

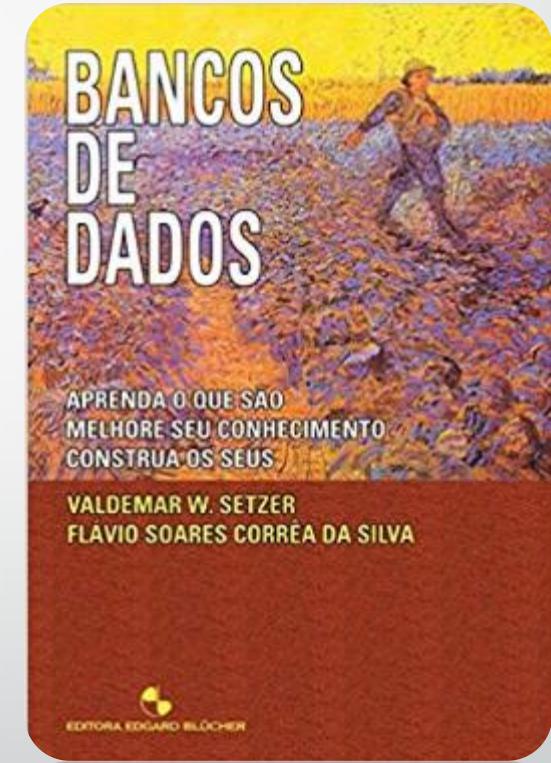
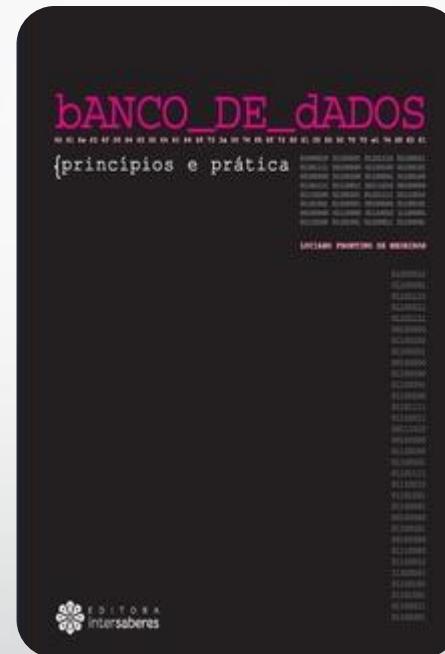
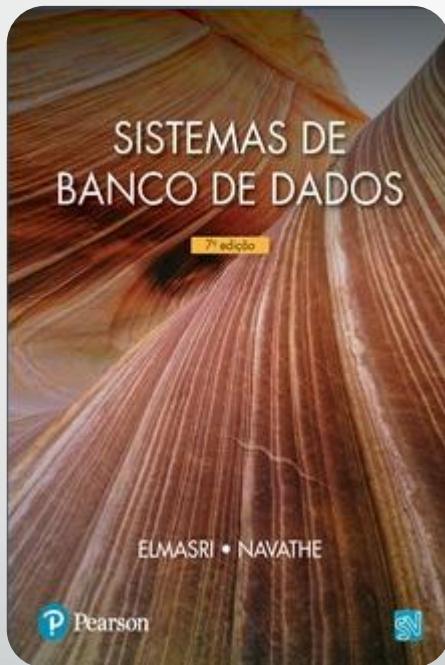
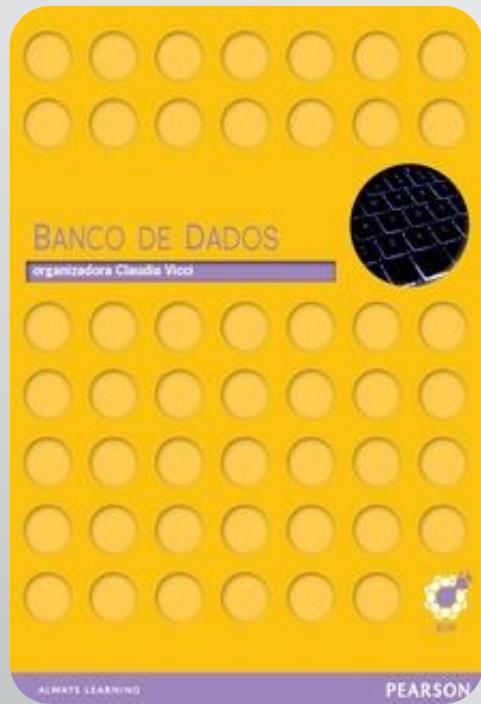
S112 – Banco de Dados

Professor MSc. Eng. Márcio José de Lemos

E-mail: marcio.lemos@senairs.org.br

<http://lattes.cnpq.br/4769158065464009>

Bibliografia Básica



Projeto de Banco de Dados

Um projeto de banco de dados

- É **sub-dividido** em etapas onde o objetivo é a criação de um banco de dados otimizado que atenda as expectativas do cliente.
- E nesse contexto, os modelos de dados são muito importantes para a **transmissão de idéias entre o cliente e o projetista**, bem como facilitar a manutenção do banco de dados no futuro.

PRIMEIRA ETAPA: ANÁLISE DE REQUISITOS

- A **primeira etapa** do projeto de banco de dados é a identificação dos requisitos que o banco de dados deve atender.
- Nesta fase devem ser realizadas entrevistas com as pessoas envolvidas no processo, cria-se uma descrição textual macro do processo (**texto conhecido como mini-mundo**), modelos externos (que devem ser entendidos por todos).

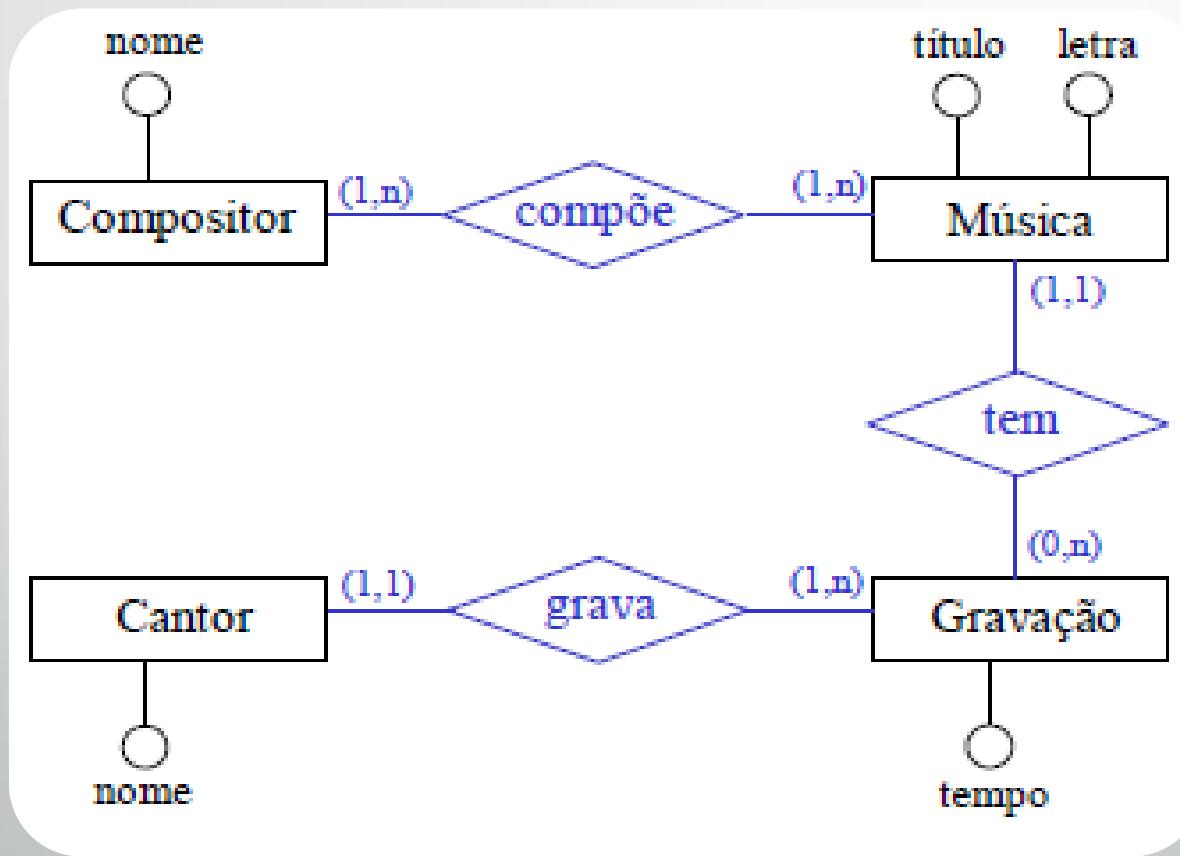
PRIMEIRA ETAPA: ANÁLISE DE REQUISITOS

- Este é o momento em que as regras de negócio devem ser identificadas, se a fase de análise de requisitos for mal executada e se identificar regras de negócio que não representam a realidade, tudo o que for feito em seguida no projeto será perda de tempo.
- Por isso, esta é considerada a parte mais importante do projeto.

O projeto de um novo banco de dados dá-se em três fases, descritas a seguir:

- **Modelagem conceitual** – nesta primeira fase, é construído um modelo conceitual, na forma de um diagrama entidade-relacionamento.
- Este modelo captura as necessidades da organização em termos de **armazenamento de dados** de forma independente de implementação.

O projeto de um novo banco de dados dá-se em três fases, descritas a seguir:

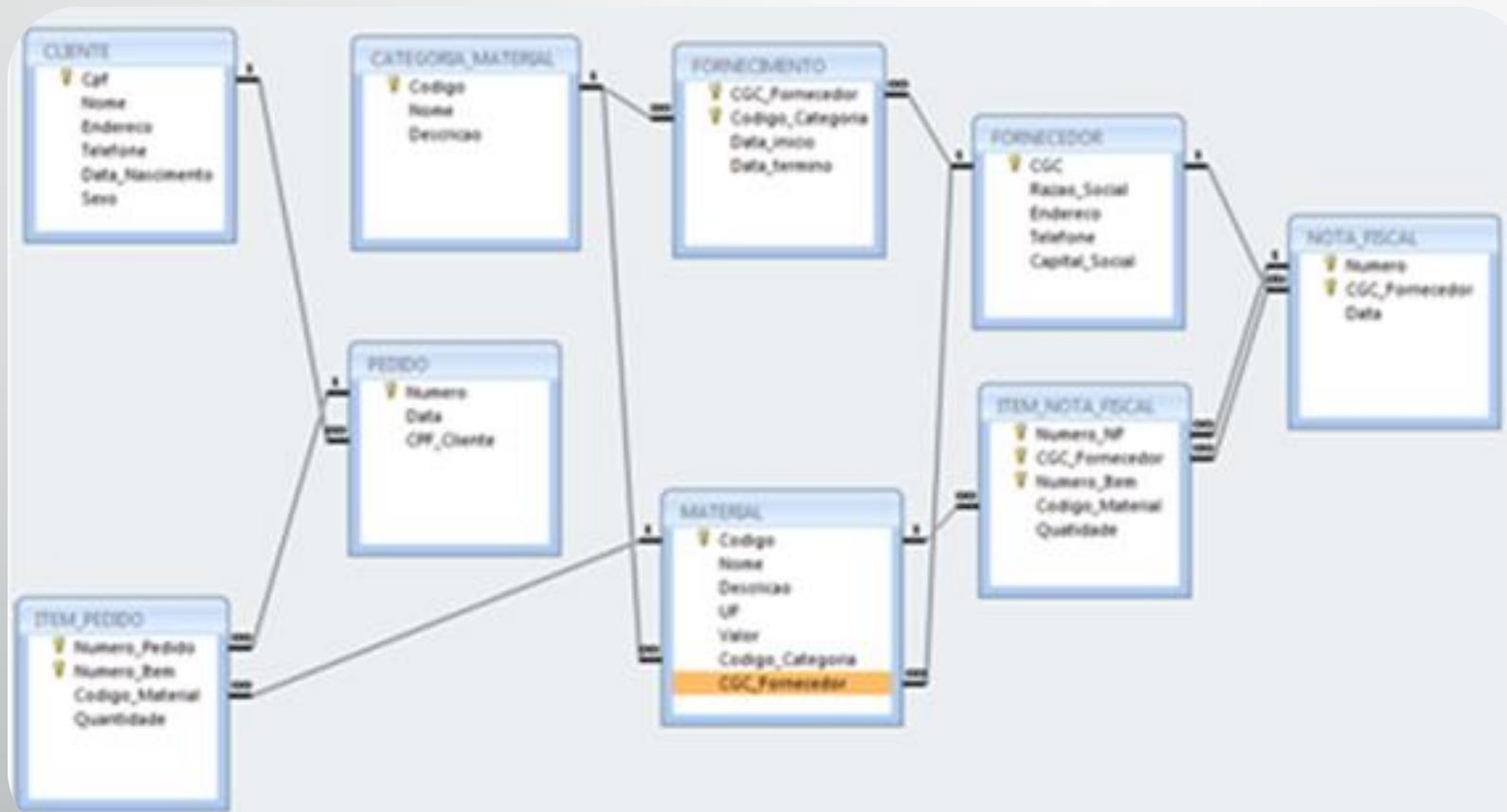




O projeto de um novo banco de dados dá-se em três fases, descritas a seguir:

- Projeto lógico – a etapa de projeto lógico objetiva transformar o modelo conceitual obtido na primeira fase em um modelo lógico.
- O modelo lógico define como o banco de dados será implementado em um SGBD específico.

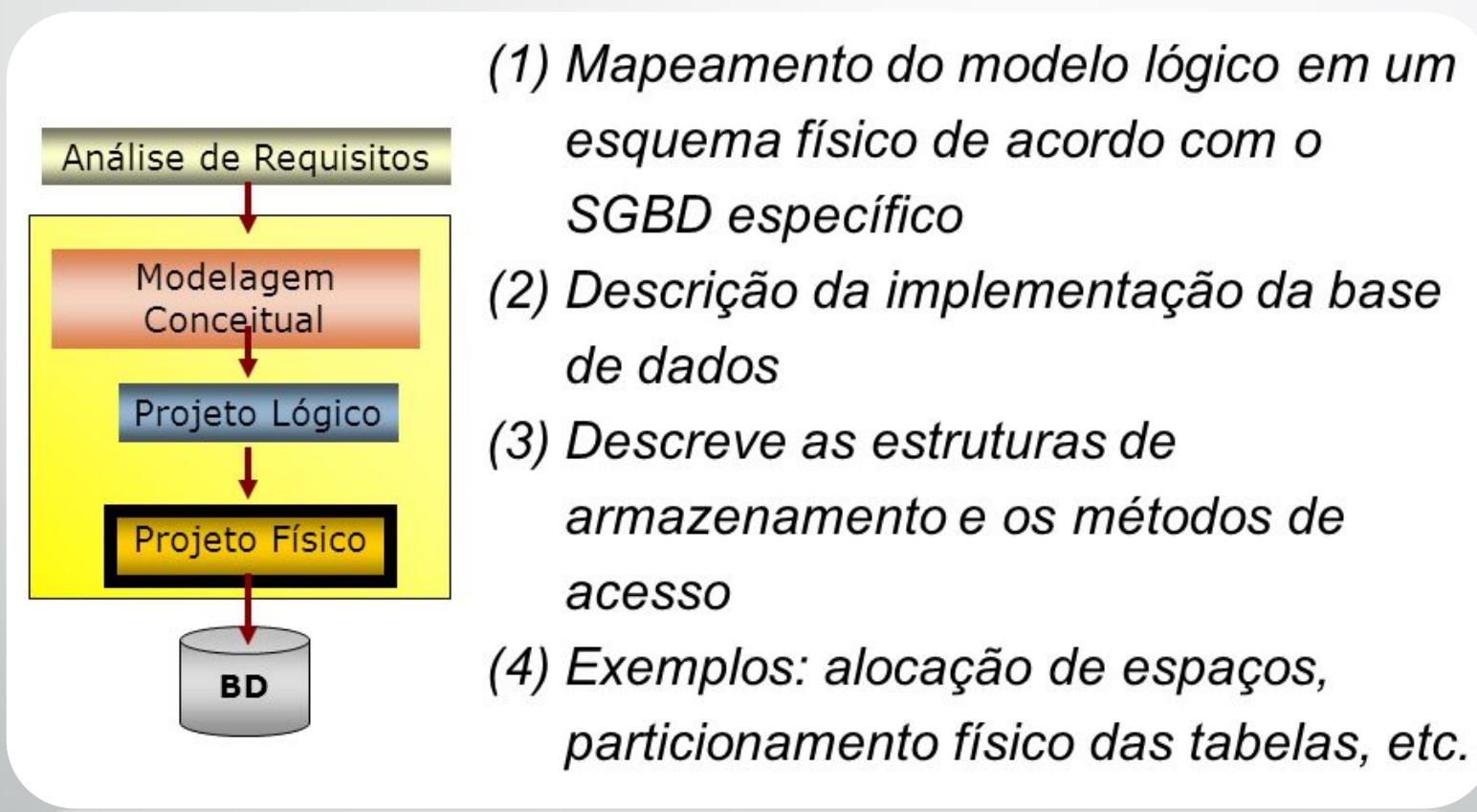
O projeto de um novo banco de dados dá-se em três fases, descritas a seguir:



O projeto de um novo banco de dados dá-se em três fases, descritas a seguir:

- Projeto físico – na etapa de projeto físico, o modelo do banco de dados é enriquecido com detalhes que influenciam no desempenho do banco de dados, mas não interfere em sua funcionalidade.
- O modelo obtido neste passo é o modelo físico do banco de dados.
- Este processo normalmente é chamado de sintonia (“tuning”). -> continuo.

O projeto de um novo banco de dados dá-se em três fases, descritas a seguir:



A sintaxe dos scripts

- Aqui, se define a sintaxe dos scripts para o produto específico: **Oracle, SQL Server, MySQL ou PostgreSQL**, por exemplo.
- **Cada um tem suas especificações** de instalação e melhores práticas, bem como **plataformas** onde serão executados.

A sintaxe dos scripts

- Por exemplo, o Oracle pode ser usado em vários sistemas operacionais, como Windows, Linux e Solaris.
- Já o SQL Server é instalado no sistema operacional do seu fabricante, a Microsoft.
- Eses detalhes são discutidos nessa fase, levando-se em conta aspectos técnicos e orçamentários, como aquisição de licenças e especialistas no produto para suporte.



A sintaxe dos scripts

- Nessa fase também se aplicam as regras de segurança: quem terá acesso a qual informação no banco de dados, bem como rotinas de *backup* e restauração.
- Soluções de administração do banco de dados, replicação ou redundância também podem ser definidos aqui.

Dessa forma, percebemos duas peculiaridades no projeto de banco de dados:

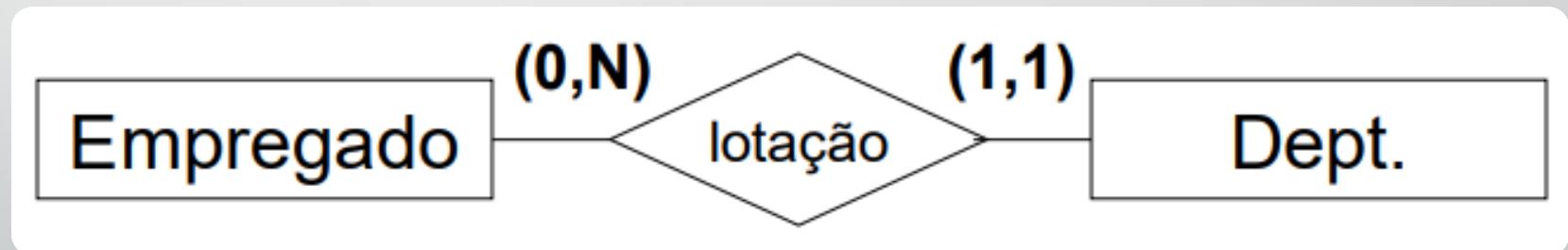
- No início do projeto, a tônica é a necessidade do solicitante. Ouvi-lo nessa fase é de suma importância para traduzir o desenho em um modelo coerente e satisfatório.
- No final do projeto, o foco maior está em como implementar o projeto, garantindo a performance e disponibilidade do sistema.

Cardinalidade de Relacionamento

- **Mínima** – Indica se a ocorrência de uma entidade em um relacionamento é obrigatória ou opcional

1 – obrigatória

0 – opcional



Classificação de Relacionamento

- A cardinalidade máxima pode ser utilizada para classificar relacionamentos binários

Envolvem duas entidades

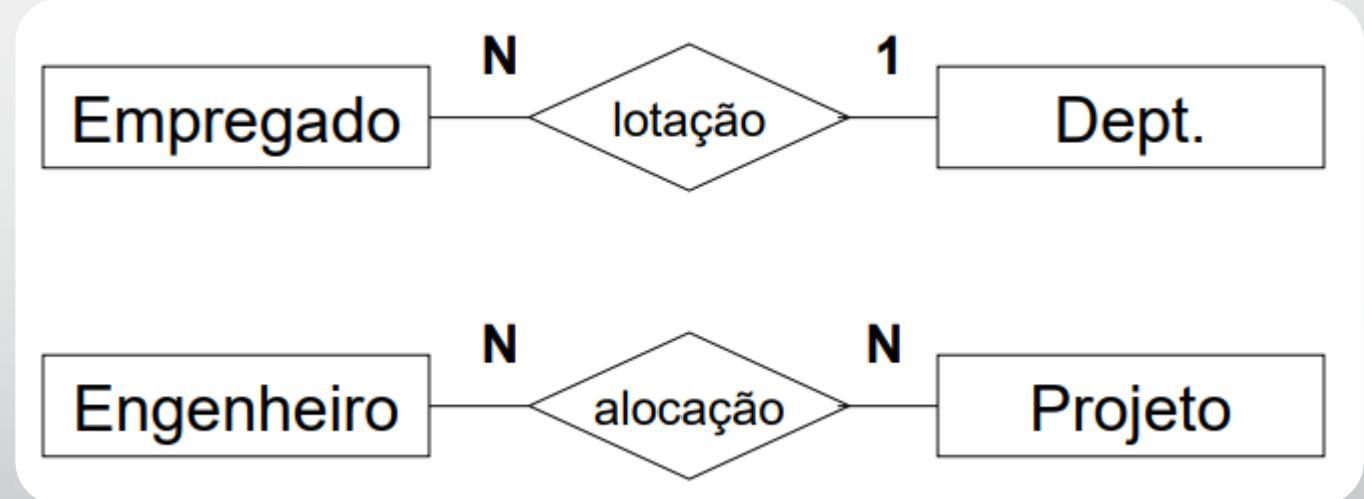
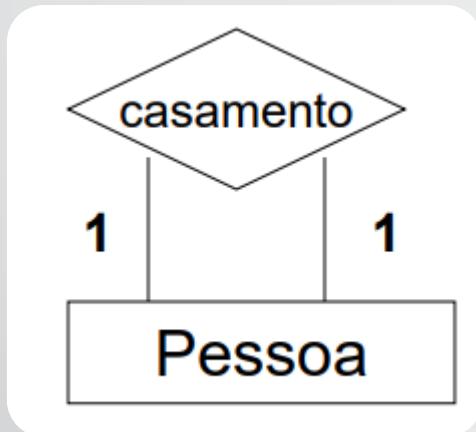
Relacionamentos binários:

N:N (muitos para muitos)

1:N (um para muitos)

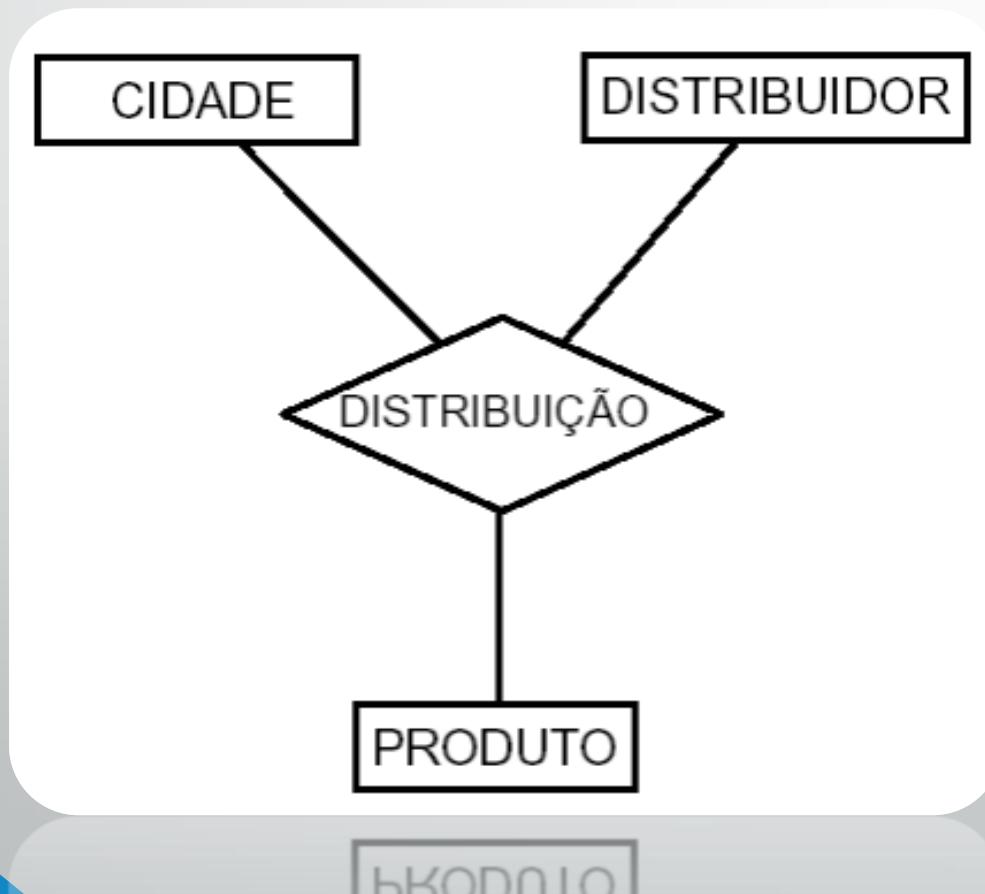
1:1 (um para um)

Classificação de Relacionamento

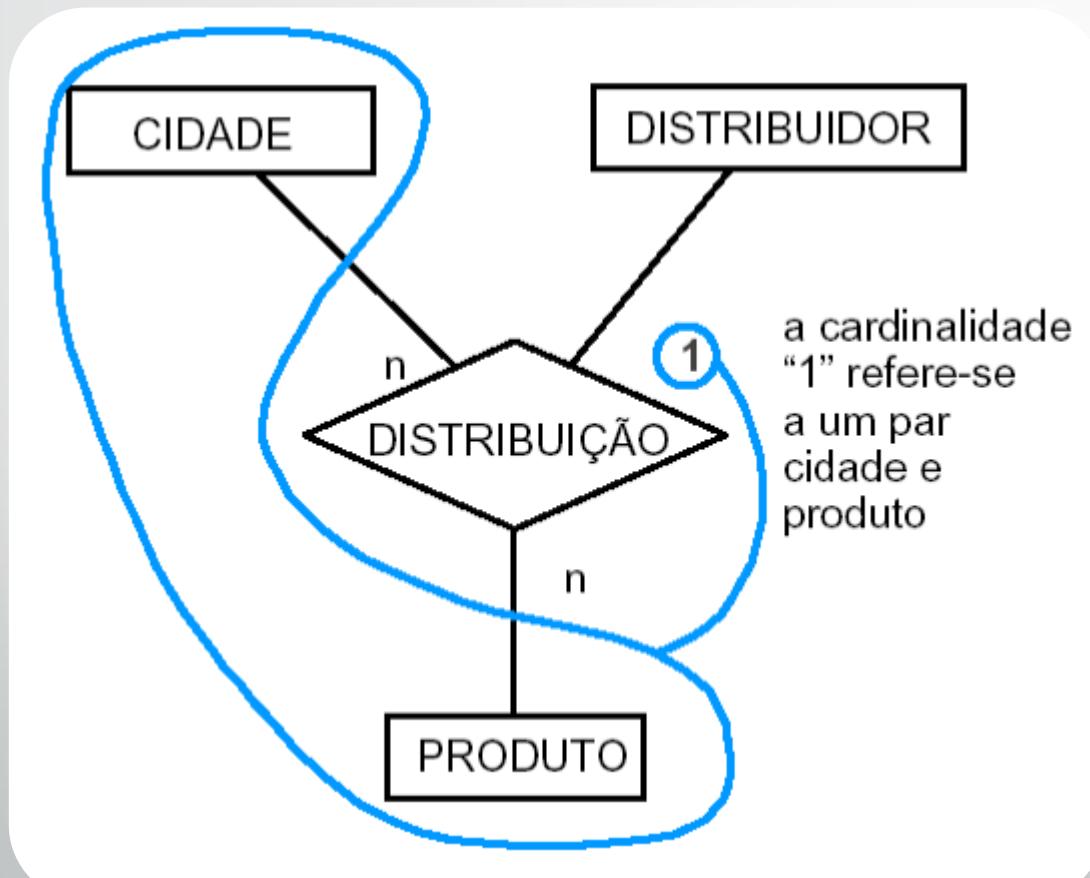


Relacionamento N-ário

Associação entre três entidades:

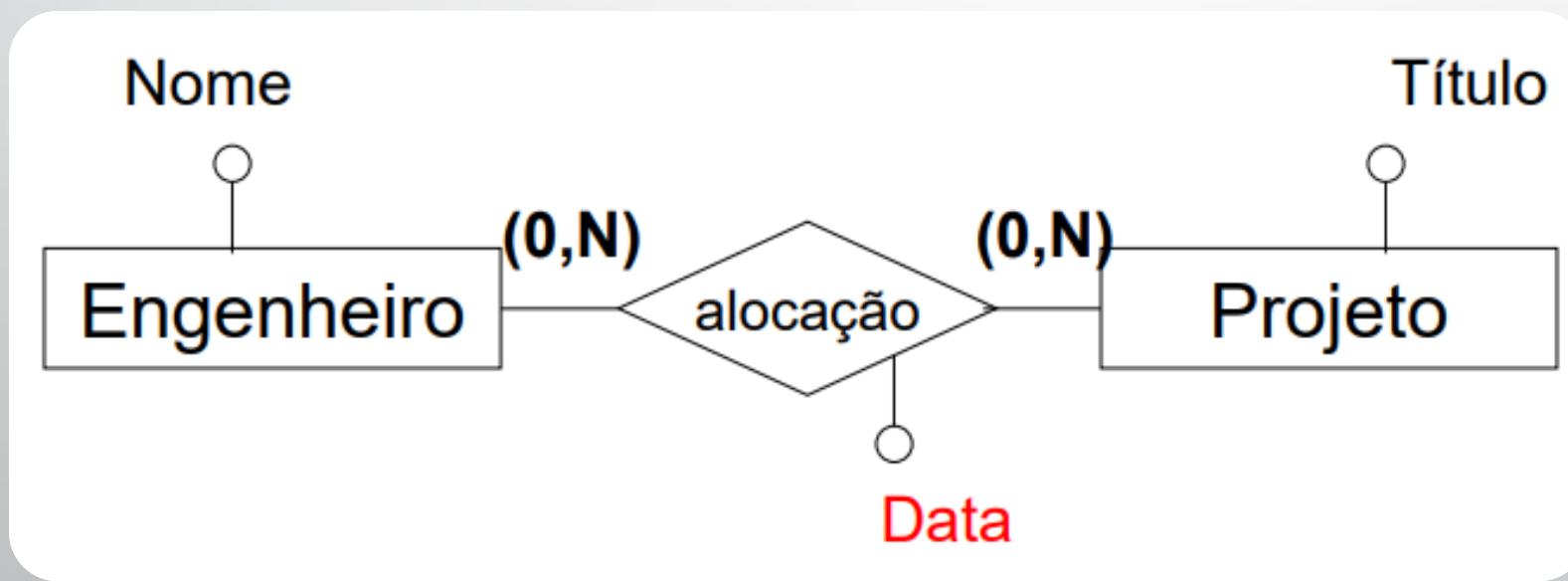


Relacionamento N-ário



MER: Atributo

Informação associada a cada ocorrência de uma entidade ou relacionamento:





Cardinalidade de Atributo

Cardinalidades definem classificações para os atributos

- Obrigatório ou opcional
- Monovalorado ou multivvalorado

Cardinalidade de Atributo

Cardinalidade mínima:

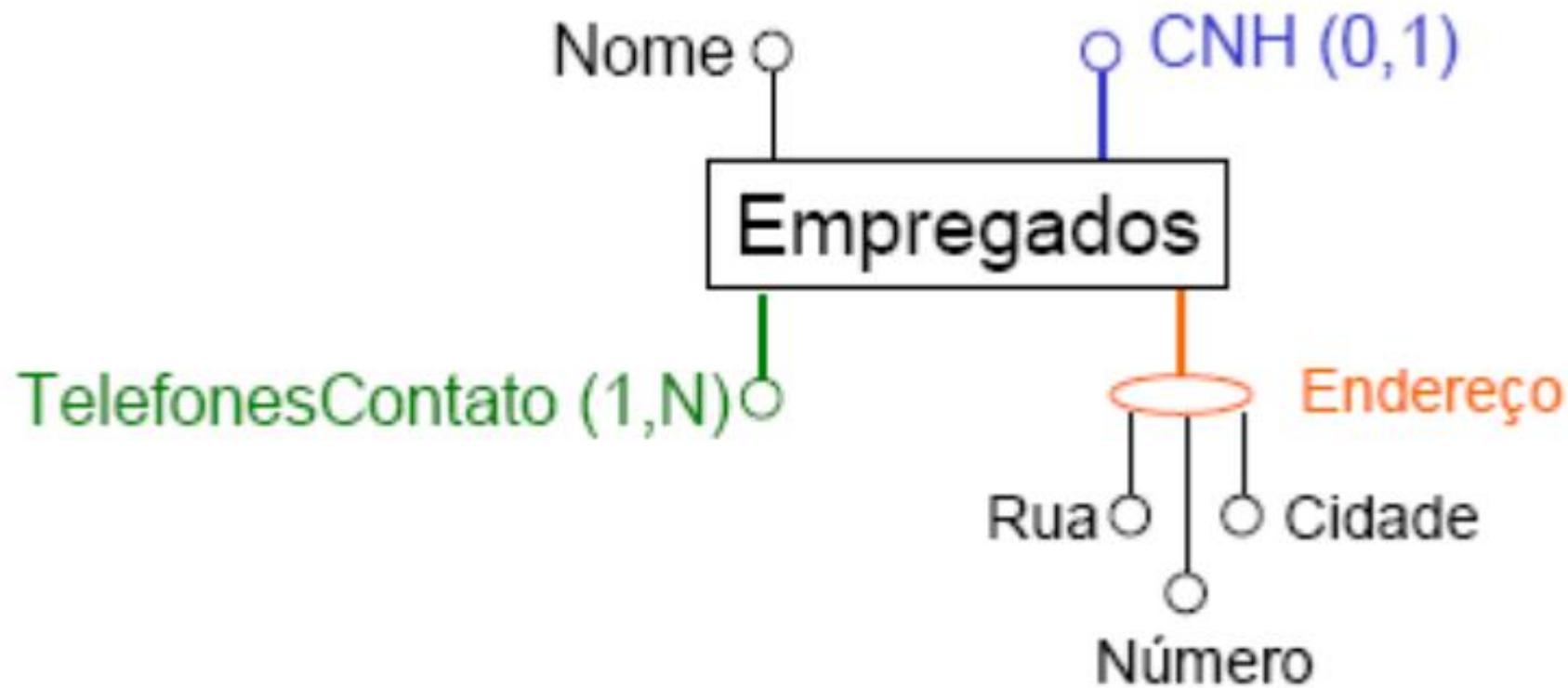
- **Atributo obrigatório** (cardinalidade 1)
 - Cada entidade possui no mínimo um valor associado
- **Atributo opcional** (cardinalidade 0 (ZERO))
 - Pode conter valores nulos

Cardinalidade de Atributo

Cardinalidade máxima

- **Atributo monovalorado (cardinalidade 1)**
 - Cada entidade possui no máximo um valor associado
- **Atributo multivalorado (cardinalidade “N”)**
 - Pode conter “N” valores
 - SQL não possui representação para atributos multivalorados.

Cardinalidade de Atributo



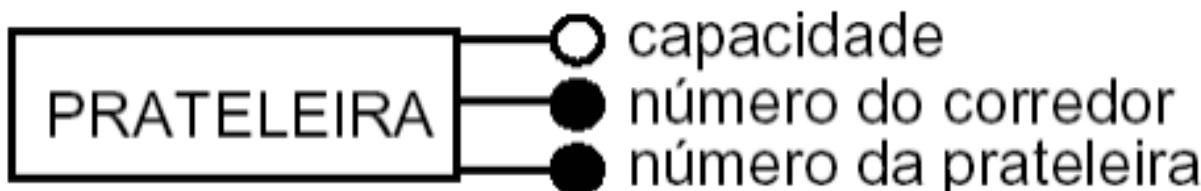
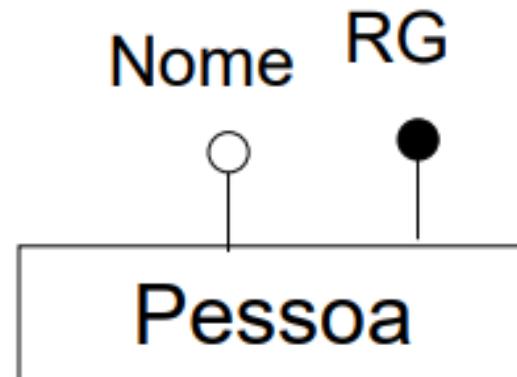
Atributo Identificador

Identificador:

Conjunto de propriedades cujos valores distinguem as ocorrências das entidades

- **Identificam unicamente uma entidade**
 - Pessoa possui único RG
 - RG é um atributo identificador

Atributo Identificador



Generalização/Especialização

Uma especialização de uma entidade representa um caso especial da mesma

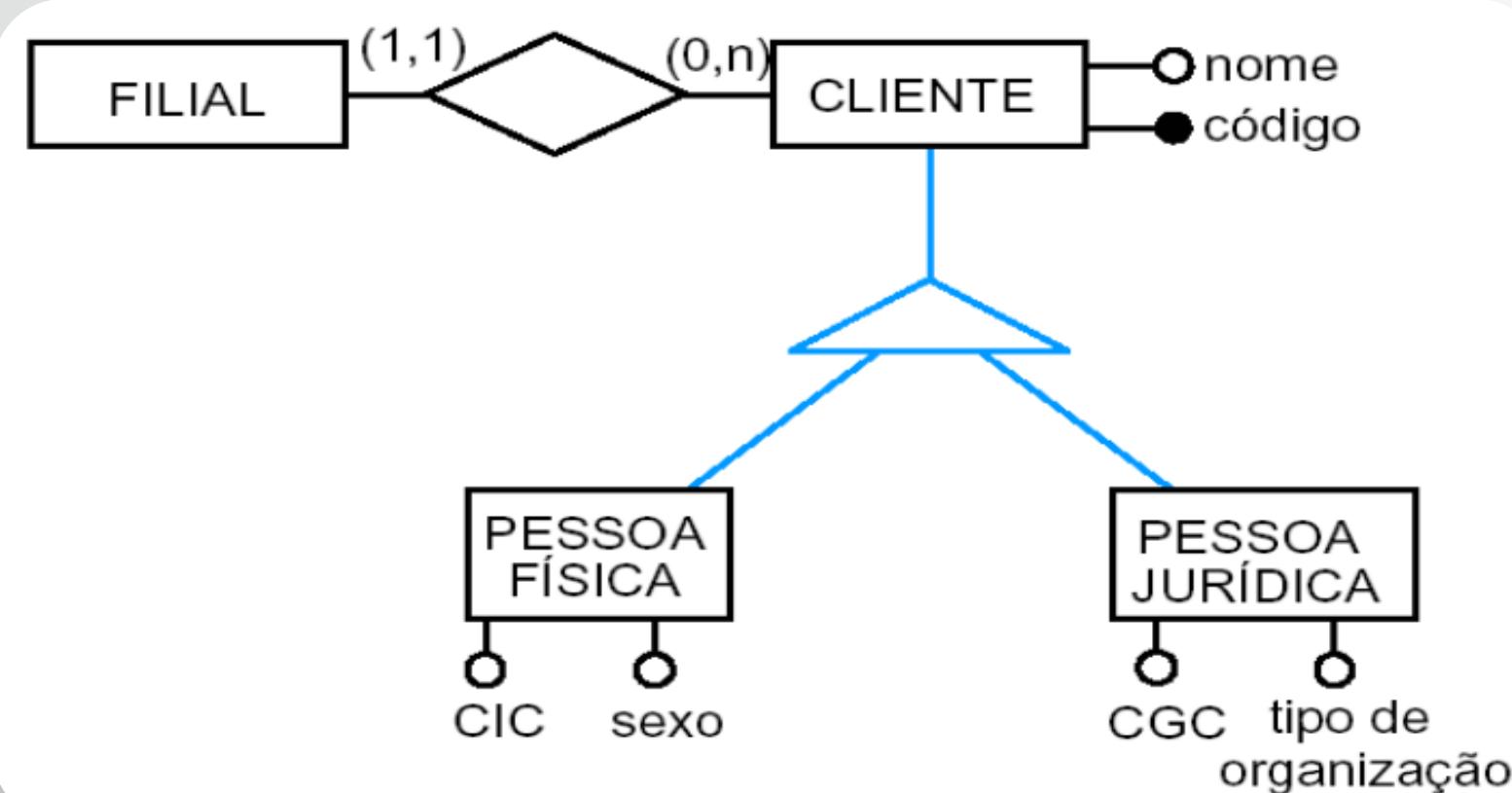
- Ex.: Carro e moto são tipos especiais de veículos automotores

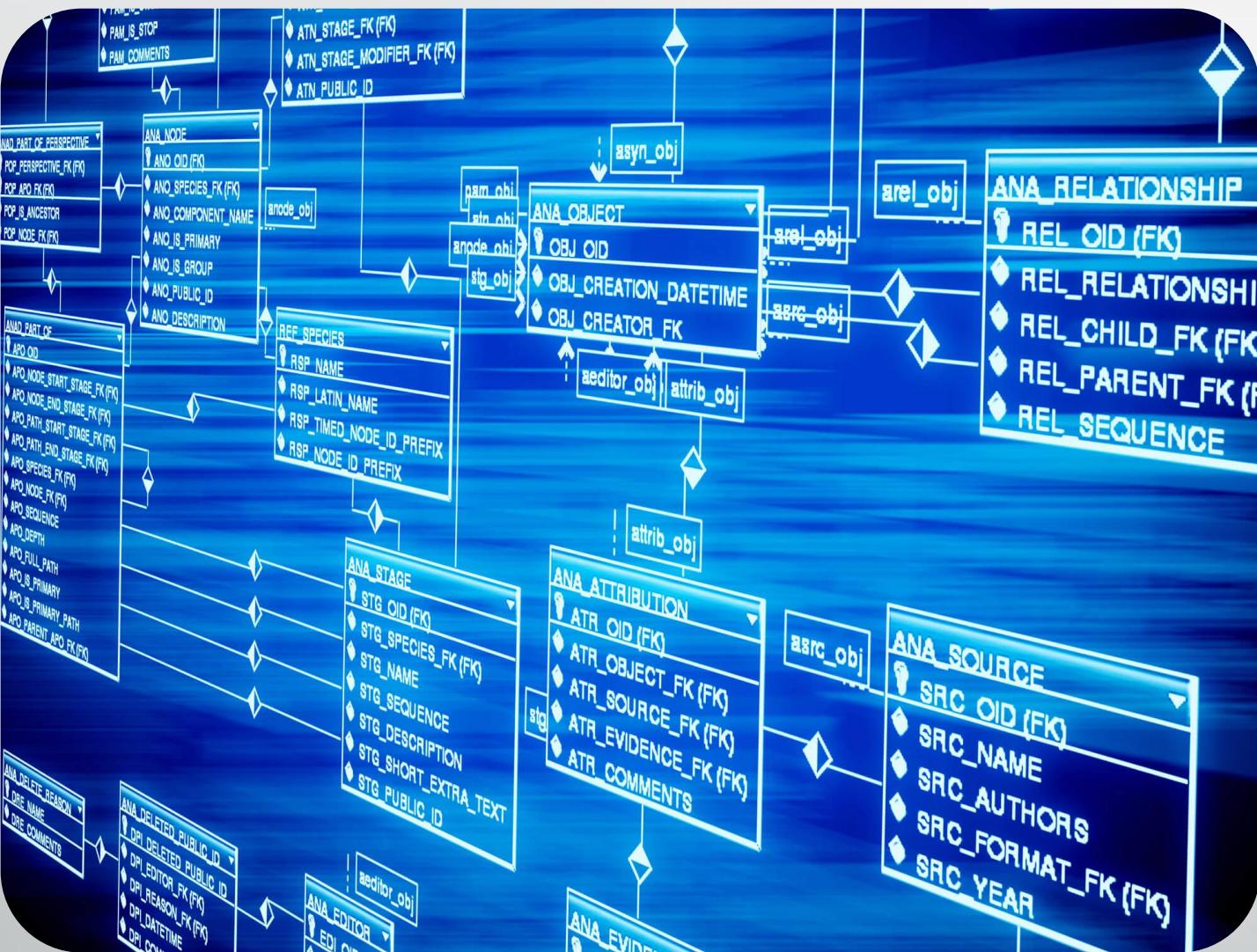
Atributo Identificador

A generalização é a entidade genérica representando uma classe de indivíduos com atributos comuns

– Ex.: Veículos automotores

Generalização/Especialização





Generalização/Especialização

- Associado a estes conceitos está o conceito de herança
Herança de propriedade
 - Atributos da entidade genérica passam à entidade especializada

Considerações Finais

- **Outros aspectos do MER**
 - Tipos de generalização
 - Diferentes notações
 - Ampla utilização – padrão de fato
- **Ferramentas**
 - ER Win
 - Oracle Designer
- **Outras abordagens**
 - NIAM/ORM (técnica européia 1970)
 - UML

Álgebra Relacional

- Álgebra desenvolvida para descrever operações sobre uma base de dados relacional.
- Os objetivos sobre os quais a Álgebra opera são tabelas
- Uma operação possui como operadores e como resultado tabelas

Álgebra Relacional

Porque aprender:

- Compreendendo álgebra relacional é mais fácil aprender SQL;
- SQL incorpora cada vez mais conceitos de álgebra;
- Algoritmos de otimização de consulta definidos sobre álgebra (possível uso internamente no SGBD)

Algebras (baseado no levantamento no PGRD)

39



Álgebra Relacional

Símbolo	Operação	Sintaxe	Tipo
σ	Seleção / Restrição	σ condição (Relação)	Primitiva
π	Projeção	π expressões (Relação)	Primitiva
\cup	União	Relação1 \cup Relação2	Primitiva
\cap	Intersecção	Relação1 \cap Relação2	Adicional
-	Diferença de conjuntos	Relação1 - Relação2	Primitiva
\times	Produto cartesiano	Relação1 \times Relação2	Primitiva
$ x $	Junção	Relação1 x Relação2	Adicional
\div	Divisão	Relação1 \div Relação2	Adicional
ρ	Renomeação	ρ nome (Relação)	Primitiva
\leftarrow	Atribuição	variável \leftarrow Relação	Adicional

Álgebra Relacional

emprestimos

Nome_agencia	Nro_emprestimo	Total
Timbo	17	1000
Indaial	23	2000
Blumenau	15	1500
Pomerode	93	500
Gaspar	11	900
Blumenau	16	1300

devedores

Nome_Cliente	Nro_emprestimo
Jonas	17
Silvio	23
Henrique	15
Carlos	93
Silvio	11
William	17
Adalberto	16

contas

Nome_Cliente	Nro_conta
Jonas	11111
Silvio	22222
Henrique	33333
Jackson	44444

R

A	B	C
a	b	c
d	a	f
c	b	d

S

D	E	F
b	g	a
d	a	f

roskell	PPPPP
suplment	33333
QVILS	55555

c	p	q
q	s	t
r	u	v



Álgebra Relacional Operação de Seleção (SELECT)

- Seleciona de uma tabela, as linhas que satisfazem a um determinado predicado
- $\sigma_F(R)$

Ex. $\sigma_{\text{nome_agencia} = \text{"Blumenau"}}(\text{emprestimo})$

Nome_agencia	Nro_emprestimo	Total
Blumenau	15	1500
Blumenau	16	1300

A condição F pode envolver
⇒ operandos constantes
ou número de componente
(\$i)
⇒ operadores aritméticos
de comparação ($<$, $=$, $>$, \leq ,
 \neq , \geq)
⇒ operadores lógicos (\wedge ,
 \vee , \neg) (e, ou, não)

```
SELECT nome_agencia, nro_emprestimo, total
FROM emprestimo
WHERE nome_agencia = 'Blumenau'
```

WHERE nome_agencia = 'Blumenau'
FROM emprestimo
SELECT nome_agencia, nro_emprestimo, total

Álgebra Relacional Operação de Projeção (PROJECT)

- Seleciona uma lista de colunas de uma tabela
- $\Pi_{i_1, i_2, \dots, i_n}(R)$

Ex. $\Pi_{nro_emprestimo, total}(emprestimo)$

Nro_emprestimo	Total
17	1000
23	2000
15	1500
93	500
11	900
16	1300

SELECT nro_emprestimo, total
FROM emprestimo

EMPRESTIMO
SELECT nro_emprestimo, total

```
    #selection at the end -add back the deselected mirror modifier object
    mirror_ob.select= 1
    modifier_ob.select=1
    bpy.context.scene.objects.active = modifier_ob
    print("Selected" + str(modifier_ob)) # modifier ob is the active ob
    mirror_ob.select = 0
    ob = bpy.context.selected_objects[0]
    ob.modifiers.remove(modifier)
    ob.select = True
    bpy.ops.object.select_all(action='DESELECT')
    bpy.context.view_layer.objects.active = ob
    ob.select = True
```

Álgebra Relacional Operação de Projeção (PROJECT)

- Operação de projeção com comparação.

Ex. $\Pi_{\text{nro_emprestimo}}(\sigma_{\text{nome_agencia} = \text{"Blumenau"}} (\text{emprestimo}))$

Nro_emprestimo
15
16

```
SELECT nro_emprestimo  
FROM emprestimo  
WHERE nome_agencia = 'Blumenau'
```

WHERE nome_agencia = 'Blumenau'
FROM emprestimo
SELECT nro_emprestimo

Álgebra Relacional Operação Produto Cartesiano

$R \times S$

- O **resultado é uma tabela** cuja linhas são a combinação das linhas das tabelas R e S tomando-se uma linha de R e concatenando-a com uma linha de S

Peca

CodPeça	NomePeça	CorPeça	PesoPeça	CidadePeça
P1	Eixo	Cinza	10	Porto Alegre
P2	Rolamento	Preto	16	Santa Maria
P3	Mancal	Verde	30	Uruguaiana

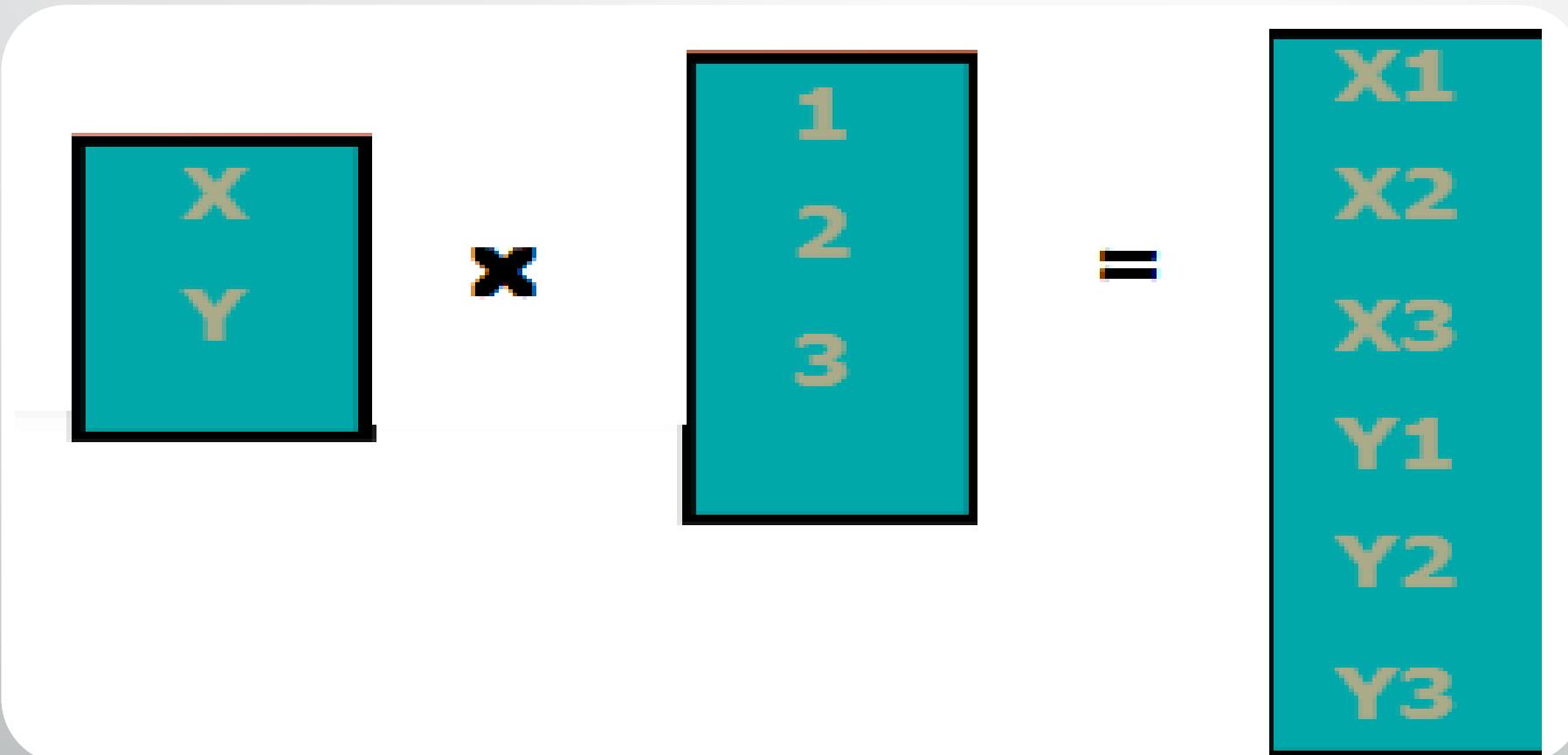
- Total de **colunas** do produto cartesiano:
 - nº colunas da primeira tabela + nº de colunas da segunda tabela
 - $6 + 3 = 9$ colunas
- Nº de **linhas** do produto cartesiano:
 - Nº de linhas da primeira tabela x nº de linhas da segunda tabela.
 - $500 \times 600 = 30.000$ linhas

Embarq

CodPeça	CodFornec	QtidEmbarq
P1	F1	300
P1	F2	400
P1	F3	200
P2	F1	300
P2	F4	350

65	E1	320
65	E2	
61	E3	
61	E4	
61	E5	

Álgebra Relacional Operação Produto Cartesiano



Exemplo de Produto Cartesiano

Obtenha os nomes de todas as peças para as quais há embarques

π NomePeça

(σ Embarq. CodPeça = Peça. CodPeça
(Embarq X Peça))

Peça	CodPeça	NomePeça	CorPeça	PesoPeça	CidadePeça
	P1	Eixo	Cinza	10	Porto Alegre
	P2	Rolamento	Preto	16	Santa Maria
	P3	Mancal	Verde	30	Uruguaiana

Embarq

Embarq	CodPeça	CodFornec	QtidEmbarq
	P1	F1	300
	P1	F2	400
	P1	F3	200
	P2	F1	300
	P2	F4	350

NomePeça
Eixo
Rolamento

Tabela
Resultante

Operação de junção (join)

A combinação de uma operação de seleção aplicada sobre uma operação de produto cartesiano é usual em aplicações de BD. É através dela que dados de tabelas relacionadas são associados.

Sintaxe:

```
<tabela1> |X| <criterio> <tabela2>
```

A junção tem como operandos duas tabelas. O resultado é equivalente a executar: $\sigma <\text{criterio}> (<\text{tabela1}> \times <\text{tabela 2}>))$

Ex. $\Pi \text{ nome_cliente, nro_emprestimo, total}$
 $(\sigma_{\text{nome_agencia} = \text{"Blumenau"}} (\text{emprestimo} \bowtie \text{devedores}))$

Nome_Cliente	Nro_emprestimo	Total
Henrique	15	1500
Adalberto	16	1300

```
SELECT nome_cliente, nro_emprestimo, total  
FROM emprestimo, devedores  
WHERE emprestimo.nro_emprestimo = devedores.nro_emprestimo  
and nome_agencia = 'Blumenau'
```

WHERE nome_agencia = 'Blumenau'
AND emprestimo.nro_emprestimo = devedores.nro_emprestimo
AND emprestimo.nro_emprestimo = devedores.nro_emprestimo

Operação de Junção (Natural)

- Operação binária que permite combinar certas relações e um produto cartesiano dentro de uma operação.

$R \bowtie S$

R			S		
A	B	C	B	C	D
a	b	c	b	c	d
d	b	c	b	c	d
b	b	f	b	c	e
c	a	d	a	d	b

A	B	C	D
a	b	c	d
a	b	c	e
d	b	c	d
d	b	c	e
c	a	d	b

Operação **implícita** = igualdade dos atributos com mesmo nome.

Cada par de atributos iguais dá origem a um único atributo, com o mesmo nome, no resultado

Operações da teoria de conjunto

- A álgebra relacional empresta da teoria de conjuntos quatro operadores: União, Interseção, Diferença e Produto Cartesiano
 - Sintaxe da **União**: $\langle\text{tabela}\rangle_1 \cup \langle\text{tabela}\rangle_2$
 - Sintaxe da **Intersecção**: $\langle\text{tabela}\rangle_1 \cap \langle\text{tabela}\rangle_2$
 - Sintaxe da **Diferença**: $\langle\text{tabela}\rangle_1 - \langle\text{tabela}\rangle_2$
 - Nos três casos, a operação possui **duas tabelas** como **operando**. E as tabelas devem ser compatíveis:
 - Possuir mesmo número de colunas;
 - o mesmo domínio para cada posição da lista de atributos;
 - quando os nomes das colunas forem diferentes, adota-se os nomes das colunas da primeira tabela.



managemen^t
database
management

Operações da teoria de conjuntos

R	S				
A	B	C	D	E	F
a	b	c	b	g	a
d	a	f	d	a	f
c	b	d			

D	E	F
b	g	a
d	a	f

R \cup S		
A	B	C
a	b	c
d	a	f
c	b	d
b	g	a

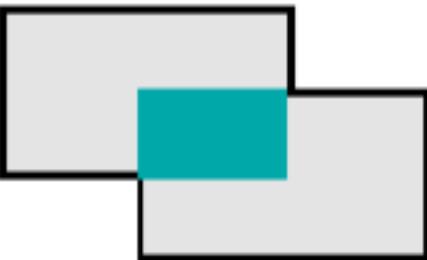
R - S		
A	B	C
a	b	c
c	b	d

R \cap S		
A	B	C
d	a	t

União



Intersecção



Diferença



Operação de União

- Operação binária que permite unir dois conjuntos ou relações de dados
- Nesta operação, os conjuntos duplicados são eliminados.

Ex. $\Pi_{\text{nome_cliente}}(\text{devedores}) \cup \Pi_{\text{nome_cliente}}(\text{contas})$

Nome_Cliente
Adalberto
Carlos
Henrique
Jackson
Jonas
Silvio
William

```
SELECT nome_cliente  
FROM devedores  
UNION  
SELECT nome_cliente  
FROM contas
```

WILLIAM
OLIVIA



Operação de Interseção

- Permite encontrar conjuntos de dados comuns entre duas relações de dados. Nesta operação, os conjuntos duplicados são eliminados.

Ex. $\Pi_{\text{nome_cliente}}(\text{devedores}) \cap \Pi_{\text{nome_cliente}}(\text{contas})$

Nome_Cliente
Henrique
Jonas
Silvio

```
SELECT nome_cliente  
      FROM devedores  
INTERSECT  
SELECT nome_cliente  
      FROM contas
```

Operação de Diferença

- Permite encontrar conjuntos de dados que estão em uma relação e que não estão em outra.

Ex. $\Pi_{\text{nome_cliente}}(\text{devedores}) - \Pi_{\text{nome_cliente}}(\text{contas})$

Nome_Cliente
Adalberto
Carlos
William

```
SELECT nome_cliente  
      FROM devedores  
MINUS  
SELECT nome_cliente  
      FROM contas
```

Operação de União

Aluno = { nome, idade, curso}

Professor = { nome, idade, depto.}

Funcionario = { nome, depto, idade}

Dom(nome) = varchar(30)
Dom(idade) = int
Dom(curso) = varchar(5)
Dom(depto) = varchar(5)

→ Aluno é compatível com Professor,
mas não é com Funcionario.

Operação de União

Apresente uma relação com todos os alunos e também com todos os professores:

Aluno = {nome, idade, curso}

{Zeca, 25, comput.
Zico, 21, eletr.
Juca, 19, odonto.
Tuca, 19, comput.}

Professor = {nome, idade, depto.}

{Ari, 35, comput.
Wilma, 32, eletr.
Zeca, 25, comput.}

Aluno U Professor = {nome, idade, curso}

{ Zeca, 25, comput.
Zico, 21, eletr.
Juca, 19, odonto.
Tuca, 19, comput.
Ari, 35, comput.
Wilma, 32, eletr. }

Tabela
Resultante

a operação de UNIÃO elimina duplicatas, mas existem SGBDs (ex.:INGRES) que permitem que o usuário determine se tuplas duplicadas devem ser eliminadas do resultado ou não.

Operação de Intersecção

Apresente uma relação de todos os alunos que **SÃO** professores

Aluno = {nome, idade, curso}
{Zeca, 25, comput.
Zico, 21, eletr.
Juca, 19, odonto.
Tuca, 19, comput.}

Professor = {nome, idade, depto.}
{Ari, 35, comput.
Wilma, 32, eletr.
Zeca, 25, comput.}

Aluno \cap Professor = {nome, idade, curso}
{ Zeca, 25, comput}

Operadores derivados

há operadores de álgebra que são deriváveis de outros. A operação de intersecção é derivável de união e diferença:

$$A \cap B = A - (A - B)$$

$$A \cup B = A - (A - B)$$

Intersecção é derivável de união e diferença:
os operadores de álgebra são deriváveis de outros.



Operação de diferença Apresente uma relação de todos os alunos que NÃO SÃO professores

Aluno = {nome, idade, curso}
{Zeca, 25, comput.
Zico, 21, eletr.
Juca, 19, odonto.
Tuca, 19, comput.}

Professor = {nome, idade, depto.}
{Ari, 35, comput.
Wilma, 32, eletr.
Zeca, 25, comput.}

Aluno - Professor = {nome, idade, curso}
{ Zico, 21, eletr.
Juca, 19, odonto.
Tuca, 19, comput.}

Note-se que a DIFERENÇA não é comutativa !

Professor - Aluno = {nome, idade, depto.}

{Ari, 35, comput.
Wilma, 32, eletr.}

Wilma, 32, eletr.
Ari, 35, comput.
Tuca, 19, comput.
Juca, 19, odonto.
Zico, 21, eletr.
Zeca, 25, comput.
Tuca, 19, comput. - ou seja - professor

Operador relacional de DIVISÃO

COL1	COL2
A	1
B	1
C	2
A	2
B	3
C	3
A	4
B	4
C	5

Tabela_A

COL2	COL1
2	C
3	

$A \div B$

Tabela_B

As linhas da Tabela A que não possuem correspondência com todas as linhas da Tabela B serão excluídas.

Em SQL

```
Select distinct col1 from Tabela_A a  
      where not exists (select null from Tabela_B b  
                        where not exists (select null  
                                          from Tabela_A  
                                          where col2=b.col2 and  
                                                a.col1=col1));
```

Operador relacional de RENOMEAR

Sintaxe

$\rho < \text{novo nome} (R)$

Exemplo

TB_ESTADO_CIVIL (estado_civil, descrição)

$\rho EA(\text{cod}, \text{desc}) (TB_ESTADO_CIVIL)$

Em SQL

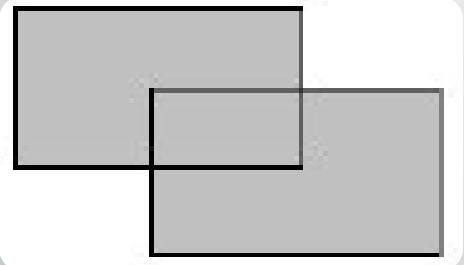
```
Select estado_civil as cod, descrição as desc  
from TB_ESTADO_CIVIL EA
```



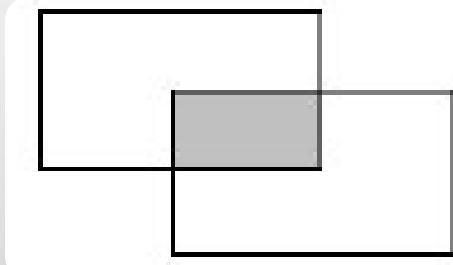
Representação Gráfica

- Operadores Tradicionais

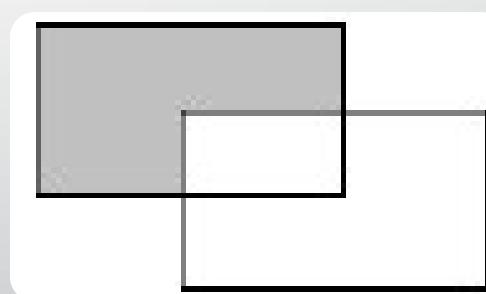
União



Intersecção

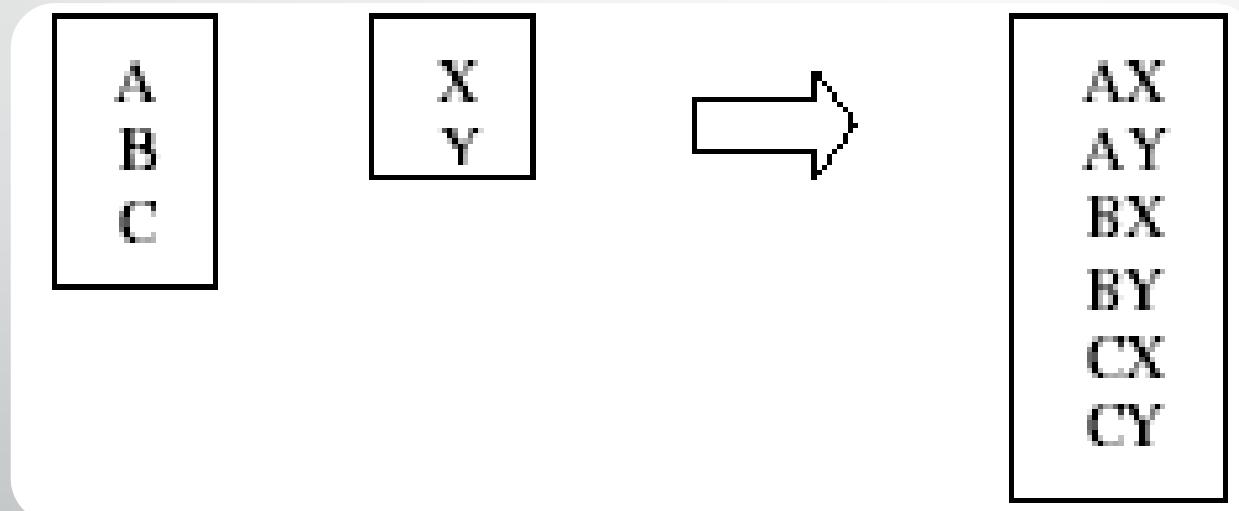


Diferença



Representação Gráfica

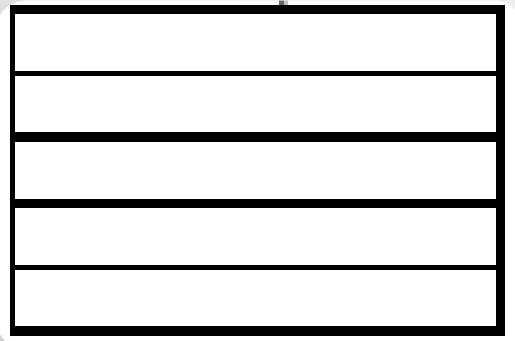
- Operadores Tradicionais
- Produto Cartesiano



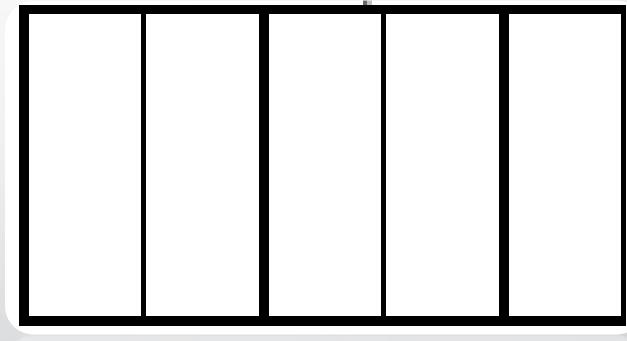
Representação Gráfica

- Operadores Relacionais

Seleção



Projeção



Simbologia

- União $R \cup S$
- Diferença $R - S$
- Prod. Cartesiano $R \times S$
- Seleção $\sigma F(R)$
- Projeção $\Pi i_1, i_2, \dots, i_m(R)$
- Junção $R \bowtie S$

FUNÇÕES DE AGREGAÇÃO (\S)

Uma função de agregação tome uma coleção de valores e retorna um único valor como resultado

avg: média

min: mínimo

max: máximo

sum: soma

count: número de valores



A álgebra relacional consiste em um conjunto de operações utilizadas para **manipular relações**.

Uma consulta em um banco de dados que

Segue o modelo relacional é realizada através da aplicação de uma serie de operações da álgebra relacional que são **executadas e retornam**.



Os dados na forma de uma tabela.

Por exemplo:

Uma consulta pode selecionar algumas tuplas de uma relação (os empregados de sexo masculino) e, adicionalmente, combinar tais tuplas com tuplas de outra relação.

- (os empregados de sexo masculino e seus respectivos dependentes).

As operações da álgebra relacional são as seguintes:

- Seleção (σ) – seleciona um subconjunto de linhas de uma relação
- Projeção (π) – apaga colunas desnecessárias de uma relação
- Produto cartesiano (\times) – permite combinar duas relações
- União ou Union (U) – tuplas na relação 1 e na relação 2
- Diferença ou Minus (-) – tuplas na relação 1 mas não na relação 2

Importante

O resultado de cada operação da álgebra relacional é **uma nova relação**, uma tabela, que pode ser manipulada por outras operações da álgebra relacional.

Assim, as operações da álgebra relacional são realizadas sobre relações inteiras (não em uma tupla (linha) da relação; o resultado dessas operações é **sempre uma nova relação, uma tabela**.

Na construção da lógica de acesso aos dados do Banco de Dados são utilizados os seguintes operadores da álgebra relacional:

- Operadores da Álgebra: PC (Produto Cartesiano), Seleção, Minus, Union e Projeção.
- Operadores Lógicos: Join, (inner e outer) e Divide

Obrigado pela atenção

Sigo à disposição pelo e-mail:

marcio.lemos@senairs.org.br