Stored Procedures

for Database Servers

Alexey Tukalo,
EFA12SF,
Information Technology,
Savonia University of Applied Sciences

November 5, 2014

1 Print out all the employees without any project

The procedure is very easy, it is possible to implement it via single T-SQL query and put the query into view, but in an according with the task I have to execute the query form procedure.

```
-- ============================================
-- Author:      Alexey Tukalo
-- Create date: 3.11.2014
-- Description: print out all the emploees with out any project
-- ============================================
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE emploeesWithOutProject
AS
BEGIN
SELECT * FROM employee WHERE NOT EXISTS (SELECT * FROM works_no WHERE employee.emp_no=works_no.emp_no)
END;
GO
```

Figure 1: The first procedure source code

The code should display employees with out any project. The employees and projects tables are shown on the picture 2.

emp_no	emp_fname	emp_lname	dept_no
1	Daria	Doncova	3
2	Sergey	Lukanov	5
3	Artur	Doile	7
4	Leo	Tolstoy	4
5	Josev	Stalin	9

emp_no	project_no	job	enter_date
2	4	3	2012-07-21
5	3	5	2014-06-03
1	2	4	2013-05-05

Figure 2: The employees and projects tables

And the output of the procedure is below.

Results Messages

	emp_no	emp_fname	emp_lname	dept_no
1	3	Artur	Doile	7
2	4	Leo	Tolstoy	4

Figure 3: The employees without any project

2 Increase all the project budgets with a %-value given as a parameter

After that I have made the second procedure the source code you can see at the pic. 4.

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================

CREATE PROCEDURE budgetIncreas (@percent int) AS
DECLARE @number INT
DECLARE projectSet CURSOR FOR SELECT project_no FROM project;
BEGIN
 OPEN projectSet;
 FETCH NEXT FROM projectSet INTO @number
 WHILE @@FETCH_STATUS=0
 BEGIN
 UPDATE project SET budget=(100+@percent) * (SELECT budget FROM project WHERE project_no=@number)/100 WHERE project_no=@number
 FETCH NEXT FROM projectSet INTO @number
END
CLOSE projectSet
END
GO

Messages
Command(s) completed successfully.
```

Figure 4: budgetIncreas() source code

The budgets before the execution was at figure 5.

project_no	project_name	budget
2	3	50
3	6	30
5	1	15

Figure 5: budgets before the execution

I have made the execution via GUI tools.

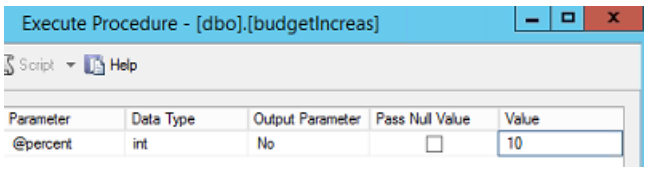

Figure 6: Execution settings

And finally I got the new budgets.

project_no	project_name	budget
2	3	55
3	6	33
5	1	16,5

Figure 7: Execution settings

3 Archive orders

I needed to write a procedure to move all old orders to special table for an archive, the procedure is shown below.

```
USE [test_db]
GO
/****** Object:  StoredProcedure [dbo].[oldOrders]    Script Date: 5.11.2014 11:33:59 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[oldOrders]  AS
DECLARE @orderid INT
DECLARE @customerid CHAR(5)
DECLARE @orderdate DATE
DECLARE @shippedate DATE
DECLARE @city CHAR(15)
DECLARE @freight MONEY
DECLARE @shipname VARCHAR(40)
DECLARE @shipaddress VARCHAR(60)
DECLARE @fax CHAR(60)
DECLARE @quantity INT
DECLARE orderCurs CURSOR FOR SELECT * FROM orders WHERE GETDATE()>DATEADD(year,2,orderdate);
BEGIN
IF NOT EXISTS (SELECT 1 FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME='archieved_orders') CREATE TABLE archieved_orders (
orderid INT NOT NULL PRIMARY KEY,
customerid CHAR(5) NOT NULL FOREIGN KEY REFERENCES customers(customerid),
orderdate DATE,
shippedate DATE,
city CHAR(15),
freight MONEY,
shipname VARCHAR(40),
shipaddress VARCHAR(60),
fax CHAR(60),
quantity INT
);
 OPEN orderCurs;
 FETCH NEXT FROM orderCurs INTO @orderid,@customerid,@orderdate,@shippedate,@city,@freight,@shipname,@shipaddress,@fax,@quantity
 WHILE @@FETCH_STATUS=0
 BEGIN
 INSERT INTO archieved_orders VALUES (@orderid,@customerid,@orderdate,@shippedate,@city,@freight,@shipname,@shipaddress,@fax,@quantity);
 DELETE FROM orders WHERE orederid=@orderid
 FETCH NEXT FROM orderCurs INTO  @orderid,@customerid,@orderdate,@shippedate,@city,@freight,@shipname,@shipaddress,@fax,@quantity
END
CLOSE orderCurse
END

Messages
Command(s) completed successfully.
```

Figure 8: The third procedure

The orders table before execution is shown on the pic below.

orederid	customerid	orderdate	shipdate	city	freighy	shipname	shipaddress	fax	quantity
3	1	2014-03-01	2014-03-08	Kuopio	2,5000	Jon	Sun st. 7	234534536 ...	3
1	1	2011-04-06	2012-06-02	Kuopio	4,0000	Ann	Tai st. 3	354546454 ...	5
2	1	2000-05-03	2001-05-08	Kajaani	2,0000	Jin	Moon st. 19	434535345 ...	2

Figure 9: Orders before the procedure execution

After the execution I have got two table pic. 10. The first table shows as the archived orders and the second one current.

orderid	customerid	orderdate	shippedate	city	freight	shipname	shipaddress	fax	quantity
1	1	2011-04-06	2012-06-02	Kuopio	4,0000	Ann	Tai st. 3	354546454 ...	5
2	1	2000-05-03	2001-05-08	Kajaani	2,0000	Jin	Moon st. 19	434535345 ...	2

orederid	customerid	orderdate	shipdate	city	freighy	shipname	shipaddress	fax	quantity
3	1	2014-03-01	2014-03-08	Kuopio	2,5000	Jon	Sun st. 7	234534536 ...	3

Figure 10: Archived_orders and orders table after the procedure execution