Tarefa 23 - Stored Procedures – MySQL - Solução
Prof. Dr. Aparecido Freitas

1. Introdução

Para a execução desta atividade iremos considerar um Banco de Dados chamado **produtodb**, com as seguintes definições de tabelas:

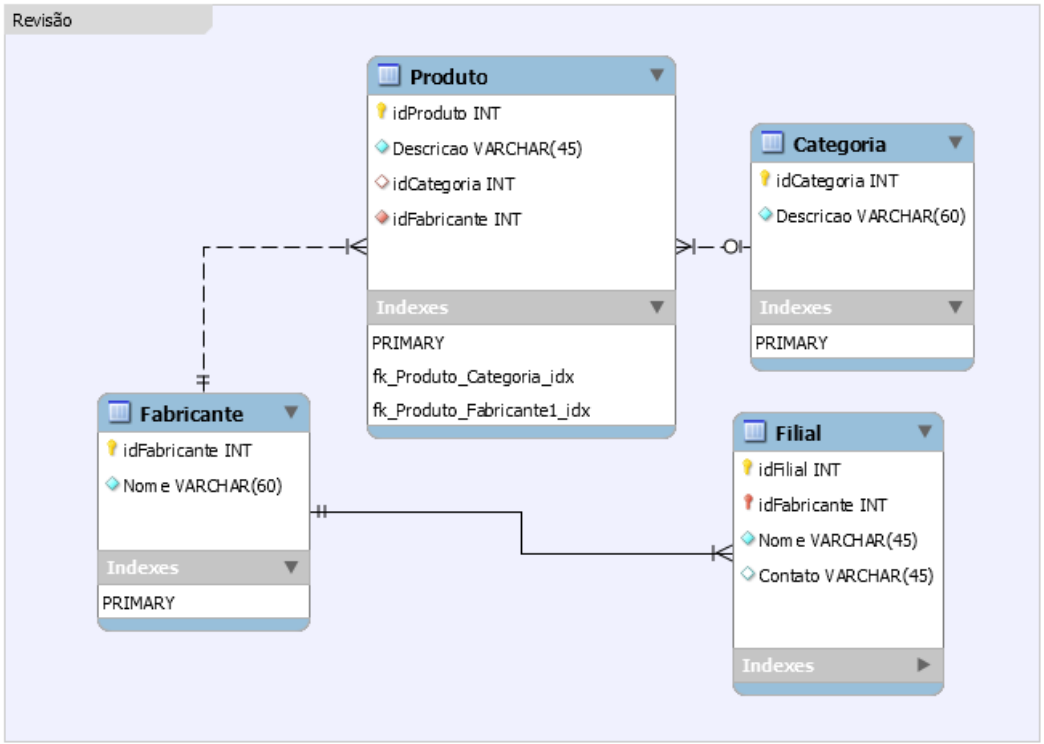
```
CREATE TABLE Fabricante (
  idFabricante INT(11) NOT NULL,
  Nome VARCHAR(60) NOT NULL,
  PRIMARY KEY (idFabricante)
);

CREATE TABLE Categoria (
  idCategoria INT(11) NOT NULL,
  Descricao VARCHAR(60) NOT NULL,
  PRIMARY KEY (idCategoria)
);

CREATE TABLE Produto (
  idProduto INT(11) NOT NULL,
  Descricao VARCHAR(45) NOT NULL,
  idCategoria INT(11) NULL DEFAULT NULL,
  idFabricante INT(11) NOT NULL,
  PRIMARY KEY (idProduto),
  INDEX fk_Produto_Categoria_idx (idCategoria ASC),
  INDEX fk_Produto_Fabricante1_idx (idFabricante ASC),
  CONSTRAINT fk_Produto_Categoria FOREIGN KEY (idCategoria) REFERENCES Categoria (idCategoria),
  CONSTRAINT fk_Produto_Fabricante1 FOREIGN KEY (idFabricante) REFERENCES Fabricante (idFabricante)
);
CREATE TABLE Filial (
  idFilial INT(11) NOT NULL,
  idFabricante INT(11) NOT NULL,
  Nome VARCHAR(45) NOT NULL,
  Contato VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (idFilial, idFabricante),
  INDEX fk_Filial_Fabricante1_idx (idFabricante ASC),
  CONSTRAINT fk_Filial_Fabricante1 FOREIGN KEY (idFabricante) REFERENCES Fabricante (idFabricante)
);
```

INSERT INTO fabricante (idFabricante, Nome) VALUES ('1', 'Nestlé');INSERT INTO fabricante (idFabricante, Nome) VALUES ('2', 'Parmalat');
INSERT INTO fabricante (idFabricante, Nome) VALUES ('3', 'Kelloggs');
INSERT INTO categoria (idCategoria, Descricao) VALUES ('1', 'Leite');
INSERT INTO categoria (idCategoria, Descricao) VALUES ('2', 'Cereais Matinais');
INSERT INTO categoria (idCategoria, Descricao) VALUES ('3', 'Achocolatado');
INSERT INTO produto (idProduto, Descricao, idCategoria, idFabricante) VALUES (1,'Leite Integral', '1', '1');
INSERT INTO produto (idProduto, Descricao, idCategoria, idFabricante) VALUES (2,'Nescau', '3', '1');
INSERT INTO produto (idProduto, Descricao, idCategoria, idFabricante) VALUES (3,'Sucrilhos', '2', '3');

Empregaremos o seguinte modelo de dados:



A utilização de Stored Procedures com apenas um comando não representa toda sua magnitude.

```
CREATE PROCEDURE sp_MinhaPrimeira() SELECT 'Olá Mundo!!!';
```

É possível a utilização de blocos de comandos, através da sintaxe a seguir:

```
CREATE PROCEDURE sp_Bloco()
BEGIN
    SELECT 'Deu Ruim!!!';
END;
```

Repare que essa instrução está com sua sintaxe correta. Então porque essa ela falhou?

O problema da procedure está relacionado aos terminadores de sintaxe do comando.

Qual o delimitador que encerra uma instrução SQL?

```
SELECT p.idProduto, p.Descricao, f.idFabricante, f.Nome
FROM Produto p
INNER JOIN fabricante f USING(idFabricante)
WHERE p.idProduto = varCodProduto;
```

Para se escrever procedures sintaticamente corretas deve-se utilizar **delimitadores**.

Vamos então reescrever nossa procedure:

```
DELIMITER $$
CREATE PROCEDURE sp_Bloco()
BEGIN

    SELECT 'Agora sim!!!';

END$$
DELIMITER ;
```

O primeiro comando é **DELIMITER $$,** que não está relacionado com a sintaxe do procedimento armazenado.

A declaração **DELIMITER** muda o delimitador padrão que é ponto e vírgula (;) para outro. Neste caso, o delimitador é alterado a partir do ponto e vírgula (;) para $$

Por que nós temos que mudar o delimitador? Porque queremos passar o procedimento armazenado para o servidor como um **TODO** ao invés de deixar o **MYSQL** interpretar cada instrução de cada vez.

Seguindo a palavra-chave **END**, usamos o delimitador **$$** para indicar o fim do procedimento armazenado.

O último comando (**DELIMITER** ;) muda o delimitador de volta para o padrão.

A Stored Procedure a seguir é utilizada para atribuir a palavra "**Novo**" em um determinado produto e em seu fabricante.

```
DELIMITER $$
CREATE PROCEDURE sp_Atualizando()
BEGIN
    UPDATE produto
    SET descricao = CONCAT(descricao, ' NOVO')
    WHERE idProduto = 1;

    UPDATE fabricante
    SET nome = CONCAT(nome, ' NOVO')
    WHERE idFabricante = 2;
END$$
DELIMITER ;
```

Crie a procedure acima definida e a execute sob o Cliente **MySQL**.

As tabelas após a execução da **Stored Procedure** deverão ter as seguintes tuplas:

```
+-----------+----------------------+-------------+--------------+
| idProduto | Descricao            | idCategoria | idFabricante |
+-----------+----------------------+-------------+--------------+
|         1 | Leite Integral  Novo |           1 |            1 |
|         2 | Nescau               |           3 |            1 |
|         3 | Sucrilhos            |           2 |            3 |
+-----------+----------------------+-------------+--------------+
3 rows in set (0.00 sec)

mysql> select * from fabricante;
+--------------+----------------+
| idFabricante | Nome           |
+--------------+----------------+
|            1 | Nestlé         |
|            2 | Parmalat  Novo |
|            3 | Kelloggs       |
+--------------+----------------+
3 rows in set (0.02 sec)
```

Crie uma **Stored Procedure** chamada **SP_PraticandoBloco**. Ao ser invocada essa procedure deve realizar a inclusão das seguintes categorias:

4 - Bebidas não Alcoólicas

5 - Suplemento Alimentar

6 - Desinfetantes

7 - Desodorantes

8 - Jogos de Vídeo Game

E a inclusão do Fabricante:

4 - Nivea

Após invocar essa **Stored Procedure** verifique se os registros foram incluídos com sucesso.

Solução:

```
DELIMITER $$
CREATE procedure sp_PraticandoBloco()
BEGIN
    INSERT INTO categoria (idCategoria, Descricao)
    VALUES (4, 'Bebidas não Alcoólicas'),
           (5, 'Suplemento Alimentar'),
           (6, 'Desinfetantes'),
           (7, 'Desodorantes'),
           (8, 'Jogos de Vídeo Game');

    INSERT INTO fabricante
    VALUES (4, 'Nivea');
END$$
DELIMITER ;
```