

## Baron Schwartz's Blog

# How to number rows in MySQL

Sat, Dec 2, 2006 in [Databases](#)    

I wrote before about a generic, cross-platform way to simulate the SQL `ROW_NUMBER()` function in any RDBMS. There is a much more efficient way to do this on MySQL with user variables.

## Background

Please see my previous article on [how to simulate the `ROW\_NUMBER\(\)` function](#) for the background. I'll use the same table structure and data in this article.

Unfortunately, that's a quadratic algorithm, so it's not something I'd do much (though I once did it over small sets of data in SQL Server 2000 at a jobsite).

## A more efficient method

In MySQL you can simultaneously assign to and read from [user variables](#) in a SELECT statement. This allows the following method of numbering rows:

```
set @type = '';
set @num = 1;

select
    type,
    variety,
    @num := if(@type = type, @num + 1, 1) as row_number,
    @type := type as dummy
from fruits;
```

type	variety	row_number	dummy
apple	fuji	1	apple
apple	gala	2	apple
apple	limbertwig	3	apple
cherry	bing	1	cherry

cherry	chelan		2	cherry
orange	navel		1	orange
orange	valencia		2	orange
pear	bartlett		1	pear
pear	bradford		2	pear
+-----+-----+-----+-----+				

How does that work? I'm restarting the row number each time the `type` column changes, by keeping track of the value it had in the last row. And I'm simultaneously incrementing and selecting the row number in each row.

The spurious `dummy` column has to be there, but if your version of MySQL supports it, you can use a subquery in the `FROM` clause to eliminate columns you don't want in the results.

## Efficiency

All I'm doing is maintaining a bit of extra memory and performing a few small comparisons and assignments for each row, so this technique is very efficient.

## Playing with fire

You can refer to the generated `row_number` column in a `HAVING` or `GROUP BY` clause, but don't burn your fingers. This technique is very much like playing with fire. The result of assigning to a variable and using it in the same statement (in the `HAVING`, for example) depends on the query plan the server chooses, the phase of the moon, and probably other things too. Before you use this technique, you should read and understand the [section on user-defined variables in the MySQL Manual](#), and decide whether it's safe for your query.

Now that you've read that section of the manual, particularly the part about the aliased expression, you should understand why the following query might be a safer paradigm when using the result in the `HAVING` clause, even though it produces another dummy column:

```
set @type = '';
set @num = 1;
```

```

select
    type,
    variety,
    @num := if(@type = type, @num + 1, 1) as dummy
_1,
    @type := type as dummy_2,
    @num as row_number
from fruits
group by type, variety
having row_number = 1;

```

```

+-----+-----+-----+-----+-----+
----+
| type   | variety | dummy_1 | dummy_2 | row_num
ber |
+-----+-----+-----+-----+-----+
----+
| apple  | fuji    | 1        | apple    | 1
|
| cherry | bing     | 1        | cherry    | 1
|
| orange | navel    | 1        | orange    | 1
|
| pear   | bartlett | 1        | pear      | 1
|
+-----+-----+-----+-----+-----+
----+

```

(If I'm wrong about that, somebody please correct me).

A safer technique is to use a subquery in the **FROM** clause. This will cause the results to be materialized in a temporary table behind the scenes. It might be less efficient for some uses, though:

```

select type, variety
from (
    select
        type,
        variety,
        @num := if(@type = type, @num + 1, 1) as ro
w_number,
        @type := type as dummy
    from fruits
) as x
where row_number = 1;

```

```

+-----+-----+
| type   | variety |
+-----+-----+
| apple  | fuji    |
| cherry | bing     |

```

orange   navel
pear   bartlett
+-----+-----+



## Conclusion

This is an efficient, flexible way to generate and use row numbers in MySQL. I'll leave it to you to find uses for it for right now, but I'm going to show you at least one application for this in an upcoming article.

*I'm Baron Schwartz, the founder and CEO of [VividCortex](#). I am the author of *High Performance MySQL* and many open-source tools for performance analysis, monitoring, and system administration. I contribute to various database communities such as Oracle, PostgreSQL, Redis and MongoDB.*

Newer


Older

## Comments

CommentsCommunity1 Login

RecommendShareSort by Oldest

Join the discussion...

 **nah** · 9 years ago

"There is a much more efficient way to do this on MySQL".

Pfff ... as always, MySQL fan-boys don't give any fact about the supposed 'efficiency' of their 'magic sql'.

Anyway, it's funny. Those fan-boys seem to discover, every day, something 'hyper-mega-cool' about RDBMS (well if you consider MySQL).

search this website

NUMBERING (well, if you consider MySQL a

of course, LOL).

^ | v · Reply · Share ›



**Xaprb** · 9 years ago

You make a very good point... I don't provide any hard numbers about efficiency. And I'm the one who says you should always do real measurements :-). It's impossible to say much about query performance on such a small data set, so I added another 100 rows for each type of fruit:

```
insert into fruits(type, variety)
  select type, concat(variety, i)
from fruits
  cross join mysql.number
where i < 100;
```

Now the queries perform differently enough to appreciate the linear vs. quadratic algorithms. This is with a totally in-memory dataset; if it were too large to fit in memory, it would be even clearer:



[see more](#)

^ | v · Reply · Share ›



**Sheeri** · 9 years ago

In the last article you specified it, and it's worth repeating here -- you want to order by type, because the numbering restarts at any change -- it doesn't magically know the count for "apple" or "cherry", it just resets on a change.

^ | v · Reply · Share ›



**Xaprb** · 9 years ago

That is a good point. The rows are numbered in the order they come back to the client, which is precisely the thing that's hard to do without this solution. If you want the rows in a different order, you just say ORDER BY whatever, and they're numbered in the order they're returned.

^ | v · Reply · Share ›



**Nicholas Thompson** · 9 years ago

**VERY** nice article! You just saved me hours (well at least many minutes) of tedious work. I adapted the incrementing variable technique you used above to

incrementing variable technique you used above to reorder the weights of menu items in Drupal (an Open Source CMS) so the items appears alphabetically...

I posted how I did it on my blog [Ordering Menu Items Alphabetically in Drupal](#).

^ | v · Reply · Share ›



**Unomi** · 9 years ago

To just count the rows incremently use something like this:

```
SELECT a.*, @num := @num + 1 b from test a, (SE
```

This results in having table 'a' with an additional column 'b' wich just counts the rows. Setting @num in the sub-selection makes it simple to put it into 1 query. The incrementing number can be adjusted.

After the 'd' (wich stands for derived and it is mandatory to have an alias), you can order, group by etc. the column 'b' still adds up.

Hope this helps people searching for a row number solution.....

- Unomi -

^ | v · Reply · Share ›



**Xaprb** · 9 years ago

That's a great solution.

^ | v · Reply · Share ›



**Alex** · 9 years ago

Anyone have a solution for SQLite? It doesn't have user-defined variables, sadly.

^ | v · Reply · Share ›



**Rafael Vale** · 9 years ago

Yo there!

Your posts helps me very much, thank u for that!  
I have a little problem and probably somebody here can help me!

I need to select the entries of a table. But i want only the MAX(id) from each product\_id of that table.

i'm trying:

```
SELECT max(id), id_pit, id_atend FROM table  
GROUP BY id_pit
```

but the result is the MAX id with the id\_atend from another id.

How can i do it?

Thanks for attention!

^ | v · Reply · Share ›



**Xaprb** · 9 years ago

You need to read two of my other articles: the "next" article linked from the top of this page, on finding the first row per group, and this one about [incorrect GROUP BY](#), which you are doing and is causing you trouble.

^ | v · Reply · Share ›



**Nick T.** · 9 years ago

I'm a new MySQL/PHP user, so please bear with me. The information you provided, Xaprb, is excellent. But is there a way I can make the row\_number column a permanent column in my database instead of just output?

I have a table very similar to that of "fruits", with several broad "types" and specific "varieties" within those "types". I need to add another column numbering these types but the table is just too big to do it by hand. The solution you mentioned above is great for output, but how can I add the column into the permanent table itself?

I tried to make a MySQL dump file of the query you gave above, and then import that into my table using PHPMyAdmin, but I kept getting errors. Is there another way?

^ | v · Reply · Share ›



**Xaprb** · 9 years ago

The simplest way is to make a multiple-column primary key on (type, variety, ai) where ai is an auto\_increment integer. This only works on MyISAM, though.

^ | v · Reply · Share ›



**rackoon** · 9 years ago



If I'm not mistaken, there is no guarantee that this method works. In the mysql-manual it says:

---SNIP---

The order of evaluation for user variables is undefined and may change based on the elements contained within a given query. In `SELECT @a, @a := @a 1 ...`, you might think that MySQL will evaluate `@a` first and then do an assignment second, but changing the query (for example, by adding a `GROUP BY`, `HAVING`, or `ORDER BY` clause) may change the order of evaluation.

The general rule is never to assign a value to a user variable in one part of a statement and use the same variable in some other part of the same statement. You might get the results you expect, but this is not guaranteed.

---SNIP---

So how do you know that in your first example `"if(@type = type, @num 1, 1)"` is evaluated before you reassign `@type` via `"@type := type"`?

^ | v · Reply · Share ›



**Xaprb** · 9 years ago

The manual is vague and inaccurate about this point. You have to read the source code and learn how the MySQL query executioner (yes, it's actually called that!) works.

^ | v · Reply · Share ›



**tadwook** · 8 years ago

thanks rackoon -- THIS METHOD DOES NOT WORK WITH order by -- just FYI

^ | v · Reply · Share ›



**Allison** · 8 years ago

Thanks, Unomi! Your solution works great for me... I was looking for an alternative to SQL Server 2005's `row_number()` function.

^ | v · Reply · Share ›



**Namrata Chauhan** · 8 years ago

This is an excellent .  
It is very helpful to me.



Thanks for this solution..

^ | v · Reply · Share ›



**Dario V** · 7 years ago

Thanks a lot.

```
set @num = 0;
select *, @num := @num + 1 as row_number from
TABLE
```

^ | v · Reply · Share ›



**simoneranucci** · 7 years ago

How do you hide dummy column with subquery???  
:'(

^ | v · Reply · Share ›



**enjoi** · 7 years ago

Very helpfull !

A question : Can this be used to update a column ?

for example select query:

```
set @num = 0;
select _pos, @num := @num + 1 as row_num
from TABLE
order by _pos;
```

result :

```
_pos row_num
1 1
4 2
7 3
8 4
... ..
```

Update query (does not work) :

```
set @num = 0;
update TABLE
set @num := @num + 1, _pos = @num
order by _pos;
```

TY for answer !

^ | v · Reply · Share ›



**Nico** · 6 years ago

Thank you for sharing!

The use of an order by clause does mix up the row

numbers for me. In case you need an order by you can use a subquery for the ordering and apply the order number in the outer select...

^ | v · Reply · Share ›



**Zilus** · 6 years ago

Awsome! I use this in a Query to order my results, sort of "x / x/" I mean, like \$num / \$total

I'll post my code and give you credits!

Thanks!

^ | v · Reply · Share ›



**Ozlem** · 5 years ago

Hi, this is a great solution however it does not work from php. I mean when I run the query through php it does not gather any result. it does not recognize first 2 lines of code, set functions...Any suggestion is appreciated.?Thanks...Ozlem.

```
set @countT = 0;
```

```
set @rowCount = 0;
```

```
SELECT MQ2.period_id, MQ2.lt_id,  
MQ2.period_name, MQ2.min_stay, MQ2.fromDate,  
MQ2.toDate, MQ2.nightly_rate, MQ2.periodDays,  
MQ2.totalDays, MQ2.fixed_rate, @countT as  
periodDaysTotal, @rowCount as rowsTotal ,
```

```
IF ((@rowCount=1),
```