INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

Módulo 5: Bases de Dados

Aula 7
Introdução ao SQL; Queries Simples

UPskill

O que significa SQL?

- SQL significa *Structure Query Language*, ou seja, Linguagem de Consulta Estruturada
- Implementa os conceitos definidos no Modelo Relacional
- SQL é uma linguagem standard para manusear bases de dados (criar BD, manipular e consultar dados)
- SQL é uma linguagem de programação de muito alto nível

A Linguagem SQL

Módulos da Linguagem SQL:

- **DDL (*Data Definition Language*)**- inclui um conjunto de comandos para a criação e alteração de tabelas, chaves estrangeiras, regras de integridade referencial e views
- **DML (*Data Manipulation Language*)**- é uma implementação da álgebra relacional, isto é, inclui instruções para efetuar interrogações a uma base de dados, bem como inserir, alterar ou anular registos em tabelas
- **DCL (*Data Control Language*)**- inclui instruções para a definição de utilizadores, grupos de utilizadores e permissões de acesso às tabelas

A Linguagem SQL

Módulos da Linguagem SQL:

- **TCL (*Transaction Control Language*)-** inclui instruções para programação de transações (sequências de instruções indissociáveis)
- **PSM (*Persistent Stored Modules*)**- vertente da linguagem que permite programar e executar programas guardados na base de dados (existentes em dois tipos: *stored procedures* e *triggers*)

O que é uma Query?

Uma Query é um pedido de informação ou de um dado. Esse pedido também pode ser entendido como uma consulta, uma solicitação, ou ainda, uma requisição.

É um componente de extrema usabilidade que também permite que o utilizador insira, atualize, selecione e exclua registos em tabelas. Numa interpretação mais simples, são comandos que, ao serem executados, retornam com informações já armazenadas, que podem ser acedidas em qualquer momento se o utilizador fizer a pergunta (comando) correcta.

Exemplos de Queries em DML

Exemplos de diversas queries em Data Manipulation Language::

- **SELECT**
- **UPDATE**
- **INSERT**
- **DELETE**
- **entre outras**

SELECT

O comando SELECT é composto por seis cláusulas:

- **SELECT** - seleciona colunas
- **FROM** - indica sobre que tabelas é efetuada a pesquisa
- **WHERE** - seleciona as linhas
- **GROUP BY** - agrupa linhas em grupo
- **HAVING** - seleciona grupos (na ausência de GROUP BY comporta-se como WHERE)
- **ORDER BY** - indica o critério de ordenação da pesquisa

Apenas as duas primeiras são obrigatórias e tem de ser respeitada a ordem indicada.

SELECT <Campos>

FROM <Tabelas>

SELECT - FROM

Na forma mais simples, a cláusula SELECT selecciona colunas de uma tabela.

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT Nome
FROM Estudante
```

Output: tabela com os campos indicados no SELECT

Nome
António Brito
Daniel Fernandes
João Antão
João Pavia
Jorge Rafael
José Serro
Pedro Romano
...

emprego digital

SELECT - FROM

Na forma mais simples, a cláusula SELECT selecciona colunas de uma tabelas.

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT Curso, Nome
FROM Estudante
```

Output: tabela com os campos indicados no SELECT

Curso	Nome
Economia	António Brito
Informática	Daniel F.
Marketing	João Antão
Sociologia	João Pavia
História	Jorge Rafael
Null	José Serro
Informática	Pedro Romano
...	...

Qualquer comando SELECT devolve uma tabela.

Sempre que, no contexto do SQL, for referida uma tabela, ela pode ser uma tabela original (definida no modelo relacional) ou o resultado de um comando SELECT.

SELECT - FROM

Na forma mais simples, a cláusula SELECT selecciona colunas de uma tabelas.

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT *
FROM Estudante
```

Output: tabela com os campos indicados no SELECT

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

SELECT **DISTINCT**

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT Curso
FROM Estudante
```

Curso
Economia
Informática
Marketing
Sociologia
História
Null
Informática
...

```
SELECT DISTINCT
Curso
FROM Estudante
```

Curso
Economia
Informática
Marketing
Sociologia
História
Null

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

SELECT <Campos>
FROM <Tabelas>
WHERE <Filtro>

SELECT - FROM - WHERE

Com o WHERE, podemos filtrar a nossa query.

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT Curso, Nome
FROM Estudante
WHERE Curso='Informática'
```

Output: tabela com os campos indicados no SELECT filtrados pela condição no WHERE

Curso	Nome
Informática	Daniel F.
Informática	Pedro Romano

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

<Filtro> é a expressão lógica que define o filtro para selecionar cada registo.

O <Filtro> pode ser constituído por várias condições lógicas combinadas entre si.

Operadores a aplicar nos Filtros

Operadores Relacionais: **<; >; =; >=; <=; <>**

Operadores lógicos: **AND; OR; NOT**

Exemplos:

```
SELECT Nome FROM ESTUDANTE WHERE Curso=’Economia’ OR Curso=’Informática
SELECT Nome FROM ESTUDANTE WHERE Curso=’Economia’ AND Curso=’Informática
SELECT Nome FROM ESTUDANTE WHERE Curso <> ’Economia’
SELECT Nome FROM ESTUDANTE WHERE Número <= 20005
SELECT Nome FROM ESTUDANTE WHERE Número <= 20005 AND Número >= 20003
```

O operador **LIKE**

O operador LIKE é utilizado em clausulas WHERE para procurar um certo padrão.

```
SELECT nome FROM Estudante
WHERE curso LIKE ‘%a’
```

Muitas vezes é usado em conjunto com as *wildcards*:

- % - zero, um ou muitos caracteres
- _ - um único caracter

O operador **LIKE**

Exemplos:

... `WHERE name LIKE` **'a%'** -> Encontra todos os valores que **começam com 'a'**
... `WHERE name LIKE` **'%a'** -> Encontra todos os valores que **acabam com 'a'**
... `WHERE name LIKE` **'%or%'** -> Encontra todos os valores com **'or' em qualquer posição**
... `WHERE name LIKE` **'_r%'** -> Encontra todos os valores com **'r' na segunda posição**
... `WHERE name LIKE` **'a_%'** -> Encontra todos os valores **começados por 'a' com lenght>=2**
... `WHERE name LIKE` **'a__%'** -> Encontra todos os valores **começados por 'a' com lenght>=3**
... `WHERE name LIKE` **'x__'** -> Encontra todos os valores **começados por 'x' com lenght=3**

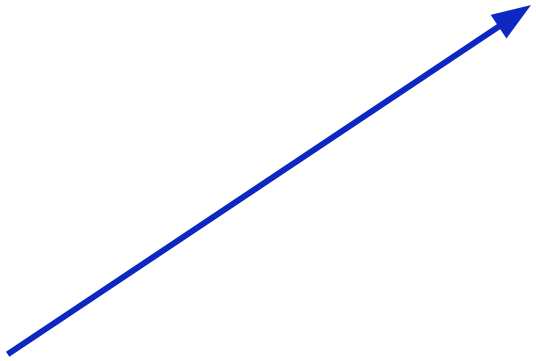
- Também existe o operador **`NOT LIKE`**, que nega o resultado da expressão **`LIKE`**.

IS NULL

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT *
FROM Estudante
WHERE Curso IS NOT NULL
```



Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20007	Pedro Romano	Informática
...	...	...

```
SELECT *
FROM Estudante
WHERE Curso IS NULL
```

Número	Nome	Curso
20006	José Serro	Null
...	...	...

SELECT <Campos>
FROM <Tabelas>
WHERE <Filtro>
ORDER BY <Campo> **ASC/DESC**

ORDER BY

Com o ORDER BY, podemos ordenar os registos devolvidos pela query.

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT *
FROM Estudante
ORDER BY Número
```

Nota: por defeito, a ordem assumida é ascendente (ASC).

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

ORDER BY

Ao ter mais do que um critério de ordenação, o segundo é utilizado em caso de empate.

Tabela “Estudante”

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
…	…	…

```
SELECT *
FROM Estudante
ORDER BY Número, Nome
```

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
…	…	…

ORDER BY

Utilizamos **ASC** ou **DESC** para ordenar de forma ascendente ou descendente.

Tabela “Estudante”

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT *
FROM Estudante
ORDER BY Número DESC
```

Número	Nome	Curso
20007	Pedro Romano	Informática
20006	José Serro	Null
20005	Jorge Rafael	História
20004	João Pavia	Sociologia
20003	João Antão	Marketing
20002	Daniel F.	Informática
20001	António Brito	Economia
...	...	...

SELECT <Campos>
FROM <Tabelas>
LIMIT <N° Registos>

LIMIT

O comando LIMIT é colocado no fim da Query e limita o número de resultados.

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
SELECT *
FROM Estudante
LIMIT 5
```

Nota: a seguir ao LIMIT, colocamos o numero de registos pretendidos (ex. `LIMIT 5`, `LIMIT 100`, etc...)

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História

INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

INSERT

O comando INSERT permite inserir uma ou várias linhas em simultâneo. Este comando é composto por duas cláusulas:

- **INSERT INTO** <tabela onde inserir> (colunas onde vão ser inseridos os valores)
- **VALUES** <valores a inserir>

INSERT INTO <Tabela> **(**<Colunas>**)**
VALUES <Valores>

INSERT

Utilizamos o comando insert para inserir um novo registo na tabela:

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
…	…	…

```
INSERT INTO Estudante
(Número, Nome, Curso)
VALUES (20008, 'Ricardo
Ribeiro', 'Informática')
```

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
20008	**Ricardo Ribeiro**	**Informática**
…	…	…

UPDATE

O comando UPDATE é composto por três cláusulas:

- **UPDATE** <tabela a alterar>
- **SET** <coluna a alterar> = <valor a definir>
- **WHERE** <seleciona as linhas a alterar>

UPDATE <Tabela>

SET <Coluna> **=** <Valor>

WHERE <Filtro>

UPDATE

Podemos **atualizar** todos os campos de uma tabela que cumpram uma condição:

Tabela "Estudante"

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
UPDATE Estudante
SET Curso='Medicina'
WHERE Curso='Informática'
```

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	**Medicina**
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	**Medicina**
...	...	...

DELETE

O comando DELETE é composto por duas cláusulas:

- **DELETE FROM** <tabela a eliminar>
- **WHERE** <seleciona as linhas que pretendemos apagar>

DELETE FROM <Tabela>
WHERE <Filtro>

DELETE

Podemos **apagar** todos os campos de uma tabela que cumpram uma condição:

Tabela “Estudante”

Número	Nome	Curso
20001	António Brito	Economia
20002	Daniel F.	Informática
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
20007	Pedro Romano	Informática
...	...	...

```
DELETE FROM Estudante
WHERE Curso=’Informática’
```

Número	Nome	Curso
20001	António Brito	Economia
20003	João Antão	Marketing
20004	João Pavia	Sociologia
20005	Jorge Rafael	História
20006	José Serro	Null
...	...	...

Configuração do ambiente

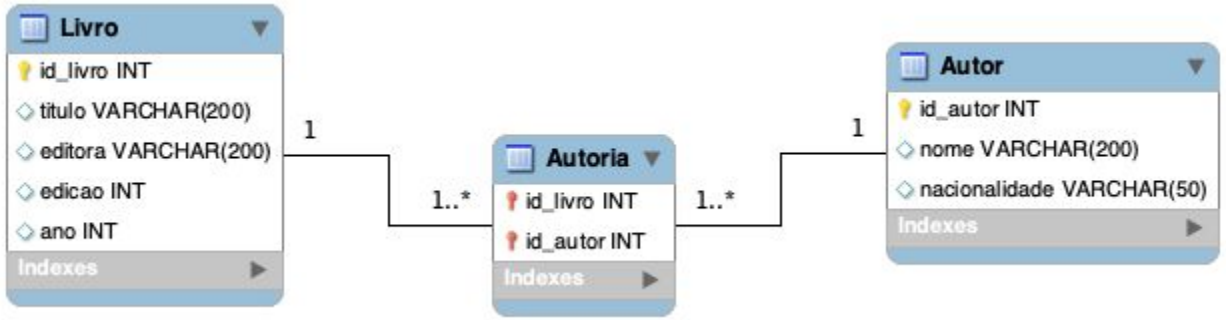
1. Instalar o Xampp: **https://www.apachefriends.org/index.html**
    a. Para mac, ir a:
    https://sourceforge.net/projects/xampp/files/XAMPP%20Mac%20OS%20X/8.1.2/
    e instalar versão mais recente **sem ser a VM**

1. Instalar o mySQL Workbench: **https://dev.mysql.com/downloads/workbench/**
    b. Se o link sugerir o “mySQL Installer for Windows”, não é necessário (é um pack com mu
    coisas), basta o download de baixo só com o Workbench

iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

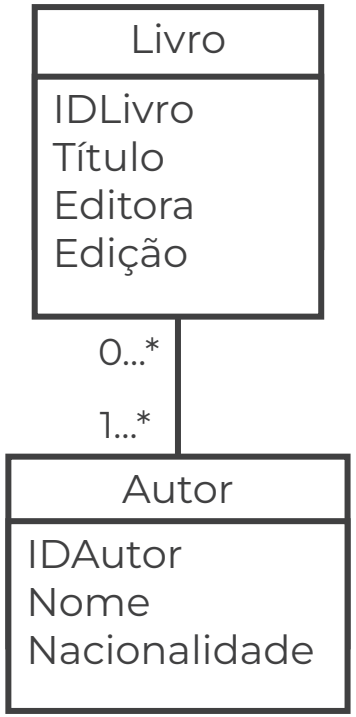
Base de dados que vamos usar

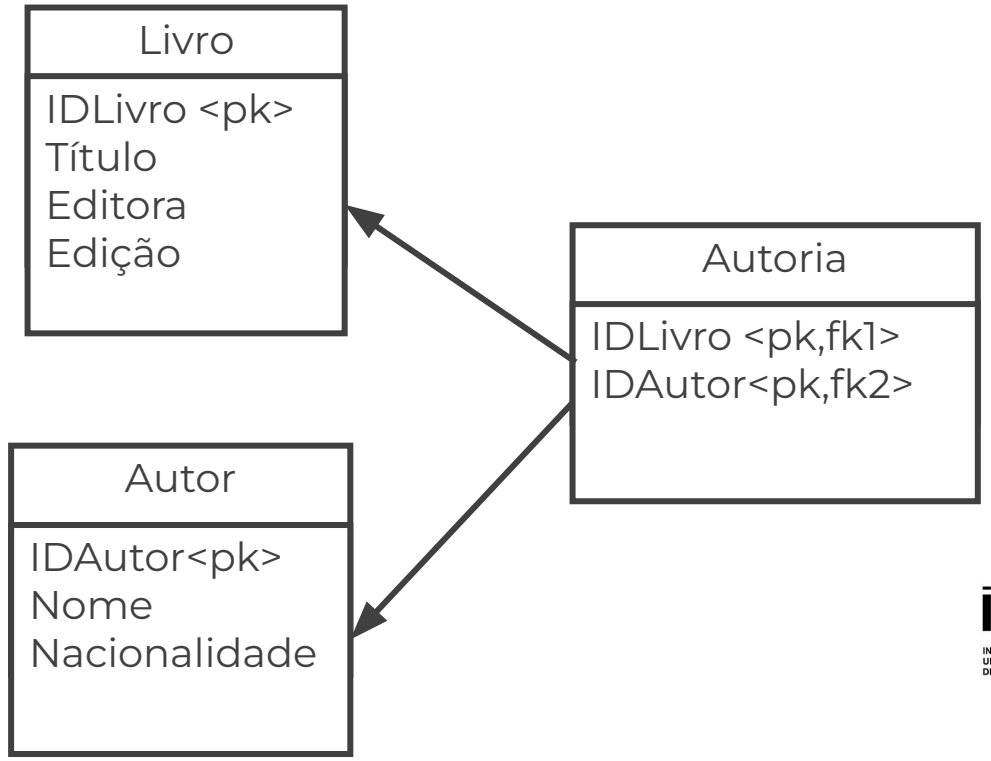
1. Criar a BD no mySQL Workbench de acordo com modelo abaixo:
2. Importar os dados com o ficheiro dados_bd_livros.sql



Exercício 1

Escreva o código SQL que permite listar **toda a informação dos livros existentes, ordenada por título.**





iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

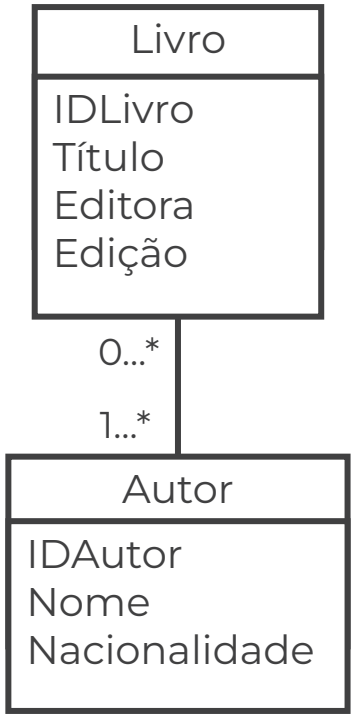
Exercício 1 - **Resolução**

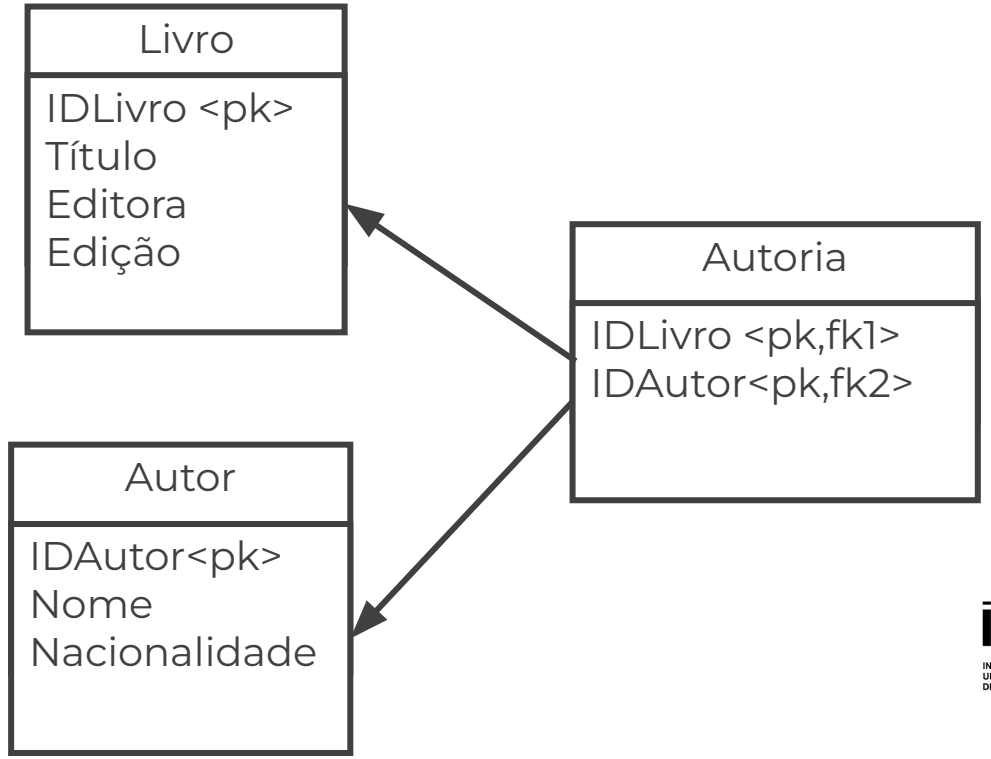
Escreva o código SQL que permite listar **toda a informação dos livros existentes, ordenada por título.**

```
SELECT *
FROM Livro
ORDER BY Título ASC
```

Exercício 2

Escreva o código SQL que permite listar **as editoras existentes.**





iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

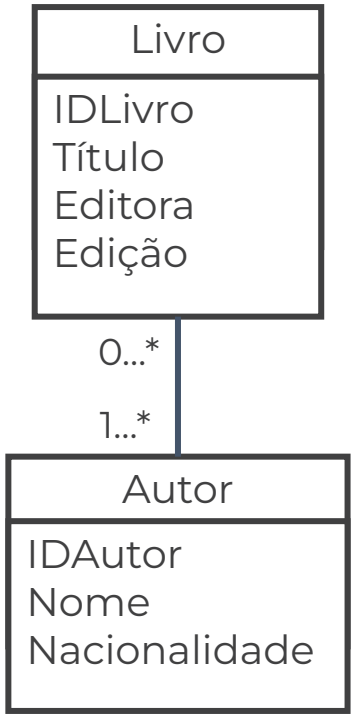
Exercício 2 - **Resolução**

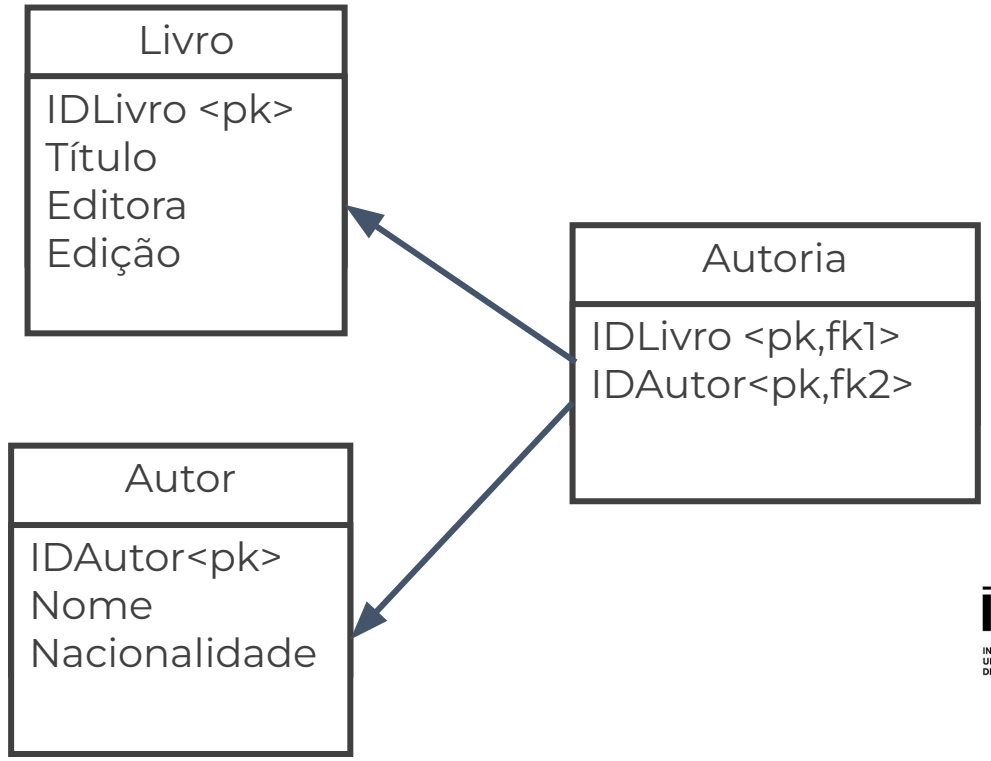
Escreva o código SQL que permite listar **as editoras existentes.**

```
SELECT DISTINCT Editora
FROM Livro
```

Exercício 3

Escreva o código SQL que permite listar as informações dos livros da editora “Relógio Dágua”.





iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

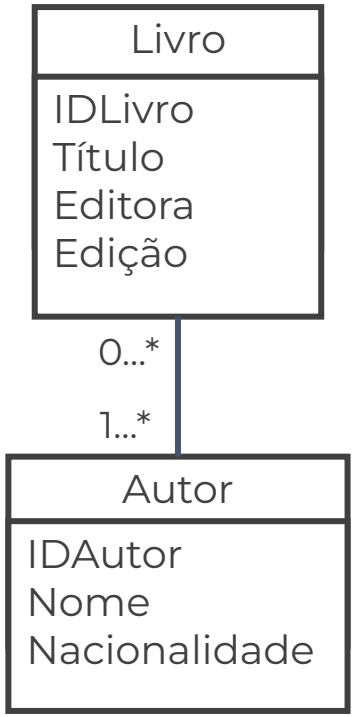
Exercício 3 - **Resolução**

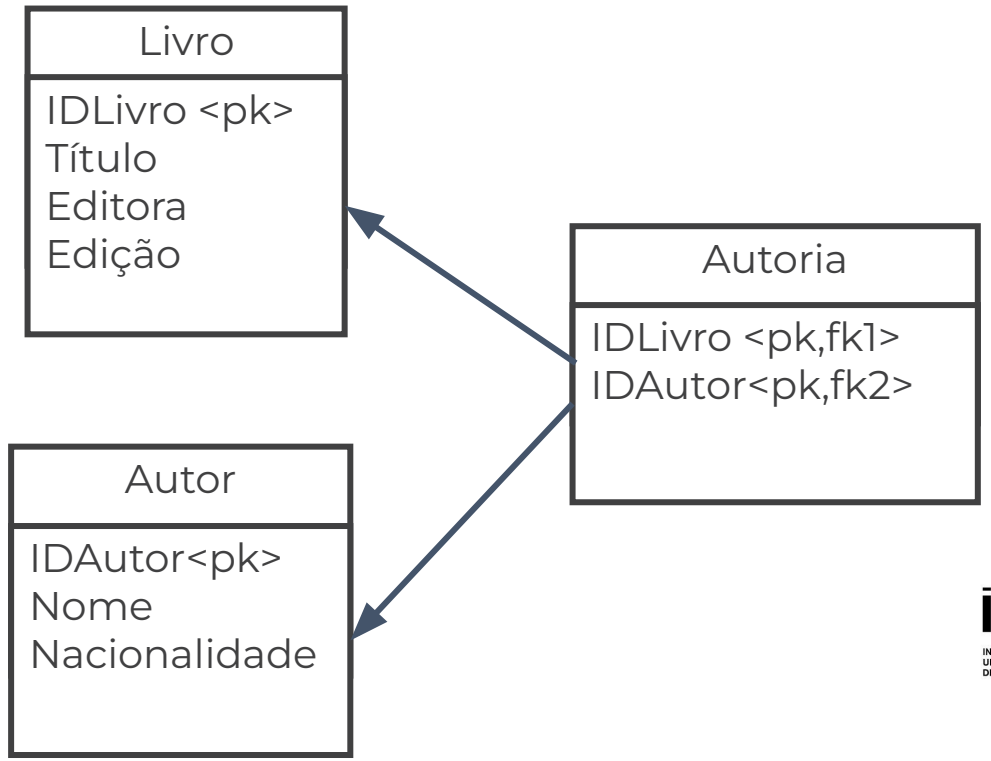
Escreva o código SQL que permite listar as informações dos livros da editora “Relógio Dágua”.

```
SELECT *
FROM livro
WHERE editora=”Relógio Dágua”
```

Exercício 4

Escreva o código SQL que permite **inserir** um novo livro no sistema. O livro tem o título "*O teu olhar*", é da editora "Dom Quixote" e trata-se da 2ª edição (de 2022).





iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

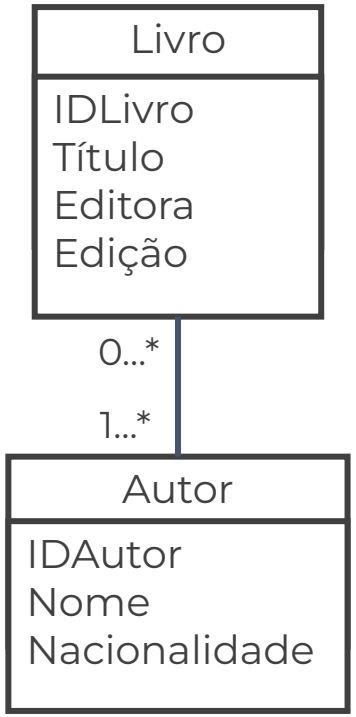
Exercício 4 - **Resolução**

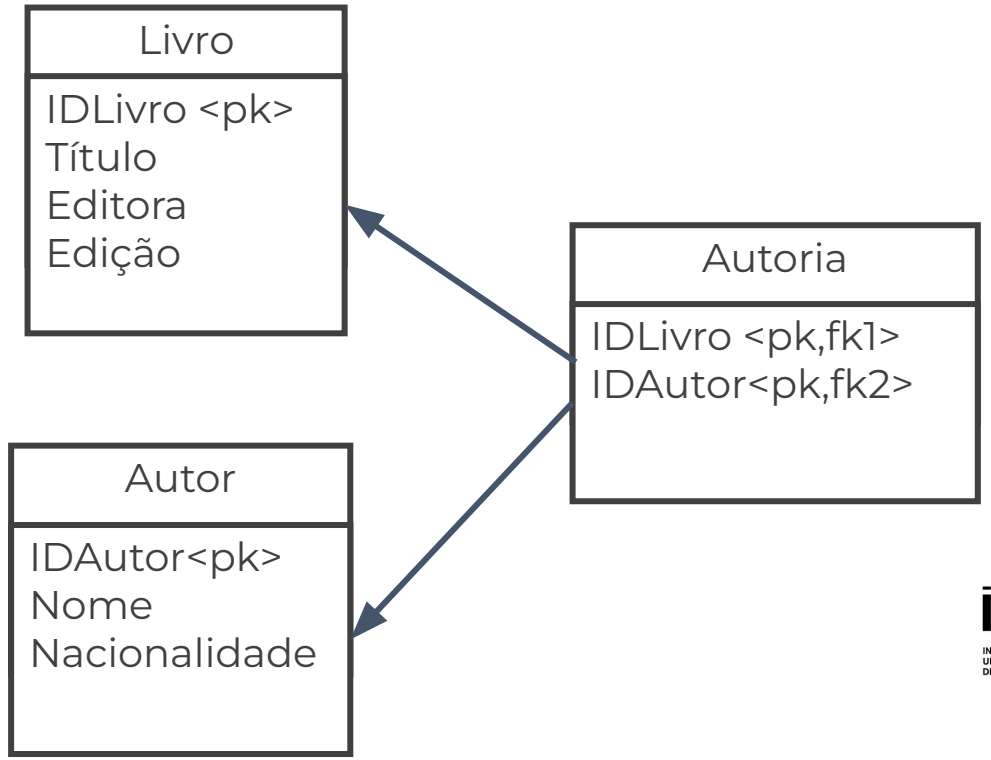
Escreva o código SQL que permite **inserir** um novo livro no sistema. O livro tem o título "*O teu olhar*", é da editora "Dom Quixote" e trata-se da 2ª edição.

```
INSERT INTO livro (titulo, editora, edicao, ano)
VALUES ("O teu olhar", "Dom Quixote", 2, 2022)
```

Exercício 5

Escreva o código SQL que permite **eliminar** do sistema o livro com id 4.





iscte INSTITUTO UNIVERSITÁRIO DE LISBOA emprego digital

Exercício 5 - **Resolução**

Escreva o código SQL que permite **eliminar** do sistema o livro com id 4.

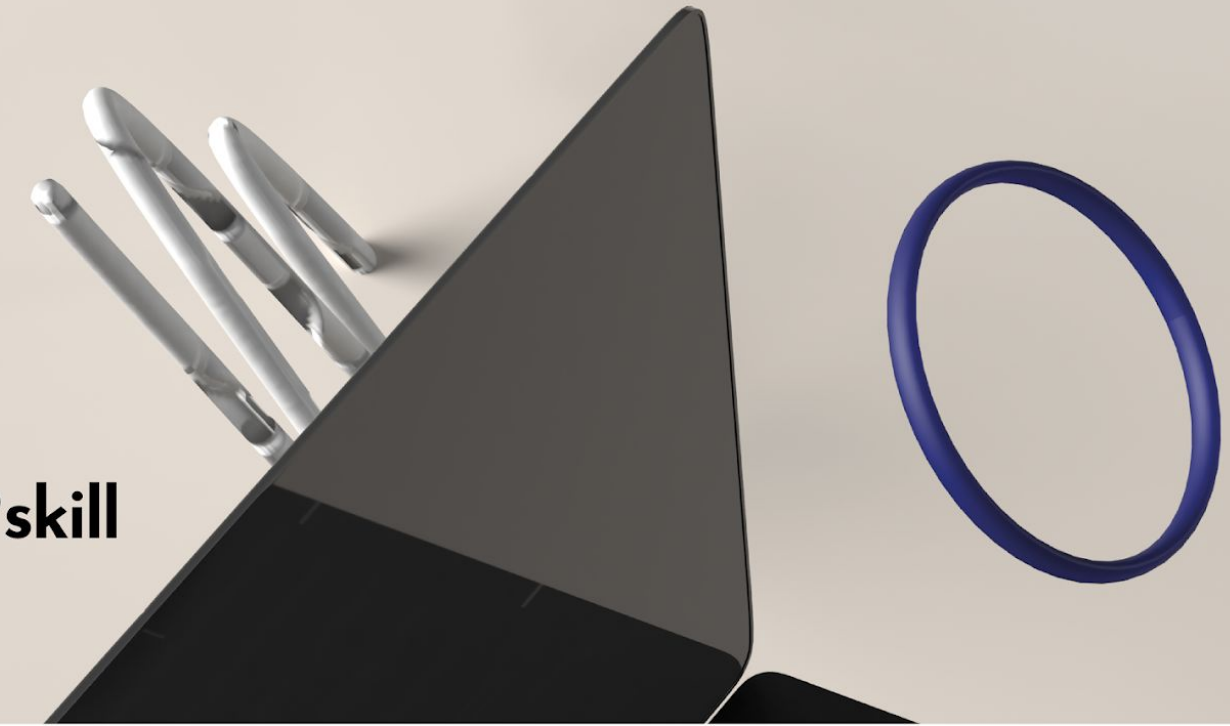
```
DELETE FROM livro
WHERE id_livro=4
```

Mais sobre **SQL**

Ótimo site para consultar “cábulas” sobre cada um dos comandos SQL:

- https://www.w3schools.com/sql/

O futuro profissional começa aqui



iscte INSTITUTO UNIVERSITÁRIO DE LISBOA

emprego digital

UPskill