Apparo Flow

How to setup the History feature

Version 3.1.1

1 Basics ____ *3*

2 Prerequisites ____ *3*

3 Enabling the feature ____ *3*

4 Settings of the History feature ____ *5*

4.1 Tab Main ____ **5**

4.1.1 Use columns as row group ____ 5

4.1.2 Decision strategy ____ 5

4.1.3 'Valid from' date column ____ 6

4.1.3.1 Validate valid from ____ 7

4.1.4 'Valid to' date column ____ 8

4.1.5 Use for current flag ____ 8

4.2 Tab Advanced ____ **8**

5 Hands on ____ *9*

5.1 Step 1 - Setting up the History feature ____ **9**

5.2 Step 2 - Preparing the Business Case ____ **10**

5.2.1 Primary key and hidden widgets ____ 10

5.2.2 Setting up TIME_FROM & TIME_TO ____ 10

5.2.3 Setting up Business Case output filter ____ 11

5.2.4 Setting up the history ____ 12

5.2.5 Result ____ 12

5.3 Optional features ____ **13**

5.3.1 Using simple auditing ____ 13

5.3.2 Using delete current data row ____ 13

1 Basics

The history feature is based on slowly changing dimensions type 2 and is intended to track data changes. Instead of overwriting a row when the user changes the value of one or several columns, the history feature creates a new row and marks it as current.

The example:

A car sharing club owns different cars and each car can be used by one member at a time only.
To track who is driving the car right now they are using Flow and a simple database table with the columns ID (which is auto generated when inserting a new row), CAR_NAME, DRIVER_NAME.

Now they also want to track who had the car recently, they decided to use the History feature, because of its capability to track data changes on selected dimensions.

2 Prerequisites

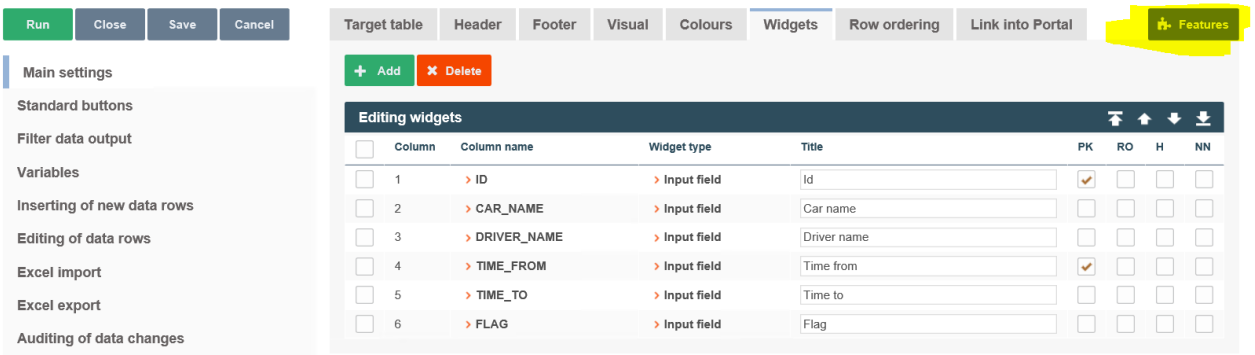
The target table of the Business Case requires 2 additional columns of type ‘date’ and one optional column for the current flag of type e.g. varchar or number.

Using an additional column ‘current flag’ makes it later far easier to filter the Business Case for the ‘current rows’ only.

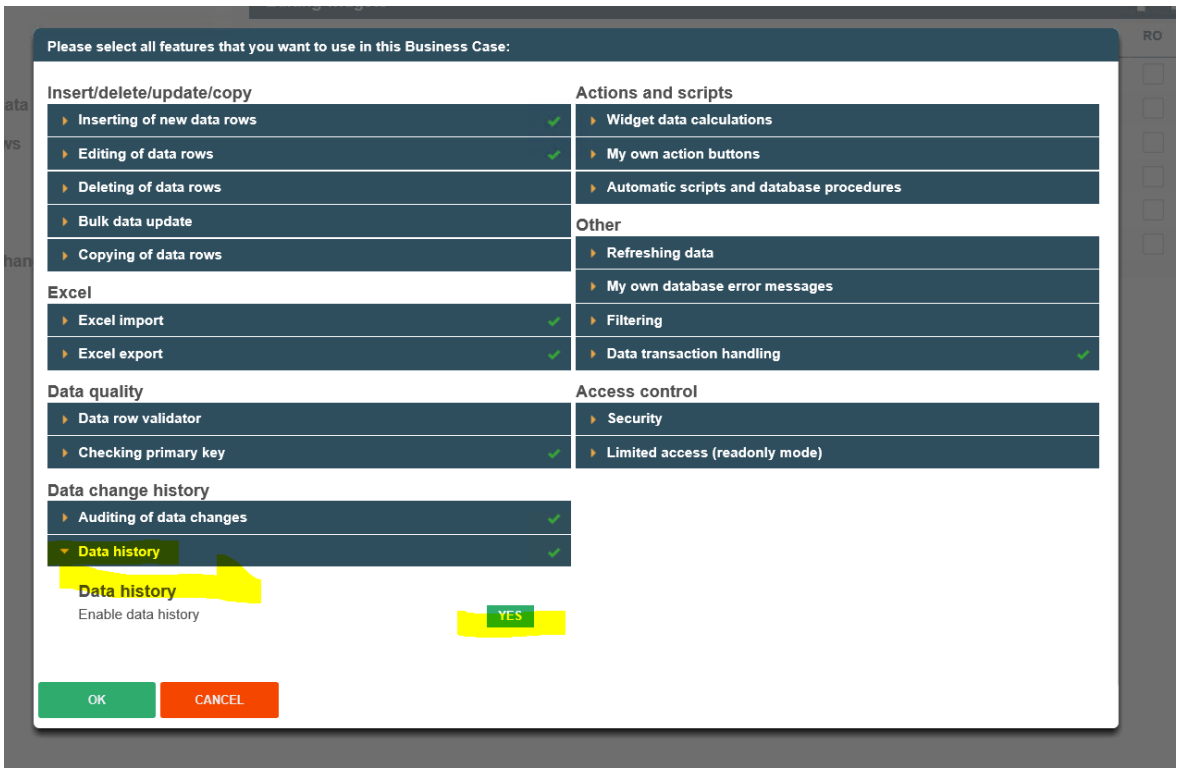
The feature ‘Inserting of new data rows’ must be enabled, including the checkbox ‘Insert area is able to update an already existing data row’.

3 Enabling the feature

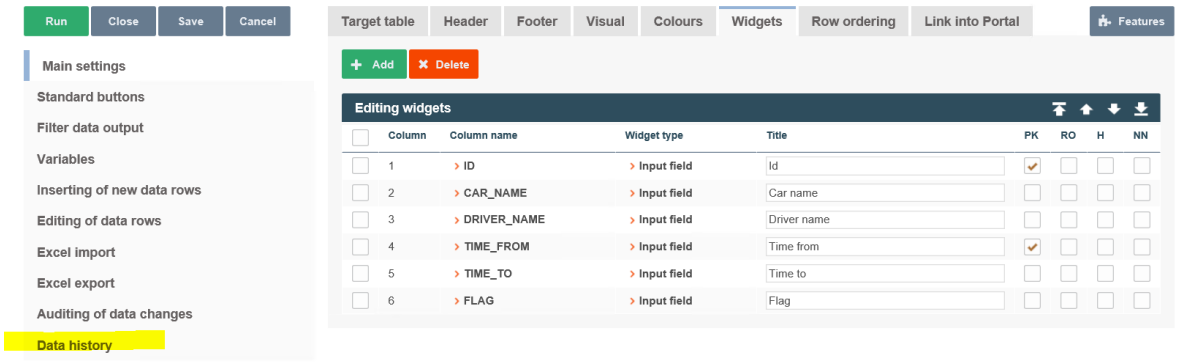
All features can be enabled/disabled using the ‘Features Button’ in the top right corner.



Now click on ‘Data history’ and enable the feature by clicking on the ‘No’ button. It turns to ‘Yes’ and shows that the feature is enabled now.



After clicking the ‘OK button’ the widget edit view refreshes and the menu ‘Data history’ appears.



4 Settings of the History feature

4.1 Tab Main

4.1.1 Use columns as row group

The row group can be imagined as the 'primary key' group of the History group.
It is used to identify the data rows belonging to a history group and can contain one or several columns.

Hints:

- The row group should build a unique identifier (like a primary key).
- Take sure that users cannot alter the values of the row group. If values of the history group are changing, this will create new history group (a new entry).

In our example the CAR_NAME is unique and can be used for the history group.
Changing any other value will create a new row for this history group and mark it as current, the old values are historized and in the most cases hidden with filters.

Hint:

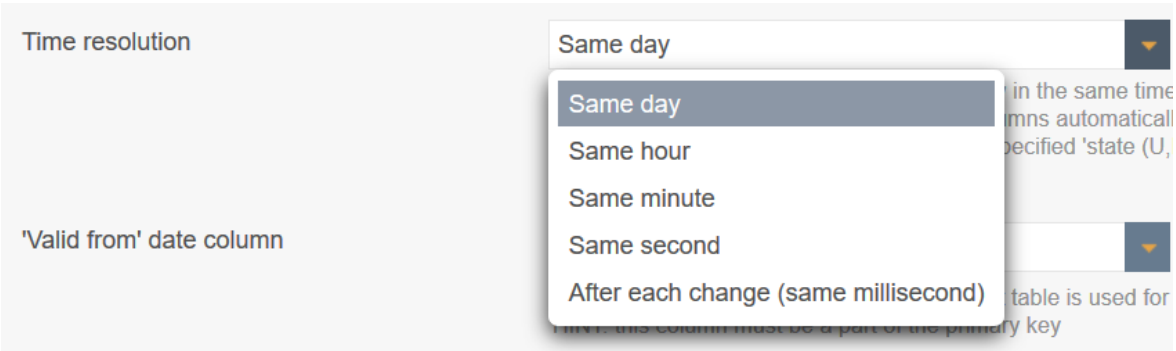
Changing CAR_NAME will create a new history group, since the unique identifier has changed.

In our simple example is 'any other value' the column DRIVER_NAME only, since all other dimensions cannot be used for the history group.

You must not use for the history group:

- Auto computed columns (as the ID column, which is generated by the DB)
- All columns mapped to the history feature (as TIME_FROM, TIME_TO, FLAG)

4.1.2 Time resolution



Sets the timeframe, within a correction (= row update) without historizing is allowed.

Let's say this property is set to '*Same minute*', which means if the user is changing the history row within one minute, then no new row is created for the history feature. '*One hour*' allows corrections withing one hour and '*After each change (same millisecond)*' will always create a history row.

Double entries comparing with current data rows are ignored automatically (no update, no insert).

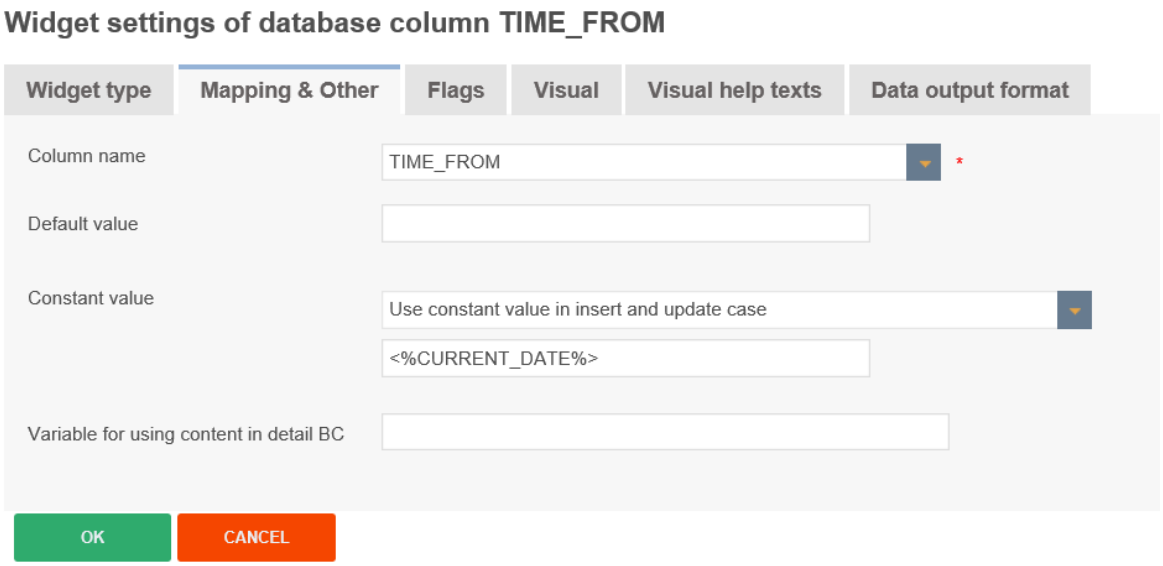
4.1.3 'Valid from' date column

This property needs to be mapped to the 'TIME_FROM' widget.
It is storing the valid from value of the history feature.

Hint:

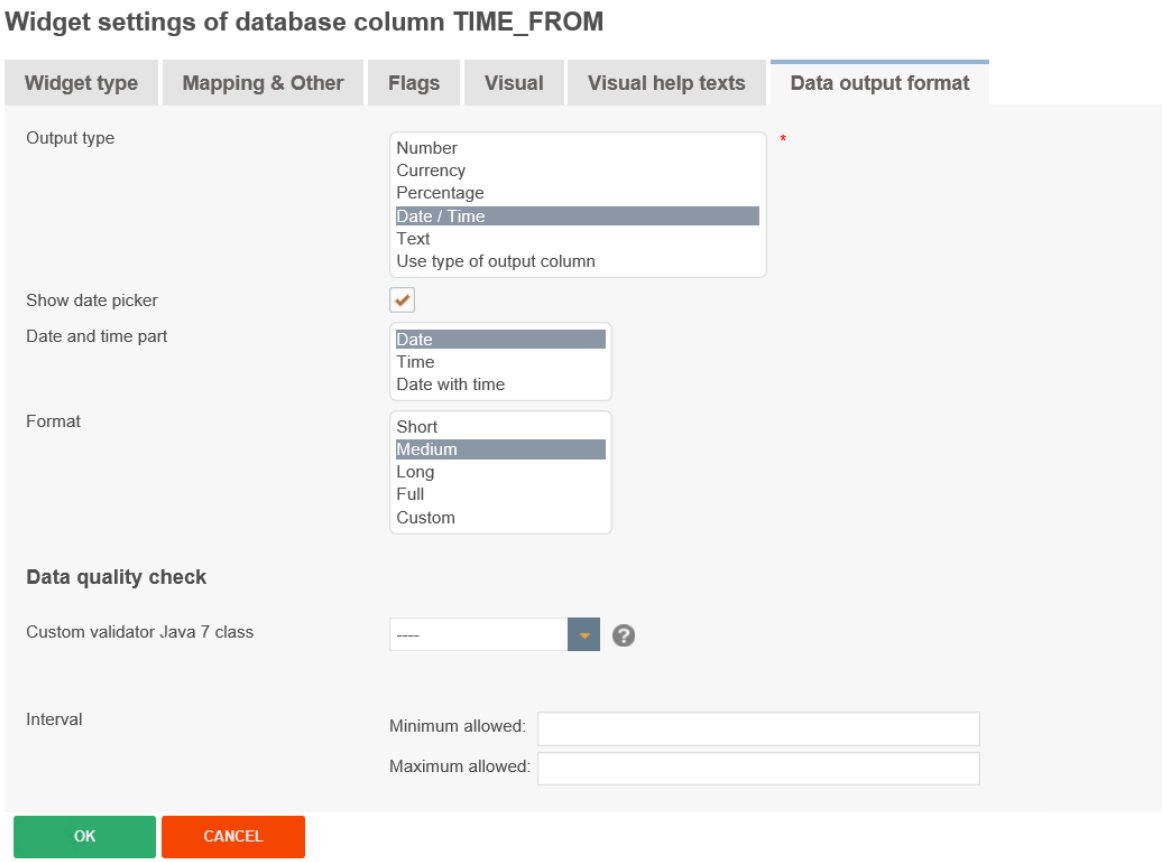
It is important that the widget for the valid from property of the history feature (TIME_FROM in our example) is part of the primary key of the Business Case.

In the most cases the **valid from** date will be generated automatically by the Business Case, using the variable <%CURRENT_DATE%> as constant value in insert and update case in the widget settings of the TIME_FROM widget (tab Mapping & Other) or this date is read from Excel document or manual entry.



Also important is the correct output format of this widget.
If the history feature is set to 'Same second' and the widget output format is storing only date & hours + minutes, it cannot work properly due the missing seconds.

We recommend to set the widgets output format to at least ‘Date with Time Medium’
Please note ‘Same day’ requires an output format of date without time.



This setting returns a date in the format ‘Oct 25, 2016 9:17:13 AM’

Hint:

For milliseconds use the custom output type e.g. 'dd.MM.yyyy:hh.mm.ss.SSS'

4.1.3.1 Validate valid from

The valid from property also contains the checkbox ‘Validate that the new 'valid from' value precede the latest existing 'valid from' value of this row group’.

Example:

The user is selecting the valid from date manually by a date picker (and not using the <%CURRENT_DATE%> variable as constant value), in this case we recommend to activate the checkbox, to make sure that a new entry will be valid (which requires a newer valid from date).

4.1.4 'Valid to' date column

In our example we map here the widget 'TIME_TO'.
Flow is automatically inserting the values for this widget.
A new history row is set to the 'infinity date' (which we will explain later) and the old historized row get's a timestamp, which shows that the history row is not longer the 'current' row.

Id	Car name	Driver name	Time from	Time to	Flag
2	Porsche-Cayenne	Silvio Speed	Oct 25, 2016 9:17:13 AM	Oct 25, 2016 9:17:35 AM	0
3	Porsche-Cayenne	Rich Racer	Oct 25, 2016 9:17:36 AM	Dec 31, 2999 12:00:00 AM	1

We changed the dimension 'Driver name' to Rich Racer at 9:17:36 AM (row with ID=3), the system created a new row with infinity date (row with ID=3) and sets the valid to date (time to) of the old row to 9:17:35 AM and marks it that way as not current. Reading time from and time to we know this row was the current history row from 9:17:13 AM to 9:17:35 AM.

Hint:

We recommend to set the data output format of the valid to widget (TIME_TO) the same way as the valid from widget (TIME_FROM).
In our example to 'Date with Time (medium)'.

4.1.5 Use for current flag

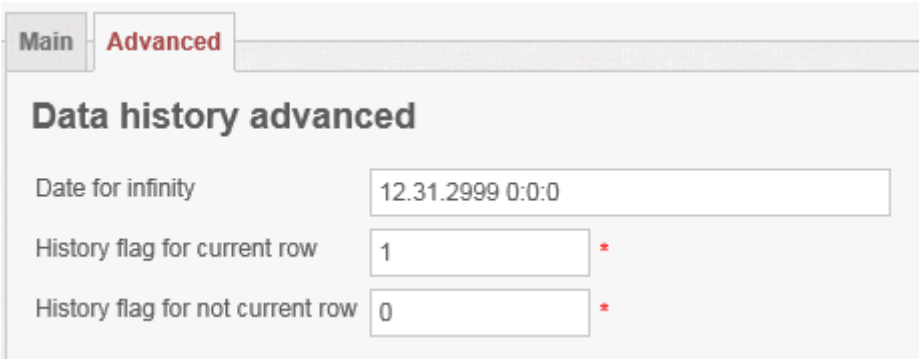
This optional property can be mapped to the current flag widget.
In the most cases the current history row will be set to 1 (when using 0 and 1 for current flag).
It allows to easier filter the Business Case.

Example:

When we use this filter: FLAG = 1, then all historized rows will not be shown to the end user in the Business Case.

4.2 Tab Advanced

It contains the settings for the infinity date and for the current flag.

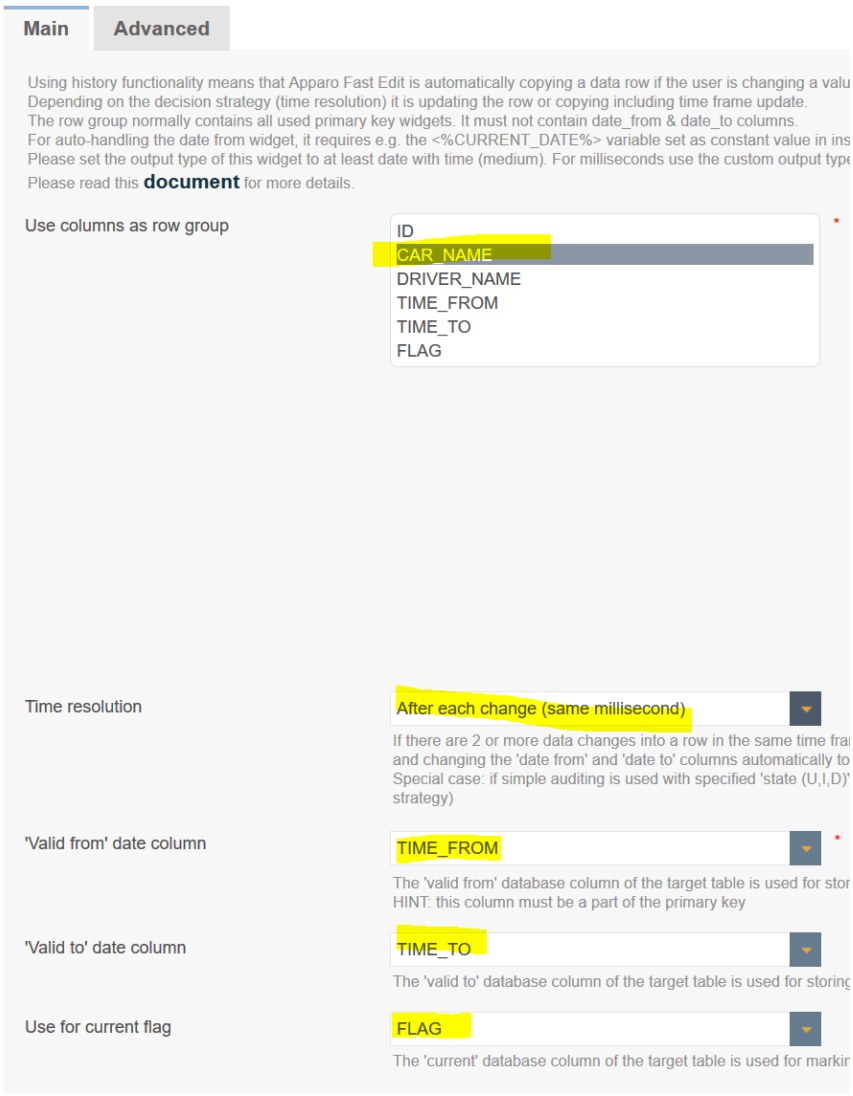


The infinity date is always used automatically to mark the current row.

5 Hands on

We will now continue with our example and setup the Business Case step by step.
As preparation we already created the Business Case, including the additional columns for the history feature.

5.1 Step 1 - Setting up the History feature



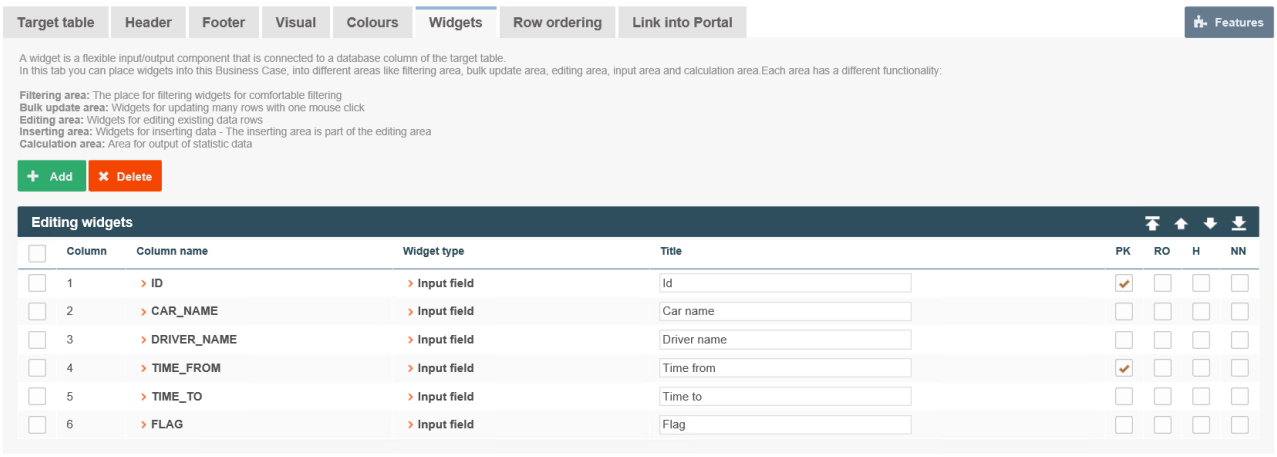
With CAR_NAME as history group, we are historizing any changes in the widget DRIVER_NAME.
Time resolution same second is storing any change when using manual input.

Hint:

ID cannot be part of the history group; the database is computing the value.

5.2 Step 2 - Preparing the Business Case

5.2.1 Primary key and hidden widgets

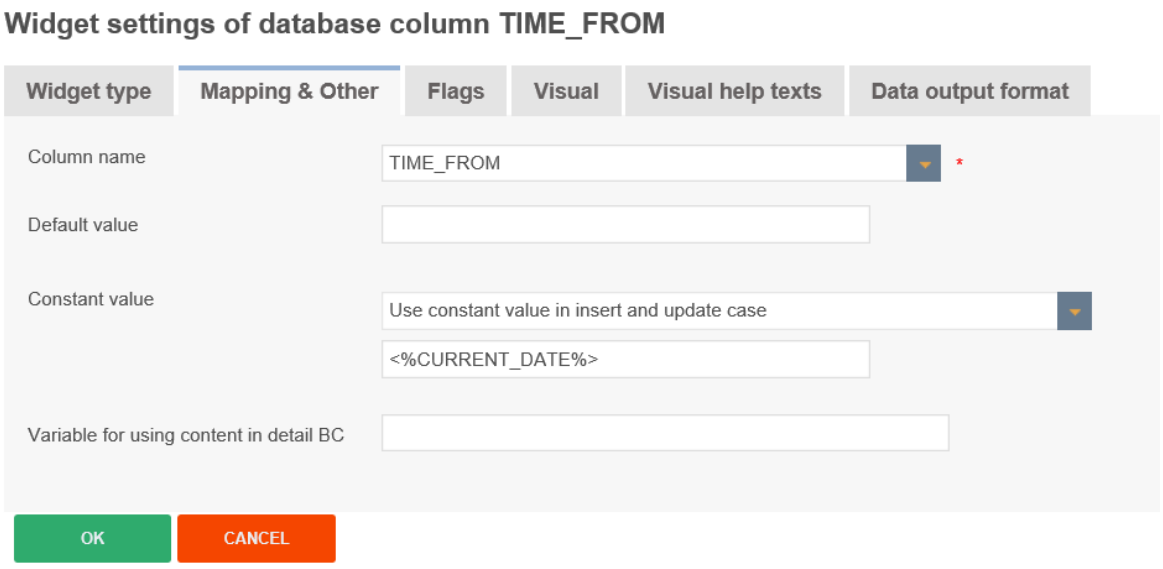


The primary key is the ID and TIME_FROM.

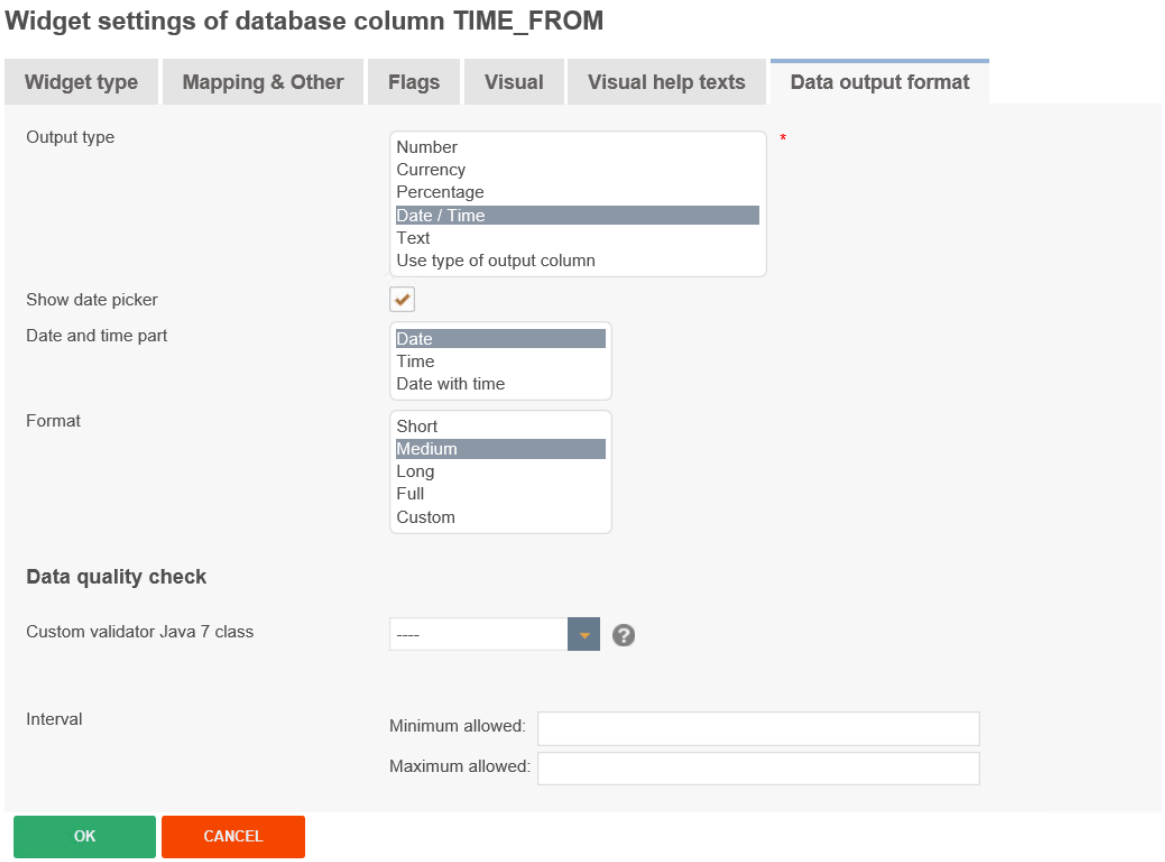
ID can be part of the primary key but not part of the history group, the primary key combination of the Business Case can differ from the primary key of the database table.

ID, TIME_FROM, TIME_TO and FLAG are set to hidden, users don't need to change them.

5.2.2 Setting up TIME_FROM & TIME_TO

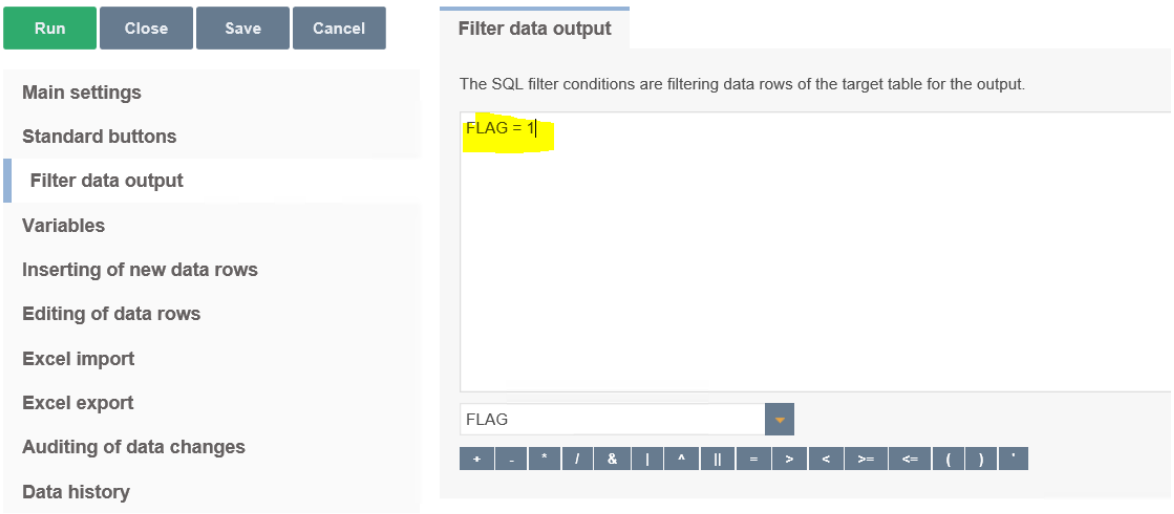


We insert CURRENT_DATE as constant value in insert and update case in the widget TIME_FROM. This allows Flow to automatically manage the history properties 'valid from' and 'valid to'.



We set the data output format of the widgets **TIME_FROM** and **TIME_TO** to Date with Time (medium)

5.2.3 Setting up Business Case output filter



FLAG=1 filters all historized rows, since those have the value 0 in the column FLAG.

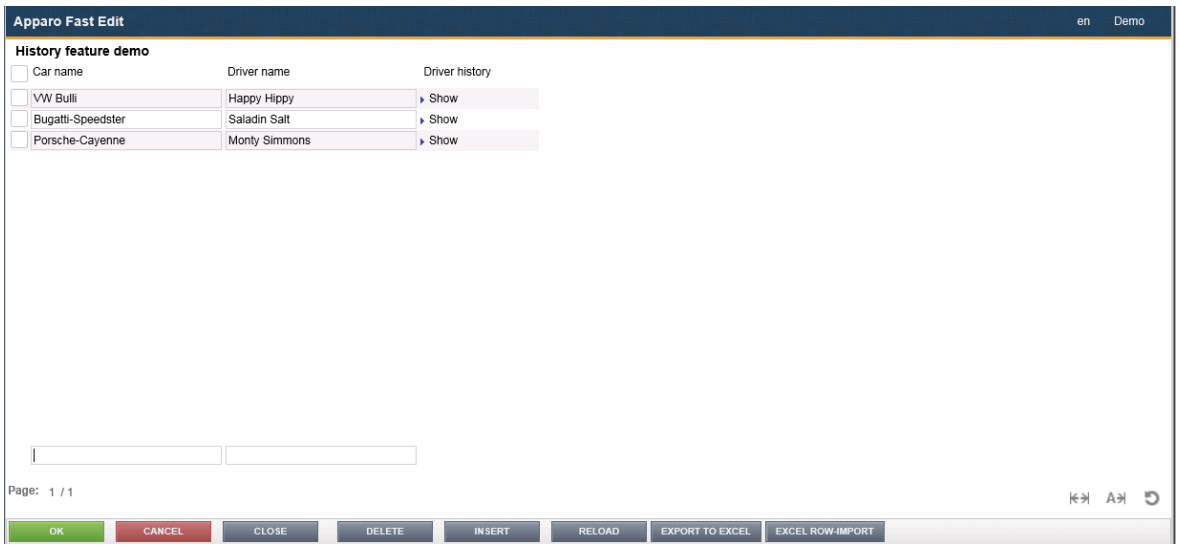
5.2.4 Setting up the history

As last part we want to add a driver history, that shows all recent drivers, which is technically seen a list of all historized rows.

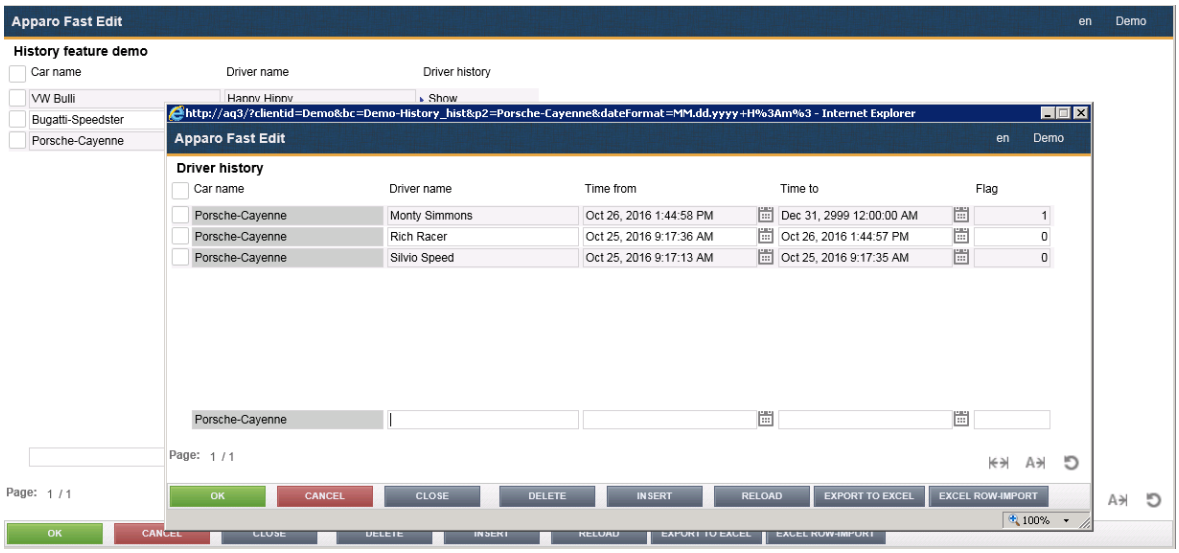
We simply use an adapted copy of the Business Case, removed the filter and the history feature and created a hyperlink widget to this Business Case.

5.2.5 Result

For the user, the history feature works totally hidden.



Clicking the Driver history link opens a popup window with all recent drivers (historized data rows)

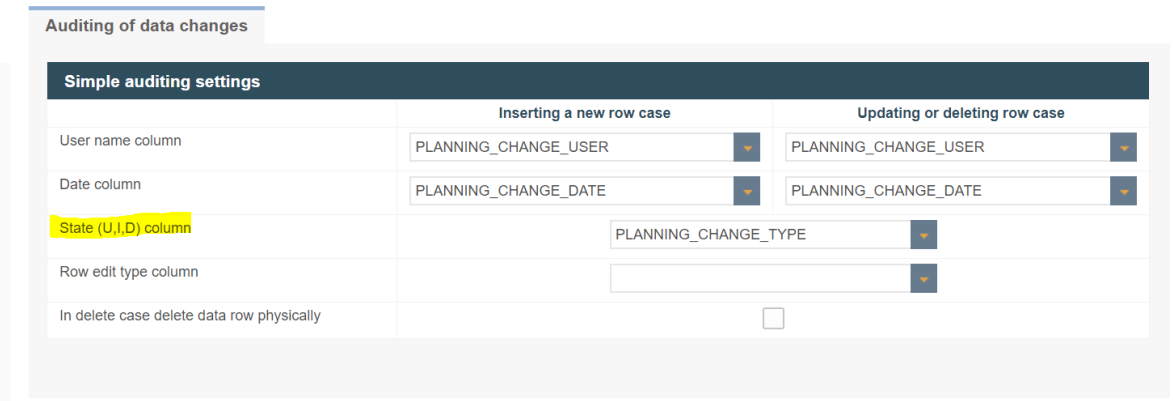


5.3 Optional features

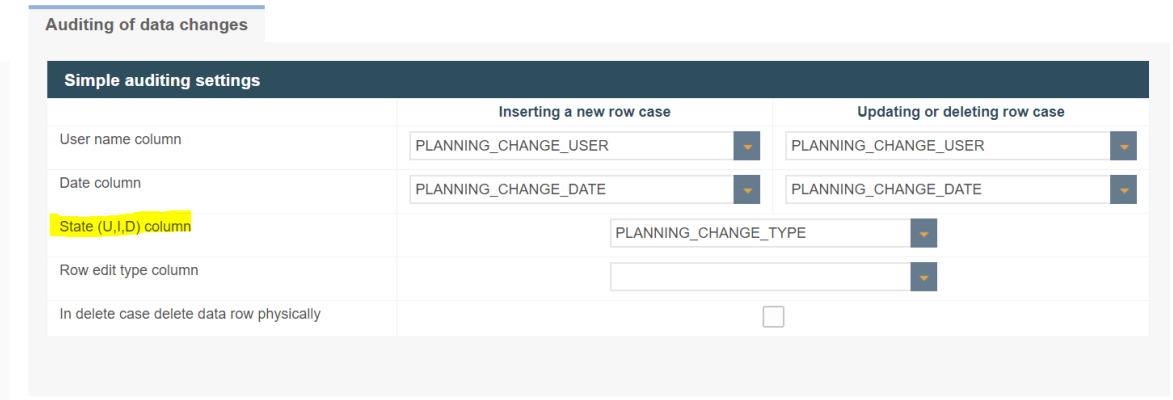
5.3.1 Using simple auditing

Simple auditing has a different behavior if history feature is enabled.

If a new data row is inserted and this is the first version of a history then state column contains “I”.
If a data row is changed and a new data row (new version) is created then state column contains “U”.



5.3.2 Using delete current data row

It is possible to delete the current data row of a history.
In this case the “Simple Auditing” feature must be enabled and the column “State” must be used.

If the user is selecting the current data row and is pressing the “Delete” button then a copy of this data row is getting the status “D” and time-from and time-to is changed to current date.