

# Fundamentos em Engenharia de Dados

## Capítulo 1. Conceitos Fundamentais

Prof. Jéssica Coelho

# Nesta aula

- ❑ Entender os conceitos de dado/informação/conhecimento/inteligência;
- ❑ Conhecer os diferentes tipos de dados;
- ❑ Conhecer as diferentes fontes de dados.

Dado



Dados são valores brutos, literais, observações ou medições do mundo real. Não estão contextualizados.

Informação



Dados agrupados dotados de relevância, de significado, que passam uma mensagem.

Conhecimento



Conjunto de informações somadas à experiência

Inteligência



Alguns autores falam sobre a Inteligência, que é a capacidade de avaliar os conhecimentos e tomar decisões sobre eles.

# Tipos de Dados

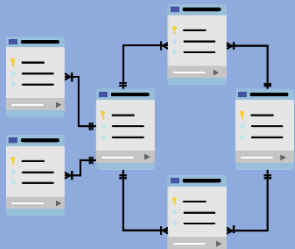
## ESTRUTURADOS

Estrutura homogênea e pré-definida

Estrutura independente dos dados

Clara distinção entre estrutura e dados

Estrutura sofre pouca alteração



## SEMIESTRUTURADOS

Estrutura heterogênea e nem sempre pré-definida

Estrutura embutida nos dados

Distinção entre estrutura e dados pouco clara

Estrutura sofre alteração com frequência



## NÃO ESTRUTURADOS

Sem esquema pré-definido

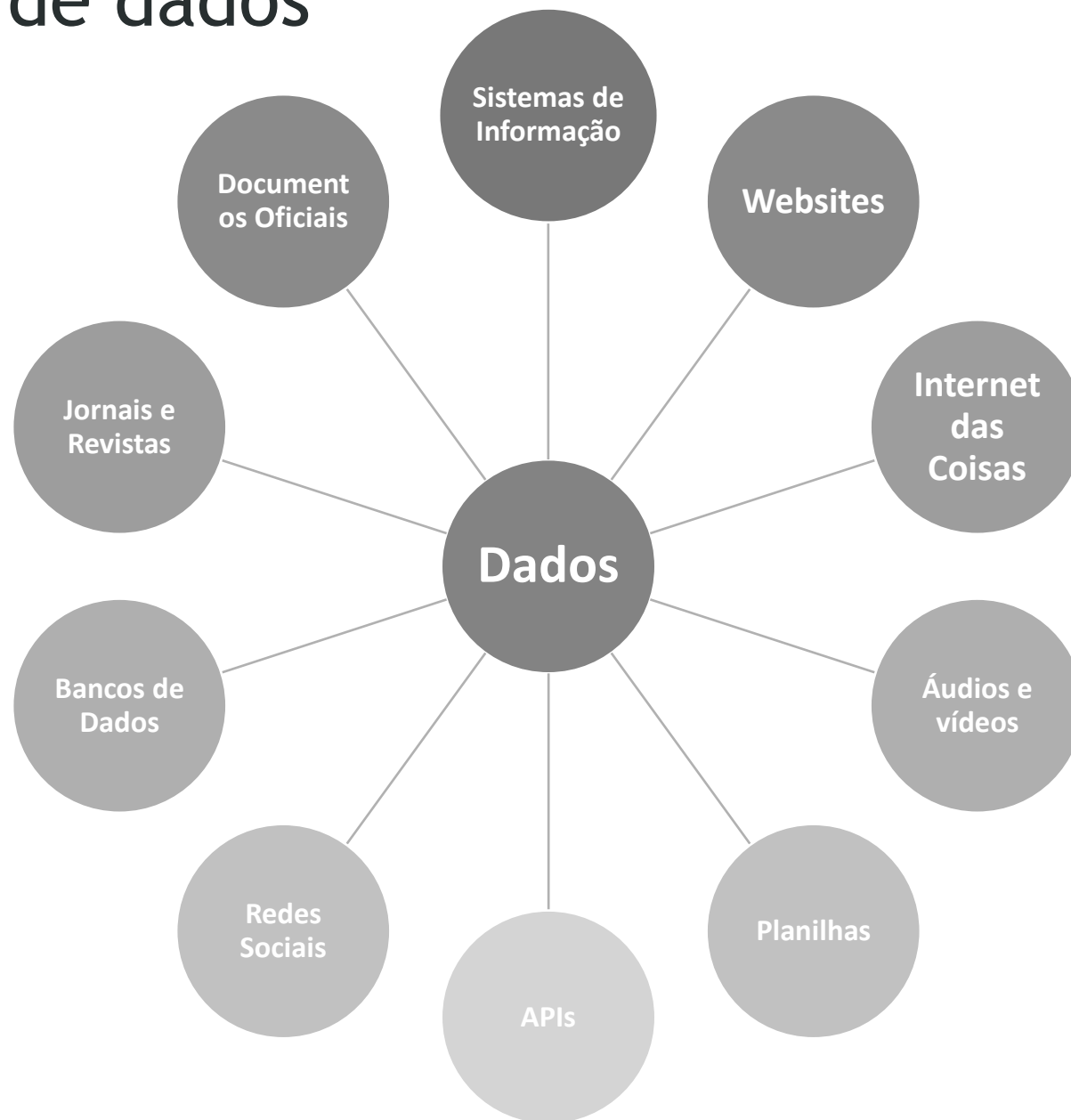
Estrutura independente dos dados

Estrutura irregular nem sempre presente

Estrutura sofre alteração com frequência



# Fontes de dados



# Conclusão

- ❑ Entendemos os conceitos de dados, informação, conhecimento e inteligência;
- ❑ Conhecemos os diferentes tipos de dados: Estruturados, Semiestruturados e Não estruturados;
- ❑ Conhecemos as fontes de dados e como isso define o formato;

# Próxima aula

- ❑ Descreveremos o termo Big Data;
- ❑ Conheceremos os tipos de análises;
- ❑ Conheceremos as responsabilidades de um Engenheiro de Dados

# Fundamentos em Engenharia de Dados

Capítulo 1.2 Big Data, Tipos de análises e o que é Engenheiro de dados

Prof. Jéssica Coelho



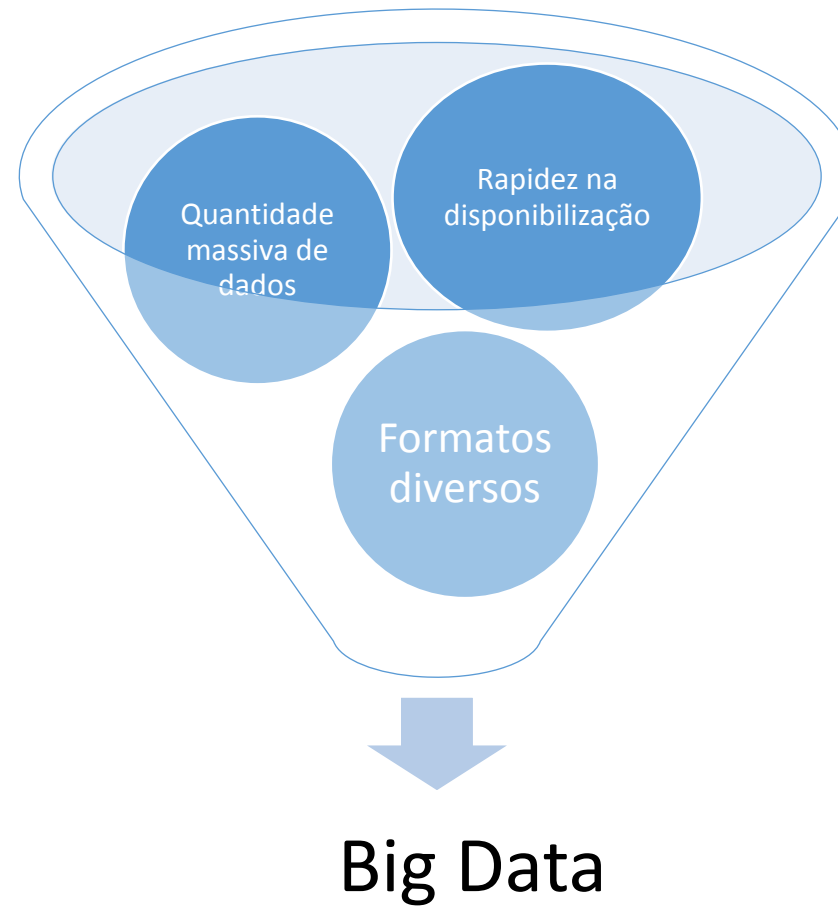
# Nesta aula

- ❑ Descreveremos o termo Big Data;
- ❑ Conheceremos os tipos de análises;
- ❑ Conheceremos as responsabilidades de um Engenheiro de Dados

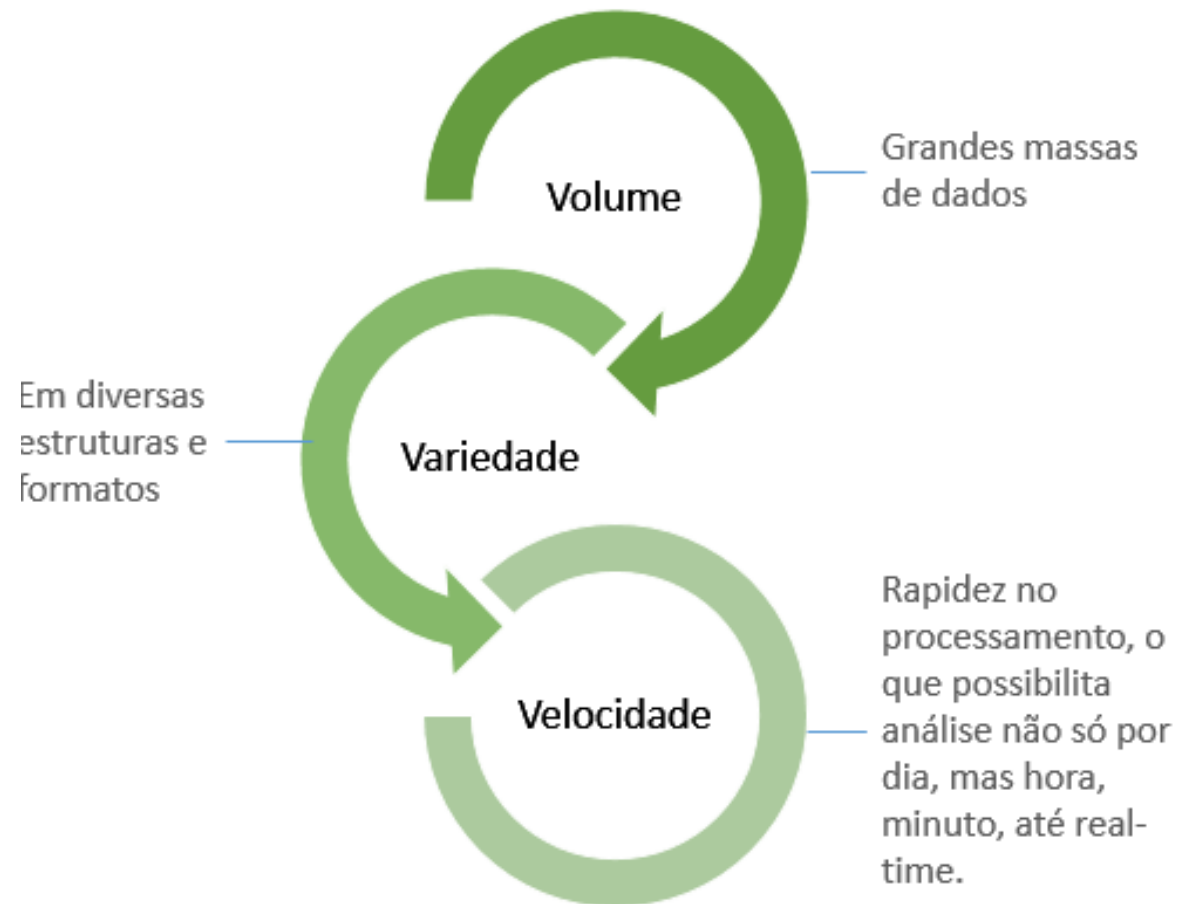
# BIG Data

Termo criado em 1990 na NASA, quando durante simulações e estudos do fluxo de ar de uma aeronave em voo se depararam com uma imensa massa de dados que não puderam ser processados.

***Então escreveram no relatório final: “Isso é um problema de Big Data”***



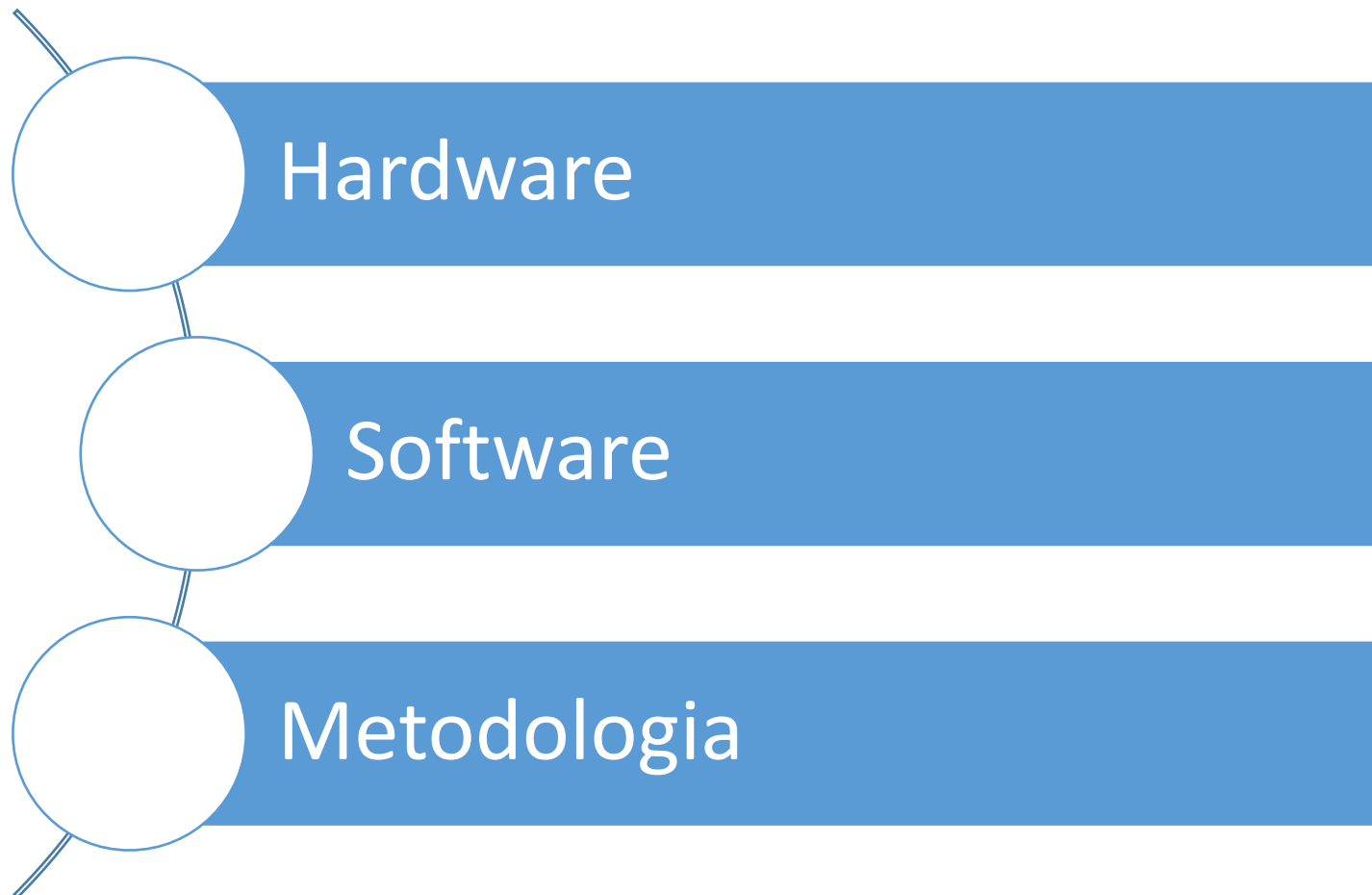
# 3 principais V's



Mas temos outros V's?



## O que o BIG Data não é?



# Análise de Dados

Técnicas para extrair informações valiosas a partir dos dados

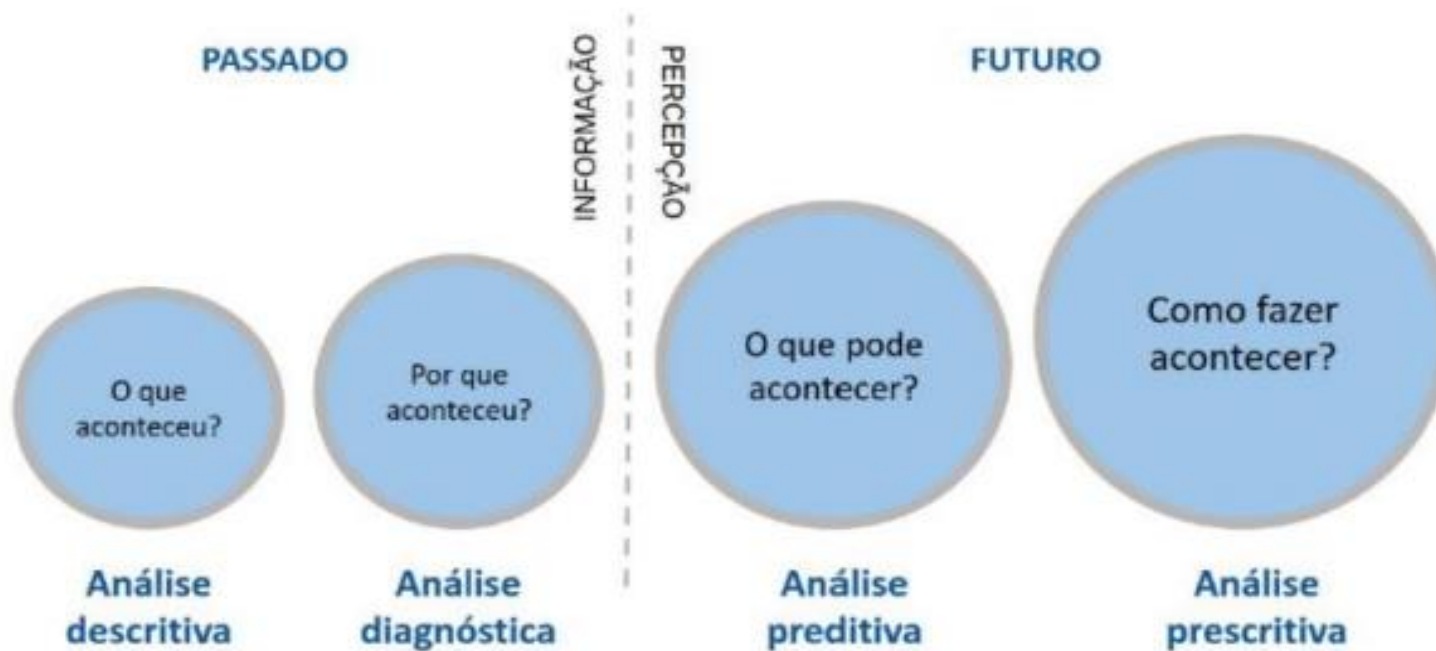
Busca identificar oportunidades de negócio

Busca entender melhor o comportamento humano

Tem como matéria prima o dado e como resultado, o conhecimento

Utiliza modelos matemáticos para encontrar padrões

# Tipos de análise



Fonte: Ferri (2019).

# O que faz um Engenheiro de Dados?

## Planejar e construir

- Modelos de dados;
- Esquema dos dados com base nos modelos.

## Coletar

- Entender a origem/tipos de dados;
- Definir estratégias para acessar e coletar os dados;

## ETL

- Construir os processos de transformação dos dados;

## Processar

- Grandes massas de dados em tempo hábil;

## Armazenar

- Decidir onde e como armazenar – DB, DW, Cubo, S3

## Disponibilizar /Segurança

- Construir as visualizações e configurar os níveis de acesso



# Conclusão

- ❑ Aprendemos o que é Big Data e seus V's;
- ❑ Aprendemos o que não é Big Data;
- ❑ Conhecemos os tipos de análises;
- ❑ E entendemos as responsabilidades de um Engenheiro de Dados

# Próxima aula

- ❑ Entenderemos o que é abstração;
- ❑ Entenderemos o que é modelagem;
- ❑ Falaremos sobre os tipos de modelagem;

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 1 - Modelagem Fundamentos

Prof. Jéssica Coelho

# Nesta aula

- ❑ Entenderemos o que é abstração;
- ❑ Entenderemos o que é modelagem;
- ❑ Falaremos sobre os tipos de modelagem;

# Abstração de Dados

Técnica de focar nos elementos principais dos dados:

Significado

Características

Como se  
relacionam

Função

Quais  
processos  
suportam

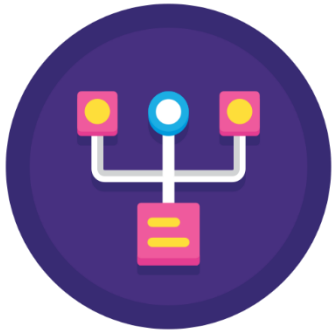
E menos em:

Como são  
armazenados

Como são  
processados  
internamente

Quais os tipos  
de dados

# Modelagem



**Modelo:** Abstração de um objeto ou evento da realidade. Geralmente representado por diagramas.

**Modelos de dados:** são representações visuais dos elementos de dados de uma empresa e as conexões entre eles. Ajudam a definir e estruturar dados no contexto de processos empresariais relevantes;

**Modelagem de dados:** conjunto de técnicas utilizadas para criar um modelo de dados que explique as características de funcionamento e de comportamento dos dados em um determinado sistema ou aplicação

# Modelos

## Conceitual

Altamente abstrato

Independente de tecnologias

O principal diagrama é o DER

## Lógico

Abstração média

Ainda independente de tecnologia

Fase de escolha do paradigma: relacional, orientado a objetos, hierárquico

## Físico

Baixa abstração

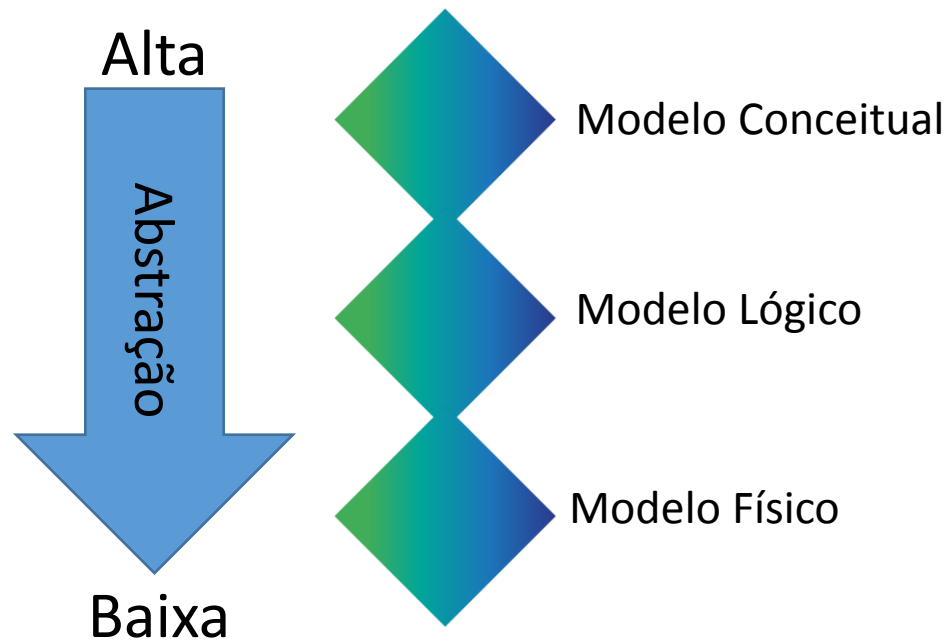
Escolha do SGBD de acordo com o paradigma escolhido

Detalhamento máximo do dado, com seus tipos e características técnicas

Respeita as restrições do SGBD escolhido

# Grau de Abstração

- Quanto maior o grau de abstração menor é o detalhe e preocupação com aspectos internos. Esse conceito é importante para entendermos os tipos de Modelos.





# Fases da modelagem



# Conclusão

- ❑ Entendemos o que é abstração;
- ❑ Entendemos o que é modelagem, seus tipos e importância;

# Próxima aula

- ❑ Entender o que é modelo conceitual;
- ❑ Conhecer o modelo entidade-relacionamento e seus principais elementos;

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 2 - Modelo Conceitual

Prof. Jéssica Coelho

# Nesta aula

- ❑ Entender o que é modelo conceitual;
- ❑ Conhecer o modelo entidade-relacionamento e seus principais elementos;

# Modelo Conceitual

Primeiro nível de modelagem em que temos as seguintes características:

É o de maior nível de abstração e mais próximo da realidade dos usuários

É construído na fase de levantamento de requisitos

O principal artefato desse modelo é o MER – Modelo de Entidade e Relacionamento

O MER é graficamente representado pelo DER – Diagrama de Entidade e Relacionamento

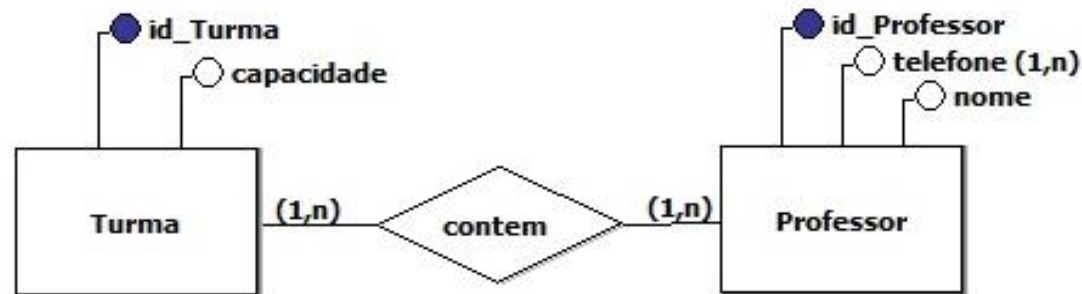
Não há preocupação ou menção à tecnologia ou ferramentas, mas apenas no entendimento dos agentes que executam as ações, suas características e como se relacionam

É usado para comunicação com a camada não técnica do processo

Pode assumir diversas notações

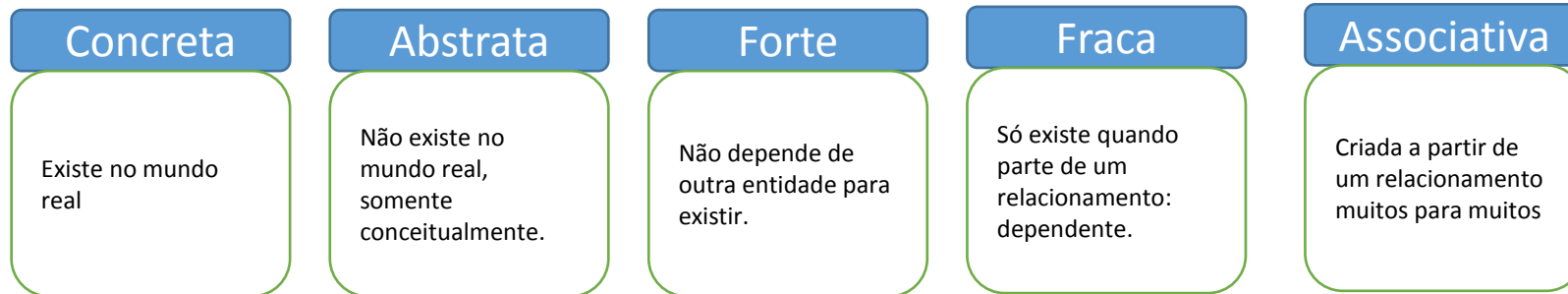
# Modelo Entidade e Relacionamento - MER

- ❑ Criado por Peter Chen (1976) baseado na teoria relacional de Edgar Codd (1970).
- ❑ Elementos principais: Entidades, atributos e relacionamentos.



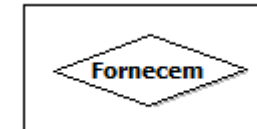
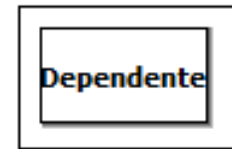
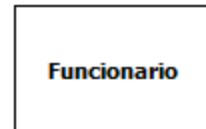
# Entidade

- Entidade é o objeto básico do modelo ER, é algo no mundo real com uma existência independente.



Notação Peter Chen

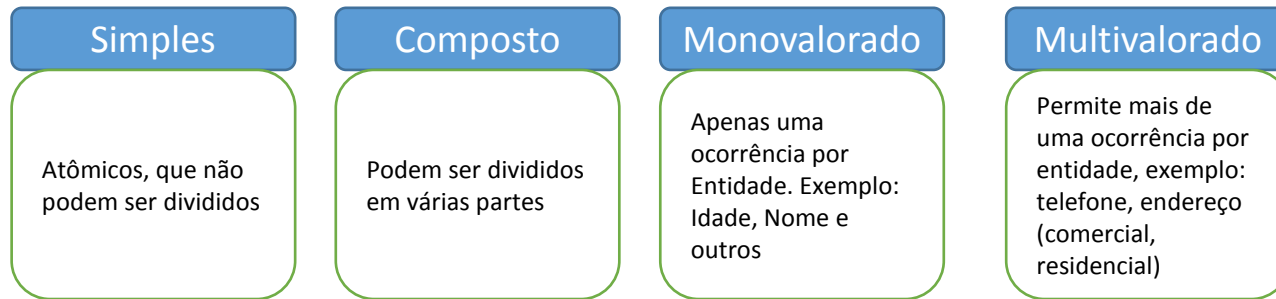
Não tem diferença na representação





# Atributo

- São características, propriedades específicas que descrevem as entidades



Notação do  
Peter Chen

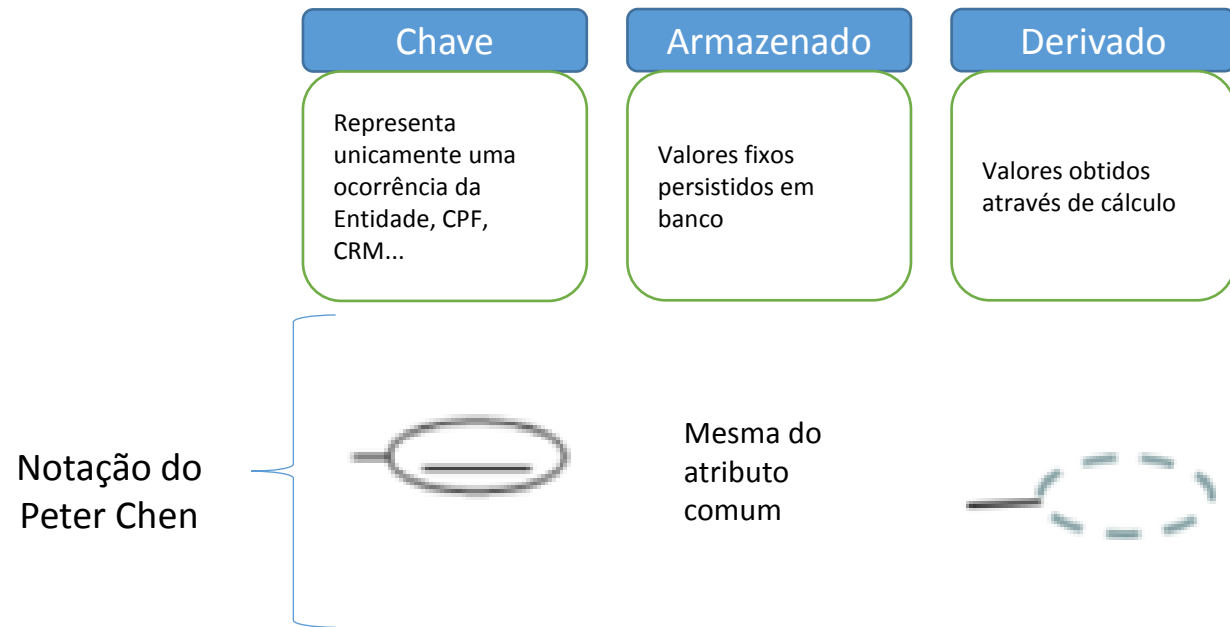


Mesma do  
atributo  
comum

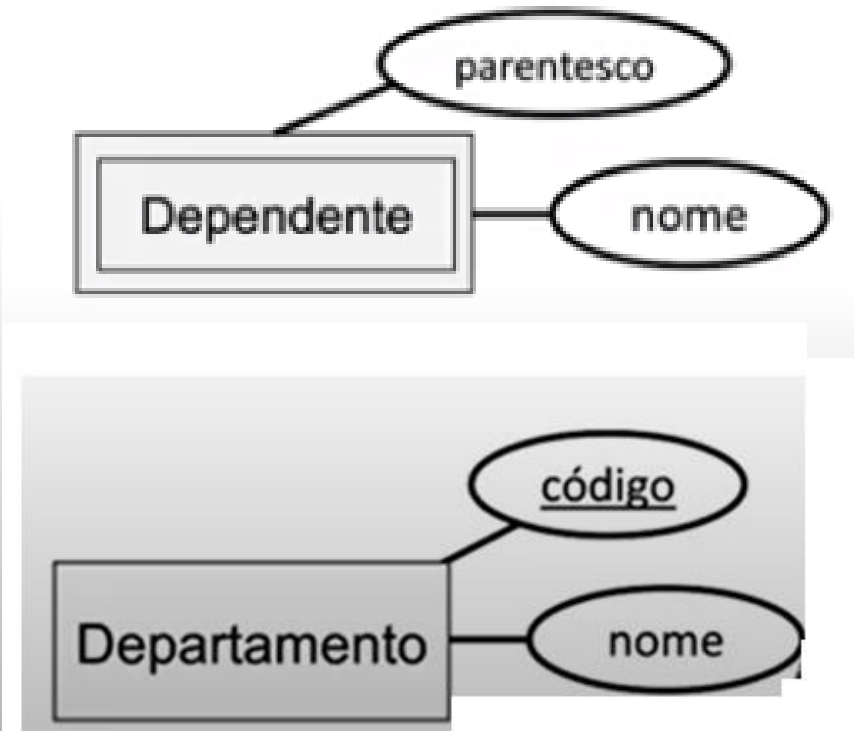
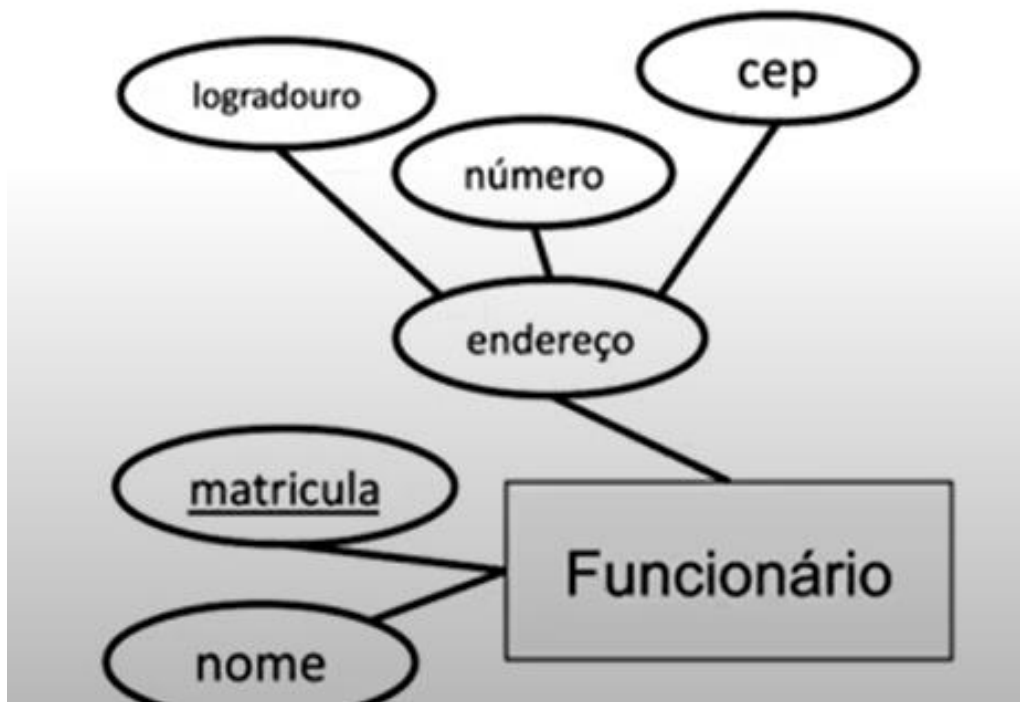


# Atributo

- São características, propriedades específicas que descrevem as entidades



# Entidades e seus atributos



# Conclusão

- ❑ Entendemos o que é modelo conceitual e conhecemos o MER;
- ❑ Conhecemos a notação de Peter Chen.

# Próxima aula

- Estudar os relacionamentos entre entidades e seus elementos;

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 3 - Modelo Conceitual - Parte 2

Prof. Jéssica Coelho

# Nesta aula

- ❑ Estudar os relacionamentos entre entidades e seus elementos;
- ❑ Estudaremos cardinalidade e obrigatoriedade de um relacionamento;

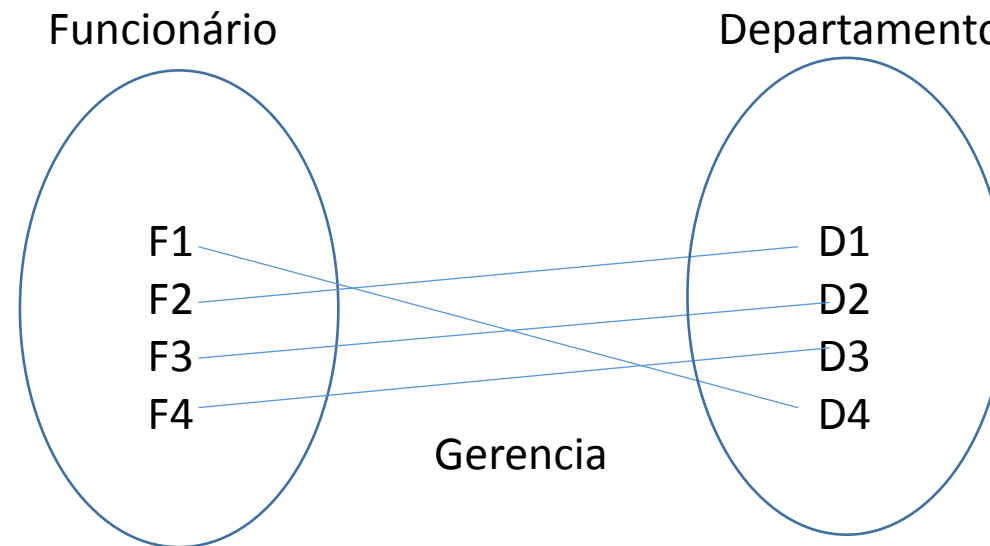
# Relacionamentos

- ❑ As entidades são relacionadas umas às outras através de um relacionamento;
- ❑ Em geral é expresso por verbo ou locução verbal;
- ❑ Restrições ou características de um relacionamento:
  - ❑ Cardinalidade: quantidade de ocorrências de (1:1, 1:N, N:N);
  - ❑ Obrigatoriedade (0,1): Se uma ocorrência de uma entidade é obrigatória ou não;
  - ❑ Participação:
    - Total: a entidade existe somente quando ela se relaciona com outra entidade;
    - Parcial: a existência de uma entidade independe do fato dela estar relacionada com outra.



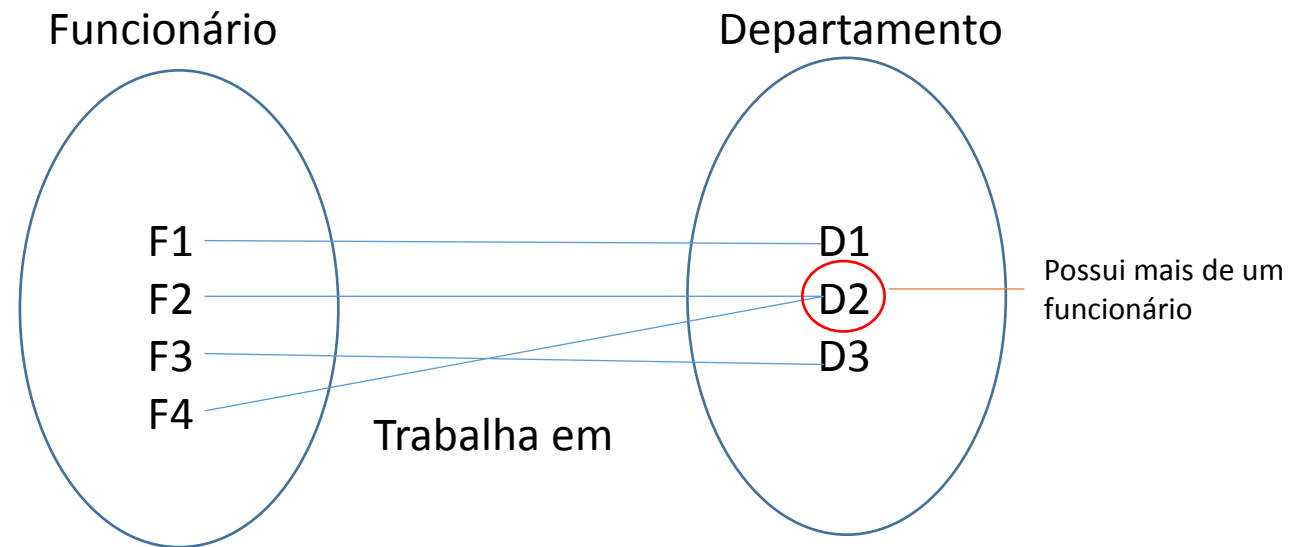
# Cardinalidade 1:1 (1..1)

Uma ocorrência da Entidade Funcionário só pode se relacionar com uma ocorrência da entidade Departamento:



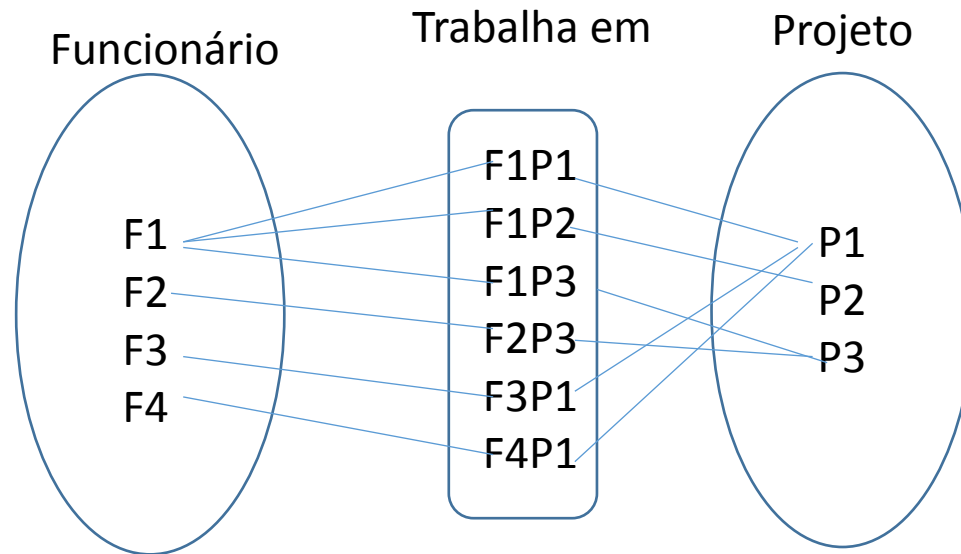
# Cardinalidade 1:N (1..\*)

Um empregado trabalha em um departamento e um departamento pode ter vários (N) empregados



# Cardinalidade N:N (\*..\*)

Empregado pode trabalhar em vários (N) projetos e um projeto pode possuir vários (N) empregados



# Obrigatoriedade:

Define se uma ocorrência de uma entidade é obrigatória ou não.

Notação (0,1) - Sendo 0 para não obrigatório e 1 para obrigatório.

# Representação gráfica (Peter Chen)

❑ Cardinalidade 1:N ou N:1



❑ Cardinalidade 1:1



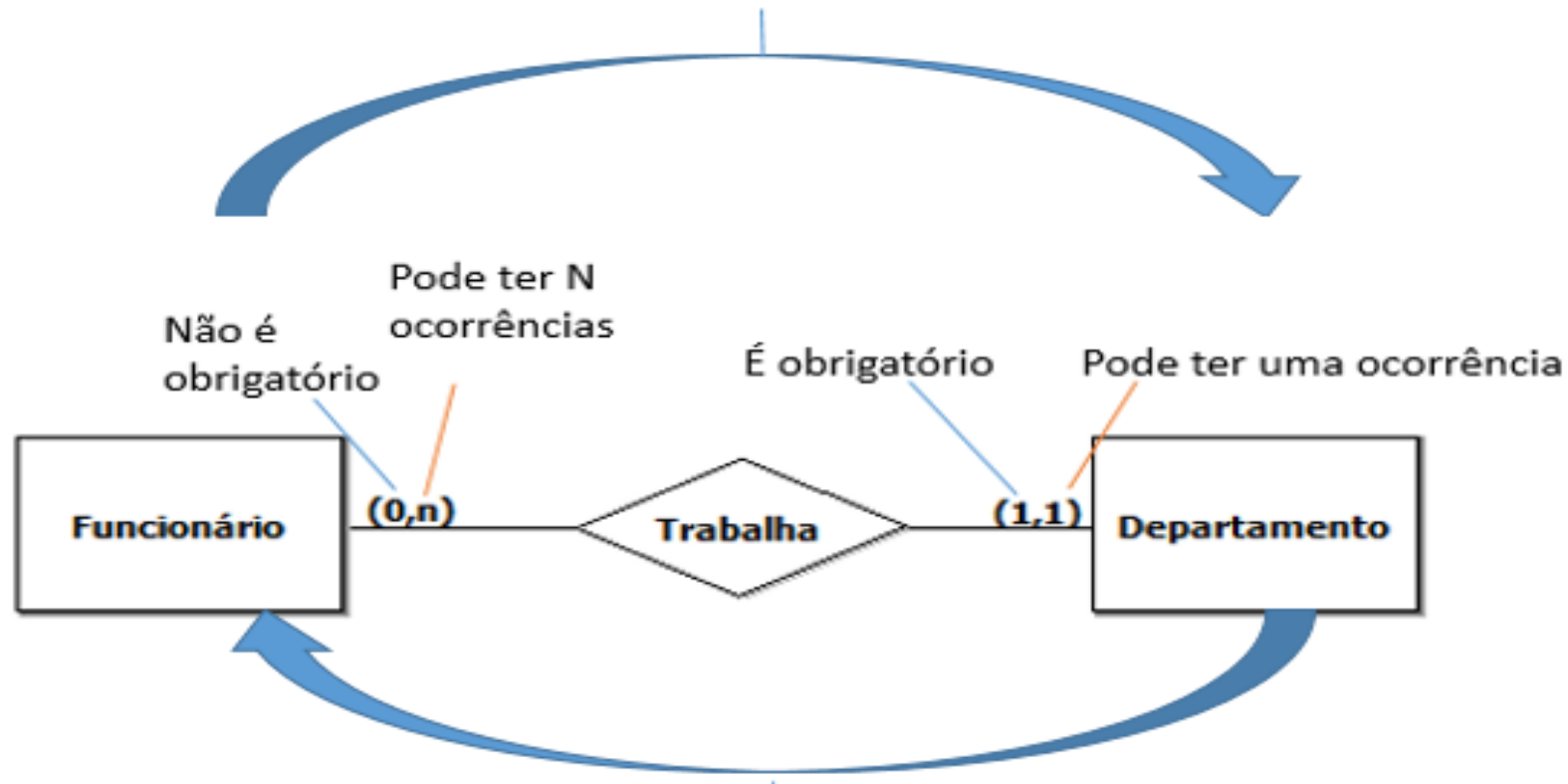
❑ Cardinalidade N:N:



Primeiro valor à esquerda representando a obrigatoriedade (0 ou 1) e após a vírgula temos a quantidade de ocorrências possíveis no relacionamento (1,N,3,4,5 e etc.)

## Exemplo de como se lê:

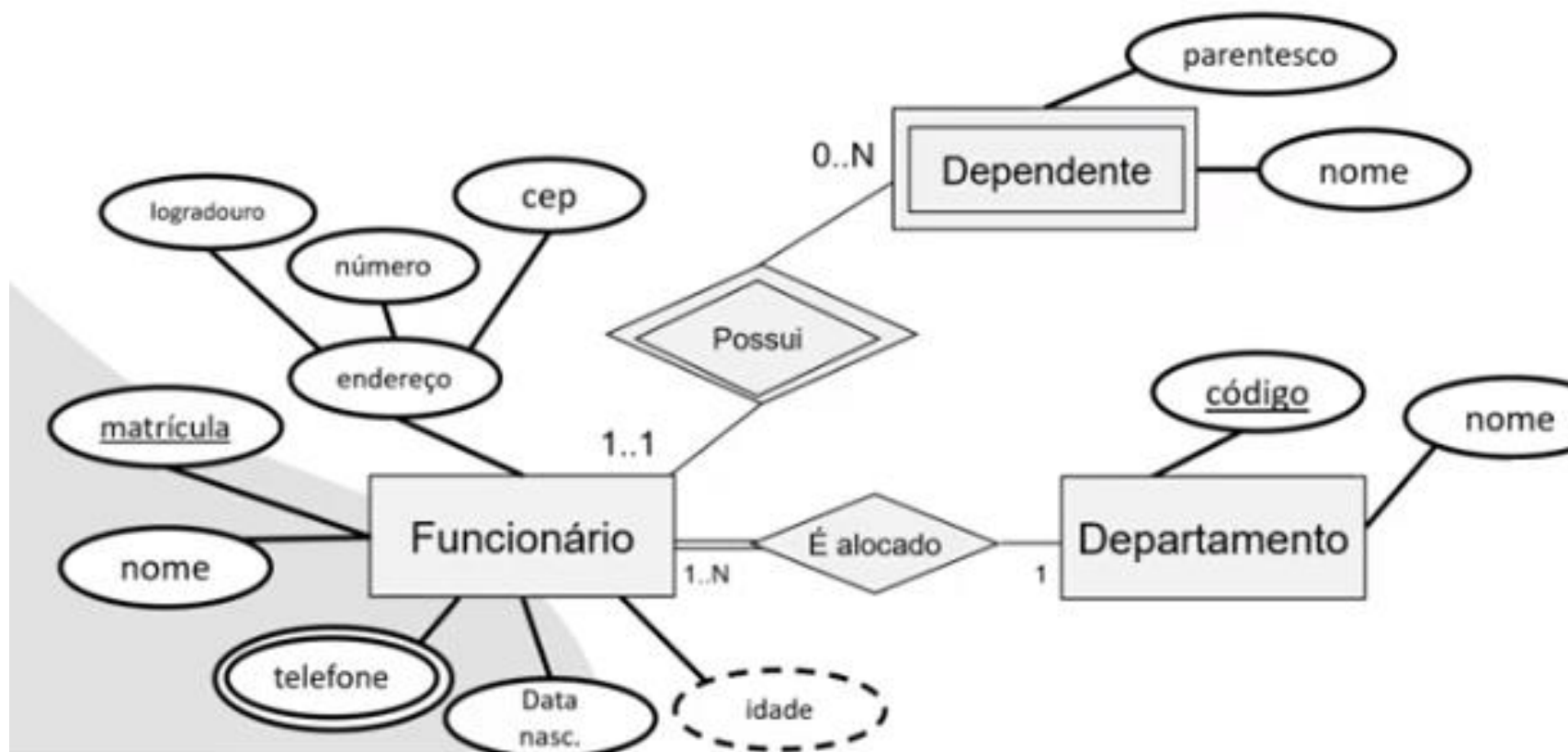
De Funcionário para Departamento lê-se: Um Funcionário trabalha em 1 (obrigatório) ou no máximo 1 (ocorrências) Departamento.



De Departamento para Funcionário lê-se: Um Departamento pode possuir 0 (não obrigatório) ou muitos - N (ocorrências) Funcionários.

# Entidade, relacionamentos e atributos

Representação Gráfica Peter Chen:



# Conclusão

- ❑ Entendemos os relacionamentos entre entidades e seus elementos;
- ❑ Estudamos cardinalidade e obrigatoriedade de um relacionamento;
- ❑ Estudamos a notação Peter Chen.



# Próxima aula

- Continuaremos estudando o MER - Modelo de Entidade e Relacionamento e seus elementos.

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 4 - Modelo Conceitual - Parte 3

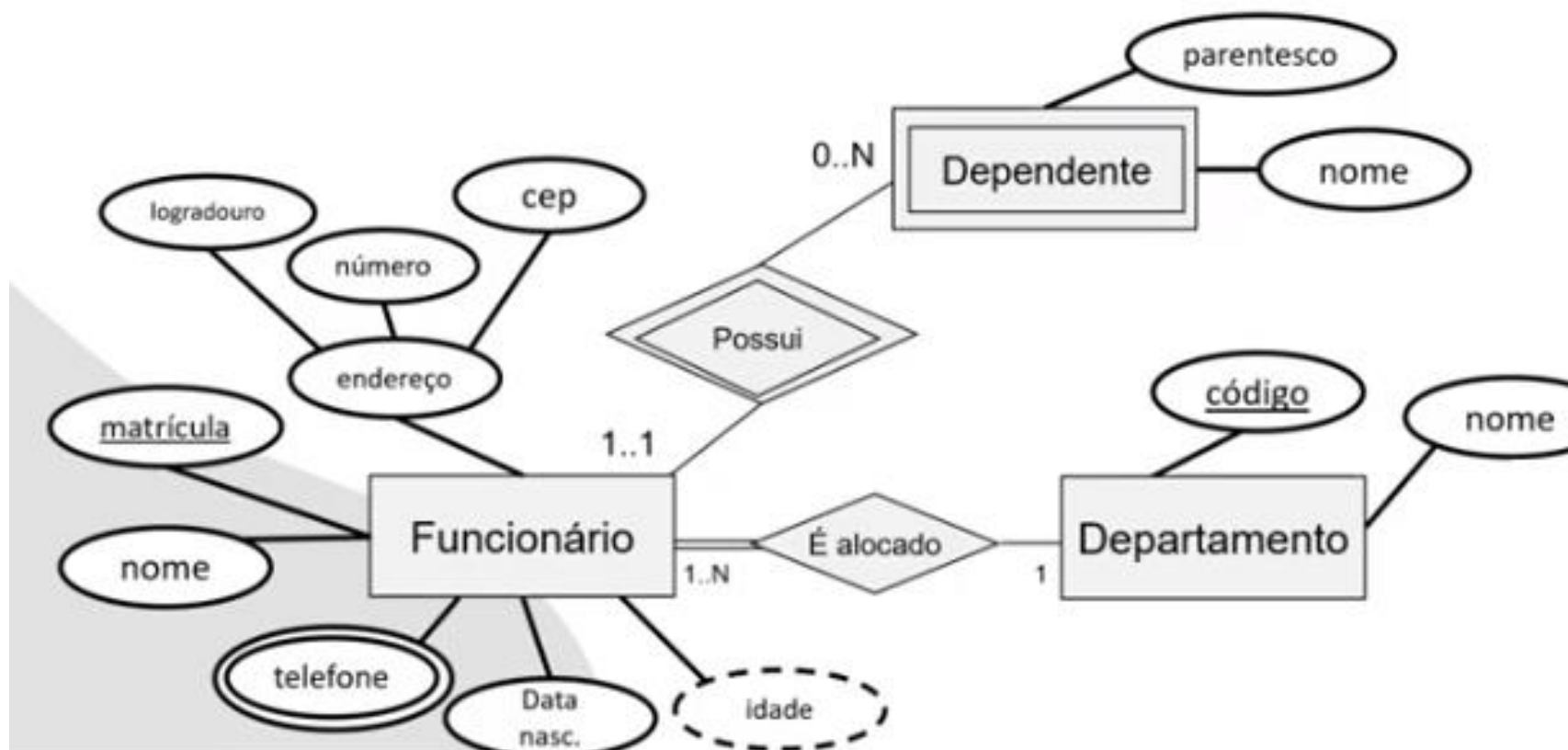
Prof. Jéssica Coelho

# Nesta aula

- Continuaremos estudando o MER - Modelo de Entidade e Relacionamento e seus elementos.

# Entidade, relacionamentos e atributos

Representação Gráfica Peter Chen:



# Características de um Relacionamento

☐ Pela quantidade de Entidades envolvidas:

☐ Binário-Ternário...;

☐ Auto-Relacionamento ou recursivo (unário)

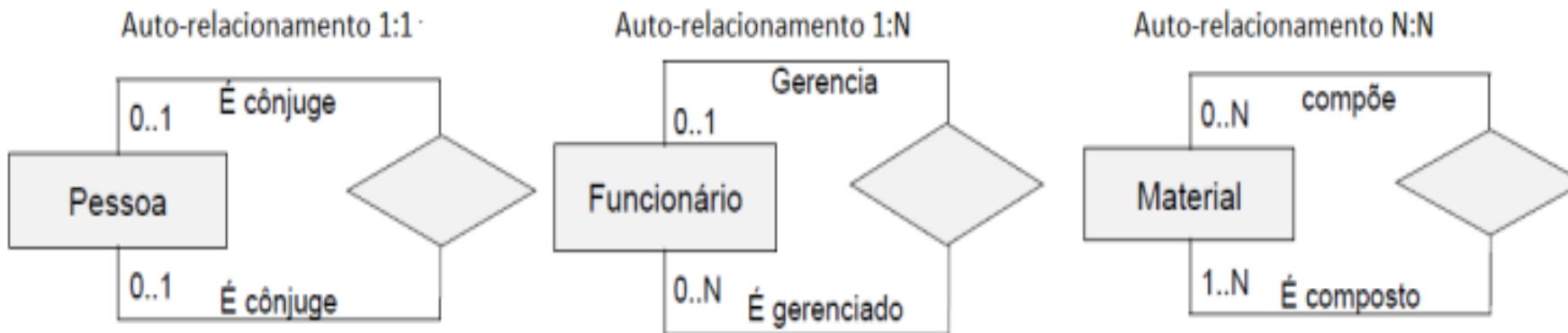
☐ Pela participação:

☐ Total;

☐ Parcial

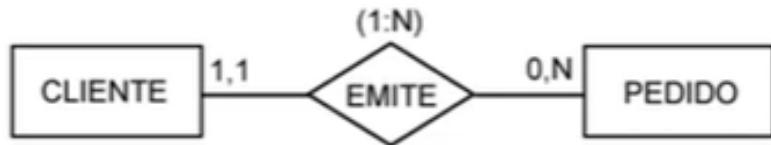
# Auto-relacionamento:

- ❑ Relacionamento entre ocorrências de uma mesma entidade (Unário)

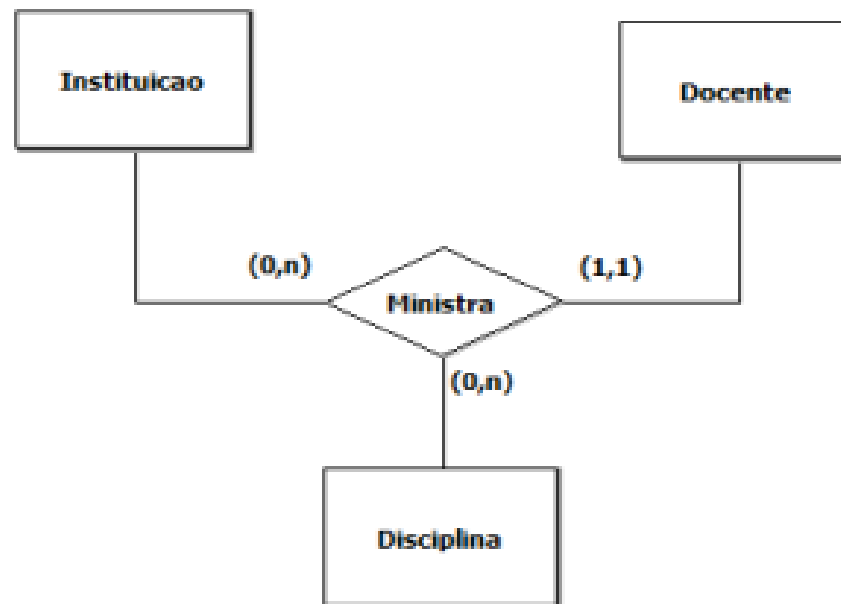


# Binário / Ternário

❑ Binário: Envolve 2 entidades:

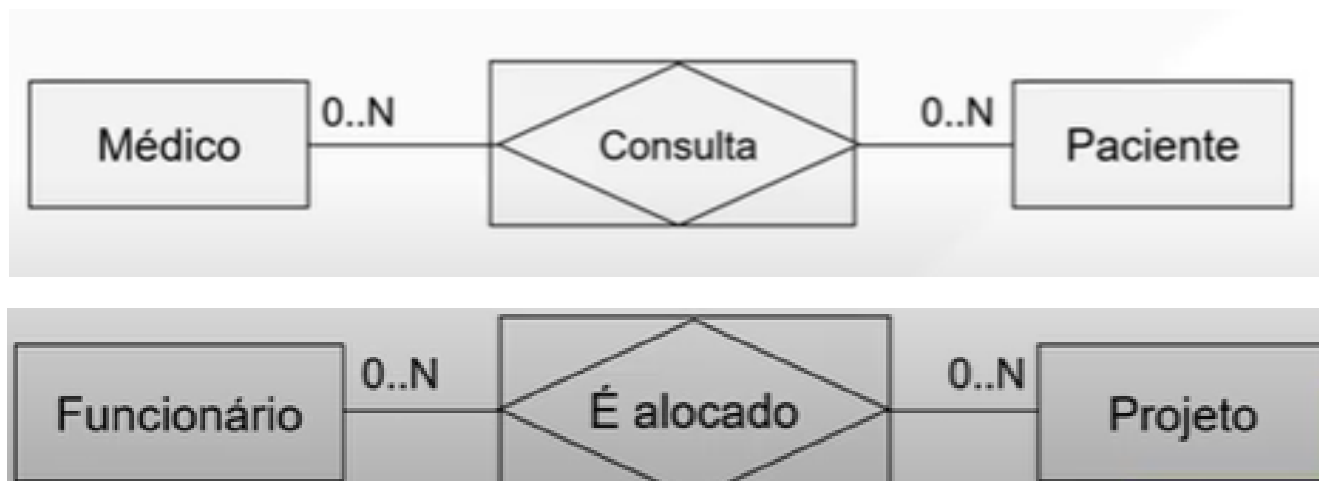


❑ Ternário: Envolve 3 entidades:



# Entidade Associativa

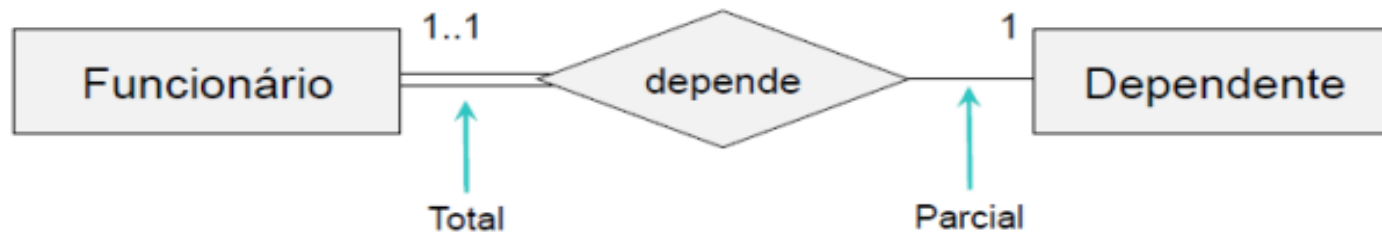
- ❑ Entidade proveniente de um relacionamento N:N (\*..\*) e pode ter seus próprios atributos:





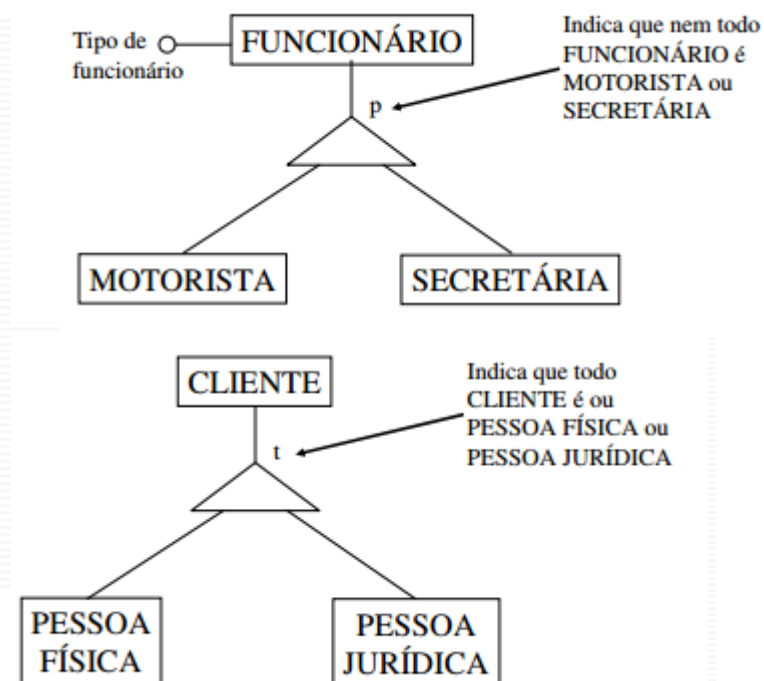
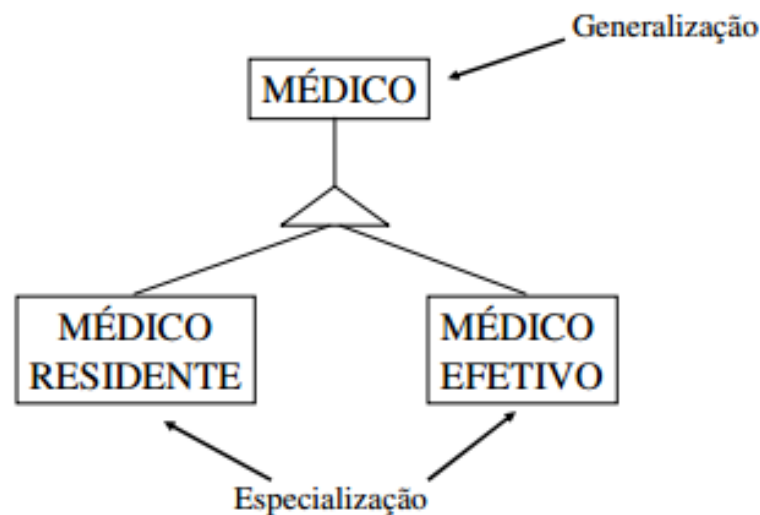
# Participação

- ❑ Total: Entidade só existe quando se relaciona com outra;
- ❑ Parcial: Entidade existe mesmo fora de um relacionamento;

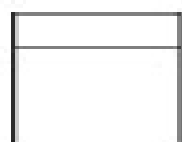


# Generalização/Especialização

- ❑ Total: Para cada ocorrência de entidade genérica obrigatoriamente tenho pelo menos uma ocorrência da especializada.
- ❑ Parcial: Nem toda ocorrência da entidade genérica possui correspondência em uma entidade especializada.



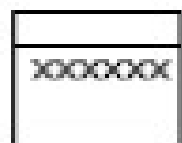
# Outras notações - James Martin



ENTIDADE



RELACIONAMENTO



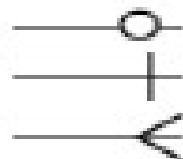
ATRIBUTO



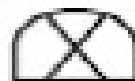
ATRIBUTO CHAVE



ENTIDADE DEPENDENTE

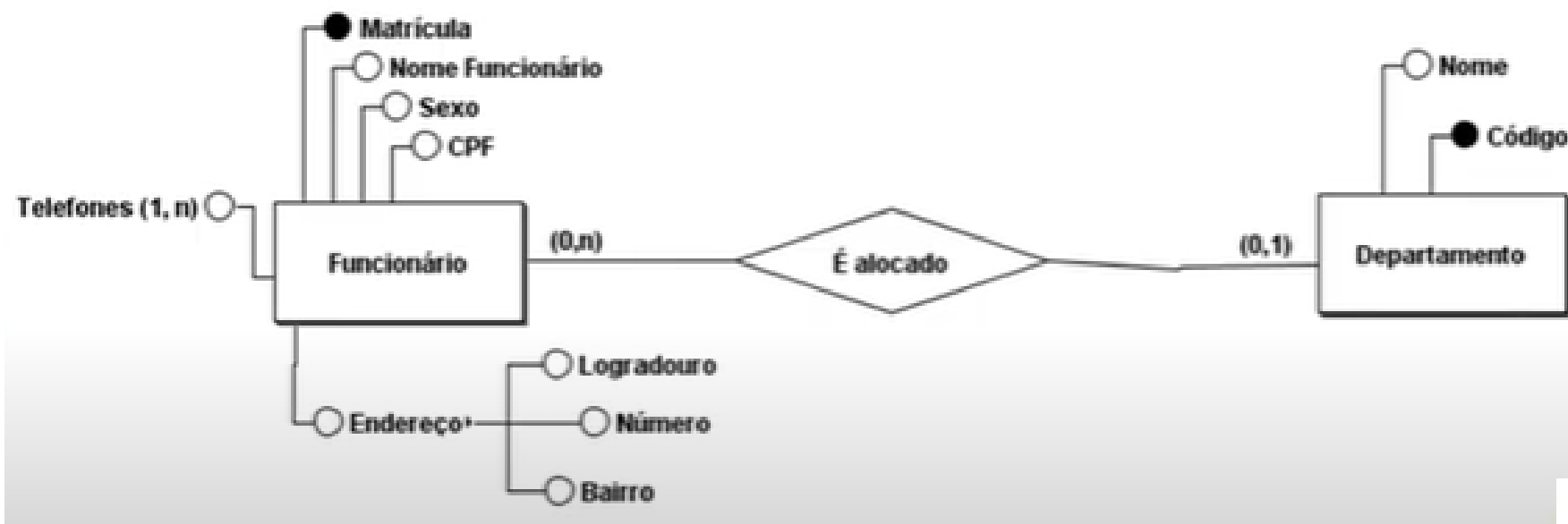


CARDINALIDADE



ESPECIALIZAÇÃO

## Outras notações - Heuser



# Conclusão

- ❑ Conhecemos mais características do Modelo de Entidade e Relacionamento.
- ❑ Conhecemos outras notações;

# Próxima aula

- ❑ Construir um MER juntos para praticarmos.

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 5 - Modelo Conceitual - Prática Parte 1

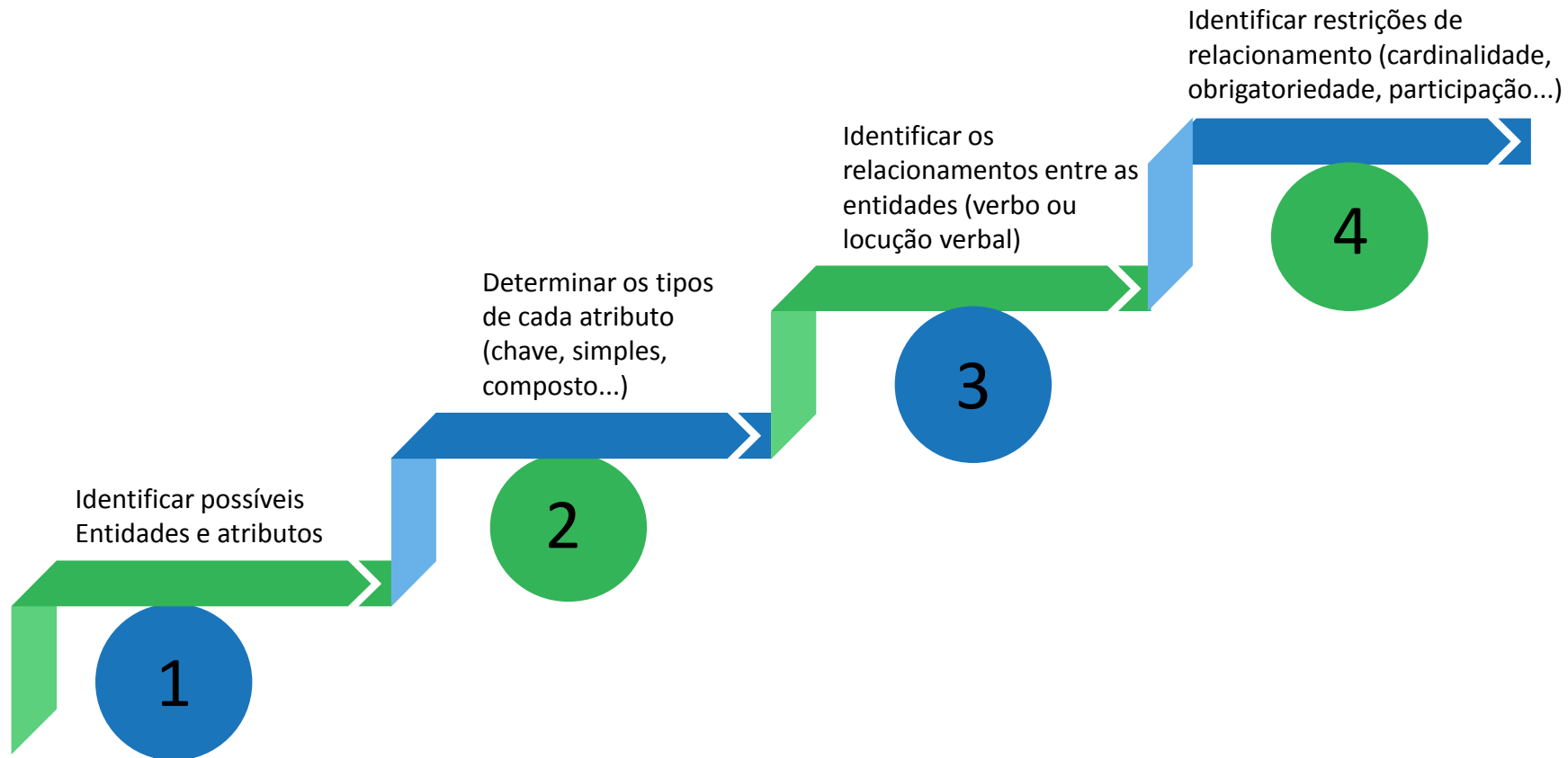
Prof. Jéssica Coelho

# Nesta aula

- ❑ Construir um MER juntos para praticarmos.



# Construção de um MER



## Exemplo de MER

- ❑ Empresa de seguros de automóveis: Cada **cliente** possui CPF, nome, sexo, endereço e telefones de contato (celular e fixo). Os **carros** possuem uma placa, marca, modelo, ano, chassi e cor. Cada carro tem um número de **sinistros** de acidentes associados a ele, sabendo que pode ter ocorrido múltiplos acidentes ou nenhum. Já os sinistros devem ser identificados por um código único e data e hora de ocorrência, local de ocorrência, condutor (que pode ou não ser o titular da apólice). Um cliente pode possuir várias apólices (mínimo uma) vigentes ou não, e cada apólice de seguro tem um identificador único e só pertence a um cliente e somente um carro, e tem data de início e fim da vigência, valor total assegurado, valor franquia associados a ela. É importante saber que o carro pode ter várias apólices vinculadas a ele, mas apenas uma vigente.

## 1

## Identificar possíveis Entidades

# Exemplo de MER

- ❑ Empresa de seguros de automóveis: Cada **cliente** possui **CPF**, **nome**, **sexo**, **endereço** e **telefones de contato (celular e fixo)**. Os **carros** possuem uma **placa**, **marca**, **modelo**, **ano**, **chassi** e **cor**. Cada carro tem um número de **sinistros** de acidentes associados a ele, sabendo que pode ter ocorrido múltiplos acidentes ou nenhum. Já os sinistros devem ser identificados por um **código único** e **data** e **hora de ocorrência**, **local de ocorrência**, **condutor** (que pode ou não ser o titular da apólice). Um cliente pode possuir várias **apólices** (mínimo uma) vigentes ou não, e cada apólice de seguro tem um **identificador único** e só pertence a um cliente e somente um carro, e tem **data de início** e **fim da vigência**, **valor total assegurado**, **valor franquia** associados a ela. É importante saber que o carro pode ter várias apólices vinculadas a ele, mas apenas uma vigente.

Tipos de  
atributos

## 2

## Identificar possíveis atributos

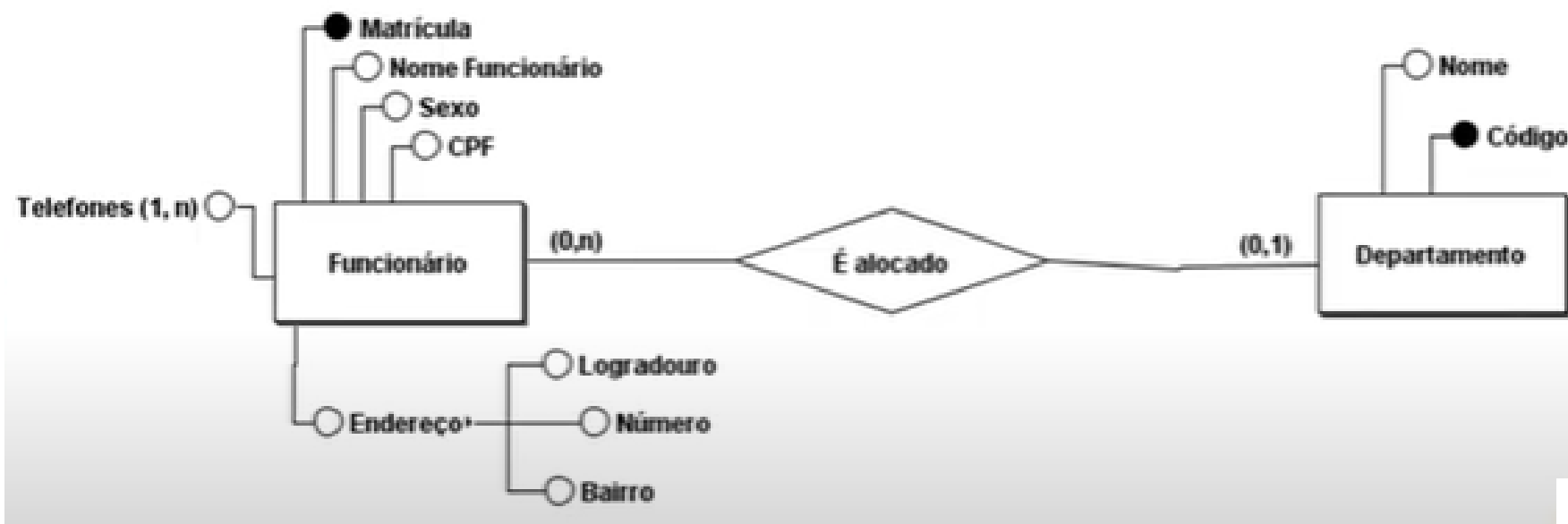
## 3

## Identificar possíveis relacionamentos

## Exemplo de MER

- ❑ Empresa de seguros de automóveis: Cada **cliente** possui **CPF**, **nome**, **sexo**, **endereço** e **telefones de contato (celular e fixo)**. Os **carros** possuem uma **placa**, **marca**, **modelo**, **ano**, **chassi** e **cor**. Cada carro tem **um** número de **sinistros** de acidentes associados a ele, sabendo que pode ter ocorrido **múltiplos acidentes ou nenhum**. Já os sinistros devem ser identificados por um **código único** e **data e hora de ocorrência**, **local de ocorrência**, **condutor** (que pode ou não ser o titular da apólice). Um cliente pode possuir **várias** **apólices** (**mínimo uma**) vigentes ou não, e cada apólice de seguro tem um **identificador único** e só pertence a **um** cliente e **somente um** carro, e tem **data de início** e **fim da vigência**, **valor total assegurado**, **valor franquia** associados a ela. É importante saber que o carro pode ter **várias** apólices vinculadas a ele, mas apenas uma vigente.

## Outras notações - Heuser e o BrModelo



# Conclusão

- ❑ Praticamos o passo a passo para encontrar as Entidades, Atributos e Relacionamentos para construção do modelo conceitual.
- ❑ Relembramos a notação de Heuser que será usado na próxima aula no BrModelo.

# Próxima aula

- ❑ Criaremos o modelo conceitual desenhado nesta aula, no BrModelo.

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 6 - Modelo Conceitual - Prática - Parte2

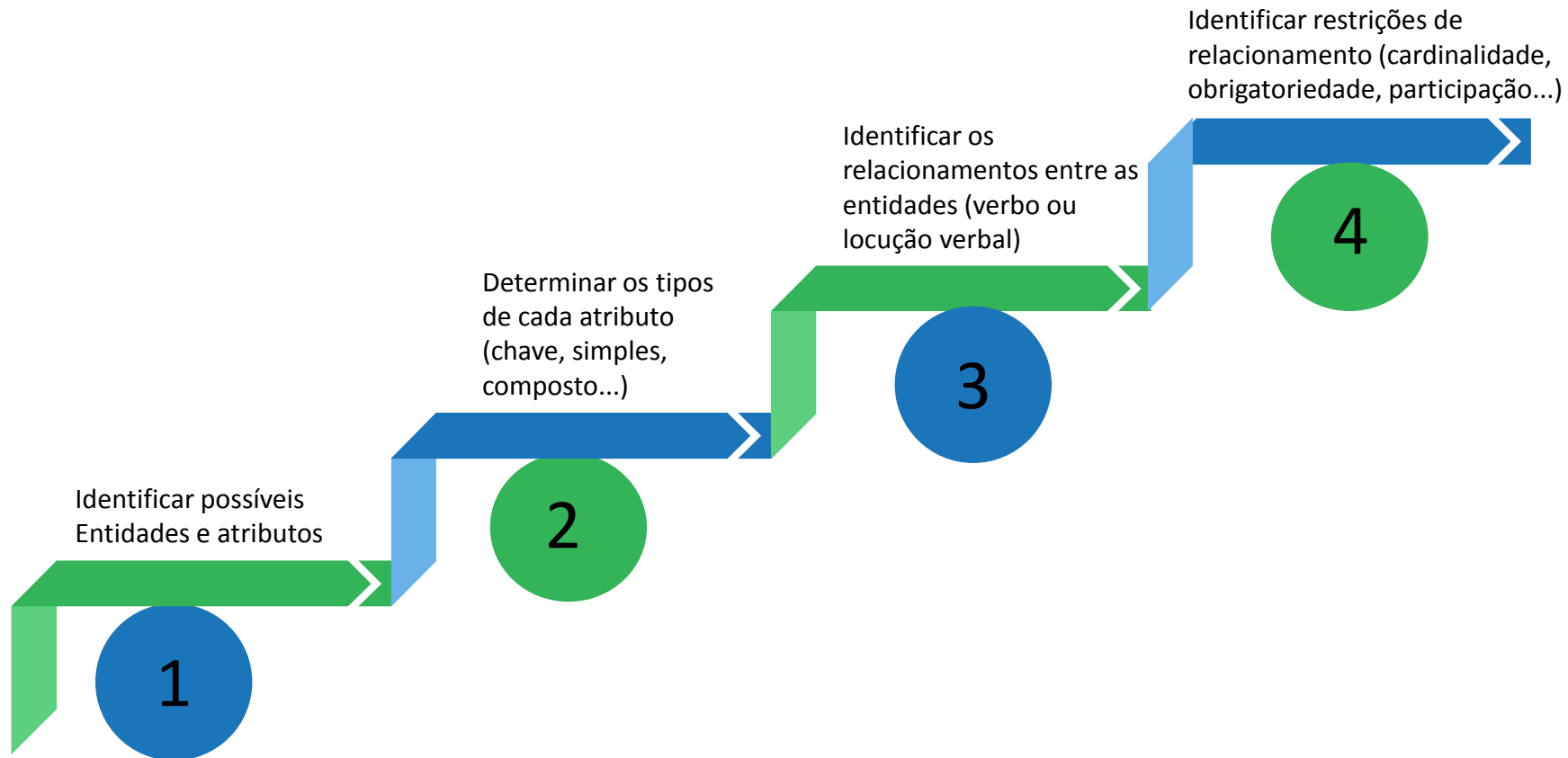
Prof. Jéssica Coelho



# Nesta aula

- ❑ Construir um MER juntos no BrModelo para praticarmos.

# Construção de um MER



## Exemplo de MER

- ❑ Empresa de seguros de automóveis: Cada cliente possui CPF, nome, sexo, endereço e telefones de contato (celular e fixo). Os carros possuem uma placa, marca, modelo, ano, chassi e cor. Cada carro tem um número de sinistros de acidentes associados a ele, sabendo que pode ter ocorrido múltiplos acidentes ou nenhum. Já os sinistros devem ser identificados por um código único e data e hora de ocorrência, local de ocorrência, condutor (que pode ou não ser o titular da apólice). Um cliente pode possuir várias apólices (mínimo uma) vigentes ou não, e cada apólice de seguro tem um identificador único e só pertence a um cliente e somente um carro, e tem data de início e fim da vigência, valor total assegurado, valor franquia associados a ela. É importante saber que o carro pode ter várias apólices vinculadas a ele, mas apenas uma vigente.

# Conclusão

- Continuamos a criação do MER no brModelo para a empresa de seguro de carros.

# Próxima aula

□ Estudaremos o Modelo Lógico.

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 7 - Modelo Lógico

Prof. Jéssica Coelho

# Nesta aula

- ❑ Conhecer o modelo lógico;
- ❑ Conhecer o modelo relacional de dados e seus elementos.

## Modelo lógico

Modelo hierárquico

Modelo de Rede

Modelo relacional

Modelo orientado a objetos

Modelo orientado a grafos

Modelo objeto-relacional

Já define  
paradigma,  
mas ainda  
não  
tecnologia



# Modelo Relacional



Definido por Edgar Cood (1970) e se tornou o modelo mais usado em 1980



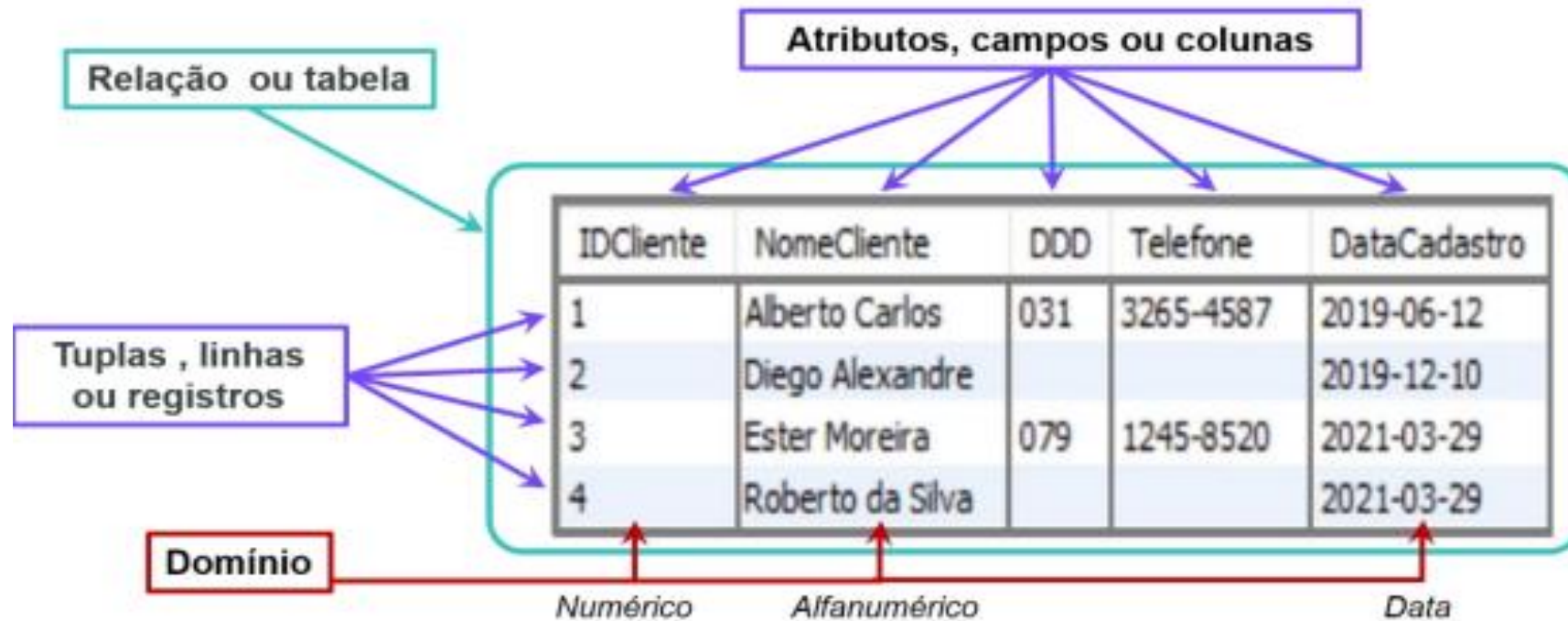
Representa um banco de dados com base na teoria de conjuntos e como estes se relacionam



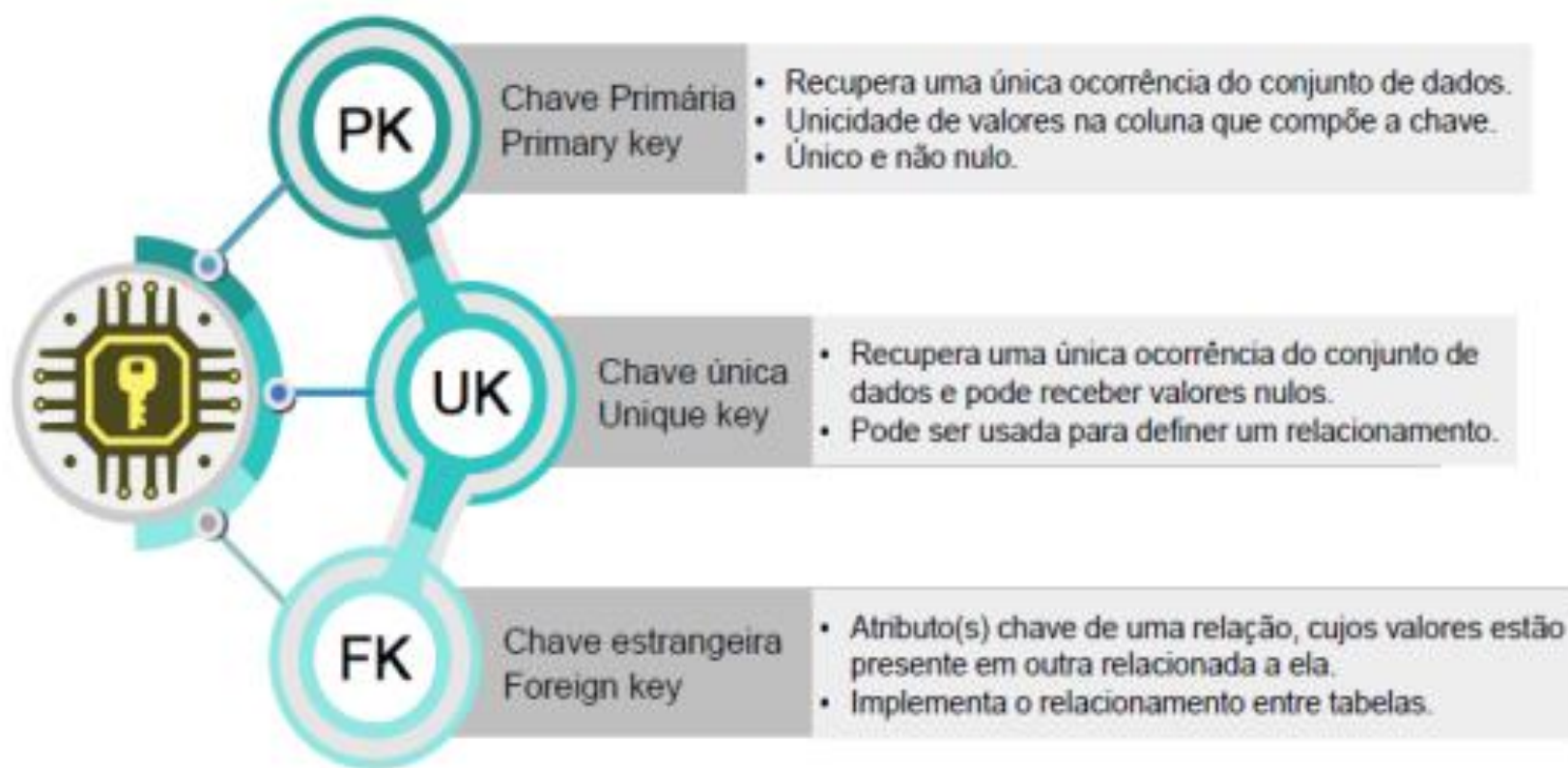
Em um banco de dados relacional, os dados estão organizados na forma de tabelas ou relações, possui colunas e se relacionam através das chaves

# Elementos do modelo relacional

- ❑ Tabela (relação) é um conjunto de linhas (tuplas);
- ❑ Cada linha pode ter um ou mais atributos, ou colunas (campos).



# Modelo relacional - Chaves



## Exemplo:

PK		UK	
IDCLIENTE	NomeCliente	DDDTelefone	CPF
1	Tamires Costa	3188559966	52016989636
2	Alberto Silva	3185552236	75878596385
3	Maria Antonia	3185965578	78945612330
4	Flávia Costa	3188778855	96988874580

Possíveis chaves ou chaves candidatas

# Relacionamento - FK

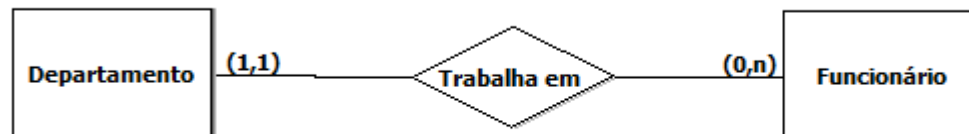
Departamento	
Cod_Departamento	Nome
1	TI
2	Engenharia
3	Compras

Primary Key

Funcionário		
Cod_Funcionario	Nome	Cod_Departamento
1	Alberto Lúcio	1
2	Maria Amélia	1
3	Carlos Eduardo	3

Foreign Key

Promove o relacionamento  
Garante integridade



# Conclusão

- ❑ Conhecemos o modelo lógico;
- ❑ Conhecemos o modelo relacional de dados e alguns dos seus elementos.
- ❑ Entendemos o que são chaves em um modelo relacional

# Na próxima aula

- Tipos de integridade no modelo relacional;

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 9 - Modelo Lógico - Prática Parte 1

Prof. Jéssica Coelho



# Nesta aula

- ❑ Construir um modelo lógico (Paradigma Relacional) baseado no MER desenvolvido na prática conceitual.

# Ferramenta BrModelo

❑ Pode baixar o BrModelo no link:  
<http://www.sis4.com/brModelo/download.html>

Neste link você terá o BrModelo3.0.zip>> basta extrair >> acessar a pasta “dist”>> abrir o arquivo brModelo.jar. Caso não consiga abrir, baixe o Java pelo link: [https://www.java.com/pt-BR/download/ie\\_manual.jsp?locale=pt\\_BR](https://www.java.com/pt-BR/download/ie_manual.jsp?locale=pt_BR)

Se tiver dúvida, este link pode ajudar:

<https://www.youtube.com/watch?v=Z8PA0MJAuqU>

# MER

- ❑ Empresa de seguros de automóveis: Cada **cliente** possui **CPF**, **nome**, **sexo**, **endereço** e **telefones de contato (celular e fixo)**. Os **carros** possuem uma **placa**, **marca**, **modelo**, **ano**, **chassi** e **cor**. Cada carro tem **um** número de **sinistros** de acidentes associados a ele, sabendo que pode ter ocorrido **múltiplos acidentes ou nenhum**. Já os sinistros devem ser identificados por um **código único** e **data e hora de ocorrência**, **local de ocorrência**, **condutor** (que pode ou não ser o titular da apólice). Um cliente pode possuir **várias** **apólices** (**mínimo uma**) vigentes ou não, e cada apólice de seguro tem um **identificador único** e só pertence a **um** cliente e **somente um** carro, e tem **data de início** e **fim da vigência**, **valor total assegurado**, **valor franquia** associados a ela. É importante saber que o carro pode ter **várias** apólices vinculadas a ele, mas apenas uma vigente.

# Conclusão

- ❑ Praticamos a criação do modelo lógico no paradigma relacional, para o MER de seguro de carros, iniciado no Modelo Conceitual.
- ❑ Praticamos a criação do modelo lógico no BrModelo.

# Próxima aula

- ✓ Estudaremos sobre SGBDs e Banco de Dados Relacionais

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 10 - Normalização de Tabelas

Prof. Jéssica Coelho

# Nesta aula

- Vamos trabalhar o conceito de Normalização

# Normalização

- ❑ É o processo de organizar os dados em um banco de dados. Isso inclui regras para criação de tabelas e seus relacionamentos, a fim de proteger os dados e tornar o banco de dados mais flexível, eliminando a redundância e a dependência inconsistente.



# Normalização

- Há seis (6) formas normais mais difundidas. Para esta apresentação, vamos utilizar as três (3) mais famosas. Vamos usar a tabela abaixo para exemplificar a sua utilização.

COD CLIENTE	NOME CLIENTE	TEL 1	TEL 2	ENDEREÇO	COD PRODUTO	NOME PRODUTO	COD FABRICANTE	NOME FABRICANTE	PREÇO	QUANTIDADE
1	Victor Sales	999997777	988556633	RUA A	3000	PRODUTO 1	300	FABRICANTE 1	50	2
1	Victor Sales	999997777	988556633	RUA A	3411	PRODUTO 2	400	FABRICANTE 2	150	3
1	Victor Sales	999997777	988556633	RUA A	8000	PRODUTO 3	300	FABRICANTE 1	40	7
2	João Batista	963665511	955458241	RUA B	3411	PRODUTO 2	400	FABRICANTE 2	150	5
3	Márcio Campos	987445566		RUA C	4522	PRODUTO 4	500	FABRICANTE 3	70	2
1	Victor Sales	999997777	988556633	RUA A	3000	PRODUTO 1	300	FABRICANTE 1	50	4

# Normalização

- ❑ 1FN: Atomicidade: Todos os atributos de uma tabela devem ser atômicos, ou seja, a tabela não pode conter grupos ou atributos repetidos com mais de um valor.

COD CLIENTE	NOME CLIENTE	TEL 1	TEL 2	ENDEREÇO
1	Victor Sales	999997777	988556633	RUA A
2	João Batista	963665511	955458241	RUA B
3	Márcio Campos	987445566		RUA C



COD CLIENTE	NOME CLIENTE	ENDEREÇO
1	Victor Sales	RUA A
2	João Batista	RUA B
3	Márcio Campos	RUA C

COD CLIENTE	TELEFONE
1	999997777
1	988556633
2	963665511
2	955458241
3	987445566

# Normalização

- ❑ 2FN: Para fazer a 2FN, é necessário aplicar a 1FN. Todos os atributos não chaves da tabela devem depender unicamente da chave primária (não podendo depender apenas de parte dela).

COD CLIENTE	COD PRODUTO	NOME PRODUTO	COD FABRICANTE	NOME FABRICANTE	PREÇO	QUANTIDADE
1	3000	PRODUTO 1	300	FABRICANTE 1	50	2
1	3411	PRODUTO 2	400	FABRICANTE 2	150	3
1	8000	PRODUTO 3	300	FABRICANTE 1	40	7
2	3411	PRODUTO 2	200	FABRICANTE 3	150	5
3	4522	PRODUTO 4	500	FABRICANTE 4	70	2
1	3000	PRODUTO 1	300	FABRICANTE 1	50	4



COD PRODUTO	NOME PRODUTO	COD FABRICANTE	NOME FABRICANTE	PREÇO
3000	PRODUTO 1	300	FABRICANTE 1	50
3411	PRODUTO 2	400	FABRICANTE 2	150
8000	PRODUTO 3	300	FABRICANTE 1	40
4522	PRODUTO 4	500	FABRICANTE 4	70

COD CLIENTE	COD PRODUTO	QUANTIDADE
1	3000	2
1	3411	3
1	8000	7
2	3411	5
3	4522	2
1	3000	4

# Normalização

- ❑ 3FN: Para fazer a 3FN, é necessário aplicar a 2FN. Os atributos não chave de uma tabela devem ser mutuamente independentes e dependentes unicamente e exclusivamente da chave primária.

COD PRODUTO	NOME PRODUTO	COD FABRICANTE	NOME FABRICANTE	PREÇO
3000	PRODUTO 1	300	FABRICANTE 1	50
3411	PRODUTO 2	400	FABRICANTE 2	150
8000	PRODUTO 3	300	FABRICANTE 1	40
4522	PRODUTO 4	500	FABRICANTE 4	70



COD PRODUTO	NOME PRODUTO	COD FABRICANTE	PREÇO
3000	PRODUTO 1	300	50
3411	PRODUTO 2	400	150
8000	PRODUTO 3	300	40
4522	PRODUTO 4	500	70

COD FABRICANTE	NOME FABRICANTE
300	FABRICANTE 1
400	FABRICANTE 2
300	FABRICANTE 1
500	FABRICANTE 4

# Conclusão

- ❑ Aprendemos sobre normalização

# Próxima aula

- ✓ Estudaremos sobre SGBDs e Banco de Dados Relacionais

# Fundamentos em Engenharia de Dados

Capítulo 2 - Aula 11 - Banco de Dados e SGBD - Prática Mysql

Prof. Jéssica Coelho

# Nesta aula

- ❑ Instalar o MySQL e o MySQL Workbench em Windows;



# Conclusão

- ❑ Instalamos o MySQL e o MySQL Workbench em Windows

# Na próxima aula

- ❑ Instalaremos o MSSQL e o SSMS em Windows

# Fundamentos em Engenharia de Dados

Capítulo 2 - Aula 11 - Banco de Dados e SGBD

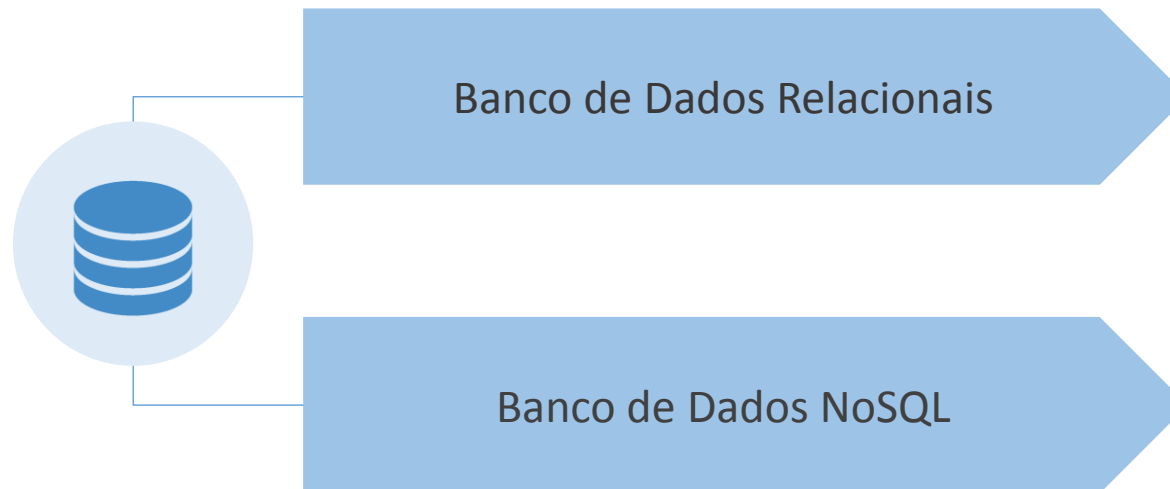
Prof. Jéssica Coelho

# Nesta aula

- ❑ Conhecer as principais características de um sistema gerenciador de banco de dados do paradigma relacional.

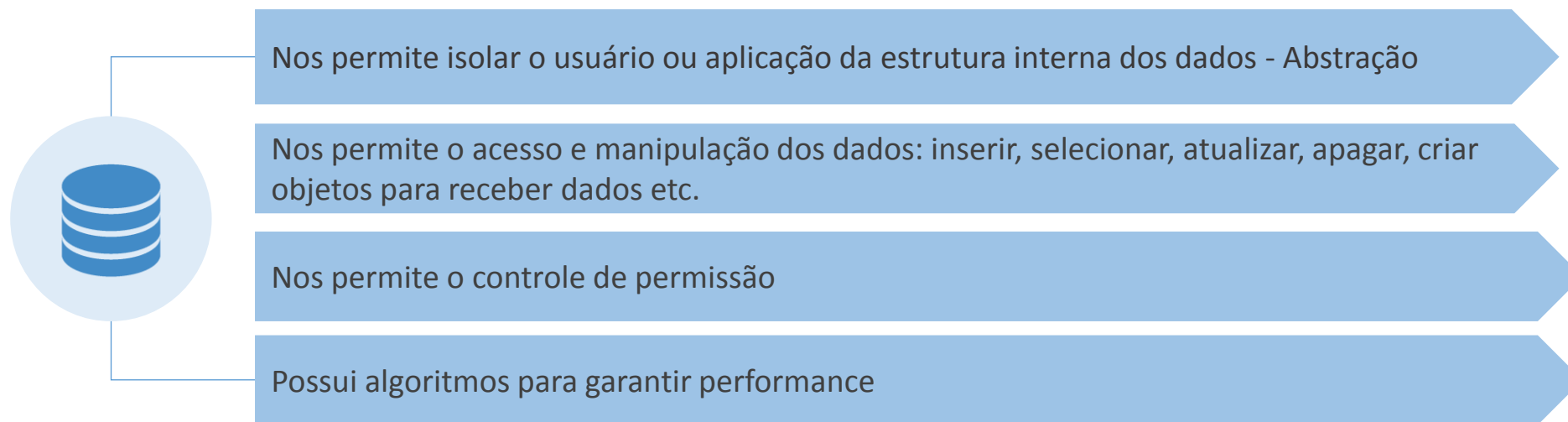
# Banco de Dados

Coleção de dados inter-relacionados, representando informações sobre um domínio específico, ou seja, sempre que for possível agrupar dados que se relacionam e tratam de um mesmo assunto posso dizer que tenho um banco de dados;



# SGBDs

São responsáveis por gerenciar múltiplos bancos de dados, acessos simultâneos, abstrair o acesso a estes dados e permitir o controle de permissão. Basicamente podemos dizer que o sistema gerenciador de banco de dados:



# SGBDs Relacionais

- ✓ Armazena dados estruturados;
- ✓ Implementam o paradigma relacional e seus elementos: tabelas, colunas, linhas e relacionamentos;
- ✓ Utiliza linguagem SQL para manipulação de dados. Structure Query Language;
- ✓ Além das tabelas possui procedures, views, triggers, funções, índices e outros;
- ✓ Implementam o conceito de ACID
  - ❑ Atomicidade;
  - ❑ Consistência;
  - ❑ Isolamento;
  - ❑ Durabilidade

# SGBDs Relacionais





## Exemplos de SGBDs Relacionais do Mercado



ORACLE®



Microsoft  
SQL Server



IBM® DB2.



PostgreSQL



SQLite



MySQL®

# Conclusão

- ❑ Conhecemos o que é banco de dados e seus principais tipos;
- ❑ Conhecemos o que é SGBD e estudamos os SGBDs Relacionais;
- ❑ Conhecemos o princípio do ACID implementado por SGBDs Relacionais;
- ❑ Conhecemos os principais SGBDs relacionais do mercado.

# Na próxima aula

- ❑ Entender o que é o modelo físico e seus aspectos principais

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 12 - Modelo Físico

Prof. Jéssica Coelho

# Nesta aula

- ❑ Estudaremos o modelo físico e seus principais elementos;

# Modelo físico

Nessa fase definimos o SGBD que será usado e suas restrições

É o banco de dados em si

Detalha os componentes da estrutura física do banco, no modelo relacional: tabelas, colunas, linhas, índices, views...

# Modelo físico

## Modelo hierárquico

- IMS e Adabas (IBM)
- DMSII (Unisys)

## Modelo relacional

- Oracle
- SQL Server
- DB2
- MySQL

## Modelo orientado a objetos

- Caché
- Jasmine.

# Modelo Físico Relacional

- ❑ Em geral é descrito por um script SQL (Structure Query Language);
- ❑ Define objetos como:
  - ❑ Tabelas, índices, procedures, visões, funções...
  - ❑ Relacionamentos através das chaves;
  - ❑ Restrições ou Constraints;
  - ❑ Colunas e seus tipos;
- ❑ Incorpora as especificidades de cada SGBD:
  - ❑ Autoincremento: Não existe no Oracle, é feito via Sequence, por exemplo;
  - ❑ Tipos de dados podem variar;



# Conclusão

- ❑ Estudamos o modelo físico e seus aspectos;
- ❑ Entendemos que existem diferenças entre os SGBDs que precisam ser consideradas para a criação dos scripts;

# Na próxima aula

- ❑ Conheceremos a ferramenta para criação dos scripts do modelo físico;

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 13 - Modelo Físico Prática MSSQL

Prof. Jéssica Coelho

# Nesta aula

- ❑ Construiremos nosso modelo físico no MySQL

# Conclusão

❑ Instalamos o MSSQL e o SSMS em Windows

# Na próxima aula

- ❑ Criar juntos um modelo relacional para praticar.

# Fundamentos em Engenharia de Dados

Capítulo 2 Aula 13 - Modelo Físico Prática MSSQL

Prof. Jéssica Coelho

# Nesta aula

- ❑ Instalaremos o MSSQL e o SSMS em Windows



# Conclusão

❑ Instalamos o MSSQL e o SSMS em Windows

# Na próxima aula

- ❑ Criar juntos um modelo relacional para praticar.

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 1 - Linguagem SQL - Introdução

Prof. Jéssica Coelho

# Nesta aula

- Apresentar a linguagem de consulta estruturada usada na criação de bancos de dados, seus objetos e na manipulação dos dados.

# SQL

SQL é uma linguagem padrão para trabalhar com bancos de dados relacionais.

A sigla significa Structure Query Language ou Linguagem de Consulta Estruturada

Linguagem comercial usada nos SGBDs Relacionais para manipular, esquemas de banco de dados, objetos e os dados em si

Possui o padrão ANSI e ISO

Baseado na álgebra relacional

# Grupos de SQL

DDL

- **Linguagem de Definição de Dados:**  
É um conjunto de instruções usado para criar e modificar estruturas/esquemas do banco relacional

DML

- **Linguagem de manipulação dos dados:**  
É um conjunto de instruções usado para modificar os dados armazenados nas tabelas;

DQL

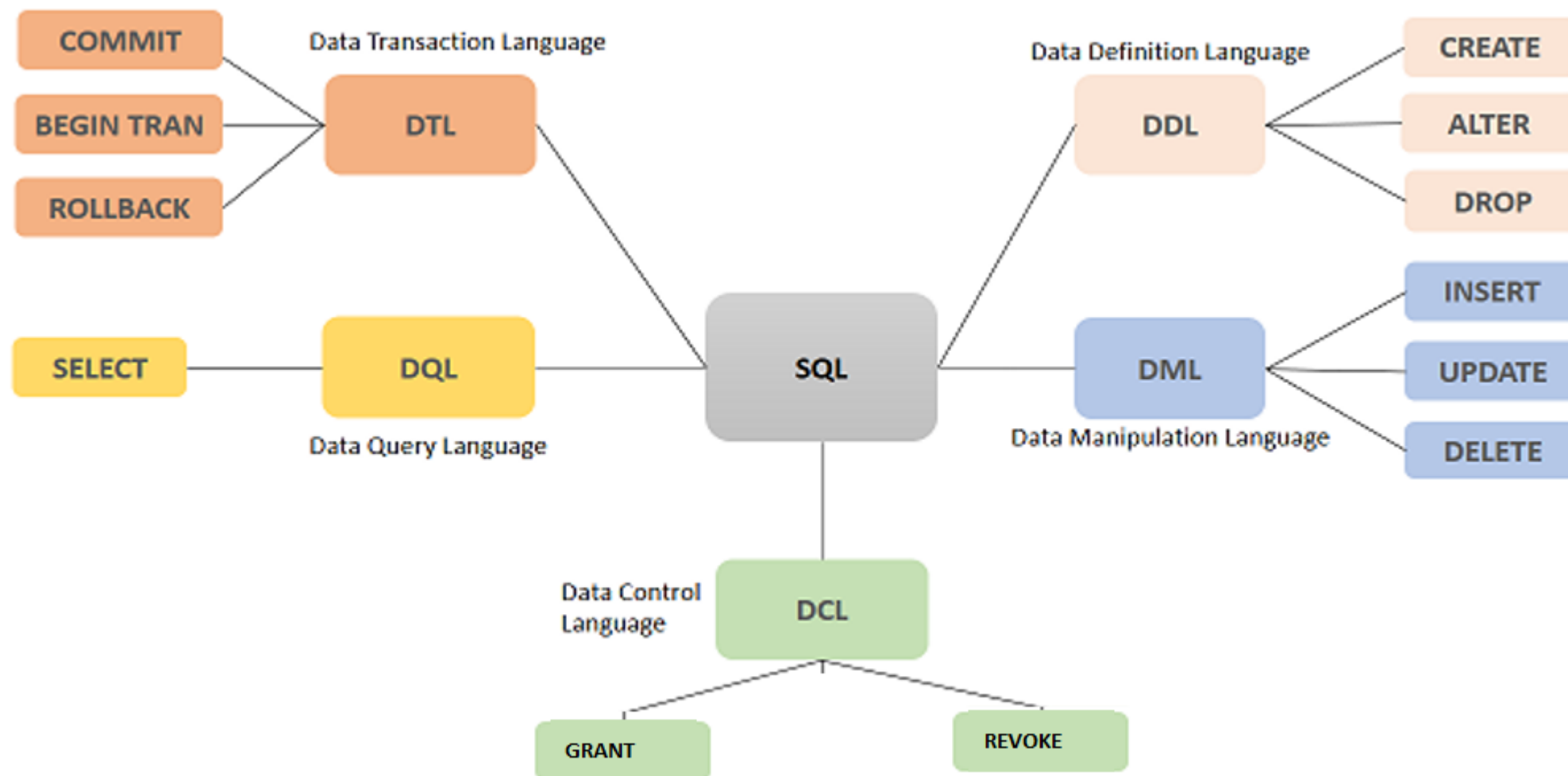
- **Linguagem de consulta (query) dos dados:**  
É um conjunto de instruções usado para consultar dados nas estruturas dos objetos do banco de dados

DCL

- **Linguagem de controle dos dados:**  
É um conjunto de instruções usado para controlar as permissões nos objetos do banco de dados

DTL

- **Linguagem de controle de transações**  
É um conjunto de instruções usado para controle transações.



# T/SQL x PL/SQL

- São os principais métodos proprietários que definem algumas especificidades de sintaxe SQL. Sendo:
- T/SQL - Transaction SQL: Extensão do SQL lançada pela Microsoft para ser usada pelo SQL Server;
  - PL/SQL - Procedural Language SQL é a extensão da linguagem SQL lançada pela SUN para ser usada pelo Oracle. É a linguagem que tem como principal característica trabalhar com seus comandos como blocos.



# Conclusão

- ❑ Visão geral da linguagem SQL.

# Na próxima aula

- ❑ Conheceremos a linguagem DDL, de definição de dados;

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 2 - Linguagem SQL - DDL

Prof. Jéssica Coelho

# Nesta aula

- ❑ Conheceremos a linguagem DDL (definição de dados);

# Linguagem de definição de dados

Subconjunto de comandos para criação e manutenção de objetos de banco de dados.

*Sempre consulte a sintaxe do SGBD na documentação*

DATABASE, TABLE, INDEX, CONSTRAINT (PK, FK, UK), ROLE, USER, PROCEDURE, FUNCTION, TRIGGER, VIEW E OUTROS

CREATE  
objeto:

Criar um objeto  
(tabela, por  
exemplo) no  
banco de dados

ALTER  
objeto:

Altera a estrutura  
ou a configuração  
de um objeto no  
banco de dados

DROP  
objeto

Exclui um objeto  
do banco de  
dados

# Objetos de banco de dados relacional

- ❑ View: objeto criado a partir de um ou mais selects. Encapsula códigos complexos. O select que originou a view é executado sempre que esta for chamada;
- ❑ Índice: É uma referência associada a uma chave que é utilizada para fins de otimização
- ❑ Procedure (procedimento armazenado): Objeto que possui um grupo de comandos de tarefas repetitivas, que aceita parâmetros de entrada para que a tarefa seja efetuada de acordo com a necessidade individual de maneira mais otimizada pois é pré-compilada.
- ❑ Function: similar à Procedure, mas com a diferença que retorna sempre um valor ou tabela.

# Objetos de banco de dados relacional

- ❑ Trigger ou Gatilho: Uma Trigger é um procedimento armazenado no banco de dados que é chamado automaticamente sempre que ocorre um evento especial no banco de dados;
- ❑ Usuários: São aqueles que interagem com os bancos de dados dentro de um Sistema Gerenciador de Banco de Dados e para os quais concedemos ou retiramos permissões;
- ❑ Roles: Uma Role ou papel é um agrupamento de permissões que pode ser concedida a usuários ou outras roles;

# Criar (CREATE)

## ❑ CRIAR - CREATE

- `CREATE DATABASE nome_do_banco_de_dados;`
- `CREATE TABLE nome_da_tabela (coluna1 tipo, coluna2 tipo, coluna3 tipo...) [lista_de_opções_para_criação];`
- `CREATE VIEW nome_da_view AS comando_SELECT_que_vai_gerar_a_view`
- `CREATE PROCEDURE nome (lista opcional de parâmetros) AS ;`
- `CREATE FUNCTION nome (lista opcional de parâmetros) AS RETURNS`



# CREATE TABLE

## Restrições de colunas

- NOT NULL
- DEFAULT *valor*
- CHECK(*condição*)

## Restrições de tabela

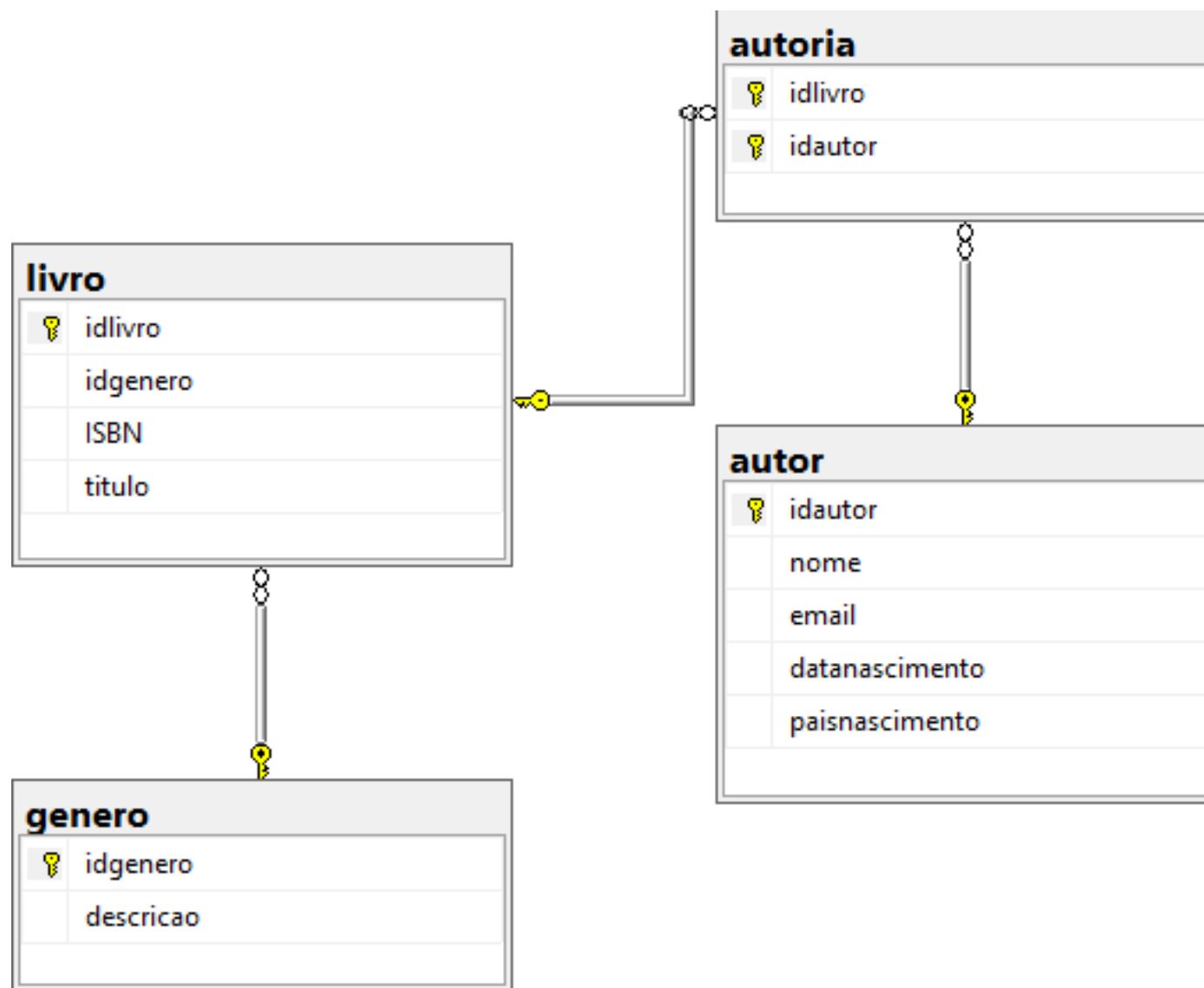
- PRIMARY KEY
- UNIQUE
- FOREIGN KEY

**Convenção de leitura:**

- | ou
- [] opcional
- [...n] itens pode ser repetido n vezes, separando por vírgula
- {} obrigatório

```
CREATE TABLE tabela (  
    atrib1 tipo [(tamanho)] [NOT NULL | DEFAULT valor]  
        [CHECK (condição)],  
    atrib2 tipo [(tamanho)] [NOT NULL | DEFAULT valor]  
        [CHECK (condição)],  
    ...  
    [CONSTRAINT nome da restrição] PRIMARY KEY (<atributos chave primária>),  
    [CONSTRAINT nome da restrição] UNIQUE (< atributos chave candidata>),  
    [CONSTRAINT nome da restrição] FOREIGN KEY (<atributos chave estrangeira>)  
        REFERENCES tabelaRef [(<chave primária>)]  
        [ON DELETE CASCADE | SET NULL | SET DEFAULT]  
        [ON UPDATE CASCADE | SET NULL | SET DEFAULT],  
    [CONSTRAINT nome da restrição] CHECK (condição)  
);
```

Considerando este modelo:



# Exemplos CREATE - SQL Server

```
CREATE DATABASE [BibliotecaIGTI]
```

```
CREATE TABLE autor  
(  
    idautor int NOT NULL IDENTITY (1,1),  
    nome varchar(50) NOT NULL,  
    email varchar(50) NULL,  
    datanascimento date NULL,  
    paismascimento varchar(50) null,  
    PRIMARY KEY (idautor)  
);
```

```
CREATE TABLE genero  
(  
    idgenero INT NOT NULL IDENTITY(1,1),  
    descricao varchar(100) NOT NULL,  
    PRIMARY KEY (idgenero)  
);
```

# Exemplos CREATE - SQL Server

```
CREATE TABLE livro
(
  idlivro int NOT NULL IDENTITY (1,1),
  idgenero int NOT NULL,
  ISBN VARCHAR(50) NOT NULL,
  titulo varchar(200) NOT NULL,
  PRIMARY KEY (idlivro),
  CONSTRAINT fk_livro_genero FOREIGN KEY (idgenero) REFERENCES genero (idgenero)
);
```

```
CREATE TABLE autoria
(
  idlivro INT NOT NULL,
  idautor INT NOT NULL,
  PRIMARY KEY (idlivro, idautor),
  CONSTRAINT fk_autoria_livro FOREIGN KEY (idlivro) REFERENCES livro (idlivro),
  CONSTRAINT fk_autoria_autor FOREIGN KEY (idautor) REFERENCES autor (idautor)
)
```

# Trabalhando com índices:

For SQL SERVER

❑ CREATE INDEX <nome do índice> on tabela <coluna(s)> [INCLUDE <colunas>]

```
CREATE INDEX IX_Livro ON Livro (Titulo)
CREATE INDEX IX_Livro_ordenado ON Livro (Titulo DESC)
```

# ALTER

- ❑ ALTER qual\_objeto\_deseja\_alterar (DATABASE, TABLE...) opção\_de\_alteração;
- ❑ MYSQL e Oracle usam esse formato: MODIFY COLUMN nome\_coluna tipo [opções de alteração]

```
ALTER TABLE AUTOR DROP COLUMN EMAIL
```

```
ALTER TABLE LIVRO ADD resumo VARCHAR(100)
```

```
ALTER TABLE LIVRO ADD CONSTRAINT NaoVazio CHECK (resumo <> '')
```

# CREATE VIEWS

❑ CREATE VIEW:

```
CREATE VIEW livroromance AS  
(  
SELECT * FROM genero  
WHERE descricao = 'romance'  
)
```



# TRUNCATE TABLE

- ❑ Exclui todas as linhas de uma tabela de base de dados. Não precisa de Commit.

idautor	nome	datanascimento	paishnascimento
2	Jessica,jessica@yahoo.com	1990-05-07	Brasil

TRUNCATE TABLE autor

idautor	nome	datanascimento	paishnascimento
---------	------	----------------	-----------------

- ❑ No SQL Server não pode ser usado se a tabela for referenciada por outra via FK;

Msg 4712, Level 16, State 1, Line 13

Cannot truncate table 'autor' because it is being referenced by a FOREIGN KEY constraint.

# Deletar objetos DROP

## ❑ DELETAR

- DROP DATABASE nome\_do\_banco\_de\_dados;
- DROP PROCEDURE nome\_da\_procedure;
- DROP FUNCTION nome\_da\_procedure;
- DROP VIEW nome\_da\_view;
- DROP INDEX nome\_da\_tabela.nome\_do\_indice; (MSSQL)
- DROP INDEX nome\_do\_indice (ORACLE);
- ALTER TABLE nome\_da\_tabela DROP INDEX nome\_do\_indice (MYSQL)

# Conclusão

- ❑ Conhecemos a linguagem de definição de dados SQL e seus comandos;
- ❑ Aprendemos sobre os objetos de banco de dados;
- ❑ Reforçamos que cada SGBD possui sua própria sintaxe e que o manual deve ser consultado.

# Próxima aula

- ✓ Vamos Praticar DDL (CREATE, ALTER e DROP) no SQL Server Management Studio;

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 2 - Linguagem SQL - DDL

Prof. Jéssica Coelho

# Nesta aula

- Praticaremos os comentários DDL estudados na aula anterior

# Conclusão

☐ Praticamos os comandos DDL

# Próxima aula

- ✓ Estudaremos os comandos DML



# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 4 - Linguagem SQL - DML

Prof. Jéssica Coelho

# Nesta aula

- ❑ Conheceremos a linguagem DML (manipulação de dados);

# Linguagem de manipulação de dados

Subconjunto de comandos que permite aos usuários manipular dados, ou seja, incluir dados em uma tabela, alterar ou excluir.

*Sempre consulte a  
sintaxe do SGBD na  
documentação*

INSERT, UPDATE, DELETE

INSERT

Insere linhas em  
uma tabela

UPDATE

Atualiza linhas de  
uma tabela

DELETE

Deleta linhas em  
uma tabela

# Linguagem de manipulação de dados

## ❑ Inclusão de linha:

```
INSERT INTO nome_tabela[(lista_atributos)] VALUES(lista_valores_atributos) [,  
(lista_valores_atributos)];
```

## ❑ Alteração de dados:

```
UPDATE nome_tabela SET nome_atributo_1 = valor [{,nome_atributo_n = valor}] [WHERE  
condição];
```

## ❑ Exclusão

```
DELETE FROM nome_tabela [WHERE condição];
```

# Conclusão

- ❑ Conhecemos o grupo de comandos DML.

# Na próxima aula

- ❑ Vamos praticar juntos no SQL Server Management Studio

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 5 - Linguagem SQL - DML - Prática

Prof. Jéssica Coelho

# Nesta aula

- Praticaremos os comandos DML



# Conclusão

☐ Praticamos os comandos DML.

# Na próxima aula

- ❑ Estudaremos o grupo de comandos do DCL

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 6 - Linguagem SQL - DCL

Prof. Jéssica Coelho

# Nesta aula

- ❑ Conheceremos a linguagem DCL (controle de acesso aos dados).

# Linguagem de controle de dados

Subconjunto de comandos que permite aos usuários controlar as permissões de acesso aos objetos de banco de dados

*Sempre consulte a  
sintaxe do SGBD na  
documentação*

GRANT, REVOKE

GRANT

Conceder  
permissão a  
objetos de banco

REVOKE

Remover  
permissão a  
objetos de banco

# Conceitos

- ❑ USUÁRIO: Aqueles que terão acesso ao banco de dados para realizar as tarefas. Pode ser uma aplicação, usuário final ou grupo de usuários;
- ❑ PERMISSÃO: Permissão é o que o usuário ou role podem executar no banco de dados.
  - ❑ Nível de banco, server ou objeto;
- ❑ ROLE: Uma role ou papel é um agrupamento de permissões que pode ser concedida a usuários ou outras roles e serve para facilitar a administração e manutenção das permissões

# Definindo permissão

## ❑ GRANT

```
GRANT <lista de privilegios> ON <objeto>  
TO <lista de usuários/roles>  
[WITH GRANT OPTION] [GRANTED BY grantor];
```

## ❑ REVOKE

- REVOKE: revoga privilégios para usuário/role.

```
REVOKE [GRANT OPTION FOR]  
<lista de privilegios> ON objeto  
FROM <usuários/roles> [RESTRICT|CASCADE] ;
```

# Algumas permissões

□ Algumas “funções” para os quais podemos conceder permissão:

- INSERT, SELECT, UPDATE, DELETE (relacionada à DML)
- REFERENCES: permissão do uso da integridade referencial.
- EXECUTE: Permissão para executar funções e procedimentos.
- USAGE: Permissão para domínios, conjuntos de caracteres, collations e translations.
- UNDER: Quando tratamos tipos definidos pelo usuário.
- TRIGGER: autorização para executar uma trigger.



# Exemplos

GRANT SELECT ON aluno to 'test\_user' (MYSQL e MSSQL)

GRANT INSERT, UPDATE, DELETE on aluno to test\_user (MYSQL e MSSQL)

GRANT SELECT on schema :: [testepermissao] to [test\_user] (MSSQL)

GRANT SELECT ON testpermissao.\* to 'test\_user'; (MYSQL)

ALTER ROLE [db\_datareader] ADD MEMBER [test\_user] (MSSQL) - ROLE

GRANT 'developer' TO 'test\_user' -ROLE

# Exemplos

## ❑ MYSQL

REVOKE DELETE FROM 'roletest'

REVOKE read from 'test\_user'

REVOKE INSERT, UPDATE, DELETE on exemplopermissao.\* FROM 'test\_user'

## ❑ MSSQL:

REVOKE DELETE FROM test\_user

REVOKE select from test\_user

REVOKE INSERT, UPDATE, DELETE on LIVRO FROM test\_user

# Conclusão

- ❑ Conhecemos o grupo de comandos DCL;
- ❑ E entendemos que existem diferentes sintaxes.

# Na próxima aula

- ❑ Conhecer o próximo grupo de comandos SQL, o DQL.

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 7 - Linguagem SQL - DQL

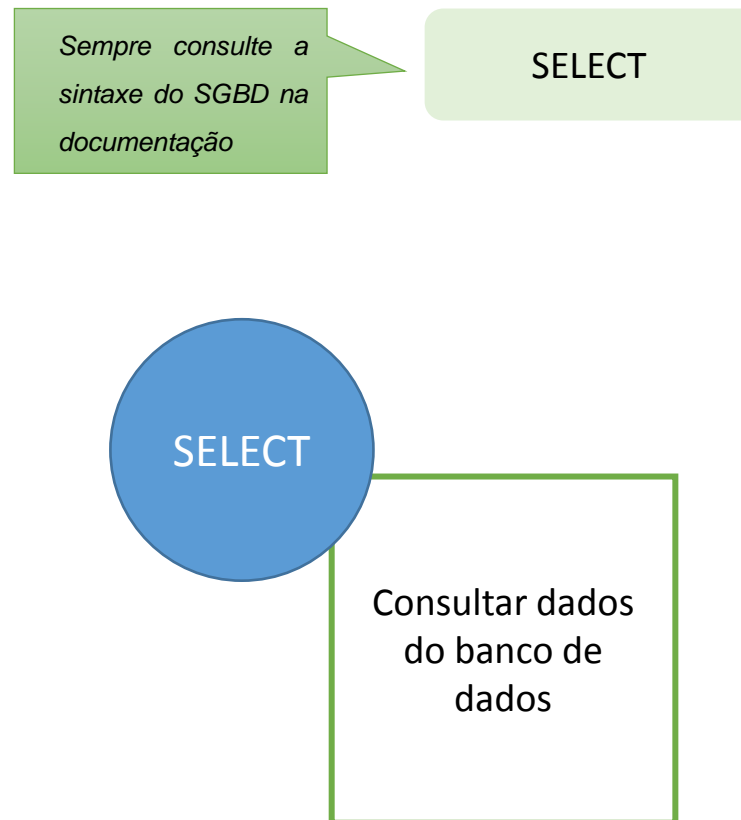
Prof. Jéssica Coelho

# Nesta aula

- ❑ Conhecer a Linguagem de Consulta de Dados.

# Linguagem de consulta de dados

- ❑ A Linguagem de Consulta de Dados ou Data Query Language (DQL) é a sublinguagem do SQL responsável por realizar consultas (leitura) aos dados



# SINTAXE PADRÃO

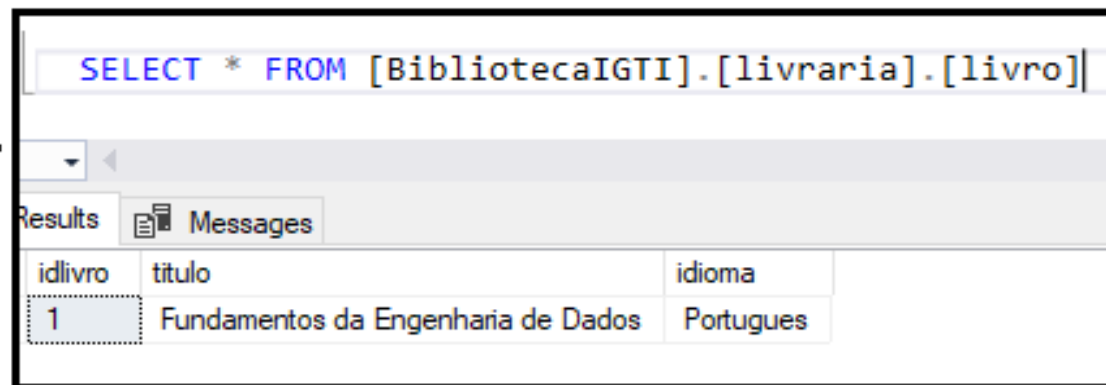
❑ Desejo SELECIONAR as seguintes colunas DA(S) TABELA(S) X [ONDE O VALOR SEJA...].

```
SELECT [predicado { * | tabela.* | [tabela.]campo1 [AS alias1] [, [tabela.]campo2  
[AS alias2] [, ...] }]  
FROM expressão tabela [, ...]  
[WHERE... ]  
[GROUP BY... ]  
[HAVING... ]  
[ORDER BY... ]
```



- ❑ Todas as colunas versus colunas específicas:

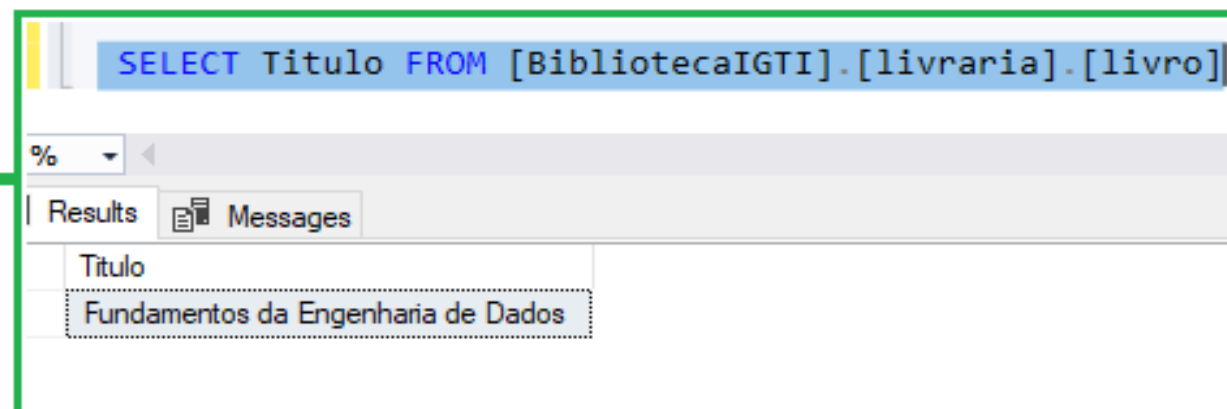
Retornar todas as colunas (\*) de uma tabela



```
SELECT * FROM [BibliotecaIGTI].[livraria].[livro]
```

idlivro	titulo	idioma
1	Fundamentos da Engenharia de Dados	Portugues

Retornar a coluna "Titulo" de uma tabela



```
SELECT Titulo FROM [BibliotecaIGTI].[livraria].[livro]
```

Titulo
Fundamentos da Engenharia de Dados

## ❑ Valores distintos - DISTINCT:

Ao adicionarmos  
**DISTINCT \***  
retornamos as linhas  
que possuem valores  
diferentes para todas  
as colunas.

Caso quiséssemos  
retornar valores  
**DISTINCTS** para a  
coluna **Titulo** bastaria  
ao invés do \* adicionar  
**SELECT DISTINCT Titulo**  
**FROM...**

```
SELECT DISTINCT * FROM Livros
```

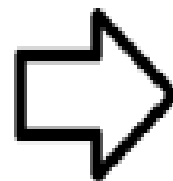
Results		Messages
Titulo	Idioma	
Fundamentos da Engenharia de Dados	Portugues	

## ❑ Valores ordenados - ORDER BY:

Listar os nomes distintos das editoras por ordem alfabética ascendente:

```
SELECT DISTINCT nome_editora  
FROM editora  
ORDER BY nome_editora;
```

```
SELECT DISTINCT nome_editora  
FROM editora  
ORDER BY nome_editora ASC;
```



nome_editora
Casa dos Espiritos
Editora Lê
Manole
Moderna
Objetiva

## ❑ Valores ordenados - ORDER BY:

Listar os nomes distintos das gênero por ordem alfabética ascendente e descendente:

```
SELECT DISTINCT descricao  
FROM genero  
ORDER BY descricao ASC;
```

descricao
Adulto
Auto-Ajuda
Autobiografia
Aventura
Biografia
Científico
Conto
Crônica
Ensaio
Fábula
Fantasia

```
SELECT DISTINCT descricao  
FROM genero  
ORDER BY descricao DESC;
```

descricao
Suspense
Sátira
Romance
Poesia
Novela
Infanto-juvenil
Infantil
Ficção
Fantasia
Fábula
Ensaio

## ❑ Operadores aritméticos:

+	Adição
-	Subtração
*	Multiplicação
/	Divisão

```
SELECT titulo_livro, preco, preco + 10 AS adição, preco - 10 AS subtração, preco * 0.1 AS multiplicação, preco / 10 AS divisão
FROM livro;
```

titulo_livro	preco	adição	subtração	multiplicação	divisão
Pelas Ruas de Calcutá	36.1	46.099998474121094	26.099998474121094	3.6099998474121096	3.6099998474121096
Devoted - Devoção	27.2	37.20000076293945	17.200000762939453	2.7200000762939456	2.720000076293945
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7	42.70000076293945	22.700000762939453	3.2700000762939454	3.2700000762939454
P.s. - Eu Te Amo	23.5	33.5	13.5	2.35	2.35
O Que Esperar Quando Você Está Esperando	37.8	47.79999923706055	27.799999237060547	3.779999923706055	3.779999923706055
As Melhores Frases Em Veja	23.9	33.89999961853027	13.899999618530273	2.3899999618530274	2.3899999618530274
Bichos Monstruosos	24.9	34.89999961853027	14.899999618530273	2.4899999618530275	2.4899999618530275
Casas Mal Assombradas	27.9	37.89999961853027	17.899999618530273	2.7899999618530273	2.7899999618530273
Memórias Póstumas de Brás Cubas	22.5	32.5	12.5	2.25	2.25
Dom Casmurro	25.9	35.89999961853027	15.899999618530273	2.5899999618530276	2.589999961853027
Dom Casmurro	35.9	45.900001525878906	25.900001525878906	3.5900001525878906	3.5900001525878906
Quincas Borba	35.9	45.900001525878906	25.900001525878906	3.5900001525878906	3.5900001525878906

# Conclusão

- ❑ Conhecemos o comando parte do grupo DQL;
- ❑ Aprendemos a sintaxe básica do SELECT;
- ❑ Aprendemos a remover linhas duplicadas e a ordenar o resultado em ordem crescente e decrescente.

# Na próxima aula

- Aprenderemos sobre os filtros.

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 8 - Linguagem SQL - DQL - Parte 2

Prof. Jéssica Coelho



# Nesta aula

- Aprenderemos sobre os filtros.

# Filtros - Cláusula WHERE

Um usuário geralmente está procurando por informações específicas no Banco de Dados. Deste modo, há a utilização do comando WHERE (ONDE), juntamente com alguns comandos específicos para só retornar ou projetar os resultados desejados.

❑ Filtros de seleção (condição):

```
SELECT column1, column2, ...  
FROM table_name  
WHERE condition;
```



Satisfaz a condição

```
SELECT column1, column2, ...  
FROM table_name  
WHERE NOT condition;
```



Não satisfaz a condição

# Filtros - Cláusula WHERE

❑ Operadores condicionais:

Condições	
=	igual
<	menor
<=	menor ou igual
>	maior
>=	maior ou igual
<>	diferente
AND	todas as condições verdadeiras
OR	pelo menos uma condição verdadeira
NOT	não satisfaz a condição
BETWEEN ... AND	dentro de um intervalo
LIKE	pesquisar por padrão
IN	possíveis valores

# Filtros - Cláusula WHERE

Considere essa tabela:

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
111	Caio	Rua 1	Física	9
222	Fernando	Rua 2	Matemática	10
333	João	Rua 3	Português	3
333	João	Rua 3	História	8
444	César	Rua 4	Física	8

Tabela Principal

# Filtros - Cláusula WHERE

Retornar todos os alunos que estudam Português:

```
SELECT *  
FROM Aluno  
WHERE disciplina = 'Português'
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
333	João	Rua 3	Português	3

# Filtros - Cláusula WHERE

Retornar todos os alunos que não possuem nota 8 - NOT:

```
SELECT *  
FROM Aluno  
WHERE NOT nota=8
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
111	Caio	Rua 1	Física	9
222	Fernando	Rua 2	Matemática	10
333	João	Rua 3	Português	3

Comando NOT

# Filtros - Cláusula WHERE

❑ Filtro de Nulo:

```
SELECT column_names  
FROM table_name  
WHERE column_name IS NULL;
```

```
SELECT column_names  
FROM table_name  
WHERE column_name IS NOT NULL;
```

# Filtros - Cláusula WHERE

## ❑ Combinando filtros AND/OR

```
SELECT column1, column2, ...  
FROM table_name  
WHERE condition1 AND condition2 AND condition3 ...;
```

```
SELECT column1, column2, ...  
FROM table_name  
WHERE condition1 OR condition2 OR condition3 ...;
```



# Filtros - Cláusula WHERE

Retornar linhas onde o nome é Caio e a disciplina estudada é Português:

```
SELECT *  
FROM Aluno  
WHERE nome_aluno='Caio' AND disciplina='Português'
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5

Comando AND

# Filtros - Cláusula WHERE

Retornar linhas onde o nome é Caio ou a disciplina estudada é Português:

```
SELECT *  
FROM Aluno  
WHERE nome_aluno='Caio' OR disciplina='Português'
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
111	Caio	Rua 1	Física	9
333	João	Rua 3	Português	3

Comando OR

# Filtros - Cláusula WHERE

❑ BETWEEN ... AND:

```
SELECT coluna1, coluna2, ...  
FROM nome_tabela  
WHERE coluna BETWEEN valor1 AND valor2
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
111	Caio	Rua 1	Física	9
333	João	Rua 3	Português	3

# Filtros - Cláusula WHERE

Retornar todos os alunos que possuem nota entre 2 e 6:

```
SELECT *
```

```
FROM Aluno
```

```
WHERE nota BETWEEN 2 AND 6
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
333	João	Rua 3	Português	3

Comando BETWEEN

## ❑ Operador Like para strings:

Ele é utilizado quando se quer encontrar registros específicos dentro de um campo texto. Podemos definir os caracteres que queremos, a posição em que ele deve aparecer e a quantidade de caracteres.

```
SELECT *  
FROM Aluno  
WHERE nome_aluno LIKE 'C%'
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
111	Caio	Rua 1	Física	9
444	César	Rua 4	Física	8

Comando LIKE %

```
SELECT *  
FROM Aluno  
WHERE nome_aluno LIKE 'C_ _ _'
```

Aluno				
matrícula	nome_aluno	endereço	disciplina	nota
111	Caio	Rua 1	Português	5
111	Caio	Rua 1	Física	9

Comando LIKE \_

A sintaxe utiliza “%” para informar se qualquer string por vir antes ou depois e o “\_” o total de caracteres. Caso a string filtrada atenda aos requisitos do filtro do LIKE, esta será retornada.

Comando	Retorna
LIKE 'A%'	Qualquer string que seja iniciada com a letra A.
LIKE '%A'	Qualquer string que seja iniciada com a letra A.
LIKE '%A%'	Qualquer string que tenha a letra A em qualquer posição.
LIKE 'A_'	String que possua dois caracteres, sendo a primeira letra A e a segunda qualquer outra.
LIKE '_A'	String que possua dois caracteres, sendo a última a letra A e a primeira qualquer outra.
LIKE '_A_'	String que possua exatamente três caracteres, em que o segundo seja obrigatoriamente a letra A.
LIKE '___'	Qualquer string com exatamente três caracteres.
LIKE '%A_'	Qualquer string que tenha a letra A na penúltima posição e a última seja qualquer outro caractere, possuindo qualquer tamanho.
LIKE '___%'	Qualquer string com pelo menos três caracteres.

# Conclusão

- ❑ Conhecemos mais aspectos e exemplos da linguagem de consulta de dados;
- ❑ Aprendemos sobre os filtros e operadores.

# Na próxima aula

- ❑ Trabalharemos com agrupamentos e agregações.



# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 8 - Linguagem SQL - DQL - Prática - Filtros

Prof. Jéssica Coelho

# Nesta aula

- Praticaremos os filtros - DQL

# Conclusão

- ❑ Praticamos os filtros do comando DQL

# Na próxima aula

- ❑ Trabalharemos com agrupamentos e agregações.

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 10 - Linguagem SQL - DQL - Parte 3

Prof. Jéssica Coelho

# Nesta aula

- ❑ Trabalharemos com agrupamentos e agregações.

# Linguagem de consulta de dados

❑ Função de agregação:

## Mínimo

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

## Máximo

```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

## Média

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

## Somatório

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

## Contagem/quantidade

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

# Linguagem de consulta de dados

Considerando a tabela abaixo:

idlivro	idgenero	ISBN	titulo	Idioma	resumo	Preco
3	1	4554654654	A menina que roubava livros	Portugues	Teste	35
4	3	658565465	Fundamentos em Engenharia de dados	Portugues	Teste	85
6	NULL	545454545	O segredo	Portugues	Test	45

Quantos livros eu tenho?

```
SELECT COUNT(*) AS QuantidadeLivros  
FROM Livro
```

Results	Messages
QuantidadeLivros	
3	

Qual a soma dos preços?

```
SELECT SUM(Preco) AS Somaprecos  
FROM Livro
```

Results	Messages
Somaprecos	
165	



# Linguagem de consulta de dados

Qual o maior e o menor preço?

```
SELECT MAX(Preco) AS MaiorPreco, MIN(Preco) AS MenorPreco  
FROM Livro
```

Results		Messages
MaiorPreco	MenorPreco	
85	35	

Qual a média de preços?

```
SELECT AVG(Preco) AS AvgPreco  
FROM Livro
```

Results		Messages
AvgPreco		
55		

# Linguagem de consulta de dados

- ❑ Agrupamentos e o comando HAVING:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s);
```

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition;
```

# Linguagem de consulta de dados

Considerando a tabela abaixo:



titulo_livro	preco
Pelas Ruas de Calcutá	36.1
Devoted - Devoção	27.2
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7
P.s. - Eu Te Amo	23.5
O Que Esperar Quando Você Está Esperando	37.8
As Melhores Frases Em Veja	23.9
Bichos Monstruosos	24.9
Casas Mal Assombradas	27.9
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	25.9
Dom Casmurro	35.9
Quincas Borba	35.9



AVG (preco) = média de preço;

(+) group by titulo\_livro = média de preço por título do livro;

(+) having avg(preco) < 30 = média de preço por título do livro, mas só retorne aqueles com média menor que 30

```
SELECT titulo_livro, avg(preco)
FROM livro
GROUP BY titulo_livro
HAVING avg(preco) < 30;
```

titulo_livro	avg(preco)
Devoted - Devoção	27.200000762939453
P.s. - Eu Te Amo	23.5
As Melhores Frases Em Veja	23.899999618530273
Bichos Monstruosos	24.899999618530273
Casas Mal Assombradas	27.899999618530273
Memórias Póstumas de Brás Cubas	22.5

# Linguagem de consulta de dados

Considerando a tabela abaixo:

**Tabela Cliente**

codigo	nome	endereco	bairro	cidade	estado	datanasc	debito	ativo
1	Alberto Carlos	Rua das Casas, 01	Centro	Mossoro	RN	1990-06-12	100	0
2	Breno Nunes	Rua da Lua, 02	Alto	Assu	RN	1975-08-10	NULL	0
3	Catarina Pontes	Rua do Sol, 04	Baixo	Apodi	RN	1987-06-15	0	0
4	Diego Alexandre	Rua da Estrela, 50	Orla	Natal	RN	1988-12-10	30	0
5	Ester Moreira	Av. do Trabalhador, 99	Cafu	Fortaleza	CE	1965-01-01	NULL	0
6	Fabricao Silva	Tv. Noite Escura, 99	Pelotas	Aracati	CE	1995-03-04	300	0
7	Gabriel Silva	Rua Projetada, SN	Centro	Piriqui	CE	1980-02-08	500	0
8	Helio Cunha	Rua de Cima, 875	NULL	Lagoinha	CE	NULL	170	0
9	Ivo Ilha	Rua de Baixo, 900	Ofeia	Recife	PE	1955-08-06	0	0
10	Julio Jales	Av. Alberto Maranhao, 83	Kivi	Olinda	PE	NULL	700	0

# Linguagem de consulta de dados

Considerando a tabela abaixo:

Qual a média de débitos de clientes para cada estado?



```
1 SELECT estado, avg(debito)
   FROM cliente
   GROUP BY estado;
```

Qual é a média de débitos de clientes para cada estado, mas só retorne as maiores que 100.



```
2 SELECT estado, avg(debito)
   FROM cliente
   GROUP BY estado
   HAVING avg(debito) > 100;
```

# Linguagem de consulta de dados

Qual o valor máximo do débito de cliente para cada estado onde o valor máximo seja inferior a 700



1 `SELECT estado, max(debito)`  
`FROM cliente`  
`GROUP BY estado`  
`HAVING max(debito) < 700;`

estado	max(debito)
CE	500
RN	100

Qual é o valor mínimo do débito de cliente para cada estado, onde este valor seja maior que 0



2 `SELECT estado, min(debito)`  
`FROM cliente`  
`GROUP BY estado`  
`HAVING min(debito) > 0;`

estado	min(debito)
CE	170

# Conclusão

- ❑ Aprendemos sobre agrupamento e agregações que podem ser realizadas em um `SELECT`.

# Na próxima aula

☐ Praticaremos os comandos.



# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 8 - Linguagem SQL - DQL - Prática

Prof. Jéssica Coelho

# Nesta aula

- Praticaremos agrupamentos e agregações.

# Conclusão

- ❑ Praticamos os agrupamentos e agregações

# Na próxima aula

❑ Estudaremos os joins.

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 12 - Linguagem SQL - DQL - Parte 4

Prof. Jéssica Coelho

# Nesta aula

- ❑ Consulta em múltiplas tabelas relacionadas entre si.

# Junções (JOINS)

O JOIN ou junção é a operação capaz de retornar os registros de um relacionamento entre tabelas.

A junção utiliza as FKs para relacionar as tabelas

Está relacionado ao conceito do Teorema de conjuntos: União, Interseção, Está contido em, Não está contido em.

# Diferentes tipos de Joins

- ❑ (INNER) JOIN: Retorna os registros que existem nas duas tabelas relacionadas;
- ❑ LEFT (OUTER) JOIN: Retorna todos os registros da tabela da esquerda e as correspondências com a tabela da direita. Se não existir correspondência, a tabela da direita é retornada com suas colunas nulas.
- ❑ RIGHT (OUTER) JOIN: Retorna todos os registros da tabela da direita e as correspondências com a tabela da esquerda. Se não existir correspondência, a tabela da esquerda é retornada com suas colunas nulas.
- ❑ FULL (OUTER) JOIN: União do LEFT com o JOIN



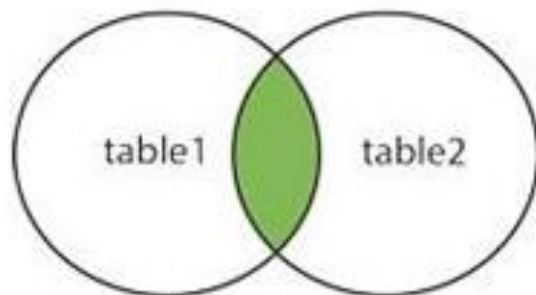
# SINTAXE

```
SELECT lista de colunas  
FROM  tabela1 [INNER | LEFT | RIGHT | FULL ]  
JOIN   tabela2 ON tabela1.coluna = tabela2.coluna;
```

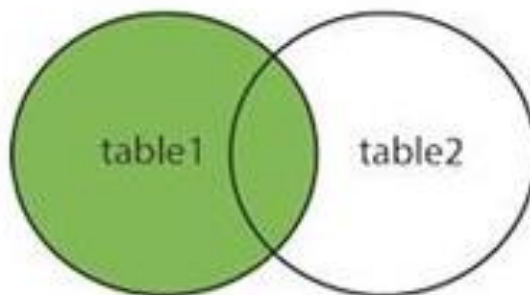
Onde selecionamos os registros de uma tabela JOIN com outra comparando as chaves FK de uma com a PK/chave única de outra na cláusula ON

# Tipos de Joins

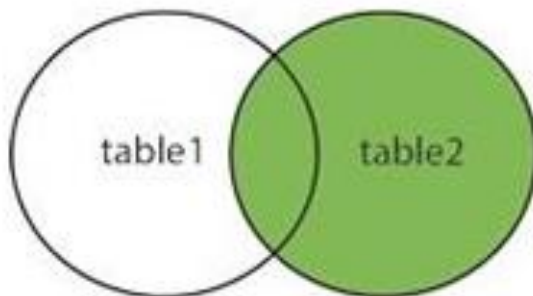
INNER JOIN



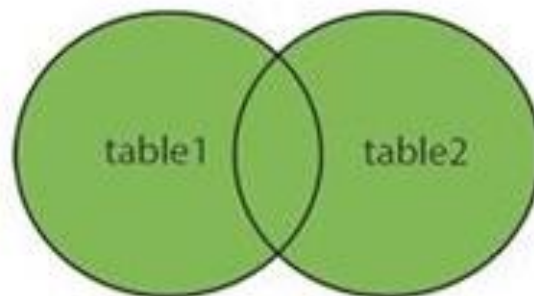
LEFT JOIN



RIGHT JOIN

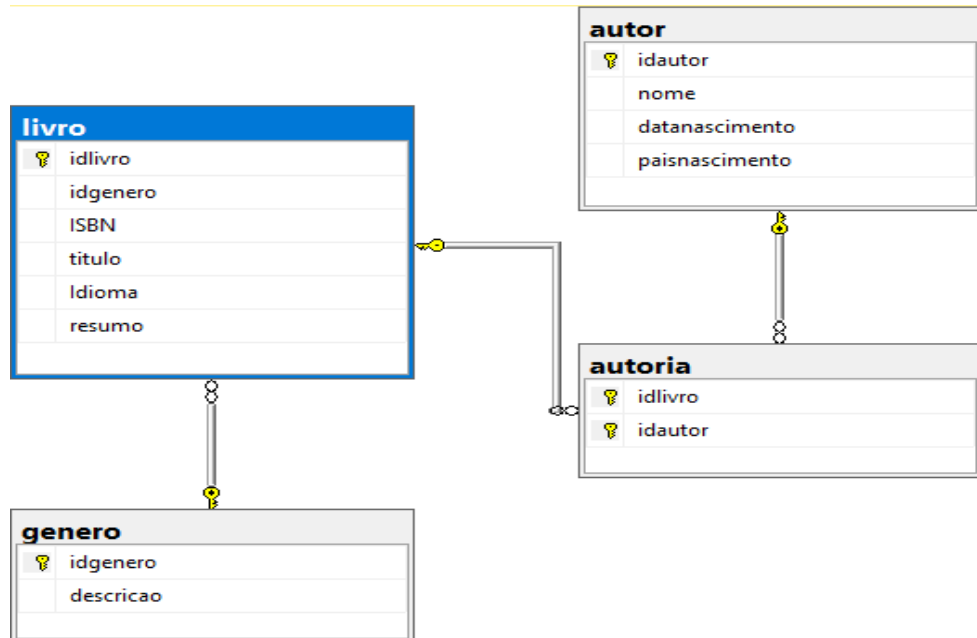


FULL OUTER JOIN



# Exemplos

❑ Considere as tabelas a seguir:



idlivro	idautor
3	3

idlivro	idgenero	ISBN	titulo	Idioma	resumo
3	1	4554654654	A menina que roubava livros	Portugues	Teste
4	3	658565465	Fundamentos em Engenharia de dados	Portugues	Teste
6	NULL	545454545	O segredo	Portugues	Test

idgenero	descricao
1	Romance
2	Suspense
3	Técnicos

idautor	nome	datanascimento	paisnascimento
3	Maria	1985-01-01	Brasil
4	José	1990-02-05	Brasil

Autoria

Livro

Gênero

Autor

Onde Livro se relaciona com Gênero através da FK idgenero e Autoria se relaciona com Autor e Livro pelas FKs Idlivro, idautor.

# Exemplos

❑ JOIN - Retornar todas os livros e seus respectivos gêneros:

```
SELECT * FROM Livro JOIN GENERO  
ON Livro.idgenero = GENERO.idgenero
```

Results							
Messages							
idlivro	idgenero	ISBN	titulo	Idioma	resumo	idgenero	descricao
3	1	4554654654	A menina que roubava livros	Portugues	Teste	1	Romance
4	3	658565465	Fundamentos em Engenharia de dados	Portugues	Teste	3	Técnicos



FK



PK de Gênero

# Exemplos

❑ LEFT JOIN - Retornar todos os gêneros tendo livros referenciados ou não:

A esquerda do Left



A direita do LEFT



```
SELECT * FROM genero LEFT JOIN Livro  
ON genero.idgenero = livro.idgenero
```

idgenero	descricao	idlivro	idgenero	ISBN	titulo	Idioma	resumo
1	Romance	3	1	4554654654	A menina que roubava livros	Portugues	Teste
2	Suspense	NULL	NULL	NULL	NULL	NULL	NULL
3	Técnicos	4	3	658565465	Fundamentos em Engenharia de dados	Portugues	Teste

Como para o gênero Suspense (a esquerda do LEFT) eu não tenho Livros cadastrados, os campos de Livro aparecem em nulo

# Exemplos

- ❑ LEFT JOIN com WHERE IS NULL – Retornar todos os Gêneros que não possuem Livros cadastrados

```
SELECT * FROM genero LEFT JOIN Livro  
ON genero.idgenero = livro.idgenero  
WHERE Livro.idlivro IS NULL
```

results							
Messages							
idgenero	descricao	idlivro	idgenero	ISBN	titulo	Idioma	resumo
2	Suspense	NULL	NULL	NULL	NULL	NULL	NULL

# Exemplos

- ❑ RIGHT JOIN: Retornar todos os gêneros tendo livros referenciados ou não:

```
SELECT * FROM Livro RIGHT JOIN genero  
ON genero.idgenero = livro.idgenero
```

esults Messages

idlivro	idgenero	ISBN	titulo	Idioma	resumo	idgenero	descricao
3	1	4554654654	A menina que roubava livros	Portugues	Teste	1	Romance
NULL	NULL	NULL	NULL	NULL	NULL	2	Suspense
4	3	658565465	Fundamentos em Engenharia de dados	Portugues	Teste	3	Técnicos

# Exemples

- ❑ FULL OUTER JOIN: Retorna todos os Livros tendo ou não gênero e todos os gêneros tendo ou não Livros cadastrados

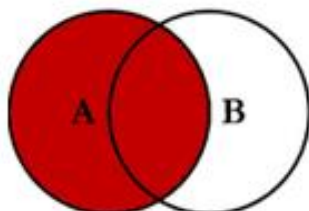
[illegible]



# Linguagem de consulta de dados

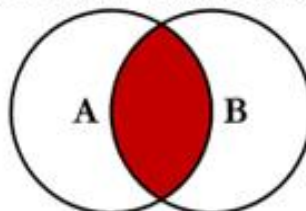
## SQL JOINS

Todos os registros de A estando ou não em B.



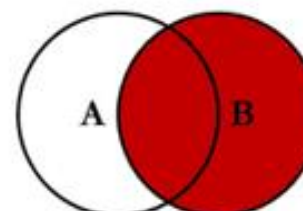
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```

Todos os registros que estão em A e em B - Interseção



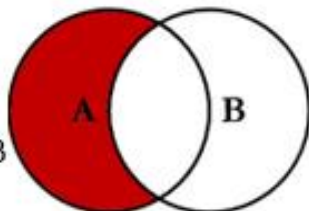
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```

Todos os registros de B estando ou não em A.



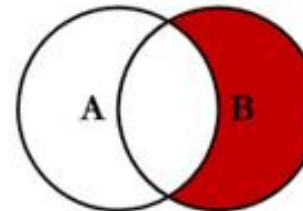
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```

Todos os registros que estão em A e não em B

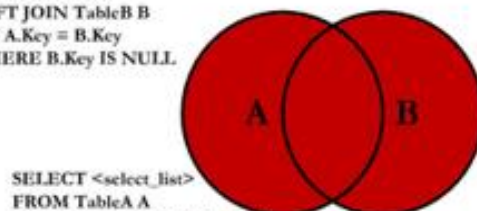


```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```

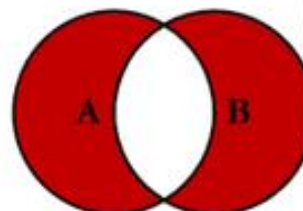
Todos os registros de B que não estão em A



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

Resultado do Left join  
+ Right join

Todos os registros que estão A ou em B mas não em ambos ao mesmo tempo

# Conclusão

- ❑ Aprendemos a realizar consultas em múltiplas tabelas;
- ❑ Aprendemos sobre as operações de JOIN;

# Na próxima aula

- ❑ Praticaremos os comandos estudados.

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 13 - Linguagem SQL - DQL - Prática Joins

Prof. Jéssica Coelho

# Nesta aula

- ❑ Praticaremos os comandos joins

# Conclusão

☐ Praticamos os joins

# Na próxima aula

□ Estudaremos sub-consultas

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 14 - Linguagem SQL - DQL - Parte 5

Prof. Jéssica Coelho



# Nesta aula

- Aprenderemos sobre as consultas aninhadas e seus operadores.

# Consultas aninhadas

❑ Consultas que envolvem mais de uma consulta, é o que chamamos de Sub-Consulta.

❑ Sintaxe:

**SELECT <Lista de colunas>**

**FROM <Nome tabela ou tabelas>**

**WHERE <Coluna> operador (SELECT <Lista de Colunas> FROM <Nome tabela ou tabelas WHERE ...>;**

❑ Operadores: IN, NOT IN, EXISTS, NOT EXIST, e os operadores condicionais já estudados (>, <, =, etc...)

## ❑ EXISTS OU NOT EXISTS

```
SELECT
    column_name
FROM Table_Name
WHERE <coluna> EXISTS (SELECT
    column_name
    FROM Table_Name
    WHERE condition);
```

```
SELECT
    column_name
FROM Table_Name
WHERE coluna NOT EXISTS
(SELECT
    column_name
    FROM Table_Name
    WHERE condition);
```

## ❑ IN / NOT IN

```
SELECT
    column_name
FROM Table_Name
WHERE IN (SELECT
    column_name
    FROM Table_Name
    WHERE condition);
```

```
SELECT
    column_name
FROM Table_Name
WHERE NOT IN (SELECT
    column_name
    FROM Table_Name
    WHERE condition);
```

# Exemplos:

```
SELECT *  
FROM Aluno  
WHERE Nota = (SELECT MAX(Nota) FROM Aluno).
```

```
CREATE table aluno (Nome varchar(max), Nota int)  
INSERT INTO aluno values ('CAIO', 7), ('MARIA', 5), ('JOAO', 10), ('MATEUS', 9), ('MARIANA', 10)  
  
SELECT * FROM Aluno  
WHERE Nota = (SELECT MAX(Nota) FROM Aluno)
```

0 %

Results Messages

	Nome	Nota
1	JOAO	10
2	MARIANA	10

# Sub-consulta no Join

```
select * from aluno
```

```
SELECT *  
FROM Aluno  
JOIN  
    (SELECT id FROM Aluno WHERE Nota = 10) AS NotaAluno  
ON Aluno.id = NotaAluno.Id
```

Results

Messages

id	nome	idade	Nota
1	Fernanda	20	5
2	Maria	40	10
3	Gabriel	50	7
4	Antonio	30	10
5	Marcos	27	2
6	Sofia	17	8

id	nome	idade	Nota	id
2	Maria	40	10	2
4	Antonio	30	10	4

# Conclusão

- ❑ Aprendemos o que são consultas aninhadas ou sub-consultas;

# Na próxima aula

- ❑ Praticaremos os comandos estudados.



# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 16 - Linguagem SQL - DQL - Parte 5

Prof. Jéssica Coelho

# Nesta aula

- Praticaremos a sub-consulta

# Na próxima aula

- ❑ Estudaremos UNION e UNION ALL

# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 16 - Linguagem SQL - DQL - Parte 6

Prof. Jéssica Coelho

# Nesta aula

- Estudaremos Union e Union ALL

# UNION

- ❑ O operador UNION combina os resultados de duas ou mais queries em um único result set, retornando todas as linhas pertencentes a todas as queries envolvidas na execução. Para utilizar o UNION, o número e a ordem das colunas precisam ser idênticos em todas as queries e os data types precisam ser compatíveis.
- ❑ Existem dois tipos do operador UNION, UNION e UNION ALL

# UNION

❏ Sintaxe:

SELECT <lista de colunas>

From tabela ou grupo de tabelas

UNION | UNION ALL

SELECT <mesma lista de colunas>

From tabela ou grupo de tabelas

# UNION

- ❑ O operador UNION, por default, executa o equivalente a um SELECT DISTINCT no result set final. Ele combina o resultado de execução das duas queries e então executa um SELECT DISTINCT a fim de eliminar as linhas duplicadas. Caso você não queira valores distintos pode utilizar UNION ALL



# UNION ALL

- ❑ O operador UNION ALL tem a mesma funcionalidade do UNION, porém, não executa o *SELECT DISTINCT* no result set final e apresenta todas as linhas, inclusive as linhas duplicadas.

# Exemplo

## UNION ALL

```
SELECT * FROM Aluno
```



id	nome	idade	Nota
1	Fernanda	20	5
2	Maria	40	10
3	Gabriel	50	7
4	Antonio	30	10
5	Marcos	27	2
6	Sofia	17	8

```
--UNION ALL
SELECT * FROM Aluno
WHERE Nota > 0
```

```
UNION ALL
```

```
SELECT * FROM Aluno
WHERE Nota < 8
```



id	nome	idade	Nota
1	Fernanda	20	5
2	Maria	40	10
3	Gabriel	50	7
4	Antonio	30	10
5	Marcos	27	2
6	Sofia	17	8
1	Fernanda	20	5
3	Gabriel	50	7
5	Marcos	27	2

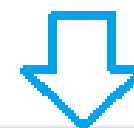
## UNION

```
--UNION
```

```
SELECT * FROM Aluno
WHERE Nota > 0
```

```
UNION
```

```
SELECT * FROM Aluno
WHERE Nota < 8
```



id	nome	idade	Nota
1	Fernanda	20	5
2	Maria	40	10
3	Gabriel	50	7
4	Antonio	30	10
5	Marcos	27	2
6	Sofia	17	8

# Conclusão

- ❑ Estudamos o que é Union e Union all

# Na próxima aula

- ❑ Praticaremos os comandos estudados.



# Fundamentos em Engenharia de Dados

Capítulo 3 Aula 17 - Linguagem SQL - DQL - Prática Union

Prof. Jéssica Coelho

# Nesta aula

□ Praticaremos Union e Union ALL

# Conclusão

□ Praticamos as cláusulas UNION e UNION ALL

# Na próxima aula

- ❑ Iniciaremos o estudo de Modelagem Multidimensional



# Fundamentos em Engenharia de Dados

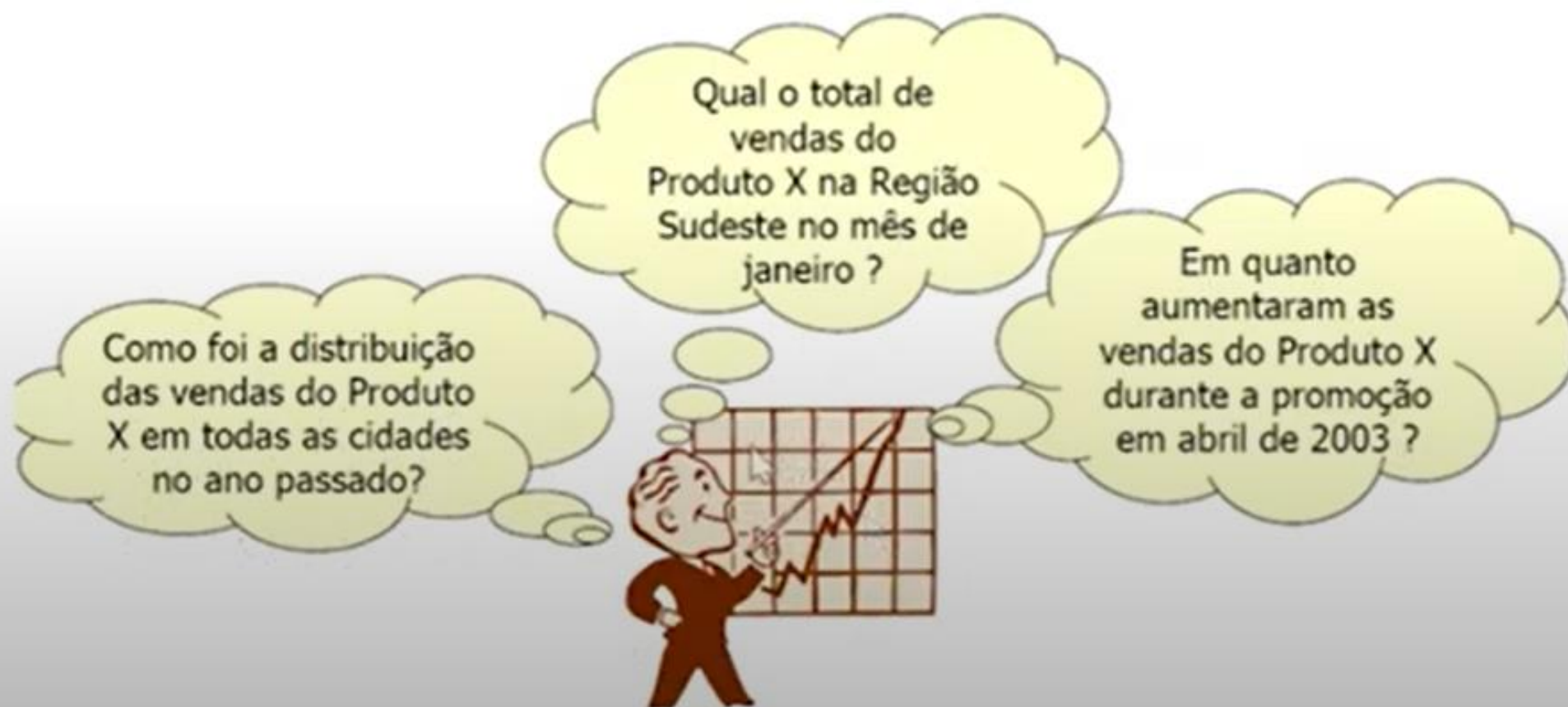
Capítulo 4 Aula 1 - Modelo dimensional

Prof. Jéssica Coelho

# Nesta aula

- ☐ Entender o que é modelagem dimensional;
- ☐ Entender o que é data warehouse;
- ☐ Entender o que é data mart;
- ☐ Entender os aspectos principais de extração, transformação e carga;
- ☐ Entender o que é OLAP e as diferenças com o OLTP.

# Visão multidimensional



# Modelagem Dimensional

- ❑ É uma modelagem focada no armazenamento de dados em arquitetura que nos permita visão multidimensional. Tem como local de armazenamento resultante o Data Warehouse;
- ❑ Data warehouse é um repositório central de informações que são usados para análises do negócio e tomada de decisão. Os dados salvos no DW fluem de diversas fontes após serem transformados, agrupados, organizados e sumarizados.

Implementa visão multidimensional;

Orientado por assunto, pelo fato do negócio;

Integrado;

Não volátil

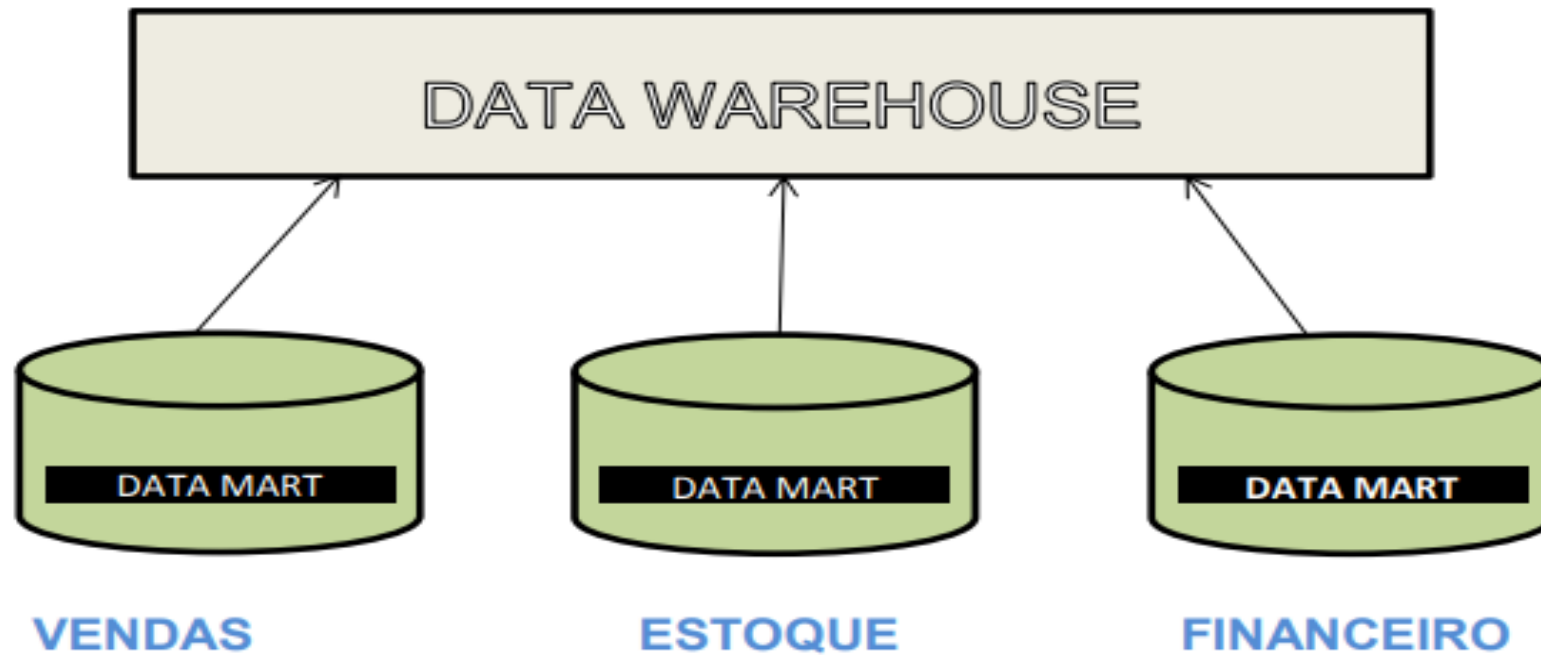
# Data Mart

Visão setorial, de área ou diferentes níveis de sumarização;

Atende às necessidades de uma determinada comunidade de usuários;

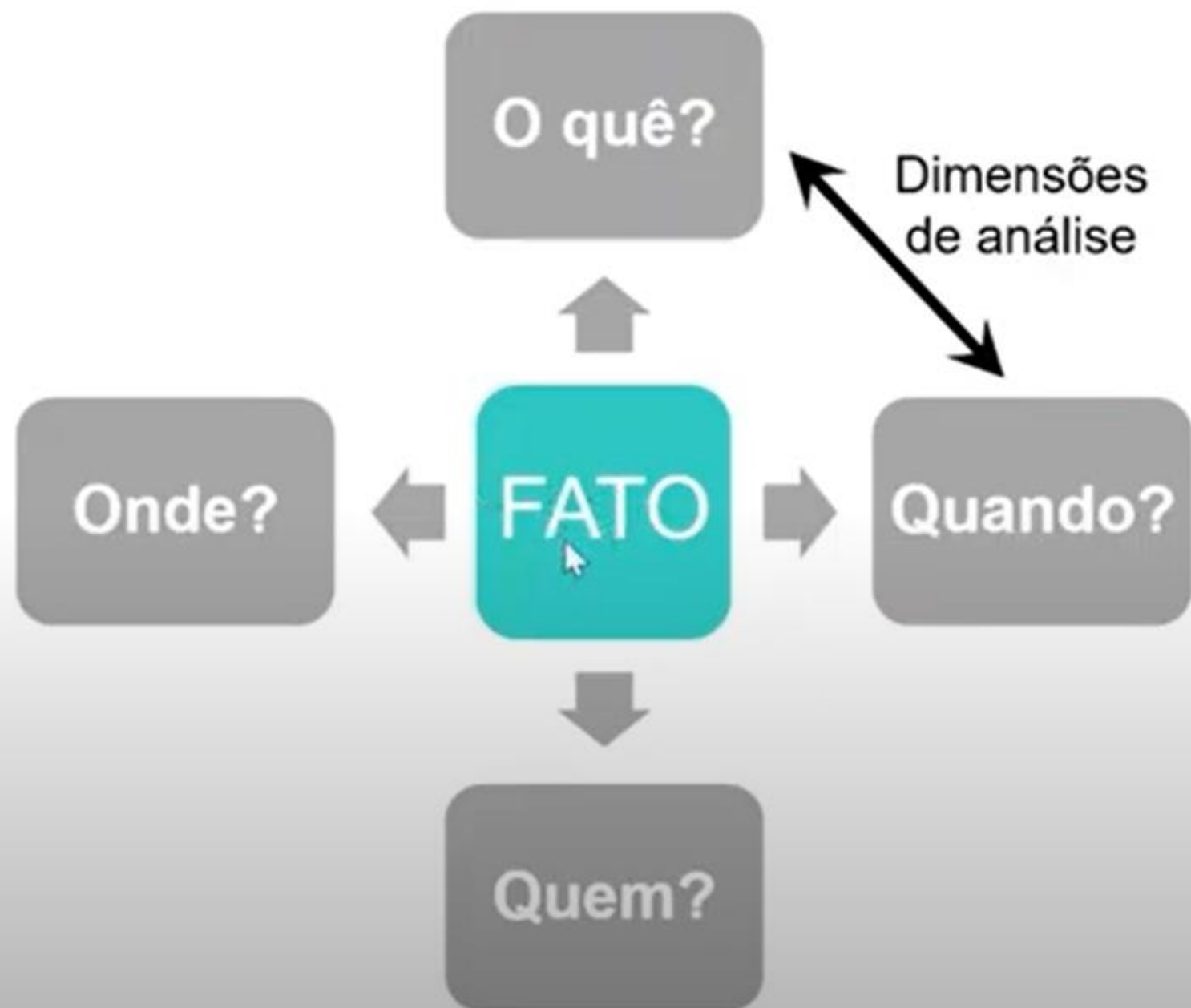
Costuma ser dito que Data Warehouse é um conjunto de Data Marts;

# Data Mart



# Elementos gerais de um DW ou DM

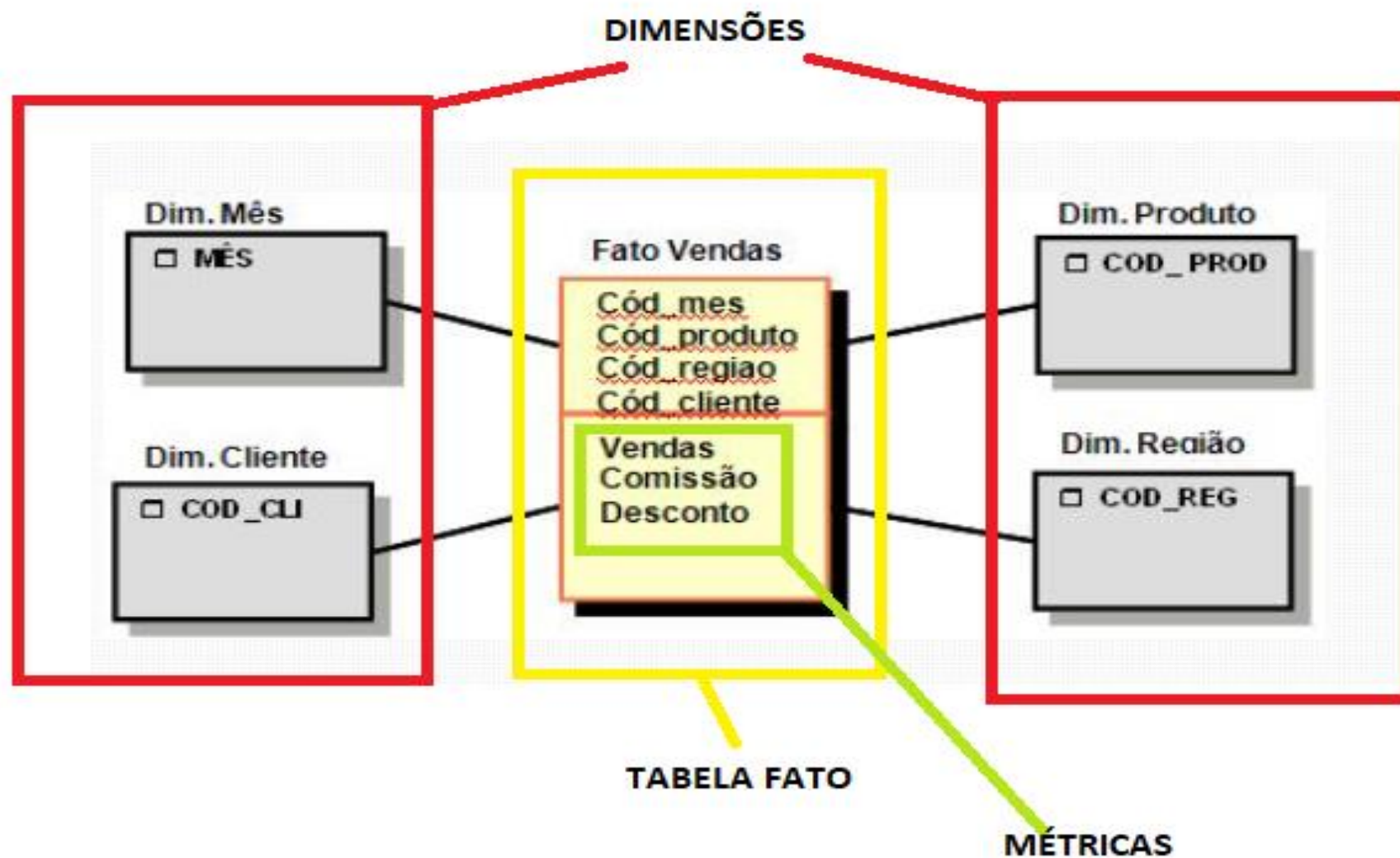
- ❑ Fatos: Acontecimento do negócio que desejo visualizar para tomada de decisão estratégica. Ex.: Vendas, Compras, Aluguel de carros. Se tornam a tabela Fato do DW;
- ❑ Dimensão: São entidades do negócio que apresentam alguma influência sobre o fato em análise;
- ❑ Métricas: são responsáveis por mensurar, monitorar e gerir as estratégias de uma empresa;
- ❑ KPIs: Indicadores chave de desempenho são criados a partir das métricas







igti

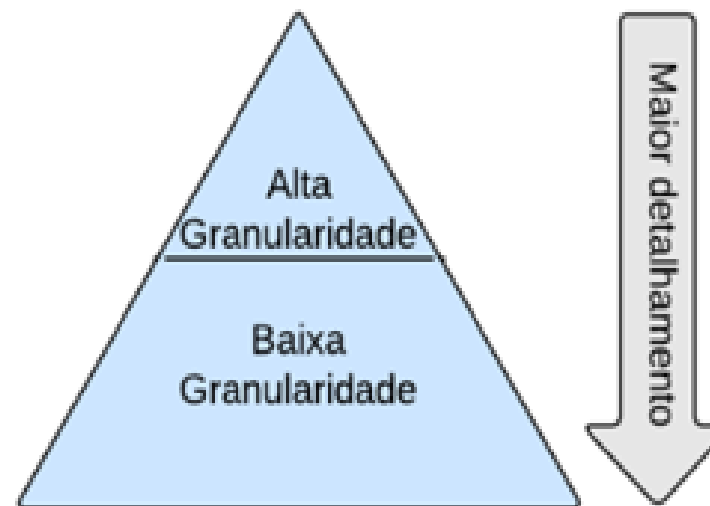
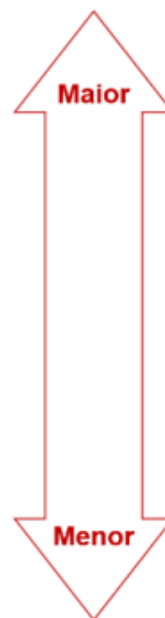


# Granularidade

- Granularidade dos dados é a concepção que determina o armazenamento e tratamento dos dados em diferentes unidades ou níveis de detalhamento:

Ocorrência	2017	2018	2019
Assaltos à mão armada	123	109	158
Furtos em residências	75	90	101
Furtos de veículos	243	250	332
Assassinatos	89	77	167
Estupros	2	24	69

Ocorrência	1º sem 2017	2º sem 2017	1º sem 2018	2º sem 2018	1º sem 2019	2º sem 2019
Assaltos à mão armada	50	73	45	64	70	88
Furtos em residências	40	35	60	30	71	30
Furtos de veículos	121	123	120	130	165	167
Assassinatos	40	49	37	40	67	100
Estupros	2	0	14	10	30	39



ETL (Extract Transformation Load) ou ETC (Extração transformação e Carga)



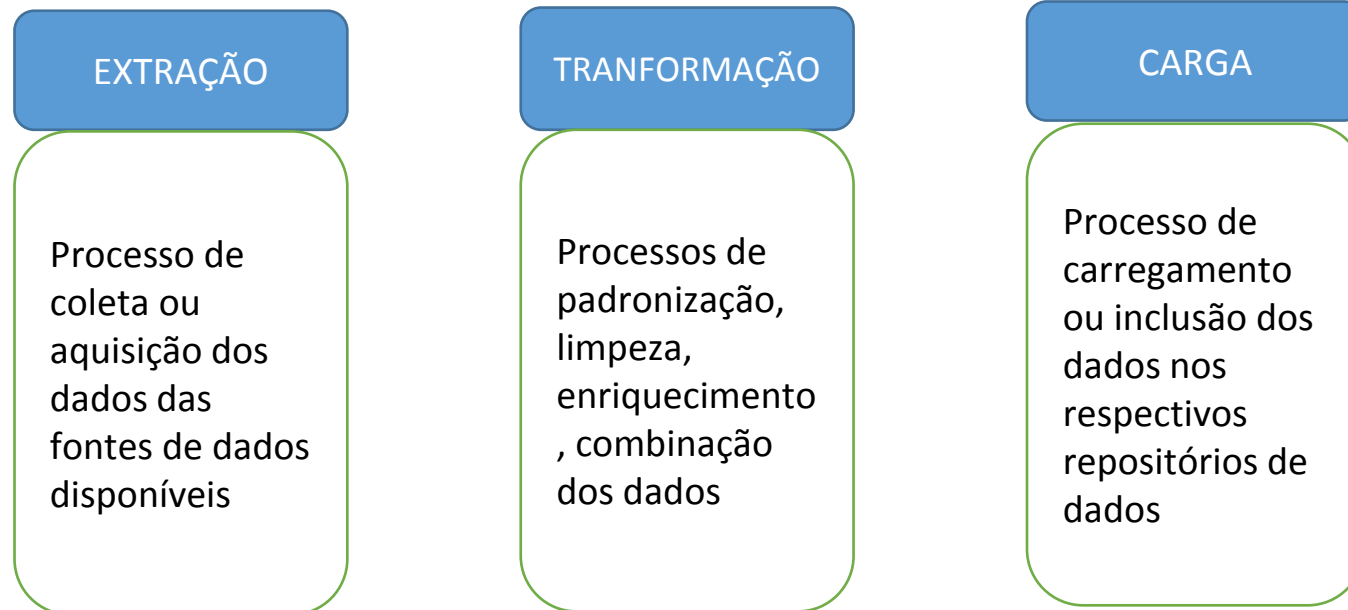
Fontes de dados

ETC/ETL

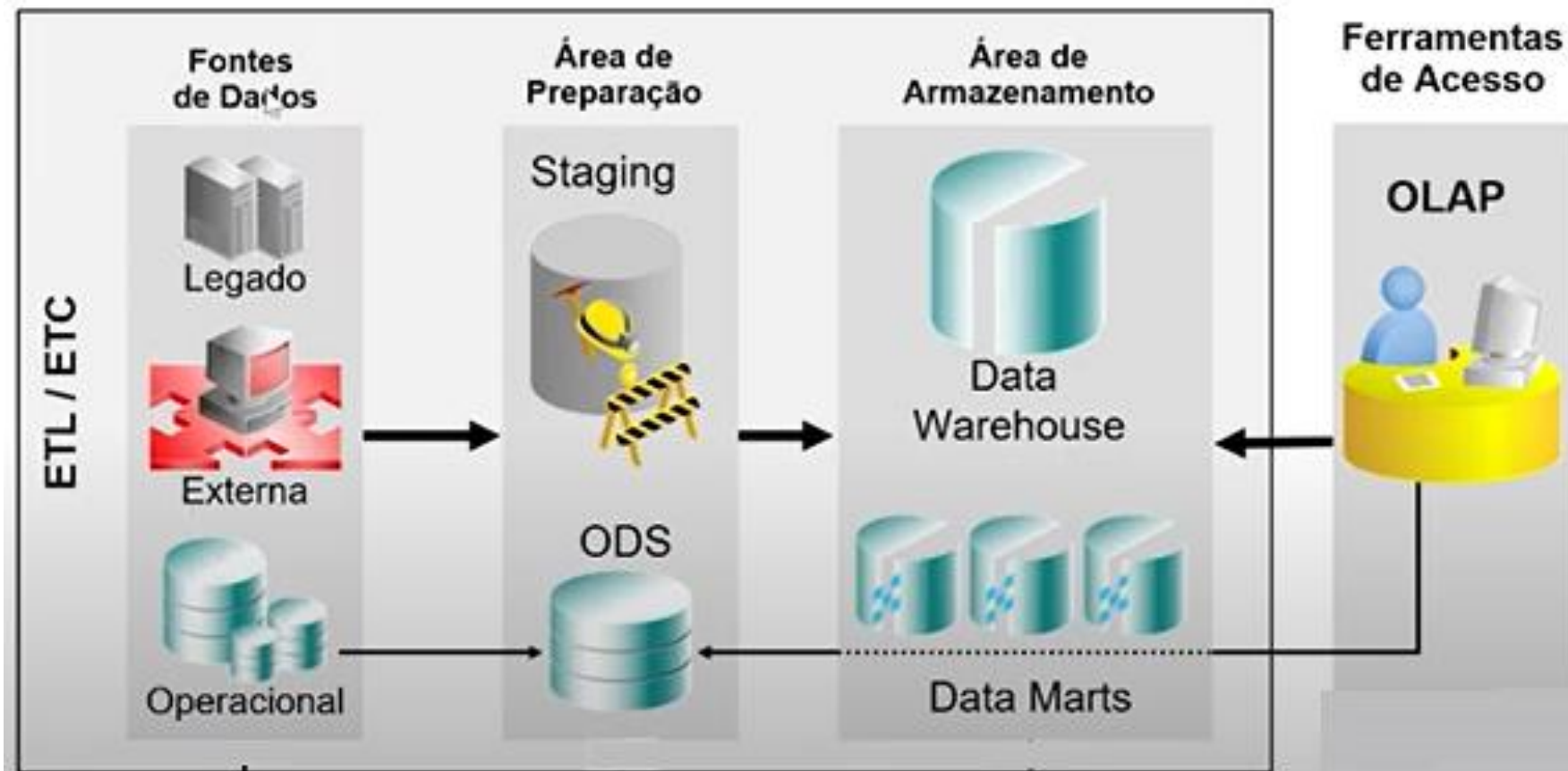
Data warehouse

## ETL (Extract Transformation Load) ou ETC (Extração transformação e Carga)

É a fase do processamento de dados em que carregamos dados de diversas fontes, transformando os padrões em um formato único, e carregados no ambiente de análise de dados, comumente DW



# Componentes do Processamento para DW



# OLTP versus OLAP

## OLTP (Online Transaction Processing)

Finalidade é controlar e executar tarefas empresariais fundamentais

Dados operacionais, atuais, refletem o presente

São a fonte original dos dados

## OLAP (Online Analytical Processing)

Finalidade é ajudar no planejamento, resolução de problemas estratégicos

Dados de consolidação históricos

Originários de vários bancos de dados OLTP

# Conclusão

- ❑ Entendemos o que é Modelagem dimensional e seus aspectos principais;
- ❑ Conhecemos o que é Data Warehouse e Data Marts;
- ❑ Conhecemos as fases de processamento de dados (ETL ou ETC);
- ❑ Conhecemos o OLAP e o comparamos com OLTP.



# Próxima aula

- ❑ Entenderemos os tipos de arquitetura de DW;
- ❑ Estudaremos as etapas da modelagem dimensional

# Fundamentos em Engenharia de Dados

Capítulo 4 Aula 2 - Modelo dimensional - Parte2

Prof. Jéssica Coelho

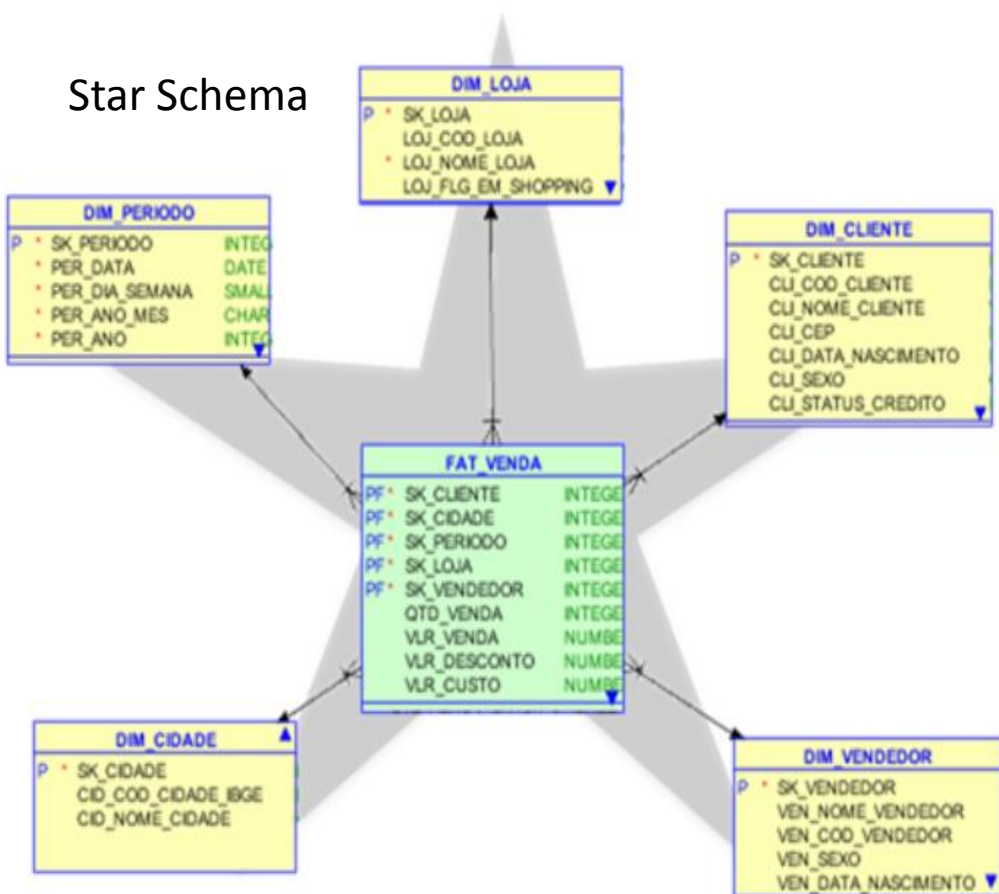
# Nesta aula

- ❑ Entenderemos os tipos de arquitetura de DW;
- ❑ Estudaremos as etapas da modelagem dimensional;
- ❑ Dicas de como identificar as fatos, dimensões e métricas de um cenário de negócio;

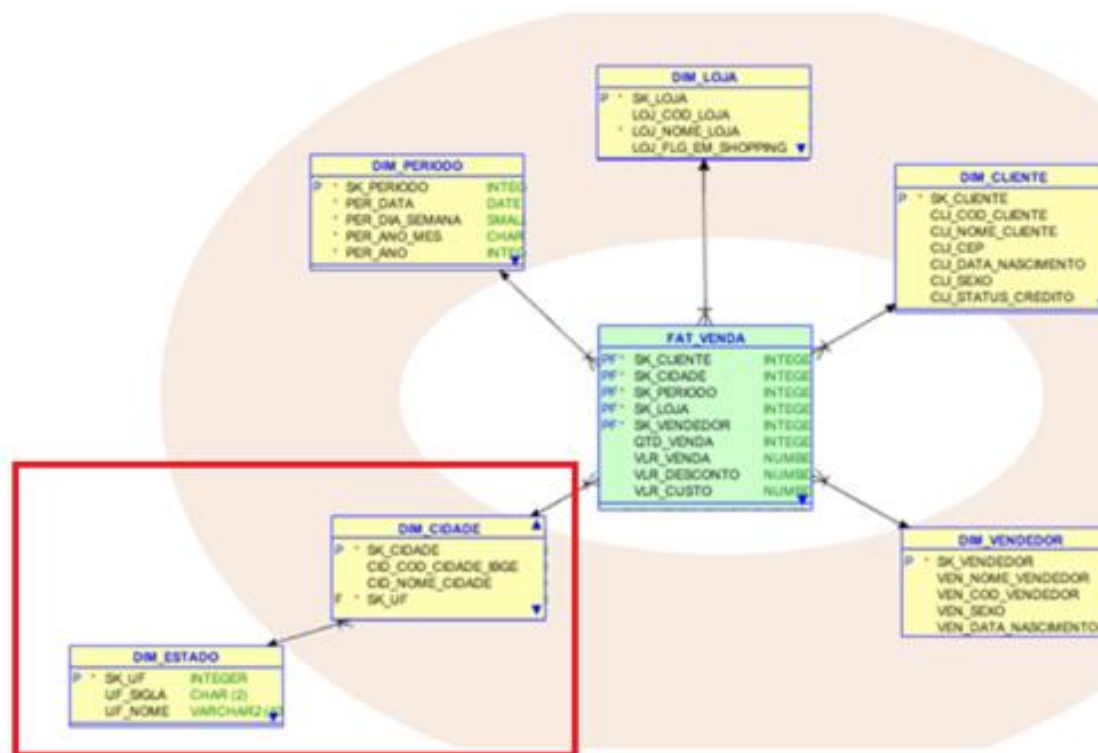
# Arquiteturas de DW

Existem basicamente duas maneiras de desenhar seu DW, ou Star Schema ou Snowflake, ambos funcionam melhor para determinado cenário:

Star Schema



Snowflake



# Etapas de uma modelagem dimensional

Levantar os  
Requisitos

Identificar os  
fatos do  
negócio que  
quero  
analisar

Identificar as  
possíveis  
métricas

Definir o grau  
de  
detalhament  
o ou  
granularidade

Identificar  
pelo que  
quero avaliar  
a métrica de  
determinado  
fato, ou seja,  
as dimensões

Identificar  
qual a  
melhor  
arquitetura:  
Snowflake  
ou  
Starschema  
ou mesmo  
híbrido

# Descobrendo Fatos e Dimensões

- ❑ Que processo/acontecimento do negócio quero avaliar? (Fato)
  - Vendas;
  - Aluguel de carros
- ❑ Quais valores usaremos para medir esses fatos? (Métricas):
  - Total de vendas;
  - Total receita;
  - Total de aluguel de carros ocorridos;
- ❑ A métrica será analisada PELO QUE? (Dimensão):
  - Região, Data...
- ❑ A que nível de detalhe queremos ir? (granularidade):
  - Quantidade por país, estado ou cidade?
  - Quantidade por dia ou mês, ou hora?

# Descobrimos Fatos e Dimensões

□ Ao final teremos:

***Quais foram as vendas no período de junho na região sul:***

Fato: Vendas

Dimensões: data e região.

Granularidade: Mês para período e regiões para a região.

# Conclusão

- ❑ Entendemos o que é Modelagem dimensional e seus aspectos principais;
- ❑ Conhecemos o que é Data Warehouse e Data Marts;
- ❑ Conhecemos como identificar Fatos, Dimensões e Métricas em um dado cenário.



# Próxima aula

- Estudaremos mais sobre OLAP e seus elementos.

# Fundamentos em Engenharia de Dados

Capítulo 4 Aula 3 - OLAP

Prof. Jéssica Coelho

# Nesta aula

- Estudaremos o OLAP e seus principais elementos;

# OLAP

conjunto de técnicas para a análise de dados multidimensionais complexos do Data Warehouse ou de outras fontes como Data Lake que será estudado em outros módulos do curso. Em resumo o DW é o armazenamento e o OLAP é o processamento destes dados em múltiplas visões

ROLAP

MOLAP

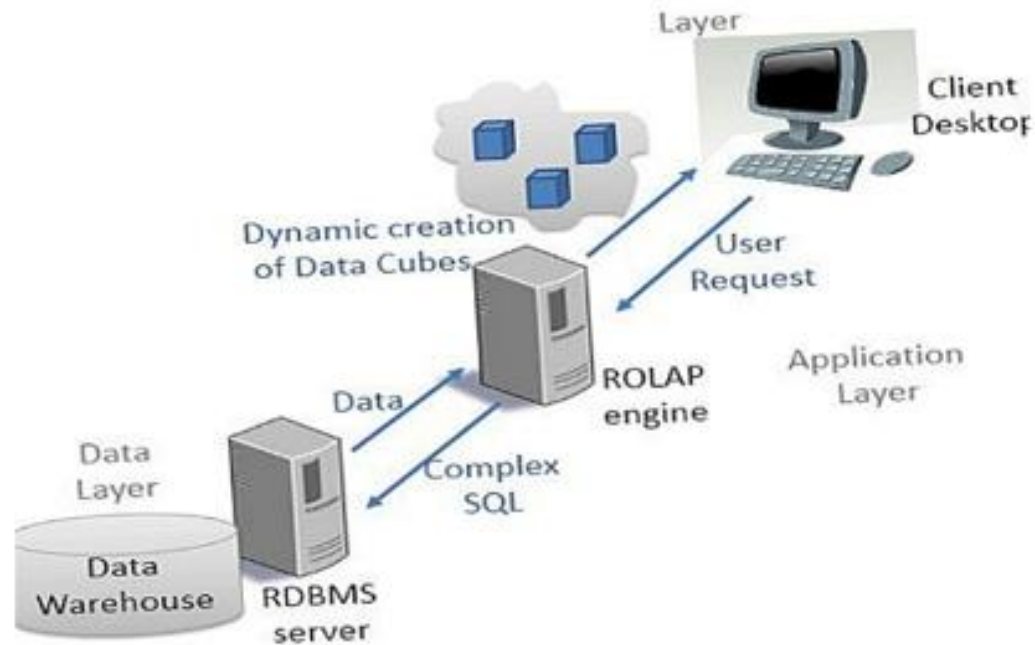
HOLAP

DOLAP

WOLAP

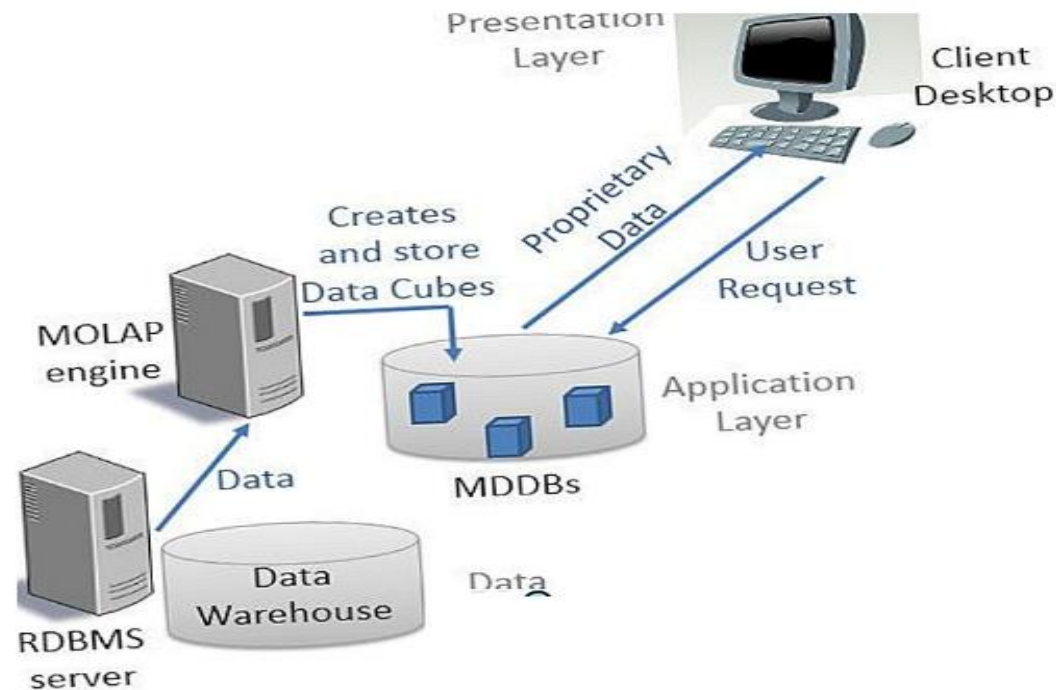
# ROLAP (Relational Online Analytical Processing)

- os dados são armazenados em bases relacionais, com linhas e colunas e para exibir os dados em uma visualização multidimensional é criada uma camada semântica de metadados que mapeia as dimensões para as tabelas relacionais.



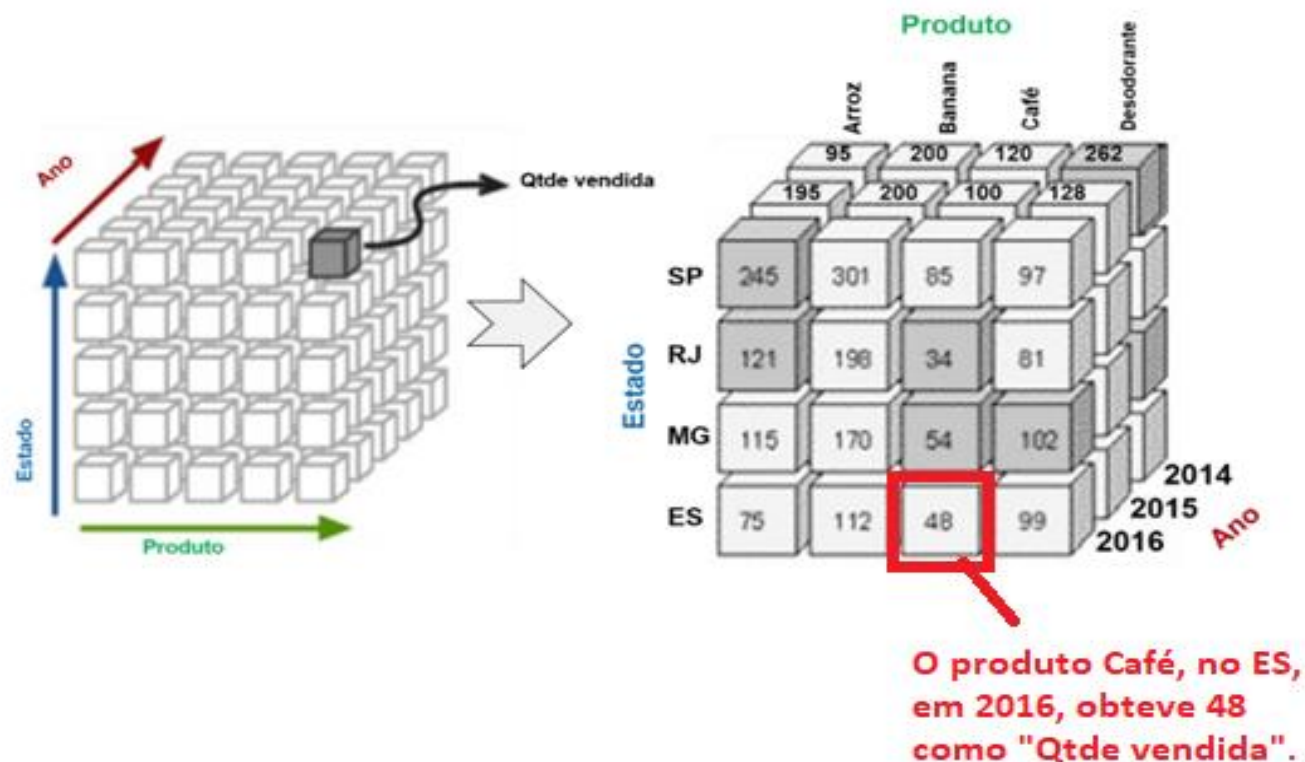
# MOLAP (Multidimensional Online Analytical Processing)

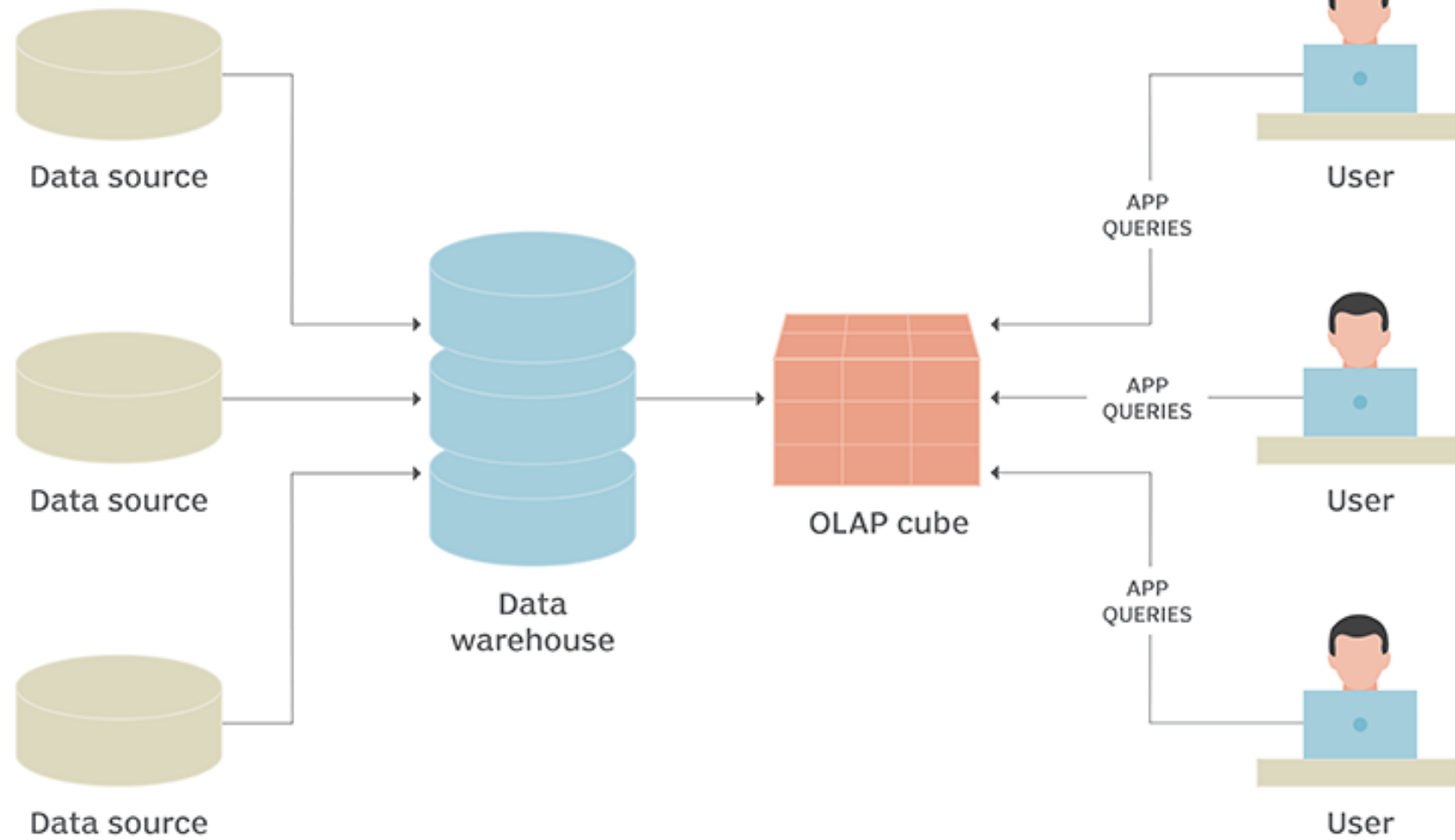
- Os dados utilizados para análise são armazenados em bancos de dados multidimensionais especializados (MDDBs). Os sistemas multidimensionais de gerenciamento de banco de dados são sistemas de softwares proprietários



# Cubo

Resultado de um processamento OLAP e nos permite, assim como sua figura geométrica correspondente, analisar o fato sob diferentes ângulos ou dimensões.



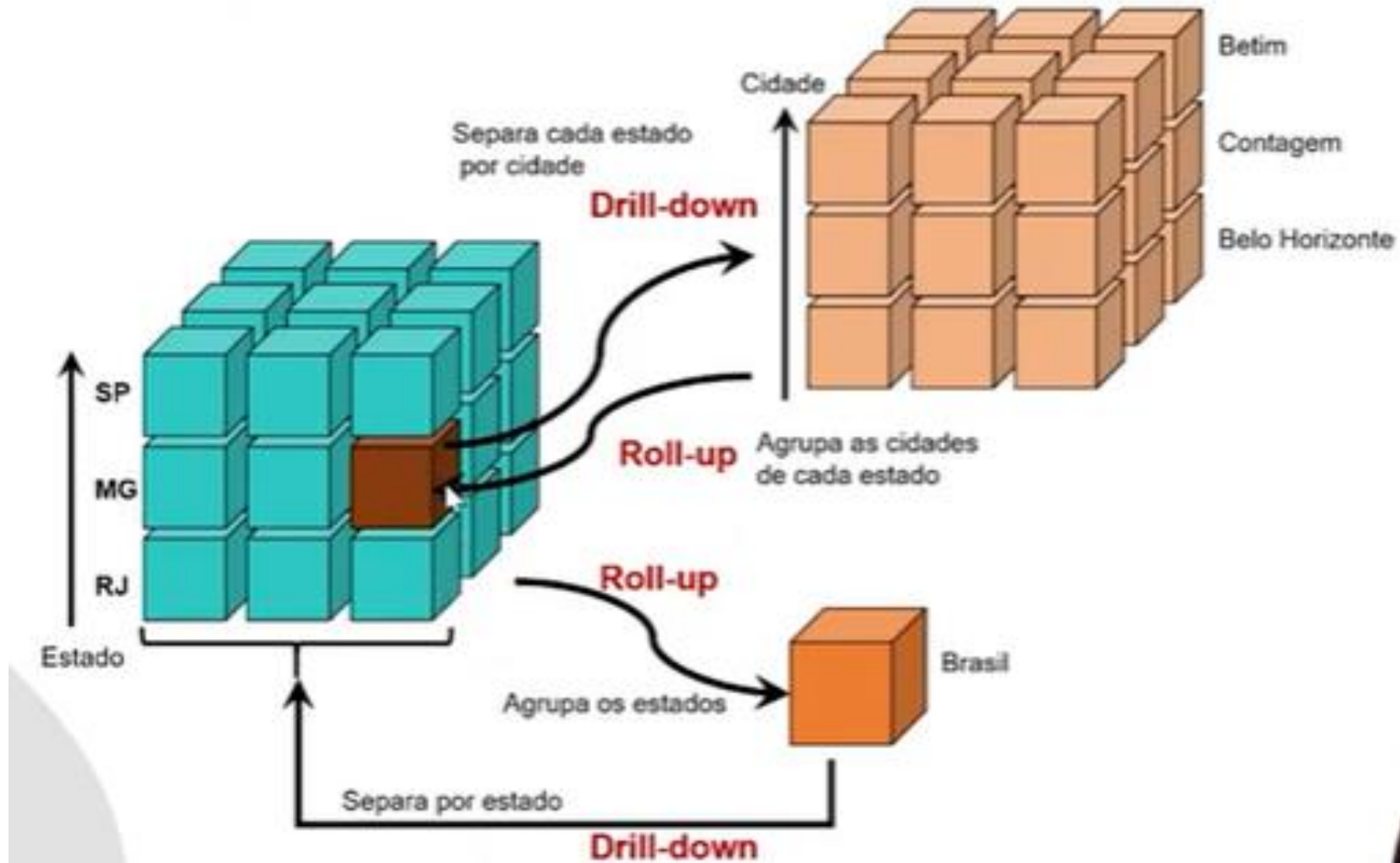




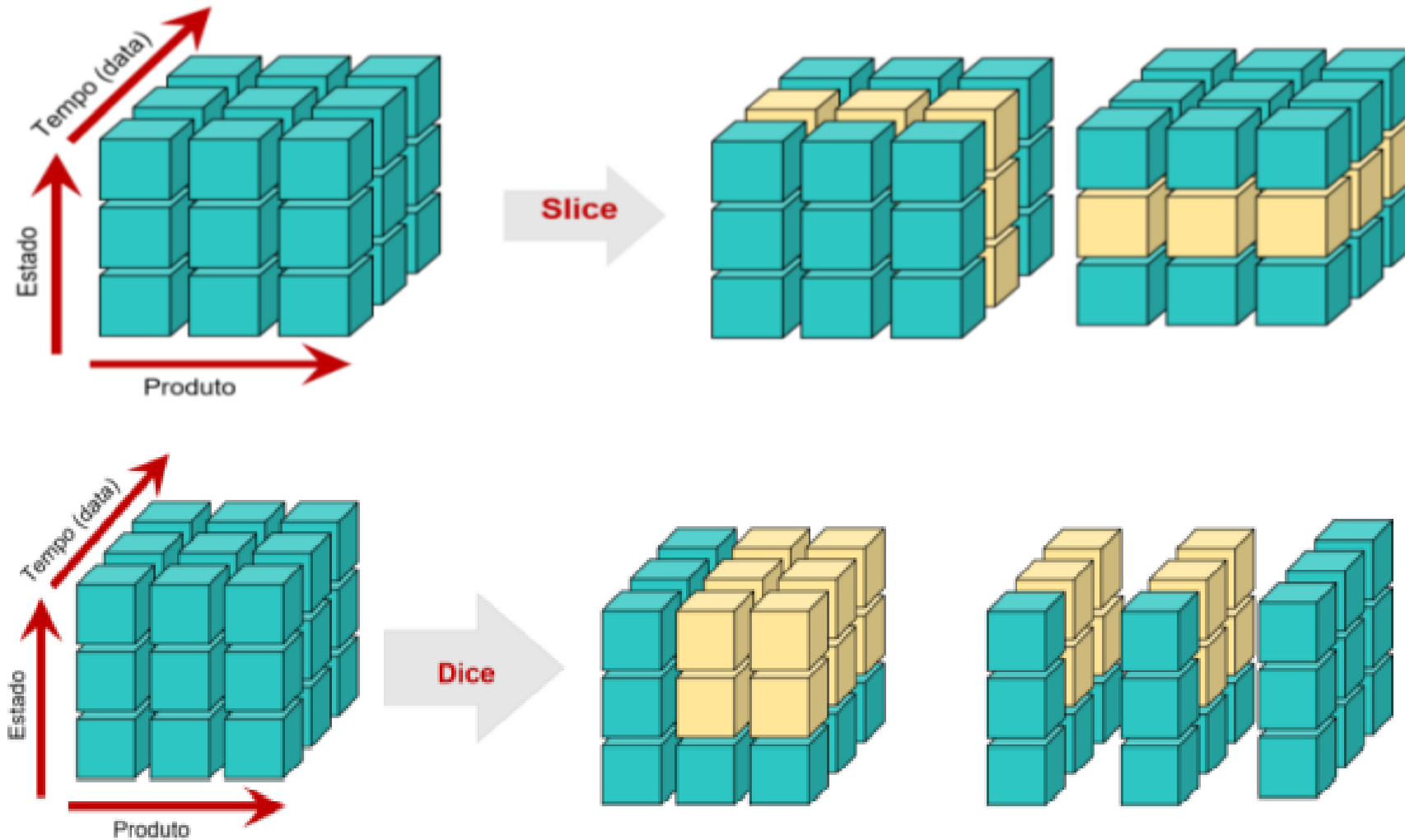
# Operações no Cubo

- ❑ Drill down / Roll up ou drill up: modifica o nível de detalhamento.
- ❑ Slice: Filtra as dimensões para só trazer parte dos dados;
- ❑ Dice: Retira uma ou mais dimensões para se analisar pelas demais;

# Drill down / Roll up ou drill up



# SLICE E DICE



# Conclusão

- ❑ Estudamos o OLAP e seus principais elementos;
- ❑ Conhecemos os tipos de arquitetura OLAP;
- ❑ Entendemos sobre cubos;
- ❑ Entendemos as operações que podemos realizar com os cubos.

# Próxima aula

- ❑ Iniciaremos a prática para Data Warehouse.

# Fundamentos em Engenharia de Dados

Capítulo 4 Aula 4 - Data hubs, Data Lakes, Data Swamps e Data Ponds

Prof. Jéssica Coelho

# Nesta aula

- Estudaremos os conceitos de Data Hubs, Data Lakes, Data Swamps e Data Ponds

# Data hub



Repositório de dados organizados, governados, a fim de permitir distribuição e compartilhamento destes dados. Foca em integrar diferentes silos de dados em um única plataforma ou arquitetura. Silos são as chamadas ilhas de informação.

Permite que você veja como os dados fluem em sua empresa. Ele coleta e compartilha dados em vários formatos para obter insights mais acessíveis.

É uma ferramenta de catálogo de dados, pode ser usada para centralizar as informações de Data Lakes, Data Warehouse e outras fontes de dados.



# Repositório de Dados

- ❑ Repositórios de dados são plataformas responsáveis por armazenar grandes conjuntos de dados para suportar análises de negócios

Data Silo

Data  
warehouse

Data mart

Metadados

Data pond

Data swamp

Data lake

Cubos

# Data Lake (Lago de Dados)

- ❑ Termo criado pelo CTO (Chief Technical Officer) do Pentaho, James Dixon;
- ❑ Repositório para dados estruturados, não estruturados e semiestruturados;
- ❑ Os dados estão armazenados no estado bruto, ou seja, sem tratamento, só indexados;
- ❑ Os dados inseridos são atrelados a metadados (tags) para identificação, localização e uso (análise);
- ❑ Plataformas de gerenciamento de dados em toda a empresa para analisar fontes de dados diferentes em seu formato nativo.

## COMO FUNCIONAM OS LAGOS DE DADOS ?

O conceito pode ser comparado a um corpo de água, um lago, onde a água flui, enchendo um reservatório e saindo.

### Dados estruturados

1. Informações em linhas e colunas.
2. Facilmente ordenado e processado com ferramentas de mineração de dados

1

O fluxo de entrada representa vários arquivos de dados brutos que variam de e-mails, planilhas, mídias sociais, etc.

2

O reservatório de água é um conjunto de dados, onde você executa análises em todos os dados.

3

O fluxo de água é o dado analisado.

4

Por meio desse processo, você é capaz de "filtrar" todos os dados rapidamente para obter informações importantes

### Dados não estruturados

1. Dados brutos e não organizados.
2. E-mails.
3. arquivos PDF.
4. Imagens, vídeo e áudio.
5. Ferramentas de mídia social.



### Data swamps

- Data Lakes não governados, gerenciados;
- Dados com baixa qualidade e sem contexto;
- Metadados ausentes ou mínimos;

### Data Pond

- Uma série de repositórios isolados de dados brutos em seu formato nativo;
- Usados como local intermediário temporário para informações brutas importantes apenas;
- Serão adicionados a um data lake.

### Data River

- Remete a fluxo de dados em tempo real;
- Streaming ou dados transacionais onde as decisões são tomadas durante o evento

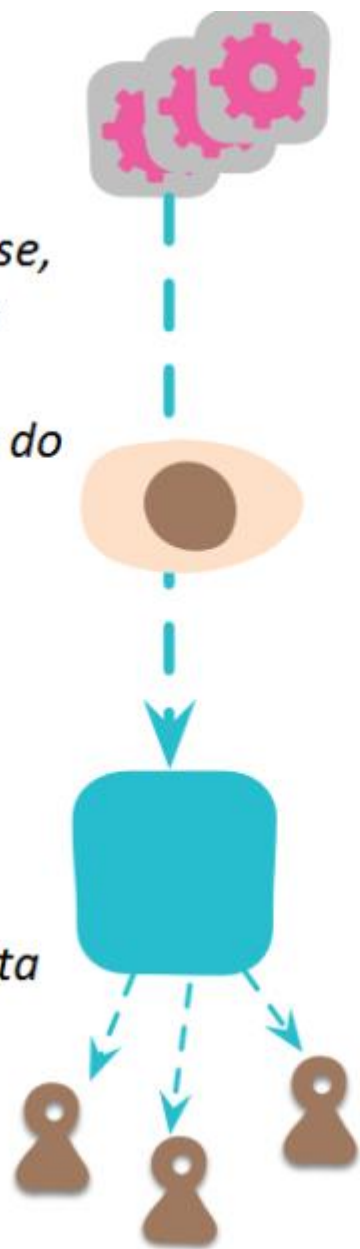
# Data Warehouse vs Data Lake

	Data warehouse	Data lake
tipos de dados	estruturados	estruturados e não estruturados
estrutura	pré-definida	aberta
esquema de dados	definido na gravação	definido na leitura
estado do dados	processado/transformado	brutos, transformado apenas quando for usado
propósito	pré-definido	aberto

# ETC vs ECT

*Com o Data Warehouse, os dados são limpos e organizados em um único esquema, antes do armazenamento*

*A análise é feita consultando diretamente no Data Warehouse*



*Com o Data Lake, os dados são armazenados em seu formato bruto*

*Os dados são selecionados e organizados de acordo com a necessidade*



# Conclusão

- ❑ Entendemos os conceitos de Data Hubs, Data Lakes, Data Swamps e Data River;
- ❑ Entendemos a diferença entre Data Lake e Data Warehouse;



# Próxima aula

- Faremos uma prática em Amazon Redshift



# Fundamentos em Engenharia de Dados

Capítulo 4 Aula 5 - Prática Redshift

Prof. Jéssica Coelho

# Nesta aula

- Faremos uma prática para subirmos um Cluster Amazon Redshift

# Conclusão

- ❑ Subimos um cluster de Redshift juntos

# Fundamentos em Engenharia de Dados

Capítulo 4 Aula 6 - Prática Modelagem Dimensional

Prof. Jéssica Coelho

# Nesta aula

- Praticaremos a criação de um DW baseado na Empresa de Seguros de Carros.