.StartSe
<tech/academy>

Gustavo Viais
Tech Lead

.StartSe

Apresentação

Banco de dados / SQL

Apresentação

- **Introdução**
  - O que é um banco de dados?
- **Conceitos BD**
  - Modelo Entidade-Relacionamento;
  - Cardinalidade;
  - Modelo Relacional;
  - Mapeamento BD.
- **Linguagem SQL**
  - Linguagem de Definição de Dados;
  - Linguagem de Manipulação de Dados;
  - Hands-on.

.StartSe
<tech/academy>

Gustavo Viais
Tech Lead

.StartSe

Apresentação

Banco de dados / SQL

Gustavo Viais
Tech Lead

Introdução

Banco de dados / SQL

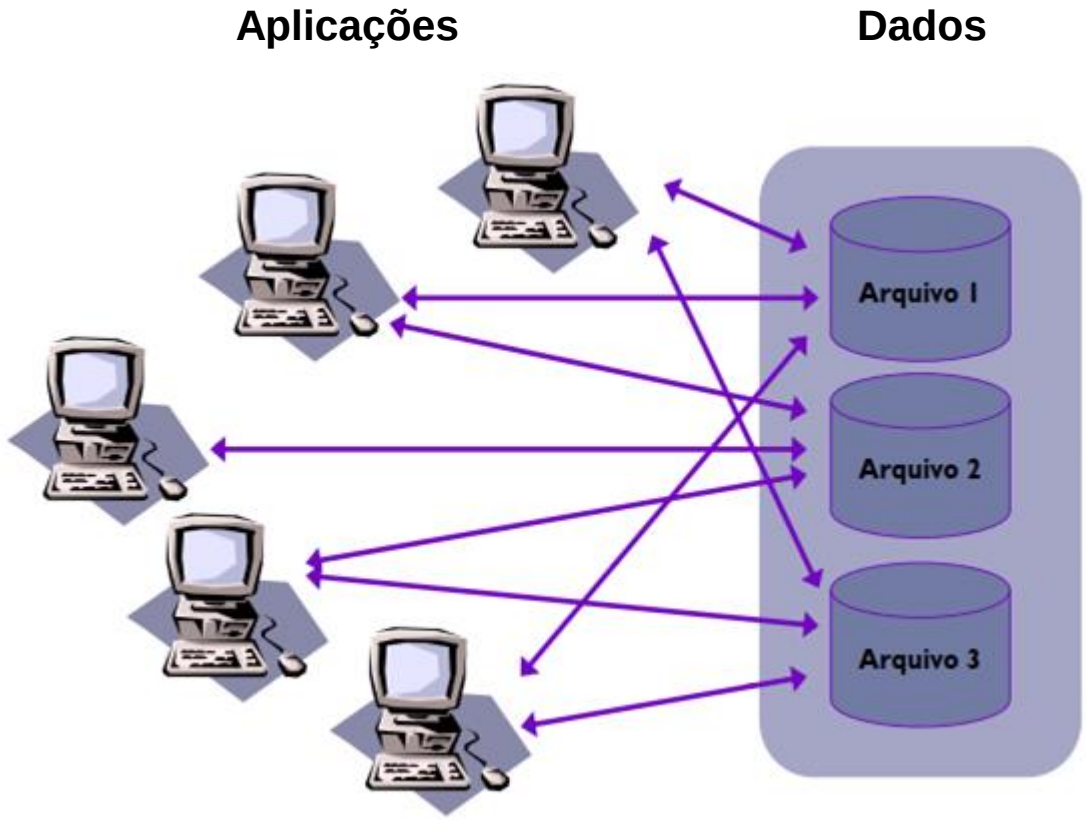
Motivação

*“Um banco de dados (BD) é uma **coleção organizada** de informações - ou dados - **estruturadas**, **normalmente** armazenadas eletronicamente em um sistema de computador.”*

Fonte: https://www.oracle.com/br/database/what-is-database/

Problema

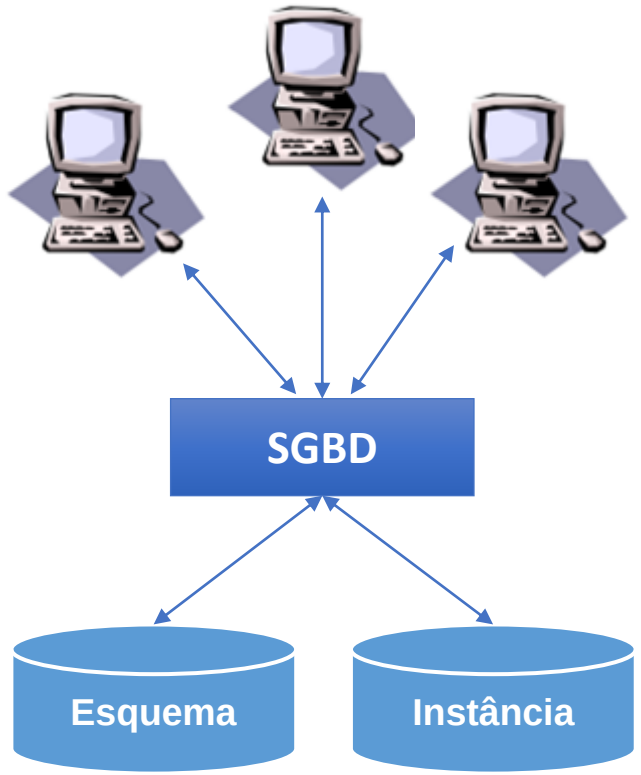
- Redundância
- Inconsistência
- Dificuldade no acesso a dados
- Isolamento dos dados
- Múltiplos usuários
- Segurança
- Integridade
- Atomicidade



Justificativa

Sistema de Gerenciamento de Bases de Dados (SGBD)

- Interface entre o banco de dados e seus usuários finais ou programas;
- Permitir recuperação, atualização e gerenciamento das informações;
- Facilitar a supervisão e o controle do bancos de dados.



Características de um SGBD

- **Controle de Redundâncias**
  - Informações armazenadas em um único lugar.
- **Compartilhamento de dados em um ambiente multiusuário**
  - Garantir concorrência ao acesso dos dados, sem erro.
- **Controle de Acesso**
  - Seleção de permissões por usuário.
- **Interfaceamento**
  - Facilidade para recuperação de informação.
- **Esquematização**
  - Mecanismos que possibilitem a compreensão do relacionamento entre as tabelas e sua manutenção.
- **Controle de Integridade**
  - Aplicações e acessos não podem comprometer integridade dos dados.
- **Backups**:
  - Facilidade para recuperar falhas de hardware e software.

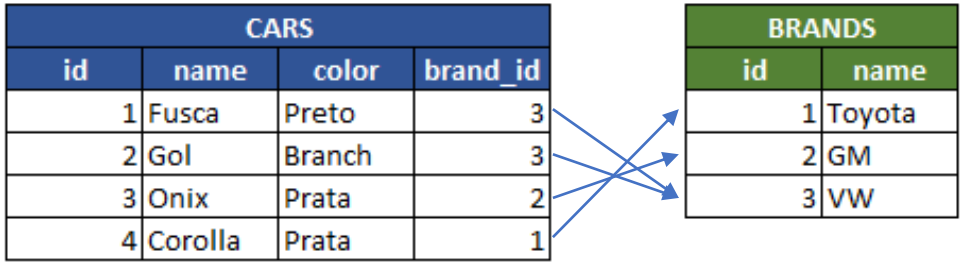
Vantagens de um SGBD

- Independência de dados;
- Consistência de dados;
- Acesso compartilhado (multiusuário e concorrente) à informação;
- Segurança;
- Controle de armazenamento, com estruturas para processamento eficiente de operações;
- Backup e recuperação de falhas;
- Múltiplas interfaces com os usuários;
- Definição e manutenção de restrições de integridade.

Tipos de bancos de dados

- **Bancos de dados relacionais**
  - Conjunto de **tabelas** com **colunas** e **linhas**;
  - **Colunas**: armazenam um tipo de dados específico (número, caractere, texto, etc...);
  - **Linhas**: representam as informações (dados) de uma determinada instância do BD;
  - Pontos de dados relacionados entre si;
  - Comumente utilizam linguagem SQL.

- **Exemplo:**



CARS			
id	name	color	brand_id
1	Fusca	Preto	3
2	Gol	Branch	3
3	Onix	Prata	2
4	Corolla	Prata	1

BRANDS	
id	name
1	Toyota
2	GM
3	VW

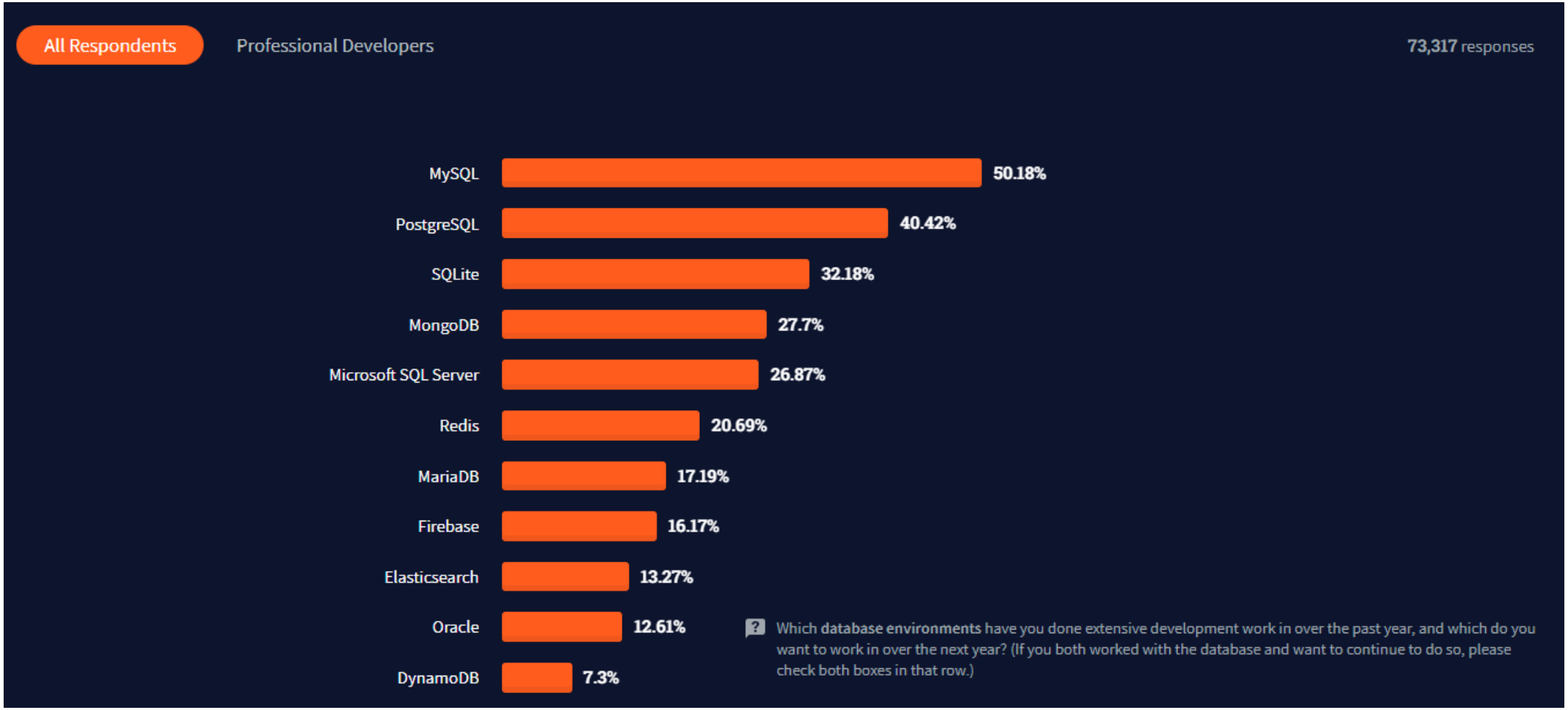
Tipos de bancos de dados

- **Outros tipos de BD:**
  - Bancos de Dados NoSQL;
  - Bancos de dados orientados a objetos;
  - Bancos de dados distribuídos;
  - Data warehouses;
  - Bancos de dados gráficos;
  - Bancos de dados OLTP;
  - Bancos de dados em nuvem.

“O melhor banco de dados para uma organização específica depende de como a organização pretende usar os dados”

Fonte: https://www.oracle.com/br/database/what-is-database/

Tipos de bancos de dados

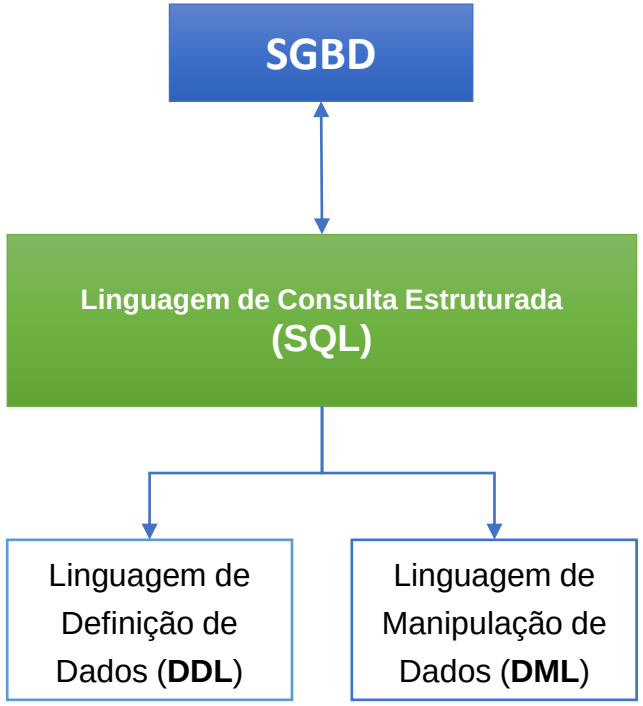


Fonte: [Stack Overflow Developer Survey, 2021]

O que é SQL?

- *Structured Query Language* (Linguagem de Consulta Estruturada);
- Desenvolvida pela primeira vez na IBM nos anos 1970;
- Usada por quase todos os bancos de dados **relacionais**;
- Linguagem declarativa;
- Utilizada para consultar, manipular, definir dados e fornecer controle de acesso;
- Embora seja amplamente utilizada, **não** é única entre os SGBDs.

Linguagens de um SGBD



"A maioria dos SGBDs usam a linguagem de consulta estruturada (SQL) para escrever e consultar dados"

Fonte: https://www.oracle.com/br/database/what-is-database/

Desafios em BD

- Absorção de aumentos significativos no volume de dados;
- Garantia da segurança de dados;
- Acompanhar a demanda;
- Gerenciamento e manutenção do BD e sua infraestrutura;
- Remoção de limites na escalabilidade.

Gustavo Viais
Tech Lead

Introdução

Banco de dados / SQL

.StartSe
<tech/academy>

Modelo Entidade-Relacionamento

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

Modelo Entidade-Relacionamento

- Criado por Peter Chen em 1976;
- Modelo Entidade-Relacionamento (MER):
  - “ *Modelo conceitual para descrever objetos (**entidades**) pertencentes a um domínio de negócios, com seus respectivos **atributos** e como eles **relacionam** entre si* ”;
  - Abstração da estrutura do BD.
- Representação gráfica das informações por meio do **Diagrama Entidade-Relacionamento (DER).**

MER - Entidades

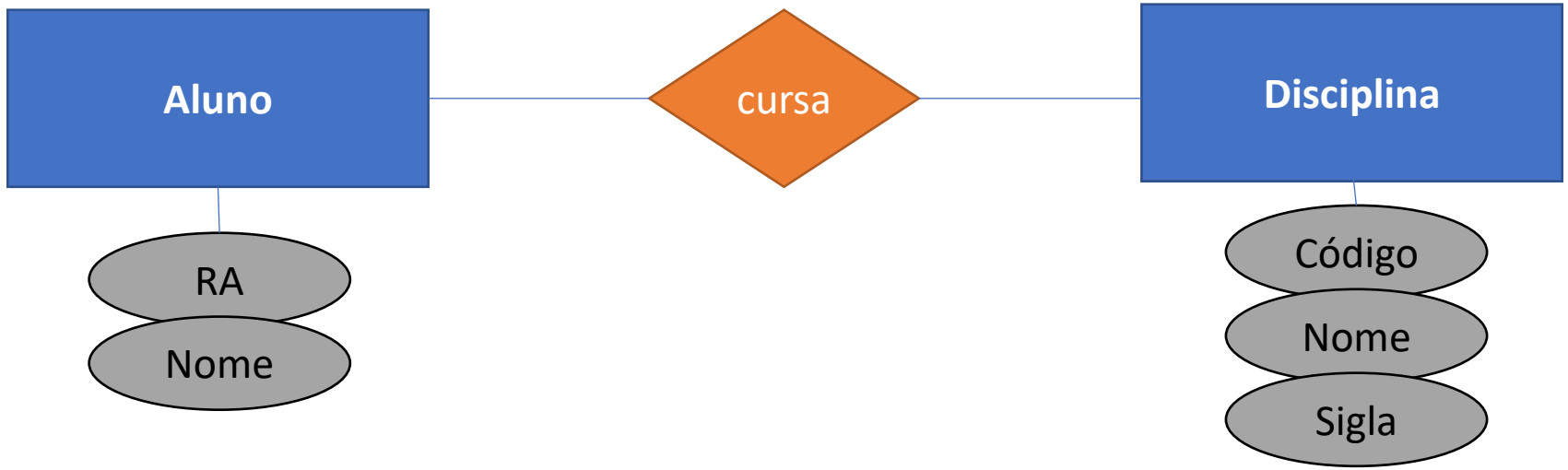
- “Coisas” do mundo real que possuem existência independente:
  - **Entidades físicas**: realmente tangíveis e visíveis no mundo real. Ex: pessoa, carro, produto, etc...
  - **Entidades lógicas**: existência conceitual, geralmente decorrente da interação entre entidades físicas. Ex: compra, venda, transação, etc...
- **DER** → representada por **retângulo**:

MER - Relacionamento

- Relação existente entre entidades do mundo real;
- Possibilita entender como uma entidade se comporta em relação às demais;
- **DER** → representado por **losango**:

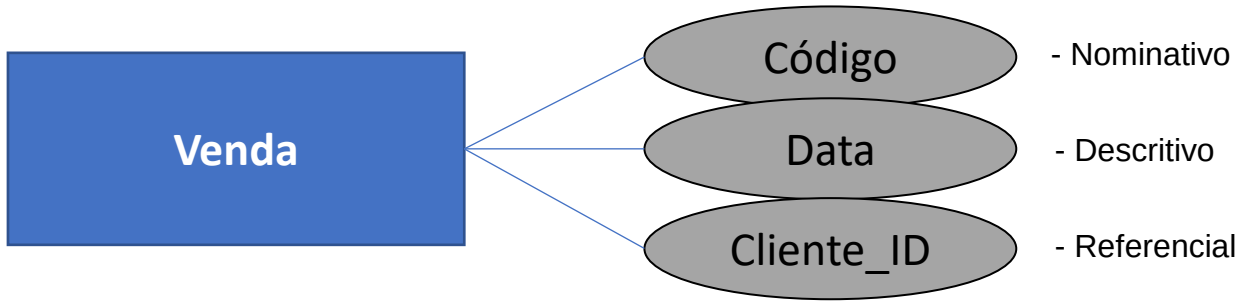
MER - Atributos

- Valores que representam as **propriedades** das entidades e dos relacionamentos no mundo real:
  - Atributos de entidades;
  - Atributos de relacionamento.
- **DER** → representado por **elipses**:



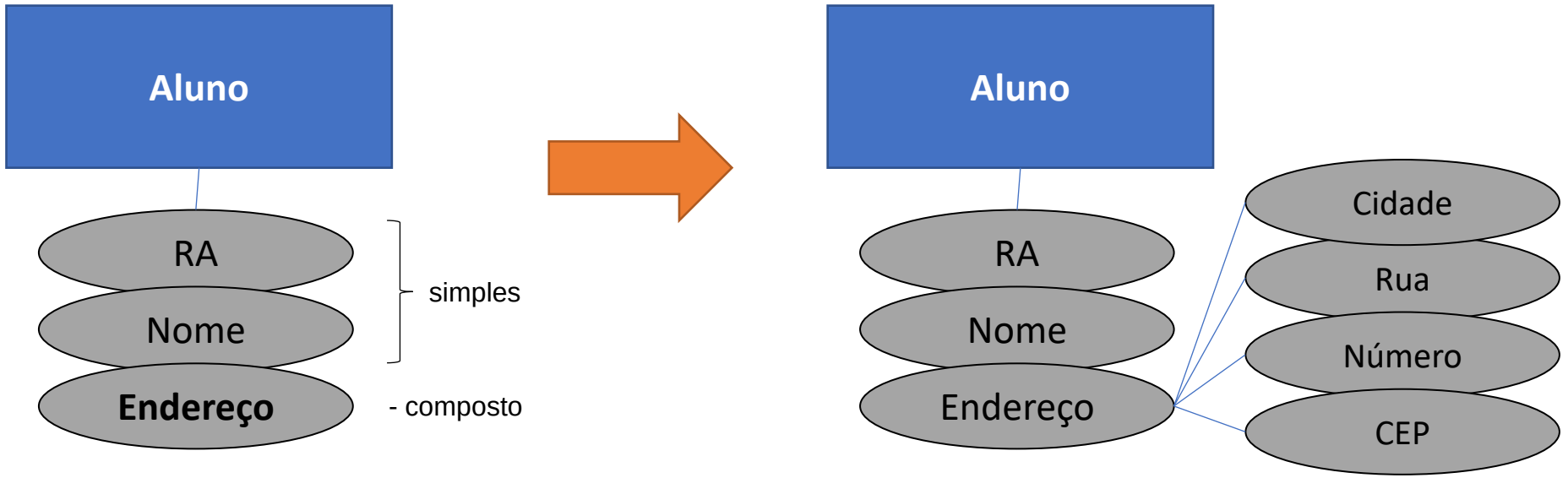
MER - Atributos

- **Classificação → Descritivos, Nominativos, Referenciais**
  - **Descritivos**: representam características intrínsecas. Ex: data, cor, gênero;
  - **Nominativos**: além de serem também descritivos, estes têm a função de definir e identificar um objeto. Ex: RA, código, sigla;
  - **Referenciais**: representa o vínculo com a entidade que está relacionada. Ex: venda com o ID do cliente que está relacionado.



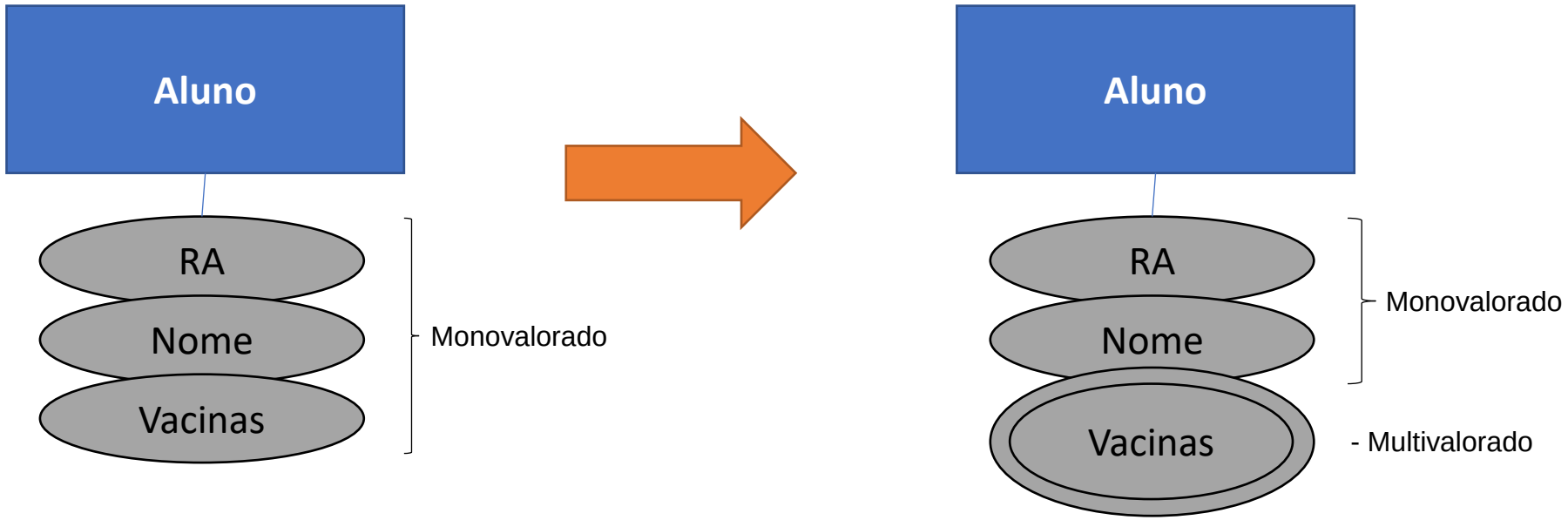
MER - Atributos

- **Tipos de atributos → Simples** vs **Composto**
  - **Simples**: atributo atômico; não divisível;
  - **Composto**: pode ser dividido em subatributos.



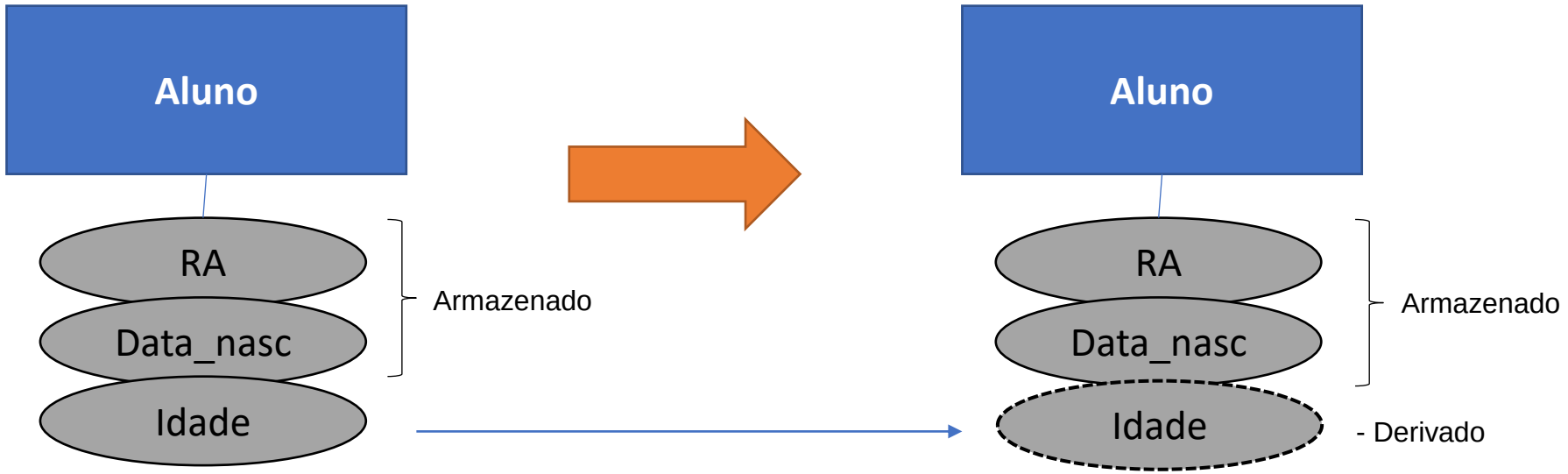
MER - Atributos

- **Tipos de atributos → Monovalorado** vs **Multivalorado**
  - **Monovalorado**: assume apenas um valor para uma entidade ou relacionamento;
  - **Multivalorado**: pode assumir mais de um valor.



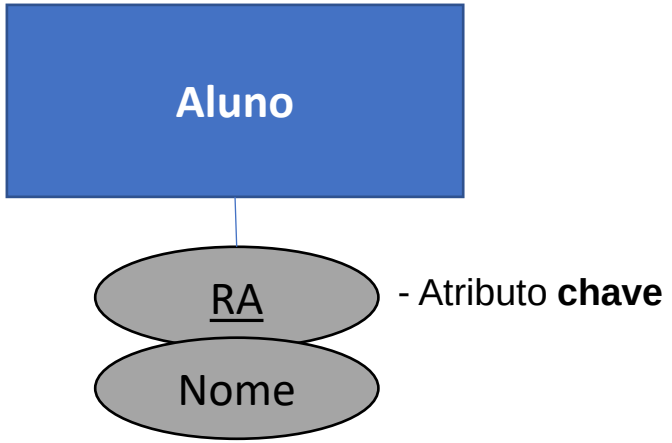
MER - Atributos

- **Tipos de atributos → Armazenado** vs **Derivado**
  - **Armazenado**: atributo próprio da entidade;
  - **Derivado (calculado)**: valor pode ser obtido por meio de outros atributos.



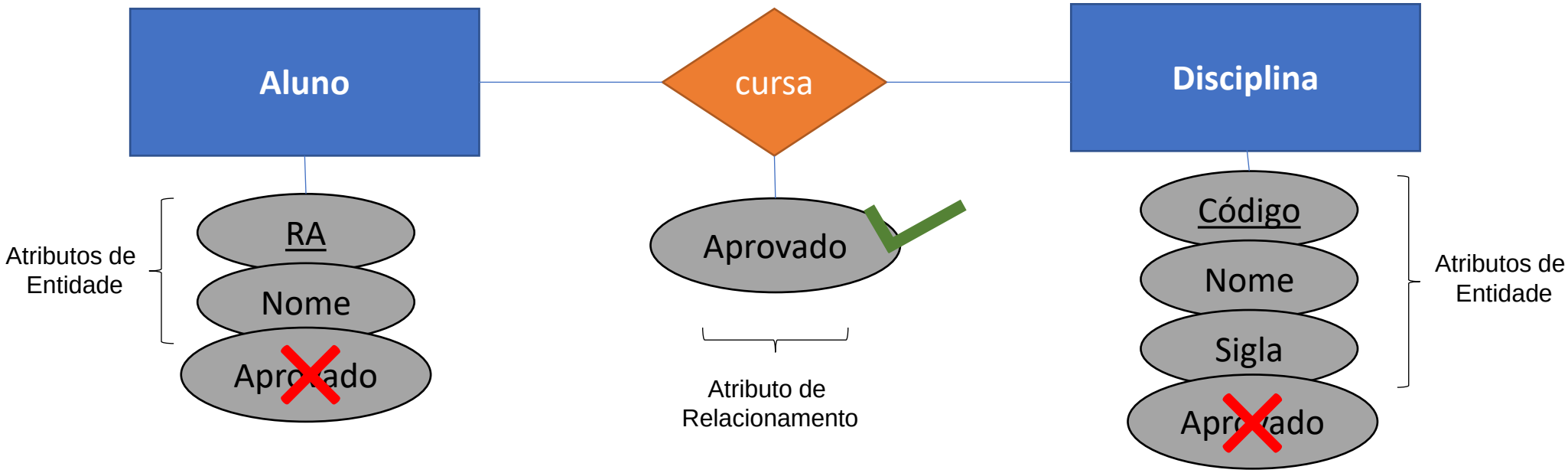
MER - Atributos

- **Tipos de atributos → Chave**
  - **Restrição de unicidade**: entidades devem conter ao menos um atributo cujo valor **identifique unicamente** o respectivo registro (instância);
  - **Principal** (mas não único) meio para consultar registros de uma entidade;
  - **DER**: representado por atributo sublinhado.



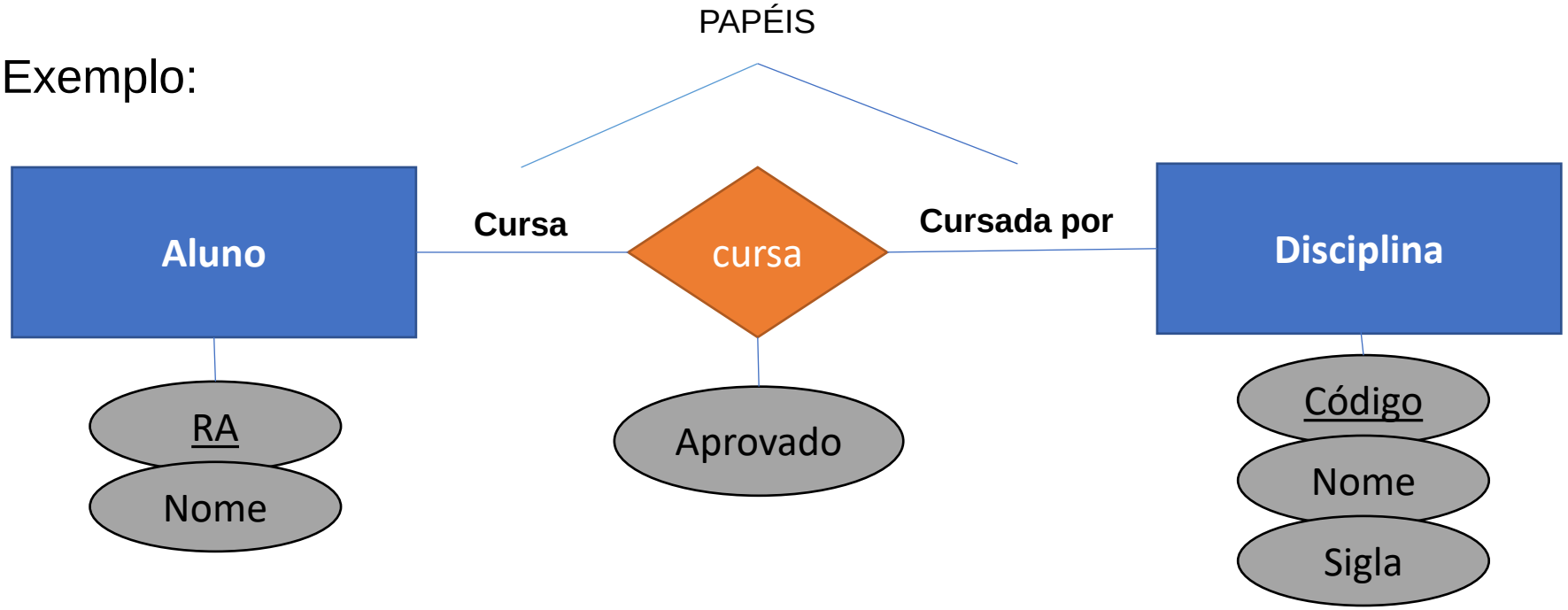
MER - Atributos

- **Atributos de Entidade** vs **Atributos de Relacionamento**
  - **Exemplo**: informação se o **aluno** foi **aprovado** em uma determinada **disciplina**.



MER – Papéis no Relacionamento

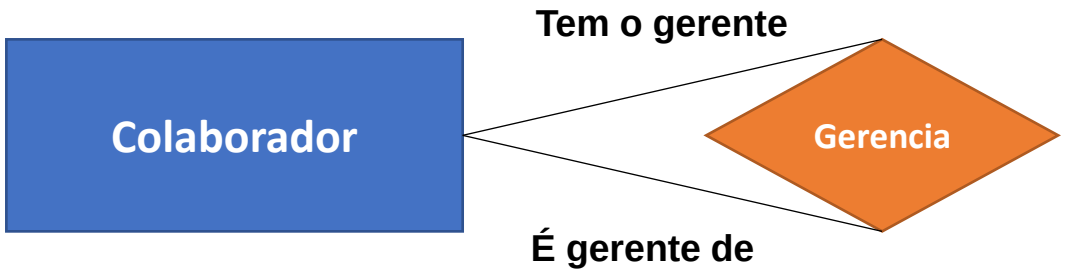
- Cada **entidade** no modelo relacional possui um **PAPEL**;
- Indicação no **DER** é opcional, porém pode facilitar o entendimento/interpretação;
- **Sugestão**: evidenciar papéis nos casos em que houver ambiguidade na interpretação.
- Exemplo:



MER – Auto relacionamento

- Quando uma mesma **entidade** possui mais de um papel em um mesmo **relacionamento**.

- Exemplo:



- **João** tem o gerente **José**;
- **José** é gerente de **João**;

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Modelo Entidade-Relacionamento

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Cardinalidade

Banco de dados / SQL

Cardinalidade

- Demonstra o número (mínimo e máximo) de entidades ao qual outra entidade pode estar relacionada;
- **DER**: ([min], [max]);
- **Exemplo**:

- Um **cliente** possui quantas **contas bancárias**?
  Mínimo **uma** e no máximo **N**

- Uma **conta bancária** está associada a quantos **clientes**?
  Mínimo **um** e no máximo **um**

Cardinalidade

- **Cardinalidade Mínima**
  - Número **mínimo** de ocorrências na qual uma entidade está relacionada a uma ocorrência de outra entidade;
  - Especifica se participação das ocorrências das entidades no relacionamento é **opcional** ou **obrigatória**;
  - Cardinalidade mínima **0** é denominada associação **opcional;**
  - Cardinalidade mínima **1** é denominada associação **obrigatória;**

***Cardinalidade mínima nem sempre é especificada no DER**

Cardinalidade

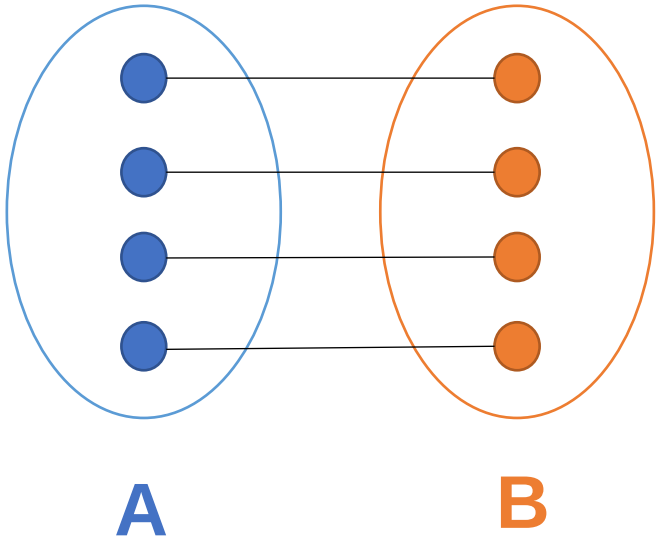
- **Cardinalidade Máxima**
  - Especifica a quantidade **máxima** de ocorrências de entidades que podem estar associadas a uma ocorrência da outra entidade (**1** ou **N**);
  - Cardinalidade **1** representa, no **máximo,** uma única ocorrência entre as entidades;
  - Cardinalidade máxima "**muitos**" é representada por **N**;

Cardinalidade

- **Combinações:**
  - Um-para-um (*oneToOne*);
  - Um-para-muitos (*oneToMany*);
  - Muitos-para-um (*ManyToOne*);
  - Muitos-para-muitos (*manyToMany*).

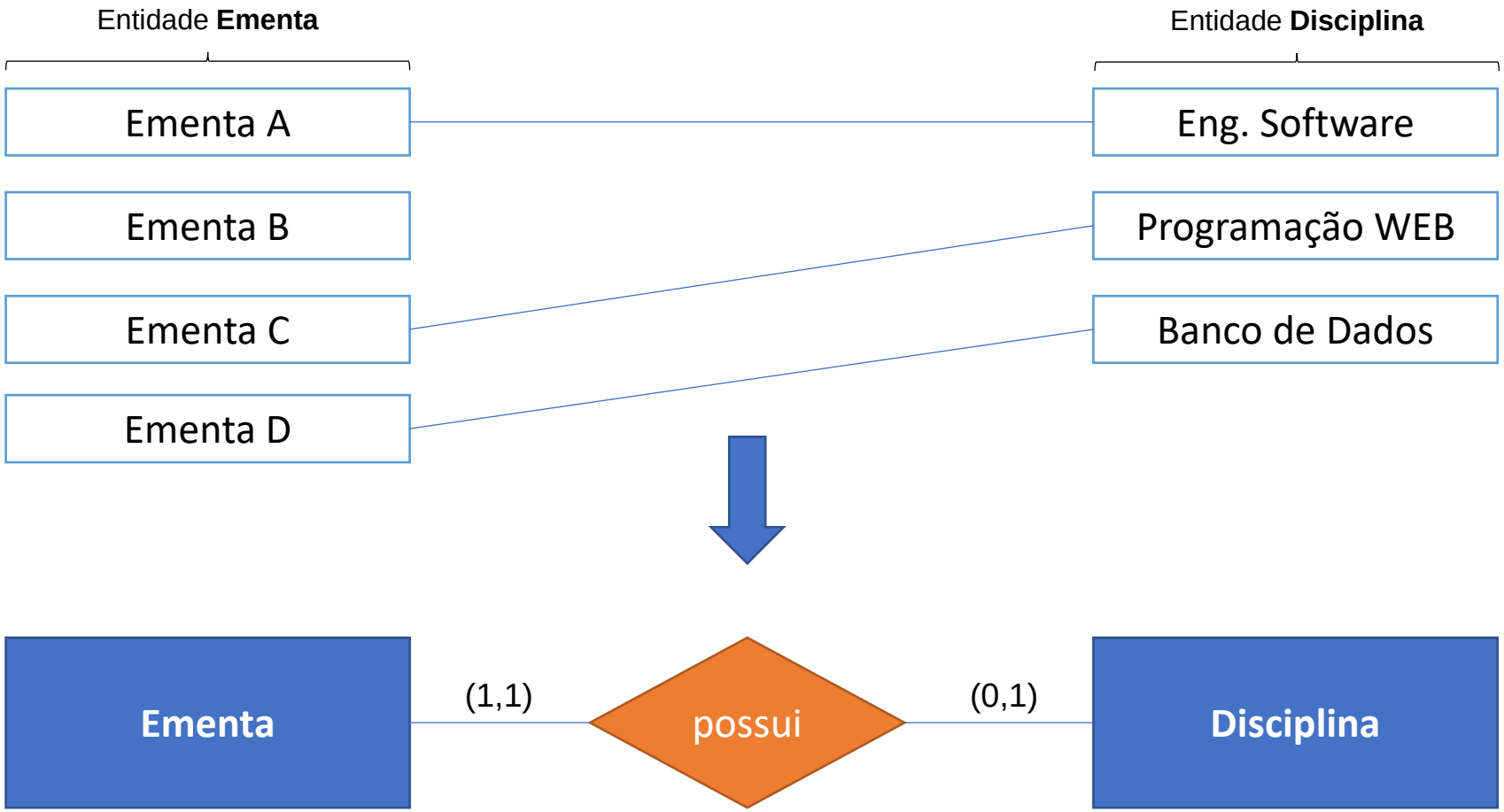
Cardinalidade Um-para-um

- **Uma** entidade **A** está associada no máximo a **uma** entidade **B**;
- **Uma** entidade **B** está associada no máximo a **uma** entidade **A**.



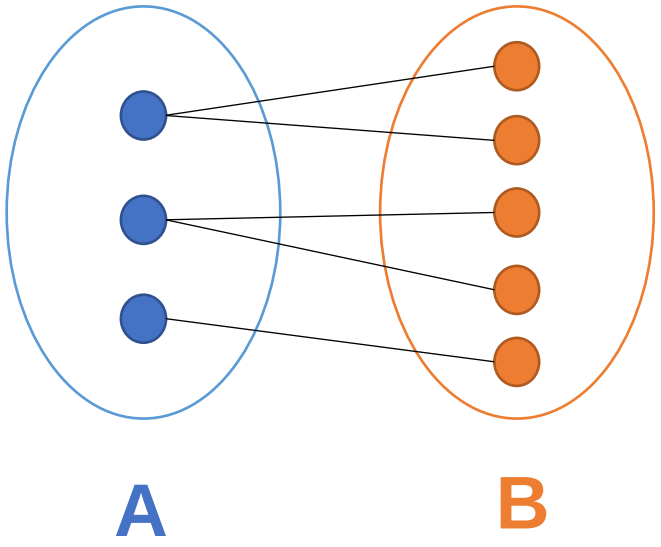
Cardinalidade Um-para-um

- **Exemplo:**



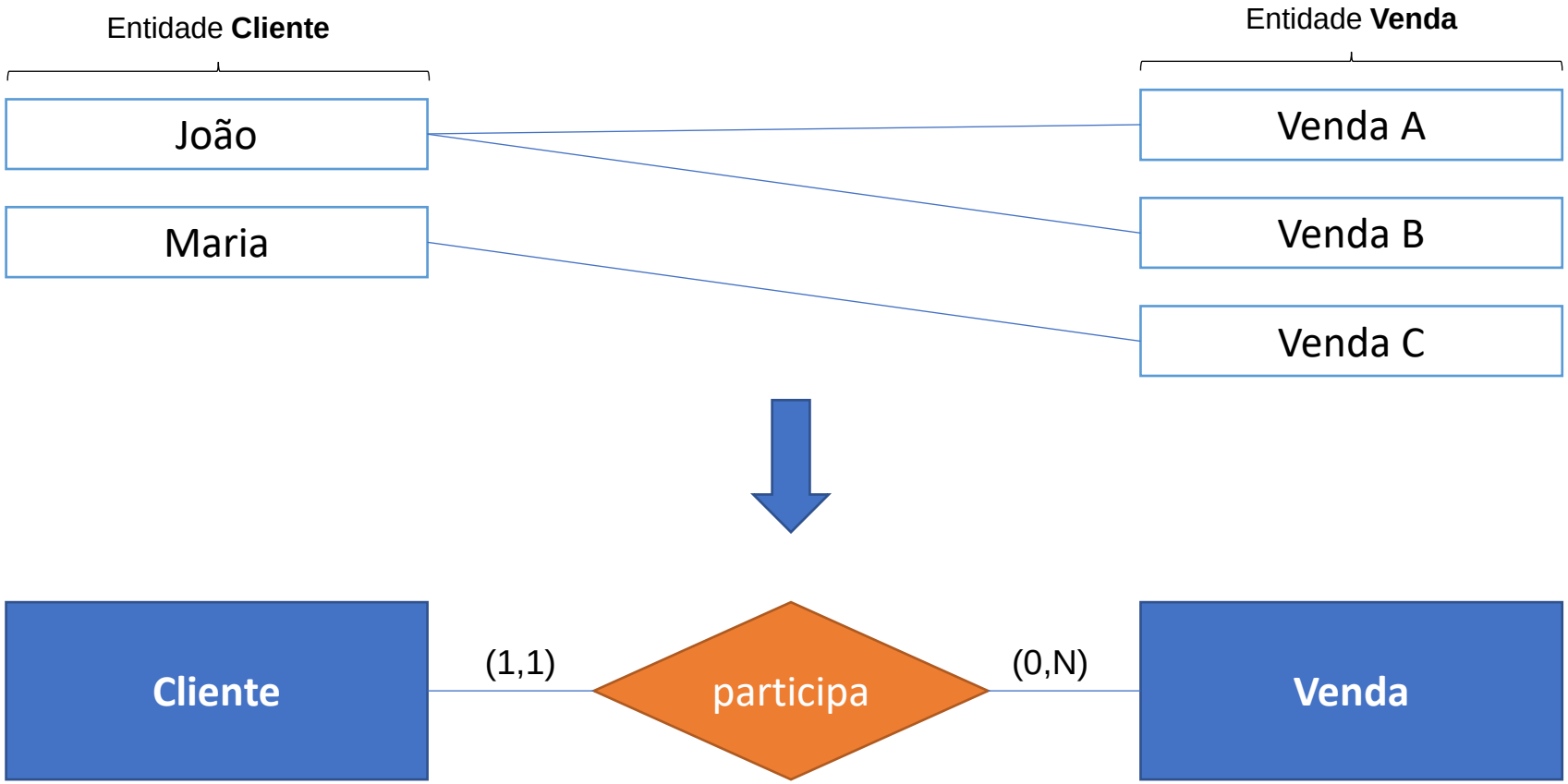
Cardinalidade Um-para-muitos

- **Uma** entidade **A** está associada a qualquer número de entidades **B;**
- **Uma** entidade **B** pode estar associada, no máximo, a **uma** entidade de **A**.



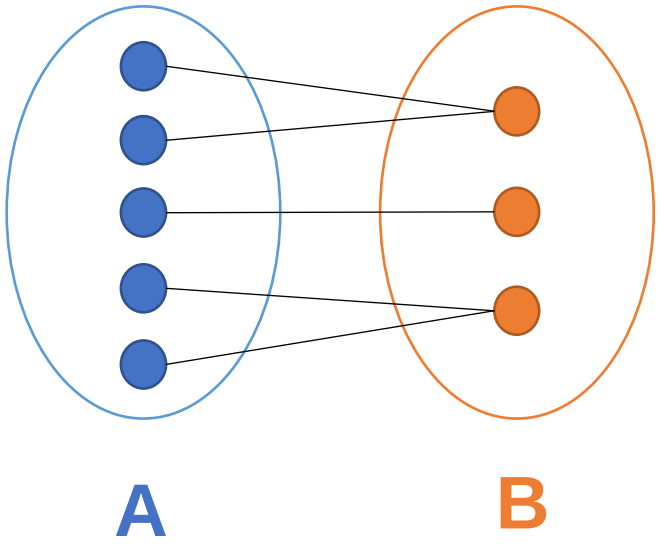
Cardinalidade Um-para-muitos

- **Exemplo:**



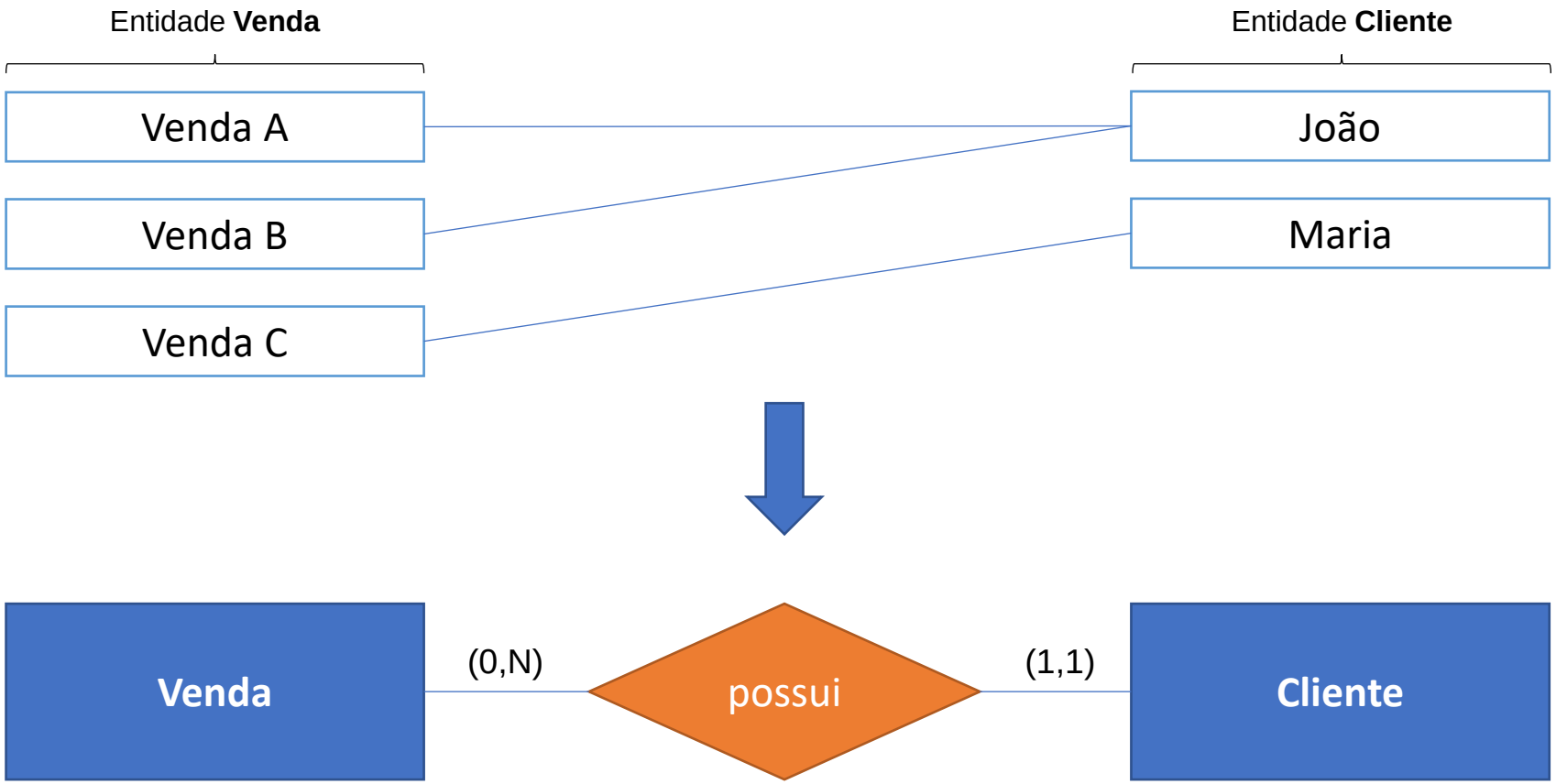
Cardinalidade Muitos-para-um

- **Uma** entidade **A** está associada, no máximo, a **uma** entidade **B;**
- **Uma** entidade **B**, no entanto, pode estar associada a qualquer número de entidades **A**.



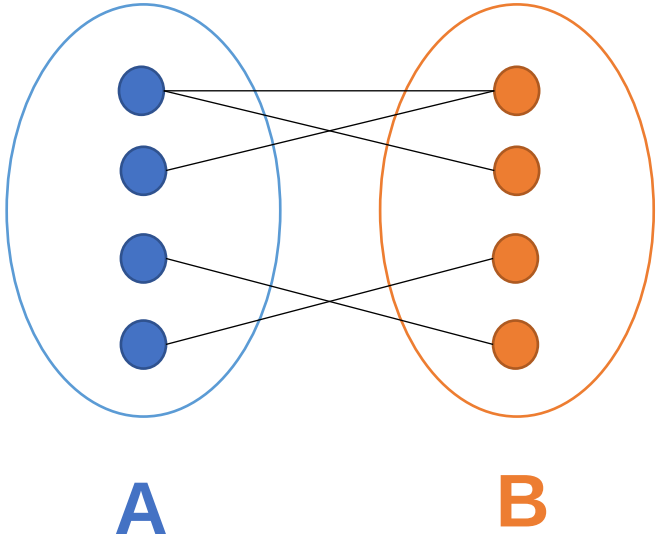
Cardinalidade Muitos-para-um

- **Exemplo:**



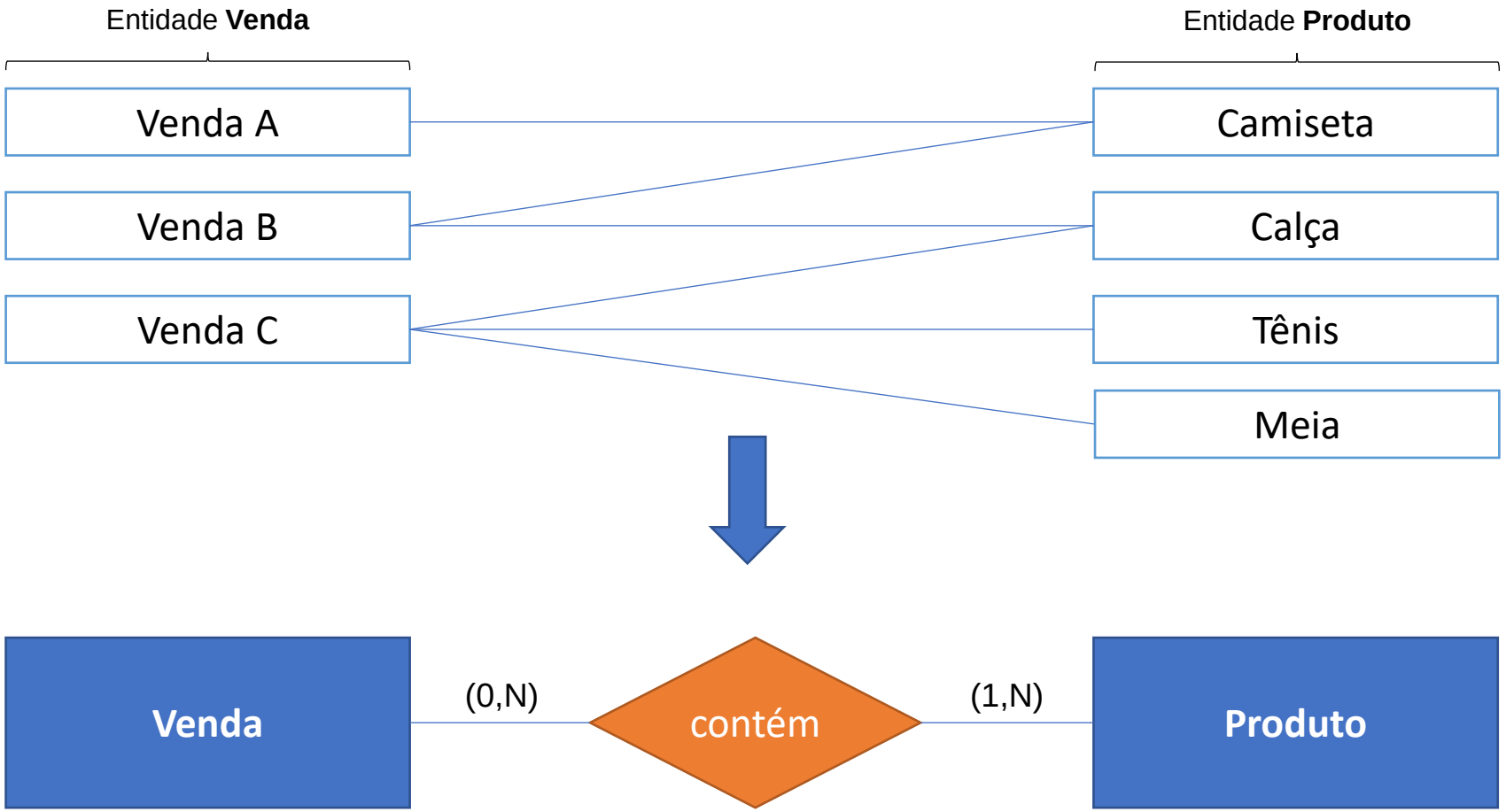
Cardinalidade Muitos-para-muitos

- **Uma** entidade **A** está associada a qualquer número de entidades de **B**;
- **Uma** entidade **B** está associada a qualquer número de entidades de **A.**



Cardinalidade Muitos-para-muitos

- **Exemplo:**



Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Cardinalidade

Banco de dados / SQL

.StartSe
<tech/academy>

Modelo relacional

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

Modelo Relacional

- Criado por Edgar F. Codd nos anos 70;

- **Conjunto de dados**
  - Entidades e relações → **Tabelas;**
  - Atributos → **Colunas** ou **campos**;

- **Operações** fundamentadas na teoria de conjuntos e álgebra
  - Projeção, produto cartesiano, seleção, junção, união e subtração;

Características

- **Dados organizados**
  - Tabelas, colunas, relacionamento, chaves, etc...
- **Integridade**
  - Restrições para dados e relacionamentos;
  - Ex: integridade **referencial** e integridade de **domínio**.
- **Manipulação**
  - Linguagens formais (álgebra e cálculo relacional) e SQL.

Organização

- Segmentado em cinco conceitos
  - **Domínio;**
  - **Atributo;**
  - **Tupla;**
  - **Relação;**
  - **Chave.**

Domínio

- **Conjunto de valores atômicos** permitidos para um dado;
- **Tipos (formatos)**
  - *string*, inteiro, data, etc...
- **Exemplos**
  - Domínio de **CPF** → *string* com exatamente 11 caracteres;
  - Domínio **Número da matrícula** → *inteiro* maior que 0;
  - Domínio **Data de Nascimento** → *data* menor ou igual a data atual.

Atributo

- **Item de dado** do BD;
  - Cabeçalho de cada coluna;
- **Atributo:** nome + domínio
- **Exemplo**
  - CPF: *char(11)*
  - matrícula: *integer*
  - data_nascimento: *date*

CPF *char(11)*	**matricula** *integer*	**data_nascimento** *date*

Tupla

- Conjunto de pares (**atributo**, **valor**)
  - Define uma **ocorrência** de um **fato** do mundo real ou de um **relacionamento** entre fatos.

- **Exemplo**
  - Aluno: **{ (nome, ‘Maria’), (CPF, 64730931092), (matrícula, 1234), (data_nascimento, ‘1992-07-10’) }**

	nome	cpf	matrícula	data_nascimento
Tupla 1	Maria	64730931092	1234	1992-07-10
Tupla 2	José	72594160091	1235	2000-01-15
Tupla 3	João	87317448001	1264	2001-10-02

Relação

- Composto por um **cabeçalho** e um **corpo**
- **Cabeçalho**
  - Número fixo de atributos (grau da relação);
  - Atributos não-ambíguos
- **Corpo**
  - Número variável de tuplas (cardinalidade da relação);
  - Ordem não é relevante

Relação

nome	cpf	matrícula	data_nascimento
Maria	64730931092	1234	1992-07-10
José	72594160091	1235	2000-01-15
João	87317448001	1264	2001-10-02

Cabeçalho

Corpo

Chave

- **Superchave**: conjunto de **atributos** que identifique **unicamente** uma tupla em uma relação
  - **Exemplos**
    - SC (Aluno) = {RA, Nome}
    - SC (Aluno) = {RA}
    - SC (Aluno) = {CPF}
    - SC (Aluno) = {CPF, Nome, Endereço}

- **Chave**: superchave com atributos **mínimos** que identifiquem uma tupla em uma relação.

Chave

- **Chave simples:**
  - Tupla pode ser identificada unicamente por meio de **um** atributo chave;
  - Exemplos: ID, CPF, RG, etc...

- **Chave Composta**
  - Identificação única de uma tupla com **dois ou mais** atributos;
  - Exemplos
    - Cidade → {Nome, Estado}
    - Voo → {Número, Data}

Chave

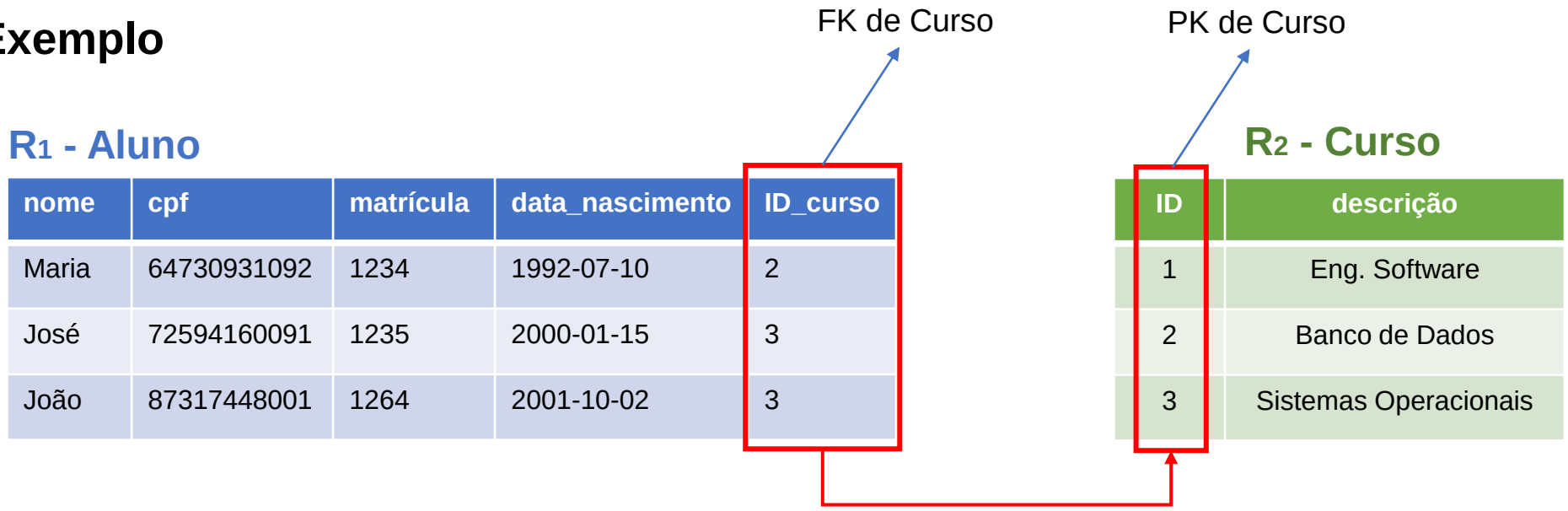
- **Chave Candidata**: possibilidade de existir mais uma chave para a mesma relação
  - **Exemplos**
    - C (Aluno) = {CPF}
    - C (Aluno) = {RA}

- **Chave PRIMÁRIA (PK)**: escolhida entre as chaves candidatas
  - Atributo é representado por sublinhado
  - **Exemplo**
    - **C (Aluno) = { CPF }** ← **Chave primária**
    - C (Aluno) = { RA } ← **Chave secundária**

Chave

- **Chave Estrangeira (FK)**
  - Atributo(s) de uma relação R_1 que consistem uma equivalência de valor com a PK de uma relação R_2;
  - Devem possuir o mesmo domínio (restrição de integridade).

- **Exemplo**



R_1 - Aluno

nome	cpf	matrícula	data_nascimento	ID_curso
Maria	64730931092	1234	1992-07-10	2
José	72594160091	1235	2000-01-15	3
João	87317448001	1264	2001-10-02	3

R_2 - Curso

ID	descrição
1	Eng. Software
2	Banco de Dados
3	Sistemas Operacionais

Esquema de Relação

- Expressão da forma **R (A_1, A_2, ..., A_n)**
  - **R**: nome da relação;
  - **Ai**: nome do atributo;
  - **n**: grau da relação.

- **Exemplo**

Estudante (<u>ra</u>, nome, endereco, data_nascimento, id_curso)

Curso (<u>id</u>, descricao)

.StartSe
<tech/academy>

Modelo relacional

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Mapeamento

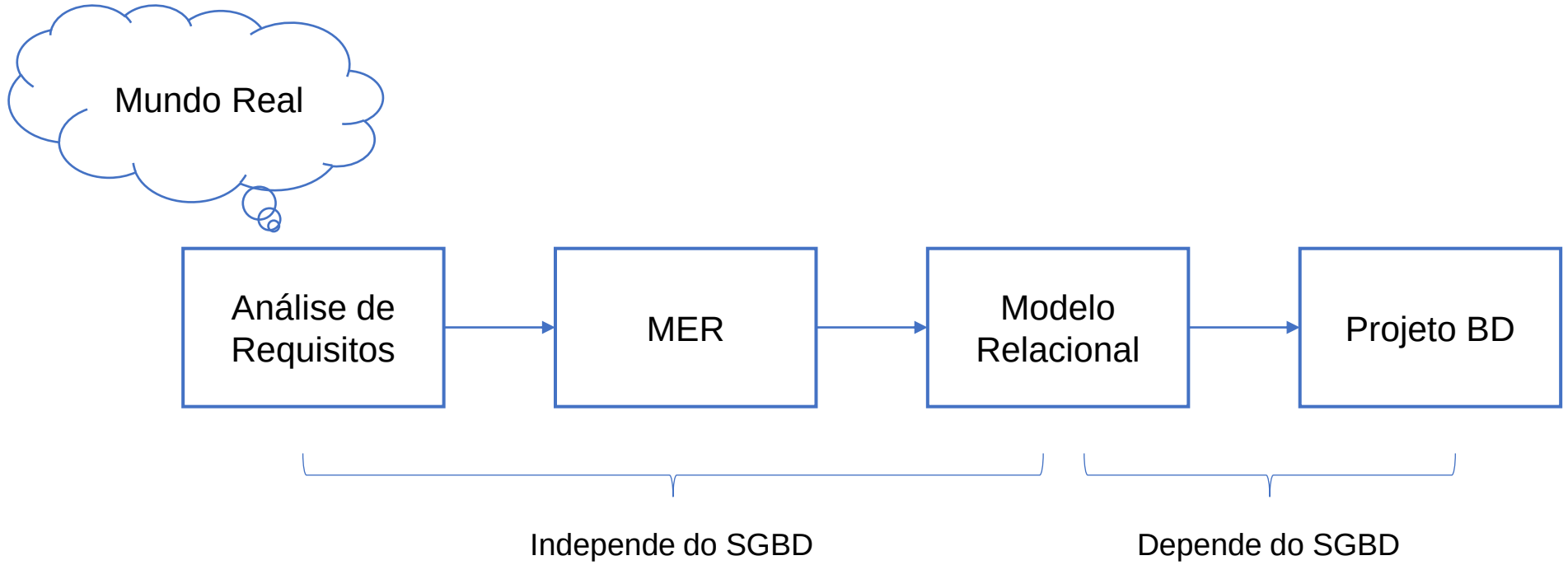
Banco de dados / SQL

Mapeamento entre Esquemas

- **MER** → modelo conceitual
- **Modelo Relacional** → modelo de implementação

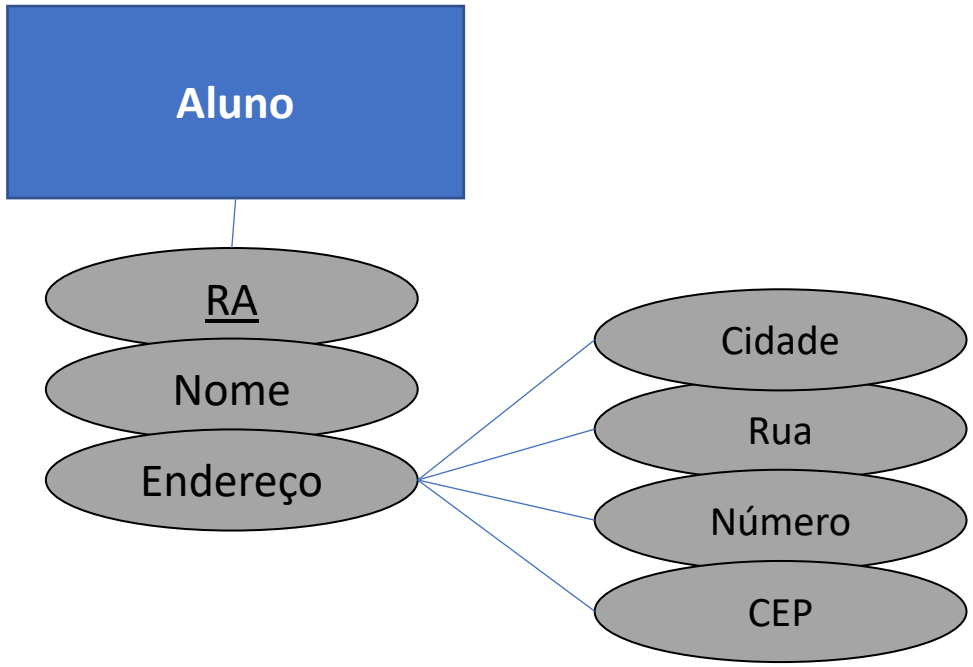
- **Mapeamento**
  - “Traduzir” representação do MER para o Modelo Relacional preservando as propriedades do modelo conceitual.

Mapeamento entre Esquemas



Mapeamento – Atributos Compostos

- São adicionados os atributos componentes (e não o atributo composto);
- **Exemplo**

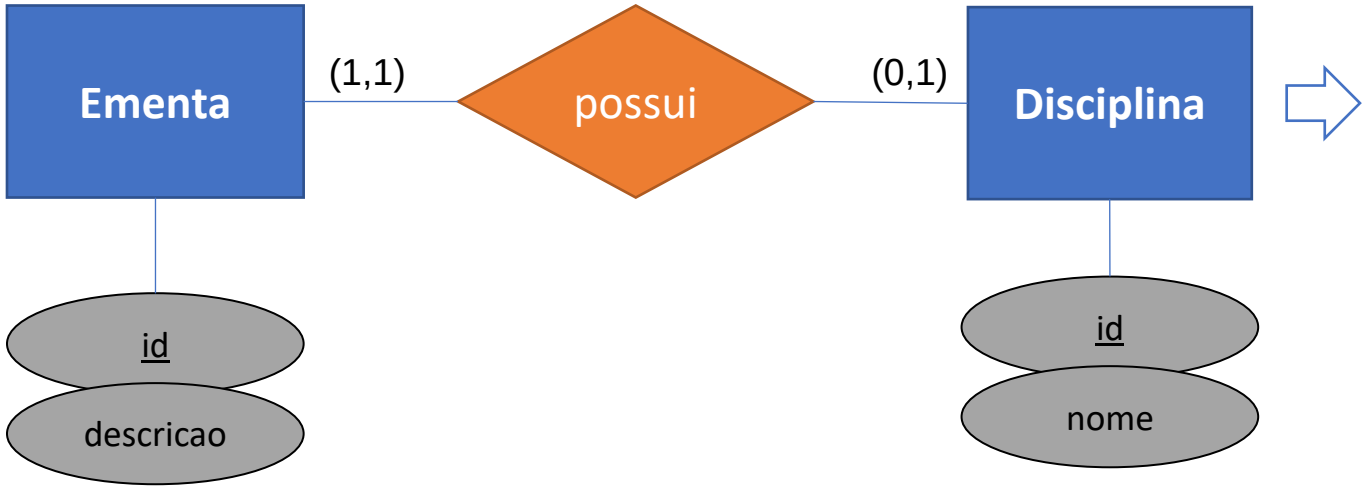


Aluno = { RA, Nome, Cidade, Rua, Número, CEP }

Mapeamento – Cardinalidade 1:1

- **Três opções**
  1. Adiciona-se a **PK** da Entidade 1 (**E1**) na Entidade 2 (**E2**) e vice-versa;
  2. Adiciona-se a **PK** de **E1** em **E2**;
  3. Adiciona-se a **PK** de **E2** em **E1**.

- **Exemplo**



① Ementa = { id, descricao, **id_disciplina** }
Disciplina = { id, nome, **id_ementa** }

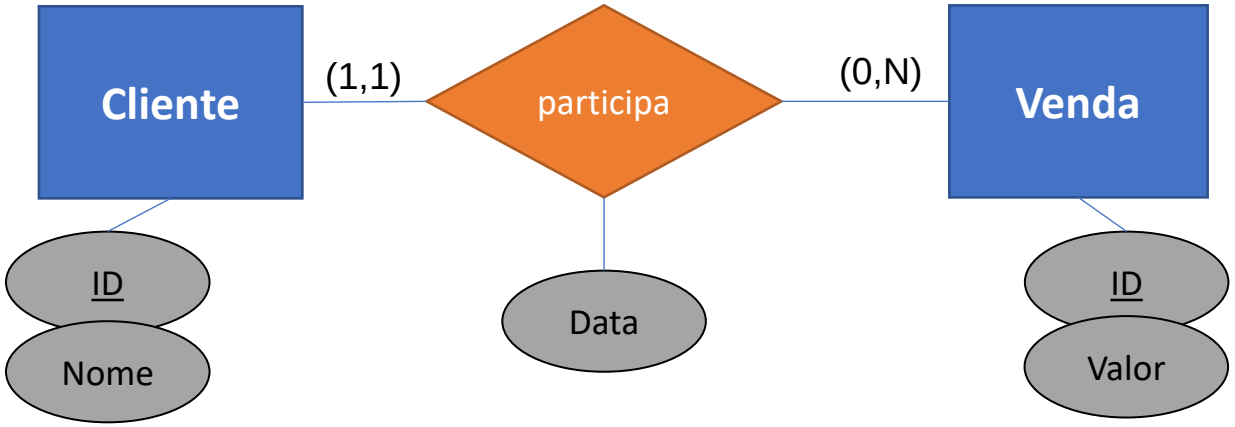
② Ementa = { id, descricao }
Disciplina = { id, nome, **id_ementa** }

③ Ementa = { id, descricao, **id_disciplina** }
Disciplina = { id, nome }

Mapeamento – Cardinalidade 1:N

- **Adiciona-se a PK de E_1 em E_2**
  - **E_1** será representada somente pelos atributos de **E_1**;
  - **E_2** será representada por:
    - Atributos de **E_2**;
    - PK de **E_1** (chave estrangeira – FK);
    - Atributos do relacionamento.

- **Exemplo**



Cliente = { id, nome }

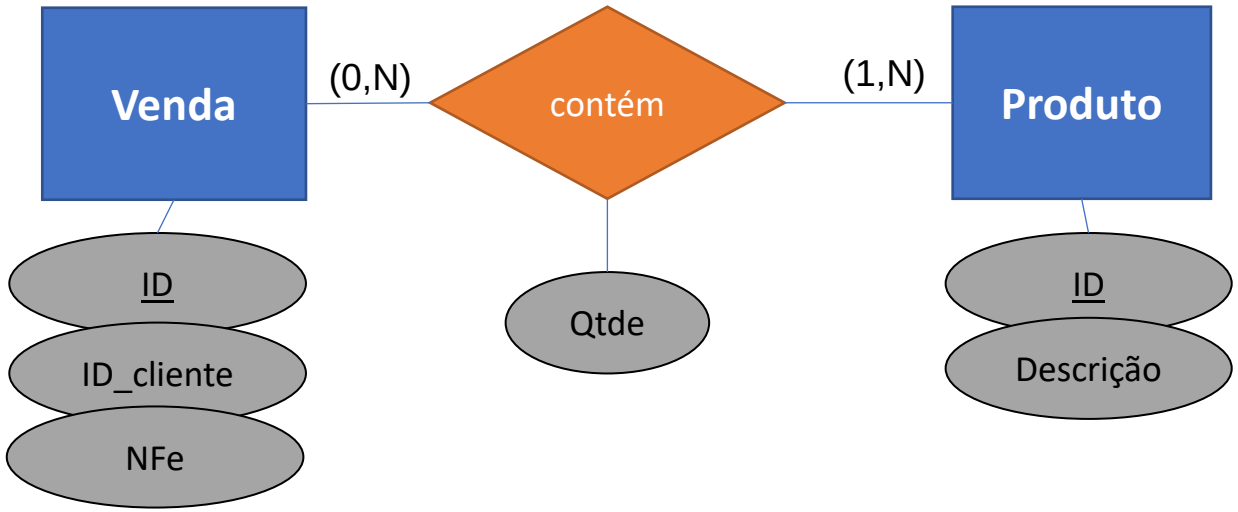
Venda = { id, valor, **id_cliente**, **data** }

Mapeamento – Cardinalidade N:N

- **E_1** será representada somente pelos atributos de **E_1**;
- **E_2** será representada somente pelos atributos de **E_2**;
- Será adicionada uma terceira tabela (**E_R**) referente ao relacionamento entre **E_1** e **E_2**, a qual será representada por:
  - Chave primária de **E_1** (FK);
  - Chave primária de **E_2** (FK);
  - Atributos do relacionamento.
- Chave primária de **E_R** será representada por:
  - PK de **E_1** + PK de **E_2**

Mapeamento – Cardinalidade N:N

- **Exemplo Mapeamento N:N**



Venda = { id, id_cliente, NFe }

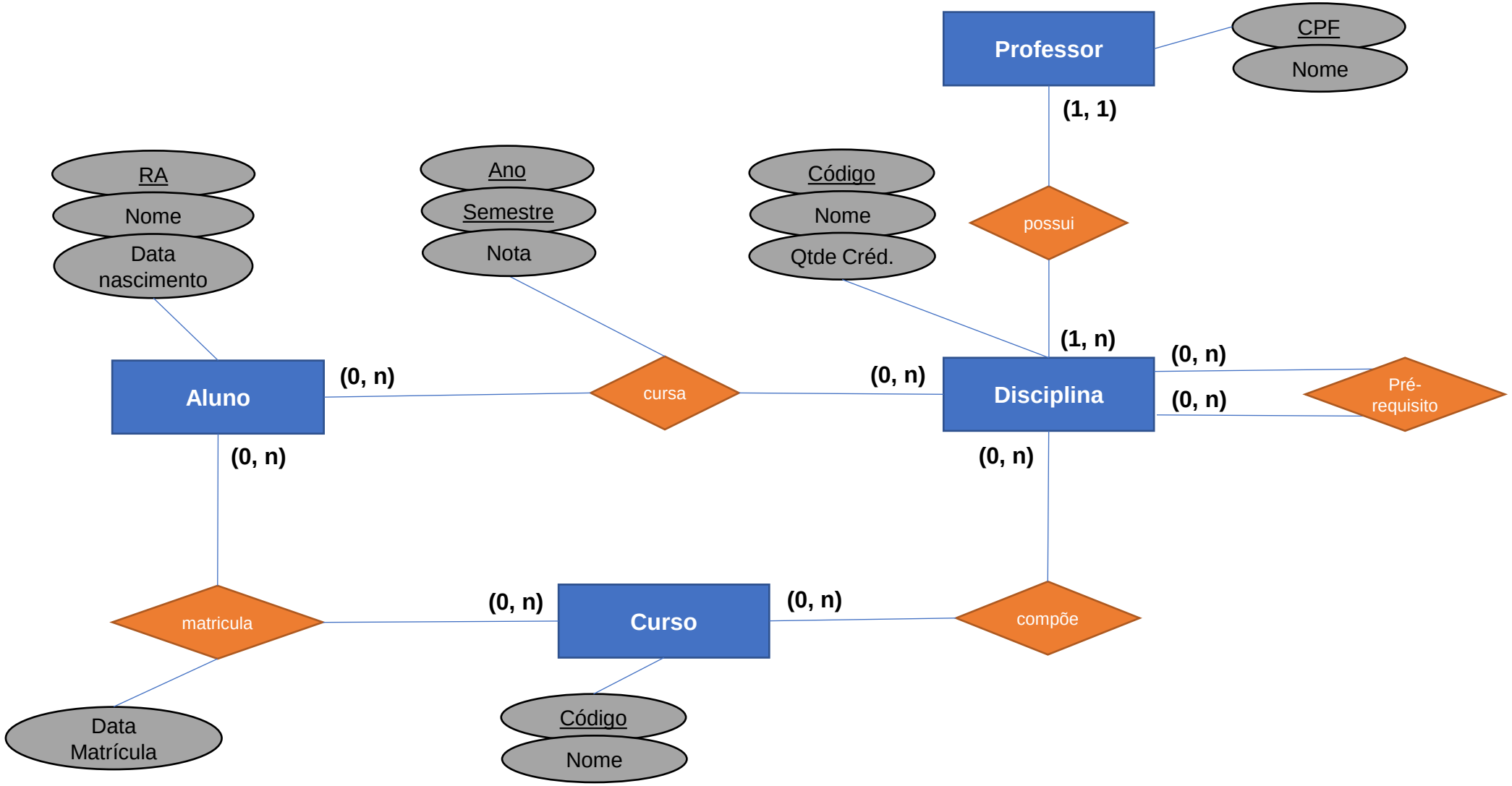
Venda_Produto = { id_venda, id_produto, qtde }

Produto = { id, valor, descricao}

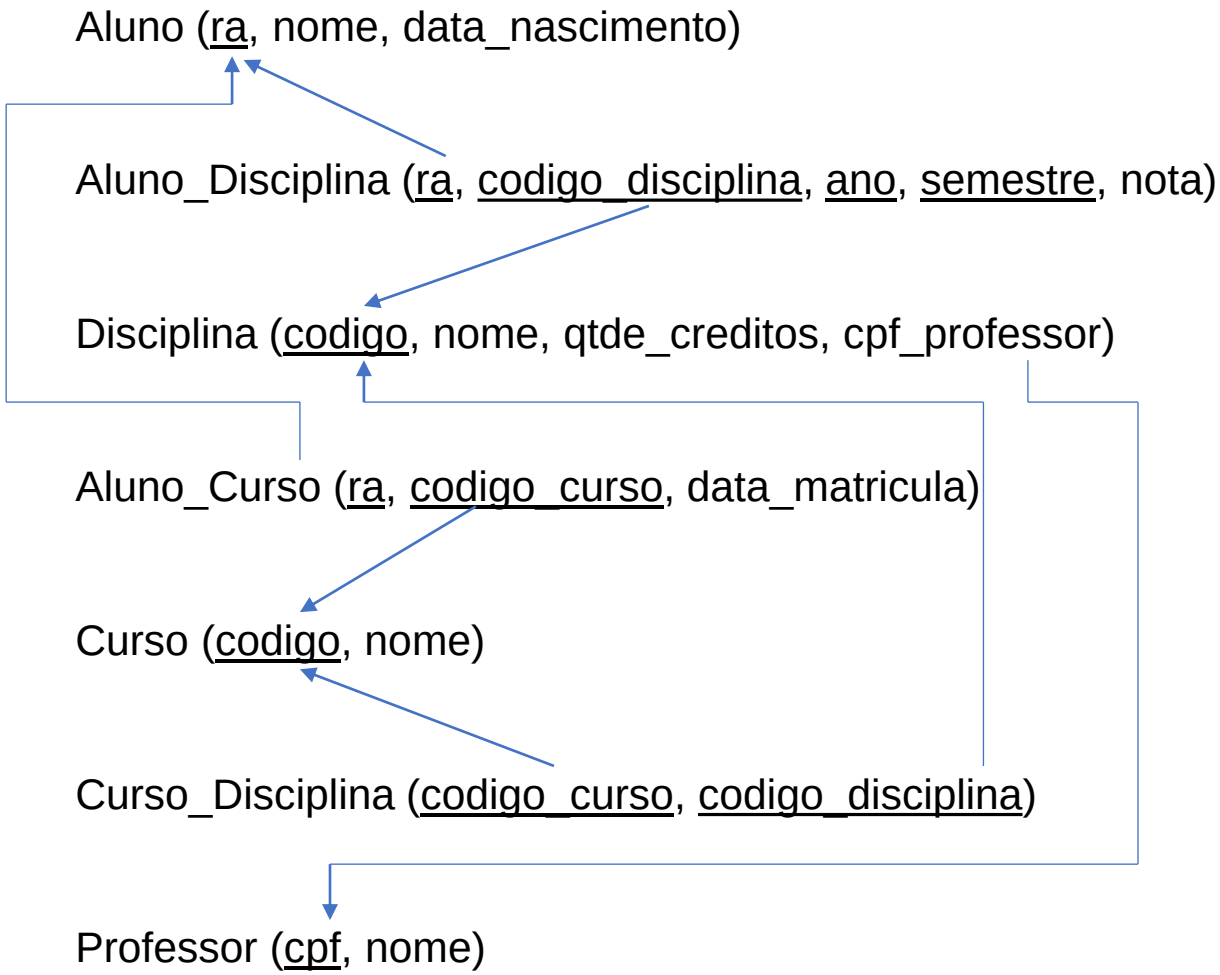
Sugestão: padronizar nomenclatura **ER**

Exemplo: Nome de **E1** + "_" + Nome de **E2**.

Exercício DER



Exemplo Modelo Relacional - Universidade



Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Mapeamento

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Preparando o ambiente

Banco de dados / SQL

MySQL

.StartSe
<tech/academy>

https://dev.mysql.com/downloads/windows/installer/8.0.html

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Preparando o ambiente

Banco de dados / SQL

.StartSe
<tech/academy>

Linguagem de Definição de Dados

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

SQL

- Dois conjuntos de comandos **principais**
  - **Linguagem de Definição de Dados**
    - *Data Definition Language (DDL);*
    - Conjunto de comandos para **definição** e **gerenciamento** do **esquema** do BD.
  - **Linguagem de Manipulação de Dados**
    - *Data Manipulation Language (DML);*
    - **Consulta**, **inserção**, **atualização** e **remoção** relacionados às **instâncias** do BD.

Linguagem de Definição de Dados

- **Principais comandos:**
  - *CREATE*: utilizado para criar objetos, como por exemplo um banco de dados e/ou uma tabela;
  - *ALTER*: usado para modificar e renomear elementos de uma tabela em um BD existente;
  - *DROP*: usado para remover um banco de dados inteiro ou uma tabela de banco de dados;
  - *TRUNCATE*: utilizado para remover todos os registros de uma tabela de banco de dados.

Linguagem de Definição de Dados

- **Criar um novo banco de dados**

```
CREATE DATABASE my_database;
```

Linguagem de Definição de Dados

- **Criar uma nova tabela**

```
CREATE TABLE my_table (
    atributo1 tipo <restrições do atributo 1> ,
    atributo2 tipo <restrições do atributo 2> ,
    ...
    atributoN tipo <restrições do atributo N> ,
    <restrições da tabela>
);
```

Linguagem de Definição de Dados

- **Tipos de Dados (*string*)**

Data Type	Description
CHAR(size)	Fixed size. Maximum size of 255 characters.
VARCHAR(size)	Variable-length strings of 65353 bytes.
TINYTEXT(size)	Maximum size of 255 characters.
TEXT(size)	Maximum size of 65,535 characters.
MEDIUMTEXT(size)	Maximum size of 16,777,215 characters.
LONGTEXT(size)	Maximum size of 4GB or 4,294,967,295 characters.

```
CREATE TABLE my_table (
    atributo1 tipo <restrições do atributo 1> ,
    atributo2 tipo <restrições do atributo 2> ,
    ...
    atributoN tipo <restrições do atributo N> ,
    <restrições da tabela>
);
```

- **Referência:**
  - https://dev.mysql.com/doc/refman/8.0/en/data-types.html

Linguagem de Definição de Dados

- **Tipos de Dados (*numéricos*)**

Data Type	Description
TINYINT	Very small integer value. Signed values range from -128 to 127. Unsigned values range from 0 to 255.
SMALLINT	Small integer value. Signed values range from -32768 to 32767. Unsigned values range from 0 to 65535.
MEDIUMINT	Medium integer value. Signed values range from -8388608 to 8388607. Unsigned values range from 0 to 16777215.
INT	Standard integer value. Signed values range from -2147483648 to 2147483647. Unsigned values range from 0 to 4294967295.
INTEGER	Same as INT
BIGINT	Big integer value. Signed values range from -9223372036854775808 to 9223372036854775807. Unsigned values range from 0 to 18446744073709551615.
FLOAT	Single precision floating point number. 32 bit (7 digits).
DOUBLE	Double precision floating point number. 64 bit (15-16 digits)
DECIMAL(m,d)	Unpacked fixed point number, where m is the total digits and d is the number of digits after the decimal; 128 bit (28-29 significant digits)

```
CREATE TABLE my_table (
    atributo1 tipo <restrições do atributo 1> ,
    atributo2 tipo <restrições do atributo 2> ,
    ...
    atributoN tipo <restrições do atributo N> ,
    <restrições da tabela>
);
```

Linguagem de Definição de Dados

- **Tipos de Dados (*data*)**

Data Type	Description
DATE	Displayed as 'YYYY-MM-DD'.
DATETIME	Displayed as 'YYYY-MM-DD HH:MM:SS'.
TIMESTAMP	Displayed as 'YYYY-MM-DD HH:MM:SS'.
TIME	Displayed as 'HH:MM:SS'.
YEAR[(2|4)]	Default is 4 digits.

```
CREATE TABLE my_table (
    atributo1 tipo <restrições do atributo 1> ,
    atributo2 tipo <restrições do atributo 2> ,
    ...
    atributoN tipo <restrições do atributo N> ,
    <restrições da tabela>
);
```

Linguagem de Definição de Dados

- **Tipos de Dados (*blob*)**

Data Type	Description
TINYBLOB	Maximum size of 255 bytes.
BLOB	Maximum size of 65,535 bytes.
MEDIUMBLOB	Maximum size of 16,777,215 bytes.
LONGTEXT	Maximum size of 4GB or 4,294,967,295 characters.

```
CREATE TABLE my_table (
    atributo1 tipo <restrições do atributo 1> ,
    atributo2 tipo <restrições do atributo 2> ,
    ...
    atributoN tipo <restrições do atributo N> ,
    <restrições da tabela>
);
```

Linguagem de Definição de Dados

- **Restrições de atributos (colunas)**
  - UNIQUE
  - NOT NULL
  - AUTO_INCREMENT
  - DEFAULT *valor*
  - CHECK *(condição)*

```
CREATE TABLE my_table (
    atributo1 tipo <restrições do atributo 1> ,
    atributo2 tipo <restrições do atributo 2> ,
    ...
    atributoN tipo <restrições do atributo N> ,
    <restrições da tabela>
);
```

Linguagem de Definição de Dados

- **Restrições de tabela**
  - PRIMARY KEY (*atributos*)
  - CHECK (*condição*)
  - UNIQUE (*atributos*)

```
CREATE TABLE my_table (
   atributo1 tipo <restrições do atributo 1> ,
   atributo2 tipo <restrições do atributo 2> ,
   ...
   atributoN tipo <restrições do atributo N> ,
   <restrições da tabela>
);
```

  - **FOREIGN KEY** (atributoFK) **REFERENCES** tabela_origem(atributoPK)

    [ON DELETE | ON UPDATE] [RESTRICT | CASCADE | SET NULL | SET DEFAULT]

.StartSe
<tech/academy>

Linguagem de Definição de Dados

Banco de dados / SQL

Gustavo Viais
Tech Lead

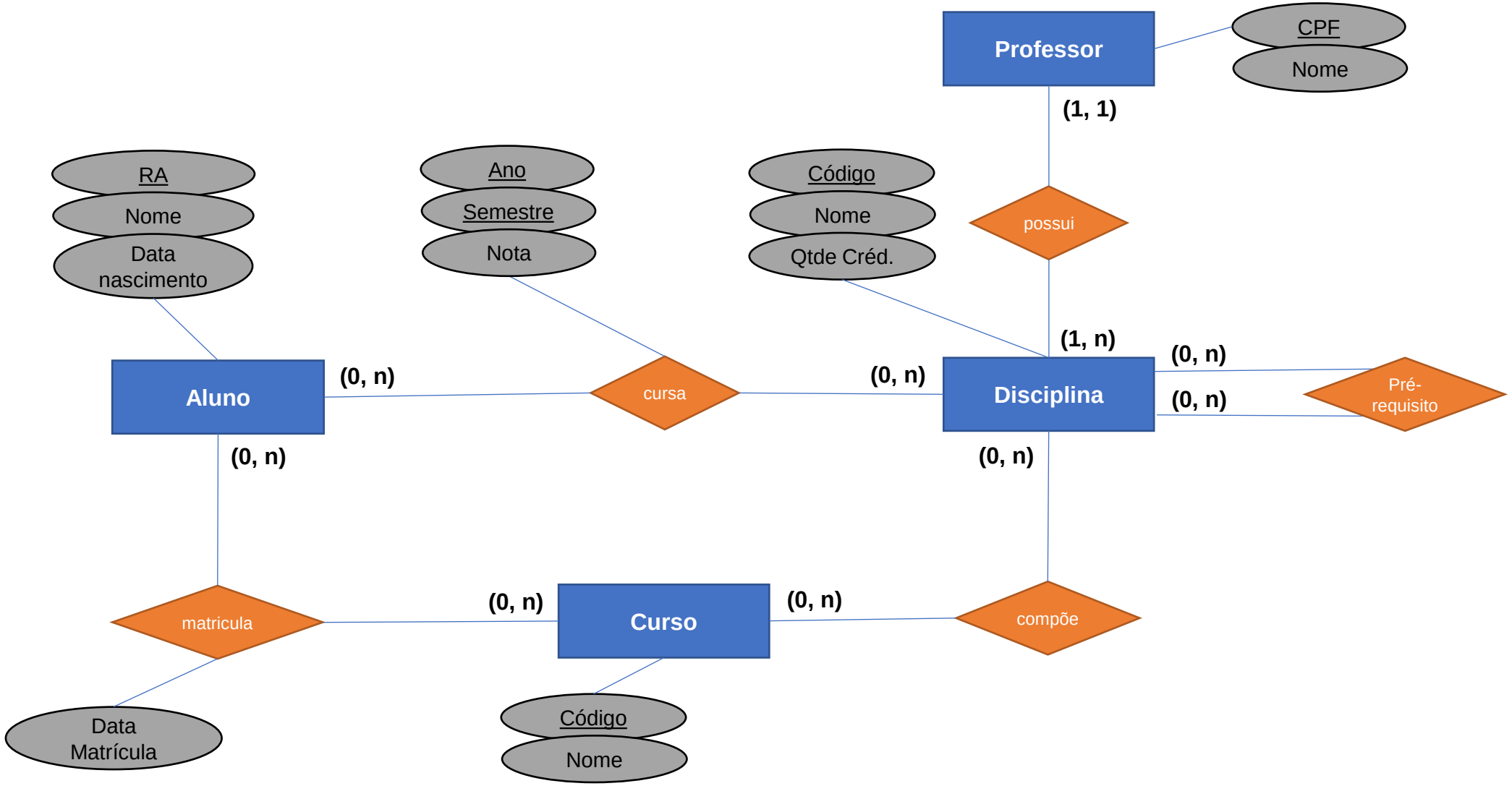
.StartSe

Gustavo Viais
Tech Lead

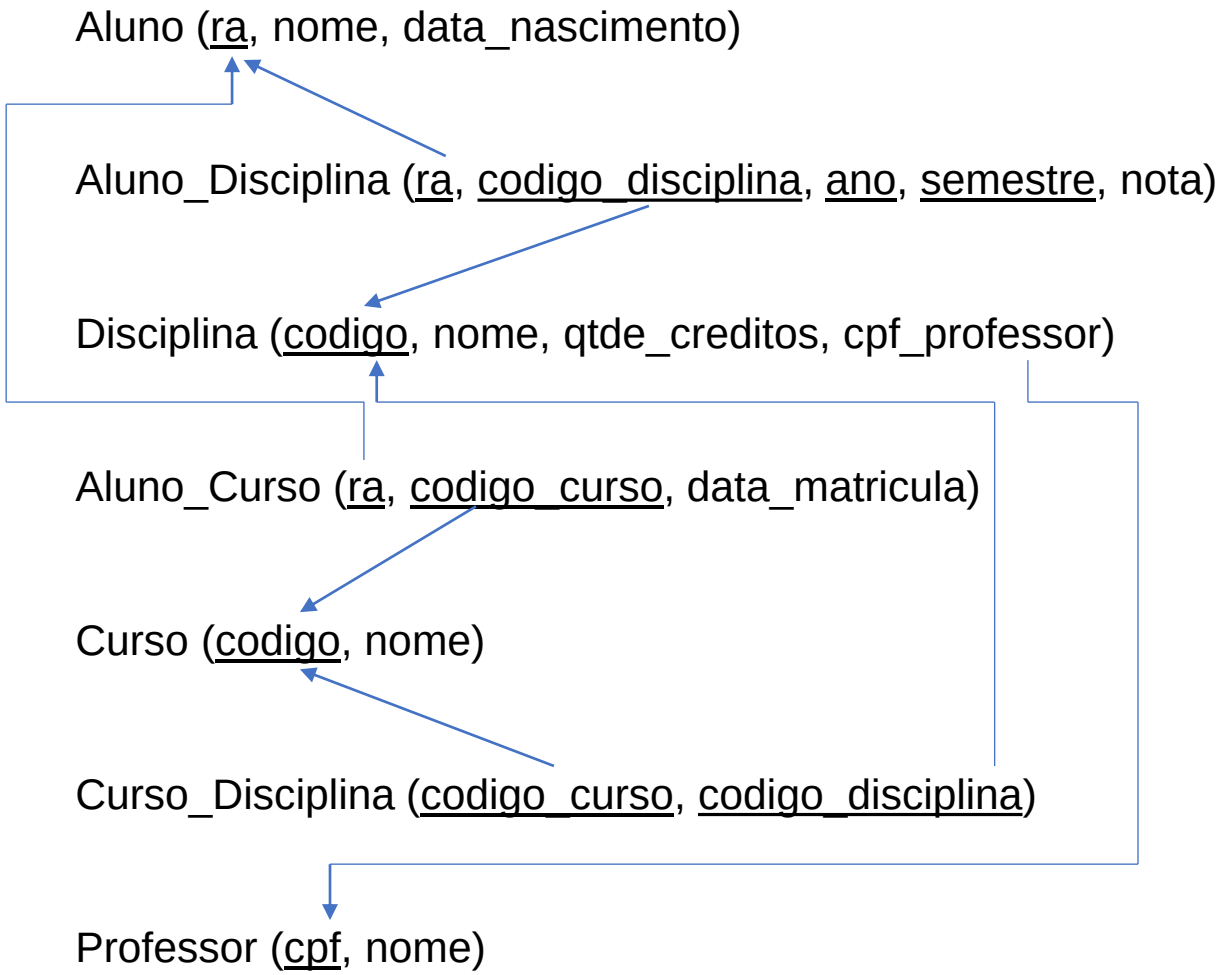
Hands-on - Criando o BD e tabelas

Banco de dados / SQL

Exemplo DER - Universidade



Exemplo Modelo Relacional - Universidade



Gustavo Viais
Tech Lead

Hands-on - Criando o BD e tabelas

Banco de dados / SQL

.StartSe
<tech/academy>

Alterando e removendo tabelas

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

ALTER TABLE

- Utilizado para alterar/adicionar atributos em uma determinada tabela;

```
ALTER TABLE my_table <ação>;
```

- **Exemplo ações**
  - ADD atributo *tipo* <restrições do atributo>
  - DROP atributo [CASCADE | RESTRICT]
  - ALTER atributo SET DEFAULT <valor>
  - ALTER atributo DROP DEFAULT

DROP TABLE

- Utilizado para excluir uma tabela do BD;
- Uma tabela **não** pode ser **removida** se outra entidade depender dela:
  - Analogia: árvore genealógica;
  - Objetivo: garantir consistência relacional.

```
DROP TABLE my_table;
```

.StartSe
<tech/academy>

Alterando e removendo tabelas

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Linguagem de Manipulação de Dados

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

Linguagem de Manipulação de Dados

- **Principais comandos:**
  - *INSERT*: utilizado para adicionar tuplas em tabelas;
  - *UPDATE*: usado para atualizar dados contidos nas tabelas;
  - *DELETE*: usado para remover tuplas das tabelas;
  - *SELECT*: utilizado para consultar os dados contidos nas instâncias do BD.

INSERT

- **Adicionar tupla em tabela**

```
INSERT INTO my_table (atributo1, atributo2, ..., atributoN)
        VALUES (valorAtributo1, valorAtributo2, ..., valorAtributoN)
```

INSERT

- **Exemplo**

```
INSERT INTO aluno (ra, nome, data_nascimento)
        VALUES (1234, 'José da Silva', '2000-02-01')
```

.StartSe
<tech/academy>

Linguagem de Manipulação de Dados

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Gustavo Viais
Tech Lead

.StartSe

Consultando dados em tabelas

Banco de dados / SQL

SELECT

- Retorna informações contidas nas tabelas, de acordo com as condições e projeções selecionadas

```
SELECT
        <lista de colunas> → projeção
FROM
        <tabela(s)>
WHERE
        <condições>
```

Projeção * representa “todas as colunas” das respetivas tabelas

OPERADORES CONDICIONAIS

- Utilizado para identificar a(s) tupla(s) que serão retornadas nas operações de **consulta**, **atualização** e **remoção** de dados;

- **Alguns operadores**
  - = , <>, >, >=, <, <= → Ex: idade = 20, valor >= 0, data_final < ‘2022-12-31’
  - AND, OR, NOT → Ex: NOT(idade = 20) AND data_final < ‘2022-12-31’
  - IN, NOT IN → Ex: ano IN (2021, 2022), versão NOT IN (‘1.0’, ‘2.0’)
  - BETWEEN → Ex: data_venda BETWEEN ‘2022-01-01’ and ‘2022-12-31’
  - IS NULL → Ex: data_nascimento IS NULL

OPERADORES CONDICIONAIS

- LIKE
  - Compara sequência de caracteres dentro de uma *string*;
  - % é utilizado para representar qualquer valor antes e/ou após a sequência;
  - Exemplo → atributo LIKE '%*string*%'

.StartSe
<tech/academy>

Gustavo Viais
Tech Lead

.StartSe

Consultando dados em tabelas

Banco de dados / SQL

.StartSe
<tech/academy>

Atualizando e excluindo dados em tabelas

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

UPDATE

- **Atualizar dados das tabelas**

```
UPDATE my_table SET
        atributo1 = <novo valor>,
        atributo2  = <novo valor>,
        ...
WHERE
        <condições>
```

UPDATE

- **Exemplo**

```
UPDATE aluno SET
        nome = ‘José da Silva’
WHERE
        ra = 1234 AND
        nome LIKE ‘José Silva’
```

DELETE

- **Remover tupla(s) de tabelas**

```
DELETE FROM my_table
WHERE
        <condições>
```

Cuidado com o UPDATE e DELETE sem WHERE !!

DELETE

- **Remover tupla(s) de tabelas**

```
DELETE FROM aluno
WHERE
        ra = 1234 AND
        data_nascimento = ‘2000-05-20’
```

.StartSe
<tech/academy>

Atualizando e excluindo dados em tabelas

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Junção entre tabelas

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

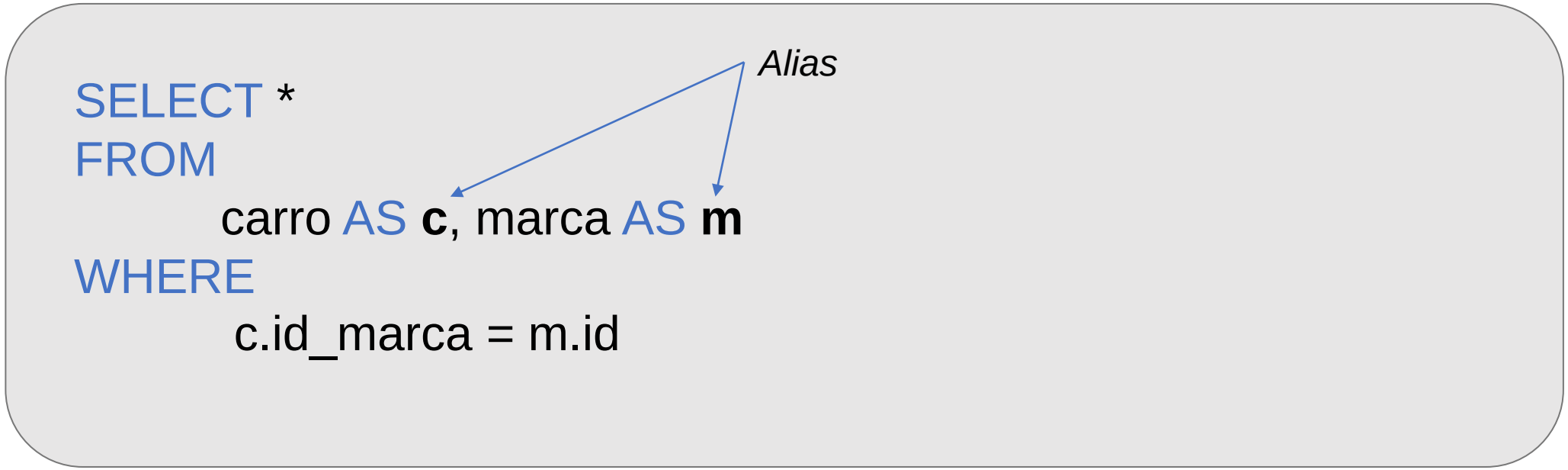
JUNÇÃO ENTRE TABELAS

- Relacionar dados/tuplas de acordo com as respectivas chaves

```
SELECT
        <lista de colunas>
FROM
        A, B
WHERE
        A.id_B = B.id
```

JUNÇÃO ENTRE TABELAS

- Exemplo



JUNÇÃO ENTRE TABELAS

- Cláusula **JOIN**

```
SELECT
        <lista de colunas>
FROM
        A
JOIN
        B  ON  A.id_B = B.id
```

JUNÇÃO ENTRE TABELAS

- Exemplo

```
SELECT *
FROM
        carro c
JOIN
        marca m  ON  c.id_marca = m.id
```

UNION e UNION ALL

- **UNION**: combina os resultados que são distintos entre duas consultas
- **UNION ALL**: combina todos resultados entre duas consultas

```
SELECT
        atributo1
FROM
        tabela_A

[ UNION | UNION ALL ]

SELECT
        atributo1
FROM
        tabela_B
```

Projeções devem ser compatíveis!!

.StartSe
<tech/academy>

Junção entre tabelas

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

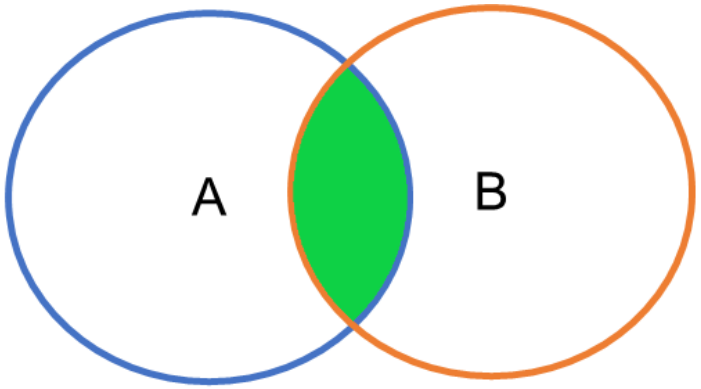
Tipos de JOIN

Banco de dados / SQL

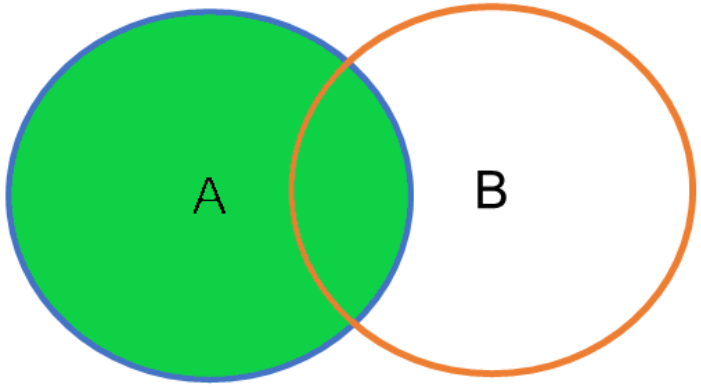
TIPOS DE JOIN

.StartSe
<tech/academy>

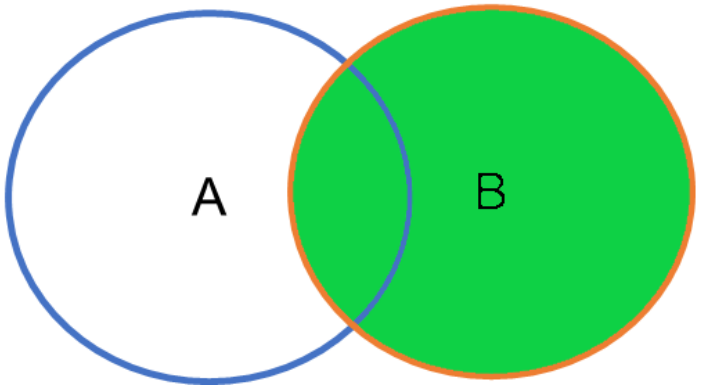
[INNER] JOIN



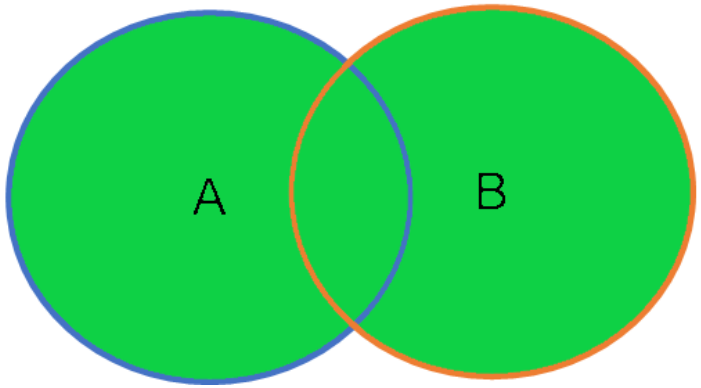
LEFT JOIN



RIGHT JOIN



FULL JOIN



INNER JOIN

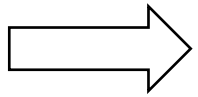
- Retorna **apenas** as tuplas que possuem correspondência em **R** e **S**

R

A	B	C
1	a	z
2	s	x
3	a	y
4	d	x

S

A	D
1	f
3	g
5	h



R INNER JOIN **S**

R.A	S.A	B	C	D
1	1	a	z	f
3	3	a	y	g

LEFT JOIN

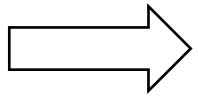
- Retorna **todas** as tuplas de **R** e preenche com valores nulos as tuplas de **S** que não correspondem à(s) chave(s) de **R**

R

A	B	C
1	a	z
2	s	x
3	a	y
4	d	x

S

A	D
1	f
3	g
5	h



R LEFT JOIN **S**

R.A	S.A	B	C	D
1	1	a	z	f
2	*null*	s	x	*null*
3	3	a	y	g
4	*null*	d	x	*null*

RIGHT JOIN

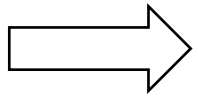
- Retorna **todas** as tuplas de **S** e preenche com valores nulos as tuplas de **R** que não correspondem à(s) chave(s) de **S**

R

A	B	C
1	a	z
2	s	x
3	a	y
4	d	x

S

A	D
1	f
3	g
5	h



R RIGHT JOIN **S**

R.A	S.A	B	C	D
1	1	a	z	f
3	3	a	y	g
null	5	*null*	*null*	h

FULL JOIN

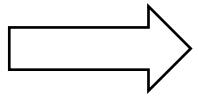
- Retorna **todas** as tuplas de **R** e **S**, preenchendo com valores nulos as tuplas de que não possuem chaves correspondentes entre **R** e **S**

R

A	B	C
1	a	z
2	s	x
3	a	y
4	d	x

S

A	D
1	f
3	g
5	h



R FULL JOIN **S**

R.A	S.A	B	C	D
1	1	a	z	f
2	*null*	s	x	*null*
3	3	a	y	g
4	*null*	d	x	*null*
null	5	*null*	*null*	h

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Tipos de JOIN

Banco de dados / SQL

.StartSe
<tech/academy>

Agregação, ordenação e distinção

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

FUNÇÕES DE AGREGAÇÃO

- Processa conjunto de valores de uma coluna, resultando em um único valor agregado;
- Exemplo de funções:
  - COUNT (*coluna*) → retorna a quantidade de registros da coluna selecionada
  - MAX (*coluna*) → retorna o valor máximo do atributo da coluna selecionada
  - MIN (*coluna*) → retorna o valor mínimo do atributo da coluna selecionada
  - SUM (*coluna*) → retorna a soma dos atributos da coluna selecionada
  - AVG (*coluna*) → retorna a média de valor dos atributos da coluna selecionada

FUNÇÕES DE AGREGAÇÃO

- Exemplo de consulta:

```
SELECT
        MAX(nota),
        MIN(nota)
FROM
        turma t
WHERE
        t.id = 1
```

FUNÇÕES DE AGREGAÇÃO

- Exemplo de consulta:

```
SELECT
        COUNT(*)
FROM
        alunos a
INNER JOIN
        turma t         ON  a.ra = t.ra
WHERE
        t.id = 1
```

ORDENAÇÃO

- Estabelece a ordem das tuplas resultantes de acordo com a lógica utilizada;
- **ORDER BY** *coluna* [ASC | DESC]
  - ASC
    - Ordena tuplas de acordo com a ordem **crescente** do atributo selecionado;
    - Valor default (quando não especificado na *query*).
  - DESC
    - Ordena tuplas de acordo com a ordem **decrescente** do atributo selecionado.

ORDENAÇÃO

- Exemplo de consulta com ordenação:

```
SELECT *
FROM
        alunos a
INNER JOIN
        turma t         ON  a.ra = t.ra
WHERE
        t.id = 1
ORDER BY
        a.nome
```

DISTINÇÃO

- Seleciona apenas tuplas que possuem valores diferentes entre as respectivas colunas;

- **DISTINCT**

```
SELECT DISTINCT
        coluna1, coluna2, ..., colunaN
FROM
        <tabela(s)>
WHERE
        <condições>
```

DISTINÇÃO

- Exemplo de consulta com distinção:

```
SELECT DISTINCT
        a.ra,
        a.nome
FROM
        alunos a
INNER JOIN
        turma t         ON  a.ra = t.ra
WHERE
        t.id = 1
```

.StartSe
<tech/academy>

Agregação, ordenação e distinção

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

.StartSe
<tech/academy>

Agrupamento

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe

AGRUPAMENTO

- Unifica tuplas com valores idênticos, de acordo com a(s) coluna(s) selecionada(s)/agrupada(s);
- **GROUP BY** *coluna1, coluna2, ..., colunaN*
  - [HAVING] <*condição*>
    - Cláusula opcional;
    - Aplica condições (filtro) às tuplas já agrupadas.

AGRUPAMENTO

- Exemplo de consulta com agrupamento:

```
SELECT
        a.ra,
        a.nome,
        MIN(av.nota),
        MAX(av.nota)
FROM
        alunos a
INNER JOIN
        avaliacoes av  ON  a.ra = av.ra
WHERE
        av.id_turma = 1
GROUP BY
         a.ra, a.nome
HAVING
        AVG(t.nota) >= 6.0
```

.StartSe
<tech/academy>

Agrupamento

Banco de dados / SQL

Gustavo Viais
Tech Lead

.StartSe