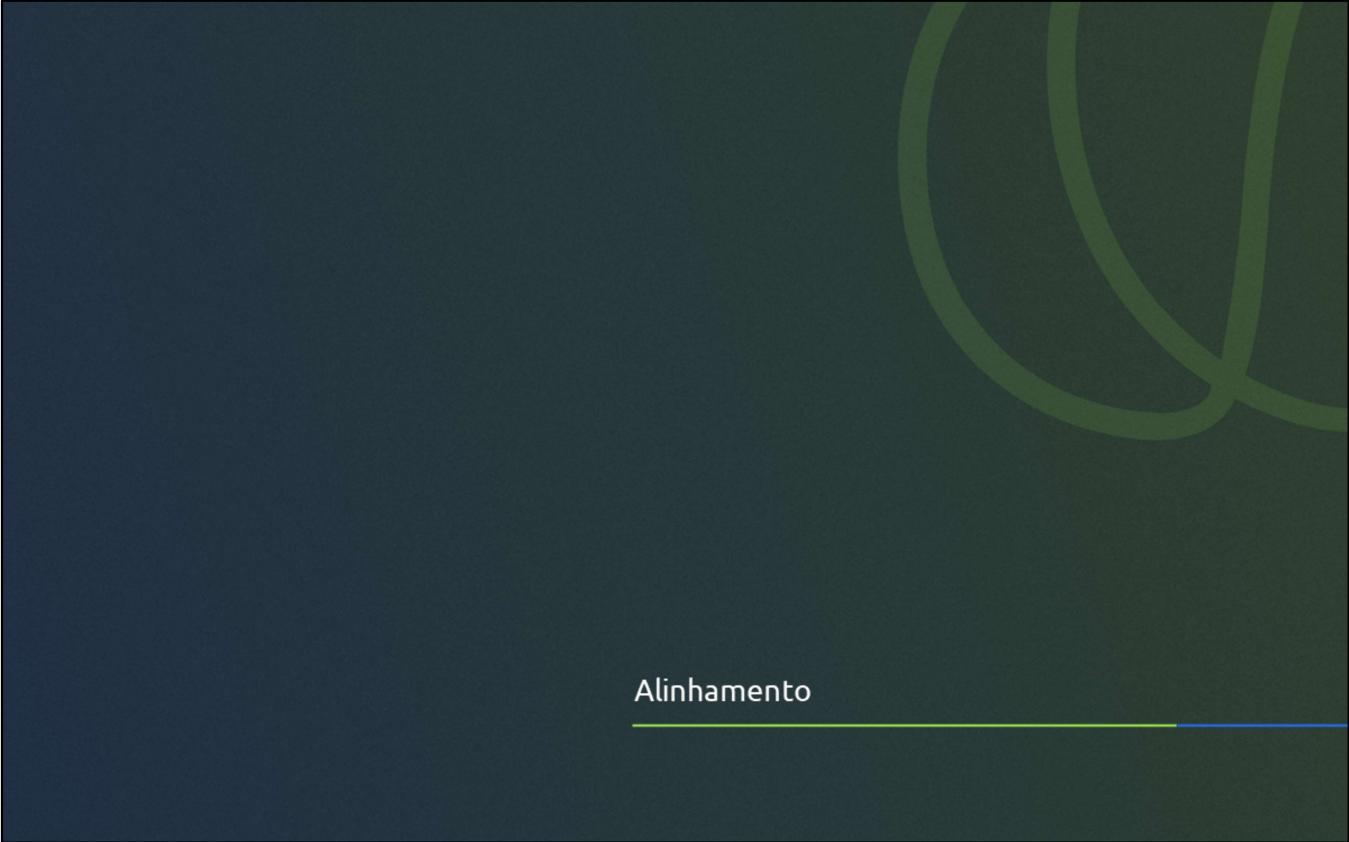




Banco de Dados I

Roberto Pontes



Alinhamento

Apresentação

Professor



Apixonado por Dados e Inovação aplicados a construir uma sociedade melhor e mais igualitária. Sou cientista de dados mestrando em Ciência da Computação com ênfase em Ciência de Dados no Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ). Pós-graduado em Big Data, bacharel em Engenharia de Controle e Automação pelo CEFET/RJ e técnico em Informática no CEFET/RJ. Atualmente sou Cientista de Dados na Radix Software e Engenharia atuando no segmento de varejo e Professor de Ciência de Dados na ADA. Anteriormente atuei como Desenvolvedor Python, com conhecimento de Django, PHP e em integração/migração de bancos de dados relacionais SQL (MySQL, PostgreSQL e MSSQL Server). Na área acadêmica tenho afinidade às áreas de processamento de imagem onde atuei durante a graduação. Atualmente estou dedicado às áreas de otimização linear e aprendizado de máquina.

 [/robertosgpontes](https://www.linkedin.com/in/robertosgpontes/)



<https://www.linkedin.com/in/robertosgpontes/>

Apresentação

Turma



NOME
FORMAÇÃO
ESTADO

1. Já trabalhou com banco de dados?
2. Qual a sua expectativa para o módulo?



O Módulo

Acordos Iniciais

1. Não teremos momento de chamada;
2. Caso te chame pelo nome e não tenha resposta por voz ou pelo chat vou entender que você não está presente;
3. A participação é sempre bem-vinda;
4. Intervalo: 15min;
5. Todos os prazos são negociáveis até a última aula do módulo (sei que imprevistos acontecem, podem ficar calmos);



1. Este tópico inclui o momento da finalização da aula, quando fechar a turma os alunos que permanecerem serão chamados nominalmente para verificar se estão presentes
2. Participação é produtivo pro andamento da turma
3. O intervalo será dado entre 20h20min ~ 20h50min a depender o andamento da aula;
4. Sou muito razoável, podem ficar tranquilos quanto a pedir para adiar, só vamos discutir juntos a viabilidade

O Módulo

O que será avaliado?

As habilidades e competências esperadas ao final do módulo:

- O que são banco de dados e SGBDs
- Instalação, configuração básica e execução de comandos em um SGBD relacional
- Modelagem de dados e modelo entidade-relacionamento (MER)
- Modelo relacional, normalização, chave primária e chave estrangeira
- Consultas SQL simples e uso de funções de formatação de colunas
- Consultas SQL complexas (agregação, junção e união)
- Comandos SQL para criação, alteração e remoção de objetos do esquema do banco
- Comandos SQL para inserção, atualização e deleção de dados de dados do banco



O Módulo

Como será avaliado?

- Participação
- Lista de Exercícios
- Trabalho Individuais/Em Grupo (Semana 1 e 2) + Projeto Final (Semana 2 e 3);
- Ou Lista/Prova



1. O projeto final e os trabalhos em grupo serão com os mesmos grupos
2. Os grupos serão montados de maneira “aleatória”
 - a. Formulário de votação de alunos com maior conhecimento técnico
3. Formaremos 6 Grupos com Quatro pessoas a partir da Semana 2

O Módulo

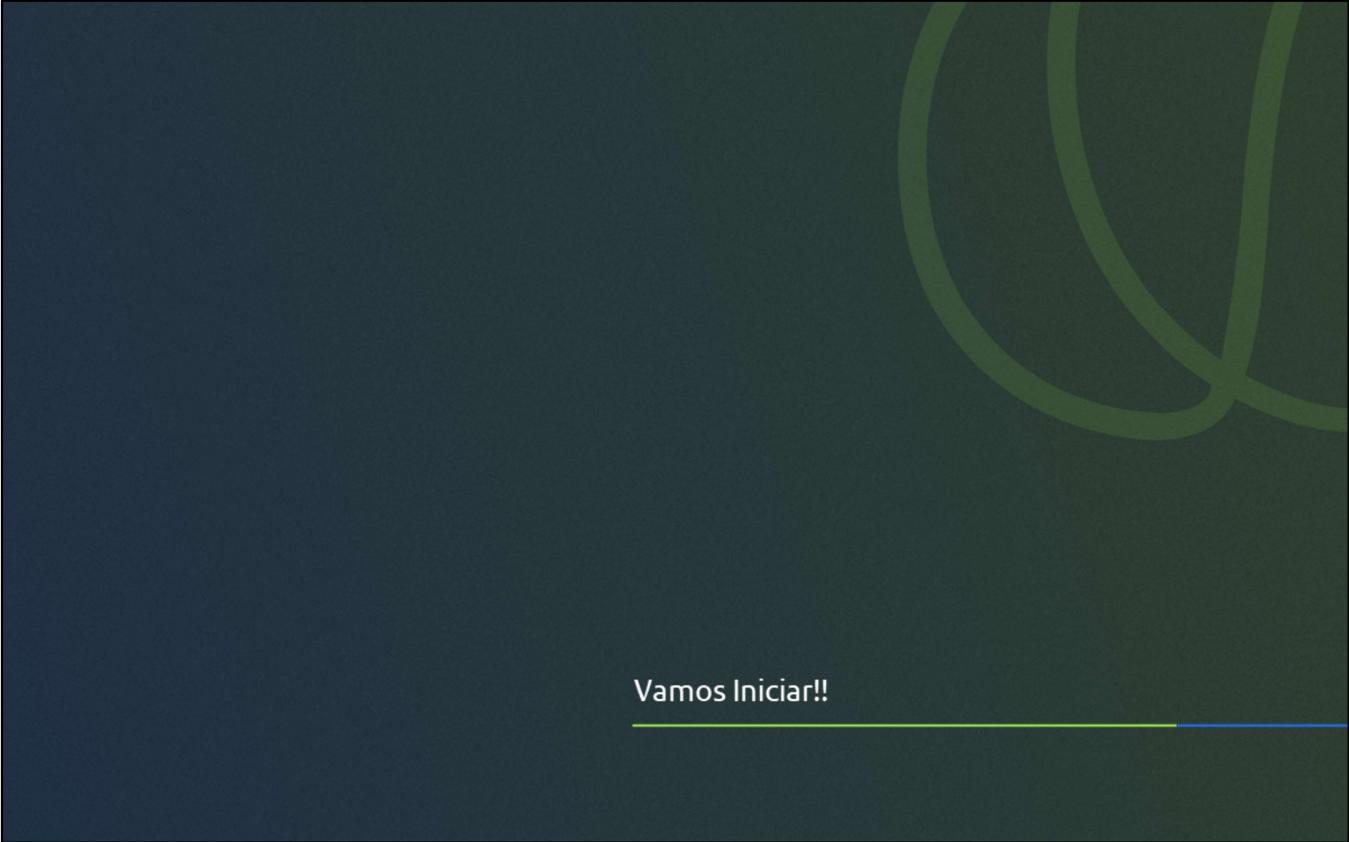
O que posso ler pra me ajudar?

ELMASRI, R.; NAVATHE, S.B. Fundamentals of Database Systems, 7th Edition.

NIELD, T. Introdução à Linguagem SQL: Abordagem Prática Para Iniciantes.

VIESCAS, J. SQL Queries for Mere Mortals: A Hands-On Guide to Data Manipulation in SQL



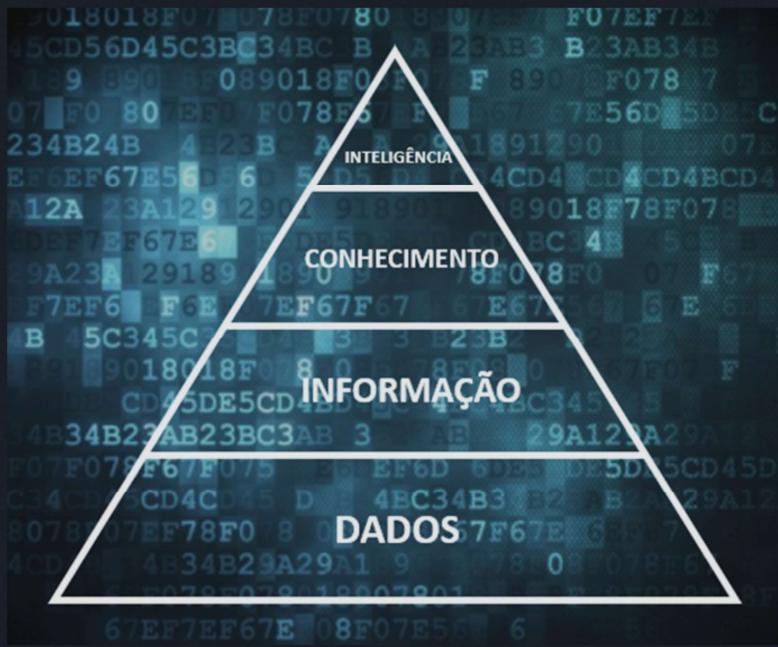


Vamos Iniciar!!

<https://www.linkedin.com/pulse/uma-introdu%C3%A7%C3%A3o-banco-de-dados-adatechbr/?trackingId=hgyMuDG5rTib%2BBkL0Tqz3w%3D%3D>

Introdução

O que são Dados



1. Dados

Apesar da palavra **dados** ser muito utilizada para se referir a **informações**, estes dois conceitos possuem diferentes significados em TI.

Dados são fatos ou observações “crus”. **Dados** são **registros, fatos brutos coletados** que **não** possuem qualquer **significado, contexto ou nexo com a realidade**.

Mais especificamente, os **dados** são **medidas objetivas e quantitativas dos atributos (características) de entidades como pessoas, lugares, coisas e eventos (conjunto de fatos)**.

Os **dados** são uma parte pequena da informação, que sozinhos não fazem sentido!

1. Informação

Quando os **dados são estruturados, organizados, processados, contextualizados ou interpretados**, há a **geração de informação**.

1. Conhecimento

Conhecimento (ou Capital Intelectual) é a habilidade de transformar a informação em ações reais. O conhecimento é uma mistura de **elementos estruturados de forma intuitiva** e, portanto, é difícil de ser colocado em palavras ou de ser plenamente entendido em termos lógicos.

1. Inteligência

A **inteligência** é o **dom humano** capaz de “digerir” as informações, por meio da **análise, e transformá-las em conhecimento útil**. Pode ser vista como **o conhecimento que foi sintetizado e aplicado a determinada situação** para ganhar maior profundidade e consciência.

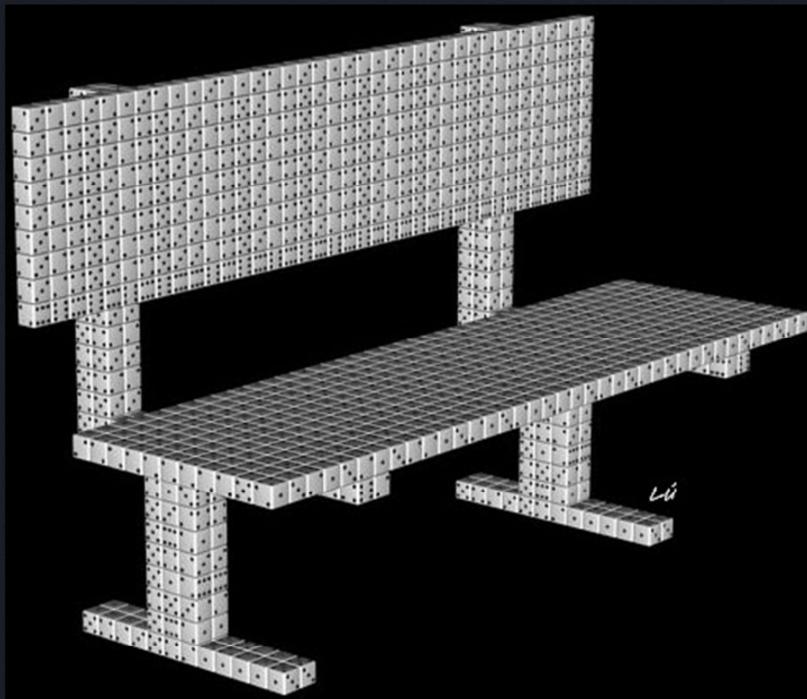
Baseia-se na experiência e intuição e, portanto, é **habilidade puramente humana**. É a **faculdade humana de conhecer, compreender, raciocinar, pensar e interpretar**. Envolve exercício de ponderação para a tomada da melhor decisão, bem como noções de ética, bom e ruim, certo e errado.

<https://www.estrategiaconcursos.com.br/blog/dado-informacao-conhecimento-inteligencia/#>

<https://www.pontodosconcursos.com.br/post/dado-informa%C3%A7%C3%A3o-conhecimento-e-intelig%C3%A3o>

Introdução

O que são Banco de Dados



A [Oracle](#) define Banco de Dados como “...uma coleção organizada de informações — ou dados — estruturadas, normalmente armazenadas eletronicamente em um sistema de computador. Um Banco de Dados é geralmente controlado por um Sistema de Gerenciamento de Banco de Dados (SGBD)”.

Introdução

Por que usar SGBDs



VAMOS PESQUISAR

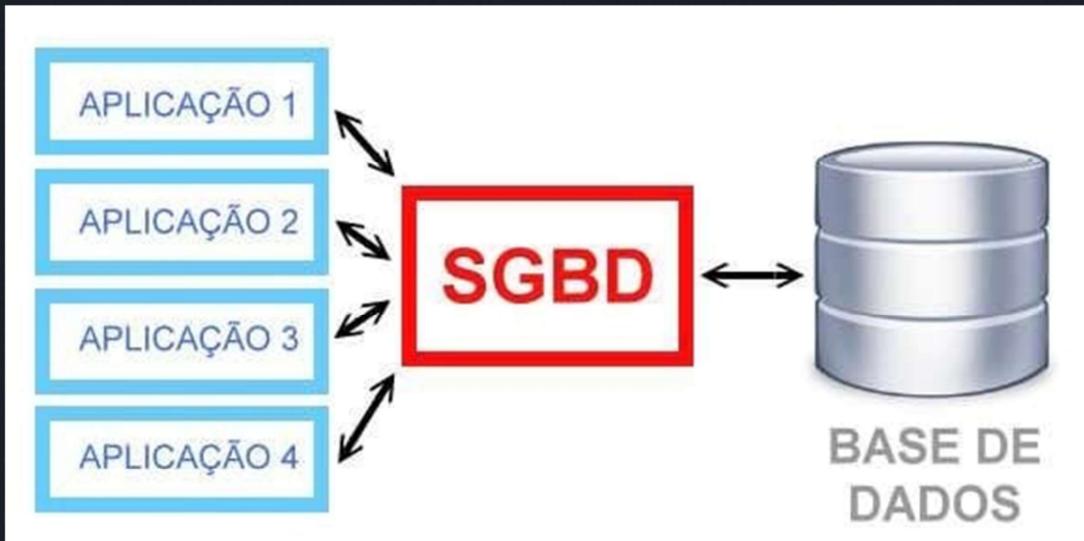
1. Como era o mundo antes dos SGBDs?
2. Como as pessoas armazenavam e recuperavam os dados?
3. Quais eram os problemas destes métodos?



Serão formados grupos de 5~6 pessoas aleatoriamente que terão 10 min para pesquisar e montar uma apresentação oral de no máximo 3 min respondendo de forma lúdica e clara.

Introdução

O que são SGBDs



Um sistema de gerenciamento de banco de dados (DBMS) é um sistema computadorizado que permite aos usuários criar e manter um banco de dados. O DBMS é um sistema de software de uso geral que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicativos. Definir um banco de dados envolve especificar os tipos de dados, estruturas e restrições dos dados a serem armazenados no banco de dados. A definição do banco de dados ou informações descritivas também são armazenadas pelo DBMS na forma de um catálogo ou dicionário de banco de dados; é chamado de metadados. Construir o banco de dados é o processo de armazenar os dados em algum meio de armazenamento controlado pelo DBMS. A manipulação de um banco de dados inclui funções como consultar o banco de dados para recuperar dados específicos, atualizar o banco de dados para refletir as mudanças no mundo real e gerar relatórios dos dados. Compartilhar um banco de dados permite que vários usuários e programas accessem o banco de dados simultaneamente.

<https://www.linkedin.com/pulse/sqbd-o-que-%C3%A9-vo%C3%A3-precisa-saber-jonatha-soares/?originalSubdomain=pt>

Introdução

O que são SGBDs



Relacionais e NoSQL

É importante notar que existem tipos de SGBDs, entre eles, os mais conhecidos e utilizados são:

- SGBDs **relacionais**, difundidos e consolidados já há algum tempo;
- E os conhecidos como **NoSQL (Not Only SQL)**. Estes últimos, cada vez mais vem conquistando espaço nas organizações.

Modelos de Base de Dados

Alguns dos modelos de base de dados são: Modelo Plano, Modelo em Rede, Modelo Hierárquico, Modelo Relacional, Orientado a objetos, e Objeto-Relacional.

Modelo plano: também conhecido como tabular, consiste de matrizes simples, bidimensionais, compostas por elementos de dados: inteiros, números reais, etc.

Modelo Relacional: os dados são classificados em tabelas, também conhecidas como relações, cada uma das quais consiste em colunas e linhas. Esse é o modelo mais comum e mais conhecido.

Modelo Orientado a objetos: define o banco de dados como uma coleção de objetos,

ou elementos de software reutilizáveis, com recursos e métodos associados.

Modelo Objeto-Relacional: este modelo de banco de dados híbrido combina a simplicidade do modelo relacional com algumas das funcionalidades avançadas do modelo de banco de dados orientado a objetos

<https://www.devmedia.com.br/gerenciamento-de-banco-de-dados-analise-comparativa-de-sgbds/30788>

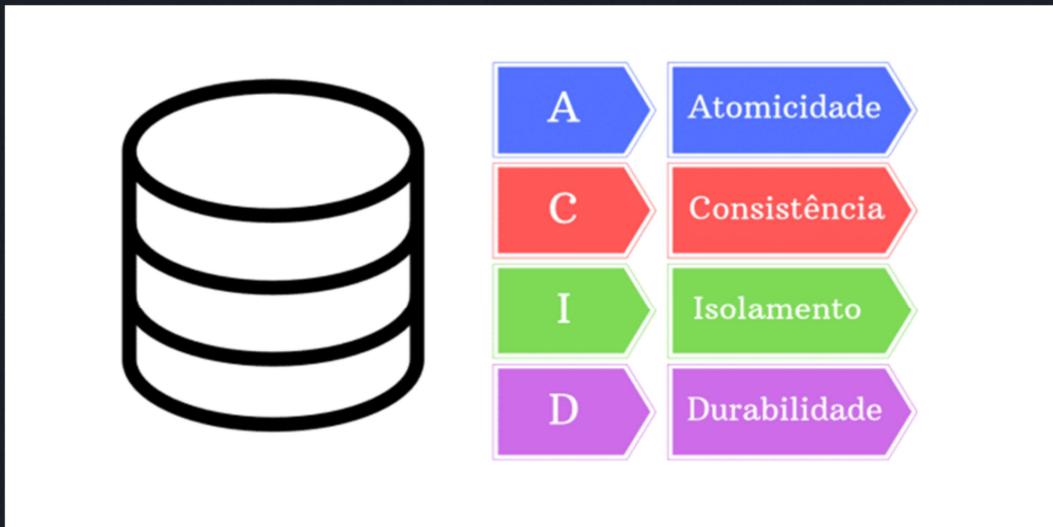
<https://www.iugu.com/iugu4devs/blog/nosql-vs-sql>

<https://becode.com.br/principais-sgbds/>

<https://dynamicssoft.com/blog/post/6-bancos-de-dados-mais-populares-usados-em-2022-para-desenvolvimento-de-aplicativos>

Introdução

O que são SGBDs - ACID



A solução para armazenamento em bancos de dados foi evoluindo durante as décadas para suprir o aumento exponencial dos sistemas de informação. As propriedades conhecidas como ACID (Atomicidade, consistência, isolamento e durabilidade) buscam garantir que essa solução possa atender as demandas cada vez mais exigentes do mercado de tecnologia.

- **Atomicidade:** garante que a transação será executada na sua totalidade ou não será executada, evitando que um processo seja feito parcialmente.
- **Consistência:** garante que os dados são confiáveis, caso alguma anomalia aconteça o estado inicial será retornado.
- **Isolamento:** as transações são independentes, ou seja, uma não pode impactar as outras.
- **Durabilidade:** garante que os dados estejam disponíveis mesmo após falhas.

Introdução

PostgreSQL



O PostgreSQL é um poderoso sistema de banco de dados objeto-relacional de código aberto que usa e estende a linguagem SQL combinada com muitos recursos que armazenam e dimensionam com segurança as cargas de trabalho de dados mais complicadas. As origens do PostgreSQL remontam a 1986 como parte do projeto POSTGRES da Universidade da Califórnia em Berkeley e tem mais de 35 anos de desenvolvimento ativo na plataforma principal.

O PostgreSQL conquistou uma forte reputação por sua arquitetura comprovada, confiabilidade, integridade de dados, conjunto robusto de recursos, extensibilidade e dedicação da comunidade de código aberto por trás do software para oferecer consistentemente soluções inovadoras e de alto desempenho. O PostgreSQL é executado em todos os principais sistemas operacionais, é compatível com ACID desde 2001 e possui complementos poderosos, como o popular extensor de banco de dados geoespacial PostGIS.

<https://www.postgresql.org/>

<https://www.postgresql.org/download/>

<https://www.pgadmin.org/download/>

<https://dbeaver.io/download/>

Introdução

PostgreSQL



<https://github.com/robertosgpontes/BancoDeDados>

Trabalho 1

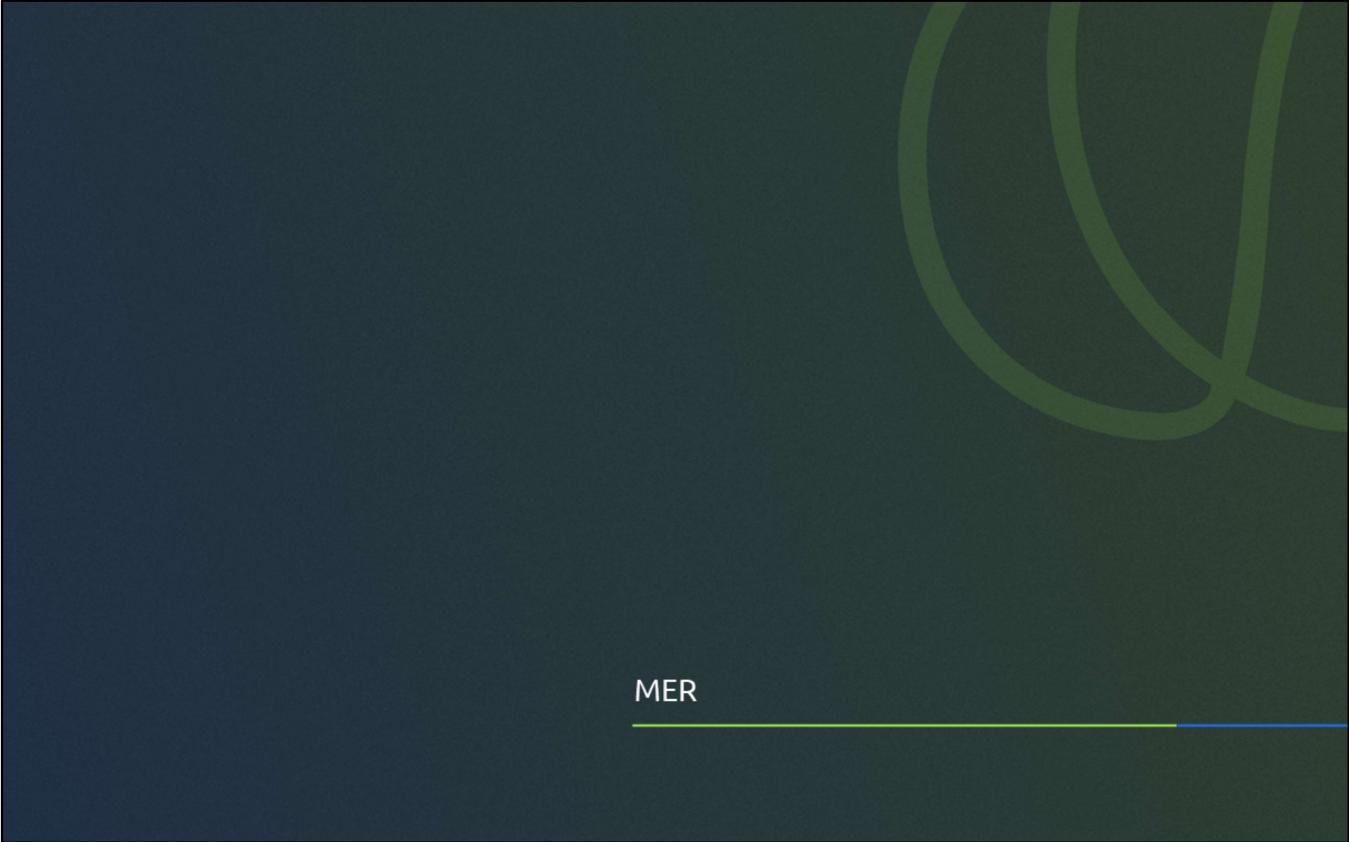
PostgreSQL

Instanciamento de um banco de dados PostgreSQL usando Docker. E acesso do banco de dados via pgAdmin ou DBeaver

- Como entregável da tarefa teremos um tutorial como o passo a passo desde a instalação do Docker até o acesso no pgAdmin ou DBeaver com:
 - a. Links
 - b. Capturas de Tela
 - c. Comando
- Prazo: Até segunda-feira 19/12 às 19h.



<https://docs.docker.com/desktop/install/windows-install/>



MER

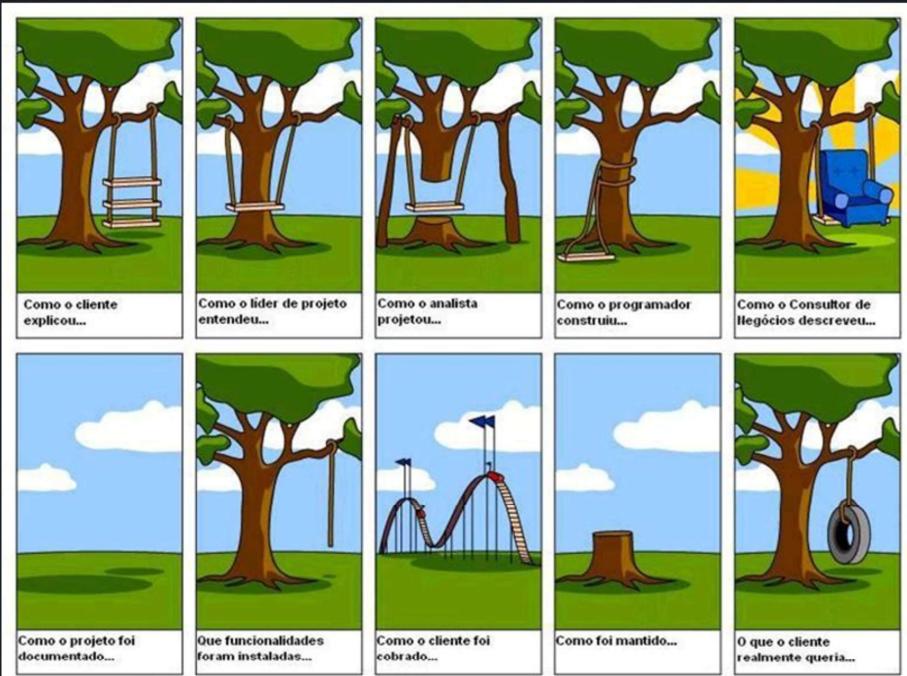
<https://www.linkedin.com/pulse/uma-introdu%C3%A7%C3%A3o-banco-de-dados-adatechbr/?trackingId=hgyMuDG5rTib%2BBkL0Tqz3w%3D%3D>

<https://sites.google.com/site/uniplibancodedados1/home>

<https://docente.ifrn.edu.br/nickersonferreira/disciplinas/programacao-com-acesso-a-banco-de-dados-3o-ano/aula-02-modelo-entidade-relacionamento-e-relacional/view>

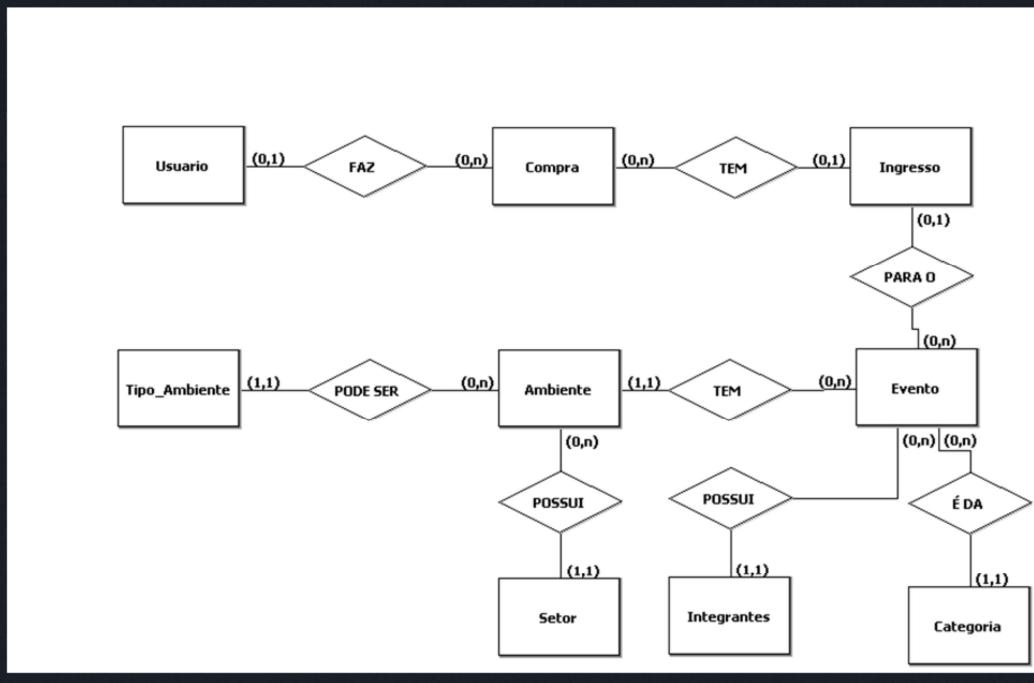
Introdução

Por que modelar?



Introdução

O que é MER?



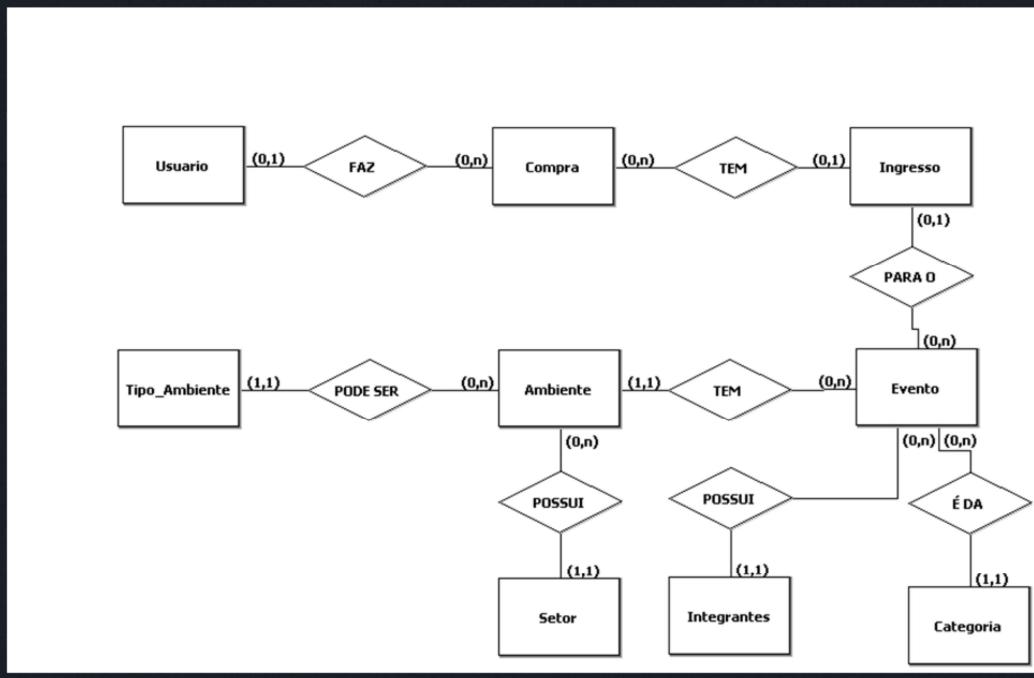
O Modelo Entidade Relacionamento (também chamado Modelo ER, ou simplesmente MER), como o nome sugere, é um modelo conceitual utilizado na Engenharia de Software para descrever os objetos (entidades) envolvidos em um domínio de negócios, com suas características (atributos) e como elas se relacionam entre si (relacionamentos).

Em geral, este modelo representa de forma abstrata a estrutura que possuirá o banco de dados da aplicação. Obviamente, o banco de dados poderá conter várias outras entidades, tais como chaves e tabelas intermediárias, que podem só fazer sentido no contexto de bases de dados relacionais.

<https://www.devmedia.com.br/mer-e-der-modelagem-de-bancos-de-dados/14332>

Introdução

O que é MER?



Entidades

Os objetos ou partes envolvidas um domínio, também chamados de entidades, podem ser classificados como físicos ou lógicos, de acordo sua existência no mundo real.

Entidades físicas: são aquelas realmente tangíveis, existentes e visíveis no mundo real, como um cliente (uma pessoa, uma empresa) ou um produto (um carro, um computador, uma roupa). Já as entidades lógicas são aquelas que existem geralmente em decorrência da interação entre ou com entidades físicas, que fazem sentido dentro de um certo domínio de negócios, mas que no mundo externo/real não são objetos físicos (que ocupam lugar no espaço). São exemplos disso uma venda ou uma classificação de um objeto (modelo, espécie, função de um usuário do sistema).

As entidades são nomeadas com substantivos concretos ou abstratos que representem de forma clara sua função dentro do domínio. Exemplos práticos de entidades comuns em vários sistemas são Cliente, Produto, Venda, Turma, Função, entre outros.

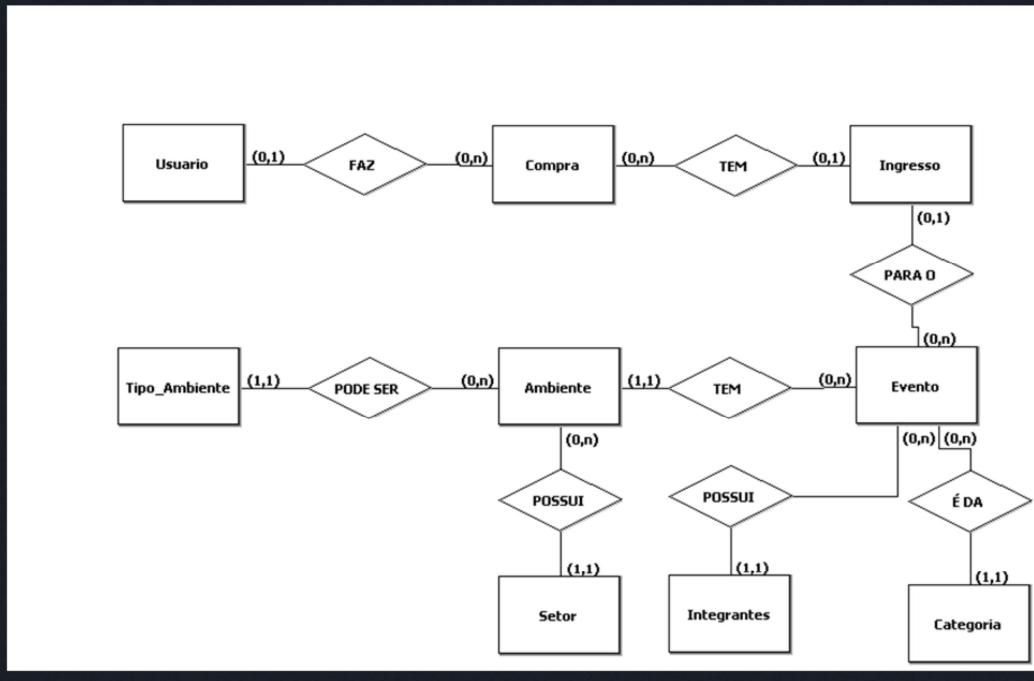
Podemos classificar as entidades segundo o motivo de sua existência:

- **Entidades fortes:** são aquelas cuja existência independe de outras entidades, ou seja, por si só elas já possuem total sentido de existir. Em um sistema de vendas, a entidade produto, por exemplo, independe de quaisquer outras para existir.

- **Entidades fracas:** ao contrário das entidades fortes, as fracas são aquelas que dependem de outras entidades para existirem, pois individualmente elas não fazem sentido. Mantendo o mesmo exemplo, a entidade venda depende da entidade produto, pois uma venda sem itens não tem sentido.
- **Entidades associativas:** esse tipo de entidade surge quando há a necessidade de associar uma entidade a um relacionamento existente. Na modelagem Entidade-Relacionamento não é possível que um relacionamento seja associado a uma entidade, então tornamos esse relacionamento uma entidade associativa, que a partir daí poderá se relacionar com outras entidades. Para melhor compreender esse conceito, tomemos como exemplo uma aplicação de vendas em que existem as entidades Produto e Venda, que se relacionam na forma muitos-para-muitos, uma vez que em uma venda pode haver vários produtos e um produto pode ser vendido várias vezes (no caso, unidades diferentes do mesmo produto). Em determinado momento, a empresa passou a entregar brindes para os clientes que comprassem um determinado produto. A entidade Brinde, então, está relacionada não apenas com a Venda, nem com o Produto, mas sim com o item da venda, ou seja, com o relacionamento entre as duas entidades citadas anteriormente. Como não podemos associar a entidade Brinde com um relacionamento, criamos então a entidade associativa "Item da Venda", que contém os atributos identificadores das entidades Venda e Produto, além de informações como quantidade e número de série, para casos específicos. A partir daí, podemos relacionar o Brinde com o Item da Venda, indicando que aquele prêmio foi dado ao cliente por comprar aquele produto especificamente.

Introdução

O que é MER?



Relacionamentos

Uma vez que as entidades são identificadas, deve-se então definir como se dá o relacionamento entre elas. De acordo com a quantidade de objetos envolvidos em cada lado do relacionamento, podemos classificá-los de três formas:

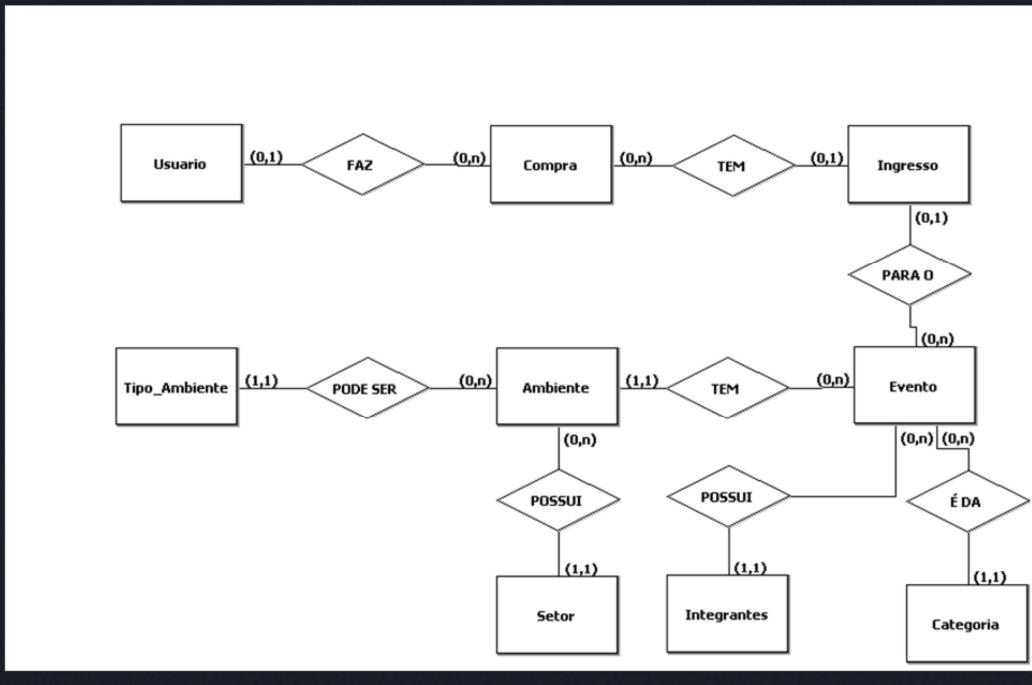
- **Relacionamento 1..1 (um para um)**: cada uma das duas entidades envolvidas referenciam obrigatoriamente apenas uma unidade da outra. Por exemplo, em um banco de dados de currículos, cada usuário cadastrado pode possuir apenas um currículo na base, ao mesmo tempo em que cada currículo só pertence a um único usuário cadastrado.
- **Relacionamento 1..n ou 1..*** (**um para muitos**): uma das entidades envolvidas pode referenciar várias unidades da outra, porém, do outro lado cada uma das várias unidades referenciadas só pode estar ligada uma unidade da outra entidade. Por exemplo, em um sistema de plano de saúde, um usuário pode ter vários dependentes, mas cada dependente só pode estar ligado a um usuário principal. Note que temos apenas duas entidades envolvidas: usuário e dependente. O que muda é a quantidade de unidades/exemplares envolvidas de cada lado.
- **Relacionamento n..n ou *..*** (**muitos para muitos**): neste tipo de relacionamento cada entidade, de ambos os lados, podem referenciar múltiplas

unidades da outra. Por exemplo, em um sistema de biblioteca, um título pode ser escrito por vários autores, ao mesmo tempo em que um autor pode escrever vários títulos. Assim, um objeto do tipo autor pode referenciar múltiplos objetos do tipo título, e vice-versa.

Os relacionamentos em geral são nomeados com verbos ou expressões que representam a forma como as entidades interagem, ou a ação que uma exerce sobre a outra. Essa nomenclatura pode variar de acordo com a direção em que se lê o relacionamento. Por exemplo: um autor escreve vários livros, enquanto um livro é escrito por vários autores.

Introdução

O que é MER?



Atributos

Atributos são as características que descrevem cada entidade dentro do domínio. Por exemplo, um cliente possui nome, endereço e telefone. Durante a análise de requisitos, são identificados os atributos relevantes de cada entidade naquele contexto, de forma a manter o modelo o mais simples possível e consequentemente armazenar apenas as informações que serão úteis futuramente. Uma pessoa possui atributos pessoais como cor dos olhos, altura e peso, mas para um sistema que funcionará em um supermercado, por exemplo, essas informações dificilmente serão relevantes.

Os atributos podem ser classificados quanto à sua função da seguinte forma:

- **Descritivos:** representam características intrínsecas de uma entidade, tais como nome ou cor.
- **Nominativos:** além de serem também descritivos, estes têm a função de definir e identificar um objeto. Nome, código, número são exemplos de atributos nominativos.
- **Referenciais:** representam a ligação de uma entidade com outra em um relacionamento. Por exemplo, uma venda possui o CPF do cliente, que a relaciona com a entidade cliente.

Quanto à sua estrutura, podemos ainda classificá-los como:

- **Simples:** um único atributo define uma característica da entidade. Exemplos: nome, peso.
- **Compostos:** Para definir uma informação da entidade, são usados vários atributos. Por exemplo, o endereço pode ser composto por rua, número, bairro, etc.

Alguns atributos representam valores únicos que identificam a entidade dentro do domínio e não podem se repetir. Em um cadastro de clientes, por exemplo, esse atributo poderia ser o CPF. A estes chamamos de Chave Primária.

Introdução

Notação de Peter Chen



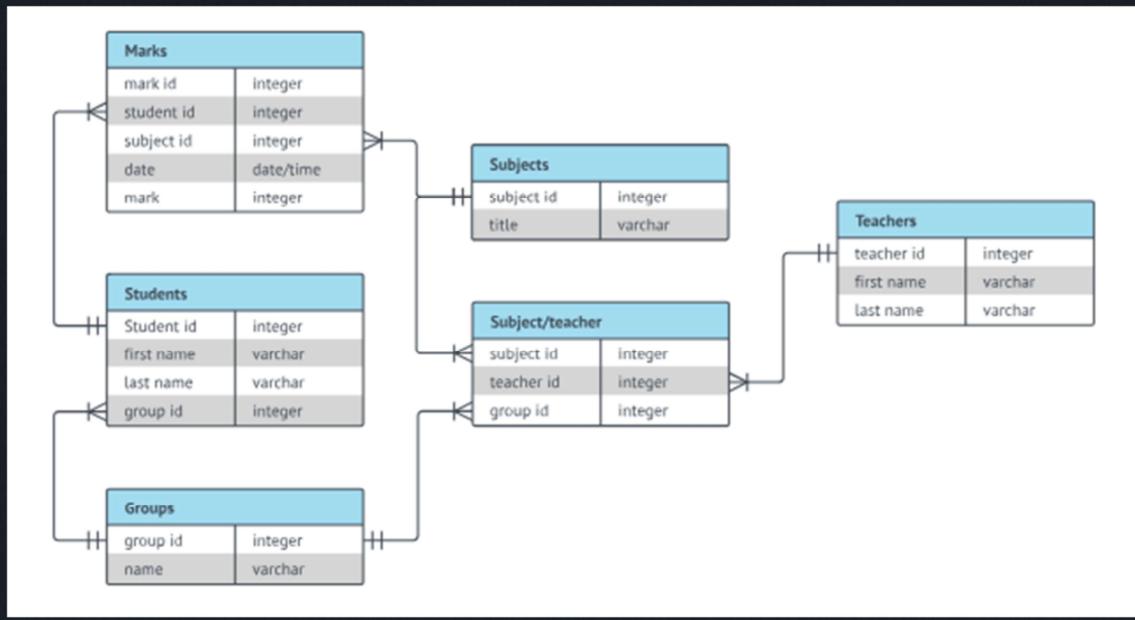
Notações de Peter Chen

<http://docente.ifrn.edu.br/elieziosoares/disciplinas/programacao-com-acesso-a-banco-de-dados/7-notacoes-e-r>

<https://www.sis4.com/brModelo/>

Introdução

Notação de James Martin



Notação de James Martin (também conhecida como a notação pé de galinha)

<http://docente.ifrn.edu.br/elieziosoares/disciplinas/programacao-com-acesso-a-banco-de-dados/7-notacoes-e-r>

<https://prezi.com/vs8tx3hcwawi/banco-de-dados-notacao-de-james-martin/>

<https://app.tryeraser.com/>

Introdução

PostgreSQL



Fábrica de Roupas

Uma fábrica de roupas exclusivas (cada modelo, único, é projetado por estilistas famosos) deseja um sistema para controlar sua produção. A fábrica conta atualmente com 1230 funcionários sendo que a maior parte dos mesmos são costureiras trabalhando na atividade fim. A fábrica possui aproximadamente 600 máquinas de costura de diversos tipos (overlock, zig zag, costura reta, etc) de diversos fabricantes.

Para ingressar como costureira, a funcionária é avaliada para determinar em que tipo de máquina ela possui habilitação. Cada máquina pode realizar um ou mais tipos de costura.

Cada peça de roupa é produzida integralmente por uma costureira em uma máquina, sendo que neste período nem a costureira, nem a máquina podem ser alocados para outra coisa.

A remuneração das costureiras é mensal, baseado em uma alíquota fixa (15%) sobre o preço de venda de cada peça. Nenhuma costureira pode receber menos que um determinado valor mínimo que é negociado no momento da contratação de cada uma. As costureiras são divididas em supervisões, cada uma possuindo uma supervisora que é a responsável pela

qualidade do que é produzido, e pela monitoração das máquinas que estão em conserto. A máquina só vai para conserto após o término da produção da peça. A supervisora da costureira que estava produzindo nesta máquina se torna a responsável pela monitoração de seu conserto.

A fábrica necessita das seguintes informações:

- a) Relatório de peças produzidas por uma costureira num determinado período, no seguinte formato: modelo da peça, descrição do modelo, data e hora de início e término da fabricação, código da máquina de costura, localização da máquina e o fabricante.
- b) Relatório das costureiras sem produção no período (matrícula da costureira, nome, Valor Mínimo Negociado).
- c) Relação das máquinas disponíveis, informando para cada uma o seu fabricante e o(s) tipo(s) de costura que possui.
- d) Quais costureiras estão disponíveis e habilitadas a trabalhar em um tipo de máquina no momento.
- e) Relação das máquinas que estiveram mais de 10 vezes em conserto, contendo: código da máquina e para cada conserto, matrícula e nome da supervisora responsável, data início e término do conserto.

Introdução

PostgreSQL



1. Um berçário deseja informatizar suas operações. Quando um bebê nasce, algumas informações são armazenadas sobre ele, tais como: nome, data do nascimento, peso do nascimento, altura, a mãe deste bebê e o médico que fez seu parto. Para as mães, o berçário também deseja manter um controle, guardando informações como: nome, endereço, telefone e data de nascimento. Para os médicos, é importante saber: CRM, nome, telefone celular e especialidade.
2. Uma floricultura deseja informatizar suas operações. Inicialmente, deseja manter um cadastro de todos os seus clientes, mantendo informações como: RG, nome, telefone e endereço. Deseja também manter um cadastro contendo informações sobre os produtos que vende, tais como: nome do produto, tipo (flor, vaso, planta,...), preço e quantidade em estoque. Quando um cliente faz uma compra, a mesma é armazenada, mantendo informação sobre o cliente que fez a compra, a data da compra, o valor total e os produtos comprados.
3. Uma escola tem várias turmas. Uma turma tem vários professores, sendo que um professor pode ministrar aulas em mais de uma turma. Uma turma tem sempre aulas na mesma sala, mas uma sala pode estar associada a várias turmas (com horários diferentes).
4. Uma biblioteca deseja manter informações sobre seus livros. Inicialmente, quer armazenar para os livros as seguintes características: ISBN, título, ano editora e autores deste livro. Para os autores, deseja-se manter: nome e

nacionalidade. Cabe salientar que um autor pode ter vários livros, assim como um livro pode ser escrito por vários autores. Cada livro da biblioteca pertence a uma categoria. A biblioteca deseja manter um cadastro de todas as categorias existentes, com informações como: código da categoria e descrição. Uma categoria pode ter vários livros associados a ela.

5. Uma firma vende produtos de limpeza, e deseja melhor controlar os produtos que vende, seus clientes e os pedidos. Cada produto é caracterizado por um código, nome do produto, categoria (ex. detergente, sabão em pó, sabonete, etc), e seu preço. A categoria é uma classificação criada pela própria firma. A firma possui informações sobre todos seus clientes. Cada cliente é identificado por um código, nome, endereço, telefone, status ("bom", "médio", "ruim"), e o seu limite de crédito. Guarda-se igualmente a informação dos pedidos feitos pelos clientes. Cada pedido possui um número e guarda-se a data de elaboração do pedido. Cada pedido pode envolver de um a vários produtos, e para cada produto, indica-se a quantidade deste pedida.

FNs e Modelo Físico

<https://www.linkedin.com/pulse/uma-introdu%C3%A7%C3%A3o-banco-de-dados-adatechbr/?trackingId=hgyMuDG5rTib%2BBkL0Tqz3w%3D%3D>

<https://www.luis.blog.br/normalizacao-de-dados-e-as-formas-normais.html>

<https://spaceprogrammer.com/bd/normalizando-um-banco-de-dados-por-meio-das-3-principais-formas/>

<https://www.estrategiaconcursos.com.br/blog/banco-dados-forma-normal/#>

<https://www.devmedia.com.br/guia-simplificado-para-as-5-formas-normais-artigo-revista-sql-magazine-87/21043>

Introdução

Por que Normalizar?

A regra de ouro que devemos observar no projeto de um banco de dados baseado no Modelo Relacional de Dados é a de "não misturar assuntos em uma mesma Tabela".

Os objetivos da normalização são muitos, entre eles destaco:

- Minimização de redundâncias e inconsistências;
- Facilidade de manipulação do banco de dados;
- Ganho de performance no SGBD;
- Facilidade de manutenção do sistema de Informação;
- Entre outros.



Introdução

1FN

A orientação da 1^a. forma normal é que todos os atributos do modelo sejam atômicos, indivisíveis. Em outras palavras, não são permitidos valores duplicados, tampouco campos possuindo mais de um valor simultaneamente.



1FN - 1^a Forma Normal: todos os atributos de uma tabela devem ser atômicos, ou seja, a tabela não deve conter grupos repetidos e nem atributos com mais de um valor. Para deixar nesta forma normal, é preciso identificar a chave primária da tabela, identificar a(s) coluna(s) que tem(êm) dados repetidos e removê-la(s), criar uma nova tabela com a chave primária para armazenar o dado repetido e, por fim, criar uma relação entre a tabela principal e a tabela secundária. Por exemplo, considere a tabela Pessoas a seguir.

PESSOAS = {ID + NOME + ENDERECO + TELEFONES}

Ela contém a chave primária ID e o atributo TELEFONES é um atributo multivalorado e, portanto, a tabela não está na 1FN. Para deixá-la na 1FN, vamos criar uma nova tabela chamada TELEFONES que conterá PESSOA_ID como chave estrangeira de PESSOAS e TELEFONE como o valor multivalorado que será armazenado.

PESSOAS = { ID + NOME + ENDERECO }

TELEFONES = { PESSOA_ID + TELEFONE }

<https://www.luis.blog.br/primeira-forma-normal-1fn-normalizacao-de-dados.html>

Introdução

2FN

A 2^a. forma normal indica que os atributos que não são chave dependem unicamente da chave primária da tabela. Em outras palavras, não pode haver dependência parcial.



2FN - 2^a Forma Normal: antes de mais nada, para estar na 2FN é preciso estar na 1FN. Além disso, todos os atributos não chaves da tabela devem depender unicamente da chave primária (não podendo depender apenas de parte dela). Para deixar na segunda forma normal, é preciso identificar as colunas que não são funcionalmente dependentes da chave primária da tabela e, em seguida, remover essa coluna da tabela principal e criar uma nova tabela com esses dados. Por exemplo, considere a tabela ALUNOS_CURSOS a seguir.

ALUNOS_CURSOS = { ID_ALUNO + ID_CURSO + NOTA + DESCRICAO_CURSO }

Nessa tabela, o atributo DESCRICAO_CURSO depende apenas da chave primária ID_CURSO. Dessa forma, a tabela não está na 2FN. Para tanto, cria-se uma nova tabela chamada CURSOS que tem como chave primária ID_CURSO e atributo DESCRICAO retirando, assim, o atributo DESCRICAO_CURSO da tabela ALUNOS_CURSOS.

ALUNOS_CURSOS = {ID_ALUNO + ID_CURSO + NOTA}

CURSOS = {ID_CURSO + DESCRICAO}

<https://www.luis.blog.br/segunda-forma-normal-2fn-normalizacao-de-dados.html>

Introdução

3FN

A 3^{a.} forma normal prega que os atributos que não são chave devem ser independentes entre si e dependentes única e exclusivamente da chave primária da tabela. Não é permitido dependência transitiva.



A terceira forma normal é parte daquilo que chamamos de normalização de dados para fins de planejamento de bases de dados computacionais. É uma forma de analisar e refinar a estrutura dos dados a fim de torná-los íntegros e exclusivos, evitando repetições desnecessárias e possíveis sobrecargas no gerenciador de banco de dados.

Uma tabela está na **Terceira Forma Normal** 3FN se ela estiver na 2FN e se nenhuma coluna não-chave depender de outra coluna não-chave.

Na **terceira forma normal** temos de eliminar aqueles campos que podem ser obtidos pela equação de outros campos da mesma tabela.

Procedimentos:

- a) Identificar todos os atributos que são funcionalmente dependentes de outros atributos não chave;
- b) Removê-los.

A chave primária da nova entidade será o atributo do qual os atributos removidos são funcionalmente dependentes.

<https://www.luis.blog.br/terceira-forma-normal-3fn-normalizacao-de-dados.html>

Introdução



1. Um berçário deseja informatizar suas operações. Quando um bebê nasce, algumas informações são armazenadas sobre ele, tais como: nome, data do nascimento, peso do nascimento, altura, a mãe deste bebê e o médico que fez seu parto. Para as mães, o berçário também deseja manter um controle, guardando informações como: nome, endereço, telefone e data de nascimento. Para os médicos, é importante saber: CRM, nome, telefone celular e especialidade.
2. Uma floricultura deseja informatizar suas operações. Inicialmente, deseja manter um cadastro de todos os seus clientes, mantendo informações como: RG, nome, telefone e endereço. Deseja também manter um cadastro contendo informações sobre os produtos que vende, tais como: nome do produto, tipo (flor, vaso, planta,...), preço e quantidade em estoque. Quando um cliente faz uma compra, a mesma é armazenada, mantendo informação sobre o cliente que fez a compra, a data da compra, o valor total e os produtos comprados.
3. Uma escola tem várias turmas. Uma turma tem vários professores, sendo que um professor pode ministrar aulas em mais de uma turma. Uma turma tem sempre aulas na mesma sala, mas uma sala pode estar associada a várias turmas (com horários diferentes).
4. Uma biblioteca deseja manter informações sobre seus livros. Inicialmente, quer armazenar para os livros as seguintes características: ISBN, título, ano editora e autores deste livro. Para os autores, deseja-se manter: nome e

nacionalidade. Cabe salientar que um autor pode ter vários livros, assim como um livro pode ser escrito por vários autores. Cada livro da biblioteca pertence a uma categoria. A biblioteca deseja manter um cadastro de todas as categorias existentes, com informações como: código da categoria e descrição. Uma categoria pode ter vários livros associados a ela.

5. Uma firma vende produtos de limpeza, e deseja melhor controlar os produtos que vende, seus clientes e os pedidos. Cada produto é caracterizado por um código, nome do produto, categoria (ex. detergente, sabão em pó, sabonete, etc), e seu preço. A categoria é uma classificação criada pela própria firma. A firma possui informações sobre todos seus clientes. Cada cliente é identificado por um código, nome, endereço, telefone, status ("bom", "médio", "ruim"), e o seu limite de crédito. Guarda-se igualmente a informação dos pedidos feitos pelos clientes. Cada pedido possui um número e guarda-se a data de elaboração do pedido. Cada pedido pode envolver de um a vários produtos, e para cada produto, indica-se a quantidade deste pedida.

Introdução



Fábrica de Roupas

Uma fábrica de roupas exclusivas (cada modelo, único, é projetado por estilistas famosos) deseja um sistema para controlar sua produção. A fábrica conta atualmente com 1230 funcionários sendo que a maior parte dos mesmos são costureiras trabalhando na atividade fim. A fábrica possui aproximadamente 600 máquinas de costura de diversos tipos (overlock, zig zag, costura reta, etc) de diversos fabricantes.

Para ingressar como costureira, a funcionária é avaliada para determinar em que tipo de máquina ela possui habilitação. Cada máquina pode realizar um ou mais tipos de costura.

Cada peça de roupa é produzida integralmente por uma costureira em uma máquina, sendo que neste período nem a costureira, nem a máquina podem ser alocados para outra coisa.

A remuneração das costureiras é mensal, baseado em uma alíquota fixa (15%) sobre o preço de venda de cada peça. Nenhuma costureira pode receber menos que um determinado valor mínimo que é negociado no momento da contratação de cada uma. As costureiras são divididas em supervisões, cada uma possuindo uma supervisora que é a responsável pela

qualidade do que é produzido, e pela monitoração das máquinas que estão em conserto. A máquina só vai para conserto após o término da produção da peça. A supervisora da costureira que estava produzindo nesta máquina se torna a responsável pela monitoração de seu conserto.

A fábrica necessita das seguintes informações:

- a) Relatório de peças produzidas por uma costureira num determinado período, no seguinte formato: modelo da peça, descrição do modelo, data e hora de início e término da fabricação, código da máquina de costura, localização da máquina e o fabricante.
- b) Relatório das costureiras sem produção no período (matrícula da costureira, nome, Valor Mínimo Negociado).
- c) Relação das máquinas disponíveis, informando para cada uma o seu fabricante e o(s) tipo(s) de costura que possui.
- d) Quais costureiras estão disponíveis e habilitadas a trabalhar em um tipo de máquina no momento.
- e) Relação das máquinas que estiveram mais de 10 vezes em conserto, contendo: código da máquina e para cada conserto, matrícula e nome da supervisora responsável, data início e término do conserto.

DDL - *Data Definition Language*

DDL

```
CREATE DATABASE name;
```



[PostgreSQL: Documentation: 15: CREATE DATABASE](#)

<https://www.postgresql.org/docs/current/sql-createdatabase.html>

<https://www.postgresqltutorial.com/postgresql-administration/postgresql-create-database/>

DDL

```
CREATE TABLE [ IF NOT EXISTS ] table_name ( [  
    { column_name data_type [ column_constraint [ ... ] ]  
    | table_constraint  
} ] );
```



[PostgreSQL: Documentation: 15: CREATE TABLE](#)

<https://www.postgresql.org/docs/current/sql-createtable.html>

<https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-create-table/>

DDL

```
CREATE TABLE [ IF NOT EXISTS ] table_name  
[ (column_name [, ...] ) ]  
AS query
```



[PostgreSQL: Documentation: 15: CREATE TABLE AS](#)

<https://www.postgresql.org/docs/current/sql-createtableas.html>

DDL

```
CREATE TABLE [ IF NOT EXISTS ] table_name  
[ (column_name [, ...] ) ]  
AS query
```



[PostgreSQL: Documentation: 15: CREATE TABLE AS](#)

<https://www.postgresql.org/docs/current/sql-createtableas.html>

<https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-create-table-as/>

DDL

CONSTRAINTS

- CHECK CONSTRAINTS
- NOT-NULL CONSTRAINTS
- UNIQUE CONSTRAINTS
- PRIMARY KEYS
- FOREIGN KEYS
- EXCLUSION CONSTRAINTS



[PostgreSQL: Documentation: 15: 5.4. Constraints](#)

<https://www.postgresql.org/docs/15/ddl-constraints.html>

DDL

```
CREATE INDEX index_name ON table_name [USING method]
(
    column_name [ASC | DESC] [NULLS {FIRST | LAST }],
    ...
);
```



[PostgreSQL: Documentation: 15: CREATE INDEX](#)

<https://www.postgresql.org/docs/15/ddl-constraints.html> <https://www.postgresql.org/docs/15/sql-createindex.html>

<https://www.postgresqltutorial.com/postgresql-indexes/postgresql-create-index/>

DDL

- ALTER TABLE

- DROP TABLE

```
DROP TABLE [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ]
```



- ALTER TABLE

<https://www.postgresql.org/docs/current/sql-altertable.html>

- DROP TABLE

<https://www.postgresql.org/docs/15/sql-droppable.html>

DDL

CREATE VIEW vs CREATE MATERIALIZED VIEW

VIEW	MATERIALIZED VIEW
Database object that allows generating a logical subset of data from one or more tables	Logical view of data driven by the select query in which the result of the query stores in the disk
Not stored in the disk	Stored in the disk
Slower	Faster
It is necessary to update the view each time using it	It is not necessary to update the materialized view each time using it



**MATERIALIZED VIEW vs
VIEW in PostgreSQL | by
Nidhi Gupta | Analytics
Vidhya | Medium**
(<https://medium.com/analytics-vidhya/materialized-view-vs-view-in->

postgresql-1ddb4fe86cb7)

What is the
Difference
Between View
and Materialized

View -

Pediaa.Com

(<https://pediaa.com/what-is-the-difference-between-view-and-materialized-view/>

view/)

CREATE VIEW

(<https://www.postgresql.org/docs/15/sql-createview.html>)

CREATE MATERIALIZED VIEW

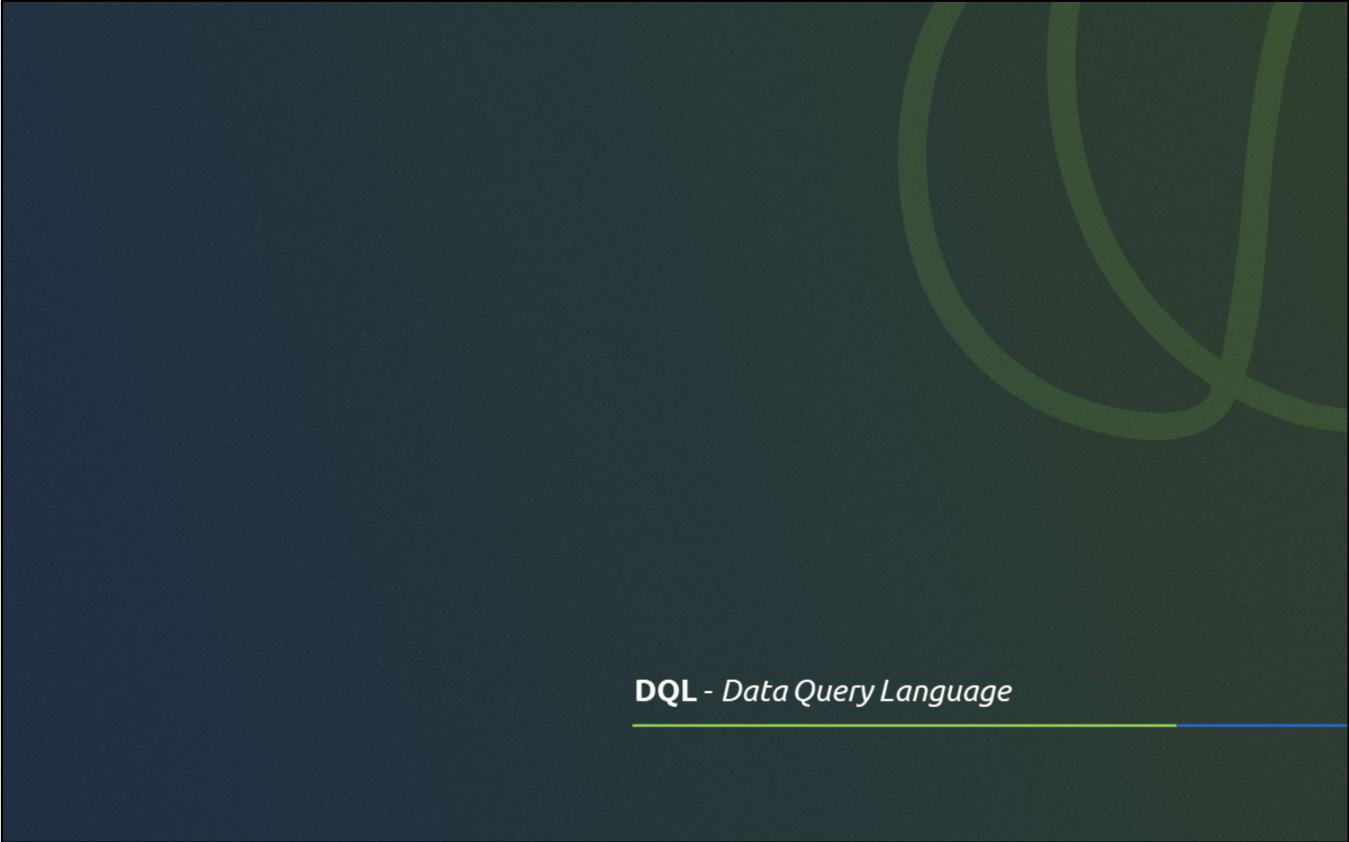
(<https://www.postgresql.org/docs/15/sql-creatematerializedview.html>)

DROP VIEW

(<https://www.postgresql.org/docs/15/sql-dropview.html>)

REFRESH MATERIALIZED VIEW

(<https://www.postgresql.org/docs/15/sql-refreshmaterializedview.html>).



DQL - *Data Query Language*

DQL

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
SELECT [ ALL | DISTINCT [ ON ( expression [, ...] ) ] ]
       [ * | expression [ [ AS ] output_name ] [, ...] ]
       [ FROM from_item [, ...] ]
       [ WHERE condition ]
       [ GROUP BY [ ALL | DISTINCT ] grouping_element [, ...] ]
       [ HAVING condition ]
       [ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
       [ ORDER BY expression [ ASC | DESC | USING operator ] [ NULLS { FIRST | LAST } ] [, ...] ]
       [ LIMIT { count | ALL } ]
```



<https://www.postgresql.org/docs/15/sql-select.html>

<https://www.postgresql.org/docs/15/functions-conditional.html#FUNCTIONS-CASE>

Obrigada