Dynamic SQL and BIML
Birds of a feather

Why cursors are important for dynamic SQL

Posted on 2017-07-18 by Jonas Henriksson

My Journey into SQL Server started with MS Access. At first I built tables, forms and reports with the help of the GUI. After a while I realized that I could build more functionality by programming the JET engine. A common pattern was to populate a record set and then move through it in some loop structure to achieve what you wanted. When I took the step into SQL Server and find out about cursors it felt familiar, "I know this, I can solve a lot of problems with cursors". As months with SQL Server became years I realized that there was a lot of problems with cursors. Performance problems is well documented, but I also feel solutions built on cursor are harder to follow and debug. Cursors were no longer my best friend. Your code should be based on sets not on cursors. Actually it came to the point were cursors became the lack must test for a SQL developer: "Oh, you've solved this with a cursor. Was that really necessary?"

But "Just when You thought cursors were out… they were pulled back in" to paraphrase Micheal Corleone in Godfather III. If you are working with dynamic SQL that is. The typical pattern for dynamic SQL is that you want to do the same thing for more than one object. It could be issuing ALTER TABLEs statement, issuing index rebuilds or even SELECT statements. If you just wanted to do this for one object you might as well build a static object. Hence dynamic SQL solutions will have a lot of cursors so you need a cursor among your templates. In the dynamic SQL world cursors are nothing to ashamed about. On the contrary, they shall be embraced and you should have a good template that you can use over and over again.

The template below, Simple Cursor, is intended for building a script that is executed and monitored from SSMS. This allows for you to read the output in the result as well as the messages windows in SSMS and take action if needed. Progress could be outputted to the messages windows with the flexible PRINT command.

It you would like to build a cursor that is executed from some kind of application, e.g. SQL Server Agent, you need to put a lot of thought into error handling. At the very least it should communicate what statement caused the error and a valid error message. This will be the topic for later posts.

```
/* Simple Cursor template from www.dynamic-sql-and-biml.com */
SET XACT_ABORT, NOCOUNT ON;

DECLARE @SQL varchar(max);

DECLARE curSQL CURSOR LOCAL STATIC READ_ONLY
FOR
SELECT 'USE ' + name + ';'
+ char(10)
+ 'SELECT DbName = DB_NAME(), NoTables = count(*) from sys.tables;'
FROM sys.databases;

OPEN curSQL;

if CURSOR_STATUS('local','curSQL') <= 0
  RAISERROR('Cursor curSQL is either empty or has failed somehow', 16, 1);
FETCH NEXT FROM curSQL INTO @SQL;

WHILE @@FETCH_STATUS = 0
BEGIN
```

```
if @SQL is null RAISERROR('SQL statement is null', 16, 1);

  exec(@SQL);
  print @SQL;

FETCH NEXT FROM curSQL INTO @SQL;
END;

if CURSOR_STATUS('local','curSQL') >= 0 CLOSE curSQL;
if CURSOR_STATUS('local','curSQL') >= -2 DEALLOCATE curSQL;
```

This entry was posted in Dynamic SQL and tagged Cursor. Bookmark the permalink.

Dynamic SQL and BIML
Proudly powered by WordPress.