How to concatenate text from multiple rows into a single text string in SQL server?

1709

Consider a database table holding names, with three rows:

```
Peter
Paul
Mary
```

535

Is there an easy way to turn this into a single string of `Peter, Paul, Mary` ?

sql | sql-server | csv | string-concatenation | group-concat

edited Feb 19 '18 at 20:41
Steve Chambers
22.7k 11 102 146

asked Oct 11 '08 at 23:49
JohnnyM
11.3k 10 30 35

22 For answers specific to SQL Server, try this question. – Matt Hamilton Oct 12 '08 at 0:03

15 For MySQL, check out Group_Concat from this answer – Pykler May 6 '11 at 19:48

24 I wish the next version of SQL Server would offer a new feature to solve multi-row string concatination elegantly without the silliness of FOR XML PATH. – Pete Alvin Oct 2 '14 at 11:47

3 Not SQL, but if this is a once-only thing, you can paste the list into this in-browser tool convert.town/column-to-comma-separated-list – Stack Man May 27 '15 at 7:56

1 In Oracle you can use the LISTAGG(COLUMN_NAME) from 11g r2 before that there is an unsupported function called WM_CONCAT(COLUMN_NAME) which does the same. – Richard Jul 6 '17 at 6:32

45 Answers

If you are on SQL Server 2017 or Azure, see Mathieu Renda answer.

1261

I had a similar issue when I was trying to join two tables with one-to-many relationships. In SQL 2005 I found that `XML PATH` method can handle the concatenation of the rows very easily.

If there is a table called `STUDENTS`

```
SubjectID       StudentName
----------      -------------
1               Mary
1               John
1               Sam
2               Alaina
2               Edward
```

Result I expected was:

```
SubjectID       StudentName
----------      -------------
1               Mary, John, Sam
2               Alaina, Edward
```

I used the following `T-SQL` :

```
SELECT Main.SubjectID,
       LEFT(Main.Students,Len(Main.Students)-1) As "Students"
FROM
    (
        SELECT DISTINCT ST2.SubjectID,
            (
                SELECT ST1.StudentName + ',' AS [text()]
                FROM dbo.Students ST1
                WHERE ST1.SubjectID = ST2.SubjectID
                ORDER BY ST1.SubjectID
                FOR XML PATH ('')
            ) [Students]
        FROM dbo.Students ST2
    ) [Main]
```

```
SELECT DISTINCT ST2.SubjectID,
    SUBSTRING(
        (
            SELECT ','+ST1.StudentName  AS [text()]
            FROM dbo.Students ST1
            WHERE ST1.SubjectID = ST2.SubjectID
            ORDER BY ST1.SubjectID
            FOR XML PATH ('')
        ), 2, 1000) [Students]
FROM dbo.Students ST2
```

edited Aug 2 '18 at 11:20

StefanJCollier
1,080 11 20

answered Feb 13 '09 at 11:53
Ritesh

11 Great solution. The following may be helpful if you need to handle special characters like those in HTML: Rob Farley: Handling special characters with FOR XML PATH('). – user140628 Apr 17 '13 at 12:35

8 Apparently this doesn't work if the names contain XML characters such as `<` or `&` . See @BenHinman's comment. – Sam Aug 13 '13 at 1:26

22 NB: This method is reliant on undocumented behavior of `FOR XML PATH ('')` . That means it should not be considered reliable as any patch or update could alter how this functions. It's basically relying on a deprecated feature. – Bacon Bits Nov 13 '14 at 18:54

24 @Whelkaholism The bottom line is that `FOR XML` is intended to generate XML, not concatenate arbitrary strings. That's why it escapes `&` , `<` and `>` to XML entity codes (`&` , `<` , `>`). I assume it also will escape `"` and `'` to `&quot;` and `&apos;` in attributes as well. It's *not* `GROUP_CONCAT()` , `string_agg()` , `array_agg()` , `listagg()` , etc. even if you can kind of make it do that. We *should* be spending our time demanding Microsoft implement a proper function. – Bacon Bits Mar 23 '15 at 14:15

11 Good news: MS SQL Server will be adding `string_agg` in v.Next. and all of this can go away. – Jason C Apr 6 '17 at 0:32

952

This answer may return unexpected results For consistent results, use one of the FOR XML PATH methods detailed in other answers.

Use `COALESCE` :

Just some explanation (since this answer seems to get relatively regular views):

- Coalesce is really just a helpful cheat that accomplishes two things:

1) No need to initialize `@Names` with an empty string value.

2) No need to strip off an extra separator at the end.

- The solution above will give incorrect results if a row has a *NULL* Name value (if there is a *NULL*, the *NULL* will make `@Names` *NULL* after that row, and the next row will start over as an empty string again. Easily fixed with one of two solutions:

```
DECLARE @Names VARCHAR(8000)
SELECT @Names = COALESCE(@Names + ', ', '') + Name
FROM People
WHERE Name IS NOT NULL
```

or:

```
DECLARE @Names VARCHAR(8000)
SELECT @Names = COALESCE(@Names + ', ', '') +
    ISNULL(Name, 'N/A')
FROM People
```

Depending on what behavior you want (the first option just filters *NULL*s out, the second option keeps them in the list with a marker message [replace 'N/A' with whatever is appropriate for you]).

edited Apr 30 at 12:02
Martin Smith
356k 60 596 709

answered Oct 12 '08 at 0:18

Chris Shaffer
28.7k 4 41 60

67 To be clear, coalesce has nothing to do with creating the list, it just makes sure that NULL values are not included. – Graeme Perrow Feb 13 '09 at 12:02

16 @Graeme Perrow It doesn't exclude NULL values (a WHERE is required for that -- this will *lose results* if one of the input values is NULL), and it *is required in this approach* because: NULL + non-NULL -> NULL and non-NULL + NULL -> NULL; also @Name is NULL by default and, in fact, that property is used as an implicit sentinel here to determine if a ', ' should be added or not – user166390 Aug 15 '10 at 18:57

offered by @Ritesh – R. Schreurs Aug 2 '13 at 8:10

10 This is not a reliable method of concatenation. It is unsupported and should not be used (per Microsoft, e.g. support.microsoft.com/en-us/kb/287515, connect.microsoft.com/SQLServer/Feedback/Details/704389). It can change without warning. Use the XML PATH technique discussed in stackoverflow.com/questions/5031204/… I wrote more here: marc.durdin.net/2015/07/… – Marc Durdin Jul 15 '15 at 0:23

336

One method not yet shown via the `XML data()` command in MS SQL Server is:

Assume table called NameList with one column called FName,

```
SELECT FName + ', ' AS 'data()'
FROM NameList
FOR XML PATH('')
```

returns:

```
"Peter, Paul, Mary, "
```

Only the extra comma must be dealt with.

Edit: As adopted from @NReilingh's comment, you can use the following method to remove the trailing comma. Assuming the same table and column names:

```
STUFF(REPLACE((SELECT '#!' + LTRIM(RTRIM(FName)) AS 'data()' FROM NameList
FOR XML PATH('')),' #!',', '), 1, 2, '') as Brands
```

edited Apr 25 '16 at 15:28
user1477388
13.6k 21 102 206

answered Apr 5 '11 at 21:19
jens frandsen
3,369 1 10 2

13 holy s**t thats amazing! When executed on its own, as in your example the result is formatted as a hyperlink, that when clicked (in SSMS) opens a new window containing the data, but when used as part of a larger query it just appears as a string. Is it a string? or is it xml that i need to treat differently in

'14 at 22:40

6 @Baodad That appears to be part of the deal. You can workaround by replacing on an added token character. For example, this does a perfect comma-delimited list for any length: `SELECT STUFF(REPLACE((SELECT '#!'+city AS 'data()' FROM #cityzip FOR XML PATH ('')),' #!',', '),1,2,'') –` NReilingh Feb 29 '16 at 18:12

1 Wow, actually in my testing using data() and a replace is WAY more performant than not. Super weird. – NReilingh Feb 29 '16 at 18:33

293

+150

SQL Server 2017+ and SQL Azure: STRING_AGG

Starting with the next version of SQL Server, we can finally concatenate across rows without having to resort to any variable or XML witchery.

STRING_AGG (Transact-SQL)

Without grouping

```
SELECT STRING_AGG(Name, ', ') AS Departments
FROM HumanResources.Department;
```

With grouping :

```
SELECT GroupName, STRING_AGG(Name, ', ') AS Departments
FROM HumanResources.Department
GROUP BY GroupName;
```

With grouping and sub-sorting

```
SELECT GroupName, STRING_AGG(Name, ', ') WITHIN GROUP (ORDER BY Name ASC) AS Departments
FROM HumanResources.Department
GROUP BY GroupName;
```

2 This works with SQL Azure. Great answer! – user2721607 Oct 11 '17 at 20:27

2 This also worked for me in Azure SQL. Brilliant! – Kevin Stone Jan 4 '18 at 20:31

270

In SQL Server 2005

```
SELECT Stuff(
  (SELECT N', ' + Name FROM Names FOR XML PATH(''),TYPE)
  .value('text()[1]','nvarchar(max)'),1,2,N'')
```

In SQL Server 2016

you can use the FOR JSON syntax

i.e.

```
SELECT per.ID,
Emails = JSON_VALUE(
   REPLACE(
     (SELECT _ = em.Email FROM Email em WHERE em.Person = per.ID FOR JSON PATH)
    ,'"},{"_":"',', '),'$[0]._'
)
FROM Person per
```

And the result will become

```
Id  Emails
1   abc@gmail.com
2   NULL
3   def@gmail.com, xyz@gmail.com
```

And in SQL Server 2017, Azure SQL Database

You can use the new STRING_AGG function

edited Jul 23 '18 at 16:55

community wiki
9 revs, 7 users 70%
Steven Chong

3 Good use of the STUFF function to nix the leading two characters. – David Aug 11 '11 at 23:12

3 I like this solution best, because I can easily use it in a select list by appending 'as <label>'. I am not sure how to do this with the solution of @Ritesh. – R. Schreurs Aug 2 '13 at 8:27

12 This is better than the accepted answer because this option also handles un-escaping XML reserverd characters such as `<`, `>`, `&`, etc. which `FOR XML PATH('')` will automatically escape. – BateTech Apr 7 '14 at 21:35

This is an awesome response as it resolved the issue and provides the best ways of doing things in different versions of SQL now I wish I could use 2017/Azure – Chris Ward May 21 '18 at 14:27

111

In MySQL there is a function, GROUP_CONCAT(), which allows you to concatenate the values from multiple rows. Example:

```
SELECT 1 AS a, GROUP_CONCAT(name ORDER BY name ASC SEPARATOR ', ') AS people
FROM users
WHERE id IN (1,2,3)
GROUP BY a
```

edited Feb 26 '12 at 18:37

Peter Mortensen
14.1k 19 88 114

answered Oct 12 '08 at 0:10

Darryl Hein
70.6k 83 191 245

2 Used to love this one, have not seen a alternative to this function with any other Db yet! – Binoj Antony Jun 18 '09 at 6:05

1 This totally solved my problem. I was trying to pull all the payment dates for a given charge on an account, this solved it perfectly. Thanks! – Maximus

@DarrylHein for my needs i used the separator as above. But this cuts me some chars at the very end of the output. This is very strange and seems to be a bug. I dont have a solution, i just workedaround. – gooleem Dec 6 '16 at 9:28

54

Use ***COALESCE*** - Learn more from here

For an example:

> 102
>
> 103
>
> 104

Then write below code in sql server,

```
Declare @Numbers AS Nvarchar(MAX) -- It must not be MAX if you have few numbers
SELECT  @Numbers = COALESCE(@Numbers + ',', '') + Number
FROM   TableName where Number IS NOT NULL

SELECT @Numbers
```

Output would be:

```
102,103,104
```

edited Sep 22 '17 at 23:27 — Graham — 4,008 — 14 — 42 — 63

answered Apr 5 '16 at 7:08 — pedram — 5,161 — 6 — 42 — 67

2 This is really the best solution IMO as it avoids the encoding issues that FOR XML presents. I used `Declare @Numbers AS Nvarchar(MAX)` and it worked fine. Can you explain why you recommend not using it please? – EvilDr Aug 3 '16 at 15:01

47

Postgres arrays are awesome. Example:

Create some test data:

```
postgres=# \c test
You are now connected to database "test" as user "hgimenez".
test=# create table names (name text);
CREATE TABLE
test=# insert into names (name) values ('Peter'), ('Paul'), ('Mary');
INSERT 0 3
test=# select * from names;
 name
-------
 Peter
 Paul
 Mary
(3 rows)
```

Aggregate them in an array:

```
test=# select array_agg(name) from names;
 array_agg
-------------------
 {Peter,Paul,Mary}
(1 row)
```

Convert the array to a comma delimited string:

```
test=# select array_to_string(array_agg(name), ', ') from names;
 array_to_string
-------------------
 Peter, Paul, Mary
(1 row)
```

DONE

1 1 11.9k 3 32 40

If you need more than one column, for example their employee id in brackets use the concat operator: `select array_to_string(array_agg(name||'('||id||')' ` – Richard Fox Feb 27 '15 at 11:50

Not applicable to sql-server, only to mysql – GoldBishop May 4 '17 at 15:03

45

Oracle 11g Release 2 supports the LISTAGG function. Documentation here.

```
COLUMN employees FORMAT A50

SELECT deptno, LISTAGG(ename, ',') WITHIN GROUP (ORDER BY ename) AS employees
FROM   emp
GROUP BY deptno;

    DEPTNO EMPLOYEES
---------- --------------------------------------------------
        10 CLARK,KING,MILLER
        20 ADAMS,FORD,JONES,SCOTT,SMITH
        30 ALLEN,BLAKE,JAMES,MARTIN,TURNER,WARD

3 rows selected.
```

Warning

Be careful implementing this function if there is possibility of the resulting string going over 4000 characters. It will throw an exception. If that's the case then you need to either handle the exception or roll your own function that prevents the joined string from going over 4000 characters.

edited Mar 14 '13 at 14:30

answered Mar 8 '12 at 16:29

Alex

7,313 7 58 75

1 For older versions of Oracle, wm_concat is perfect. Its use is explained in the link gift by Alex. Thnks Alex! – tcsconelli Jul 20 '15 at 13:04

In SQL Server 2005 and later, use the query below to concatenate the rows.

33

```
DECLARE @t table
(
    Id int,
    Name varchar(10)
)
INSERT INTO @t
SELECT 1,'a' UNION ALL
SELECT 1,'b' UNION ALL
SELECT 2,'c' UNION ALL
SELECT 2,'d'

SELECT ID,
stuff(
(
    SELECT ','+ [Name] FROM @t WHERE Id = t.Id FOR XML PATH('')
),1,1,'')
FROM (SELECT DISTINCT ID FROM @t ) t
```

edited Oct 20 '14 at 8:49

George Garchagudashvili
5,400 12 34 50

answered Jul 6 '11 at 12:46

Yogesh Bhadauirya
1,155 8 10

2 I believe this fails when the values contain XML symbols such as < or & . – Sam Aug 13 '13 at 1:36

I don't have access to a SQL Server at home, so I'm guess at the syntax here, but it's more or less:

26

```
DECLARE @names VARCHAR(500)

SELECT @names = @names + ' ' + Name
FROM Names
```

unnecessary one) – Marc Gravell ♦ Oct 12 '08 at 9:10

3 the only problem with this approach (which i use all the time) is that you can't embed it – ekkis Nov 23 '12 at 22:22

To get rid of the leading space change the query to `SELECT @names = @names + CASE WHEN LEN(@names)=0 THEN '' ELSE ' ' END + Name FROM Names` – Tian van Heerden Mar 4 '16 at 9:15

Also, you have to check that Name is not null, you can do it by doing: `SELECT @names = @names + ISNULL(' ' + Name, '')` – Vita1ij Mar 18 '16 at 10:49

25

You need to create a variable that will hold your final result and select into it, like so.

Easiest Solution

```
DECLARE @char VARCHAR(MAX);

SELECT @char = COALESCE(@char + ', ' + [column], [column])
FROM [table];

PRINT @char;
```

answered Nov 15 '16 at 21:07

Tigerjz32

2,684 3 18 31

24

A recursive CTE solution was suggested, but no code provided. The code below is an example of a recursive CTE -- note that although the results match the question, the data doesn't *quite* match the given description, as I assume that you really want to be doing this on groups of rows, not all rows in the table. Changing it to match all rows in the table is left as an exercise for the reader.

```
;with basetable as
(   SELECT id, CAST(name as varchar(max))name,
        ROW NUMBER() OVER(Partition By id     order by seq) rw,
```

```
            )g(id, name, seq)
),
rCTE as (
    SELECT recs, id, name, rw from basetable where rw=1
    UNION ALL
    SELECT b.recs, r.ID, r.name +', '+ b.name name, r.rw+1
    FROM basetable b
         inner join rCTE r
    on b.id = r.id and b.rw = r.rw+1
)
SELECT name FROM rCTE
WHERE recs = rw and ID=4
```

answered Aug 9 '12 at 21:06

jmoreno
11.1k 2 39 70

For the flabbergasted: this query inserts 12 rows (a 3 columns) into a temporary basetable, then creates a recursive Common Table Expression (rCTE) and then flattens the `name` column into a comma-separated string for 4 *groups* of `id` s. At first glance, I think this is more work than what most other solutions for SQL Server do. – knb Jul 24 '17 at 13:34

1 @knb: not sure if that is praise,condemnation,or just surprise. The base table is because I like my examples to actually work, it doesn't really have anything to do with the question. – jmoreno Jul 25 '17 at 2:20

23

Starting with PostgreSQL 9.0 this is quite simple:

```
select string_agg(name, ',')
from names;
```

In versions before 9.0 `array_agg()` can be used as shown by hgmnz

edited Oct 20 '14 at 10:42

answered Nov 16 '12 at 23:15

a_horse_with_no_name
316k 48 482 588

a_horse_with_no_name May 17 '13 at 12:11

21

In SQL Server vNext this will be built in with the STRING_AGG function, read more about it here: https://msdn.microsoft.com/en-us/library/mt790580.aspx

edited Jan 10 '17 at 15:42

answered Nov 21 '16 at 11:27

Henrik Fransas
905 9 13

17

Using XML helped me in getting rows separated with commas. For the extra comma we can use the replace function of SQL Server. Instead of adding a comma, use of the AS 'data()' will concatenate the rows with spaces, which later can be replaced with commas as the syntax written below.

```
REPLACE(
        (select FName AS 'data()'  from NameList  for xml path(''))
         , ' ', ', ')
```

edited Feb 26 '12 at 18:42

Peter Mortensen
14.1k 19 88 114

answered Apr 7 '11 at 11:16

Diwakar
179 1 2

2 This is the best answer here in my opinon. The use of declare variable is no good when you need to join in another table, and this is nice and short. Good work. – David Roussel Jun 2 '11 at 16:22

7 that's not working good if FName data has spaces already, for example "My Name" – binball Jun 8 '11 at 15:16

Really it is working for me on ms-sql 2016 Select REPLACE((select Name AS 'data()' from Brand Where Id IN (1,2,3,4) for xml path('')) , ' ', ', ') as allBrands – Rejwanul Reja Apr 28 '17 at 10:13

An empty list will result in NULL value. Usually you will insert the list into a table column or program variable: adjust the 255 max length to your need.

(Diwakar and Jens Frandsen provided good answers, but need improvement.)

edited Feb 26 '12 at 20:03

Peter Mortensen
14.1k 19 88 114

answered Feb 3 '12 at 10:39

Daniel Reis
10.1k 5 36 66

There is a space before the comma when using this :(– slayernoah Nov 18 '15 at 18:23

1 Just replace `', '` with `','` if you don't want the extra space. – Daniel Reis Nov 18 '15 at 23:17

12

```
SELECT STUFF((SELECT ', ' + name FROM [table] FOR XML PATH('')), 1, 2, '')
```

Here's a sample:

```
DECLARE @t TABLE (name VARCHAR(10))
INSERT INTO @t VALUES ('Peter'), ('Paul'), ('Mary')
SELECT STUFF((SELECT ', ' + name FROM @t FOR XML PATH('')), 1, 2, '')
--Peter, Paul, Mary
```

edited Feb 28 '18 at 16:04

answered Jan 25 '18 at 4:55

Max Szczurek
3,440 2 14 25

Thanks so much for giving the smallest possible solution, along with a working example! I had no idea why the top-voted answer works, nor how to replicate it. – jpaugh Mar 19 '18 at 14:21

This puts the stray comma at the beginning.

However, if you need other columns, or to CSV a child table you need to wrap this in a scalar user defined field (UDF).

You can use XML path as a correlated subquery in the SELECT clause too (but I'd have to wait until I go back to work because Google doesn't do work stuff at home :-)

edited Feb 26 '12 at 18:39

Peter Mortensen
14.1k 19 88 114

answered Oct 13 '08 at 17:24

gbn
352k 60 495 586

9

With the other answers, the person reading the answer must be aware of a specific domain table such as vehicle or student. The table must be created and populated with data to test a solution.

Below is an example that uses SQL Server "Information_Schema.Columns" table. By using this solution, no tables need to be created or data added. This example creates a comma separated list of column names for all tables in the database.

```
SELECT
    Table_Name
    ,STUFF((
        SELECT ',' + Column_Name
        FROM INFORMATION_SCHEMA.Columns Columns
        WHERE Tables.Table_Name = Columns.Table_Name
        ORDER BY Column_Name
        FOR XML PATH ('')), 1, 1, ''
    )Columns
FROM INFORMATION_SCHEMA.Columns Tables
GROUP BY TABLE_NAME
```

answered May 4 '16 at 19:31

Mike Barlow - BarDev
7,285 15 54 77

```
SELECT question_id,
   LISTAGG(element_id, ',') WITHIN GROUP (ORDER BY element_id)
FROM YOUR_TABLE;
GROUP BY question_id
```

as @user762952 pointed out, and according to Oracle's documentation http://www.oracle-base.com/articles/misc/string-aggregation-techniques.php, the WM_CONCAT() function is also an option. It seems stable, but Oracle explicitly recommends against using it for any application SQL, so use at your own risk.

Other than that, you will have to write your own function; the Oracle document above has a guide on how to do that.

edited May 23 '17 at 11:47
Community ♦
1 1

answered May 13 '13 at 16:02
ZeroK
370 3 7

I really liked elegancy of Dana's answer. Just wanted to make it complete.

7

```
DECLARE @names VARCHAR(MAX)
SET @names = ''

SELECT @names = @names + ', ' + Name FROM Names

-- Deleting last two symbols (', ')
SET @sSql = LEFT(@sSql, LEN(@sSql) - 1)
```

edited May 23 '17 at 11:55

community wiki
3 revs, 2 users 92%
Oleg Sakharov

If you are deleting the last two symbols ', ', then you need to add ', ' after Name ('SELECT \@names = \@names + Name + ', ' FROM Names'). That way the last two chars will always be ', '. – Justin T Dec 18 '15 at 11:04

In my case I needed to get rid of the *leading* comma so change the query to SELECT @names = @names + CASE WHEN LEN(@names)=0 THEN '' ELSE '

6

```
DECLARE @names VARCHAR(500)
SELECT @names = CONCAT(@names, ' ', name)
FROM Names
select @names
```

answered Feb 12 '15 at 12:01

Rapunzo
625 4 18 40

It would be nice to know *why* CONCAT works. A link to MSDN would be nice. – DaveBoltman Sep 20 '16 at 8:15

6

This answer will require some privilege in server to work.

Assemblies are a good option for you. There are a lot of sites that explain how to create it. The one I think is very well explained is this one

If you want, I have already created the assembly, and it is possible to download the DLL here.

Once you have downloaded it, you will need to run the following script in your SQL Server:

```
CREATE Assembly concat_assembly
   AUTHORIZATION dbo
   FROM '<PATH TO Concat.dll IN SERVER>'
   WITH PERMISSION_SET = SAFE;
GO

CREATE AGGREGATE dbo.concat (

    @Value NVARCHAR(MAX)
  , @Delimiter NVARCHAR(4000)

) RETURNS NVARCHAR(MAX)
EXTERNAL Name concat_assembly.[Concat.Concat];
GO
```

Observe that the path to assembly may be accessible to server. Since you have successfully done all the steps, you can use the function like:

```
SELECT dbo.Concat(field1, ',')
FROM Table1
```

Hope it helps!!!

answered May 8 '15 at 1:39

Nizam
3,398 3 33 53

5

I usually use select like this to concatenate strings in SQL Server:

```
with lines as
(
  select
    row_number() over(order by id) id, -- id is a line id
    line -- line of text.
  from
    source -- line source
),
result_lines as
(
  select
    id,
    cast(line as nvarchar(max)) line
  from
    lines
  where
    id = 1
  union all
  select
    l.id,
    cast(r.line + N', ' + l.line as nvarchar(max))
  from
```

```
select top 1
  line
from
  result_lines
order by
  id desc
```

edited Jul 6 '11 at 7:06

answered Jul 6 '11 at 6:58

Vladimir Nesterovsky
422 5 11

5

If you want to deal with nulls you can do it by adding a where clause or add another COALESCE around the first one.

```
DECLARE @Names VARCHAR(8000)
SELECT @Names = COALESCE(COALESCE(@Names + ', ', '') + Name, @Names) FROM People
```

answered Jul 27 '11 at 20:05

Pramod
59 1 3

5

MySQL complete Example:

We have Users which can have many Data's and we want to have an output, where we can see all users Datas in a list:

Result:

```
_____________________________
| id   |  rowList           |
|-------------------------/
| 0    | 6, 9               |
| 1    | 1,2,3,4,5,7,8,1    |
|____________________________|
```

```
CREATE TABLE `Data` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=latin1;


INSERT INTO `Data` (`id`, `user_id`) VALUES
(1, 1),
(2, 1),
(3, 1),
(4, 1),
(5, 1),
(6, 0),
(7, 1),
(8, 1),
(9, 0),
(10, 1);


CREATE TABLE `User` (
  `id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


INSERT INTO `User` (`id`) VALUES
(0),
(1);
```

Query:

```
SELECT User.id, GROUP_CONCAT(Data.id ORDER BY Data.id) AS rowList FROM User LEFT JOIN
Data ON User.id = Data.user_id GROUP BY User.id
```

answered Jul 22 '15 at 7:51

user1767754

10.9k 5 78 90


14.1k 19 88 114


49 1

4

This can be useful too

```
create table #test (id int,name varchar(10))
--use separate inserts on older versions of SQL Server
insert into #test values (1,'Peter'), (1,'Paul'), (1,'Mary'), (2,'Alex'), (3,'Jack')

DECLARE @t VARCHAR(255)
SELECT @t = ISNULL(@t + ',' + name, name) FROM #test WHERE id = 1
select @t
drop table #test
```

returns

```
Peter,Paul,Mary
```

answered Oct 25 '13 at 8:14


endo64
1,615 17 24

5 Unfortunately this behavior seems not to be officially supported. MSDN says: "If a variable is referenced in a select list, it should be assigned a scalar value or the SELECT statement should only return one row." And there are people who observed problems: sqlmag.com/sql-server/multi-row-variable-assignment-and-order – blueling Dec 5 '13 at 9:11

3

This method applies to Teradata Aster database only as it utilizes its NPATH function.

Again, we have table Students

```
SubjectID       StudentName
```

Then with NPATH it is just single SELECT:

```
SELECT * FROM npath(
  ON Students
  PARTITION BY SubjectID
  ORDER BY StudentName
  MODE(nonoverlapping)
  PATTERN('A*')
  SYMBOLS(
    'true' as A
  )
  RESULT(
    FIRST(SubjectID of A) as SubjectID,
    ACCUMULATE(StudentName of A) as StudentName
  )
);
```

Result:

```
SubjectID       StudentName
----------      -------------
1               [John, Mary, Sam]
2               [Alaina, Edward]
```

answered Nov 15 '13 at 20:26

topchef
13.7k 7 50 92

1 2 next

protected by Community ♦ Aug 2 '11 at 14:32

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).