



Delegação Regional do Norte
Centro de Emprego e Formação Profissional do Porto

Bases de Dados

- SQL – Parte II –

- Operações de Manipulação e Seleção -

António Sousa

antonioribeirosousa@gmail.com



INSERT

- Permite adicionar tuplos a uma tabela.

```
INSERT INTO <TABELA> [ (<ATRIB_1>, ..., <ATRIB_N>) ]  
[ VALUES (<VAL_A1>, ..., <VAL_AN>), ..., (<VAL_M1>, ..., <VAL_MN>) | <CONSULTA> ] ;
```

- Quando não se indica os atributos da tabela assume-se que são todos e pela ordem definida quando da sua criação. Os valores são inseridos por essa mesma ordem.



INSERT

- Quando se explicita o nome dos atributos é possível definir a ordem e indicar apenas parte dos atributos. Os valores são inseridos na ordem definida.
- Os atributos com declarações NOT NULL e sem declarações DEFAULT devem ser sempre indicados.
- Nos atributos não indicados é inserido o valor por defeito (se definido) ou o valor NULL.



INSERT

- Adicionar um novo tuplo a uma tabela.

```
INSERT INTO DEPARTAMENTO VALUES  
( 'Investigação', 10, '487563546', '2016-11-  
30' );
```

- Adicionar dois novos tuplos a uma tabela indicando apenas parte dos atributos.

```
INSERT INTO DEPARTAMENTO (Num, Nome)  
VALUES (11, 'Publicidade'), (12, 'Recursos Humanos');
```



DELETE

- Permite remover tuplos de uma tabela.

DELETE FROM <TABELA> [**WHERE** <COND>] ;

- Quando não se indica a condição WHERE todos os tuplos são removidos mas a tabela mantém-se na BD embora vazia.
- Esta operação apenas remove diretamente tuplos de uma tabela. No entanto, esta operação pode propagar-se a outras tabelas para manter as restrições de integridade referencial.



DELETE

- **ON DELETE SET NULL:** coloca o valor NULL na chave externa dos tuplos que referenciam os tuplos removidos.
- **ON DELETE SET DEFAULT:** coloca o valor por defeito na chave externa dos tuplos que referenciam os tuplos removidos.
- **ON DELETE CASCADE:** remove todos os tuplos que referenciam os tuplos removidos.



DELETE

- Remover zero ou um tuplo de uma tabela.

```
DELETE FROM EMPREGADO WHERE NumBI = '487563546';
```

- Remover zero, um ou vários tuplos de uma tabela.

```
DELETE FROM EMPREGADO WHERE Salário > 2000;
```

- Remover zero, um ou vários tuplos de uma tabela.

```
DELETE FROM EMPREGADO WHERE NumDep IN (SELECT  
Num FROM DEPARTAMENTO WHERE Nome =  
'Produção');
```

- Remover todos os tuplos de uma tabela.

```
DELETE FROM EMPREGADO;
```



UPDATE

- Permite alterar os valores dos atributos de um ou mais tuplos de uma tabela.

UPDATE <TABELA>

SET <ATRIB_1> = <VAL_1>, ..., <ATRIB_N> =
<VAL_N>

WHERE <COND>;

- Esta operação apenas altera directamente tuplos de uma tabela. No entanto, esta operação pode propagar-se a outras tabelas para manter as restrições de integridade referencial.



UPDATE

- **ON UPDATE SET NULL:** coloca o valor NULL na chave externa dos tuplos que referenciam os tuplos alterados.
- **ON UPDATE SET DEFAULT:** coloca o valor por defeito na chave externa dos tuplos que referenciam os tuplos alterados.
- **ON UPDATE CASCADE:** actualiza com os novos valores a chave externa dos tuplos que referenciam os tuplos alterados.



UPDATE

- Alterar um tuplo de uma tabela.

```
UPDATE PROJECTO SET Localização = 'Gaia', NumDep =4 WHERE  
Num = 1;
```



SELECT-FROM-WHERE

- Permite consultar a base de dados para obter informação.

```
SELECT <ATRIB_1>, ..., <ATRIB_N> FROM <TABELA_1>,  
..., <TABELA_M> [WHERE <COND>];
```

- Para que uma tabela seja um conjunto é necessário especificar restrições do tipo PRIMARY KEY ou UNIQUE sobre os atributos da tabela ou utilizar a opção DISTINCT nas consultas.
- Em SQL, os operadores de comparação são {=, <, <=, >, >=, <>}.



Delegação Regional do Norte
Centro de Emprego e Formação Profissional do Porto

SELECT-FROM-WHERE

- Obtenha o número do BI, primeiro e último nome de todos os empregados.

```
SELECT NumBI, NomeP, NomeF FROM EMPREGADO;
```



Delegação Regional do Norte
Centro de Emprego e Formação Profissional do Porto

SELECT-FROM-WHERE

Obtenha o número do BI dos empregados que trabalham no departamento 3 e cujo salário é superior a 1500 euros.

```
SELECT NumBI FROM EMPREGADO WHERE NumDep = 3 AND  
Salário;
```



SELECT-FROM-WHERE

- Obtenha o nome dos empregados que trabalham no departamento de Marketing.

SELECT NomeP, NomeF

FROM EMPREGADO, DEPARTAMENTO

WHERE Nome = 'Marketing' **AND**
NumDep = Num;



SELECT *

- Permite substituir a lista de atributos a selecionar, significando que se pretende selecionar todos os atributos sem os mencionar explicitamente.

SELECT *

FROM <TABELA_1>, ..., <TABELA_M>

[**WHERE** <COND>];

- Obtenha os empregados que trabalham no departamento 4 e cujo salário é superior a 2000 euros.

SELECT *

FROM EMPREGADO

WHERE NumDep = 3 AND Salário > 1500;



SELECT DISTINCT

- Permite remover os tuplos em duplicado do resultado da consulta.

```
SELECT DISTINCT <ATRIB_1>, ..., <ATRIB_N>  
FROM <TABELA_1>, ..., <TABELA_M>  
[WHERE <COND>];
```

- O SQL não remove automaticamente os tuplos em duplicado porque:
 - É uma operação dispendiosa pois requer ordenar tuplos e remover duplicados.
 - Quando se aplicam funções de agregação não é usual excluir as repetições do resultado.
 - O utilizador pode querer ver os tuplos em duplicado no resultado da consulta.

- Obtenha todos os diferentes primeiros nomes dos empregados.

```
SELECT DISTINCT NomeP  
FROM EMPREGADO;
```




Operadores Aritméticos

- Os operadores aritméticos +, -, * e / podem ser utilizados para fazer cálculos com atributos do tipo numérico.

<ATRIB> [+ | - | * | /] <VAL>

- Para todos os projectos localizados no Porto, obtenha o nome dos empregados e os respectivos salários se estes fossem aumentados em 5%.

```
SELECT NomeP, NomeF, Salário * 1.05
FROM EMPREGADO, TRABALHA_EM, PROJECTO
WHERE Nome = 'Porto' AND Num = NumProj AND
EmpBI = NumBI;
```



Ambiguidade no nome dos atributos

- Se as relações envolvidas numa consulta tiverem atributos com o mesmo nome pode ocorrer uma situação ambígua. Por exemplo, as relações

DEPARTAMENTO (Nome, Num, GerenteBI,
GerenteData)

PROJECTO (Nome, Num, Localização, NumDep)

têm atributos com o mesmo nome.



Ambiguidade no nome dos atributos

- A solução é **qualificar** os atributos com o prefixo do nome da relação a que pertencem:

DEPARTAMENTO.Nome

DEPARTAMENTO.Num

PROJECTO.Nome

PROJECTO.Num



Ambiguidade no nome dos atributos

- Para todos os projectos localizados no Porto, obtenha o nome do projecto e o último nome do respectivo gerente.

```
SELECT PROJECTO.Nome, NomeF  
FROM EMPREGADO, DEPARTAMENTO, PROJECTO WHERE PROJECTO.Nome =  
'Porto'  
AND PROJECTO.NumDep = DEPARTAMENTO.Num  
AND GerenteBI = NumBI;
```



Ambiguidade no nome dos atributos

- Se uma consulta referenciar uma mesma relação mais do que uma vez pode ocorrer uma situação ambígua. Por exemplo, o atributo SuperBI da relação EMPREGADO referencia a própria relação.
- A solução é renomear cada uma das duas relações:
EMPREGADO **AS** E
EMPREGADO **AS** S



Ambiguidade no nome dos atributos

- Obter para cada empregado, o seu último nome e o último nome do respectivo supervisor.

```
SELECT E.NomeF, S.NomeF  
FROM EMPREGADO AS E, EMPREGADO AS S  
WHERE E.SuperBI = S.NumBI;
```



Renomear Atributos

- O SQL permite que os atributos que aparecem no resultado de uma consulta sejam renomeados de modo a fazerem mais sentido.
<ATRIB> **AS** <NOVO_NOME>
- Para todos os projectos localizados no Porto, obtenha o nome dos empregados e os respectivos salários se estes fossem aumentados em 5%.

```
SELECT NomeP, NomeF, Salário * 1.05 AS Aumento_Salário
FROM EMPREGADO, TRABALHA_EM, PROJECTO
WHERE Nome = 'Porto' AND Num = NumProj AND EmpBI =
NumBI;
```



Renomear Atributos

- Obter para cada empregado, o seu último nome e o último nome do respectivo supervisor.

```
SELECT E.NomeF AS Nome_Empregado, S.NomeF  
AS Nome_Supervisor  
FROM EMPREGADO AS E, EMPREGADO AS S  
WHERE E.SuperBI = S.NumBI;
```




BETWEEN

- Permite definir um intervalo numérico de comparação.

<ATRIB> **BETWEEN** <VAL_1> **AND** <VAL_2>

- Obtenha os empregados que trabalham no departamento 4 e cujo salário é superior a 2000 euros e inferior a 4000 euros.



BETWEEN

```
SELECT *  
FROM EMPREGADO  
WHERE NumDep = 4 AND (Salário BETWEEN 2000 AND 4000);
```



LIKE

- Permite comparar atributos do tipo string com sequências de texto padrão.

<ATRIB> [**NOT**] **LIKE** <PADRÃO>

- Existem dois caracteres com significado especial:
 - % representa um número arbitrário de caracteres.
 - _ representa um qualquer caracter.



IS NULL

- Permite verificar se os valores de um atributo são NULL (não conhecido, em falta ou não aplicável).

```
<ATRIB> IS [NOT] NULL
```

- Obtenha o nome dos empregados que não têm supervisor.

```
SELECT NomeP, NomeF  
FROM EMPREGADO  
WHERE SuperBI IS NULL;
```



ORDER BY

- Permite definir a ordem dos tuplos do resultado.

```
ORDER BY <ATRIB_1> [ASC | DESC],  
..., <ATRIB_N> [ASC | DESC]
```

- Obtenha por ordem alfabética o nome dos empregados que trabalham no departamento 4 e cujo salário é superior a 2000 euros.

```
SELECT NomeP, NomeF  
FROM EMPREGADO  
WHERE NumDep = 4 AND Salário > 2000  
ORDER BY NomeP, NomeF;
```