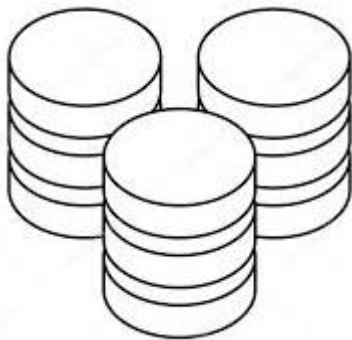


Banco de Dados II

Select
Operações de Conjunto
JOIN



Profa. Damires Souza
damires@ifpb.edu.br



Comando Select

Select [distinct] {*,colunas [alias],
expressões, funções}

From {tabelas [alias]}

[Where condições]

[Group by colunas]

[Having condição]

[Order by colunas [ASC|DESC]];

Da Álgebra Relacional

Seleção

$\sigma_F (R)$

Projeção

$\pi_{i_1, i_2, \dots, i_m}(R)$

Produto Cartesiano

$R \times S$

Junção

$R \bowtie S$

União

$R \cup S$

Interseção

$R \cap S$

Diferença

$R - S$

Visualizando as operações

Seleção

Projeção

--	--	--	--	--

Produto Cartesiano

A
B
C

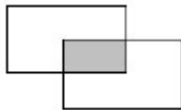
X
Y



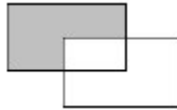
AX
AY
BX
BY
CX
CY

Junção

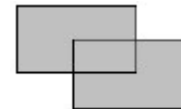
Interseção



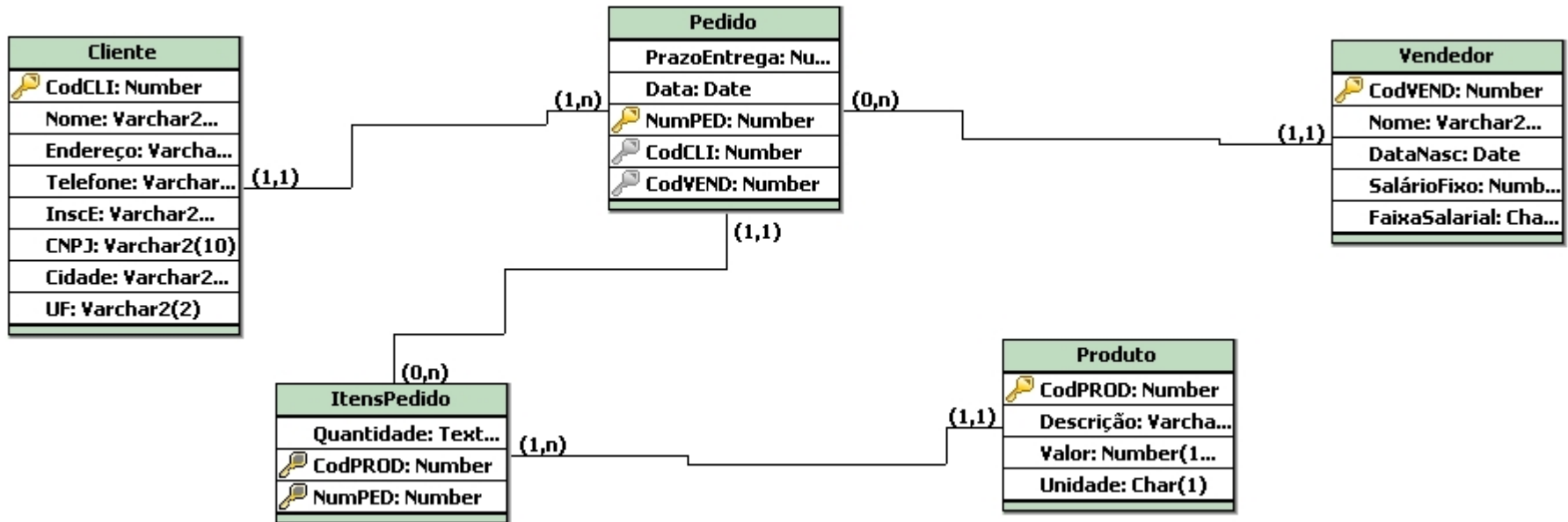
Diferença



União



Banco Pedidos



Exemplo: Álgebra Relacional e SQL

$\pi_{\text{nome}} (\sigma_{\text{cidade} = \text{'João Pessoa'}} (\text{cliente}))$

=

Select nome

From cliente

Where cidade = 'João Pessoa';

Exemplo: Álgebra Relacional e SQL

$$\pi_{(\text{nome_cli}, \text{UF})} (\sigma_{((\text{uf} = \text{'PB'} \vee \text{uf} = \text{'PE'}) \wedge (\text{prazoentrega} > 15)), (\text{cliente} \bowtie \text{Pedido}))}$$

Select nome, UF

from cliente **JOIN** pedido

on **cliente.codcli = pedido.codcli**

where UF in ('PB','PE') and prazoentrega > 15;

=> Quais clientes têm pedidos com prazo de entrega maior que 15 dias e são da Paraíba ou Pernambuco?

Exemplo: SQL e Álgebra Relacional

$\pi_{\text{nome_cli}} (\sigma_{\text{cidade} = \text{'Recife'}}) (\text{cliente}) \cup$
 $\pi_{\text{nome_vend}} (\text{vendedor})$

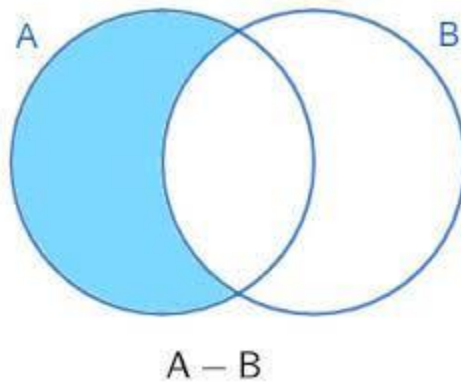
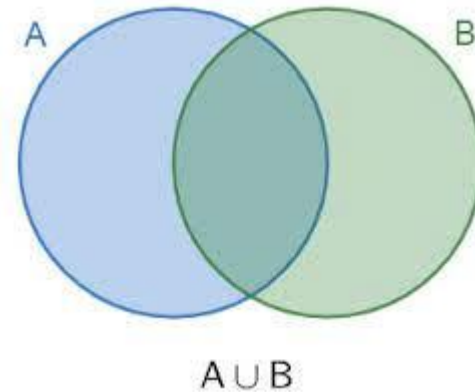
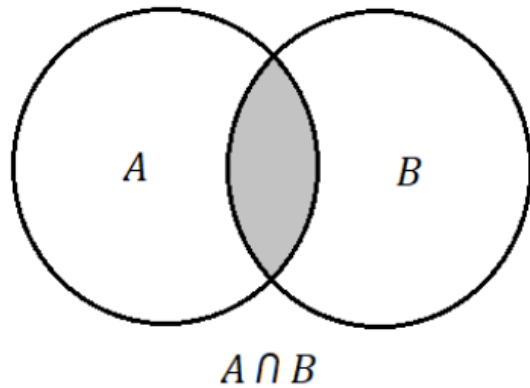
(Select nome
from cliente
where cidade like 'Recife')

UNION

(Select nome
from vendedor);

⇒ **Selecione os clientes cuja cidade seja 'Recife' e vendedores existentes.**

Operações de conjuntos



Operação de Conjunto - UNION

Tabela_q1 UNION [ALL] tabela_q2

Select nome

from cliente

where cidade like 'Recife'

UNION

Select nome

from vendedor;

q1

q2

	Data Output	Explain	Mess
	nome character varying (30)		
1	Maria Portela		
2	Josefa Cirino		
3	Ariane Dutra		
4	Claudia Dias		
5	Joao Peregrino		
6	Juan Gomes		
7	Cassandra Doura		

- O Union automaticamente elimina tuplas duplicadas
 - Union ALL mantém duplicadas

Operação de Conjunto - INTERSECT

```
tabela_q1 INTERSECT [ ALL] tabela_q2
```

```
select codcli  
from cliente  
where UF = 'PB'
```

q1

INTERSECT

```
select codcli  
from pedido;
```

q2

Data Output			E
	codcli integer		
1		2	
2		3	
3		5	

- O Intersect automaticamente elimina tuplas duplicadas
 - Intersect ALL mantém duplicadas

Operação de Conjunto – EXCEPT/MINUS

tabela_q1 **EXCEPT/MINUS** [ALL] tabela_q2

select codcli
from cliente

q1

Except

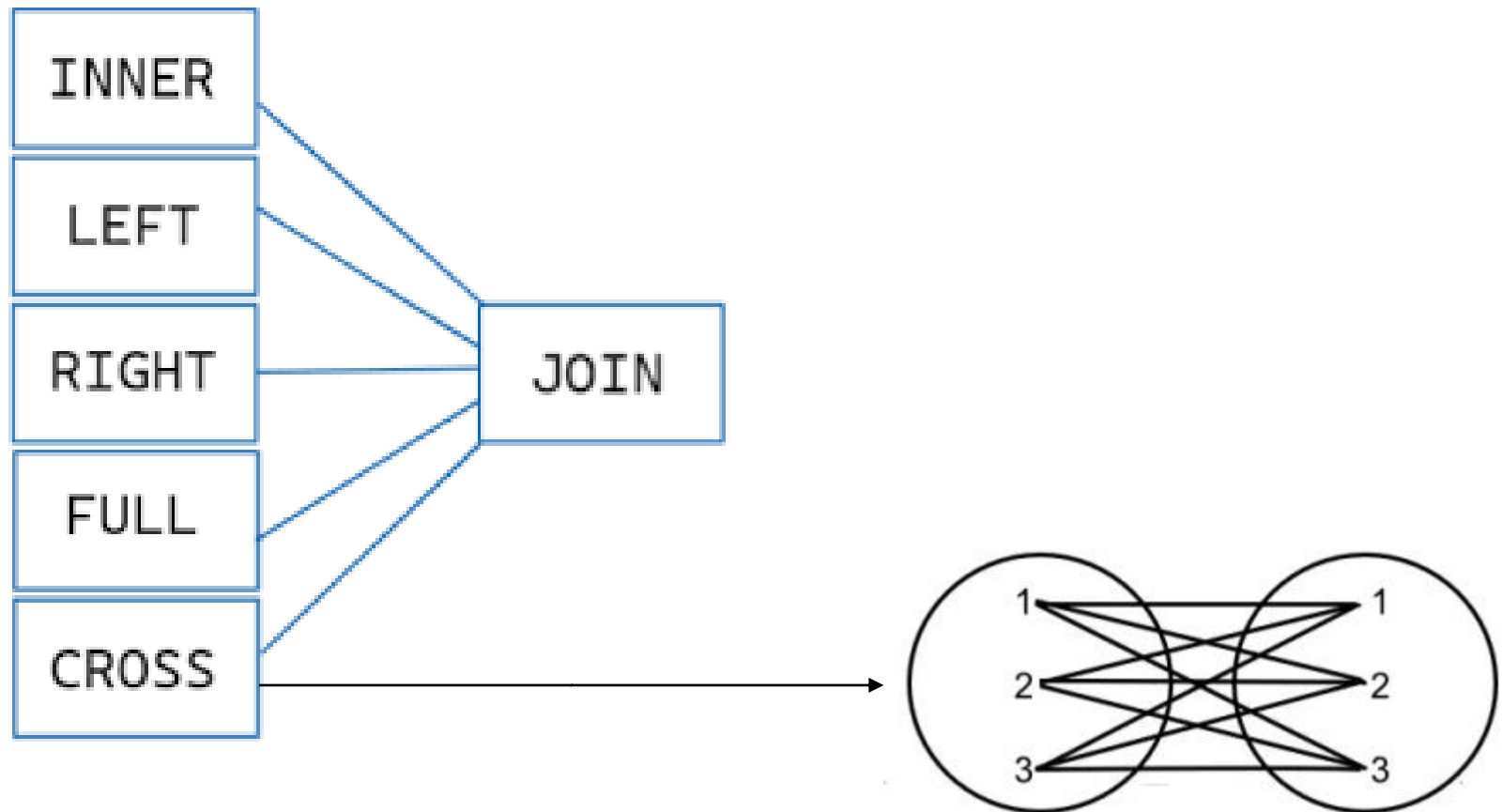
select codcli
from pedido;

q2

Data Output			E
	codcli integer		
1		9	
2		6	
3		7	
4		8	

- O Except/Minus automaticamente elimina tuplas duplicadas
 - Except/Minus ALL mantém duplicadas

Tipos de JOIN



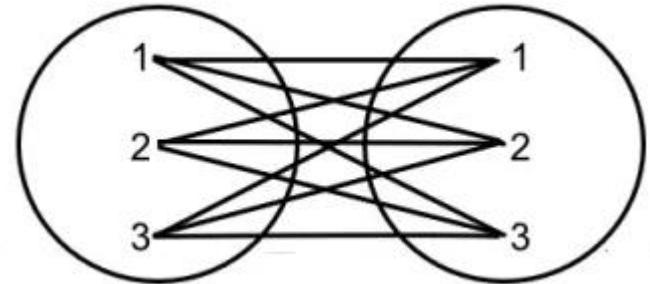
Cross JOIN

➤ Produto Cartesiano

Select **cliente.codcli**, nome,
numped, **pedido.codcli**
from **cliente**, **pedido**;

≡

Select **cliente.codcli**, nome,
numped, **pedido.codcli**
from cliente **cross JOIN** pedido;



	codcli integer	codcli integer	nome character varying (30)	numped integer
5	5	1	Ana Moura	1
6	6	1	Cassandra Doura	1
7	7	1	Cicero Novaes	1
8	8	1	Marcos Araruna	1
9	9	1	Marcia Araruna2	1
10	1	2	Claudia Dias	2
11	2	2	Joaquim Moraes	2
12	3	2	Janaina Rodrigues	2

54 linhas

Inner Join

- Consiste em **juntar** duas ou mais tabelas, ligando-as através da **Chave Primária** de uma e a **Chave Estrangeira** da outra
 - São apresentados os registros em que exista **ligação** entre as tabelas
 - Existe uma “**qualificação**”: Produto cartesiano + qualificação

Tabela1.campoPK = tabela2.campoFK

Inner Join

```
Select cliente.codcli, pedido.codcli,  
       nome, numped  
from cliente, pedido  
where cliente.codcli  
       = pedido.codcli;
```

Ou

```
Select cliente.codcli, pedido.codcli,  
       nome, numped  
from cliente JOIN pedido on  
       cliente.codcli = pedido.codcli;
```

codcli integer	codcli integer	nome character varying (30)	numped integer
1	1	Claudia Dias	1
2	2	Joaquim Moraes	2
3	3	Janaina Rodrigues	3
4	4	Maria Portela	4
5	5	Ana Moura	5
5	5	Ana Moura	6

6 linhas

Qualificação, apelidos/*aliases*

➤ Qualificadores de Nome:

- Produto.descricao
- Artista.nome

➤ Operação **renomear**: Produto p, Itens I

- P.descricao
- P.codprod = I.codprod

```
Select nome, p.numprod
```

```
From cliente c INNER JOIN pedido p
```

```
ON c.codcli = p.codcli;
```

Outro exemplo

Select v.nome

From vendedor v join pedido p

on v.codvend = p.codvend

join itenspedido i on p.numped = i.numped

join produto pr on i.codprod = pr.codprod

Where i.quantidade > 5 and pr.descricao = 'Chocolate';

O que a query retorna?

E essa???

```
Select cidade, count(*)  
from cliente C join pedido P on C.codcli =  
    P.codcli  
join vendedor V on P.codvend = V.codvend  
Group by cidade;
```

Grupos

Cidade	Numped	cliente	vendedor
João Pessoa	1	Mariana	Pedro
	6	João	Maria
	5	Joaquim	Pedro
	3	André	Ana
Recife	2	Valter	Maria
	4	Paulo	Pedro

Select **cidade**, count(*)
 from **cliente** C join **pedido** P
 on C.codcli = P.codcli
 join **vendedor** V on P.codvend = V.codvend
Group by cidade;



	Data Output	Explain	Messages	Not
	<div>cidade</div> <div>character varying (15)</div>		count	bigint
1	Recife			2
2	Joao Pessoa			4

Group by e having

```
select cliente.uf, count(*)  
from cliente  
group by uf  
having count(*) > 2;
```

```
select v.faixacomissao, avg(salariofixo)  
from vendedor v  
where v.faixacomissao <> 'B'  
group by v.faixacomissao  
having avg(salariofixo) > 3000;
```

Novo BD e tabelas

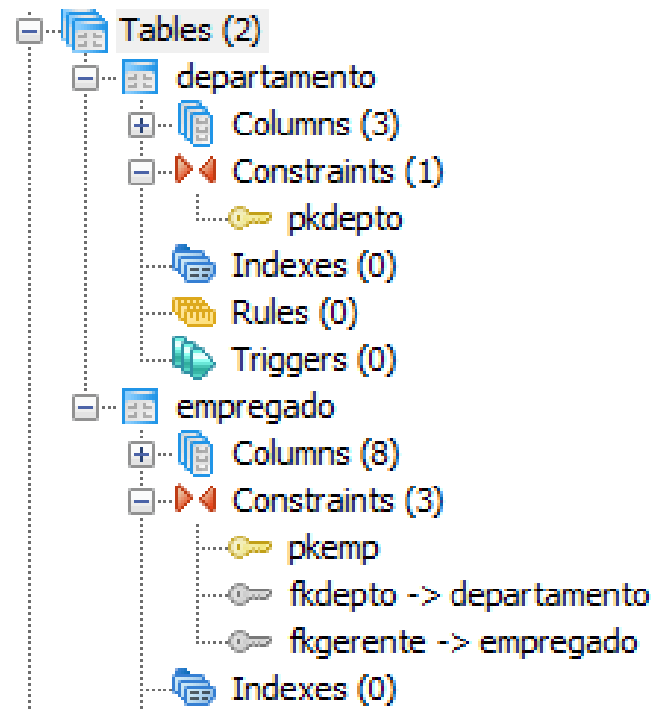
New!!!

Database: **Empregados**

Tabelas:

Departamento(CodDepto, Nome, Local)

Empregado (Matricula, PrimeiroNome, Sobrenome, Dataadmissao, Cargo, Salario, **gerente**, CodDepto)



**** Após criar o banco, rodar scripts de criação de tabelas e de povoamento.**

Consulta Inner Join

Select e.sobrenome as "Empregado", d.nome as "Departamento", e.dataadmissao as "Data de Admissão"

From empregado **e** **join** departamento **d**
on e.coddepto = d.coddepto;

O que a query retorna?

Tabela Empregado

PK

FK

matricula [PK] integer	primeironome character varying (15)	sobrenome character varying (15)	dataadmissao date	cargo character varying (30)	salario numeric (13,2)	gerente integer	coddepto integer
1	Jão	Guedes	2021-02-11	Analista de Sistemas Junior	3400.00	[null]	1
2	José	Batista	2021-02-11	Analista de Sistemas Pleno	4200.00	1	1
3	Ana Maria	Silva	2021-02-11	Analista de Sistemas Junior	3400.00	1	1
4	Ricardo	Neves	2021-02-11	Analista de Sistemas Pleno	4200.00	2	1
5	Valter	Moura	2021-02-11	Analista de Suporte Junior	3400.00	2	1
7	Mariana	Oliveira	2021-02-11	Designer de Interface	4800.00	1	[null]

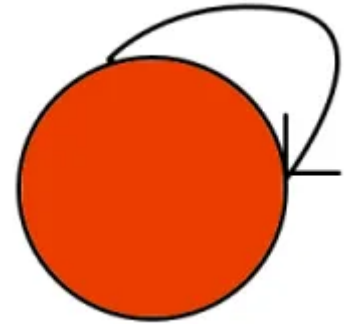
PK

FK

	matricula [PK] integer	primeironome character varying (15)	gerente integer
1	1	Jão	[null]
2	2	José	1
3	3	Ana Maria	1
4	4	Ricardo	2
5	5	Valter	2
6	7	Mariana	1

Self Join

- Utilizado quando uma tabela tem um relacionamento recursivo



Select e.primeironome as
"Empregado", g.primeironome as
"Gerente"

From (empregado e join
empregado g
on e.gerente = g.matricula);

	Data Output	Explain	Messages	Notifications
	Empregado character varying (15)		Gerente character varying (15)	
1	José		Jão	
2	Ana Maria		Jão	
3	Ricardo		José	
4	Valter		José	
5	Mariana		Jão	

OUTER JOIN

➤ LEFT OUTER JOIN

- Retorna as linhas comuns às duas tabelas mais o conjunto de linhas não comuns da tabela da esquerda

➤ RIGHT OUTER JOIN

- Retorna as linhas comuns às duas tabelas mais as linhas não comuns que existem na tabela da direita

Exemplos

Select d.nome as Departamento, e.primeironome as

Empregado

From departamento d **left**

on d.coddepto = e.coddepto

	departamento character varying (20)	empregado character varying (15)
1	Informática	Jão
2	Informática	José
3	Informática	Ana Maria
4	Informática	Ricardo

	departamento character varying (20)	empregado character varying (15)
1	Informática	Jão
2	Informática	José
3	Informática	Ana Maria
4	Informática	Ricardo
5	Informática	Valter
6	[null]	Mariana

left outer join empregado e

on d.coddepto = e.coddepto

order by d.nome;

FULL OUTER JOIN

Select d.nome as Departamento,
e.primeironome as Empregado

From departamento d **full join** empregado
e on d.coddepto = e.coddepto;

	 departamento character varying (20)	 empregado character varying (15)
1	Informática	Jão
2	Informática	José
3	Informática	Ana Maria
4	Informática	Ricardo
5	Informática	Valter
6	[null]	Mariana
7	Pessoal	[null]

Select d.nome as Departamento,
e.primeironome as Funcionario
From departamento d left join empregado e
on d.coddepto = e.coddepto
Where e.coddepto is null
Order by d.nome;

O que a query retorna?