



PROGRAMA DE CAPACITAÇÃO

**DATA SCIENCE + DATA INTELLIGENCE**

# NC 03 – Introdução ao uso de SQL para processos de extração e transformação de dados.

Prof. Dr.: José Carlos Conti



# Ementa



- SQL (Linguagem Estruturada de Consulta) tem sido, há muito tempo, bastante útil para extração e consulta a bancos de dados relacionais, para capturar matéria prima para análises, relatórios e indicadores.
- É fundamental que os profissionais dominem SQL para acelerar o processo de extração, tratamento e análise de dados.



# Conteúdos a serem trabalhados

1

- Conceituação e importância da linguagem SQL

3

- Uso da cláusula SELECT para consulta ao banco de dados

5

- Trabalhando com operações lógicas.

7

- Cruzamento de dados de diferentes tabelas (JOIN).

2

- Instalação do SQL Server Express.

4

- Uso de filtros (cláusula WHERE) nas consultas.

6

- Agrupando dados em uma consulta por meio de GROUP BY.

8

- Práticas usando SQL Server Express e Python

# Programação das aulas (15h – 5 sessões de 3h cada)



# Softwares a serem trabalhados

- Jupyter Notebooks (Python 3)
- SQL Server Express

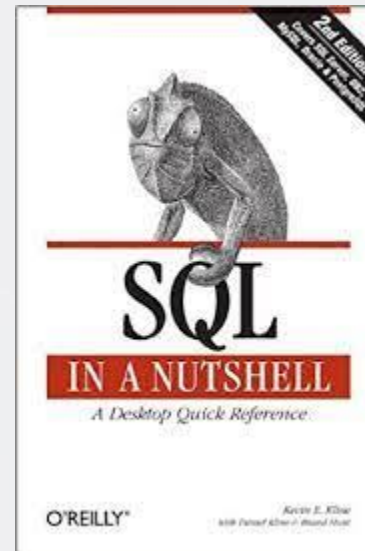
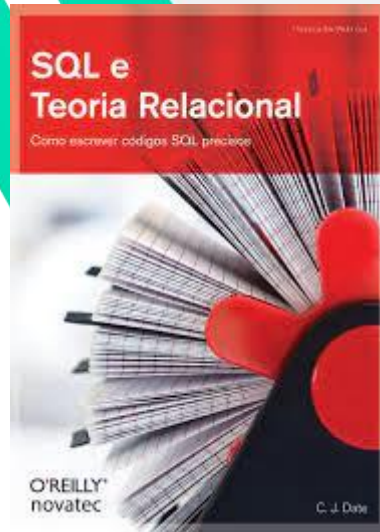
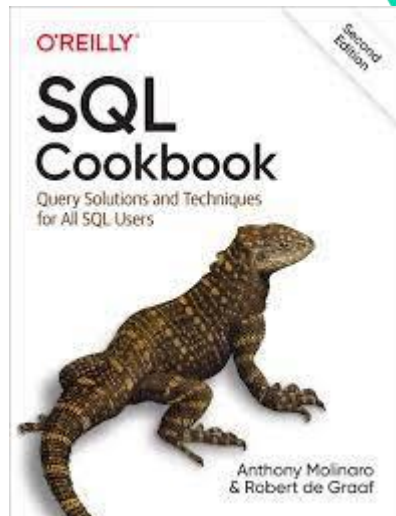




# Bibliografia básica



- SQL Cookbook – Anthony Molinaro, Robert de Graaf
- SQL e Teoria Relacional – C.J. Date
- SQL in a Nutshell – Kevin Kline, Daniel Kline, Brand Hunt
- Getting Started with SQL – Thomas Nield



- Revisão: Atividades da aula anterior

1. Consulte os dados da tabela t\_vendas, selecione todos os campos.
2. Consulte os dados da tabela t\_vendas, selecione os campos: codigo\_cliente, loja, tamanho\_pedido.
3. Consulte os dados da tabela t\_vendas, selecione os campos: codigo\_cliente, loja, tamanho\_pedido, acrescente na condição WHERE loja = 'Rio de Janeiro'.
4. Consulte os dados da tabela t\_vendas, selecione todos os campos , acrescente na condição WHERE loja in ('Rio de Janeiro', 'Salvador')
5. Consulte os dados da tabela t\_vendas, selecione todos os campos, acrescente na condição WHERE tamanho\_pedido > 5 e codigo\_cliente entre 100 e 200.
6. Consulte os dados da tabela t\_vendas, crie o campo UF consultando o campo LOJA em que: LOJA = Rio de Janeiro a UF será RJ; LOJA = São Paulo, a UF será SP; LOJA = Salvador a UF será BA; LOJA = Belo Horizonte a UF será MG.
7. Consulte os dados da tabela t\_clientes, selecione todos os campos.
8. Consulte os dados da tabela t\_clientes, selecione todos os cliente em que a UF = SP.
9. Consulte os dados da tabela t\_clientes, selecione todos os clientes em que o código do cliente esteja entre 10 e 20.
10. Consulte os dados da tabela t\_clientes, selecione todos os clientes em que a UF = RJ e o código do cliente seja maior que 50.



- Atividades:

11. Consulte os dados da tabela t\_clientes, selecione todos os campos e crie um campo novo gr\_cliente, considere código\_cliente quando for  $\leq 100$  o gr\_cliente será 'Até 100', quando o código\_cliente estiver entre 101 e 200 o gr\_cliente será 'De 101 a 200', quando o código\_cliente estiver entre 201 e 300 o gr\_cliente será 'De 201 a 300', quando o código\_cliente estiver entre 301 a 400, o gr\_cliente será 'De 301 a 400', quando o código\_cliente estiver entre 401 a 500 o gr\_cliente será 'De 401 a 500', quando o código\_cliente for  $> 500$ , o gr\_cliente será ' $> 500$ '.

12. Consulte os dados da tabela t\_clientes, selecione todos os campos e crie um campo novo flag\_camp, considere o campo UF, se for igual SP ou RJ será igual a 1, se UF for diferente de SP ou RJ, será igual a 0.

13. Consulte os dados da tabela t\_clientes, selecione todos os campos e crie um campo novo flag\_seleção, considere o campo código do cliente, se estiver entre 200 a 400, o valor será igual a 1, caso contrário, o valor será 0.

# Aula 3

## Trabalhando com operações lógicas



-  
-  
-

x



Go to [www.menti.com](https://www.menti.com) and use the code 9938 1826

**P04 - Qual sintaxe representa a seleção de dados sobre vendas com o volume de pedidos igual a 5 e a loja de Porto Alegre ?**

Mentimeter

0

```
select * from
cliente.dbo.t_vendas
WHERE
TAMANHO_PEDIDO >= 5
and LOJA = 'Porto*';
```

0

```
select LOJA, MARCA,
TAMANHO_PEDIDO
from
cliente.dbo.t_vendas
where
TAMANHO_PEDIDO = 5
and LOJA = 'Porto
Alegre';
```

0

```
select * from
cliente.dbo.t_vendas
WHERE
TAMANHO_PEDIDO = 5
and LOJA = 'Porto*';
```

# Trabalhando com operações lógicas

## Cláusula Where e seus Operadores lógicos no SQL

Os operadores lógicos testam a legitimidade das condições em consultas SQL

BETWEEN

NOT BETWEEN

=

AND

NOT IN

<>

OR

IN

NOT LIKE

LIKE

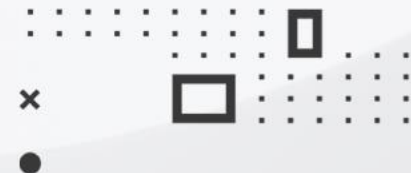
>

>=

<

<=

# Trabalhando com operações lógicas



## Atividades:

- Usando o operador AND
- O exemplo a seguir seleciona informações sobre vendas que possuem o volume de pedidos igual a 5 e a loja de Porto Alegre.

```
select LOJA, MARCA, TAMANHO_PEDIDO from cliente.dbo.t_vendas
where TAMANHO_PEDIDO = 5 and LOJA = 'Porto Alegre';
```

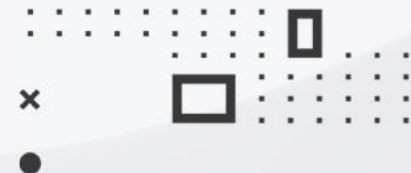
133 %

Resultados Mensagens

	LOJA	MARCA	TAMANHO_PEDIDO
1	Porto Alegre	Xiaomi	5
2	Porto Alegre	LG	5
3	Porto Alegre	Samsung	5
4	Porto Alegre	Samsung	5
5	Porto Alegre	Motorola	5
6	Porto Alegre	Acer	5
7	Porto Alegre	Sony	5
8	Porto Alegre	Sony	5
9	Porto Alegre	LG	5
10	Porto Alegre	Xiaomi	5
11	Porto Alegre	Motorola	5



# Trabalhando com operações lógicas



## Atividades:

- Usando o operador OR
- O exemplo a seguir seleciona informações sobre vendas que possuem o volume de pedidos igual a 5 e a loja de Porto Alegre.

```
select LOJA, MARCA, TAMANHO_PEDIDO from cliente.dbo.t_vendas
where TAMANHO_PEDIDO = 5 and LOJA = 'Porto Alegre';
```

133 %

Resultados Mensagens

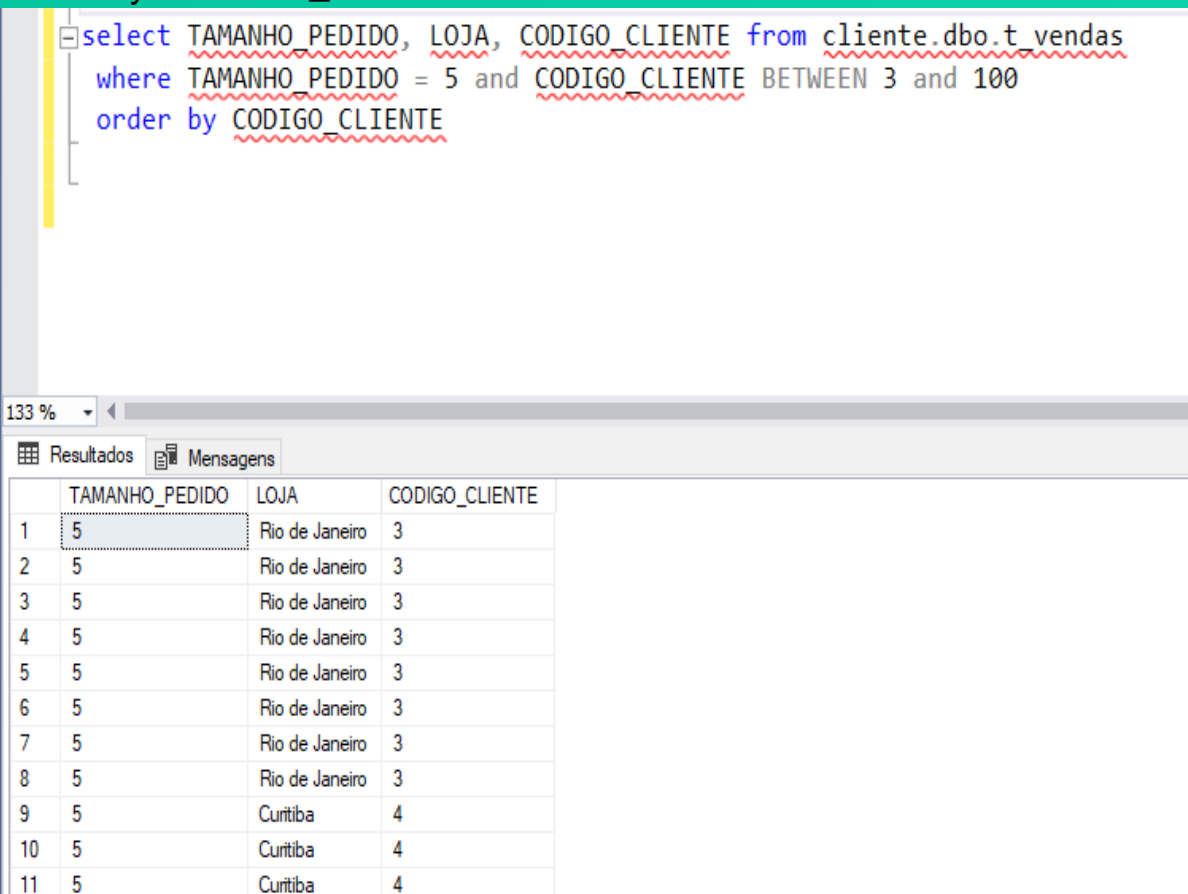
	LOJA	MARCA	TAMANHO_PEDIDO
1	Porto Alegre	Xiaomi	5
2	Porto Alegre	LG	5
3	Porto Alegre	Samsung	5
4	Porto Alegre	Samsung	5
5	Porto Alegre	Motorola	5
6	Porto Alegre	Acer	5
7	Porto Alegre	Sony	5
8	Porto Alegre	Sony	5
9	Porto Alegre	LG	5
10	Porto Alegre	Xiaomi	5
11	Porto Alegre	Motorola	5

# Trabalhando com operações lógicas

## Usando BETWEEN

- A próxima consulta usa a cláusula BETWEEN para limitar o tamanho do pedido para um grupo específico de código do cliente e ordena pelo código do cliente.

```
select TAMANHO_PEDIDO, LOJA, CODIGO_CLIENTE from cliente.dbo.t_vendas  
where TAMANHO_PEDIDO = 5 and CODIGO_CLIENTE BETWEEN 3 and 100  
order by CODIGO_CLIENTE
```

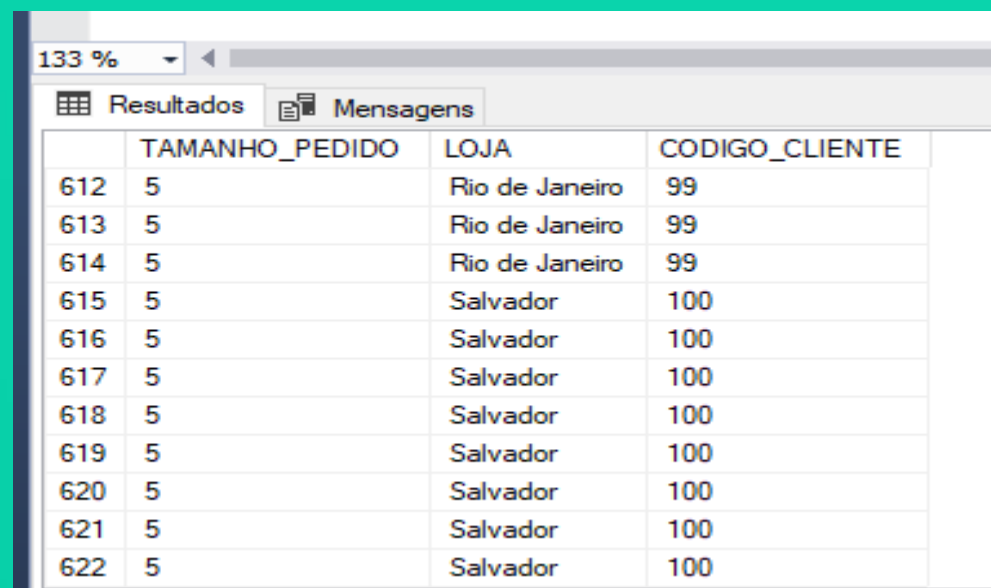


The screenshot shows a SQL query window with the following text:

```
select TAMANHO_PEDIDO, LOJA, CODIGO_CLIENTE from cliente.dbo.t_vendas  
where TAMANHO_PEDIDO = 5 and CODIGO_CLIENTE BETWEEN 3 and 100  
order by CODIGO_CLIENTE
```

Below the query window, the 'Resultados' (Results) tab is active, displaying a table with 11 rows and 4 columns: TAMANHO\_PEDIDO, LOJA, and CODIGO\_CLIENTE. The first row is highlighted.

	TAMANHO_PEDIDO	LOJA	CODIGO_CLIENTE
1	5	Rio de Janeiro	3
2	5	Rio de Janeiro	3
3	5	Rio de Janeiro	3
4	5	Rio de Janeiro	3
5	5	Rio de Janeiro	3
6	5	Rio de Janeiro	3
7	5	Rio de Janeiro	3
8	5	Rio de Janeiro	3
9	5	Curitiba	4
10	5	Curitiba	4
11	5	Curitiba	4



The screenshot shows a SQL query window with the following text:

```
select TAMANHO_PEDIDO, LOJA, CODIGO_CLIENTE from cliente.dbo.t_vendas  
where TAMANHO_PEDIDO = 5 and CODIGO_CLIENTE BETWEEN 3 and 100  
order by CODIGO_CLIENTE
```

Below the query window, the 'Resultados' (Results) tab is active, displaying a table with 11 rows and 4 columns: TAMANHO\_PEDIDO, LOJA, and CODIGO\_CLIENTE. The first row is highlighted.

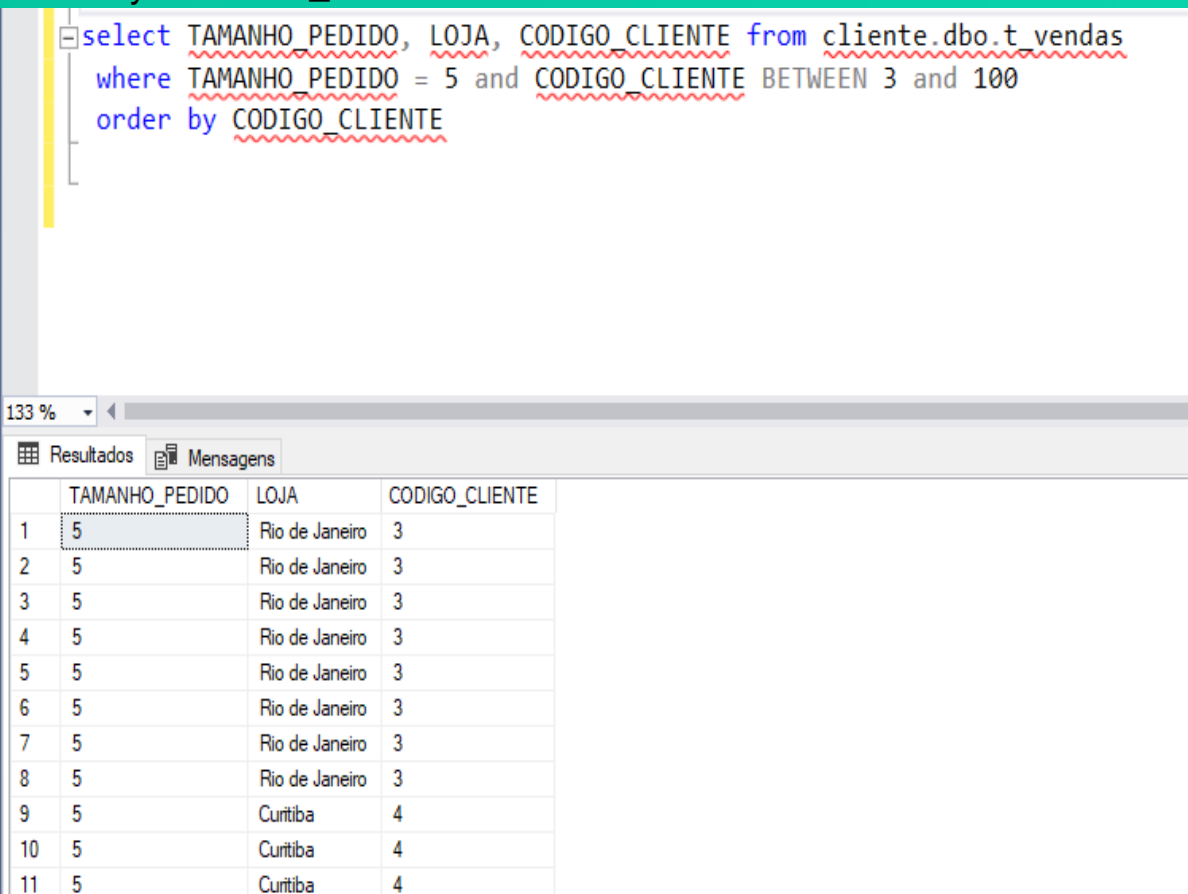
	TAMANHO_PEDIDO	LOJA	CODIGO_CLIENTE
612	5	Rio de Janeiro	99
613	5	Rio de Janeiro	99
614	5	Rio de Janeiro	99
615	5	Salvador	100
616	5	Salvador	100
617	5	Salvador	100
618	5	Salvador	100
619	5	Salvador	100
620	5	Salvador	100
621	5	Salvador	100
622	5	Salvador	100

# Trabalhando com operações lógicas

## Usando NOT BETWEEN

- A próxima consulta usa a cláusula BETWEEN para limitar o tamanho do pedido para um grupo específico de código do cliente e ordena pelo código do cliente.

```
select TAMANHO_PEDIDO, LOJA, CODIGO_CLIENTE from cliente.dbo.t_vendas  
where TAMANHO_PEDIDO = 5 and CODIGO_CLIENTE BETWEEN 3 and 100  
order by CODIGO_CLIENTE
```

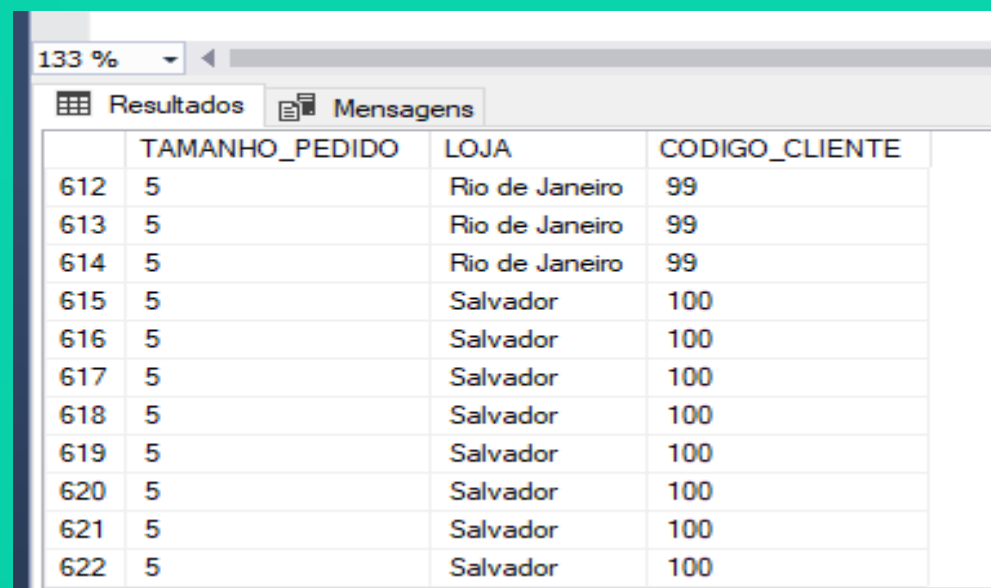


The screenshot shows a SQL query window with the following text:

```
select TAMANHO_PEDIDO, LOJA, CODIGO_CLIENTE from cliente.dbo.t_vendas  
where TAMANHO_PEDIDO = 5 and CODIGO_CLIENTE BETWEEN 3 and 100  
order by CODIGO_CLIENTE
```

Below the query window, the 'Resultados' (Results) tab is active, displaying a table with 11 rows and 4 columns: TAMANHO\_PEDIDO, LOJA, and CODIGO\_CLIENTE. The first row is highlighted.

	TAMANHO_PEDIDO	LOJA	CODIGO_CLIENTE
1	5	Rio de Janeiro	3
2	5	Rio de Janeiro	3
3	5	Rio de Janeiro	3
4	5	Rio de Janeiro	3
5	5	Rio de Janeiro	3
6	5	Rio de Janeiro	3
7	5	Rio de Janeiro	3
8	5	Rio de Janeiro	3
9	5	Curitiba	4
10	5	Curitiba	4
11	5	Curitiba	4



The screenshot shows a SQL query window with the following text:

```
select TAMANHO_PEDIDO, LOJA, CODIGO_CLIENTE from cliente.dbo.t_vendas  
where TAMANHO_PEDIDO = 5 and CODIGO_CLIENTE BETWEEN 3 and 100  
order by CODIGO_CLIENTE
```

Below the query window, the 'Resultados' (Results) tab is active, displaying a table with 11 rows and 4 columns: TAMANHO\_PEDIDO, LOJA, and CODIGO\_CLIENTE. The first row is highlighted.

	TAMANHO_PEDIDO	LOJA	CODIGO_CLIENTE
612	5	Rio de Janeiro	99
613	5	Rio de Janeiro	99
614	5	Rio de Janeiro	99
615	5	Salvador	100
616	5	Salvador	100
617	5	Salvador	100
618	5	Salvador	100
619	5	Salvador	100
620	5	Salvador	100
621	5	Salvador	100
622	5	Salvador	100

# Trabalhando com operações lógicas

- Comparando OR e IN usando o Join entre duas tabelas

01 O exemplo a seguir seleciona uma lista de nomes de clientes que compraram nas lojas de Rio de Janeiro, Salvador ou Porto Alegre.

02 Entretanto, você recupera os mesmos resultados usando IN.

A seguir apresenta-se parte do conjunto de resultados das duas consultas...

```
SELECT b.codigo_cliente, b.nome_cliente, a.loja
FROM cliente.dbo.t_clientes as b
JOIN cliente.dbo.t_vendas as a
ON b.codigo_cliente = a.codigo_cliente
WHERE a.loja = 'Rio de Janeiro'
OR a.loja = 'Salvador'
OR a.loja = 'Porto Alegre'
order by b.codigo_cliente
```

	codigo_cliente	nome_cliente	loja
1	3	Jose A3	Rio de Janeiro
2	3	Jose A3	Rio de Janeiro
3	3	Jose A3	Rio de Janeiro
4	3	Jose A3	Rio de Janeiro
5	3	Jose A3	Rio de Janeiro
6	3	Jose A3	Rio de Janeiro
7	3	Jose A3	Rio de Janeiro
8	3	Jose A3	Rio de Janeiro
9	3	Jose A3	Rio de Janeiro
10	3	Jose A3	Rio de Janeiro
11	3	Jose A3	Rio de Janeiro

01 → 

```
SELECT b.codigo_cliente, b.nome_cliente, a.loja
FROM cliente.dbo.t_clientes as b
JOIN cliente.dbo.t_vendas as a
ON b.codigo_cliente = a.codigo_cliente
WHERE a.loja = 'Rio de Janeiro'
OR a.loja = 'Salvador'
OR a.loja = 'Porto Alegre'
order by b.codigo_cliente
```

02 → 

```
SELECT b.codigo_cliente, b.nome_cliente, a.loja
FROM cliente.dbo.t_clientes as b
JOIN cliente.dbo.t_vendas as a
ON b.codigo_cliente = a.codigo_cliente
WHERE a.loja in ('Rio de Janeiro','Salvador','Porto Alegre')
order by b.codigo_cliente
```

	codigo_cliente	nome_cliente	loja
1	3	Jose A3	Rio de Janeiro
2	3	Jose A3	Rio de Janeiro
3	3	Jose A3	Rio de Janeiro
4	3	Jose A3	Rio de Janeiro
5	3	Jose A3	Rio de Janeiro
6	3	Jose A3	Rio de Janeiro
7	3	Jose A3	Rio de Janeiro
8	3	Jose A3	Rio de Janeiro
9	3	Jose A3	Rio de Janeiro
10	3	Jose A3	Rio de Janeiro
11	3	Jose A3	Rio de Janeiro

# Trabalhando com operações lógicas

- Usando IN

- O exemplo a seguir encontra todos os códigos de clientes para a tabela de clientes e seleciona na tabela vendas os clientes em que o código do cliente corresponde aos resultados da sub consulta SELECT onde o tamanho do pedido da tabela de vendas seja maior ou igual a 5.

```
SELECT b.codigo_cliente, b.nome_cliente, a.tamanho_pedido
FROM cliente.dbo.t_clientes as b
JOIN cliente.dbo.t_vendas as a
ON b.codigo_cliente = a.codigo_cliente
WHERE b.codigo_cliente IN
(SELECT a.codigo_cliente
FROM cliente.dbo.t_vendas
WHERE a.tamanho_pedido >= 5)
```

A seguir apresenta-se parte do conjunto de resultados...

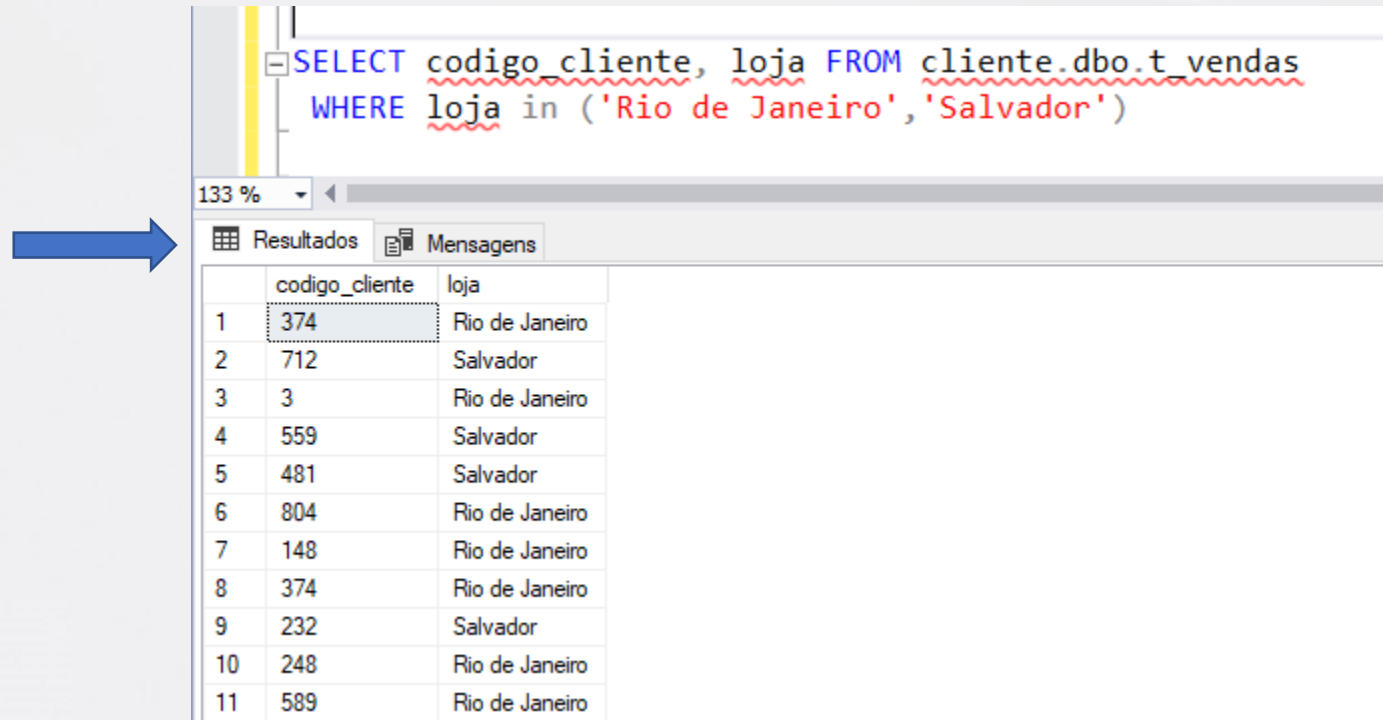
	codigo_cliente	nome_cliente	tamanho_pedido
1	64	Jose A64	5
2	712	Pedro H712	5
3	163	Joao B163	5
4	57	Jose A57	5
5	481	Antonio E481	5
6	148	Joao B148	5
7	235	Joaquim C235	5
8	232	Joaquim C232	5
9	459	Antonio E459	5
10	773	Pedro H773	5
11	213	Joaquim C213	5



# Trabalhando com operações lógicas

- Usando IN com uma lista de expressões

- O exemplo a seguir localiza todos os códigos os clientes na tabela de vendas que compraram nas lojas do Rio de Janeiro ou Salvador.



```
SELECT codigo_cliente, loja FROM cliente.dbo.t_vendas
WHERE loja in ('Rio de Janeiro', 'Salvador')
```

133 %

Resultados Mensagens

	codigo_cliente	loja
1	374	Rio de Janeiro
2	712	Salvador
3	3	Rio de Janeiro
4	559	Salvador
5	481	Salvador
6	804	Rio de Janeiro
7	148	Rio de Janeiro
8	374	Rio de Janeiro
9	232	Salvador
10	248	Rio de Janeiro
11	589	Rio de Janeiro

# Trabalhando com operações lógicas

- Usando IN com uma lista de expressões

- O exemplo a seguir consulta todas as marcas dos produtos vendidos na tabela de vendas que possuam os valores: Samsung, Motorola, Philco e Semp.

```
select * from cliente.dbo.t_vendas  
where marca in ('Samsung', 'Motorola', 'Philco', 'Semp')
```

.77 %

Resultados Mensagens

	SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
1	HL1008	1	Rio de Janeiro	2016-01-01	Philco	374	2000	1250
2	HL1021	1	Recife	2016-01-01	Samsung	165	3000	1400
3	HL1008	5	Salvador	2016-01-01	Philco	712	2000	1250
4	HL1020	4	Recife	2016-01-01	Motorola	144	1500	650
5	HL1021	3	Rio de Janeiro	2016-01-01	Samsung	3	3000	1400
6	HL1021	5	Fortaleza	2016-01-01	Samsung	163	3000	1400
7	HL1008	5	São Paulo	2016-01-01	Philco	57	2000	1250
8	HL1009	4	Rio de Janeiro	2016-01-01	Motorola	804	1400	750
9	HL1014	3	Curitiba	2016-01-01	Samsung	774	1100	550
10	HL1003	4	Curitiba	2016-01-01	Samsung	560	4500	2800
11	HL1008	4	Guarulhos	2016-01-01	Philco	630	2000	1250

# Trabalhando com operações lógicas

- Usando NOT IN com uma lista de expressões

- O exemplo a seguir localiza todos os códigos os clientes na tabela de vendas que não compraram nas lojas do Rio de Janeiro ou Salvador.

```
select * from cliente.dbo.t_vendas  
where loja not in ('Rio de Janeiro', 'Salvador')
```

	SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
1	HL1018	5	Porto Alegre	2016-01-01	Xiaomi	64	1200	650
2	HL1001	2	Fortaleza	2016-01-01	LG	796	2600	1700
3	HL1021	1	Recife	2016-01-01	Samsung	165	3000	1400
4	HL1020	4	Recife	2016-01-01	Motorola	144	1500	650
5	HL1019	2	São Paulo	2016-01-01	Apple	698	6500	2800
6	HL1005	4	Nova Iguaçu	2016-01-01	Canon	569	1500	850
7	HL1007	4	Campinas	2016-01-01	Dell	510	2300	1200
8	HL1021	5	Fortaleza	2016-01-01	Samsung	163	3000	1400
9	HL1008	5	São Paulo	2016-01-01	Philco	57	2000	1250
10	HL1002	4	Fortaleza	2016-01-01	Apple	157	2500	1500
11	HL1007	3	São Paulo	2016-01-01	Dell	235	2300	1200

# Trabalhando com operações lógicas

- Usando NOT IN

- O exemplo a seguir localiza os clientes que não têm um tamanho de pedido maior ou igual a 5. NOT IN localiza os clientes que não correspondem aos itens da lista de valores.

```
SELECT b.codigo_cliente, b.nome_cliente, a.tamanho_pedido
FROM cliente.dbo.t_clientes as b
JOIN cliente.dbo.t_vendas as a
ON b.codigo_cliente = a.codigo_cliente
WHERE b.codigo_cliente NOT IN
(SELECT a.codigo_cliente
FROM cliente.dbo.t_vendas
WHERE a.tamanho_pedido >= 5)
```

133 %

Resultados Mensagens

	codigo_cliente	nome_cliente	tamanho_pedido
1	796	Pedro H796	2
2	374	Maria D374	1
3	165	Joao B165	1
4	144	Joao B144	4
5	698	Gabriel G698	2
6	3	Jose A3	3
7	559	Miguel F559	2
8	569	Miguel F569	4
9	510	Miguel F510	4
10	804	Isa I804	4
11	157	Joao B157	4

# Trabalhando com operações lógicas

- Usando Between

```
select * from cliente.dbo.t_vendas  
where tamanho_pedido between 5 and 50
```

177 %

Resultados Mensagens

	SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
1	HL1018	5	Porto Alegre	2016-01-01	Xiaomi	64	1200	650
2	HL1008	5	Salvador	2016-01-01	Philco	712	2000	1250
3	HL1021	5	Fortaleza	2016-01-01	Samsung	163	3000	1400
4	HL1008	5	São Paulo	2016-01-01	Philco	57	2000	1250
5	HL1024	5	Salvador	2016-01-01	Acer	481	2400	1150
6	HL1005	5	Rio de Janeiro	2016-01-01	Canon	148	1500	850
7	HL1001	5	São Paulo	2016-01-01	LG	235	2600	1700
8	HL1023	5	Salvador	2016-01-01	Samsung	232	1500	550
9	HL1005	5	Recife	2016-01-02	Canon	459	1500	850
10	HL1005	5	São Paulo	2016-01-02	Canon	773	1500	850
11	HL1023	5	Rio de Janeiro	2016-01-02	Samsung	213	1500	550



# Trabalhando com operações lógicas

- Usando Not Between

```
--select * from cliente.dbo.t_vendas  
where tamanho_pedido not between 5 and 50
```

177 %

Resultados Mensagens

	SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
1	HL1001	2	Fortaleza	2016-01-01	LG	796	2600	1700
2	HL1008	1	Rio de Janeiro	2016-01-01	Philco	374	2000	1250
3	HL1021	1	Recife	2016-01-01	Samsung	165	3000	1400
4	HL1020	4	Recife	2016-01-01	Motorola	144	1500	650
5	HL1019	2	São Paulo	2016-01-01	Apple	698	6500	2800
6	HL1021	3	Rio de Janeiro	2016-01-01	Samsung	3	3000	1400
7	HL1013	2	Salvador	2016-01-01	Nikon	559	1550	700
8	HL1005	4	Nova Iguaçu	2016-01-01	Canon	569	1500	850
9	HL1007	4	Campinas	2016-01-01	Dell	510	2300	1200
10	HL1009	4	Rio de Janeiro	2016-01-01	Motorola	804	1400	750
11	HL1002	4	Fortaleza	2016-01-01	Apple	157	2500	1500

# Trabalhando com operações lógicas

- Usando = , > = , >

```
select * from cliente.dbo.t_vendas  
where tamanho_pedido = 5
```

77 %

Resultados Mensagens

	SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
1	HL1018	5	Porto Alegre	2016-01-01	Xiaomi	64	1200	650
2	HL1008	5	Salvador	2016-01-01	Philco	712	2000	1250
3	HL1021	5	Fortaleza	2016-01-01	Samsung	163	3000	1400
4	HL1008	5	São Paulo	2016-01-01	Philco	57	2000	1250
5	HL1024	5	Salvador	2016-01-01	Acer	481	2400	1150
6	HL1005	5	Rio de Janeiro	2016-01-01	Canon	148	1500	850
7	HL1001	5	São Paulo	2016-01-01	LG	235	2600	1700
8	HL1023	5	Salvador	2016-01-01	Samsung	232	1500	550
9	HL1005	5	Recife	2016-01-02	Canon	459	1500	850
10	HL1005	5	São Paulo	2016-01-02	Canon	773	1500	850
11	HL1023	5	Rio de Janeiro	2016-01-02	Samsung	213	1500	550

# Trabalhando com operações lógicas

- Usando = , > = , >

```
select * from cliente.dbo.t_vendas  
where tamanho_pedido >= 5
```

SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
HL1018	5	Porto Alegre	2016-01-01	Xiaomi	64	1200	650
HL1008	5	Salvador	2016-01-01	Philco	712	2000	1250
HL1021	5	Fortaleza	2016-01-01	Samsung	163	3000	1400
HL1008	5	São Paulo	2016-01-01	Philco	57	2000	1250
HL1024	5	Salvador	2016-01-01	Acer	481	2400	1150
HL1005	5	Rio de Janeiro	2016-01-01	Canon	148	1500	850
HL1001	5	São Paulo	2016-01-01	LG	235	2600	1700
HL1023	5	Salvador	2016-01-01	Samsung	232	1500	550
HL1005	5	Recife	2016-01-02	Canon	459	1500	850
HL1005	5	São Paulo	2016-01-02	Canon	773	1500	850
HL1023	5	Rio de Janeiro	2016-01-02	Samsung	213	1500	550

# Trabalhando com operações lógicas

- Usando = , > = , >

```
select * from cliente.dbo.t_vendas  
where tamanho_pedido > 5
```

Resultados Mensagens

SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
-----	----------------	------	------------	-------	----------------	----------------	----------------

```
select tamanho_pedido, count(*) as qtd from  
cliente.dbo.t_vendas  
group by tamanho_pedido  
order by tamanho_pedido
```

177 %

Resultados Mensagens

	tamanho_pedido	qtd
1	1	4710
2	2	4785
3	3	4828
4	4	4677
5	5	4794

# Trabalhando com operações lógicas

- Usando = , <= , <

```
select * from cliente.dbo.t_clientes  
where codigo_cliente = 100
```

.77 %

Resultados Mensagens

	CODIGO_CLIENTE	NOME_CLIENTE	DATA_NASCIMENTO	EMAIL	TELEFONE_CELULAR	UF
1	100	Jose A100	1951-01-07	jose a100@email.com	11911111210	SP



# Trabalhando com operações lógicas

- Usando = , <= , <

```
select * from cliente.dbo.t_clientes  
where codigo_cliente <= 100
```

177 %

Resultados Mensagens

	CODIGO_CLIENTE	NOME_CLIENTE	DATA_NASCIMENTO	EMAIL	TELEFONE_CELULAR	UF
1	1	Jose A1	1950-09-30	jose a1@email.com	1191111111	SP
2	2	Jose A2	1950-10-01	jose a2@email.com	1191111112	SP
3	3	Jose A3	1950-10-02	jose a3@email.com	1191111113	SP
4	4	Jose A4	1950-10-03	jose a4@email.com	1191111114	SP
5	5	Jose A5	1950-10-04	jose a5@email.com	1191111115	SP
6	6	Jose A6	1950-10-05	jose a6@email.com	1191111116	SP
7	7	Jose A7	1950-10-06	jose a7@email.com	1191111117	SP
8	8	Jose A8	1950-10-07	jose a8@email.com	1191111118	SP
9	9	Jose A9	1950-10-08	jose a9@email.com	1191111119	SP
10	10	Jose A10	1950-10-09	jose a10@email.com	1191111120	SP
11	11	Jose A11	1950-10-10	jose a11@email.com	1191111121	SP

```
where codigo_cliente <= 100
```

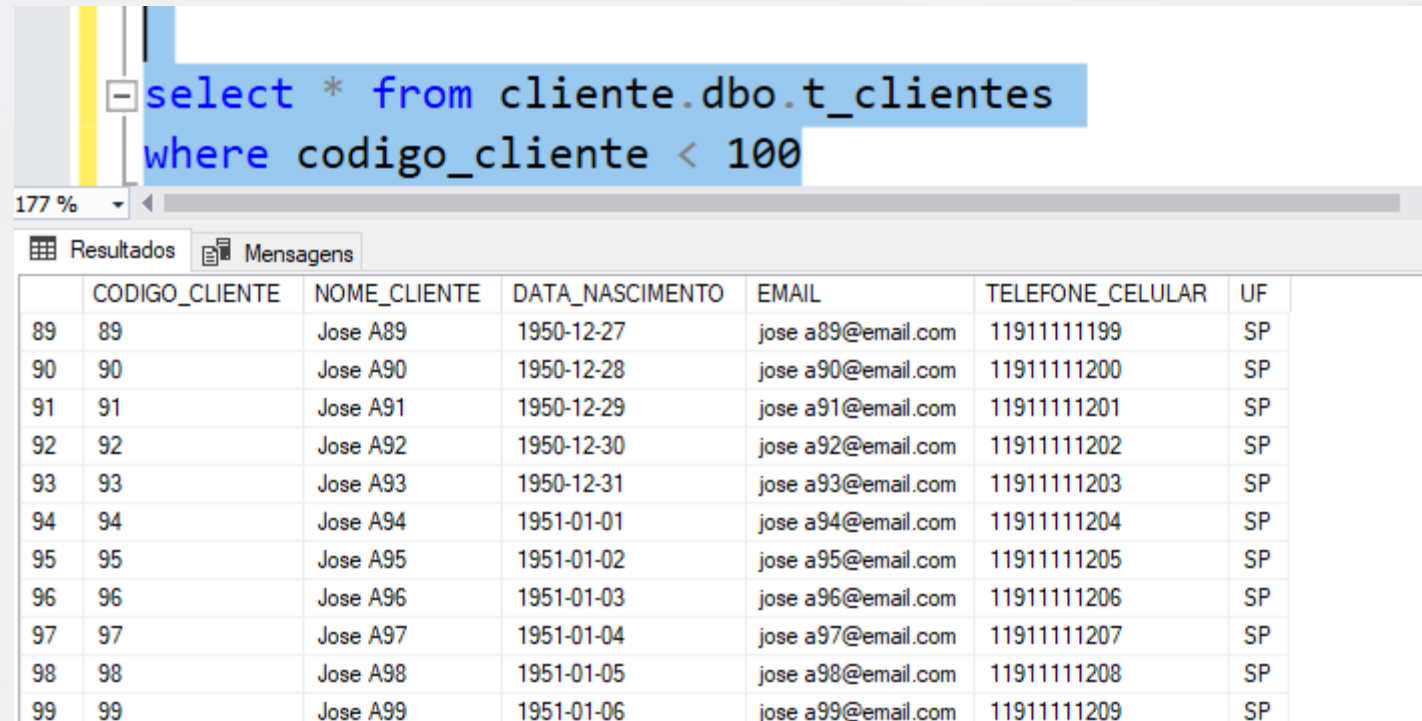
177 %

Resultados Mensagens

	CODIGO_CLIENTE	NOME_CLIENTE	DATA_NASCIMENTO	EMAIL	TELEFONE_CELULAR	UF
90	90	Jose A90	1950-12-28	jose a90@email.com	11911111200	SP
91	91	Jose A91	1950-12-29	jose a91@email.com	11911111201	SP
92	92	Jose A92	1950-12-30	jose a92@email.com	11911111202	SP
93	93	Jose A93	1950-12-31	jose a93@email.com	11911111203	SP
94	94	Jose A94	1951-01-01	jose a94@email.com	11911111204	SP
95	95	Jose A95	1951-01-02	jose a95@email.com	11911111205	SP
96	96	Jose A96	1951-01-03	jose a96@email.com	11911111206	SP
97	97	Jose A97	1951-01-04	jose a97@email.com	11911111207	SP
98	98	Jose A98	1951-01-05	jose a98@email.com	11911111208	SP
99	99	Jose A99	1951-01-06	jose a99@email.com	11911111209	SP
100	100	Jose A100	1951-01-07	jose a100@email.c...	11911111210	SP

# Trabalhando com operações lógicas

- Usando = , <= , <



The screenshot shows a SQL query editor with the following text:

```
select * from cliente.dbo.t_clientes  
where codigo_cliente < 100
```

Below the query, the results are displayed in a table. The table has 7 columns: CODIGO\_CLIENTE, NOME\_CLIENTE, DATA\_NASCIMENTO, EMAIL, TELEFONE\_CELULAR, and UF. The results show 11 rows of data, all with CODIGO\_CLIENTE values from 89 to 99.

	CODIGO_CLIENTE	NOME_CLIENTE	DATA_NASCIMENTO	EMAIL	TELEFONE_CELULAR	UF
89	89	Jose A89	1950-12-27	jose a89@email.com	11911111199	SP
90	90	Jose A90	1950-12-28	jose a90@email.com	11911111200	SP
91	91	Jose A91	1950-12-29	jose a91@email.com	11911111201	SP
92	92	Jose A92	1950-12-30	jose a92@email.com	11911111202	SP
93	93	Jose A93	1950-12-31	jose a93@email.com	11911111203	SP
94	94	Jose A94	1951-01-01	jose a94@email.com	11911111204	SP
95	95	Jose A95	1951-01-02	jose a95@email.com	11911111205	SP
96	96	Jose A96	1951-01-03	jose a96@email.com	11911111206	SP
97	97	Jose A97	1951-01-04	jose a97@email.com	11911111207	SP
98	98	Jose A98	1951-01-05	jose a98@email.com	11911111208	SP
99	99	Jose A99	1951-01-06	jose a99@email.com	11911111209	SP

# Trabalhando com operações lógicas

- Usando LIKE

```
select * from cliente.dbo.t_vendas  
where loja like ('%Rio%')
```

77 %

Resultados Mensagens

	SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
1	HL1008	1	Rio de Janeiro	2016-01-01	Philco	374	2000	1250
2	HL1021	3	Rio de Janeiro	2016-01-01	Samsung	3	3000	1400
3	HL1009	4	Rio de Janeiro	2016-01-01	Motorola	804	1400	750
4	HL1005	5	Rio de Janeiro	2016-01-01	Canon	148	1500	850
5	HL1010	2	Rio de Janeiro	2016-01-01	Apple	374	1900	1150
6	HL1003	4	Rio de Janeiro	2016-01-01	Samsung	248	4500	2800
7	HL1008	1	Rio de Janeiro	2016-01-02	Philco	589	2000	1250
8	HL1023	5	Rio de Janeiro	2016-01-02	Samsung	213	1500	550
9	HL1001	1	Rio de Janeiro	2016-01-03	LG	125	2600	1700
10	HL1017	5	Rio de Janeiro	2016-01-03	Samsung	374	5200	3500
11	HL1014	4	Rio de Janeiro	2016-01-03	Samsung	572	1100	550

# Trabalhando com operações lógicas

- Usando NOT LIKE

```
select * from cliente.dbo.t_vendas  
where loja not like ('%Rio%')
```

77 %

Resultados Mensagens

	SKU	TAMANHO_PEDIDO	LOJA	DATA_VENDA	MARCA	CODIGO_CLIENTE	PRECO_UNITARIO	CUSTO_UNITARIO
1	HL1018	5	Porto Alegre	2016-01-01	Xiaomi	64	1200	650
2	HL1001	2	Fortaleza	2016-01-01	LG	796	2600	1700
3	HL1021	1	Recife	2016-01-01	Samsung	165	3000	1400
4	HL1008	5	Salvador	2016-01-01	Philco	712	2000	1250
5	HL1020	4	Recife	2016-01-01	Motorola	144	1500	650
6	HL1019	2	São Paulo	2016-01-01	Apple	698	6500	2800
7	HL1013	2	Salvador	2016-01-01	Nikon	559	1550	700
8	HL1005	4	Nova Iguaçu	2016-01-01	Canon	569	1500	850
9	HL1007	4	Campinas	2016-01-01	Dell	510	2300	1200
10	HL1021	5	Fortaleza	2016-01-01	Samsung	163	3000	1400
11	HL1008	5	São Paulo	2016-01-01	Philco	57	2000	1250

# Trabalhando com operações lógicas

- **Atividades – Execute as seguintes consultas:**

1. **Selecione os campos da tabela t\_vendas e na cláusula Where combine os operadores = e between.**
2. **Selecione os campos da tabela t\_vendas e na cláusula Where combine os operadores > e > =**
3. **Consulte a tabela t\_vendas e combine os operadores < e <=**
4. **Consulte a tabela t\_vendas e combine os operadores >= e <=**
5. **Consulte a tabela t\_vendas e combine os operadores not between e >=**
6. **Consulte a tabela t\_clientes e mostre um exemplo com o operador AND**
7. **Consulte a tabela t\_clientes e mostre um exemplo com o operador OR**
8. **Consulte a tabela t\_clientes e combine os operadores between e IN**
9. **Consulte a tabela t\_clientes e combine os operadores IN e NOT IN**
10. **Consulte a tabela t\_clientes e combine os operadores LIKE e NOT LIKE.**

# Links para Downloads

- <https://www.sublimetext.com/3>
- <https://www.jetbrains.com/pt-br/pycharm/download/download-thanks.html?platform=windows&code=PCC>
- <https://www.anaconda.com/products/individual>
- <https://public.tableau.com/pt-br/s/download/thanks>
- <https://www.microsoft.com/pt-br/download/confirmation.aspx?id=55994>
- <https://docs.microsoft.com/pt-br/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>

.....  
Dúvidas ?  
Obrigado !!

[conti30@gmail.com](mailto:conti30@gmail.com)