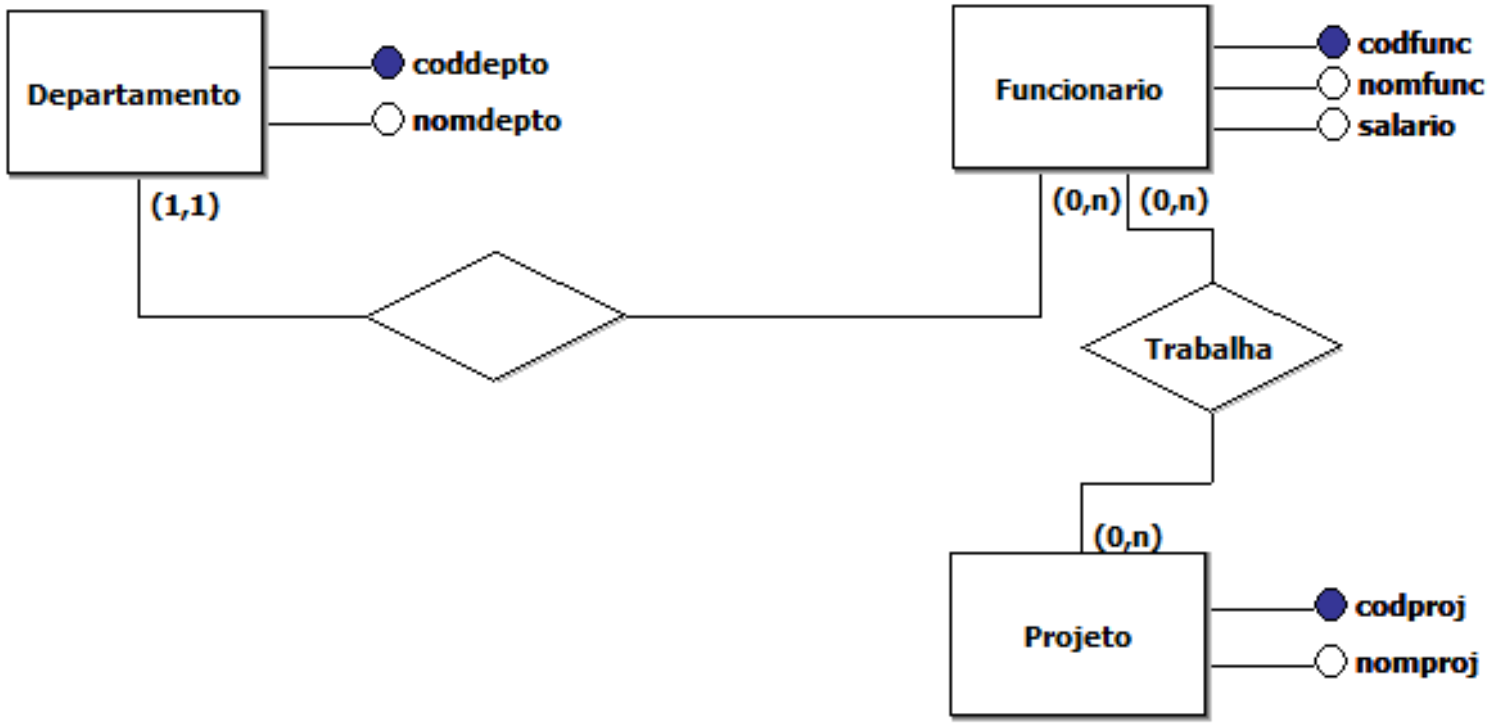
SQL & Álgebra Relacional

- *Junção*
- *Junção Externa*
- *União*
- *Intersecção*
- *Diferença*
- *Divisão*
- *Non-equi join*
- *Renomear*
- *Agrupamento*

Modelo


Departamento
coddepto
nomdepto
(1,1)
Funcionario
codfunc
nomfunc
salario
(0,n)
(0,n)
Trabalha
(0,n)
Projeto
codproj
nomproj

SQL

```
insert into departamento values (1,'d1');
insert into departamento values (2,'d2');
insert into departamento values (3,'d3');

insert into funcionario values
(1,'Joao',2,'1000.00');
insert into funcionario values
(2,'Maria',1,'3000.00');
insert into funcionario values
(3,'Jose',2,'4000.00');
insert into funcionario values
(4,'Ana',NULL,'3000.00');
```

```
insert into projeto values (1,'p1');
insert into projeto values (2,'p2');
insert into projeto values (3,'p3');

insert into trabalha values (1,1);
insert into trabalha values (1,2);
insert into trabalha values (1,4);
insert into trabalha values (2,1);
insert into trabalha values (2,2);
insert into trabalha values (3,1);
insert into trabalha values (3,4);
```

Junção

– Produto cartesiano seguido de seleção...

$|\times|$

<tabela1> $|\times|$ <condição de junção> <tabela2>

Funcionário $|\times|$ <funcionário.coddepto = departamento.coddepto> Departamento

```
Select  *
        From funcionário inner join departamento
                        On funcionario.coddepto = departamento.coddepto;
```

Álgebra - EquiJunção

$$\langle tabela1 \rangle *_{(\langle lista1 \rangle),(\langle lista2 \rangle)} \langle tabela2 \rangle$$

$$Funcionário *_{(coddepto),(coddepto)} Departamento$$

Junção Natural

<tabela1> * <tabela2>

Funcionário * Departamento

```
select * from
        departamento natural inner join funcionario;
```

Melhor não usar!!! https://www.postgresqltutorial.com/postgresql-natural-join/

Junção Externa - Esquerda

Símbolos:

=*	left outer join
*=	right outer join
=*=	full outer join

Exemplos:

Funcionario =* (coddepto), (coddepto) departamento

```
Select * from funcionario left join departamento
        On funcionario.coddepto = departamento.coddepto;
```

Junção Externa – Direita

Símbolos:

=*	left outer join
*=	right outer join
=*=	full outer join

Funcionario *= (coddepto),(coddepto) departamento

```
Select * from funcionario right join departamento
        On funcionario.coddepto = departamento.coddepto;
```

Junção Externa - Full

Símbolos:

=*	left outer join
*=	right outer join
=*=	full outer join

Funcionario =*= (coddepto),(coddepto) departamento

```
Select * from funcionario full join departamento
        On funcionario.coddepto =departamento.coddepto;
```

União

Reunir todos elementos de 2 conjuntos. Por exemplo, mostrar os códigos dos funcionários do departamento 2 mais os funcionários que trabalham em algum projeto, não importando o departamento

Sintaxe: <tabela1> $\cup$ <tabela2>

Na álgebra pode-se fazer uso de tabelas temporárias

Exemplo:

$TEMP1 \Leftarrow \pi$ codfunc (σ coddepto = 2 (funcionario))

$TEMP2 \Leftarrow \pi$ codfunc (trabalha)

$TEMP1 \cup TEMP2$

Em SQL:

```
Select codfunc from funcionario where coddepto = 2
Union
Select codfunc from trabalha;
```

Intersecção

Reunir apenas os elementos que participem de ambos os conjuntos. Por exemplo, mostrar os funcionários que trabalham em um departamento e esteja trabalhando em algum projeto

Sintaxe: <tabela1> $\cap$ <tabela2>

Exemplo:

TEMP1 $\Leftarrow \pi$ codfunc (σ coddepto = 2 (funcionario))
TEMP2 $\Leftarrow \pi$ codfunc (trabalha)

TEMP1 $\cap$ TEMP2

Em SQL:

```
Select codfunc from funcionario where coddepto = 2
Intersect
Select codfunc from trabalha
```

Ou

```
Select codfunc from funcionario where coddepto = 2
        and exists  (select * from trabalha
                where trabalha.codfunc = funcionario.codfunc);
```

Diferença

Reunir apenas os elementos que pertencem a um conjunto mas não pertencem ao outro, por exemplo mostrar os funcionário que trabalham em um departamento, mas não trabalham em nenhum projeto.

Sintaxe: <tabela1> - <tabela2>

Exemplo:

TEMP1 ⇐ π codfunc (σ coddepto = 2 (funcionario))
TEMP2 ⇐ π codfunc (trabalha)

TEMP1 - TEMP2

Em SQL:

No PostgresQL no lugar de minus usar except

```
Select codfunc from funcionario where coddepto = 2
minus
Select codfunc from funcionario
    where exists (select * from trabalha where trabalha.codfunc = funcionario.codfunc);
```

OU

```
Select codfunc from funcionario where coddepto =2
    and not exists (select * from trabalha where trabalha.codfunc = funcionario.codfunc);
```

Divisão

Resolve problemas do tipo: Obter os funcionários que trabalhem em **todos** os projetos
Tem duas tabelas como operandos
Os nomes e domínios da tabela2 devem estar contidos na tabela1
O resultado é uma tabela com as colunas e domínios que aparecem na tabela1 mas não aparecem na tabela 2.
Para que esta linha apareça no resultado é necessário sua concatenação com cada linha da tabela2 apareça também na tabela1.

Divisão

projeto

Codproje	Nomproje
1	p1
2	p2
3	p3

trabalha

Codproje	Codfunc
1	1
1	2
1	4
2	1
2	2
3	1
3	4

Sintaxe: <tabela1> ÷ <tabela2>

Exemplo:

TEMP1 ⇐ π funcionario.codfunc, nomfunc, codproje
(σ funcionario.codfunc = trabalha.codfunc
(trabalha X funcionario))

codfunc	Nomfunc	codproje
1	Joao	1
2	Maria	2
1	Joao	2
1	Joao	3
2	Maria	1
4	Ana	1
4	Ana	3

TEMP2 ⇐ π codproje (projeto)

TEMP1 ÷ TEMP2

Divisão

```
select funcionario.codfunc, nomfunc from funcionario
        where not exists
                (select * from projeto
                        where not exists
                                (select * from trabalha
                                        Where trabalha.codproj = projeto.codproj
                                        and
                                        funcionario.codfunc = trabalha.codfunc));
```

Non-equi join

Utilizada quando no caso de um join entre duas tabelas, não existe uma coluna em uma tabela1 que corresponda diretamente a uma coluna de uma tabela2

π codfaixa,desfaixa,salmin,salmax,nomfunc,salario
(funcionário (X) (salario >= salmin and salario <= salmax) faixasal)

```
select codfaixa,desfaixa,salmin,salmax,nomfunc,salario
        from funcionario,faixasal
        where funcionario.salario >= salmin and funcionario.salario <= salmax;
```

Codfaixa	desfaixa	salmin	salmax	nomfunc	salario
1	baixo	0	1000	joao	1000
3	alto	2001	3000	maria	3000
3	alto	2001	3000	ana	3000
4	muito alto	3001	10000	jose	4000

Renomear

Símbolo: ρ (rô)

Sintaxe: ρ <novo_nome> (<tabela>)

Self-join

codfunc	nomfunc	codchefe
1	Joao	2
2	Maria	4
3	jose	2
4	ana	

Exemplo:

π chefe.nomfunc, funcionario.nomfunc
(σ funcionario.codchefe = chefe.codfunc
(funcionario X ρ chefe (funcionario))

```
select chefe.nomfunc as nomchefe ,funcionario.nomfunc
    from funcionario,funcionario as chefe
    where funcionario.codchefe = chefe.codfunc;
```

nomchefe	nomfunc
maria	Joao
ana	Maria
maria	Jose

Agrupamento

```
SELECT [DISTINCT]  * | coluna(s) | funções
        FROM
        TABELA(S)
        [WHERE expressão]
        [GROUP BY expressão]
        [HAVING expressão]
        [ORDER BY critério]
```

Agrupamento

```
SELECT coddepto, count(*)
        FROM
        FUNCIONARIO
        GROUP BY coddepto
```

Agrupamento

```
SELECT coddepto, count(*)
        FROM
        FUNCIONARIO
        GROUP BY coddepto
        HAVING COUNT(*) > 1
```