

# **Bases de Dados**

**Módulo 9c: Junções com diversas tabelas**

**Prof. André Bruno de Oliveira**

19/04/24 11:21

# Junções de diversas tabelas

- É muito comum que hajam diversas tabelas em um banco de dados. Estas tabelas costumam possuir vínculos entre si através de atributos com mesmo significado. Durante nossos módulos das aulas anteriores vimos alguns exemplos que envolvem duas ou mais relações. É importante estar familiarizado com este tipo de operação para resolver query no SQL.
  - Como exemplo, podemos elucidar uma junção entre 4 relações na álgebra relacional:

$M1 \bowtie M2 \bowtie M3 \bowtie M4$

O comando equivalente em SQL é dado por:

```
SELECT * FROM t1 NATURAL JOIN t2 NATURAL JOIN t3 NATURAL JOIN t4;
```

# Junções de diversas tabelas

- Considere o seguinte exemplo:
  - Exemplo SQL de junção natural de quatro tabelas: t1, t2, t3 e t4.
  - Em uma situação como essa, seja na Álgebra Relacional ou em SQL, o principal segredo é entender que a operação de junção é feita aos pares, respeitando a ordem em que os pares são especificados. Isto significa que primeiro vai ser efetuada a junção de t1 com t2. O conjunto de resultados dessa operação será então utilizado para sofrer a junção com a tabela t3. A seguir, o resultado obtido será utilizado para sofrer a junção com a tabela t4.

```
SELECT * FROM t1  
    NATURAL JOIN t2  
    NATURAL JOIN t3  
    NATURAL JOIN t4;
```

# Junções de diversas tabelas

- A tabela abaixo inclui a descrição de quatro relações do nosso banco de dados **Empregados** usados nesta aula prática: Departamento, Funcionário, Projeto e EquipeProjeto.

Nome da relação	Descrição
<i>Departamento</i> (codDepto, nomDepto)	Departamentos da empresa.
Funcionario(matFunc, nome, idade, salario, codDepto)	Matrícula (“matFunc”), nome, idade, salário e código do departamento (“codDepto”) dos funcionários da empresa.
Projeto(codProj, nomProj, uf)	Projetos da empresa. Cada projeto tem um código identificador único (“codProj”), um nome (“nomProj”) e é conduzido em alguma UF do Brasil (ex: 'RJ', 'DF', 'MG', ...)
EquipeProjeto(matFunc, codProj)	Armazena os funcionários alocados a cada projeto.

# Junções de diversas tabelas

- Para esta aula prática os seguintes conteúdos serão usados nas tabelas (instâncias) para este banco de dados.

*Departamento*

	codDepto	nomDepto
1	1	Engenharia
2	2	Arquitetura
3	3	RH

*Funcionario*

	matFunc	nome	idade	salario	codDepto
1	M1	Jennifer	30	5000	1
2	M2	Hector	22	3000	3
3	M3	Molina	49	4200	1
4	M4	Larry	25	1500	2
5	M5	Ellison	37	4200	1
6	M6	Yuko	31	3000	2

*Projeto*

	codProj	nomProj	uf
1	50	PROJETO ALPHA	RJ
2	99	PROJETO BETA	DF

*Equipeprojeto*

	matFunc	codProj
1	M1	50
2	M3	50
3	M4	50
4	M1	99
5	M5	99

# Junções de diversas tabelas –NATURAL JOIN

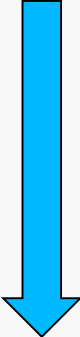
- Considere o seguinte exemplo:
  - Exemplo de SELECT que recupera a matrícula, nome dos funcionários e o nome do departamento onde os mesmos trabalham. O resultado final está ordenado pela matrícula.

```
SELECT f.matFunc, f.nome, d.nomDepto
```

```
FROM Funcionario f NATURAL JOIN Departamento d
```

```
ORDER BY matFunc;
```

*Resultado Final*



	matFunc	nome	nomDepto
1	M1	Jennifer	Engenharia
2	M2	Hector	RH
3	M3	Molina	Engenharia
4	M4	Larry	Arquitetura
5	M5	Ellison	Engenharia
6	M6	Yuko	Arquitetura

# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:
  - Veja que para recuperar as informações desejadas foi preciso realizar a junção entre as tabelas *Funcionario* (que contém o nome e matrícula do funcionário) e a tabela *Departamento* (que contém o nome do departamento). Esse par de tabelas pode participar de uma operação de join porque possui um atributo em comum: o código do departamento (“codDepto”).

```
SELECT f.matFunc, f.nome, d.nomDepto  
FROM Funcionario f NATURAL JOIN Departamento d  
ORDER BY matFunc;
```

*Funcionario*

	matFunc	nome	idade	salario	codDepto
1	M1	Jennifer	30	5000	1
2	M2	Hector	22	3000	3
3	M3	Molina	49	4200	1
4	M4	Larry	25	1500	2
5	M5	Ellison	37	4200	1
6	M6	Yuko	31	3000	2

*Departamento*

	codDepto	nomDepto
1	1	Engenharia
2	2	Arquitetura
3	3	RH

*Resultado Final*

	matFunc	nome	nomDepto
1	M1	Jennifer	Engenharia
2	M2	Hector	RH
3	M3	Molina	Engenharia
4	M4	Larry	Arquitetura
5	M5	Ellison	Engenharia
6	M6	Yuko	Arquitetura

# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:

SQL

```
SELECT f.matFunc, f.nome, d.nomDepto
FROM Funcionario f NATURAL JOIN Departamento d
ORDER BY matFunc;
```

Equivalente na AR

$\pi_{\text{matFunc, nome, nomeDepto}}(\text{Funcionario} \bowtie \text{Departamento})$

*Funcionario*

	matFunc	nome	idade	salario	codDepto
1	M1	Jennifer	30	5000	1
2	M2	Hector	22	3000	3
3	M3	Molina	49	4200	1
4	M4	Larry	25	1500	2
5	M5	Ellison	37	4200	1
6	M6	Yuko	31	3000	2

*Departamento*

	codDepto	nomDepto
1	1	Engenharia
2	2	Arquitetura
3	3	RH

*Resultado Final*

	matFunc	nome	nomDepto
1	M1	Jennifer	Engenharia
2	M2	Hector	RH
3	M3	Molina	Engenharia
4	M4	Larry	Arquitetura
5	M5	Ellison	Engenharia
6	M6	Yuko	Arquitetura

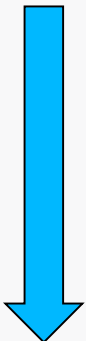


# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:
  - Exemplo de SELECT com NATURAL JOIN de três tabelas.

```
SELECT f.matFunc, f.nome, d.nomDepto, e.codProj
FROM Funcionario f
NATURAL JOIN EquipeProjeto e
NATURAL JOIN Departamento d
ORDER BY matFunc;
```

*Resultado Final*



	matFunc	nome	nomDepto	codProj
1	M1	Jennifer	Engenharia	50
2	M1	Jennifer	Engenharia	99
3	M3	Molina	Engenharia	50
4	M4	Larry	Arquitetura	50
5	M5	Ellison	Engenharia	99

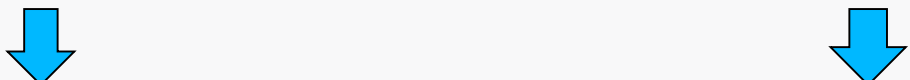
# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:
  - Vamos compreender como este SELECT é resolvido.
  - Primeiro as junções entre tabelas são sempre feitas aos pares e seguindo a ordem especificada no FROM. Neste caso, o SELECT faz primeiro a junção natural entre *Funcionario* e *EquipeProjeto* (chave matFunc), obtendo **um resultado intermediário** (uma relação temporária na memória de trabalho do SGBD) que é apresentado na página a seguir. Neste resultado intermediário, teremos um conjunto contendo todas as tuplas de *Funcionario* corretamente “casadas” com as tuplas de *EquipeProjeto*.
  - **É encontrado o resultado intermediário 1** produzido por *Funcionario* NATURAL JOIN *EquipeProjeto*. Usou-se o campo comum matFun.

	matFun	nome	idade	salario	codDep	matFun	codProj
1	M1	Jennifer	30	5000	1	M1	50
2	M1	Jennifer	30	5000	1	M1	99
3	M3	Molina	49	4200	1	M3	50
4	M4	Larry	25	1500	2	M4	50
5	M5	Ellison	37	4200	1	M5	99

# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:
  - Conceitualmente, a operação de junção não descarta nenhum atributo do par de tabelas envolvidas. Por isso, o **resultado intermediário 1** contém todos os atributos de *Funcionario* e todos os de *EquipeProjeto*.
  - Veja que os funcionários de matrícula **M2** e **M6** não aparecem neste **resultado intermediário 1**, pois não estão alocados a nenhum projeto e a funcionária de matrícula **M1** aparece duas vezes, pois ela está alocada a dois projetos.



	matFun	nome	idade	salario	codDep	matFun	codProj
1	M1	Jennifer	30	5000	1	M1	50
2	M1	Jennifer	30	5000	1	M1	99
3	M3	Molina	49	4200	1	M3	50
4	M4	Larry	25	1500	2	M4	50
5	M5	Ellison	37	4200	1	M5	99

# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:
  - Vamos compreender como este SELECT é resolvido.
  - Numa segunda etapa, o conjunto de **resultado intermediário 1** será então utilizado no **NATURAL JOIN** com a tabela *Departamento*. Como resultado (ainda um resultado intermediário), teremos um conjunto contendo todas as tuplas de **Resultado intermediário 1** corretamente casadas com as tuplas de *Departamento*. A variável em comum entre ambas as relações desta vez é “**codDep**” e o atributo que **é acrescentado é “NomDep**”.
  - **É encontrado o resultado intermediário 2** produzido por *Resultado intermediário 1* NATURAL JOIN *Departamento*. Usou-se o campo comum matFun.

	matFun	nome	idade	salario	codDep	matFun	codProj	codDep	nomDep
1	M1	Jennifer	30	5000	1	M1	50	1	Engenharia
2	M1	Jennifer	30	5000	1	M1	99	1	Engenharia
3	M3	Molina	49	4200	1	M3	50	1	Engenharia
4	M4	Larry	25	1500	2	M4	50	2	Arquitetura
5	M5	Ellison	37	4200	1	M5	99	1	Engenharia

# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:
  - Vamos compreender como este SELECT é resolvido.
  - Na etapa final, estando todas as junções resolvidas e não existindo nenhuma cláusula WHERE na consulta, o SGBD gera o resultado final projetando os atributos que o usuário especificou na “SELECT list” (lista ao lado da palavra SELECT): f.matFunc, f.nome, d.nomDepto, e.codProj.

## SQL

```
SELECT f.matFunc, f.nome, d.nomDepto, e.codProj
FROM Funcionario f
NATURAL JOIN EquipeProjeto e
NATURAL JOIN Departamento d
ORDER BY matFunc;
```

## Resultado Final

	matFunc	nome	nomDepto
1	M1	Jennifer	Engenharia
2	M2	Hector	RH
3	M3	Molina	Engenharia
4	M4	Larry	Arquitetura
5	M5	Ellison	Engenharia
6	M6	Yuko	Arquitetura

## Equivalente na AR

$$\pi_{\text{matFunc, nome, nomeDepto, codProjeto}}(\text{Funcionario} \bowtie \text{EquipeProjeto} \bowtie \text{Departamento})$$

# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:
  - Exemplo que modifica a consulta do exemplo anterior para exibir o nome do projeto ao invés do código.

## SQL

```
SELECT f.matFunc, f.nome, d.nomDepto, p.nomProj
```

```
FROM Funcionario f
```

```
NATURAL JOIN EquipeProjeto e
```

```
NATURAL JOIN Projeto p
```

```
NATURAL JOIN Departamento d
```

```
ORDER BY matFunc;
```

## Resultado Final

	matFunc	nome	nomDepto	nomProj
1	M1	Jennifer	Engenharia	PROJETO ALPHA
2	M1	Jennifer	Engenharia	PROJETO BETA
3	M3	Molina	Engenharia	PROJETO ALPHA
4	M4	Larry	Arquitetura	PROJETO ALPHA
5	M5	Ellison	Engenharia	PROJETO BETA

## Equivalente na AR

$\pi_{\text{matFunc, nome, nomeDepto, nomProj}}(\text{Funcionario} \bowtie \text{EquipeProjeto} \bowtie \text{Projeto} \bowtie \text{Departamento})$

# Junções de diversas tabelas – NATURAL JOIN

- Considere o seguinte exemplo:

- Neste exemplo, a tabela Projeto precisou ser acrescentada ao FROM, pois o nome do projeto (“nomProj”) está armazenado nesta tabela. Como não existem atributos em comum entre Funcionario e Projeto, primeiro é preciso realizar a junção de Funcionario com EquipeProjeto (possuem “matFunc” como atributo em comum). O resultado produzido poderá então sofrer uma junção com Projeto (possuirão “codProj” como variável em comum).

## SQL

```
SELECT f.matFunc, f.nome, d.nomDepto, p
FROM Funcionario f
NATURAL JOIN EquipeProjeto e
NATURAL JOIN Projeto p
NATURAL JOIN Departamento d
ORDER BY matFunc;
```

## Resultado Final

	matFunc	nome	nomDepto	nomProj
1	M1	Jennifer	Engenharia	PROJETO ALPHA
2	M1	Jennifer	Engenharia	PROJETO BETA
3	M3	Molina	Engenharia	PROJETO ALPHA
4	M4	Larry	Arquitetura	PROJETO ALPHA
5	M5	Ellison	Engenharia	PROJETO BETA

## Equivalente na AR

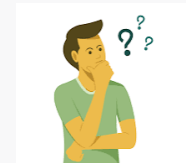
$\pi_{\text{matFunc, nome, nomeDepto, nomProj}}(\text{Funcionario} \bowtie \text{EquipeProjeto} \bowtie \text{Projeto} \bowtie \text{Departamento})$

# Junções de diversas tabelas – Junção Theta

- Considere o seguinte exemplo:
  - Exemplo anterior de expressão SQL equivalente ao exemplo anterior com uso do WHERE.
  - A Junção theta neste exemplo é usada para realizar a junção das quatro tabelas. A expressão SQL e a expressão AR são equivalentes as expressões do exemplo anterior. É importante mencionar que de acordo com o espaço de memória e tempo de processamento de CPU expressões equivalentes em SQL podem gerar custo maior para o SGBD resolver.

## SQL

```
SELECT f.matFunc, f.nome, d.nomDepto, p.nomProj
FROM Funcionario f, EquipeProjeto e, Projeto p, Departamento d
WHERE f.matFunc= e.matFun AND e.codProj=p.codProj AND f.codDpto=d. codDpto
ORDER BY matFunc;
```



## Equivalente na AR

$$\pi_{f.matfunc, f.nome, d.nomdepto, p.nomproj} \left( \sigma_{f.matfunc = e.matfun \text{ AND } e.codproj = p.codproj \text{ AND } f.coddpto = d.coddpto} (\rho_f \text{funcionario} \times \rho_e \text{equipeprojeto} \times \rho_p \text{projeto} \times \rho_d \text{departamento}) \right)$$



# Junções de diversas tabelas

- \* \* **IMPORTANTE:** Como observação final é importante comentar que não apenas a junção natural, mas também a junção theta e o produto cartesiano são resolvidos aos pares, respeitando a ordem de especificação dos mesmos dentro do **SELECT**.

Obrigado