escola britânica de artes criativas & tecnologia

# **Módulo** | SQL: Selecionando & Ordenando

Caderno de **Aula**
Professor Mariane Neiva

## Tópicos

1. Restrição de colunas;
2. Selecionando dados;
3. Ordenando e limitando os resultados;

## Aulas

Nessa aula, usaremos a seguinte tabela:

```
CREATE TABLE transacoes (
  id_cliente INT,
  id_transacao INT,
  data_compra DATE,
  valor FLOAT,
  id_loja varchar(25)
);
```

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 | 2021-08-92 | 10.00 | subway |
| 3 | 8154567758 | 2021-08-15 | 1100.00 | shopee |

Para realizar os testes no SQL, clique aqui

# 1. Restrição de colunas

## 1.1 Tipos de chaves

### 1.1.1 Chave primária (PRIMARY KEY)

Para identificar uma tabela, podemos utilizar uma chave primária. Existem algumas regras que devem ser respeitadas na coluna designada a ser chave primária:

1. O valor da coluna não pode se repetir na tabela
2. O valor da coluna não pode ser nulo

Você pode utilizar a chave primária de uma tabela para identificar uma instância em outra tabela.

```
CREATE TABLE <nome_tabela> (
  <nome_da_coluna_primaria> <tipo_da_coluna_primaria> PRIMARY KEY,
  <nome_da_coluna_2> <tipo_da_coluna_2>,
  ...
);
```

No exemplo, podemos colocar a chave primeira como o **id_transacao**, já que toda transação deve ser única (indice não se repete) e não nula (não pode existir um valor de transação que não foi preenchido):

```
CREATE TABLE transacoes (
  id_cliente INT,
  id_transacao INT PRIMARY KEY,
  data_compra DATE,
  valor FLOAT,
  id_loja varchar(25)
);
```

> Atenção: existe a opção de utilizar o AUTO_INCREMENT para que o valor de id_transacao seja automaticamente adicionado de 1 a cada nova transação no banco de dados. Entenda mais clicando aqui.

## 1.1.2. Chave estrangeira (FOREIGN KEY)

Quando queremos relacionar duas tabelas, utilizamos a chave estrangeira. Isso significa que uma coluna na tabela atual, está relaciona com alguma instância de outra tabela. No exemplo para o MySQL:

```
CREATE TABLE <nome_tabela> (
  <nome_da_coluna> <tipo_da_coluna>,
  <nome_da_coluna> <tipo_da_coluna>,
  ...,
  <nome_da_coluna> <tipo_da_coluna>,
```

```
  FOREIGN KEY <nome_da_coluna> REFERENCES <nome_tabela_da_chave_relacionada>(<nome_da_coluna_na_tabe
);
```

**Exemplo**: Na nossa tabela transações, faz sentido ter a chave primária como **id_transacoes** como fizemos anteriormente e deixar a chave estrangeira como o **id_cliente**. Isso porque, deve existir uma tabela clientes onde esse **id_cliente** é a chave primeira.

```
CREATE TABLE transacoes(
  id_cliente INT,
  id_transacao INT PRIMARY KEY,
  data_compra DATE,
  valor FLOAT,
  id_loja varchar(25),
  FOREIGN KEY (id_cliente) REFERENCES clientes(id_cliente)
);
```

**Atenção:** Não há alteração visual quando adicionamos primary key ou foreign key na tabela. Por isso não mostraremos o resultado aqui.

> Mais informações e outros formatos: clique aqui

## 1.2. Tipos de restrições

Algumas regras podem ser estabelecidas aos dados da sua tabela. Definindo-as na hora da criação da tabela, garantirá segurança na inserção dos dados. Veremos algumas nessa aula.

### 1.2.1 Valores não nulos (NOT NULL)

Define que os valores da coluna não podem ser nulos.

```
CREATE TABLE <nome_tabela> (
  <nome_da_coluna> <tipo_da_coluna> NOT NULL ,
  <nome_da_coluna_2> <tipo_da_coluna_2> ,
  ...
);
```

No nosso exemplo (unindo tudo que já aprendemos):

```
CREATE TABLE transacoes (
  id_cliente INT,
  id_transacao INT PRIMARY KEY,
  data_compra DATE,
  valor FLOAT NOT NULL,
  id_loja varchar(25),
  FOREIGN KEY (id_cliente) REFERENCES clientes(id_cliente)
);
```

## 1.2.2 Valores únicos (UNIQUE)

Define que os valores da coluna devem ser únicos.

```
CREATE TABLE <nome_tabela> (
  <nome_da_coluna> <tipo_da_coluna> UNIQUE ,
  <nome_da_coluna_2> <tipo_da_coluna_2> ,
  ...
);
```

No nosso exemplo (unindo tudo que já aprendemos):

```
CREATE TABLE transacoes (
  id_cliente INT,
  id_transacao INT PRIMARY KEY,
  data_compra DATE UNIQUE,
  valor FLOAT NOT NULL,
  id_loja varchar(25),
  FOREIGN KEY (id_cliente) REFERENCES clientes(id_cliente)
);
```

## 1.2.3 Checando restrições (CHECK)

Esse comando certifica que algumas condições devem ser satisfeitas ao inserir um dados na tabela.

```
CREATE TABLE <nome_tabela> (
  <nome_da_coluna> <tipo_da_coluna>  ,
  <nome_da_coluna_2> <tipo_da_coluna_2> ,
  CHECK (<condicao>)
  ...
);
```

No nosso caso, podemos avaliar se não há valor negativo nas transações:

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

**Atenção:** Não há alteração visual quando adicionamos primary key ou foreign key na tabela. Por isso não mostraremos o resultado aqui.

# 2. Selecionando dados

## 2.1. Comando SELECT

Para visualizar todos os dados da tabela, utilize asterisco (*):

```
SELECT * FROM <nome_tabela>;
```

Para visualizar colunas específicas da tabela:

```
SELECT <nome_coluna_1>,<nome_coluna_2> FROM <nome_tabela>;
```

Temos visto diversos exemplos durante os módulos 1 e 2. Mas hoje vamos além!

## 2.2. Comando AS

Você pode criar um apelido (*alias*) na hora se mostrar sua coluna com o comando **AS**:

```
SELECT <nome_da_coluna> AS <apelido_para_coluna> FROM <nome_da_tabela>;
```

ou

```
SELECT <nome_coluna1> AS <apelido_para_coluna1>, <nome_da_coluna2> AS <apelido_para_coluna2> FROM <n
```

No nosso exemplo:

```
SELECT id_cliente,id_transacao, data_compra, id_loja AS nome_loja FROM transacoes;
```

O resultado da query:

| id_cliente | id_transacao | data_compra | valor | nome_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 | 2021-07-11 | 50.74 | extra |
| 5 | 76766789 | 2021-08-02 | 10.00 | subway |
| 3 | 8154567758 | 2021-08-15 | 1100.00 | shopee |

## 2.3 Comando SELECT DISTINCT

Supondo que existam várias lojas (id_loja) diferentes no nosso banco de dados transações, com o uso do select distinct, o SQL só mostra quais são as lojas cadastradas na tabela, sem repetição.

```
SELECT DISTINCT <nome_da_coluna>  FROM <nome_da_tabela>;
```

No nosso caso:

```
SELECT DISTINCT id_loja AS nome_loja  FROM transacoes;
```

O resultado:

| nome_loja |
|---|
| magalu |
| giraffas |
| postoshell |
| subway |
| seveneleven |
| extra |
| shopee |

# 3. Ordenando e limitando resultados

## 3.1 Comando ORDER

Na hora de mostrar os resultados, é possível ordena-los de acordo com alguma varíavel(coluna) com a função **ORDER**.

```
SELECT <coluna1>,... <coluna2>, ...
FROM <nome_tabela>
ORDER BY <coluna1> ASC|DESC;
```

Clique duas vezes (ou pressione "Enter") para editar

```
SELECT id_cliente, valor
FROM transacoes
WHERE id_cliente= 1
ORDER BY valor DESC;
```

O resultado da query:

| id_cliente | valor |
|---|---|
| 1 | 2000.90 |
| 1 | 50.74 |
| 1 | 15.70 |

## 3.2. Comando LIMIT

Você pode limitar quantas linhas serão mostradas na sua seleção de dados com o comando LIMIT:

```
SELECT <nome_coluna>
FROM <nome_tabela>
LIMIT <numero_de_linhas>;
```

No nosso exemplo:

```
SELECT id_transacao, valor
FROM transacoes
LIMIT 3;
```

O resultado:

| id_transacao | valor |
|---|---|
| 768805383 | 50.74 |
| 768805399 | 30.90 |
| 818770008 | 110.00 |