

# Filters

Filters are implemented in form of a load script. These load scripts can be set for:

- **Documents:** Load scripts in documents will be applied to the whole document. Filters set here will be global and applied to all analysis parts. You can also define selections that will be applied when the analysis is reloaded.
- **Sheets:** Filters that are defined for sheets will filter down everything on the sheet. But other parts of the analysis will not be affected.
- **Components:** Filters for single components only have an effect on this component.
- **Stories:** Filters in stories will define what part of the data will actually be shown in the download. The analyst can predefine which selection / filter the download will show.

ANALYSIS SETTINGS

General settingsVariablesLoad ScriptProcess Explorer KPIs

```
clear selections;
select pinned "_CEL_O2C_ACTIVITIES"."EVENTTIME" = 2010
```

Filter Builder

the Filter Builder lets you create some basic Filters for the Load Script

First select a table and then select a column. After that click the Add button to add the Filter to the text area on the left.

You then have to edit <op> and <value>. Valid options for <op> are for example =, <, >, <=, >=.

In the Info section you can see some examples on how to use them.

Table

Add

Info

You can specify custom PQL queries here. Multiple queries are separated by a semicolon.

We highly recommend to start load scripts for analysis or sheets with "CLEAR SELECTIONS"

Examples:

1. filter "case\_table"."caseid" = 2

2. filter "case\_table"."caseid" = 2; filter "activity\_table"."activity\_text" LIKE '%PO%'

3. filter YEAR("case\_table"."case\_start\_time") = 2010

4. select "EVENTLOG"."SORTING" > 20

5. select YEAR("EVENTLOG"."EVENTTIME") > 2012

6. select pinned CALC\_THROUGHPUT(FIRST\_OCCURRENCE [Source Activity Name] TO LAST\_OCCURRENCE [Target Activity Name], REMAP\_TIMESTAMPS("table"."event time", HOURS)) BETWEEN 2011 and 2014

7. select pinned "EVENTLOG"."USER\_TYPE" = 'Batch'

Done

## Filter Syntax

Filters offer the possibility to predefine what part of the data is shown in the respective part of the analysis.

Example	Description
FILTER PROCESS EQUALS 'Create Purchase Order'	This example would filter for all cases that include the activity "Create Purchase Order".
FILTER YEAR("EVENTLOG"."EVENTTIME") = 2010	This example would filter for all occurrences in 2010.
FILTER "EVENTLOG"."USER_TYPE" != 'BATCH'	This example would filter for non-automated actions in the process.

## Selection Syntax

Selections can only be defined in document load scripts. The selections will be applied when the analysis is loaded. It is also possible to pin selections. Pinned selection cannot be deleted by a viewer. It is only possible to reset them

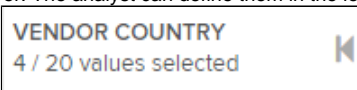
Example	Description
CLEAR SELECTIONS;	This command will delete all selections created by the user, when the analysis is reloaded. A reload is triggered when the analysis is reopened or the page is refreshed.


CLEAR SELECTIONS;  SELECT YEAR("EVENTLOG"." EVENTTIME") = 2010;	This example would create a selection that has selected all entries with in the year 2010 from the eventlogs eventtimes.
CLEAR SELECTIONS;  SELECT YEAR("EVENTLOG"." EVENTTIME") AS "YEAR" = 2010;	This example would create the same selection as our first example, but the selection would be named "Year" in the selections overview.
CLEAR SELECTIONS;  SELECT PINNED YEAR ("EVENTLOG"."EVENTTIME") AS "YEAR" = 2010;	This example create exactly the same selection as in our second example, but the selection is pinned. Pinned selections cannot be deleted but only be reset by the viewers.

## Pinned Selections

Pinned selections are not deletable for the viewer. The analyst can define them in the load script for the analysis. The command "PINNED" will

create a pinned selection from your statement:



The pinned selections cannot be deleted by viewers, but a viewer can reset them with . Otherwise pinned selections will behave like any other selection created from the analysis. They can be changed and edited over the quickselections.

When the analysis is reloaded, the pinned selections will be restored in the same state the analyst defined them.



### CLEAR SELECTIONS;

We highly recommend to start a load script in the analysis setting with the statement: "CLEAR SELECTIONS;". This will delete all previously created selections. Otherwise the created selections of the user will stay in place and the load script's selections will be added on top.

To avoid confusion for the users the old selections should be deleted by starting the load script with "CLEAR SELECTIONS;".

## Process Query Language (PQL)

Process Query Language (PQL) is an extension to the normal SQL used to query databases. PQL has been especially designed to query and filter process flows and process patterns.

Consequently, PQL offers many additional commands to improve the analysis of processes using Celonis Process Mining 4. Besides, PQL supports all standard SQL commands of the used database server (e.g. MS SQL, SAP HANA and Oracle). Therefore all standard commands and functions of the database server can be used in SAP Process Mining by Celonis 4, too.

For details, take a look at the [PQL section](#).

Some of the most common examples:

Statement	Description	Example
A OR B	Returns true if one of the arguments is true, otherwise false.	'true' OR 'false' = 'true'
A AND B	Returns true if both of the arguments are true, otherwise false.	'true' AND 'false' = 'false'
A EQUAL B	Returns true if A is equal to B, otherwise false.	3 = 3 = 'true'
A NOT_EQ TO B	Returns true if A is not equal to B, otherwise false.	4 != 5 = 'true'
A > B	Returns true if A is greater than B, otherwise false.	2 > 4 = 'false'

A >= B	Returns true if A is greater than or equal to B, otherwise false.	3 >= 3 = 'true'
A < B	Returns true if A is less than B, otherwise false.	5 < 3 = 'false'
A <= B	Returns true if A is less than or equal to B, otherwise false.	2 <= 3 = 'true'
A + B	Returns the result of adding the values of A and B.	3 + 3 = 6
A - B	Returns the result of subtracting the values of A and B.	4 - 1 = 3
A * B	Returns the result of multiplying the values of A and B.	2*3 = 6
A / B	Returns the result of dividing the values of A and B.	8 / 2 = 4
X    Y	Returns the interchained strings of the inputs. Result is a string. Numeric value inputs are converted to a string.	PQL is No.'  1 = PQL is No.1
PROCESS EQUALS 'Activity1'	Returns the cases that include the defined activity.	PROCESS_EQUAL 'Delivery'
PROCESS EQUALS START 'Activity 1' TO 'Activity 2' TO ANY TO 'Activity 3' END	Returns the cases that fulfill the defined process. TO ANY can also be included in the process. Another syntax would be:  PROCESS # ^ 'Activity 1' -> 'Activity 2' -> * -> 'Activity 3' \$.  # implicates START, ^ = FROM, -> = TO, * = ANY and \$ = END	PROCESS EQUALS START 'Create Purchase Order' TO ANY TO 'Delivery' END  PROCESS # ^'Create Purchase Order' -> * -> 'Delivery' \$
A LIKE 'abcd'	Returns 'true' if S1 equals S2. Premise: S1 and S2 are strings. Be careful: output is boolean.	LIKE("Windows","Apple") = 'false'
A IN ('abcd', 'abc', 'abcdefg')	All values of A within an intervall (B,C)	(4, 5, 2, 8) BETWEEN 3 AND 5 = 4, 5
ISNULL(X)	Returns '1' if the input column contains a null-element and '0' in all other cases.	ISNULL('Salary') = 1
CASE WHEN X LIKE Y THEN A ELSE B END	Returns A if X is equal to Y. Otherwise, B is returned.	CASE WHEN 'apples' LIKE 'oranges' THEN 'wow' ELSE 'impossible' END