Os comandos INSERT INTO, UPDATE e DELETE pertencem à parte da linguagem SQL responsável pela manipulação de dados (DML).

1- Comando INSERT INTO

1.1 Inserção de registos simples

O comando INSERT permite realizar a introdução de registos numa tabela da Base de dados.

Sintaxe:

```
INSERT INTO  <nome_tabela> (<campo1> , <campo2>, … , <campoN>)
VALUES (valor1, valor2, … , valorN),
          (valor11, valor21, … , valorN1),
                …
          (valor1x, valor2x, … , valorNx)
Ou
INSERT INTO  <nome_tabela>  VALUES (valor1, valor2, … , valorN),
                                          (valor11, valor21, … , valorN1),
                                             …
                                          (valor1x, valor2x, … , valorNx)
```

NOTAS (1):

- O formato abreviado pode ser utilizado quando queremos introduzir dados em todos os campos da tabela, devemos, no entanto, respeitar a ordem da disposição da criação dos campos na tabela.

  Exemplo: Considera a seguinte tabela (tab1)

C1	C2	C3	C4
V1	V2	V3	V4
V9	V8	V5	V7
…	…	…	…

  Insert into tab1 values (v1, v2, v3, v4)

- Os campos que não forem colocados no comando INSERT são preenchidos com o valor NULL.

  Exemplo: insert into tab1 (C1, C3) values (v1, v2) Resultado:

V1		V2	

- O formato abreviado pode ser usado mesmo não tendo dados para todos os campos, basta preencher os restantes campos com NULL.

  Exemplo: INSERT INTO tab1 VALUES (v1, Null, v2, null)

- O Access usa o formato #mm/dd/yyyy# (entre cardinais) para representar uma data no formato habitual. A introdução também pode ser feita usando #mm-dd-yyyy#. Em, por exemplo MySQL e Oracle, é usado o formato 'yyyy-mm-dd'.

Exemplo1: INSERT INTO ficha (nome, datanasc, telefone) VALUES ('Ana', '1990-12-24', '275333555')

Exemplo2: Introduzir na tabela Postal, uma linha com os valores 6200 e Covilhã, referente aos campos CodPostal e Localidade.

INSERT INTO Postal (CodPostal, Localidade) **VALUES** (6200, 'Covilhã')

O comando INSERT poderá falhar:

- Ao tentar inserir mais do que uma vez a mesma chave primária;
- Ao tentar inserir mais do que uma vez o mesmo valor num campo com índice único;
- Ao tentar inserir um valor NULL um campo NOT NULL;
- Se o tipo de dados enviados na componente VALUES não estiver de acordo com cada um dos campos (por exemplo, inserir um valor do tipo texto num campo do tipo inteiro);
- Se algum dos campos obrigatórios for ignorado;
- Se o nº de campos for diferente do nº de valores associados;
- Os valores presentes na lista de valores (VALUES) devem corresponder a cada uma das colunas da lista de campos ou, se esta não existir, deve corresponder ao conjunto e ordem dos campos na tabela;
- Se existir algum tipo de restrição na tabela a que os dados não obedeçam.

1.2 Inserção de Conjuntos de Registos

Podemos também inserir numa tabela conjuntos de registos que existem noutra (s) tabela(s). Desde que as suas estruturas sejam iguais - campos, com as mesmas caraterísticas, podendo não ser selecionados o nº total de campos da tabela de origem.

Sintaxe:

INSERT INTO <Nome_Tabela> [(<Campo1>, …<CampoN>)]
SELECT …

Exemplo: Supõe que existe uma tabela denominada Adultos com uma estrutura exatamente igual à estrutura de uma tabela denominada **Pessoas (campos: Id, Nome, Idade, Salario, Telefone, CodPostal, Localidade)**.
Pretende-se introduzir na tabela Adultos o conjunto das Pessoas com 18 anos ou mais.

INSERT INTO Adultos **SELECT** * FROM Pessoas WHERE Idade >=18

Ou

INSERT INTO Adultos (Id, Nome, Idade, Salario, Telefone, CodPostal)
SELECT * FROM Pessoas WHERE Idade >=18

2 - Comando UPDATE

O comando UPDATE permite alterar os valores já existentes nos campos de uma única tabela.

Sintaxe:

```
UPDATE <Nome_Tabela> [Alias]
SET <Campo1> = { Expressão1, Query1},
      <Campo2> ={ Expressão2, Query2},
      ...
      <CampoN> ={ ExpressãoN, QueryN}, …
[WHERE Condição]
```

NOTA:

- Se não for colocada a cláusula WHERE no comando UPDATE, as alterações serão realizadas em todos os registos da tabela;
- No Access não é válida a utilização de Querys (consultas) ligadas à cláusula SET;
- O *Alias* é outra designação que pode reduzir ou alterar o nome real da tabela, colocando-se à frente desta e passará a ser outra forma de identificar a tabela.

UPDATE não gera um conjunto de resultados. Para além disso, depois de atualizares os registos utilizando uma consulta atualizar, não podes anular a operação. Se pretendes saber quais foram os registos atualizados, examina primeiro os resultados de uma consulta de seleção que utiliza os mesmos critérios e, em seguida, executa a consulta de atualização.

Mantém sempre cópias de segurança dos dados. Se atualizares os registos incorretos, poderás obtê-los a partir das cópias de segurança.

Exemplos (tabela Pessoa com os campos: id, Nome, idade, Salario, telefone, CodPostal):

1- Aumentar o salário 5% a todas as pessoas.

```
UPDATE Pessoa
SET Salario = Salario * 1.05
```

2- Aumentar 10% o salário apenas à Sílvia Martins subtraindo-lhe 5 anos à idade.

```
UPDATE Pessoa
SET Salario = Salario * 1.1
      Idade = Idade – 5
WHERE Nome = "Sílvia Martins"
```

3- (válido para SQL Server e Oracle, por exemplo)

O salário do Ambrósio Pastel deverá passar a ser igual ao menor salário da empresa e o seu campo Telefone deverá ter o valor NULL.

```
UPDATE Pessoa
SET Salario = (SELECT MIN (Salario) FROM Pessoa),
    Telefone =NULL
WHERE Nome = 'Ambrósio Pastel'
```

3- Comando DELETE

Este comando permite apagar conjuntos de linhas existentes numa única tabela.

Sintaxe:

```
DELETE FROM <Nome_Tabela>
[WHERE Condição]
```

NOTA: Tal como nos comandos Select e Update a cláusula WHERE restringe o conjunto de registos que irão ser apagados, se não for usada todas as linhas da tabela serão apagadas.

Exemplos:

1-Apagar todas as linhas da tabela Pessoas

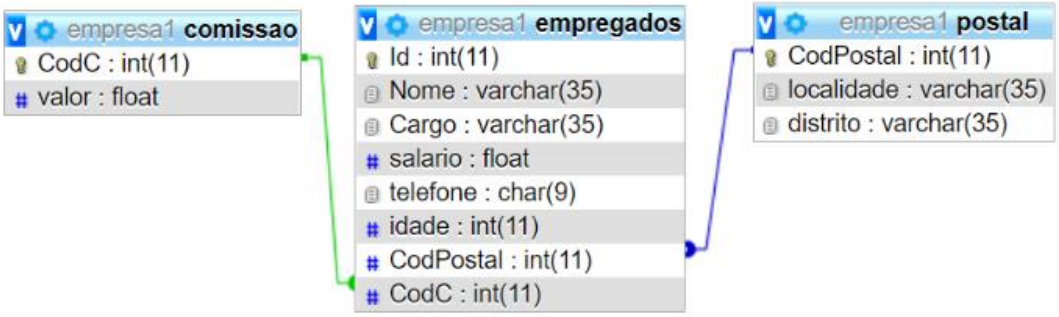
```
DELETE FROM Pessoas;
```

2- Apagar da tabela Pessoas o sujeito Bartolomeu Fastio.

```
DELETE FROM Pessoas
WHERE Nome Like 'Bartolomeu Fastio';     em substituição da clausula like usar  =' Bartolomeu Fastio'
```

EXERCÍCIOS

NOTA: Considera o seguinte esquema de tabelas:



Escreve o SQL que permita:

1- Introduzir na tabela postal um registo com o seguinte conteúdo (usa as 2 sintaxes estudadas:

Nome do campo	Valor do campo
CodPostal	1000
localidade	Lisboa
distrito	Lisboa

2- Supondo que a tabela **funcionarios** tem a mesma estrutura da tabela **empregados**.
Copiar todos os registos da tabela empregados para a tabela funcionarios.

3- Continua a supor que a tabela **funcionarios** tem a mesma estrutura que a tabela **empregados**.
Copiar para a tabela **funcionarios** os registos da tabela empregados, correspondente aos indivíduos que auferem dum salário superior a 1000 euros.
Ordenar os registos por Nome.

4- Altera os registos da tabela empregados de modo a que todos os indivíduos com mais de 40 anos tenham um acréscimo no salário de 100.

5- Adicionar o prefixo "01" ao telefone de todos os indivíduos que têm telefone e que vivem em Lisboa. Usa as tabelas empregados e postal (esta última para referência à localidade onde vivem os empregados).

6- Substituir o Nome Bruno pelo Nome Carlos na tabela empregados.

7- Apagar as localidades da tabela postal que não estão associadas a ninguém da tabela empregados.

8- Criar a tabela empregados2 a partir da tabela empregados.

Cria um ficheiro do word com o nome F5EX9.docx, para onde copiarás o sql relativo a cada exercício.

9- Inicializa o serviço de **Wamp**, e através do PHPMyAdmin elabora as seguintes operações:

a. Inserir os seguintes dados nas tabelas:

postal

CodPostal	localidade	distrito
6200	Covilhã	C.Branco
6230	Fundão	C.Branco
6320	Sabugal	Guarda
1000	Lisboa	Lisboa
4450	Maia	Porto

comissao

CodC	valor
1	350
2	300
3	150
4	50

empregados

Id	Nome	cargo	salario	telefone	idade	CodPostal	CodC
1	Aníbal	Tesoureiro/a	950,00	275564589	35	6200	2
2	Anita	Assistente Administrativa	1000,00	275587123	25	6200	2
3	Cruz	CEO	2500,00	275587936	48	6200	1
4	Sousa	Gestor/a de Marketing	1100,00	275698714	40	6230	1
5	Xavier	Técnico/a Especializado/a	900,00	275698574	33	1000	3
6	Bruno	Técnico/a Especializado/a	900,00	271222333	36	6320	3
7	Alexandra	Técnico/a Especializado/a	900,00	271666555	28	1000	3
8	Carla	Motorista	800,00	275888999	26	6230	4

b. Criar a tabela funcionarios com as mesmas caraterísticas da tabela empregados.

c. Reproduz e executa o sql relativo a todos os exercícios (guarda o sql no ficheiro F5EX9.docx identificando a alínea de cada exercício).

d. Exporta a base de dados e envia-a para a disciplina de PSD, na classroom do Google Apps.

REVER : criar queries

10. Apresente a instrução SQL que permite efetuar a seleção de dados adequada a cada um dos seguintes pedidos (copia a querie e resultado para um ficheiro, com o nome F5Ex10a.docx):

a. O nome e o salário de todos os empregados com cargo Técnico/a Especializado/a

b. O nome dos empregados com salário entre 950 e 2000 ou cujo código da comissão (CodC) esteja compreendido entre 1 e 4.

c. O nome e cargo dos empregados cujo código da comissão não esteja compreendido entre 1 e 4.

d. O nome e cargo de todos os empregados da Covilhã.

e. O nome e cargo de todos os empregados com comissão igual a 300 ou 350.

f. O nome e cargo de todos os empregados da Covilhã, com comissão igual a 150 ou 300.

g. O nome dos empregados, o salário e a metade deste. Nota: É necessário definir um campo para calcular a metade do ordenado.

h. A média dos salários por cada localidade.

Bibliografia

Damas, L. (2005). SQL. Lisboa: FCA
Tavares, F. (2015). MySql. Lisboa: FCA
Ex10 -exercícios da ficha n.º5, da UFCD 0814 (20/21), da prof.ª Anabela Rocha