

Transpose rows to columns in MySQL

In MySQL, transposing data in rows to columns is not a trivial job, especially when the number of distinct values in a column is undetermined and the values are unknown. However, there is a nice solution for this job. To achieve this goal, the magic is to use `group_concat` function in MySQL. When the number of distinct values and the values are known before you write the SQL statement, it can be done with the `group_concat` function. However, to get a nice solution for the undetermined number of unknown values, there are couple of steps: 1) create a `sql_segment` with `group_concat` function; 2) create a complete SQL statement on fly; 3) execute the constructed SQL statement.

1. Create a `sql_segment` with `group_concat`

Suppose we have a table called `table1`, it has a number columns, like `f1`, `t1`, and `v1`. We want to transpose the data based on values in `f1` and create column headers based on `f1` and `t1`. The following SQL statement can create a piece of SQL segment for using in the next step.

```
declare _SQL text(4096); SET SESSION group_concat_max_len = 100000; SET
@_SQL = null; SELECT GROUP_CONCAT(DISTINCT CONCAT(
'group_concat(if(a.f1 = ', f1, ', if(v1<0, null, round(v1,1)), null)) AS `',
concat(t1,'_',f1), '`' ) ) INTO @_SQL FROM ( select distinct f1, t1 from
table1 group by f1, t1 order by f1 ) d;
```

2. Create a complete SQL statement

Create the complete SQL statement. All data in table "`table1`" will be grouped by `f2,f3`. The simple statement will be like the following.

```
SET @_SQL = CONCAT('f2,f3,', @_SQL, ' from table1 group by f2, f3' );
```

3. Execute the constructed SQL statement

Use the following statements to execute the SQL statement stored in `@_SQL`.

```
prepare s1 from @_SQL; execute s1; deallocate prepare s1;
```