avaloq
An NEC Company

Foreign Exchange

Customization Upgrade Guide
Release 5.4
Area: FX & Money Market

Latest version of this document

The latest version of this document can be found at https://docs.avaloq.com

Feedback

Please send any feedback to documentation@avaloq.com

Avaloq Evolution Ltd. Allmendstr. 140 ¦ CH-8027 Zürich ¦ Switzerland

Version History

While the document ID of a Customization Upgrade Guide remains the same, the content is different in every release. This is because the document describes the customization adaptations that are required between two specific releases. To preserve your existing customization, read the entire document and make the necessary adaptations before you upgrade your system to a new release.

The sections below are new or updated:

Version / Date	Section	Description of the change
5.4v0 / 20 April 2022		This is a new document.

Contents

1 Introduction

This document describes the changes necessary to keep the existing functionality of components in the FX & Money Market area after an upgrade of the system to Release 5.4.

This document is for customization specialists.

2 Upgrade steps

2.1 Base parameter item: map_maturity_to_val_date

In Release 5.4 the default value of the `map_maturity_to_val_date` item of the `avq.doc.fxtr` base parameter has changed from **"+"** to **"-"**. This is to align with the desired behaviour going forward for new clients. For a description of the item, see *FX Trade - Customization Guide (doc ID: 2131)*.

If you haven't explicitly set this base parameter item to "+" or "-" in your customization (therefore relying on the default value), you must now **explicitly set the value to "+"** in Release 5.4 to keep the current behaviour of mapping the maturity date to `doc.val_date` in the FXTR business type.

If you previously had `map_maturity_to_val_date` set to "+" and are switching to "-", you must:

- Check your customization to ensure that the maturity date of FXTR orders is always identified by `doc_fxtr.maturity_date` and not by `doc.val_date`
- Perform a mandatory data migration of existing forex forward orders

For more on these steps, see "Switching map_maturity_to_val_date from "+" to "-" (optional)" below.

If you already have `map_maturity_to_val_date` set to "-" on your current release, this change does not affect you.

2.2 Switching map_maturity_to_val_date from "+" to "-" (optional)

This section is only relevant if you previously had `map_maturity_to_val_date` set to "+" and want to change to "-". If you want to continue using the previous logic, just make sure `map_maturity_to_val_date` is set to "+" in your customization.

The following table gives examples of how dates are set depending on the setting of the base parameter item (Opening Order FX Forward on 07.07.2017 with Maturity 09.08.2017):

Field	Base par item = "+"	Base par item = "-"
`doc_fxtr.trx_date`	07.07.2021	07.07.2021
`doc_fxtr.spot_date`	09.07.2021	09.07.2021
`doc_fxtr.val_date`	**09.08.2021**	07.07.2021
`doc_fxtr.maturity_date`	09.08.2021 or NULL (depends on customization)	**09.08.2017**

If you are switching `map_maturity_to_val_date` from "+" to "-", follow the steps described below.

Check customization

If you have created your own forms, or you generate FX trades from messages, make sure you set the maturity of the FXTR order in `doc_fxtr.maturity_date`. When consuming the maturity date in customization, make sure it is always identified by `doc_fxtr.maturity_date` and not by `doc.val_date`.

Migrate existing orders

Migrate existing FXTR orders to update the dates as they would have been set with the new base parameter value. Not doing this could lead to errors in areas like asset evaluation.

To migrate existing orders:

1. **Read the current last pillar date**

   Run the following query with your specific bu_ids:

   ```
   SELECT last_pillar FROM base WHERE bu_id = :bu_id
   ```

   You will need this date in step **4**. If the `last_pillar` date is not set (the query returns null), you can skip steps **2** and **4** below

2. **Run the Base Utility task (task ID: 802)**

   

   You can skip this step if the query in step **1** returned null.

   Source: TASK_BASE_UTIL

   Set the **Pillar Date** parameter to a value like "-1yv". It is important that you go further back in history than the minimum maturity date you will set in step **3** below.

   Do not set any of the other parameters for the task.

3. **Run a migration script**

   In the script below, enter the minimum maturity date (`c_maturity_date`) and additional order type groups (if you use others to open FXTR orders) to select the required FXTR orders.

   Note that BGP 928 must be up and running to do the bookings.

   

   You can first run the script without any modifications by commenting out the following part:

   ```
   update doc_fxtr

   set    maturity_date = c.old_val_date
   where doc_id = c.doc_id;
   be#.evt#move(
     i_evt_id        => c.evt_id
    ,i_new_book_date => l_new_book_date
    ,i_new_val_date  => c.new_val_date
   );
   ```

   This allows you to check the log entries with the affected asset_ids. When you are happy, you can undo the comments and run the script.

   The migration script:

   ```
   declare
     c_maturity_date                       constant date    := to_date('xx.xx.xxxx', 'dd.mm.yyyy'); --
   minimal Maturity Date that is updated
     --
     c_vbda_activ_on_instn                 constant boolean := be#.vbda#is_activ_on_instn;
     l_new_book_date                               date;
     l_bu_id                                       pls_integer;
   begin
     install#.log#write('Start val_date migration script');
     install#.log#write('   Migrates trades, which have maturity later or equal >'||c_maturity_
   date||'<');
     session#.open_session;
     for c in (
       select dc.id         doc_id
   ```

```
          ,dc.val_date   old_val_date
          ,dc.trx_date   new_val_date
          ,ev.id         evt_id
          ,ev.book_date
          ,ev.done_date
          ,dc.bp_imed_id bu_id
          ,oa.obj_id     asset_id
    from doc       dc
        ,obj_asset oa
        ,evt3      ev
    where dc.order_type_grp_id in ( def_order_type_grp_fxtr.ofx
                                   ,def_order_type_grp_fxtr.opn_nov_in
                                   ,def_order_type_grp_fxtr.opn_ndf
                                   ,def_order_type_grp_fxtr.opn_ndf_nov_in
                                   ,def_order_type_grp_fxtr.opn_trsy_swap
                                  ) - If you use additional order type groups for opening FXTR orders
add them to this list
      and oa.obj_id            = dc.asset_id
      and dc.val_date          = oa.maturity_date
      and oa.maturity_date    >= c_maturity_date
      and ev.doc_id            = dc.id
      and ev.id                > 0
      and ev.pko_id            = def_be_pko.tech
      and ev.evt_status_id    != def_evt_status.sim
      and not exists (select null from evt3 where id = -ev.id)
    order by dc.bp_imed_id
  ) loop
    if   l_bu_id is null
      or l_bu_id != c.bu_id
    then
      session#.open_session(i_bu_id => c.bu_id);
      l_bu_id := c.bu_id;
    end if;
    if c_vbda_activ_on_instn then
      l_new_book_date := greatest(c.done_date, c.new_val_date);
    end if;
    install#.log#write('   Asset >'||c.asset_id||'< : Moving value date of order #' || c.doc_id || '
from ' || c.old_val_date || ' to '|| c.new_val_date ||', moving book date from ' || c.book_date || '
to ' || l_new_book_date);
    update doc_fxtr

    set    maturity_date = c.old_val_date
    where doc_id = c.doc_id;
    be#.evt#move(
      i_evt_id        => c.evt_id
     ,i_new_book_date => l_new_book_date
     ,i_new_val_date  => c.new_val_date
    );
  end loop;
  install#.log#write('End val_date migration script');
end;
```

4. **Rerun the Base Utility task (task ID: 802)**

   

   You can skip this step if the query in step **1** returned null.

   Enter the pillar date that you obtained in step **1** and rerun the task to set the system back to its original state.

5. **Run the Evaluation of Financial Products task (task ID: 1848)**

   Source: TASK_EVAL

Run this task for the FXTR asset class with the appropriate scenario ("std" is used on the MDB). Check for errors of type "the capital is already redeemed".

If these errors occur, check the order type group of the related orders and add this order type group to the script in step **3**. Then redo steps **1–5** and check the log entries again.

6. **Recalculate desired offline pillars**

   For all online pillars of date type "value date" or "book date", the incremental recalculation is automatically triggered due to the value and book date modification of the FXTR orders.

   For the offline pillars – like the balance pillar using book date – you must decide if you want to recalculate them or not. For example, you probably would *not* want to recalculate a year-end (book date) balance pillar, because the balance sheet has already been published.

   The veri date–based balance pillar is not affected: all pillars (online and offline) that use date types other than "value date" or "book date" (e.g. "veri date") **are not affected by the migration** and will not change.

Lot accounting: FXTR only has a "position accrual" lot, which is affected in the same way as the position in a "book date" balance pillar: it will appear or disappear in line with the position bookings.