Aula 6 - Comandos de SQL – Stored Procedure

Prof. Anésio Freire

Definição de Stored Procedure:

Stored Procedure, traduzido Procedimento armazenado, é uma biblioteca de comandos em SQL para utilização junto ao banco de dados.

Ela armazena tarefas repetitivas e aceita parâmetros de entrada para que a tarefa seja efetuada de acordo com a necessidade individual.

Uma Stored Procedure pode reduzir o tráfego na rede, melhorar a performance de um banco de dados, criar tarefas agendadas, diminuir riscos, criar rotinas de processamento, etc.

Por todas estas e outras funcionalidades é que as stored procedures são de extrema importância para os DBAs e desenvolvedores.

Existem 5 Procedimentos básicos que podemos criar:

- ***Procedimentos Locais*** - São criados a partir de um banco de dados do próprio usuário;
- ***Procedimentos Temporários*** - Existem dois tipos de procedimentos temporários: **Locais**, que devem começar com # e **Globais**, que devem começar com ##;
- ***Procedimentos de Sistema*** - Armazenados no banco de dados padrão do SQL Server (Master), podemos indentifica-los com as siglas sp, que se origina de stored procedure. Tais procedures executam as tarefas administrativas e podem ser executadas a partir de qualquer banco de dados.
- ***Procedimentos Remotos*** - Podemos usar Queries Distribuídas para tais procedures. São utilizadas apenas para compatibilidade.
- ***Procedimentos Estendidos*** - Diferente dos procedimentos já citados, este tipo de procedimento recebe a extensão .dll e são executadas fora do SGBD SQL Server. São identificadas com o prefixo xp.

Quando utilizar procedures

- Quando temos várias aplicações escritas em diferentes linguagens, ou rodam em plataformas diferentes, porém executam a mesma função.
- Quando damos prioridade à consistência e segurança.

Os bancos (Itaú, Bradesco, Real, etc), por exemplo, em geral, utilizam stored procedures para todas as operações em comum. Os procedimentos podem assegurar que as operações sejam registradas de forma correta e segura.

Por que é mais seguro?

Seguindo a linha de raciocínio dos bancos, utilizando stored procedures outras aplicações e usuários não conseguiriam nenhum tipo de acesso às tabelas do banco de dados de forma direta.

Eles poderiam apenas executar as stored procedures, que rodam ações específicas e determinadas pelos DBAs e desenvolvedores.

Aula 6 - Comandos de SQL – Stored Procedure

Prof. Anésio Freire

Criando uma Stored Procedure

Este é um exemplo de uma stored procedure que executa uma consulta utilizando um filtro por descrição, em uma tabela específica de nosso banco de dados.

```
USE BancoDados
GO
CREATE PROCEDURE Busca --- Declarando o nome da procedure
@CampoBusca VARCHAR (20) --- Declarando variável (note que utilizamos
o @ antes do nome da variável)
AS
SELECT Codigo, Descrição --- Consulta
FROM NomeTabela
WHERE Descricao = @CampoBusca --- Utilizando variável como filtro para
a consulta
```

Exemplo para o MYSQL:

```
 CREATE PROCEDURE Busca --- Declarando o nome da procedure
(CampoBusca VARCHAR (20)) --- Declarando variável (
AS
SELECT Codigo, Descrição --- Consulta
FROM NomeTabela
WHERE Descricao = CampoBusca --- Utilizando variável como filtro para
a consulta
```

Para executar uma procedure basta utilizar a cláusula EXECUTE seguido pelo nome da procedure e na frente o valor a ser utilizado como parâmetro.

```
EXECUTE Busca 'iMasters'
```

Para executar uma procedure, no MYSQL, basta utilizar a cláusula CALL seguido pelo nome da procedure e na frente o valor a ser utilizado como parâmetro entre parênteses..

```
CALL Busca ('iMasters')
```

Para deletar uma procedure é necessário utilizar a cláusula DROP PROCEDURE como no exemplo abaixo. No MYSQL é igual.

```
DROP PROCEDURE Busca
```

Exemplo com passagem de parâmetros

Podemos receber parâmetros, e utilizarmos eles em instruções SQL que serão executadas dentro da Stored Procedure:

Aula 6 - Comandos de SQL – Stored Procedure

Prof. Anésio Freire

```
CREATE PROCEDURE TESTE @PAR1 INT
AS
BEGIN
UPDATE TABELA1 SET CAMPO1 = 'NOVO_VALOR'
WHERE CAMPO2 = @PAR1
END
```

Percebam que no exemplo acima, não utilizamos parênteses, pois Stored Procedures são um pouco diferentes de funções.

Como uma Stored Procedure fica armazenada no banco de dados , ela já é pré-compilada e o SQL Server a executa mais rapidamente. Um exemplo de execução desta Stored Procedure, no Query Analyzer :

/* Chama a Stored Procedure TESTE passando 10 como primeiro parâmetro */

```
EXECUTE TESTE 10
```

Outra vantagem das Stored Procedures é que um programa chamador , seja ele uma página ASP ou um programa em VB, Delphi, Java, etc., só precisa chamar o nome da Stored Procedure, que pode conter diversos comandos Transact-SQL embutidos dentro dela, evitando assim um tráfego de rede maior, resultando em resposta mais rápida.

Uma Stored Procedure pode ainda retornar valores para a aplicação. Aqui temos um detalhe: o SQL Server permite o retorno de dados em forma de uma tabela após a execução ou um valor de retorno normal. Exemplo:

```
CREATE PROCEDURE TESTE @PAR1 INT
AS
BEGIN
SELECT @PAR1*@PAR1 AS QUADRADO
END
```

No exemplo acima a aplicação chamadora (cliente) pode capturar o retorno da Stored Procedure através do campo chamado QUADRADO, que contém somente um valor de retorno: o parâmetro elevado ao quadrado. Agora no próximo exemplo:

```
CREATE PROCEDURE TESTE @PAR1 INT
AS
BEGIN
SELECT CAMPO1 , CAMPO2 FROM TABELA1
WHERE CAMPO3 = @PAR1
END
```

Aula 6 - Comandos de SQL – Stored Procedure

Prof. Anésio Freire

Retorna para o cliente uma tabela contendo dois campos , CAMPO1 e CAMPO2, e podem ser capturados pela aplicação chamadora como se fosse uma tabela.

O uso de Stored Procedure é encorajado, mais deve-se utilizar este recurso com cuidado pois se utilizado em excesso o SQL Server pode ser sobrecarregado, mas ao mesmo tempo podemos obter um ganho de performance considerável, dependendo do caso. Minhas 'regrinhas' para o uso de Stored Procedures:

* Não faça Stored Procedures que somente fazem um Select ou Update ou Delete. Para isso envie a instrução diretamente.

* Use sempre transações, para poder 'voltar' os dados em caso de problemas

* Retorne somente o necessário, evitando tráfego na rede desnecessário.

* Use uma nomeclatura coerente para as Stored Procedures e as variáveis dentro dela

* SEMPRE endente seu código ao entrar em uma estrutura de bloco.

* Comente o máximo possível do seu código através do -- ou do /* e */

Utilizando Estruturas de Decisão

A linguagem que utilizamos para os comandos de um Stored Procedure é a linguagem T-SQL. A seguir vou apresentar as principais estruturas de controle disponíveis.

Este é o velho conhecido comando IF existente em qualquer linguagem:

```
IF teste_booleano
    Comandos_Se_Verdadeiro
Else
    Comandos_Se_Falso
```

Um Exemplo:

```
USE Northwind
GO
CREATE PROCEDURE sp_ExIF
  @pais1 nVarchar(15),
  @pais2 nVarchar(15),
AS
/* CRIA AS VARIAVEIS LOCAIS */
DECLARE @TotPed1 int, @TotPed2 int
DECLARE @mensagem Char(100)
/* DEFINE O VALOR DE CADA VARIÁVEL */
SET @TotPed1 = (SELECT Count(OrderID) FROM Orders WHERE
ShipCountry=@pais1)
SET @TotPed2 = (SELECT Count(OrderID) FROM Orders WHERE
ShipCountry=@pais2)
/* EXECUTO O TESTE, UTILIZANDO IF...ELSE */
IF (@TotPed1) > (@TotPed2)
BEGIN
   SET @mensagem = 'O Número de pedidos do primeiro país é maior'
END
ELSE
BEGIN
  IF (@TotPed1) < (@TotPed2)
  BEGIN
    SET @mensagem = 'O número de pedidos do segundo país é maior'
```

```
  END
  ELSE
  BEGIN
    SET @mensagem = 'O número de pedidos dos dois países é igual'
  END
END
PRINT(@mensagem)
```

Aula 6 - Comandos de SQL – Stored Procedure

Prof. Anésio Freire

A Estrutura WHILE...CONTINUE

Esta estrutura faz com que um conjunto de comandos continue sendo executado, enquanto uma determinada condição for verdadeira. A sintaxe para este comando é a seguinte:

```
WHILE Teste
BEGIN
Comando1
Comando2
Comando3
...
Comando4
END
```

Vamos a um exemplo:

```
USE Northwind
GO
CREATE PROCEDURE sp_CalculaSoma
@numero int
AS
/*CRIA AS VARIÁVEIS PARA UM CONTROLE DE LAÇO
INICIALIZA A VARIÁVEL COM O VALOR 1 */
DECLARE @soma int, @contador int
SET @soma=0
SET @contador=1
WHILE (@contador<=numero)
BEGIN
  SET @soma=@soma+@contador
  INSERT INTO SomaNaturais VALUES (@contador,@soma)
  SET @contador = @contador+1
END
```

Para executar esta Stored Procedure entre com o seguinte comando:

```
EXEC sp_CalculaSoma 10
```

Aula 6 - Comandos de SQL – Stored Procedure

Prof. Anésio Freire

Definição de Function:

Além de procedimentos, o SQL permite que armazenemos funções no servidor de banco de dados. A função pode receber e tratar diversos parâmetros da mesma maneira que o procedimento. A diferença entre um procedimento e uma função reside no fato de que a função sempre retorna um valor. Assim, para criar uma função, utilizamos o comando *Create Function*. Como a função retorna um valor, devemos especificar o tipo de dados retornado pela função. O exemplo de estrutura a seguir apresenta como a função é similar ao procedimento.

```
CREATE FUNCTION NOME_FUNÇÃO (
            nome_parm1 tipo_parm1,
            nome_parm2 tipo_parm2,
            ...)

            RETURN TIPO_RETORNO AS

BLOCO_DE_COMANDOS_SQL
```

Exemplos:

```
CREATE FUNCTION sf_Fat(@N Integer) Return int as
BEGIN
      DECLARE @Fator INT, @I INT
      SET @Fator = 1
      SET @I = 1
      IF (@N <= 1)
            RETURN (@Fator)
      ELSE
            WHILE (@I <= @N)
                  BEGIN
                        SET @Fator = @Fator * @I;
                        SET @I = @I + 1
                  END
            RERUTN (@Fator)
END
```

Em seguida execute a instrução:

```
Select sf_Fat (5)
```

No Mysql se cria função assim:

```
CREATE FUNCTION nome_função (parâmetros)
RETURNS tipo_dados_de retorno
BLOCO_DE_COMANDOS_SQL
```

Exemplo:

```
CREATE FUNCTION fn_teste (a DECIMAL(10,2), b INT)
RETURN INT
RETURN a*b;
```

Para executar a função:

```
SELECT fn_teste(2.5, 4) AS Resultado;
```