

Analista de Dados

Módulo | SQL:

Filtrando & Seleção Condicional

Caderno de Aula

Professor [Mariane Neiva](#)

▼ Tópicos

1. Filtros and, or, in e between;
 2. Filtros like e wildcards;
 3. Seleção condicional;
-

▼ Aulas

Nessa aula, usaremos a seguinte tabela:

```
CREATE TABLE transacoes (  
  id_cliente INT,  
  id_transacao INT PRIMARY KEY,  
  data_compra DATE UNIQUE,  
  valor FLOAT NOT NULL,  
  id_loja varchar(25),  
  CHECK (valor > 0)  
);
```

Também temos os seguintes valores inseridos na tabela

```
INSERT INTO transacoes VALUES (1,768805383,'2021-06-10',50.74,'magalu');
INSERT INTO transacoes VALUES (2,768805399,'2021-06-13',30.90,'giraffas');
INSERT INTO transacoes VALUES (3,818770008,'2021-06-05',110.00,'postoshell');
INSERT INTO transacoes VALUES (1,76856563,'2021-07-10',2000.90,'magalu');
INSERT INTO transacoes VALUES (1,767573759,'2021-06-20',15.70,'subway');
INSERT INTO transacoes VALUES (3,818575758,'2021-06-25',2.99,'seveneleven');
INSERT INTO transacoes VALUES (4,764545534,'2021-07-11',50.74,'extra');
INSERT INTO transacoes VALUES (5,76766789,'2021-08-02',10.00,'subway');
INSERT INTO transacoes VALUES (3,8154567758,'2021-08-15',1100.00,'shopee');
```

Como resultado da função SELECT, temos a seguinte tabela:

id_cliente	id_transacao	data_compra	valor	id_loja
1	768805383	2021-06-10	50.74	magalu
2	768805399	2021-06-13	30.90	giraffas
3	818770008	2021-06-05	110.00	postoshell
1	76856563	2021-07-10	2000.90	magalu
1	767573759	2021-06-20	15.70	subway
3	818575758	2021-06-25	2.99	seveneleven
4	764545534	2021-07-10	50.74	extra
5	76766789	2021-08-02	10.00	subway
3	8154567758	2021-08-15	1100.00	shopee

▼ 1. Filtros and, or, in e between

Temos visto o comando SELECT de diversas formas durante as aulas, hoje vamos complementá-lo ainda mais com os comandos **where**, **and**, **or**, **in** e **between**.

Atenção: o comando **where** pode ser utilizado em conjunto com diversas funcionalidades como DELETE e UPDATE (mostramos isso no módulo 2)!

▼ 1.1 Comando WHERE + AND

O comando **AND** seleciona aquelas instâncias que condizem com duas (ou mais) condições.

```
SELECT <coluna1>, <coluna2>, ...
FROM <nome_tabela>
WHERE <condicao1> AND <condicao2>, ..., AND <condicao3>;
```

No nosso exemplo:

```
SELECT *
FROM transacoes
WHERE valor > 30 AND data_compra > 2020-07-01;
```

O resultado:

id_cliente	id_transacao	data_compra	valor	id_loja
1	76856563	2021-07-10	2000.90	magalu
4	764545534	2021-07-11	50.74	extra
3	8154567758	2021-08-15	1100.00	shopee

1.2 Comando WHERE + OR

O comando **OR** seleciona aquelas instâncias que condizem com pelo menos um dos critérios.

```
SELECT <coluna1>, <coluna2>, ...
FROM <nome_tabela>
WHERE <condicao1> OR <condicao2>, ..., OR <condicao3>;
```

No nosso exemplo:

```
SELECT *
FROM transacoes
WHERE valor > 30 OR data_compra > 2020-07-01;
```

O resultado:

id_cliente	id_transacao	data_compra	valor	id_loja
1	768805383	2021-06-10	50.74	magalu
2	768805399	2021-06-13	30.90	giraffas
3	818770008	2021-06-05	110.00	postoshell
1	76856563	2021-07-10	2000.90	magalu
4	764545534	2021-07-11	50.74	extra
5	76766789	2021-08-92	10.00	subway
3	8154567758	2021-08-15	1100.00	shopee

1.3 Comando WHERE + IN

Ao invés de utilizar vários comandos OR, podemos utilizar o comando **IN**

```
SELECT <coluna1>, <coluna2>, ..., <coluna>
FROM <nome_tabela>
WHERE <coluna> IN (<valor1>, <valor2>, ...)
```

No nosso exemplo:

```
SELECT *
FROM transacoes
WHERE id_loja IN ('magalu','subway') AND valor > 10;
```

O resultado:

id_cliente	id_transacao	data_compra	valor	id_loja
1	768805383	2021-06-10	50.74	magalu
1	76856563	2021-07-10	2000.90	magalu
1	767573759	2021-06-20	15.70	subway

1.4 Comando WHERE + BETWEEN

Com o comando BETWEEN, podemos encontrar instâncias entre < valor1 > e < valor2 >

```
SELECT <coluna1>, <coluna2>, ..., <coluna>
FROM <nome_tabela>
WHERE <coluna> BETWEEN <valor1> AND <valor2>
```

No nosso exemplo:

```
SELECT *
FROM transacoes
WHERE valor BETWEEN 60 AND 1000;
```

O resultado:

id_cliente	id_transacao	data_compra	valor	id_loja
3	818770008	2021-06-05	110.00	postoshell

2. Filtros like e wildcards

Muitas vezes precisamos encontrar valores de texto dentro da tabela. Em alguns casos, poderíamos utilizar buscas exatas com as condições where e o '='.

No entanto, nem sempre temos o valor exato da busca e para isso podemos utilizar os *wildcards* (na tabela) junt o com a palavra **LIKE**. No SQL temos os seguintes elementos para utilizar:

Elemento	Onde usar
%	Pode ser substituído por 0 ou mais caracteres
_	Pode ser substituído por 1 caracter
[]	Pode ser substituído pelos caracteres indicados dentro dos []
^	Pode ser substituído pelos caracteres que NÃO estão indicados dentro dos [^]

Como utilizar

```
SELECT <coluna1>, <coluna2>, ...  
FROM <nome_tabela>  
WHERE <coluna> LIKE <padrao_utilizando_wildcards>;
```

No nosso caso para encontrar 'magalu' podemos utilizar

```
WHERE id_loja LIKE 'mag%'  
WHERE id_loja LIKE 'mag_lu'  
WHERE id_loja LIKE '[mn]agalu'  
WHERE id_loja LIKE 'mag[^ou]lu'  
WHERE id_loja LIKE 'maga[a-k]u'
```

Vamos testar uma das formas?

```
SELECT *  
FROM transacoes  
WHERE id_loja LIKE 'mag%'
```

id_cliente	id_transacao	data_compra	valor	id_loja
1	768805383	2021-06-10	50.74	magalu
1	76856563	2021-07-10	2000.90	magalu

Os wildcards podem ser diferentes de acordo com a ferramenta de banco de dados. Verifique a documentação específica.

▼ 3. Seleção condicional

Caso tenhamos diversas condições e para cada uma delas queremos retornar uma seleção diferente, podemos utilizar o comando **CASE** associado com **WHEN**, **THEN** e **ELSE** (opcional).

```
CASE
  WHEN <condicao1> THEN <comando1>
  WHEN <condicao2> THEN <comando2>
  ...
  WHEN <condicaoN> THEN <comandoN>
  ELSE <comando_caso_nenhuma_condicao_nao_seja_satisfeita>
END
```

Na tradução literal, podemos ler da seguinte forma:

- Caso a <condicao1> seja satisfeita, faça o <comando1> e saia
- Caso a <condicao1> não seja satisfeita, check a <condicao2>
- Caso a <condicao2> seja satisfeita, faça o <comando2> e saia
- Siga checando as condições até encontrar uma que seja satisfeita
- Caso nenhuma condição seja satisfeita, execute o
<comando_caso_nenhuma_condicao_nao_seja_satisfeita>

O comando pode ser utilizando dentro de vários comando como SELECT, UPDATE e DELETE e até isoladamente (com o SELECT, UPDATE E DELETE,... dentro do < comandoN >).

Vamos ao exemplo?

```
SELECT id_cliente, id_loja, valor,
CASE
  WHEN valor > 1000 THEN 'Compra com alto valor'
  WHEN valor < 1000 THEN 'Compra com baixo valor'
END
AS classeValor
FROM transacoes
WHERE valor > 50;
```

O resultado:

id_cliente	valor	id_loja	classeValor
1	50.74	magalu	Compra com baixo valor
3	110.00	postoshell	Compra com baixo valor

SQL - 4 - C			
id_cliente	valor	id_loja	classeValor
1	2000.90	magalu	Compra com alto valor
4	50.74	extra	Compra com baixo valor
3	1100.00	shopee	Compra com alto valor