Banco de Dados I - SQL

Tecnólogo em Análise e Desenvolvimento de Sistemas - Faculdades SENAC/PE

Professor: Danilo Farias

Linguagem SQL

- **Structured Query Language (SQL)**, ou Linguagem de Consulta Estruturada ou SQL, é a linguagem de pesquisa declarativa padrão para banco de dados relacional.
- A linguagem SQL é um grande padrão de banco de dados, o que decorre da sua **simplicidad**e e **facilidade de uso**.
- **SQL** se diferencia de outras linguagens de consulta a banco de dados no sentido em que uma consulta **SQL** especifica a **forma do resultado** e **não o caminho para chegar a ele**.
- Todo **Sistema Gerenciador de Banco de Dados (SGBD)** deve oferecer aos seus usuários e administradores meios de criar definições de dados, bem como manipular esses dados armazenados em suas bases.

Linguagem SQL - Histórico

- O SQL foi desenvolvido originalmente no início dos anos 1970 nos laboratórios da IBM em San Jose, dentro do projeto System R. Em 1970, um pesquisador da IBM, Ted Codd, visionou um sistema onde o usuário seria capaz de acessar as informações através de comandos em inglês, onde as informações estariam armazenadas em tabelas.

- O nome original da linguagem era **SEQUEL**, acrônimo para **Structured English Query Language** (Linguagem de Consulta Estruturada em Inglês).

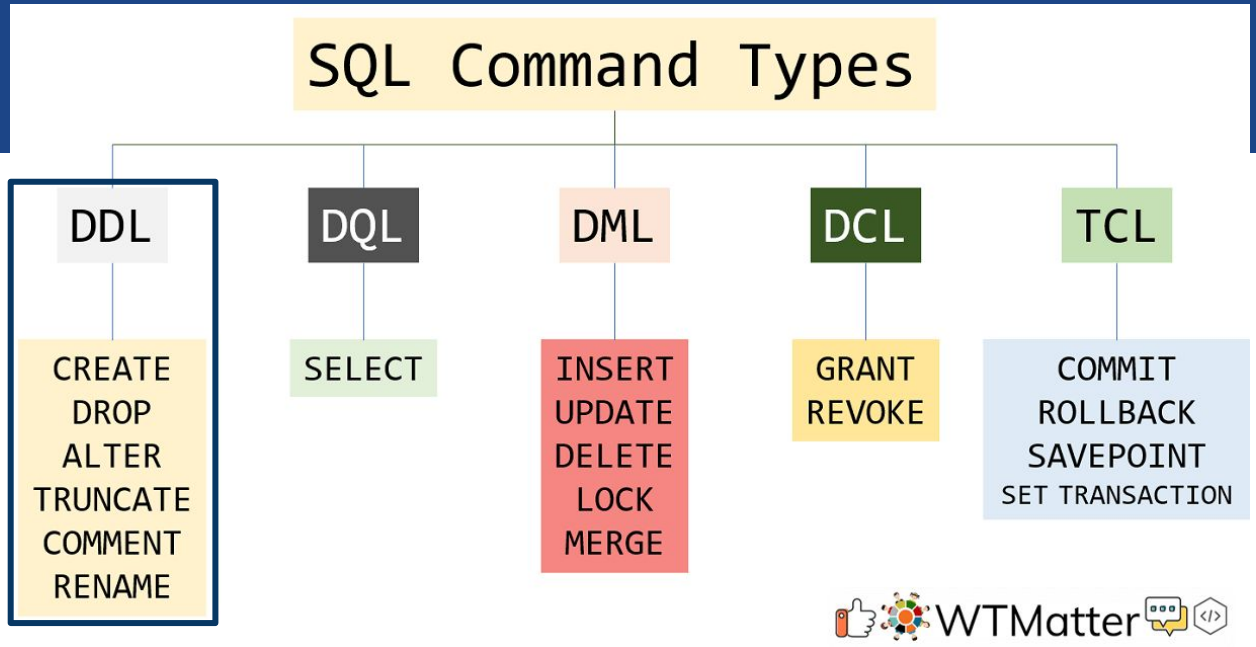
Linguagem SQL - Histórico

- Embora o SQL tenha sido originalmente criado pela IBM, rapidamente surgiram vários "**dialetos**" produzidos por outros desenvolvedores.
- Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem.
- Esta tarefa foi realizada pela **American National Standards Institute (ANSI)** em 1986 e **International Organization for Standardization (ISO)** em 1987.
- O SQL foi revisto em 1992 e a essa versão foi dado o nome de **SQL-92**. Foi revisto novamente em 1999 e 2003 para se tornar **SQL:1999 (SQL3)** e **SQL:2003**, respectivamente.

Linguagem SQL - Tipos de Comandos

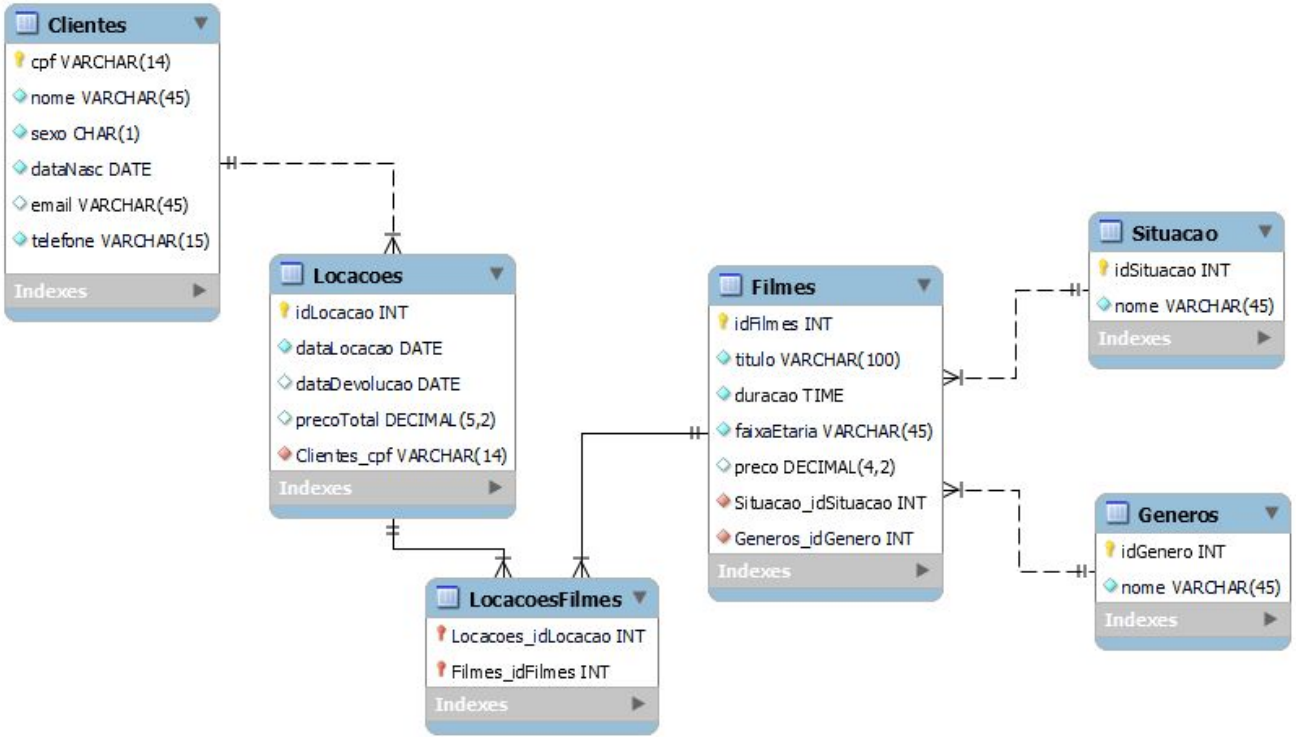
- **SQL** é caracterizada pela utilização de palavras-chaves que podem ser classificadas, de acordo com sua função, nos seguintes tipos:
  - **DML – Linguagem de Manipulação de Dados**, subconjunto da linguagem usado para inserir, atualizar e apagar dados. Ex.: INSERT, UPDATE, DELETE;
  - **DDL – Linguagem de Definição de Dados**, permite ao utilizador definir tabelas novas e elementos associados. Ex.: CREATE, DROP;
  - **DCL – Linguagem de Controle de Dados**, controla quem tem acesso para ver ou manipular dados dentro do banco de dados. Ex.: GRANT, REVOKE;
  - **DTL – Linguagem de Transação de Dados**, usado para o controle de transações no banco de dados. Ex.: START TRANSACTION;
  - **DQL – Linguagem de Consulta de Dados**, permite ao usuário especificar uma consulta (query) como uma descrição do resultado desejado. Ex.: SELECT

SQL - Linguagem de Definição de Dados (DDL)



SQL - Linguagem de Definição de Dados (DDL)

- Antes de iniciarmos nossa jornada em **SQL**, vamos primeiro modelar um **Diagrama Relacional** do nosso mini-mundo. Iremos usar como estudo de caso uma **Locadora de Filmes**, que será nosso campo de estudo.



SQL - Linguagem de Definição de Dados (DDL)

- **SQL-DDL**:
  - Para iniciar nosso projeto de construção de um Banco de Dados vamos primeiro aprender como se **cria uma Base de Dados**;
    - **CREATE DATABASE locadorafilmes;**
  - Após criar um Banco de Dados, vamos **usar** o mesmo para os próximos comandos;
    - **USE locadorafilmes;**
  - O próximo comando é o de **excluir/remover** uma **Base de Dados**
    - **DROP locadorafilmes;**

SQL - Linguagem de Definição de Dados (DDL)

- **SQL-DDL**:
  - Com o Banco de Dados criado e já em uso, vamos aprender como **criar** uma **Tabela**.
  - A sintaxe é:

```
CREATE TABLE nome_da_tabela(
        atributo 1 tipo1,
        atributo 2 tipo 2,
        ...
        atributo N tipo N
);
```

SQL - Linguagem de Definição de Dados (DDL)

- **SQL-DDL**:
  - Vamos criar nossas Tabelas de acordo com o Diagrama Relacional

```
create table Clientes(
        cpf varchar(14) not null primary key,
        nome varchar(45) not null,
        sexo char(1) not null,
        dataNasc date not null,
        email varchar(45) not null unique,
        telefone varchar(15) not null
);

desc Clientes;
```

```
create table Generos(
        idGenero int not null auto_increment primary key,
        nome varchar(45) not null
);
desc generos;

create table Situacao(
        idSituacao int not null auto_increment,
        nome varchar(45) not null,
        primary key(idSituacao)
);
drop table situacao;
desc situacao;
```

SQL - Linguagem de Definição de Dados (DDL)

- **SQL-DDL**:
  - Vamos criar nossas Tabelas de acordo com o Diagrama Relacional

```
create table Filmes(
        idFilme int not null auto_increment primary key,
        titulo varchar(100) not null,
        duracao time not null,
        ano int not null,
        faixaEtaria varchar(45) not null,
        preco decimal(4,2),
        Situacao_idSituacao int not null,
        Generos_idGenero int not null,
        foreign key(Situacao_idSituacao) references Situacao(idSituacao),
        foreign key(Generos_idGenero) references Generos(idGenero)
);
desc filmes;
```

SQL - Linguagem de Definição de Dados (DDL)

- **SQL-DDL**:
  - Vamos criar nossas Tabelas de acordo com o Diagrama Relacional

```
create table Locacoes (
	idLocacao int not null auto_increment primary key,
	dataLocacao date not null,
	dataDevolucao date,
	totalPreco decimal(5,2),
	Cliente_cpf varchar(14) not null,
	foreign key(Cliente_cpf) references Clientes(cpf)
);
desc locacoes;
```

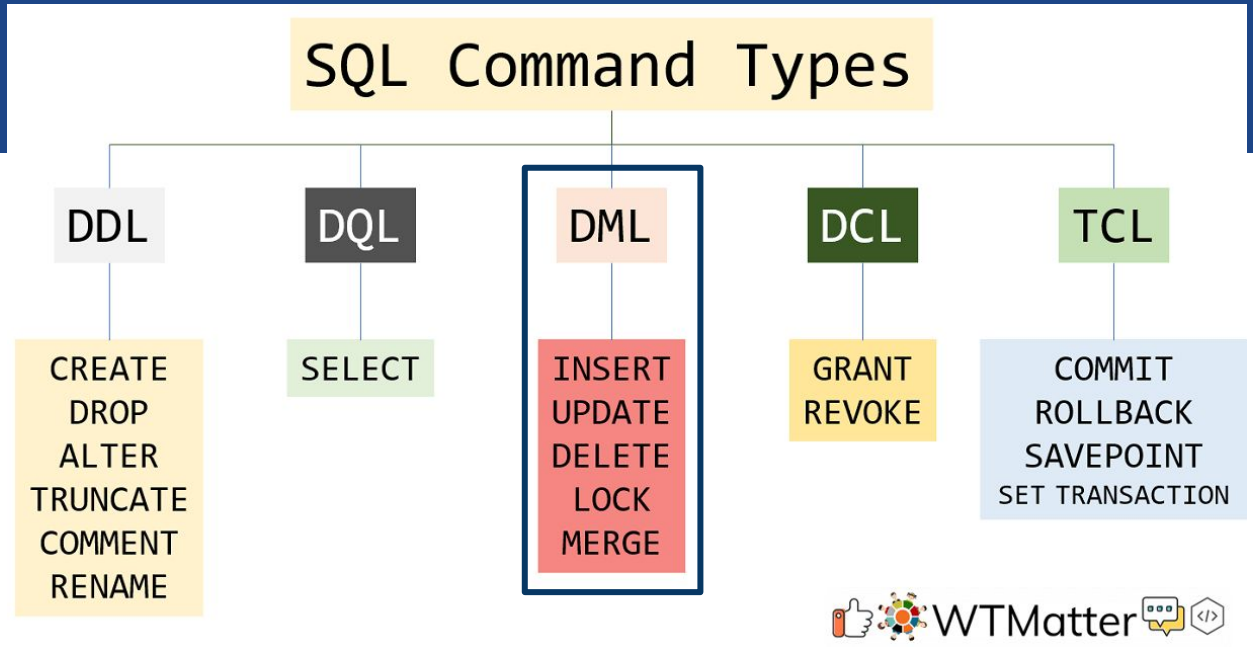
```
create table LocacoesFilmes (
	Locacoes_idLocacao int not null,
	Filmes_idFilme int not null,
	primary key(Locacoes_idLocacao, Filmes_idFilme),
	foreign key(Locacoes_idLocacao) references Locacoes(idLocacao),
	foreign key(Filmes_idFilme) references Filmes(idFilme)
);
desc LocacoesFilmes;
```

SQL - Linguagem de Definição de Dados (DDL)

- **SQL-DDL**:
  - Caso precise **modificar** alguma coluna da **Tabela** podemos usar o **Alter**

```
ALTER TABLE clientes ADD endereco VARCHAR(100);
desc clientes;

alter table clientes drop column endereco;

ALTER TABLE clientes ADD endereco VARCHAR(100) not null;

alter table filmes change column preco precoLoc decimal(4,2);

alter table filmes change column precoLoc preco decimal(3,2);
```

SQL - Linguagem de Manipulação de Dados (DML)



SQL - Linguagem de Manipulação de Dados (DML)

- **SQL-DML**:
  - **Linguagem de Manipulação de Dados**, subconjunto da linguagem usado para inserir, atualizar e apagar dados. Ex.: **INSERT**, UPDATE, DELETE;

```
insert into clientes (cpf, nome, sexo, dataNasc, email, telefone)
	value ("123.789.789-96", "Pedro Henrique", "M", "2000-09-12", "pedroh@gmail.com",
"8198685878482");

select * from clientes;

insert into clientes (cpf, nome, sexo, dataNasc, email, telefone)
	values ("123.789.789-87", "Mauro Dantas", "M", "1995-12-12", "maurodantas@gmail.com",
"81998985652"),
			("123.789.789-88", "Alvaro Monteiro", "M", "1992-01-10", "alvarom@gmail.com",
"819999985754");
```

SQL - Linguagem de Manipulação de Dados (DML)

- **SQL-DML**:
  - **Linguagem de Manipulação de Dados**, subconjunto da linguagem usado para inserir, atualizar e apagar dados. Ex.: **INSERT**, UPDATE, DELETE;

```
insert into generos (nome)
        values ('Ação'),
                ('Terro'),
                ('Comédia'),
                ('Drama'),
                ('Ficção Científica'),
                ('Desenho'),
                ('Romance');

select * from generos;
```

```
insert into situacao (nome)
        values ('Disponível'),
                ('Locado'),
                ('Rersevado'),
                ('Avariado'),
                ('Furtado'),
                ('Vendido');

select * from situacao;
```

SQL - Linguagem de Manipulação de Dados (DML)

- **SQL-DML**:
  - **Linguagem de Manipulação de Dados**, subconjunto da linguagem usado para inserir, atualizar e apagar dados. Ex.: **INSERT**, UPDATE, DELETE;

```
insert into filmes (titulo, duracao, ano, faixaEtaria, preco, Situacao_idSituacao, Generos_idGenero)
	values ('Star Wars - Despertar da Força', '2:16:00', 2015, '16+', 3.5, 1, 5),
		('Star Wars - Despertar da Força', '2:16:00', 2015, '16+', 3.5, 1, 5),
		('Star Wars - Despertar da Força', '2:16:00', 2015, '16+', 3.5, 2, 5),
		('Star Wars - Ascensão Skywalker', '2:22:00', 2019, '16+', 4.5, 1, 5),
		('Star Wars - Ascensão Skywalker', '2:22:00', 2019, '16+', 4.5, 2, 5),
		('Star Wars - Ascensão Skywalker', '2:22:00', 2019, '16+', 4.5, 1, 5),
		('Star Wars - Os Últimos Jedi', '2:32:00', 2017, '16+', 3.5, 2, 5),
		('Star Wars - Os Últimos Jedi', '2:32:00', 2017, '16+', 3.5, 1, 5),
		('Star Wars - Os Últimos Jedi', '2:32:00', 2017, '16+', 3.5, 2, 5);

select * from filmes;
```

SQL - Linguagem de Manipulação de Dados (DML)

- **SQL-DML**:
  - **Linguagem de Manipulação de Dados**, subconjunto da linguagem usado para inserir, atualizar e apagar dados. Ex.: INSERT, **UPDATE**, DELETE;

```
update filmes
	set preco = 4.5
		where ano = 2020;

update filmes
	set preco = 4.5
		where Generos_idGenero = 5;
```

```
update filmes
	set preco = preco - 1
		where Generos_idGenero = 5;

update filmes
	set preco = preco + 1;
```

Obs.: SET SQL_SAFE_UPDATES = 0;

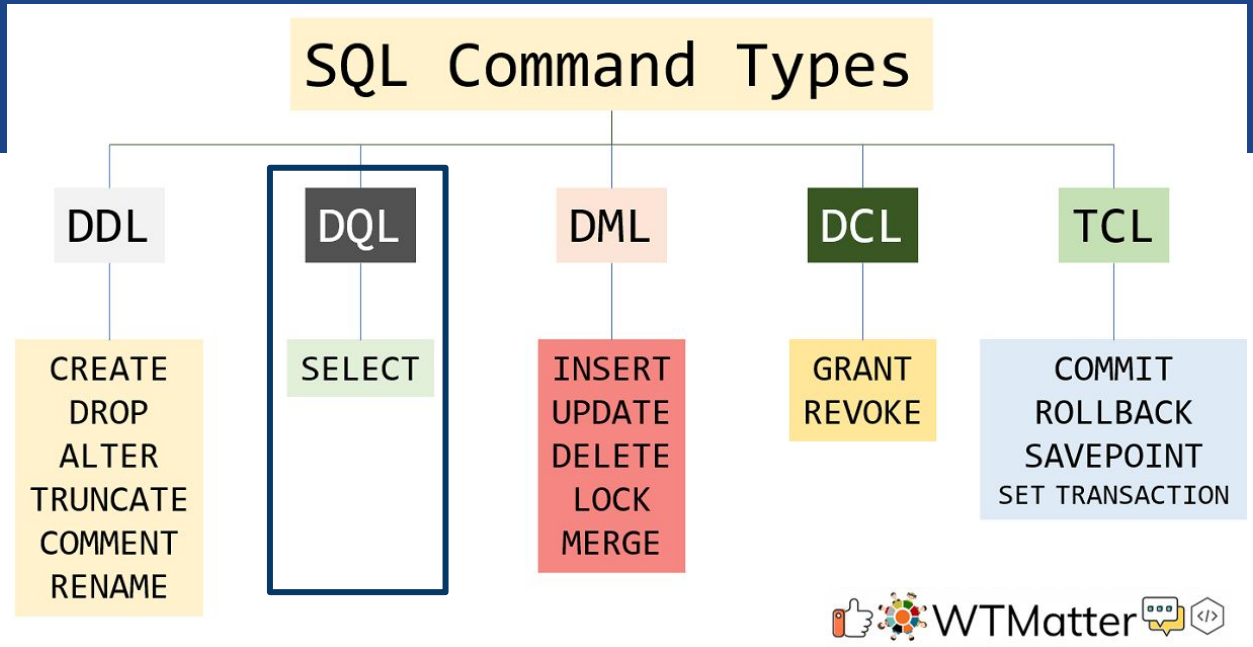
SQL - Linguagem de Manipulação de Dados (DML)

- **SQL-DML**:
  - **Linguagem de Manipulação de Dados**, subconjunto da linguagem usado para inserir, atualizar e apagar dados. Ex.: INSERT, UPDATE, **DELETE**;

```
delete from situacao
        where idSituacao = 7;

select * from situacao;
```

```
delete from situacao
        where idSituacao in (8,9,10,11,12);

delete from filmes
        where nome = 'Deadpool';

delete from clientes
        where cpf = '123.789.456-90';
```

SQL - Linguagem de Query de Dados (DQL)



SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – Linguagem de Consulta de Dados**, permite ao usuário especificar uma consulta (query) como uma descrição do resultado desejado. Ex.: **SELECT**

```
select titulo, duracao, ano from filmes;

select * from filmes;

select titulo, duracao, faixaEtaria, ano, preco
from filmes
        where Situacao_idSituacao = 1;

select titulo, duracao, faixaEtaria, ano, preco
from filmes
        where Situacao_idSituacao = 1
                order by ano;
```

```
select titulo, duracao, faixaEtaria, ano, preco
from filmes
        where Situacao_idSituacao = 1
                order by ano desc;

select titulo, duracao, faixaEtaria, ano, preco
from filmes
        where Situacao_idSituacao = 1
                order by titulo;

select nome, email, telefone from clientes
        order by nome;
```

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT - WHERE : OPERADORES ARITMÉTICOS**

```
select nome "Nome do Cliente", email as "E-mail", telefone "Telefone" from clientes
	order by nome;

select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
	where Situacao_idSituacao = 1
		and preco < 5
		order by titulo;

select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
	where Situacao_idSituacao = 1
		and preco >= 5
		order by titulo;
```

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT - WHERE : OPERADORES LÓGICOS**

```
select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
        where Situacao_idSituacao = 1
                    and generos_idgenero = 1
                            or generos_idgenero = 5
                order by titulo;

select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
        where Situacao_idSituacao = 1
                    and ano = 2018
                            or ano = 2019
                order by titulo;
```

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT - WHERE**
    - **BETWEEN** - Usado para verificar se o valor de um atributo está em um intervalo de valores. Especifica um intervalo a ser testado;
    - **LIKE** - Utilizada para comparar cadeias de caracteres usando padrões de comparação para um ou mais caracteres. Normalmente, o coringa percentual (%) substitui zero, um ou mais caracteres e o coringa sublinha (_) substitui um único caractere.
    - **IN** - Usado para verificar se o valor de um atributo está em um conjunto de valores entre parênteses. Quando o valor for compatível com um dos valores do conjunto, o registro é exibido.
    - **IS NULL** – Usado para selecionar diretamente um valor NULL

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT - WHERE**

```
select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
	where Situacao_idSituacao = 1
		and ano between 2018 and 2020
			order by titulo;
select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
	where Situacao_idSituacao = 1
		and titulo like "Sta%"
			order by titulo;
select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
	where Situacao_idSituacao = 1
		and generos_idgenero in (1, 6, 7)
			order by titulo;
```

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT - SUBCONSULTAS**

```
select idsituacao from situacao where nome like "Dispo%";

select idgenero from generos where nome like "Ação" or nome like "Romance";

select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
        where Situacao_idSituacao = (select idsituacao from situacao where nome like "Dispo%")
                and generos_idgenero in (select idgenero from generos where nome like "Ação" or nome
like "Romance")
                        order by titulo;
```

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT - SUBCONSULTAS**

```
select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
        where Situacao_idSituacao = (select idsituacao from situacao where nome like "Dispo%")
                and generos_idgenero in (select idgenero from generos where nome like "Ação" or nome
like "Romance")
                        order by ano desc, titulo;

select titulo "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
        where Situacao_idSituacao = (select idsituacao from situacao where nome like "Dispo%")
                and generos_idgenero in (select idgenero from generos where nome like "Ação" or nome
like "Romance")
                        order by titulo, ano desc;
```

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT : FUNÇÕES**
    - ABS (valor) : Retorna o valor absoluto (positivo) do valor informado;
    - AVG (valor) : Retorna a média dos valores ou valor informado;
    - FLOOR (valor) : Retorna o maior número inteiro, igual ou menor ao valor informado;
    - ROUND (valor, n) : Arredonda o valor informado para n casas decimais;
    - POWER (valor, p) : Retorna o valor informado elevado à potência p;
    - LEN (expressão) Retorna o número de caracteres contidos na expressão informada;
    - LOWER (expressão) e UPPER (expressão) : Converte para minúsculo e maiúsculo a expressão informada, respectivamente;
    - LTRIM (expressão) e RTRIM (expressão) : Remove os espaços em branco à esquerda e à direita da expressão informada, respectivamente;
    - SUBSTRING (expressão, início, tamanho) : Extrai uma parte dos caracteres da expressão, iniciando da posição informada em início, considerando a quantidade definida em tamanho;

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT : FUNÇÕES**

```
select max(preco) "Maior Preço de Locação" from filmes;
select min(preco) "Menor Preço de Locação" from filmes;

select avg(preco) from filmes;
select round(avg(preco), 2) "Preço Médio de Locação" from filmes;

select upper(titulo) "Título", duracao "Duração", faixaEtaria "Fixa Etária", ano "Ano", preco "Preço" from filmes
        where Situacao_idSituacao = (select idsituacao from situacao where nome like "Dispo%")
                and generos_idgenero in (select idgenero from generos where nome like "Ação" or nome like "Romance")
                        order by titulo, ano desc;
```

SQL - Linguagem de Manipulação de Dados (DQL)

- **SQL-DQL**:
  - **DQL – SELECT - CONSULTA EM MÚLTIPLAS TABELAS**

```
select f.titulo "Título", f.duracao "Duração",
g.nome "Gênero", f.faixaEtaria "Fixa Etária",
f.ano "Ano", f.preco "Preço", s.nome "Situação"
        from filmes f, generos g, situacao s
                where f.Generos_idGenero =
        g.idGenero
                and f.Situacao_idSituacao =
        s.idSituacao
                        order by f.titulo;
```

```
select f.titulo "Título", f.duracao "Duração",
f.faixaEtaria "Faixa Etaria", f.ano "Ano", g.nome
"Gênero", s.nome "Situação", f.preco "Preço"
        from filmes f, generos g, situacao s
                where f.Situacao_idSituacao =
s.idSituacao
                and f.Genero_idGenero = g.idGenero;
```

SQL - Linguagem de Manipulação de Dados (DQL)

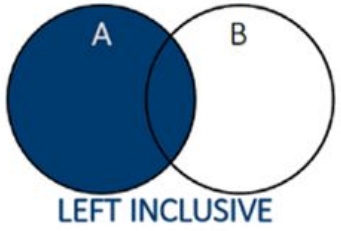
- **SQL-DQL**:
  - **DQL – SELECT - CONSULTA EM MÚLTIPLAS TABELAS**

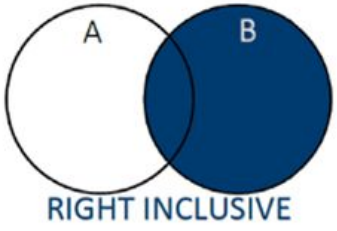
```
select f.titulo "Título", f.duracao "Duração",
f.faixaEtaria "Faixa Etaria", f.ano "Ano",
g.nome "Gênero", s.nome "Situação", f.preco
"Preço"
        from filmes f, generos g, situacao s
                where f.Situacao_idSituacao =
        s.idSituacao
                and f.Genero_idGenero = g.idGenero
                and f.preco <= 5
                and f.Situacao_idSituacao = 1
                        order by f.titulo;
```

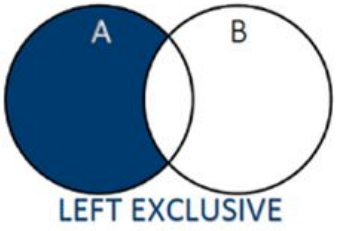
```
select f.titulo "Título", f.duracao "Duração",
f.faixaEtaria "Faixa Etaria", f.ano "Ano",
g.nome "Gênero", s.nome "Situação", f.preco
"Preço"
        from filmes f, generos g, situacao s
                where f.Situacao_idSituacao =
        s.idSituacao
                and f.Genero_idGenero = g.idGenero
                and f.ano between 2018 and 2021
                and f.Situacao_idSituacao = 1
                        order by f.titulo;
```

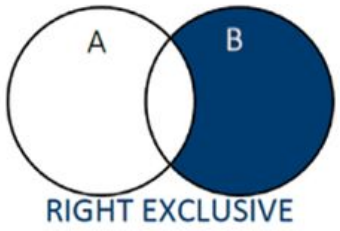
SQL - Linguagem de Manipulação de Dados (DQL)

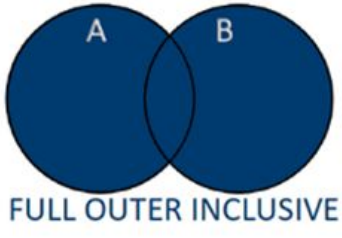
- **SQL-DQL**:
  - **DQL – SELECT - JOINS**
    - https://www.devmedia.com.br/sql-select-guia-para-iniciantes/29530

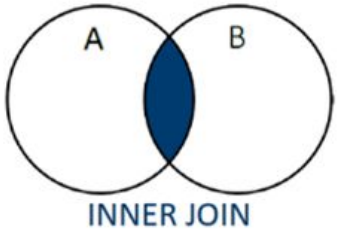


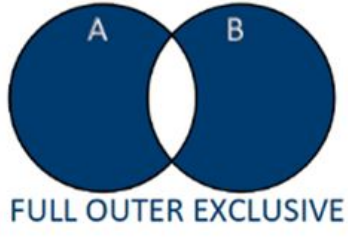












SQL - Linguagem de Manipulação de Dados (DQL)

LEFT JOIN

Everything on the left
+
anything on the right that matches

```
SELECT *
FROM TABLE_1
LEFT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI LEFT JOIN

Everything on the left
that is NOT on the right

```
SELECT *
FROM TABLE_1
LEFT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_2.KEY IS NULL
```

RIGHT JOIN

Everything on the right
+
anything on the left that matches

```
SELECT *
FROM TABLE_1
RIGHT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI RIGHT JOIN

Everything on the right
that is NOT on the left

```
SELECT *
FROM TABLE_1
RIGHT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_1.KEY IS NULL
```

OUTER JOIN

Everything on the right
+
Everything on the left

```
SELECT *
FROM TABLE_1
OUTER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

ANTI OUTER JOIN

Everything on the left and right
that is unique to each side

```
SELECT *
FROM TABLE_1
OUTER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
WHERE TABLE_1.KEY IS NULL
OR TABLE_2.KEY IS NULL
```

INNER JOIN

Only the things that match on the left AND the right

```
SELECT *
FROM TABLE_1
INNER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

CROSS JOIN

All combination of rows from the right and the left (cartesean product)

```
SELECT *
FROM TABLE_1
CROSS JOIN TABLE_2
```

Dúvidas?

Banco de Dados I - SQL

Tecnólogo em Análise e Desenvolvimento de Sistemas - Faculdades SENAC/PE

Professor: Danilo Farias