

# Universidade Presbiteriana Mackenzie



Banco de Dados – Aula 17  
Linguagem SQL  
SELECT com várias tabelas  
(LEFT, RIGHT e FULL JOIN)

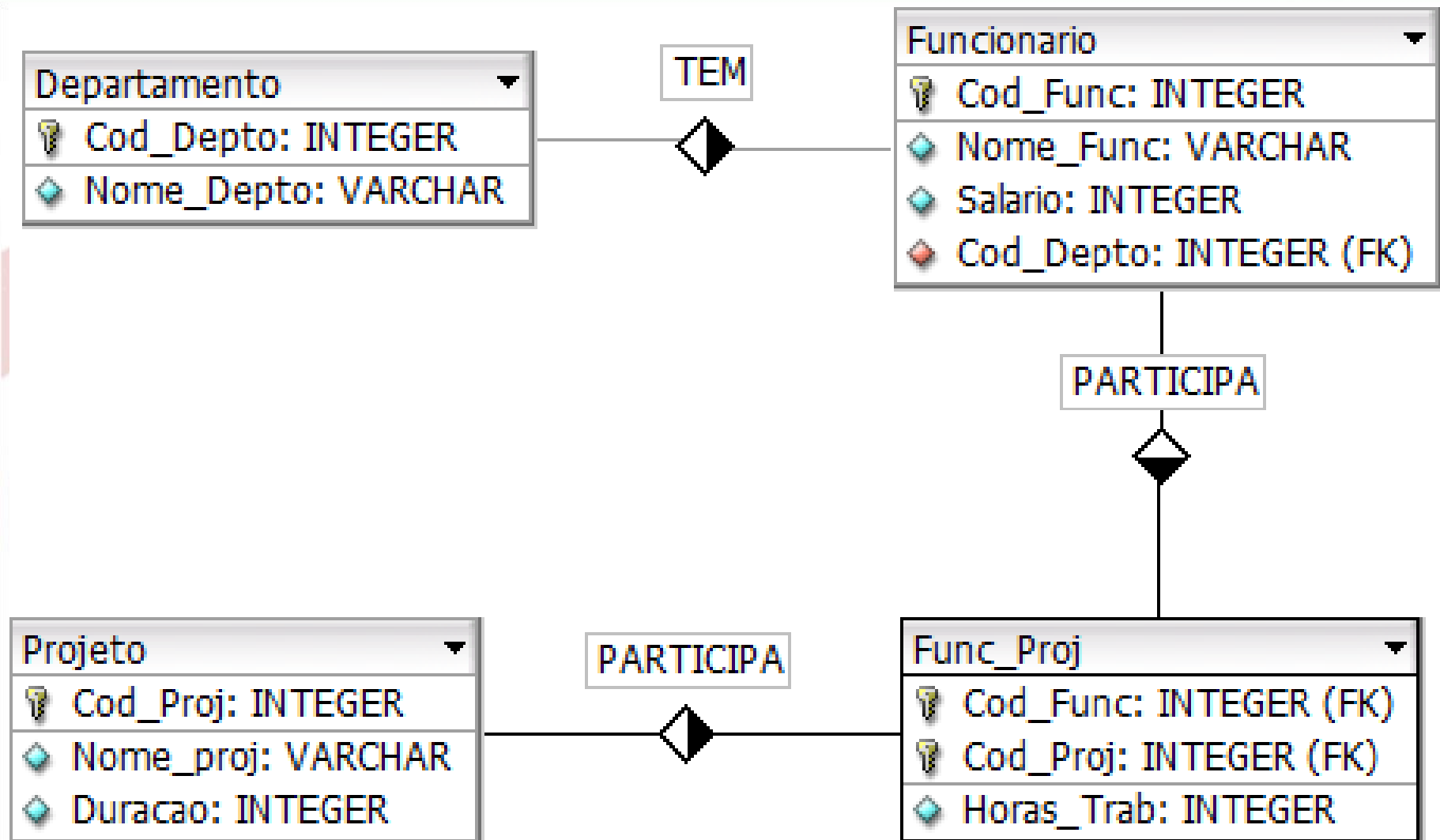
**Profa. Elisângela Botelho Gracias**

Faculdade de Computação e Informática

# Introdução

**Considere o seguinte Banco de Dados para esta aula (chave primária está sublinhada)**

- Departamento = {Cod\_Depto, Nome\_Depto}
- Funcionario = {Cod\_Func, Nome\_Func, Salario, Cod\_Depto}
  - Cod\_depto é chave estrangeira que referencia o atributo Cod\_depto da tabela Departamento
- Projeto = {Cod\_Proj, Nome\_Proj, Duracao}
- Func\_Proj = {Cod\_Func, Cod\_Proj, Horas\_Trab}
  - Cod\_Func é chave estrangeira que referencia o atributo Cod\_Func da tabela Funcionario
  - Cod\_Proj é chave estrangeira que referencia o atributo Cod\_Proj da tabela Projeto



**Departamento**

Cod_Depto	Nome_Depto
1	Marketing
2	Vendas
3	Dados
4	Pesquisa

**Funcionário**

Cod_Func	Nome_Func	Salario	Cod_Depto
101	Joao da Silva	2000	2
102	Mario Souza	1500	1
103	Sergio Santos	2400	2
104	Maria Castro	1200	1
105	Marcio Santana	1400	4

**Projeto**

Cod_Proj	Nome_Proj	Duracao
1001	Sistema A	2
1002	Sistema B	6
1003	Sistema X	4

**Func\_Proj**

Cod_Func	Cod_Proj	Horas_Trab
101	1001	24
101	1002	160
102	1001	56
102	1003	45
103	1001	86
103	1003	64
104	1001	46

-- Script de Criação do BD Projeto

```
DROP TABLE Func_Proj CASCADE CONSTRAINT;  
DROP TABLE Projeto CASCADE CONSTRAINT;  
DROP TABLE Funcionario CASCADE CONSTRAINT;  
DROP TABLE Departamento CASCADE CONSTRAINT;
```

```
CREATE TABLE Departamento  
(Cod_Depto INTEGER,  
Nome_Depto VARCHAR(20) NOT NULL,  
PRIMARY KEY(Cod_Depto));
```

```
CREATE TABLE Funcionario  
(Cod_Func INTEGER,  
Nome_Func VARCHAR(20) NOT NULL,  
Salario INTEGER NOT NULL,  
Cod_Depto INTEGER,  
PRIMARY KEY(Cod_Func),  
FOREIGN KEY (Cod_Depto) REFERENCES Departamento (Cod_Depto));
```

```
CREATE TABLE Projeto  
(Cod_Proj INTEGER,  
Nome_Proj VARCHAR(20) NOT NULL,  
Duracao INTEGER NOT NULL,  
PRIMARY KEY(Cod_Proj));
```

```
CREATE TABLE Func_Proj  
(Cod_Func INTEGER,  
Cod_Proj INTEGER,  
Horas_Trab INTEGER,  
PRIMARY KEY(Cod_Func, Cod_Proj),  
FOREIGN KEY (Cod_Func) REFERENCES Funcionario(Cod_Func),  
FOREIGN KEY (Cod_Proj) REFERENCES Projeto(Cod_Proj));
```

```
INSERT INTO Departamento (Cod_Depto, Nome_Depto) VALUES (1, 'Marketing');  
INSERT INTO Departamento (Cod_Depto, Nome_Depto) VALUES (2, 'Vendas');  
INSERT INTO Departamento (Cod_Depto, Nome_Depto) VALUES (3, 'Dados');  
INSERT INTO Departamento (Cod_Depto, Nome_Depto) VALUES (4, 'Pesquisa');
```

```
INSERT INTO Funcionario (Cod_Func, Nome_Func, Salario, Cod_Depto) VALUES (101, 'Joao da Silva Santos', 2000, 2);  
INSERT INTO Funcionario (Cod_Func, Nome_Func, Salario, Cod_Depto) VALUES (102, 'Mario Souza', 1500, 1);  
INSERT INTO Funcionario (Cod_Func, Nome_Func, Salario, Cod_Depto) VALUES (103, 'Sergio Silva Santos', 2400, 2);  
INSERT INTO Funcionario (Cod_Func, Nome_Func, Salario, Cod_Depto) VALUES (104, 'Maria Castro', 1200, 1);  
INSERT INTO Funcionario (Cod_Func, Nome_Func, Salario, Cod_Depto) VALUES (105, 'Marcio Silva Santana', 1400, 4);
```

```
INSERT INTO Projeto (Cod_Proj, Nome_Proj, Duracao) VALUES (1001, 'SistemaA', 2);  
INSERT INTO Projeto (Cod_Proj, Nome_Proj, Duracao) VALUES (1002, 'SistemaB', 6);  
INSERT INTO Projeto (Cod_Proj, Nome_Proj, Duracao) VALUES (1003, 'SistemaX', 4);
```

```
INSERT INTO Func_Proj (Cod_Func, Cod_Proj, Horas_Trab) VALUES (101, 1001, 24);  
INSERT INTO Func_Proj (Cod_Func, Cod_Proj, Horas_Trab) VALUES (101, 1002, 160);  
INSERT INTO Func_Proj (Cod_Func, Cod_Proj, Horas_Trab) VALUES (102, 1001, 56);  
INSERT INTO Func_Proj (Cod_Func, Cod_Proj, Horas_Trab) VALUES (102, 1003, 45);  
INSERT INTO Func_Proj (Cod_Func, Cod_Proj, Horas_Trab) VALUES (103, 1001, 86);  
INSERT INTO Func_Proj (Cod_Func, Cod_Proj, Horas_Trab) VALUES (103, 1003, 64);  
INSERT INTO Func_Proj (Cod_Func, Cod_Proj, Horas_Trab) VALUES (104, 1001, 46);  
COMMIT;
```

# JUNÇÃO EXTERNA



# Junção Externa

- Uma junção entre duas tabelas gera um resultado onde tem-se linhas que se combinam sobre a coluna de junção
- O operador de junção externa gera o resultado da junção (as linhas combinadas) mais as linhas não combinadas



# Junção Externa

- Uma junção externa de um lado gera um resultado com as linhas combinadas mais as linhas não combinadas com base em uma das tabelas
- A linguagem SQL utiliza as palavras-chave LEFT JOIN e RIGHT JOIN para produzir a junção externa de um lado

# JUNÇÃO EXTERNA

## LEFT JOIN

# Junção Externa – LEFT JOIN

- A palavra-chave LEFT JOIN gera um resultado contendo as linhas combinadas e as linhas não combinadas da tabela da esquerda

# Junção Externa – LEFT JOIN

- Junção externa completa das tabelas Departamento e Funcionario, utilizando o LEFT JOIN:

```
SELECT *  
FROM Departamento D LEFT JOIN Funcionario F  
ON (D.Cod_depto = F.Cod_depto)  
ORDER BY D.Nome_Depto ASC;
```

# Junção Externa – LEFT JOIN

- Junção externa completa das tabelas Departamento e Funcionario, utilizando o LEFT JOIN:

```
SELECT *  
FROM Departamento D LEFT JOIN Funcionario F  
ON (D.Cod_depto = F.Cod_depto)  
ORDER BY D.Nome_Depto ASC;
```



# Junção Externa – LEFT JOIN

- A junção externa completa das tabelas Departamento e Funcionario gera o seguinte resultado:

Cod_Depto	Nome_Depto	Cod_Func	Nome_Func	Salario	Cod_Depto
3	Dados	NULL	NULL	NULL	NULL
1	Marketing	102	Mario Souza	1500	1
1	Marketing	104	Maria Castro	1200	1
4	Pesquisa	105	Marcio Silva Santana	1400	4
2	Vendas	101	Joao da Silva Santos	2000	2
2	Vendas	103	Sergio Silva Santos	2400	2



# Junção Externa – LEFT JOIN

- A junção externa completa das tabelas Departamento e Funcionario gera o seguinte resultado:

Cod_Depto	Nome_Depto	Cod_Func	Nome_Func	Salario	Cod_Depto
3	Dados	NULL	NULL	NULL	NULL
1	Marketing	102	Mario Souza	1500	1
1	Marketing	104	Maria Castro	1200	1
4	Pesquisa	105	Marcio Silva Santana	1400	4
2	Vendas	101	Joao da Silva Santos	2000	2
2	Vendas	103	Sergio Silva Santos	2400	2

# Junção Externa – LEFT JOIN

- A junção externa completa das tabelas Departamento e Funcionario gera o seguinte resultado:

Cod_Depto	Nome_Depto	Cod_Func	Nome_Func	Salario	Cod_Depto
3	Dados	NULL	NULL	NULL	NULL
1	Marketing	102	Mario Souza	1500	1
1	Marketing	104	Maria Castro	1200	1
4	Pesquisa	105	Marcio Silva Santana	1400	4

Observe que o departamento 'Dados' não tem funcionário, portanto, os **dados referentes a um funcionário aparecem como nulos**

# Junção Externa – LEFT JOIN

- Exemplo (LEFT JOIN): Obtenha os nomes de TODOS os departamentos da empresa, com os nomes dos funcionários que trabalham em cada um.

```
SELECT D.Nome_Depto, F.Nome_func  
FROM Departamento D LEFT JOIN Funcionario F  
ON (D.Cod_depto = F.Cod_depto)  
ORDER BY D.Nome_Depto;
```

Nome_Depto	Nome_func
Dados	NULL
Marketing	Mario Souza
Marketing	Maria Castro
Pesquisa	Marcio Silva Santana
Vendas	Joao da Silva Santos
Vendas	Sergio Silva Santos

# Junção Externa – LEFT JOIN

- Exemplo (LEFT JOIN): Obtenha os nomes de todos os departamentos, com os nomes dos funcionários em cada um.

**Observe que o departamento 'Dados' não tem funcionário, mas apareceu no resultado**

Nome\_func

depto

Nome_Depto	Nome_func
Dados	NULL
Marketing	Mario Souza
Marketing	Maria Castro
Pesquisa	Marcio Silva Santana
Vendas	Joao da Silva Santos
Vendas	Sergio Silva Santos



# JUNÇÃO EXTERNA RIGHT JOIN





# Junção Externa – RIGHT JOIN

- A palavra-chave RIGHT JOIN gera um resultado contendo as linhas combinadas e as linhas não combinadas da tabela da direita
- Assim, o resultado de uma junção externa de um lado depende da direção (DIREITA ou ESQUERDA) e da posição dos nomes das tabelas



# Junção Externa – RIGHT JOIN

- Exemplo (RIGHT JOIN): Obtenha os nomes de todos os departamentos da empresa, com os nomes dos funcionários que trabalham em cada um.

```
SELECT D.Nome_Depto, F.Nome_func  
FROM Funcionario F RIGHT JOIN Departamento D  
ON (D.Cod_depto = F.Cod_depto)  
ORDER BY D.Nome_Depto;
```

Nome_Depto	Nome_func
<b>Dados</b>	<b>NULL</b>
Marketing	Mario Souza
Marketing	Maria Castro
Pesquisa	Marcio Silva Santana
Vendas	Joao da Silva Santos
Vendas	Sergio Silva Santos

# Junção Externa – RIGHT JOIN

- Exemplo (RIGHT JOIN): Obtenha os nomes de todos os departamentos da empresa, com os nomes dos funcionários que trabalham em cada um.

```
SELECT D.Nome_Depto, F.Nome_func  
FROM Funcionario F RIGHT JOIN Departamento D  
ON (D.Cod_depto = F.Cod_depto)  
ORDER BY D.Nome_Depto;
```

Nome_Depto	Nome_func
<b>Dados</b>	<b>NULL</b>
Marketing	Mario Souza
Marketing	Maria Castro
Pesquisa	Marcio Silva Santana
Vendas	Joao da Silva Santos
Vendas	Sergio Silva Santos

# Junção Externa – RIGHT JOIN

- Observe que os exemplos anteriores, utilizando LEFT JOIN e RIGHT JOIN, trazem o mesmo resultado, com alterações apenas na ordem em que as tabelas aparecem na cláusula FROM.

# Exemplos de LEFT JOIN e RIGHT JOIN

- Exemplo: Obtenha os nomes de TODOS os departamentos da empresa e a quantidade de funcionários pertencentes a cada um deles (retorne mesmo aqueles departamentos onde não tem funcionários)
- Observe os exemplos a seguir utilizando LEFT JOIN e RIGHT JOIN

# Exemplos de LEFT JOIN e RIGHT JOIN

- LEFT JOIN

```
SELECT D.Nome_Depto, COUNT(F.Cod_Func) AS Total  
FROM Departamento D LEFT JOIN Funcionario F  
ON (D.Cod_depto = F.Cod_depto)  
GROUP BY D.Nome_Depto  
ORDER BY D.Nome_Depto ASC;
```

Nome_Depto	Total
<b>Dados</b>	<b>0</b>
Marketing	2
Pesquisa	1
Vendas	2



# Exemplos de LEFT JOIN e RIGHT JOIN

- LEFT JOIN

```
SELECT D.Nome_Depto, COUNT(F.Cod_Func) AS Total
FROM Departamento D LEFT JOIN Funcionario F
ON (D.Cod_depto = F.Cod_depto)
GROUP BY D.Nome_Depto
ORDER BY D.Nome_Depto ASC;
```


Nome_Depto	Total
Dados	0
Marketing	2
Pesquisa	1
Vendas	2



# Exemplos de LEFT JOIN e RIGHT JOIN

- LEFT JOIN

```
SELECT D.Nome_Depto, COUNT(F.Cod_Func) AS Total  
FROM Departamento D LEFT JOIN Funcionario F  
ON (D.Cod_depto = F.Cod_depto)  
GROUP BY D.Nome_Depto
```



Observe que o departamento de 'Dados' tem ZERO funcionários, portanto, atente-se ao atributo que colocar no COUNT

Nome_Depto	Total
Dados	0
Marketing	2
Pesquisa	1
Vendas	2

# Exemplos de LEFT JOIN e RIGHT JOIN

- **RIGHT JOIN**

```
SELECT D.Nome_Depto, COUNT(F.Cod_Func) AS Total
FROM Funcionario F RIGHT JOIN Departamento D
ON (D.Cod_depto = F.Cod_depto)
GROUP BY D.Nome_Depto
ORDER BY D.Nome_Depto ASC;
```

Nome_Depto	Total
<b>Dados</b>	<b>0</b>
Marketing	2
Pesquisa	1
Vendas	2

# Exemplos de LEFT JOIN com mais de 2 tabelas

**LEFT JOIN:** Obtenha o nome de todos os funcionários e o nome dos projetos que cada um trabalhou (retorne mesmo aqueles funcionários que não trabalharam em nenhum projeto ainda)

```
SELECT F.Nome_Func, P.Nome_Proj  
FROM Funcionario F LEFT JOIN Func_Proj FP  
ON (F.Cod_Func = FP.Cod_Func)  
LEFT JOIN Projeto P ON (FP.Cod_Proj = P.Cod_Proj)  
ORDER BY P.Nome_Proj ASC;
```

# Exemplos de LEFT JOIN com mais de 2 tabelas

**LEFT JOIN:** Obtenha o nome de todos os funcionários e o nome dos projetos que cada um trabalhou (retorne mesmo aqueles funcionários que não trabalharam em nenhum projeto ainda)

```
SELECT F.Nome_Func, P.Nome_Proj  
FROM Funcionario F LEFT JOIN Func_Proj FP  
ON (F.Cod_Func = FP.Cod_Func)  
LEFT JOIN Projeto P ON (FP.Cod_Proj = P.Cod_Proj)  
ORDER BY P.Nome_Proj ASC;
```

# Junção Externa – LEFT JOIN

- O resultado da consulta anterior gera o seguinte resultado:

Nome_Func	Nome_Proj
Marcio Silva Santana	NULL
Maria Castro	SistemaA
Sergio Silva Santos	SistemaA
Joao da Silva Santos	SistemaA
Mario Souza	SistemaA
Joao da Silva Santos	SistemaB
Mario Souza	SistemaX
Sergio Silva Santos	SistemaX



# Junção Externa – LEFT JOIN

- O resultado da consulta anterior gera o seguinte resultado:

Nome_Func	Nome_Proj
<b>Marcio Silva Santana</b>	<b>NULL</b>
Maria Castro	SistemaA



Observe que o funcionário ‘Marcio Silva Santana’, que não participou de nenhum projeto, apareceu no resultado, mas sem nenhum projeto vinculado a ele

Mario Souza	SistemaX
Sergio Silva Santos	SistemaX



# Exemplos de RIGHT e LEFT JOIN na mesma consulta

Observe que o exemplo anterior pode ser feito utilizando tanto o LEFT JOIN quanto o RIGHT JOIN, trazendo o mesmo resultado:

```
SELECT F.Nome_Func, P.Nome_Proj  
FROM Func_Proj FP RIGHT JOIN Funcionario F  
ON (FP.Cod_Func = F.Cod_Func)  
LEFT JOIN Projeto P  
ON (FP.Cod_Proj = P.Cod_Proj);
```

# Exemplos de RIGHT e LEFT JOIN na mesma consulta

Observe que o exemplo anterior pode ser feito utilizando tanto o LEFT JOIN quanto o RIGHT JOIN, trazendo o mesmo resultado:

```
SELECT F.Nome_Func, P.Nome_Proj  
FROM Func_Proj FP RIGHT JOIN Funcionario F  
ON (FP.Cod_Func = F.Cod_Func)  
LEFT JOIN Projeto P  
ON (FP.Cod_Proj = P.Cod_Proj);
```

## Exemplos de RIGHT e INNER JOIN na mesma consulta

Observe que o exemplo anterior pode ser feito utilizando tanto o INNER JOIN quanto o RIGHT JOIN, trazendo o mesmo resultado:

```
SELECT F.Nome_Func, P.Nome_Proj  
FROM Func_Proj FP INNER JOIN Projeto P  
ON (FP.Cod_Proj = P.Cod_Proj)  
RIGHT JOIN Funcionario F  
ON (FP.Cod_Func = F.Cod_Func);
```

## Exemplos de RIGHT e INNER JOIN na mesma consulta

Observe que o exemplo anterior pode ser feito utilizando tanto o INNER JOIN quanto o RIGHT JOIN, trazendo o mesmo resultado:

```
SELECT F.Nome_Func, P.Nome_Proj  
FROM Func_Proj FP INNER JOIN Projeto P  
ON (FP.Cod_Proj = P.Cod_Proj)  
RIGHT JOIN Funcionario F  
ON (FP.Cod_Func = F.Cod_Func);
```



# JUNÇÃO EXTERNA FULL JOIN

# Junção Externa – FULL JOIN

- A palavra-chave FULL JOIN gera um resultado contendo as linhas combinadas mais todas as linhas não combinadas das tabelas envolvidas neste FULL JOIN



# Junção Externa – FULL JOIN

- Exemplo (FULL JOIN): Vamos inserir mais um funcionário, mas ele não pertencerá, inicialmente, a nenhum departamento.
- Lembrando que: se você não definiu a chave estrangeira Cod\_Depto da tabela Funcionario como NOT NULL, é permitido que um funcionário não pertença a nenhum departamento

**INSERT**

**INTO** Funcionario (Cod\_Func, Nome\_Func, Salario)

**VALUES** (106, 'Antonio Silva', 2800);

# Junção Externa – FULL JOIN

- Exemplo (FULL JOIN): Agora, temos:
  - um departamento que não tem funcionário – que é o departamento de 'Dados '
  - um funcionário que não pertence a nenhum departamento – que é o funcionário 'Antonio Silva'

# Junção Externa – FULL JOIN

- Exemplo (FULL JOIN): Obtenha todas as informações de funcionários e departamentos. Retorne mesmo aqueles departamentos que não tem funcionários e mesmo aqueles funcionários que não pertencem a nenhum departamento

**SELECT \***

**FROM Departamento D FULL JOIN Funcionario F**

**ON (D.Cod\_depto = F.Cod\_depto);**


# Junção Externa – FULL JOIN

- Exemplo (FULL JOIN): Obtenha todas as informações de funcionários e departamentos. Retorne mesmo aqueles departamentos que não tem funcionários e mesmo aqueles funcionários que não pertencem a nenhum departamento

Cod_Depto	Nome_Depto	Cod_Func	Nome_Func	Salario	Cod_Depto
1	Marketing	102	Mario Souza	1500	1
1	Marketing	104	Maria Castro	1200	1
2	Vendas	101	Joao da Silva Santos	2000	2
2	Vendas	103	Sergio Silva Santos	2400	2
3	Dados	NULL	NULL	NULL	NULL
4	Pesquisa	105	Marcio Silva Santana	1400	4
NULL	NULL	106	Antonio Silva	2800	NULL

## Função Externa – FULL JOIN

Observe que, além das **linhas combinadas**, foram retornados o **departamento de 'Dados'** que não tem funcionário, e os dados do **funcionário 'Antonio Silva'** que não pertence a nenhum departamento



Cod_Depto	Nome_Depto	Cod_Func	Nome_Func	Salario	Cod_Depto
	Marketing	102	Mario Souza	1500	1
	Marketing	104	Maria Castro	1200	1
2	Vendas	101	Joao da Silva Santos	2000	2
2	Vendas	103	Sergio Silva Santos	2400	2
3	Dados	NULL	NULL	NULL	NULL
4	Pesquisa	105	Marcio Silva Santana	1400	4
NULL	NULL	106	Antonio Silva	2800	NULL



# Obrigado

Profa. Elisângela Botelho Gracias  
[elisangela.botelho@mackenzie.br](mailto:elisangela.botelho@mackenzie.br)

