Custom Transformation - use cases with advanced SQL queries

This version of the app's documentation is outdated. Please find the information you're looking for here:

- Custom Transformation - use cases with advanced SQL queries

<table>
<thead>
<tr>
<th>On this page you will learn about:</th>
<th>Read more:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- This version of the app's documentation is outdated. Please find the information you're looking for here:</td>
<td>- Default automatic presets</td>
</tr>
<tr>
<td>- Using If / Then construct</td>
<td></td>
</tr>
<tr>
<td>- Comparing two dates using If / Then construct</td>
<td></td>
</tr>
<tr>
<td>- Using formulas</td>
<td></td>
</tr>
<tr>
<td>- Merging tables using filtration</td>
<td></td>
</tr>
<tr>
<td>- Calculating the number of workdays in a period of time</td>
<td></td>
</tr>
</tbody>
</table>

Using If / Then construct

Use case:

You have a table with the stationary orders. You need to output a new column in the table with the order priority according to the total sum of money: less than $1000 is Low, from $1000 to $2000 is Medium, and more than $2000 is High.

Solution:

1. Switch the page to the edit mode.
2. Insert the Table Transformer macro and paste the table or the macros outputting tables within the macro body.
3. Select the macro and click Edit.
4. In the Presets tab select Custom transformation and click Next.
5. Enter the following SQL query:

```
SELECT *
CASE
    WHEN 'Subtotal' < 1000
        THEN "LOW"
    WHEN 'Subtotal' >= 1000
        AND 'Subtotal' < 2000
        THEN "MEDIUM"
    ELSE "HIGH"
END AS 'Priority'
FROM T1
```

*CASE WHEN ... THEN ... ELSE ... END* goes through conditions and return a value when the first condition is met.

AS '...' outputs a new 'Priority' column.

6. Click Next.
7. Define the table settings and view options if needed.
8. Save the macro and the page.
Comparing two dates using If / Then construct

Use case:
You have a table with two columns: Target date and Completion date. You need to compare and rate these dates:

- If the completion date is within 5 days, the status is **ON TIME**
- If the completion date is more than 5 days late, but less than 10 days, the status is **LATE**
- If the completion date is more than 10 days, the status is **VERY LATE**

Solution:
1. Switch the page to the edit mode.
2. Insert the Table Transformer macro and paste the tables or the macros outputting tables within the macro body.
3. Select the macro and click **Edit**.
4. In the **Presets** tab select **Custom transformation** and click **Next**.
5. Enter the following SQL query:
SELECT *
,
CASE
  WHEN
    (('Completion date' - 'Target date') / "24h") < 5
  THEN "ON TIME"
  WHEN
    (('Completion date' - 'Target date') / "24h") > 5
    AND
    (('Completion date' - 'Target date') / "24h") <= 10
  THEN "LATE"
  ELSE "VERY LATE"
END
  AS 'Rating'
FROM T1

6. Click Next.
7. Define the date format.
8. Save the macro and the page.

Using formulas

Use case:

You have a table with the number of added lines of code and the number of defects. You need to calculate the quality of the code. The number of defects less than one defect per 10 lines indicates the good quality of the code. Otherwise the quality of the code is bad.

Solution:

1. Switch the page to the edit mode.
2. Insert the Table Transformer macro and paste the tables or the macros outputting tables within the macro body.
3. Select the macro and click Edit.
4. In the Presets tab select Custom transformation and click Next.
5. Enter the following SQL query:

SELECT *
,
{ 'Number of defects' * 10 / 'Lines of code added'
  AS 'Defects per 10 lines',
  CASE
    WHEN
      'Number of defects' * 10 / 'Lines of code added'
    < 1
    THEN "Good"
    ELSE "Bad"
  END
  AS 'Code quality'
FROM T1
(‘...’ * 10 / ‘...’) AS ‘...’ calculates the number of defects per 10 lines of code and outputs the ‘Defects per 10 lines’ column.

CASE WHEN ... THEN ... ELSE ... END AS ‘...’ goes through conditions and return a value when the first condition is met and outputs the ‘Code quality’ column.

6. Click Next.
7. Define the table settings and view options if needed.
8. Save the macro and the page.

You can merge two columns containing text with the help of formulas:

```sql
SELECT *
  ,
  ('First name' + ' ' + 'Last name')
AS 'Full name'
FROM T1
```
Merging tables using filtration

Use case:

You have two tables where the columns differ.

You need to merge two tables where the entries meet the conditions:

- transaction date in Table 1: October 2018
- customer type in Table 2: business

Solution:

1. Switch the page to the edit mode.
2. Insert the Table Transformer macro and paste the tables or the macros outputting tables within the macro body.
3. Select the macro and click Edit.
4. In the Presets tab select Custom transformation and click Next.
5. Enter the following SQL query:
Calculating the number of workdays in a period of time

Use case:
You have a table containing a period of time (start date and end date).
You need to count the number of workdays during this time period.

Advanced use case:
You have an additional table with national holidays which you also shouldn’t consider as workdays.

Solution:

1. Switch the page to the edit mode.
2. Insert the Table Transformer macro and paste the table within the macro body.
3. Select the macro and click Edit.
4. Switch to the SQL query tab.
5. Enter the following SQL query:

```
SELECT *
FROM T1
JOIN T2
ON T1.'Transaction ID' = T2.'Transaction ID'
WHERE
(  
  'Transaction Date' >= '10 / 1 / 2018'
  AND 'Transaction Date' < '11 / 1 / 2018'
)
  AND
(  
  'Customer Type' = "Business"
)
WHERE ('...' >= "10/1/2018" AND '...' < "11/1/2018") AND ('...' = "...") extracts only those records that fulfill a specified condition.
```

6. Click Next.
7. Define the table settings and view options if needed.
8. Save the macro and the page.
SELECT *
' Days' - 2 * ('Days' / 7)::integer -
CASE
  WHEN 'Days' % 7 = 0 THEN 0
  WHEN 'Start date'::Date->getDay() = 0 THEN 1
  WHEN 'Start date'::Date->getDay() + 'Days' % 7 = 6 THEN 1
  WHEN 'Start date'::Date->getDay() + 'Days' % 7 > 6 THEN 2
  ELSE 0
END AS 'Work days'
FROM
(SELECT *
(SELECT T1.'End date' - T1.'Start date') / "1d" + 1
AS 'Days' FROM T1) AS TT

6. Click Next.
7. Define the table settings and view options if needed.
8. Save the macro and the page.

Solution for the advanced use case:

Enter the following SQL query:

```
SELECT *
' Days' - 2 * ('Days' / 7)::integer -
CASE
  WHEN 'Days' % 7 = 0 THEN 0
  WHEN 'Start date'::Date->getDay() = 0 THEN 1
  WHEN 'Start date'::Date->getDay() + 'Days' % 7 = 6 THEN 1
  WHEN 'Start date'::Date->getDay() + 'Days' % 7 > 6 THEN 2
  ELSE 0
END -
(SELECT COUNT(*) FROM
(SELECT *, 'Holiday'::Date->getDay() AS 'Day' FROM T2)
WHERE 'Holiday' >= TT.'Start date' AND 'Holiday' <= TT.'End date'
AND 'Day' > 0 AND 'Day' < 6)
AS 'Work days'
FROM
(SELECT *, ('End date' - 'Start date') / "1d" + 1
AS 'Days' FROM T1) AS TT
```