Banco de Dados

Aula 16 - Linguagem SQL – Funções

Apresentação

Na aula anterior, estudamos como criar, executar e apagar procedimentos armazenados (*stored procedures*), bem como utilizá-los em conjunto com estruturas de controle de fluxo de dados. Nesta aula, você vai estudar o conceito de funções, que são usadas para encapsular operações úteis para manipulação do acesso de dados em um sistema de banco de dados. Aprenderemos a criar, executar e apagar essas estruturas, bem como utilizá-las no processamento de consultas. Também será discutido como se pode declarar variáveis no SQL.

Vídeo 01 - Apresentação

Objetivos

- Criar funções com e sem parâmetros no sistema MySQL.
- Declarar variáveis locais que podem ser utilizadas dentro das funções.
- Utilizar as funções para auxílio no processamento de consultas.

Funções

Assim como ocorre com os procedimentos, é possível ter uma sequência de comandos SQL encapsulados em estruturas denominadas **funções**. Como você viu em disciplinas anteriores, que trataram de lógica e programação, a principal diferença entre uma função e um procedimento está no fato de que a função obrigatoriamente deve retornar um valor. Nesta disciplina, já trabalhamos, em aulas anteriores, com funções internas, pré-definidas pelo próprio SGBD, como AVG(), SUM(), COUNT() etc. Mas o usuário pode definir suas próprias funções com parâmetros de entrada e variáveis locais. É possível no SQL construir dois tipos de funções:

- **Funções escalares:** estruturas semelhantes a funções internas, que retornam um único valor.
- **Funções com valor de tabela:** estruturas semelhantes a visões com a diferença de aceitarem parâmetros de entrada, que retornam uma tabela como resultado do seu processamento.

No caso do MySQL, não é permitido que uma função retorne uma tabela. Desse modo, vamos estudar apenas como criar e utilizar as **funções escalares**.

Funções escalares

Uma função escalar retorna um único valor de dados de um tipo predefinido e pode ser utilizada do mesmo modo que uma função interna, sendo mais usada como:

- Uma expressão na lista de um comando SELECT.
- Uma expressão em uma cláusula WHERE ou HAVING.
- Uma expressão em uma cláusula ORDER BY ou GROUP BY.
- Uma expressão na cláusula SET de um comando UPDATE.
- Uma expressão na cláusula VALUES de um comando INSERT.

A instrução para criar uma função é simples, basta utilizar o comando CREATE FUNCTION em conjunto com a lista de parâmetros (caso necessário) a serem usados. A sintaxe de criação de uma função é descrita a seguir.

```
mysql> CREATE FUNCTION  nome_da_funcao (parâmetros de entrada)
   RETURNS tipo_de_retorno
   BEGIN
      comandos em SQL
      RETURN valor_de_retorno;
   END;
```

Nessa expressão, no campo *nome_da_funcao* deve-se inserir o nome que se deseja atribuir para a função. Esse nome deve seguir as mesmas regras usadas para os nomes das tabelas, que foi visto na **Aula 9**. Em seguida, caso haja parâmetros, deve-se fornecer a lista de parâmetros de entrada, que segue o mesmo formato definido na aula anterior sobre **procedimentos armazenados**. Mesmo não havendo parâmetros, devem ser inseridos os parênteses () na criação da função. A cláusula RETURNS define o tipo de dado que será retornado pela função, por exemplo, um número inteiro, um conjunto de caracteres etc. Finalmente, entre as palavras BEGIN e END devem ser descritos os comandos SQL que definem a operação a ser executada. Lembrando que antes do END deve ser usada a cláusula RETURN que define a variável ou expressão que irá retornar o resultado da função.

Para termos o retorno da função por meio de uma variável, é necessário sabermos como se faz a declaração de variáveis locais no MySQL. Isso é feito usando o comando DECLARE, cuja sintaxe é apresentada a seguir.

```
mysql> DECLARE nome_da_variavel tipo_de_dado;
```

Nessa expressão, no campo *nome_da_variavel* deve-se inserir o nome da variável que está sendo criada e esse nome deve seguir as mesmas regras usadas para os nomes das tabelas, que foi visto na **Aula 9**. O mesmo acontece para a definição do tipo de dado que será atribuído à variável.

Agora, vamos exercitar a criação de funções no banco de dados **sistvendas** para entendermos melhor o seu conceito? Se você quiser relembrar a definição desse banco de dados, consulte a **Aula 13**. Para começar, vamos criar uma função que retorne a quantidade total de produtos vendidos. O comando para criar essa função é descrito no destaque a seguir.

```
mysql> DELIMITER |
CREATE FUNCTION  Total_Vendas()
   RETURNS int
   BEGIN
      DECLARE qtde_vendas int;
      SELECT sum(comp_total) INTO qtde_vendas
      FROM compras;
      RETURN qtde_vendas;
   END
|
```

Observe que foi criada uma função denominada Total_Vendas, sem qualquer parâmetro, que retorna um valor do tipo inteiro. Os comandos que definem sua operação encontram-se entre as palavras chave BEGIN e END.

Analisando cuidadosamente os comandos, vemos o uso do comando DECLARE para definir uma variável intitulada qtde_vendas do tipo inteiro. Em seguida, é feita uma consulta (SELECT) da soma de todos os produtos vendidos da tabela compras. Esse valor é armazenado na variável qtde_vendas usando a cláusula INTO. Finalmente, a cláusula RETURN define que a função retorna o valor contido na variável qtde_vendas.

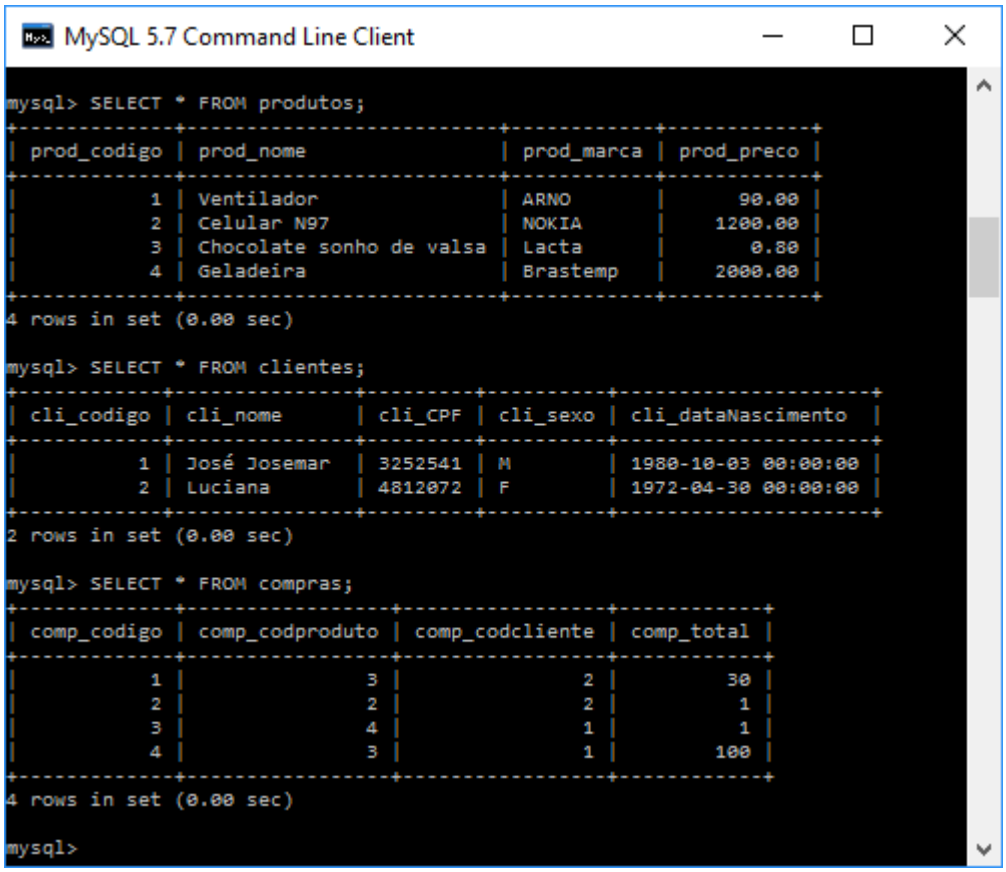
Agora, a partir dessa função, vamos criar outra que retorne a quantidade total de unidades vendidas de um produto dado o seu código. O comando para criar essa função é descrito a seguir.

```
mysql> DELIMITER |
CREATE FUNCTION  Total_Vendas2 (codigo_produto int)
   RETURNS int
   BEGIN
      DECLARE qtde_vendas int;
      SELECT sum(comp_total) INTO qtde_vendas
      FROM compras
      WHERE comp_codproduto = codigo_produto;
      RETURN qtde_vendas;
   END
|
```

Compare a diferença entre as duas funções apresentadas. Observe que a função denominada Total_Vendas2 contém um parâmetro de entrada do tipo inteiro. E que é feita uma consulta (SELECT) da soma de todos os produtos da tabela **compras** cujo código equivale ao parâmetro de entrada codigo_produto.

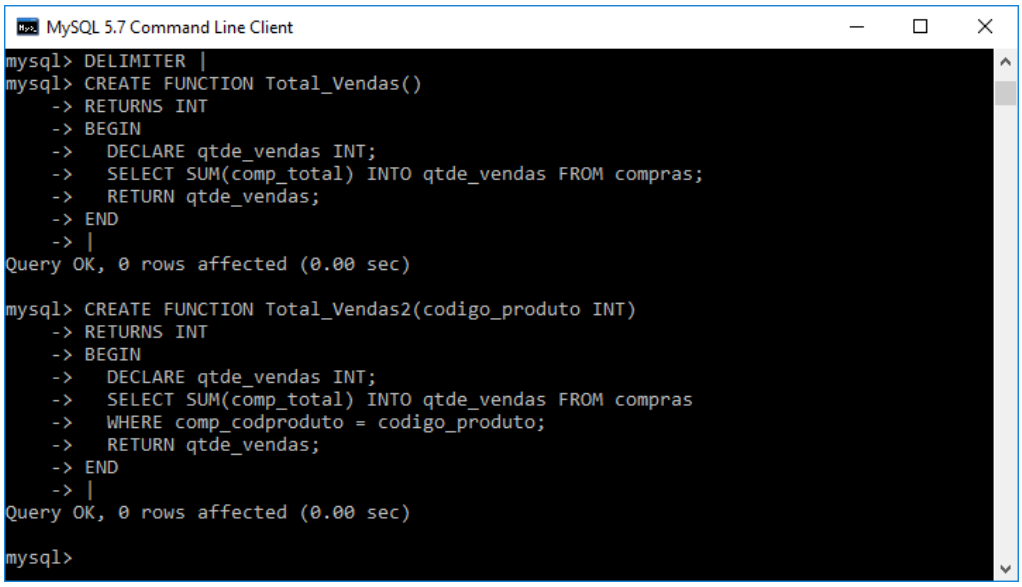
Para analisarmos a aplicação das funções no banco de dados **sistvendas**, apresentam-se, na **Figura 1**, os dados presentes nas tabelas do banco de dados. A **Figura 2** ilustra a resposta do SGBD após a criação da função.

Figura 01 - Tela do MySQL após os comandos SELECT * FROM **produtos**, SELECT * FROM **clientes** e SELECT * FROM **compras**.



Fonte: MySQL 5.7 Command Line Client

Figura 02 - Tela do MySQL após os comandos CREATE FUNCTION.



Fonte: MySQL 5.7 Command Line Client

Note que está sendo usado o caractere "|" como delimitador de comandos, do mesmo modo como no procedimento, isso é feito para que possamos usar o caractere ";" no meio da função.

Vídeo 02 - Introdução a Funções

Atividade 01

1. O que são funções?
2. Qual a diferença entre funções e procedimentos?
3. Quais os tipos de funções que podem ser definidas em SQL?
4. Acesse o banco de dados sispagamentos, visto na Aula 14, e elabore o que se pede.

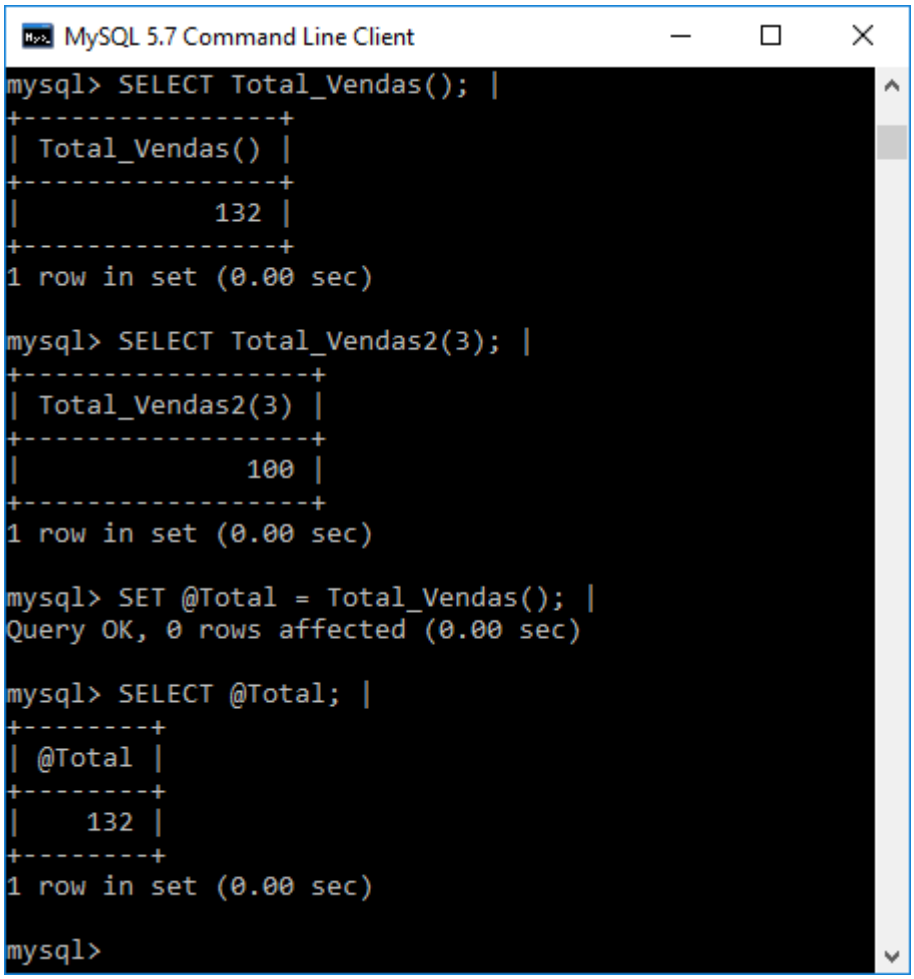
1. Crie uma função que retorne o desconto de INSS total dos empregados.
2. Crie uma função que retorne o total de descontos aplicados no salário de um determinado empregado.
3. Crie uma função que retorne a quantidade total de empregados de um determinado sexo.

Utilizando funções escalares

Agora que já sabemos criar funções, vamos exemplificar duas formas de executá-las, imprimindo seu resultado na tela. Observe os comandos ilustrados no destaque a seguir e, na **Figura 3**, a resposta do SGBD à sua execução.

```
mysql> SELECT Total_Vendas();
mysql> SELECT Total_Vendas2(3);
mysql> SET @Total = Total_Vendas();
mysql> SELECT @Total;
```

Figura 03 - Tela do MySQL após diversos comandos SELECT e SET.



Fonte: MySQL 5.7 Command Line Client

Os dois primeiros exemplos utilizam o comando SELECT para imprimir na tela o resultado das funções Total_Vendas() e Total_Vendas2(), sendo passado nesta última o parâmetro 3, ou seja, se deseja saber o número de itens vendidos do produto de código igual a 3. Resultado similar pode ser obtido utilizando o comando SET para atribuir o resultado de uma função a uma variável e em seguida imprimindo o mesmo na tela usando SELECT. Observe que nesse caso o SGBD reconhece que uma variável está sendo definida pela inserção do caractere “@” antes do seu nome e que não é necessário associar um tipo específico a ela, sendo uma forma alternativa de se definir uma variável.

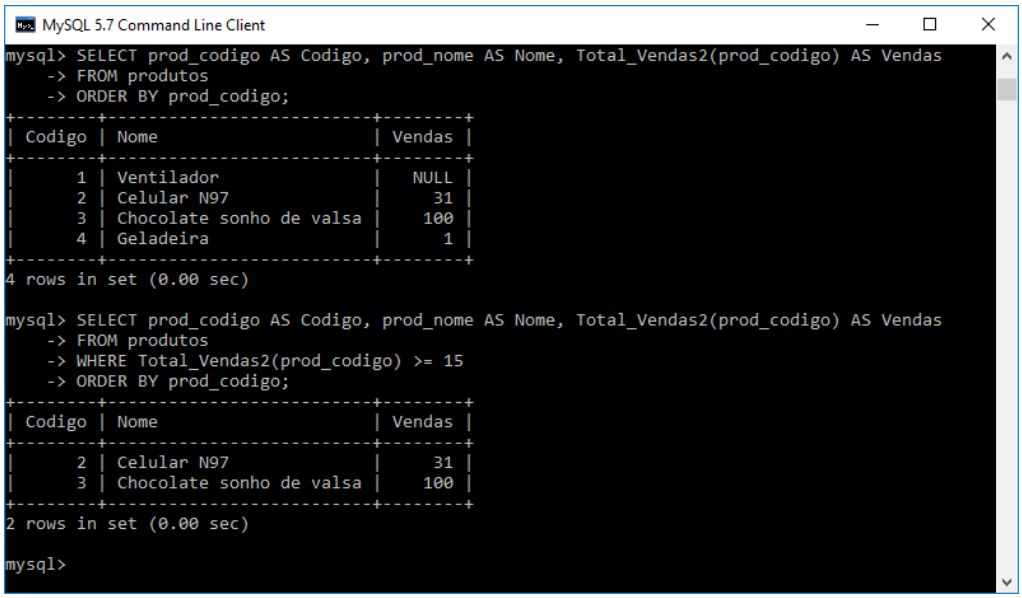
Embora já saibamos como criar uma função e obter o resultado da sua execução, ainda não mostramos como ela pode ser utilizada na manipulação de dados num SGBD. Para isso, vamos exemplificar a seguir, dois casos de uso da função Total_Vendas2() criada anteriormente.

```
mysql> SELECT prod_codigo AS Codigo, prod_nome AS Nome,
   Total_Vendas2(prod_codigo) AS Vendas
   FROM produtos
   ORDER BY prod_codigo;

mysql> SELECT prod_codigo AS Codigo, prod_nome AS Nome,
   Total_Vendas2(prod_codigo) AS Vendas
   FROM produtos
   WHERE Total_Vendas2(prod_codigo) >= 15
   ORDER BY prod_codigo;
```

No primeiro exemplo, a função Total_Vendas2() é usada numa consulta para gerar as linhas de uma coluna denominada Vendas, mostrando assim o total de vendas de cada produto associado a tabela **produtos**. No segundo exemplo, temos um caso similar, porém, a função também é utilizada na cláusula WHERE para filtrar os produtos com venda superior ou igual a 15 unidades. A **Figura 4** apresenta o resultado das consultas anteriores ao serem processadas no *MySQL*.

Figura 04 - Tela do *My*SQL após diversos comandos SELECT utilizando a função Total_Vendas2().

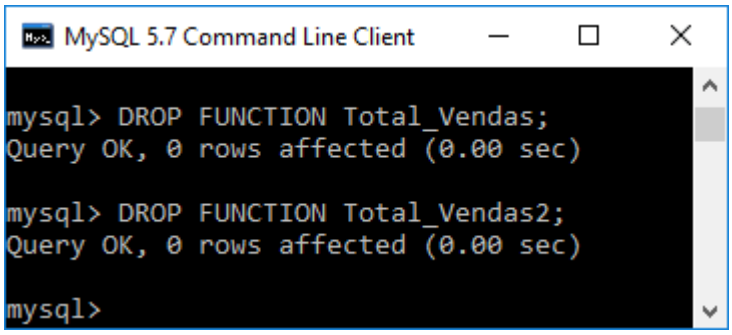


Fonte: *My*SQL 5.7 Command Line Client

Finalmente, para excluir uma função do banco de dados, você deve utilizar o comando DROP FUNCTION, o qual tem sua sintaxe descrita no destaque a seguir. E a resposta do SGBD a esse comando aplicado nas funções criadas anteriormente é ilustrada na **Figura 5**.

```
mysql>DROP FUNCTION  nome_da_funcao;
```

Figura 05 - Tela do MySQL após os comandos DROP FUNCTION.



Fonte: MySQL 5.7 Command Line Client

Vídeo 03 - Funções: Estrutura de Controle Condicional

Atividade 02

1. Vamos praticar um pouco para que você se familiarize com a utilização de funções escalares. Entre no banco de dados sispagamentos (Aula 14). Considerando as funções que você criou na atividade anterior, elabore o que se pede.

    a. Execute as funções e imprima seu resultado na tela.

    b. Utilize as funções para obter uma relação dos dados dos empregados que têm um desconto de INSS maior que R$ 50,00.

    c. Utilize as funções para obter uma relação dos dados dos empregados que têm um salário líquido (salário bruto menos descontos) menor que R$ 1000,00.

    d. Exclua do banco de dados as funções anteriores.

Conclusão

Vídeo 04 - Funções: Estrutura de Controle de Repetição.

Encerramos por aqui nossa aula sobre funções na linguagem SQL. Na próxima aula, aprenderemos os conceitos de segurança de sistemas e segurança de banco de dados e como podemos controlar o acesso aos nossos dados.

Faça a autoavaliação com atenção e veja se precisa parar e refletir mais sobre a criação e o uso de funções. Escreva no seu caderno todos os comandos SQL (e respectivas funções) aprendidos nesta aula, para não esquecer.

Bons estudos e boa sorte!

Resumo

Nesta aula, estudamos a criação e utilização das funções. Aprendemos a usar os comandos CREATE FUNCTION e DROP FUNCTION para criar e apagar uma função. Conhecemos o comando DECLARE para declaração de variáveis e vimos que também é possível declarar e inicializar uma variável usando o comando SET e o caractere “@”. Além disso, vimos o uso das funções no processamento de consultas.

Autoavaliação

1. Considere o banco de dados **CursoX**, criado na autoavaliação da Aula 9, cuja estrutura de tabelas é apresentada a seguir.

ATRIBUTO	TIPO	DESCRIÇÃO
aluno_cod	Número inteiro	Código do aluno
aluno_nome	Alfanumérico	Nome do aluno
aluno_endereco	Alfanumérico	Endereço do aluno
aluno_cidade	Alfanumérico	Cidade do aluno

Tabela: Alunos

ATRIBUTO	TIPO	DESCRIÇÃO
dis_cod	Número inteiro	Código da disciplina
dis_nome	Alfanumérico	Nome da disciplina

ATRIBUTO	TIPO	DESCRIÇÃO
dis_carga	Número inteiro	Carga horária da disciplina
dis_professor	Alfanumérico	Professor da disciplina

Tabela: Disciplina

ATRIBUTO	TIPO	DESCRIÇÃO
prof_cod	Número inteiro	Código do professor
prof_nome	Alfanumérico	Nome do professor
prof_endereco	Alfanumérico	Endereço do professor
prof_cidade	Alfanumérico	Cidade do professor

Tabela: Professores

a. Crie uma função que calcule a quantidade de professores e alunos que moram em uma determinada cidade.

b. Crie uma função que calcule a carga horária média de um determinado professor. Elabore uma consulta usando a função criada.

c. Crie uma função que receba o código do professor como parâmetro de entrada e retorne a cidade em que ele mora. Depois, elabore uma consulta para listar a quantidade de professores por cidade em que residem.

d. Crie uma função que retorne a disciplina de maior carga horária de um professor. Depois, use essa função para gerar uma tabela com o nome do professor e nome da disciplina de maior carga horária dele.

Referências

BEIGHLEY, L. **Use a cabeça SQL.** Rio de Janeiro: Editora AltaBooks, 2008.

MYSQL 5.7 Reference Manual. Disponível em: <http://dev.mysql.com/doc/refman/5.7/en/>. Acesso em: 28 jan. 2017.