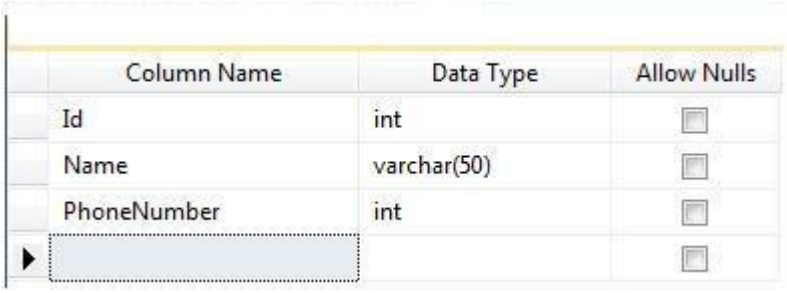
MS SQL 2011 (Denali) – With Result Set

Модификация возвращаемого набора данных (NEW)

В оригинальном звучании и в жизни эта возможность звучит как **With Result Set**. Эта штука позволяет менять имена и типы данных в возвращаемом хранимой процедурой наборе данных.

Перед тем, как мы углубимся в детали использования данной возможности, рассмотрим, как предыдущие версии SQL серверов обходились с данными, которые возвращает хранимая процедура. Какие возможности они предоставляли для работы с результатом.

Для демонстрации работы будем использовать в качестве примера таблицу **tbl_Test** состоящую из 3 колонок.



Column Name	Data Type	Allow Nulls
Id	int	☐
Name	varchar(50)	☐
PhoneNumber	int	☐
		☐

```
-- Drop the table if it exists
IF EXISTS (SELECT * FROM sys.objects WHERE name = N'tbl_Test' AND type = 'U')
    DROP TABLE tbl_Test
GO
SET ANSI_NULLS ON
GO

--Create the table
CREATE TABLE [dbo].[tbl_Test](
        [Id] [int] NOT NULL,
        [Name] [varchar](50) NOT NULL,
        [PhoneNumber] [int] NOT NULL
) ON [PRIMARY]
GO
```

Теперь запишем туда немного информации. Пусть это будет 1000 записей:

```
--Populate the Cte with some records
;With Cte(Id,Name,PhoneNo) As (
Select
        Id = 1
        ,Name='Name' + CAST( 1 As Varchar(50))
        , PhoneNo=12345678
Union All
Select
        Id+1
        ,Name= 'Name' + CAST( Id+1 As Varchar(50))
        , PhoneNo+1
From Cte
Where Id <1000
)
--Insert the records into the table
Insert Into dbo.tbl_test
Select * From Cte
Option( Maxrecursion 0)
```

```
--Display the records
Select *
From tbl_Test
```

! Рекомендую взять скрипт на заметку!

Выполнение скрипта выше выведет примерно следующий набор данных (часть)

```
Id        Name      PhoneNumber
1         Name1     12345678
2         Name2     12345679
3         Name3     12345680
4         Name4     12345681
5         Name5     12345682
```

Теперь напишем процедуру, которая будет выводить данные запроса к таблице **tbl_Test**

```
CREATE PROCEDURE dbo.Usp_FetchRecords
AS
BEGIN

        Select
                Id
                ,Name
                ,PhoneNumber
         From dbo.tbl_Test
END
```

Для того, чтобы получить итоговый набор данных по результатам выполнения хранимой процедуры существует несколько подходов. Некоторые из них описываются и обсуждаются в статье Эрланда Зомарскога (Erland Sommarskog) в его статье. Мы воспользуемся одним из подходов основанном на временных таблицах.

Использование временных таблиц

```
--If the #Temp object exists in the tempdb, then drop it
--Если #Temp существует в базе tempdb, тогда прибить ее
IF OBJECT_ID('tempdb..#Temp') IS NOT NULL
BEGIN
Drop Table #Temp
END

--Create a temporary table
--Создание временной таблицы
CREATE TABLE #Temp
(
        Id int,
        EmpName Varchar(50),
        PhoneNo int
)

--Insert records into the Temporary table from the executed stored proc
--Заполнить временную таблицу записями, возвращенными выполненной процедурой
INSERT INTO #Temp
(
        Id
        ,EmpName
        ,PhoneNo
)
EXEC dbo.Usp_FetchRecords

--Display the records inserted into the temporary table
--Вывести записи внесенные во временную таблицу
```

```
Select * from #Temp
```

Подход описанный выше работает на отлично, если мы заранее знаем какие колонки и какого типа требуются на выходе.

Недостатки этого и похожих подходов:

- Ни один из подходов не предлагает прямого решения проблемы. В любом случае требуется поддержка со стороны временных таблиц или же переменных. Потребляет место в базе данных, под временную таблицу.
- Время выполнение запроса увеличивается
- В случае, когда требуется Open Raw Set или Open query запросы, необходимо явным образом включить функцию *Ad Hoc Distributed Queries* и только потом начинать работу.
- В случае задействования временных таблиц или переменных табличного типа, нужно заранее знать структуру ответа процедуры.

Новый подход MS SQL 2011

Новая версия позволяет избавится от упомянутых недостатков и сейчас мы увидим как именно.

```
EXEC Usp_FetchRecords

WITH RESULT SETS(
       (      [Emp Id] int,
              [Emp Name] varchar(50),
              [Phone Number] varchar(50)
       )
)
```

Вывод данных будет такой:

```
Emp Id   Emp Name Phone Number
1        Name1            12345678
2        Name2            12345679
3        Name3            12345680
4        Name4            12345681
5        Name5            12345682
…
```

Общий синтаксис использования **With Result Set**

```
WITH RESULT SETS(
       (
             Column Name1 DataType [Size]
            , Column Name2 DataType [Size]
            ,  . .  . .  .  . .  . . . . .
            ,  . . . . . . . . . . . . . .
            , Column Name-n DataType [Size]

       )
      ,
       (
             Column Name1 DataType [Size]
            , Column Name2 DataType [Size]
            ,  . .  . .  .  . .  . . . . .
            ,  . . . . . . . . . . . . . .
            , Column Name-n DataType [Size]
       )
       . . . . . . . . . . . . . . . . . .
       . .  . . . . . . . . . . . . . . .
       ,
       (
```

```
             Column Name1 DataType [Size]
            , Column Name2 DataType [Size]
            ,  . .  .  .  .  .  .  .  .  .
            ,  . . . . . . . . . . . . . .
            , Column Name-n DataType [Size]
       )
)
```

Таким образом можно произвольно менять названия колонок в итоговом наборе данных. Можно менять тип данных в рамках, которые позволительны для неявного приведения типов. В противном случае база данных сгенерирует ошибку.

Т.е. в примере ниже база выдаст ошибку о неправомерном приведении типов. Мы пытаемся вернуть тип *int* в то время как поле объявлено как *varchar(50)*.

```
EXEC Usp_FetchRecords
WITH RESULT SETS(
       (      [Emp Id] int,
              [Emp Name] int, -- изменили на тип int
              [Phone Number] varchar(50)
       )
)
```

Во время выполнения скрипта получим следующую ошибку:

Msg 8114, Level 16, State 2, Procedure Usp_FetchRecords, Line 5 Error converting data type varchar to int.

Запрос, который был продемонстрирован выше предназначен для преобразования одиночного результирующего набора данных с применением With Result Set. Однако эта техника, как можно видеть из общего синтаксиса может быть распространена на несколько итоговых наборов. Сейчас будет пример как это сделать.

Представим себе, что у нас есть хранимая процедура, которая возвращает два набора данных.

```
CREATE PROCEDURE [dbo].[Usp_ModifiedFetchRecords]
AS
BEGIN

       Select
                     Id
                     ,Name
                     ,PhoneNumber
              From dbo.tbl_Test;

              Select
                     Id
                     ,Name
              From dbo.tbl_Test
              Where PhoneNumber % 2 = 0
END
```

Второй select возвращает абонентов с четными телефонными номерами. Пример выполнения может быть такой (часть результата)

Results | Messages

2	2	Name2	12345679
3	3	Name3	12345680
4	4	Name4	12345681
5	5	Name5	12345682
6	6	Name6	12345683
7	7	Name7	12345684
8	8	Name8	12345685
9	9	Name9	12345686
10	10	Name10	12345687

	Id	Name
1	1	Name1
2	3	Name3
3	5	Name5
4	7	Name7
5	9	Name9
6	11	Name11
7	13	Name13
8	15	Name15
9	17	Name17
10	19	Name19

Теперь попробуем применить With Result Set, чтобы получить более удобоваримый результат без изменения самой хранимой процедуры.

```
EXEC Usp_ModifiedFetchRecords

WITH RESULT SETS(
        (       [Emp Id From First Result Set] int,
                [Emp Name From First Result Set] varchar(50),
                [Phone Number From First Result Set] varchar(50)
        )  ,
        (       [Emp Id From Second Result Set] int,
                [Emp Name From Second Result Set] varchar(50)
        )
)
```

Результат выполнения теперь будет в духе:

Results | Messages

	Emp Id From First Result Set	Emp Name From First Result Set	Phone Number From First Result Set
1	1	Name1	12345678
2	2	Name2	12345679
3	3	Name3	12345680
4	4	Name4	12345681

	Emp Id From Second Result Set	Emp Name From Second Result Set
1	1	Name1
2	3	Name3
3	5	Name5
4	7	Name7
5	9	Name9
6	11	Name11

В данном случае хранимая процедура возвращает два результирующих набора данных, но если мы попробуем обработать в With Result Set только один из них, то получим ошибку от SQL движка.

Msg 11535, Level 16, State 1, Procedure Usp_ModifiedFetchRecords, Line 11 EXECUTE statement failed because its WITH RESULT SETS clause specified 1 result set(s), and the statement tried to send more result sets than this.

Способ получения данных из With Result Set

Порой может потребоваться дополнительно обработать значение, полученное с помощью With Result Set. В таком случае можно использовать временные таблицы или переменный табличного типа.

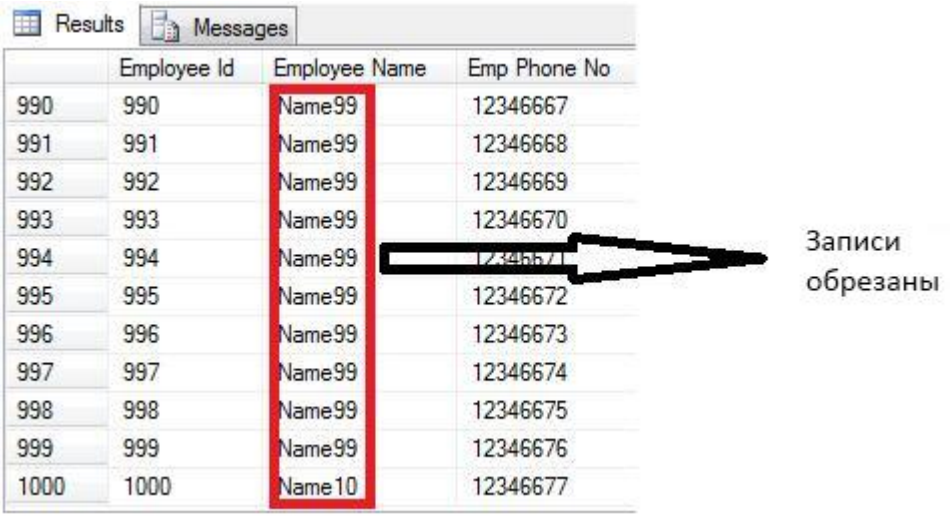
Рассмотрим подход с использованием переменных табличного типа.

```
Declare @tblStoreWithResultSetsData Table
(   [Employee Id] int
  , [Employee Name] varchar(50)
  , [Emp Phone No] int
)

insert into @tblStoreWithResultSetsData
EXEC Usp_FetchRecords
WITH RESULT SETS(
       (      [Emp Id] int,
              [Emp Name] varchar(6), -- подтверждение концепции,
                                     --  изменим длину строки до 6.
                                     -- запись будет обрезана
              [Phone Number] varchar(50)
       )
)

Select * From @tblStoreWithResultSetsData
```

Результат будет ожидаемым, имя работника сократиться до 6 символов. Это можно увидеть на следующем скриншоте (последние 10 записей)



Возможное применение:

1. *Преобразование данных в SSIS пакетах будет проще*, дополнительное описание смотрите в статье здесь.
2. *Изменение типов данных без изменения схемы*. Представьте что .Net приложение ожидает значение булевого типа, а значение в таблице закодировано типом int или

char(1). В общем можно применить конвертацию значений с помощью конструкции Case When Then Else. Но ведь проще и приятнее сразу изменить тип данных на bit (в случае с int).

3. Еще один пример приложения With Result Set, когда .Net программа ждет int, а в базе данных колонка имеет тип float.
4. *Возможная невосприимчивость DAL к изменениям схемы.* Имеется в виду положительная невосприимчивость, когда с помощью With Result Set задаем имена колонок для результирующего набора данных. Тогда будет неважно как имена меняются в самой базе. Эдакий аналог VIEW для хранимых процедур.

Ограничения:

Нельзя делать выборочное изменение колонок в итоговом наборе данных. Например следующий скрипт вызовет ошибку при выполнении:

```
EXEC Usp_FetchRecords
WITH RESULT SETS(
       (      [Emp Id] int,
              [Phone Number] varchar(50)
       )
)
```

Так как процедура возвращает набор из *трех* колонок. Ошибка будет такая:

Msg 11537, Level 16, State 1, Procedure Usp_FetchRecords, Line 5 EXECUTE statement failed because its WITH RESULT SETS clause specified 2 column(s) for result set number 1, but the statement sent 3 column(s) at run time.

Дальше будет интереснее, так что оставайтесь на связи. Перевод.

Hard'n'heavy!

Violet Tape