Stored Procedures

Stored Procedures

- **Definição**
  - Conjunto de comandos SQL que são compilados e armazenados no servidor
- **Características**
  - Podem ser armazenados no banco de dados e acionados por qualquer programa aplicativo que tenha autorização para execução

Stored Procedures

- **Características**
  - Uma **Stored Procedure (sp)** é executada no lado do servidor e seu plano de execução fica na memória, agilizando as próximas chamadas
  - Podem receber um ou mais parâmetros formais
    - Pode retornar diversos valores como **parâmetro de saída (output)**

Stored Procedures

- **Classificação das Stored Procedures**
  - **Stored Procedure:** definidas pelo usuário
  - **System Stored Procedure:** disponibilizadas pelo sistema de banco de dados para execução de tarefas administrativas
    - Podem ser executadas em qualquer banco
    - Estão armazenadas no banco de dados **master**
    - Todas as stored procedures iniciam por "sp_"

Stored Procedures

- **System Stored Procedure:**
  - Exemplos de System Stored Procedures:
    - **sp_helpdb**: exibe informações sobre um BD específico
    - **sp_addtype**: define novos tipos de dados
    - **sp_droplogin**: remove um usuário do SQL Server

Stored Procedures

- **Vantagens: Desempenho**

- Ex.: Seja a consulta

```
SELECT codigop, nome, COUNT(*)
FROM Projeto p, Alocacao a
WHERE p.codproj = a.codigop
GROUP BY p.codproj, p.nome
```

- Se vários usuários realizarem esta consulta o tráfego de rede será alto.

- No entanto, se criarmos uma *stored procedure* para executar esta consulta, os usuário necessitaram apenas de um comando para executar a consulta anterior:

EXEC nomeProcedimento;

Stored Procedures

- Manutenção
  - Facilita o gerenciamento
  - Encapsulam rotinas de uso freqüente no próprio servidor, estando disponível para várias aplicações
  - Parte da lógica do sistema pode ser armazenada no próprio BD, em vez de ser codificada em várias aplicações

Stored Procedures

- **Sintaxe**

  CREATE PROCEDURE <nome> [parâmetro]

  AS <instrução SQL>

- **Regras**
  - O **nome** da *procedure* deve seguir as regras para criação de identificadores
  - Nome do parâmetro deve iniciar por @ e deve ser único na lista de argumentos, seguido do seu tipo
    - @mes int, @ano int, @nome varchar
  - Todos os parâmetros são considerados de entrada, exceto se houver OUTPUT após sua definição

Stored Procedures

- Para ser executada, é necessário fazer uma chamada ao comando EXECUTE

  EXEC[UTE] <nome da sp> [valor do parâmetro]

Stored Procedures

```
CREATE PROCEDURE MostraEmpregadosDep
  @nomeDep varchar(50) = 'Pessoal'
AS
    SELECT e.mat, e.nome, e.endereco, e.salario
    FROM Empregados e, Departamento d
    WHERE e.codD = d.codD and
              d.nomeD = @nomeDep
```

- Uma chamada a este procedimento seria:

  EXEC MostraEmpregadosDep 'Informatica'

Stored Procedures

- **Exemplo:** Criar uma procedure que exiba o título e a editora de cada livro.

```
CREATE PROCEDURE listar
AS
    SELECT l.titulo, e.nome
    FROM livro l, editora e
    WHERE l.codEdit = e.codEdit

EXEC listar
```

Stored Procedures

- **Exemplo:** Criar uma *procedure* que exiba o total de salários pagos a um determinado setor da empresa.

```
CREATE PROCEDURE sptotalSal
@setor char(3)
AS
        SELECT SUM(salario)
        FROM funcionario
        WHERE setor = @setor

EXEC sptotalSal 'INF'
```

Stored Procedures - Exemplos

- Modelo Relacional Exemplo: SCP
- Crie Stored Procedures:
  - SP1 tem como parâmetro de entrada o código do funcionário, e mostra todas as datas dos pedidos que foram feitos por este funcionário.
  - SP2 mostra todos os pedidos realizados no dia 23/08/2007

Stored Procedures - Exemplos

- SP3 tem como parâmetros de entrada o código do cliente e o código da cidade que o cliente mora, e mostra os nomes dos produtos comprados por este cliente, e o estado que o cliente mora.
- SP4 tem como parâmetro de entrada a sigla do setor, e mostra quantos funcionários existe neste setor, e uma soma do salários pagos aos funcionários.

Stored Procedures com parâmetros de saída

- Obtenha a quantidade de mensagens que o usuário "aluno1" recebeu.

```
CREATE PROCEDURE getNumMens
@usuario char(8),
@quant smallint OUTPUT
AS
    SELECT @quant=COUNT(*)
```

Criação da procedure getNumMens com 1 parâmetro de entrada (@usuario) e 1 parâmetro de saída (@quant).
O valor retornado pelo SELECT é armazenado na variável @quant

Stored Procedures com parâmetros de saída

- Chamada:

1. USE javamail

O banco javamail é selecionado

2. DECLARE @quant smallint

Em seguida, é declarada a variável de saída @quant.

3. EXECUTE getNumMens 'aluno1', @quant OUTPUT

Na execução da procedure, 'aluno1' é o parâmetro de entrada e @quant recebe o valor que virá de dentro da procedure

4. SELECT @quant quantidade

Por fim, o comando SELECT se encarrega de exibir o resultado que a variável @quant recebeu

Stored Procedures com parâmetros de saída

1. USE javamail
2. DECLARE @quant smallint
3. EXECUTE getNumMens 'aluno1',
@quant OUTPUT
4. SELECT @quant quantidade

Stored Procedures com parâmetros de saída

- Criar uma procedure que receba o câmbio do dolar, a quantidade de reais, e retorne o valor convertido em dolares.

```
CREATE PROCEDURE spCambio
                @dolar decimal(4,2),
                @reais decimal(7,2),
                @final decimal(7,2) output
                AS
                        SELECT @final = @reais/@dolar
```

Stored Procedures com parâmetros de saída

- Executando a procedure...

```
DECLARE @res decimal(7,2)
EXEC spCambio 2.33, 2500,
        @res output
SELECT @res Conversao
```