ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Structured Query Language

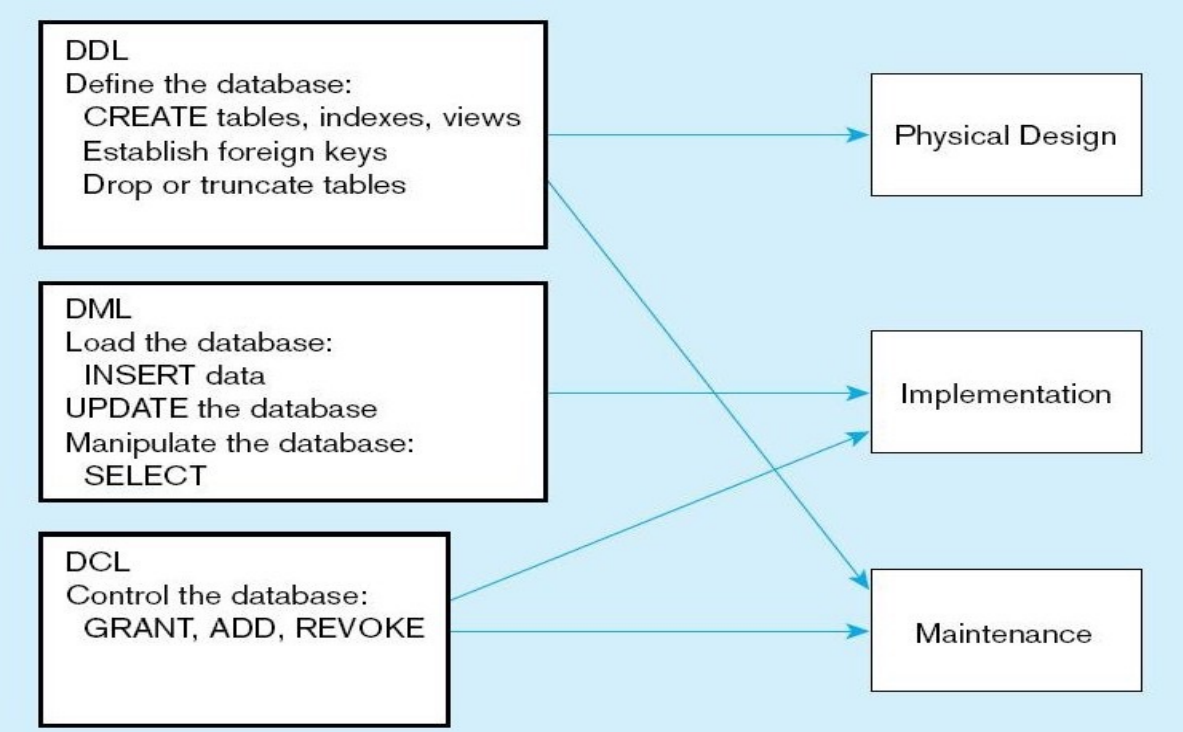
Data Definition Language (DDL)

DEE
Departamento de Engenharia Electrotécnica

Licenciatura em Engenharia de Telecomunicações e Informática

SQL- Strutured Query Language

- SQL é mais que uma linguagem de interrogação estruturada. Inclui características para a definição da estrutura de dados, para alterar os dados de uma base de dados, e para especificar esquemas de segurança. Estas características agrupam-se do seguinte modo:



Data Definition Language -DDL

DEE Departamento de Engenharia Electrotécnica

Create Database

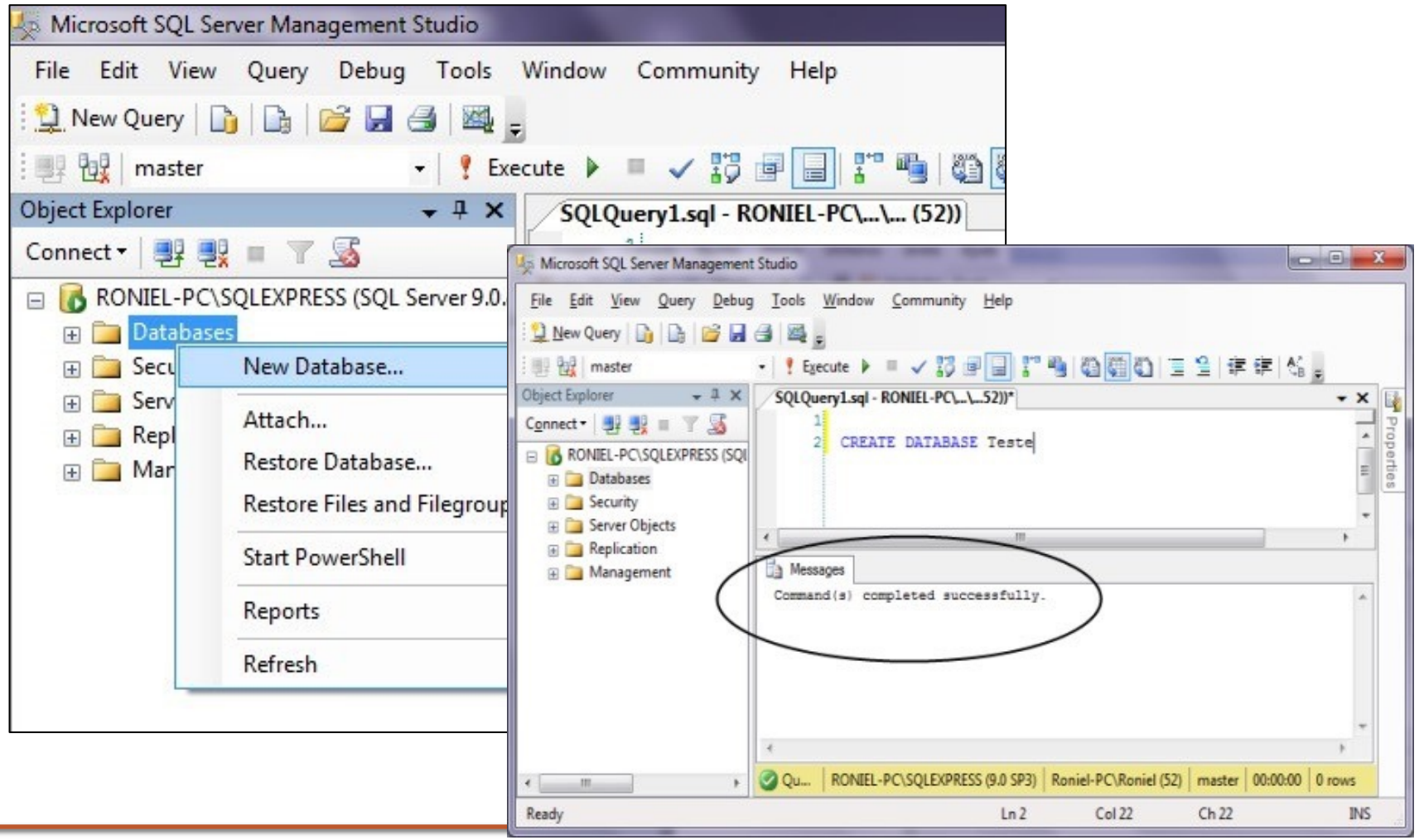
- Antes de começarmos a criar a nossa primeira **base de dados** , vamos entender melhor o que é um base de dados.
- **"Database** ou **Base de Dados** é um contêiner que guarda todas as tabelas e outras estruturas SQL relacionadas àquelas tabelas"
- Bases de Dados são constituídos por **tabelas**, que é a parte interna de uma base de dados. Estas tabelas contêm dados em linhas e colunas.

Tabela(relação) Empregados

Nr_Emp	Nome	Nrcc	Categoria	Salário	Nrdepart	Data_nasc
1	António Sousa	111111	programador	1000	3	20-03-2000
2	Ana Amaral	111112	dba	1000	4	20-04- 1970
3		111123	programador	2000	2	12-04-1990
4	Carlos Silva	234567	operador	100	1	20-08-2080

Data Definition Language -DDL

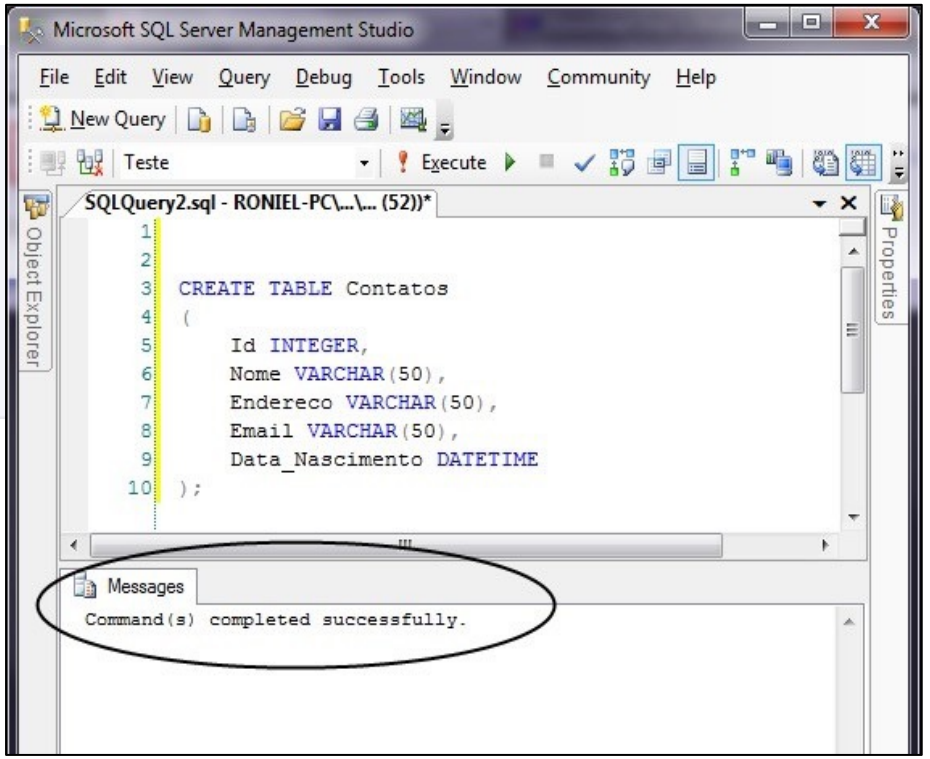
- **Primeiro devemos criar uma DATABASE**
  - Pode ser feito de duas formas
    - modo designer e
    - por linha de comando.



Data Definition Language -DDL

Create Table – o comando mais simples para criar tabelas

```
CREATE TABLE table_name
(
  column1 datatype [ NULL | NOT NULL ],
  column2 datatype [ NULL | NOT NULL ],
  ...
);
```



Tipo de Dados

Na linguagem **SQL** (Structured Query Language) existem três tipos de dados que são agrupados em três grupos

- **Strings**
- **Dados Numéricos**
- **Dados relacionados com Tempo ou Datas**

Tipos de Dados	Descrição
Character(n) ou Char(n)	String de comprimento Fixo (n > 0)
Character ou Char	Um único caráter
Character Varying(n) ouVarchar(n)	String de Comprimento Variável (n > 0)
Bit(n)	String com n Bits (n > 0) Comprimento Fixo
Bit Varying(n)	String com n Bits (n > 0) Comprimento Variável
Numeric ou Numeric(n) ouNumeric(n,d)	Valor numérico constituído de por n dígitos e sinal com d casas decimais (0 <= d <= n; n > 0)
Integer ou Int	Inteiro (com sinal)
Smallint	Pequeno Inteiro (Short int) com sinal
Float	Nº de Dupla Precisão (Ponto Flutuante)
Date	Data
Time	Hora
Timestamp	Intervalo Temporal

https://learn.microsoft.com/pt-br/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver16

Alter table

- **ALTER TABLE Statement**
  - The SQL Server (Transact-SQL) ALTER TABLE statement is used to add, modify, or drop columns in a table.

```
ALTER TABLE table_name
  ADD column_name column_definition;
```

```
ALTER TABLE employees
  ADD last_name VARCHAR(50);
```

```
ALTER TABLE table_name
  ADD column_1 column_definition,
      column_2 column_definition,
      ...
      column_n column_definition;
```

```
ALTER TABLE employees
  ADD last_name VARCHAR(50),
      first_name VARCHAR(40);
```

Alter table/Rename table

DEE Departamento de Engenharia Electrotécnica

Alterar nome da coluna

```
ALTER TABLE table_name
  ALTER COLUMN column_name column_type;
```

```
ALTER TABLE employees
  ALTER COLUMN last_name VARCHAR(75) NOT NULL;
```

Drop coluna

```
ALTER TABLE table_name
  DROP COLUMN column_name;
```

```
ALTER TABLE employees
  DROP COLUMN last_name;
```

Renomear do nome da coluna ou da tabela

```
sp_rename 'table_name.old_column_name', 'new_column_name', 'COLUMN';
```

```
sp_rename 'employees.last_name', 'lname', 'COLUMN';
```

Eliminar a tabela

```
DROP TABLE table_name;
```

```
sp_rename 'employees', 'emps';
```

Restrições de integridade

- São regras que definem a validade dos dados.
- Por exemplo, para a seguinte relação:

Não pode ser negativo

Nr_Emp	Nome	Nrcc	Categoria	Salário	Nrdepart	Data_nasc
1	António Sousa	111111	programador	1000	3	20-03-2000
2	Ana Amaral	111112	dba	1000	4	20-04- 1970
3		111123	programador	2000	2	12-04-1990
4	Carlos Silva	234567	operador	100	1	20-08-2080

Não pode ser nulo

O nome não pode ser nulo

Demasiado pequeno

O salário tem que ser superior ao salario mínimo

Data inválida

A data de nascimento tem que ser maior que 01-01-1960 e menor que 01-01-2000

As regras anterior vão fazer parte da definição da tabela

Restrições de integridade

- As regras do exemplo anterior denominam-se de **restrições de integridade.**
- Tipos de restrições de integridade:
  - **Domínio** – são regras que se aplicam aos atributos de uma dada tabela, definindo o domínio de cada atributo
  - **Entidade** - Referem-se à **chave primaria** da tabela que não pode ter valores nulos nem valores duplicados
  - **Referencial** - é uma restrição que relaciona duas tabelas. Temos um novo conceito: **chave estrangeira**
  - **Negocio** - restrições complexas que não podem ser definidas na estrutura da tabela. São verificadas pelos programas de aplicação;
    - **Exemplos :** o salario de um empregado não pode diminuir, só aumentar; ou

      o empregado não pode ganhar mais do que o seu chefe

Restrições de integridade

NULL E DEFAULT

```
CREATE TABLE EMPREGADO
( Nr_empr  int,
Nome varchar(50)  NOT NULL,
Nrcc int,
Categoria  varchar(30) DEFAULT "programador"
Salário int Default 600,
Nr_depart int,
Data_nascimento  date);
```

Nr_Emp	Nome	Nrcc	Categoria	Salário	Nrdepart	Data_nasc
1	António Sousa	111111	programador	1000	3	20-03-2000
2	Ana Amaral	111112	dba	1000	4	20-04-1970
3		111123	programador	2000	2	12-04-1990
4	Carlos Silva	234567	operador	100	1	20-08-2080

O nome não pode ser nulo

A categoria por omissão é programador para todos os empregados

RESTRIÇÃO – UNIQUE

UNIQUE

```
CREATE TABLE EMPREGADO
(    Nr_empr  int,
     Nome varchar(50)  NOT NULL, Nrcc  int,
     Categoria  varchar(30) default "programador" Salário int,
     Nr_depart int,
     Data_nascimento  date,
     CONSTRAINT nrcc_unique  UNIQUE ( Nrcc));
```

Nr_Emp	Nome	Nrcc	Categoria	Salário	Nrdepart	Data_nasc
1	António Sousa	111111	programador	1000	3	20-03-2000
2	Ana Amaral	111112	dba	1000	4	20-04-1970
3		111123	programador	2000	2	12-04-1990
4	Carlos Silva	234567	operador	100	1	20-08-2080

O número do cartão de cidadão deve ser único

RESTRIÇÃO – CHECK

```
CREATE TABLE table_name
(
  column1 datatype [ NULL | NOT NULL ],
  column2 datatype [ NULL | NOT NULL ],

  ...

  CONSTRAINT constraint_name
    CHECK [ NOT FOR REPLICATION ] (column_name condition)

);
```

CREATE TABLE EMPREGADO

(Nr_empr int,

Nome varchar(50) NOT NULL, Nrcc

int,

Categoria varchar(30) default “programador”,

Salário int,

Nr_depart int,

Data_nascimento date,

CONSTRAINT nrcc_unique **UNIQUE** (Nrcc),

CONSTRAINT salario_maior **CHECK** (salário >600));

Nr_Emp	Nome	Nrcc	Categoria	Salário	Nrdepart	Data_nasc
1	António Sousa	111111	programador	1000	3	20-03-2000
2	Ana Amaral	111112	dba	1000	4	20-04-1970
3		111123	programador	2000	2	12-04-1990
4	Carlos Silva	234567	operador	100	1	20-08-2080

O salário tem que ser superior ao salario mínimo

Constraints – ADD, DROP

DEE Departamento de Engenharia Electrotécnica

Adicionar

```
ALTER TABLE employees
ADD CONSTRAINT employees_unique UNIQUE (employee_number);
```

```
ALTER TABLE employees
ADD CONSTRAINT check_last_name
  CHECK (last_name IN ('Smith', 'Anderson', 'Jones'));
```

Eliminar constraints

```
ALTER TABLE table_name
DROP CONSTRAINT constraint_name;
```

Não podemos alterar restrições. Temos que eliminar e depois recriar.

Restrição de Entidade – Primary Key

- **Chave Primária**
- uma tabela não pode conter linhas duplicadas, porque isso criaria ambiguidades na recuperação.
- Para garantir a singularidade, cada tabela deve ter uma coluna (ou um conjunto de colunas), chamada **chave primária**, que identifica exclusivamente todos os registos da tabela.
- Podem existir vários atributos cujos valores identificam exclusivamente uma ocorrência dessa tabela**: chaves candidatas. A chave primária é uma das chaves candidatas.**

Nr_Emp	Nome	Nrcc	Categoria	Salário	Nrdepart	Data_nasc
1	António Sousa	111111	programador	1000	3	20-03-2000
2	Ana Amaral	111112	programador	1000	4	20-04- 970
3	José	111123	programador	2000	2	12-04-1990
4	Carlos Silva	234567	programador	650	1	20-08-2080

- ***Nrcc*** e ***Nr_Emp*** são atributos que são chaves candidatas da entidade ***empregado***

PRIMARY KEY

```
CREATE TABLE table_name
(
  column1 datatype [ NULL | NOT NULL ] [ PRIMARY KEY ],
  column2 datatype [ NULL | NOT NULL ],
  ...
);
```

```
CREATE TABLE employees
( employee_id INT PRIMARY KEY,
  last_name VARCHAR(50) NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  salary MONEY
);
```

OR

```
CREATE TABLE table_name
(
  column1 datatype [ NULL | NOT NULL ],
  column2 datatype [ NULL | NOT NULL ],
  ...
  CONSTRAINT constraint_name PRIMARY KEY (column1, column2, ... column_n)
);
```

```
CREATE TABLE employees
( employee_id INT,
  last_name VARCHAR(50) NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  salary MONEY,
  CONSTRAINT employees_pk PRIMARY KEY (employee_id)
);
```

Qdo declaramos um atributo como chave primária da tabela,o SGBD não deixa que a tabela tenha dois registos com o mesmo valor nesse atributo e também não permite que esse atributo tenha o valor null

PRIMARY KEY

- Pode acontecer que a **chave primaria seja constituída por mais do que um atributo** e neste caso temos uma chave primaria composta
- Se a chave é composta então,

```
CREATE TABLE employees
( last_name VARCHAR(50) NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  salary MONEY,
  CONSTRAINT employees_pk PRIMARY KEY (last_name, first_name)
);
```

Restrição Referencial– Foreign Key

- **Chave Estrangeira**
  - As tabelas na base de dados devem estar relacionadas.
  - Uma tabela está relacionada com outra tabela se possui a chave primária da outra tabela.
    - A tabela que recebeu a chave primaria da outra tabela possui uma **chave estrangeira.**

- **Chave estrangeira** é um atributo ou conjunto de atributos que aparecem como chave primária numa outra tabela relacionada
- Ela permite estabelecer um ligação entre tabelas(relações)

Nr_Emp	Nome	Nrcc	Categoria	Salário	Nrdepart	Data_nasc
1	António Sousa	111111	programador	-1000	3	20-03-2000
2	Ana Amaral	111112	dba	1000	4	20-4- 1970
3		111123	programador	2000	2	12-04-1990
4	Carlos Silva	234567	operador	1	1	20-08-2080

Nrdepart	Nome	Local
1	DEE	Felgueiras
2	DEI	Porto
3	Gestão	Vila do Conde
4	DEQ	Braga

RESTRIÇÃO CHAVE ESTRANGEIRA

```
CREATE TABLE child_table
(
  column1 datatype [ NULL | NOT NULL ],
  column2 datatype [ NULL | NOT NULL ],
  ...

  CONSTRAINT fk_name
    FOREIGN KEY (child_col1, child_col2, ... child_col_n)
    REFERENCES parent_table (parent_col1, parent_col2, ... parent_col_n)
    ON DELETE CASCADE
    [ ON UPDATE { NO ACTION | CASCADE | SET NULL | SET DEFAULT } ]
);
```

```
CREATE TABLE products
( product_id INT PRIMARY KEY,
  product_name VARCHAR(50) NOT NULL,
  category VARCHAR(25)
);

CREATE TABLE inventory
( inventory_id INT PRIMARY KEY,
  product_id INT NOT NULL,
  quantity INT,
  min_level INT,
  max_level INT,
  CONSTRAINT fk_inv_product_id
    FOREIGN KEY (product_id)
    REFERENCES products (product_id)
    ON DELETE CASCADE
);
```

ON UPDATE Optional - Especifica o que deve acontecer aos dados da tabela filha quando os dados da tabela pai são atualizados.

NO ACTION - É usado em conjunto com as cláusulas **ON DELETE** ou **ON UPDATE**.

Significa que **nenhuma ação** será realizada sobre os dados da tabela filha quando os dados correspondentes na tabela pai forem apagados ou atualizados.

Como inserir dados nas Tabelas

- **Insert Into**

```
INSERT INTO table
(column1, column2, ... )
VALUES
( DEFAULT | NULL | expression1,
  DEFAULT | NULL | expression2,
  ...
);
```

```
INSERT INTO employees
(employee_id, last_name, first_name)
DEFAULT VALUES;
```

```
INSERT INTO employees
(employee_id, last_name, first_name)
VALUES
(10, 'Anderson', 'Sarah');
```

```
INSERT INTO employees
(employee_id, last_name, first_name)
VALUES
(10, 'Anderson', 'Sarah'),
(11, 'Johnson', 'Dale');
```

Como alterar dados nas Tabelas

- **Update**

```
UPDATE table
SET column1 = expression1,
    column2 = expression2,
    ...
[WHERE conditions];
```

```
UPDATE employees
SET first_name = 'Kyle',
    employee_id = 14
WHERE last_name = 'Johnson';
```

```
UPDATE employees
SET last_name = 'Johnson'
WHERE employee_id = 10;
```

Como eliminar dados nas Tabelas

DEE Departamento de Engenharia Electrotécnica

- **Delete**

```
DELETE [ TOP (top_value) [ PERCENT ] ]
FROM table
[WHERE conditions];
```

```
DELETE FROM employees
WHERE first_name = 'Sarah';
```

```
DELETE FROM employees
WHERE last_name = 'Johnson'
AND employee_id >= 80;
```

```
DELETE TOP(3)
FROM employees
WHERE last_name = 'Johnson';
```

Data Definition Language -DDL

Aula Prática

Exercício

Exercício

1. Crie uma base de Dados com o nome **Escola_DB**

2. Crie as seguintes tabelas com os respetivos domínios (tipo de dados):

Disciplina (coddisc, designacao, creditos, anocurricular, semestre, codCurso)

coddisc : alfanumérico com tamanho máximo de 5 caracteres.
designacao : alfanumérico com tamanho variavel de 50 caracteres. creditos: numero inteiro
anocurricular: numero inteiro
semestre : numero
codcurso: inteiro
numero inteiro

Curso (codcurso, Nome)

codcurso : numérico com 2 carateres
nome: alfanumérico com tamanho fixo de 10 carateres.

Aluno(numaluno, nome, nif, nrcc, local, datanascimento, sexo, codcurso)

numaluno : numérico de 7 dígitos
nome: alfanumérico com tamanho variável de 30 caracteres. nif: numero inteiro
nrcc : numero inteiro
local: alfanumérico com tamanho fixo de 10 caracteres.
datanascimento: data
sexo: alfanumérico com tamanho fixo de 1 caracter.
codcurso : numérico com 2 carateres

Notas (numaluno, coddisc, nota)

numaluno: numérico de 7 dígitos
coddisc: : alfanumérico com tamanho máximo de 4 caracteres.
nota : inteiro

Exercício (continuação)

3. Adicione as restrições as seguintes restrições:
   - O nome do curso e da disciplina não pode ser null
   - O atributo sexo só deve ser feminino, masculino, sem género
   - A nota que o aluno obteve na disciplina deve estar compreendida entre 0 e 20.
   - O numero de contribuinte e o numero do cartão de cidadão devem ser únicos
   - A data de nascimento deve ser superior a 1970
4. Identifique as chaves primarias das tabelas (relações)
5. Crie as chaves primárias para cada uma das tabelas (relações)
6. Identifique as chaves estrangeiras das tabelas (relações)
7. Adicione as chaves estrangeiras para cada uma das tabelas (relações)

Exercício (continuação)

8. Indique se as seguintes instruções são ou não possíveis e justifique.

a) insert into curso(codcurso, nome) values(1, 'LETI')

b) insert into curso(codcurso, nome) values(1, 'LEI')

c) insert into curso(a, nome, nif, nrcc, local, datanascimento, sexo, codcurso) values (1, 'Sandra',11111111, 2222222, 'Porto', 20/10/1975, 'F',1)

d) insert into aluno(numaluno, nome, nif, nrcc, local, datanascimento, sexo, codcurso) values (2, 'José', 11111111, 2222221, 'Porto', 20/10/1975, 'F',1)

e) insert into aluno(numaluno, nome, nif, nrcc, local, datanascimento, sexo, codcurso) values (3, 'Carlos', 11111115, 2222231, 'Lisboa', 20/10/1960, 'F',2)

f) insert into aluno(numaluno, nome, nif, nrcc, local, datanascimento, sexo, codcurso) values (4, 'Ana' 11111215, 2322231, 'Lisboa', 20/10/1960, 'O',2)

g) Insert into disciplina(coddisc, designação, créditos, anocurricular, semestre,codCurso) values ('BDDAD', 4, 2,1, 1)

h) Insert into disciplina(coddisc, designação, créditos, anocurricular, semestre, codCurso) values ('ESOFT', 4, 2,1, 9)

i) insert into nota(numAluno, coddisc nota) values(1, 'BDDAD', 18)

j) insert into nota(numAluno, coddisc nota) values(1, 'BDDAD', 10)

k) insert into nota(numAluno, coddisc nota) values(1, 'SIMOV', 18)

l) insert into nota(numAluno, coddisc nota) values(2, 'BDDAD', , 23)