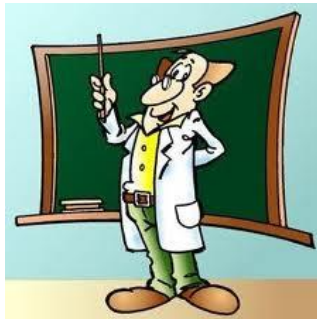




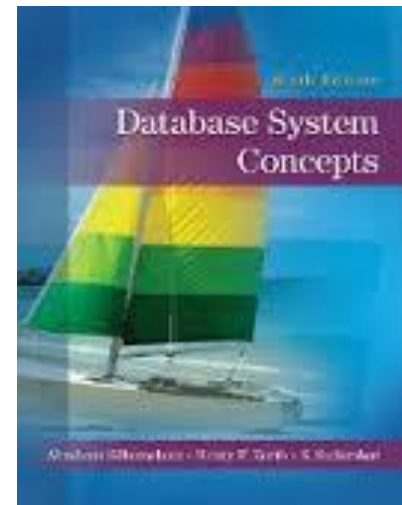
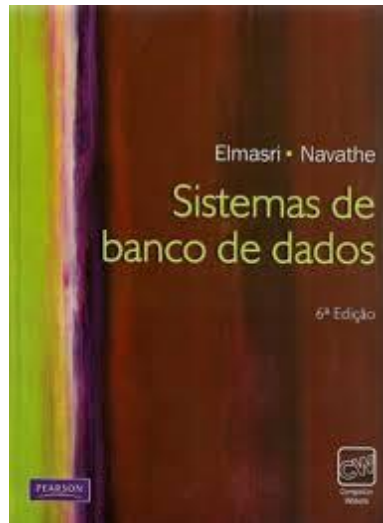
## Unidade 5 – Álgebra Relacional



Prof. Aparecido V. de Freitas  
Doutor em Engenharia  
da Computação pela EPUVSP  
[aparecidovfreitas@gmail.com](mailto:aparecidovfreitas@gmail.com)

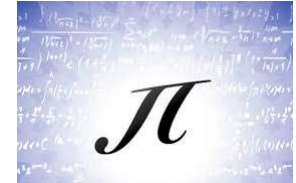


# Bibliografia





# Álgebra Relacional

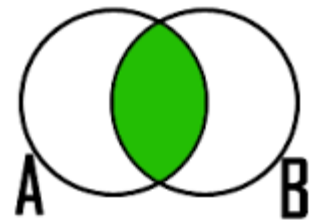


- ✓ Historicamente, a Álgebra Relacional e o Cálculo Relacional foram desenvolvidos antes da Linguagem SQL.
- ✓ SQL é baseada nos conceitos da Álgebra Relacional e do Cálculo Relacional.
- ✓ Para se capacitar na escrita de SQL complexa, necessita-se de conhecimento dos conceitos da Álgebra Relacional.



# Lembrando do Modelo Relacional...

- ✓ Um modelo de dados contém um conjunto de operações para manipular o Banco de Dados, além da definição do esquema e das restrições.
- ✓ O conjunto de operações do Modelo Relacional é a Álgebra Relacional.
- ✓ Uma sequência de operações da Álgebra Relacional forma uma Expressão da Álgebra Relacional.
- ✓ Na Álgebra Relacional vale a **Propriedade do Fechamento**: O resultado de uma expressão com relações **resulta em uma nova relação**.



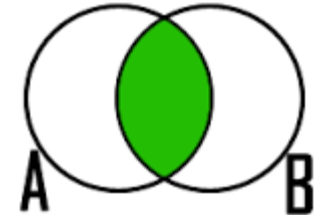


# Qual a importância da Álgebra Relacional ?





# Álgebra Relacional



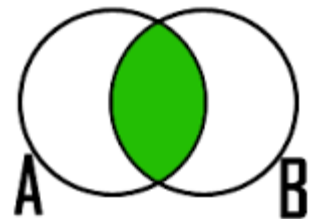
- ✓ Fornece alicerce formal para as operações do Modelo Relacional;
- ✓ É usada como base para a implementação e otimização de consultas em DBMS;
- ✓ Os conceitos da Álgebra Relacional estão incorporados na Linguagem SQL;
- ✓ Módulos internos dos DBMS são baseados em conceitos da Álgebra Relacional.





# Cálculo Relacional

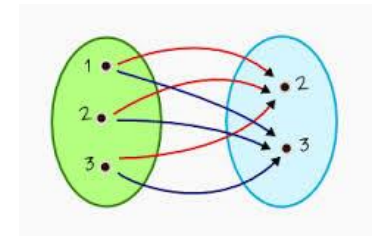
- ✓ Oferece uma **linguagem declarativa** de nível mais alto para a especificação de consultas relacionais.
- ✓ No Cálculo Relacional não existe ordem de operações para se especificar a forma pela qual será recuperado o resultado de uma consulta. (SQL é baseado no Cálculo Relacional de Tupla)
- ✓ Esse é o principal fator que distingue a Álgebra Relacional do Cálculo Relacional.





# Operações da Álgebra Relacional

- ✓ Operações de **Teoria dos Conjuntos**: (UNIÃO, INTERSECÇÃO, DIFERENÇA e PRODUTO CARTESIANO)
- ✓ Operações para **Banco de Dados Relacionais**: SELEÇÃO, PROJEÇÃO e JUNÇÃO.



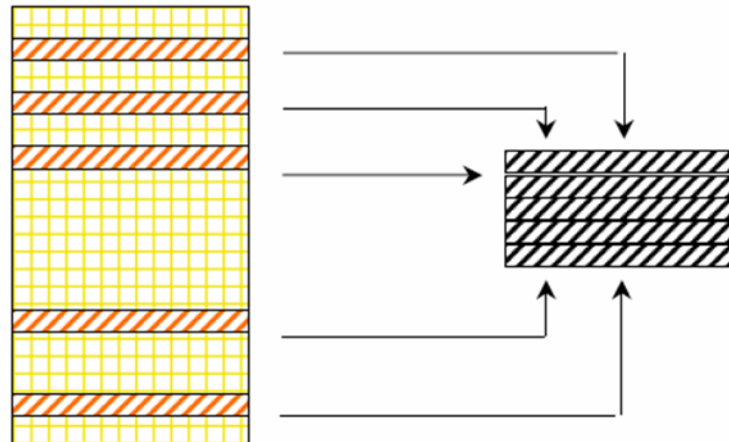




# Operações Relacionais Unárias

## SELEÇÃO ( $\sigma$ )

- ✓ A operação SELEÇÃO é usada para escolher um **subconjunto** das tuplas de uma relação que satisfaça a uma condição de SELEÇÃO.
- ✓ Pode-se considerar a operação de SELEÇÃO como um **filtro** que mantém apenas as tuplas que satisfazem a uma condição qualificadora.
- ✓ Pode ser vista como uma **partição horizontal** no conjunto de tuplas.





# SELEÇÃO ( $\sigma$ )

- ✓ O símbolo  $\sigma$  (sigma) é usado para indicar a operação de SELEÇÃO.
- ✓ Exemplo: Seleção das tuplas da relação FUNCIONARIO do departamento 345.

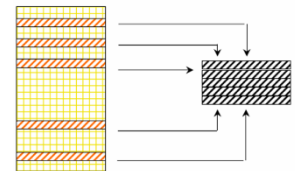
$\sigma$  ( FUNCIONARIO )

DEPTO=345

- ✓ Exemplo: Seleção das tuplas da relação FUNCIONARIO com salário superior a \$15.000.

$\sigma$  ( FUNCIONARIO )

SALARIO>15.000





# SELEÇÃO ( $\sigma$ )

- ✓ De forma geral, na Álgebra Relacional a operação de SELEÇÃO é indicada por:

$$\sigma_{\langle \text{Condição de Seleção} \rangle} (R)$$

- ✓ **R** é uma expressão da Álgebra Relacional, cujo resultado é uma Relação. A mais simples expressão desse tipo é uma Relação de Banco de Dados. A relação resultante tem os mesmos atributos de **R**.
- ✓ A Condição de Seleção é uma expressão **booleana** especificada nos **atributos de R**.



# SELEÇÃO ( $\sigma$ )

- ✓ Buscar os dados dos *empregados* que estão com salário menor que 2.000,00

$\sigma_{\text{salario} < 2000}$  (Empregado)

## Empregado

codEmp	Nome	Salario	idade	codDep
200	Pedro	3.000,00	45	001
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001
203	Ana	1.800,00	25	002

## Resultado

codEmp	Nome	Salario	idade	codDep
203	Ana	1.800,00	25	002



# Cláusulas da Condição de Seleção

 $\sigma$  $(R)$ 

&lt;Condição de Seleção&gt;

A condição de Seleção pode ser especificada por:



<nome do atributo> <operador de comparação> <valor constante> OU  
<nome do atributo> <operador de comparação> <nome do atributo>

- ✓ **Nome do atributo**: é um atributo de R;
- ✓ **Operador de comparação**: =, <, <=, >, >=, <>
- ✓ **Valor constante**: é um valor do domínio do atributo
- ✓ Podem ser ligadas pelos operadores **AND**, **OR** e **NOT**



# SELEÇÃO ( $\sigma$ )

- ✓ Buscar os dados dos empregados com salario maior que 2000 e com menos 45 anos

$\sigma_{\text{salario} > 2000 \text{ AND } \text{idade} < 45}$  (Empregado)

## Empregado

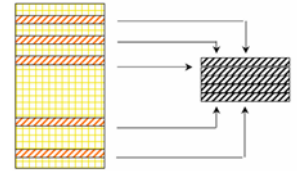
codEmp	Nome	Salario	idade	codDep
200	Pedro	3.000,00	45	001
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001
203	Ana	1.800,00	25	002

## Resultado

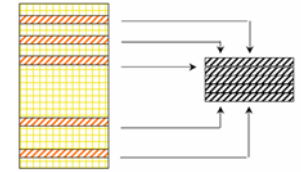
codEmp	Nome	Salario	idade	codDep
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001



# SELEÇÃO – Observações



- ✓ A condição de SELEÇÃO é aplicada independentemente para cada tupla individualmente em R.
- ✓ Se a condição for avaliada como **TRUE**, então a tupla é **selecionada**.
- ✓ O operador SELEÇÃO é **unário**; ou seja, aplicado a uma **única** RELAÇÃO.
- ✓ O **grau** da relação resultante (seu número de atributos) é o mesmo que o grau de R.
- ✓ O número de tuplas na relação resultante é sempre **menor ou igual** ao número de tuplas em R, ou seja,  $|\sigma_c(R)| \leq |R|$  para qualquer condição C.
- ✓ A fração de tuplas selecionada por uma condição é conhecida como **SELETIVIDADE** da condição.
- ✓ Em SQL, a condição SELEÇÃO normalmente é especificada na cláusula **WHERE**.



# SELEÇÃO – Propriedades

✓ A operação SELEÇÃO é comutativa.

$$\sigma_{\langle \text{condicao1} \rangle} \left( \sigma_{\langle \text{condicao2} \rangle} (R) \right) = \sigma_{\langle \text{condicao2} \rangle} \left( \sigma_{\langle \text{condicao1} \rangle} (R) \right)$$

✓ Portanto, uma sequência de SELEÇÃO pode ser aplicada em qualquer ordem.





# SELEÇÃO – Propriedade Comutativa

$$\sigma_{\text{salario} > 2000} \quad ( \quad \sigma_{\text{idade} < 45} \quad ( \text{Empregado} ) \quad )$$

## Empregado

codEmp	Nome	Salario	idade	codDep
200	Pedro	3.000,00	45	001
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001
203	Ana	1.800,00	25	002

Resultado Parcial: condição: idade < 45

codEmp	Nome	Salario	idade	codDep
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001
203	Ana	1.800,00	25	002

Resultado: condição: salario > 2000

codEmp	Nome	Salario	idade	codDep
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001



# SELEÇÃO – Propriedade Comutativa

$$\sigma_{\text{idade} < 45} \text{ ( } \sigma_{\text{salario} > 2000 \text{ ( Empregado) )}$$

## Empregado

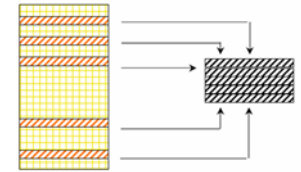
codEmp	Nome	Salario	idade	codDep
200	Pedro	3.000,00	45	001
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001
203	Ana	1.800,00	25	002

Resultado Parcial: condição: salario > 2000

codEmp	Nome	Salario	idade	codDep
200	Pedro	3.000,00	45	001
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001

Resultado: condição: idade < 45

codEmp	Nome	Salario	idade	codDep
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001



## SELEÇÃO – Propriedade CASCATA

- ✓ Uma sequência de operadores **SELEÇÃO** pode ser reduzida a uma única operação **SELEÇÃO** por meio de uma condição conjuntiva **AND**.

$$\sigma_{\langle \text{condicao1} \rangle} \left( \sigma_{\langle \text{condicao2} \rangle} (R) \right) = \sigma_{\langle \text{condicao2} \rangle \text{ AND } \langle \text{condicao1} \rangle} (R)$$

- ✓ Exemplo:

equivalentes

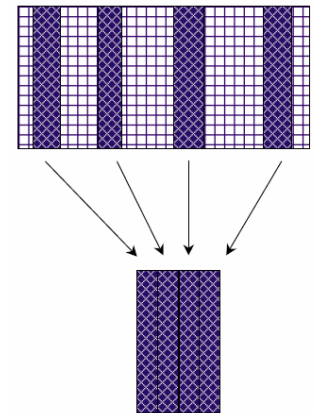
$$\sigma_{\text{idade} < 45} \left( \sigma_{\text{salario} > 2000} (\text{Empregado}) \right)$$

$$\sigma_{\text{salario} > 2000 \text{ AND } \text{idade} < 45} (\text{Empregado})$$



# Projeção ( $\pi$ )

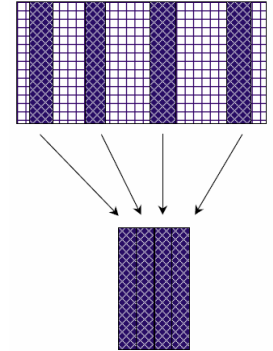
- ✓ A operação PROJEÇÃO considera certas colunas da tabela.
- ✓ O resultado é uma relação que contém apenas os atributos de interesse.
- ✓ Pode ser vista como uma **partição vertical** no conjunto de tuplas.





# Sintaxe – Projeção ( $\pi$ )

$\pi$  <lista de atributos> (<R>)



onde:

- ✓ <lista de atributos> é uma sublista desejada de atributos da relação R.
- ✓ <R> é o nome da relação ou uma expressão da álgebra relacional de onde a lista de atributos será considerada.



# PROJEÇÃO – Exemplo

- ✓ Buscar o nome e a idade de todos os empregados;

$$\pi_{\text{nome, idade}}(\text{Empregado})$$

## Empregado

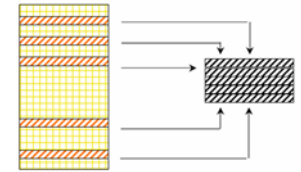
codEmp	Nome	Salario	idade	codDep
200	Pedro	3.000,00	45	001
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001
203	Ana	1.800,00	25	002

## Resultado

Nome	idade
Pedro	45
Paulo	43
Maria	38
Ana	25



# PROJEÇÃO – Propriedades



$\pi$  <lista de atributos> (<R>)

- ✓ R, em geral, é uma expressão da Álgebra Relacional cujo resultado é uma relação (**Propriedade do Fechamento**). No caso, mais simples, R é apenas o nome de uma relação do Banco de Dados.
- ✓ O resultado da operação PROJEÇÃO tem apenas os **mesmos atributos** especificados na <lista de atributos> e na **mesma ordem** em que eles aparecem na lista. Logo, seu grau é igual ao número de atributos especificados em <lista de atributos>.
- ✓ Se a <lista de atributos> inclui apenas atributos não chave de R, tuplas duplicadas provavelmente ocorrerão. A operação PROJEÇÃO remove quaisquer tuplas duplicadas, de modo que o resultado da operação é um conjunto de tuplas distintas.



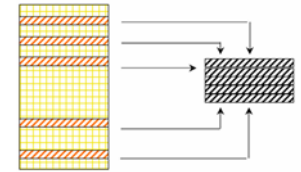
# Porque a operação de PROJEÇÃO elimina tuplas duplicadas ?







# PROJEÇÃO – Duplicadas



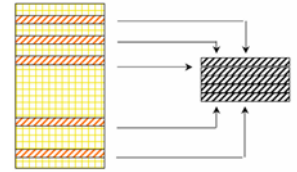
$\pi$  <lista de atributos> (<R>)

- ✓ Se as tuplas duplicadas não fossem eliminadas, o resultado seria um multiconjunto de tuplas, em vez de um conjunto. Isso não é permitido no modelo relacional.
- ✓ A eliminação de duplicadas envolve classificação ou alguma outra técnica algorítmica para detectar duplicadas, o que – naturalmente – aumenta o processamento da operação.





## PROJEÇÃO – Observações



$$\pi \langle \text{lista de atributos} \rangle (\langle R \rangle)$$

- ✓ O número de tuplas em uma relação resultante de uma operação de PROJEÇÃO é sempre menor ou igual ao número de tuplas em R.
- ✓ Se a lista de atributos é uma superchave de R – ou seja, inclui alguma chave de R –, a relação resultante tem o mesmo número de tuplas de R.



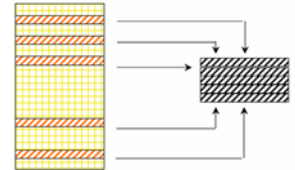
Vale a propriedade comutativa na  
operação de PROJEÇÃO ?





# PROJEÇÃO – Propriedade Comutativa

✓ A operação PROJEÇÃO **NÃO** é comutativa .



$$\pi_{\langle \text{lista1} \rangle} \left( \pi_{\langle \text{lista2} \rangle} (R) \right) \neq \pi_{\langle \text{lista2} \rangle} \left( \pi_{\langle \text{lista1} \rangle} (R) \right)$$

✓ Exemplo

$$\pi_{\langle \text{cpf} \rangle} \left( \pi_{\langle \text{nome, cpf} \rangle} (\text{Empregado}) \right) \neq \pi_{\langle \text{nome, cpf} \rangle} \left( \pi_{\langle \text{cpf} \rangle} (\text{Empregado}) \right)$$

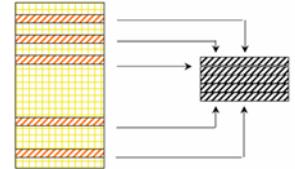


Expressão **incorreta**, pois a lista do lado direito não contém todos os atributos do lado esquerdo....



# PROJEÇÃO – Equivalência ao SQL

- ✓ Em SQL, a lista de atributos de PROJEÇÃO é especificada na cláusula SELEÇÃO de uma consulta.
- ✓ Caso se remova a palavra chave DISTINCT, então as duplicadas não serão removidas, o que não é permitido no modelo relacional.
- ✓ Exemplo


$$\pi_{\langle \text{sexo}, \text{salario} \rangle} (\text{Funcionario})$$

- ✓ Equivale em SQL a:

```
SELECT DISTINCT SEXO, SALARIO FROM FUNCIONARIO;
```



# SELEÇÃO e PROJEÇÃO

- ✓ Operadores diferentes podem ser aninhados
- ✓ Exemplo: Buscar o nome e o salario dos empregados com mais de 40 anos

$$\pi_{\text{nome, salario}} (\sigma_{\text{idade} > 40} (\text{Empregado}))$$

## Empregado

codEmp	Nome	Salario	idade	codDep
200	Pedro	3.000,00	45	001
201	Paulo	2.200,00	43	001
202	Maria	2.500,00	38	001
203	Ana	1.800,00	25	002

## Resultado

Nome	Salario
Pedro	3.000,00
Paulo	2.200,00



## Sequências de Operações e a operação RENAME

- ✓ As relações obtidas nas operação vistas anteriormente não possuem nome;
- ✓ É comum aplicar-se várias operações relacionais em sequência para se obter uma consulta, por exemplo: Uma SELEÇÃO seguida de uma PROJEÇÃO.
- ✓ Pode-se aplicar uma operação de cada vez e criar-se uma relação intermediária.

### ✓ Expressão em linha:

$$\pi_{Pnome, Unome, Salario}(\sigma_{Dnr=5}(FUNCIONARIO))$$

### ✓ Sequência de operações:

$$\begin{aligned} FUNCS\_DEPT5 &\leftarrow \sigma_{Dnr=5}(FUNCIONARIO) \\ RESULTADO &\leftarrow \pi_{Pnome, Unome, Salario}(FUNCS\_DEP5) \end{aligned}$$



# Operação de Rename

- ✓ Se nenhuma renomeação for aplicada, os nomes dos atributos na relação resultante das operações SELEÇÃO e PROJEÇÃO permanecem inalterados.
- ✓ A Álgebra Relacional define um operador formal, chamado  $\rho$  que pode renomear o nome da relação ou os nomes dos atributos, ou ambos.

$$\rho_{S(B_1, B_2, \dots, B_n)}(R) \quad \text{OU} \quad \rho_S(R) \quad \text{OU} \quad \rho_{(B_1, B_2, \dots, B_n)}(R)$$

Renomeia tanto a  
relação quanto os  
atributos (na ordem)

Renomeia somente  
a relação

Renomeia somente os  
atributos (na ordem)





## Operação de Rename em SQL

- ✓ A renomeação em SQL pode ser feita por meio de ALIAS (apelidos) usando a palavra chave AS.

```
SELECT      F.Pnome AS Primeiro_nome,  
            F.Unome AS Ultimo_nome,  
            F.Salario AS Salario
```

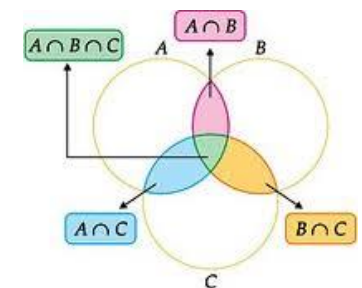
```
FROM FUNCIONARIO AS F
```

```
WHERE  F.Dnr = 5 ;
```



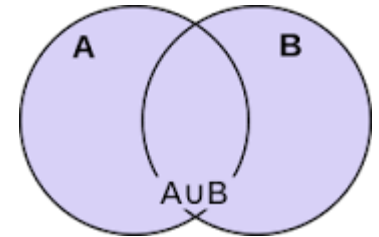
# Operações com base na Teoria dos Conjuntos

- ✓ Operações **UNIÃO**, **INTERSECÇÃO** e **DIFERENÇA** mesclam os elementos de DOIS conjuntos;
- ✓ São operações **binárias**, ou seja, aplicada à dois conjuntos de dados;
- ✓ No modelo relacional, as duas relações precisam ter o mesmo tipo de tuplas. Essa condição é chamada **COMPATIBILIDADE DE TIPO**.
- ✓ Duas relações  $R(A_1, A_2, \dots, A_n)$  e  $S(B_1, B_2, \dots, B_n)$  são compatíveis de tipo se tiverem o mesmo grau  $n$  e se  $\text{dom}(A_i) = \text{dom}(B_i)$ , para  $1 \leq i \leq n$ . Isso significa que as duas relações têm o mesmo número de atributos e cada par correspondente de atributos tem o mesmo domínio.





# Operação de UNIÃO

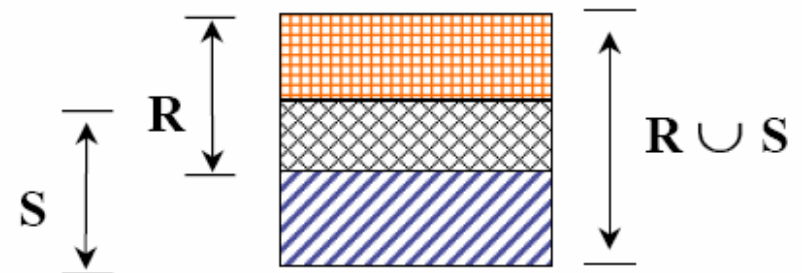


- ✓  $R \cup S$ ;
- ✓ Inclui todas as tuplas que estão em R ou em S ou tanto em R quanto em S;
- ✓ Tuplas duplicadas são eliminadas.

x	y	z
1	1	1
1	2	2
2	2	3
3	1	1

x	y	z
1	1	1
1	2	1
1	2	3

x	y	z
1	1	1
1	2	1
1	2	2
1	2	3
2	2	3
3	1	1

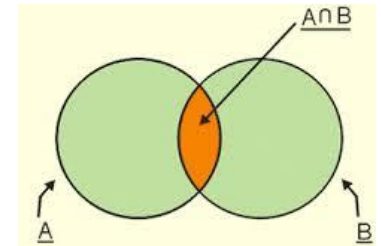
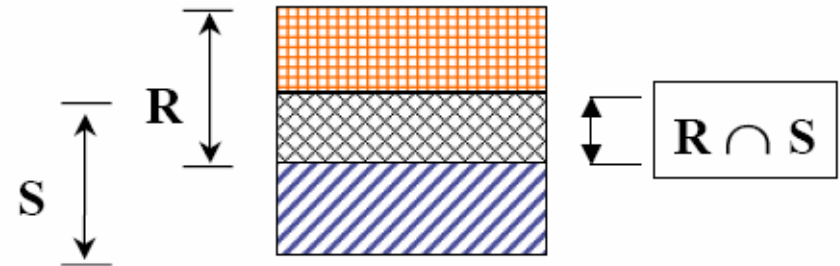




# Operação de INTERSECÇÃO

- ✓  $R \cap S$ ;
- ✓ Inclui todas as tuplas que estão tanto em R quanto em S.

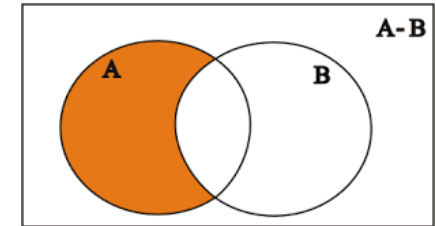
R			S			$R \cap S$		
x	y	z	x	y	z	x	y	z
1	1	1	1	1	1	1	1	1
1	2	2	1	2	1	1	2	1
2	2	3	3	1	1	3	1	1
3	1	1						





# Operação de SUBTRAÇÃO (DIFERENÇA)

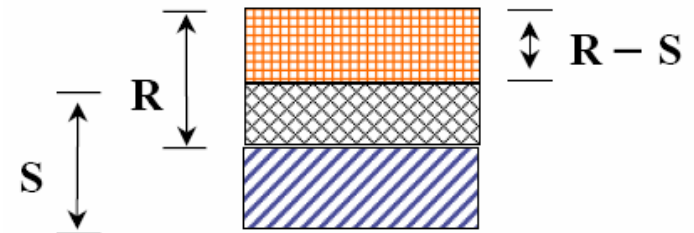
- ✓  $R - S$ ;
- ✓ Inclui todas as tuplas que estão em R, mas não em S.



x	y	z
1	1	1
1	2	2
2	2	3
3	1	1

x	y	z
1	1	1
1	2	1
3	1	1

x	y	z
1	2	2
2	2	3

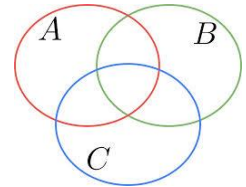




# Operações com Conjuntos – Observações

- ✓ Tanto **UNIÃO** quanto **INTERSECÇÃO** são operações comutativas:  

$$R \cup S = S \cup R \quad \text{e} \quad R \cap S = S \cap R;$$



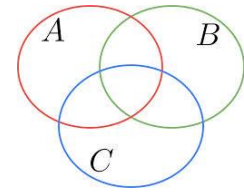
- ✓ Tanto **UNIÃO** quanto **INTERSECÇÃO** podem ser tratadas como operações n-árias, aplicáveis a qualquer número de relações, pois ambas são associativas:  

$$R \cup (S \cup T) = (R \cup S) \cup T \quad \text{e} \quad R \cap (S \cap T) = (R \cap S) \cap T$$

- ✓ A operação **SUBTRAÇÃO** **não** é comutativa. Em geral:  $R - S \neq S - R$

- ✓ A operação **INTERSECÇÃO** pode ser expressa em termos de **UNIÃO** e **DIFERENÇA**:  

$$R \cap S = ((R \cup S) - (R - S)) - (S - R)$$



# Operações com Conjuntos em SQL

- ✓ Em SQL, existem 3 operações – **UNION**, **INTERSECT** e **EXCEPT** que correspondem às operações de conjunto UNIÃO, INTERSECÇÃO e SUBTRAÇÃO.
- ✓ Além disso, existem operações de multiconjunto (**UNION ALL**, **INTERSECT ALL** e **EXCEPT ALL**) que **não** eliminam tuplas duplicadas.

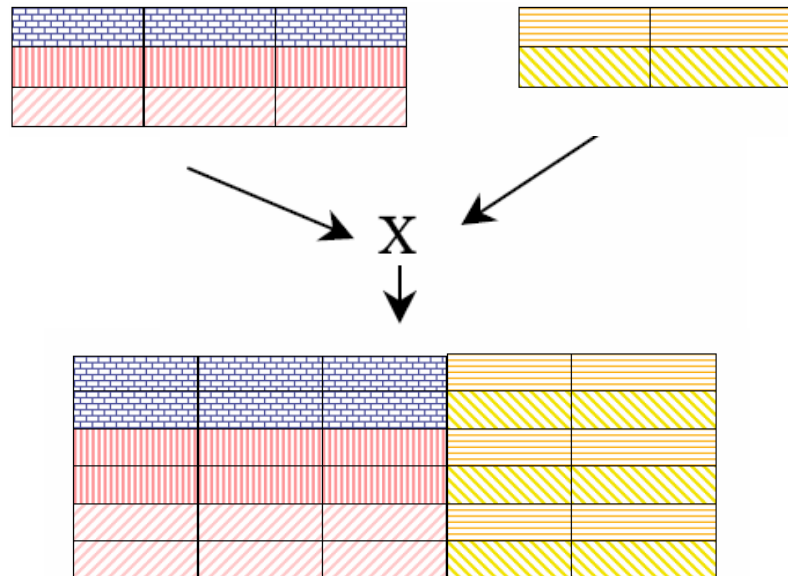




# Operações PRODUTO CARTESIANO

*Produto Cartesiano*  
 $A \times B = \{(x, y) | x \in A \text{ e } y \in B\}$   
*Par ordenado*

- ✓ Também conhecida por **PRODUTO CRUZADO** ou **JUNÇÃO CRUZADA**;
- ✓ Indicada por **X** ;
- ✓ Também é uma operação binária, mas as relações sobre as quais ela é aplicada não precisam ser compatíveis na união;
- ✓ Produz um novo elemento combinando cada membro (tupla) de uma relação (conjunto) com cada membro (tupla) de outra relação (conjunto).







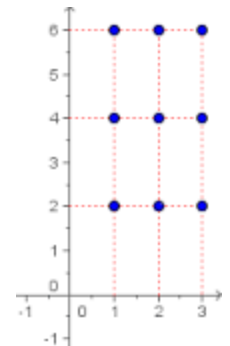
Produto  
Cartesiano

$$A \times B = \{(x, y) | x \in A \text{ e } y \in B\}$$

Par ordenado

## PRODUTO CARTESIANO – Observações

- ✓ O resultado de  $R (A_1, A_2, \dots, A_n) \times S (B_1, B_2, \dots, B_m)$  é uma relação  $Q$  com grau  $n + m$  atributos  $Q (A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)$  nessa ordem.
- ✓ Logo, se  $R$  tem  $n_r$  tuplas (indicado por  $|R| = n_r$ ) e  $S$  tem  $m_s$  tuplas, então  $R \times S$  terá  $n_r * m_s$  tuplas.
- ✓ Com frequência, a operação **PRODUTO CARTESIANO**, aplicada isoladamente, não tem significado prático.



Join

## Operação JUNÇÃO

- ✓ Indicada por  $\bowtie$ .
- ✓ Permite processar relacionamentos (tuplas relacionadas) entre duas relações.
- ✓ O resultado da JUNÇÃO é uma relação Q com  $n + m$  atributos Q ( $A_1, A_2, \dots, A_n, B_1, B_2, \dots B_m$ ) nessa ordem, que tem uma tupla para cada combinação de tuplas - uma de R e outra de S – sempre que a combinação satisfaz a condição de junção.
- ✓ Essa é a principal diferença entre **PRODUTO CARTESIANO** e **JUNÇÃO**.
- ✓ Em **JUNÇÃO** apenas combinações de tuplas que satisfazem a **condição** de junção aparecem no resultado, enquanto que no **PRODUTO CARTESIANO** todas as combinações de tuplas são incluídas no resultado.





# Exemplo JUNÇÃO ⋈

Join

- ✓ Necessita-se recuperar o **nome do Gerente** de cada **Departamento**;

DEPARTAMENTO			FUNCIONARIO	
PK			PK	
<u>idDepto</u>	NomeDepto	idGerente	<u>idFunc</u>	NomeFunc
10	COMPRAS	1000	1000	Paulo de Souza Alves
20	ENGENHARIA	1200	1100	José da Silva
40	VENDAS	2000	1200	Pedro Rangel de Souza
55	FINANCEIRO	5000	1450	Angela Silva Medeiros
			2000	Alceu de Almeida
			3000	Saulo de Araujo
			3400	Pedro Silva
			5000	Roberto Boschetti
			5400	Ana de Souza Almeida



# Exemplo JUNÇÃO $\bowtie$

Join

✓ Aplicando-se a operação de **JUNÇÃO** com a condição **idGerente = idFunc**

**DEPARTAMENTO**

<u>idDepto</u>	NomeDepto	idGerente
----------------	-----------	-----------

**FUNCIONARIO**

<u>idFunc</u>	NomeFunc
---------------	----------

**A** ← **DEPARTAMENTO**  $\bowtie_{\text{idGerente = idFunc}}$  **FUNCIONARIO**



**A**

<u>idDepto</u>	NomeDepto	idGerente	idFunc	NomeFunc
10	COMPRAS	1000	1000	Paulo de Souza Alves
20	ENGENHARIA	1200	1200	Pedro Rangel de Souza
40	VENDAS	2000	2000	Alceu de Almeida
55	FINANCEIRO	5000	5000	Roberto Boschetti



## Exemplo JUNÇÃO $\bowtie$

Join

- ✓ Aplicando-se a operação de **PROJEÇÃO** na relação intermediária

**A**

<u>idDepto</u>	NomeDepto	idGerente	<u>idFunc</u>	NomeFunc
----------------	-----------	-----------	---------------	----------

RESULTADO  $\leftarrow \pi$  NomeDepto, NomeFunc (A)



**RESULTADO**

NomeDepto	NomeFunc
COMPRAS	Paulo de Souza Alves
ENGENHARIA	Pedro Rangel de Souza
VENDAS	Alceu de Almeida
FINANCEIRO	Roberto Boschetti



## Exemplo JUNÇÃO



- ✓ Renomeando-se os atributos da relação RESULTADO

### RESULTADO

NomeDepto	NomeFunc
-----------	----------

$\rho$  (NomeDepto, NomeGerente) **(RESULTADO)**



### RESULTADO

NomeDepto	NomeGerente
COMPRAS	Paulo de Souza Alves
ENGENHARIA	Pedro Rangel de Souza
VENDAS	Alceu de Almeida
FINANCEIRO	Roberto Boschetti



## JUNÇÃO – Observações

- ✓ Quando a condição de junção for **TRUE**, a combinação de **TUPLA** é incluída na relação resultante;
- ✓ Uma condição geral tem a forma: **<condição> AND <condição> .... AND <condição>**
- ✓ As condições podem ter os operadores de comparação { =, <, >, <=, >=, ≠ } ;
- ✓ Uma JUNÇÃO com essa condição geral é chamada **JUNÇÃO THETA**;
- ✓ Na **JUNÇÃO THETA**, as informações das relações originais não são preservadas, uma vez que as tuplas combinadas que resultam em **FALSE** não aparecem no resultado.



## EQUIJOIN

- ✓ O uso mais comum de **JUNÇÃO** envolve condições de junção em apenas comparações de **IGUALDADE**;
- ✓ Esse tipo de **JUNÇÃO**, em que se usa somente o operador de comparação **=**, é chamado **EQUIJUNÇÃO** ou **EQUIJOIN**;
- ✓ Na **EQUIJUNÇÃO** sempre se tem um ou mais pares de atributos com **VALORES IDÊNTICOS**.







## EQUIJOIN – Exemplo

A ← DEPARTAMENTO ⋈ **idGerente = idFunc** FUNCIONARIO

**A**

<u>idDepto</u>	NomeDepto	idGerente	idFunc	NomeFunc
10	COMPRAS	1000	1000	Paulo de Souza Alves
20	ENGENHARIA	1200	1200	Pedro Rangel de Souza
40	VENDAS	2000	2000	Alceu de Almeida
55	FINANCEIRO	5000	5000	Roberto Boschetti



## JUNÇÃO NATURAL

- ✓ Corresponde a uma EQUIJUNÇÃO (condição de junção de igualdade) no qual **elimina-se** o segundo atributo (desnecessário) uma vez que possuem valores idênticos.
- ✓ A definição padrão de **JUNÇÃO NATURAL** requer que os **dois atributos** de junção tenham o **MESMO NOME** nas duas relações. Se isto não ocorrer, deve-se aplicar uma operação de renomeação antes da operação de junção.
- ✓ A **JUNÇÃO NATURAL** é indicada por um **\***.



Join



# JUNÇÃO NATURAL – Exemplo

- ✓ Suponha que se queira combinar cada tupla de **PROJETO** com uma tupla de **DEPARTAMENTO** que controla um projeto.

## DEPARTAMENTO

<b>PK</b>	
<u>idDepto</u>	NomeDepto
10	COMPRAS
20	ENGENHARIA
40	VENDAS
55	FINANCEIRO
90	RH

<b>PK</b>		<b>FK</b>
<u>idProj</u>	NomeProj	DeptoResponsavel
1000	Controle de Pedidos	10
1200	Projeto ABX	20
2000	Projeto Vendas Otimizadas	40
3000	Saulo de Araujo	55

- ✓ Nesse exemplo, as tuplas a serem combinadas devem ser relacionadas pelos atributos **idDepto** na relação de **DEPARTAMENTO** e **DeptoResponsavel** na relação **PROJETO**.

Join

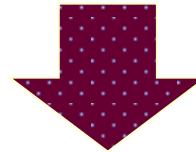


# JUNÇÃO NATURAL – Exemplo

- ✓ Renome-se o nome de um dos atributos, para deixá-los com mesmo nome.

<u>idProj</u>	NomeProj	DeptoResponsavel
---------------	----------	------------------

$\rho$  (idProj, NomeProj, idDepto) **(PROJETO)**



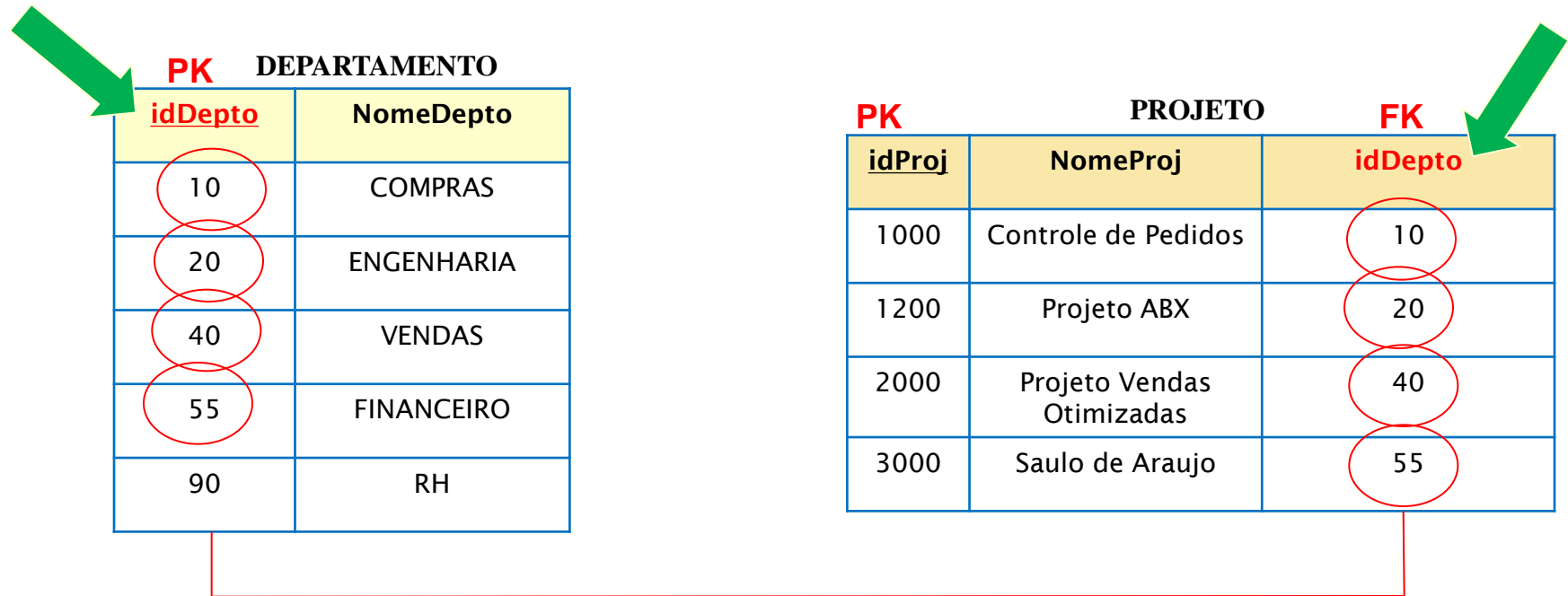
<b>PK</b>	<b>PROJETO</b>	<b>FK</b>
<u>idProj</u>	NomeProj	idDepto
1000	Controle de Pedidos	10
1200	Projeto ABX	20
2000	Projeto Vendas Otimizadas	40
3000	Saulo de Araujo	55

Join



## JUNÇÃO NATURAL - Exemplo

- ✓ As relações agora ficam com o atributo de join com mesmo nome (**idDeppto**) nas relações de **DEPARTAMENTO** e **PROJETO**.



Join



## JUNÇÃO NATURAL – Exemplo

- ✓ Aplica-se o **JOIN NATURAL**, com o atributo de junção **idDeppto**.
- ✓ Somente um valor de atributo de junção será mantido na relação resultando.

DEPARTAMENTO		PROJETO		
PK				FK
	<u>idDeppto</u>		NomeProj	<u>idDeppto</u>
	10		Controle de Pedidos	10
	20		Projeto ABX	20
	40		Projeto Vendas Otimizadas	40
	55		Saulo de Araujo	55
	90			

RESULTADO ← DEPARTAMENTO \*<sub>(idDeppto)</sub> PROJETO

<u>idDeppto</u>	NomeDeppto	<u>idProj</u>	NomeProj
10	COMPRAS	1000	Controle de Pedidos
20	ENGENHARIA	1200	Projeto ABX
40	VENDAS	2000	Projeto Vendas Otimizadas
55	FINANCEIRO	3000	Saulo de Araujo



# JUNÇÃO NATURAL

- ✓ Uma definição mais geral, para **JUNÇÃO NATURAL** é:

$$Q \leftarrow R *_{(<lista1>),( <lista2>)} S$$

- ✓ Nesse caso, **<lista1>** especifica uma lista de **i** atributos de **R** e **<lista2>** uma lista de **i** atributos de **S**. As listas são usadas para formar condições de comparação de igualdade entre pares de atributos correspondentes. (As condições passam por um **AND**).

Join



## JUNÇÃO – Observações

- ✓ Se nenhuma combinação de tuplas satisfazer a condição de junção, o resultado de uma **JUNÇÃO** é uma relação **VAZIA**;
- ✓ Se não houver condição de junção, todas as combinações de tuplas se qualificam e a **JUNÇÃO** se degenera em um **PRODUTO CARTESIANO**, chamado **PRODUTO CRUZADO** ou **JUNÇÃO CRUZADA**;
- ✓ Considerando que somente as linhas que atenderem a condição de junção serão selecionadas, essa operação de **JUNÇÃO** é denominada **INNER JOIN**.



Join





## JUNÇÃO NATURAL – Observações

- ✓ As operações com JUNÇÃO **NATURAIS** e **EQUIJUNÇÃO** são também conhecidas por **JUNÇÕES INTERNAS** (inner joins).
- ✓ São chamadas de **inners joins** para diferenciá-las de uma variação de JUNÇÃO diferente conhecida por **outer joins**.
- ✓ Informalmente, uma junção interna é um tipo de operação definida como uma combinação de PRODUTO CARTESIANO e SELEÇÃO.

Join




## OUTER JOIN (Junção Externa)

- ✓ Corresponde a uma extensão da operação **JUNÇÃO**;
- ✓ As operações de **JUNÇÃO** vistas até aqui combinam com tuplas que satisfazem a condição de junção. Ou seja, as tuplas **sem** uma tupla correspondente (ou relacionada) são **eliminadas** do resultado de **JUNÇÃO**;
- ✓ As tuplas com valores **NULL** nos atributos de junção também são eliminadas;
- ✓ **Junções externas (outer joins)** são junções especiais desenvolvidas para o caso em que o usuário deseja manter todas as tuplas em **R**, ou todas em **S**, ou todas aquelas nas duas relações no resultado da Junção, independentemente delas possuírem ou não tuplas correspondentes na outra relação.

Join



## LEFT OUTER JOIN

- ✓ Indicada por 
- ✓ Tuplas correspondentes em **R** e **S** são selecionadas;
- ✓ Tuplas de **R** sem correspondente em **S**, são também selecionadas e os atributos de **S** são preenchidos com o valor **NULL**.

OUTER LEFT JOIN.s...Testes (sa (232))\*

```
SELECT F.nomeFuncionario  
      , C.nomeCargo  
FROM   FUNCIONARIO AS F  
       LEFT OUTER JOIN CARGO AS C ON ( F.codCargo = C.codCargo )
```

Results Messages


	nomeFuncionario	nomeCargo
1	JOÃO	CAIXA
2	MARIA	VENDEDOR
3	CARLOS	CAIXA
4	TADEU	NULL



Join



## RIGHT OUTER JOIN

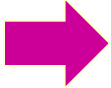
- ✓ Indicada por 
- ✓ Tuplas correspondentes em **R** e **S** são selecionadas;
- ✓ Tuplas de **S** sem correspondente em **R**, são também selecionadas e os atributos de **R** são preenchidos com o valor **NULL**.

OUTER RIGHT JOIN...Testes (sa (232))

```
SELECT F.nomeFuncionario  
      ,C.nomeCargo  
FROM   FUNCIONARIO AS F  
       RIGHT OUTER JOIN CARGO AS C ON ( F.codCargo = C.codCargo )
```

Results Messages


	nomeFuncionario	nomeCargo
1	JOÃO	CAIXA
2	CARLOS	CAIXA
3	MARIA	VENDEDOR
4	NULL	GERENTE



Join



# FULL OUTER JOIN

- ✓ Indicada por 
- ✓ **Tuplas** correspondentes em **R** e **S** são selecionadas;
- ✓ Mantém todas as tuplas nas relações da **esquerda** e da **direita** quando nenhuma tupla correspondente for encontrada, preenchendo com valores **NULL** conforme a necessidade.

OUTER FULL JOIN.s...Testes (sa (232))\*

```
SELECT F.nomeFuncionario
      ,C.nomeCargo
FROM      FUNCIONARIO AS F
FULL OUTER JOIN CARGO      AS C ON ( F.codCargo = C.codCargo )
```

Results Messages

	nomeFuncionario	nomeCargo
1	JOÃO	CAIXA
2	MARIA	VENDEDOR
3	CARLOS	CAIXA
4	TADEU	NULL
5	NULL	GERENTE

Join