



PCS3413

Engenharia de Software e Banco de Dados

Aulas SQL – parte 1

Vamos usar o <https://sqlbolt.com/> antes de iniciarmos com Postgre

SQL – STRUCTURED QUERY LANGUAGE

SQL

- Álgebra Relacional
- Definição de dados
 - DDL – Data Definition Language
- Manipulação de dados
 - DML – Data Manipulation Language

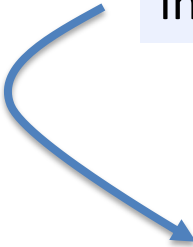
Manipulação de dados

- operações de modificam o estado das tabelas:
 - INSERT
 - DELETE
 - UPDATE
- Operações que acessam (recuperam) dados de tabelas:
 - SELECT


Inserção de dados

```
INSERT INTO nometab values (valor-campo1, valor-campo2, ..., valor-campoN)
```

```
insert into Produtos values (1, 'queijo', 9.99);
```



numProduto	nomeProduto	precoProduto
1	queijo	9.99



```
create table Produtos  
(numProduto numeric (3),  
nomeProduto varchar (30),  
preçoProduto numeric (9,2),  
Primary Key (numProduto) );
```

Inserção de dados - continuação

```
INSERT INTO nometab (nome-campo1, ..., nome-campon) values (valor-campo1, ..., valor-campon)
```



quando a tabela tem vários campos e quero intruzir valores apenas para alguns dos campos

```
insert into Produtos (numProduto, nomeProduto, preçoProduto) values (1, 'queijo', 9.99);
```

```
insert into Produtos (nomeProduto, preçoProduto, numProduto) values ( 'queijo', 9.99, 1);
```

inserindo valores apenas para os campos número e nome:

Inserção de dados - continuação

a) comando completo

```
INSERT INTO Produtos (numProduto, nomeProduto) VALUES (2, 'presunto');
```

numProduto	nomeProduto	precoProduto
1	queijo	9.99
2	presunto	

difícil: imagine se a tabela tem vários campos(ex. 10) e no insert apenas para alguns está sendo definido valor não nulo

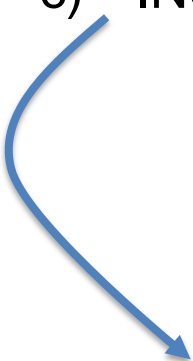
b) comando resumido

```
INSERT INTO Produtos VALUES (2, 'presunto', null);
```

Inserção de dados - continuação

```
create table Produtos  
(numProduto numeric (3),  
nomeProduto varchar(30),  
preçoProduto numeric (9,2) default 11.5,  
primary key (numProduto));
```

c) INSERT INTO Produtos VALUES (3, 'salame');



numProduto	nomeProduto	preçoProduto
1	queijo	9.99
2	presunto	
3	salame	11.5

Remoção de dados

DELETE FROM nometab

Produtos

numProduto	nome	preço
1	queijo	8.00
2	presunto	11.50
3	salame	11.50
4	copa	18.00

Delete from Produtos;

todas as linhas são removidas

DELETE FROM nometab

WHERE condição

Delete from Produtos
where preço = 11.50;

numProduto	nomeProduto	preçoProduto
1	queijo	8.00
4	copa	18.00

Modificação de dados

```
UPDATE nometab  
SET nomecampo = novovalor  
WHERE condição
```

Produtos

numProduto	nomeProduto	preçoProduto
1	queijo	8.00
2	presunto	11.50
3	salame	11.50
4	copa	18.00

dar aumento de 10% a todos os produtos.

```
update Produtos  
set preçoProduto= preçoProduto *1.10;
```

Produtos

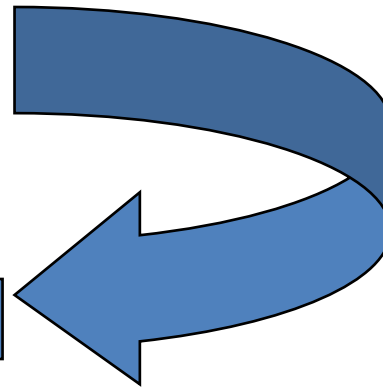
numProduto	nomeProduto	preçoProduto
1	queijo	8.80
2	presunto	12.65
3	salame	12.65
4	copa	19.80

mude o nome do produto 1 para queijo branco e aumente seu valor em mais 0,20.

```
update Produtos  
set preçoProduto = preçoProduto + 0.20, nomeProduto = 'queijo branco'  
where numProduto = 1
```

Produtos

numProduto	nomeProduto	preçoProduto
1	queijo branco	9.00
2	presunto	12.65
3	salame	12.65
4	copa	19.80



OPERAÇÕES QUE ACESSAM (RECUPERAM) DADOS
DE TABELAS

Busca em Tabelas

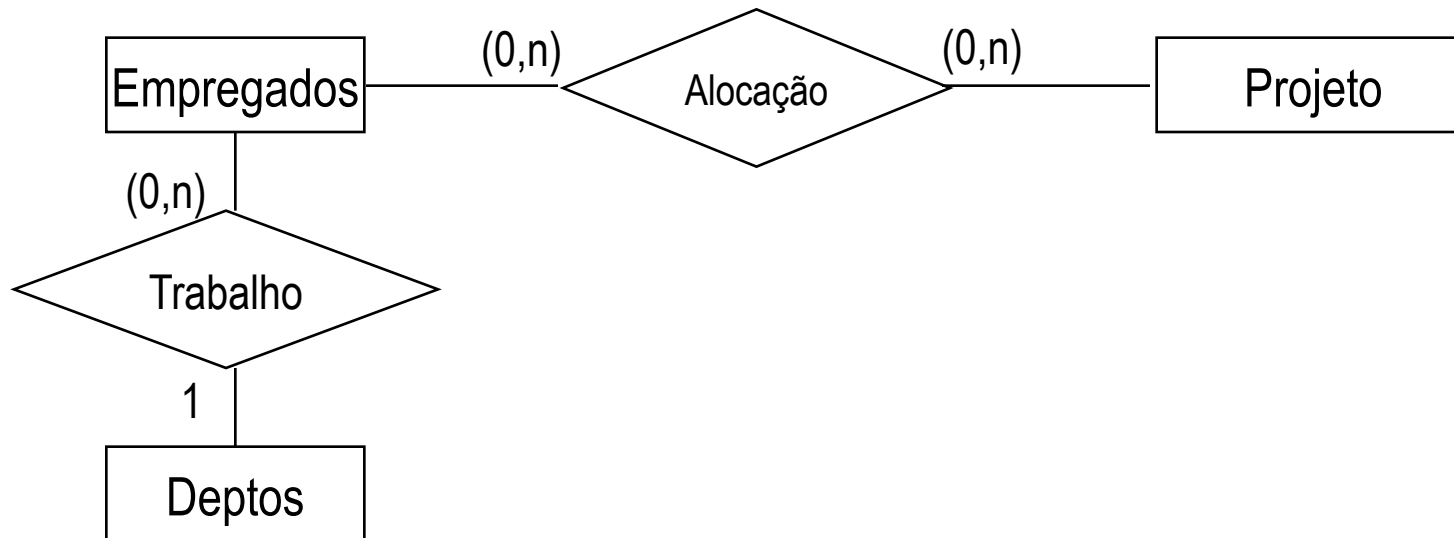
consulta típica em SQL:

select A_1, A_2, \dots, A_n

from r_1, r_2, \dots, r_n

where P

Esquema de Banco de Dados



Deptos (Cod_Depto, Nome_Depto, Região_Depto)

Empregados (Id_Emp, Nome, Sobrenome, Salário, Cod_Depto)

Projeto (Sigla_Proj, Descr_Proj, DataIni_Proj, DataFim_Proj)

Alocação (Id_Emp, Sigla_Proj)

Busca Simples

```
select campo1, campo2, ..., campoN  
from nome-table
```

para trazer todos os campos da tabela na cláusula from use * ou para trazer apenas alguns dos atributos, indique quais.

Deppto

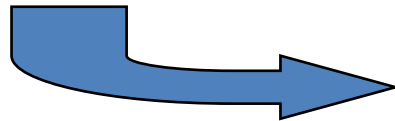
Cod_Depto	Nome_Depto	Região_Depto
41	Recursos Humanos	Osasco
42	Recursos Humanos	São Paulo
43	Informática	São Paulo
44	Marketing	São Paulo

1)
select *
from Deppto

ou

```
select cod_depto, nome_depto, região_depto  
from Deppto
```

```
select cod_depto, nome_depto  
from Depto
```



Cod_Depto	Nome_Depto
41	Recursos Humanos
42	Recursos Humanos
43	Informática
44	Marketing

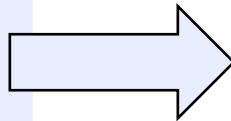
Alocação

Id_Emp	Sigla_Proj
0962	beta
0942	beta
0943	zebra
0962	alfa

Listar todos os projetos nos quais existem empregados alocados.

2)

```
select  Sigla_Proj  
from    Alocação
```



Sigla_Proj
beta
beta
zebra
alfa

Eliminar linhas com valores repetidos
no resultado da consulta

```
select  [distinct] nome-campo  
from    nome-table
```

3)

```
select distinct Sigla_Proj  
from    Alocação
```

Sigla_Proj
beta
zebra
alfa

Operadores de Comparação

=	igual
<>	diferente
>	maior
<	menor
>=	maior ou igual
<=	menor ou igual

Busca Qualificada

```
select campo1,..., campon  
from nome-tabela  
where <condição>
```

para trazer todos os campos da tabela na cláusula from use *
ou para trazer apenas alguns dos atributos, indique quais.

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

Listar o Identificador dos empregados que ganham acima de 2500

4)

```
select Id_Emp  
from Empregados  
where Salário > 2500
```

Id_Emp
0942
0944

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

5)

```
select Id_Emp
from   Empregados
where  Salário > 2500 and Cod_Depto = 42
```

Id_Emp
0942

6)

```
select Sobrenome, Nome, Salário
from   Empregados
where  Salário >= 2500 and Cod_Depto <> 42
```

Sobrenome	Nome	Salário
Santos	João	2500
Traviatta	Helena	3000

Operadores de Comparação para campos string

- LIKE

“Info%”	corresponde a todas strings que iniciam por Info
“% ati%”	corresponde a todas strings que possuem ati no meio
“ _ _ _ ”	corresponde a todas strings com três caracteres
“ _ _ % ”	corresponde a todas strings com pelo menos três caracteres

Deppto

Cod_Depto	Nome_Depto	Região_Depto
41	Recursos Humanos	Osasco
42	Recursos Humanos	São Paulo
43	Informática	São Paulo
44	Marketing	São Paulo

12)

```
select *  
from Deppto  
where Nome_Depto like 'Info%'
```

Cod_Depto	Nome_Depto	Região_Depto
43	Informática	São Paulo

13)

```
select *  
from Deppto  
where Região_Depto not like '_ão Paulo'
```

Cod_Depto	Nome_Depto	Região_Depto
41	Recursos Humanos	Osasco

Cláusula as

- para renomear atributos e tabelas

Não aceito em alguns gerenciadores e opcionais em outros

14)

```
select Cod_Depto as Número, Nome_Depto as Nome  
from Depto as D  
where D.Nome_Depto like '%R%';
```

14)

```
select Cod_Depto Número, Nome_Depto Nome  
from Depto D  
where D.Nome_Depto like '%R%';
```

Número	Nome
41	Recursos Humanos
42	Recursos Humanos

Cláusula Order By

- ordenação ascendente (default) ou descendente

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

15)

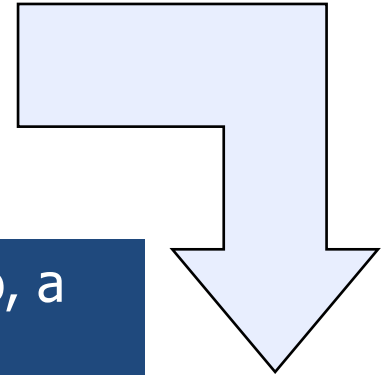
```
select Id_Emp as Emp, Sobrenome  
from Empregados  
where salário >= 2500 and Cod_Depto = 42  
order by Id_Emp desc
```

Emp	Sobrenome
0962	Alves
0942	Jordão

16)

```
select  Sobrenome, Cod_Depto
from    Empregados
where   salário > 2000
order by Cod_Depto, Sobrenome desc
```

se vários empregados trabalham no mesmo Depto, a segunda ordenação é obedecida



Sobrenome	Cod_Depto
Jordão	42
Alves	42
Traviatta	43
Santos	44

PARTE 2

Mais de uma tabela

17)

```
select Nome_Depto, Sobrenome  
from Depto, Empregados  
where Depto.Cod_Depto = Empregados. Cod_Depto
```

Depto ⊗ Empregados

Depto. Cod_Depto	Nome_Depto	Região_Depto	Id_Emp	Nome	Sobrenome	Salário	Empregados. Cod_Depto
41	Recursos Humanos	Osasco	0962	João	Alves	2500	42
41	Recursos Humanos	Osasco	0942	Ana Maria	Jordão	2550	42
41	Recursos Humanos	Osasco	0943	João	Santos	2500	44
41	Recursos Humanos	Osasco	0944	Helena	Traviatta	3000	43
42	Recursos Humanos	São Paulo	0962	João	Alves	2500	42
42	Recursos Humanos	São Paulo	0942	Ana Maria	Jordão	2550	42
42	Recursos Humanos	São Paulo	0943	João	Santos	2500	44
42	Recursos Humanos	São Paulo	0944	Helena	Traviatta	3000	43
44	Marketing	São Paulo	0944	Helena	Traviatta	3000	43

Depto. Cod_Depto	Nome_Depto	Região_Depto	Id_Emp	Nome	Sobrenome	Salário	Empregados. Cod_Depto
42	Recursos Humanos	São Paulo	0962	João	Alves	2500	42
42	Recursos Humanos	São Paulo	0942	Ana Maria	Jordão	2550	42
43	Informática	São Paulo	0944	Helena	Traviatta	3000	43
44	Marketing	São Paulo	0943	João	Santos	2500	44

apenas Nome_Depto e Sobrenome

Nome_Depto	Sobrenome
Recursos Humanos	Alves
Recursos Humanos	Jordão
Informática	Traviatta
Marketing	Santos

Junção

- forma o produto cartesiano de seus dois argumentos (relações) e efetuam a seleção obedecendo a equivalência dos atributos que aparecem em ambas relações.
 - Junção Natural – remove atributos em duplicidade
 - Junção Teta – não remove atributos em duplicidade.

Junção (Join)

a) Junção Natural – natural join

```
18)
select *
from Depto natural join Empregados
```

Cod_Depto	Nome_Depto	Região_Depto	Id_Emp	Nome	Sobrenome	Salário
42	Recursos Humanos	São Paulo	0962	João	Alves	2500
42	Recursos Humanos	São Paulo	0942	Ana Maria	Jordão	2550
43	Informática	São Paulo	0944	Helena	Traviatta	3000
44	Marketing	São Paulo	0943	João	Santos	2500

Junção - continuação

b) junção Teta

corresponde a junção natural, sendo que a condição deve ser definida.

```
19)
select *
from Depto inner join Empregado on
Depto.Cod_Depto = Empregado.Cod_Depto
```

Cod_Depto	Nome_Depto	Região_Depto	Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
42	Recursos Humanos	São Paulo	0962	João	Alves	2500	42
42	Recursos Humanos	São Paulo	0942	Ana Maria	Jordão	2550	42
43	Informática	São Paulo	0944	Helena	Traviatta	3000	43
44	Marketing	São Paulo	0943	João	Santos	2500	44

resultado mantém campos repetidos



mais Junções

- Junção externa (outer join)
 - uma extensão da operação de junção para tratar informações omitidas.
 - as linhas que não têm correspondência nas duas tabelas são mantidas no resultado, porém associa-se nulo (null) à informação desconhecida para aquela linha.
- Junção externa à esquerda (left outer join)
 - mantém todas as linhas da tabela à esquerda e completa com nulos as informações desconhecidas.
- Junção externa à direita (right outer join)
 - mantém todas as linhas da tabela à direita e completa com nulos as informações desconhecidas.
- Junção externa total (full outer join)
 - mantém todas as linhas das duas tabelas e completa com nulos as informações desconhecidas.

Depto

Cod_Depto	Nome_Depto	Região_Depto
41	Recursos Humanos	Osasco
42	Recursos Humanos	São Paulo
43	Informática	São Paulo
44	Marketing	São Paulo

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

20)

select *

from Depto natural left outer join Empregados

Cod_Depto	Nome_Depto	Região_Depto	Id_Emp	Nome	Sobrenome	Salário
41	Recursos Humanos	Osasco				
42	Recursos Humanos	São Paulo	0962	João	Alves	2500
42	Recursos Humanos	São Paulo	0942	Ana Maria	Jordão	2550
43	Informática	São Paulo	0944	Helena	Traviatta	3000
44	Marketing	São Paulo	0943	João	Santos	2500

Junção externa – continuação Right Outer Join

- Right outer join

o esquema mostra a relação de empregado e empregados em tempo integral:

empregado (nomeEmp, rua, cidade)

trabalhadorTI (nomeEmp, nomeAg, salário)

empregado

nomeEmp	rua	cidade
João	Joinha	Osasco
Roberto	Alvarenga	São Paulo
Sérgio	Primícia	Taubaté
Antônio	Selva	Guarulhos

trabalhadorTI

nomeEmp	nomeAg	salário
João	Osasco	1500
Roberto	Osasco	1300
Ana	Pinheiros	5300
Antônio	Pinheiros	1500

21)

select *

from empregado natural right outer join trabalhadorTI

nomeEmp	rua	cidade	nomeAg	salário
João	Joinha	Osasco	Osasco	1500
Roberto	Alvarenga	São Paulo	Osasco	1300
Antônio	Selva	Guarulhos	Pinheiros	1500
Ana			Pinheiros	5300

junção externa – continuação Full Outer Join

- full outer join

```
22)
select *
from empregado natural full outer join trabalhadorTI
```

nomeEmp	rua	cidade	nomeAg	salário
João	Joinha	Osasco	Osasco	1500
Roberto	Alvarenga	São Paulo	Osasco	1300
Antônio	Selva	Guarulhos	Pinheiros	1500
Sérgio	Primícia	Taubaté		
Ana			Pinheiros	5300

Junção com using

- condição da junção com using é similar à da junção natural, exceto que os atributos da junção são os A_1, A_2, \dots, A_n (argumento de using) em vez de todos os atributos comuns a ambas relações. Os atributos A_1, A_2, \dots, A_n devem ser somente os comuns a ambas relações e aparecem apenas uma vez no resultado da junção.

Junção com using - continuação

```
23)
select *
from Depto join Empregados using (Cod_Depto)
```

Cod_Depto	Nome_Depto	Região_Depto	Id_Emp	Nome	Sobrenome	Salário
42	Recursos Humanos	São Paulo	0962	João	Alves	2500
42	Recursos Humanos	São Paulo	0942	Ana Maria	Jordão	2550
43	Informática	São Paulo	0944	Helena	Traviatta	3000
44	Marketing	São Paulo	0943	João	Santos	2500

Resumo Tipo de Junções e Condições em SQL

Tipos de Junção	condições da junção
inner join left outer join right outer join full outer join	natural on <condição> using (A ₁ , A ₂ , ..., A _n)

- ◆ inner join corresponde a junção teta
- ◆ outer join corresponde a junção externa
- ◆ a condição é obrigatória para junções externas e opcional para interna (neste caso, corresponde ao produto cartesiano)
- ◆ as palavras inner e outer são opcionais
- ◆ natural aparece antes do tipo de junção
- ◆ on e using aparecem ao final da expressão de junção

SQL

continuação

(in, not in, consultas aninhadas, group by e
having)

Parte 3

Tabelas: Depto, Alocação e Empregados

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

Alocação

Id_Emp	Sigla_Proj
0962	beta
0942	beta
0943	zebra
0962	alfa

Depto

Cod_Depto	Nome_Depto	Região_Depto
41	Recursos Humanos	Osasco
42	Recursos Humanos	São Paulo
43	Informática	São Paulo
44	Marketing	São Paulo

mais operadores de comparação

- **in** - igual a algum membro da lista que segue

Listar sobrenome, nome e salário de todos os empregados que trabalham no departamento 43 ou 44

24)

```
select Sobrenome, Nome, Salário  
from Empregados  
where Cod_Depto = 43 or Cod_Depto = 44
```

Sobrenome	Nome	Salário
Santos	João	2500
Traviatta	Helena	3000

25)

```
select Sobrenome, Nome, Salário  
from Empregados  
where Cod_Depto in (43, 44)
```

mais operadores - continuação

- **not in** - diferente de todos os membros da lista

Listar sobrenome, nome e salário de todos os empregados que não trabalham no departamento 43 ou 44

26)

```
select Sobrenome, Nome, Salário  
from Empregados  
where Cod_Depto <>43 or Cod_Depto <>44
```

Sobrenome	Nome	salário
Alves	João	2500
Jordão	Ana Maria	2550
Santos	João	2500
Traviatta	Helena	3000

27) Não trabalham nos depts 43 e 44

```
select Sobrenome, Nome, Salário  
from Empregados  
where Cod_Depto not in (43, 44)
```

Sobrenome	Nome	Salário
Alves	João	2500
Jordão	Ana Maria	2550

Consultas aninhadas (subQuery)

Listar o identificador e sobrenome de todos os empregados que trabalham no departamento de Informática

28)

```
select Id_Emp, Sobrenome
from Empregados
where Cod_Depto in (select Cod_Depto
                      from Depto
                      where Nome_Depto = 'Informática');
```

Cod_Depto
43



Id_Emp	Sobrenome
0944	Traviatta

Ex. com delete

remova todos os empregados que não estejam alocados a nenhum projeto.

```
delete from Empregados  
where id_emp not in (select id_emp  
                    from Alocação)
```

Empregados

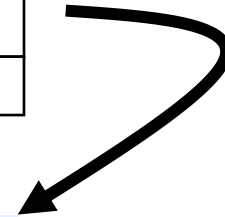
Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

Alocação

Id_Emp	Sigla_Proj
0962	beta
0942	beta
0943	zebra
0962	alfa

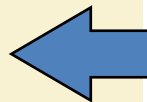
mais operadores - continuação

any	similar a in
some	similar a in
all	compara com todos da lista



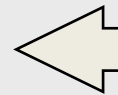
usados junto =, <>, >, <, >=, <=

= some
= any



mesmo que
in

<> all

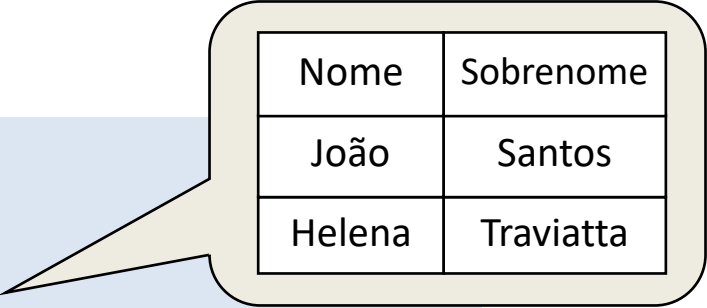


mesmo que
not in

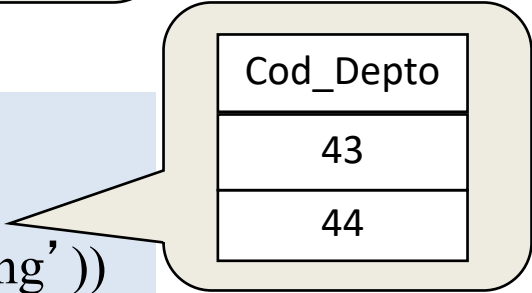
mais exemplos - continuação

Listar todos os empregados que trabalham nos departamentos Informática ou Marketing e que ganham acima de 2000. Apresentar no resultado nome e sobrenome.

```
29)
select Nome, Sobrenome
from Empregados
where salário > 2000 and Cod_Depto = any
( select Cod_Depto
  from Depto
  where nome_Depto in ( 'Informática', 'Marketing' ) )
```



Nome	Sobrenome
João	Santos
Helena	Traviatta



Cod_Depto
43
44

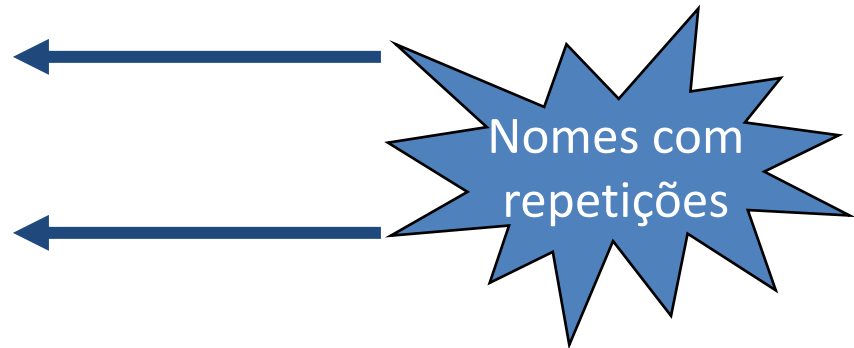
União (U)

- dado duas relações 'r' e 's':
 - número de atributos devem ser iguais
 - domínio do i-ésimo atributo de r deve ser igual ao do i-ésimo atributo de s

União - continuação

Ex_Empregados

Nome	Sobrenome	Cod_Depto
Maria	Alves	42
Ana Maria	Jordão	42
Maria	Alves	44
Helena	Almeida	43



Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

União - continuação

- Liste o nome de todos os empregados, incluindo daqueles que já não pertencem mais a empresa.

```
select nome  
from Empregados  
UNION  
select nome  
from Ex_Empregados
```

Ex_Empregados

Nome	Sobrenome	Cod_Depto
Maria	Alves	42
Ana Maria	Jordão	42
Maria	Alves	44
Helena	Almeida	43

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0965	Willian Antonio	Trigueiro	2500	44
0944	Helena	Traviatta	3000	43

União não tem
valores
repetidos

Nome
Maria
Ana Maria
João
Helena
Willian Antonio

Diferença (-)

- dado duas relações 'r' e 's':
 - número de atributos devem ser iguais
 - domínio do i-ésimo atributo de r deve ser igual ao do i-ésimo atributo de s

Listar todos os empregados que não estão alocados a nenhum projeto.

```
select id_emp  
from Empregados  
except  
select id_emp  
from Alocação
```

repetições são
eliminadas
do resultado

	Id_emp
1	965
2	944

ou

```
select Id_Emp  
from Empregados  
where Id_Emp not in (select Id_Emp from Alocação)
```

Funções Agregadas

- tomam uma coleção de valores como entrada e retornam um único valor
 - Média (average) : avg
 - Mínimo (minimum): min
 - Máximo (maximum): max
 - Total (total): sum
 - Contagem (count): count

Calcular a média de salários.

```
30)
select avg (Salário)
from Empregados;
```

avg(Salário)

2637,5

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

funções agregadas - continuação

Listar nome e sobrenome dos empregados que ganham acima da média de salários.

31)

```
select  Nome, Sobrenome
from    Empregados
where   Salário > (select avg (Salário)
                    from Empregados);
```

avg(Salário)

2637,5

Nome	Sobrenome
Helena	Traviatta

Dar um aumento de 5% para todos os empregados que ganham abaixo da média salarial da empresa.

```
update Empregados
set Salário = Salário * 1,05
where Salário < (select avg (Salário)
                  from Empregados)
```

avg(Salário)

2637,5

Empregados



Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2625	42
0942	Ana Maria	Jordão	2677,5	42
0943	João	Santos	2625	44
0944	Helena	Traviatta	3000	43

Apresente o sobrenome e o salário dos que ganham o maior salário na empresa.

32)

```
select Sobrenome, Salário  
from Empregados  
where Salário = all (select max(Salário)  
                     from Empregados);
```

Max(Salário)

3000



Sobrenome	Salário
Traviatta	3000

Total de empregados existentes.

33)

```
select  count (Id_emp)
from    Empregados;
```



count(Id_Emp)
4

Total gasto com salários.

34)

```
select  sum (Salário)
from    Empregados;
```



sum(Salário)
10550

Group by

- forma grupos
 - linhas com mesmos valores para os atributos da cláusula group by são colados em um grupo

Listar para cada departamento a respectiva média de salários.

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

35)
select Cod_Depto, **avg**(Salário)
from Empregados
group by Cod_Depto;

Cod_Depto	avg(Salário)
42	2525
44	2500
43	3000

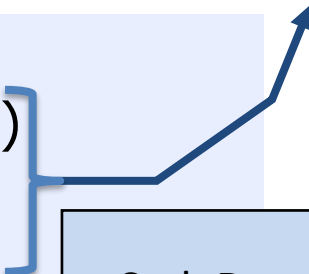
Having

- condição aplicada ao grupo

Listar todos os departamentos com média de salário maior que 2500.

Cod_Depto	avg(Salário)
42	2525
44	2500
43	3000

36)
select Cod_Depto, **avg** (Salário)
from Empregados
Group by Cod_Depto
having **avg** (salário) > 2500;



Cod_Depto	AVG(Salário)
42	2525
43	3000

se as cláusulas WHERE e HAVING aparecem na mesma consulta, o predicado de WHERE é primeiro aplicado. As linhas que satisfazem a cláusula where são agrupadas (GROUP BY) e a cada grupo é aplicado o predicado de HAVING.

Empregados

Id_Emp	Nome	Sobrenome	Salário	Cod_Depto
0962	João	Alves	2500	42
0942	Ana Maria	Jordão	2550	42
0943	João	Santos	2500	44
0944	Helena	Traviatta	3000	43

Deptos, excetuando o depto 42, com média salarial acima de 2500 reais.

37)

```
select      Cod_Depto, avg (Salário) as Salário
from        Empregados
Where       Cod_Depto <> 42
Group by   Cod_Depto
having      avg (salário) > 2500;
```

Cod_Depto	Salário
43	3000