COMBINING TABLES VERTICALLY WITH PROC SQL

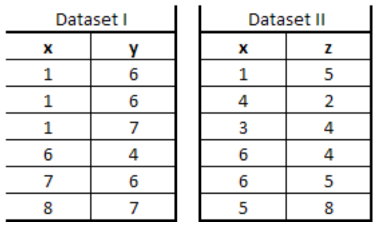
Deepanshu Bhalla 17 Comments PROC SQL, SQL

This tutorial explains how to combine / append data sets vertically with PROC SQL. Suppose you have two data sets and we need to combine these two datasets vertically. For example, if a dataset A contains 10 records and dataset B contains 10 records. I want combined dataset would contain 20 records.

Create data sets in SAS

```
data dat1;
input x y;
cards;
1 6
1 6
1 7
6 4
7 6
8 7
;
run;
data dat2;
input x z;
cards;
1   5
4   2
3   4
```

```
6   4
6   5
5   8
;
run;
```



Dataset I	
x	y
1	6
1	6
1	7
6	4
7	6
8	7

Dataset II	
x	z
1	5
4	2
3	4
6	4
6	5
5	8

Input Data Sets

1. UNION Operator

It displays all rows from both the tables and removes duplicate records from the combined dataset. By adding **ALL** keyword, it allows **duplicate rows** in the combined dataset.

Important Point

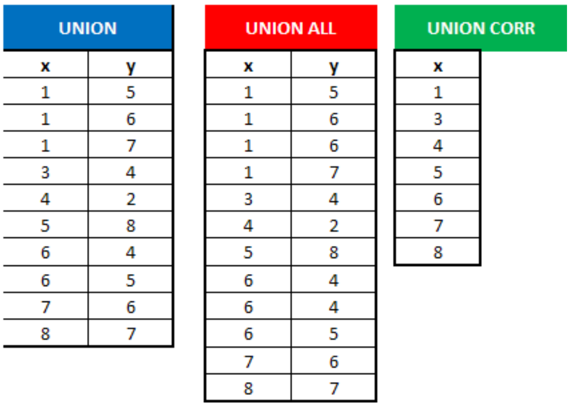
> **UNION** is performed by **position not by column name.** Hence, common columns in each **SELECT** statement should be in the same order. If **CORR keyword** is included, PROC SQL **matches the columns by name**.

ALL Keyword

> **ALL** keyword allows duplicates in the concatenated dataset.

CORR Keyword

> **CORR** keyword tells SAS to **match the columns in table by name** and not by position. Columns that do not match by name are excluded from the result table, except for the **OUTER UNION** operator



UNION	
x	y
1	5
1	6
1	7
3	4
4	2
5	8
6	4
6	5
7	6
8	7

UNION ALL	
x	y
1	5
1	6
1	6
1	7
3	4
4	2
5	8
6	4
6	4
6	5
7	6
8	7

UNION CORR
x
1
3
4
5
6
7
8

UNION Operator

```
proc sql;

create table out7 as

select *

from dat1

UNION

select *

from dat2;

quit;
```

```
proc sql;

create table out8 as

select *

from dat1

UNION ALL

select *

from dat2;

quit;
```

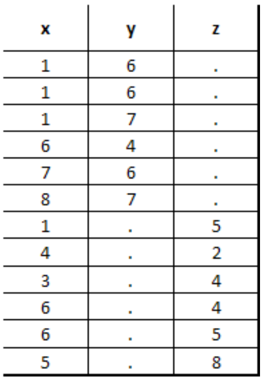
```
proc sql;

create table out9 as

select *

from dat1

UNION CORR

select *
```

```
from dat2;
quit;
```

2. OUTER UNION CORR

It appends (concatenates) two tables. It is equivalent to **SET statement** in Data Step. It allows duplicates in the concatenated table. **The ALL keyword is not required with OUTER UNION.**

```
proc sql;
create table out10 as
select *
from dat1
OUTER UNION CORR
select *
from dat2;
quit;
```

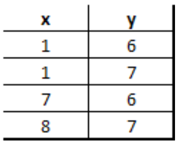


x	y	z
1	6	.
1	6	.
1	7	.
6	4	.
7	6	.
8	7	.
1	.	5
4	.	2
3	.	4
6	.	4
6	.	5
5	.	8

OUTER UNION CORR

3. Except Operator

It returns **unique** rows from the first query that are not found in the second query. **(Non matched Rows)**. It **removes duplicate records** (where all columns in the results are the same) - row 2nd in table1.



x	y
1	6
1	7
7	6
8	7

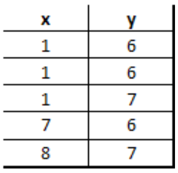
Except Operator

```
proc sql;
create table out1 as
select *
from dat1
EXCEPT
select *
from dat2;
quit;
```

Except ALL

It allows duplicate records in the combined dataset and does not remove duplicates.

```
proc sql;
create table out2 as
select *
from dat1
EXCEPT ALL
select *
from dat2;
quit;
```

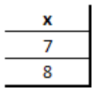


x	y
1	6
1	6
1	7
7	6
8	7

Except ALL

Except CORR

It displays only columns that have the same name (or **common**) in both the tables.



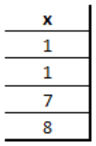
x
7
8

Except CORR

It returns all unique rows in the first table (**based on the common column**) that do not appear in the second table.

```
proc sql;
create table out3 as
select *
from dat1
EXCEPT CORR
select *
from dat2;
quit;
```

Except ALL CORR



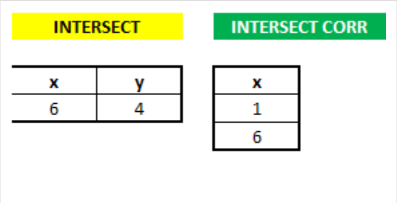
x
1
1
7
8

Except ALL Corr

4. INTERSECT Operator

It selects unique rows that are common to both the tables.

```
proc sql;
create table out5 as
select *
from dat1
INTERSECT
select *
from dat2;
quit;
```



INTERSECT

x	y
6	4

INTERSECT CORR

x
1
6

SQL : Intersect Operator

Related Posts

- Difference between WHERE and HAVING Clause in GROUP BY
- NODUPKEY with PROC SQL
- SAS : Random Sampling with PROC SQL
- Proc SQL Self Join Tutorial
- SAS CASE WHEN Statement: Learn with Examples

Proc SQL Tutorials : Top 15 Proc SQL Tutorials

Spread the Word!

Share in Share Tweet

Deepanshu founded ListenData with a simple objective - Make analytics easy to understand and follow. He has over 10 years of experience in data science. During his tenure, he worked with global clients in various domains like Banking, Insurance, Private Equity, Telecom and HR.

While I love having friends who agree, I only learn from those who don't

Let's Get Connected

Email in LinkedIn

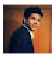
17 Responses to "Combining Tables Vertically with PROC SQL"

Vishal agarwal April 26, 2016 at 10:49 AM

Thanks for providing easy to understand examples and language which helps t grab complex things much easier.

Reply Delete

Replies



Deepanshu Bhalla April 26, 2016 at 10:53 AM

Glad you found it helpful. Cheers!

Delete

Reply

Unknown July 6, 2016 at 11:17 AM

It's reallllllly easy to understand

Reply Delete

Replies

Deepanshu Bhalla July 9, 2016 at 6:02 AM

Thank you for stopping by my blog.

Delete

Reply

Unknown July 30, 2016 at 2:52 AM

Appreciate your work

Reply Delete

Manish August 28, 2016 at 2:20 PM

Its very good and understandable....

Reply Delete

Aarthi October 29, 2016 at 5:41 AM

Thanks a lot :)

Reply Delete

Anonymous March 1, 2017 at 12:16 PM

Hi sir,
Are you clinical sas programmer?

Reply Delete

Replies

sadsadsa May 18, 2017 at 11:42 PM

This comment has been removed by the author.

Delete

Reply

Unknown June 2, 2017 at 11:57 AM

Thank you for the easy understand example and explanation. It is very good for learners to go through

Reply Delete

Unknown February 23, 2018 at 4:34 AM

Hi
Can you please explain the difference between inner join and intersect with same example.

Reply Delete

Replies

Anonymous August 7, 2018 at 9:36 AM

inner join

inner join is a equi join and we can use non equi join also. when condition is true , matched records are displayed

intersect is used to derived common records in both table
A table;
id
1
2
3
4

B table;
id
1
3
4

inner join :

select * from a, b where a.id=b.id;
select * from a inner join b on a.id=b.id;

intersect

select id from a intersect select id from b;
1
3
4

Delete

Reply

Unknown April 21, 2019 at 12:49 PM

can you explain CORR with some brief example

Reply Delete

Anonymous May 15, 2019 at 9:44 PM

THANKS A TON FOR MAKING IT CLEAR AND EASY TO LEARN

Reply Delete

Anonymous January 8, 2020 at 12:48 AM

Thanks for sharing it with us! Can you please guide me how to append multiple files using a sql function at once other than typing "Union all corr select * from C"? Thank you.

Reply Delete

Unknown March 20, 2021 at 7:19 PM

very usefull, thank you

Reply Delete

annonymous November 27, 2021 at 8:01 PM

I have doubt that, in the example datasets variable names are different then how it is combining them ?
does while combining it only sees the row values and not variable names ?

Reply Delete

To leave a comment, click the button below to sign in with Google.

SIGN IN WITH GOOGLE

← PREV NEXT →