# Alert Calculation: Step Description

Alert Calculation step is aimed to generate alert flags for generated forecast values. Alert can be calculated based on any forecast versions generated in the system namely:

* Disaccumulated and Disaggregated Hybrid Forecast
* Disaggregated Hybrid Forecast
* Hybrid Forecast
* Statistical (VF) Forecast
* ML Forecast

This step is needed if business process in the customer side assumes user manual control of the analytical forecast.

### Code Realization Requirements

The code should be created in a form of a SAS macro set on SAS VIYA platform.

# Input Data

The initial data for the step is listed below. These tables should be present in the system before this step for example in STG or in DDS area.

• IN\_PRODUCT (see data requirements)

• IN\_LOCATION (see data requirements)

## ACC\_AGG\_HYBRID\_FORECAST\_<target variable postfix>

Hybrid Forecast values that is disaggregated to sku/location level and disaccumulated to daily granularity for particuler.

|  |  |
| --- | --- |
| **ACC\_AGG\_HYBRID\_FORECAST** | |
| Column Name | Description |
| **PRODUCT\_LVL\_ID<m>** | Level m =**IB\_VF\_PRODUCT\_LVL**of product hierarchy |
| **LOCATION\_LVL\_ID<n>** | Level n =**IB\_VF\_LOCATION\_LVL** of location hierarchy |
| **CUSOMER\_LVL\_ID<k>** | Level k =**IB\_VF\_CUSTOMER\_LVL** of customer hierarchy |
| **DISTR\_CHANNEL\_LVL\_ID<l>** | Level l =**IB\_VF\_DISTR\_LVL** of distribution channel hierarchy |
| **PERIOD\_DT** | Date of sales (calendar day) |
| **SEGMENT\_NAME** | Time Series VF Segment Name |
| **VF\_FORECAST\_VALUE** | VF forecast value |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new or old |
| **ML\_FORECAST\_VALUE** | ML forecasted value |
| **HYBRID\_FORECAST\_VALUE** | HYBRID forecast value |

## AGG\_HYBRID\_FORECAST\_<target variable postfix> (optional)

Hybrid Forecast values that is disaggregated to sku/location level.

|  |  |
| --- | --- |
| **AGG\_HYBRID\_FORECAST** | |
| Column Name | Description |
| **PRODUCT\_LVL\_ID<m>** | Level m =**IB\_VF\_PRODUCT\_LVL**of product hierarchy |
| **LOCATION\_LVL\_ID<n>** | Level n =**IB\_VF\_LOCATION\_LVL** of location hierarchy |
| **CUSOMER\_LVL\_ID<k>** | Level k =**IB\_VF\_CUSTOMER\_LVL** of customer hierarchy |
| **DISTR\_CHANNEL\_LVL\_ID<l>** | Level l =**IB\_VF\_DISTR\_LVL** of distribution channel hierarchy |
| **PERIOD\_DT** | Date of sales (calendar day) |
| **SEGMENT\_NAME** | Time Series VF Segment Name |
| **VF\_FORECAST\_VALUE** | VF forecast value |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new or old |
| **ML\_FORECAST\_VALUE** | ML forecasted value |
| **HYBRID\_FORECAST\_VALUE** | HYBRID forecast value |

## HYBRID\_FORECAST\_<target variable postfix> (optional)

Hybrid Forecast values that at original granularity level.

|  |  |  |
| --- | --- | --- |
| **HYBRID\_FORECAST** | | |
| Column Name | Description |  |
| **PRODUCT\_LVL\_ID<m>** | Level m =**IB\_VF\_PRODUCT\_LVL**of product hierarchy |  |
| **LOCATION\_LVL\_ID<n>** | Level n =**IB\_VF\_LOCATION\_LVL** of location hierarchy |  |
| **CUSOMER\_LVL\_ID<k>** | Level k =**IB\_VF\_CUSTOMER\_LVL** of customer hierarchy |  |
| **DISTR\_CHANNEL\_LVL\_ID<l>** | Level l =**IB\_VF\_DISTR\_LVL** of distribution channel hierarchy |  |
| **PERIOD \_DT** | Date of sales (calendar day) |  |
| **SEGMENT\_NAME** | Time Series VF Segment Name |  |
| **VF\_FORECAST\_VALUE** | VF forecast value |  |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |  |
| **ASSORTMENT\_TYPE** | new, or old |  |
| **ML\_FORECAST\_VALUE** | ML forecasted value |  |
| **HYBRID\_FORECAST\_VALUE** | HYBRID forecast value |  |

## VF\_FORECAST\_<target variable postfix> (optional)

Forecast values from VF forecast step that is disaggregated to sku/location level and disaccumulated to daily granularity.

|  |  |  |
| --- | --- | --- |
| **VF\_FORECAST\_TABLE** | | |
| Column Name | Description |  |
| **PRODUCT\_LVL\_ID<m>** | Level m =**IB\_VF\_PRODUCT\_LVL**of product hierarchy |  |
| **LOCATION\_LVL\_ID<n>** | Level n =**IB\_VF\_LOCATION\_LVL** of location hierarchy |  |
| **CUSOMER\_LVL\_ID<k>** | Level k =**IB\_VF\_CUSTOMER\_LVL** of customer hierarchy |  |
| **DISTR\_CHANNEL\_LVL\_ID<l>** | Level l =**IB\_VF\_DISTR\_LVL** of distribution channel hierarchy |  |
| **PERIOD\_DT** | Date of sales (week) |  |
| **FORECAST\_VALUE** | VF forecast value |  |

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## VF\_SEGMENTS\_<target variable postfix> (optional)

Name of Segments for each TS in VF\_FORECAST\_TABLE

|  |  |  |
| --- | --- | --- |
| **VF\_SEGMENTS** | | |
| Column Name | Description |  |
| **PRODUCT\_LVL\_ID<m>** | Level m =**IB\_VF\_PRODUCT\_LVL**of product hierarchy |  |
| **LOCATION\_LVL\_ID<n>** | Level n =**IB\_VF\_LOCATION\_LVL** of location hierarchy |  |
| **CUSOMER\_LVL\_ID<k>** | Level k =**IB\_VF\_CUSTOMER\_LVL** of customer hierarchy |  |
| **DISTR\_CHANNEL\_LVL\_ID<l>** | Level l =**IB\_VF\_DISTR\_LVL** of distribution channel hierarchy |  |
| **SEGMENT\_NAME** | VF segment name |  |

|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## ML\_FORECAST\_<target variable postfix> (optional)

Forecast values from ML forecast step, this table include only those triples PRODUCT\_ID|LOCATION\_ID|PERIOD\_DT.

|  |  |
| --- | --- |
| **ML\_FORECAST** | |
| Column Name | Description |
| **PRODUCT\_LVL\_ID<m>** | Level m =**IB\_VF\_PRODUCT\_LVL**of product hierarchy |
| **LOCATION\_LVL\_ID<n>** | Level n =**IB\_VF\_LOCATION\_LVL** of location hierarchy |
| **CUSOMER\_LVL\_ID<k>** | Level k =**IB\_VF\_CUSTOMER\_LVL** of customer hierarchy |
| **DISTR\_CHANNEL\_LVL\_ID<l>** | Level l =**IB\_VF\_DISTR\_LVL** of distribution channel hierarchy |
| **PERIOD\_DT** | Date of sales (calendar day) |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | ‘new’ or ‘old’ |
| **FORECAST\_VALUE** | ML forecasted value |

## RESTORED\_DEMAND\_<target variable postfix>

Demand information regarding the past till last known day of the history.

|  |  |
| --- | --- |
| **DM.RESTORED\_DEMAND** | |
| Column Name | Description |
| **PRODUCT\_ID** | Product ID (the lowest level of the product hierarchy) |
| **LOCATION\_ID** | Location ID |
| **CUSOMER\_ID** | Customer id |
| **DISTR\_CHANNEL\_ID** | Distribution Channel ID |
| **PERIOD\_DT** | Date of sales (calendar day) |
| **SALES\_QTY\_R** | Restored demand |
| **PROMO\_FLG** | 1|promo event was active  0|no promo event |
| **SALES\_QTY** | Total sales in units per day (w/o returns) |
| **STOCK\_QTY** | Stock qty (BOP) |
| **DEFICIT\_FLG1** | 1|primary deficit occurred  0|no primary deficit |
| **DEFICIT\_FLG2** | 1|secondary deficit occurred  0|no secondary deficit |

## FORECAST\_FLAG

Product/location lifecycle history containing the following fields is used as an input:

|  |  |
| --- | --- |
| **DM.FORECAST\_FLAG** | |
| Column Name | Description |
| **PRODUCT\_ID** | Product ID (the lowest level of the product hierarchy) |
| **LOCATION\_ID** | Location ID |
| **CUSOMER\_ID** | Customer id |
| **DISTR\_CHANNEL\_ID** | Distribution Channel ID |
| **PERIOD\_START\_DT** | Period start date |
| **PERIOD\_END\_DT** | Period end date |
| **STATUS** | One of the following status: active, blocked, out-of-sale |

## CONFIG\_PARAMETERS

The following config parameters are used within the step.

### ALERT\_PARAMETERS

|  |  |  |
| --- | --- | --- |
| **CONFIG.****ALERT\_PARAMETERS** | |  |
| Column Name | Description |  |
| **Tgt\_type** | One of 3 types of the target variable:   * SELLIN – means CPG sales to its customer, * SELLOUT – means CPG’s customer sales to their clients,   POS – means sales in the point of sales, can be relevant for both Retailer and CPG | POS |
| **alert\_id** | Alert ID |  |
| **al\_product\_lvl** | Data aggregation level regarding product hierarchy, e.g. 8 means PRODUCT\_ID , 7 means PRODUCT\_LVL\_ID7, 6 means PRODUCT\_LVL\_ID6, etc |  |
| **al\_location\_lvl** | Data aggregation level regarding location hierarchy, e.g. 6 means LOCATION\_ID , 5 means LOCATION\_LVL\_ID5, 4 means LOCATION \_LVL\_ID4, etc |  |
| **al\_customer\_lvl** | Data aggregation level regarding customer hierarchy, e.g. 6 means CUSTOMER\_ID , 5 means CUSTOMER\_LVL\_ID5, 4 means CUSTOMER\_LVL\_ID4, etc |  |
| **al\_distr\_channel\_lvl** | Data aggregation level regarding distribution channel dimension hierarchy, e.g. 2 means DISTR\_CHANNEL\_ID , 1 means DISTR\_CHANNEL\_LVL\_ID1 |  |
| **al\_tim\_lvl** | Data aggregation level regarding time dimension, e.g. day, month, week.2, etc |  |
| **alert\_threshold\_val** | Critical alert threshold value |  |
| **Input\_table\_table** | Name of the table and target variable for which alert must be calculated |  |
| **Input\_column** | Name of the target in input variable for which alert must be calculated |  |

Default structure of the ALERT\_PARAMTERS config table see in alert\_parameters file.

|  |  |
| --- | --- |
| **CONFIG.CONFIG\_PARAMETERS** | |
| Column Name | Description |
| **IB\_ALERT\_GRANULARITY** | Granularity that must be included into alert calculation, default = SKU/All StoresLocations/WEEK |
| **IB\_ALERT\_FORECAST\_LIST** | Which output forecast tables must be used for alert calculation |
| **IB\_ALERT\_MINCRITICAL\_VALUE** | Minimal value that can be meet as a forecast for active assortment |
| **IB\_ALERT\_MAXCRITICAL\_RATIO** |  |
| **IB\_ALERT\_MINCRITICAL\_RATIO** |  |
| **IB\_ALERT\_BASE\_PAST\_PERIOD** |  |
|  |  |

### CONFIG

|  |  |
| --- | --- |
| **CONFIG parameters init** | |
| Column Name | Description |
| **IB\_FF\_ACTIVE\_STATUS\_LIST** | List of statuses in FORECAST\_FLAG table that relates to active life period, default = (‘active’) |
| **IB\_FC\_HORIZ** | Forecast horizon in weeks |
| **IB\_ALERT\_MIN\_VAL** | Minimal value that can be used as a benchmark for comparison with forecast, default = 0.1 |
| **IB\_ALERT\_MIN\_OBS** | Minimal number of observations to calculate a benchmark for comparison, default = 10 |
| **IB\_MAX\_NP\_HISTORY** | Maximal length of new product length in days, default = 30 |

## INITIAL\_GLOBAL parameters

All parameters are listed in initial\_global file.

|  |  |
| --- | --- |
| **INITIAL\_GLOBAL parameters init** | |
| Column Name | Description |
| **IB\_HIST\_END\_DT** | Last known date (i.e. sales and stock information is known) |

## Other Dependencies

All string comparison within design algorithm is case-insensitive. All steps of the algorithm below should be applied to all target variables **tgt\_type** which have **act\_flag** = 1.

# Algorithm Definition

Assumption: All KPIs inside input table must be checked within each alert.

## Common utility steps

1. To aggregate a table on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level:
   1. Table left join PRODUCT on product\_lvl\_id<m>

left join LOCATION on location\_lvl\_id<n>

left join CUSTOMER on Customer\_lvl\_id<k>

left join DISTR\_CHANNEL on distr\_channel\_lvl\_id<l>

and add columns:

product\_lvl\_id<al\_product\_lvl>,

location\_lvl\_id<al\_location\_lvl>,

customer\_lvl\_id<al\_customer\_lvl>,

distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>

e.g. if al\_product\_lvl = 8 then add column product\_id

and delete previous columns product\_lvl\_id<m>, location\_lvl\_id<n>, Customer\_lvl\_id<k>, distr\_channel\_lvl\_id<l>

* 1. Provide aggregation (aggregation method for other columns will be provided separately) table Group by

product\_lvl\_id<al\_product\_lvl>,

location\_lvl\_id<al\_location\_lvl>,

customer\_lvl\_id<al\_customer\_lvl>,

distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>

1. To aggregate a table on al\_time\_lvl:
   1. transform time variable using intnx-function (e.g intnx('week.2', period\_dt, 6));
   2. Provide aggregation (aggregation method for other columns will be provided seprarately) table Group by transformed period\_dt

## Alert 1: Abnormally lForecast ow value is missings for regular assortment or is zero

See detailed NAREG and ZEROREG in requirement document.

The total forecast for a regular SKU (i.e. one that is listed at least at one location (for Retailer, store) or a customer (for CPG) per week, for which the forecast is being built) is not built (i.e. Forecast value is NA or is zero).

**Inputs:** DISACC\_DISAGG\_HYBRID\_FORECAST, ALERT\_PARAMETERS, RESTORED\_DEMAND, FORECAST\_FLAG

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=1.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

* 1. Aggregate **I1.HYBRID\_FORECAST\_VALUE**

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):See s\_alert\_calculations04\_05\_alerts.sas macro56–109 (macro %alert1)97.

FORECAST\_VALUE = AVERAGE (**I1.HYBRID\_FORECAST\_VALUE)**

FROM I1

GROUP BY (see Common utility steps3.1 1) and (see Common utility steps3.1 2)

* 1. Aggregate FORECAST\_FLAG

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1):

/\*Calculate min start\_dt and max end\_dt on aggregated level\*/

MIN\_START\_DT MAX(**IB\_HIST\_END\_DT+1,**

MIN(FORECAST\_FLAG.PERIOD\_START\_DT)),

MAX\_END\_DT MIN(**IB\_HIST\_END\_DT+IB\_FC\_HORIZ,**

MAX(FORECAST\_FLAG.PERIOD\_END\_DT)),

FROM FORECAST\_FLAG

WHERE STATUS in **IB\_FF\_ACTIVE\_STATUS\_LIST**

GROUP BY (see Common utility steps3.1 1)

* 1. Define timestamps on aggregated time level in b., for which forecast can be forecasted.

If ALERT\_TIME\_GRANULRITY = week.2, then provide all weeks from MIN\_START\_DT to MAX\_END\_DT. In example on Figure 1, all weeks 10, 11, 12, 13, 14 must be added.

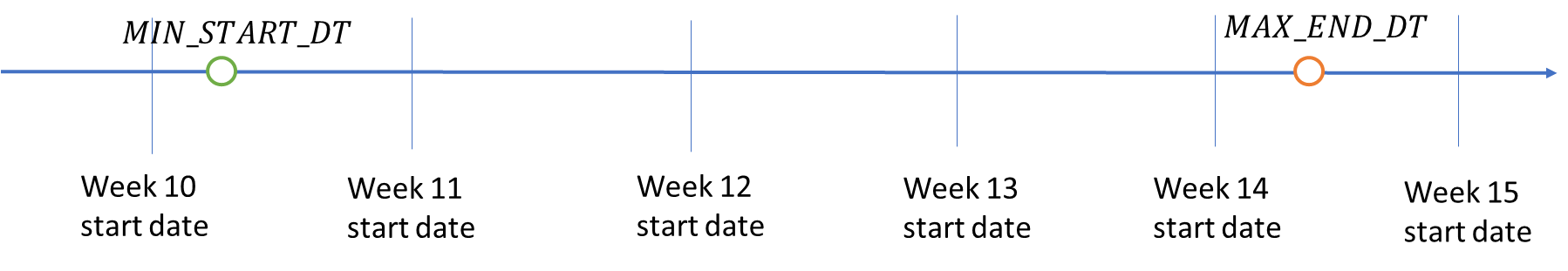


Figure 1. Defining related timestamps to interval

AS result b. is added with PERIOD\_DT column with values on period\_dt level

* 1. c. left join a. on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>. PERIOD\_DT

and leave only whose rows where a. FORECAST\_VALUE is missing or a.FORECAST\_VALUE<= **IB\_ALERT\_MIN\_VALUE**

* 1. Add other columns to output table:
     1. KPI\_NM = get value from table\_column variable after the name of target variable type **tgt\_type**:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST\_pos’

* + 1. ALERT\_TYPE = CASE WHEN FORECAST\_VALUE is missing THEN ‘NAREG’ ELSE ‘ZEROREG’ END
    2. STAT\_NOM\_NM = ‘Forecast value’
    3. STAT\_DEN\_NM = ‘na’
    4. STAT\_NOM\_VAL = d.FORECAST\_VALUE
    5. STAT\_DEN\_VAL = 1
    6. ALERT\_THRESHOLD = missing
    7. ALERT\_STAT\_VAL = d.FORECAST\_VALUE

Define list of Locations and a list of Products that must have forecast values at least in one period in the forecast period

Select only those pairs Location/Product for which

FORECAST\_FLAG.PERIOD\_END\_DT>= IB\_HIST\_END\_DT

Select Distinct Products from a

Select Distinct Locations from a.

Define Product Hierarchy Level elements for which alert should be calculated

1.b left join PRODUCT and save

at the week where forecast is built)

Alert is raised when the forecast is absent

sum of forecast over all horizons

is less than or equal to k=0.001)\*/

If forecast level in HYBRID FORECAST is in IB\_OUTPUT\_PRODUCT\_LVL, IB\_OUTPUT\_LOCATION\_LVL, (IB\_OUTPUT\_CUSTOMER\_LVL) then just keep the same table.

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPEPRODUCT\_ID** | Alert typeProduct ID (the lowest level of the product hierarchy) |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM\_VAL** | Statistics Nominator |
| **STAT\_DEN\_VAL** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |
| **STORE\_LOCATION\_ID** | Location ID |
| **PERIOD\_START\_DT** | Date of sales (calendar day) |
| **SEGMENT\_NAME** | Name of segment that was linked to a pair product/location within VF Project (can be missing) |
| **VF\_FORECAST\_VALUE** | VF forecast value |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new, or old |
| **ML\_FORECAST\_VALUE** | forecasted value |

## Alert 2: Abnormally high values for regular assortmentw.r.t last year data

The average demand forecast is more than **alert\_threshold** = 2 times higher than the actual restored demand for the average store for the same period last year.

**Inputs:** ALERT\_PARAMETERS, RESTORED\_DEMAND

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=2.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

* 1. Aggregate **I1.HYBRID\_FORECAST\_VALUE**

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

STAT\_NOM\_VAL = AVERAGE (**I1.HYBRID\_FORECAST\_VALUE)**

FROM I1

GROUP BY (see Common utility steps3.1 1) and (see Common utility steps3.1 2)

* 1. Define **target\_type** from target\_table\_column: e.g. if target\_table\_column = ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE then **target\_type** = ‘pos’
  2. Aggregate RESTORED\_DEMAND\_<**target\_type**>

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

/\*Calculate min start\_dt and max end\_dt on aggregated level\*/

STAT\_DEN\_VAL = AVERAGE (**SALESTGT\_QTY\_R)**

FROM RESTORED\_DEMAND\_**target\_type**

GROUP BY (see Common utility steps3.1 1) and (see Common utility steps3.1 2)

* 1. Shift historical value one year ahead:
     1. If al\_time\_lvl like ‘day’ then

period\_dt = intnx(al\_time\_lvl, period\_dt, 365);

* + 1. If al\_time\_lvl like ‘week’ then

period\_dt = intnx(al\_time\_lvl, period\_dt, 52)

* + 1. If al\_time\_lvl like ‘month’ then

period\_dt = intnx(al\_time\_lvl, period\_dt, 12)

* 1. a left join d. on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>. period\_dt

and leave only whose rows where d.STAT\_DEN\_VAL in not missing and d.STAT\_DEN\_VAL> **IB\_ALERT\_MIN\_VAL**.

* 1. Leave alerting rows:
     1. Calculate

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/STAT\_DEN\_VAL

* + 1. Leave only those rows for which ALERT\_STAT\_VAL>ALERT\_PARAMETERS.**ALERT\_THRESHOLD**
  1. Add other columns to output table:
     1. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

* + 1. ALERT\_TYPE = ‘INCRREG’
    2. STAT\_NOM\_NM = ‘Average Forecast Value’
    3. STAT\_DEN\_NM = ‘Average Demand Value a year ago’
    4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

s\_alert\_calculation111–94 (macro %alert2)

## Alert 3: Abnormally low values w.r.t last year data

The average demand forecast is more than **alert\_threshold** = 2 times lower than the actual restored demand for the average store for the same period last year.

**Inputs:** ALERT\_PARAMETERS, RESTORED\_DEMAND

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=3.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

Steps a. – e. are the same as for alert 2 see 3.3.

1. Leave alerting rows:
   * 1. Calculate

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/STAT\_DEN\_VAL

* + 1. Leave only those rows for which ALERT\_STAT\_VAL<1/ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

1. Add other columns to output table:
   * 1. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

* + 1. ALERT\_TYPE = ‘DECRREG’
    2. STAT\_NOM\_NM = ‘Average Forecast Value’
    3. STAT\_DEN\_NM = ‘Average Demand Value a year ago’
    4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

**Inputs:** T1

s\_alert\_calculation96–78 (macro %alert3)**Transformation algorithm:**

1. Keep the Same Table

Output: As a result of this step, a table of the following structure is constructed, *T2.*

|  |  |
| --- | --- |
| Column Name | Description |
| **PRODUCT\_ID** | Product ID (the lowest level of the product hierarchy) |
| **STORE\_LOCATION\_ID** | Location ID |
| **PERIOD\_START\_DT** | Date of sales (calendar day) |
| **SEGMENT\_NAME** | Name of segment that was linked to a pair product/location within VF Project (can be missing) |
| **VF\_FORECAST\_VALUE** | VF forecast value |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new, or old |
| **ML\_FORECAST\_VALUE** | forecasted value |

## Alert 4: Abnormally high values w.r.t last dataProviding LOCATION Life Cycle information

The average demand forecast exceeds the average weekly actual restored demand of the last l = 12 weeks by more than **alert\_threshold** = 5 times

**Inputs:** ALERT\_PARAMETERS, RESTORED\_DEMAND

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=4.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

Steps a. – b. are the same as for alert 2 see 3.3.

1. Aggregate RESTORED\_DEMAND\_<**target\_type**>

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

/\*Calculate min start\_dt and max end\_dt on aggregated level\*/

DEMAND = AVERAGE (**SALES\_QTY\_R)**

FROM RESTORED\_DEMAND\_**target\_type**

GROUP BY (see Common utility steps3.1 1) and (see Common utility steps3.1 2)

1. Calculate average DEMAND value for the last L = (84 days/12 weeks/3 months) of historical data:

ASSUMPTION: Provide external parameter regarding L (84 days/12 weeks/3 months)

STAT\_DEN\_VAL = AVERAGE (DEMAND)

FROM c.

WHERE period\_dt >

CASE WHEN al\_time\_lvl like ‘day’

THEN intnx(**al\_time\_lvl**, IB\_HIST\_END\_DT, 84)

ELSE CASE WHEN **al\_time\_lvl** like ‘week’

THEN intnx(**al\_time\_lvl**, IB\_HIST\_END\_DT, 12)

ELSE CASE WHEN **al\_time\_lvl** like ‘month’

THEN intnx(**al\_time\_lvl**, IB\_HIST\_END\_DT, 3)

/\*Assumption: al\_time\_lvl is one of 3 above types\*/

ELSE intnx(al\_time\_lvl, IB\_HIST\_END\_DT, 0)

END END END

GROUP BY product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>

1. a left join d. on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>.

and leave only whose rows where d.STAT\_DEN\_VAL in not missing and d.STAT\_DEN\_VAL> **IB\_ALERT\_MIN\_VAL**.

1. Leave alerting rows:
   * 1. Calculate

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/STAT\_DEN\_VAL

* + 1. Leave only those rows for which ALERT\_STAT\_VAL>ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

1. Add other columns to output table:
   * 1. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

* + 1. ALERT\_TYPE = ‘HIGHREG’
    2. STAT\_NOM\_NM = ‘Average Forecast Value’
    3. STAT\_DEN\_NM = ‘Average Demand Value within last 3 months’
    4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

s\_alert\_calculation79–60 (macro %alert4)**Inputs:** T2

**Transformation algorithm:**

1. Keep the Same Table

Output: As a result of this step, a table of the following structure is constructed, *T3.*

|  |  |
| --- | --- |
| Column Name | Description |
| **PRODUCT\_ID** | Product ID (the lowest level of the product hierarchy) |
| **STORE\_LOCATION\_ID** | Location ID |
| **PERIOD\_START\_DT** | Date of sales (calendar day) |
| **SEGMENT\_NAME** | Name of segment that was linked to a pair product/location within VF Project (can be missing) |
| **VF\_FORECAST\_VALUE** | VF forecast value |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new, or old |
| **ML\_FORECAST\_VALUE** | forecasted value |

## Alert 5: Abnormally low values w.r.t last dataProviding CUSTOMER Life Cycle information

The average demand forecast is lower than the average weekly actual recovered demand of the last 12 weeks by more than **alert\_threshold** = 5 times

**Inputs:** ALERT\_PARAMETERS, RESTORED\_DEMAND

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=5.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

Steps a. – e. are the same as for alert 4 see 3.5.

1. Leave alerting rows:
   * 1. Calculate

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/STAT\_DEN\_VAL

* + 1. Leave only those rows for which ALERT\_STAT\_VAL<1/ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

1. Add other columns to output table:
   * 1. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

* + 1. ALERT\_TYPE = ‘LOWREG’
    2. STAT\_NOM\_NM = ‘Average Forecast Value’
    3. STAT\_DEN\_NM = ‘Average Demand Value within last 3 months’
    4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

## s\_alert\_calculation6242 (macro %alert5)Alert 6: Abnormally maximum deviation w.r.t last data

The maximum deviation of the forecast from the last full completed week, in =2 times greater than the difference between the maximum and minimum forecast on the history of the last 3 month

**Inputs:** ALERT\_PARAMETERS, RESTORED\_DEMAND

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=6.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

1. Aggregate **I1.HYBRID\_FORECAST\_VALUE**

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

\_VALUE = AVERAGE (**I1.HYBRID\_FORECAST\_VALUE)**

FROM I1

GROUP BY product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>, period\_dt

1. Define **target\_type** from target\_table\_column: e.g. if target\_table\_column = ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE then **target\_type** = ‘pos’
2. Aggregate RESTORED\_DEMAND\_<**target\_type**>

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

/\*Calculate min start\_dt and max end\_dt on aggregated level\*/

\_VALUE= AVERAGE (**SALESTGT\_QTY\_R)**

FROM RESTORED\_DEMAND\_**target\_type**

GROUP BY GROUP BY product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>, period\_dt

1. Union results of a and c. there should be 6 columns after this step in output table

product\_lvl\_id<**al\_product\_lvl**m>,

location\_lvl\_id<**al\_location\_lvl**n>,

customer\_lvl\_id<**al\_customer\_lvl**k>,

distr\_channel\_lvl\_id<**al\_distr\_channel\_lvl**l>,

PERIOD\_DT

\_VALUE

1. Calculate first difference for each quad product\_lvl\_id<m> | location\_lvl\_id<n> | customer\_lvl\_id<k> | distr\_channel\_lvl\_id :

\_DIFVALUE (period) = VALUE(period) - VALUE(previous period)

* 1. it can be done in two steps:
     1. shift elements in period\_dt column 1 timestamp ahead (where measure of timestamp is **al\_time\_lvl)** see d. in alert 2.
     2. original table left join i. on product\_lvl\_id<m> | location\_lvl\_id<n> | customer\_lvl\_id<k> | distr\_channel\_lvl\_id. period\_dt
     3. calculate \_DIFVAL as

1. Calculate benchmark statistics:

STAT\_DEN\_VAL = AVERAGE (\_DIFVALUE)

FROM e.

WHERE period\_dt <= IB\_HIST\_END\_DTand

period\_dt >

CASE WHEN al\_time\_lvl like ‘day’

THEN intnx(al\_time\_lvl, IB\_HIST\_END\_DT, 84)

ELSE CASE WHEN al\_time\_lvl like ‘week’

THEN intnx(al\_time\_lvl, IB\_HIST\_END\_DT, 12)

ELSE CASE WHEN al\_time\_lvl like ‘month’

THEN intnx(al\_time\_lvl, IB\_HIST\_END\_DT, 3)

/\*Assumption: al\_time\_lvl is one of 3 above types\*/

ELSE intnx(al\_time\_lvl, IB\_HIST\_END\_DT, 0)

END END END

GROUP BY product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>

1. e left join f. on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>.

and leave only whose rows

where d.STAT\_DEN\_VAL in not missing

and d.STAT\_DEN\_VAL> **IB\_ALERT\_MIN\_VAL**

AND e.PERIOD > IB\_HIST\_END\_DT

Rename e.\_VALUE to STAT\_NOM\_VAL

1. Leave alerting rows:
   * 1. Calculate

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/STAT\_DEN\_VAL

* + 1. Leave only those rows for which ALERT\_STAT\_VAL>ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

1. Add other columns to output table:
   * 1. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

* + 1. ALERT\_TYPE = ‘DEVWK’
    2. STAT\_NOM\_NM = ‘Forecast Deviation’
    3. STAT\_DEN\_NM = ‘Demand Deviation within last 3 months’
    4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

s\_alert\_calculation4449 (macro %alert6)

**Inputs:** T3

**Transformation algorithm:**

1. Keep the Same Table

Output: As a result of this step, a table of the following structure is constructed, HYBRID\_FORECAST\_DISAGG (detailed description see in section 3.8)*.*

## Alert 7 Non-zero forecast for assortments with missing forecast flags

The alert indicates the existence of the forecasts for assortments that should not be forecasted for the specified period.

**Inputs:** ALERT\_PARAMETERS, FORECAST\_FLAG

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=7.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

Steps a.-c. are the same as for alert 1, see 3.2.

* 1. a. left join c. on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>. PERIOD\_DT

and leave only whose rows where c.PERIOD\_DT is missing and a.FORECAST\_VALUE > **IB\_ALERT\_MIN\_VALUE**

* 1. Add other columns to output table:
     1. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

* + 1. ALERT\_TYPE = ‘ZEROFLG’
    2. STAT\_NOM\_NM = ‘Forecast value’
    3. STAT\_DEN\_NM = ‘na’
    4. STAT\_NOM\_VAL = d.FORECAST\_VALUE
    5. STAT\_DEN\_VAL = 1
    6. ALERT\_THRESHOLD = missing
    7. ALERT\_STAT\_VAL = d.FORECAST\_VALUE

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM\_VAL** | Statistics Nominator |
| **STAT\_DEN\_VAL** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

## Alert 8 Abnormal share of new assortment forecast

Idea: [New product demand forecast] / [Average demand forecast for the same period for the range group from the corresponding product + 1 level of the hierarchy > ALERT\_THERSHOLD = 20 or < 1/ALERT\_THRESHOLD.

**Inputs:** ALERT\_PARAMETERS, FORECAST\_FLAG, RESTORED\_DEMAND\_<**target\_type**>TGT\_VAR\_CONFIG

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=8.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

* 1. Aggregate **I1.HYBRID\_FORECAST\_VALUE**

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

/\*Assumption: Average transformation is neede to

STAT\_NOM\_VAL = AVERAGE (**I1.HYBRID\_FORECAST\_VALUE)**

PERIOD\_DT

FROM I1

GROUP BY product\_lvl\_id<al\_product\_lvl>, product\_lvl\_id<**tgt\_var\_config.al\_product\_agg\_lvl>** /\*is needed below\*/**,** location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>, PERIOD\_DT(see Common utility steps3.1 1) and (see Common utility steps3.1 2)

* 1. Aggregate FORECAST\_FLAG to further splitting quadruples product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl> to new and regular assortmentruple

/\*Calculate min start\_dt and max end\_dt on aggregated level\*/

MIN\_START\_DT MAX(**IB\_HIST\_END\_DT+1,**

MIN(FORECAST\_FLAG.PERIOD\_START\_DT)),

FROM FORECAST\_FLAG

WHERE STATUS in **IB\_FF\_ACTIVE\_STATUS\_LIST**

GROUP BY ( product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>see Common utility steps3.1 1)

* 1. Calculate average demand for regular assortment Define target\_type from target\_table\_column: e.g. if target\_table\_column = ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE then target\_type = ‘pos’
  2. Aggregate RESTORED\_DEMANDon **tgt\_var\_config\_<target\_type>**
  3. **on. al\_product\_agg\_lvl** al\_product\_lvl/al\_location\_lvl /al\_customer\_lvl /al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

/\*Calculate min start\_dt and max end\_dtaverage demand on on aggregated level for regular assortment\*/

STAT\_DEN\_VAL = AVERAGE (STAT\_NOM\_VALTGTSALES\_QTY\_R)

STAT\_COUNT = COUNT(product\_lvl\_id<**al\_product\_lvl**>)

FROM RESTORED\_DEMAND\_target\_typea.

JOIN b on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>

/\*select only regular assortment\*/

WHERE **IB\_END\_HIST\_DT** - **MIN\_START\_DT** > **IB\_MAX\_NP\_HISTORY**

GROUP BY product\_lvl\_id<**tgt\_var\_config.al\_product\_agg\_lvl**>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>, PERIOD\_DT

(see Common utility steps3.1 1) and (see Common utility steps3.1 2)

* 1. Select only new assortment statistics:Shift historical value one year ahead:

If al\_time\_lvl like ‘day’ then

period\_dt = intnx(**al\_time\_lvl**, period\_dt, 365);

If al\_time\_lvl like ‘week’ then

period\_dt = intnx(**al\_time\_lvl**, period\_dt, 52)

If al\_time\_lvl like ‘month’ then

period\_dt = intnx(**al\_time\_lvl**, period\_dt, 12)

a left join cd. on product\_lvl\_id<tgt\_var\_config.**tgt\_var\_config.al\_product\_agg\_lvl** al\_product\_agg\_lvl>product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>, PERIOD\_DT. period\_dt

left join b on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>

SELECT a.STAT\_NOM\_VAL,

c.STAT\_DEN\_VAL

/\*select only regular assortment\*/

WHERE **IB\_END\_HIST\_DT** – b.**MIN\_START\_DT** <= **IB\_MAX\_NP\_HISTORY**

/\*leave only those rows with enough statistics\*/

cb.STAT\_COUNT>= **IB\_ALERT\_MIN\_OBS**

and leave only whose rows where d.STAT\_DEN\_VAL in not missing and d.STAT\_DEN\_VAL> **IB\_ALERT\_MIN\_VAL**.

* 1. Leave alerting rows:
     1. Calculate

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/STAT\_DEN\_VAL

* + 1. Leave only those rows for which ALERT\_STAT\_VAL>ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

**OR**

ALERT\_STAT\_VAL<1/ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

* 1. Add other columns to output table:
     1. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

* + 1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

* + 1. ALERT\_TYPE = ‘SHRNEW’
    2. STAT\_NOM\_NM = ‘Forecast for New Assortment
    3. STAT\_DEN\_NM = ‘Average Forecast for Regular Assortment’
    4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

## Alert 9 Abnormally low value for new assortment

Forecast value is lower then 1 peace a year

Abnormally low value for new assortment LOWNEW Forecast of demand for a new product less than 1 piece per year (less than 0.0027 piece on average per day) MED

**Inputs:** ALERT\_PARAMETERS, FORECAST\_FLAG, TGT\_VAR\_CONFIG

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=98.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

1. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table\_column do the next steps.

For example table **I1.HYBRID\_FORECAST\_VALUE** is used as element ALERT\_PARAMETERS.target\_table\_column

* 1. Aggregate **I1.HYBRID\_FORECAST\_VALUE**

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1) and on al\_time\_lvl (see Common utility steps3.1 2):

STAT\_NOM\_VAL = SUM(**I1.HYBRID\_FORECAST\_VALUE)**

STAT\_COUNT = COUNT(**I1.HYBRID\_FORECAST\_VALUE)**

PERIOD\_DT = missing

FROM I1

GROUP BY product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>,

* 1. Aggregate FORECAST\_FLAG and select NEW quadruples

on al\_product\_lvl/al\_location\_lvl/al\_customer\_lvl/al\_distr\_channel\_lvl level (see Common utility steps3.1 1):

/\*Calculate min start\_dt and max end\_dt on aggregated level\*/

MIN\_START\_DT MIN(FORECAST\_FLAG.PERIOD\_START\_DT),

FROM FORECAST\_FLAG

WHERE STATUS in **IB\_FF\_ACTIVE\_STATUS\_LIST**

1. Select only new assortment statistics:

a left join b on product\_lvl\_id<al\_product\_lvl>, location\_lvl\_id<al\_location\_lvl>, customer\_lvl\_id<al\_customer\_lvl>, distr\_channel\_lvl\_id<al\_distr\_channel\_lvl>

SELECT STAT\_NOM\_VAL

STAT\_DEN\_VAL = 1

PERIOD\_DT = missing

/\*select only regular assortment\*/

WHERE **IB\_END\_HIST\_DT** – b.**MIN\_START\_DT** <= **IB\_MAX\_NP\_HISTORY**

AND STAT\_COUNT>= IB\_ALERT\_MIN\_OBS

1. Leave alerting rows, calculate:

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/ STAT\_DEN\_VAL

Leave only those rows

WHERE ALERT\_STAT\_VAL<ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

1. Add other columns to output table:
2. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

1. ALERT\_TYPE = ‘LOWNEW’
2. STAT\_NOM\_NM = ‘Total forecast value on the whole forecasting period’
3. STAT\_DEN\_NM = ‘na’
4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input collumn name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

## Alert 10 NONSEAS time series model is used for long time series

|  |  |  |
| --- | --- | --- |
| **Non-seasonal models applied** | **NONSEAS** | **A non-seasonal *ESM* or *ARIMA* model without seasonal variables is used to predict a time series with a history of more than 2 years.** |

**IDEA:** A non-seasonal *ESM* or *ARIMA* model without seasonal variables is used to predict a time series with a history of more than 2 years.

Alert is applicable only for VF Forecast table

**Inputs:** ALERT\_PARAMETERS, FORECAST\_FLAG, TGT\_VAR\_CONFIG

**Transformation algorithm:**

1. Use only those rows from ALERT\_PARAMETERS that are related to ALERT\_ID=810.
2. For each row of ALERT\_PARAMETERS, for each element of ALERT\_PARAMETERS.target\_table

Where target\_table like ‘%VF%’

do the next steps.

* 1. **I1 left join VF\_SEGMENTS**

ASSUMPTION: there is only one VF\_SEGMENTS table

on product\_lvl\_id<m>/location\_lvl\_id<n>/customer\_lvl\_<k>/distr\_channel\_lvl\_id<k> level (see Common utility steps3.1 1):

and SELECT VF\_SEGMENTS.**SEGMENT\_NAME**

* 1. Aggregate FORECAST\_FLAG and select NEW quadruples

on vfal\_product\_lvl/vfal\_location\_lvl/vfal\_customer\_lvl/vfal\_distr\_channel\_lvl level (see Common utility steps3.1 1):

/\*Calculate min start\_dt and max end\_dt on aggregated level\*/

MIN\_START\_DT MIN(FORECAST\_FLAG.PERIOD\_START\_DT),

FROM FORECAST\_FLAG

WHERE STATUS in **IB\_FF\_ACTIVE\_STATUS\_LIST**

* 1. Select only assortment with more than 2 years history:

a left join b on product\_lvl\_id<vf\_al\_product\_lvl>, location\_lvl\_id<alvf\_location\_lvl>, customer\_lvl\_id<alvf\_customer\_lvl>, distr\_channel\_lvl\_id<alvf\_distr\_channel\_lvl>

SELECT STAT\_NOM\_VAL = missing

STAT\_DEN\_VAL = 1

PERIOD\_DT = missing

/\*select only regular assortment\*/

WHERE **IB\_END\_HIST\_DT** – b.**MIN\_START\_DT** >= **2\*365**

ASSUMPTION: alert must be parametrized

1. Leave alerting rows, calculate:

ALERT\_STAT\_VAL = STAT\_NOM\_VAL/ STAT\_DEN\_VAL

Leave only those rows

WHERE VF\_SEGMENTS.**SEGMENT\_NAME** is not LIKE ‘%\_SEASON%’ and is not like ‘SEASON%’

1. Add other columns to output table:
2. KPI\_NM = get value from table\_column variable after the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

kpi\_nm = ‘pos.HYBRID\_FORECAST\_VALUE’

1. INPUT\_TABLE = get value from table\_column variable before the name of target variable:

i.e. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST\_VALUE

INPUT\_TABLE = ‘ACC\_AGG\_HYBRID\_FORECAST’

1. ALERT\_TYPE = ‘NONSEAS’
2. STAT\_NOM\_NM = ‘Time Series Segment’
3. STAT\_DEN\_NM = ‘na’
4. ALERT\_THRESHOLD = ALERT\_PARAMETERS.**ALERT\_THRESHOLD**

Output: As a result of this step, a table of the following structure is constructed, *T1.*

|  |  |
| --- | --- |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id<m>,** | Level m = al\_product\_lvlof product hierarchy |
| **location\_lvl\_id<n>,** | Level n = al\_location\_lvl of location hierarchy |
| **customer\_lvl\_id<k>,** | Level k = al\_customer\_lvlof customer hierarchy |
| **distr\_channel\_lvl\_id<l>** | Level l = al\_distr\_channel\_lvlof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input column name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |

## Combine all alerts

**Inputs:** T1, T2, T3, T4, T5, T6, T7, T8, T9, T10

**Transformation algorithm:**

1. Union all tables T1, T2, T3, T4, T5, T6, T7, T8, T9, T10 into one tables\_alert\_calculation51–605 (macro %s\_alert\_calculation)586

Output: As a result of this step, the OutAlerts table is constructed, see details in 3.8 3.13.

## Output from the Algorithm

1. Table with all types of forecast alerts

|  |  |
| --- | --- |
| **OutAlertsHYBRID\_FORECAST** | |
| Column Name | Description |
| **PRODUCT\_ID** | Product ID (the lowest level of the product hierarchy) |
| **STORE\_LOCATION\_ID** | Location ID |
| **PERIOD\_START\_DT** | Date of sales (calendar day) |
| **SEGMENT\_NAME** |  |
| **VF\_FORECAST\_VALUE** | VF forecast value |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new, or old |
| **ML\_FORECAST\_VALUE** | ML forecasted value |
| **HYBRID\_FORECAST\_VALUE** | HYBRID forecast value |
| Column Name | Description |
| **ALERT\_TYPE** | Alert type |
| **product\_lvl\_id,** | Element id of product hierarchy |
| **product\_lvl** | Level of product hierarchy |
| **location\_lvl\_id,** | Element id of product hierarchy |
| **location\_lvl** | Level of location hierarchy |
| **customer\_lvl\_id,** | Element id of customer hierarchy |
| **customer\_lvl** | Level of customer hierarchy |
| **distr\_channel\_lvl\_id** | Element id of distr\_channel hierarchy |
| **distr\_channel\_lvl** | Levelof distribution channel hierarchy |
| **PERIOD\_DT** | Start period of date in al\_time\_lvl format |
| **KPI\_NM** | Target variable name |
| **INPUT\_TABLE** | Input table and input column name, e.g. ACC\_AGG\_HYBRID\_FORECAST\_pos.HYBRID\_FORECAST, |
| **STAT\_NOM\_NM** | Statistics Nominator Name |
| **STAT\_DEN\_NM** | Statistics Denominator Name |
| **STAT\_NOM** | Statistics Nominator |
| **STAT\_DEN** | Statistics Denominator |
| **ALERT\_THRESHOLD** | Alert threshold value |
| **ALERT\_STAT\_VAL** | Alert statistics value, i.e. STAT\_NOM/STAT\_DEN |