

Explorando a Linguagem de Consulta a Banco de Dados SQL – Structured Query Language

Formação de Banco de Dados Relacional

Juliana Mascarenhas

Tech Education Specialist DIO / Owner @Simplificandoredes
e @SimplificandoProgramação

Mestre em modelagem computacional | Cientista de dados

@in/juliana-mascarenhas-ds/

Objetivo Geral

Criação do projeto Lógico a partir do esquema conceitual, para posteriormente utilizar a linguagem SQL para criação do Banco de Dados relacionado. Dessa forma, serão apresentados o mapeamento ER/Relacional, assim como a estrutura e comandos da linguagem de consulta a banco de dados SQL.

Pré-requisitos

- Pensamento Computacional
- Modelagem de dados
- Modelo Entidade Relacionamento

Percorso

Etapa 1

Modelo Relacional e Mapeamento ER/Relacional

Etapa 2

Primeitos Passos com SQL

Etapa 3

Explorando Queries com SQL

Etapa 4

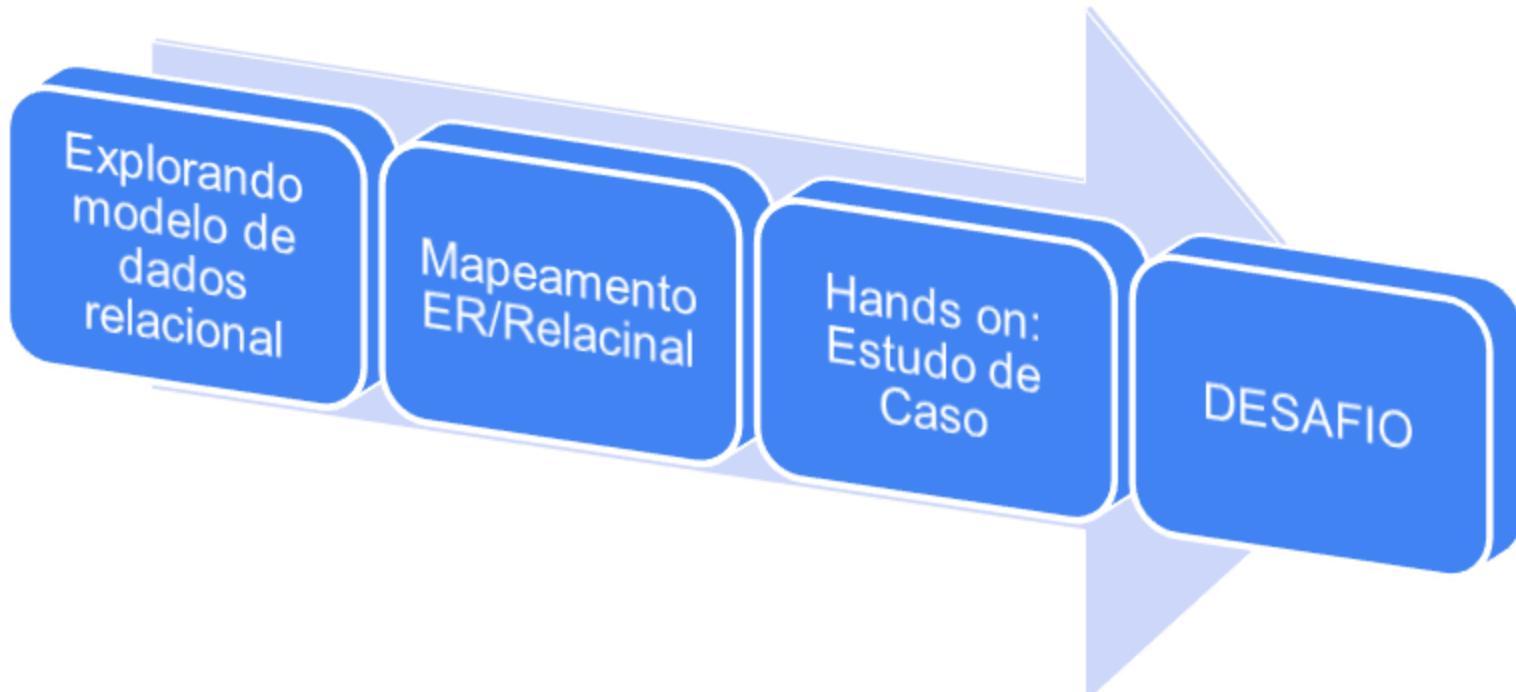
Agrupando Registros e Tabelas com SQL

Etapa 1

Modelo Relacional e Mapemamento ER/Relacional

// Explorando a Linguagem de Consulta a Banco de Dados SQL

Modelo Relacional



Processo



Processo



Conceitos formais do Modelo Relacional

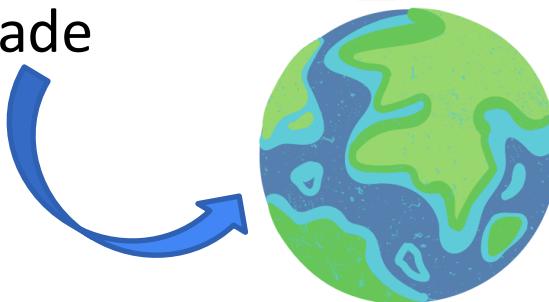
Conceitos

Coleção de Relação

- Modelo Relacional
- Tabelas X Arquivos
- Entidade

Conceitos

- Modelo Relacional
- Tabelas X Arquivos
- Entidade



Coleção de Relação

Flat Files

Classe

Objeto



Conceitos

Domínio

- Tupla
- Atributo
- Relação

COURSE				
Course_name	Course_number	Credit_hours	Department	
Intro to Computer Science	CS1310	4	CS	
Data Structures	CS3320	4	CS	
Discrete Mathematics	MATH2410	3	MATH	
Database	CS3380	3	CS	

Conceitos

- Valores atômicos
- Grupos: especificando nomes



Brasil

Local

Contato de Usuários

Domínio

Nomes



$$\begin{aligned} & B \int_{m-\pi/2}^{m+\pi/2} \frac{dx}{\sin^2 x} = \int_{x=m}^{\pi/2} \frac{dx}{\sin^2 x} = \int_0^{\pi/2-x} \frac{d(x-m)}{\sin^2 x} = \int_0^{\pi/2} \frac{d(\pi/2-x)}{\sin^2 x} = \\ & + \frac{1}{2} \pi^2 = 3,1415 \quad \text{A-C} \\ & \frac{B}{2} \int_{m-\pi/2}^{m+\pi/2} \frac{dx}{\sin^2 x} = \frac{\pi}{2} \quad \text{C} \\ & B = \frac{\pi}{2} \cdot 2 = \pi \\ & \Delta T = \frac{\pi}{2} \cdot \frac{1000}{300} = 523,6^{\circ} \quad \text{D-G} \\ & \Delta x = 4 - 3^2 = 7 \quad \text{E-H} \\ & (x-y) = 2x+3y \quad \text{F-I} \\ & \frac{x^2}{2} = 27,19 \quad \text{G-J} \\ & x = 2,79 \quad \text{H-K} \\ & P = \frac{1}{2} \cos x + \tan y \quad \text{I-L} \\ & = (x-y)^2 \quad \text{J-M} \end{aligned}$$

Nota de Alunos



ID Social

Conceitos

- Esquema Relacional
- A_i : atributo
- D : domínio

$R(A_1, A_2, A_3, \dots, A_n)$

Degree

STUDENT(Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa)

Descrição de uma relação

Conceitos

Definição - R

STUDENT(Name: string, Ssn: string, Home_phone: string, Address: string, Office_phone: string, Age: integer, Gpa: real)

- Esquema Relacional
- Ai: atributo
- D: domínio

r: STUDENT(Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa)

Descrição de uma relação

Conceitos

- Esquema Relacional
- A_i : atributo
- D : domínio

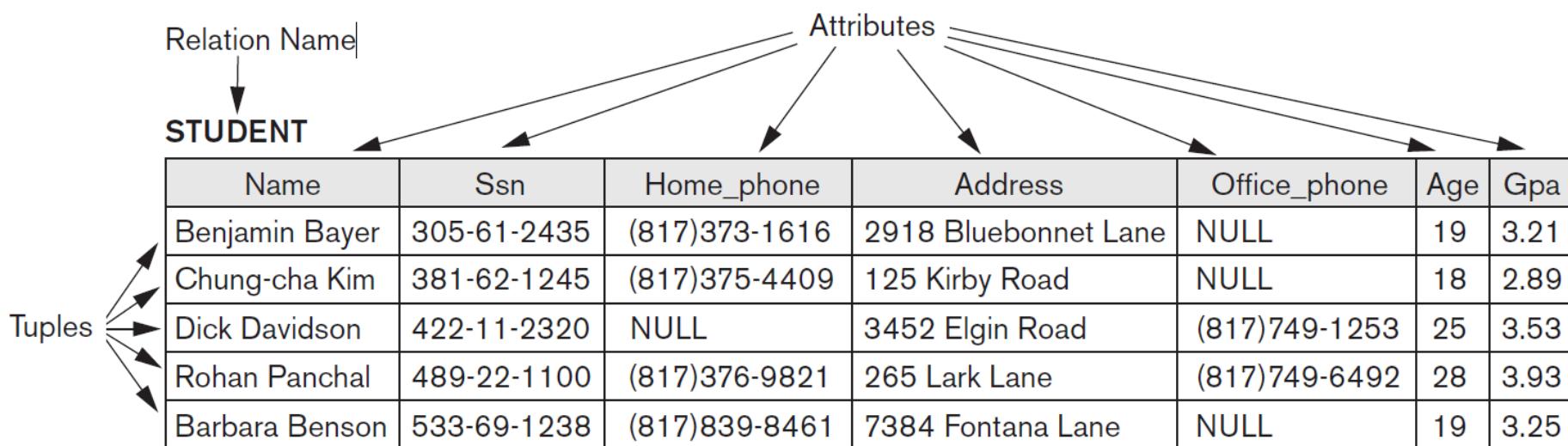
$$r = \{t_1, t_2, t_3, \dots, t_m\}$$

$$t = \langle v_1, v_2, v_3, \dots, v_n \rangle$$

$$1 \leq i \leq n$$

Descrição de uma relação

Conceitos



Conceitos

Cardinalidade

$$r(R) \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times \dots \times \text{dom}(A_n))$$

$$|\text{dom}(A_1)| \times |\text{dom}(A_2)| \times \dots \times |\text{dom}(A_n)|$$

Total de instâncias

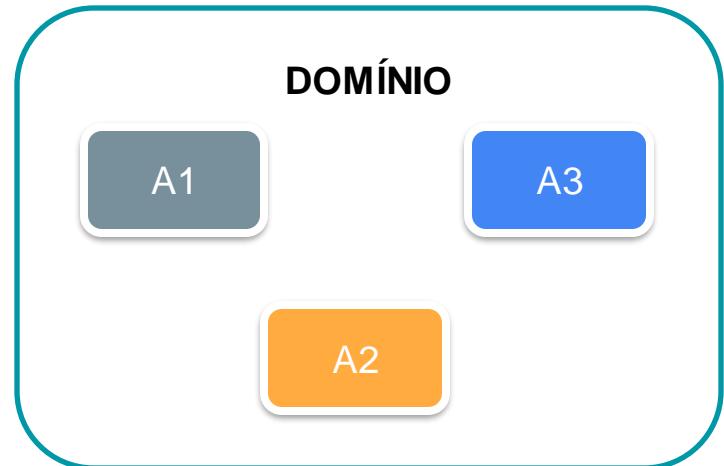
The diagram illustrates the components of a relation. At the top, 'Relation Name' points to the label 'STUDENT'. Below it, 'Attributes' points to the column headers: Name, Ssn, Home_phone, Address, Office_phone, Age, and Gpa. At the bottom, 'Tuples' points to the data rows of the table.

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Conceitos

Coleção de Relação

- Domínio
- Papel (Role)



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93

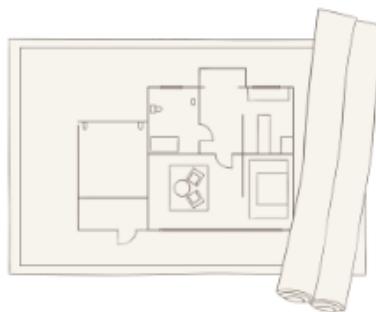
Características da Relação

Características

- Relação != Arquivo != Tabela

Características da Relação

Relação



Arquivos



Tabelas

Ordenação de Tuplas na Relação

Características da Relação



Relation Name | STUDENT

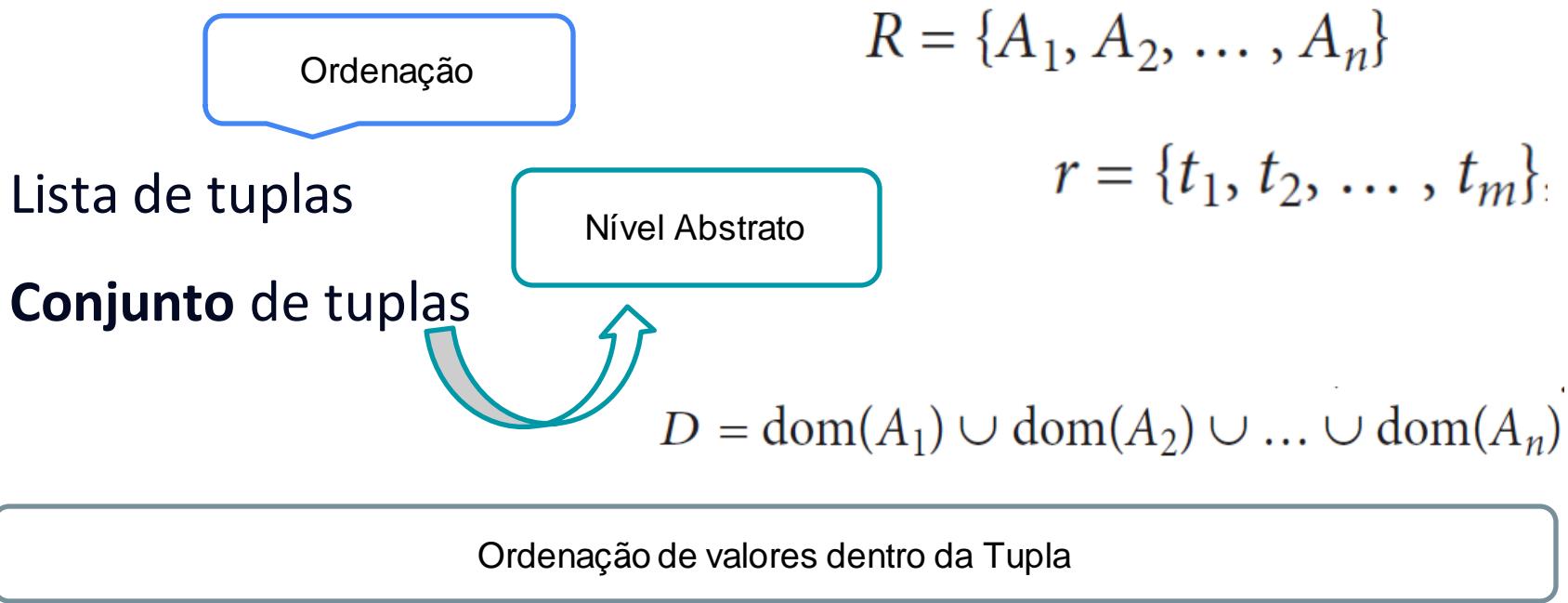
Attributes

Tuples

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Ordenação de Tuplas na Relação

Características da Relação



Características da Relação

Lista de tuplas

Conjunto de tuplas

set of (<attribute>, <value>)



Ordenação de valores dentro da Tupla

Características da Relação

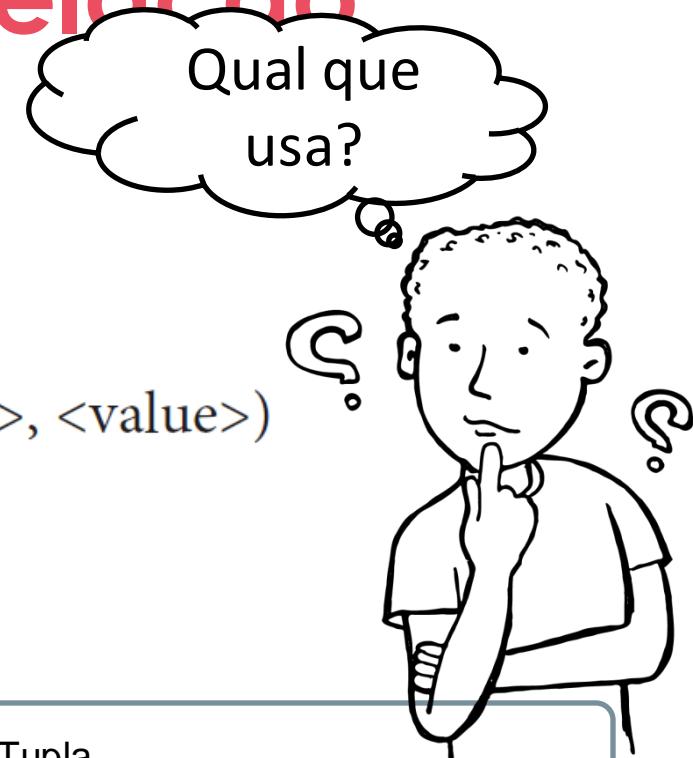
Lista de tuplas

Conjunto de tuplas

set of (<attribute>, <value>)



Ordenação de valores dentro da Tupla



Características da Relação

- Valores Nulos

Escritório



Casa



Valor desconhecido

Valor existente mas indisponível

Atributo não se aplica

Valor indefinido

Valores nulos nas tuplas

Características da Relação

- Valores Nulos



Valor desconhecido

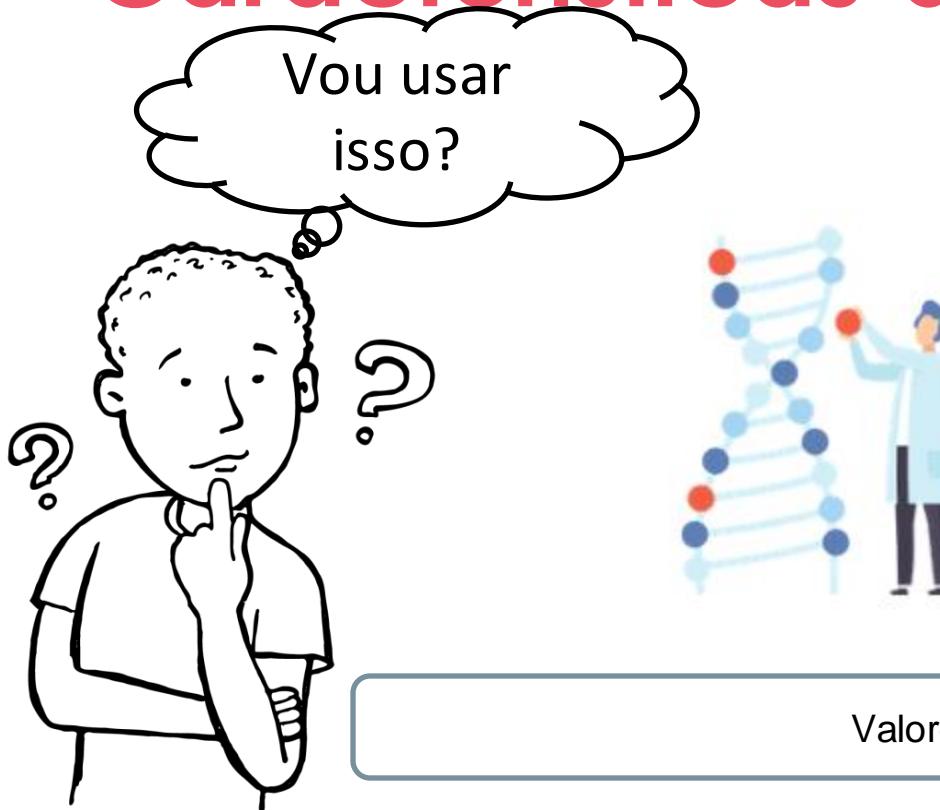
Valor existente mas indisponível

Atributo não se aplica

Valor indefinido

Valores nulos nas tuplas

Características da Relação



Valor desconhecido

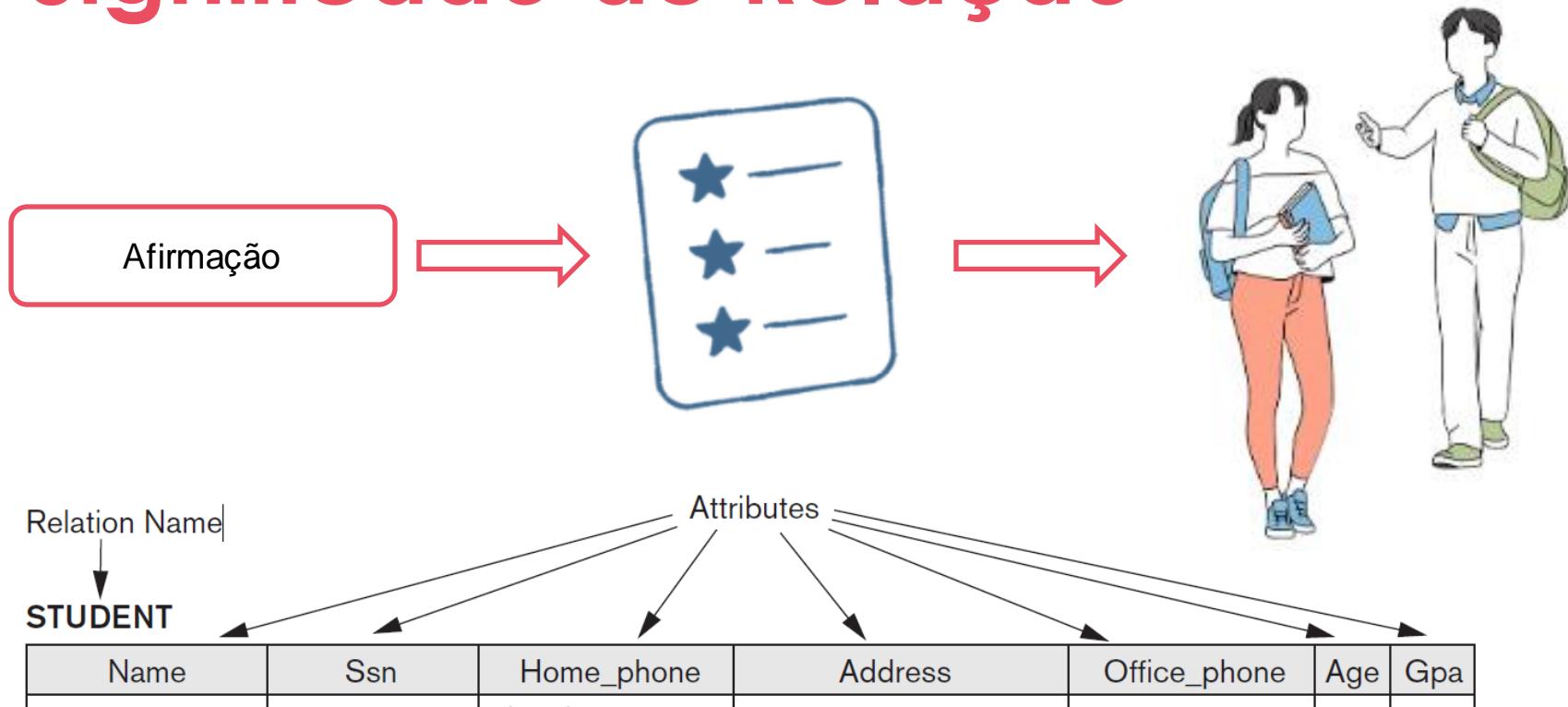
Valor existente mas indisponível

Atributo não se aplica

Valor indefinido

Valores nulos nas tuplas

Significado de Relação



Significado de Relação

Lúcia correu semana passada

Predicado

O **predicado**, formado por um ou mais verbos, é aquilo que se declara sobre a ação do sujeito, concordando em número e pessoa com ele.

Preposições

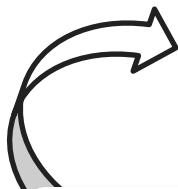
Tuples

STUDENT							
Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa	
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21	
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89	
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53	
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93	
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25	

Significado de Relação

Predicado

O **predicado**, formado por um ou mais verbos, é aquilo que se declara sobre a ação do sujeito, concordando em número e pessoa com ele.



Afirmações declarativas



Lógica de Predicados

Validade de Argumentos

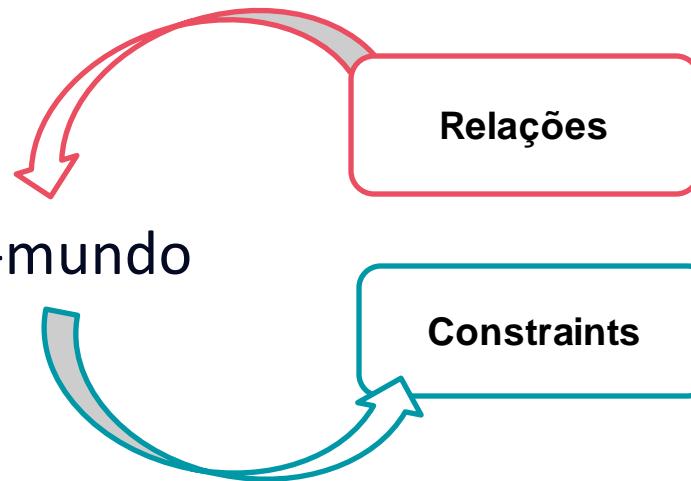
Características da Relação

- Uppercase: Relações - Q, R, S
 - Lowercase: Estado de Relação - q, r, s
 - Letras: Tuplas – t, u, v
 - Nome atributo = papel
- $t = \langle v_1, v_2, \dots, v_n \rangle$
- 
- Nome
(ESTUDANTE)

Constraints do Modelo Relacional

Constraints

- Composição do mini-mundo



Dados do Contexto de Banco de Dados

Constraints

- Categorizadas



Características das
relações

Inheret Model-based Constraints

Schema-base Constraints

DDL

Application-base Constraints

Dados do Contexto de Banco de Dados

Constraints

- Dependência de dados

Dependências funcionais

Dependências Multivaloradas

Processo de Normalização



Constraints baseadas no Esquema do Banco de Dados

Constraints de Domínio

Integer

Float

$A = \langle v_1 \rangle - \text{dom}(A)$

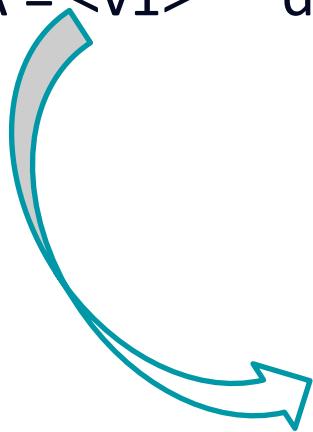


Diagram illustrating the components of a relation:

- Relation Name:** STUDENT
- Attributes:** Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa
- Tuples:** Represented by the rows in the table.

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Constraints de Domínio

$A = \langle v_1 \rangle - \text{dom}(A)$



The diagram shows a relation named 'STUDENT' with attributes: Name, Ssn, Home_phone, Address, Office_phone, Age, and Gpa. Below the relation name, an arrow points to the 'Name' attribute, labeled 'Attributes'. Another arrow points from the 'Attributes' label to the entire table. A large curved arrow on the left points to the table, labeled 'Tuples' at its base. The table data is as follows:

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Constraint de Chave

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Relação -> conjunto de tuplas

Repetição

Uniqueness Constraint

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

SK = Subconjunto de atributos

$$t_1[\text{SK}] \neq t_2[\text{SK}]$$

Uniqueness Constraint

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Chave & Superchave

$$t_1[\text{SK}] \neq t_2[\text{SK}]$$

Constraint de Chave

+ de uma
possível chave?



CAR

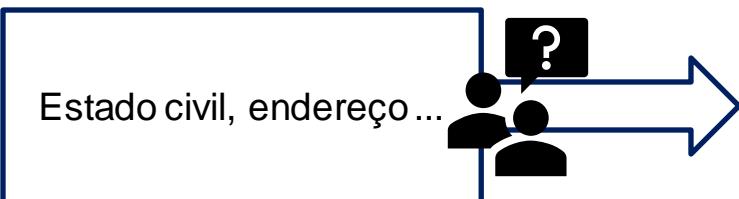
License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

Chave candidata

Constraint de Chaves

- Uniquinness constraints
- Chave primária e candidata

Valores nulos em atributos?



STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

SGBDs e esquemas relacionais

Esquema relacional

SGBD É um subconjunto de relações

$$S = \{R_1, R_2, \dots, R_m\}$$

Estado de um SGBD relacional

$$\text{DB} = \{r_1, r_2, \dots, r_m\}$$

Esquema relacional

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

DEPT_LOCATIONS

<u>Dnumber</u>	Dlocation
----------------	-----------

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	Dependent_name	Sex	Bdate	Relationship
-------------	----------------	-----	-------	--------------

Figure 5.5

Schema diagram for the COMPANY relational database schema.

Esquema relacional

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	<u>Manager_start_date</u>
-------	----------------	---------------------------

DEPT_LOCATIONS

<u>Dnumber</u>	
----------------	--

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	Dependent_name	Sex	Bdate	Relationship
-------------	----------------	-----	-------	--------------

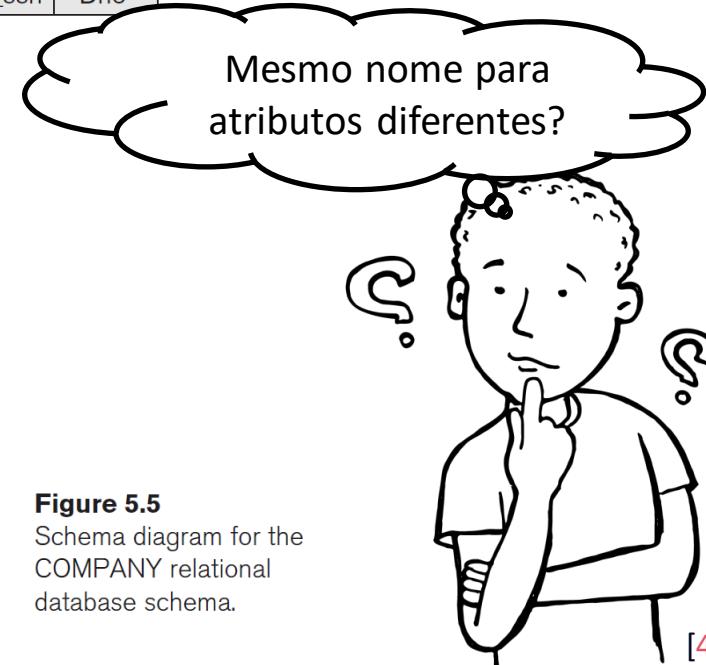


Figure 5.5

Schema diagram for the COMPANY relational database schema.

Esquema relacional

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Address	Sex	Salary	Super_ssn
-------	-------	-------	------------	---------	-----	--------	-----------

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

DEPT_LOCATIONS

<u>Dnumber</u>	Dlocation
----------------	-----------

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	Dependent_name	Sex	Bdate	Relationship
-------------	----------------	-----	-------	--------------

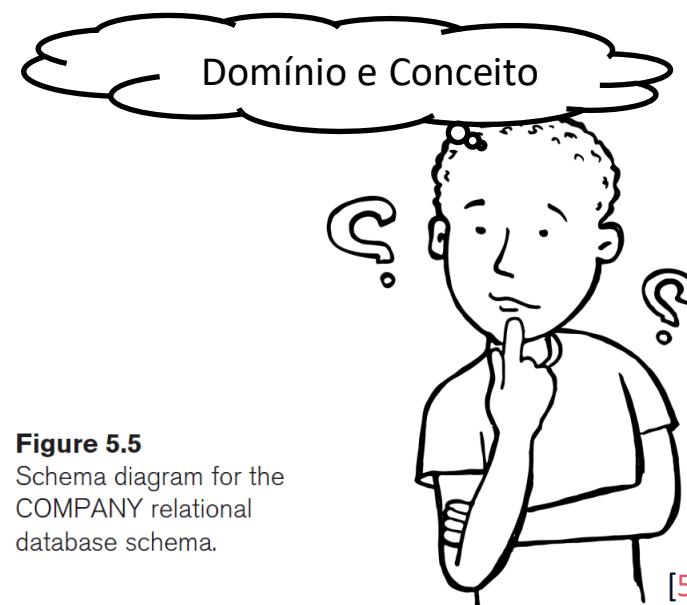


Figure 5.5

Schema diagram for the COMPANY relational database schema.

Esquema relacional

SGBD {
 $S = \{R_1, R_2, \dots, R_m\}$
 $DB = \{r_1, r_2, \dots, r_m\}$

Restrições de integridade



Status do SGBD

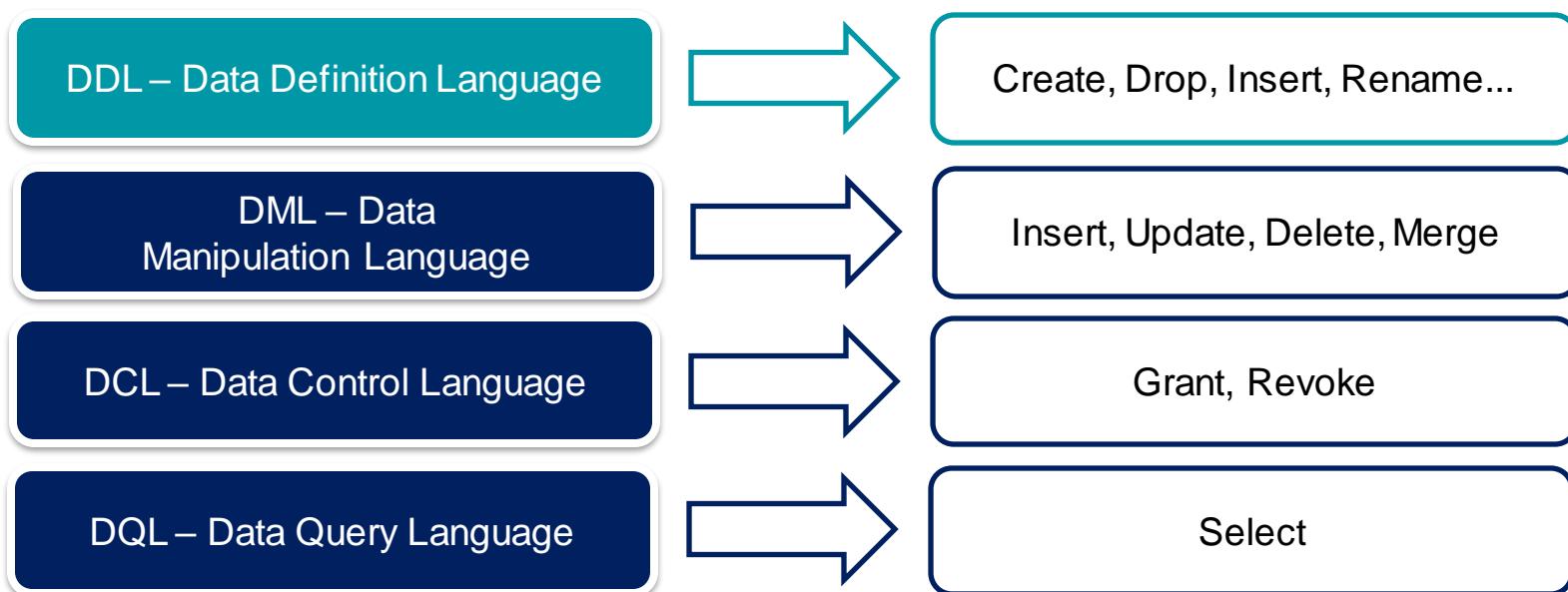
Definição

DDL – Data Definition Language



Create, Drop, Insert, Rename...

Definição



Estado SGBD - Company

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPT_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Estado SGBD - Company

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

WORKS_ON

Essn	Pho	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

Integridade

SGBD



Integridade

- Constraints de Integridade
- Constraints de domínio, chave, Not Null

Integridade de entidade

Integridade Referencial



Integridade Referencial e de Entidade e Chaves Estrangeiras

Integridade

- Chave Primária != NULL
- Consistência entre entidades



Integridade de Entidade

PROJECT

A diagram illustrating a primary key constraint. A white arrow points down to the 'Pnumber' column header. A blue arrow points from the 'Dnum' column header to the right edge of the table, indicating that 'Dnum' cannot have null values.

Pname	<u>Pnumber</u>	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

Integridade Referencial

DEPARTMENT

A diagram illustrating a referential integrity constraint. A white arrow points down to the 'Dname' column header. A blue arrow points from the 'Dnumber' column header to the left edge of the table, indicating that 'Dnumber' must reference a valid department ID in the DEPARTMENT table.

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

Integridade

Chave estrangeira



Integridade de Entidade

- Chave Primária != NULL
- Consistência entre entidades

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4



DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19



Chave estrangeira

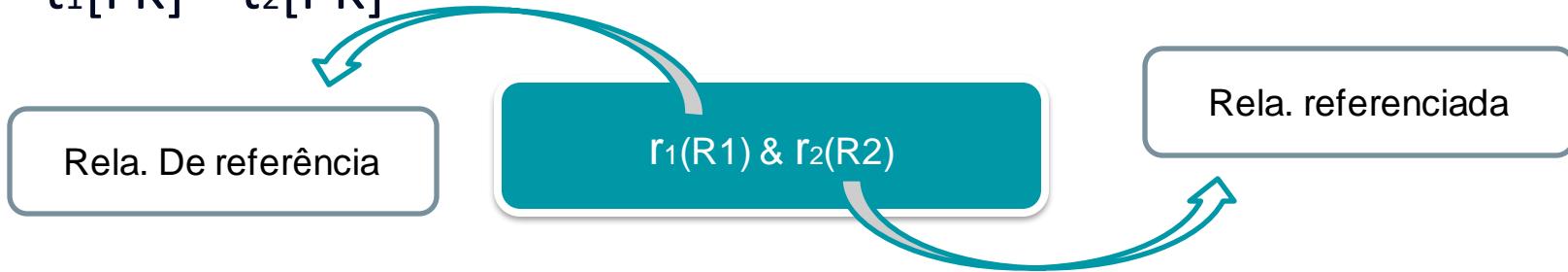
Regras:

- Domínio dos atributos de FK = Domínio dos atributos da PK
- $t_1[FK] = t_2[PK]$

Chave estrangeira

Regras:

- Domínio dos atributos de FK = Domínio dos atributos da PK
- $t_1[\text{FK}] = t_2[\text{PK}]$



Integridade Referencial

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

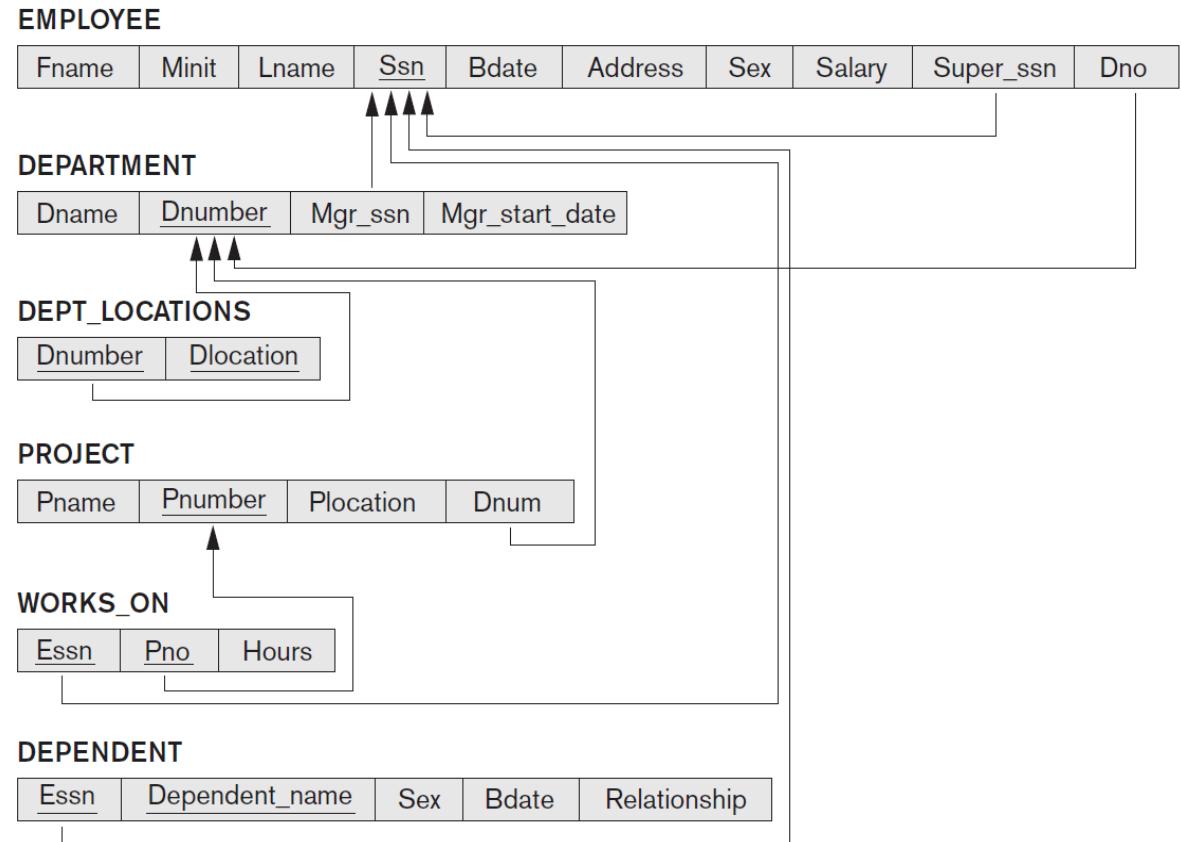
DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Chave estrangeira



DDL

Outras Constraints

Constraints - semântica

- Salário máximo do supervisor
- Carga horária máxima/semana



Triggers &
Assertions

Depende do contexto do banco de dados

Outras Constraints

Restrições de Estado

O salário do empregado só
pode aumentar ...

Restrições de Transição

Triggers & Assertions

Aplicação



Mapeando o modelo ER/Relacional

Mapeando

Design Lógico do Projeto

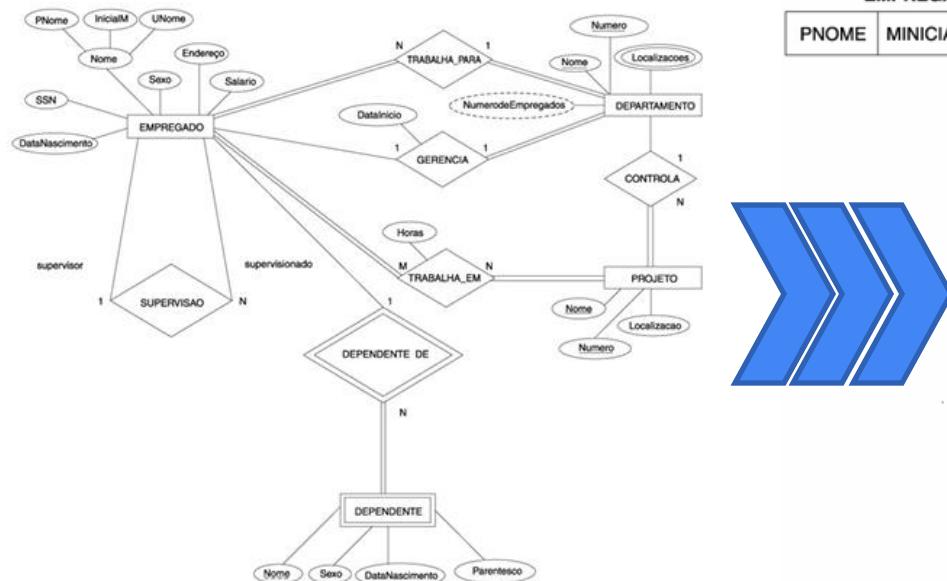
Mapeamento do Modelo de dados

ER ou EER

Relacional

Logical database design

Projeto Lógico



EMPREGADO									
PNAME	MINICIAL	UNOME	SSN	DATANASC	ENDERECO	SEXO	SALARIO	SUPERSSN	DNO

DEPARTAMENTO			
DNAME	DNUMBER	GERSSN	GERDATAINICIO

DEPTO_LOCALIZACOES	
DNUMBER	DLOCALIZACAO

PROJETO			
PJNAME	PNUMBER	PLOCALIZACAO	DNUM

TRABALHA_EM		
ESSN	PNO	HORAS

DEPENDENTE				
ESSN	NOME_DEPENDENTE	SEXO	DATANASC	PARENTESCO

Fonte: livro de referência - Navathe

Mapeamento ER para Modelo Relacional

Algortimo de Mapeamento

Referência

- Atributos de valor único (single-values)
- PK – Primary Key & Unique Key
- Constraint de Entidade e Referencial

Algoritmo de Mapeamento

- Relação -> E (A1,A2,A3,...,A4)
- PK Simples ou Composta?
- Unique Constraint
- FK – Foreign Key



Entidades Fortes (Regulares)

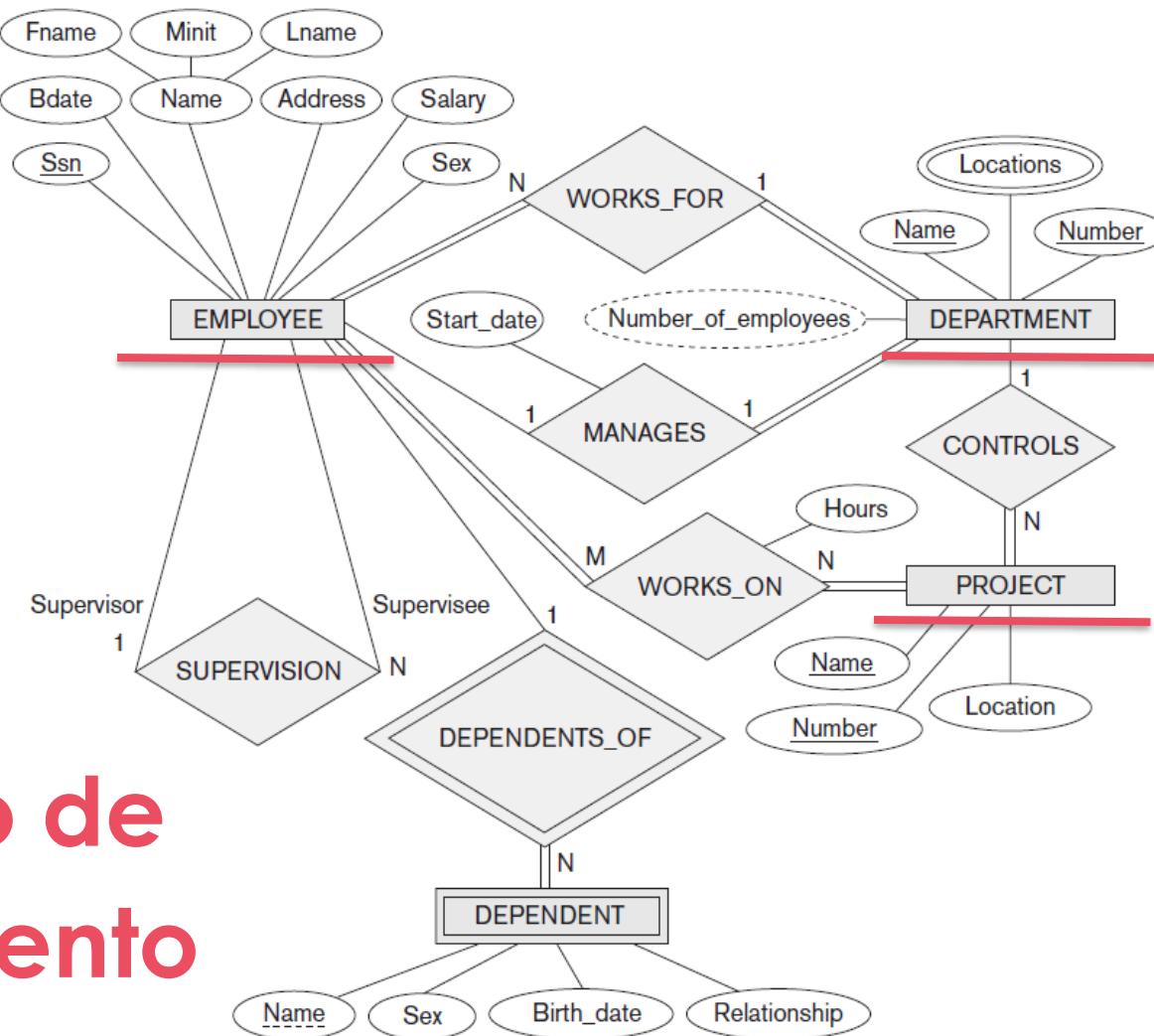
Algoritmo de Mapeamento

- Relação -> E (A1,A2,A3,...,A4)
- PK Simples ou Composta?
- Unique Constraint
- FK – Foreign Key



PROJECT
DEPARTMENT
EMPLOYEE

Entidades Fortes (Regulares)



Algoritmo de Mapeamento

Strong Entity Mapping

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary
-------	-------	-------	------------	-------	---------	-----	--------

DEPARTMENT

Dname	<u>Dnumber</u>
-------	----------------

PROJECT

Pname	<u>Pnumber</u>	Plocation
-------	----------------	-----------

Relação de Entidade – **Entity Relation**

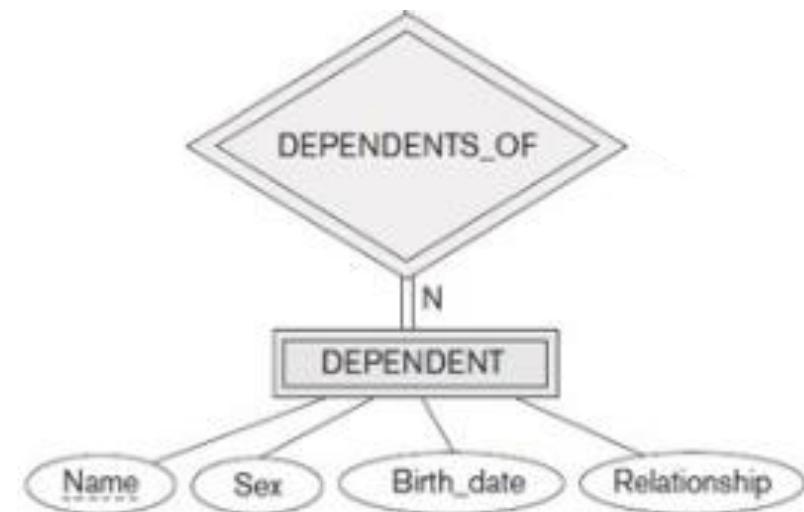
Algortimo de Mapeamento

- Relação -> E (A1,A2,A3,...,A4)
- PK Simples ou Composta?
- Unique Constraint
- FK – Foreign Key



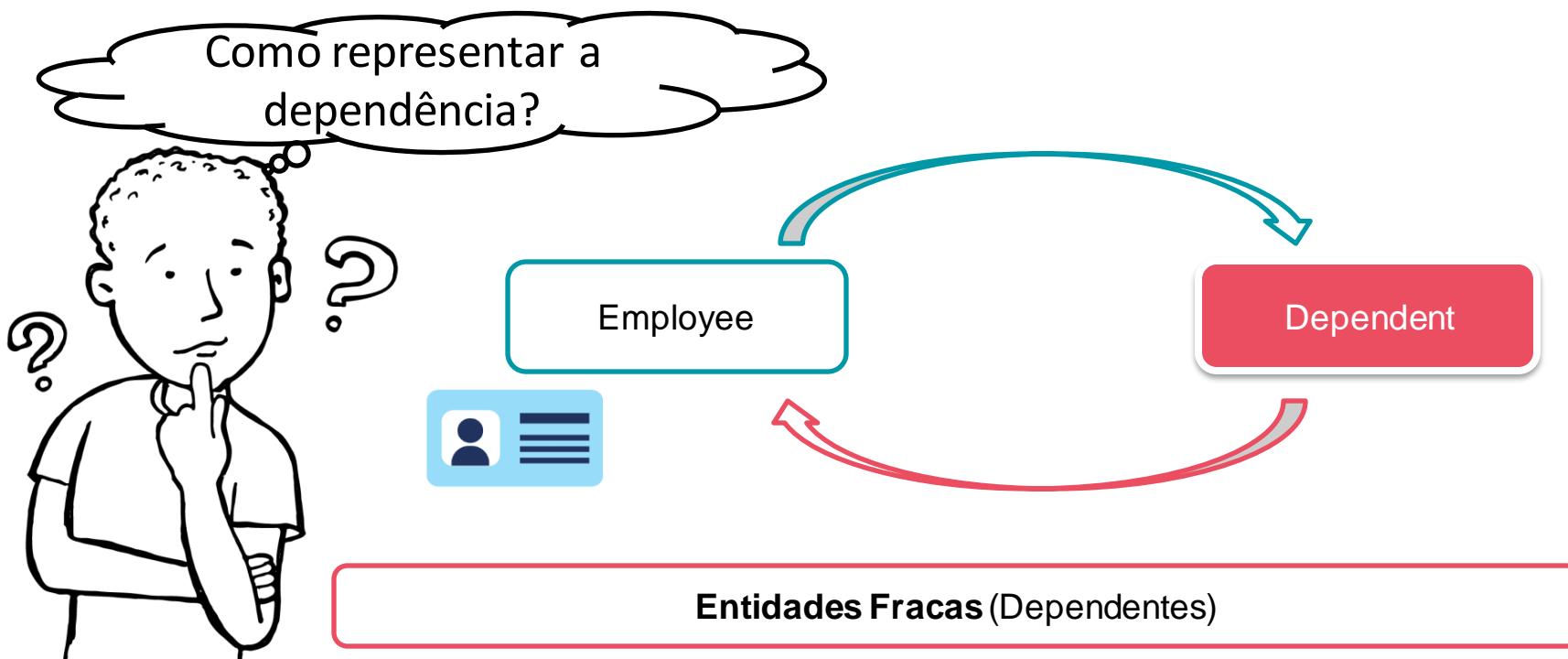
Entidades Fracas (Dependentes)

Algortimo de Mapeamento

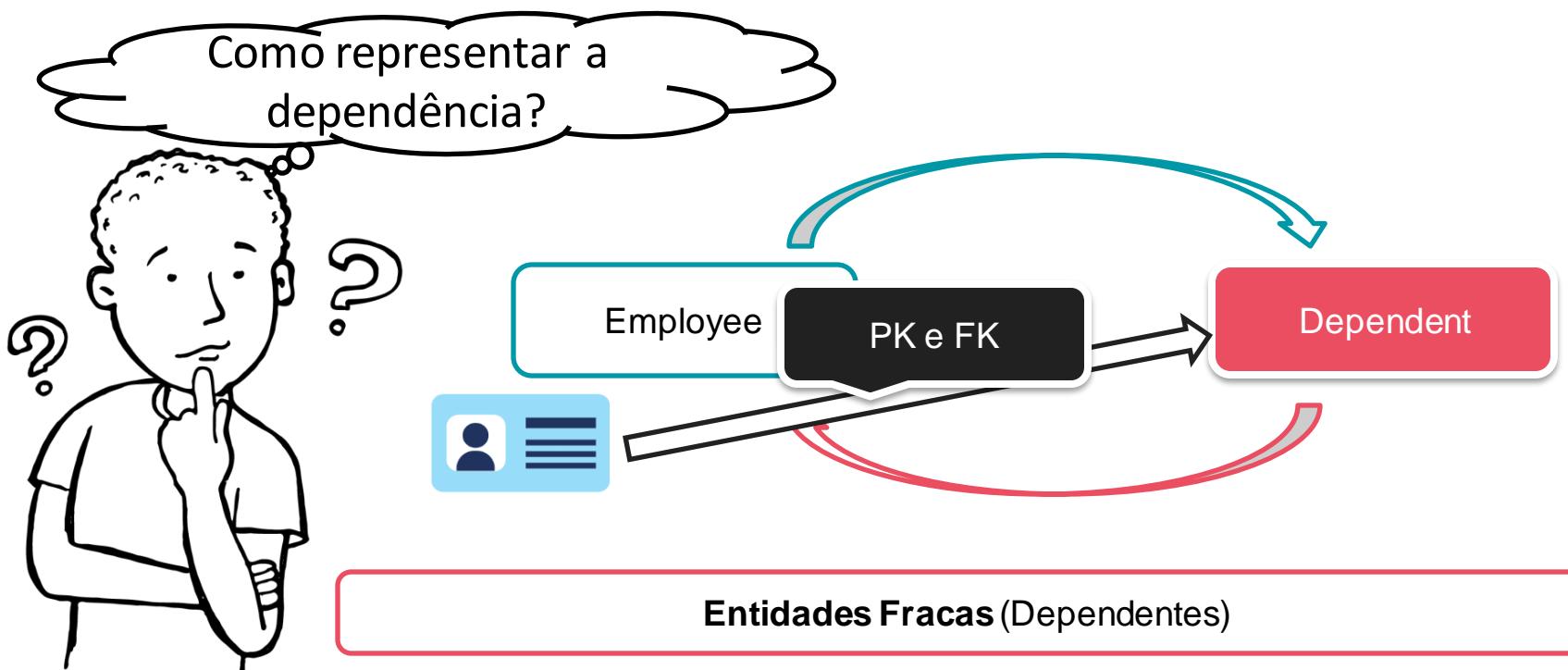


Entidades Fracas (Dependentes)

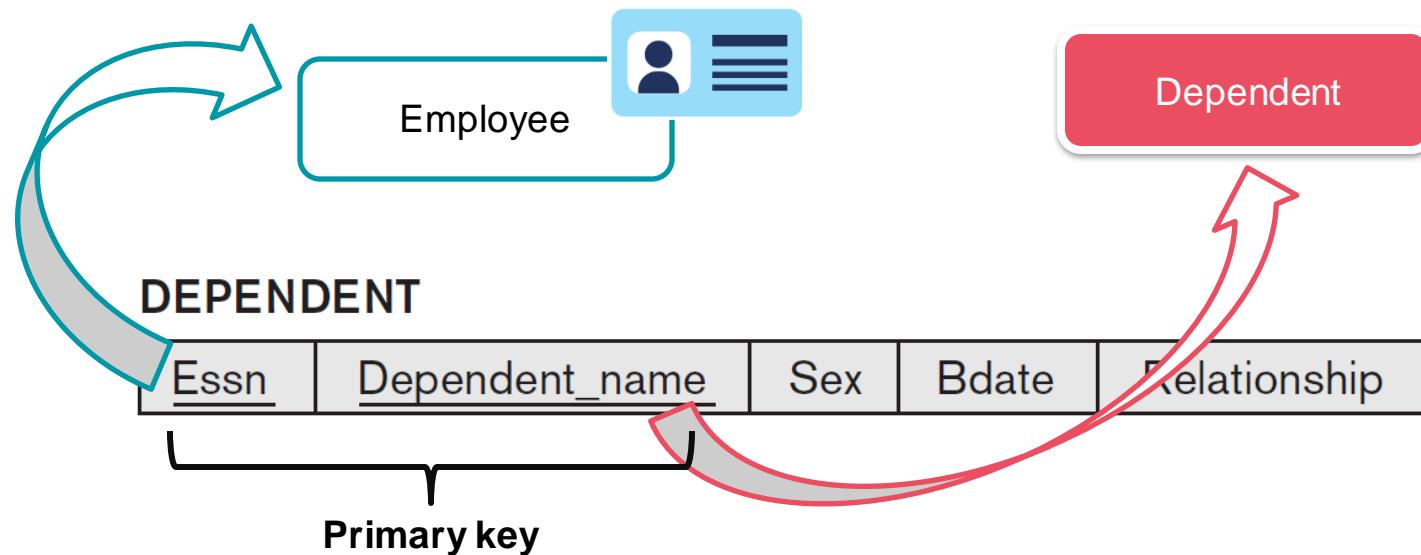
Algortimo de Mapeamento



Algortimo de Mapeamento



Mapeamento - Entidade Fraca



Entidade Fraca identificada pela Chave (PK e FK)

Algoritmo de Mapeamento

- Foreign Key
- Merge dos relacionamentos
- Cross-reference

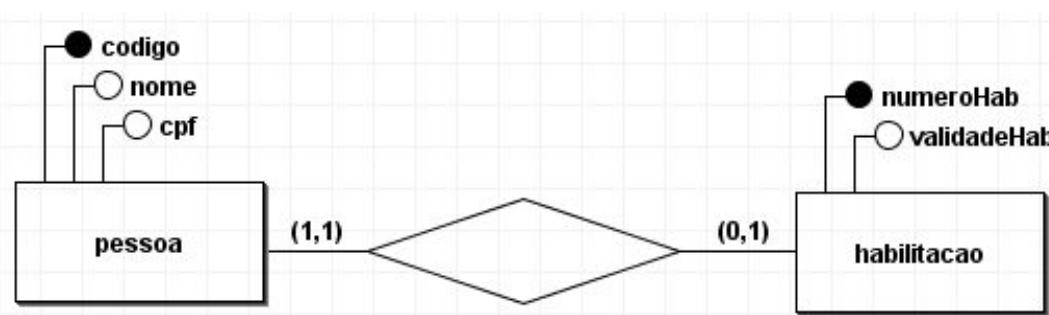
Abordagem



Relacionamento Binário 1:1

Algoritmo de Mapeamento

- PK de S \rightarrow FK em T



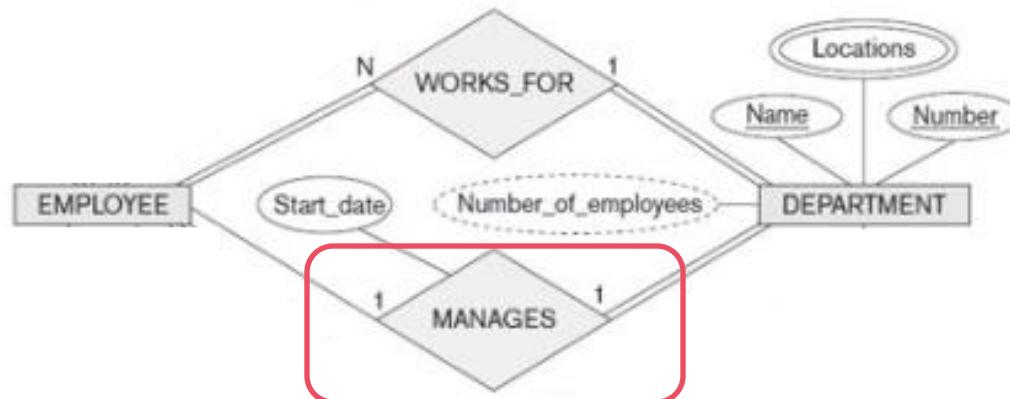
Participação Total

Abordagem FK

Relacionamento Binário 1:1

Algoritmo de Mapeamento

- PK de S -> FK em T



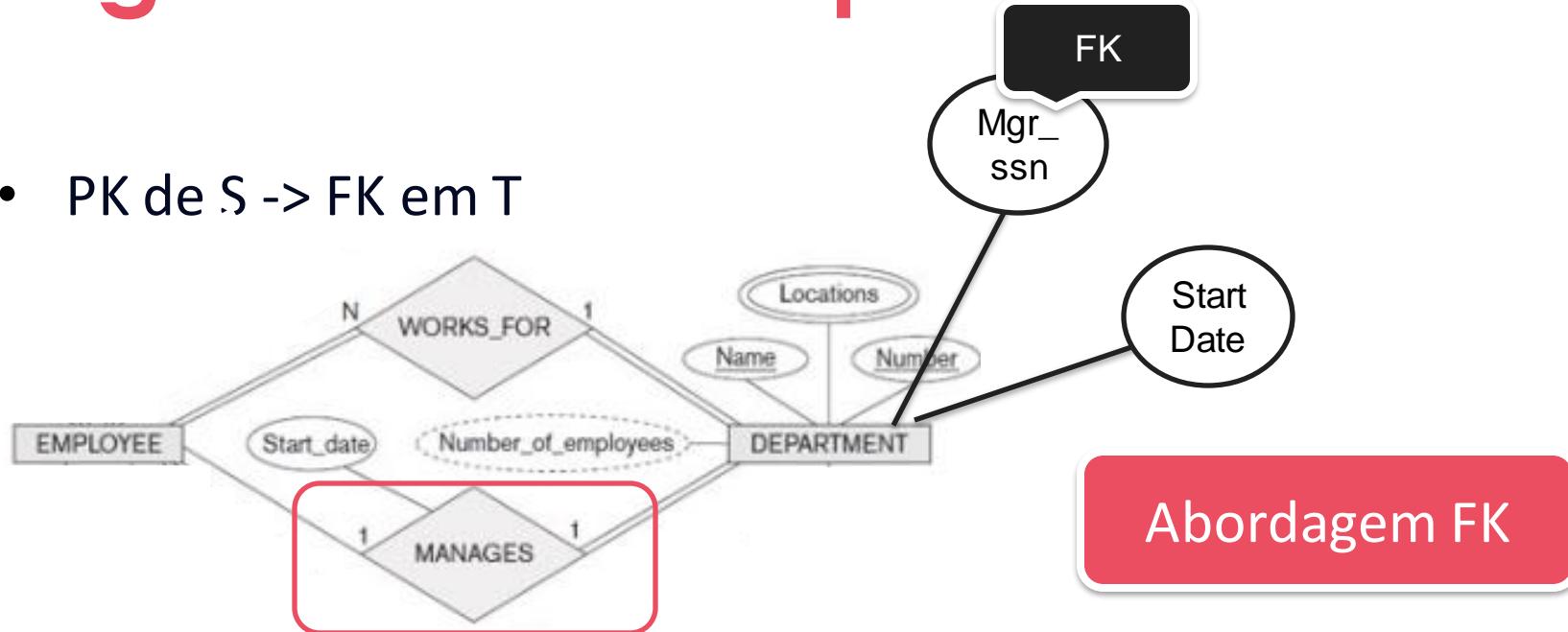
Participação Total

Abordagem FK

Relacionamento Binário 1:1

Algoritmo de Mapeamento

- PK de S -> FK em T



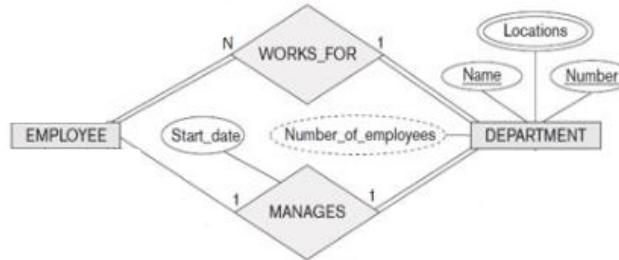
Abordagem FK

Relacionamento Binário 1:1

Algoritmo de Mapeamento

Relacionamento Total

- 2 entidades em 1 entidade



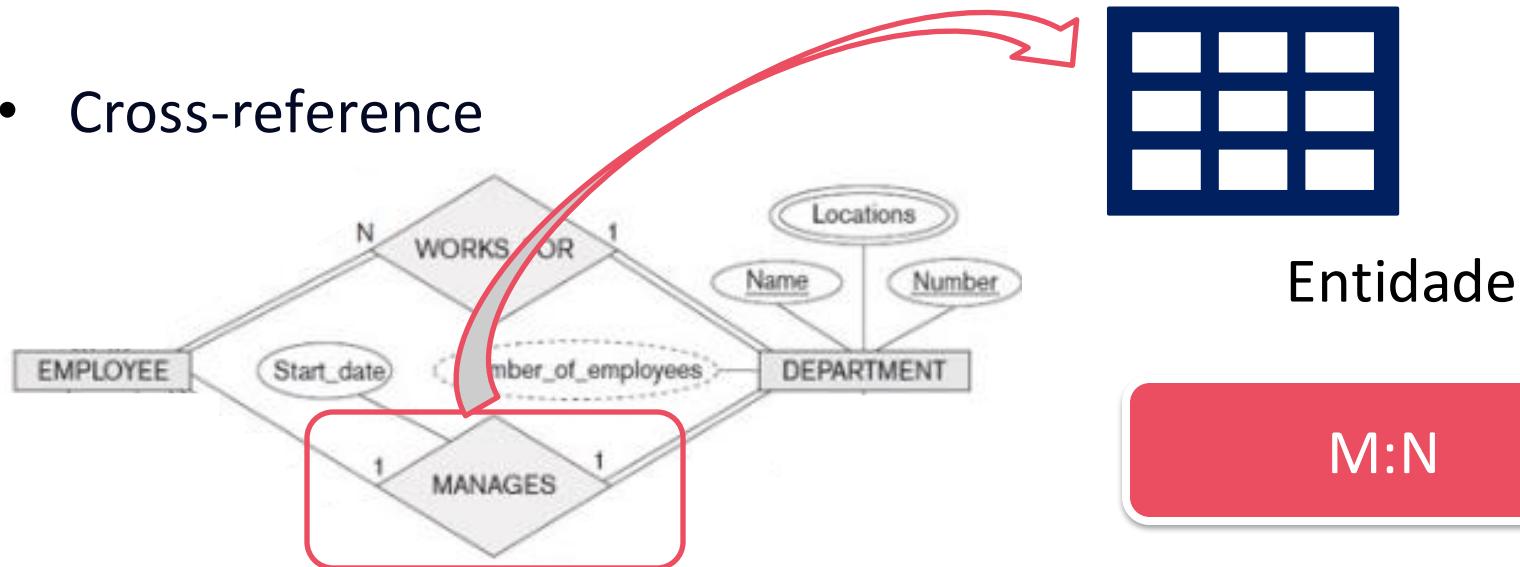
MgrDep

Merge

Relacionamento Binário 1:1

Algoritmo de Mapeamento

- Cross-reference



Relacionamento Binário 1:1

Algoritmo de Mapeamento

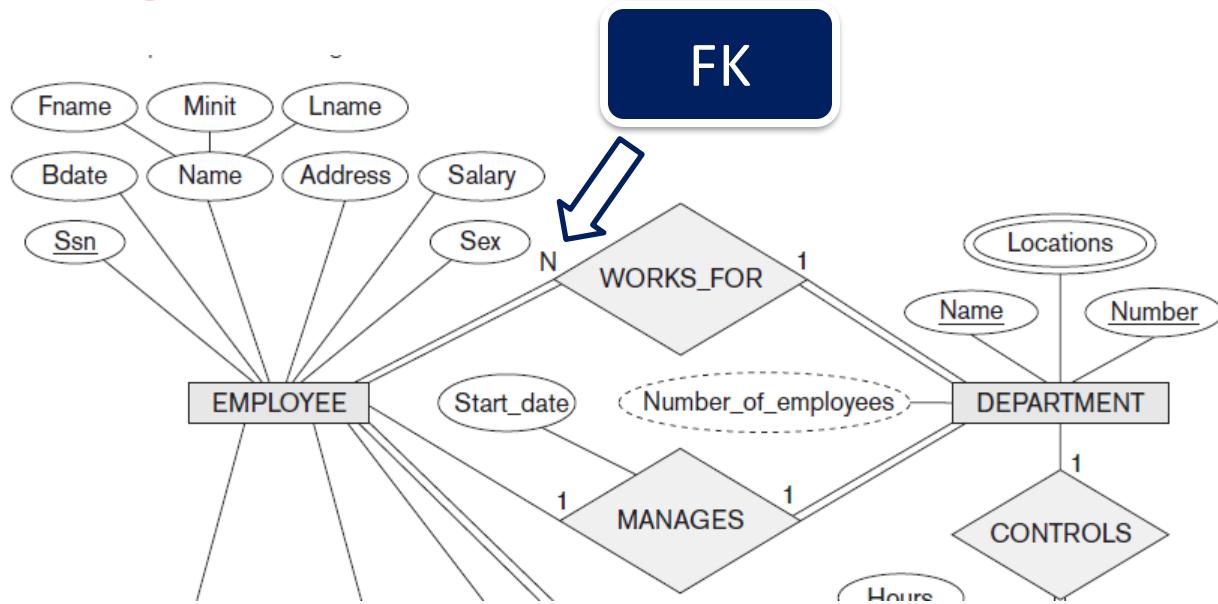
Abordagens:

- Foreign Key
- Relationship relation

1:N

Relacionamento Binário 1:N

Algoritmo de Mapeamento

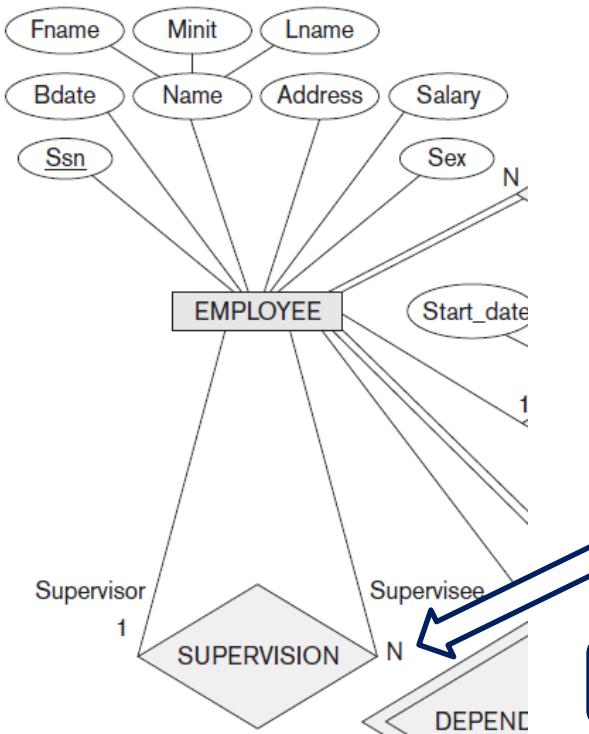


Relacionamento Binário
1:N

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
			▲▲▲						

Algoritmo de Mapeamento



EMPLOYEE

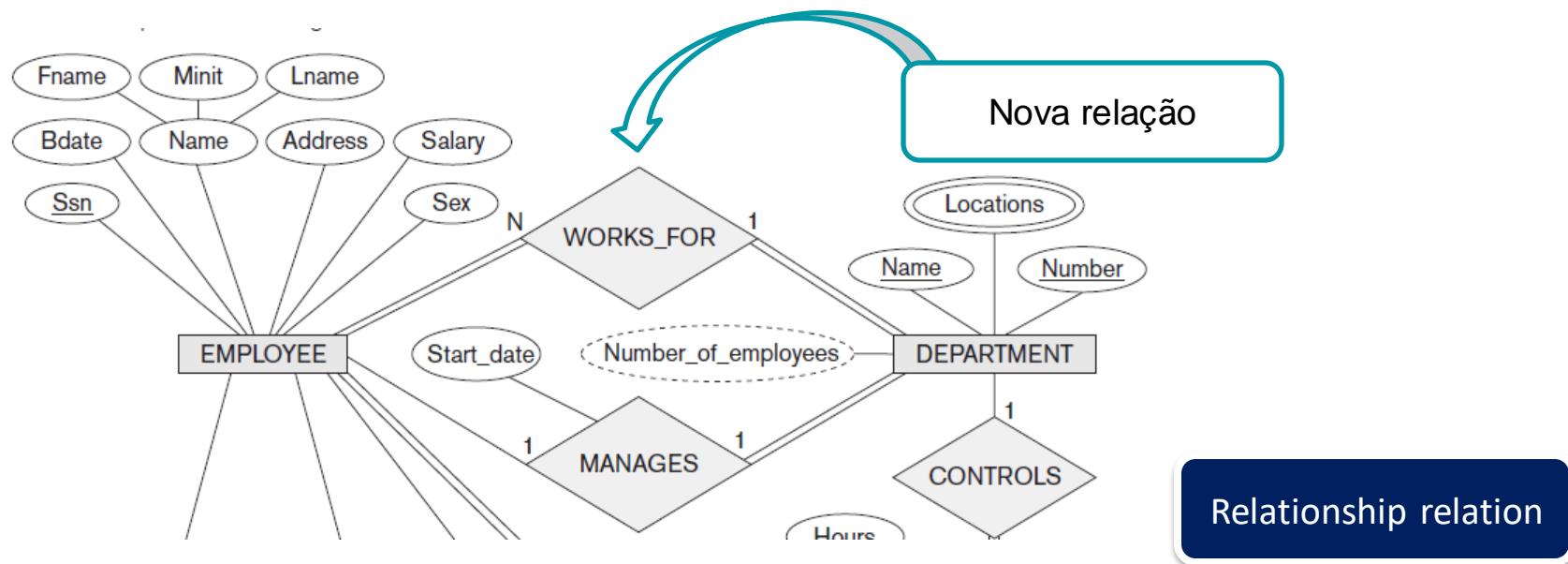
Fname	Minit	Lname	Ssn	Bdate	Address
-------	-------	-------	-----	-------	---------

Ssn

FK

Relacionamento Binário 1:N

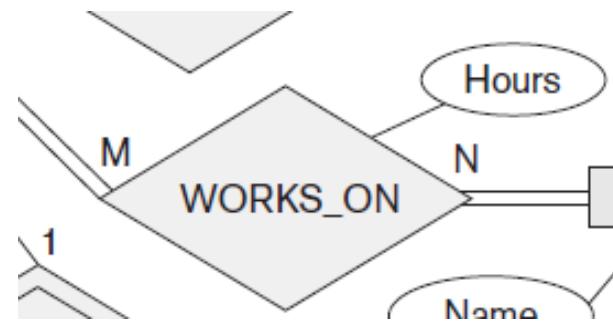
Algoritmo de Mapeamento



Algoritmo de Mapeamento

Abordagem:

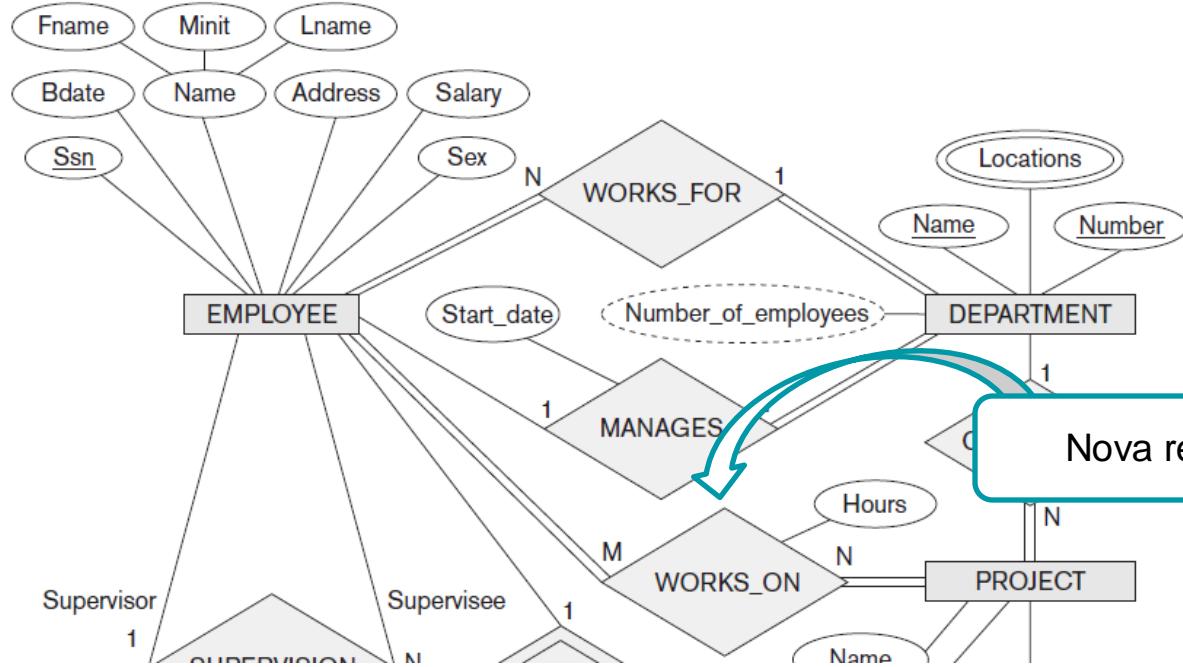
- Relationship relation



M:N

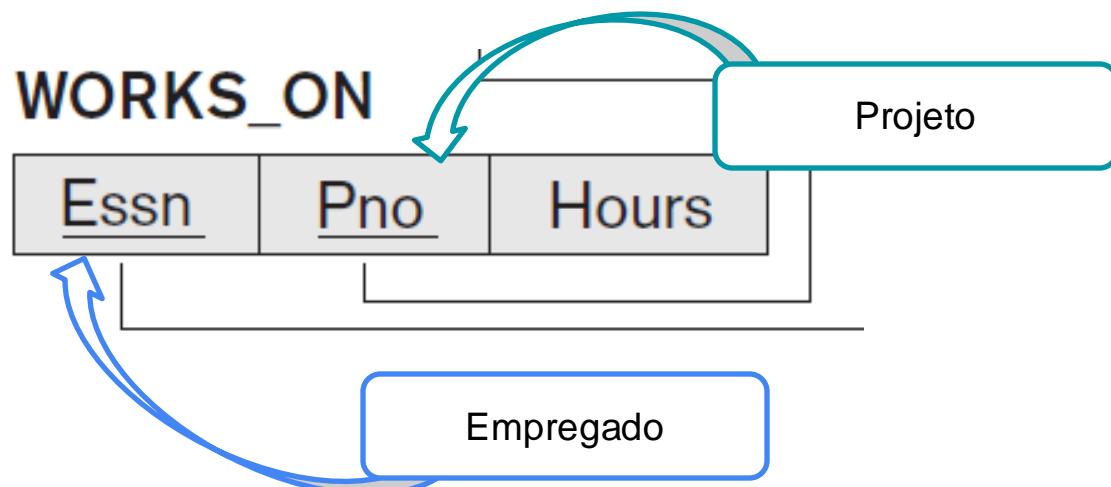
Relacionamento Binário N:M

Algoritmo de Mapeamento



Relacionamento
Binário N:M

Algoritmo de Mapeamento



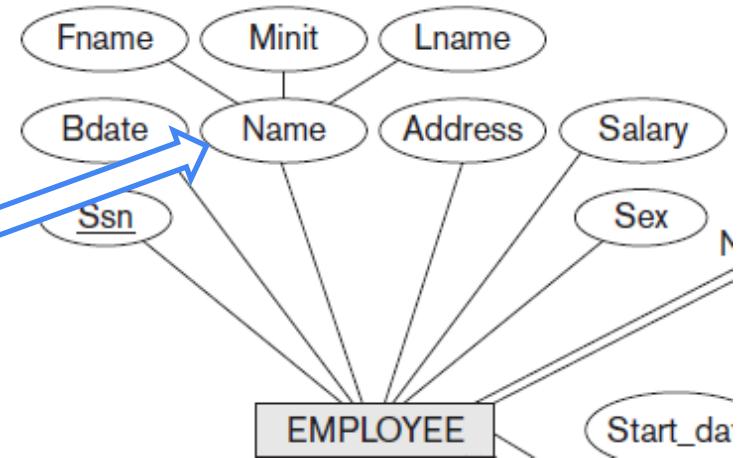
Relacionamento
Binário N:M

Algoritmo de Mapeamento

Abordagem:

- Componentes simples

Fname | Minit | Lname

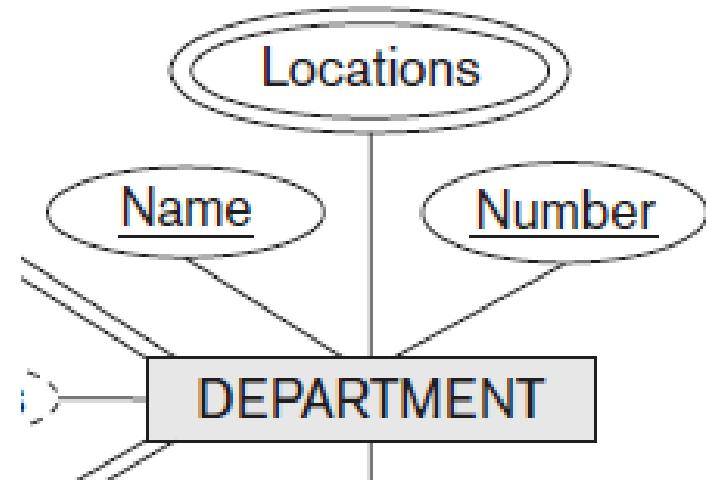


Atributos Compostos

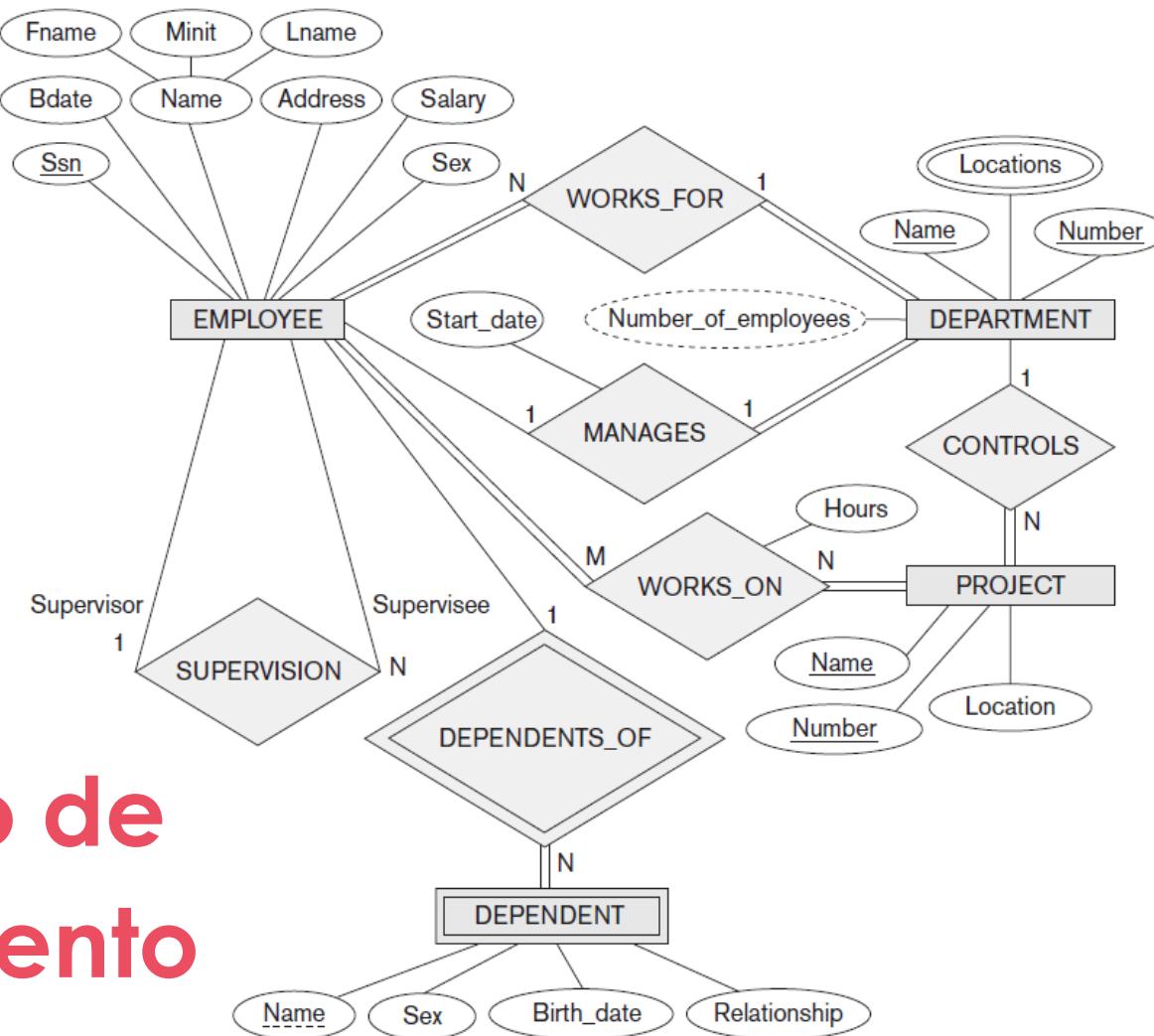
Algoritmo de Mapeamento

Abordagem:

- Relationship Relation



Atributos Multivalorados



Algoritmo de Mapeamento

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	-----	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
-------	---------	---------	----------------

DEPT_LOCATIONS

Dnumber	Dlocation
---------	-----------

PROJECT

Pname	Pnumber	Plocation	Dnum
-------	---------	-----------	------

WORKS_ON

Essn	Pno	Hours
------	-----	-------

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
------	----------------	-----	-------	--------------

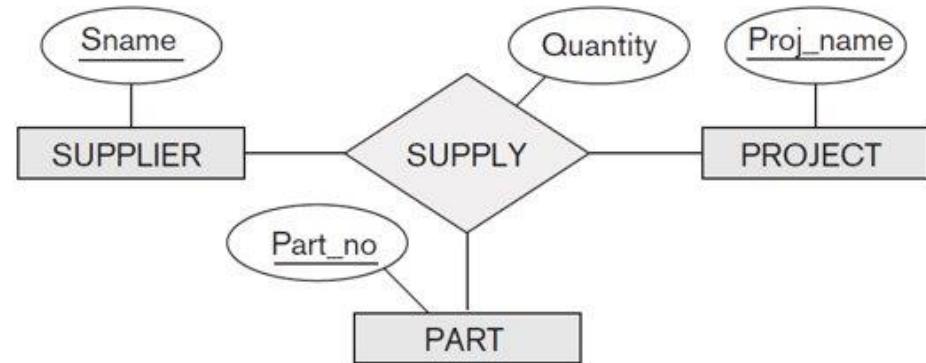
Figure 9.2

Result of mapping the COMPANY ER schema into a relational database schema.

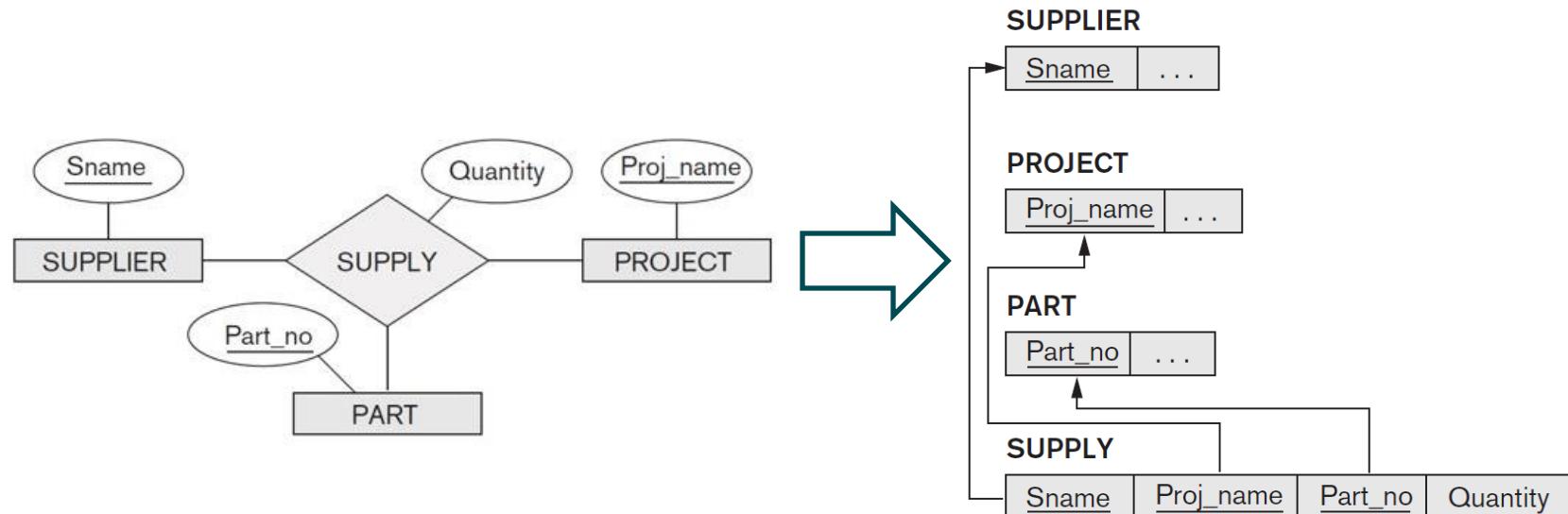
Algoritmo de Mapeamento

Abordagem:

- Relationship relation



Algoritmo de Mapeamento



Algoritmo de Mapeamento

Modelo ER

- Entidade
- Relação 1:1 ou 1:N
- Relação M:N
- Relacionamento N-ário

Modelo Relacional

- Relação (entidade)
- FK ou relação
- Relação ou 2 FKS
- Relação ou N FKS

Resumo

Algoritmo de Mapeamento

Modelo ER

- Atributo Simples
- Atributos Composto
- Atributo Multivalorado
- Conjunto de valores
- Atributo chave

Modelo Relacional

- Atributo
- Conjunto de atributos
- Relação ou FK
- Domínio
- PK (ou secundária)

Resumo

Mapeando os components do EER

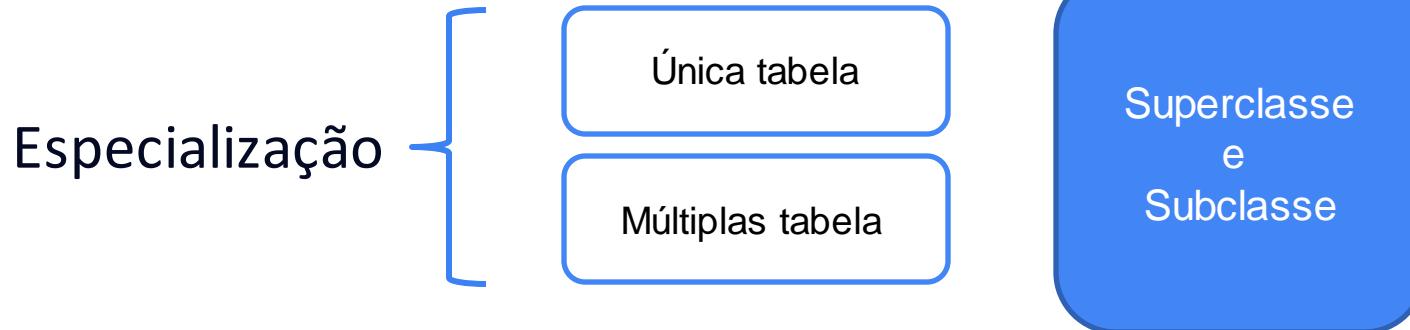
Algoritmo de Mapeamento



{SECRETARY, TECHNICIAN, ENGINEER}

SubClasse

Algoritmo de Mapeamento



{SECRETARY, TECHNICIAN, ENGINEER}

SubClasse

Algoritmo de Mapeamento

Opções

- Múltiplas relações - Sub/superclasse
- Múltiplas relações - Subclasse
- Relação única - 1 tipo de atributo
- Relação única - Múltiplos tipos de atributo

{SECRETARY, TECHNICIAN, ENGINEER}

SubClasse

Algoritmo de Mapeamento

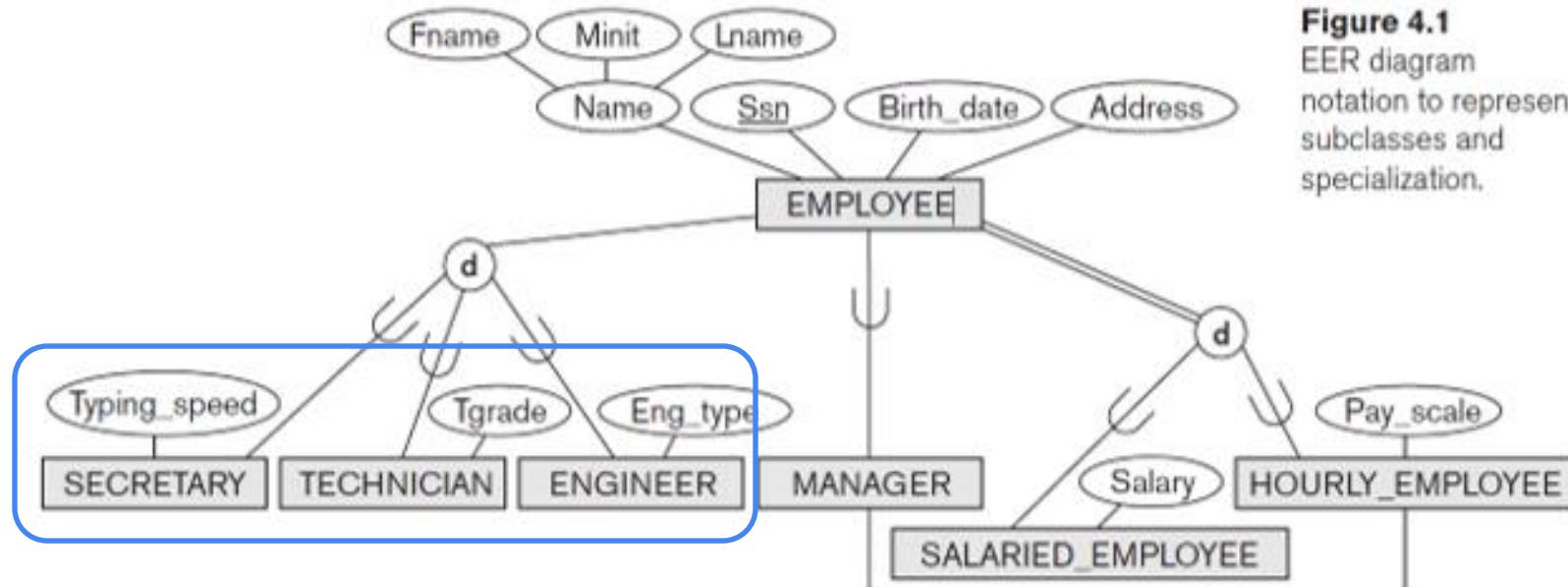


Figure 4.1
EER diagram
notation to represent
subclasses and
specialization.

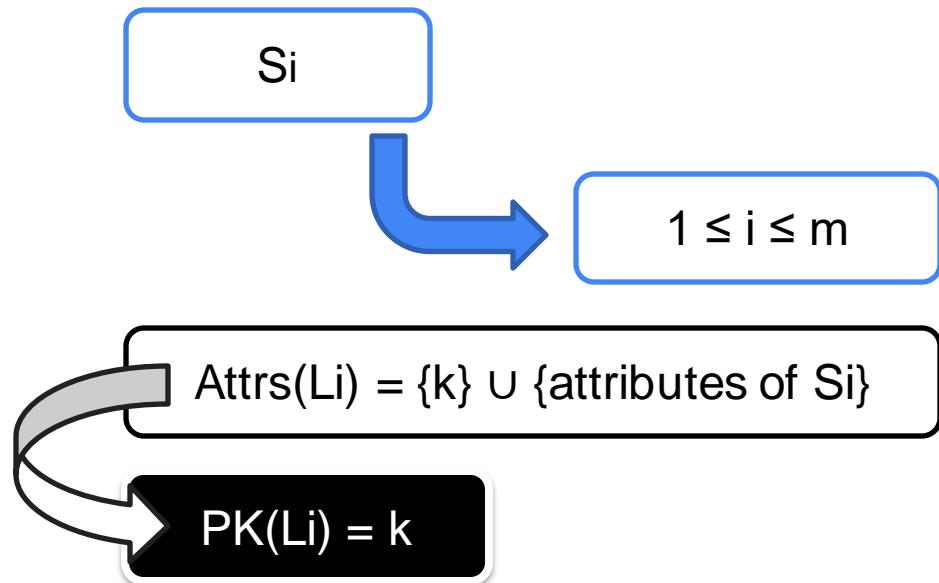
Algoritmo de Mapeamento

Relação L

- $\text{Attrs}(L) = \{k, a_1, \dots, a_n\}$

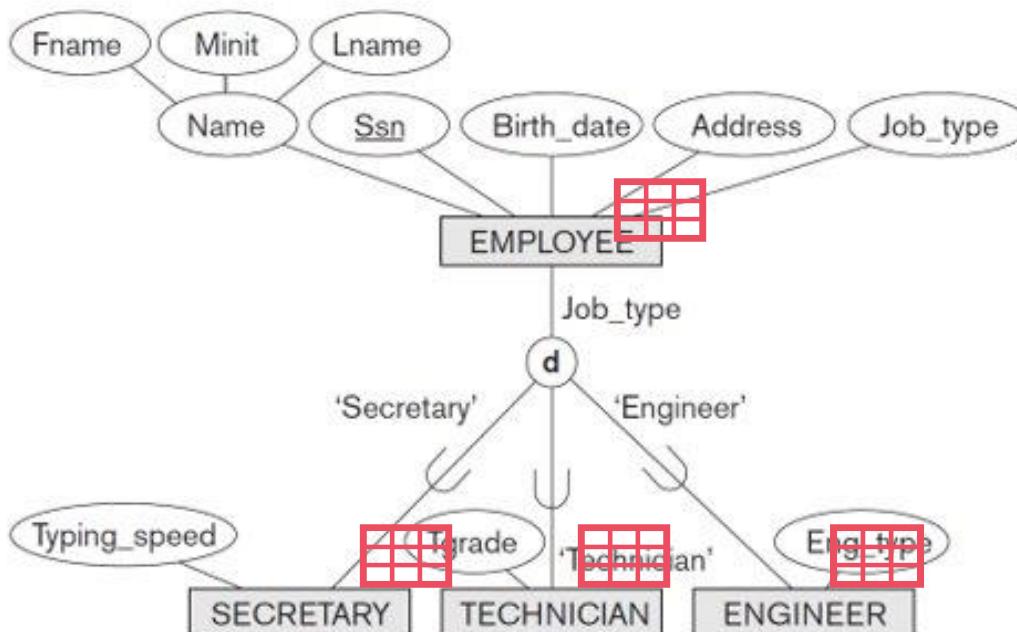
Relação Li

- $\text{Attrs}(L_i) = \{k\}$



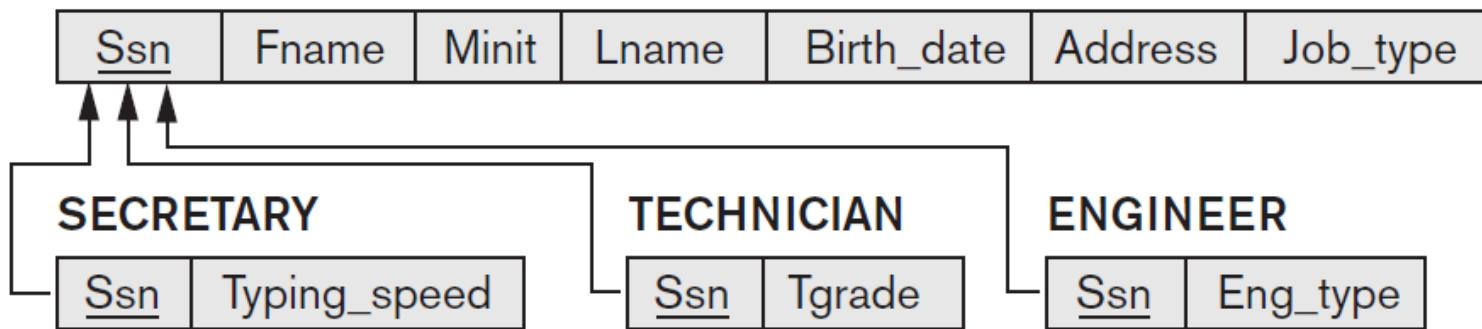
Múltiplas relações - Sub/superclasse

Algoritmo de Mapeamento



Algoritmo de Mapeamento

(a) EMPLOYEE



Múltiplas relações - Sub/superclasse

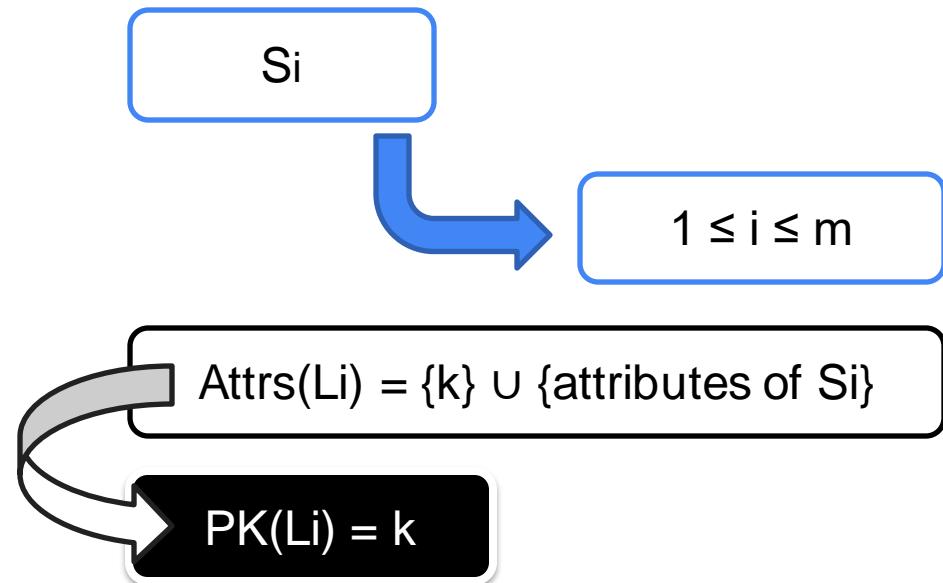
Algoritmo de Mapeamento

~~Relação L~~

- $\text{Attrs}(L) = \{k, a_1, \dots, a_n\}$

Relação Li

- $\text{Attrs}(Li) = \{k\}$

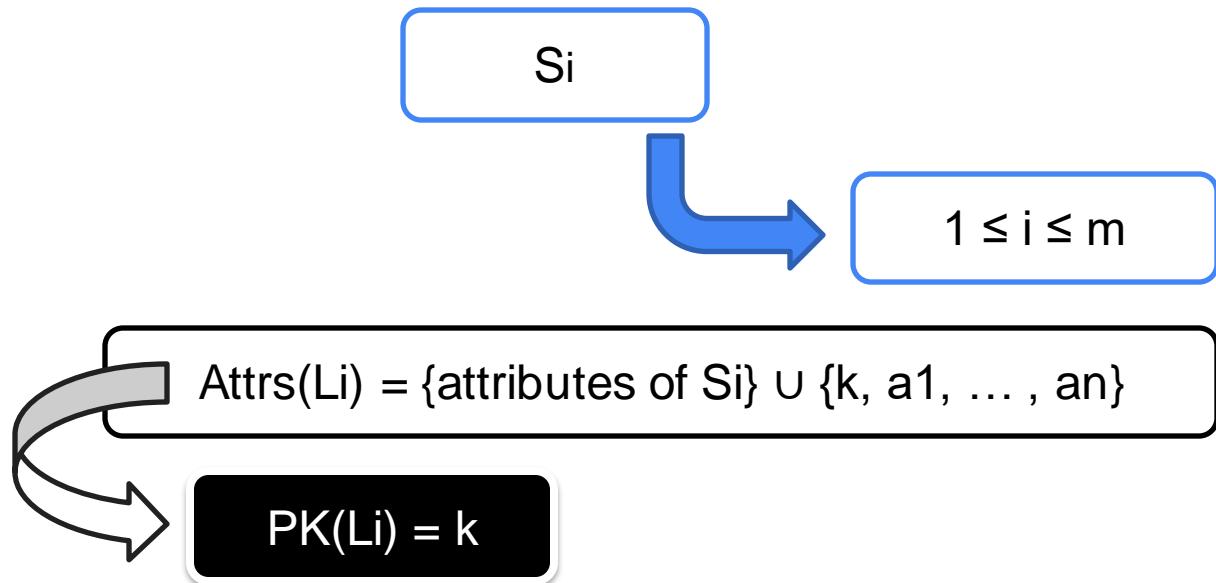


Múltiplas relações - Subclasse apenas

Algoritmo de Mapeamento

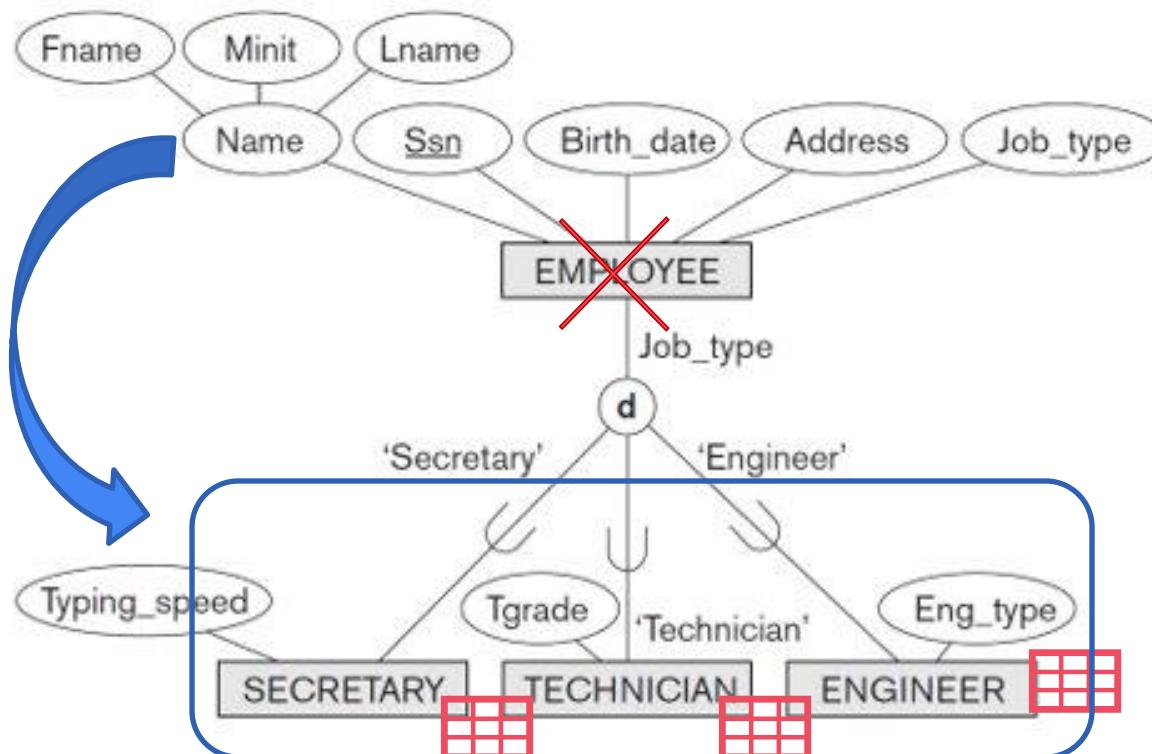
Relação L_i

- $\text{Attrs}(L_i) = \{k\}$



Múltiplas relações - Subclasse apenas

Algoritmo de Mapeamento



Algoritmo de Mapeamento

(b) CAR

<u>Vehicle_id</u>	License_plate_no	Price	Max_speed	No_of_passengers
-------------------	------------------	-------	-----------	------------------

TRUCK

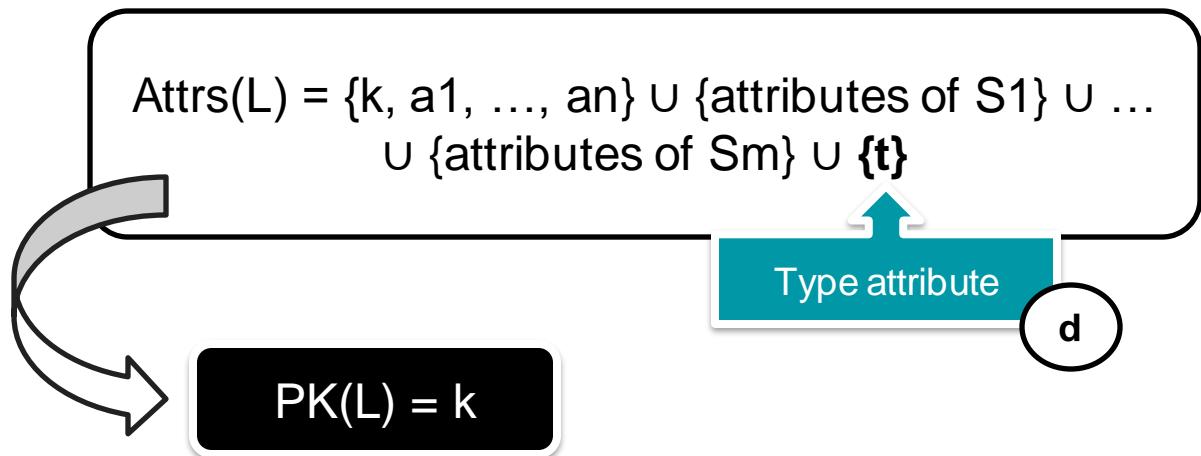
<u>Vehicle_id</u>	License_plate_no	Price	No_of_axles	Tonnage
-------------------	------------------	-------	-------------	---------

Múltiplas relações - Subclasse apenas

Algortimo de Mapeamento

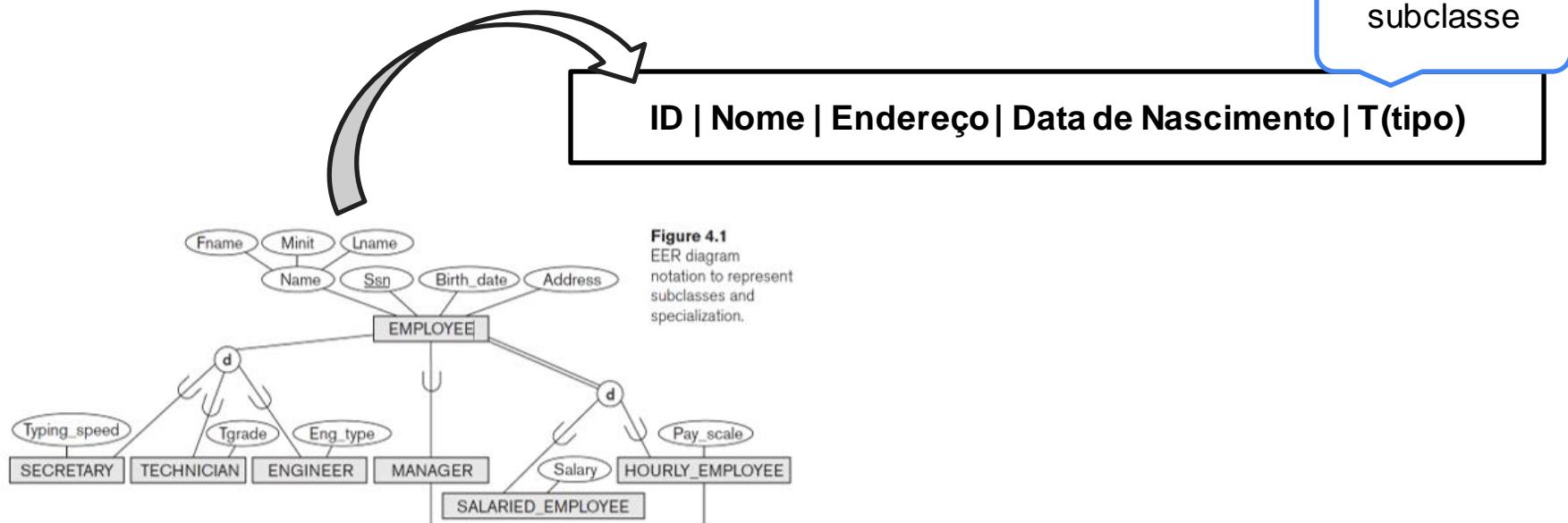
Relação L

- $\text{Attrs}(L) = \{k\}$



Relação única - 1 tipo de atributo

Algoritmo de Mapeamento



Relação única - 1 tipo de atributo

Algoritmo de Mapeamento

(c) EMPLOYEE

Ssn	Fname	Minit	Lname	Birth_date	Address	Job_type	Typing_speed	Tgrade	Eng_type
-----	-------	-------	-------	------------	---------	----------	--------------	--------	----------

Atributos de tipo

Relação única - 1 tipo de atributo

Algoritmo de Mapeamento

Relação L

- $\text{Attrs}(L) = \{k\}$

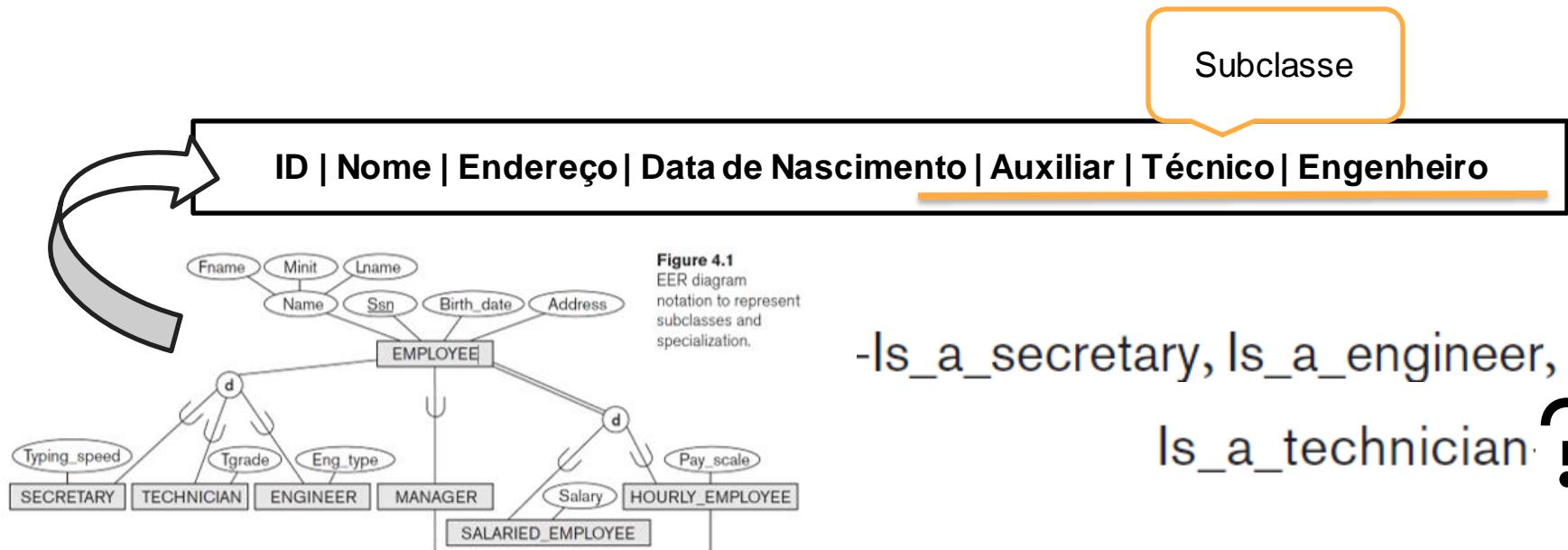
$\text{Attrs}(L) = \{k, a_1, \dots, a_n\} \cup \{\text{attributes of } S_1\} \cup \dots \cup \{\text{attributes of } S_m\} \cup \{t_1, t_2, \dots, t_m\}$

True OR False

$\text{PK}(L) = k$

Relação única - Múltiplos atributos Type

Algoritmo de Mapeamento

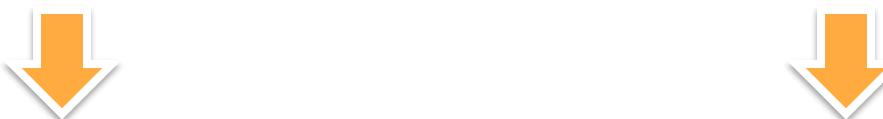


Relação única - Múltiplos atributos Type

Algoritmo de Mapeamento

Múltiplos Atributos de tipo

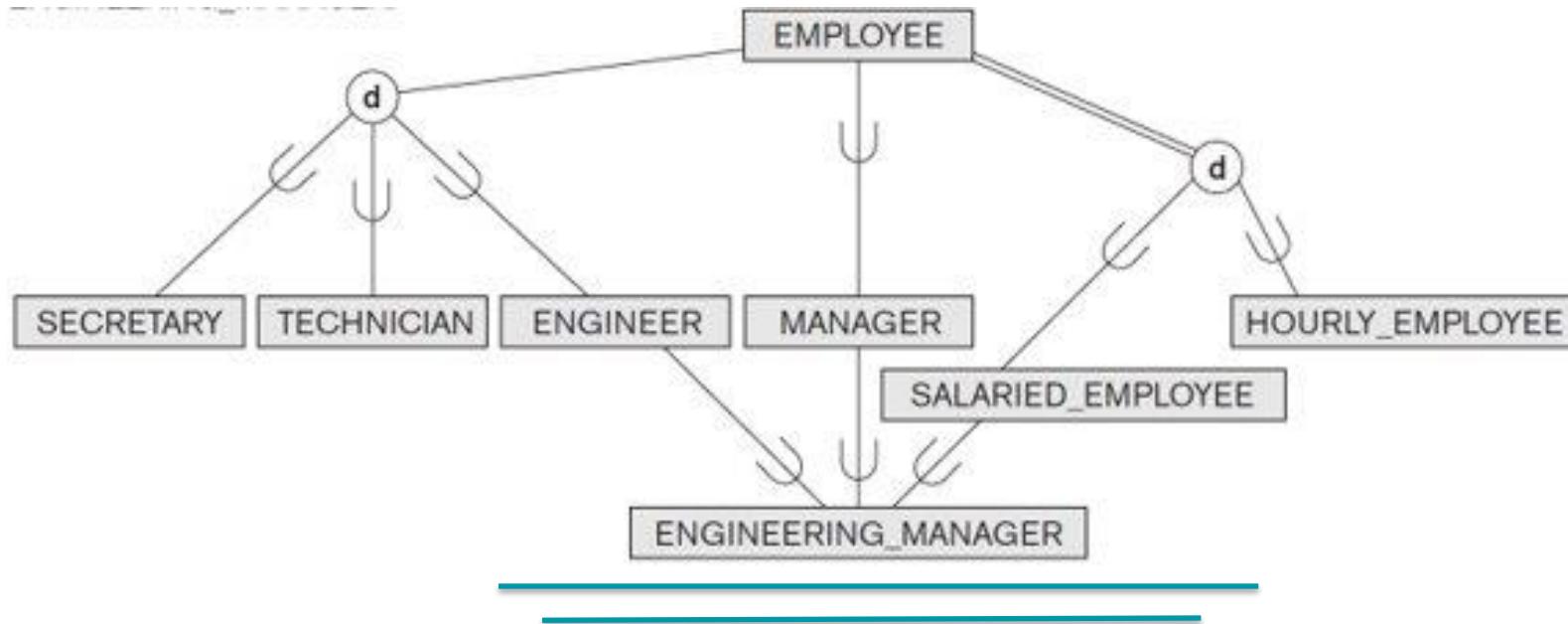
(d) PART



Part_no	Description	Mflag	Drawing_no	Manufacture_date	Batch_no	Pflag	Supplier_name	List_price

Relação única - Múltiplos atributos Type

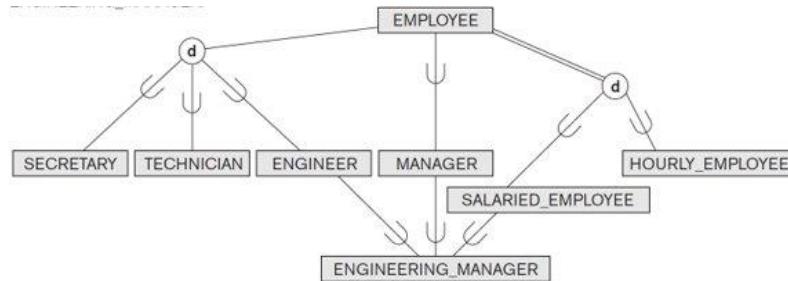
Algoritmo de Mapeamento



Herança Múltipla

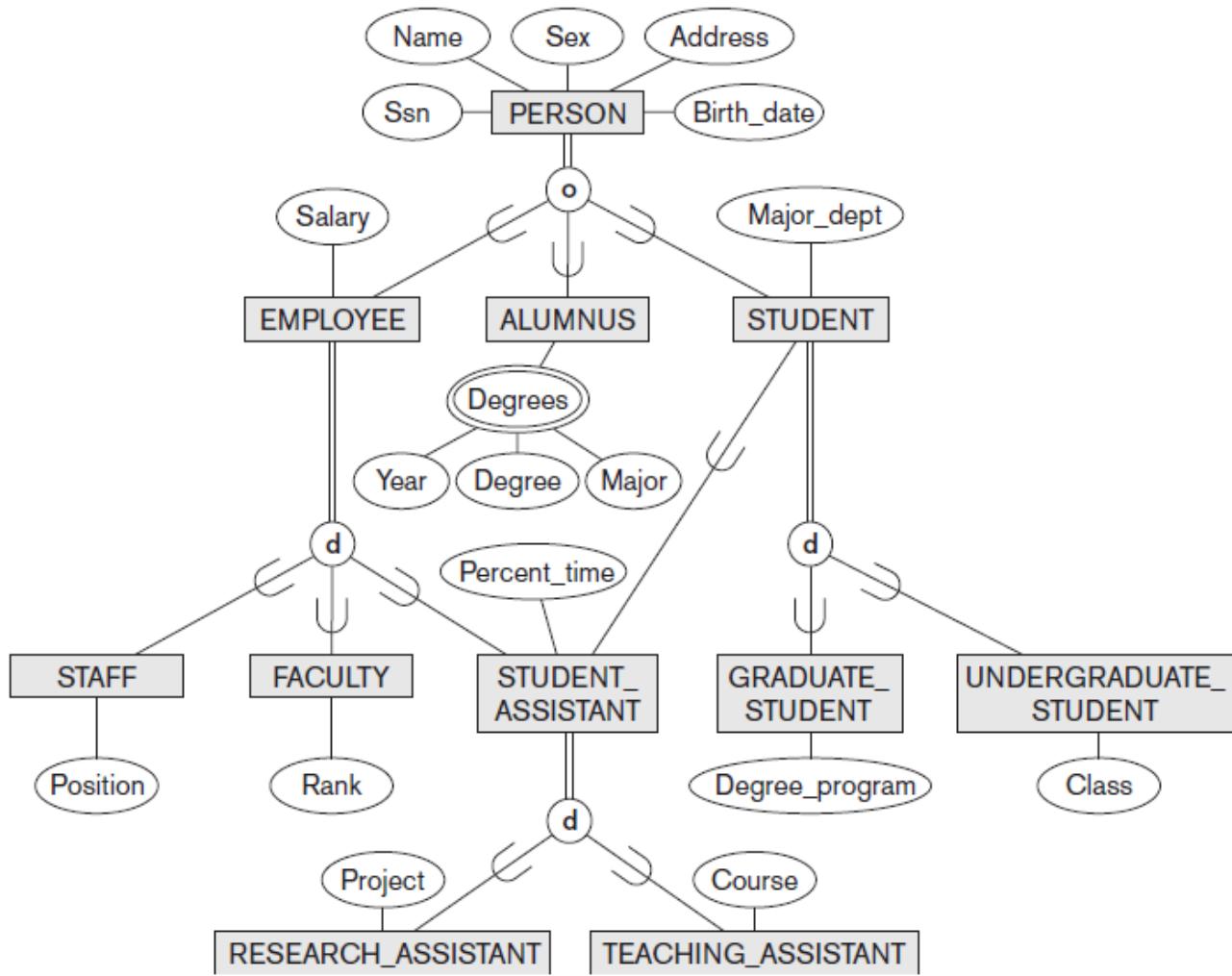
Algoritmo de Mapeamento

- Mesmo atributo chave
- Mapeamento: Opções anteriores



Herança Múltipla

Herança Múltipla



Algoritmo de Mapeamento

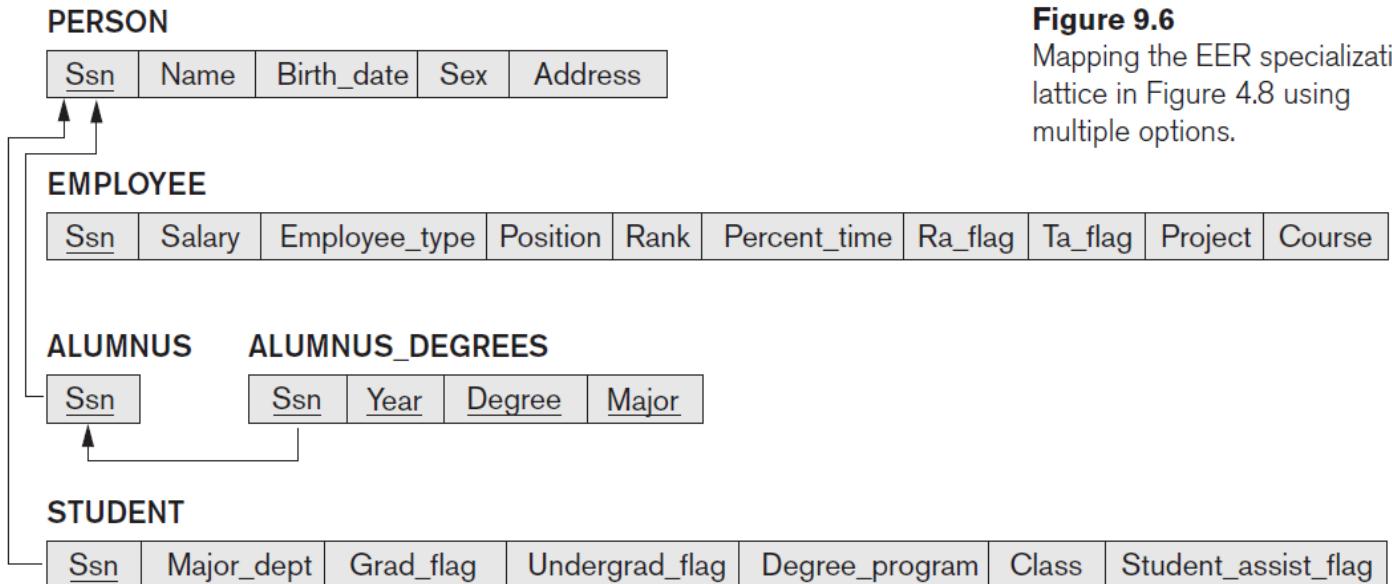


Figure 9.6

Mapping the EER specialization lattice in Figure 4.8 using multiple options.

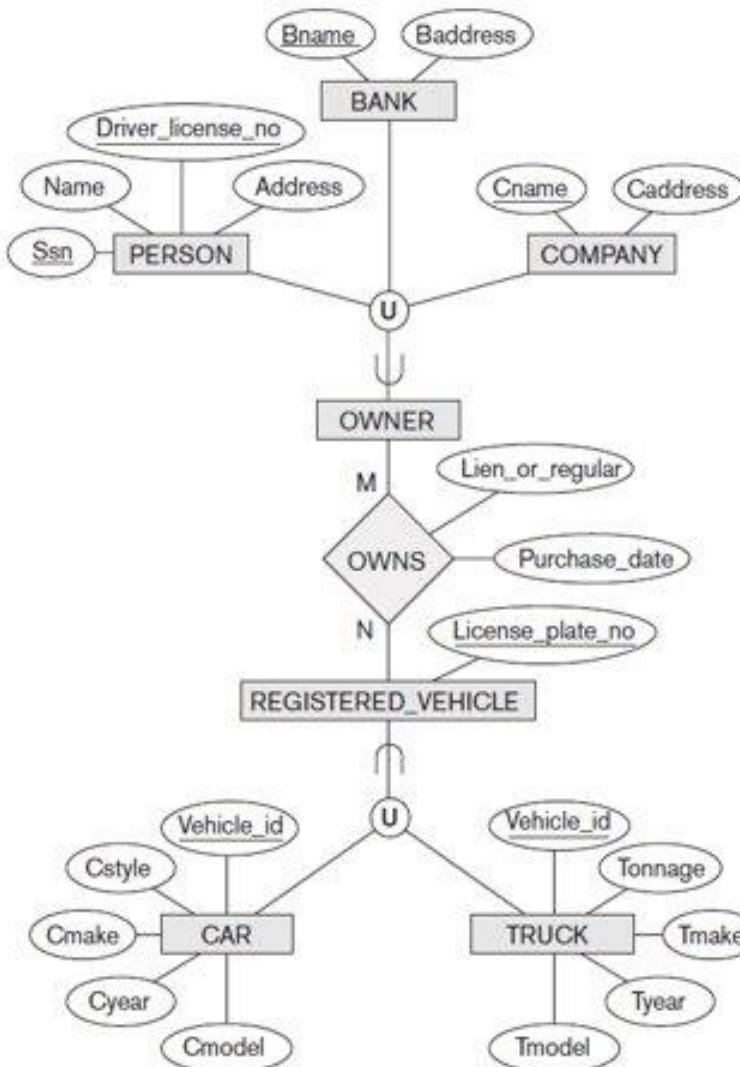
Algoritmo de Mapeamento

- Superclasses
- Tipos de entidades diferentes



UNION TYPE

Algoritmo de Mapeamento



Algoritmo de Mapeamento

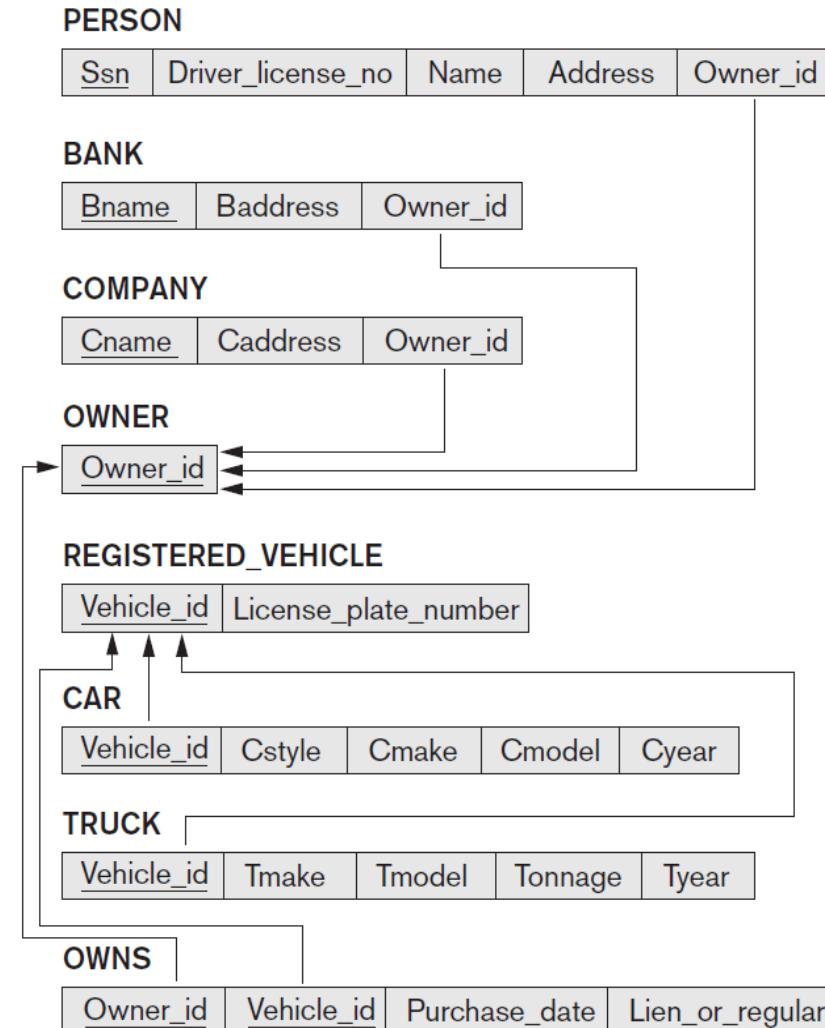
Chave substituta



Chave Primária

Chave Estrangeira

UNION TYPE



Handos on: Estudo de caso

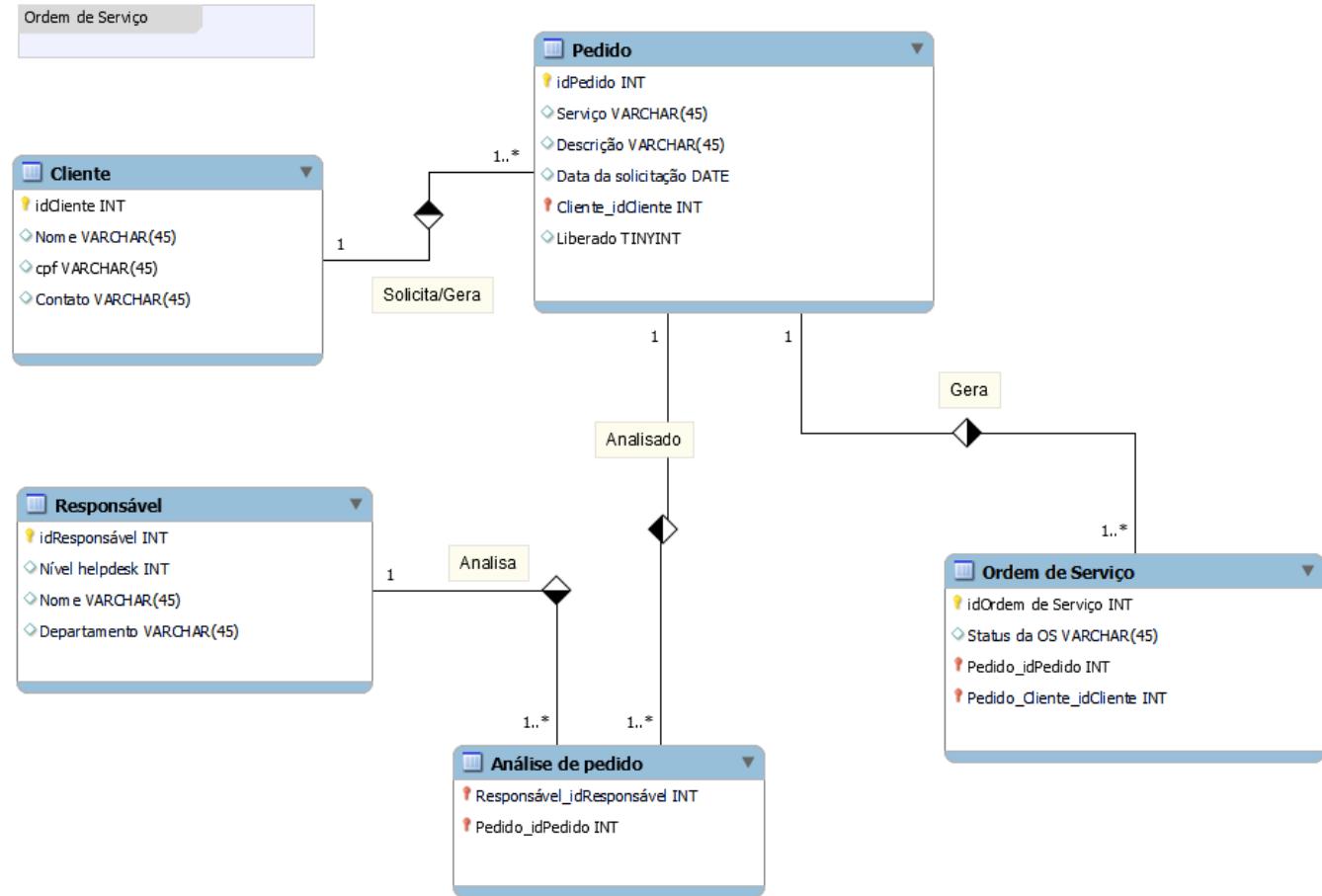
Hands on

Objetivo:

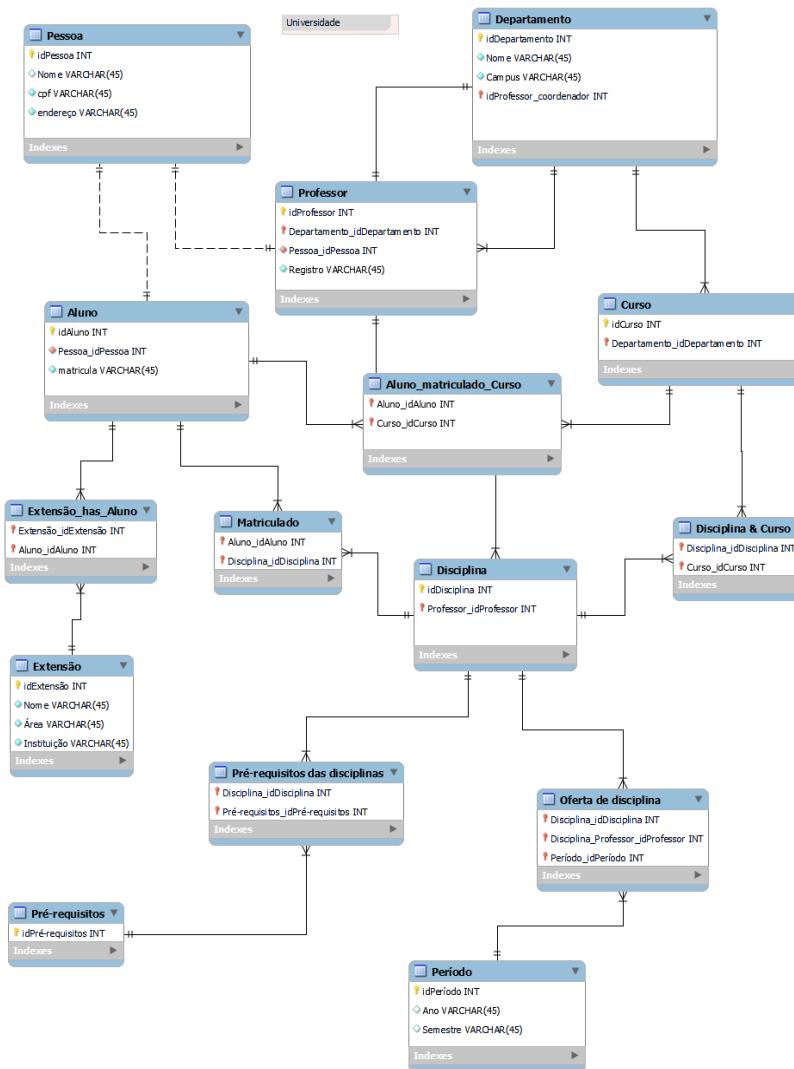
- Mapear os esquemas conceituais de OS,
Universidade e E-commerce

Herança Múltipla

Hands on: OS



Hands on: Universidade



Etapa 2

Explorando a Linguagem SQL

// Explorando a Linguagem de Consulta a Banco de Dados SQL

Explorando a SQL



Introdução ao SQL

1970

Linguagem para:

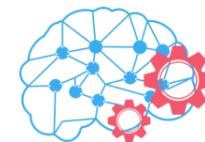
- Manipulação de dados
- Execução de operações

DBAs

BI

Devs

DS



Introdução ao SQL

1970

Objetivo

- Modificação de dados e estrutura
- Adicionar, remover ou atualizar linhas
- Recuperação de um subconjunto de info do BD

OLTP

OTAP



Introdução ao SQL

1970

Classificações

- **DDL** – Data Definition Language
- **DML** – Data Manipulation Language
- **DCL** – Data Control Language
- **DQL** – Data Query Language



Introdução ao SQL

Classificações

- DDL – Data Definition Language

CREATE | DROP | ALTER

INSERT | UPDATE | DELETE

RENAME | TRUNCATE | MERGE



Introdução ao SQL

Classificações

- DDL – Data Definition Language

USUÁRIO
ESQUEMA
STATEMENTS
INDEXING



Introdução ao SQL

Classificações

- **DDL** – Data Definition Language

13.1.12 CREATE DATABASE Statement

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name
      [create_option] ...

create_option: [DEFAULT] {
    CHARACTER SET [=] charset_name
    | COLLATE [=] collation_name
    | ENCRYPTION [=] {'Y' | 'N'}
}
```

Introdução ao SQL

Classificações

- DDL – Data Definition Language

Primeiro Exemplo



Introdução ao SQL

Classificações

- **DDL – Data Definition Language**

Primeiro Exemplo



```
mysql> use teste;
Database changed
mysql> CREATE TABLE person
-> (person_id SMALLINT UNSIGNED,
-> fname VARCHAR(20),
-> lname VARCHAR(20),
-> gender ENUM('M','F'),
-> birth_date DATE,
-> street VARCHAR(30),
-> city VARCHAR(20),
-> state VARCHAR(20),
-> country VARCHAR(20),
-> postal_code VARCHAR(20),
-> CONSTRAINT pk_person PRIMARY KEY (person_id)
-> );
Query OK, 0 rows affected (0.08 sec)
```

```
MySQL OK, 0 rows affected (0.08 sec)
-> ;
-> CONSTRAINT pk_person PRIMARY KEY (person_id)
-> to person code automatically
```

Hands On! **Criando objeto pessoa**

**“Falar é fácil.
Mostre-me o código!”**

Linus Torvalds

Introdução ao SQL

Classificações

- **DML** – Data Manipulation Language

INSERT | UPDATE

DELETE | MERGE



Introdução ao SQL

Classificações

GRANT | REVOKE

- **DCL** – Data Control Language

SELECT

- **DQL** – Data Query Language



Introdução ao SQL

Statement

- Comando/instrução
- Reconhecido pelo BD
- Retorno: registro de dados



Introdução ao SQL

Cláusulas SQL

- Função -> instrução
- Algumas são obrigatórias

```
SELECT * FROM table_name;
```



Introdução ao SQL

Cadê o FROM?

```
SELECT now();
```

- Datetime da máquina

```
SELECT now() FROM dual;
```



Introdução ao SQL

Cláusulas SQL

SELECT
FROM
WHERE

ORDER BY
GROUP BY
HAVING

```
SELECT * FROM table_name;
```



Introdução ao SQL

Introdução ao SQL

Termos – SQL

Identificador

Operador

Constante

Expressão



Introdução ao SQL

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

DEPT_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

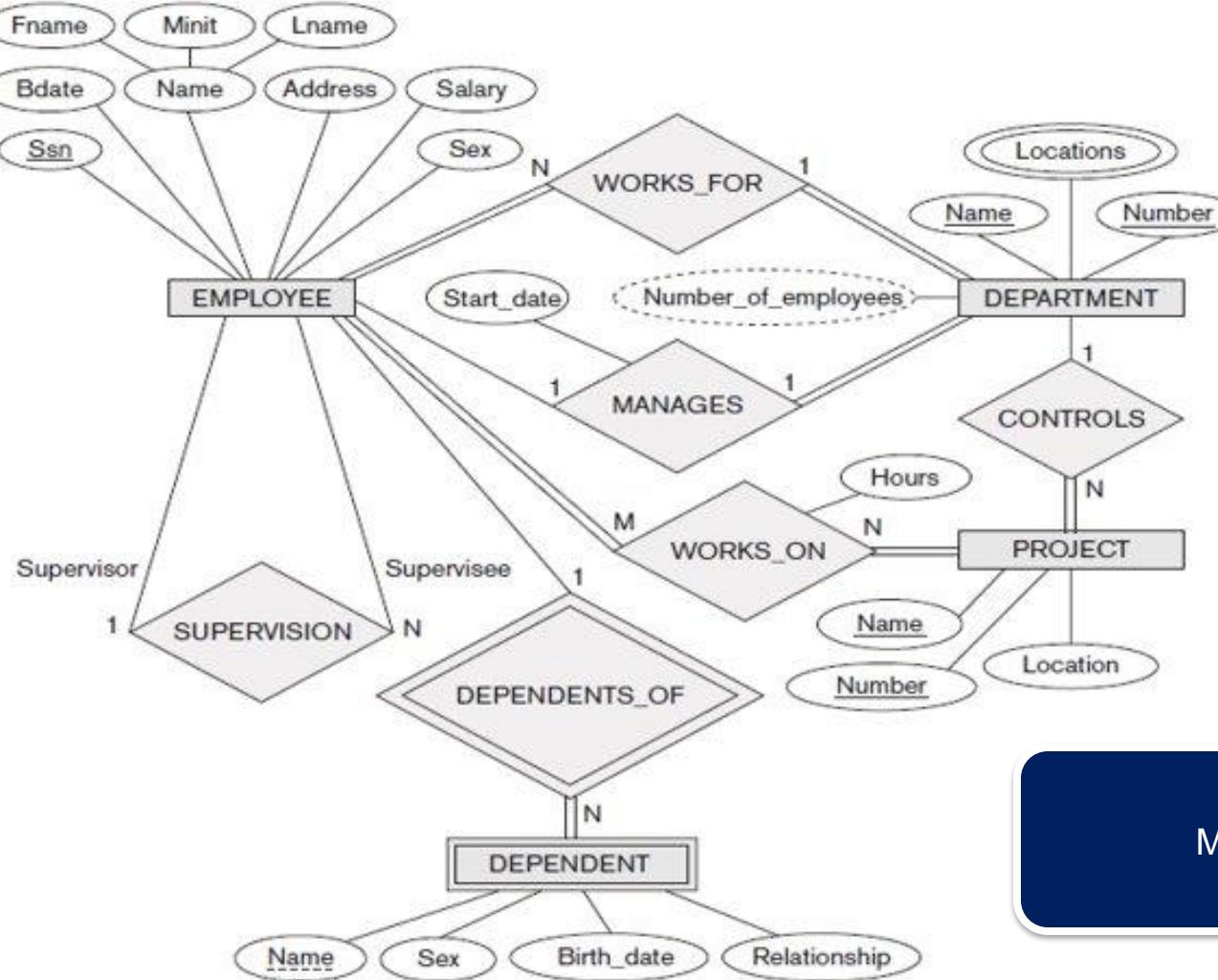
DEPENDENT

<u>Essn</u>	Dependent_name	Sex	Bdate	Relationship
-------------	----------------	-----	-------	--------------

MODELO RELACIONAL

Figure 5.5

Schema diagram for the COMPANY relational database schema.



MODELO ER

Hands On! **Manipulando o esquema do BD**

**“Falar é fácil.
Mostre-me o código!”**

Linus Torvalds

Tipos de dados no Mysql

Tipos de dados

Caracteres

- Fixo - 255 bytes
- Variável - 65.535 bytes

```
char(20) /* fixed-length */  
varchar(20) /* variable-length */
```



Tipos de dados

Texts types

- mediumtext - 16,777,215 bytes
- longtext - 4,294,967,295 characters
- ...

documents,

emails

, XML



Tipos de dados

Conjunto de caracteres

- Alfabeto, Português, Inglês...

SHOW CHARACTER SET;

Charset	Description	Default collation	Maxlen
big5	Big5 Traditional Chinese	big5_chinese_ci	2
dec8	DEC West European	dec8_swedish_ci	1
cp850	DOS West European	cp850_general_ci	1
hp8	HP West European	hp8_english_ci	1
koi8r	KOI8-R Relcom Russian	koi8r_general_ci	1
latin1	cp1252 West European	latin1_swedish_ci	1
latin2	ISO 8859-2 Central European	latin2_general_ci	1
swe7	7bit Swedish	swe7_swedish_ci	1
ascii	US ASCII	ascii_general_ci	1
ujis	EUC-JP Japanese	ujis_japanese_ci	3
sjis	Shift-JIS Japanese	sjis_japanese_ci	2
hebrew	ISO 8859-8 Hebrew	hebrew_general_ci	1
tis620	TIS620 Thai	tis620_thai_ci	1
euckr	EUC-KR Korean	euckr_korean_ci	2
koi8u	KOI8-U Ukrainian	koi8u_general_ci	1
gb2312	GB2312 Simplified Chinese	gb2312_chinese_ci	2
greek	ISO 8859-7 Greek	greek_general_ci	1
cp1250	Windows Central European	cp1250_general_ci	1
gbk	GBK Simplified Chinese	gbk_chinese_ci	2
latin5	ISO 8859-9 Turkish	latin5_turkish_ci	1

...

Fonte: livro 2 de referência

Tipos de dados

Conjunto de caracteres

- Alfabeto, Português, Inglês...

SHOW CHARACTER SET;

Maxlen > 1 = multibyte char set

Charset	Description	Default collation	Maxlen
big5	Big5 Traditional Chinese	big5_chinese_ci	2
dec8	DEC West European	dec8_swedish_ci	1
cp850	DOS West European	cp850_general_ci	1
hp8	HP West European	hp8_english_ci	1
koi8r	KOI8-R Relcom Russian	koi8r_general_ci	1
latin1	cp1252 West European	latin1_swedish_ci	1
latin2	ISO 8859-2 Central European	latin2_general_ci	1
swe7	7bit Swedish	swe7_swedish_ci	1
ascii	US ASCII	ascii_general_ci	1
ujis	EUC-JP Japanese	ujis_japanese_ci	3
sjis	Shift-JIS Japanese	sjis_japanese_ci	2
hebrew	ISO 8859-8 Hebrew	hebrew_general_ci	1
tis620	TIS620 Thai	tis620_thai_ci	1
euckr	EUC-KR Korean	euckr_korean_ci	2
koi8u	KOI8-U Ukrainian	koi8u_general_ci	1
gb2312	GB2312 Simplified Chinese	gb2312_chinese_ci	2
greek	ISO 8859-7 Greek	greek_general_ci	1
cp1250	Windows Central European	cp1250_general_ci	1
gbk	GBK Simplified Chinese	gbk_chinese_ci	2
latin5	ISO 8859-9 Turkish	latin5_turkish_ci	1

...

Fonte: livro 2 de referência

Tipos de dados

Texts data

- < 64KB

Text type	Maximum number of bytes
Tinytext	255
Text	65,535
Mediumtext	16,777,215
Longtext	4,294,967,295

Fonte: livro 2 de referência



Tipos de dados

Table 2-4. MySQL floating-point types

Type	Numeric range
Float(p,s)	-3.402823466E+38 to -1.175494351E-38 and 1.175494351E-38 to 3.402823466E+38
Double(p,s)	-1.7976931348623157E+308 to -2.2250738585072014E-308 and 2.2250738585072014E-308 to 1.7976931348623157E+308

Dados numéricos

Table 2-3. MySQL integer types

Type	Signed range	Unsigned range
Tinyint	-128 to 127	0 to 255
Smallint	-32,768 to 32,767	0 to 65,535
Mediumint	-8,388,608 to 8,388,607	0 to 16,777,215
Int	-2,147,483,648 to 2,147,483,647	0 to 4,294,967,295
Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	0 to 18,446,744,073,709,551,615



Tipos de dados

Dados temporais

Table 2-5. MySQL temporal types

Type	Default format	Allowable values
Date	YYYY-MM-DD	1000-01-01 to 9999-12-31
Datetime	YYYY-MM-DD HH:MI:SS	1000-01-01 00:00:00 to 9999-12-31 23:59:59
Timestamp	YYYY-MM-DD HH:MI:SS	1970-01-01 00:00:00 to 2037-12-31 23:59:59
Year	YYYY	1901 to 2155
Time	HHH:MI:SS	-838:59:59 to 838:59:59



Constraints em SQL

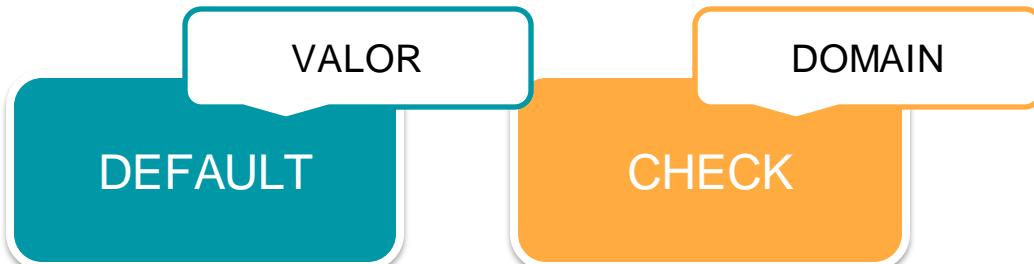
Constraint

- NOT NULL
- PK & SK



Constraints

- NOT NULL
- PK & SK



Constraints

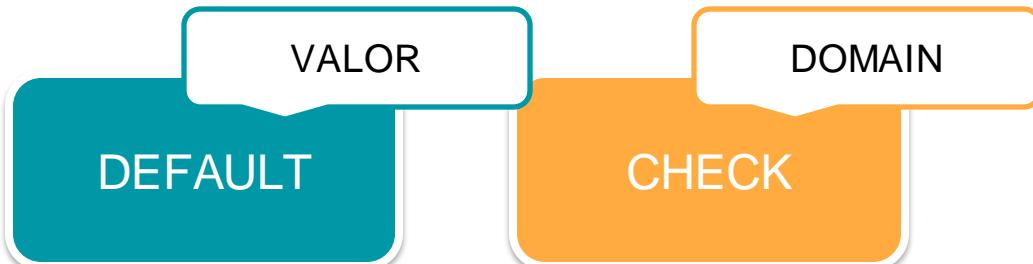
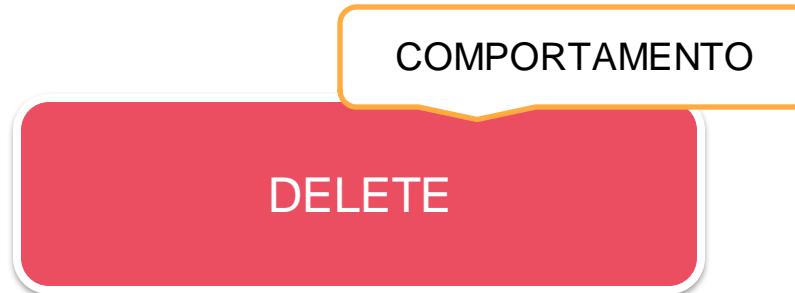
- NOT NULL
- PK & SK

```
CREATE DOMAIN D_NUM AS INTEGER  
CHECK (D_NUM > 0 AND D_NUM < 21);
```



Constraints

- NOT NULL
- PK & SK



Constraints

- Primary Key
- UNIQUE

CREATE TABLE

Dnumber INT **PRIMARY KEY**,

Dname VARCHAR(15) **UNIQUE**,



Violação de Constraints

- Primary Key
- UNIQUE

Referential Triggered Action

Dnumber INT **PRIMARY KEY**,

Dname VARCHAR(15) **UNIQUE**,



Violação de Constraints

- Nomear
- CHECK

Row-based

CHECK (Dept_create_date <= Mgr_start_date);

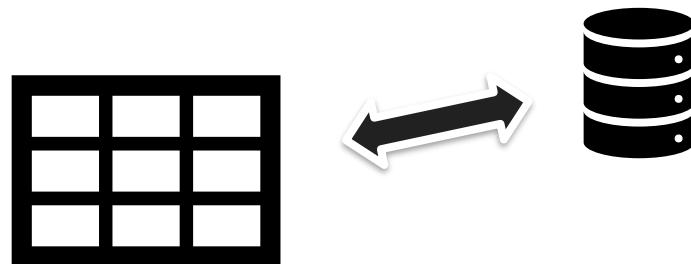


QUERIES / INSERTION COM SQL

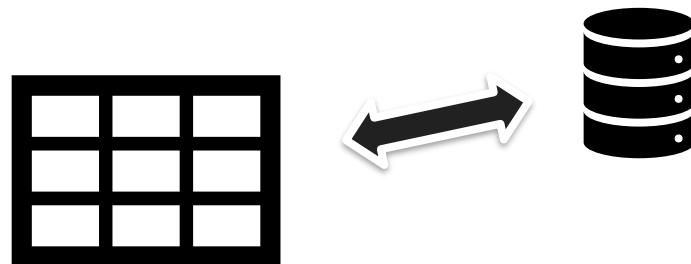
SQL Queries

Comportamento

- Multiset – not set
- Duplicações (Redundâncias)



SQL Queries



Comportamento

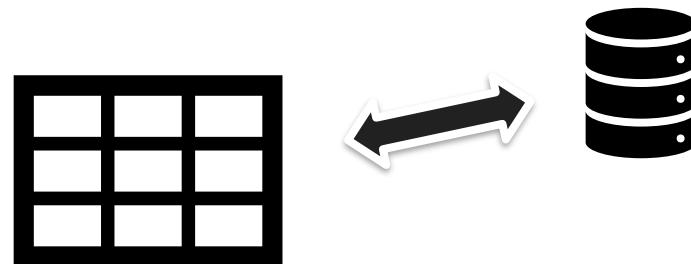
- Multiset – not set
- Duplicações (Redundâncias)

Custoso – melhor prevenir

Pode ser desejada pelo usuário



SQL Queries



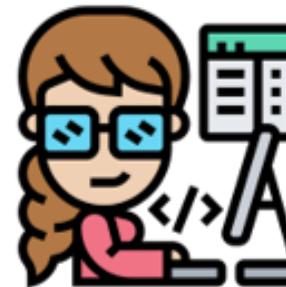
Comportamento

- Multiset – not set
- Duplicações (Redundâncias)

DISTINCT

```
SELECT      ALL Salary  
FROM        EMPLOYEE;
```

```
SELECT      DISTINCT Salary  
FROM        EMPLOYEE;
```

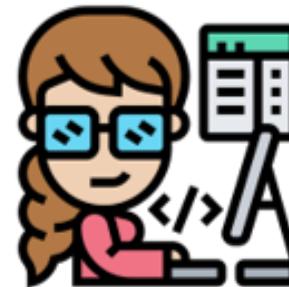


SQL Queries

Mapping

SELECT <lista de atributos> FROM <tabela> WHERE <condição>

```
SELECT      Bdate, Address  
FROM        EMPLOYEE  
WHERE       Fname = 'John' AND Minit = 'B' AND Lname = 'Smith';
```



SQL Queries

Mapping

Projeção de atributos

SELECT <lista de atributos> FROM <tabela> WHERE <condição>

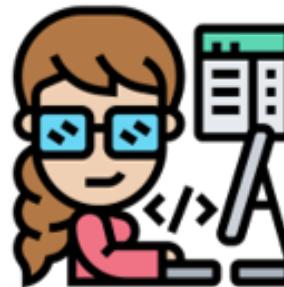
```
SELECT      Bdate, Address  
FROM        EMPLOYEE  
WHERE       Fname = 'John' AND Minit = 'B' AND Lname = 'Smith';
```



SQL Queries

Operadores

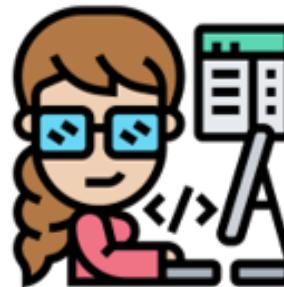
- =, <, <=, >, >=, e <> - SQL
- +, -, *, / - operadores básicos
- AND, OR, XOR, NOT – op. booleanos
- TRUE, FALSE – operadores lógicos



SQL Queries

DML

- Subconjunto do SQL
- Mais utilizado
- Comandos: INSERT, DELETE, UPDATE



SQL Queries

NULL & Not NULL

DML

INSERT INTO <table> (<list-attributes>) VALUES (<list-values>);



SQL Queries

NULL & Not NULL

DML

INSERT INTO <table> (<list-attributes>) VALUES (<list-values>);

Valores das PK?



SQL Queries

NULL & Not NULL

DML

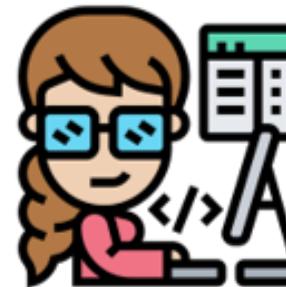
INSERT INTO <table> (<list-attributes>) VALUES (<list-values>);

Valores das PK?

SMALLINT UNSIGNED

0 to 65535

INT, BIGINT



SQL Queries

```
mysql> DESC person;
```

Field	Type	Null	Key	Default	Extra
person_id	smallint(5) unsigned		PRI	NULL	auto_increment
.					

DML

```
INSERT INTO <table> (<list-attributes>) VALUES (<list-values>);
```

Valores das PK?

SMALLINT UNSIGNED

0 to 65535

INT, BIGINT



EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

DEPT_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

Inserindo as Constraints

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	Dependent_name	Sex	Bdate	Relationship
-------------	----------------	-----	-------	--------------

Figure 5.5

Schema diagram for the COMPANY relational database schema.

Hands On! **Constraints com SQL**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Nomes, Aliasing e Variação de tuplas

Nomes de atributos

CREATE TABLE EMPLOYEE

(Fname	VARCHAR(15)	NOT NULL,
Minit	CHAR,	
Lname	VARCHAR(15)	NOT NULL,
Ssn	CHAR(9)	NOT NULL,
Bdate	DATE,	
Address	VARCHAR(30),	
Sex	CHAR,	
Salary	DECIMAL(10,2),	
Super_ssn	CHAR(9),	
Dno	INT	NOT NULL,

PRIMARY KEY (Ssn),

CREATE TABLE DEPARTMENT

(Dname	VARCHAR(15)	NOT NULL,
Dnumber	INT	NOT NULL,
Mgr_ssn	CHAR(9)	NOT NULL,
Mgr_start_date	DATE,	

PRIMARY KEY (Dnumber),

UNIQUE (Dname),

FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn));

Suponha...



Identificando a Tabela

```
SELECT      Fname, EMPLOYEE.Name, Address  
FROM        EMPLOYEE, DEPARTMENT  
WHERE       DEPARTMENT.Name = 'Research' AND  
              DEPARTMENT.Dnumber = EMPLOYEE.Dnumber;
```

Suponha...

```
SELECT      EMPLOYEE.Fname, EMPLOYEE.LName,  
              EMPLOYEE.Address  
FROM        EMPLOYEE, DEPARTMENT  
WHERE       DEPARTMENT.DName = 'Research' AND  
              DEPARTMENT.Dnumber = EMPLOYEE.Dno;
```



Nomes de atributos

Quando usar?

- Ambiguidade
- Esclarecimento

```
SELECT      EMPLOYEE.Fname, EMPLOYEE.LName,  
            EMPLOYEE.Address  
FROM        EMPLOYEE, DEPARTMENT  
WHERE       DEPARTMENT.DName = 'Research' AND  
            DEPARTMENT.Dnumber = EMPLOYEE.Dno;
```



Nomes de atributos

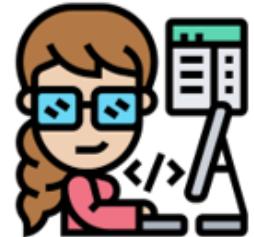
Quando usar?

- Diferenciar atributos

```
SELECT      E.Fname, E.Lname, S.Fname, S.Lname  
FROM        EMPLOYEE AS E, EMPLOYEE AS S  
WHERE       E.Super_ssn = S.Ssn;
```



Renomeando



Renomear atributos?

- Nomenclatura diferente

Nova nomenclatura

EMPLOYEE AS E(Fn, Mi, Ln, Ssn, Bd, Addr, Sex, Sal, Sssn, Dno)

Renomeando



Renomear atributos?

- Nomenclatura diferente

Referências múltiplas

```
SELECT      E.Fname, E.LName, E.Address  
FROM        EMPLOYEE AS E, DEPARTMENT AS D  
WHERE       D.DName = 'Research' AND D.Dnumber = E.Dno;
```

Hands On! **Alias (AS Statement) em Queries SQL**

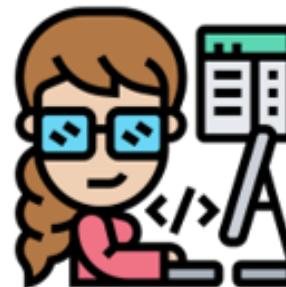
*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

When Good Statements Go Bad

Problemas

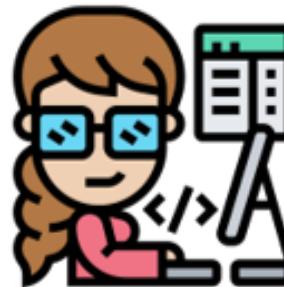
- PK & FK inexistente
- Valores violados
- Conversão inválida de datas



Problemas

- PK inexistente

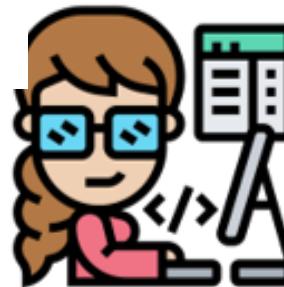
```
mysql> INSERT INTO person
->   (person_id, fname, lname, gender, birth_date)
-> VALUES (1, 'Charles','Fulton', 'M', '1968-01-15');
ERROR 1062 (23000): Duplicate entry '1' for key 'PRIMARY'
```



Problemas

- FK inexistente

```
mysql> INSERT INTO favorite_food (person_id, food)
-> VALUES (999, 'lasagna');
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint
fails ('bank'.favorite_food', CONSTRAINT 'fk_fav_food_person_id' FOREIGN KEY
('person_id') REFERENCES 'person' ('person_id'))
```



Problemas

- Violação de valores

```
mysql> UPDATE person  
-> SET gender = 'Z'  
-> WHERE person_id = 1;  
ERROR 1265 (01000): Data truncated for column 'gender' at row 1
```



Problemas

- Conversão inválida

```
mysql> UPDATE person
      -> SET birth_date = 'DEC-21-1980'
      -> WHERE person_id = 1;
ERROR 1292 (22007): Incorrect date value: 'DEC-21-1980' for column 'birth_date'
at row 1
```

```
mysql> UPDATE person
      -> SET birth_date = str_to_date('DEC-21-1980' , '%b-%d-%Y')
      -> WHERE person_id = 1;
Query OK, 1 row affected (0.12 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

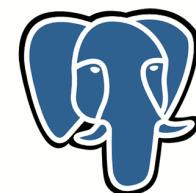


MySQL & PostgreSQL

Postgresql

- Otimizado para aplicações complexas
- Grande volume de dados \ informações críticas
- Ex:

E-commerce grande/médio porte



Postgre**SQL**

MySQL

- Possui agilidade e versatilidade
- Operações mais simples
- Processamento e tempo curto de resposta
- Ex:

Site | Fórum | Portal



Ponderações

- Precisa de Rollback? MySQL não possui
- Precisa de agilidade?
- Fácil utilização
- Operação mais simplificada

Enterprise

MySQL

Desafio 1 : Design do Esquema de BD Relacional para Aplicação Oficina

Design de Esquema Relacional - Oficina

- a. Declare suas relações usando o SQL DDL.
- b. Especifique um número de consultas em SQL que são necessárias para seu banco de dados Oficina.
- c. Com base no uso esperado do banco de dados, determine as *constraints* que fazem sentido para seu contexto
- d. Implemente seu banco de dados com MySQL
- e. Exporte o script SQL, adicione no Github para validação

Etapa 3

Explorando Queries com SQL

// Explorando a Linguagem de Consulta a Banco de Dados SQL

Explorando comandos DDL – Data Definition Language

Introdução ao SQL

Classificações

- DDL – Data Definition Language

CREATE | DROP | ALTER

INSERT | UPDATE | DELETE

RENAME | TRUNCATE | MERGE



Introdução ao SQL

UPDATE

DROP

DELETE

ALTER

Introdução ao SQL

UPDATE

- Modificação a partir de uma condição

DROP

DELETE

ALTER

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

Introdução ao SQL

UPDATE

DROP

- Remoção de estruturas

DELETE

ALTER

```
DROP DATABASE databasename;
```

Introdução ao SQL

UPDATE

DROP

DELETE

- Remoção de registros

ALTER

```
DELETE FROM table_name WHERE condition;
```

Introdução ao SQL

UPDATE

DROP

DELETE

ALTER

- Modificação da estrutura do BD

```
ALTER TABLE Customers  
ADD Email varchar(255);
```

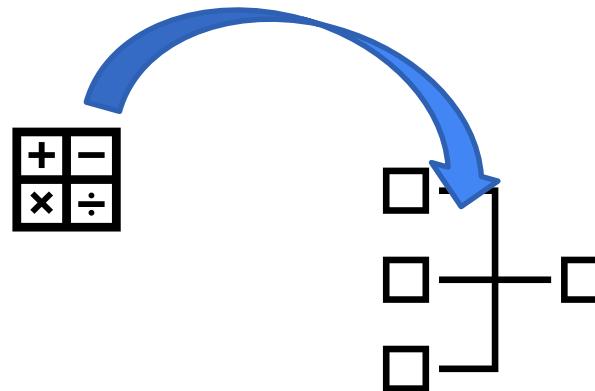
Hands On! **Constraints com SQL**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Expressões em Statements SQL

Expressões



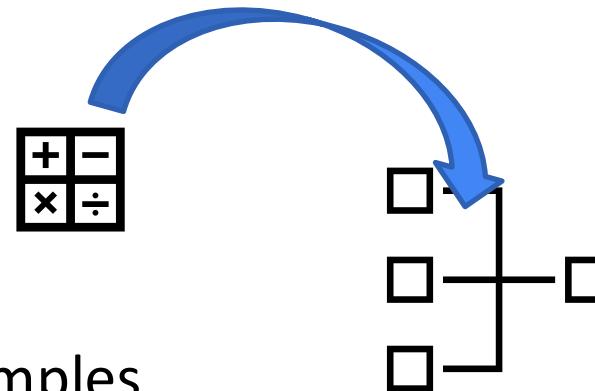
Atributos aritméticos

Operações matemáticas

Tipos de dados: string e numérico

```
SELECT Fname, Lname, Salary, Salary * 0.11 AS INSS FROM  
EMPLOYEE;
```

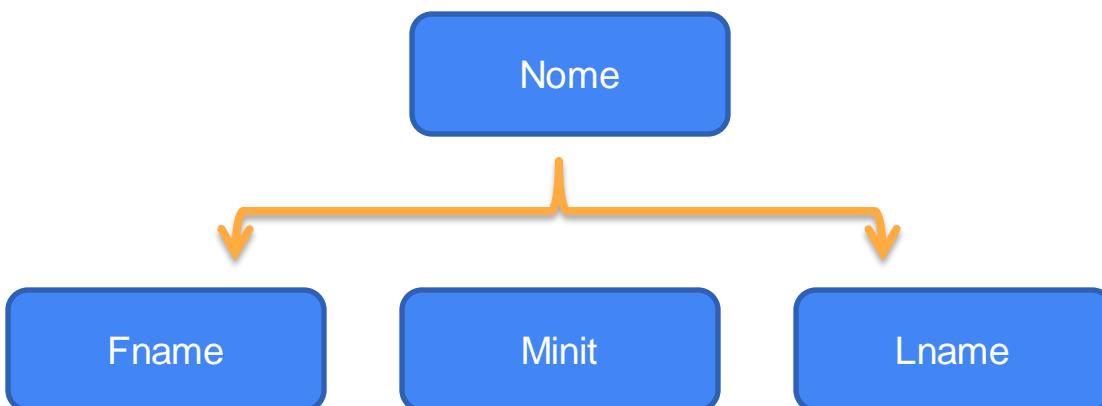
Expressões



- Armazenamento: dados simples
- Resultado de expressões: dados complexos
- Plus: utilizar alias

```
SELECT Fname, Lname, Salary, Salary * 0.11 AS INSS FROM  
EMPLOYEE;
```

Concatenando Strings



Expressões - Strings

Atributos distintos -> único sentido

Nome:

- Fname, Mint, Lname



Expressões - Strings

Atributos distintos -> único sentido

Nome:

- Fname, Minit, Lname

```
SELECT concat(Fname, ' ', Minit, ' ', Lname) AS  
Complete_name FROM EMPLOYEE;
```



Hands On! Utilizando Expressões em SQL

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Introdução a operação de conjuntos com SQL

Like | Between

- Comparação
- String Matching
- Caracteres especiais: % e _



```
SELECT      Fname, Lname  
FROM        EMPLOYEE  
WHERE       Address LIKE '%Houston,TX%';
```

Strings

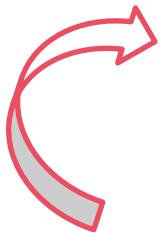
- 'AB_CD%EF'
- " -> ""



'AB_CD%\EF' ESCAPE '\'

Like | Between

- **Intervalo**
- Numérico
- Ex: salário, idade ...



```
SELECT *  
FROM EMPLOYEE  
WHERE (Salary BETWEEN 30000 AND 40000) AND Dno = 5;
```

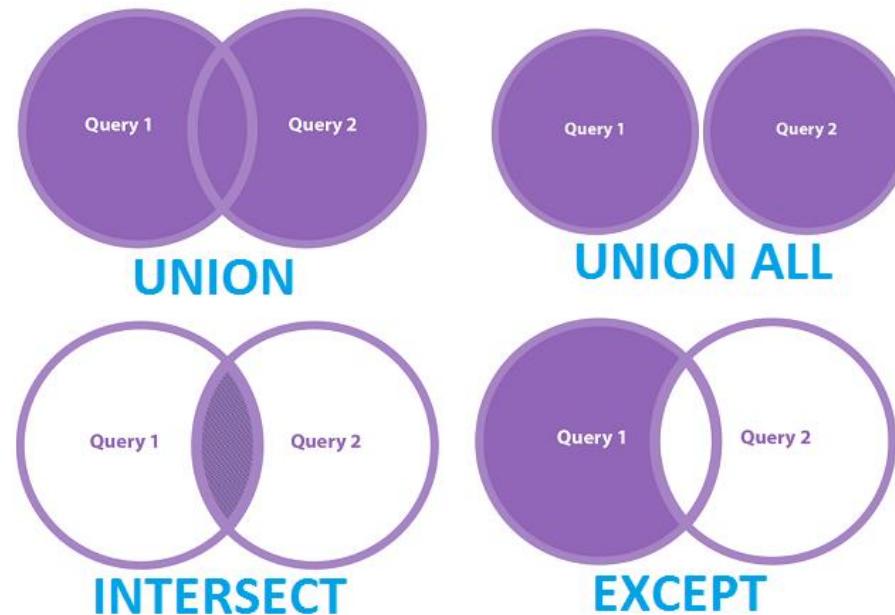
Hands On! Utilizando like e between com SQL

“Falar é fácil.
Mostre-me o código!”

Linus Torvalds

Comandos baseados em Operações Matemáticas: **UNION, INTERSECTION & EXCEPT**

Union | Intersection | Except

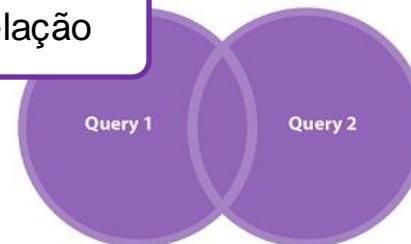


Fonte: c-sharpcorner

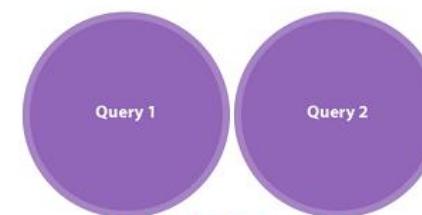
Union | Intersection | Except

- UNION
- UNION ALL
- INTERSECTION
- EXCEPT

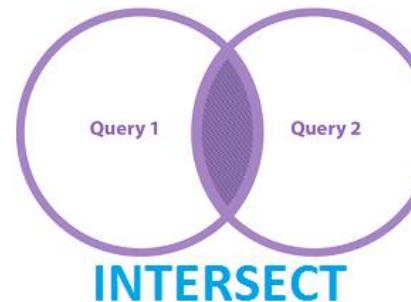
Mesmo Tipo de Relação



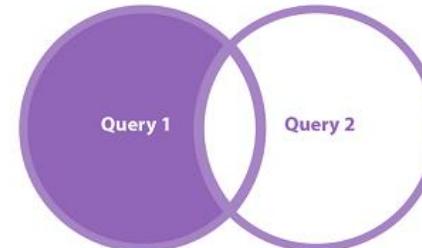
UNION



UNION ALL



INTERSECT



EXCEPT

Union | Intersection | Except

$R \rightarrow S$

A

a1
a2

Mesmo Tipo de Relação

- UNION
- UNION ALL
- INTERSECTION
- EXCEPT

R
A
a1
a2
a2
a3

S
A
a1
a2
a4
a5

(b) T
A
a1
a1
a2
a2
a2
a2
a3
a4
a5

$R \rightarrow S$

A

a3

$S \rightarrow R$

A

a4
a5

Union | Intersection | Except

Mesmo Tipo de Relação

- UNION
 (SELECT
 FROM
 WHERE
 DISTINCT Pnumber
 PROJECT, DEPARTMENT, EMPLOYEE
 Dnum = Dnumber **AND** Mgr_ssn = Ssn
 AND Lname = 'Smith')
- UNION ALL
- INTERSECTION
 UNION
 (SELECT
 FROM
 WHERE
 DISTINCT Pnumber
 PROJECT, WORKS_ON, EMPLOYEE
 Pnumber = Pno **AND** Essn = Ssn
 AND Lname = 'Smith');
- EXCEPT

Hands On! **Utilizando operadores com SQL**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Nested Queries com SQL

Nestes Queries

- Comparação por atributos buscados
- Nested Query – Consulta aninhada
- Consulta externa e interna



Nestes Queries

- Comparação por atributos buscados
- Nested Query – Consulta aninhada
- Consulta externa e interna



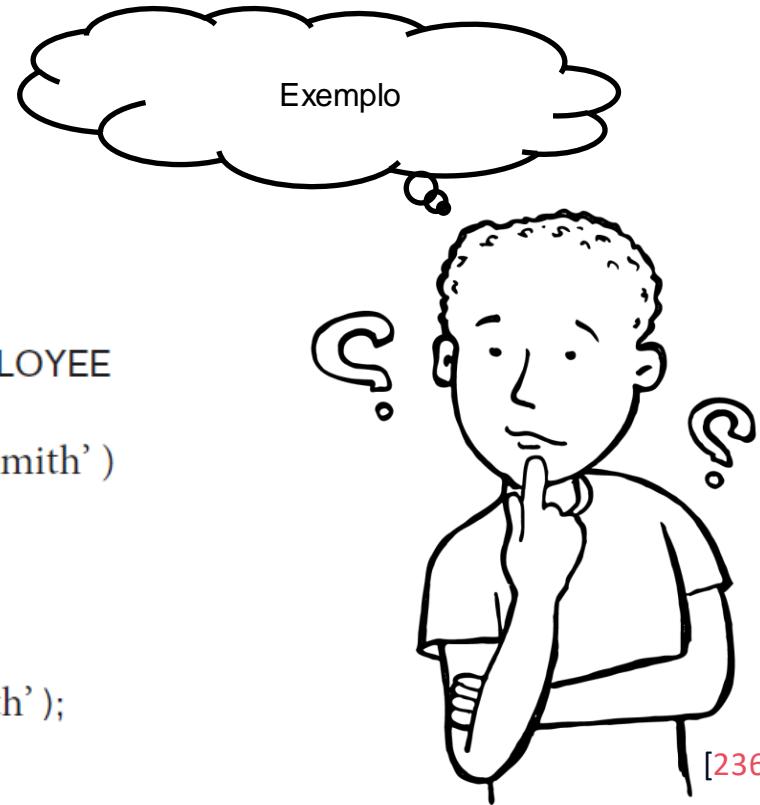
SELECT

FROM

WHERE

Nestes Queries

```
SELECT      DISTINCT Pnumber
FROM
WHERE      Pnumber IN
          ( SELECT      Pnumber
            FROM        PROJECT, DEPARTMENT, EMPLOYEE
            WHERE       Dnum = Dnumber AND
                        Mgr_ssn = Ssn AND Lname = 'Smith' )
          OR
          Pnumber IN
          ( SELECT      Pno
            FROM        WORKS_ON, EMPLOYEE
            WHERE       Essn = Ssn AND Lname = 'Smith' );
```



Nestes Queries

- IN – comparação com set
- = – comparação com unidades
- Retorno: tabela



SELECT

FROM

WHERE

Nestes Queries

Keywords:

- ANY, SOME, ALL

Operadores:

- >, >=, <, <=, and <>

SELECT

FROM

WHERE



Nestes Queries Correlacionadas

```
SELECT      E.Fname, E.Lname  
FROM        EMPLOYEE AS E, DEPENDENT AS D  
WHERE       E.Ssn = D.Essn AND E.Sex = D.Sex  
           AND E.Fname = D.Dependent_name;
```

SELECT

FROM

WHERE

Cláusulas EXISTS & UNIQUE

- **EXISTS:** TRUE se o resultado da consulta aninhada contiver pelo menos uma tupla
- **NOT EXISTS:** TRUE se o resultado da consulta aninhada não contiver tuplas
- **UNIQUE:** retorna TRUE se único



Cláusulas EXISTS & UNIQUE

Q16B:

```
SELECT      E.Fname, E.Lname
FROM        EMPLOYEE AS E
WHERE       EXISTS ( SELECT      *
                      FROM        DEPENDENT AS D
                      WHERE       E.Ssn = D.Essn AND E.Sex = D.Sex
                                  AND E.Fname = D.Dependent_name);
```

Conjuntos Explícitos

Query 17. Retrieve the Social Security numbers of all employees who work on project numbers 1, 2, or 3.

Q17: **SELECT** **DISTINCT** Essn
 FROM WORKS_ON
 WHERE Pno **IN** (1, 2, 3);

Etapa 4

Criando Queries com Funções/Cláusulas de Agrupamentos

// Explorando a Linguagem de Consulta a Banco de Dados SQL

Cláusulas de Ordenação com SQL

Ordenação em Queries SQL



Ordenação em Queries SQL



Ordenação em Queries SQL

ORDER BY

- Cláusula SQL
- Ordenação por coluna
- Expressões baseadas em dados
- + de um valor



Ordenação em Queries SQL

ORDER BY

- Cláusula SQL
- Ordenação por coluna
- Expressões baseadas em dados
- + de um valor

```
Select Fname, Lname, Dno from  
employee order by (Dno);
```



```
Select Fname, Lname, Dno from  
employee;
```



Ordenação em Queries SQL



Ascendente

- Sentido da ordenação

Descendente

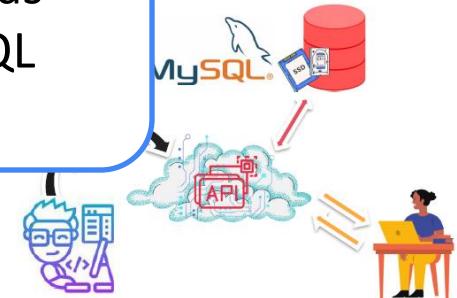


Ordenação em Queries SQL

```
SELECT <attribute list>  
FROM <table list>  
[ WHERE <condition> ]  
[ ORDER BY <attribute list> ];
```



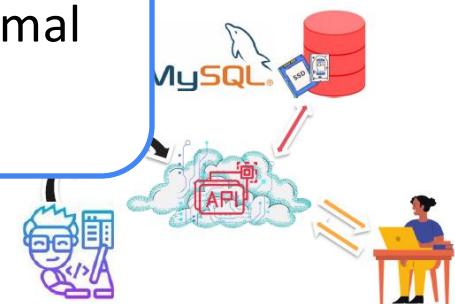
Estrutura das
Cláusulas SQL



WHERE STATEMENT?

Critério a ser considerado na query ao recuperar a informação. A tupla (linha ou instância) deve conter a condição definida em where

Definição formal



Ordenação em Queries SQL

Ordenação

```
SELECT <attribute list>  
FROM <table list>  
[ ORDER BY <attribute list>  
DESC];
```



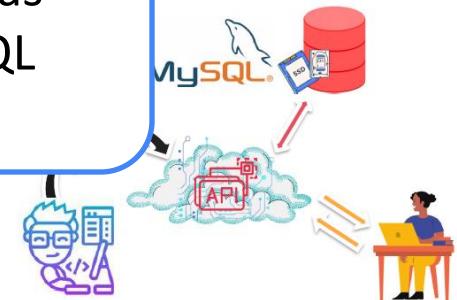
Ordenação em Queries SQL

Exemplo

```
SELECT idAccount  
FROM accounts  
ORDER BY amount LIMIT 5;
```

Expressão

Estrutura das
Cláusulas SQL



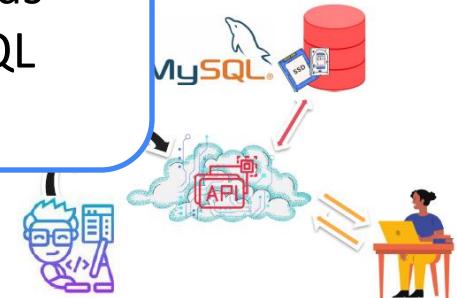
Ordenação em Queries SQL

Order + Expressões

```
SELECT <attribute list>  
FROM <table list>  
[ WHERE <condition> ]  
[ ORDER BY <attribute list> ];
```



Estrutura das
Cláusulas SQL



Ordenação em Queries SQL

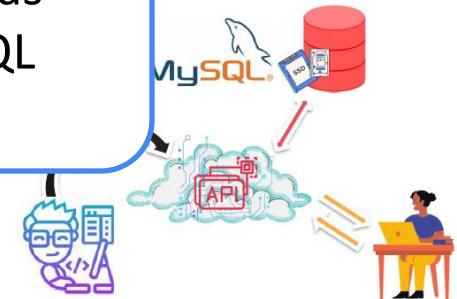
Exemplo

```
SELECT idAccount  
FROM accounts  
WHERE type = 'PJ'  
ORDER BY amount LIMIT 5;
```

Expressão



Estrutura das
Cláusulas SQL



Hands On! **Ordenando dados recuperados com SQL**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Ordenação com SQL

Utilizando Expressões

Classificação de dados

ORDER BY

- Cláusula SQL
- Ordenação por coluna
- Expressões baseadas em dados
- + de um valor



Classificação de dados

Ordenação dos dados para trazer consistência e sequência

Alias para fornecer nomenclatura adequada à dados derivados ou armazenados



Classificação de dados

Order + Expressões

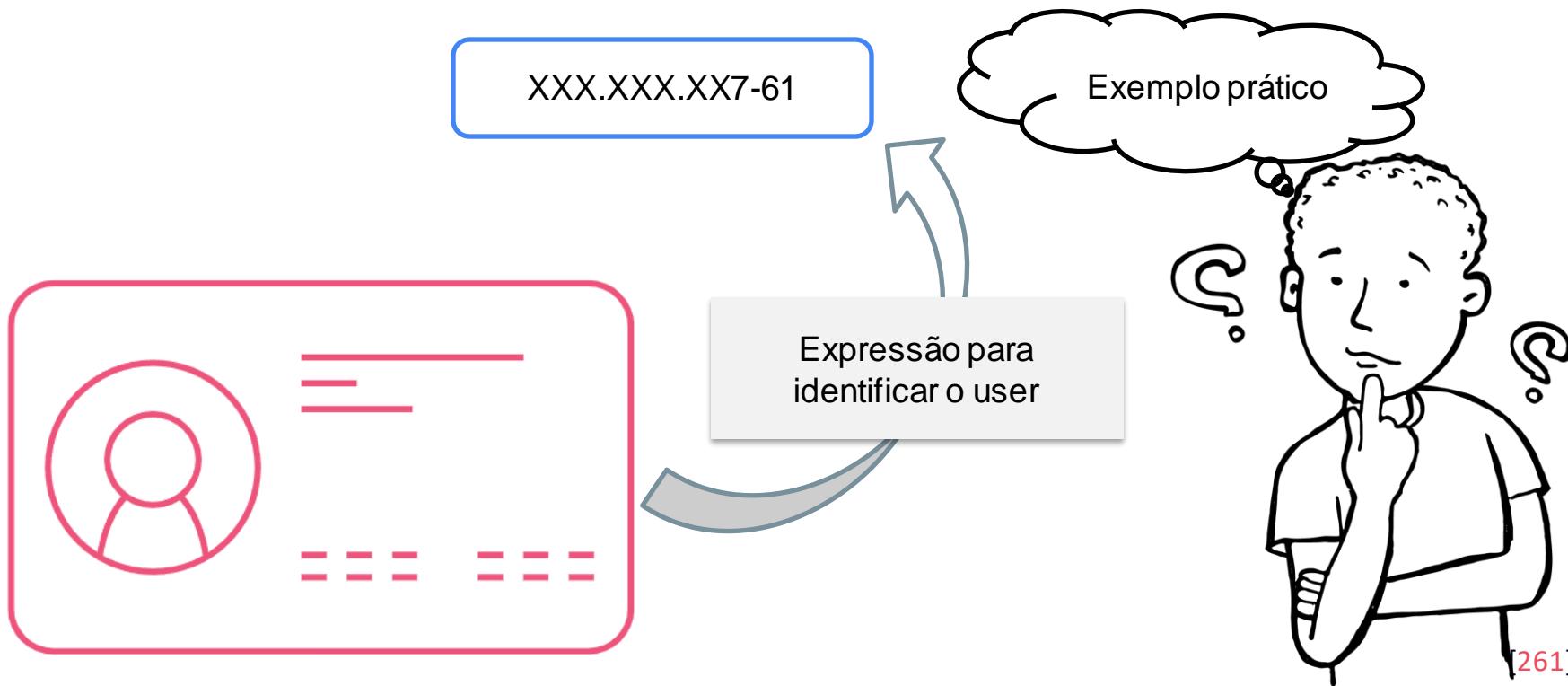
```
SELECT <attribute list>  
FROM <table list>  
[ WHERE <condition> ]  
[ ORDER BY <attribute list> ];
```

Tratar dados não
armazenados

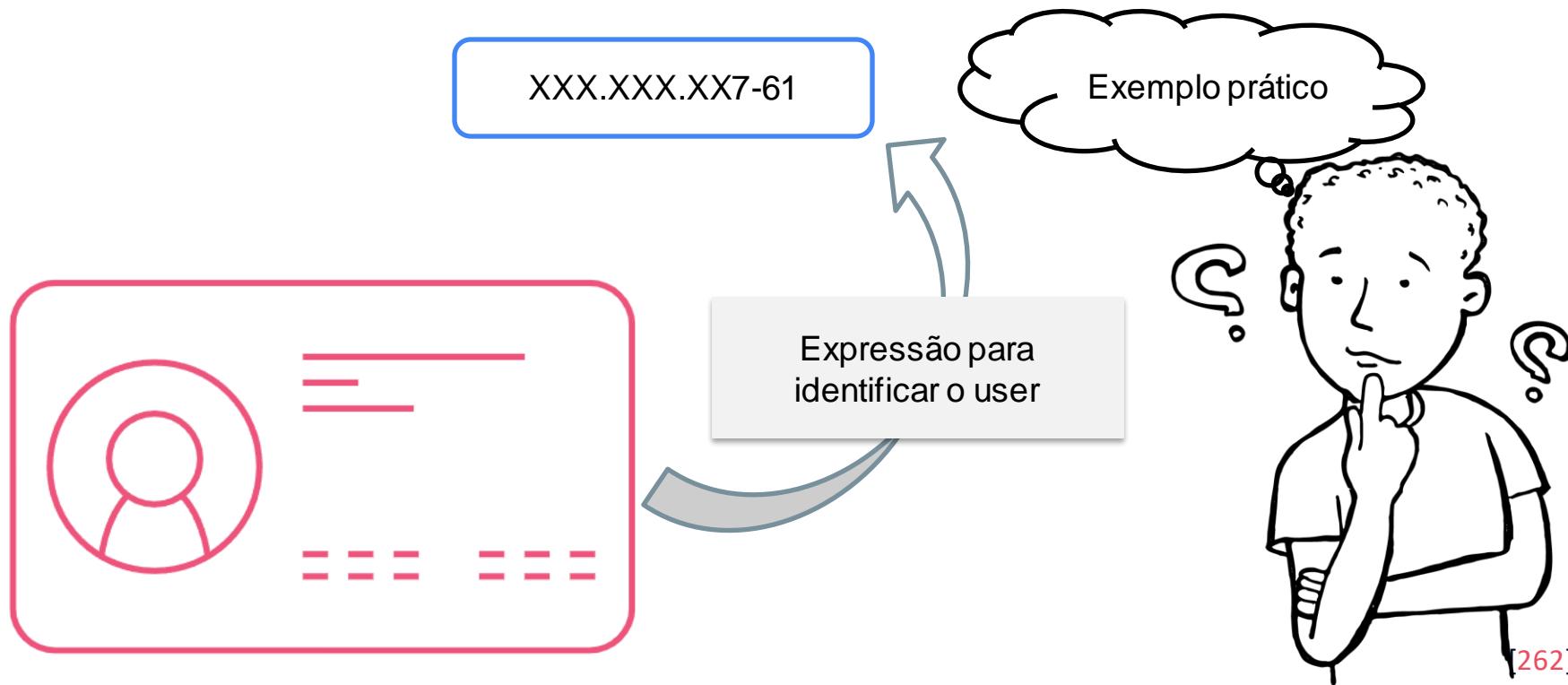
Adicionar expressão



Classificação de dados



Classificação de dados



Classificação de dados

- Operadores numéricos
- Nomes dos atributos
- + de um atributo
- DESC ou ASC

Como ordenar os registros?



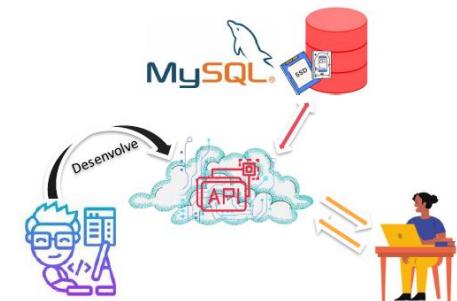
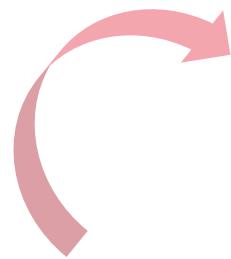
Classificação de dados

```
SELECT <attribute list>
FROM <table list>
[ WHERE <condition> ]
[ ORDER BY <atribute a, atrIBUTE b> ]
[DESC | ASC];
```



Classificação de dados

```
SELECT <attribute list>
FROM <table list>
[ WHERE <condition> ]
[ ORDER BY <EXPRESSION> ]
[DESC | ASC];
```



Hands On! **Ordenando dados recuperados com SQL**

**“Falar é fácil.
Mostre-me o código!”**

Linus Torvalds

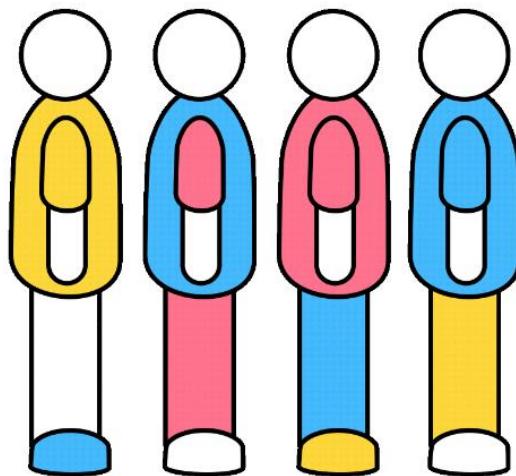
Funções de Agregação com SQL

Funções de Agrupamento

Agregar

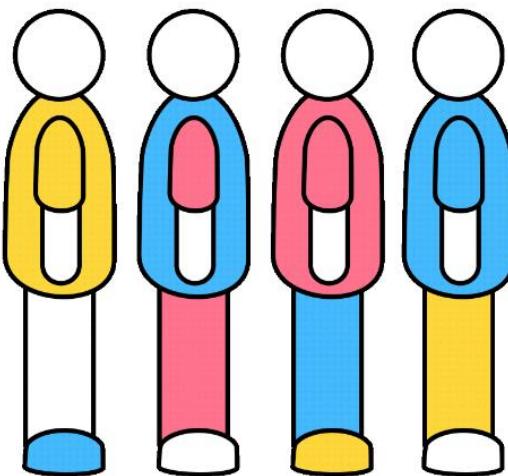
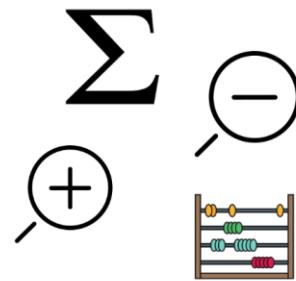
Aglutinar

Agrupar



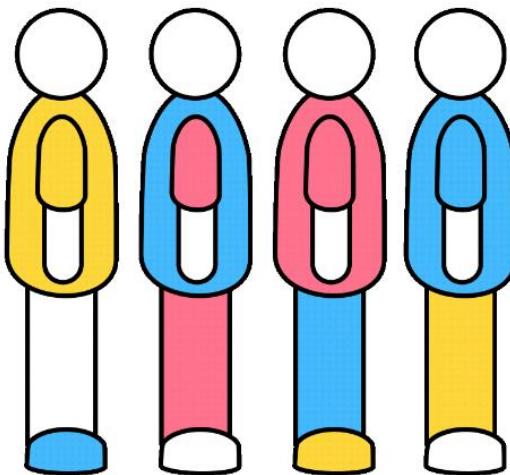
Agrupamento de Registros

- COUNT
- SUM
- MIN
- MAX
- AVERAGE



Agrupamento de Registros

- COUNT: registros
- SUM: somatório
- MIN: valor mínimo - atributo
- MAX: valor máximo - atributo
- AVERAGE: média de valor - atributo

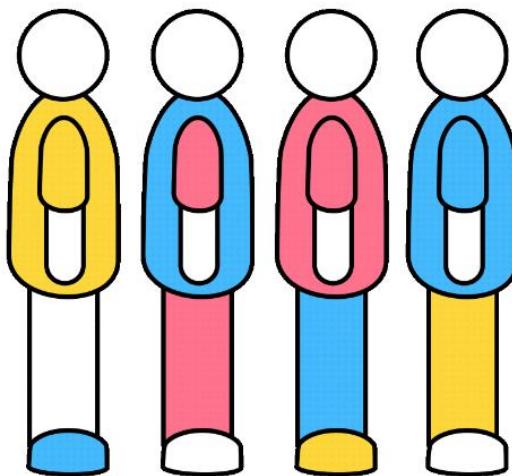


Agrupamento de Registros

- COUNT: registros
- SUM: somatório
- MIN: valor mínimo - atributo
- MAX: valor máximo - atributo
- AVERAGE: média de valor - atributo

SELECT

HAVING



Agrupamento de Registros

Station number | report code | year | month | day | temp | pressure | visibility | wind speed |
temperature | precipitation | fog | rain | hail | thunder | tornado

Grid view Form view

Total rows loaded: 28000

station number	report code	year	month	day	temp	dew point	station pressure	visibility	wind speed	temperature	precipitation	snow depth	fog	rain	hail	thunder	tornado
1	143080 34DDA7	2002	12	21	36	33.8	987.4	3.4	0.2	69	0	NULL	true	true	true	true	true
2	766440 39537B	1998	10	1	83.3	72.7	1014.6	5.9	6.7	62	0	NULL	false	false	false	false	false
3	176010 C3C6D5	2001	5	18	69.1	55.7	NULL	7.3	4.3	31	0	NULL	false	false	false	false	false
4	125500 145150	2007	10	14	39.7	33	NULL	6.9	2.5	79	0	NULL	false	false	false	false	false
5	470160 FF616A	1967	7	29	72.4	65.6	NULL	9.2	1.2	38	0.04	NULL	false	false	false	false	false
6	821930 1F8A7B	1953	6	18	81.3	72.8	1007.1	12.4	3.6	31	0	NULL	false	false	false	false	false
7	470070 D02B0B	1961	6	27	77	73.4	NULL	7.9	3	58	1.93	NULL	false	false	false	false	false
8	719200 C74611	1978	2	5	1.6	-4.4	962.9	14.9	13.3	84	0	9.8	false	false	false	false	false
9	477460 737090	1962	8	14	84.5	72.3	1009.6	24.1	5.1	40	0	NULL	false	false	false	false	false
10	598550 CSC6E6	2006	10	15	82	72.9	NULL	14.2	1.7	39	0	NULL	false	false	false	false	false
11	471100 6A6704	1990	9	19	62.5	50.5	NULL	6	4.1	62	0	NULL	false	false	false	false	false
12	29800 921894	1986	12	26	16.8	13.9	NULL	6.6	14.7	46	0.02	8.7	false	false	false	false	false
13	292090 515EFF	1953	5	8	37.8	34.2	NULL	6	42	NULL	NULL	NULL	false	false	false	false	false
14	484750 A38C90	1988	6	24	87.5	72.6	NULL	8.7	3.1	54	0	NULL	false	false	false	false	false
15	724500 777F09	1988	3	28	63.1	47.2	958.1	13.6	18.3	82	0.01	NULL	false	false	false	false	false
16	30110 858C82	2004	10	31	49.8	47.6	NULL	19.6	38	0	NULL	NULL	false	false	false	false	false
17	410610 D6A909	1991	6	27	96.8	42.1	930.9	6.2	3.3	54	0	NULL	false	false	false	false	false
18	724320 207979	1988	3	4	35.1	33.1	999.4	3.1	9.3	40	0.23	NULL	true	true	true	true	true
19	941830 229317	2007	4	19	76.3	66.5	994.9	NULL	4	80	0	NULL	false	false	false	false	false
20	743920 2ABE7D	1996	5	21	70	57.6	NULL	5.8	7.5	75	0	NULL	true	true	true	true	true
21	150630 4ADF04	1996	12	10	32	28.3	990.4	4.9	3.9	44	0	NULL	false	false	false	false	false
22	724460 998082	1986	10	2	65.4	64.2	976	5.5	6.5	80	0.61	NULL	true	true	true	true	true
23	992210 3481BD	2000	5	5	45.4	NULL	1011.9	NULL	14.6	45	0	NULL	false	false	false	false	false
24	932920 EB6580	2009	5	17	59.4	52.1	NULL	12.4	7.3	72	0	NULL	false	false	false	false	false
25	906440 612C7C	1977	2	22	81	72	NULL	20.5	5.6	86	0.01	NULL	false	false	false	false	false
26	722788 2C7749	1995	10	16	89.7	35.6	NULL	35	3.7	33	0	NULL	false	false	false	false	false
27	700450 37D0AC	1998	11	8	7.3	33.3	1013.7	6.3	16.3	46	0	0.0	NULL	NULL	NULL	NULL	NULL

Fonte: Getting Start with SQL

Grid view Form view

Total rows loaded: 28000

	station number	report code	year	month	day	temp	dew point	station pressure	visibility	wind speed	temperature	precipitation	snow depth	fog	rain	hail	thunder	tornado
1	143080 34DDA7		2002	12	21	36	33.0	987.4	3.4	0.2	69	0	NULL	true	true	true	true	true
2	766440 395378		1998	10	1	83.3	72.7	1014.6	5.9	6.7	62	0	NULL	false	false	false	false	false
3	176010 C3C6D5		2001	5	18	69.1	55.7	NULL	7.3	4.3	31	0	NULL	false	false	false	false	false
4	125600 145150		2007	10	14	39.7	33	NULL	6.9	2.5	79	0	NULL	false	false	false	false	false
5	470160 EF616A		1967	7	29	72.4	65.6	NULL	9.2	1.2	38	0.04	NULL	false	false	false	false	false
6	821930 1F8A7B		1953	6	18	81.3	72.8	1007.1	12.4	3.6	31	0	NULL	false	false	false	false	false
7	478070 D028DB		1981	6	27	77	73.4	NULL	7.9	3	58	1.93	NULL	false	false	false	false	false
8	719200 C74611		1978	2	5	1.6	-4.4	962.9	14.9	13.3	84	0	9.8	false	false	false	false	false
9	477460 737090		1962	8	14	84.5	72.3	1009.6	24.1	5.1	40	0	NULL	false	false	false	false	false
10	598550 C5C66E		2006	10	15	82	72.9	NULL	14.2	1.7	39	0	NULL	false	false	false	false	false
11	471100 6A6704		1990	9	19	62.5	50.5	NULL	6	4.1	62	0	NULL	false	false	false	false	false
12	29880 921894		1986	12	26	16.8	13.9	NULL	6.6	14.7	46	0.02	8.7	false	false	false	false	false
13	292090 515EFF		1953	5	8	37.8	34.2	NULL	NULL	6	42	NULL	NULL	false	false	false	false	false
14	484750 A38C90		1988	6	24	87.5	72.6	NULL	8.7	3.1	54	0	NULL	false	false	false	false	false
15	724500 777F09		1988	3	28	63.1	47.2	958.1	13.6	18.3	82	0.01	NULL	false	false	false	false	false
16	30110 85F8C2		2004	10	31	49.8	47.6	NULL	NULL	19.6	38	0	NULL	false	false	false	false	false
17	410610 D6A909		1991	6	27	96.8	42.1	990.9	6.2	3.3	54	0	NULL	false	false	false	false	false
18	724320 207979		1988	3	4	35.1	33.1	999.4	3.1	9.3	40	0.23	NULL	true	true	true	true	true
19	941830 229317		2007	4	19	76.3	66.5	994.9	NULL	4	80	0	NULL	false	false	false	false	false
20	743920 2ABE7D		1995	5	21	70	57.6	NULL	5.8	7.5	75	0	NULL	true	true	true	true	true
21	150630 4ADE04		1996	12	10	32	28.3	990.4	4.9	3.9	44	0	NULL	false	false	false	false	false
22	724460 B9B8B2		1986	10	2	65.4	64.2	976	5.5	6.5	80	0.61	NULL	true	true	true	true	true
23	992210 346180		2000	5	5	45.4	NULL	1011.9	NULL	14.6	45	0	NULL	false	false	false	false	false
24	932920 EB6580		2009	5	17	59.4	52.1	NULL	12.4	7.3	72	0	NULL	false	false	false	false	false
25	986440 612C7C		1977	2	22	81	72	NULL	20.5	5.6	86	0.01	NULL	false	false	false	false	false
26	722788 2C7749		1995	10	16	89.7	35.6	NULL	35	3.7	33	0	NULL	false	false	false	false	false
27	700460 22D24C		1998	11	4	7.2	42.2	1003.7	6.2	46.7	46	0	0.04	NULL	NULL	NULL	NULL	NULL

Agrupamento de Registros

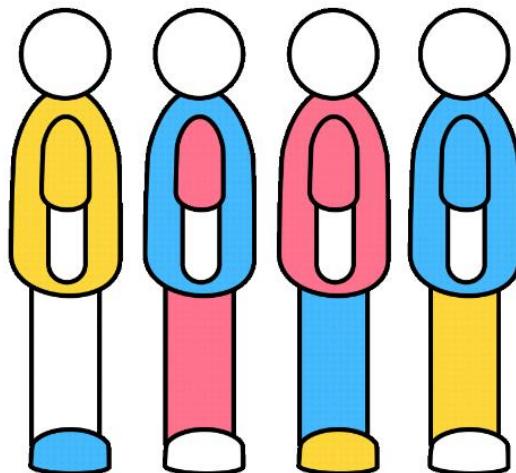
```
SELECT year, COUNT(*) AS record_count FROM station_data WHERE tornado = 1 GROUP BY year;
```

```
SELECT year, month, COUNT(*) AS record_count FROM station_data WHERE tornado = 1 GROUP BY year, month
```

```
SELECT COUNT(*) AS record_count FROM station_data;
```

Agrupamento de Registros

```
SELECT month, round(AVG(temp),2) as avg_temp  
FROM station_data  
WHERE year >= 2000  
GROUP BY month
```



Hands On! **Exemplos com BD Employee**

**“Falar é fácil.
Mostre-me o código!”**

Linus Torvalds

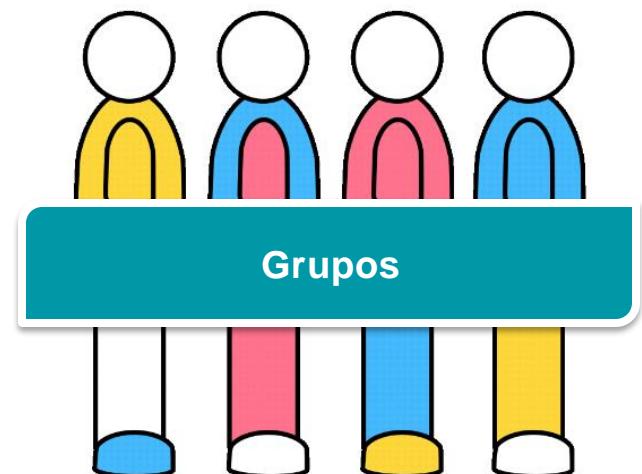
GROUP BY: Cláusula de Agrupamento com SQL

Funções de Agrupamento

Agregar

Aglutinar

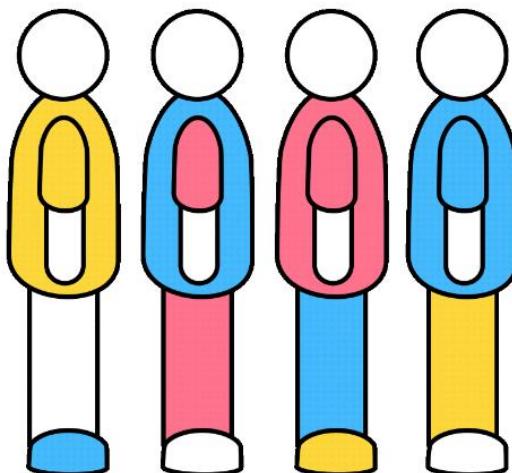
Agrupar



Agrupamento de Registros

- Atributos de Relacionamentos
- Grupos de valores

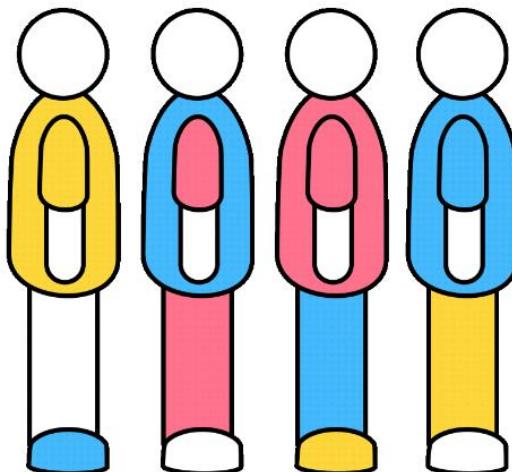
```
SELECT      Dno, COUNT (*), AVG (Salary)  
FROM        EMPLOYEE  
GROUP BY    Dno;
```



Agrupamento de Registros

- Atributos de Relacionamentos
- Grupos de valores

```
SELECT      Dno, COUNT (*), AVG (Salary)
FROM        EMPLOYEE
GROUP BY    Dno;
```



Agrupamento de Registros

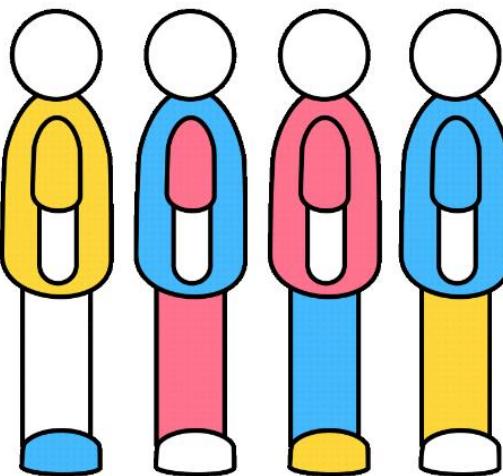
(a)

Fname	Minit	Lname	Ssn	...	Salary	Super_ssn	Dno
John	B	Smith	123456789	...	30000	333445555	5
Franklin	T	Wong	333445555		40000	888665555	5
Ramesh	K	Narayan	666884444		38000	333445555	5
Joyce	A	English	453453453		25000	333445555	5
Alicia	J	Zelaya	999887777		25000	987654321	4
Jennifer	S	Wallace	987654321		43000	888665555	4
Ahmad	V	Jabbar	987987987		25000	987654321	4
James	E	Bong	888665555		55000	NULL	1

Grouping EMPLOYEE tuples by the value of Dno

Dno	Count (*)	Avg (Salary)
5	4	33250
4	3	31000
1	1	55000

Result of Q24



Hands On! **Exemplos com BD Employee**

**“Falar é fácil.
Mostre-me o código!”**

Linus Torvalds

Entendendo o HAVING Statement

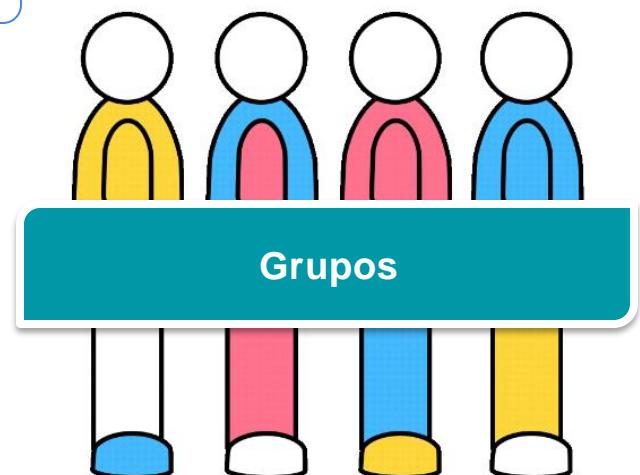
Funções de Agrupamento

Agregar

Aglutinar

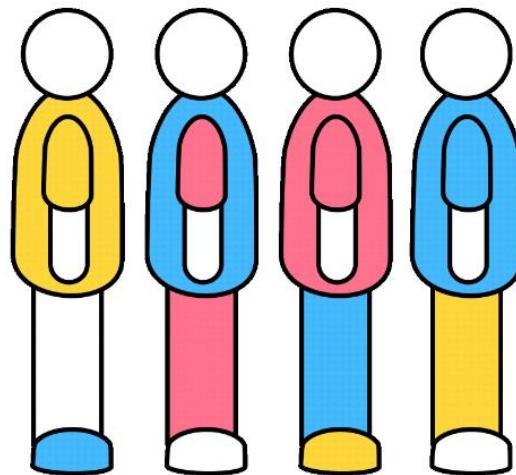
Agrupar

Condição?



HAVING Statement

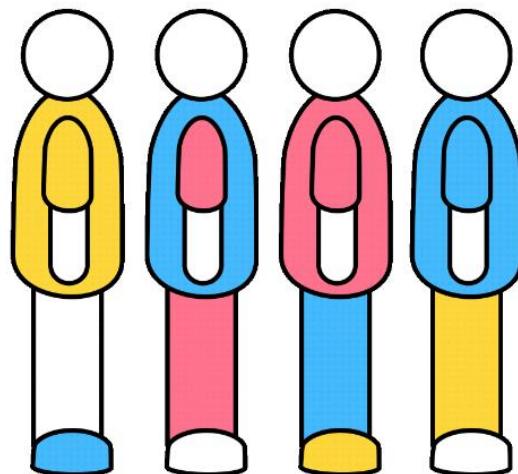
- **SELECT:** seleciona tabela e atributos
- **GROUP BY:** grupos de valores
- **HAVING:** Filtragem dos grupos



HAVING Statement

- **SELECT:** seleciona tabela e atributos
- **GROUP BY:** grupos de valores
- **HAVING:** Filtragem dos grupos

Condição sobre a informação sobre o grupo



HAVING Statement

```
SELECT Pnumber, Pname, COUNT (*)
FROM PROJECT, WORKS_ON
WHERE Pnumber = Pno
GROUP BY Pnumber, Pname
HAVING COUNT (*) > 2;
```

The diagram illustrates the execution flow of the HAVING clause. It starts with a large table of project and works-on data, which is then grouped by Pname and Pnumber. Arrows point from specific rows to a smaller summary table that includes Pname and Count (*). A note at the bottom right specifies "Result of Q26 (Pnumber not shown)".

Pname	Pnumber	...	Essn	Pno	Hours
ProductY	2		123456789	2	7.5
ProductY	2		453453453	2	20.0
ProductY	2		333445555	2	10.0
Computerization	10		333445555	10	10.0
Computerization	10		999887777	10	10.0
Computerization	10		987987987	10	35.0
Reorganization	20		333445555	20	10.0
Reorganization	20		987654321	20	15.0
Reorganization	20		888665555	20	NULL
Newbenefits	30		987987987	30	5.0
Newbenefits	30		987654321	30	20.0
Newbenefits	30		999887777	30	30.0

Pname	Count (*)
ProductY	3
Computerization	3
Reorganization	3
Newbenefits	3

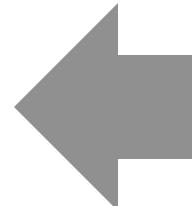
Result of Q26
(Pnumber not shown)

After applying the HAVING clause condition

HAVING Statement

```
SELECT <attribute and function list>
FROM <table list>
[ WHERE <condition> ]
[ GROUP BY <grouping attribute(s)> ]
[ HAVING <group condition> ]
[ ORDER BY <attribute list> ];
```

Estrutura da
Query SQL



HAVING Statement

```
SELECT <attribute and function list>
FROM <table list>
[ WHERE <condition> AND <condition2> ]
[ GROUP BY <grouping attribute(s)> ]
[ HAVING <group condition> ]
[ ORDER BY <attribute list> ];
```

Query interna

HAVING Statement



Preferência de execução das cláusulas

```
SELECT <attribute and function list>
FROM <table list>
[ WHERE <condition> AND <condition2> ]
[ GROUP BY <grouping attribute(s)> ]
[ HAVING <group condition> ]
[ ORDER BY <attribute list> ];
```

Query interna

Hands On! **Exemplos com BD Employee**

**“Falar é fácil.
Mostre-me o código!”**

Linus Torvalds

Ainda está em dúvida
sobre ORDER e GROUP?

AGRUPAMENTO

ORDER BY

GROUP BY

Nome
Maria
João
Cristina
Mario
....



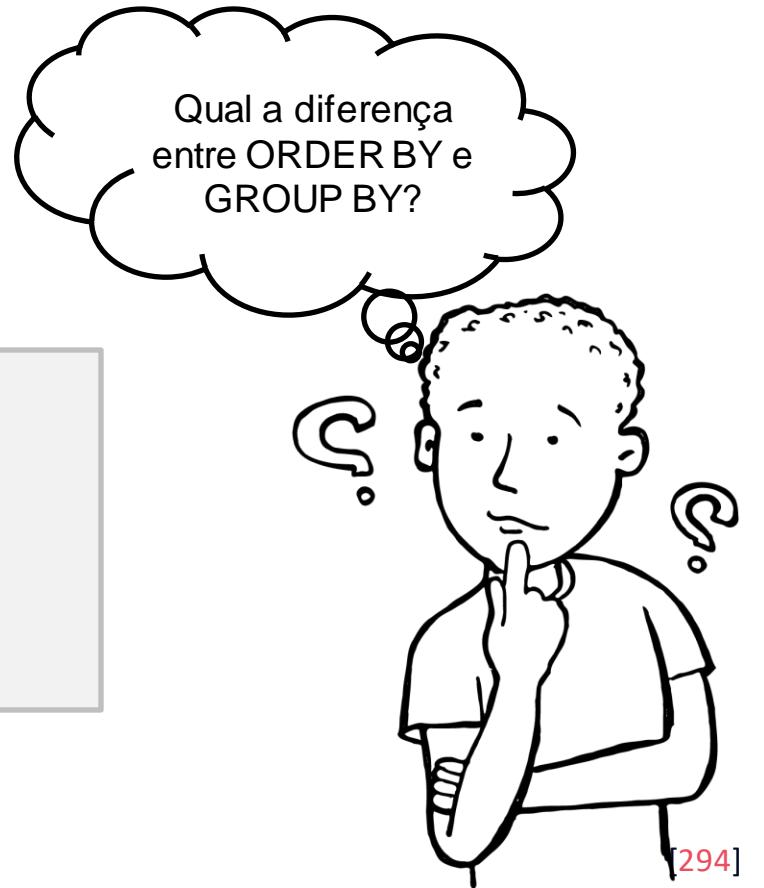
AGRUPAMENTO

ORDER BY

Nome
-----+-----
Maria
João
Cristina
Mario
....



Nome
-----+-----
Cristina
João
Maria
Mario
....



AGRUPAMENTO

ORDER BY

Nome
-----+
Maria
João
Cristina
Mario
....



SELECT nome
FROM tabela
ORDER BY nome

Qual a diferença
entre ORDER BY e
GROUP BY?



AGRUPAMENTO

ORDER BY

GROUP BY

CarModel
-----+
C350
C350
C350
C23
....



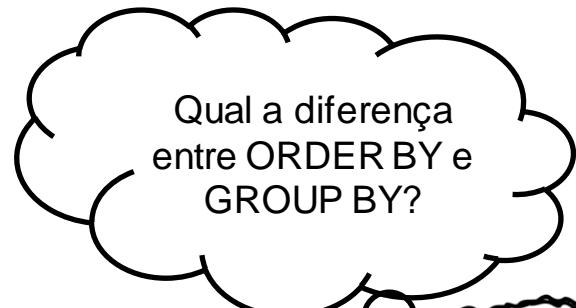
AGRUPAMENTO

GROUP BY

CarModel
C350
C350
C350
C23
...



CarModel
C23
C350
...



AGRUPAMENTO

GROUP BY

CarModel |
-----+
C350
C350
C350
C23
....



```
SELECT carro  
FROM tabela  
GROUP BY carro  
ORDER BY carro
```



Agrupamento em Queries SQL

- Cláusula SQL
- Ordenação por coluna
- Expressões baseadas em dados
- + de um valor

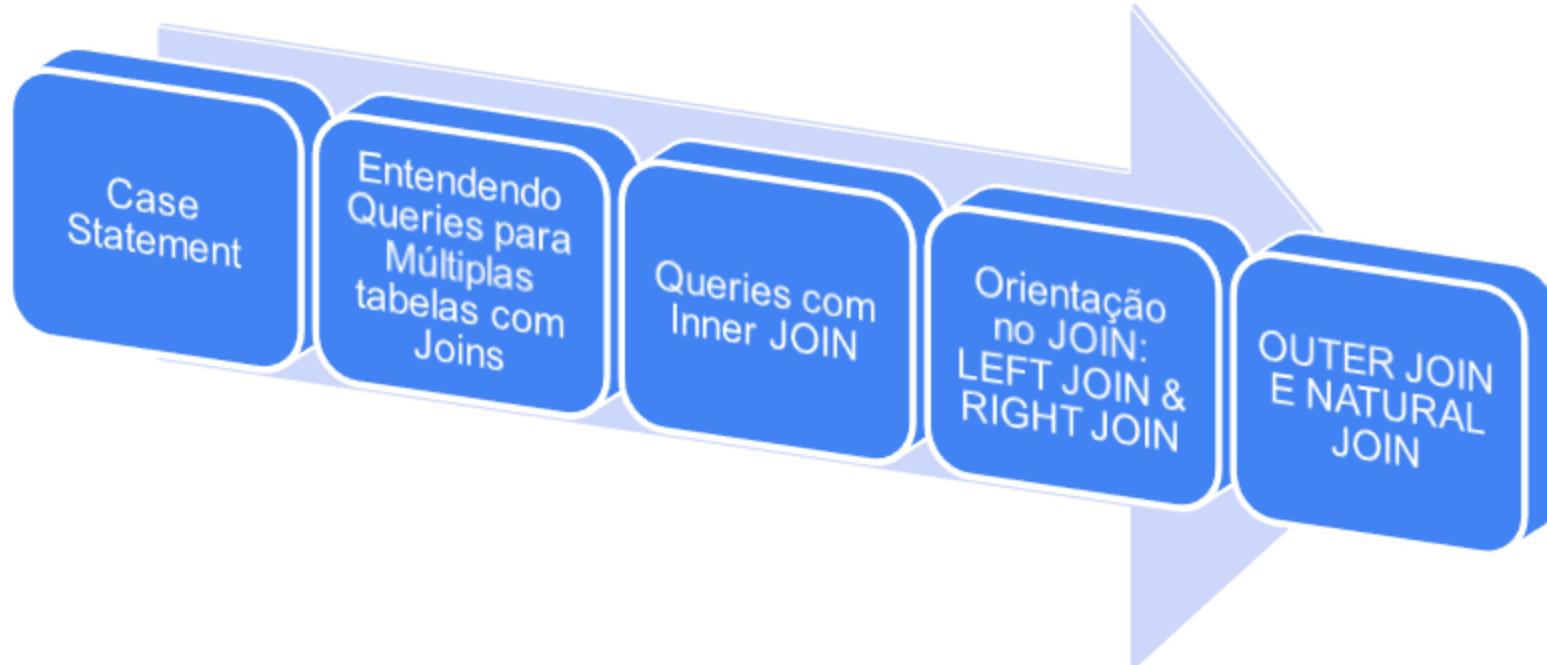


Etapa 5

Agrupando Registros e Tabelas com JOIN

// Explorando a Linguagem de Consulta a Banco de Dados SQL

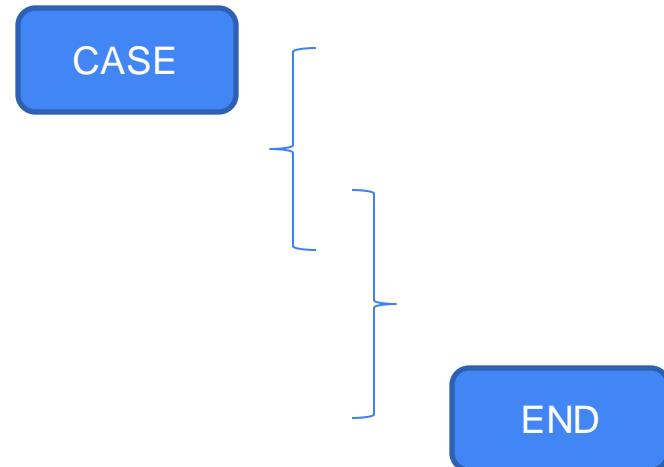
Explorando a SQL



CASE Statement

CASE Statement

- Troca de um valor
- Condição para troca
- Mapeamento de valores de correspondência



CASE Statement

- Troca de um valor
- Condição para troca
- Mapeamento de valores de correspondência



CASE Statement

```
UPDATE      EMPLOYEE  
SET          Salary =  
CASE        WHEN      Dno = 5      THEN Salary + 2000  
            WHEN      Dno = 4      THEN Salary + 1500  
            WHEN      Dno = 1      THEN Salary + 3000  
            ELSE          Salary + 0 ;
```



CASE Statement

Exemplo de mapeamento
do clima

Grid view Form view

Total rows loaded: 28000

	report_code	year	month	day	wind_speed	wind_severity
22	6757E5	1992	10	9	43.9	HIGH
23	E681EA	1984	2	10	44.9	HIGH
24	F7D723	2003	5	23	45.3	HIGH
25	B6A78C	1973	7	17	42.3	HIGH
26	1C8A2A	1998	11	21	50	HIGH
27	34DDA7	2002	12	21	0.2	LOW
28	39537B	1998	10	1	6.7	LOW
29	C3C6D5	2001	5	18	4.3	LOW
30	145150	2007	10	14	2.5	LOW
31	EF616A	1967	7	29	1.2	LOW

Fonte: livro Getting Started with SQL

CASE Statement

```
SELECT report_code, year, month, day, wind_speed,  
CASE  
WHEN wind_speed >= 40 THEN 'HIGH'  
WHEN wind_speed >= 30 AND wind_speed < 40 THEN 'MODERATE'  
ELSE 'LOW'  
END as wind_severity  
FROM station_data
```

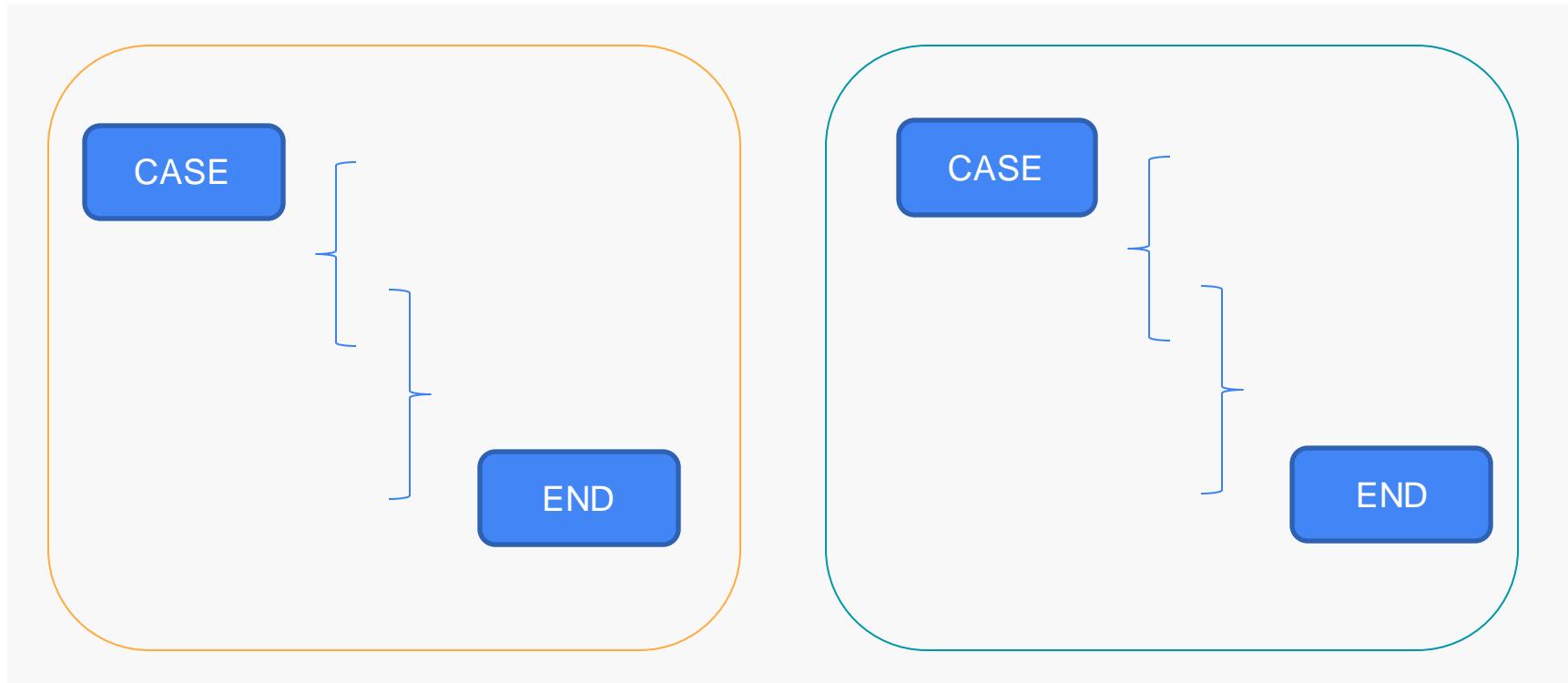
CASE Statement

SGBD entende a operação

```
SELECT report_code, year, month, day, wind_speed,  
CASE  
WHEN wind_speed >= 40 THEN 'HIGH'  
WHEN wind_speed >= 30 THEN 'MODERATE'  
ELSE 'LOW'  
END as wind_severity  
FROM station_data
```

Agrupando Queries com CASE Statement

Agrupando CASE Statement



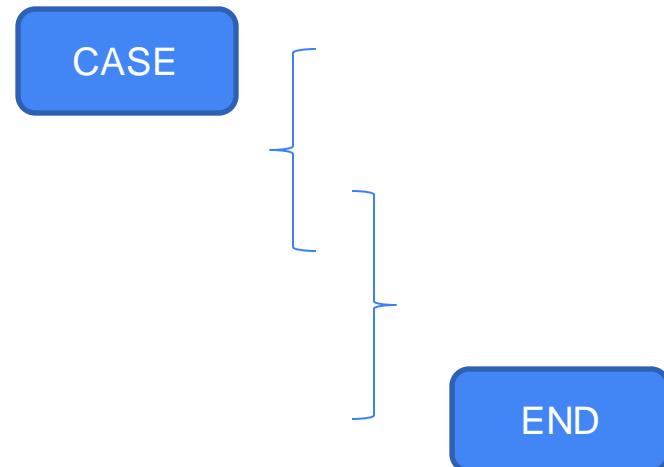
CASE Statement

```
SELECT year,  
CASE  
WHEN wind_speed >= 40 THEN 'HIGH'  
WHEN wind_speed >= 30 THEN 'MODERATE'  
ELSE 'LOW'  
END as wind_severity,  
COUNT(*) as record_count  
FROM station_data  
GROUP BY 1, 2
```

O caso zero/null trick

CASE Statement

- Filtros – valores distintos
- SELECT QUERY
- WHERE



CASE Statement

Tornado precipitation

```
SELECT year, month,  
SUM(precipitation) AS tornado_precipitation  
FROM station_data  
WHERE tornado = 1  
GROUP BY year, month
```

CASE Statement

Non-Tornado precipitation

```
SELECT year, month,  
SUM(precipitation) AS non_tornado_precipitation  
FROM station_data  
WHERE tornado = 0  
GROUP BY year, month
```

CASE Statement

```
SELECT year, month,  
SUM(CASE WHEN tornado = 1 THEN precipitation ELSE 0  
END) as tornado_precipitation,  
SUM(CASE WHEN tornado = 0 THEN precipitation ELSE 0  
END) as non_tornado_precipitation  
FROM station_data  
GROUP BY year, month
```

CASE Statement

Grid view Form view

Total rows loaded: 831

	year	month	tornado_precipitation	non_tornado_precipitation
811	2008	8 0		3.45
812	2008	9 0		1.77
813	2008	10 0		8.88
814	2008	11 0		2.14
815	2008	12 0.52		8.19
816	2009	1 0		4.02
817	2009	2 0		0.84
818	2009	3 0		14.41
819	2009	4 0		4.08
820	2009	5 0		8.4
821	2009	6 0.41		3.32

Fonte: livro Getting Start with SQL

Resultado

CASE Statement

AVG

MAX

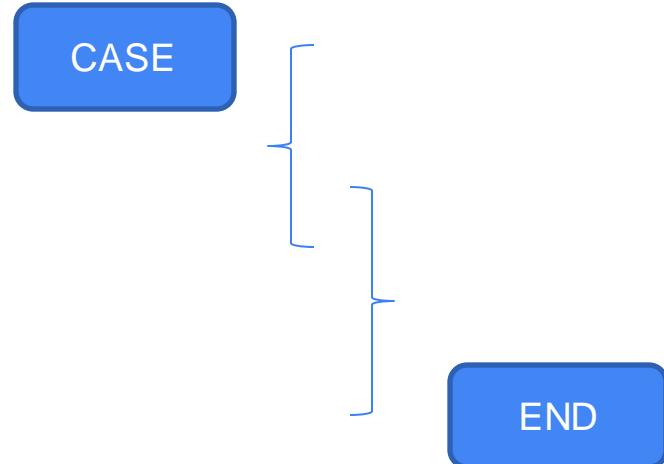
MIN

```
SELECT year, month,  
SUM(CASE WHEN tornado = 1 THEN precipitation ELSE 0  
END) as tornado_precipitation,  
SUM(CASE WHEN tornado = 0 THEN precipitation ELSE 0  
END) as non_tornado_precipitation  
FROM station_data  
GROUP BY year, month
```

CASE Statement

- MAX, AVG, MIN
- NAD, OR, NOT
- COUNT

Combinar operadores com CASE



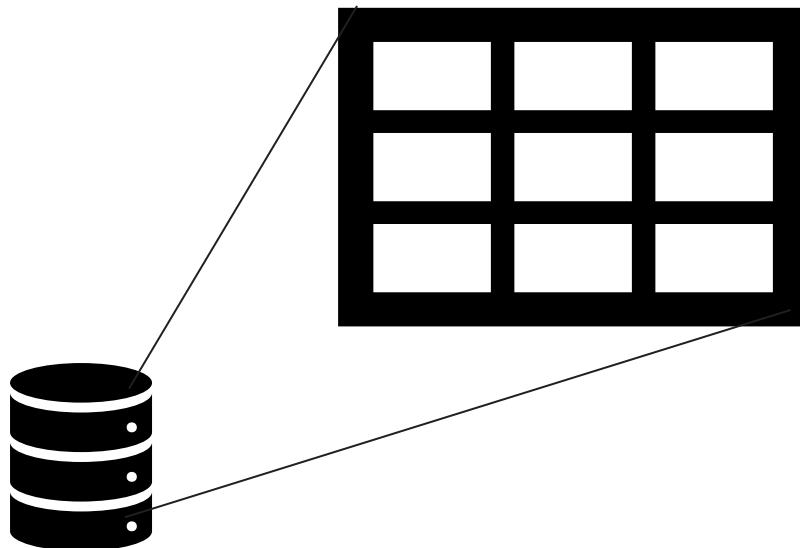
Hands On! **Projeto**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

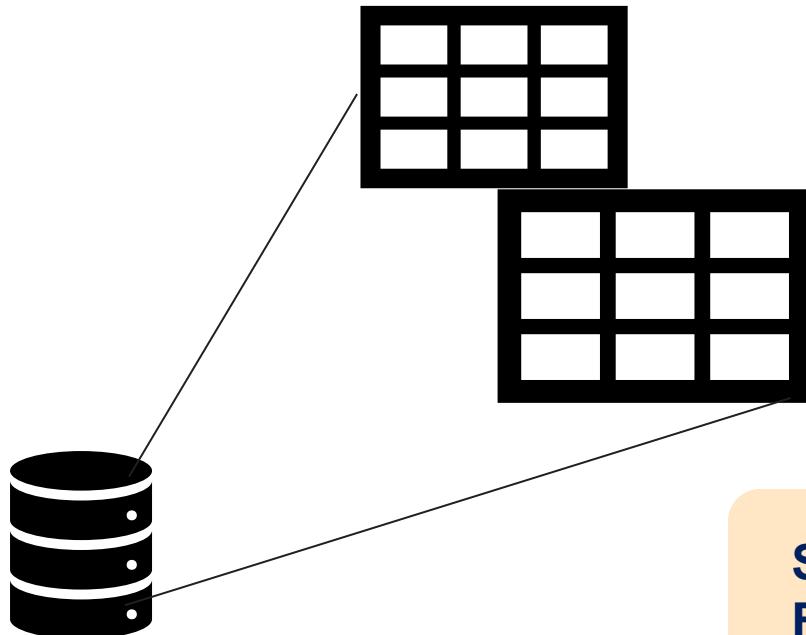
Entendendo Queries de Múltiplas Tabelas com JOINS

JOIN Statement



```
SELECT e.fname, e.lname  
FROM employee e
```

JOIN Statement



```
SELECT e.fname, e.lname, d.name  
FROM employee e JOIN department d
```

JOIN Statement

```
mysql> SELECT e.fname, e.lname, d.name  
      -> FROM employee e JOIN department d;
```

fname	lname	name
Michael	Smith	Operations
Michael	Smith	Loans
Michael	Smith	Administration
Susan	Barker	Operations
Susan	Barker	Loans
Susan	Barker	Administration
Robert	Tyler	Operations
Robert	Tyler	Loans
Robert	Tyler	Administration
Susan	Hawthorne	Operations
Susan	Hawthorne	Loans
Susan	Hawthorne	Administration

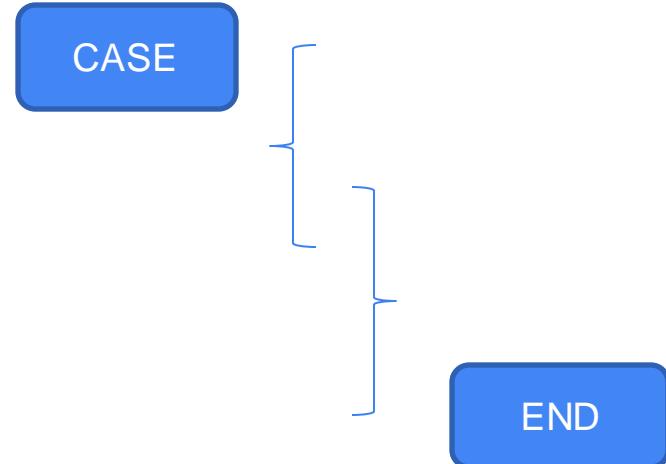


```
SELECT e.fname, e.lname, d.name  
FROM employee e JOIN department d
```

JOIN Statement

- JOIN: repetição de registros
- Produto Cartesiano

Combinar operadores com CASE



CROSS JOIN Statement

```
SELECT pt.name, p.product_cd, p.name  
FROM product p CROSS JOIN product_type pt;
```

name	product_cd	name
Customer Accounts	AUT	auto loan
Customer Accounts	BUS	business line of credit
Customer Accounts	CD	certificate of deposit
Customer Accounts	CHK	checking account
Customer Accounts	MM	money market account
Customer Accounts	MRT	home mortgage
Customer Accounts	SAV	savings account
Customer Accounts	SBL	small business loan
Insurance Offerings	AUT	auto loan
Insurance Offerings	BUS	business line of credit
Insurance Offerings	CD	certificate of deposit
Insurance Offerings	CHK	checking account
Insurance Offerings	MM	money market account
Insurance Offerings	MRT	home mortgage
Insurance Offerings	SAV	savings account
Insurance Offerings	SBL	small business loan
Individual and Business Loans	AUT	auto loan
Individual and Business Loans	BUS	business line of credit
Individual and Business Loans	CD	certificate of deposit
Individual and Business Loans	CHK	checking account
Individual and Business Loans	MM	money market account
Individual and Business Loans	MRT	home mortgage
Individual and Business Loans	SAV	savings account
Individual and Business Loans	SBL	small business loan

24 rows in set (0.00 sec)

Leitura para CROSS JOIN: Learning SQL

JOIN Statement

```
mysql> SELECT e.fname, e.lname, d.name  
      -> FROM employee e JOIN department d;
```

fname	lname	name
Michael	Smith	Operations
Michael	Smith	Loans
Michael	Smith	Admini
Susan	Barker	Operat
Susan	Barker	Loans
Susan	Barker	Admini
Robert	Tyler	Operations
Robert	Tyler	Loans
Robert	Tyler	Administration
Susan	Hawthorne	Operations
Susan	Hawthorne	Loans
Susan	Hawthorne	Administration
...		

O que faltou?



**SELECT e.fname, e.lname, d.name
FROM employee e JOIN department d**

JOIN Statement

```
SELECT e.fname, e.lname, d.name  
FROM employee e JOIN department d  
ON e.dept_id = d.dept_id ;
```



JOIN Statement

SELECT <atributes list>

FROM <table1> ft **JOIN** <table2> st

CROSS

O que acontece
OUTER JOIN?



JOIN Statement

SELECT <atributes list>

FROM <table1> ft **INNER JOIN** <table2> st

ON ft.common_attribute =

st.common_attribute



Queries com INNER JOIN

JOIN Statement

```
SELECT e.fname, e.lname, d.name  
FROM employee e JOIN department d  
ON e.dept_id = d.dept_id ;
```



JOIN Statement

```
SELECT e.fname, e.lname, d.name  
FROM employee e JOIN department d  
ON e.dept_id = d.dept_id ;
```

Falha: linhas excluídas



INNER JOIN

- Mais comum dos JOINs
- Linhas não correspondentes excluídas
- Atributo de junção

Mesclar Tabelas

1 + atributos junção

2 + Tabelas

INNER JOIN

- Mais comum dos JOINs
- Linhas não correspondentes excluídas
- Atributo de junção

Mesclar Tabelas

1 + atributos junção

2 + Tabelas

ON tab_a.atributo = tab_b.atributo

USING (atributo c/ mesmo nome)

INNER JOIN

Exemplo:

CUSTOMER							
CUSTOMER_ID	NAME	REGION	STREET ADDRESS	CITY	STATE	ZIP	
1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014	
2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201	
3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032	
4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104	
5	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782	

CUSTOMER_ORDER							
ORDER_ID	ORDER_DATE	SHIP_DATE	CUSTOMER_ID	PRODUCT_ID	ORDER_QTY	SHIPPED	
1	2015-04-20	2015-04-23	3	5	300	false	
2	2015-04-18	2015-04-22	5	4	375	false	
3	2015-04-15	2015-04-18	1	1	450	false	
4	2015-04-17	2015-04-20	3	2	500	false	
5	2015-04-18	2015-04-21	3	2	600	false	

Figure 8-4. A one-to-many relationship between CUSTOMER and CUSTOMER_ORDER

Fonte: livro Getting Started with SQL

INNER JOIN

```
SELECT ORDER_ID, CUSTOMER.CUSTOMER_ID, ORDER_DATE, SHIP_DATE, NAME,  
STREET_ADDRESS, CITY, STATE, ZIP, PRODUCT_ID, ORDER_QTY  
FROM CUSTOMER INNER JOIN CUSTOMER_ORDER  
ON CUSTOMER.CUSTOMER_ID = CUSTOMER_ORDER.CUSTOMER_ID
```

Atributos de junção

Hands On! **Projeto**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Agrupamento com mais de 3 Tabelas com JOIN

3 ou + tabelas com JOIN



```
mysql> SELECT a.account_id, c.fed_id  
-> FROM account a INNER JOIN customer c  
->   ON a.cust_id = c.cust_id  
-> WHERE c.cust_type_cd = 'B';  
+-----+-----+  
| account_id | fed_id |  
+-----+-----+  
| 24 | 04-1111111 |  
| 25 | 04-1111111 |  
| 27 | 04-2222222 |  
| 28 | 04-3333333 |  
| 29 | 04-4444444 |  
+-----+-----+  
5 rows in set (0.15 sec)
```

3 ou + tabelas com JOIN

JOIN com três
tabelas

```
mysql> SELECT a.account_id, c.fed_id, e.fname, e.lname
-> FROM account a INNER JOIN customer c
->   ON a.cust_id = c.cust_id
->   INNER JOIN employee e
->   ON a.open_emp_id = e.emp_id
-> WHERE c.cust_type_cd = 'B';
+-----+-----+-----+-----+
| account_id | fed_id | fname | lname |
+-----+-----+-----+-----+
|      24 | 04-1111111 | Theresa | Markham |
|      25 | 04-1111111 | Theresa | Markham |
|      27 | 04-2222222 | Paula  | Roberts |
|      28 | 04-3333333 | Theresa | Markham |
|      29 | 04-4444444 | John   | Blake  |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

3 ou + tabelas com JOIN

- 3 tabelas
- 2 tipos de JOINs
- 2 subcláusulas

```
mysql> SELECT a.account_id, c.fed_id, e.fname, e.lname
->   FROM account a INNER JOIN customer c
->     ON a.cust_id = c.cust_id
->   INNER JOIN employee e
->     ON a.open_emp_id = e.emp_id
-> WHERE c.cust_type_cd = 'B';
+-----+-----+-----+-----+
| account_id | fed_id | fname | lname |
+-----+-----+-----+-----+
|      24 | 04-1111111 | Theresa | Markham |
|      25 | 04-1111111 | Theresa | Markham |
|      27 | 04-2222222 | Paula  | Roberts |
|      28 | 04-3333333 | Theresa | Markham |
|      29 | 04-4444444 | John   | Blake  |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

3 ou + tabelas com JOIN

```
mysql> SELECT a.account_id, c.fed_id, e.fname, e.lname  
-> FROM customer c INNER JOIN account a  
-> ON a.cust_id = c.cust_id  
-> INNER JOIN employee e  
-> ON a.open_emp_id = e.emp_id  
-> WHERE c.cust_type_cd = 'B';
```

account_id	fed_id	fname	lname
24	04-1111111	Theresa	Markham
25	04-1111111	Theresa	Markham
27	04-2222222	Paula	Roberts
28	04-3333333	Theresa	Markham
29	04-4444444	John	Blake

5 rows in set (0.09 sec)

```
mysql> SELECT a.account_id, c.fed_id, e.fname, e.lname  
-> FROM account a INNER JOIN customer c  
-> ON a.cust_id = c.cust_id  
-> INNER JOIN employee e  
-> ON a.open_emp_id = e.emp_id  
-> WHERE c.cust_type_cd = 'B';
```

account_id	fed_id	fname	lname
24	04-1111111	Theresa	Markham
25	04-1111111	Theresa	Markham
27	04-2222222	Paula	Roberts
28	04-3333333	Theresa	Markham
29	04-4444444	John	Blake

5 rows in set (0.00 sec)

3 ou + tabelas com JOIN

```
mysql> SELECT a.account_id, c.fed_id, e.fname, e.lname  
-> FROM employee e INNER JOIN account a  
-> ON e.emp_id = a.open_emp_id  
-> INNER JOIN customer c  
-> ON a.cust_id = c.cust_id  
-> WHERE c.cust_type_cd = 'B';
```

account_id	fed_id	fname	lname
24	04-1111111	Theresa	Markham
25	04-1111111	Theresa	Markham
27	04-2222222	Paula	Roberts
28	04-3333333	Theresa	Markham
29	04-4444444	John	Blake

5 rows in set (0.00 sec)

```
mysql> SELECT a.account_id, c.fed_id, e.fname, e.lname  
-> FROM account a INNER JOIN customer c  
-> ON a.cust_id = c.cust_id  
-> INNER JOIN employee e  
-> ON a.open_emp_id = e.emp_id  
-> WHERE c.cust_type_cd = 'B';
```

account_id	fed_id	fname	lname
24	04-1111111	Theresa	Markham
25	04-1111111	Theresa	Markham
27	04-2222222	Paula	Roberts
28	04-3333333	Theresa	Markham
29	04-4444444	John	Blake

5 rows in set (0.00 sec)

Hands On! **Projeto**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

A Ordem Importa em Queries com JOIN?

Ordem de Queries JOIN

- SGBD interpreta e escolhe a sequência de comandos
- SGBD escolhe um ponto de partida

```
mysql> SELECT a.account_id, c.fed_id, e.fname, e.lname  
-> FROM account a INNER JOIN customer c  
->   ON a.cust_id = c.cust_id  
->   INNER JOIN employee e  
->   ON a.open_emp_id = e.emp_id  
-> WHERE c.cust_type_cd = 'B';
```

account_id	fed_id	fname	lname
24	04-1111111	Theresa	Markham
25	04-1111111	Theresa	Markham
27	04-2222222	Paula	Roberts
28	04-3333333	Theresa	Markham
29	04-4444444	John	Blake

5 rows in set (0.00 sec)

Ordem de Queries JOIN

STRAIGHT_JOIN



FORCE ORDER



Tem como especificar a ordem para o SGBD?

INNER JOIN

```
SELECT STRAIGHT_JOIN a.account_id, c.fed_id, e.fname, e.lname  
FROM customer c INNER JOIN account a  
ON a.cust_id = c.cust_id  
INNER JOIN employee e  
ON a.open_emp_id = e.emp_id  
WHERE c.cust_type_cd = 'B';
```

Forçando ordem de junção

Outros tópicos com JOIN Statement

Nested (Subqueries) com JOIN Statement

Subqueries com JOIN

```
SELECT a.account_id, a.cust_id, a.open_date, a.product_cd  
FROM account a INNER JOIN  
    (SELECT emp_id, assigned_branch_id  
     FROM employee  
     WHERE start_date < '2007-01-01' AND (title = 'Teller' OR title =  
     'Head Teller')) e  
ON a.open_emp_id = e.emp_id  
INNER JOIN  
    (SELECT branch_id  
     FROM branch  
     WHERE name = 'Woburn Branch') b  
ON e.assigned_branch_id = b.branch_id;
```

Subqueries COM JOIN

```
mysql> SELECT emp_id, assigned_branch_id  
-> FROM employee  
-> WHERE start_date < '2007-01-01'  
-> AND (title = 'Teller' OR title = 'Head Teller');
```

emp_id	assigned_branch_id
8	1
9	1
10	2
11	2
13	3
14	3
16	4
17	4
18	4

9 rows in set (0.03 sec)

Subqueries COM JOIN

```
mysql> SELECT branch_id  
      ->   FROM branch  
      -> WHERE name = 'Woburn Branch';  
  
+-----+  
| branch_id |  
+-----+  
|      2 |  
+-----+  
1 row in set (0.02 sec)
```

```
mysql> SELECT emp_id, assigned_branch_id  
      ->   FROM employee  
      -> WHERE start_date < '2007-01-01'  
      ->     AND (title = 'Teller' OR title = 'Head Teller');  
+-----+-----+  
| emp_id | assigned_branch_id |  
+-----+-----+  
|      8 |              1 |  
|      9 |              1 |  
|     10 |              2 |  
|     11 |              2 |  
|     13 |              3 |  
|     14 |              3 |  
|     16 |              4 |  
|     17 |              4 |  
|     18 |              4 |  
+-----+-----+  
9 rows in set (0.03 sec)
```

Subqueries com JOIN

```
SELECT a.account_id, a.cust_id, a.open_date, a.product_cd  
FROM account a INNER JOIN  
    (SELECT emp_id, assigned_branch_id  
     FROM employee  
     WHERE start_date < '2007-01-01' AND (title = 'Teller' OR title =  
        'Head Teller')) e  
ON a.open_emp_id = e.emp_id  
INNER JOIN  
    (SELECT branch_id  
     FROM branch  
     WHERE name = 'Woburn Branch') b  
ON e.assigned_branch_id = b.branch_id;
```

Mesma Tabela em uma Query JOIN

JOIN Statement

E quando precisamos
utilizar a mesma tabela em
um JOIN?

Mas preciso disso?



JOIN Statement

Chave estrangeira
refenciando a mesma
tabela

Exemplo 1

EMPREGADO

PNAME	MINITIAL	UNOME	SSN	DATANASC	ENDERECO	SEXO	SALARIO	SUPERSSN	DNO
-------	----------	-------	-----	----------	----------	------	---------	----------	-----

DEPARTAMENTO

DNAME	DNUMBER	GERSSN	GERDATAINICIO
-------	---------	--------	---------------

DEPTO_LOCALIZACOES

DNUMBER	DLOCALIZACAO
---------	--------------

PROJETO

PJNAME	PNUMBER	PLOCALIZACAO	DNUM
--------	---------	--------------	------

TRABALHA_EM

ESSN	PNO	HORAS
------	-----	-------

DEPENDENTE

ESSN	NOME_DEPENDENTE	SEXO	DATANASC	PARENTESCO
------	-----------------	------	----------	------------

JOIN Statement

```
mysql> SELECT a.account_id, e.emp_id,
->   b_a.name open_branch, b_e.name emp_branch
-> FROM account a INNER JOIN branch b_a
->   ON a.open_branch_id = b_a.branch_id
-> INNER JOIN employee e
->   ON a.open_emp_id = e.emp_id
-> INNER JOIN branch b_e
->   ON e.assigned_branch_id = b_e.branch_id
-> WHERE a.product_cd = 'CHK';
```

Exemplo 2

account_id	emp_id	open_branch	emp_branch
10	1	Headquarters	Headquarters
14	1	Headquarters	Headquarters
21	1	Headquarters	Headquarters
1	10	Woburn Branch	Woburn Branch
4	10	Woburn Branch	Woburn Branch
7	13	Quincy Branch	Quincy Branch
13	16	So. NH Branch	So. NH Branch
18	16	So. NH Branch	So. NH Branch
24	16	So. NH Branch	So. NH Branch
28	16	So. NH Branch	So. NH Branch

10 rows in set (0.16 sec)

Condição de Junção e Filtros

JOIN Statement

```
SELECT cust_id, name  
FROM business;
```

```
mysql> SELECT cust_id, name  
-> FROM business;  
+-----+  
| cust_id | name          |  
+-----+  
|      10 | Chilton Engineering  
|      11 | Northeast Cooling Inc.  
|      12 | Superior Auto Body  
|      13 | AAA Insurance Inc.  
+-----+  
4 rows in set (0.01 sec)
```

JOIN Statement

```
SELECT cust_id, name  
FROM customer;
```

```
mysql> SELECT cust_id  
-> FROM customer;
```

cust_id
1
2
3
4
5
6
7
8
9
10
11
12
13

```
13 rows in set (0.02 sec)
```

JOIN Statement

```
SELECT a.account_id, a.product_cd, c.fed_id,  
FROM account a INNER customer c  
ON a.cust_id = c.cust_id  
WHERE c.cust_type_cd = 'B';
```

account_id	product_cd	fed_id
24	CHK	04-1111111
25	BUS	04-1111111
27	BUS	04-2222222
28	CHK	04-3333333
29	SBL	04-4444444

5 rows in set (0.01 sec)

JOIN Statement

```
SELECT a.account_id, a.product_cd, c.fed_id,  
FROM account a INNER customer c  
ON a.cust_id = c.cust_id  
AND c.cust_type_cd = 'B';
```

O QUE ACONTECE SE
ERRARMOS A SINTAXE?



JOIN Statement

```
mysql> SELECT e.fname, e.lname, d.name  
    -> FROM employee e JOIN department d;
```

fname	lname	name
Michael	Smith	Operations
Michael	Smith	Loans
Michael	Smith	Administration
Susan	Barker	Operations
Susan	Barker	Loans
Susan	Barker	Administration
Robert	Tyler	Operations
Robert	Tyler	Loans
Robert	Tyler	Administration
Susan	Hawthorne	Operations
Susan	Hawthorne	Loans
Susan	Hawthorne	Administration



```
SELECT e.fname, e.lname, d.name  
FROM employee e JOIN department d
```

Hands On! **Projeto**

*“Falar é fácil.
Mostre-me o código!”*

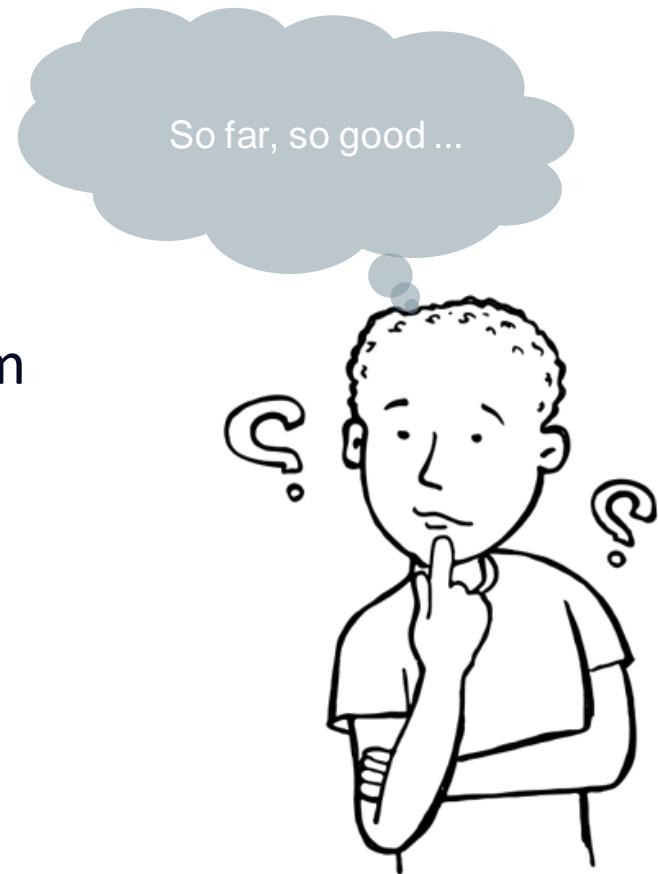
Linus Torvalds

Como utilizar o OUTER JOIN Statement?

OUTER JOIN

Linhas sem correspondencias não eram exibidas

Valores de ambas tabelas



OUTER JOIN

```
mysql> SELECT cust_id, name  
      -> FROM business;
```

```
+-----+  
| cust_id | name  
+-----+  
| 10     | Chilton Engineering  
| 11     | Northeast Cooling Inc.  
| 12     | Superior Auto Body  
| 13     | AAA Insurance Inc.  
+-----+  
4 rows in set (0.01 sec)
```

```
mysql> SELECT cust_id  
      -> FROM customer;
```

```
+-----+  
| cust_id |  
+-----+  
| 1       |  
| 2       |  
| 3       |  
| 4       |  
| 5       |  
| 6       |  
| 7       |  
| 8       |  
| 9       |  
| 10      |  
| 11      |  
| 12      |  
| 13      |  
+-----+
```

```
13 rows in set (0.02 sec)
```

OUTER JOIN

O que acontece no
OUTER JOIN?



OUTER JOIN

```
mysql> SELECT cust_id  
-> FROM customer;
```

```
+-----+  
| cust_id |  
+-----+
```

1
2
3
4
5
6
7
8
9

```
mysql> SELECT cust_id, name  
-> FROM business;  
+-----+-----+  
| cust_id | name  
+-----+-----+  
| 10 | Chilton Engineering  
| 11 | Northeast Cooling Inc.  
| 12 | Superior Auto Body  
| 13 | AAA Insurance Inc.  
+-----+-----+
```

4 rows in set (0.01 sec)

```
+-----+  
| cust_id |  
+-----+
```

10
11
12
13

13 rows in set (0.02 sec)

O que acontece no
OUTER JOIN?



OUTER JOIN

Matchs nas tabelas

```
mysql> SELECT cust_id, name  
-> FROM business;  
+-----+  
| cust_id | name  
+-----+  
| 10 | Chilton Engineering  
| 11 | Northeast Cooling Inc.  
| 12 | Superior Auto Body  
| 13 | AAA Insurance Inc.  
+-----+  
4 rows in set (0.01 sec)
```

```
mysql> SELECT cust_id  
-> FROM customer;  
+-----+  
| cust_id |  
+-----+  
| 1 |  
| 2 |  
| 3 |  
| 4 |  
| 5 |  
| 6 |  
| 7 |  
| 8 |  
| 9 |  
+-----+  
| 10 |  
| 11 |  
| 12 |  
| 13 |  
+-----+  
13 rows in set (0.02 sec)
```

O que acontece no
OUTER JOIN?



OUTER JOIN

SELECT <atributes list>

FROM <table1> ft **OUTER JOIN** <table2> st

ON ft.common_attribute =

st.common_attribute



OUTER JOIN

SELECT Tabela de orientação

FROM <table1> ft **OUTER JOIN** <table2> st

ON ft.common_attribute =

st.common_attribute



OUTER JOIN

```
SELECT Tabela de orientação  
FROM <table1> ft OUTER Condição de junção t  
ON ft.common_attribute =  
st.common_attribute
```



Hands On! **Projeto**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Orientação em Queries

JOIN: LEFT e RIGHT JOIN

LEFT e RIGHT JOIN

CROSS JOIN

Tabela 1: 10 linhas

Tabela 2: 20 linhas

```
mysql> SELECT c.cust_id, b.name  
-> FROM customer c LEFT OUTER JOIN business b  
-> ON c.cust_id = b.cust_id;
```

cust_id	name
1	NULL
2	NULL
3	NULL
4	NULL
5	NULL
6	NULL
7	NULL
8	NULL
9	NULL
10	Chilton Engineering
11	Northeast Cooling Inc.
12	Superior Auto Body
13	AAA Insurance Inc.

13 rows in set (0.00 sec)

Hands On! **Projeto**

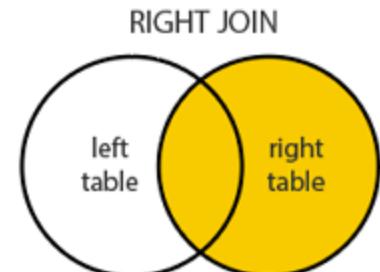
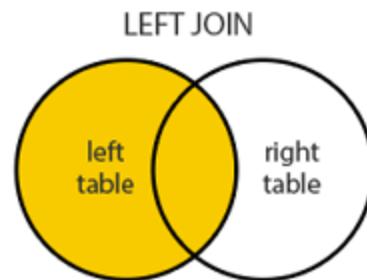
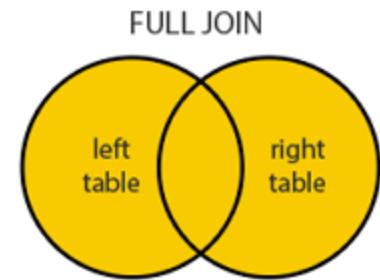
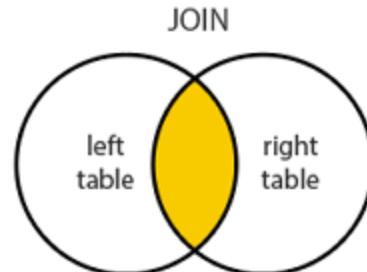
*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

Entendendo melhor os tipos de JOINs

INNER ou OUTER JOIN?

A tabela referenciada modifica o resultado da query

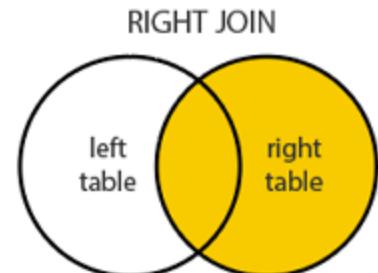
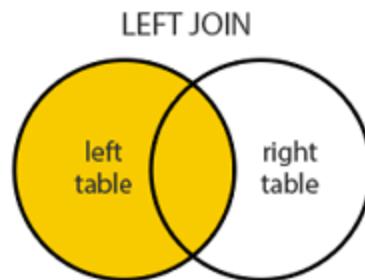
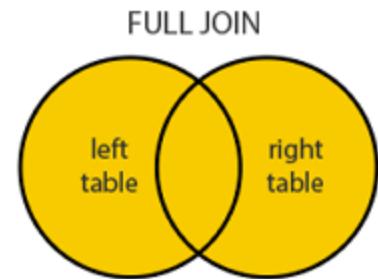
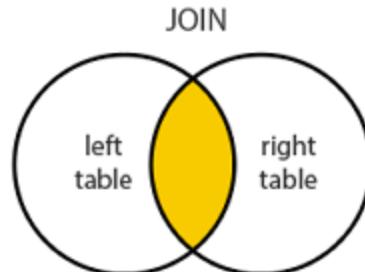


Fonte: dofactory.com

INNER ou OUTER JOIN?

A tabela referenciada modifica o resultado da query

Só usa com OUTER JOIN?



Fonte: dofactory.com

Diferenças

INNER JOIN

FULL OUTER JOIN

JOIN

FULL OUTER JOIN

LEFT JOIN

LEFT OUTER JOIN

RIGTH JOIN

RIGHT OUTER JOIN

Diferenças

INNER JOIN

FULL OUTER JOIN

JOIN

LEFT JOIN



RIGTH JOIN

FULL OUTER JOIN

LEFT OUTER JOIN

RIGHT OUTER JOIN

Diferenças

INNER JOIN

FULL OUTER JOIN

JOIN

Interseção

LEFT JOIN

FULL OUTER JOIN

Matchs

RIGTH JOIN

LEFT OUTER JOIN

RIGHT OUTER JOIN

INNER ou OUTER JOIN?

LEFT JOIN



Everything on the left
+
anything on the right that matches

```
SELECT *
FROM TABLE_1
LEFT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI LEFT JOIN



Everything on the left
that is NOT on the right

```
SELECT *
FROM TABLE_1
LEFT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_2.KEY IS NULL
```

RIGHT JOIN



Everything on the right
+
anything on the left that matches

```
SELECT *
FROM TABLE_1
RIGHT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI RIGHT JOIN



Everything on the right
that is NOT on the left

```
SELECT *
FROM TABLE_1
RIGHT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_1.KEY IS NULL
```

OUTER JOIN



Everything on the right
+
Everything on the left

```
SELECT *
FROM TABLE_1
OUTER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI OUTER JOIN



Everything on the left and right
that is unique to each side

```
SELECT *
FROM TABLE_1
OUTER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_1.KEY IS NULL
OR TABLE_2.KEY IS NULL
```

INNER JOIN



Only the things that match on the
left AND the right

```
SELECT *
FROM TABLE_1
INNER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

CROSS JOIN



All combination of rows from the
right and the left (cartesian product)

```
SELECT *
FROM TABLE_1
CROSS JOIN TABLE_2
```

Diferenças

INNER JOIN

FULL OUTER JOIN

JOIN

CROSS JOIN

& FULL OUTER JOIN

LEFT JOIN

LEFT OUTER JOIN

RIGTH JOIN

RIGHT OUTER JOIN

Diferenças

CROSS JOIN

&

FULL OUTER JOIN

Tabela 1: 10 linhas

Tabela 2: 20 linhas

Diferenças

CROSS JOIN

&

FULL OUTER JOIN

Tabela 1: 10 linhas

Tabela 2: 20 linhas

CROSS JOIN: $10 * 20 = 200$ linhas

Diferenças

CROSS JOIN

&

FULL OUTER JOIN

Tabela 1: 10 linhas

Tabela 2: 20 linhas

CROSS JOIN: $10 * 20 = 200$ linhas

5 linhas como combinação

Diferenças

CROSS JOIN

&

FULL OUTER JOIN

Tabela 1: 10 linhas

Tabela 2: 20 linhas

INNER JOIN: interseção = 5 linhas

5 linhas como combinação

Diferenças

CROSS JOIN

&

FULL OUTER JOIN

Tabela 1: 10 linhas

Tabela 2: 20 linhas

FULL OUTER JOIN: combinação = 25 linhas

5 linhas como combinação

Explorando ainda mais: **NATURAL JOIN**

NATURAL JOIN



NATURAL JOIN

- NATURAL JOIN entre R & S
- Condição de junção implícita
- Atributos comuns



NATURAL JOIN

- NATURAL JOIN entre R & S
- Condição de junção implícita
- Atributos comuns



Atributo com mesmo nome

EMPLOYEE.Dno = DEPT.Dno

```
SELECT Fname, Lname, Address  
FROM (EMPLOYEE NATURAL JOIN  
      (DEPARTMENT AS DEPT (Dname, Dno, Mssn, Msdate)))  
WHERE Dname = 'Research';
```

NATURAL JOIN

Atributo com mesmo nome

EMPLOYEE.Dno = DEPT.Dno

```
SELECT a.account_id, a.cust_id,  
c.cust_type_cd, c.fed_id  
FROM account a NATURALJOIN customer c;
```



account_id	cust_id	cust_type_cd	fed_id
1	1	I	111-11-1111
2	1	I	111-11-1111
3	1	I	111-11-1111
4	2	I	222-22-2222
5	2	I	222-22-2222
6	3	I	333-33-3333
7	3	I	333-33-3333
8	4	I	444-44-4444
9	4	I	444-44-4444
10	4	I	444-44-4444

NATURAL JOIN

Atributo com mesmo nome

EMPLOYEE.Dno = DEPT.Dno

account_id	cust_id	cust_type_cd	fed_id
1	1	I	111-11-1111
2	1	I	111-11-1111
3	1	I	111-11-1111
4	2	I	222-22-2222
5	2	I	222-22-2222
6	3	I	333-33-3333
7	3	I	333-33-3333
8	4	I	444-44-4444
9	4	I	444-44-4444
10	4	I	444-44-4444

11	5	I	555-55-5555
12	6	I	666-66-6666
13	6	I	666-66-6666
14	7	I	777-77-7777
15	8	I	888-88-8888
16	8	I	888-88-8888
17	9	I	999-99-9999
18	9	I	999-99-9999
19	9	I	999-99-9999
20	10	B	04-1111111
21	10	B	04-1111111
22	11	B	04-2222222
23	12	B	04-3333333
24	13	B	04-4444444

24 rows in set (0.02 sec)

NATURAL JOIN

Se se não houver atributos com mesmo nome?



Hands On! **Projeto**

*“Falar é fácil.
Mostre-me o código!”*

Linus Torvalds

DESAFIO DE PROJETO

Construindo seu Primeiro Projeto Lógico de BD

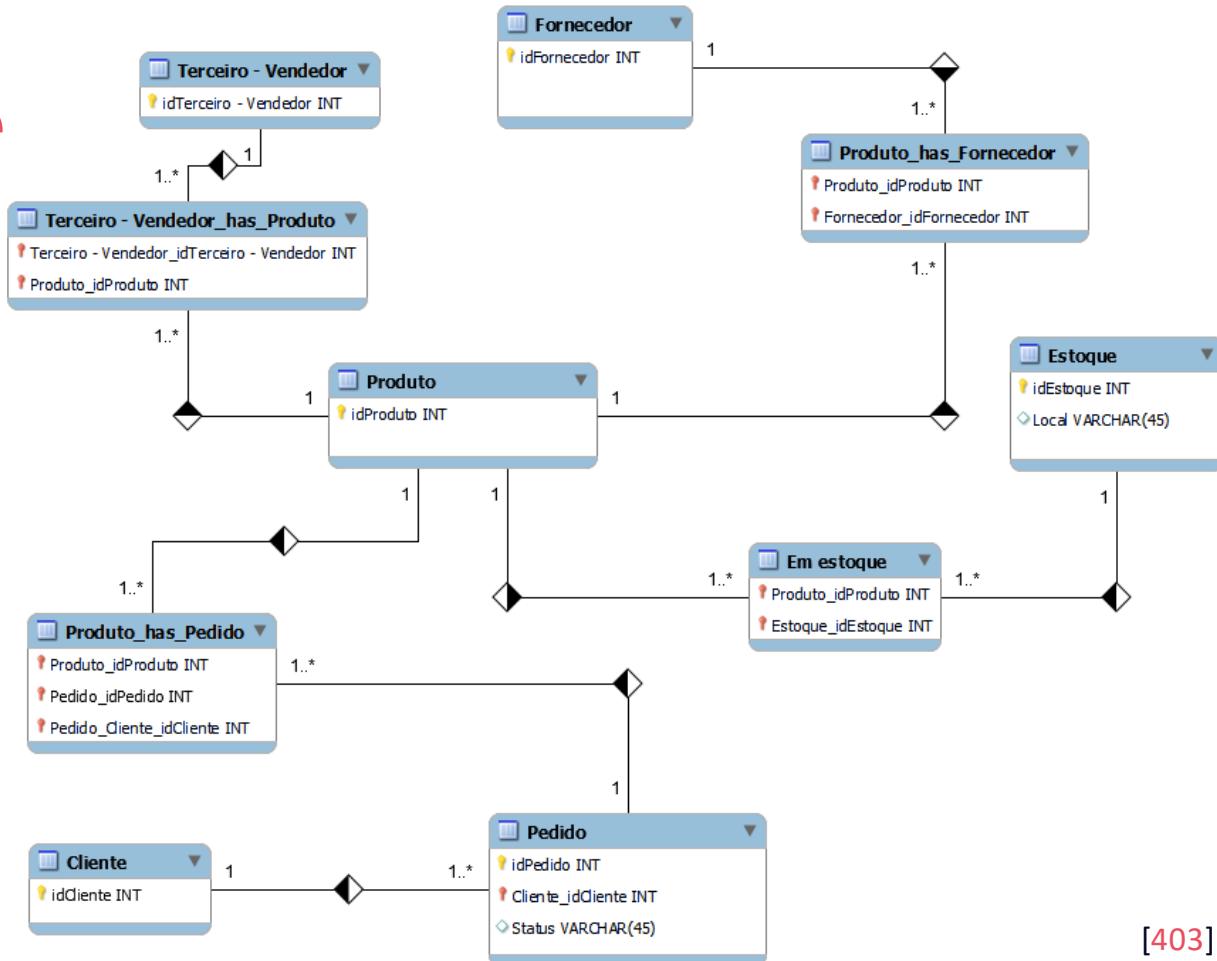
// Explorando a Linguagem de Consulta a Banco de Dados SQL

Desafio de Projeto

1. Mapeamento do esquema ER para Relacional
2. Definição do script SQL para criação do esquema de banco de dados
3. Persistência de dados para testes
4. Recuperação de informações com queries SQL



E-commerce



DESAFIO DE PROJETO

Entendendo o Desafio

// Explorando a Linguagem de Consulta a Banco de Dados SQL

Desafio de Projeto - Diretrizes

- Não há um mínimo de queries a serem realizadas
- Os tópicos supracitados devem estar presentes nas queries
- Elabore perguntas que podem ser respondidas pelas consultas
- As cláusulas podem estar presentes em mais de uma query



Desafio de Projeto - Diretrizes

Após a criação do esquema lógico, realize a criação do Script SQL para criação do esquema do banco de dados. Posteriormente, realize a persistência de dados para realização de testes. Especifique ainda queries mais complexas do que apresentadas durante a explicação do desafio.



Desafio de Projeto - Diretrizes

Sendo assim, crie queries SQL com as cláusulas abaixo:

- Recuperações simples com SELECT Statement
- Filtros com WHERE Statement
- Crie expressões para gerar atributos derivados
- Defina ordenações dos dados com ORDER BY
- Condições de filtros aos grupos – HAVING Statement
- Crie junções entre tabelas para fornecer uma perspectiva mais complexa dos dados



DESAFIO DE PROJETO

Construindo um Projeto Lógico de BD do zero

// Explorando a Linguagem de Consulta a Banco de Dados SQL

Desafio - Oficina

Agora você irá criar um esquema lógico a partir do esquema conceitual que criou para o cenário de uma oficia. Utilize o algoritmo de mapeamento apresentado para determinar qual a melhor opção para mapear seus componentes para modelo relacional. Pense que posteriormente você irá criar o esquema do banco de dados com base neste esquema lógico.

Desafio de Projeto

1. Mapeamento do esquema ER para Relacional
2. Definição do script SQL para criação do esquema de banco de dados
3. Persistência de dados para testes
4. Recuperação de informações com queries SQL



Desafio de Projeto - Diretrizes

- Não há um mínimo de queries a serem realizadas
- Os tópicos supracitados devem estar presentes nas queries
- Elabore perguntas que podem ser respondidas pelas consultas
- As cláusulas podem estar presentes em mais de uma query

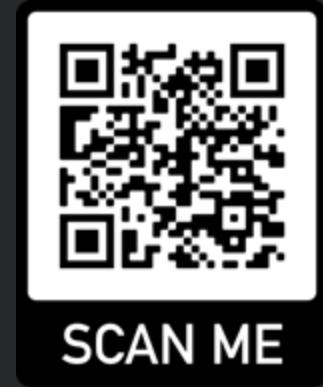


Referências Bibliográficas



Dúvidas?

- > Fórum/Artigos
- > Comunidade Online (Discord)



Para saber mais

Livros de Referência

- ELMASRI, Ramez; B. NAVATHE, Shamkant. **Sistema de Banco de Dados**. 7. ed. São Paulo: Pearson, 2018.

Livros complementares

- RODZVILLA, John. A Review of “**Learning SQL**” Beaulieu, Alan. Sebastopol, CA: O'Reilly, 2009, 320 pp., \$39.99, ISBN 978-0-596-52083-0. 2010.
- TEOREY, Toby J. **Database modeling and design**. Morgan Kaufmann, 1999.
- NIELD, Thomas. **Getting Started with SQL: A Hands-On Approach for Beginners**. " O'Reilly Media, Inc.", 2016.

Links Úteis

Documentação Oficial

- Documentação: <http://dev.mysql.com/doc/>
- Download: <http://dev.mysql.com/downloads/>
- Site oficial : <http://www.mysql.com/>
- Suporte: <http://www.mysql.com/support/>
- <https://dev.mysql.com/doc/refman/8.0/en/preface.html>

Links Úteis

<https://support.microsoft.com/pt-br/office/acesso-sql-conceitos-b%C3%A1sicos-vocabul%C3%A1rio-e-sintaxe-444d0303-cde1-424e-9a74-e8dc3e460671#:~:text=SQL%20cl%C3%A1usulas,as%20cl%C3%A1usulas%20SQL%20mais%20comuns.>

<https://www.analyticsvidhya.com/blog/2020/11/guide-data-types-mysql-data-science-beginners/#:~:text=LONGTEXT%20can%20store%20the%20maximum,any%20long%2Dform%20text%20strings.>

<https://dev.mysql.com/doc/refman/8.0/en/create-database.html>

<https://www.digitalocean.com/community/tutorials/how-to-create-a-new-user-and-grant-permissions-in-mysql>

https://basedosdados.org/dataset/br-me-caged?bdm_table=microdados_antigos

Links Úteis

<https://docs.microsoft.com/pt-br/sql/relational-databases/tables/add-columns-to-a-table-database-engine?view=sql-server-ver16>

<https://www.mysqltutorial.org/mysql-primary-key/#:~:text=Because%20MySQL%20works%20faster%20with,that%20the%20table%20may%20have%20.%20>

<https://www.c-sharpcorner.com/article/the-complete-reference-set-operations-in-ms-sql-union-all-intersect-excep/#:~:text=UNION%20ALL%20combines%20two%20or,in%20the%20second%20result%20set.>

<https://dotnettutorials.net/lesson/differences-between-union-except-and-intersect-operators-in-sql-server/#:~:text=UNION%20ALL%3A%20Combine%20two%20or,no%20matching%20to%20each%20other>

Dúvidas?

- > Fórum/Artigos
- > Comunidade Online (Discord)

