FullCycle

Bancos de Dados

FullCycle

Bancos de Dados
Módulo 1

FullCycle

Aula 1
Quem é o seu professor

Lourenço Taborda

Arquiteto de Solução

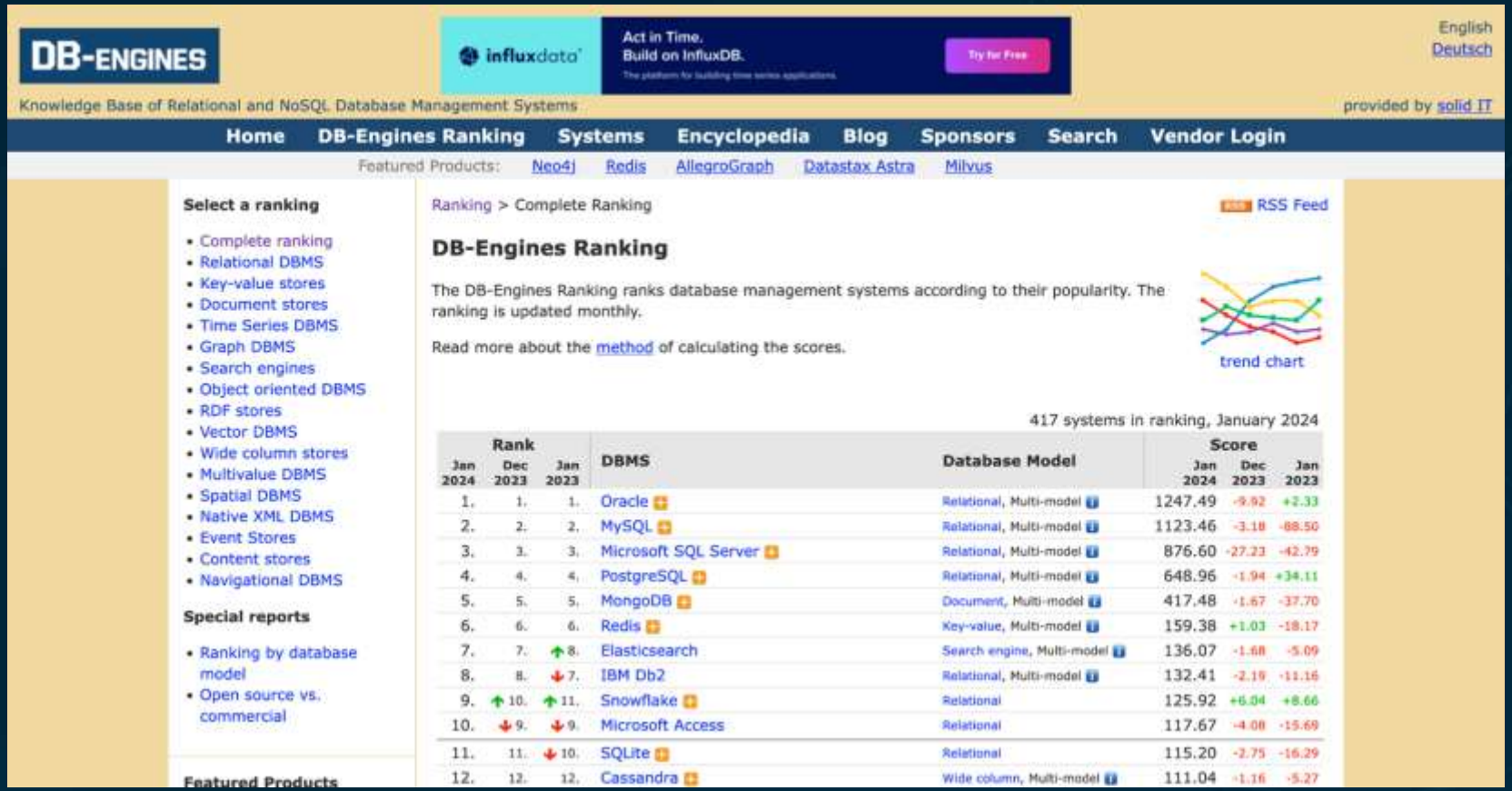
https://linktr.ee/devtbd

FullCycle

Aula 2
Bancos de Dados Populares no DB-Engines

FullCycle

DB-Engines Ranking - popularity ranking of database management systems



DB-ENGINES

Knowledge Base of Relational and NoSQL Database Management Systems

English
Deutsch

provided by solid IT

Home | DB-Engines Ranking | Systems | Encyclopedia | Blog | Sponsors | Search | Vendor Login

Featured Products: Neo4j Redis AllegroGraph Datastax Astra Milvus

Select a ranking

- Complete ranking
- Relational DBMS
- Key-value stores
- Document stores
- Time Series DBMS
- Graph DBMS
- Search engines
- Object oriented DBMS
- RDF stores
- Vector DBMS
- Wide column stores
- Multivalue DBMS
- Spatial DBMS
- Native XML DBMS
- Event Stores
- Content stores
- Navigational DBMS

Special reports

- Ranking by database model
- Open source vs. commercial

Featured Products

Ranking > Complete Ranking

RSS Feed

DB-Engines Ranking

The DB-Engines Ranking ranks database management systems according to their popularity. The ranking is updated monthly.

Read more about the method of calculating the scores.

trend chart

417 systems in ranking, January 2024

Rank Jan 2024	Rank Dec 2023	Rank Jan 2023	DBMS	Database Model	Score Jan 2024	Score Dec 2023	Score Jan 2023
1.	1.	1.	Oracle	Relational, Multi-model	1247.49	-9.92	+2.33
2.	2.	2.	MySQL	Relational, Multi-model	1123.46	-3.18	-88.50
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	876.60	-27.23	-42.79
4.	4.	4.	PostgreSQL	Relational, Multi-model	648.96	-1.94	+34.11
5.	5.	5.	MongoDB	Document, Multi-model	417.48	-1.67	-37.70
6.	6.	6.	Redis	Key-value, Multi-model	159.38	+1.03	-18.17
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model	136.07	-1.68	-5.09
8.	8.	↓ 7.	IBM Db2	Relational, Multi-model	132.41	-2.19	-11.16
9.	↑ 10.	↑ 11.	Snowflake	Relational	125.92	+6.04	+8.66
10.	↓ 9.	↓ 9.	Microsoft Access	Relational	117.67	-4.08	-15.69
11.	11.	↓ 10.	SQLite	Relational	115.20	-2.75	-16.29
12.	12.	12.	Cassandra	Wide column, Multi-model	111.04	-1.16	-5.27

FullCycle

Aula 3
Vantagem Competitiva nos Dados

FullCycle

Information Systems Strategy Triangle
Triângulo da Estratégia dos Sistemas de Informação

"Information Systems Strategy Triangle é um **framework simples** para compreender o **impacto dos sistemas** de informação nas organizações.

Nas empresas de sucesso, a Estratégia de **Negócio direciona** a Estratégia Organizacional e a Estratégia da Informação."

Information Systems Strategy Triangle
Referências:
Keri Pearlson & Carol Saunders, Managing and Using Information Systems A Strategic Approach (Hoboken: John Wiley & Sons, 2010)

FullCycle

Information Systems Strategy Triangle
Triângulo da Estratégia dos Sistemas de Informação

"A Estratégia Organizacional e a Estratégia da Informação são dependentes da Estratégia de Negócio.

Mudanças em qualquer estratégia requer o mudanças nas demais para manter a harmonia.

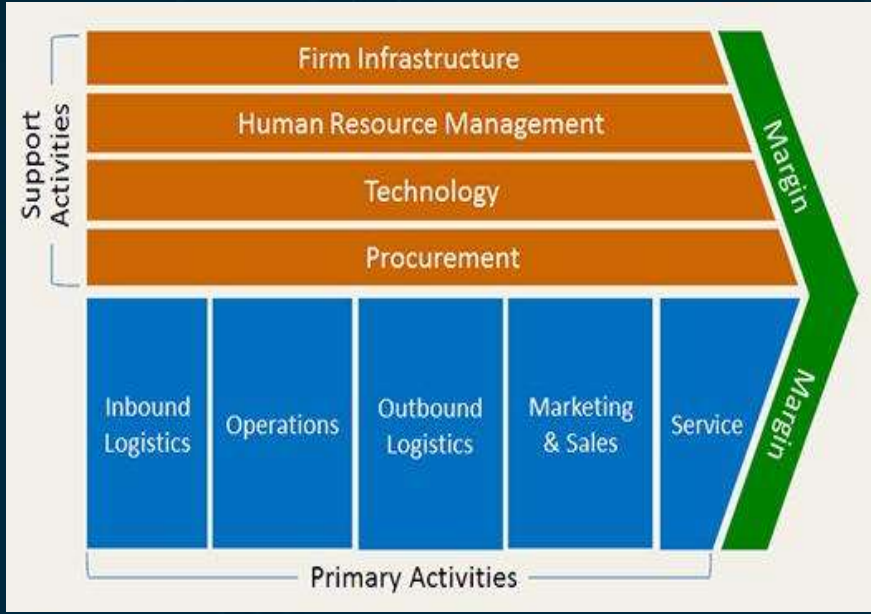
Estratégia da **Informação sempre envolve consequências para o negócio**."

Information Systems Strategy Triangle
Referências:
Keri Pearlson & Carol Saunders, Managing and Using Information Systems A Strategic Approach (Hoboken: John Wiley & Sons, 2010)

FullCycle

Value Chain
Cadeia de Valor

“Michael Porter, da Harvard Business School, descreve um negócio em termos de atividades primárias e atividades de suporte (…) que descrevem como um **negócio transforma insumos em produtos com valor agregado**.”



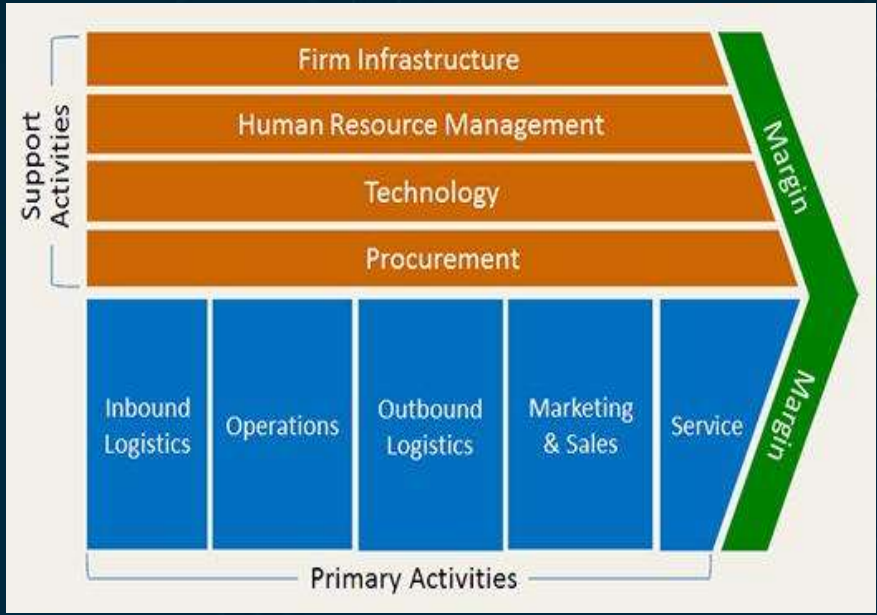
Value Chain (Cadeia de Valor)
Referências:
Michael Porter, Competitive Advantage (New York: Free Press, 1985)
Keri Pearlson & Carol Saunders, Managing and Using Information Systems A Strategic Approach (Hoboken: John Wiley & Sons, 2010)

FullCycle

Value Chain
Cadeia de Valor

“A análise da Cadeia de Valor (Value Chain Analysis) é útil para examinar o fluxo de informação em um negócio.

Além disso, permite a análise das **atividades específicas** pelas quais as organizações podem criar **vantagem competitiva (valor)**.”



Value Chain (Cadeia de Valor)
Referências:
Michael Porter, Competitive Advantage (New York: Free Press, 1985)
Keri Pearlson & Carol Saunders, Managing and Using Information Systems A Strategic Approach (Hoboken: John Wiley & Sons, 2010)

FullCycle

Contexto de Negócio e Geração de Valor

Referência:
The Data Warehousing Institute, Business Intelligence Fundamentals, 2011

FullCycle

Aula 4
Definição de Banco de Dados

FullCycle

Definição de Banco de Dados

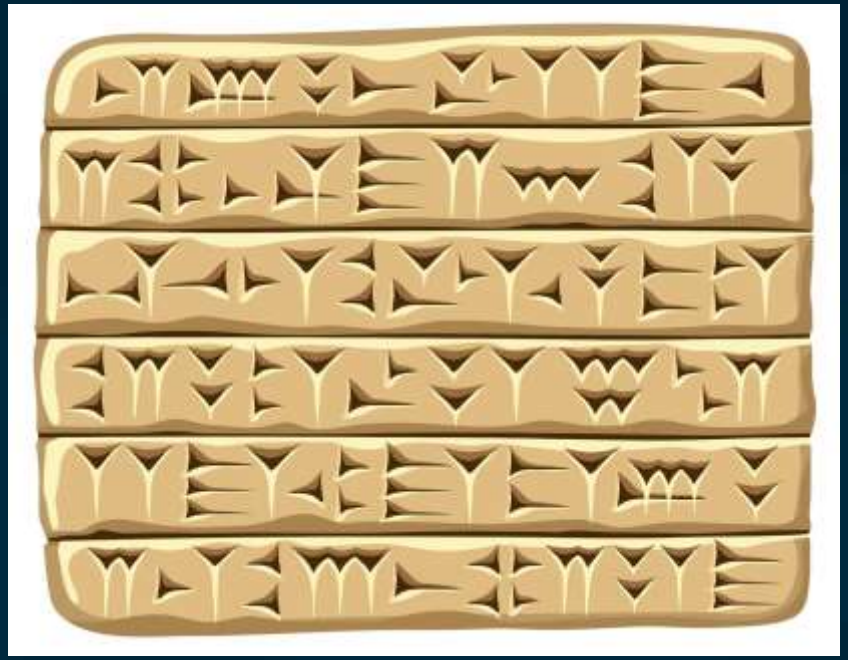


Imagem de valadzionak_volha no Freepik.

Imagem de cristina_gottardi no Freepik.

Definição de Banco de Dados

Database defined

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

"What Is a Database?" em <https://www.oracle.com/database/what-is-database/>.
Acessado em 20 de Janeiro de 2024 às 21:15.

Definição de Banco de Dados

Database defined

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

"What Is a Database?" em <https://www.oracle.com/database/what-is-database/>.
Acessado em 20 de Janeiro de 2024 às 21:15.

Definição de Banco de Dados

What is a Database?

Think of a “data+base” or a database as a place where you can put data or information of different types for as long as you want.

Formally, a database is an organized collection of structured or unstructured information stored electronically on a machine locally or in the cloud. Databases are managed using a Database Management System (DBMS). The DBMS acts as an interface between the end user (or an application) and the database. Databases use a query language for storing or retrieving data.

"What is a Database?" em <https://www.mongodb.com/databases>.
Acessado em 20 de Janeiro de 2024 às 21:20.

Definição de Banco de Dados

What is a Database?

Think of a “data+base” or a database as a place where you can put data or information of different types for as long as you want.

Formally, a database is an organized collection of structured or unstructured information stored electronically on a machine locally or in the cloud. Databases are managed using a Database Management System (DBMS). The DBMS acts as an interface between the end user (or an application) and the database. Databases use a query language for storing or retrieving data.

"What is a Database?" em <https://www.mongodb.com/databases>.
Acessado em 20 de Janeiro de 2024 às 21:20.

Definição de Banco de Dados

Banco de dados ou base de dados:

- é uma **coleção organizada** de
  - **informações** ou
  - **dados estruturados** e **não estruturados**
    - armazenados **eletronicamente**
      - em uma **máquina local** ou
      - em **nuvem**.
- é controlado por um **Sistema Gerenciador de Banco de Dados (SGBD)**,
  - traduzido do inglês Database Management System (DBMS).
- usa uma **linguagem de consulta** (query language) para escrever e ler os dados.

FullCycle

Aula 5
Dados Estruturados, Semi-Estruturados e Não-Estruturados

FullCycle

Imagem traduzida de "Structured Vs Unstructured Data: An Overview" em <https://www.mongodb.com/unstructured-data/structured-vs-unstructured>. Acessado em 20 de Janeiro de 2024 às 23:00.

FullCycle

Aula 6
Bancos de Dados SQL e NoSQL

Tipos Comuns de Banco de Dados

SQL

NoSQL

NewSQL

Tipos Comuns de Banco de Dados

- Em geral, existem 2 tipos comuns de bancos de dados:

1) Bancos de Dados de Linguagem de Consulta Estruturada Structured Query Language (**SQL**)

2) e Bancos de Dados **NoSQL**.

O banco de dados de uma aplicação deve ser determinado por suas **necessidades** e **restrições**.

Texto traduzido de "CNCF Landscape Databases" em <https://landscape.cncf.io/guide#app-definition-and-development--database>.
Acessado em 30 de Janeiro de 2024 às 23:30.

FullCycle

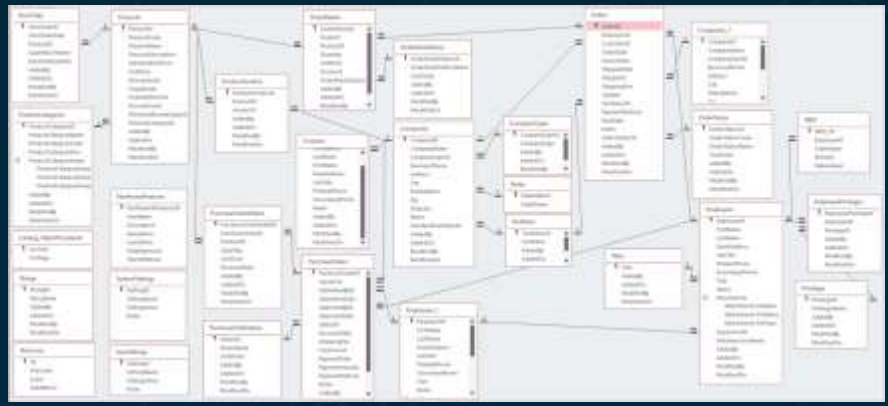
Tipos Comuns de Banco de Dados

Bancos de Dados Relacionais (SQL)

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

Texto traduzido de "What Is a Database" em <https://www.oracle.com/database/what-is-database>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Projetados na década de **1970**.
- Geralmente usa Structured Query Language (**SQL**) para operações como criação, leitura, atualização e exclusão de dados (**CRUD**).
- Armazena dados em **tabelas com colunas e linhas**, que podem ser unidas por campos conhecidos como **chaves estrangeiras**, estabelecendo relacionamentos entre as tabelas.



https://support.content.office.net/en-us/media/559a04f2-11b2-44b8-ae4a-92284d1576bd.png

- **PostgreSQL**, **MySQL**, **Microsoft SQL Server** e **Oracle** são exemplos.

Bancos de Dados Relacionais (SQL)

"***Future users of large data banks must be protected from having to know how the data is organized in the machine***. *A prompting service which supplies such information is not a satisfactory solution.*

Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed." **(Edgar Codd,1970)**.

FullCycle

Tipos Comuns de Banco de Dados

Bancos de Dados Não Relacionais (NoSQL)

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Comumente chamados de **Bancos de Dados NoSQL**.
- Amadureceu devido às **aplicações web** modernas cada vez mais **complexas**.
- As variedades desses bancos de dados proliferaram na década de **2010**.
- Exemplos populares são **MongoDB**, **Cassandra**, **DynamoDB**, **Redis** e **Neo4j**.

Not Only SQL

FullCycle

Aula 7
Tipos de Bancos de Dados NoSQL

FullCycle

Tipos Comuns de Banco de Dados

Bancos de Dados Não Relacionais (NoSQL)

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Os tipos mais populares destes banco de dados são:

Bancos de Dados NoSQL de
Documento

Bancos de Dados NoSQL de
Chave-Valor

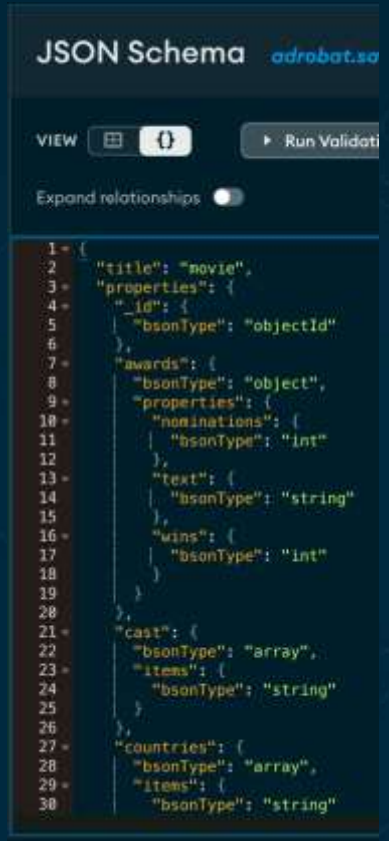
Bancos de Dados NoSQL
Colunar

Bancos de Dados NoSQL de
Grafo

Bancos de Dados NoSQL de Documento

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

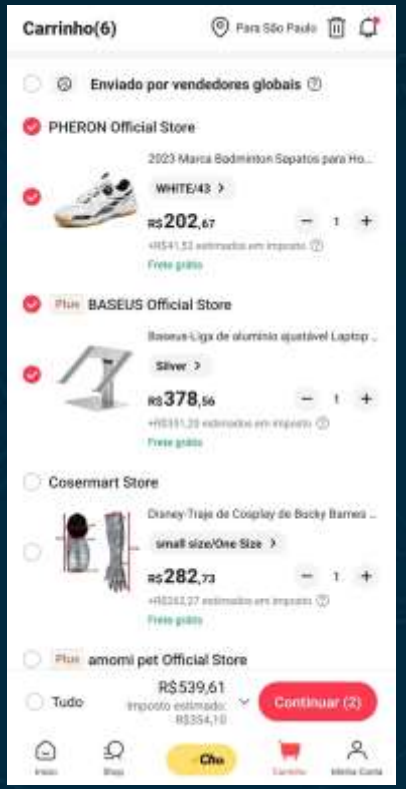
- Armazena dados em documentos **JSON**, **BSON** ou **XML**.
- Elementos específicos podem ser **indexados** para consultas mais rápidas.
- Formato muito mais **próximo dos objetos** de dados usados na aplicação.
- **Menos tradução** é necessária para usar e acessar os dados.
- Flexibilidade de **retrabalhar suas estruturas** de documentos.
- Os **dados se tornam como código** e ficam sob o controle dos desenvolvedores.
- **Não requer intervenção dos administradores** de banco de dados para alterar a estrutura de um banco de dados.



Bancos de Dados NoSQL de Chave-Valor

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Imagem de <https://webimages.mongodb.com/_com_assets/cms/ku7azn477vrdsl6w4-key-value-database.png>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Tipo mais simples de banco de dados NoSQL.
- Cada elemento é armazenado como **um par chave-valor** que consiste em um nome de atributo (“chave”) e um conteúdo (“valor”).
- Este banco de dados é como um RDBMS com **duas colunas**: o nome do atributo (como "estado") e o valor (como "SP").
- Os casos de uso incluem **carrinhos de compras**, **preferências** e **perfis de usuário**.



Bancos de Dados NoSQL Colunar

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Também denominados “**orientados à coluna**” e “**coluna larga**”.
- Organizados como um **conjunto de colunas**.
- Consultas diretamente nas colunas selecionadas, **sem consumir memória com dados indesejados**.
- Colunas são do mesmo tipo e se beneficiam de uma **compactação mais eficiente**, tornando as leituras ainda mais rápidas. Entregam **alta velocidade em operações de agregação**.
- A maneira como os dados são gravados **dificulta a consistência**, pois as gravações de todas as colunas no banco de dados orientado a colunas exigem **vários eventos de gravação no disco**.

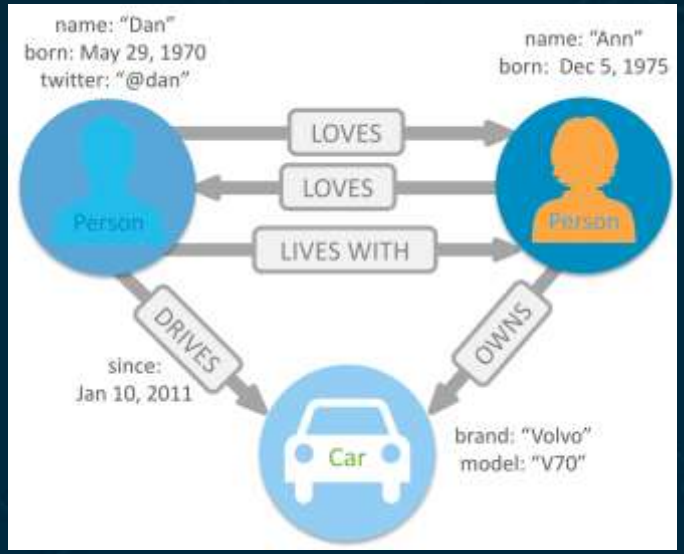
Chave Primária	“Coluna larga”				
id	song_order	album	artist	song_id	title
62c36092...	4	No One Rides for Free	Fu Manchu	7db1a490...	Ojo Rojo
62c36092...	3	Roll Away	Back Door Slam	2b09185b...	Outside Woman Blues
62c36092...	2	We Must Obey	Fu Manchu	8a172618...	Moving in Stereo
62c36092...	1	Tres Hombres	ZZ Top	a3e63f8f...	La Grange

https://docs.datastax.com/en/cql-oss/3.3/cql/images/screenshots/select_desc.png

Banco de Dados NoSQL de Grafo

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Concentra-se no **relacionamento entre os elementos de dados**.
- Cada elemento está contido como um **nó**.
- As **conexões** entre os elementos do banco de dados são chamadas de **links** ou **relacionamentos**.
- Otimizado para **capturar e pesquisar as conexões** entre os elementos, superando a sobrecarga associada ao JOIN de várias tabelas em SQL.
- Os casos de uso incluem **detecção de fraudes** e **redes sociais**.



https://dist.neo4j.com/wp-content/uploads/graph-example.png

FullCycle

Aula 8
Bancos de Dados NewSQL
Outros Tipos de Banco de Dados

Bancos de Dados NewSQL

- Com a ascensão do Kubernetes e sua capacidade de oferecer suporte a aplicações stateful, vimos uma **nova geração de bancos de dados** aproveitar as **vantagens da conteinerização**.

- Esses novos bancos de dados nativos da nuvem visam trazer os **benefícios de escalabilidade e disponibilidade** do Kubernetes para os bancos de dados.

Texto traduzido de "CNCF Landscape Databases" em <https://landscape.cncf.io/guide#app-definition-and-development--database>.
Acessado em 30 de Janeiro de 2024 às 23:30.

Bancos de Dados NewSQL

- **NewSQL** é uma classe de sistemas de gerenciamento de **banco de dados relacionais** que busca fornecer a **escalabilidade de sistemas NoSQL** para cargas de trabalho de processamento de transações online (**OLTP**), mantendo as **garantias ACID** de um sistema de banco de dados tradicional.
    - OLTP: OnLine Transaction Processing
    - ACID: Atomicidade - Consistência - Isolamento - Durabilidade

Texto traduzido de "NewSQL" em <https://en.wikipedia.org/wiki/NewSQL>.
Acessado em 30 de Janeiro de 2024 às 23:30.

Bancos de Dados NewSQL

STONEBRAKER e CATTEL (2011) definem cinco características de um **NewSQL**:

1. **Linguagem SQL** como meio de interação entre o SGBD e a aplicação;
2. Suporte para **transações ACID**;
3. Controle de **concorrência não bloqueante**, para que as leituras e escritas não causem conflitos entre si;
4. Arquitetura que forneça um maior **desempenho** por nó de processamento;
5. Arquitetura **escalável**, com **memória distribuída** e com capacidade de funcionar em um **cluster** com um grande número de nós.

Adaptado de "NewSQL" em <https://www.devmedia.com.br/conheca-a-geracao-de-banco-de-dados-nosql-e-newsql/33202>.
Acessado em 30 de Janeiro de 2024 às 23:30.

Bancos de Dados Hierárquicos

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Desenvolvido na década de **1960**, o banco de dados hierárquico se parece com uma **árvore genealógica**.
- Um único objeto (o “pai”) possui um ou mais objetos abaixo dele (o “filho”).
- Nenhum filho pode ter mais de um pai.
- Oferece **alto desempenho** pois possui acesso fácil e tempo de consulta rápido como consequência do **padrão rígido** e **complexo** de navegação pela árvore.
- O Registro do Windows é um exemplo desse sistema.

Bancos de Dados Orientados a Objetos

Texto traduzido de "Types of Databases" em <https://www.mongodb.com/databases/types>.
Acessado em 20 de Janeiro de 2024 às 23:00.

Texto traduzido de "What Is a Database" em <https://www.oracle.com/database/what-is-database>.
Acessado em 20 de Janeiro de 2024 às 23:00.

- Informação representada na forma de objetos, como na **Programação Orientada a Objetos**.
- Armazenam e gerenciam objetos no disco de um servidor de banco de dados.
- As consultas de dados em **relacionamentos complexos** são rápidas e poderosas.

- Um exemplo de banco de dados orientado a objetos é o **MongoDB Realm**, onde a linguagem de consulta constrói objetos nativos por meio do SDK escolhido.

```
class Carro extends Realm.Object {
  static schema = {
    name: 'Carro',
    properties: {
      fabricante: 'string',
      modelo: 'string',
      quilometragem: {type: 'int', default: 0},
      timestamp: {
        type: 'int',
        default: () => Math.round(new Date().getTime() / 1000),
      },
    },
  };
}
```

FullCycle

Evolução dos Bancos de Dados

1960-1980:
File Systems
Hierárquicos
Rede

1980-2000:
soluções RDBMS
Distribuídas,
Mini-computadores

2000-2005:
bolha .COM, nova escala,
início do NOSQL,
whitepapers

2005-2010
Open Source

2010-2015:
Adoção de banco de dados na nuvem
DBaaS
Hadoop/Streams

2015:2022
Adoção de plataformas de dados :
inovação,transformação

2023+
Surgimento de GenAI
Pulverização de NoSQL
ex: Vector Search

Armazenamento de Dados

VSAM
IMS
Adabas
IDMS

TERADATA
SQL Server
MySQL
ORACLE
PostgreSQL
INFORMIX
INGRES
DB2
amazon Try Prime
facebook
Google
Cassandra
HBASE
mongoDB
CouchDB
riak
redis
Azure CosmosDB
Google BigQuery
amazon DynamoDB
SAP HANA
amazon web services
Microsoft Azure
databricks
snowflake
Chroma
RocksDB

Bancos de Dados Relacional

System R

Object-Relational,
Shared nothing
Stored Procedures
Triggers,
Otimização baseada em custo

Dado Geoespacial,
Série Temporal,
Shared Disk,
Processamento em Grid

NoSQL,
Poliglota,
Armazenamento distribuído,
In-memory

JSON
Armazenamento Colunar
Computação em Nuvem,
BigData

Plataforma de Dados

FullCycle

Compreenda os Requisitos do seu Caso de Uso

FullCycle

Aula 9
Computação em Nuvem: Características Essenciais

É uma abordagem de negócio e de entrega de tecnologia da informação que é composta por **5 características essenciais**:

1) **autosserviço sob demanda**,

2) **amplo acesso pela rede**,

3) **compartilhamento dos recursos tecnológicos coletivos entre múltiplos clientes**,

4) **elasticidade rápida** e **aparentemente infinita** e

5) **medição, controle e observabilidade do consumo** dos recursos.

Texto traduzido de MELL, Peter. GRANCE, Timothy. The National Institute of Standards and Technology Definition of Cloud Computing. National Institute of Standards and Technology. Disponível em: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>. Acesso em: 06 de setembro de 2021.

FullCycle

Aula 10
Computação em Nuvem: Modelos de Serviço e Responsabilidade Compartilhada

Computação em Nuvem (Cloud Computing)

É uma abordagem de negócio e de entrega de tecnologia da informação que apresenta **3 modelos de serviços**:

1) Software como Serviço (**SaaS**, Software as a Service),
2) Plataforma como Serviço (**PaaS**, Platform as a Service) e
3) Infraestrutura como Serviço (**IaaS**, Infrastructure as a Service).

Texto traduzido de MELL, Peter. GRANCE, Timothy. The National Institute of Standards and Technology Definition of Cloud Computing. National Institute of Standards and Technology. Disponível em: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>. Acesso em: 06 de setembro de 2021.

FullCycle

Computação em Nuvem (Cloud Computing)

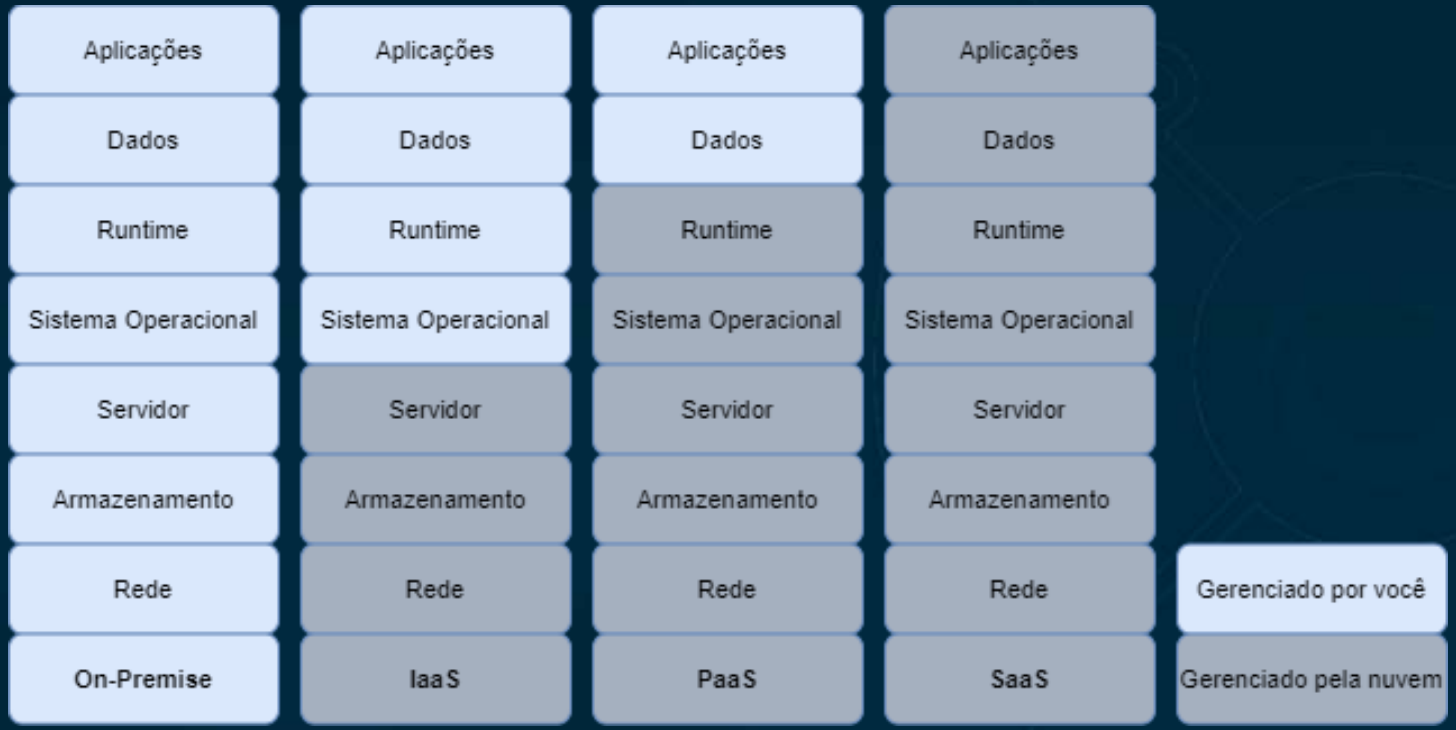


Diagrama gerado a partir de MELL, Peter. GRANCE, Timothy. The National Institute of Standards and Technology Definition of Cloud Computing. National Institute of Standards and Technology. Disponível em: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>. Acesso em: 06 de setembro de 2021.

FullCycle

Aula 11
Computação em Nuvem: Modelos de Implantação e Nativo da Nuvem

Computação em Nuvem (Cloud Computing)

É uma abordagem de negócio e de entrega de tecnologia da informação que disponibiliza **4 modelos de implantação** (deploy):

1) Nuvem **Privada**,

2) Nuvem **Comunitária**,

3) Nuvem **Pública** e

4) Nuvem **Híbrida** composta pela combinação dos modelos de serviço anteriores.

Texto traduzido de MELL, Peter. GRANCE, Timothy. The National Institute of Standards and Technology Definition of Cloud Computing. National Institute of Standards and Technology. Disponível em: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>. Acesso em: 06 de setembro de 2021.

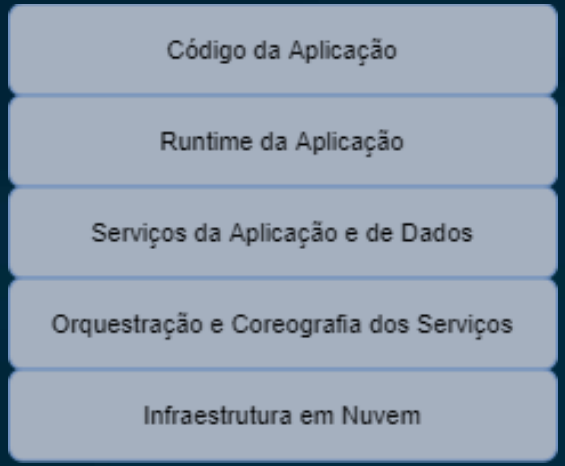
Nativo da Nuvem (Cloud Native)

Uma aplicação nativa da nuvem é um programa de computador especificamente projetado para realizar o seu propósito de negócio e que **satisfaz às características essenciais da computação em nuvem**, adota os modelos de serviço e implantação adequadamente e de forma otimizada.
As aplicações nativas da nuvem são especificamente projetadas para tirar **vantagem das inovações da computação em nuvem**. Facilmente integram-se com suas respectivas arquiteturas em nuvem, **alavancando os recursos em nuvem e capacidades elásticas**.

Texto traduzido de Cloud Native Apps. Cloud Native Computing Foundation. Disponível em: <https://github.com/cncf/glossary/blob/main/definitions/cloud_native_apps.md>. Acesso em 06 de setembro de 2021.

FullCycle

Nativo da Nuvem (Cloud Native)



Aplicações que não foram projetadas para a nuvem não podem tirar vantagem da escalabilidade e resiliência do ambiente em nuvem.

As aplicações nativas da nuvem são resilientes, gerenciáveis e alavancadas pelo conjunto de serviços do provedor de nuvem que as acompanham, tais como alto nível de observabilidade, automação e previsibilidade.

Diagrama gerado de Cloud Native Apps. Cloud Native Computing Foundation. Disponível em: <https://github.com/cncf/glossary/blob/main/definitions/cloud_native_apps.md>. Acesso em 06 de setembro de 2021.

FullCycle

Aula 12
Cargas de Trabalho Transacionais e Analíticas

Cargas de Trabalho Transacionais (OLTP) e Analíticas (OLAP)

OLTP → OnLine Transaction Processing

- em livre tradução: Processamento de Transações Online.

É um tipo de processamento de dados que consiste na execução de várias transações que ocorrem simultaneamente (transações bancárias online, compras, entrada de pedidos ou envio de mensagens de texto, por exemplo).

Normalmente envolve **inserir, atualizar e/ou excluir pequenas quantidades de dados** em um armazenamento de dados para coletar, gerenciar e **proteger essas transações**, sejam elas econômicas ou financeiras (em sua definição original).

Interpretação do autor sobre a Definição de OLTP em O que é OLTP. Oracle. Disponível em: <https://www.oracle.com/br/database/what-is-oltp>. Acesso em 11 de fevereiro de 2024.

Cargas de Trabalho Transacionais (OLTP) e Analíticas (OLAP)

OLAP → OnLine Analytical Processing

- em livre tradução: Processamento Analítico Online.

OLTP permite a execução em tempo real de um grande número de transações por um grande número de pessoas, enquanto o **processamento analítico online (OLAP) geralmente envolve a consulta dessas transações** (também chamadas de registros) **em um banco de dados para fins analíticos**.

O OLAP ajuda as empresas a extrair insights de dados de transações para que possam usá-los em tomadas de decisões mais informadas.

Interpretação do autor sobre a Definição de OLTP em O que é OLTP. Oracle. Disponível em: <https://www.oracle.com/br/database/what-is-oltp>. Acesso em 11 de fevereiro de 2024.

FullCycle

Cargas de Trabalho Transacionais (OLTP) e Analíticas (OLAP)

OLTP	OLAP
Executa em **tempo real** de um grande número de transações de banco de dados por um grande número de pessoas.	Envolve a **consulta de muitos registros** (até mesmo todos os registros) em um banco de dados para **fins analíticos.**
Tempos de resposta **extremamente rápidos**.	Requer tempos de resposta **mais lentos** do que os exigidos pelo OLTP.
Modifica pequenas quantidades de dados **com frequência** e geralmente envolve um equilíbrio de **leituras e gravações**.	**Não modifica** os dados de forma alguma; cargas de trabalho são geralmente de **leitura intensiva**.

Interpretação do autor sobre a Definição de OLTP em O que é OLTP. Oracle. Disponível em: <https://www.oracle.com/br/database/what-is-oltp>. Acesso em 11 de fevereiro de 2024.

FullCycle

Cargas de Trabalho Transacionais (OLTP) e Analíticas (OLAP)

OLTP	OLAP
Usa **dados indexados** para melhorar os tempos de resposta.	Armazena dados em **formato colunar** para permitir **acesso fácil** a um grande número de registros.
Exige **backups frequentes** ou **simultâneos**.	Exige **backups** muito **menos frequentes**.
Exige relativamente **menos espaço** de armazenamento.	Armazenam **grandes quantidades de dados históricos**.

Interpretação do autor sobre a Definição de OLTP em O que é OLTP. Oracle. Disponível em: <https://www.oracle.com/br/database/what-is-oltp>. Acesso em 11 de fevereiro de 2024.

FullCycle

Cargas de Trabalho Transacionais (OLTP) e Analíticas (OLAP)

OLTP	OLAP
Geralmente executa **consultas simples** envolvendo apenas **um ou alguns registros**.	Executa c**onsultas complexas** envolvendo um **grande número de registros**.

Portanto, **OLTP é um sistema de modificação de dados em tempo real**, enquanto **OLAP é um sistema de armazenamento de dados multidimensional histórico em tempo real** usado para **recuperar grandes quantidades de dados para fins analíticos**. OLAP geralmente fornece análises sobre dados que **foram capturados por um ou mais sistemas OLTP**.

Interpretação do autor sobre a Definição de OLTP em O que é OLTP. Oracle. Disponível em: <https://www.oracle.com/br/database/what-is-oltp>. Acesso em 11 de fevereiro de 2024.

FullCycle

Aula 13
Sistema Gerenciador de Banco de Dados em Nuvem

FullCycle

Sistema Gerenciador de Banco de Dados em Nuvem (Cloud DBMS)

Sistemas Gerenciadores de Banco de Dados em nuvem (SGBDs em nuvem):

- do inglês Cloud Database Management Systems (cloud DBMSs)

São produtos de software que armazenam e manipulam dados e que são entregues principalmente como software como serviço (SaaS) na nuvem.

Os SGBDs em nuvem podem, **opcionalmente**, serem capazes de executar localmente (on-premises) ou em configurações híbridas, multi nuvem ou inter nuvem.

Podem ser usados para **trabalho transacional e/ou analítico**.

Podem ter recursos de um **ecossistema de dados mais amplo**.

Interpretação do autor sobre a definição dos Sistemas Gerenciadores de Banco de Dados em nuvem exposto em Magic Quadrant for Cloud Database Management Systems de 18 de dezembro de 2023. Gartner. Disponível em: <https://www.gartner.com/doc/reprints?id=1-2F9JOOFF&ct=231010&st=sb>. Acesso em 11 de fevereiro de 2024.

Sistema Gerenciador de Banco de Dados em Nuvem (Cloud DBMS)

Sistemas Gerenciadores de Banco de Dados em nuvem entregam pelo menos 1 destes casos de uso:

1. **OLTP**: foco transacional com esquema de dados fixo e estável.
2. **Transações leves**: volumes muito elevados de transações simples com alta simultaneidade, potencialmente com consistência relaxada.
3. **Inteligência operacional**: grande número de usuários simultâneos que executam consultas analíticas curtas, ao mesmo tempo em que entrega cargas de trabalho operacionais.
4. **Data Warehouse Tradicional**: dados estruturados históricos de múltiplas origens.
5. **Data Warehouse Lógico**: camada lógica ou virtual para uma variedade de origens
6. **Data Lake** e **Machine Learning**: armazenamento e processamento de dados com diferentes estruturas e origens.

Interpretação do autor sobre a definição dos Sistemas Gerenciadores de Banco de Dados em nuvem exposto em Magic Quadrant for Cloud Database Management Systems de 18 de dezembro de 2023. Gartner. Disponível em: <https://www.gartner.com/doc/reprints?id=1-2F9JOOFF&ct=231010&st=sb>. Acesso em 11 de fevereiro de 2024.

Sistema Gerenciador de Banco de Dados em Nuvem (Cloud DBMS)

Sistemas Gerenciadores de Banco de Dados em nuvem entregam estas capacidades:

1. Disponibilidade de software como serviço (SaaS) em nuvens públicas ou privadas.
2. Gestão dos dados em armazenamento em nuvem, ou seja, não é infraestrutura como serviço (IaaS).
3. Persistir dados no armazenamento controlado pelo próprio SGBD em nuvem.
4. Componentes de gerenciamento de dados que armazenam, leem, atualizam e gerenciam dados. Não é um aglomerado de diferentes ferramentas.
5. Suporte a operações transacionais ou analíticas ou ambas.
6. Opcionalmente, suportar múltiplos modelos de dados e tipos de dados (relacionais, não-relacionais, geoespaciais, séries temporais e outros).

Interpretação do autor sobre a definição dos Sistemas Gerenciadores de Banco de Dados em nuvem exposto em Magic Quadrant for Cloud Database Management Systems de 18 de dezembro de 2023. Gartner. Disponível em: <https://www.gartner.com/doc/reprints?id=1-2F9JOOFF&ct=231010&st=sb>. Acesso em 11 de fevereiro de 2024.

FullCycle

Aula 14
Databases, Data Warehouses e Data Lakes
Parte 1

FullCycle

Databases, Data Warehouses e Data Lakes

Databases	Data Warehouses	Data Lakes
Coleção organizada de dados ou informações acessadas **eletronicamente**.	Coletam dados de muitas origens, inclusive **múltiplos OLTP**. Casos de uso de relatórios (ex.: vendas trimestrais por loja) e predição (ex.: forecast de vendas baseado na tendência histórica).	
Processamento de transações online (**OLTP**).	Processamento analítico online (**OLAP**).	
Relacionais armazenam dados em **tabelas com linhas e colunas fixas**.	Armazena informações **altamente estruturadas** de várias fontes.	Repositório de dados de **fontes distintas** que são armazenados em seu formato **original** e **bruto**.

Databases vs. Data Warehouses vs. Data Lakes. MongoDB. Disponível em: <https://www.mongodb.com/databases/data-lake-vs-data-warehouse-vs-database>. Acesso em 12 de fevereiro de 2024.

FullCycle

Databases, Data Warehouses e Data Lakes

Databases	Data Warehouses	Data Lakes
N**ão relacionais** (**NoSQL**) armazenam dados em uma **variedade de modelos** (JSON, BSON, pares de chave-valor, tabelas com linhas e colunas dinâmicas, além de vértices e arestas.	É um **banco de dados gigante otimizado para leitura** de dados **atuais e históricos** de **um ou mais sistemas** para analisar, procurar insights e criar **Business Intelligence** (BI, relatórios e painéis gerenciais).	Foco na análise dos dados para **obter insights**. Também usado simplesmente para **armazenamento barato**, com a ideia de que os dados poderão ser usados para **análise no futuro**.
Armazenam dados **estruturados** e/ou **semiestruturados**.	**Dados estruturados** predominantemente. Esquema relacional pré definido e fixo.	V**ários formatos**, incluindo JSON, BSON, CSV, TSV, Avro, ORC e Parquet.

Databases vs. Data Warehouses vs. Data Lakes. MongoDB. Disponível em: <https://www.mongodb.com/databases/data-lake-vs-data-warehouse-vs-database>. Acesso em 12 de fevereiro de 2024.

FullCycle

Databases, Data Warehouses e Data Lakes

Databases	Data Warehouses	Data Lakes
Funcionalidades de **segurança** (**criptografia** em repouso, em trânsito e em uso; **autenticação**, **autorização**, **auditoria** etc).	Armazena desde dados brutos sem transformação até dados **altamente selecionados, limpos, curados, filtrados e agregados**.	Os dados **não precisam ser transformados** para serem ingeridos. Armazenam grandes quantidades de dados atuais e históricos.
Transações ACID (Atomicidade, Consistência, Isolamento, Durabilidade) para garantir integridade dos dados.	Processos de extração, transformação e carga (**ETL**) **movem dados** da origem para o data warehouse regularmente (ex.: cargas diárias ou horárias).	A **ingestão** de dados é incrivelmente **eficiente**, sem **planejamento prévio**. Dados podem ser processados com OLAP e visualizados com BI.

Databases vs. Data Warehouses vs. Data Lakes. MongoDB. Disponível em: <https://www.mongodb.com/databases/data-lake-vs-data-warehouse-vs-database>. Acesso em 12 de fevereiro de 2024.

FullCycle

Aula 15
Databases, Data Warehouses e Data Lakes
Parte 2

FullCycle

Databases, Data Warehouses e Data Lakes

Databases	Data Warehouses	Data Lakes
Interação facilitada por meio de Linguagens de consulta e APIs.	**Ferramentas de BI** permitem a exploração dos dados.	Público varia de acordo com a estrutura dos dados. **Analistas de negócios alcançam insights com dados mais estruturados**. Com **dados mais desestruturados**, a análise de dados exige experiência de pessoas desenvolvedoras, cientistas e engenheiros dados.
Índices para desempenho de consultas. Podem entregar buscas **textuais** e **vetoriais**.	**Modelagem dimensional de dados** para desempenho de consultas.	
Topologias flexíveis para isolar cargas de trabalho (nó analítico). Deploy **local**, nuvem **privada**, nuvem **pública**, nuvem **híbrida** e/ou **multi nuvem**.	Devido à sua natureza altamente estruturada, a **análise dos dados é simplificada**.	

Databases vs. Data Warehouses vs. Data Lakes. MongoDB. Disponível em: <https://www.mongodb.com/databases/data-lake-vs-data-warehouse-vs-database>. Acesso em 12 de fevereiro de 2024.

FullCycle

Databases, Data Warehouses e Data Lakes

Databases

Relacionais
- Oracle
- MySQL
- Microsoft SQL Server
- PostgreSQL

Documentais
- MongoDB
- CouchDB

Chave-Valor
- Redis
- DynamoDB

Colunares (Coluna Larga)
- Cassandra
- HBase

Grafos
- Neo4j
- Amazon Neptune

Data Warehouses

- Amazon Redshift
- Google BigQuery
- IBM DB2 Warehouse
- Microsoft Azure Synapse
- Oracle Autonomous Data Warehouse
- Snowflake
- Teradata Vantage

Data Lakes

Armazenamento
- AWS S3
- Azure Data Lake Storage Gen2
- Google Cloud Storage

Processamento e consulta
- MongoDB Atlas Data Lake
- AWS Athena
- Presto
- Starburst
- Databricks SQL Analytics

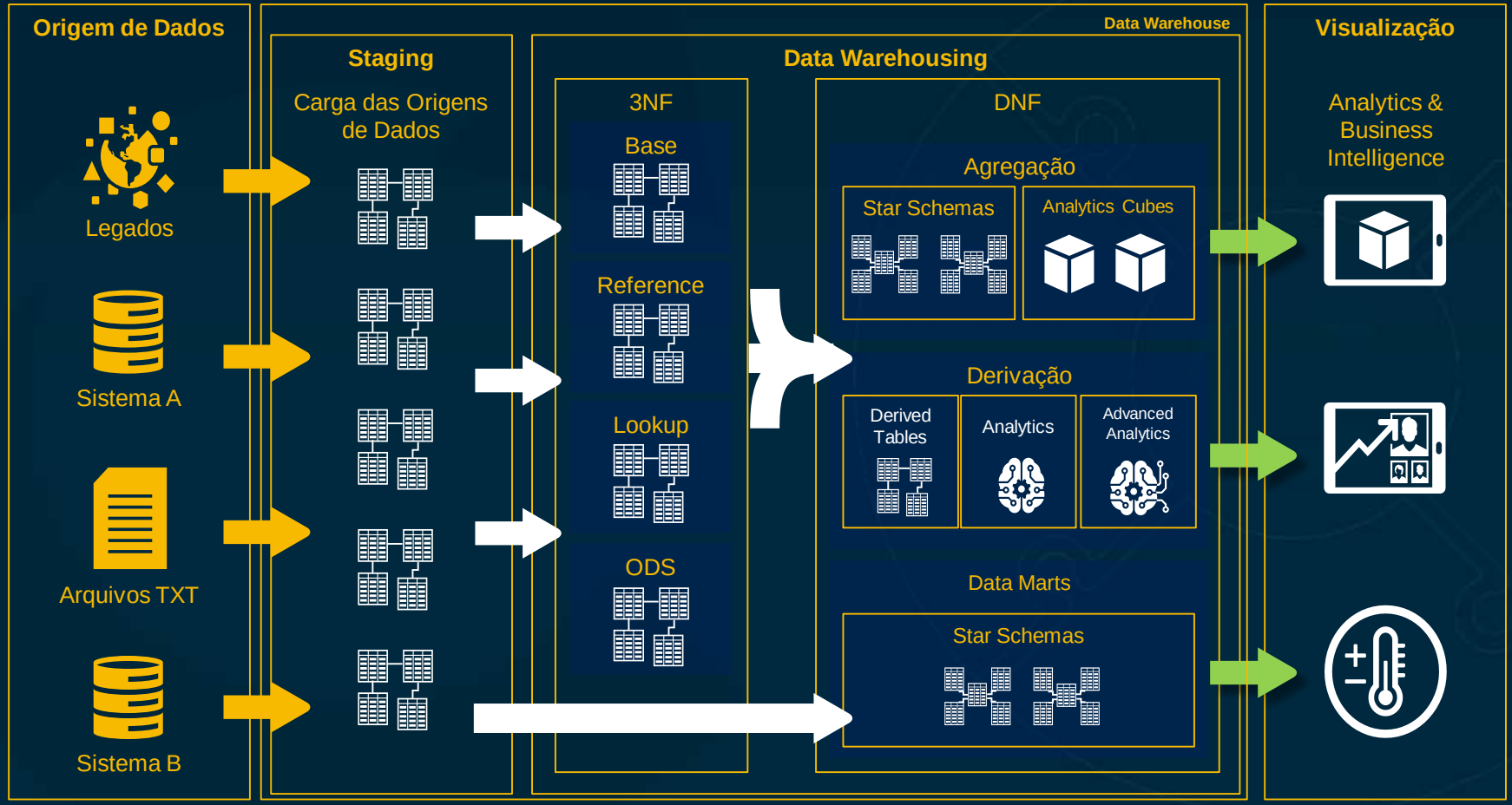
Databases vs. Data Warehouses vs. Data Lakes. MongoDB. Disponível em: <https://www.mongodb.com/databases/data-lake-vs-data-warehouse-vs-database>. Acesso em 12 de fevereiro de 2024.

FullCycle

Aula 16
Arquitetura Data Warehouse

FullCycle

Arquitetura Data Warehouse



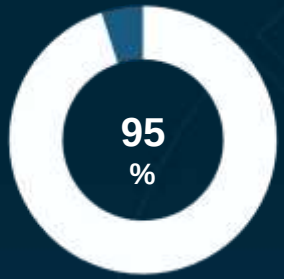
Diagramação realizada pelo autor baseada em seus estudos, sua experiência e vivência de projetos desta natureza.

FullCycle

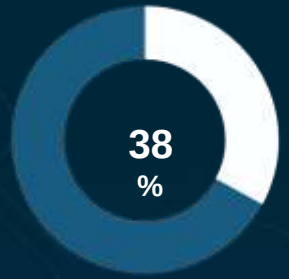
Aula 17
A Fama do Data Warehouse

A fama do Data Warehouse

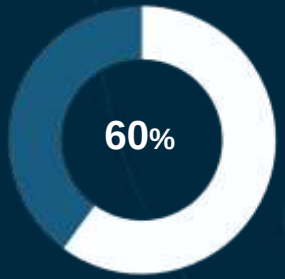
- **Manual**
- **Complexo e caro**
- **Lento**



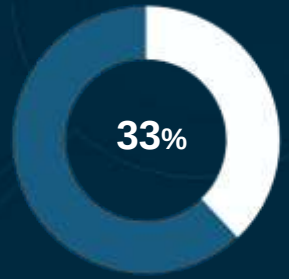
Requer extenso envolvimento manual



Aquisição inicial e custos contínuos com manutenção



Muito complexo de gerenciar



Lento e demorado de implementar

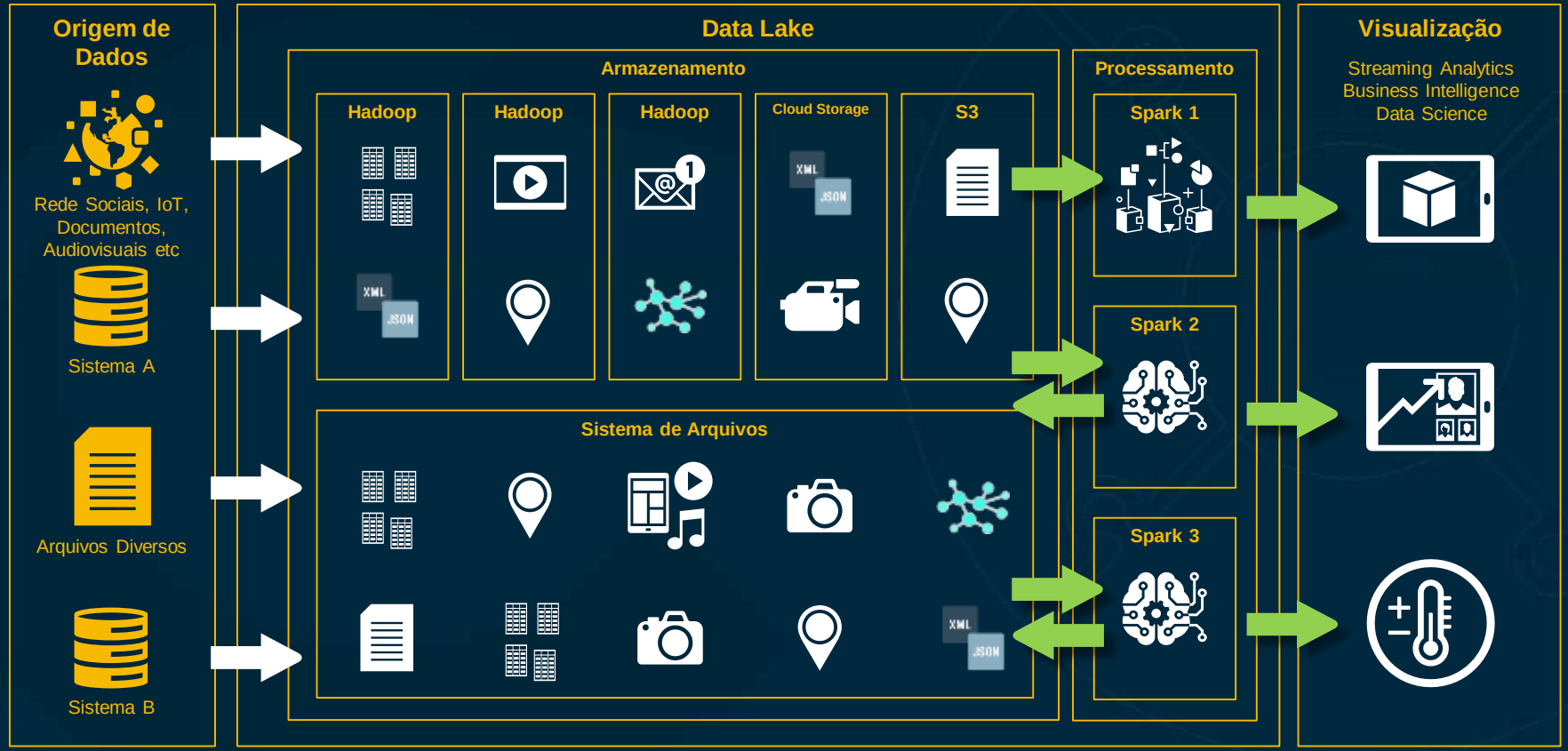
Fonte: Dimensional Research – The State of the Data Warehouse

FullCycle

Aula 18
Arquitetura Data Lake

FullCycle

Arquitetura Data Lake



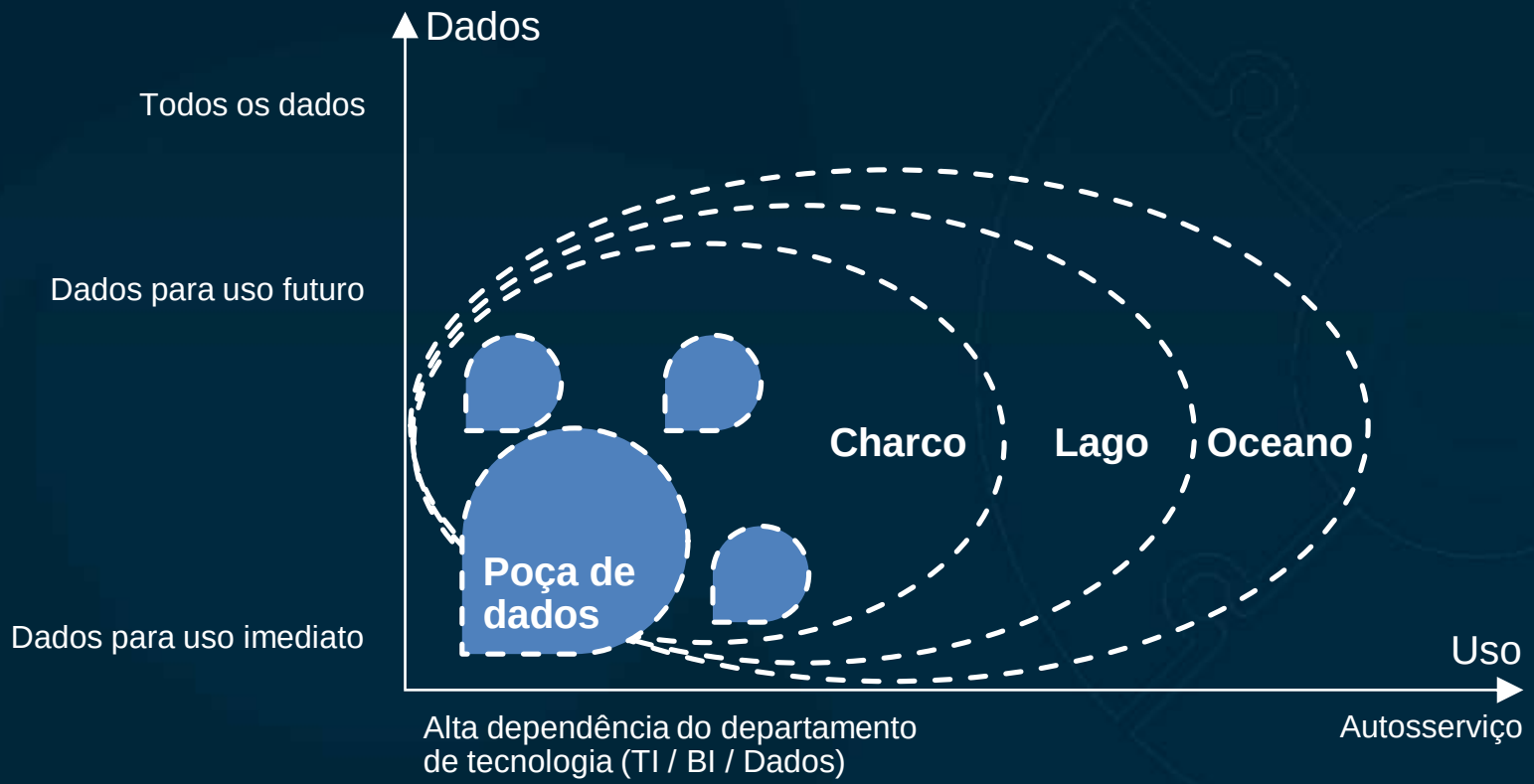
Diagramação realizada pelo autor baseada em seus estudos, sua experiência e vivência de projetos desta natureza.

FullCycle

Aula 19
Maturidade do Data Lake

FullCycle

Maturidade do Data Lake



Referência: The Enterprise Big Data Lake. O'Reilly Media. 2019. ISBN 9781491931554.

Foto de Greg Rakozy no Unsplash

FullCycle

Aula 20
A Fama do Data Lake

A fama do Data Lake

Falta de atomicidade e isolamento transacional

Inconsistência de dados e qualidade de dados reduzida

Caótico e complexo

Referência: Learning Spark, 2nd Edition. O'Reilly Media. 2020. ISBN 9781492050049.

FullCycle

Aula 21
Arquitetura Lakehouse

Lakehouse: um novo paradigma que combina elementos de Data Lake e Data Warehouse.

TRANSAÇÃO ACID

CONFORMIDADE DE ESQUEMA

FORMATOS DIVERSOS E ABERTOS

CARGAS MISTAS

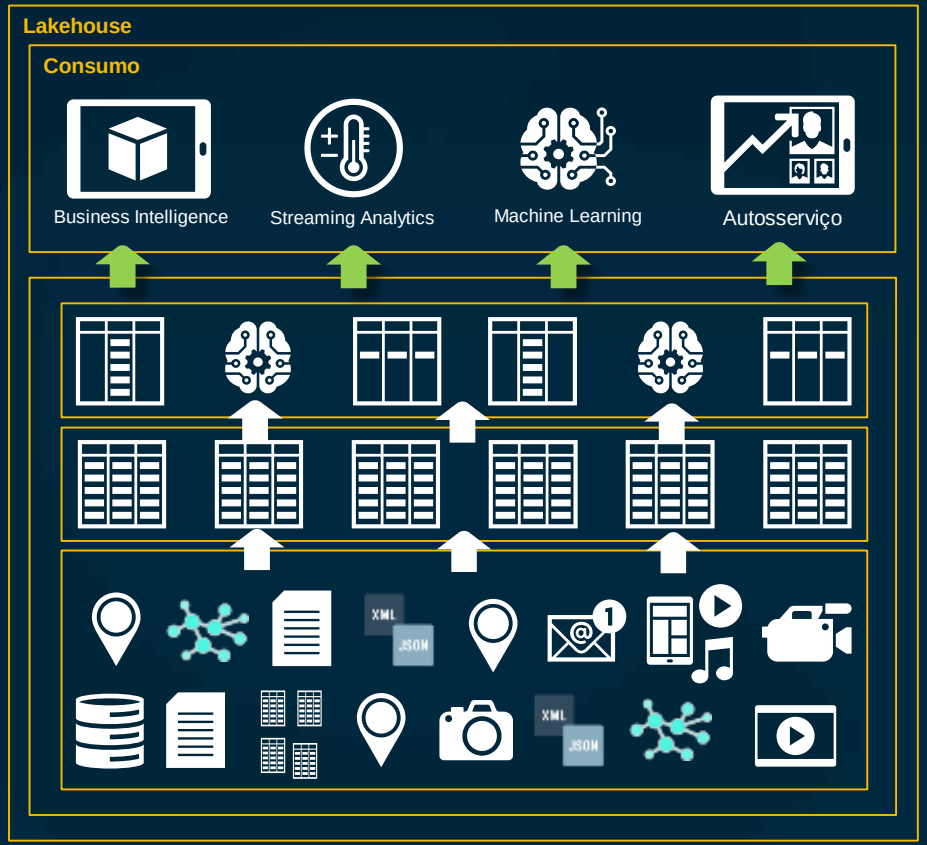
UPSERT & DELETES PARALELOS

GOVERNANÇA DE DADOS

Referência: Learning Spark, 2nd Edition. O'Reilly Media. 2020. ISBN 9781492050049.

FullCycle

Arquitetura Lakehouse



Plataforma única para consumo

Motor de alto desempenho para consultas

Camada transacional estruturada

Data Lake para todos os dados

Referências:

https://www.youtube.com/watch?v=yumysN3XwbQ

https://databricks.com/wp-content/uploads/2020/08/p975-armbrust.pdf

Projetos Lakehouse

APACHE HUDI

Hadoop Update Delete and Incremental

Focado em upserts e deletes em Chave-Valor

Combina formatos colunares e lineares

APACHE ICEBERG

Focado em propósito geral de armazenamento em tabelas únicas de grande tamanho

Evolução de esquema e particionamento, versionamento e isolamento serializado

DELTA LAKE

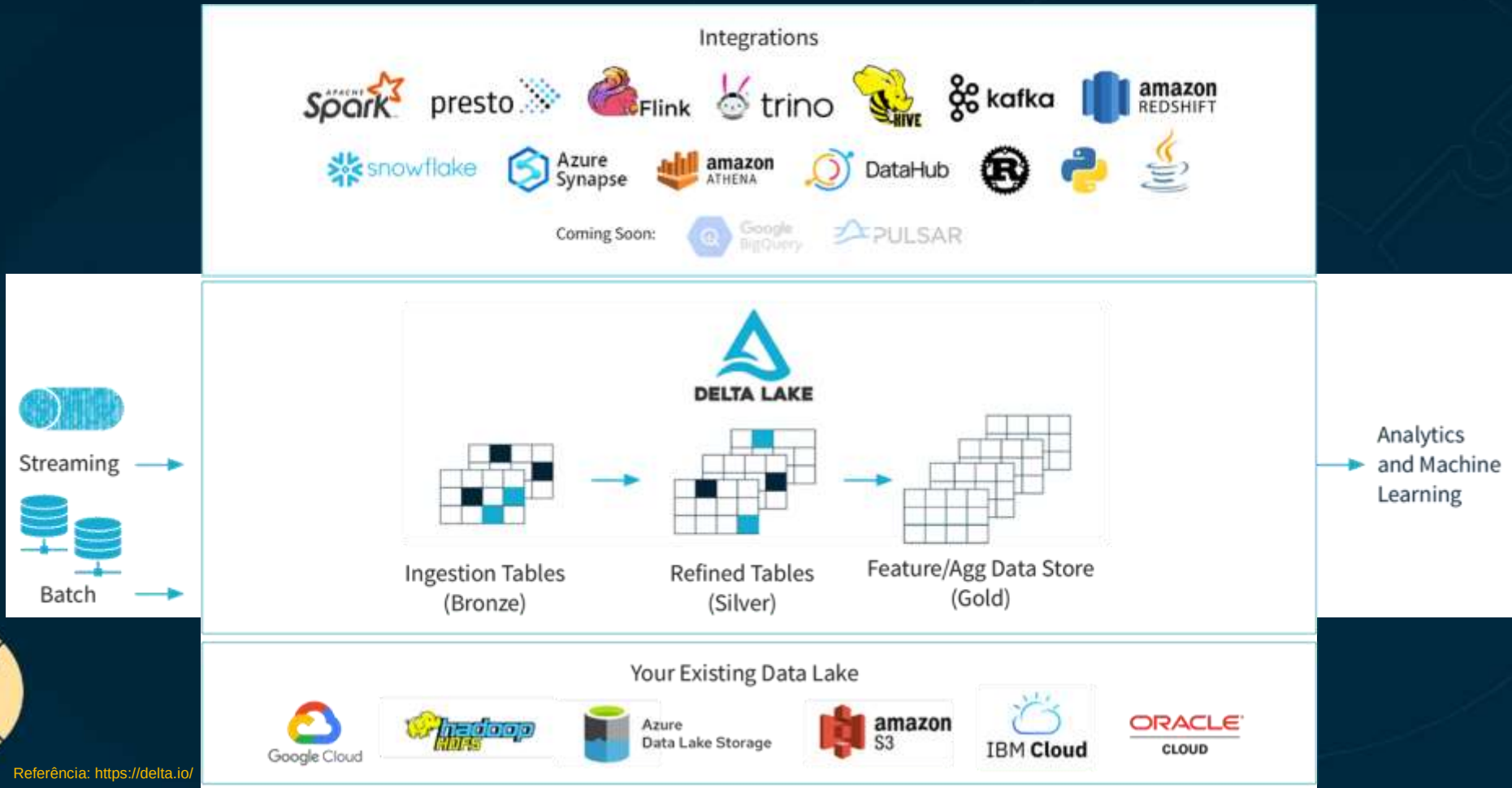
Mantido pela Linux Foundation e construído pelos criadores do Apache Spark

Formato aberto de armazenamento com suporte transacional e evolução de esquema

Referência: Learning Spark, 2nd Edition. O'Reilly Media. 2020. ISBN 9781492050049.

FullCycle

Arquitetura Lakehouse



Referência: https://delta.io/

FullCycle

Aula 22
Estratégia de Dados

FullCycle

Estratégia de Dados

FullCycle

Aula 23
Repositório Ideal de Dados

Repositório Ideal de Dados

O repositório ideal de dados entrega:

- **Escalabilidade**
- **Desempenho**
- **Transação ACID** — Atomicidade, Consistência, Isolamento, Durabilidade
- **Formatos diversos** — Estruturado, Semiestruturado, Não-estruturado
- **Cargas mistas** — SQL para BI, Batch de ETL, Streaming, AI e ML
- **Acessibilidade**

Referência: Learning Spark, 2nd Edition. O'Reilly Media. 2020. ISBN 9781492050049.

FullCycle

Aula 24
ACID

ACID

Atomicidade

Define todos os elementos que compõem uma transação completa do banco de dados.
Todas as operações são bem-sucedidas ou nenhuma delas.

Consistência

Define as regras para manter os pontos de dados em um estado correto após uma transação.
Alterações feitas em uma transação serão consistentes com as restrições vigentes (database constraints).

Isolamento

Mantém o efeito de uma transação invisível para as demais até que esta seja confirmada.
Todas as transações são executadas em um ambiente isolado, sem interferir umas nas outras.

Durabilidade

Garante que as alterações de dados se tornem permanentes quando a transação for confirmada.

"O que é RDBMS (Banco de Dados Relacional)?". Oracle. Disponível em <https://www.oracle.com/br/database/what-is-a-relational-database/>. Acessado em 13 de Fevereiro de 2024 às 14:30.
"What Does ACID Compliance Mean?". MongoDB. Disponível em <https://www.mongodb.com/databases/acid-compliance>. Acessado em 13 de Fevereiro de 2024 às 21:40.

Transação

É a representação de uma operação econômica no mundo real.
Exemplos: uma compra de produto; um saque de conta-corrente.

Em computação a transação representa o **estado transitório do dado** o qual é alterado por uma ou mais operações de transformação deste dado.

Um banco de dados transacional ou OLTP (Online Transaction Processing) permite adicionar ou modificar um grande conjunto de dados com muita concorrência e bastante desempenho, permitindo o processamento em tempo real de uma operação de negócio.

Transação ACID

ACID garante que os dados estarão em um **estado consistente e esperado** após a execução de um grupo de operações de leitura e gravação, isto é, uma transação que só **será bem-sucedida se todas as operações da transação forem bem-sucedidas**.

As transações podem impactar um único registro ou vários registros.

Uma **transação aderente à Atomicidade, Consistência, Isolamento e Durabilidade** é, portanto, uma transação ACID.

"What Does ACID Compliance Mean?". MongoDB. Disponível em <https://www.mongodb.com/databases/acid-compliance>. Acessado em 13 de Fevereiro de 2024 às 21:40.

FullCycle

Aula 25
Restrições de Banco de Dados (Database Constraints)

Consistência, integridade, precisão e confiabilidade

As restrições de banco de dados são um recurso importante dos sistemas de gerenciamento de banco de dados, pois garantem que as r**egras definidas na criação do modelo de dados sejam aplicadas** quando os dados são manipulados (inseridos, atualizados ou excluídos) em um banco de dados.

É uma **prática comum definir regras para os dados** de um banco de dados. Isto evita dados incorretos em uma coluna. Por exemplo: uma sequência de texto em uma coluna numérica ou um valor nulo em uma coluna com obrigatoriedade de preenchimento.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Restrições de Banco de Dados (Database Constraints)

Tipos comuns de restrições

DEFAULT **CHECK**

NOT NULL

UNIQUE **PRIMARY KEY**

FOREIGN KEY

Nem todos os sistemas de gerenciamento de banco de dados suportam os mesmos tipos de restrições.

Quando isso acontece, pode haver recursos ou considerações especiais para cada sistema específico.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Restrições de Banco de Dados (Database Constraints)

Restrição	Tabela	Coluna
DEFAULT	Apenas 1 coluna	Apenas 1 coluna
CHECK	Múltiplas colunas	Apenas 1 coluna
NOT NULL	Não aplicável	Apenas 1 coluna
UNIQUE	Múltiplas colunas	Apenas 1 coluna
PRIMARY KEY	Múltiplas colunas	Apenas 1 coluna
FOREIGN KEY	Múltiplas colunas	Apenas 1 coluna

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Aula 26
Restrições de Banco de Dados: DEFAULT (Database Constraints)

DEFAULT

Define um **valor inicial padrão a ser usado para uma determinada coluna** quando nenhum dado for fornecido no **momento da inserção**.

Se uma coluna com uma restrição DEFAULT for omitida na instrução INSERT, o banco de dados usará automaticamente o valor definido.

Se não houver DEFAULT definido e a coluna for omitida, o banco de dados atribui um valor NULL.

Os padrões podem ser valores fixos ou chamadas para funções SQL do sistema ou definidas pelo usuário.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constrainis-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

DEFAULT

Em Oracle:

ALTER TABLE empregado **ADD** (UPDATED_ON DATE **DEFAULT** SYSDATE);

Em MySQL:

CREATE TABLE empregado (uid **BINARY**(16) **DEFAULT** (UUID_TO_BIN(UUID())));

```
mysql> SELECT BIN_TO_UUID(uid) AS uid FROM empregado;
+--------------------------------------+
| uid                                  |
+--------------------------------------+
| f1109174-94c9-11e8-971d-3bf1095aa633 |
| f110cf9a-94c9-11e8-971d-3bf1095aa633 |
+--------------------------------------+
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

DEFAULT

Em SQL Server:

```
ALTER TABLE produto ADD CONSTRAINT DF_Produto_DataEntrada
DEFAULT GETDATE() FOR DataEntrada;

ALTER TABLE produto ADD CONSTRAINT DF_Produto_EstoqueAtual
DEFAULT 0 FOR EstoqueAtual;
```

Em PostgreSQL:

```
CREATE TABLE empregado (
    EmpregadoId INT NOT NULL,
    DataInicio DATE NOT NULL DEFAULT CURRENT_DATE
);
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Aula 27
Restrições de Banco de Dados: CHECK (Database Constraints)

CHECK

Impõe uma regra na(s) coluna(s) da tabela e define uma **condição lógica** a cada vez que uma linha é inserida ou atualizada, a **condição é verificada automaticamente** e um erro é gerado se a condição for falsa. A condição pode ser uma **expressão que avalia uma ou mais colunas**. Também pode incluir valores hardcoded, funções SQL do sistema ou definidas pelo usuário.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constrainis-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

CHECK

Em Oracle:

```
ALTER TABLE empregado ADD CONSTRAINT ck_empregado_salario CHECK (salary > 0);
```

Em MySQL:

```
CREATE TABLE t1 (
  CHECK (c1 <> c2),
  c1 INT CHECK (c1 > 10),
  c2 INT CONSTRAINT c2_positive CHECK (c2 > 0),
  c3 INT CHECK (c3 < 100),
  CONSTRAINT c1_nonzero CHECK (c1 <> 0),
  CHECK (c1 > c3)
);
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

CHECK

Em SQL Server:

```
ALTER TABLE Produto ADD CONSTRAINT CK_Produto_EstoqueAtual
CHECK (EstoqueAtual >= 0);

ALTER TABLE Produto ADD CONSTRAINT CK_Produto_Preco
CHECK (Preco > 0);
```

Em PostgreSQL:

```
CREATE TABLE empregado (
        EmpregadoId INT NOT NULL,
        Idade INT CHECK (Idade>=18)
);
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Aula 28
Restrições de Banco de Dados: NOT NULL (Database Constraints)

NOT NULL

Como o próprio nome indica, **impede que a coluna que a implementa armazene valores nulos**. Em outras palavras, deve sempre ter um valor.

Por padrão, **todas as colunas de uma tabela aceitam valores nulos** (NULL). Uma restrição NOT NULL impede que uma coluna aceite NULL como valor.

Em geral, as restrições NOT NULL não são nomeadas explicitamente.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

NOT NULL

Em Oracle:

```
CREATE TABLE impostos (   imposto_id NUMBER,
                          nome_imposto VARCHAR2(255) NOT NULL,
                                                     aliquota NUMBER(9,2),
PRIMARY KEY (imposto_id));

ALTER TABLE empregado MODIFY (primeiro_nome NOT NULL, ultimo_nome NOT NULL,
ultima_atualizacao NOT NULL);
```

Em MySQL:

```
CREATE TABLE impostos (
    imposto_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, nome VARCHAR(100)
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

NOT NULL

Em SQL Server:

```
ALTER TABLE Produto MODIFY      Produto_ID VARCHAR(20) NOT NULL,
                                Produto_Nome VARCHAR(100) NOT NULL,
                                Preco MONEY NOT NULL;
```

Em PostgreSQL:

```
CREATE TABLE PRODUTO (
    Produto_ID INT NOT NULL,
    Produto_Nome VARCHAR(100) NOT NULL
);
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Aula 29
Restrições de Banco de Dados: UNIQUE (Database Constraints)

UNIQUE

Use esta restrição para que **um valor de coluna específico seja exclusivo para cada registro da tabela**. Esta restrição proíbe a coluna que a implementa de armazenar valores duplicados. Também pode conter diversas colunas; nesse caso, a combinação destas colunas deve ser única.

As chaves exclusivas (UNIQUE KEYs) são definidas no nível da tabela e podem incluir uma ou mais colunas. Eles garantem que os valores de uma linha não se repetem em outra. Você pode criar quantas chaves exclusivas forem necessárias em cada tabela para garantir que todas as regras de negócios associadas à exclusividade sejam aplicadas.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

UNIQUE

Em Oracle:

```
ALTER TABLE departmento  ADD CONSTRAINT nome_departamento_local_uk
                         UNIQUE (nome, local);
```

Em MySQL:

```
CREATE TABLE departamento (
    departamento_ID INT,
    nome VARCHAR(100),
    UNIQUE (nome)
);
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

UNIQUE

Em SQL Server:

```
ALTER TABLE produto ADD CONSTRAINT UK_Produto_ProdutoID
UNIQUE (ProdutoID);
ALTER TABLE produto ADD CONSTRAINT UK_Produto_ProdutoNome
UNIQUE (ProdutoNome);
```

Em PostgreSQL:

```
CREATE TABLE departamento (
    departamento_ID INT NOT NULL,
    departamento_nome VARCHAR(100) NOT NULL,
    CONSTRAINT departamento_nome_uk UNIQUE (departamento_nome),
    email VARCHAR(50) UNIQUE );
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Aula 30
Restrições de Banco de Dados: PRIMARY KEY (Database Constraints)

PRIMARY KEY

É **definida em uma ou mais colunas de uma tabela**. Indica que esta coluna (ou conjunto de colunas) **deve identificar exclusivamente cada linha da tabela**. As colunas que fazem parte da **PRIMARY KEY** devem obedecer às restrições **UNIQUE e NOT NULL ao mesmo tempo**, ou seja, a(s) coluna(s) não pode(m) conter valores duplicados ou nulos.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

PRIMARY KEY

Em Oracle:

```
CREATE TABLE empregado ( "EMPREGADO_ID" NUMBER,
"PRIMEIRO_NOME" VARCHAR2(30),
"ULTIMO_NOME" VARCHAR2(30),
"GERENTE" NUMBER,
PRIMARY KEY ("EMPREGADO_ID")
);
```

Em MySQL:

```
CREATE TABLE departamento ( departamento_ID INT, nome VARCHAR(100),
PRIMARY KEY (departamento_ID)
);
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

PRIMARY KEY

Em SQL Server:

```
ALTER TABLE produto ADD CONSTRAINT PK_Produto
PRIMARY KEY (ProdutoID);
```

Em PostgreSQL:

```
CREATE TABLE produto (
    produto_ID INT NOT NULL,
    produto_nome VARCHAR(100) NOT NULL,
    CONSTRAINT pk_produto PRIMARY KEY (produto_ID)
);
```

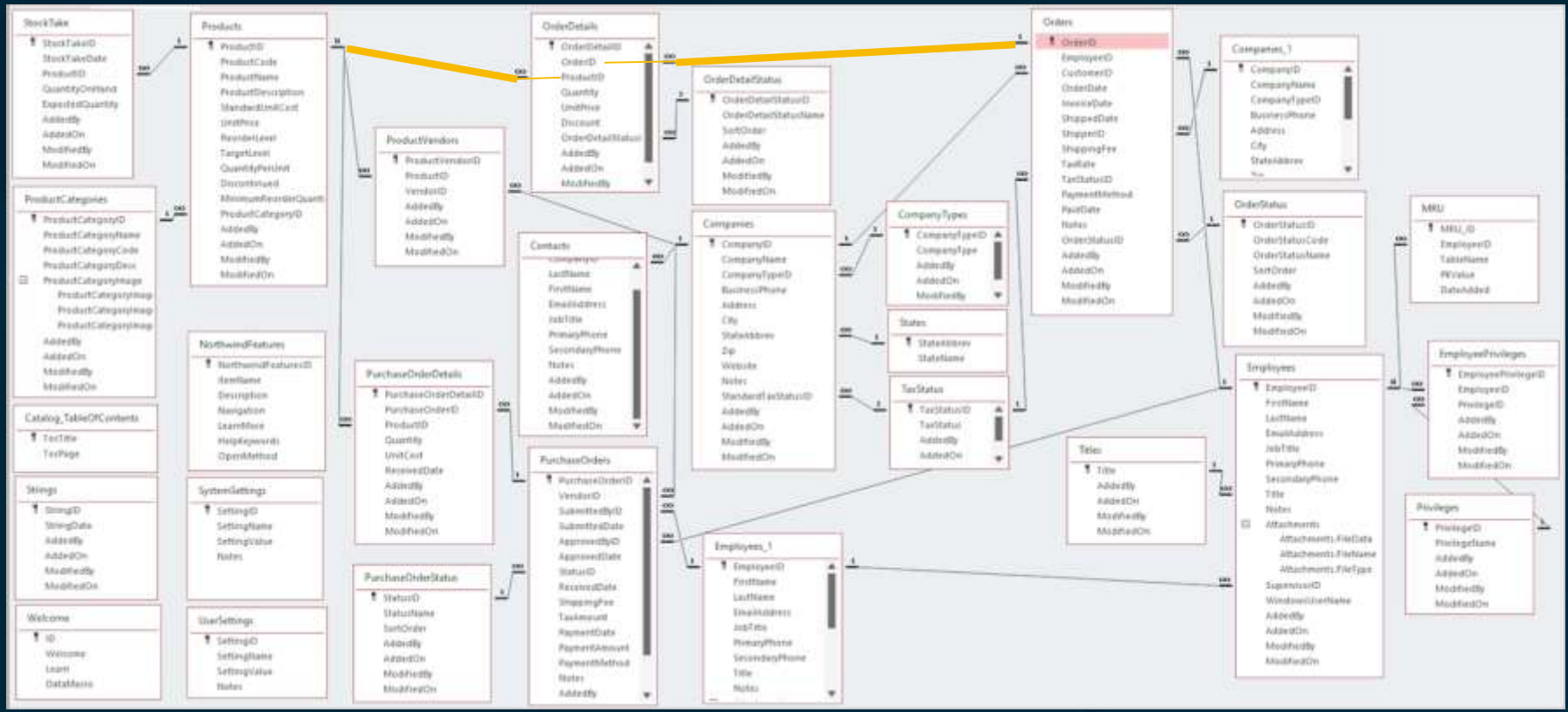
"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Aula 31
Restrições de Banco de Dados: FOREIGN KEY (Database Constraints)

FullCycle

Restrições de Banco de Dados (Database Constraints) - FOREIGN KEY



https://support.content.office.net/en-us/media/559a04f2-11b2-44b8-ae4a-92284d1576bd.png

FOREIGN KEY

É definida em uma ou mais colunas de uma tabela como **uma referência à(s) coluna(s) PRIMARY KEY** de outra tabela. A restrição FOREIGN KEY cria um **relacionamento entre as tabelas**.

É vital para manter a integridade referencial em um banco de dados. Garante que cada linha em uma tabela filha (como Pedido) tenha uma e somente uma linha associada em uma tabela pai (como Produto).

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FOREIGN KEY

As chaves estrangeiras são criadas em tabelas filhas e “referenciam” uma tabela pai. Para poder fazer referência a uma tabela, deve existir uma restrição que garanta a exclusividade (**UNIQUE** ou **PRIMARY KEY**) para as colunas referenciadas da tabela pai.

Cada valor inserido ou atualizado nas colunas que fazem parte de uma **FOREIGN KEY existe exatamente uma vez na tabela pai**. Não é possível inserir ou atualizar uma linha com referência a outra linha que não exista na tabela pai. O registro **pai não poderá ser excluído** caso existam filhos.

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FOREIGN KEY

Em Oracle:

```
CREATE TABLE empregado (
  "EMPREGADO_ID" NUMBER,
  "GERENTE" NUMBER,
      CONSTRAINT "GERENTE_FK"
      FOREIGN KEY ("GERENTE")
      REFERENCES empregado ("EMPREGADO_ID") ENABLE,
  "DEPARTMENTO_ID" NUMBER,
      CONSTRAINT "DEPARTMENTO_FK"
      FOREIGN KEY ("DEPARTMENTO_ID")
      REFERENCES departamento ("DEPARTMENTO_ID") ENABLE
);
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FOREIGN KEY

Em MySQL:

```
CREATE TABLE product_order (
    product_no INT NOT NULL AUTO_INCREMENT,          product_category INT NOT
NULL,
    product_id INT NOT NULL,
            customer_id INT NOT NULL,

    PRIMARY KEY(product_no),
    INDEX (product_category, product_id),
    INDEX (customer_id),

    FOREIGN KEY (product_category, product_id) REFERENCES product(category, id)
    FOREIGN KEY (customer_id) REFERENCES customer(id)
) ENGINE=INNODB;
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What Are They, And How To Define Them In Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FOREIGN KEY

Em SQL Server:

```
ALTER TABLE OrderDetail ADD CONSTRAINT FK_OrderDetail_ProductID
    FOREIGN KEY (ProductID) REFERENCES Product (ProductID);
ALTER TABLE PurchaseDetail ADD CONSTRAINT FK_PurchaseDetail_ProductCode
    FOREIGN KEY (ProductCode) REFERENCES Product (ProductCode);
```

Em PostgreSQL:

```
ALTER TABLE livraria ADD CONSTRAINT livraria_livro
        FOREIGN KEY (ISBN) REFERENCES livro (ISBN)
        NOT DEFERRABLE
        INITIALLY IMMEDIATE
;
```

"What Are the Different Types of Database Constraints?". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constrainis-types/>. Acessado em 14 de Fevereiro de 2024 às 12:45.
"Database Constraints: What They Are and How to Define Them in Vertabelo". Vertabelo. Disponível em <https://vertabelo.com/blog/database-constraints/>. Acessado em 14 de Fevereiro de 2024 às 12:50.
"Defining, Constraining, and Manipulating Your Entities". Oracle. Disponível em <https://blogs.oracle.com/connect/post/defining-constraining-and-manipulating-your-entities>. Acessado em 14 de Fevereiro de 2024 às 12:55.

FullCycle

Aula 32
Comentários sobre Restrições de Banco de Dados (Database Constraints)

FIQUE ATENTO ÀS DOCUMENTAÇÕES ESPECÍFICAS!

Cada banco de dados apresenta implementações específicas de restrições.

3.143 ALL_CONSTRAINTS

ALL_CONSTRAINTS describes constraint definitions on tables accessible to the current user.

Related Views

- DBA_CONSTRAINTS describes all constraint definitions in the database.
- USER_CONSTRAINTS describes constraint definitions on tables in the current user's schema.

Column	Datatype	NULL	Description
OWNER	VARCHAR2(128)		Owner of the constraint definition
CONSTRAINT_NAME	VARCHAR2(128)		Name of the constraint definition If the constraint is defined in an application usage domain, this column displays a system-generated constraint name. The constraint name specified in the application usage domain is displayed in DOMAIN_CONSTRAINT_NAME.
CONSTRAINT_TYPE	VARCHAR2(1)		Type of the constraint definition: • C - Check constraint on a table • P - Primary key • U - Unique key • R - Referential integrity • V - With check option, on a view • O - With read only, on a view • H - Hash expression • F - Constraint that involves a REF column • S - Supplemental logging

"3.143 ALL_CONSTRAINTS". Oracle. Disponível em <https://docs.oracle.com/en/database/oracle/oracle-database/23/refrn/ALL_CONSTRAINTS.html#GUID-9C96DA92-CFE0-4A3F-9061-C5ED17B43EFE>. Acessado em 14 de Fevereiro de 2024 às 12:55.

NoSQL também dispõe de técnicas para integridade!

Esquema flexível de dados não significa obrigatoriamente inconsistência.

Método	Descrição	Impacto no Desempenho	Caso de Uso
Transações	As atualizações para múltiplas coleções ocorrem em uma única operação atômica.	Potencialmente alto, devido à contenção de leitura	Seu aplicativo deve sempre retornar dados atualizados e pode tolerar possíveis impactos negativos no desempenho durante períodos de leituras intensas.
Documentos Embarcados	Modifique o esquema do aplicativo para incorporar dados relacionados em uma única coleção.	Baixo a moderado, dependendo do tamanho do documento e dos índices	Seu aplicativo sempre lê e atualiza os dados relacionados ao mesmo tempo. Esta solução simplifica seu esquema e evita a necessidade de operações de $lookup.
MongoDB Atlas Triggers	Quando ocorre uma atualização em uma coleção, os gatilhos atualizam automaticamente outra coleção.	Baixa a moderada, com possíveis atrasos no processamento de eventos acionados	Seu aplicativo pode tolerar a leitura de dados ligeiramente desatualizados. Os usuários poderão ver dados desatualizados se executarem uma consulta imediatamente após uma atualização, mas antes que o gatilho termine de atualizar a segunda coleção.

Imagem traduzida de "Data Consistency" em <https://www.mongodb.com/docs/manual/data-modeling/data-consistency/#std-label-data-modeling-data-consistency>. Acessado em 20 de Janeiro de 2024 às 23:00.

Validação do Esquema JSON em MongoDB

```
db.createCollection("estudantes", {
   validator: {
      $jsonSchema: {
         bsonType: "object",
         title: "Validação de Objeto Estudante",
         required: [ "local", "curso", "nome", "ano" ],
         properties: {
            nome: {
               bsonType: "string",
               description: "'nome' é obrigatório e composto por caracteres"
            },
            ano: {
               bsonType: "int",
               minimum: 2017,
               maximum: 3017,
               description: "'ano' é obrigatório e número inteiro [ 2017, 3017 ]"
            }
         }
      }
   }
} )
```

https://www.mongodb.com/docs/manual/core/schema-validation/specify-json-schema/

Validação do Esquema Grafo em Neo4j

```
@prefix neo4j: <http://neo4j.com/myvoc#> .
@prefix sh: <http://www.w3.org/ns/shacl#> .

neo4j:PersonShape a sh:NodeShape ;
  sh:targetClass neo4j:Person ;
  sh:property [
    sh:path neo4j:name ;
    sh:pattern "^\w[\s\w\.]*$" ;
    sh:maxCount 1 ;
    sh:datatype xsd:string ;
  ];
  sh:property [
    sh:path neo4j:ACTED_IN ;
    sh:class neo4j:Movie ;
    sh:nodeKind sh:IRI ;
  ] ;
.
```

"target"	"propertyOrRelationshipPath"	"param"	"value"
"Person"	"name"	"datatype"	"string"
"Person"	"name"	"pattern"	"^\w[\s\w\.]*$"
"Person"	"name"	"maxCount"	1

https://neo4j.com/labs/neosemantics/4.0/validation/

Definição do Esquema de Dados em DynamoDB

```
{
    TableName : "Music",
    KeySchema: [
        {
            AttributeName: "Artist",
            KeyType: "HASH", //Partition key
        },
        {
            AttributeName: "SongTitle",
            KeyType: "RANGE" //Sort key
        }
    ],
    AttributeDefinitions: [
        {
            AttributeName: "Artist",
            AttributeType: "S"
        },
        {
            AttributeName: "SongTitle",
            AttributeType: "S"
        }
    ],
    ProvisionedThroughput: {       // Only specified if using provisioned mode
        ReadCapacityUnits: 1,
        WriteCapacityUnits: 1
    }
}
```

https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/SQLtoNoSQL.CreateTable.html

Definição do Esquema de Dados em Cassandra

```
CREATE KEYSPACE reservation WITH replication = {'class':
  'SimpleStrategy', 'replication_factor' : 3};

CREATE TYPE reservation.address (
  street text,
  city text,
  state_or_province text,
  postal_code text,
  country text );

CREATE TABLE reservation.reservations_by_confirmation (
  confirm_number text,
  hotel_id text,
  start_date date,
  end_date date,
  room_number smallint,
  guest_id uuid,
  PRIMARY KEY (confirm_number) )
  WITH comment = 'Q6. Find reservations by confirmation number';
```

https://cassandra.apache.org/doc/4.1/cassandra/data_modeling/data_modeling_schema.html

FullCycle

Aula 33
Teorema CAP
Parte 1

FullCycle

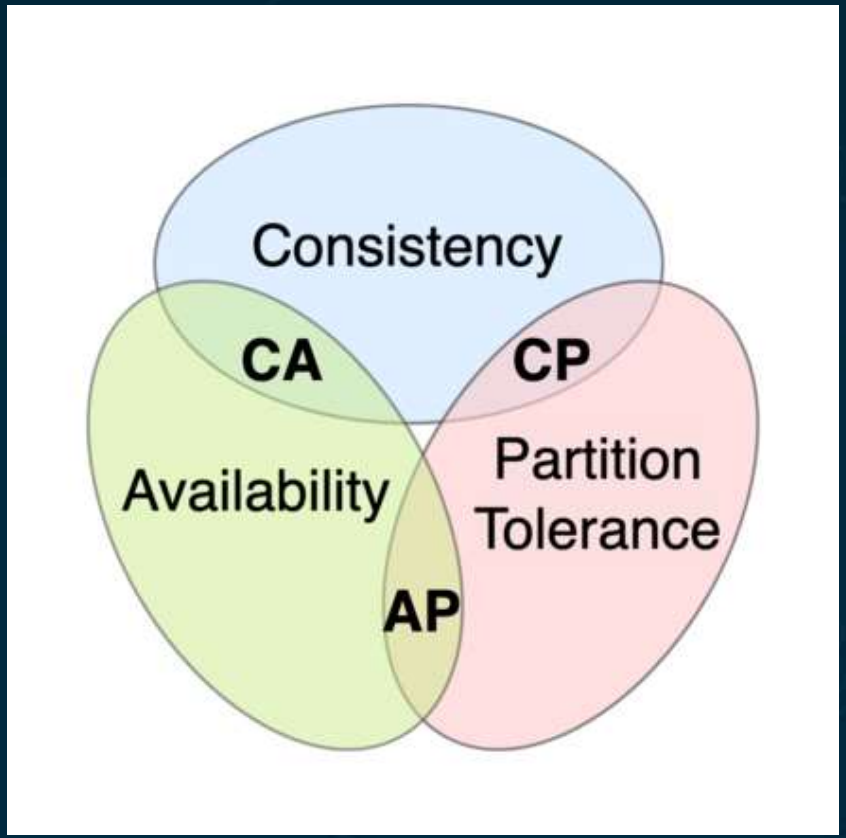
Teorema CAP - Definição Clássica (2000)

Também chamado de **Teorema de Brewer**:

Qualquer **armazenamento de dados distribuído** pode fornecer **apenas 2 das 3 garantias**:

Consistência - Consistency
Cada leitura recebe a escrita mais recente ou um erro.

Disponibilidade - Availability
Cada requisição recebe uma resposta sem a garantia de que esta contém a escrita mais recente.



"CAP theorem". Wikipedia. Disponível em <https://en.wikipedia.org/wiki/CAP_theorem>. Acessado em 16 de Fevereiro de 2024 às 13:00.

FullCycle

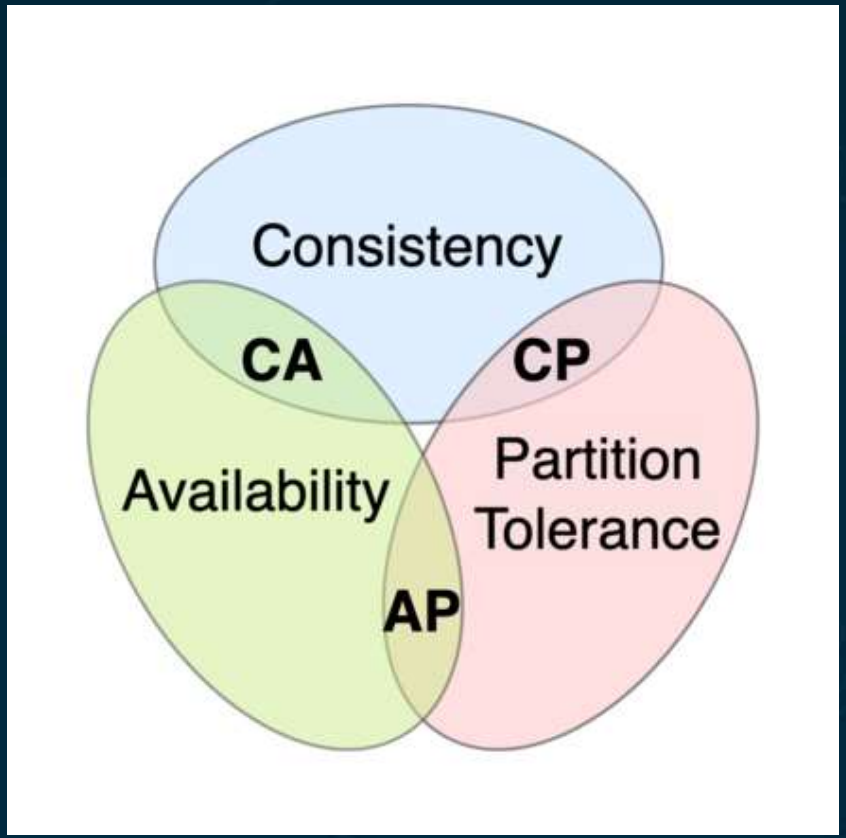
Teorema CAP - Definição Clássica (2000)

Tolerância à Partição - Partition Tolerance
O sistema **continua a operar** mesmo com perdas e/ou atrasos de mensagens ocasionados pela rede entre os nós ou seja, **partição de rede**.

Durante uma partição de rede, decide-se entre:

1) **cancelar a operação** e **diminuir a disponibilidade**, mas, **garantir a consistência**

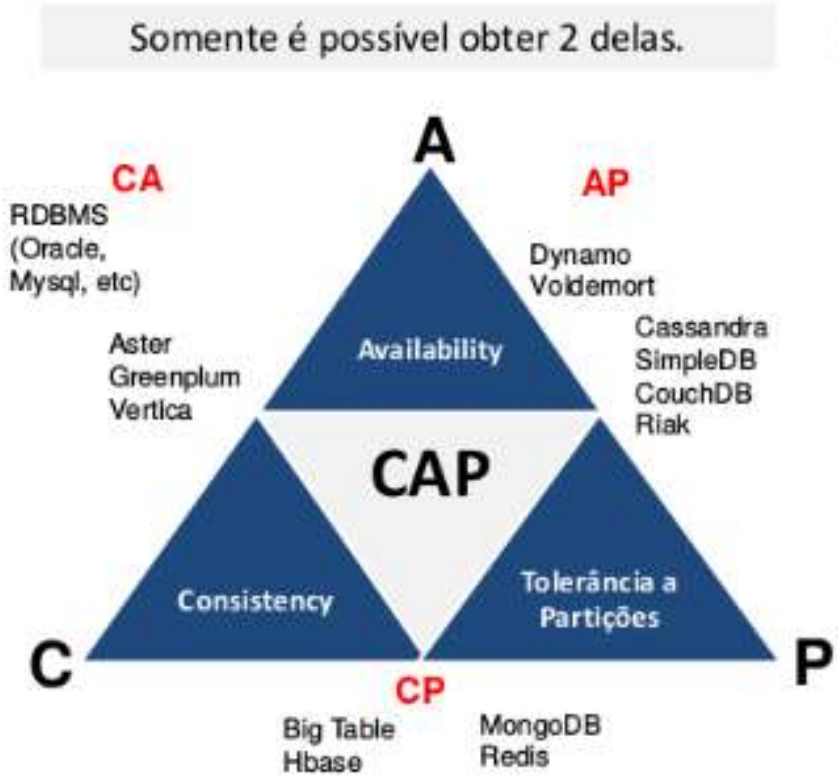
1) **prosseguir com a operação** e **fornecer disponibilidade**, mas corre o **risco de inconsistência**.



"CAP theorem". Wikipedia. Disponível em <https://en.wikipedia.org/wiki/CAP_theorem>. Acessado em 16 de Fevereiro de 2024 às 13:00.

FullCycle

Teorema CAP - Definição Clássica (2000)



Sistemas distribuídos, escolher 2:

- ✓ **Consistency**: Todos os nodes veem os mesmos dados
- ✓ **Availability**: Todos os pedidos recebem uma resposta
- ✓ **Partition Tolerance**: Sistema funcional após falhas de rede

O **segredo** na hora de escolher a ferramenta está no **entendimento do negócio e suas regras**.

"SQL, NoSQL ou newSQL?". Vinícius Vale. Disponível em <https://www.linkedin.com/pulse/sql-nosql-ou-newsql-vinicius-aquino-do-vale>. Acessado em 16 de Fevereiro de 2024 às 13:00.

FullCycle

Aula 34
Teorema CAP
Parte 2

Teorema CAP - 12 anos depois (2012)

Qualquer sistema de dados compartilhados em rede pode ter **apenas 2 das 3 propriedades** desejáveis do Teorema CAP.

Ao **manipular explicitamente as partições de rede**, as pessoas desenvolvedoras podem **otimizar a consistência e a disponibilidade**, conseguindo assim **alguma compensação** entre os três estados desejáveis de Consistência, Disponibilidade e Tolerância à Partição.

Embora as pessoas desenvolvedoras ainda precisem **escolher entre consistência e disponibilidade quando há partições presentes**, há uma variedade incrível de flexibilidade para lidar com partições e a respectiva recuperação destas.

"CAP Twelve Years Later: How the rules have changed". InfoQ. Disponível em <https://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed/>. Acessado em 16 de Fevereiro de 2024 às 13:30.

Teorema CAP - 12 anos depois (2012)

O objetivo do CAP moderno é maximizar combinações de consistência e disponibilidade que façam sentido para a aplicação.

O movimento NoSQL trata da criação de **possibilidades que se concentram primeiro na disponibilidade e depois na consistência;** bancos de dados que aderem às propriedades ACID (atomicidade, consistência, isolamento e durabilidade) **fazem o oposto**.

Quando em P, **a escolha entre C e A pode ocorrer muitas vezes** dentro do mesmo sistema com granularidade muito fina; não apenas os subsistemas podem fazer escolhas diferentes, mas **a escolha pode mudar de acordo com a operação** ou mesmo com os **dados** específicos ou com o **usuário** envolvido.

"CAP Twelve Years Later: How the rules have changed". InfoQ. Disponível em <https://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed/>. Acessado em 16 de Fevereiro de 2024 às 13:30.

Teorema CAP - 12 anos depois (2012)

CAP permite C e A perfeitos na maioria das vezes.

Quando partições estão presentes ou são percebidas, uma estratégia que **detecte partições e as considere explicitamente** é adequada em 3 etapas: **detectar partições**, **entrar em um modo de partição explícito** que possa **limitar algumas operações** e iniciar um **processo de recuperação** para restaurar a consistência e compensar erros cometidos durante uma partição.

"CAP Twelve Years Later: How the rules have changed". InfoQ. Disponível em <https://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed/>. Acessado em 16 de Fevereiro de 2024 às 13:30.

FullCycle

Aula 35
Teorema CAP
Parte 3

FullCycle

Relembrando ACID para falar de CAP (e, depois, de BASE)

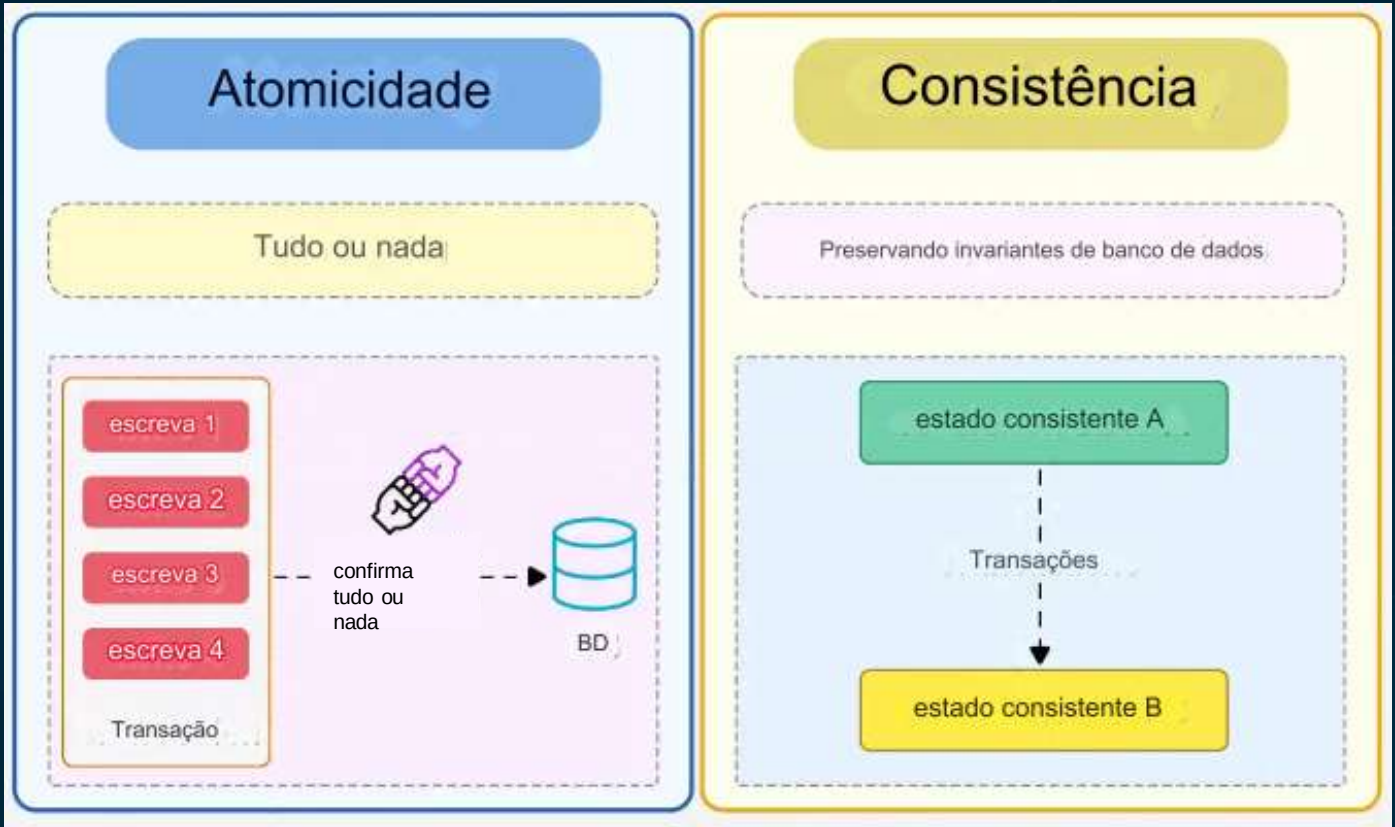


Imagem traduzida de "What does ACID mean?" em <https://blog.bytebytego.com/p/ep89-what-does-acid-mean>.
Acessado em 16 de Fevereiro de 2024 às 12:30.

FullCycle

Relembrando ACID para falar de CAP (e, depois, de BASE)

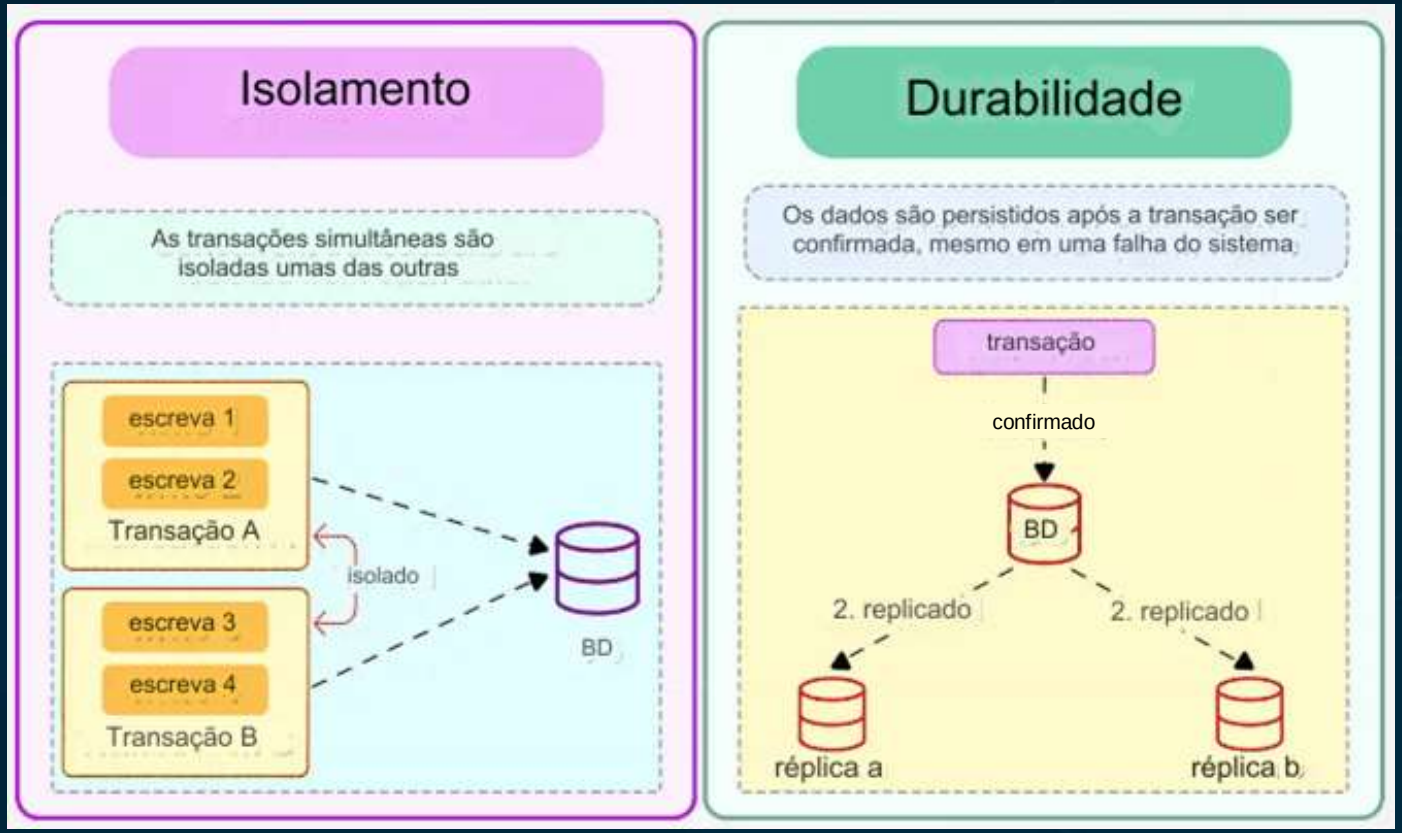


Imagem traduzida de "What does ACID mean?" em <https://blog.bytebytego.com/p/ep89-what-does-acid-mean>.
Acessado em 16 de Fevereiro de 2024 às 12:30.

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

O Teorema CAP é **demasiado simplista e mal compreendido** para a tomada de decisão sobre qual banco de dados utilizar.

Consistência na CAP significa linearização, que é uma **noção muito específica e muito forte de consistência**. Em particular, **não tem nada a ver com o C no ACID**, embora esse C também signifique “consistência”.

Consistência no ACID é a garantia de que uma transação só pode levar o banco de dados **de um estado consistente para outro consistente**.

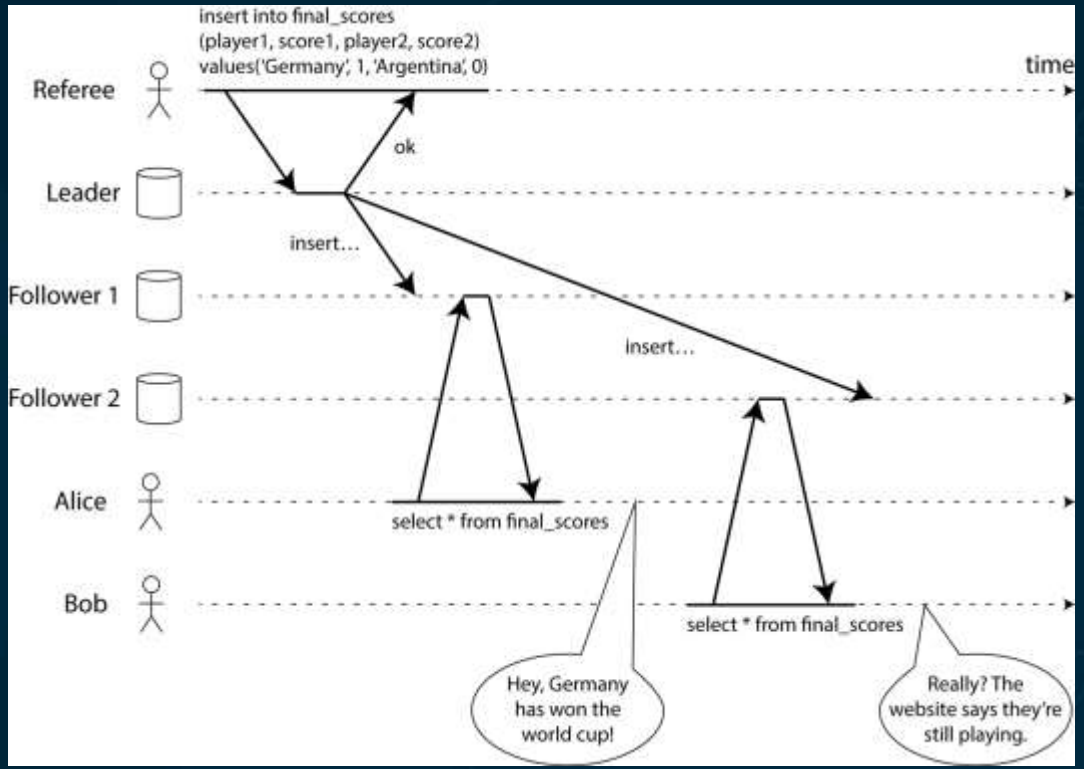
"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

FullCycle

Aula 36
Teorema CAP
Parte 4

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

Se a operação B tiver sido iniciada após a conclusão com êxito da operação A, então, a operação B deverá ver o sistema no mesmo estado em que estava na conclusão da operação A ou em um estado mais recente.



"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

FullCycle

Aula 37
Teorema CAP
Parte 5

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

Disponibilidade no CAP é definida como:

“cada solicitação recebida por um nó sem falha [de banco de dados] no sistema deve resultar em uma resposta [sem erro]”.

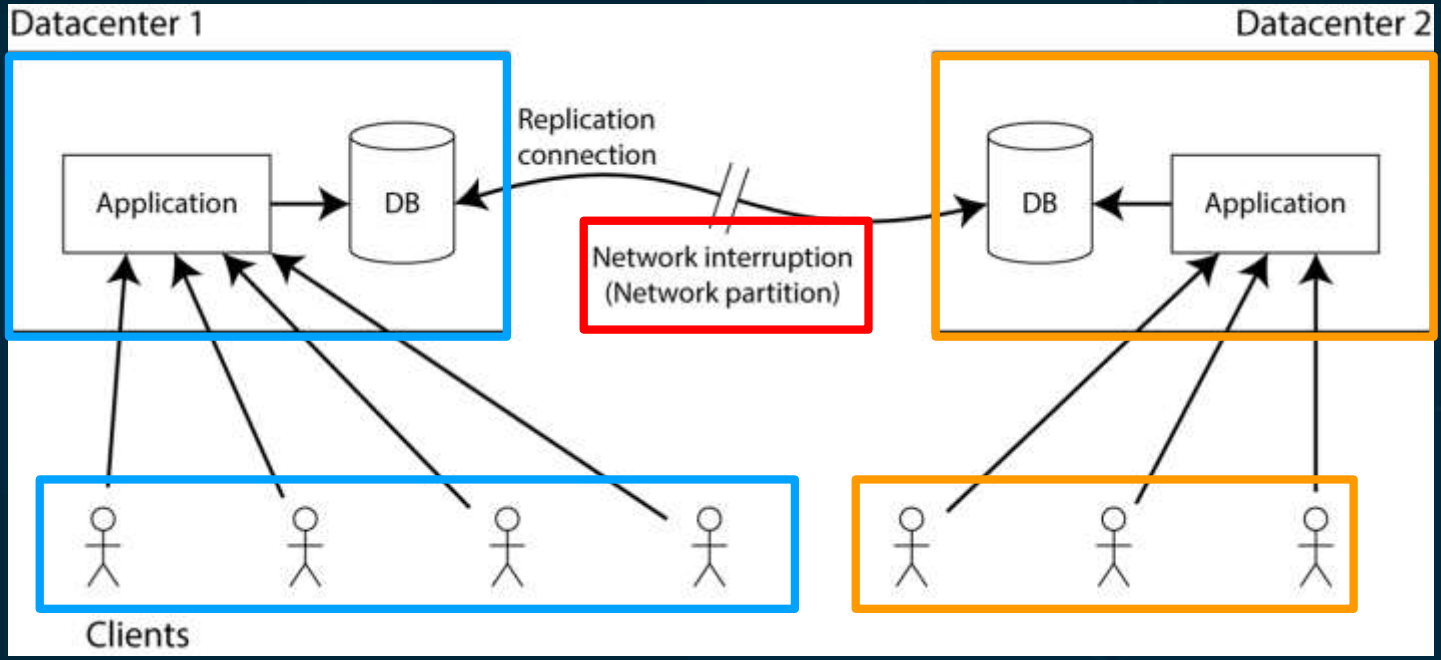
Não é suficiente que **algum** nó seja capaz de lidar com a solicitação: **qualquer** nó que não apresente falha precisa ser capaz de lidar com ela.

Muitos sistemas chamados de “altamente disponíveis” (ou seja, com baixo tempo de inatividade), na verdade, não atendem a essa definição de disponibilidade.

"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

Replicação: sempre que dados forem gravados em um datacenter, também deverão ser gravados na réplica do outro datacenter.



"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

Escolha possíveis:

1) O aplicativo continua tendo permissão para escrever no banco de dados. Portanto **permanece totalmente disponível em ambos os datacenters**. Entretanto, enquanto o link de replicação for interrompido, quaisquer alterações gravadas em um datacenter não aparecerão no outro datacenter. **Isso viola a linearizabilidade**.

2) Para não perder a linearização, todas as leituras e escritas precisam ocorrer em um único datacenter. No outro datacenter (que não pode ser atualizado devido à falha no link de replicação), o banco de dados deve parar de aceitar leituras e gravações até que a partição de rede seja reparada e o banco de dados esteja sincronizado novamente.

"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

Assim, embora o outro banco de dados **não tenha falhado**, ele não pode processar solicitações, portanto **não está disponível para CAP**.

Portanto, se um sistema escolhe a linearização, isso **não significa necessariamente que uma partição de rede leva automaticamente a uma interrupção do aplicativo**.

Disponibilidade na vida real $\neq$ **Disponibilidade no Teorema CAP**

"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

FullCycle

Aula 38
Teorema CAP
Parte 6

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

Na prática, os sistemas **multi datacenter** são frequentemente projetados com **replicação assíncrona** e, portanto, n**ão linearizáveis**.

No entanto, **a razão para essa escolha é** muitas vezes a **latência** das redes de longa distância, e não apenas o desejo de tolerar falhas de datacenter e de rede.

"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

Muitos sistemas não são linearizáveis (consistentes) nem CAP-disponíveis.

Considere qualquer **banco de dados replicado com um único líder**, que é a forma padrão na **maioria dos bancos de dados relacionais**.

Se um cliente for particionado do líder, **não poderá escrever**.

Mesmo que ele possa ler de um seguidor (uma réplica somente leitura), o fato de não poder escrever significa que **toda configuração de líder único não é CAP-disponível**. Tais configurações são frequentemente comercializadas como “alta disponibilidade”.

"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

Teorema CAP - Não chame seu banco de dados de CP ou AP (2015)

Se a replicação de líder único não é CAP-disponível ("AP") então é “CP”?

Se for permitido que o aplicativo faça leituras de um seguidor e a **replicação for assíncrona** (o padrão na maioria dos bancos de dados), um seguidor poderá ficar um pouco atrás do líder quando ocorrer a leitura dele.

Nesse caso, **suas leituras não serão linearizáveis**, ou seja, **não é CAP-consistente**.

Nem, nem… Nem CP, nem AP.

"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.

FullCycle

Aula 39
Teorema PACELC

CP/AP: uma falsa dicotomia

O fato de não termos conseguido classificar nenhum banco de dados como **inequivocamente “AP” ou “CP”** deve significar algo: **esses simplesmente não são os rótulos corretos para descrever sistemas**.

Como **alternativa ao CAP**, propomos um simples f**ramework de sensibilidade ao atraso** para dispor sobre as compensações entre garantias de consistência e tolerância de rede em um banco de dados replicado (A Critique of the CAP Theorem. Martin Kleppmann).

"Please stop calling databases CP or AP". Martin Kleppmann. Disponível em <https://martin.kleppmann.com/2015/05/11/please-stop-calling-databases-cp-or-ap.html>. Acessado em 16 de Fevereiro de 2024 às 18:00.
"A Critique of the CAP Theorem". Martin Kleppmann. Disponível em <https://arxiv.org/pdf/1509.05393.pdf>. Acessado em 16 de Fevereiro de 2024 às 18:00.

PACELC

Se **P** então compensação entre **AC** **E**lse (senão) compensação entre **LC**
(Daniel Abadi, 2010)

Quando há particionamento é necessário decidir entre disponibilidade (A) ou consistência (C).

Quando não há particionamento é necessário decidir entre latência (L) ou consistência (C).

"PACELC theorem". Wikipedia. Disponível em <https://en.wikipedia.org/wiki/PACELC_theorem>. Acessado em 16 de Fevereiro de 2024 às 18:00.

FullCycle

Teorema PACELC

Um requisito de alta disponibilidade implica que o sistema deve replicar dados.

Assim que um sistema distribuído replica dados, surge uma compensação entre consistência e latência.

DDBS	P+A	P+C	E+L	E+C
Aerospike[8]	✓	paid only	optional	✓
Bigtable/HBase		✓		✓
Cassandra	✓		✓[a]	
Cosmos DB		✓	✓[b]	
Couchbase		✓	✓	✓
Dynamo	✓		✓[a]	
DynamoDB		✓	✓	✓
FaunaDB[10]		✓	✓	✓
Hazelcast IMDG[6][7]	✓	✓	✓	✓
Megastore		✓		✓
MongoDB	✓			✓
MySQL Cluster		✓		✓
PNUTS		✓	✓	
PostgreSQL	✓	✓	✓	✓
Riak	✓		✓[a]	
VoltDB/H-Store		✓		✓

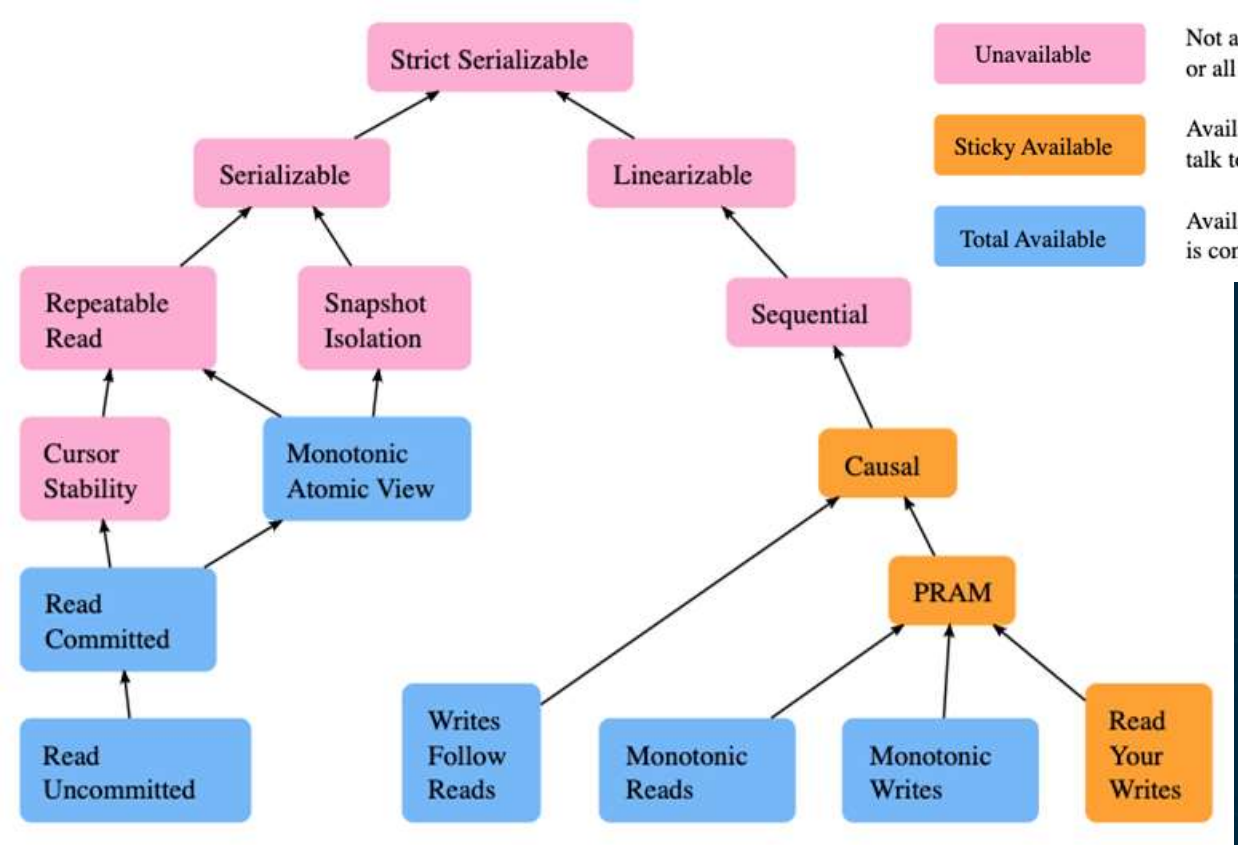
"PACELC theorem". Wikipedia. Disponível em <https://en.wikipedia.org/wiki/PACELC_theorem>. Acessado em 16 de Fevereiro de 2024 às 18:00.

FullCycle

Aula 40
Modelos de Consistência

FullCycle

Modelos de Consistência



"Consistency Models". JEPSEN. Disponível em <https://jepsen.io/consistency>. Acessado em 16 de Fevereiro de 2024 às 18:00.

FullCycle

Aula 41
BASE

BA

Basically Available: basicamente disponível, ou seja, o sistema **parece** estar funcionando o tempo todo.

S

Soft State: em estado leve, o sistema **não precisa ser consistente** o tempo todo.

E

Eventual Consistency: eventualmente consistente, o sistema torna-se consistente com o tempo.

"SQL, NoSQL ou newSQL?". Vinícius Vale. Disponível em <https://www.linkedin.com/pulse/sql-nosql-ou-newsql-vinicius-aquino-do-vale>. Acessado em 16 de Fevereiro de 2024 às 13:00.

Resolução de Conflitos

Para garantir a **convergência das réplicas**, um sistema deve **reconciliar as diferenças** entre múltiplas cópias de dados distribuídos. Isto consiste em duas partes:

1) troca de versões ou atualizações de dados entre servidores (frequentemente conhecido como **antientropia**);

2) escolher um **estado final apropriado** quando ocorrerem atualizações simultâneas, chamado **reconciliação**.

"Eventual Consistency". Wikipedia. Disponível em <https://en.wikipedia.org/wiki/Eventual_consistency>. Acessado em 16 de Fevereiro de 2024 às 13:00.

FullCycle

Aula 42
BASE
Momento da Reconciliação

BASE: Momento da Reconciliação

Resolução de Conflitos

A **reconciliação de escritas simultâneas** deve ocorrer algum tempo **antes da próxima leitura** e pode ser agendada em instantes diferentes:

- Reparo de leitura: A correção é feita quando uma leitura encontra uma inconsistência. Isso **retarda a operação de leitura**.
- Reparo de escrita: A correção ocorre durante uma operação de gravação, **retardando a operação de escrita**.
- Reparo assíncrono: A correção não faz parte de uma operação de leitura ou gravação.

"Eventual Consistency". Wikipedia. Disponível em <https://en.wikipedia.org/wiki/Eventual_consistency>. Acessado em 16 de Fevereiro de 2024 às 13:00.

FullCycle

Aula 43
Bloqueio e Simultaneidade

Bloqueio e Simultaneidade

Conflitos podem surgir em um banco de dados quando **vários usuários ou aplicativos tentam alterar os mesmos dados ao mesmo tempo**.

Técnicas de **bloqueio e simultaneidade reduzem o potencial de conflitos**, mantendo a integridade dos dados.

O bloqueio impede que outros usuários e aplicativos acessem dados enquanto estão sendo atualizados.

"O que é RDBMS (Banco de Dados Relacional)?". Oracle. Disponível em <https://www.oracle.com/br/database/what-is-a-relational-database/>. Acessado em 13 de Fevereiro de 2024 às 14:30.

Bloqueio e Simultaneidade

Em alguns bancos de dados, **o bloqueio se aplica à tabela inteira**, o que cria um impacto negativo no desempenho do aplicativo.

Outros aplicam bloqueios no **nível de registro**, deixando os **outros registros dentro da tabela disponíveis**, ajudando a garantir um melhor desempenho do aplicativo.

A **simultaneidade** gerencia a atividade quando vários usuários ou aplicativos fazem consultas **ao mesmo tempo** no mesmo banco de dados. Esse recurso fornece o acesso correto de acordo com as políticas definidas para o controle de dados.

"O que é RDBMS (Banco de Dados Relacional)?". Oracle. Disponível em <https://www.oracle.com/br/database/what-is-a-relational-database/>. Acessado em 13 de Fevereiro de 2024 às 14:30.

FullCycle

Aula 44
Bloqueio Otimista

Bloqueio Otimista

- Pressupõe que **conflitos e erros são raros**.
- A maioria das transações pode ser **concluída sem interferência**.
- Não bloqueia os dados antes da leitura ou escrita, mas, **verifica se há alterações no final da transação**.
- Se outra transação tiver modificado os dados, a transação atual será abortada e deverá ser tentada novamente.
- Adequado para cenários onde as **operações de leitura são mais frequentes** do que as operações de gravação e onde o bloqueio dos dados causaria muita degradação do desempenho.

FullCycle

Bloqueio Otimista

Leitura Inicial	Marcação ou Cópia dos Dados	Execução da Transação	Verificação dos Conflitos	Resolução dos Conflitos
Leitura dos dados necessários à transação.	A transação marca ou faz uma **cópia dos dados** que leu.	A transação é **executada** como se não houvesse concorrência. Nenhum bloqueio é colocado.	**Verifica se outros processos modificaram os dados lidos originalmente**. Se os dados foram alterados por outra transação desde o momento em que foram lidos pela transação atual então isso é um possível conflito.	Se um conflito for detectado, **a transação pode ser abortada e reiniciada**, ou uma estratégia de resolução de conflitos pode ser aplicada para garantir a consistência dos dados.

Bloqueio Otimista

Vantagens

- **Mitiga o risco** de impasse (**deadlock**).
- Melhor **desempenho e escalabilidade** em sistemas com **muitas transações concorrentes**.

Desvantagens

- Exige que as pessoas desenvolvedoras implementem a **detecção e resolução de conflitos nas aplicações**.
- **Aumenta a latência e a complexidade** das transações.
- **Potencialmente reduz a consistência** e a confiabilidade dos dados.

FullCycle

Aula 45
Bloqueio Pessimista

Bloqueio Pessimista

- Recursos são bloqueados exclusivamente para a transação que os está acessando.
- Assume que **conflitos e erros são comuns**.
- A maioria das **transações precisa de acesso exclusivo aos dados**.
- **Bloqueia os dados antes da leitura ou escrita** e evita que outras transações os modifiquem até que a transação atual seja **confirmada ou revertida**.
- Adequado para cenários onde as **operações de escrita são mais frequentes** do que as operações de leitura e onde **cancelar e repetir transações seria muito caro ou complexo**.

FullCycle

Bloqueio Pessimista

Requisição de Bloqueio

A transação solicita explicitamente um **bloqueio exclusivo** para o recurso. Existem diferentes **tipos de bloqueios**, como **leitura** e **escrita**.

Concessão de Bloqueio

Se o recurso não estiver bloqueado por outra transação, o sistema **concede o bloqueio** à transação solicitante. Se o recurso já estiver bloqueado, a transação solicitante é colocada em espera até que o bloqueio seja liberado.

Acesso ao Recurso

Uma vez que a transação tenha adquirido o bloqueio, ela pode acessar o recurso de forma exclusiva. N**enhum outro processo pode acessar ou modificar o recurso até que o bloqueio seja liberado**.

Liberação do Bloqueio

Após a transação ter concluído suas operações no recurso, ela **libera o bloqueio**, permitindo que outros processos acessem o recurso.

Bloqueio Pessimista

Vantagens

- Simplifica a lógica e o código das transações.
- Reduz a latência e a complexidade.
- **Favorece a consistência** e a confiabilidade dos dados.

Desvantagens

- **Cria** o risco de impasse (**deadlock**).
- **Sobrecarga** de bloqueio e desbloqueio.
- **Limita** a simultaneidade e a taxa de transferência (**throughput**).

FullCycle

Aula 46
Tratamento de Conflitos

Bloqueio Pessimista

- **Não elimina** a possibilidade de conflitos e erros.

- Transfere a **responsabilidade** de tratamento destes cenários na aplicação **para o banco de dados**.

- Cenários comuns:
  - Impasse (**deadlock**);
  - Esgotamento de tempo limite (**timeout**) e
  - Escalada de bloqueio (**lock escalation**).

Bloqueio Pessimista: Cenários Comuns

- Impasse (**deadlock**)
  - duas ou mais transações estão aguardando que a outra libere um recurso que precisa para continuar, criando uma situação em que nenhuma das transações pode progredir.

- Esgotamento de tempo limite (**timeout**)
  - Período máximo de espera permitido para que uma operação seja concluída antes que seja considerada falha ou interrompida.

Bloqueio Pessimista: Cenários Comuns

- Escalada de bloqueio (**lock escalation**)
  - O banco de dados decide **consolidar bloqueios individuais** que estão **em níveis mais baixos** (em linhas ou registros individuais) **em um nível mais alto** (como uma tabela inteira) para reduzir a sobrecarga de gerenciamento de bloqueios e melhorar a eficiência.

Bloqueio Pessimista: Tratamento de Conflitos

- Geralmente, o **banco de dados detecta e resolve** por meio do cancelamento de uma das transações e liberando seus bloqueios.

- A aplicação deve **capturar a exceção e tentar novamente** a transação cancelada.

- Para **controlar o comportamento de bloqueio** e reduzir a contenção de outras transações, a aplicação também deve:
  - **evitar bloquear muitas linhas ou colunas**;
  - **usar níveis de isolamento** e
  - usar dicas **(hints) de bloqueio** apropriados.

FullCycle

Aula 47
Níveis de Isolamento

Problemas da ausência de isolamento entre transações:

Leitura Suja
Dirty Read

Uma **transação A** atualiza um registro e **não confirma (commit)** as alterações.

O banco de dados permite que **a transação B leia** este registro **antes da confirmação** de A.

Leitura Irrepetível
Non-repeatable read

Leituras consecutivas podem recuperar **resultados diferentes** quando é permitido que outra transação faça atualizações entre estas leituras.

Leitura Fantasma
Phantom Read

Transação A faz duas leituras da mesma consulta enquanto a **transação B** insere ou exclui linhas e há a **alteração no número de linhas recuperadas** pela transação A em sua segunda leitura.

Níveis de Isolamento

Bancos de dados resolvem estes problemas com **níveis** de isolamento:

Leitura Não Confirmada **Read Uncommitted**	Leitura Confirmada **Read Committed**	Leitura Repetível **Repeatable Read**	Serializável **Serializable**
Isolamento Ausente	Alto Isolamento	Isolamento Baixo	Isolamento Mediano
Não há isolamento. É permitido às transações lerem dados não confirmados de outras transações. Não resolve nenhum dos problemas de isolamento.	**Resolve apenas a leitura suja.** Transações leem somente dados confirmados.	**Não resolve a leitura fantasma.** O recurso é bloqueado durante a transação. **Modo padrão de isolamento** em muitos bancos de dados.	**Resolve os problemas citados.** Transações concorrentes **parecem** que são executadas **em série**.

FullCycle

Aula 48
Tipos de Bloqueio

Tipos de Bloqueio

Bloqueio de Leitura
Read Lock (R)

Aplicado quando uma transação precisa **ler dados** de um recurso.

Permite que várias transações leiam os mesmos dados **simultaneamente**.

Pode **bloquear transações de escrita**, resultando em atrasos para transações de modificação.

Bloqueio de Escrita
Write Lock (W)

Aplicado quando uma transação precisa **modificar ou escrever dados** em um recurso.

Impede que outras transações leiam, atualizem ou excluam os dados enquanto o bloqueio estiver ativo.

Garante a **exclusividade durante operações de escrita**.

Pode causar bloqueios e atrasos durante muitas transações concorrentes.

FullCycle

Tipos de Bloqueio

Bloqueio Compartilhado
Shared Lock (S)

Permite que várias transações **leiam os mesmos dados simultaneamente**.

Impede que outras transações modifiquem os dados durante a vigência do bloqueio.

Pode ter bloqueios de escrita e atrasos em atualizações.

Bloqueio Exclusivo
Exclusive Lock (X)

Aplicado quando uma transação precisa de **acesso exclusivo** a um recurso para modificar ou escrever dados nele.

Impede que outras transações acessem o recurso enquanto o bloqueio estiver vigente.

Evita conflitos de **escrita**.

Bloqueio de Atualização
Update Lock (U)

É uma combinação de bloqueio de leitura e bloqueio de escrita.

Permite que uma transação **leia e atualize os dados de forma exclusiva**.

Evita leituras sujas. Similar ao Bloqueio Compartilhado, porém, com mais flexibilidade.

FullCycle

Tipos de Bloqueio

Bloqueio de Intenção Compartilhado
Intent Shared Lock (IS)

Indica a intenção de uma transação de adquirir bloqueios de **leitura em níveis mais baixos**.

Usado para coordenar bloqueios em diferentes níveis de granularidade.

Bloqueio de Intenção Exclusivo
Intent Exclusive Lock (IX)

Indica a intenção de uma transação de adquirir bloqueios de **escrita em níveis mais baixos**.

Usado para coordenar bloqueios em diferentes níveis de granularidade.

Bloqueio de Intenção Compartilhado Exclusivo
Shared Intent Exclusive Lock (SIX)

Indica a intenção de uma transação de adquirir bloqueios de **leitura e escrita em níveis mais baixos**.

Indica que outros processos podem adquirir **bloqueios de leitura em níveis inferiores**.

FullCycle

Aula 49
Níveis de Bloqueio

FullCycle

Níveis de Bloqueio

Banco de Dados

Bloqueio aplicado em **todo o banco de dados**.

Impede demais transações de acessar qualquer recurso do banco de dados durante a vigência do bloqueio.

Tabela

Bloqueio aplicado em **toda a tabela**.

Impede demais transações de acessar qualquer parte da tabela.

Simples de implementar e entender.

Página

Bloqueio aplicado em **blocos de dados** (**páginas**) que contêm **várias linhas**.

Menor granularidade em comparação com o bloqueio de linha, pois **várias linhas são bloqueadas juntas**.

Pode conter dados de múltiplos objetos.

Objeto

Bloqueio aplicado em **objetos individuais do banco de dados**, sejam estes tabelas, índices ou procedimentos armazenados.

Pode ser **mais complexo** de implementar e gerenciar.

Linha

Cada **linha** pode ser **bloqueada independentemente**.

Permite que **outras linhas** na mesma tabela sejam **acessadas e modificadas** por outras transações **simultaneamente**.

FullCycle

Níveis de Bloqueio

Banco de Dados

Mais simples.

Alta exclusividade.

Baixa concorrência.

Atrasos **prolongados**.

Tabela

Simples.

Evita problemas de **conflito**.

Baixa concorrência.

Risco de atrasos e gargalos.

Página

Reduz carga de gestão do SGBD.

Menos bloqueios mantidos e coordenados.

Risco de bloqueio em massa.

Objeto

Granularidade **flexível**.

Minimiza bloqueios.

Melhora desempenho e concorrência se **bem** implementado.

Implementação **complexa**.

Risco de atrasos.

Linha

Granularidade **fina**.

Minimiza bloqueios.

Maior concorrência.

Maior carga de gestão do SGBD em tabelas com **muitas linhas**.

FullCycle

Continuação em Banco de Dados Módulo 2