доц. д-р Цветанка Георгиева-Трифонова

СЪХРАНЕНИ ПРОЦЕДУРИ

СЪДЪРЖАНИЕ

- Роля, създаване и извикване на съхранени процедури
- Създаване на съхранени процедури с параметри
- Използване на резултатите, върнати от изходните параметри

- *Съхранена процедура*
  - последователност от SQL конструкции, съхранени в база от данни на SQL Server;
- Предимства
  - предоставя по-добра производителност;
  - осигурява изолация от промени в критериите и логиката на системата.

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА

```
CREATE PROCEDURE stored_procedure_name
[ { @parameter_name_IN datatype [=default_value]
    | @parameter_name_OUT datatype OUTPUT } [, …] ]
 AS
    sql_queries
```

- Изпълняване на съхранена процедура

```
{EXECUTE | EXEC}
[@return_value =] stored_procedure_name
[ { [@parameter_name_IN =] parameter_value
  | [@parameter_name_OUT =]
    @received_var_name OUTPUT }[, …] ]
```

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА – ПРИМЕР

```
CREATE PROCEDURE SalesByEmployee
AS
    SELECT FirstName, LastName,
          ( SELECT COUNT(*)FROM Sales s
            WHERE s.EmployeeID = e.EmployeeID )
           AS TotalSales
    FROM Employees e
    ORDER BY FirstName, LastName
```

- Изпълняване

```
EXECUTE SalesByEmployee
```

или

```
EXEC SalesByEmployee
```

	FirstName	LastName	TotalSales
1	Атанас	Лазаров	0
2	Ваня	Хростова	1
3	Георги	Хростов	0
4	Катя	Цветанова	0
5	Стела	Миланова	1
6	Стоян	Георгиев	2

ПРОМЕНЯНЕ И ИЗТРИВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА

- Променяне

```
ALTER PROCEDURE stored_procedure_name
[ { @parameter_name_IN datatype [=default_value]
    | @parameter_name_OUT datatype OUTPUT } [, …] ]
  AS
    sql_queries
```

- Изтриване

```
DROP PROCEDURE stored_procedure_name
```

ИЗПОЛЗВАНЕ НА ВЪРНАТИТЕ СТОЙНОСТИ

```
CREATE PROCEDURE CurrentEmployees
 AS
    SELECT EmployeeID,
           FirstName, Surname, LastName,
           Title, HireDate, TerminationDate,
           ManagerEmpID, StoreName
    FROM Employees e
    INNER JOIN Stores s ON e.StoreID=s.StoreID
    WHERE TerminationDate IS NULL

    RETURN @@ROWCOUNT
```

ИЗПОЛЗВАНЕ НА ВЪРНАТИТЕ СТОЙНОСТИ (2)

- За получаване на стойността, която връща дадена съхранена процедура, се използва EXEC конструкция от вида:

```
EXEC @some_var_name = stored_procedure_name
```

- Пример

```
DECLARE @count int
EXEC @count = CurrentEmployees
PRINT @count -- неявно се преобразува в низ
```

	EmployeeID	FirstName	Surname	LastName	Title	HireDate	TerminationDate	ManagerEmpID	StoreName
1	1	Стоян	Маринов	Георгиев	мениджър	2001-10-01 00:00:00.000	NULL	NULL	Искър
2	3	Ваня	Иванова	Хростова	продавач	2000-10-08 00:00:00.000	NULL	2	Искър
3	4	Стела	Георгиева	Миланова	продавач	2008-10-08 00:00:00.000	NULL	2	Глория
4	6	Катя	Петрова	Цветанова	продавач	2000-05-25 00:00:00.000	NULL	5	Глория

Results Messages

```
(4 row(s) affected)
4
```

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА С ПАРАМЕТРИ

```
CREATE PROCEDURE stored_procedure_name
  @parameter_name_IN datatype,
        -- входен параметър
  @parameter_name_OUT datatype OUTPUT
        -- изходен параметър
  AS
      sql_queries
```

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА С ПАРАМЕТРИ – ПРИМЕР

- Съхранена процедура с един входен параметър:

```
CREATE PROCEDURE SalesForEmployee
    @EmployeeID  int
 AS
    SELECT LastName, FirstName,
          (SELECT SUM(TotalForSale*(1-Discount))
           FROM Sales s
           WHERE e.EmployeeID = s.EmployeeID )
         AS TotalSales
    FROM Employees e
    WHERE e.EmployeeID = @EmployeeID
```

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА С ПАРАМЕТРИ – ПРИМЕР (2)

- Съхранена процедура с един входен параметър:

```
CREATE PROCEDURE SalesForEmployee
    @EmployeeID  int
 AS
    SELECT LastName, FirstN
          (SELECT SUM(TotalP
           FROM Sales s
           WHERE e.Employeel
          AS TotalSales
    FROM Employees e
    WHERE e.EmployeeID = @EmployeeID
```

	LastName	FirstName	TotalSales
1	Георгиев	Стоян	44.450000
2	Хростов	Георги	NULL
3	Хростова	Ваня	1.800000
4	Миланова	Стела	10.100000
5	Пазаров	Атанас	NULL
6	Цветанова	Катя	NULL

ИЗПЪЛНЯВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА С ВХОДНИ ПАРАМЕТРИ

```
EXEC [@some_var_name =] stored_procedure_name
     [@parameter_name_IN1 =] parameter_value1
  [,…]
```

- Пример

```
EXEC SalesForEmployee 1
```

ИЛИ

```
EXEC SalesForEmployee @EmployeeID = 1
```

	LastName	FirstName	TotalSales
1	Георгиев	Стоян	44.450000

ПОДРАЗБИРАЩИ СЕ СТОЙНОСТИ НА ВХОДНИТЕ ПАРАМЕТРИ

```
CREATE PROCEDURE stored_procedure_name
  @parameter_name_IN datatype = default_value [,…]
  AS
       sql_queries
```

- Изпълняване на процедурата може да се извърши чрез:

```
EXEC stored_procedure_name
```

или

```
EXEC stored_procedure_name DEFAULT
```

ПОДРАЗБИРАЩИ СЕ СТОЙНОСТИ НА ВХОДНИТЕ ПАРАМЕТРИ – ПРИМЕР

```
ALTER PROCEDURE SalesForEmployee
   @EmployeeID int = 1
 AS
   SELECT LastName, FirstName,
         (SELECT SUM(TotalForSale*(1-Discount))
          FROM Sales s
          WHERE e.EmployeeID = s.EmployeeID )
         AS TotalSales
   FROM Employees e
   WHERE e.EmployeeID = @EmployeeID
```

- Изпълняване на процедурата

```
EXEC SalesForEmployee
```

или

```
EXEC SalesForEmployee DEFAULT
```

ПОДРАЗБИРАЩИ СЕ СТОЙНОСТИ НА ВХОДНИТЕ ПАРАМЕТРИ – ПРИМЕР (2)

```
CREATE PROCEDURE GetSuppliersRaw
      @CompanyName varchar(50) = NULL
AS
   IF @companyName IS NOT NULL
        SELECT * FROM Suppliers
        WHERE CompanyName LIKE @CompanyName + '%'
   ELSE
        SELECT * FROM Suppliers

   RETURN @@ROWCOUNT
```

- Изпълняване на процедурата

```
EXEC GetSuppliersRaw 'A'
```

или

```
DECLARE @count int
EXEC @count = GetSuppliersRaw 'Ал'
RAISERROR('%d реда са избрани.', 11, 1, @count)
```

ПОДРАЗБИРАЩИ СЕ СТОЙНОСТИ НА ВХОДНИТЕ ПАРАМЕТРИ – ПРИМЕР (3)

```
CREATE PROCEDURE GetSuppliersRaw
     @CompanyName varchar(50) = NULL
AS
   IF @companyName IS NOT NULL
        SELECT * FROM Suppliers
        WHERE CompanyName LIKE @CompanyName + '%'
   ELSE
        SELECT * FROM Suppliers

   RETURN @@ROWCOUNT
```

	SupplierID	CompanyName	ContactName	ContactTitle	Address	City	PostalCode	PhoneNumber	FaxNumber	EmailAddress	WebAddress
1	2	АЛЕМИТ ООД	Марин Христов	управител	бул. Хр. Ботев, 5	Велико Търново	5000	676654564	3434343	marin@gmail.com	www.alemit.bg
2	3	АЛПИ ООД	Милка Маринова	управител	бул. България, 2	Пясковец	5200	3432545	878754	milka@gmail.com	www.alpi.bg

Results | Messages

```
(2 row(s) affected)
Msg 50000, Level 11, State 1, Line 3
2 реда са избрани.
```

```
liersRaw 'Ал'
избрани.', 11, 1, @count)
```

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА С ПАРАМЕТРИ – ПРИМЕР

```
CREATE PROCEDURE get_s
    @SaleID int
 AS
  SELECT SaleID,
    CONVERT(char(10), SaleDate, 104) AS Sale_date,
    EmployeeID, CustomerID
  FROM Sales
  WHERE SaleID = @SaleID
```

	SaleID	Sale_date	EmployeeID	CustomerID
1	1	30.01.2010	1	1
2	2	30.01.2010	1	3
3	3	31.01.2010	3	2
4	4	31.01.2010	4	2

- Изпълняване на процедурата

```
EXEC get_sale 1
```

	SaleID	Sale_date	EmployeeID	CustomerID
1	1	30.01.2010	1	1

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА С ПАРАМЕТРИ – ПРИМЕР (2)

```
CREATE PROCEDURE get_sale_details
    @saleID int
  AS
   SELECT SaleID, sd.ProductID, ProductName,
      Quantity, sd.Price,
      sd.Quantity*sd.Price*(1-sd.Discount) AS Cost
   FROM SaleDetails sd
   INNER JOIN Products p
      ON sd.ProductID = p.ProductID
   WHERE SaleID = @SaleID
```

- Изпълняване на процедурата

```
EXEC get_sale_details 1
```

СЪЗДАВАНЕ НА СЪХРАНЕНА ПРОЦЕДУРА С ПАРАМЕТРИ – ПРИМЕР (3)

```
CREATE PROCEDURE get_sale_details
    @saleID int
AS
  SELECT SaleID
     Quantity,
     sd.Quantity
  FROM SaleDeta
  INNER JOIN Pr
     ON sd.Produc
  WHERE SaleID
```

	SaleID	ProductID	ProductName	Quantity	Price	Cost
1	1	1	ябълки	2.000	3.00	6.000000000
2	1	3	домати	1.000	2.75	2.750000000
3	2	2	портокали	5.000	4.00	18.000000000
4	2	4	картофи	2.000	3.50	7.000000000
5	2	9	шоколад	10.000	2.60	26.000000000
6	3	5	чипс	1.000	1.80	1.800000000
7	4	7	макарони	2.000	2.80	5.600000000
8	4	8	еклери	3.000	1.50	4.500000000

	SaleID	ProductID	ProductName	Quantity	Price	Cost
1	1	1	ябълки	2.000	3.00	6.000000000
2	1	3	домати	1.000	2.75	2.750000000

- Изпълняване на процедурата

```
EXEC get_sale_details 1
```

ИЗПОЛЗВАНЕ НА РЕЗУЛТАТИТЕ, ВЪРНАТИ ОТ ИЗХОДНИТЕ ПАРАМЕТРИ

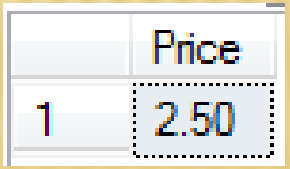
```
EXEC [@return_value =] stored_procedure_name
 [@parameter_name_OUT =] @received_var_name OUTPUT
```

- Пример

```
CREATE PROCEDURE Get_Product_price
  @ProductID int, @price money OUTPUT
 AS
  SELECT @price = price
  FROM Products
  WHERE ProductID = @ProductID
```

- Изпълняване на процедурата

```
DECLARE @price money
EXEC Get_Product_price 1, @price OUTPUT
SELECT @price AS Price
```



	Price
1	2.50

ИЗПОЛЗВАНЕ НА РЕЗУЛТАТИТЕ, ВЪРНАТИ ОТ ИЗХОДНИТЕ ПАРАМЕТРИ – ПРИМЕР

```
CREATE PROCEDURE spInsertSale
    @CustomerID int,
    @EmployeeID int,
    @SaleDate datetime = NULL,
    @SaleID int OUTPUT
AS
    DECLARE @e int

    INSERT INTO Sales
            (CustomerID, EmployeeID, SaleDate)
    VALUES (@CustomerID, @EmployeeID, @SaleDate)

    SELECT @e = @@ERROR, @SaleID = @@IDENTITY
    RETURN @e
```

- Изпълняване на процедурата

```
DECLARE @e int, @Ident int

BEGIN TRANSACTION

EXEC @e = spInsertSale @CustomerID = 1,
     @EmployeeID = 2, @SaleDate = '2009/10/06',
     @SaleID = @Ident OUTPUT
IF @e <> 0
   BEGIN
      ROLLBACK TRANSACTION
      RAISERROR('Грешката е %d.', 16, 1, @e)
   END
ELSE
  BEGIN
    INSERT INTO SaleDetails
           (SaleID, ProductID, Price, Quantity)
    VALUES (@Ident, 87, 10, 20)
    COMMIT TRANSACTION
  END
```

ЗАДАЧИ

- *1*. Да се създаде съхранена процедура, извеждаща:
  - ***1.1.*** клиентите, за които колоната `CustomerID` започва с дадена последователност от цифри, предадена като параметър или да изведе всички клиенти, ако не се предаде стойност;
  - ***1.2.*** продажбите, за които колоната `CustomerID` започва с дадена последователност от цифри, предадена като параметър или да изведе всички продажби, ако не се предаде стойност;
  - ***1.3.*** доставната цена на продукт с идентификатор, предаден като параметър със стойност по подразбиране `NULL` или да изведе средната аритметична стойност на доставните цени на всички продукти, ако не се предаде стойност.

ЗАДАЧИ

- *2.* Да се създаде съхранена процедура, чрез която се извежда различна информация от таблицата за продажбите в зависимост от деня от седмицата, в който е поискана. От вторник до петък показва продажбата на най-голяма стойност и сумата от продажбите за предишния ден. От събота до понеделник показва продажбата на най-голяма стойност и сумата от продажбите за изминалата седмица.
- *3.* Да се създаде съхранена процедура, изтриваща всички доставчици, за които колоната `CompanyName` започва с даден символен низ, предаден като параметър. Ако изтриването на редове генерира грешка, върнатата от процедурата стойност да е –1; ако се предаде празен низ или `NULL`, върнатата от процедурата стойност да е –2; в противен случай – броя на изтритите редове. Да се напише код, който извиква процедурата и извежда съответното съобщение за липсващ префикс; за наличие на грешка; за броя на изтритите редове.

ЗАДАЧИ

- *4*. Да се създаде съхранена процедура, изчисляваща отстъпката на дадена по идентификатора си продажба, т.е. изчисляваща и актуализираща стойността на колоната `Discount` в таблицата за продажбите `Sales` в зависимост от стойностите на `TotalForSale` и `CurrentBalance`: за обща стойност на продажбата над 100 и текущ баланс на клиента над 1000 да се зададе отстъпка 15%; за обща стойност на продажбата над 100 и текущ баланс на клиента под 1000 – 10%; за обща стойност на продажбата под 100 и текущ баланс на клиента над 1000 – 12%; за обща стойност на продажбата под 100 и текущ баланс на клиента под 1000 – 0%.

ЗАДАЧИ

- 5. Да се създаде съхранена процедура, извеждаща служителите, назначени след дадена дата, предадена като параметър или да изведе всички служители, ако не се предаде стойност. Върнатата от процедурата стойност да е броя на редовете, които се извличат. Да се напише код, който извиква процедурата и извежда съобщение за броя на извлечените редове.
- 6. Да се създаде съхранена процедура, която увеличава наличното количество на продукт с идентификатор, предаден като параметър или да изведе съобщение, ако не се предаде стойност. Ако продукт със зададения идентификатор не съществува или ако актуализирането генерира грешка, да се изведе съответното съобщение; ако се изпълни успешно, да се изведе съобщение, което потвърждава извършената промяна и да се извлече новата стойност на колоната `Stock` за съответния продукт.

ЗАДАЧИ

7. Да се създаде съхранена процедура, въвеждаща нов ред в таблицата `Sales`, с входни параметри за клиент, служител и дата, изходен параметър за идентификатор на продажбата. Ако датата на продажба за реда, който трябва да бъде добавен, е преди повече от една седмица, конструкцията за добавяне на нов ред не се изпълнява, извежда се дефинирано от потребителя съобщение за грешка, а върнатата от процедурата стойност е номера на грешката. В противен случай конструкцията за добавяне на нов ред се изпълнява, върнатата от процедурата стойност е номера на грешката, генерирана след въвеждането на продажбата. Да се напише код, който извиква процедурата, така че ако е получена грешка, извежда съобщение с номера на съответната грешка; в противен случай извежда добавения ред в таблицата `Sales`.

ЗАДАЧИ

- *8.* Да се създаде съхранена процедура, която чрез изходен параметър връща наличното количество на продукт с идентификатор, предаден като входен параметър. Да се напише код, който изпълнява съхранената процедура и извежда съобщение, ако не съществува продукт със зададения идентификатор; в противен случай извежда получената от изходния параметър стойност.