Procedimentos armazenados

Prof. Márcio Bueno
{bd2tarde,bd2noited}@marciobueno.com

Material da Profª Maria Lígia B. Perkusich

Procedimentos armazenados (i)

- Definição
  - Um procedimento armazenado *(stored procedure)* é uma coleção nomeada de comandos SQL que é armazenada no servidor (compiladas), como um procedimento/ função, para eventuais processamentos.

Procedimentos armazenados (ii)

- Considere a consulta:

```
SELECT codigop, nome FROM Projeto p, Alocacao a
WHERE p.codproj = a.codigop
```

- Podemos criar um procedimento armazenado para a mesma, como:

```
CREATE PROCEDURE DBO.AlocadoProjeto  AS
 SELECT codigop, nome FROM Projeto p JOIN
Alocacao a  ON p.codproj = a.codigop
```

- Para executar

```
EXEC AlocadoProjeto;
```

Procedimentos armazenados (iii)

- É um método de encapsular tarefas repetitivas que são executadas mais eficientemente
  - Pré-compiladas
- Procedimento armazenado suporta
  - Variáveis definidas pelo usuário;
  - Execução condicional;
  - Chamadas a outros procedimentos armazenados;
  - Parâmetros de entrada e saída;
  - e outras características poderosas de programação.

Criando um procedimento armazenado (i)

- Quando um procedimento armazenado é criado, seus comandos são verificados sintaticamente. Se estiverem corretos, o procedimento é criado.
  - No SQL Server o nome e o código do procedimento são armazenados nas tabelas do sistema ***sysobjects*** e ***syscomments***, respectivamente.
    - Sp_helptext AlocadoProjeto
  - Um erro é retornado se algum erro de sintaxe for detectado, e o procedimento não é criado.

Criando um procedimento armazenado (ii)

- Um *processo de resolução tardia* permite que os procedimentos referenciem objetos que não existam quando o procedimento é criado.
  - Este processo permite flexibilidade porque procedimentos armazenados e outros objetos que ele referencia não precisa ser criado em uma ordem em particular.
- Os objetos precisam existir quando o procedimento for executado.

Criando um procedimento armazenado (iii)

- Quando o procedimento é executado pela primeira vez, ou se ele for recompilado, o processador de consulta ler o procedimento armazenado em um processo chamado resolução.
- Em seguida o otimizador de consultas analisa as declarações no procedimento e cria um plano que contém o método mais rápido para acessar os dados → memória
  - Quantidade de dados nas tabelas, índices, junções, etc.

Compilação

- É o processo de analisar o procedimento e criar um plano de execução que será colocado na memória - tabela *sysprocedures*
- A memória contém os planos de execução mais valiosos.
- Fatos que incrementam o valor de um plano incluem:
  - Tempo necessário para compilar
  - Uso freqüente
- O espaço necessário para armazenar o plano decrementa seu valor

Vantagens fundamentais

- Quando o procedimento é executado pela primeira vez, o plano de consulta é lido e completamente otimizado na forma de um *plano de procedimento* e então executado
  - Isso evita a analise sintática, resolução a cada execução da consulta
- Depois de ser executado pela primeira vez, o plano de um procedimento armazenado é armazenado na memória
  - Da próxima vez que se utilizar esse procedimento na mesma sessão, ele será lido da memória e executado

Vantagens (i)

- Desempenho
  - Reduz o tráfego na rede.
    - Ao invés dos usuários enviarem centenas de comandos SQL pela rede, os usuários podem executar operações complexas enviando apenas um comando, o que reduz a quantidade de requisições entre os clientes e os servidor.
  - Exemplo:
    - EXEC AlocadoProjeto

Vantagens (ii)

- Fácil Gerenciamento
  - Procedimentos armazenados podem encapsular as regras do negócio.
    - Tais regras, se mudarem, poderão ser trocadas em um único lugar.
  - Todos os usuários podem usar o mesmo procedimento para assegurar consistência ao acesso a dados e modificação.

Vantagens (iii)

- Segurança
  - Os usuários não precisam saber detalhes da implementação, ou das tabelas que serão usadas.
  - Se um conjunto de procedimentos armazenados suporta todas as funcionalidades que os usuários precisam, esses usuários nunca precisarão acessar as tabelas diretamente.
  - Usuários podem ter acesso aos procedimentos armazenados, mesmo que não tenham acesso direto as tabelas ou visões destas.

Sintaxe

CREATE PROC[EDURE] nome_do_proc

[{ @parâmetro tipo_de_dado}
[=default|NULL] [OUTPUT]] [,....n]

[WITH {RECOMPILE | ENCRYPTION |
{RECOMPILE, ENCRYPTION}]

AS instrução_de_SQL [...n]

Exemplos – sem parâmetros

- Os comandos abaixo criam um procedimento armazenado que lista todos os livros que estão em atraso em um banco de dados de uma biblioteca.
- Criando o procedimento

```
USE  biblioteca
CREATE PROC dbo.livros_atraso [WITH RECOMPILE]
AS
  SELECT * FROM dbo.emprestimo
  WHERE data_limite < GETDATE( )
```

- Executando o procedimento

```
USE  biblioteca
EXEC livros_atraso
```

Exemplos - com parâmetros de entrada (i)

- Criando o procedimento:

```
CREATE PROC dbo.empregados
@nome varchar(50) = 'Pessoal'
AS
SELECT e.matricula, e.nome, e.endereco,
e.salario
FROM Empregados e JOIN Departamento d
ON e.cod_depto = d.codigo
WHERE d.nome = @nome
```

- Uma chamada a este procedimento seria:

```
EXEC empregados 'Informatica'
```

Exemplos - com parâmetros de entrada (ii)

- Criando o procedimento:

```
CREATE PROCEDURE InsereEmpregado
@mat int, @nome varchar(30), @endereco
 varchar(60), @salario float, @depto int
AS INSERT INTO Empregado
    VALUES (@mat, @nomeE, @endereco, @salario,
    @depto)
```

- Executando o procedimento

```
EXEC InsereEmpregado  ‘324’, ‘Maria’, ‘Rua das
 Rosas, 100’, ‘R$ 1000,00’, ‘10’
```

Exemplo - com parâmetros de entrada (iii)

```
CREATE PROCEDURE find_isbn
   @titulo string = null,
   @ traducao char(8) = 'portugues'   AS
         IF @titulo is null
          BEGIN
           PRINT "Entre com o título"
           RETURN
         END
SELECT isbn, titulo, traducao
FROM item I JOIN titulo t ON t.titulo_no = I.titulo_no
WHERE t.titulo LIKE @titulo AND I.traducao LIKE @traducao
```

Exemplos - com parâmetros de saída (i)

- Criando o procedimento:

```
CREATE PROCEDURE  DBO.Soma
@x int, @y int,
@resp int OUTPUT
AS
  SET @resp = @x + @y
```

Executando o procedimento

```
DECLARE @resposta int
EXEC Soma 100, 400, @resposta OUTPUT
(ou EXEC Soma y=400, x=100, resp=@resposta OUTPUT)
Select 'Resposta = ', @resposta
```

Exemplos - com parâmetros de saída (ii)

- Criando o procedimento:

```
CREATE PROCEDURE  DBO.Multiplica
@x int, @y int,
@resp int OUTPUT
AS
    SET @resp = @x * @y
```

- Executando o procedimento

```
DECLARE @resposta int
EXEC Multiplica 2, 3, @resposta OUTPUT
Select 'A resposta é: ', @resposta
```

Executando procedimentos (i)

- Executado um procedimento armazenado por ele mesmo
  - Exec livros_atraso

- Com recompilação
  - EXEC pauthors
    WITH RECOMPILE

Executando procedimentos (ii)

- Executado um procedimento armazenado dentro de uma declaração INSERT
  - INSERT INTO clientes
    EXEC empregado_cliente
- Criando o procedimento

```
CREATE PROC empregado_cliente AS
  SELECT lastname, firstname, address, city
  FROM empregados
  WHERE data_contratacao = GETDATE()
```

- O seguinte declaração executa o procedimento

```
INSERT into clientes
  EXEC empregado_cliente
```

Forçando a recompilação

- Você pode forçar todos os procedimentos armazenados e gatilhos que fazem referência a uma tabela particular serem recompiladas em seu próximo tempo de execução executando o procedimento armazenado ***sp_recompile***:
  - EXEC sp_recompile autores

Removendo procedimentos armazenados

USE Empresa

DROP PROC Empregados