#

**Article:** Alternative way to get table’s row count

**Date**: 19/03/2012

**Posted by**: HeelpBook Staff

**Source**: [Link](http://www.mssqlcity.com/Articles/KnowHow/RowCount.htm)

**Permalink**: [Link](http://heelpbook.altervista.org/2012/sql-server-alternative-way-to-get-the-tables-row-count/)

# **SQL Server – Alternative way to get the table’s row count**

To get the total row count in a table, we usually use the following select statement:

SELECT count(\*) FROM table\_name

*This query performs full table scan to get the row count*.

You can check it by setting **SET SHOWPLAN ON** for **SQL Server 6.5** or **SET SHOWPLAN\_TEXT ON** for **SQL Server 7.0/2000**. So, if the table is very big, it can take a lot of time. In this example, the **tbTest** table will be created and **10000** rows will be inserted into this table:

CREATE TABLE tbTest (

 id int identity primary key,

 Name char(10)

)

GO

DECLARE @i int

SELECT @i = 1

WHILE @i <= 10000

 BEGIN

 INSERT INTO tbTest VALUES (LTRIM(str(@i)))

 SELECT @i = @i + 1

 END

GO

There is another way to determine the total row count in a table. You can use the **sysindexes** system table for this purpose. There is **ROWS** column in the **sysindexes** table. This column contains the total row count for each table in your database. So, you can use the following select statement instead of above one:

SELECT rows FROM sysindexes WHERE id = OBJECT\_ID('table\_name') AND indid < 2

There are physical read and logical read operations. A logical read occurs if the page is currently in the cache. If the page is not currently in the cache, a physical read is performed to read the page into the cache. To see how many logical or physical read operations were made, you can use **SET STATISTICS IO ON** command.

This is the example:

SET STATISTICS IO ON

GO

SELECT count(\*) FROM tbTest

GO

SELECT rows FROM sysindexes WHERE id = OBJECT\_ID('tbTest') AND indid < 2

GO

SET STATISTICS IO OFF

GO

This is the result:

-----------

10000

(1 row(s) affected)

Table 'tbTest'. Scan count 1, logical reads 32, physical reads 0, read-ahead reads 0.

rows

-----------

10000

(1 row(s) affected)

Table 'sysindexes'. Scan count 1, logical reads 2, physical reads 0, read-ahead reads 0.

So, you can improve the speed of the first query in several times.

This works for **SQL Server 6.5** and **SQL Server 7.0/2000/2005** as well.