

## ✓ Aula 22

```
1 '''
2 Criar um banco de dados chamado loja
3
4 1 - Acesse Cloud SQL
5 2 - Crie uma instância com MySQL
6 3 - Pelo card da instância criada acesse o Shell
7 4 - Pelo shell conecte-se ao Mysql
8 5 - create database loja;
9 '''
10
11 '''
12 Agora vamos criar 6 tabelas e suas relações referentes a uma loja de utensílios
13 domésticos
14 tabelas: Cliente, Pedido, Produto, ItemPedido, Categoria, ProdutoCategoria
15
16
17 create table Cliente (
18     id_cliente INT PRIMARY KEY,
19     nome VARCHAR (50),
20     endereco VARCHAR (100)
21 );
22
23 CREATE TABLE Pedido (
24     id_pedido INT PRIMARY KEY,
25     id_cliente INT,
26     descricao VARCHAR(100),
27     FOREIGN KEY (id_cliente) REFERENCES Cliente(id_cliente)
28 );
29
30 CREATE TABLE Produto (
31     id_produto INT PRIMARY KEY,
32     nome VARCHAR(50),
33     preco DECIMAL(10, 2)
34 );
35
36 CREATE TABLE ItemPedido (
37     id_item INT PRIMARY KEY,
38     id_pedido INT,
39     id_produto INT,
40     quantidade INT,
41     FOREIGN KEY (id_pedido) REFERENCES Pedido(id_pedido),
42     FOREIGN KEY (id_produto) REFERENCES Produto(id_produto)
43 );
44
45 CREATE TABLE Categoria (
46     id_categoria INT PRIMARY KEY,
47     nome VARCHAR(50)
48 );
49
50 CREATE TABLE ProdutoCategoria (
51     id_produto INT,
52     id_categoria INT,
53     PRIMARY KEY (id_produto, id_categoria),
54     FOREIGN KEY (id_produto) REFERENCES Produto(id_produto),
55     FOREIGN KEY (id_categoria) REFERENCES Categoria(id_categoria)
56 );
57
58 Agora vamos popular as 6 tabelas com dados referentes a 20 transações comerciais
59 dessa loja
60
61
62 INSERT INTO Cliente (id_cliente, nome, endereco)
63 VALUES
64     (1, 'Douglas', 'SP'),
65     (2, 'Daniela', 'SP'),
66     (3, 'Ester', 'MG'),
67     (4, 'Maria', 'BA'),
68     (5, 'Douglas', 'PE'),
69     (6, 'Raimundo', 'CE'),
70     (7, 'Jucimara', 'GO'),
71     (8, 'Marlene', 'AC'),
72     (9, 'Douglas', 'MA'),
73     (10, 'Daniela', 'PI'),
74     (11, 'Pedro', 'PB'),
75     (12, 'Joaquim', 'RS'),
76     (13, 'Julio', 'TO'),
77     (14, 'Carla', 'DF'),
```

```
78 (15, 'Bianca', 'PA'),
79 (16, 'Sandra', 'SP'),
80 (17, 'Celso', 'SP'),
81 (18, 'Silvana', 'BA'),
82 (19, 'Prof Raquel', 'BA'),
83 (20, 'Pedro', 'SP'),
84 (21, 'Erick', 'BA'),
85 (22, 'Denise', 'DF'),
86 (23, 'Diana', 'SP'),
87 (24, 'Dagmar', 'DF'),
88 (25, 'Deiverson', 'SP');
89 '''
90 '''
91 INSERT INTO Pedido (id_pedido, id_cliente, descricao)
92 VALUES
93 (1, '1', 'Caneca'),
94 (2, '1', 'Garfo'),
95 (3, '2', 'Faca'),
96 (4, '5', 'Copo'),
97 (5, '19', 'Copo'),
98 (6, '19', 'Colher'),
99 (7, '15', 'Colher'),
100 (8, '8', 'Colher'),
101 (9, '2', 'Copo'),
102 (10, '19', 'Colher'),
103 (11, '6', 'Copo'),
104 (12, '11', 'Colher'),
105 (13, '10', 'Garfo'),
106 (14, '4', 'Garfo'),
107 (15, '3', 'Colher'),
108 (16, '3', 'Garfo'),
109 (17, '2', 'Copo'),
110 (18, '17', 'Garfo'),
111 (19, '14', 'Caneca'),
112 (20, '13', 'Caneca');
113 '''
114 '''
115 INSERT INTO Produto (id_produto, nome, preco)
116 VALUES
117 (1, 'Caneca', 15.99),
118 (2, 'Garfo', 5.99),
119 (3, 'Faca', 7.99),
120 (4, 'Colher', 3.99),
121 (5, 'Copo', 10.99);
122 '''
123
124 '''
125 INSERT INTO ItemPedido (id_item, id_pedido, id_produto, quantidade)
126 VALUES
127 (1, 1, 1, 2),
128 (2, 2, 2, 5),
129 (3, 3, 3, 7),
130 (4, 4, 5, 10),
131 (5, 5, 5, 11),
132 (6, 6, 4, 30),
133 (7, 7, 4, 32),
134 (8, 8, 4, 44),
135 (9, 9, 5, 13),
136 (10, 10, 4, 33),
137 (11, 11, 5, 34),
138 (12, 12, 4, 32),
139 (13, 13, 2, 33),
140 (14, 14, 2, 34),
141 (15, 15, 4, 27),
142 (16, 16, 2, 18),
143 (17, 17, 5, 17),
144 (18, 18, 2, 33),
145 (19, 19, 1, 50),
146 (20, 20, 1, 47);
147 '''
148 '''
149 INSERT INTO Categoria (id_categoria, nome)
150 VALUES
151 (1, 'Comer'),
152 (2, 'Beber');
153
154 INSERT INTO ProdutoCategoria (id_produto, id_categoria)
155 VALUES
156 (1, 2),
157 (2, 1),
158 (3, 1),
159 (4, 1),
160 (5, 2).
```

```

161 '''

```

## ✎ Agora vamos aprender os principais comandos de consulta para o Analista de Dados

```

1 '''
2 colocar os selects * from para consulta de cada banco populado
3 '''
4 '''
5 -- seta o banco de dados, indica que usaremos o banco de dados loja
6 USE loja;
7
8 SELECT * FROM Cliente;
9 -- Lendo o comando: Seleciona todas as colunas da tabela: Cliente
10
11 SELECT * FROM Pedido;
12 -- Lendo o comando: Seleciona todas as colunas da tabela: Pedido
13
14 SELECT * FROM Produto;
15 -- Lendo o comando: Seleciona todas as colunas da tabela: Produto
16
17 SELECT * FROM ItemPedido;
18 -- Lendo o comando: Seleciona todas as colunas da tabela: ItemPedido
19
20 SELECT * FROM Categoria;
21 -- Lendo o comando: Seleciona todas as colunas da tabela: Categoria
22
23 SELECT * FROM ProdutoCategoria;
24 -- Lendo o comando: Seleciona todas as colunas da tabela: ProdutoCategoria
25
26 -- Warning: Evite o uso do *, seja específico, adicione apenas as colunas que irá precisar na sua análise.
27 -- Exemplo:
28 SELECT id_produto, quantidade FROM ItemPedido;
29
30 -- Lendo o comando: Seleciona as colunas: id_produto e quantidade da tabela ItemPedido
31 '''

```

## ✎ WHERE — São as condições

```

1 '''
2 SELECT id_produto, quantidade FROM ItemPedido
3 WHERE quantidade >= 40;
4 Lendo o comando: Seleciona as colunas: id_produto e quantidade da tabela: ItemPedido, quando a quantidade for maior ou igual a 10
5 '''
6
7 '''
8 SELECT * FROM Cliente
9 WHERE endereco = 'SP';
10
11 -- Lendo o comando: Seleciona todas as colunas, quando o endereço for igual a SP
12 '''

```

## ✎ DISTINCT

```

1 '''
2 SELECT distinct id_produto, quantidade FROM ItemPedido
3 WHERE quantidade >= 10;
4
5 -- Lendo o comando: Seleciona distintamente as colunas: id_produto e quantidade da tabela: ItemPedido, quando o quando a quantidade for
6 '''
7
8 '''
9 SELECT distinct descricao FROM Pedido;
10
11 -- Lendo o comando: Seleciona distintamente as colunas: id_produto e quantidade da tabela: ItemPedido, quando o quando a quantidade for
12 '''

```

## ✎ LIMIT — Retornar os n primeiros registros

```

1 '''
2 SELECT * FROM Cliente
3 LIMIT 5;
4 Lendo o comando: Seleciona todos os atributos da tabela Cliente e mostra
5 apenas os cinco primeiros registros
6 '''

```

## ✓ BOOLEANOS ( E | OU | NÃO ) Bom para combinar com WHERE

```

1 '''
2
3 -- Exemplo de AND
4 SELECT nome, endereco
5 FROM Cliente
6 WHERE endereco = 'BA'
7 AND id_cliente = 19;
8 Lendo o comando:
9 Seleciona as colunas: nome e endereco da tabela: Clientes,
10 apenas os registros com endereco igual a 'BA' e id_cliente igual a 19
11
12
13 -- Exemplo de OR
14 SELECT nome, endereco
15 FROM Cliente
16 WHERE endereco = 'SP'
17 OR endereco = 'MG';
18 Lendo o comando:
19 Seleciona as colunas: nome e endereco da tabela: Clientes,
20 apenas os registros com endereco igual a 'SP' ou endereco a 'MG'
21
22 -- Exemplo de NOT
23 SELECT nome, endereco
24 FROM Cliente
25 WHERE NOT endereco = 'SP';
26 Lendo o comando:
27 Seleciona as colunas: nome e endereco da tabela: Clientes,
28 apenas os registros com endereco não seja igual a 'SP'

```

## ✓ Funções de agregação ( SUM , AVG , MAX , MIN , COUNT e ROUND )

```

1 '''
2 Somatório da quantidade de produtos vendidos
3
4 SELECT SUM(quantidade) AS soma_qtde
5 FROM ItemPedido;
6 '''
7
8 '''
9 Média da quantidade de produtos vendidos
10
11 SELECT AVG(quantidade) AS media_qtde,
12 round(AVG(quantidade),2) AS media_qtde_arredondado
13 FROM ItemPedido;
14 '''
15
16
17 '''
18 Maior quantidade de produtos vendidos
19
20 SELECT MAX(quantidade) AS maior_quantidade_venda
21 FROM ItemPedido;
22 '''
23 '''
24 SELECT MIN(quantidade) menor_quantidade_venda
25 FROM ItemPedido;
26 '''
27
28 '''
29 Contagem de produtos (quantos produtos temos na nossa base?)
30
31 SELECT count(id_produto) as qtd_produtos
32 FROM Produto;
33 '''

```

## ✓ Aula 23

### ✓ GROUP BY

```
1 '''
2 SELECT endereco,
3     count(*) AS 'freq'
4 FROM Cliente
5 GROUP BY endereco;
6 '''
7
8 '''
9 Lendo o código: Comando que seleciona o atributo endereço, cria e seleciona a
10 coluna count renomeada com freq da tabela Cliente e agrupa por endereco
11 '''
```

### ✓ HAVING

```
1 '''
2 SELECT endereco,
3     count(*)
4 FROM Cliente
5 GROUP BY endereco
6 HAVING count(*) > 1;
7 '''
8
9 '''
10 Lendo o código: Consulta que seleciona endereco e contagem de endereco da
11 tabela Cliente, Agrupado por endereço tendo apenas (HAVING) os endereços que se
12 repetem pelo menos duas vezes.
13 '''
```

### ✓ UNION

```
1 '''
2 Serve para unir duas consultas diferentes e mostá-la na mesma tabela
3
4 SELECT id_pedido, id_produto
5     FROM ItemPedido
6 WHERE id_produto = 1
7 UNION
8 SELECT id_pedido, id_produto
9     FROM ItemPedido
10 WHERE id_produto = 4;
11
12 Lendo o código: Unindo duas consultas, sendo a primeira como a seleção de
13 id_pedido e id_produto da tabela ItemPedido onde o id_produto é 1, e a segunda
14 como a seleção de id_pedido e id_produto da tabela ItemPedido onde o
15 id_produto é 4
16 '''
```

### ✓ IN BETWEEN

```
1 '''
2 SELECT
3   id_pedido,
4   id_produto
5 FROM ItemPedido
6 WHERE id_produto = 1 OR id_produto = 2;
7
8 Alternativa seria usar IN
9
10 SELECT
11   id_pedido,
12   id_produto
13 FROM ItemPedido
14 WHERE id_produto IN (1,2);
15 '''
16
17 '''
18 Essas duas consultas anteriores trazem exatamente a mesma resposta, porém de
19 formas diferentes
20 '''
21 '''
22 SELECT
23   id_pedido,
24   id_produto
25 FROM ItemPedido
26 WHERE id_produto = 1 OR id_produto = 2;
27 '''
28 '''
29 Lendo a consulta: Selecionar id_pedido e id_produto da tabela ItemPedido
30 onde id_produto seja igual a 1 ou igual a dois usando where combinado com or
31 '''
32 '''
33 SELECT
34   id_pedido,
35   id_produto
36 FROM ItemPedido
37 WHERE id_produto IN (1,2);
38
39 '''
40
41 '''
42 Agora vamos imaginar que em relação ao que tivemos de retorno na consulta
43 anterior nosso objetivo seja ainda mais restritivo: Agora além de
44 Selecionar id_pedido e id_produto da tabela ItemPedido onde id_produto seja
45 igual a 1 ou igual a dois queremos também que o número do pedido esteja
46 entre (BETWEEN) 1 e 9, pois o nosso objetivo é selecionar
47
48 SELECT
49   id_pedido,
50   id_produto
51 FROM ItemPedido
52 WHERE id_produto IN (1,2) AND
53   id_pedido BETWEEN 1 AND 9;
54
55 Lendo a consulta: Selecionar id_pedido e id_produto da tabela ItemPedido
56 onde id_produto seja igual a 1 ou igual a dois usando IN E além disso, o número
57 do pedido esteja entre (BETWEEN) 1 e 9, pois o nosso objetivo é selecionar
58 somente
59 os id_pedido de 1 dígito.
60
61 '''
62
63
```

## ✓ LIKE

```
1 '''
2 Agora vamos para a tabela Produto e no atributo nome vamos realizar uma consulta
3 que nos retorne todos os produtos que terminem com a letra o e os seus
4 respectivos ids
5
6 SELECT
7   id_produto,
8   nome
9 FROM Produto
10 WHERE nome LIKE '%o';
11
12 Agora vamos para a tabela Produto e no atributo nome vamos realizar uma consulta
13 que nos retorne todos os produtos que comecem com a letra C e os seus
14 respectivos ids
15
16 SELECT
17   id_produto,
18   nome
19 FROM Produto
20 WHERE nome LIKE 'C%';
21 '''
22
23 '''
24 Agora vamos para a tabela Produto e no atributo nome vamos realizar uma consulta
25 que nos retorne todos os produtos que contenha a sequencia lh e os seus
26 respectivos ids
27
28
29 SELECT
30   id_produto,
31   nome
32 FROM Produto
33 WHERE nome LIKE '%lh%';
34
35 '''
```

#### NULL e NOT NULL

```
1 '''
2 Verificar a existência de valores nulos para o atributo nome na tabela Produto
3
4 SELECT nome
5 FROM Produto
6 WHERE nome IS NULL;
7
8
9 Verificar a existência de valores não nulos para o atributo
10 nome na tabela Produto
11
12 SELECT nome
13 FROM Produto
14 WHERE nome IS NOT NULL;
15 '''
16
```

#### ORDER BY

```
1 '''
2 Vamos selecionar todos os atributos da tabela ItemPedido, ordenando de forma
3 Crescente por quantidade
4
5 SELECT * FROM ItemPedido
6 ORDER BY quantidade ASC;
7
8 Vamos selecionar todos os atributos da tabela ItemPedido, ordenando de forma
9 Decrescente por quantidade
10
11 SELECT id_pedido FROM ItemPedido
12 ORDER BY quantidade DESC;
13 '''
```

#### TRIM - Remove os espaços

```
1 '''
2 SELECT nome,
3   TRIM(nome) AS nome_no_space
4 FROM Produto;
5 '''
```

REPLACE Substitui um caracter ou sequencia de caracteres

```
1 '''
2 SELECT nome,
3   REPLACE(nome, 'o', 'oo') AS nome_zuado
4 FROM Produto;
5 '''
```

LPAD

```
1 '''
2 Vamos deixar todos os ids de pedidos na tabela ItemPedido com cinco dígitos,
3 Assim ele preencherá todos os valores com zeros à esquerda que forem necessários
4 para o números der dígitos solicitado
5
6 SELECT id_pedido,
7   LPAD(id_pedido, 5, "0")
8 FROM ItemPedido;
9 '''
```

SUBSTRING

```
1 '''
2 SELECT nome,
3   SUBSTRING(nome, 1, 2) AS subs
4 FROM Produto;
5 '''
```

UPPER

```
1
2 '''
3 Deixando todos os valores de um atributo com todas as letras maiúsculas
4 SELECT nome,
5   UPPER(nome)
6 FROM Produto;
7 '''
8 '''
9 Deixando todos os valores de um atributo com todas as letras minúsculas
10 SELECT nome,
11   LOWER(nome)
12 FROM Produto;
13 '''
14 '''
15 SELECT nome,
16   LENGTH(nome)
17 FROM Produto;
18 '''
```

## ✓ CAST

```
1 '''
2 SELECT id_cliente,
3   concat('Cliente - ', CAST(id_cliente AS CHAR)) AS id_cliente_texto
4 FROM Cliente;
5 '''
```

## ✓ CASE — Semelhante ao if/else



```

1 '''
2 SELECT id_pedido, quantidade,
3       CASE WHEN quantidade > 30 THEN 'Atacado'
4       ELSE 'Varejo' END
5 FROM ItemPedido;
6 '''
7 '''
8 Lendo o código: Selecione os atributos id_pedido, quantidade
9 (1ª linha do bloco de código) da tabela ItemPedido
10 (4ª linha do bloco de código) e quando a quantidade for maior do que 30
11 devolva em uma coluna separada a classificação daquele pedido como atacado
12 caso seja menor que 30 classifique como varejo e depois encerre (END)
13 '''

```

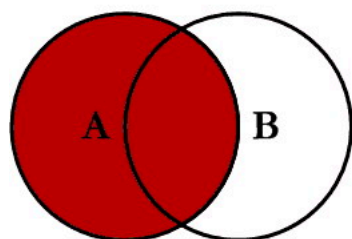
## VIEW

```

1 '''
2 CREATE OR REPLACE VIEW query AS
3 SELECT id_pedido, quantidade,
4       CASE WHEN quantidade > 30 THEN 'Atacado'
5       ELSE 'Varejo' END
6 FROM ItemPedido;
7
8 '''

```

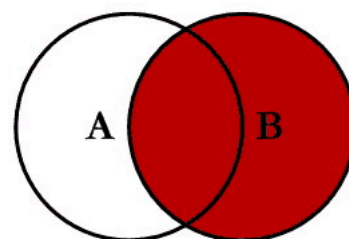
# SQL JOINS



```

SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key

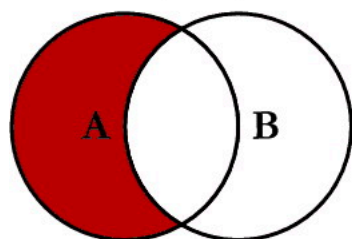
```



```

SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key

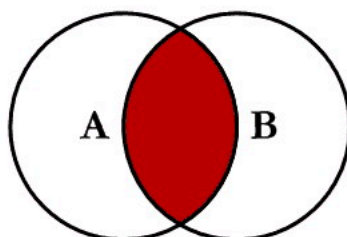
```



```

SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL

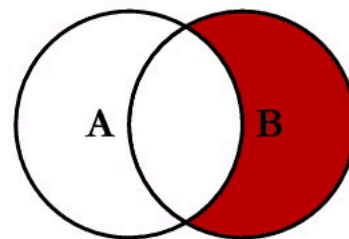
```



```

SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key

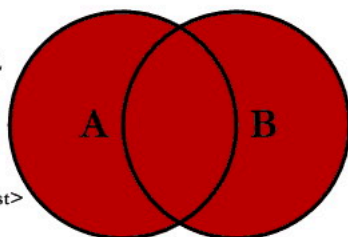
```



```

SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL

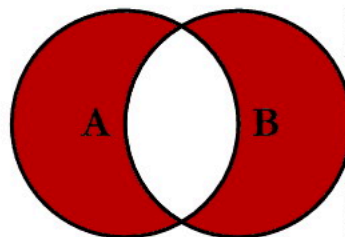
```



```

SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key

```



```

SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL

```

© C.L. Moffatt, 2008

```

1 '''
2 Inserindo novos clientes na tabela Cliente para melhorar a visualização dos joins
3 INSERT INTO Cliente (id_cliente, nome, endereco)
4 VALUES
5   (21, 'Erick', 'BA'),
6   (22, 'Denise', 'DF'),
7   (23, 'Diana', 'SP'),
8   (24, 'Dagmar', 'DF'),
9   (25, 'Deiverson', 'SP');
10 '''

```

```
1 '''
2 SELECT *
3   FROM ItemPedido v
4   INNER JOIN ProdutoCategoria c
5     ON v.id_produto = c.id_produto;
6
7 '''
8
9
10 '''
11 SELECT *
12   FROM Cliente c
13   LEFT JOIN Pedido p
14     ON p.id_cliente = c.id_cliente;
15 '''
16 '''
17 SELECT *
18   FROM Cliente c
```