http://www.sqlservercentral.com/articles/T-SQL/61461/
Printed 2007/11/15 07:48AM

TSQL Lab I - Playing around with ROW_NUMBER

By Jacob Sebastian, 2007/11/01

Introduction

In this series of articles, I will present a few real-life problems and a few different ways to solve them. In this series we may not discuss new stuff. Rather, we will be going through things that you already know. More specifically, we would focus on performing some of the day-to-day tasks using TSQL functions and KEYWORDS that we already know.

This article is about a TSQL query that I had to write recently to generate certain values based on certain rules. I have an excel sheet which contains the Item Numbers of a warehouse management application. I need to generate a new number for each item based on a set of rules.

The Problem

I need to generate a new Item Code for the items in the warehouse. The current Code is 12 digit long and it is getting difficult to perform manual data entry. The idea is to generate a new Item Code for each item. Here is the sample data that I have. (For privacy reasons, I am not presenting the actual data)

Category	Item Description
LAYH	**Item 1**
LAYH	Item 2
LAYH	Item 3
LSA	Item 4
LSA	Item 5
LSA	Item 6
LSA	Item 7
LSA	Item 8
LSA	Item 9
LSA	Item 10
LSA	Item 11
LSA	Item 12
LSA	Item 13
LSA	Item 14
LSA	Item 15
LSA	Item 16
LSA	Item 17
LSA	Item 18
LSA	Item 19
LSA	Item 20
UT	Item 21
UT	Item 22

I need to generate the new Item code based on the following rules.

- The Item code should be generated from the Category Code (first column)
- The new Item code should be 6 characters long.
- Append a sequence number at the end of the category code to generate the new number.

- Prefix the sequence number with "0"s (one or more zeros) to make the length of the code to 6 characters.
- Length of category code will vary from 2 to 4 characters.
- Some of the categories have close to 1000 items under them. So the sequence number should use 0-9 as well as A-Z to make sure that we have enough combination of values. For example, after 09 the next value should be 0A, should go until 0Z and then jump to 10

Here is the expected result and here is the excel file that I am using.

Category	Item Description	Item ID
LAYH	**Item 1**	**LAYH00**
LAYH	Item 2	LAYH01
LAYH	Item 3	LAYH02
LSA	Item 4	LSA000
LSA	Item 5	LSA001
LSA	Item 6	LSA002
LSA	Item 7	LSA003
LSA	Item 8	LSA004
LSA	Item 9	LSA005
LSA	Item 10	LSA006
LSA	Item 11	LSA007
LSA	Item 12	LSA008
LSA	Item 13	LSA009
LSA	Item 14	LSA00A
LSA	Item 15	LSA00B
LSA	Item 16	LSA00C
LSA	Item 17	LSA00D
LSA	Item 18	LSA00E
LSA	Item 19	LSA00F
LSA	Item 20	LSA00G
UT	Item 21	UT0000
UT	Item 22	UT0001

Writing the query

There are at least 2 (or even more) approaches that we can jump start with. I did not want to use a CURSOR. Hence I thought of writing a query that generates the new item code by applying all the rules. I wanted to use ROW_NUMBER() with ORDER BY and PARTITION BY to generate a sequence number within each group.

What is little difficult is generating the sequence number the way we need. We do not need a numeric sequence number. But we need to generate the sequence number using a mixture of digits and letters. I wanted to use a CTE to generate such a sequence.

Solving this problem includes two parts. The first part is to generate a sequence number within each group by using ROW_NUMEBR() by PARTITION. The second part is to generate the custom sequence number that replaces numeric sequence number generated by the first part.

Accessing the data from the excel sheet

This is pretty simple. You can use OPENROWSET to do so. The following code shows that.

```
WITH items AS (
      SELECT * FROM OPENROWSET('Microsoft.Jet.OLEDB.4.0',
      'Excel 8.0;DATABASE=c:\temp\items.xls', 'Select * from [items$]')
)
SELECT * FROM items
```

```
/*
OUTPUT

Category Item Description
-------- ----------------
LAYH     Item 1
LAYH     Item 2
LAYH     Item 3
LSA      Item 4
LSA      Item 5
LSA      Item 6
LSA      Item 7
LSA      Item 8
LSA      Item 9
LSA      Item 10
LSA      Item 11
LSA      Item 12
LSA      Item 13
LSA      Item 14
LSA      Item 15
LSA      Item 16
LSA      Item 17
LSA      Item 18
LSA      Item 19
LSA      Item 20
UT       Item 21
UT       Item 22
*/
```

At the next step, let us generate the sequence number for each category. We will be using ROW_NUMBER() and PARTITION BY to generate the sequence number.

```
WITH
items AS (
      SELECT * FROM OPENROWSET('Microsoft.Jet.OLEDB.4.0',
      'Excel 8.0;DATABASE=c:\temp\items.xls', 'Select * from [items$]')
),
ItemCode AS (
      SELECT
            Category,
            [Item Description],
            ROW_NUMBER() OVER ( PARTITION BY Category ORDER BY Category)
AS sequence
      FROM items
)

SELECT * FROM ItemCode ORDER BY Category

/*
OUTPUT

Category Item Description Sequence
-------- ---------------- --------------------
LAYH     Item 1           1
LAYH     Item 2           2
LAYH     Item 3           3
LSA      Item 4           1
LSA      Item 5           2
LSA      Item 6           3
LSA      Item 7           4
LSA      Item 8           5
LSA      Item 9           6
LSA      Item 10          7
LSA      Item 11          8
LSA      Item 12          9
LSA      Item 13          10
LSA      Item 14          11
LSA      Item 15          12
LSA      Item 16          13
LSA      Item 17          14
```

```
LSA       Item 18          15
LSA       Item 19          16
LSA       Item 20          17
UT        Item 21          1
UT        Item 22          2
*/
```

At the next stage, let us generate the custom sequence number that we needed. Let us use a CTE to do that. Our sequence number will start with "00" and will end with "ZZ".

```
WITH seq AS (
    SELECT '0' AS ch, 0 AS sr
      UNION SELECT '1',1
      UNION SELECT '2',2
      UNION SELECT '3',3
      UNION SELECT '4',4
      UNION SELECT '5',5
      UNION SELECT '6',6
      UNION SELECT '7',7
      UNION SELECT '8',8
      UNION SELECT '9',9
      UNION SELECT 'A',10
      UNION SELECT 'B',11
      UNION SELECT 'C',12
      UNION SELECT 'D',13
      UNION SELECT 'E',14
      UNION SELECT 'F',15
      UNION SELECT 'G',16
      UNION SELECT 'H',17
      UNION SELECT 'I',18
      UNION SELECT 'J',19
      UNION SELECT 'K',20
      UNION SELECT 'L',21
      UNION SELECT 'M',22
      UNION SELECT 'N',23
      UNION SELECT 'O',24
      UNION SELECT 'P',25
      UNION SELECT 'Q',26
      UNION SELECT 'R',27
      UNION SELECT 'S',28
      UNION SELECT 'T',29
      UNION SELECT 'U',30
      UNION SELECT 'V',31
      UNION SELECT 'W',32
      UNION SELECT 'X',33
      UNION SELECT 'Y',34
      UNION SELECT 'Z',35
),
NewSeq AS (
SELECT
      ROW_NUMBER() OVER(ORDER BY a.sr, b.sr) AS SrNo,
      a.ch + b.ch  AS Digit
FROM seq a
CROSS JOIN seq b
)

SELECT * FROM newseq

/*
OUTPUT:

SrNo                 Digit
-------------------- -----
1                    00
2                    01
3                    02
4                    03
5                    04
6                    05
7                    06
8                    07
```

```
9                    08
10                   09
11                   0A
12                   0B
13                   0C
14                   0D
15                   0E
16                   0F
17                   0G
18                   0H
19                   0I
20                   0J
21                   0K
22                   0L
23                   0M
24                   0N
25                   0O
26                   0P
27                   0Q
28                   0R
29                   0S
30                   0T
31                   0U
32                   0V
33                   0W
34                   0X
35                   0Y
36                   0Z
37                   10
*/
```

Now, let us go to the final step. Here is the code which generates the new *Item Code*.

```
WITH seq AS (
    SELECT '0' AS ch, 0 AS sr
      UNION SELECT '1',1
      UNION SELECT '2',2
      UNION SELECT '3',3
      UNION SELECT '4',4
      UNION SELECT '5',5
      UNION SELECT '6',6
      UNION SELECT '7',7
      UNION SELECT '8',8
      UNION SELECT '9',9
      UNION SELECT 'A',10
      UNION SELECT 'B',11
      UNION SELECT 'C',12
      UNION SELECT 'D',13
      UNION SELECT 'E',14
      UNION SELECT 'F',15
      UNION SELECT 'G',16
      UNION SELECT 'H',17
      UNION SELECT 'I',18
      UNION SELECT 'J',19
      UNION SELECT 'K',20
      UNION SELECT 'L',21
      UNION SELECT 'M',22
      UNION SELECT 'N',23
      UNION SELECT 'O',24
      UNION SELECT 'P',25
      UNION SELECT 'Q',26
      UNION SELECT 'R',27
      UNION SELECT 'S',28
      UNION SELECT 'T',29
      UNION SELECT 'U',30
      UNION SELECT 'V',31
      UNION SELECT 'W',32
      UNION SELECT 'X',33
      UNION SELECT 'Y',34
      UNION SELECT 'Z',35
),
```

```
NewSeq AS (
      SELECT
            ROW_NUMBER() OVER(ORDER BY a.sr, b.sr) AS SrNo,
            a.ch + b.ch  AS CustomSequence
      FROM seq a
      CROSS JOIN seq b
),
items AS (
      SELECT * FROM OPENROWSET('Microsoft.Jet.OLEDB.4.0',
      'Excel 8.0;DATABASE=c:\temp\items.xls', 'Select * from [items$]')
),
ItemCode AS (
      SELECT
            Category,
            [Item Description],
            ROW_NUMBER() OVER ( PARTITION BY Category ORDER BY Category)
AS sequence
      FROM items
)

SELECT
      Category,
      [Item Description],
      Category + replicate('0', 4-len(Category)) + CustomSequence AS
NewCode
FROM ItemCode i
INNER JOIN NewSeq s ON i.sequence = s.Srno
ORDER BY Category, NewCode

/*
OUTPUT:

Category Item Description NewCode
-------- ---------------- --------
LAYH     Item 1           LAYH00
LAYH     Item 2           LAYH01
LAYH     Item 3           LAYH02
LSA      Item 4           LSA000
LSA      Item 5           LSA001
LSA      Item 6           LSA002
LSA      Item 7           LSA003
LSA      Item 8           LSA004
LSA      Item 9           LSA005
LSA      Item 10          LSA006
LSA      Item 11          LSA007
LSA      Item 12          LSA008
LSA      Item 13          LSA009
LSA      Item 14          LSA00A
LSA      Item 15          LSA00B
LSA      Item 16          LSA00C
LSA      Item 17          LSA00D
LSA      Item 18          LSA00E
LSA      Item 19          LSA00F
LSA      Item 20          LSA00G
UT       Item 21          UT0000
UT       Item 22          UT0001
*/
```

Conclusions

As I had mentioned at the beginning of this article, there are always more than one way to do a given programming task. What I am presenting here may not be the best for your specific requirement. However, this might give you a hint or help you to clear a syntactic question that will help you to write the TSQL code for your specific requirement.
