

Bases de Dados

Álgebra Relacional
– Partes I e 2

Profa. Elaine Parros Machado de Sousa



“Existem muitas formas de debugar um sql. Da contagem até a fé. Mas depois que passa deste último nível, só falta álgebra relacional. E é incrível como ela não falha.”

Marcus Araujo - BCC 2015



Álgebra Relacional

- **Álgebra Relacional**
 - linguagem de consulta **procedural**
- **Operação da Álgebra Relacional** ⇒ definida sobre uma ou mais relações
 - resultado ⇒ relação
- **Expressão da Álgebra Relacional** ⇒ sequência de operações
- **Consulta** ⇒ expressa como uma **expressão da álgebra relacional**



Álgebra Relacional

- **OBS:** na perspectiva **algébrica**, uma relação é um elemento imutável e atômico
 - AR **não tem** operações de definição de relações ou de inclusão/modificação/remoção de tuplas
 - AR permite apenas definição de consultas



Operações da Álgebra Relacional

- 3 grupos principais:
 - 1) Operações Relacionais Unárias
 - Seleção
 - Projeção
 - 2) Operações Relacionais Binárias
 - Junção
 - Divisão
 -



Operações da Álgebra Relacional

- ...
- **3) Operações sobre Conjuntos**
 - União
 - União Exclusiva
 - Interseção
 - Diferença
 - Produto Cartesiano
- Além disso...
 - Operação de *Assignment*
 - Operação de *Rename*



Operações Relacionais Unárias

- **Operações Unárias da Álgebra Relacional**
⇒ levam em conta a **estrutura** das relações
- Basicamente 2 operações:
 - Seleção (σ)
 - Projeção (π)

Seleção (σ)

Aluno = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zico, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

- Exemplo:
 - “*Selecione os dados dos alunos que fazem odontologia*”

select * from Aluno where Curso = ‘odontologia’ **SQL**

$\sigma_{(\text{curso} = \text{'odontologia'})} \text{ Aluno}$

Álgebra Relacional

Resultado:

{<Juca, 21, odontologia>}



Operações Relacionais Unárias

- **Seleção - $\sigma_{(\text{condição})} R$**
 - resultado: relação que contém o subconjunto das tuplas de R que satisfazem à condição de seleção *<condição>*
 - condição de seleção: **operação de comparação** de um atributo da relação com:
 - uma constante
 - com outro atributo da própria relação ⇒ comparação de valores de atributos da mesma tupla



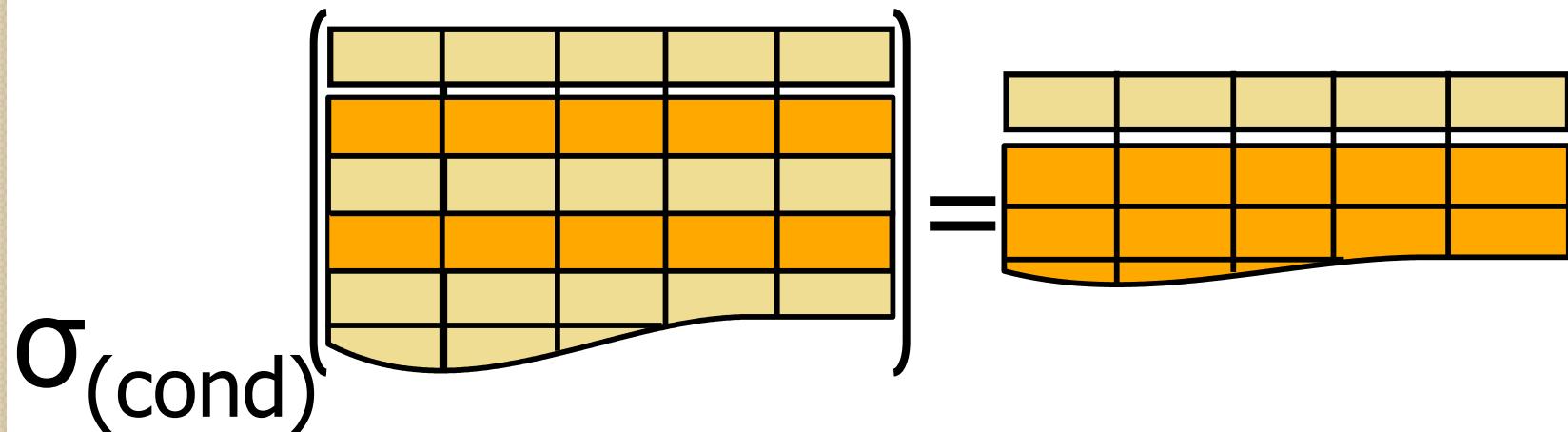
Operações Relacionais Unárias

- Seleção pode combinar várias condições concatenadas por operadores lógicos **AND** e **OR**
 - Ex:

$\sigma_{((\text{curso} = \text{'odontologia'}) \text{ OR } (\text{idade} < 25))} \text{ Aluno}$

Seleção

- Seleção \Rightarrow **particionamento horizontal**
 - escolha de algumas “linhas” da tabela





Seleção

- Operador Seleção é **Comutativo**
 - $\sigma_{(\text{condição A})}(\sigma_{(\text{condição B})}R) = \sigma_{(\text{condição B})}(\sigma_{(\text{condição A})} R)$
- Uma sequência de seleções pode ser executada em qualquer ordem, ou pode ser transformada numa única seleção
 - $\sigma_{(\text{condição 1})}(\sigma_{(\text{condição 2})}(\dots(\sigma_{(\text{condição n})}(R))))$
 - $\sigma_{((\text{condição 1}) \text{ AND } (\text{condição 2}) \text{ AND } \dots \text{ AND } (\text{condição n}))}(R)$



Seleção

- Operador **Seleção**
 - aplicado a cada tupla da relação R
 - (**grau** de $\sigma_{(\text{condição})}(R)$) = (**grau** de R)
 - **seletividade** da condição de seleção:
fração de tuplas selecionadas
 - $| \sigma_{(\text{condição})}(R) | \leq | R |$

Projeção(π)

Aluno = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zico, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

- **Consulta:** Seleciona nome e idade dos Alunos

SQL

```
select Nome, Idade from Aluno
```

AR

$\pi_{(\text{Nome}, \text{Idade})} \text{ Aluno}$



{<Zeca, 25>,
<Zico, 18>,
<Juca, 21>,
<Tuca, 18> }



Operações Relacionais Unárias

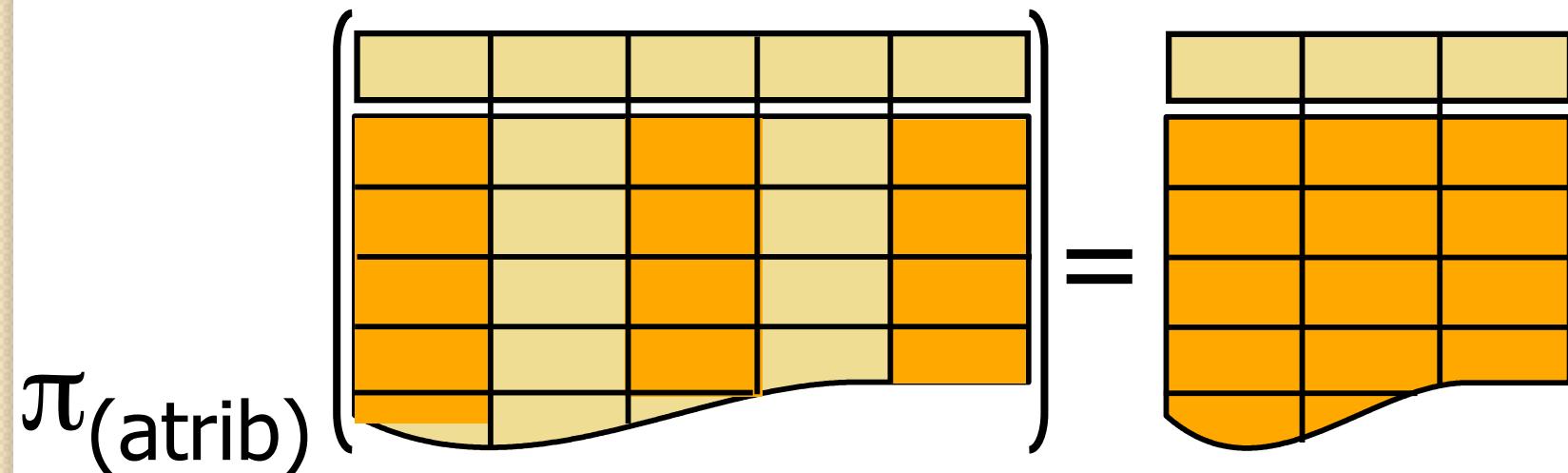
- **Projeção** - $\pi_{(\text{atributos})} R$
 - resultado: **relação** que tem apenas os atributos indicados na lista de **<atributos>**
 - **<atributos>**: subconjunto do conjunto de atributos da relação

Projeção

- O resultado de uma operação de projeção é uma relação \Rightarrow **não há tuplas repetidas**
 - Por definição, tuplas **repetidas são eliminadas**
 - se lista de *<atributos>* contém chave da relação \Rightarrow resultado naturalmente não tem tuplas repetidas
 - se lista de *<atributos>* não contém chave \Rightarrow tuplas repetidas são eliminadas, se for o caso

Projeção

- Projeção \Rightarrow **particionamento vertical**
 - escolha de algumas “Colunas” da tabela





Projeção

- Operador de Projeção
 - não é Comutativo
 - se $\langle lista \ B \rangle$ contém $\langle lista \ A \rangle$, então vale a igualdade:
 - $\pi_{\langle lista \ A \rangle}(\pi_{\langle lista \ B \rangle}(R)) = \pi_{\langle lista \ A \rangle}(R)$
 - (**grau** de $\pi_{\langle lista \rangle}(R)$) = $|\langle lista \rangle|$
 - $|\pi_{\langle lista \rangle}(R)| \leq |R|$

Exemplo

Matricula= {NomeA, Disciplina, Nota}

{<Zeca, SCC-125, 8.5>,
<Zeca, SCC-148, 8.0>,
<Zeca, SCC-180, 7.5>,
<Zico, SCC-148, 5.2>,
<Juca, SCC-125, 6.0>,
<Juca, SCC-148, 7.0>}

“Listar as notas que cada aluno tirou na disciplina SCC-125”

$\pi_{(\text{nome}, \text{nota})}(\sigma_{(\text{disciplina} = \text{'SCC-125'})}(\text{Matricula}))$

{<Zeca, SCC-125, 8.5>,
<Juca, SCC-125, 6.0>}

{<**Zeca, 8.5**>,
<**Juca, 6.0**>}



Operações da Álgebra Relacional

- ***Assignment* (\leftarrow):**

- Dois modos de utilização

- 1) Atribuir um nome a uma relação que armazena resultados intermediários de uma expressão algébrica

- **Nome \leftarrow Expressão Algébrica Relacional**

$\text{Aluno} = \{\underline{\text{Nome}}, \text{Idade}, \text{Curso}\}$

Odonto $\leftarrow \sigma_{(\text{curso} = \text{'odontologia'})} \text{Aluno}$

Resultado $\leftarrow \pi_{(\text{Nome}, \text{Idade})} \text{Odonto}$



Operações da Álgebra Relacional

- **Assignment (\leftarrow) :**

2) Renomear os atributos numa relação intermediária ou resultante de uma sequência de operações

- **NomeRelação (A₁, A₂, ...) \leftarrow Expressão**

Aluno = {Nome, Idade, Curso}

Odonto (Aluno, Idade) \leftarrow $\pi_{(\text{Nome, Idade})} \sigma_{(\text{curso} = \text{'odontologia'})}$ Aluno



Operações da Álgebra Relacional

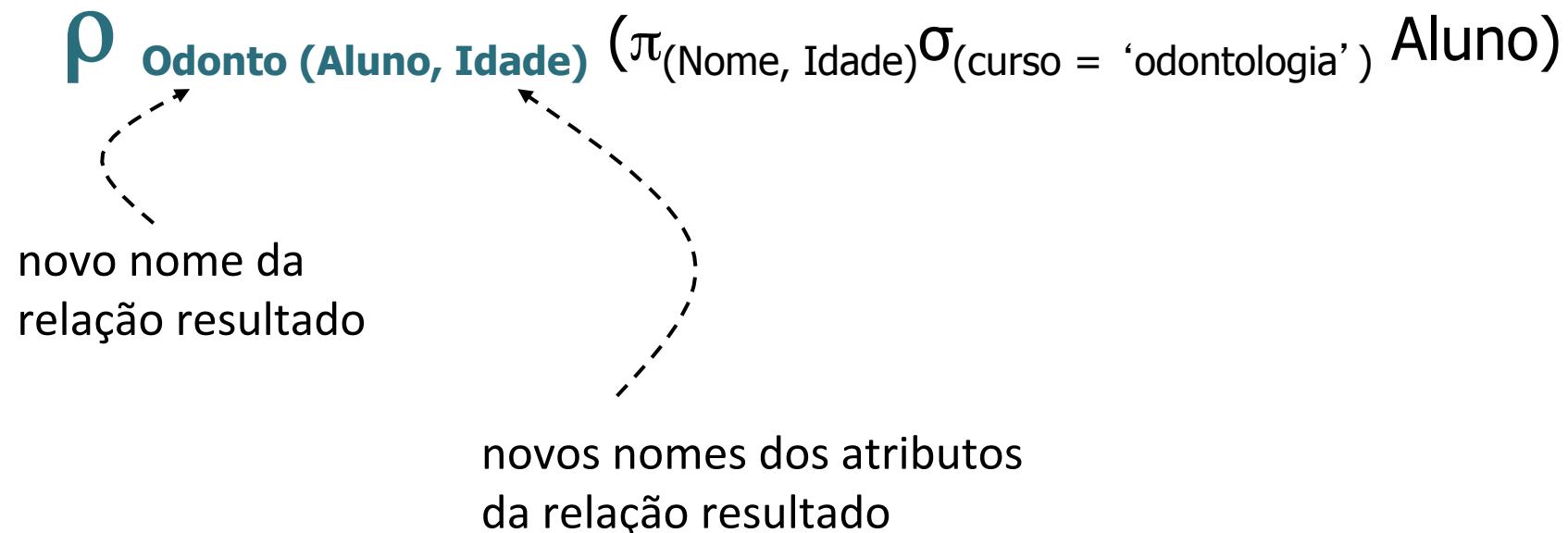
- **RENAME** (ρ)

- permite renomear uma relação ou os atributos de uma relação
- dada uma relação $R(A_1, A_2, \dots, A_n)$:
 - $\rho_{s(B1, B2, \dots, Bn)}(R)$ – renomeia atributos e relação
 - $\rho_s(R)$ – renomeia somente relação
 - $\rho_{(B1, B2, \dots, Bn)}(R)$ – renomeia somente atributos

Operações da Álgebra Relacional

- **RENAME** (ρ)

Aluno = {Nome, Idade, Curso}



Exercício

Pesquise se há como fazer
em SQL as operações de
Assignment e Rename



Operações Relacionais Binárias

- Junção
- Divisão

Exemplo

```
Aluno = {Nome, Idade, Curso}  
{<Zeca, 25, computação>,  
<Zico, 18, eletrônica>,  
<Juca, 21, odontologia>,  
<Tuca, 18, computação> }
```

```
Matricula= {NomeA, Disciplina, Nota}  
{<Zeca, SCC-125, 8.5>,  
<Zico, SCC-148, 5.2>,  
<Juca, SCC-125, 6.0>,  
<Juca, SCC-148, 7.0>}
```

“Listar as disciplinas em que os alunos de computação se matricularam”

Em SQL

```
select M.Disciplina  
from Aluno A join Matricula M  
on A.Nome = M.NomeA  
where A.Curso = 'computacao'
```



Junção

Aluno = {Nome, Idade, Curso}

Matricula= {NomeA, Disciplina, Nota}

R \bowtie (*condição da junção*) **S**

“Listar as disciplinas em que os alunos de computação se matricularam”

$\pi_{(\text{Disciplina})} (\sigma_{(\text{Curso} = \text{“computação”})} (\text{Matricula} \bowtie \text{Aluno}))$
 $\quad \quad \quad (\text{NomeA} = \text{Nome})$



Junção

- **R \bowtie (condição da junção) S**
 - condição da Junção:
 - $<\text{condição}> \text{ AND } <\text{condição}> \text{ AND } \dots <\text{condição}>$
 - $<\text{condição}>$: comparação entre atributos, ou conjunto de atributos:
 - **Atrib_R θ Atrib_S**
 - Atrib_R - atributo da relação R
 - Atrib_S - atributo da relação S
 - Atrib_R e Atrib_S são **atributos de junção** – **compatibilidade de domínio**
 - θ - **operador de comparação** válido no domínio desses atributos

Junção

- $Q \leftarrow R \boxtimes_{(condição\ de\ junção)} S$
 - resultado: Q tem uma tupla para cada combinação das tuplas de R com as tuplas de S que satisfaz a condição de junção
 - $|Q| \leq |R \times S|$
 - (grau de Q) = (grau de R) + (grau de S)
 - tuplas de R que não têm correspondente em S **não estão** no resultado (e vice-versa)
 - tuplas com valores nulos para atributos de junção **não estão** no resultado

Junção

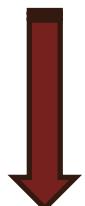
Aluno = {Nome, Idade, Curso}

Matricula= {NomeA, Disciplina, Nota}

“Listar as disciplinas em que os alunos de computação se matricularam”

$$\pi_{(\text{Disciplina})} (\sigma_{(\text{Curso} = \text{“computação”})} (\text{Matricula} \bowtie \text{Aluno})) \\ (\text{NomeA} = \text{Nome})$$

É possível melhorar a eficiência da consulta?



SIM!!!!

$$\pi_{(\text{Disciplina})} (\text{Matricula} \bowtie (\sigma_{(\text{Curso} = \text{“computação”})} \text{Aluno})) \\ (\text{NomeA} = \text{Nome})$$



Tipos de Junção

- Junções Internas (*inner joins*)
 - junção theta
 - equi-junção
 - junção natural
- Junções externas (*outer joins*)
 - *left outer join*
 - *right outer join*
 - *full outer join*



Junções Internas

- Junção Theta (θ -join)
 - θ é **qualquer operador** válido no domínio dos atributos de junção
 - atributos de junção das duas relações aparecem na relação resultado
 - variação mais genérica

Exemplo: Junção-θ

Aluno = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zico, 18, eletrônica>,
<Juca, 21, odontologia>,
→ <**Tuca**,18, computação> }

Matricula= {NomeA, Disciplina, Nota}

{<Zeca, SCC-125, 8.5>,
<Zico, SCC-148, 5.2>,
<Juca, SCC-125, 6.0>,
<Juca, SCC-148, 7.0 >}

Aluno \bowtie Matricula
(Nome = NomeA)

{**Nome**, Idade, Curso, **NomeA**, Disciplina, Nota}
{<**Zeca**, 25, computação, **Zeca**, SCC-125, 8.5>,
<**Zico**, 18, eletrônica, **Zico**, SCC-148, 5.2>,
<**Juca**, 21, odontologia, **Juca**, SCC-125, 6.0>,
<**Juca**, 21, odontologia, **Juca**, SCC-148, 7.0 >}

Exemplo: Junção-θ

Aluno = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zico, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

Matricula= {NomeA, Disciplina, Nota}

{<Zeca, SCC-125, 8.5>,
<Zico, SCC-148, 5.2>,
<Juca, SCC-125, 6.0>,
<Juca, SCC-148, 7.0 >}

Aluno \bowtie Matricula
(Nome \bowtie NomeA)

{**Nome**, Idade, Curso, **NomeA**, Disciplina, Nota}
{<Zeca, 25, computação, Zico, SCC-148, 5.2>,
<Zeca, 25, computação, Juca, SCC-125, 6.0>,
<Zeca, 25, computação, Juca, SCC-148, 7.0 >,
<.....>}

Junções Internas

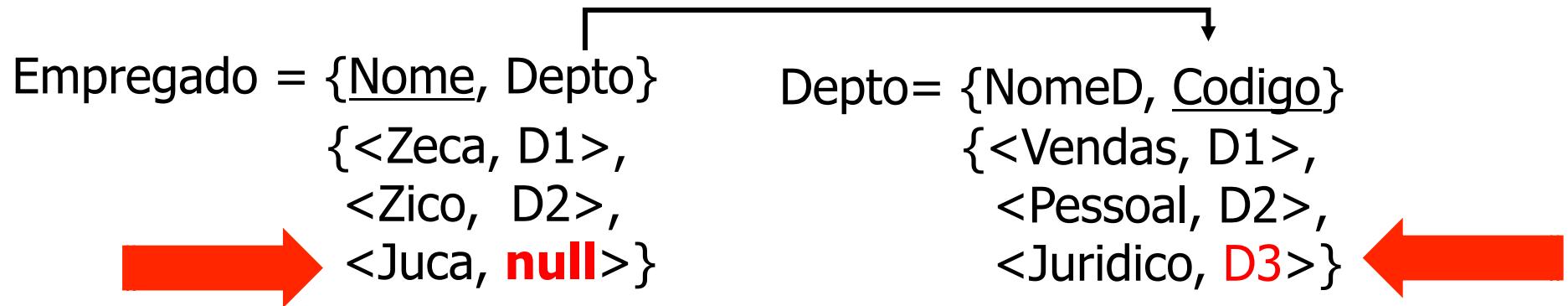
- **Equi-Junção (*Equi-join*)**
 - θ é um operador de **igualdade**
 - os atributos de junção das duas relações aparecem na relação resultado

Duas maneiras de representar *equi-join*:

$R \bowtie_{(AtribR = AtribS)} S$ ou

$R \bowtie_{(AtribR, AtribS)} S$

Exemplo: Equi-Junção



Empregado \bowtie Depto
(Dept, Codigo)

{Nome, **Dept**, NomeD, **Codigo**}
{<Zeca, **D1**, Vendas, **D1D2**, Pessoal, **D2**>}

Junções Internas

- **Junção Natural - R^*S**
 - semelhante à Equi-Junção
 - apenas os atributos de junção de uma das relações aparecem na relação resultado
 - requer que os **atributos de junção** tenham **nomes iguais** nas duas relações, ou que sejam renomeados
- 

Exemplo: Junção Natural

Empregado = {Nome, Depto}
{<Zeca, D1>,
<Zico, D2>,
<Juca, null>}

Departamento= {NomeD, Codigo}
{<Vendas, D1>,
<Pessoal, D2>,
<Juridico, D3>}

Empregado * $\rho_{(\text{NomeD}, \text{Dept})}$ Departamento

{Nome, **Dept**, NomeD}
{<Zeca, D1, Vendas>,
<Zico, D2, Pessoal>}



Junções Internas

- Resumindo... 3 tipos de Junção Interna (*inner joins*)
 - Junção - $\theta \rightarrow R \bowtie_{(\text{condição de junção})} S$
 - Equi-Junção $\rightarrow R \bowtie_{(\text{Atrib}_R, \text{Atrib}_S)} S$
 - Junção Natural $\rightarrow R * S$

Em SQL?

Junções Externas

- Junções externas (*outer joins*)
 - *Left Outer Join*
 - *Right Outer Join*
 - *Full Outer Join*

Ideia geral: retornam...

tuplas da junção interna

+

tuplas de uma tabela que não tenham
correspondência com tuplas da outra
tabela envolvida na junção



Junções Externas

- ***Left Outer Join*** – $R \bowtie_{\text{(condição de junção)}} S$
 - resultado:
 - tuplas que atendem à condição de junção
 - + - tuplas de R que não têm correspondentes em S

Exemplo: *Left Join*

Aluno = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zico, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

Matricula= {NomeA, Disciplina, Nota}

{<Zeca, SCC-125, 8.5>,
<Zico, SCC-148, 5.2>,
<Juca, SCC-125, 6.0>,
<Juca, SCC-148, 7.0 >}

“Selecionar as informações de todos os alunos e, para os que estão matriculados, os códigos e notas das disciplinas que cursam.”

Aluno \bowtie Matricula
(Nome = NomeA)

{Nome, Idade, Curso, NomeA, Disciplina, Nota}

{<Zeca, 25, computação, Zeca,
<Zico, 18, eletrônica, Zico,
<Juca, 21, odontologia, Juca,
<Juca, 21, odontologia, Juca,
<Tuca, 18, computação, null, null, null >}}

SCC-125, 8.5>,
SCC-148, 5.2>,
SCC-125, 6.0>,
SCC-148, 7.0 >},

junção
interna

junção
externa 43

Exemplo: *Left Join*

Aluno = {Nome, Idade, Curso}

```
{<Zeca, 25, computação>,
 <Zico, 18, eletrônica>,
 <Juca, 21, odontologia>,
 <Tuca, 18, computação> }
```

Matricula= {NomeA, Disciplina, Nota}

```
{<Zeca, SCC-125, 8.5>,
 <Zico, SCC-148, 5.2>,
 <Juca, SCC-125, 6.0>,
 <Juca, SCC-148, 7.0 >}
```

“Selecionar as informações de todos os alunos e, para os que estão matriculados, os códigos e notas das disciplinas que cursam.”

Aluno \bowtie Matricula
(Nome = NomeA)

{**Nome, Idade, Curso, NomeA, Disciplina, Nota**}

```
{<Zeca, 25, computação, Zeca, SCC-125, 8.5>,
 <Zico, 18, eletrônica, Zico, SCC-148, 5.2>,
 <Juca, 21, odontologia, Juca, SCC-125, 6.0>,
 <Juca, 21, odontologia, Juca, SCC-148, 7.0 >,
 <Tuca, 18, computação, null, null, null >}
```

E se inverter
as tabelas?

Exemplo: *Left Join*

Aluno = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zico, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

Matricula= {NomeA, Disciplina, Nota}

{<Zeca, SCC-125, 8.5>,
<Zico, SCC-148, 5.2>,
<Juca, SCC-125, 6.0>,
<Juca, SCC-148, 7.0 >}

“Selecionar as informações de todos os alunos e, para os que estão matriculados, os códigos e notas das disciplinas que cursam.”

Aluno \bowtie Matricula
(Nome = NomeA)

{Nome, Idade, Curso, NomeA, Disciplina, Nota}

{<Zeca, 25, computação, Zeca, SCC-125, 8.5>,
<Zico, 18, eletrônica, Zico, SCC-148, 5.2>,
<Juca, 21, odontologia, Juca, SCC-125, 6.0>,
<Juca, 21, odontologia, Juca, SCC-148, 7.0 >,
<Tuca, 18, computação, null, null, null >}

Em SQL?

Junções Externas

- ***Right Outer Join*** – $R \bowtie_{\text{(condição de junção)}} S$
 - resultado:
 - tuplas que atendem à condição de junção
 - + - tuplas de S que não têm correspondentes em R

Exemplo: *Right Join*

Empregado = {Nome, Depto}
{<Zeca, D1>,
<Zico, D2>,
<Juca, *null*>}

Departamento= {NomeD, Codigo}
{<Vendas, D1>,
<Pessoal, D2>,
<**Juridico**, D3>}

“Selecionar as informações de todos os departamentos e, se houver, dos empregados que trabalham neles.”

Empregado \bowtie Departamento
(Dept = Codigo)

{Nome, Depto, NomeD, Código}
{<Zeca, D1, Vendas, D1>,
<Zico, D2, Pessoal, D2>,
<**null**, **null**, **Juridico**, **D3**>}

Em SQL?

Junções Externas

- ***Full Outer Join*** – $R \bowtie_{\text{(condição de junção)}} S$
 - resultado:
 - tuplas que atendem à condição de junção
 - +
 - tuplas de R que não têm correspondentes em S
 - +
 - tuplas de S que não têm correspondentes em R

Exemplo: *Full Join*

Empregado = {Nome, Depto}
{<Zeca, D1>,
<Zico, D2>,
<**Juca**, *null* > }

Departamento= {NomeD, Codigo}
{<Vendas, D1>,
<Pessoal, D2>,
<**Juridico**, **D3**>}

Empregado \bowtie Departamento
(Dept = Codigo)

{Nome, Depto, NomeD, Código}
{<Zeca, D1, Vendas, D1>,
<Zico, D2, Pessoal, D2>,
<**Juca**, *null*, *null*, *null*>,
<*null*, *null*, **Juridico**, **D3**>}

Em SQL?

Exercício

Aluno = {Nome, Nusp, DataNasc}

Disciplina = {Sigla, Nome, NCred}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

- Em AR: selecione Nome, NUSP de todos os alunos e, para os que estiverem matriculados em alguma disciplina do SCC, sigla da disciplina, número da turma e quantidade de alunos na turma.
 - faça 2 versões e compare a eficiência de cada uma

Operações Relacionais Binárias

- Junção
- Divisão



Exemplo:

Matricula= {NomeA, Disciplina, Nota}

{<Zeca, SCC-125, 8.5>,
<Zeca, SCC-148, 8.0>,
<Zeca, SCC-180, 7.5>,
<Zico, SCC-148, 5.2>,
<Juca, SCC-125, 6.0>,
<Juca, SCC-148, 7.0>}

Aulas = {NomeP, Disciplina}

{<João, SCC-125>,
< João, SCC-148>,
<Eva, SCC-180>}

“Quais alunos cursam
todas as disciplinas
ministradas pelo Prof.
João? “

Exemplo (cont.)

①

Selecionar as disciplinas ministradas por João

Aulas = {NomeP, Disciplina}
{<João, SCC-125>,
<João, SCC-148>,
<Eva, SCC-180>}

“Quais alunos cursam
todas as disciplinas
ministradas pelo Prof.
João? “

$S \leftarrow \pi_{\{Disciplina\}}(\sigma_{(NomeP = "João")}(Aulas))$

$S = \{Disciplina\}$
{<SCC-125>,
<SCC-148>}

Exemplo (cont.)

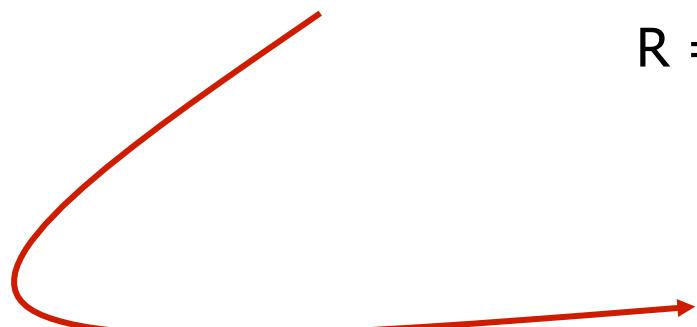
② Selecionar as disciplinas cursadas por cada aluno

Matricula= {NomeA, Disciplina, Nota}

{<Zeca, SCC-125, 8.5>,
<Zeca, SCC-148, 8.0>,
<Zeca, SCC-180, 7.5>,
<Zico, SCC-148, 5.2>,
<Juca, SCC-125, 6.0>,
<Juca, SCC-148, 7.0>}

“Quais alunos cursam
todas as disciplinas
ministradas pelo Prof.
João? “

$R \leftarrow \pi_{\{\text{NomeA}, \text{Disciplina}\}}(\text{Matricula})$



$R = \{\text{NomeA}, \text{Disciplina}\}$
{<Zeca, SCC-125>,
<Zeca, SCC-148>,
<Zeca, SCC-180>,
<Zico, SCC-148>,
<Juca, SCC-125>,
<Juca, SCC-148>}

Exemplo (cont.)

3 Aplicar operação de divisão

T ← R ÷ S

Exemplo (cont.)

$$(\pi_{\{\text{NomeA}, \text{Disciplina}\}}(\text{Matricula})) \div (\pi_{\{\text{Disciplina}\}}(\sigma_{(\text{NomeP} = "João")})(\text{Aulas}))$$

R = {NomeA, Disciplina}

{<Zeca, **SCC-125<Zeca, **SCC-148**>,
<Zeca, SCC-180>,
<Zico, **SCC-148**>,
<Juca, **SCC-125**>,
<Juca, **SCC-148**>}**

S = {Disciplina}

{<**SCC-125**>,
<**SCC-148**>}

T = {NomeA}

{<Zeca>,
<Juca>}

Divisão

- **Operação de Divisão - $R \div S$**
 - $T \leftarrow R \div S$
 - S é uma relação cujos atributos (B) são um subconjunto dos atributos (A) da relação R
 - $T(C) \leftarrow R(A) \div S(B)$, com:
 - $B \subseteq A$
 - $C = A - B$
 - uma tupla t pertence ao resultado $T(C)$ se existirem tuplas t_R em R tal que $t_R[C] = t$, e com $t_R[B] = t_S$ para toda tupla t_S em S

Em outras palavras: para t aparecer no resultado, t tem que aparecer em R em combinação com cada tupla de S



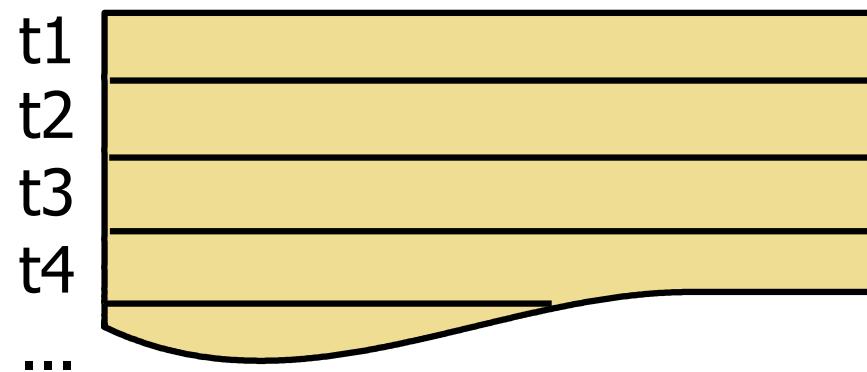
Divisão

- **Exercício:** pesquise como fazer uma divisão em SQL
 - Dica: livro [Elmasri&Navathe]

Operações sobre Conjuntos

- Operações usuais da **Teoria dos Conjuntos**
- Na Álgebra Relacional cada relação é considerada um **conjunto** de tuplas

$$R = \{t_1, t_2, t_3, \dots, t_n\}$$





Operações sobre Conjuntos

- Operações sobre Conjuntos \Rightarrow levam em consideração apenas **estrutura** da relação, e **não a semântica**
- Operações Binárias sobre Conjuntos \Rightarrow a maioria exige **Compatibilidade de Domínio** das relações

Exemplo

Aluno = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zeca, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

Professor = {Nome, Idade, Depto}

{<Zeca, 25, computação>,
<Ari, 30, computação>,
<Eva, 27, eletrônica>}

Dom(Dept) = Dom(Curso)

União

Aluno \cup Professor = {Nome, Idade, Curso}

{<Zeca, 25, computação>,
<Zeca, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação>,
<Ari, 30, computação>,
<Eva, 27, eletrônica>}



Operações sobre Conjuntos

- **União $\rightarrow R \cup S$**
 - resultado: todas as tuplas de S e todas as tuplas de R;
 - tuplas repetidas são eliminadas
 - requer compatibilidade de domínio
 - convenção: relação resultado tem os nomes dos atributos da primeira relação
 - é possível renomear
 - operação comutativa e associativa
 - em SQL?

Exemplo

Aluno = {Nome, Idade, Depto}

{<Zeca, 25, computação>,
<Zeca, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

Professor = {Nome, Idade, Depto}

{<Zeca, 25, computação>,
<Ari, 30, computação>,
<Eva, 27, eletrônica>}

Interseção

Aluno ∩ Professor = {Nome, Idade, Depto}

{<Zeca, 25, computação>}



Operações sobre Conjuntos

- **Interseção $\rightarrow R \cap S$**
 - resultado: apenas as tuplas que estão, simultaneamente, em R e em S
 - requer compatibilidade de domínio
 - convenção: relação resultado tem os nomes dos atributos da primeira relação
 - é possível renomear
 - operação comutativa e associativa
 - em SQL?

Exemplo

Aluno = {Nome, Idade, Depto}

{<Zeca, 25, computação>,
<Zeca, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

Professor = {Nome, Idade, Depto}

{<Zeca, 25, computação>,
<Ari, 30, computação>,
<Eva, 27, eletrônica>}

Diferença

Aluno - Professor = {Nome, Idade, Depto}

{<Zeca, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação>}

Professor - Aluno = {Nome, Idade, Depto}

{<Ari, 30, computação>,
<Eva, 27, eletrônica>}



Operações sobre Conjuntos

- **Diferença $\rightarrow R - S$**
 - resultado: tuplas que estão em R mas não estão em S
 - requer compatibilidade de domínio
 - convenção: relação resultado tem os nomes da atributos da primeira relação
 - é possível renomear
 - operação não comutativa
 - em SQL?

Exemplo

Aluno = {Nome, Idade, Depto}

{<Zeca, 25, computação>,
<Zeca, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação> }

Professor = {Nome, Idade, Depto}

{<Zeca, 25, computação>,
<Ari, 30, computação>,
<Eva, 27, eletrônica>}

União Exclusiva

Aluno \cup | Professor = {Nome, Idade, Depto}

{<Zeca, 18, eletrônica>,
<Juca, 21, odontologia>,
<Tuca, 18, computação>,
<Ari, 30, computação>,
<Eva, 27, eletrônica>}



Operações sobre Conjuntos

- **União Exclusiva $\rightarrow R \cup | S$**
 - resultado: tuplas que estão em S ou em R, mas não as tuplas que estão em ambas
 - requer compatibilidade de domínio
 - convenção: relação resultado tem os nomes da atributos da primeira relação
 - é possível renomear
 - operação comutativa
 - em SQL?

Exemplo:

Oferece = {Curso, DeptO}
{<Comp, SCC>,
<Comp, SMA>,
<Matem, SMA>}

Disciplina = {Codigo,DeptD}
{<SCC182, SCC>,
<SCC181, SCC>}

Produto Cartesiano

Oferece X Disciplina = {Curso, DeptO, Codigo, DeptD}

→ {<Comp, SCC, SCC182, SCC >,
 <Comp, SCC, SCC181, SCC >,
 <Comp, SMA, SCC182, SCC >,
 <Comp, SMA, SCC181, SCC >,
 <Matem, SMA, SCC182, SCC >,
 <Matem, SMA, SCC181, SCC >}

Exemplo:

Oferece = {Curso, DeptO}

{<Comp, SCC>,
<Comp, SMA>,
<Matem, SMA>}

Disciplina = {Codigo,DeptD}

{<SCC182, SCC>,
<SCC181, SCC>}

Produto Cartesiano

Oferece X Disciplina = {Curso, DeptO, Codigo, DeptD}

{<Comp, SCC, SCC182, SCC >,
<Comp, SCC, SCC181, SCC>,
<Comp, SMA, SCC182, SCC >,
<Comp, SMA, SCC181, SCC >,
<Matem, SMA, SCC182, SCC >,
<Matem, SMA, SCC181, SCC >}

Em SQL:

```
select * from Oferece, Disciplina
```



Operações sobre Conjuntos

- **Produto Cartesiano $\rightarrow R \times S$**
 - resultado: relação que tem como atributos a **concatenação dos atributos** da relação R e da relação S
 - tuplas: todas as **combinações** possíveis de tuplas de R com tuplas de S
 - não requer Compatibilidade de Domínio



Operações sobre Conjuntos

- **Operações sobre Conjuntos em SQL?**

Exercício: pesquise quais são e como usar os comandos em SQL correspondentes a operadores sobre conjuntos da Álgebra Relacional.



Leitura recomendada

- R. Elmasri, S. Navathe: *Sistemas de Banco de Dados*
 - 4^a Edição
 - Capítulo 6
 - 6^a Edição
 - Capítulo 6

Exercícios

empregado = {nomeEmpregado, CPF, rua, cidade,
telefone, idade}

trabalha = {empregado, companhia, salário}

companhia = {nome, CNPJ}

filial = { CNPJ, cidade}



Exercícios

- **Q1:** Liste nome e cidade de todos os empregados da IBM que ganham mais de dez mil dólares por mês
- **Q2:** Liste os nomes de todos os empregados que não trabalham para a IBM.
- **Q3:** Liste o nome de todas as companhias com filiais em **todas** as cidades onde há unidades da IBM
- **Q4:** Liste nome e CPF de todos os empregados e, para os que tiverem trabalhando, liste o CNPJ e o nome da(s) companhia(s) em que trabalham.
- **Q5: (DESAFIO)** Liste o nome de cada empregado que mora numa cidade onde há filial da(s) companhia(s) em que trabalha.