**Аргументы**

NEXT  
Возвращает строку результата сразу же за текущей строкой и перемещает указатель текущей строки на возвращенную строку. Если инструкция FETCH NEXT выполняет первую выборку в отношении курсора, она возвращает первую строку в результирующем наборе. NEXT является параметром по умолчанию выборки из курсора.

PRIOR  
Возвращает строку результата, находящуюся непосредственно перед текущей строкой и перемещает указатель текущей строки на возвращенную строку. Если инструкция FETCH PRIOR выполняет первую выборку из курсора, не возвращается никакая строка и положение курсора остается перед первой строкой.

FIRST  
Возвращает первую строку в курсоре и делает ее текущей.

LAST  
Возвращает последнюю строку в курсоре, и делает ее текущей.

ABSOLUTE { *n*| @*nvar*}  
Если *n* или @*nvar* является положительным, возвращает строку *n* строк из начала курсора и делает возвращенную строку новой текущей строкой. Если *n* или @*nvar* является отрицательным, возвращает строку *n* строк в конец курсора и делает возвращенную строку новой текущей строкой. Если *n* или @*nvar* равно 0, строки не возвращаются. *n* должен быть целочисленной константой, а @*nvar* должен иметь тип данных **smallint**, **tinyint** или **int**.

RELATIVE { *n*| @*nvar*}  
Если *n* или @*nvar* является положительным, возвращает строку *n* строк за текущую строку и делает возвращенную строку новой текущей строкой. Если *n* или @*nvar* является отрицательным, возвращает строку *n* строк перед текущей строкой и делает возвращенную строку новой текущей строкой. Если *n* или @*nvar* равно 0, возвращает текущую строку. Если FETCH RELATIVE указана с *n* или @*nvar*, заданными с отрицательными числами или 0 в первой выборке, выполненной с курсором, строки не возвращаются. *n* должен быть целочисленной константой, а @*nvar* должен иметь тип данных **smallint**, **tinyint** или **int**.

GLOBAL  
Указывает, что аргумент *cursor\_name* ссылается на глобальный курсор.

*cursor\_name*  
Имя открытого курсора, из которого должна быть произведена выборка. Когда имеется как глобальный, так и локальный курсор с именем *cursor\_name*, то *cursor\_name* ссылается на глобальный курсор, если задано GLOBAL, и на локальный, если GLOBAL не задано.

@*cursor\_variable\_name*  
Имя переменной курсора, ссылающейся на открытый курсор, из которого должна быть произведена выборка.

INTO @*variable\_name*[ ,...*n*]  
Позволяет поместить данные из столбцов выборки в локальные переменные. Каждая переменная из списка, слева направо, связывается с соответствующим столбцом в результирующем наборе курсора. Тип данных каждой переменной должен соответствовать типу данных соответствующего столбца результирующего набора, или должна обеспечиваться поддержка неявного преобразования в тип данных этого столбца. Количество переменных должно совпадать с количеством столбцов в списке выбора курсора.

**Комментарии**

Если параметр SCROLL не указан в инструкции DECLARE CURSOR в стиле ISO, NEXT является единственным поддерживаемым параметром FETCH. Если SCROLL указан в стиле ISO DECLARE CURSOR, поддерживаются все параметры FETCH.

При использовании расширений курсора Transact-SQL DECLARE применяются следующие правила.

* Если указан параметр FORWARD\_ONLY или FAST\_FORWARD, NEXT является единственным поддерживаемым параметром FETCH.
* Если DYNAMIC, FORWARD\_ONLY или FAST\_FORWARD не указаны и указан один из параметров KEYSET, STATIC и SCROLL, поддерживаются все параметры FETCH.
* Курсоры DYNAMIC SCROLL поддерживают все параметры FETCH, кроме ABSOLUTE.

Функция @@FETCH\_STATUS возвращает состояние последней инструкции FETCH. Те же данные записываются в столбец fetch\_status в курсоре, возвращаемом процедурой sp\_describe\_cursor. Эти сведения о состоянии должны использоваться для определения действительности данных, возвращаемых инструкцией FETCH перед попыткой выполнения любой операции с этими данными. Дополнительные сведения см. в разделе документации [@@FETCH\_STATUS (Transact-SQL)](https://learn.microsoft.com/ru-ru/sql/t-sql/functions/fetch-status-transact-sql?view=sql-server-ver16).

**Разрешения**

Разрешения FETCH по умолчанию предоставляются всем допустимым пользователям.

**Примеры**

**A. Использование инструкции FETCH в простом курсоре**

В следующем примере объявляется простой курсор для строк в таблице Person.Person с фамилией, начинающийся с B, и используется FETCH NEXT для пошагового выполнения строк. Инструкции FETCH возвращают значение для столбца, указанного в инструкции DECLARE CURSOR в качестве однострочного результирующего набора.

SQLКопировать

USE AdventureWorks2012;

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

**Б. Использование инструкции FETCH для сохранения значений в переменных**

Следующий пример аналогичен примеру A, за исключением того, что выход инструкций FETCH сохраняется в локальных переменных, а не возвращается непосредственно клиенту. Инструкция PRINT объединяет переменные в одну строку и возвращает их клиенту.

SQLКопировать

USE AdventureWorks2012;

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

**В. Объявление курсора SCROLL и использование других параметров инструкции FETCH**

В следующем примере создается курсор SCROLL, с помощью которого можно получить полные возможности прокрутки с помощью параметров LAST, PRIOR, RELATIVE и ABSOLUTE.

SQLКопировать

USE AdventureWorks2012;

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