

# How to split a comma-separated value to columns



I have a table like this

114

Value	String
1	Cleo, Smith



I want to separate the comma delimited string into two columns

46

Value	Name	Surname
1	Cleo	Smith

I need only two fixed extra columns



sql-server

sql-server-2008

csv

edited Oct 1 '13 at 21:32



Charles

46.1k 12 88 125

asked May 14 '12 at 10:40



Gurru

640 2 9 20

possible duplicate of [How to split a single column values to multiple column values?](#) – Pondlife May 14 '12 at 10:43

## 33 Answers

1

2

next

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook

```
[column_id] INT IDENTITY(1, 1) NOT NULL,
[value] NVARCHAR(MAX)
)
```

AS

BEGIN

```
DECLARE @value NVARCHAR(MAX),
```

```
    @pos INT = 0,
```

```
    @len INT = 0
```

```
SET @string = CASE
```

```
    WHEN RIGHT(@string, 1) != @delimiter
```

```
    THEN @string + @delimiter
```

```
    ELSE @string
```

```
END
```

```
WHILE CHARINDEX(@delimiter, @string, @pos + 1) > 0
```

```
BEGIN
```

```
    SET @len = CHARINDEX(@delimiter, @string, @pos + 1) - @pos
```

```
    SET @value = SUBSTRING(@string, @pos, @len)
```

```
    INSERT INTO @out_put ([value])
```

```
    SELECT LTRIM(RTRIM(@value)) AS [column]
```

```
    SET @pos = CHARINDEX(@delimiter, @string, @pos + @len) + 1
```

```
END
```

```
RETURN
```

```
END
```

[edited Mar 1 at 0:47](#)
[answered Feb 28 at 13:34](#)

[Blixter](#)
**84** 9

Your purpose can be solved using following query -

**118**

```
Select Value , Substring(FullName, 1,Charindex(',', FullName)-1) as Name,
Substring(FullName, Charindex(',', FullName)+1, LEN(FullName)) as Surname
from Table1
```

Join **Stack Overflow** to learn, share knowledge, and build your career.

[Email Sign Up](#)
[OR SIGN IN WITH](#)


```

CREATE FUNCTION Split (
    @InputString          VARCHAR(8000),
    @Delimiter             VARCHAR(50)
)
RETURNS @Items TABLE (
    Item                  VARCHAR(8000)
)
AS
BEGIN
    IF @Delimiter = ''
    BEGIN
        SET @Delimiter = ','
        SET @InputString = REPLACE(@InputString, ' ', @Delimiter)
    END

    IF (@Delimiter IS NULL OR @Delimiter = '')
        SET @Delimiter = ','

    --INSERT INTO @Items VALUES (@Delimiter) -- Diagnostic
    --INSERT INTO @Items VALUES (@InputString) -- Diagnostic

    DECLARE @Item          VARCHAR(8000)
    DECLARE @ItemList      VARCHAR(8000)
    DECLARE @DelimIndex    INT

    SET @ItemList = @InputString
    SET @DelimIndex = CHARINDEX(@Delimiter, @ItemList, 0)
    WHILE (@DelimIndex != 0)
    BEGIN
        SET @Item = SUBSTRING(@ItemList, 0, @DelimIndex)
        INSERT INTO @Items VALUES (@Item)

        -- Set @ItemList = @ItemList minus one less item
        SET @ItemList = SUBSTRING(@ItemList, @DelimIndex+1, LEN(@ItemList)-
@DelimIndex)
        SET @DelimIndex = CHARINDEX(@Delimiter, @ItemList, 0)
    END -- End WHILE

    IF @Item IS NOT NULL -- At least one delimiter was encountered in @InputString
    BEGIN

```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



**RETURN**

```
END -- End Function
GO
```

```
---- Set Permissions
--GRANT SELECT ON Split TO UserRole1
--GRANT SELECT ON Split TO UserRole2
--GO
```

edited Sep 21 '16 at 22:31



Noctis

9,845 3 31 69

answered May 14 '12 at 10:43



Romil Kumar Jain

16.6k 7 43 81

1 Look at the numbers table solution DelimitedSplit8K by Jeff Moden in @ughai answer below too. – [Ruskin](#) Mar 6 '18 at 6:31

1 SQL 2016 now comes with a split function – [tvanharp](#) Jan 10 at 21:57



41



```
;WITH Split_Names (Value,Name, xmlname)
AS
(
    SELECT Value,
           Name,
           CONVERT(XML, '<Names><name>'
+ REPLACE(Name, ',', ' </name><name>') + '</name></Names>') AS xmlname
    FROM tblnames
)

SELECT Value,
xmlname.value('/Names[1]/name[1]', 'varchar(100)') AS Name,
xmlname.value('/Names[1]/name[2]', 'varchar(100)') AS Surname
FROM Split_Names
```

and also check the link below for reference

Join **Stack Overflow** to learn, share knowledge, and build your career.



Email Sign Up

OR SIGN IN WITH



Google



- 3 This is better.. it's simple and short. – [Kimchi Man](#) Oct 20 '14 at 18:55
- 2 I really love this way. CHARINDEX and SUBSTRING are a mess when you have more than 2 values to split (Eg. 1,2,3). Thanks a lot – [jotapdiez](#) Jun 3 '15 at 17:46 
- 2 Great idea. Three times as slow as the CHARINDEX plus SUBSTRING mess though, at least for me. :-( – [Michel de Ruiter](#) Nov 8 '16 at 12:38
- 2 Great solution however some characters are illegal in the XML (for example '&') so I had to wrap each field in a CDATA tag... CONVERT(XML, '<Names><name><![CDATA[' + REPLACE(Name, ',', ' ')></name><name><![CDATA[' + ' ]></name></Names>') AS xmlname – [Tony](#) Mar 28 '18 at 12:05 
- 1 @Tony needed to update the code from Tony to CONVERT(XML, '<Names><name><![CDATA[' + REPLACE(address1, ',', ' ')></name><name><![CDATA[' + ' ]></name></Names>') AS xmlname (Missing the final s on </Names>) – [Ryan Buddicom](#) Aug 16 '18 at 10:41

xml base answer is simple and clean

38

refer [this](#)

```
DECLARE @S varchar(max),
        @Split char(1),
        @X xml

SELECT @S = 'ab,cd,ef,gh,ij',
       @Split = ','

SELECT @X = CONVERT(xml, '<root> <myvalue>' +
REPLACE(@S,@Split,'</myvalue> <myvalue>') + '</myvalue> </root> ')

SELECT T.c.value('.', 'varchar(20)'),           --retrieve ALL values at once
       T.c.value('/root/myvalue)[1]', 'VARCHAR(20)' , --retrieve index 1 only, which is the
'ab'
       T.c.value('/root/myvalue)[2]', 'VARCHAR(20)',
       T.c.value('/root/myvalue)[3]', 'VARCHAR(20)'
FROM @X.nodes('/root/myvalue') T(c)
```

edited Feb 28 at 7:23

answered Jun 19 '13 at 7:32

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

 Google

Facebook 

I think this is cool

26

```
SELECT value,
    PARSENAME(REPLACE(String,',','.'),2) 'Name' ,
    PARSENAME(REPLACE(String,',','.'),1) 'Sur Name'
FROM table WITH (NOLOCK)
```

edited Dec 1 '14 at 15:13



Knerd

895 2 18 44

answered Jun 20 '14 at 5:39



Azar

1,442 8 15

4 yes, but PARSENAME wont go above 4 values. – [sifar786](#) Jul 7 '14 at 15:40

3 U r requirement is only for name and surname only na – [Azar](#) Jul 8 '14 at 3:49

Great performance! Beside that, it also supports nulls, – [Julio Nobre](#) Dec 6 '14 at 1:52

You also need to be aware that PARSENAME will return NULL for items longer than 128 chars. – [Luis Cazares](#) Feb 12 at 19:50

With CROSS APPLY

23

```
select ParsedData.*
from MyTable mt
cross apply ( select str = mt.String + ',,' ) f1
cross apply ( select p1 = charindex( ',', str ) ) ap1
cross apply ( select p2 = charindex( ',', str, p1 + 1 ) ) ap2
cross apply ( select Nname = substring( str, 1, p1-1 )
                , Surname = substring( str, p1+1, p2-p1-1 )
            ) ParsedData
```

answered May 13 '15 at 17:23



Lavisa

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook



15



There are multiple ways to solve this and many different ways have been proposed already. Simplest would be to use `LEFT` / `SUBSTRING` and other string functions to achieve the desired result.

### Sample Data

```
DECLARE @tbl1 TABLE (Value INT,String VARCHAR(MAX))

INSERT INTO @tbl1 VALUES(1,'Cleo, Smith');
INSERT INTO @tbl1 VALUES(2,'John, Mathew');
```

### Using String Functions like `LEFT`

```
SELECT
    Value,
    LEFT(String,CHARINDEX(',',String)-1) AS Fname,
    LTRIM(RIGHT(String,LEN(String) - CHARINDEX(',',String) )) AS Lname
FROM @tbl1
```

This approach fails if there are more 2 items in a String. In such a scenario, we can use a splitter and then use `PIVOT` or convert the string into an XML and use `.nodes` to get string items. XML based solution have been detailed out by aads and bvr in their solution.

The answers for this question which use splitter, all use `WHILE` which is inefficient for splitting. Check this [performance comparison](#). One of the best splitters around is `DelimitedSplit8K`, created by Jeff Moden. You can read more about it [here](#)

### Splitter with `PIVOT`

```
DECLARE @tbl1 TABLE (Value INT,String VARCHAR(MAX))

INSERT INTO @tbl1 VALUES(1,'Cleo, Smith');
INSERT INTO @tbl1 VALUES(2,'John, Mathew');
```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



## Output

Value	Fname	Lname
1	Cleo	Smith
2	John	Mathew

**DelimitedSplit8K** by Jeff Moden

**CREATE FUNCTION** [dbo].[DelimitedSplit8K]

*/\*\*\*\*\**

*Purpose:*

*Split a given string at a given delimiter and return a list of the split elements (items).*

*Notes:*

- 1. Leading a trailing delimiters are treated as if an empty string element were present.*
- 2. Consecutive delimiters are treated as if an empty string element were present between them.*
- 3. Except when spaces are used as a delimiter, all spaces present in each element are preserved.*

*Returns:*

*iTVF containing the following:*

*ItemNumber = Element position of Item as a BIGINT (not converted to INT to eliminate a CAST)*

*Item = Element value as a VARCHAR(8000)*

*Statistics on this function may be found at the following URL:*

*<http://www.sqlservercentral.com/Forums/Topic1101315-203-4.aspx>*

*CROSS APPLY Usage Examples and Tests:*

*--*

*=====*

*-- TEST 1:*

*-- This tests for various possible conditions in a string using a comma as the delimiter. The expected results are*

*-- laid out in the comments*

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

 Google

Facebook 



```

;
--===== Create and populate a test table on the fly (this is NOT a part of the
solution).
-- In the following comments, "b" is a blank and "E" is an element in the left to
right order.
-- Double Quotes are used to encapsulate the output of "Item" so that you can see
that all blanks
-- are preserved no matter where they may appear.
SELECT *
INTO #JBMTTest
FROM (
    SELECT 0, NULL UNION ALL --# & type of Return Row(s) --1 NULL
    SELECT 1, SPACE(0) UNION ALL --1 b (Empty String)
    SELECT 2, SPACE(1) UNION ALL --1 b (1 space)
    SELECT 3, SPACE(5) UNION ALL --1 b (5 spaces)
    SELECT 4, ',' UNION ALL --2 b b (both are empty strings)
    SELECT 5, '55555' UNION ALL --1 E
    SELECT 6, ',55555' UNION ALL --2 b E
    SELECT 7, ',55555,' UNION ALL --3 b E b
    SELECT 8, '55555,' UNION ALL --2 b B
    SELECT 9, '55555,1' UNION ALL --2 E E
    SELECT 10, '1,55555' UNION ALL --2 E E
    SELECT 11, '55555,4444,333,22,1' UNION ALL --5 E E E E E
    SELECT 12, '55555,4444,,333,22,1' UNION ALL --6 E E b E E E
    SELECT 13, ',55555,4444,,333,22,1,' UNION ALL --8 b E E b E E E b
    SELECT 14, ',55555,4444,,,333,22,1,' UNION ALL --9 b E E b b E E E b
    SELECT 15, ' 4444,55555 ' UNION ALL --2 E (w/Leading Space) E
    (w/Trailing Space)
    SELECT 16, 'This,is,a,test.' UNION ALL --E E E E
    ) d (SomeID, SomeValue)
;
--===== Split the CSV column for the whole table using CROSS APPLY (this is the
solution)
SELECT test.SomeID, test.SomeValue, split.ItemNumber, Item = QUOTENAME(split.Item, '')
FROM #JBMTTest test
CROSS APPLY dbo.DelimitedSplit8K(test.SomeValue, ',') split
;
--
=====

-- TEST 2:
-- This tests for various "alpha" splits and COLLATION using all ASCII characters from 0
to 255 as a delimiter against

```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

 Google

 Facebook

```
=====
WITH
cteBuildAllCharacters (String,Delimiter) AS
(
    SELECT TOP 256
        'ABCDEFGHJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789',
        CHAR(ROW_NUMBER() OVER (ORDER BY (SELECT NULL))-1)
    FROM master.sys.all_columns
)
SELECT ASCII_Value = ASCII(c.Delimiter), c.Delimiter, split.ItemNumber, Item =
QUOTENAME(split.Item, '')
    FROM cteBuildAllCharacters c
    CROSS APPLY dbo.DelimitedSplit8K(c.String,c.Delimiter) split
    ORDER BY ASCII_Value, split.ItemNumber
;
```

-----

Other Notes:

1. Optimized for VARCHAR(8000) or less. No testing or error reporting for truncation at 8000 characters is done.
2. Optimized for single character delimiter. Multi-character delimiters should be resolved externally from this function.
3. Optimized for use with CROSS APPLY.
4. Does not "trim" elements just in case leading or trailing blanks are intended.
5. If you don't know how a Tally table can be used to replace loops, please see the following...  
<http://www.sqlservercentral.com/articles/T-SQL/62867/>
6. Changing this function to use NVARCHAR(MAX) will cause it to run twice as slow. It's just the nature of VARCHAR(MAX) whether it fits in-row or not.
7. Multi-machine testing for the method of using UNPIVOT instead of 10 SELECT/UNION ALLs shows that the UNPIVOT method is quite machine dependent and can slow things down quite a bit.

-----

Credits:

This code is the product of many people's efforts including but not limited to the following:  
 cteTally concept originally by Iztek Ben Gan and "decimalized" by Lynn Pettis (and others) for a bit of extra speed  
 and finally redacted by Jeff Moden for a different slant on readability and

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

 Google

Facebook 

improvement over Rev 05. Special thanks to "Nadrek" and "peter-757102" (aka Peter de Heer) for bringing such improvements to light. Nadrek's original improvement brought about a 10% performance gain and Peter followed that up with the content of Rev 07.

I also thank whoever wrote the first article I ever saw on "numbers tables" which is located at the following URL and to Adam Machanic for leading me to it many years ago.  
<http://sqlserver2000.databases.aspfaq.com/why-should-i-consider-using-an-auxiliary-numbers-table.html>

-----  
Revision History:

Rev 00 - 20 Jan 2010 - Concept for inline cteTally: Lynn Pettis and others.  
Redaction/Implementation: Jeff Moden  
- Base 10 redaction and reduction for CTE. (Total rewrite)

Rev 01 - 13 Mar 2010 - Jeff Moden  
- Removed one additional concatenation and one subtraction from the SUBSTRING in the SELECT List for that tiny bit of extra speed.

Rev 02 - 14 Apr 2010 - Jeff Moden  
- No code changes. Added CROSS APPLY usage example to the header, some additional credits, and extra documentation.

Rev 03 - 18 Apr 2010 - Jeff Moden  
- No code changes. Added notes 7, 8, and 9 about certain "optimizations" that don't actually work for this type of function.

Rev 04 - 29 Jun 2010 - Jeff Moden  
- Added WITH SCHEMABINDING thanks to a note by Paul White. This prevents an unnecessary "Table Spool" when the function is used in an UPDATE statement even though the function makes no external references.

Rev 05 - 02 Apr 2011 - Jeff Moden  
- Rewritten for extreme performance improvement especially for larger strings approaching the 8K boundary and for strings that have wider elements. The redaction of this code involved

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



the final element (not followed by a delimiter) in the string to be split has been greatly simplified by using the ISNULL/NULLIF combination to determine when the CHARINDEX returned a 0 which indicates there are no more delimiters to be had or to start with. Depending on the width of the elements, this code is between 4 and 8 times faster on a single CPU box than the original code especially near the 8K boundary.

- Modified comments to include more sanity checks on the usage example, etc.
- Removed "other" notes 8 and 9 as they were no longer applicable.

Rev 06 - 12 Apr 2011 - Jeff Moden  
 - Based on a suggestion by Ron "Bitbucket" McCullough, additional test rows were added to the sample code and  
 the code was changed to encapsulate the output in pipes so that spaces and empty strings could be perceived  
 in the output. The first "Notes" section was added. Finally, an extra test was added to the comments above.

Rev 07 - 06 May 2011 - Peter de Heer, a further 15-20% performance enhancement has been discovered and incorporated into this code which also eliminated the need for a "zero" position in the cteTally table.

\*\*\*\*\*

```

----- Define I/O parameters
    (@pString VARCHAR(8000), @pDelimiter CHAR(1))
RETURNS TABLE WITH SCHEMABINDING AS
RETURN
----- "Inline" CTE Driven "Tally Table" produces values from 0 up to 10,000...
    -- enough to cover NVARCHAR(4000)
    WITH E1(N) AS (
        SELECT 1 UNION ALL SELECT 1 UNION ALL SELECT 1 UNION ALL
        SELECT 1 UNION ALL SELECT 1 UNION ALL SELECT 1 UNION ALL
        SELECT 1 UNION ALL SELECT 1 UNION ALL SELECT 1 UNION ALL SELECT 1
    ),
    E2(N) AS (SELECT 1 FROM E1 a, E1 b), --10E+2 or 100 rows
    E4(N) AS (SELECT 1 FROM E2 a, E2 b), --10E+4 or 10,000 rows max
cteTally(N) AS (----- This provides the "base" CTE and limits the number of rows right
up front
                -- for both a performance gain and prevention of accidental
"overruns"

```

```
SELECT TOP (TSNIII (DATALENGTH(@nString)-1)) ROW_NUMBER() OVER (ORDER BY
```

Join Stack Overflow to learn, share knowledge, and build your career.

```

@pDelimiter
),
cteLen(N1,L1) AS(----- Return start and Length (for use in substring)
    SELECT s.N1,
           ISNULL(NULLIF(CHARINDEX(@pDelimiter,@pString,s.N1),0)-s.N1,8000)
    FROM cteStart s
)
----- Do the actual split. The ISNULL(NULLIF combo handles the length for the final
element when no delimiter is found.
SELECT ItemNumber = ROW_NUMBER() OVER(ORDER BY l.N1),
       Item       = SUBSTRING(@pString, l.N1, l.L1)
FROM cteLen l
;

GO

```

answered Jun 11 '15 at 9:18



ughai

9,059 3 21 43

Try this (change instances of ' ' to ',' or whatever delimiter you want to use)

15

```

CREATE FUNCTION dbo.Wordparser
(
    @multiwordstring VARCHAR(255),
    @wordnumber      NUMERIC
)
returns VARCHAR(255)
AS
BEGIN
    DECLARE @remainingstring VARCHAR(255)
    SET @remainingstring=@multiwordstring

    DECLARE @numberofwords NUMERIC
    SET @numberofwords=(LEN(@remainingstring) - LEN(REPLACE(@remainingstring, ' ',

```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook

```

    word VARCHAR(255)
)

WHILE @numberofwords > 1
BEGIN
    SET @word=LEFT(@remainingstring, CHARINDEX(' ', @remainingstring) - 1)

    INSERT INTO @parsedwords(word)
    SELECT @word

    SET @remainingstring= REPLACE(@remainingstring, Concat(@word, ' '), '')
    SET @numberofwords=(LEN(@remainingstring) - LEN(REPLACE(@remainingstring, '
', ''))) + 1)

    IF @numberofwords = 1
        BREAK

    ELSE
        CONTINUE
END

IF @numberofwords = 1
    SELECT @word = @remainingstring
INSERT INTO @parsedwords(word)
SELECT @word

RETURN
(SELECT word
 FROM @parsedwords
 WHERE line = @wordnumber)

END

```

Example usage:

```

SELECT dbo.Wordparser(COLUMN, 1),
       dbo.Wordparser(COLUMN, 2),
       dbo.Wordparser(COLUMN, 3)
FROM   TABLE

```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH





10



I think PARSENAME is the neat function to use for this example, as described in this article: <http://www.sqlshack.com/parsing-and-rotating-delimited-data-in-sql-server-2012/>

The PARSENAME function is logically designed to parse four-part object names. The nice thing about PARSENAME is that it's not limited to parsing just SQL Server four-part object names – it will parse any function or string data that is delimited by dots.

The first parameter is the object to parse, and the second is the integer value of the object piece to return. The article is discussing parsing and rotating delimited data - company phone numbers, but it can be used to parse name/surname data also.

Example:

```
USE COMPANY;  
SELECT PARSENAME('Whatever.you.want.parsed',3) AS 'ReturnValue';
```

The article also describes using a Common Table Expression (CTE) called 'replaceChars', to run PARSENAME against the delimiter-replaced values. A CTE is useful for returning a temporary view or result set.

After that, the UNPIVOT function has been used to convert some columns into rows; SUBSTRING and CHARINDEX functions have been used for cleaning up the inconsistencies in the data, and the LAG function (new for SQL Server 2012) has been used in the end, as it allows referencing of previous records.

answered Dec 23 '15 at 9:51



JasonP

199 2 3



10



With SQL Server 2016 we can use string\_split to accomplish this:

```
create table commasep (  
  id int identity(1,1)  
  ,string nvarchar(100) )
```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



answered Sep 7 '16 at 18:20



Kannan Kandasamy

9,654 2 13 28

I am using SQL Server 2016 but it gives an error Invalid object name 'string\_split' – [ibiza](#) Dec 14 '16 at 16:28

- 1 Can you check the compatibility level of your database? It must be 130 which is sql server 2016. You can use this query select \* from sys.databases – [Kannan Kandasamy](#) Dec 14 '16 at 19:54

right, I see 120 so it must be only the client (Microsoft SQL Server Management Studio) that is 2016 and not the database server per se because if I go to Help -> About, I see SQL Server 2016 Management Studio v13.0.15000.23. Thanks – [ibiza](#) Dec 14 '16 at 20:16

We can create a function as this

9

```
CREATE Function [dbo].[fn_CSVToTable]
(
    @CSVList Varchar(max)
)
RETURNS @Table TABLE (ColumnData VARCHAR(100))
AS
BEGIN
    IF RIGHT(@CSVList, 1) <> ','
    SELECT @CSVList = @CSVList + ','

    DECLARE @Pos BIGINT,
            @OldPos BIGINT
    SELECT @Pos = 1,
           @OldPos = 1

    WHILE @Pos < LEN(@CSVList)
    BEGIN
        SELECT @Pos = CHARINDEX(',', @CSVList, @OldPos)
        INSERT INTO @Table
        SELECT LTRIM(RTRIM(SUBSTRING(@CSVList, @OldPos, @Pos - @OldPos))) Col001

        SELECT @OldPos = @Pos + 1
    END
END
```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google





edited Mar 26 '14 at 11:03



Lucas Zamboulis

2,431 5 20 26

answered Mar 26 '14 at 10:42



Himansz

180 2 12

I think following function will work for you:

7

**You have to create a function in SQL first. Like this**

```
CREATE FUNCTION [dbo].[fn_split](
@str VARCHAR(MAX),
@delimiter CHAR(1)
)
RETURNS @returnTable TABLE (idx INT PRIMARY KEY IDENTITY, item VARCHAR(8000))
AS
BEGIN
DECLARE @pos INT
SELECT @str = @str + @delimiter
WHILE LEN(@str) > 0
BEGIN
SELECT @pos = CHARINDEX(@delimiter,@str)
IF @pos = 1
INSERT @returnTable (item)
VALUES (NULL)
ELSE
INSERT @returnTable (item)
VALUES (SUBSTRING(@str, 1, @pos-1))
SELECT @str = SUBSTRING(@str, @pos+1, LEN(@str)-@pos)
END
RETURN
END
```

You can call this function, like this:

```
select * from fn_split('1,24,5','(',')')
```

Join **Stack Overflow** to learn, share knowledge, and build your career.

[Email Sign Up](#)

OR SIGN IN WITH



Google



```
Data VARCHAR(200)
)

insert into @test
(ID, Data)
Values
('1', 'Cleo,Smith')

insert into @test
(ID, Data)
Values
('2', 'Paul,Grim')

select ID,
(select item from fn_split(Data,',') where idx in (1)) as Name ,
(select item from fn_split(Data,',') where idx in (2)) as Surname
from @test
```

## Result will like this:

	ID	Name	Surname
1	1	Cleo	Smith
2	2	Paul	Grim

answered Aug 28 '17 at 7:32



[Muhammad Awais](#)

2,161 1 27 22

Using loops to split string is horribly inefficient. Here are several better options for that split function. [sqlperformance.com/2012/07/t-sql-queries/split-strings](https://sqlperformance.com/2012/07/t-sql-queries/split-strings) – [Sean Lange](#) Oct 4 '18 at 16:22



Use Parsename() function

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



```

Union
select 'Bab,Karimi' as FullName
)

SELECT PARSENAME(REPLACE(FullName,',','.'),2) as Name,
        PARSENAME(REPLACE(FullName,',','.'),1) as Family
FROM cte

```

## Result

Name	Family
Aria	Karimi
Bab	Karimi
Joe	Karimi

answered Mar 18 '15 at 4:36



Mirak

3,231 2 16 16

6

```

SELECT id,
        Substring(NAME, 0, Charindex(',', NAME)) AS firstname,
        Substring(NAME, Charindex(',', NAME), Len(NAME) + 1) AS lastname
FROM spilt

```

edited Apr 12 at 17:44



DxTx

1,857 1 9 22

answered Apr 14 '15 at 18:42



anonymous

85 1 1

6 It would be useful if you could expand on your answer, and use the code formatting tools as well. – Politank-Z Apr 14 '15 at 18:59

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook

```

with cte as
(
    select SUBSTRING(@csv,1,charindex(',',@csv,1)-1) as val,
    SUBSTRING(@csv,charindex(',',@csv,1)+1,len(@csv)) as rem
    UNION ALL
    select SUBSTRING(a.rem,1,charindex(',',a.rem,1)-1) as val,
    SUBSTRING(a.rem,charindex(',',a.rem,1)+1,len(a.rem))
    from cte a where LEN(a.rem)>=1
) select val from cte

```

edited May 26 '16 at 6:25



Sampada

2,478 7 17 34

answered Mar 14 '16 at 11:47



Rangani

146 1 5

---

Work like a charm! – [yu yang Jian](#) Jan 15 at 8:34

---

Using instring function :)

4

```

select Value,
    substring(String,1,instr(String," ") -1) Fname,
    substring(String,instr(String," ") +1) Sname
from tablename;

```

Used two functions,

1. substring(string, position, length) ==> returns string from position to length
2. instr(string,pattern) ==> returns position of pattern.

If we don't provide length argument in substring it returns until end of string

answered Sep 30 '15 at 8:36



WoodChopper

2,242 3 21 39

---

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook

3

I encountered a similar problem but a complex one and since this is the first thread i found regarding that issue i decided to post my finding. i know it is complex solution to a simple problem but i hope that i could help other people who go to this thread looking for a more complex solution. i had to split a string containing 5 numbers (column name: levelsFeed) and to show each number in a separate column. for example: 8,1,2,2,2 should be shown as :

1	2	3	4	5
-----				
8	1	2	2	2

Solution 1: using XML functions: this solution for the slowest solution by far

```
SELECT Distinct FeedbackID,
, S.a.value('/H/r)[1]', 'INT') AS level1
, S.a.value('/H/r)[2]', 'INT') AS level2
, S.a.value('/H/r)[3]', 'INT') AS level3
, S.a.value('/H/r)[4]', 'INT') AS level4
, S.a.value('/H/r)[5]', 'INT') AS level5
FROM (
    SELECT *,CAST (N'<H><r>' + REPLACE(levelsFeed, ',', '</r><r>') + '</r> </H>' AS
XML) AS [vals]
    FROM Feedbacks
) as d
CROSS APPLY d.[vals].nodes('/H/r') S(a)
```

Solution 2: using Split function and pivot. (the split function split a string to rows with the column name Data)

```
SELECT FeedbackID, [1],[2],[3],[4],[5]
FROM (
    SELECT *, ROW_NUMBER() OVER (PARTITION BY feedbackID ORDER BY (SELECT null)) as rn
    FROM (
        SELECT FeedbackID, levelsFeed
        FROM Feedbacks
    ) as a
    CROSS APPLY dbo.Split(levelsFeed, ',')
) as SourceTable
PIVOT
(
    MAX(data)
    FOR data IN [1],[2],[3],[4],[5])
```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

 Google

Facebook 

```

SELECT FeedbackID,
SUBSTRING(levelsFeed,0,CHARINDEX(',',levelsFeed)) AS level1,
PARSENAME(REPLACE(SUBSTRING(levelsFeed,CHARINDEX(',',levelsFeed)+1,LEN(levelsFeed)),',','').
AS level2,
PARSENAME(REPLACE(SUBSTRING(levelsFeed,CHARINDEX(',',levelsFeed)+1,LEN(levelsFeed)),',','').
AS level3,
PARSENAME(REPLACE(SUBSTRING(levelsFeed,CHARINDEX(',',levelsFeed)+1,LEN(levelsFeed)),',','').
AS level4,
PARSENAME(REPLACE(SUBSTRING(levelsFeed,CHARINDEX(',',levelsFeed)+1,LEN(levelsFeed)),',','').
AS level5
FROM Feedbacks

```

since the levelsFeed contains 5 string values i needed to use the substring function for the first string.

i hope that my solution will help other that got to this thread looking for a more complex split to columns methods

answered Jul 14 '15 at 9:42



Yossi  
34 6

3

```

DECLARE @INPUT VARCHAR (MAX)='N,A,R,E,N,D,R,A'
DECLARE @ELIMINATE_CHAR CHAR (1)=','
DECLARE @L_START INT=1
DECLARE @L_END INT=(SELECT LEN (@INPUT))
DECLARE @OUTPUT CHAR (1)

WHILE @L_START <=@L_END
BEGIN
SET @OUTPUT=(SUBSTRING (@INPUT,@L_START,1))
IF @OUTPUT!=@ELIMINATE_CHAR
BEGIN
PRINT @OUTPUT
END
SET @L_START=@L_START+1

```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



---

I used your code, it's simple but there are spelling errors in ELIMINATE\_CHAT it should be ELIMINATE\_CHAR and START AT the end of the script should be L\_START. thank you. – [Wessam El Mahdy](#) Mar 6 '17 at 21:47

---



You can use an inbuilt `STRING_SPLIT` function which is available only under compatibility level 130. If your database compatibility level is lower than 130, SQL Server will not be able to find and execute `STRING_SPLIT` function. You can change a compatibility level of database using the following command:



```
ALTER DATABASE DatabaseName SET COMPATIBILITY_LEVEL = 130
```

Syntax

```
STRING_SPLIT ( string , separator )
```

[see documentation here](#)

answered Jul 9 '17 at 8:54



[bwanamaina](#)

86 1 8

---

Nice. But it does not apply to SQL Server less than 2016 – [Lokesh](#) Dec 5 '18 at 6:16

---

True, in my answer i indicated that it will only be available in compatibility level 130 and later. – [bwanamaina](#) Dec 8 '18 at 9:38

---



This function is most fast:

**3**

```
CREATE FUNCTION dbo.F_ExtractSubString  
(  
    @String VARCHAR(MAX),
```

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



```

SET @String = @String + @Separator
WHILE CHARINDEX(@Separator, @String, @End + 1) > 0 AND @NroSubString > 0
BEGIN
    SET @St = @End + 1
    SET @End = CHARINDEX(@Separator, @String, @End + 1)
    SET @NroSubString = @NroSubString - 1
END
IF @NroSubString > 0
    SET @Ret = ''
ELSE
    SET @Ret = SUBSTRING(@String, @St, @End - @St)
RETURN @Ret
END
GO

```

Example usage:

```

SELECT dbo.F_ExtractSubString(COLUMN, 1, ',', ''),
       dbo.F_ExtractSubString(COLUMN, 2, ',', ''),
       dbo.F_ExtractSubString(COLUMN, 3, ',', ')
FROM   TABLE

```

answered Dec 19 '17 at 14:55



[Mariano Sedano](#)

39 4

- 2 Thank you for this code snippet, which might provide some limited, immediate help. A proper explanation [would greatly improve](#) its long-term value by showing *why* this is a good solution to the problem, and would make it more useful to future readers with other, similar questions. Please [edit](#) your answer to add some explanation, including the assumptions you've made. – [Toby Speight](#) Dec 19 '17 at 15:24

mytable:

2

Value ColOne

1 ColOne Smith

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google





```

ALTER TABLE mytable ADD ColTwo nvarchar(256);
UPDATE mytable SET ColTwo = LEFT(ColOne, Charindex(',', ColOne) - 1);
--'Cleo' = LEFT('Cleo, Smith', Charindex(',', 'Cleo, Smith') - 1)
UPDATE mytable SET ColTwo = REPLACE(ColOne, ColTwo + ',', '');
--' Smith' = REPLACE('Cleo, Smith', 'Cleo' + ',')
UPDATE mytable SET ColOne = REPLACE(ColOne, ',' + ColTwo, ''), ColTwo = LTRIM(ColTwo);
--'Cleo' = REPLACE('Cleo, Smith', ',' + ' Smith', '')

```

Result:

Value	ColOne	ColTwo
1	Cleo	Smith

edited Dec 29 '14 at 22:39

answered Dec 29 '14 at 22:30



[kolunar](#)

2,192 1 20 33

it is so easy, you can take it by below query:

2

```

DECLARE @str NVARCHAR(MAX)='ControlID_05436b78-04ba-9667-fa01-9ff8c1b7c235,3'
SELECT LEFT(@str, CHARINDEX(',',@str)-1),RIGHT(@str,LEN(@str)-(CHARINDEX(',',@str)))

```

answered Mar 16 '16 at 7:40



[Mehdi najafian](#)

21 3

This worked for me

2

```

CREATE FUNCTION [dbo].[SplitString](

```

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook

```

SET @xml = N'<t>' + REPLACE(@delimited,@delimiter,'</t><t>') + '</t>'
INSERT INTO @t(val)
SELECT r.value('.', 'varchar(MAX)') as item
FROM @xml.nodes('/t') as records(r)
RETURN
END

```

answered Jan 7 '17 at 14:31



Krishna

2,500 1 16 14

---

do you know how to handle xml special chars? – [Zach Smith](#) Aug 17 '17 at 12:44

---

You may find the solution in [SQL User Defined Function to Parse a Delimited String](#) helpful (from [The Code Project](#)).

2

This is the code part from this page:

```

CREATE FUNCTION [fn_ParseText2Table]
    (@p_SourceText VARCHAR(MAX)
    ,@p_Delimiter VARCHAR(100)=';' --default to comma delimited.
    )
RETURNS @retTable
    TABLE([Position] INT IDENTITY(1,1)
    ,[Int_Value] INT
    ,[Num_Value] NUMERIC(18,3)
    ,[Txt_Value] VARCHAR(MAX)
    ,[Date_value] DATETIME
    )
AS
/*
*****
Purpose: Parse values from a delimited string
        & return the result as an indexed table
Copyright 1996, 1997, 2000, 2003 Clayton Groom (<A
href="mailto:Clayton_Groom@hotmail.com">Clayton_Groom@hotmail.com</A>)
Posted to the public domain Aug, 2004

```

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



\*/

```

BEGIN
DECLARE @w_xml xml;
SET @w_xml = N'<root><i>' + replace(@p_SourceText, @p_Delimiter, '</i><i>') + '</i>
</root>';

INSERT INTO @retTable
    ([Int_Value]
    , [Num_Value]
    , [Txt_Value]
    , [Date_value]
    )
SELECT CASE
    WHEN ISNUMERIC([i].value('.', 'VARCHAR(MAX)')) = 1
    THEN CAST(CAST([i].value('.', 'VARCHAR(MAX)') AS NUMERIC) AS INT)
    END AS [Int_Value]
, CASE
    WHEN ISNUMERIC([i].value('.', 'VARCHAR(MAX)')) = 1
    THEN CAST([i].value('.', 'VARCHAR(MAX)') AS NUMERIC(18, 3))
    END AS [Num_Value]
, [i].value('.', 'VARCHAR(MAX)') AS [txt_Value]
, CASE
    WHEN ISDATE([i].value('.', 'VARCHAR(MAX)')) = 1
    THEN CAST([i].value('.', 'VARCHAR(MAX)') AS DATETIME)
    END AS [Date_value]
FROM @w_xml.nodes('//root/i') AS [Items]([i]);
RETURN;
END;
GO

```

edited Oct 5 '17 at 13:10



Tamas Rev

5,178 4 23 45

answered May 14 '12 at 10:43



Michael Schnerring

2,217 3 15 37

- 11 Any chance you could summarize the solution here to make sure the answer doesn't become obsolete if the link ever dies? – Adam Lear ♦ Sep 18 '13 at 15:06

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



Facebook

2

**AS Begin**

```
--Declare @delimiter varchar(1)=',',@occurence int=2,@String varchar(100)='a,b,c'
```

```
Declare @result int
```

```
;with T as (
    select 1 Rno,0 as row, charindex(@delimiter, @String) pos,@String st
    union all
    select Rno+1,pos + 1, charindex(@delimiter, @String, pos + 1), @String
    from T
    where pos > 0
)
```

```
select @result=pos
from T
where pos > 0 and rno = @occurence
return isnull(@result,0)
END
```

```
declare @data as table (data varchar(100))
insert into @data values('1,2,3')
insert into @data values('aaa,bbbb,cccc')
select top 3 Substring (data,0,dbo.get_occurrence_index( ',',1,data)) ,--First Record
always starts with 0
Substring (data,dbo.get_occurrence_index( ',',1,data)+1,dbo.get_occurrence_index(
',',2,data)-dbo.get_occurrence_index( ',',1,data)-1) ,
Substring (data,dbo.get_occurrence_index( ',',2,data)+1,len(data)) , -- Last record cant
be more than len of actual data
data
From @data
```

answered May 22 '18 at 9:17



vignesh

620 2 7 26

I found that using PARSENAME as above caused any name with a period to get nulled.

1

So if there was an initial or a title in the name followed by a dot they return NULL.

I found this worked for me:

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



Facebook

Surname  
FROM Table1

edited Apr 28 '15 at 3:23

answered Apr 28 '15 at 3:17



RoadRunner

21 3

1

```
select distinct modelFileId,F4.*
from contract
cross apply (select XmlList=convert(xml, '<x>'+replace(modelFileId,';', '</x>
<x>')+ '</x>').query('.')) F2
cross apply (select mfid1=XmlNode.value('/x[1]', 'varchar(512)')
,mfid2=XmlNode.value('/x[2]', 'varchar(512)')
,mfid3=XmlNode.value('/x[3]', 'varchar(512)')
,mfid4=XmlNode.value('/x[4]', 'varchar(512)') from XmlList.nodes('x') F3(XmlNode)) F4
where modelFileId like '%;%
order by modelFileId
```

answered Dec 8 '16 at 18:51



Frank

304 2 9

1

```
Select distinct PROJ_UID,PROJ_NAME,RES_UID from E2E_ProjectWiseTimesheetActuals
where CHARINDEX(','+cast(PROJ_UID as varchar(8000))+',' , @params) > 0 and
CHARINDEX(','+cast(RES_UID as varchar(8000))+',' , @res) > 0
```

edited Mar 8 '17 at 19:20

answered Mar 8 '17 at 13:28



EJoshuaS

7,564 10 31 51



user7678586

11 1

4 While this code may answer the question, providing additional context regarding why and/or how this code answers the question improves its long-term

Join Stack Overflow to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



Facebook



```

CREATE FUNCTION [dbo].[fnSplit](@sInputList VARCHAR(8000), @sDelimiter VARCHAR(8000) =
','')
RETURNS @List TABLE (item VARCHAR(8000))
BEGIN

    DECLARE @sItem VARCHAR(8000)
    WHILE CHARINDEX(@sDelimiter, @sInputList, 0) <> 0
    BEGIN

        SELECT @sItem = RTRIM(LTRIM(SUBSTRING(@sInputList, 1, CHARINDEX(@sDelimiter,
@sInputList, 0) - 1))),
        @sInputList = RTRIM(LTRIM(SUBSTRING(@sInputList, CHARINDEX(@sDelimiter,
@sInputList, 0) + LEN(@sDelimiter), LEN(@sInputList))))

        -- Indexes to keep the position of searching
        IF LEN(@sItem) > 0

            INSERT INTO @List SELECT @sItem

    END

    IF LEN(@sInputList) > 0
    BEGIN

        INSERT INTO @List SELECT @sInputList -- Put the last item in

    END

    RETURN

END

```

edited Oct 16 '15 at 10:17



Sk93

2,685 3 28 59

answered Oct 16 '15 at 9:29



Glyn

9 2

▲ You can use split function.

Join **Stack Overflow** to learn, share knowledge, and build your career.

[Email Sign Up](#)
[OR SIGN IN WITH](#)

[Google](#)
[Facebook](#)

answered Jan 9 '17 at 12:30



vicky

1,034

13

25

that is not a built in function. we need to create a function Split in that DB. – [ashveli](#) Oct 17 '17 at 9:42

1

2

next

**protected by Community ♦** Aug 24 '18 at 6:13

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 [reputation](#) on this site (the [association bonus](#) does not count).

Would you like to answer one of these [unanswered questions](#) instead?

Join **Stack Overflow** to learn, share knowledge, and build your career.

[Email Sign Up](#)

OR SIGN IN WITH



Google

[Facebook](#)