How to Pivot and Unpivot views

Revision 20180406

NOTE
This document is confidential and proprietary of Denodo Technologies. No part of this document may be reproduced in any form by any means without prior written authorization of Denodo Technologies.

Copyright © 2019
Denodo Technologies Proprietary and Confidential
Goal

This document describes how to apply a row to column (pivot) or a column to row (unpivot) transformation in a Denodo Virtual DataPort view.

Content

This document is composed of two sections:

- **Pivot**: explains how to transform a row to a column using an example.
- **Unpivot**: explains how to transform a column to a row with an example.

1. Pivot

In the following view `df_jobs`:

<table>
<thead>
<tr>
<th>JOB</th>
<th>DEPTNO</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALYST</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>CLERK</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>CLERK</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>CLERK</td>
<td>30</td>
<td>1</td>
</tr>
<tr>
<td>MANAGER</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>MANAGER</td>
<td>20</td>
<td>1</td>
</tr>
<tr>
<td>MANAGER</td>
<td>30</td>
<td>1</td>
</tr>
<tr>
<td>PRESIDENT</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>SALESMAN</td>
<td>30</td>
<td>4</td>
</tr>
</tbody>
</table>

we want the different department numbers under DEPTNO to be columns. In order to do that:

- We have 4 department numbers that we want to use: 10, 20, 30 and 40.
- We are going to create one column for each department dept_10, dept_20, dept_30, dept_40 that will have the values that are currently in the COUNT column.
The VQL query to create the new view will look like this:

```
CREATE VIEW p_df_jobs AS SELECT job AS job,
max(case WHEN (deptno = '10') THEN count ELSE NULL END) AS dept_10,
max(case WHEN (deptno = '20') THEN count ELSE NULL END) AS dept_20,
max(case WHEN (deptno = '30') THEN count ELSE NULL END) AS dept_30,
max(case WHEN (deptno = '40') THEN count ELSE NULL END) AS dept_40
FROM df_jobs
GROUP BY job
```

It is possible to create the same view in using the graphical interface from the VDP admin tool instead of using the VQL query directly:

1. Grouping the result set to have all the attributes of each item (Group By tab):
   - Adding the 4 fields in the Output tab with the following function using the correct deptno for each field
     - `dept_10`:
       - `max(case WHEN (df_jobs.deptno = '10') THEN count ELSE NULL END)`
And the results:

<table>
<thead>
<tr>
<th>Before Pivot</th>
<th>After Pivot</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>job</strong></td>
<td><strong>dept_no</strong></td>
</tr>
<tr>
<td>ANALYST</td>
<td>20</td>
</tr>
<tr>
<td>CLERK</td>
<td>10</td>
</tr>
<tr>
<td>CLERK</td>
<td>20</td>
</tr>
<tr>
<td>CLERK</td>
<td>30</td>
</tr>
<tr>
<td>MANAGER</td>
<td>10</td>
</tr>
<tr>
<td>MANAGER</td>
<td>20</td>
</tr>
<tr>
<td>MANAGER</td>
<td>30</td>
</tr>
<tr>
<td>PRESIDENT</td>
<td>10</td>
</tr>
<tr>
<td>SALESMAN</td>
<td>30</td>
</tr>
</tbody>
</table>

This new view has pivoted the COUNT column by DEPTNO across JOB.

Another typical use case is when you have key-value data sources, i.e., for a given register natural key you have multiple records, each one with a key and its correspondent value.

<table>
<thead>
<tr>
<th>Name</th>
<th>Surname</th>
<th>ID</th>
<th>KEY</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Smith</td>
<td>C001</td>
<td>age</td>
<td>32</td>
</tr>
<tr>
<td>John</td>
<td>Smith</td>
<td>C001</td>
<td>city</td>
<td>San Jose</td>
</tr>
</tbody>
</table>
In this case, we want a new column for each one of the possible values of 'key'. The pivoted table could be created as:

```
CREATE VIEW pivoted_view AS SELECT name AS name, surname AS surname, id as id,
max(case WHEN (key = 'age') THEN value ELSE NULL END) AS age,
max(case WHEN (key = 'city') THEN value ELSE NULL END) AS city,
max(case WHEN (key = 'state') THEN value ELSE NULL END) AS state
FROM my_view
GROUP BY name, surname, id;
```

And the result would be:

<table>
<thead>
<tr>
<th>Name</th>
<th>Surname</th>
<th>ID</th>
<th>Age</th>
<th>City</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Smith</td>
<td>C001</td>
<td>32</td>
<td>San Jose</td>
<td>&lt;null&gt;</td>
</tr>
<tr>
<td>Eugene L</td>
<td>Walter</td>
<td>C080</td>
<td>27</td>
<td>&lt;null&gt;</td>
<td>CA</td>
</tr>
<tr>
<td>Melisa</td>
<td>Lim</td>
<td>C040</td>
<td>&lt;null&gt;</td>
<td>Oakland</td>
<td>&lt;null&gt;</td>
</tr>
</tbody>
</table>

2. Unpivot

In the following view df_customer_phone:

<table>
<thead>
<tr>
<th>customer_id</th>
<th>phone1</th>
<th>phone2</th>
<th>phone3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>705-421-4577</td>
<td>705-421-1421</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>734-843-9756</td>
<td>372-626-7333</td>
<td>705-521-9242</td>
</tr>
<tr>
<td>3</td>
<td>834-277-1151</td>
<td>263-421-7326</td>
<td></td>
</tr>
</tbody>
</table>
we want to have two columns, one with the customer_id and one with a single phone. The approach is to create a union with a branch for every column that we are going to unpivot. In this case phone1, phone2 and phone3 will be transformed to the column phone in our unpivoted view.
In order to do that, in this case the easiest way is to create the view manually using VQL. In other cases we may need to create a projection view for every branch and then create the view using the graphical interface:

```
CREATE VIEW unpivot_df_customer_phone AS
SELECT customer_id, phone1 as phone FROM df_customer_phone
WHERE phone1 <> ''
UNION
SELECT customer_id, phone2 as phone FROM df_customer_phone
WHERE phone2 <> ''
UNION
SELECT customer_id, phone3 as phone FROM df_customer_phone
WHERE phone3 <> ''
```

**Note:** In this case we are using `phoneX <> ''` as condition in the different union branches because we are using a delimited file as data source but `'phoneX is not null'` may be needed in other cases.

To create this view using the GUI:
- Create a projection view for every branch maintaining customer_id and one of the phone columns for each view.
  - Every view will have a condition for empty values for the same column that we are projecting on the view. This is the projection returning only phone1 column but two more views need to be created for this example, one returning phone2 and one returning phone3
    - Where Condition:
○ Output:

○ Create an Union View using those projections and include an association for the phoneX fields:
In both cases using the GUI and using VQL the final view will look like this:

Before unpivot

<table>
<thead>
<tr>
<th>customer_id</th>
<th>phone1</th>
<th>phone2</th>
<th>phone3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>705-421-4577</td>
<td>705-421-1421</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>734-843-9756</td>
<td>372-823-7333</td>
<td>705-521-9242</td>
</tr>
<tr>
<td>3</td>
<td>334-277-1151</td>
<td>253-421-7235</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>238-843-4734</td>
<td>258-421-2000</td>
<td>522-421-1421</td>
</tr>
<tr>
<td>5</td>
<td>125-341-1111</td>
<td>632-340-1521</td>
<td></td>
</tr>
</tbody>
</table>

After unpivot

<table>
<thead>
<tr>
<th>customer_id</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>705-421-1421</td>
</tr>
<tr>
<td>2</td>
<td>734-843-9242</td>
</tr>
<tr>
<td>1</td>
<td>734-421-4577</td>
</tr>
<tr>
<td>2</td>
<td>372-823-7333</td>
</tr>
<tr>
<td>3</td>
<td>734-843-6796</td>
</tr>
<tr>
<td>4</td>
<td>522-421-1421</td>
</tr>
<tr>
<td>3</td>
<td>253-421-7326</td>
</tr>
<tr>
<td>4</td>
<td>238-843-4734</td>
</tr>
<tr>
<td>5</td>
<td>632-346-1521</td>
</tr>
<tr>
<td>5</td>
<td>125-341-1111</td>
</tr>
</tbody>
</table>