

## Consolidando o seu conhecimento

Chegou a hora de você seguir todos os passos realizados por mim durante esta aula. Caso já tenha feito, excelente. Se ainda não, é importante que você execute o que foi visto nos vídeos para poder continuar com a próxima aula.

1) Abra um novo script MySQL.

2) Digite e execute:

```
USE `sucos_vendas`;
DROP procedure IF EXISTS `cliente_novo_velho`;
DELIMITER $$

USE `sucos_vendas`$$
CREATE PROCEDURE `cliente_novo_velho` (vCPF VARCHAR(20))
BEGIN
    DECLARE vResultado VARCHAR(20);
    DECLARE vDataNascimento DATE;
    SELECT DATA_DE_NASCIMENTO INTO vDataNascimento FROM
    tabela_de_clientes WHERE CPF = vCPF;
    IF vDataNascimento < '20000101' THEN
        SET vResultado = 'CLIENTE VELHO';
    ELSE
        SET vResultado = 'CLIENTE NOVO';
    END IF;
    SELECT vResultado;
END$$
DELIMITER ;
```

2) Execute a SP.

```
Call cliente_novo_velho ('19290992743');
```

Result Grid		Filter Row
vResultado		
CLIENTE NOVO		

3) Esta SP usa a estrutura IF-THEN-ELSE para classificar um cliente como sendo novo ou velho, baseado em sua idade.

4) Vamos ver outra estrutura de controle de fluxo. Digite e execute:

```
USE `sucos_vendas`;
DROP procedure IF EXISTS `acha_status_preco_2`;
DELIMITER $$

USE `sucos_vendas`$$
CREATE PROCEDURE `acha_status_preco_2` (vProduto VARCHAR(50))
BEGIN
    DECLARE vPreco FLOAT;
    DECLARE vMensagem VARCHAR(30);
```

```

SELECT PRECO_DE_LISTA INTO vPreco FROM tabela_de_produtos
WHERE codigo_do_produto = vProduto;
IF vPreco >= 12 THEN
    SET vMensagem = 'PRODUTO CARO. ';
ELSEIF vPreco >= 7 AND vPreco < 12 THEN
    SET vMensagem = 'PRODUTO EM CONTA. ';
ELSE
    SET vMensagem = 'PRODUTO BARATO. ';
END IF;
SELECT vMensagem;
END$$
DELIMITER ;

```

5) Digite e execute:

```
Call acha_status_preco_2 ('1000889');
```

6) Observe que a estrutura IF-THEN-ELSEIF permite encadear diversos testes.

7) O encadeamento de condições pode ser expresso, também, usando o comando CASE-END-CASE. Para isso digite e execute:

```

USE `sucos_vendas`;
DROP procedure IF EXISTS `acha_tipo_sabor`;
DELIMITER $$

USE `sucos_vendas`$$
CREATE PROCEDURE `acha_tipo_sabor` (vProduto VARCHAR(50))
BEGIN
    DECLARE vSabor VARCHAR(50);
    SELECT SABOR INTO vSabor FROM tabela_de_Produtos
    WHERE codigo_do_produto = vProduto;
    CASE vSabor
        WHEN 'Lima/Limão' THEN SELECT 'Cítrico';
        WHEN 'Laranja' THEN SELECT 'Cítrico';
        WHEN 'Morango/Limão' THEN SELECT 'Cítrico';
        WHEN 'Uva' THEN SELECT 'Neutro';
        WHEN 'Morango' THEN SELECT 'Neutro';
        ELSE SELECT 'Ácidos';
    END CASE;
END$$
DELIMITER ;

```

8) Digite e execute:

```
CALL acha_tipo_sabor('1000889');
```

9) Uma estrutura derivada da mostrada é o CASE-NOT-FOUND. Para isso vamos criar a SP Acha\_Tipo\_Sabor\_Erro. Digite e execute:

```

USE `sucos_vendas`;
DROP procedure IF EXISTS `acha_tipo_sabor_erro`;
DELIMITER $$

USE `sucos_vendas`$$
CREATE PROCEDURE `acha_tipo_sabor_erro`(vProduto VARCHAR(50))
BEGIN
    DECLARE vSabor VARCHAR(50);
    SELECT SABOR INTO vSabor FROM tabela_de_Produtos
    WHERE codigo_do_produto = vProduto;
    CASE vSabor
        WHEN 'Lima/Limão' THEN SELECT 'Cítrico';
        WHEN 'Laranja' THEN SELECT 'Cítrico';
        WHEN 'Morango/Limão' THEN SELECT 'Cítrico';
        WHEN 'Uva' THEN SELECT 'Neutro';
        WHEN 'Morango' THEN SELECT 'Neutro';
    END CASE;
END$$
DELIMITER ;

```

10) Esta SP difere da anterior apenas porque foi retirada a linha abaixo.

```
ELSE SELECT 'Ácidos';
```

Estamos criando uma situação onde nenhuma das condições podem ser satisfeitas.

11) Execute a SP:

```
CALL acha_tipo_sabor_erro('1004327');
```

12) Temos o erro:

Changes applied  
Error Code: 1339. Case not found for CASE statement

13) Para evitar de não incluirmos opções no CASE que contemplem todas as situações podemos acrescentar o tratamento de erro. Altere a SP executando:

```

USE `sucos_vendas`;
DROP procedure IF EXISTS `acha_tipo_sabor_erro`;
DELIMITER $$

USE `sucos_vendas`$$
CREATE PROCEDURE `acha_tipo_sabor_erro`(vProduto VARCHAR(50))
BEGIN
    DECLARE vSabor VARCHAR(50);
    DECLARE msgErro VARCHAR(30);
    DECLARE CONTINUE HANDLER FOR 1339 SET msgErro = 'O case não está completo';
    SELECT SABOR INTO vSabor FROM tabela_de_Produtos
    WHERE codigo_do_produto = vProduto;
    CASE vSabor
        WHEN 'Lima/Limão' THEN SELECT 'Cítrico';
        WHEN 'Laranja' THEN SELECT 'Cítrico';
        WHEN 'Morango/Limão' THEN SELECT 'Cítrico';
        WHEN 'Uva' THEN SELECT 'Neutro';
        WHEN 'Morango' THEN SELECT 'Neutro';
    END CASE;
    IF msgErro != '' THEN SELECT msgErro;
    END IF;
END$$
DELIMITER ;

```

```

WHEN 'Laranja' THEN SELECT 'Cítrico';
WHEN 'Morango/Limão' THEN SELECT 'Cítrico';
WHEN 'Uva' THEN SELECT 'Neutro';
WHEN 'Morango' THEN SELECT 'Neutro';
END CASE;
SELECT msgErro;
END$$
DELIMITER ;

```

14) Execute a SP:

```
CALL acha_tipo_sabor_erro('1004327');
```

Teremos o erro tratado.

![2.png](https://caelum-online-public.s3.amazonaws.com/1223-mysqlproceduresefuncions/04/2.png)

15) O Case Condisional utiliza uma estrutura de Case semelhante a usada quando executamos um comando

```
USE sucos_vendas ; DROP procedure IF EXISTS acha_status_preco_case ; DELIMITER $$
```

```

USE sucos_vendas $$ CREATE PROCEDURE acha_status_preco_case (vProduto VARCHAR(50)) BEGIN DECLARE vPreco
FLOAT; DECLARE vMensagem VARCHAR(30); SELECT PRECO_DE_LISTA INTO vPreco FROM tabela_de_produtos WHERE
codigo_do_produto = vProduto; CASE WHEN vPreco >= 12 THEN SET vMensagem = 'PRODUTO CARO.'; WHEN vPreco >= 7
AND vPreco < 12 THEN SET vMensagem = 'PRODUTO EM CONTA.'; WHEN vPreco < 7 THEN SET vMensagem = 'PRODUTO
BARATO'; END CASE; SELECT vMensagem; END$$ DELIMITER ;

```

16) Digite e execute:

```
CALL acha_status_preco_case('1004327');
```

17) A estrutura de Loop permite repetir um conjunto de comandos até que determinada condição aconteça

```
CREATE TABLE TAB_LOOPING (ID INT);
```

18) Após a criação de uma tabela auxiliar digite e execute

```
USE sucos_vendas ; DROP procedure IF EXISTS looping_while ; DELIMITER $$
```

```

USE sucos_vendas $$ CREATE PROCEDURE looping_while (vNumInicial INT, vNumFinal INT) BEGIN DECLARE vContador
INT; DELETE FROM TAB_LOOPING; SET vContador = vNumInicial; WHILE vContador <= vNumFinal DO INSERT INTO
TAB_LOOPING (ID) VALUES (vContador); SET vContador = vContador + 1; END WHILE; SELECT * FROM TAB_LOOPING;
END$$ DELIMITER ;

```

19) Vamos executar a SP passando como parâmetro um número inicial e um número **final** para criação de

```
call looping_while (1, 1000);
```

ID
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Teremos uma sequência numérica que vai de 1 a 1000.