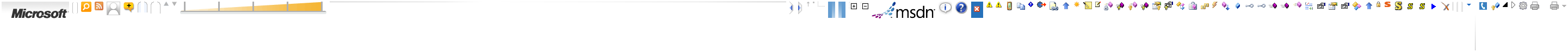
[](http://msdn.microsoft.com/en-us/default.aspx)

**FETCH (Transact-SQL)**

**SQL Server 2008 R2**

Retrieves a specific row from a Transact-SQL server cursor.

Topic link icon[Transact-SQL Syntax Conventions1](http://msdn.microsoft.com/en-us/library/ms177563.aspx)

[Syntax](javascript:void(0))

FETCH

          [ [ NEXT | PRIOR | FIRST | LAST

                    | ABSOLUTE { n | @nvar }

                    | RELATIVE { n | @nvar }

               ]

               FROM

          ]

{ { [ GLOBAL ] cursor\_name } | @cursor\_variable\_name }

[ INTO @variable\_name [ ,...n ] ]

[Arguments](javascript:void(0))

NEXT

Returns the result row immediately following the current row and increments the current row to the row returned. If FETCH NEXT is the first fetch against a cursor, it returns the first row in the result set. NEXT is the default cursor fetch option.

PRIOR

Returns the result row immediately preceding the current row, and decrements the current row to the row returned. If FETCH PRIOR is the first fetch against a cursor, no row is returned and the cursor is left positioned before the first row.

FIRST

Returns the first row in the cursor and makes it the current row.

LAST

Returns the last row in the cursor and makes it the current row.

ABSOLUTE { n| @nvar}

If n or @nvar is positive, returns the row n rows from the front of the cursor and makes the returned row the new current row. If n or @nvar is negative, returns the row n rows before the end of the cursor and makes the returned row the new current row. If n or @nvar is 0, no rows are returned. n must be an integer constant and @nvar must be smallint, tinyint, or int.

RELATIVE { n| @nvar}

If n or @nvar is positive, returns the row n rows beyond the current row and makes the returned row the new current row. If n or @nvar is negative, returns the row n rows prior to the current row and makes the returned row the new current row. If n or @nvar is 0, returns the current row. If FETCH RELATIVE is specified with n or @nvar set to negative numbers or 0 on the first fetch done against a cursor, no rows are returned. n must be an integer constant and @nvar must be smallint, tinyint, or int.

GLOBAL

Specifies that cursor\_name refers to a global cursor.

cursor\_name

Is the name of the open cursor from which the fetch should be made. If both a global and a local cursor exist with cursor\_name as their name, cursor\_name to the global cursor if GLOBAL is specified and to the local cursor if GLOBAL is not specified.

@cursor\_variable\_name

Is the name of a cursor variable referencing the open cursor from which the fetch should be made.

INTO @variable\_name[ ,...n]

Allows data from the columns of a fetch to be placed into local variables. Each variable in the list, from left to right, is associated with the corresponding column in the cursor result set. The data type of each variable must either match or be a supported implicit conversion of the data type of the corresponding result set column. The number of variables must match the number of columns in the cursor select list.

[Remarks](javascript:void(0))

If the SCROLL option is not specified in an ISO style DECLARE CURSOR statement, NEXT is the only FETCH option supported. If SCROLL is specified in an ISO style DECLARE CURSOR, all FETCH options are supported.

When the Transact-SQL DECLARE cursor extensions are used, these rules apply:

* If either FORWARD\_ONLY or FAST\_FORWARD is specified, NEXT is the only FETCH option supported.
* If DYNAMIC, FORWARD\_ONLY or FAST\_FORWARD are not specified, and one of KEYSET, STATIC, or SCROLL are specified, all FETCH options are supported.
* DYNAMIC SCROLL cursors support all the FETCH options except ABSOLUTE.

The @@FETCH\_STATUS function reports the status of the last FETCH statement. The same information is recorded in the fetch\_status column in the cursor returned by sp\_describe\_cursor. This status information should be used to determine the validity of the data returned by a FETCH statement prior to attempting any operation against that data. For more information, see [@@FETCH\_STATUS (Transact-SQL)2](http://msdn.microsoft.com/en-us/library/ms187308.aspx).

[Permissions](javascript:void(0))

FETCH permissions default to any valid user.

[Examples](javascript:void(0))

**A. Using FETCH in a simple cursor**

The following example declares a simple cursor for the rows in the Person.Person table with a last name that starts with B, and uses FETCH NEXT to step through the rows. The FETCH statements return the value for the column specified in DECLARE CURSOR as a single-row result set.

USE AdventureWorks2008R2;

GO

DECLARE contact\_cursor CURSOR FOR

SELECT LastName FROM Person.Person

WHERE LastName LIKE 'B%'

ORDER BY LastName;

OPEN contact\_cursor;

-- Perform the first fetch.

FETCH NEXT FROM contact\_cursor;

-- Check @@FETCH\_STATUS to see if there are any more rows to fetch.

WHILE @@FETCH\_STATUS = 0

BEGIN

-- This is executed as long as the previous fetch succeeds.

FETCH NEXT FROM contact\_cursor;

END

CLOSE contact\_cursor;

DEALLOCATE contact\_cursor;

GO

**B. Using FETCH to store values in variables**

The following example is similar to example A, except the output of the FETCH statements is stored in local variables instead of being returned directly to the client. The PRINT statement combines the variables into a single string and returns them to the client.

USE AdventureWorks2008R2;

GO

-- Declare the variables to store the values returned by FETCH.

DECLARE @LastName varchar(50), @FirstName varchar(50);

DECLARE contact\_cursor CURSOR FOR

SELECT LastName, FirstName FROM Person.Person

WHERE LastName LIKE 'B%'

ORDER BY LastName, FirstName;

OPEN contact\_cursor;

-- Perform the first fetch and store the values in variables.

-- Note: The variables are in the same order as the columns

-- in the SELECT statement.

FETCH NEXT FROM contact\_cursor

INTO @LastName, @FirstName;

-- Check @@FETCH\_STATUS to see if there are any more rows to fetch.

WHILE @@FETCH\_STATUS = 0

BEGIN

-- Concatenate and display the current values in the variables.

PRINT 'Contact Name: ' + @FirstName + ' ' + @LastName

-- This is executed as long as the previous fetch succeeds.

FETCH NEXT FROM contact\_cursor

INTO @LastName, @FirstName;

END

CLOSE contact\_cursor;

DEALLOCATE contact\_cursor;

GO

**C. Declaring a SCROLL cursor and using the other FETCH options**

The following example creates a SCROLL cursor to allow full scrolling capabilities through the LAST, PRIOR, RELATIVE, and ABSOLUTE options.

USE AdventureWorks2008R2;

GO

-- Execute the SELECT statement alone to show the

-- full result set that is used by the cursor.

SELECT LastName, FirstName FROM Person.Person

ORDER BY LastName, FirstName;

-- Declare the cursor.

DECLARE contact\_cursor SCROLL CURSOR FOR

SELECT LastName, FirstName FROM Person.Person

ORDER BY LastName, FirstName;

OPEN contact\_cursor;

-- Fetch the last row in the cursor.

FETCH LAST FROM contact\_cursor;

-- Fetch the row immediately prior to the current row in the cursor.

FETCH PRIOR FROM contact\_cursor;

-- Fetch the second row in the cursor.

FETCH ABSOLUTE 2 FROM contact\_cursor;

-- Fetch the row that is three rows after the current row.

FETCH RELATIVE 3 FROM contact\_cursor;

-- Fetch the row that is two rows prior to the current row.

FETCH RELATIVE -2 FROM contact\_cursor;

CLOSE contact\_cursor;

DEALLOCATE contact\_cursor;

GO

[See Also](javascript:void(0))

**Reference**

[CLOSE (Transact-SQL)3](http://msdn.microsoft.com/en-us/library/ms175035.aspx)

[DEALLOCATE (Transact-SQL)4](http://msdn.microsoft.com/en-us/library/ms188782.aspx)

[DECLARE CURSOR (Transact-SQL)5](http://msdn.microsoft.com/en-us/library/ms180169.aspx)

[OPEN (Transact-SQL)6](http://msdn.microsoft.com/en-us/library/ms190500.aspx)

**Concepts**

[Transact-SQL Cursors7](http://msdn.microsoft.com/en-us/library/ms190028.aspx)

Links Table

1http://msdn.microsoft.com/en-us/library/ms177563.aspx

2http://msdn.microsoft.com/en-us/library/ms187308.aspx

3http://msdn.microsoft.com/en-us/library/ms175035.aspx

4http://msdn.microsoft.com/en-us/library/ms188782.aspx

5http://msdn.microsoft.com/en-us/library/ms180169.aspx

6http://msdn.microsoft.com/en-us/library/ms190500.aspx

7http://msdn.microsoft.com/en-us/library/ms190028.aspx

Community Content

How to fetch cursor rowtype columns?

I need to fetch cursor rowtype fields without hardcord(that means without declare all local variables).Because i have total 47 columns in my cursor.So please provide solution for this....

http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif

[http://i1.social.s-msft.com/profile/u/avatar.jpg?displayname=svathi&size=small](http://msdn.microsoft.com/en-us/library/community/user/1630776%28d=printer%29.aspx)

* 12/3/2010
* [Svathi](http://msdn.microsoft.com/en-us/library/community/user/1630776%28d=printer%29.aspx)

I agree

I had always hated Fetching twice for the simple fact, if I change the output fields.

http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif

[http://i1.social.s-msft.com/profile/u/avatar.jpg?displayname=jason+barrera&size=small](http://msdn.microsoft.com/en-us/library/community/user/1572115%28d=printer%29.aspx)

* 10/8/2010
* [Jason Barrera](http://msdn.microsoft.com/en-us/library/community/user/1572115%28d=printer%29.aspx)

You don't have to repeat the FETCH statement twice

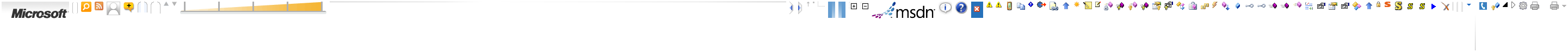
Personally I don't like to repeat the same FETCH statement before the loop and then again during the loop. Here is an alternative way to write your cursor loop to avoid doing that: http://sql-dotnet.blogspot.com/2010/09/dont-repeat-fetch-statement-with-sql.html

http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif

[http://i1.social.s-msft.com/profile/u/avatar.jpg?displayname=lex3001&size=small](http://msdn.microsoft.com/en-us/library/community/user/44008%28d=printer%29.aspx)

* 9/24/2010
* [lex3001](http://msdn.microsoft.com/en-us/library/community/user/44008%28d=printer%29.aspx)

© 2012 Microsoft. All rights reserved.

[](javascript:void(0))