# 

**Article:** SQL Server – Using a fast cursor

**Date**: 12/03/2012

**Posted by**: HeelpBook Staff

**Source**: [Link](http://www.kodyaz.com/articles/sql-cursor-example-tables-rows-count-using-sql-server-cursor.aspx)

**Permalink**: [Link](http://heelpbook.altervista.org/2012/sql-server-list-count-of-rows-in-all-tables-in-database-using-a-cursor/)

# **SQL Server – List Count of Rows in All Tables in Database using a Cursor**

In this document, sql developers will find a **SQL cursor example** t-sql code to list number of rows (record counts) in all user tables in a MS SQL Server database.

Note that with the **T-SQL** enhancements introduced with **MS SQL Server 2005** and **MS SQL Server 2008**, developers and database administrators can find ways to avoid using SQL Server cursor in their sql codes in their jobs.

Do not use frequently **sql cursor** in production system during high load times. **SQL cursors** if not done in the correct declaration can effect the performance of database applications in negative manner. So if possible prevent developers use **transact-sql cursor** in their sql scripts.

In the following **SQL Server** cursor, you will first notice the **cursor declaration** in sql script. You can **declare sql cursor** using **DECLARE cursorname CURSOR syntax**.

This **sample sql cursor** is build over a list of user tables defined in a database, and is used for listing the count of rows in each database table. Since for each table a sql select query is build and executed seperately, the sql cursor or the loop for running select task for each table consumes considerable resource.   
Do not forget, sql engine is built and optimized for batch processes not for single row processes.

So if possible avoid from using sql cursor in sql codes.

#### SQL Cursor Example

Here is a sql cursor example :

DECLARE @TableName sysname

DECLARE @SQL nvarchar(max)

DECLARE tables\_cursor CURSOR FAST\_FORWARD

FOR

SELECT name FROM sys.tables

OPEN tables\_cursor

FETCH NEXT FROM tables\_cursor INTO @TableName

WHILE @@FETCH\_STATUS = 0

BEGIN

SELECT @SQL =

'SELECT ''' + @TableName + ''', COUNT(\*) as RowsCount ' +

'FROM [' + @TableName + ']'

EXEC SP\_EXECUTESQL @SQL

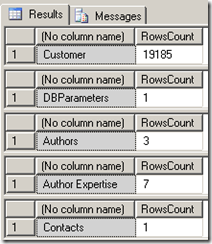
FETCH NEXT FROM tables\_cursor INTO @TableName

END

CLOSE tables\_cursor

DEALLOCATE tables\_cursor

And when I execute the above sample **sql cursor** code on one of my SQL Server databases, I get the following sample **t-sql** cursor output :



The above cursor is declared as **FAST\_FORWARD** which is faster than other cursor declaration types.