Feature at a glance

Enhanced filtering of general ledger codes

Saeed Rezavi, Solution Management, SAP Ariba
Roger Nyberg, Product Management, SAP Ariba
Target GA February 2018

CONFIDENTIAL

Feature at a Glance

Introducing: **Enhanced filtering of general ledger codes**

Customer challenge

Customers have a need to add filters to the out-of-the-box filtering of GL codes which is done by company code. However, when a customer adds a generic relation entry and adds a filter for a different field, the out-of-the-box filtering is no longer taken into consideration. For example a customer adds a generic relation entry for commodity code to general ledger, but then the filtering on company code is lost.

Meet that challenge with SAP **Ariba**

A new template has been introduced that can be used instead. In this new template, the existing out-of-the-box object matching query constraint is merged with the query constraints from Generic Relation entry.

Experience key benefits

Customers can add a filter on top of the out-of-the-box GL filtering by company code without removing the filter on company code.

Solution area

Operational Procurement / SAP Ariba Buying and Invoicing + SAP Ariba Invoice Management

Enablement information

This is an optional feature.

Value measure	●	high
Ease of implementation	◒	simple
Regional limitations	●	global

Feature at a Glance

Introducing: **Enhanced filtering of general ledger codes**

A customer wishing to uptake this new functionality has to make two changes to their customization:

- **Change the template name from GenericRelationField to ConditionBasedGenericRelationField for the GenericLedger field from:**

```
<inClass name="ariba.common.core.SplitAccounting">
    <inField name="GeneralLedger">
            <templates>
                <template name=" GenericRelationField" operator="add"/>
          </templates>
           <properties nameTableConstraintFieldPath="LineItem.CommonCommodityCode" nameTableRelationName="CommodityCodeGLMap"/>
        </inField>
</inClass>
</extension>
```

to:

```
<inClass name="ariba.common.core.SplitAccounting">
    <inField name="GeneralLedger">
            <templates>
                <template name=" ConditionBasedGenericRelationField" operator="add"/>
          </templates>
           <properties nameTableConstraintFieldPath="LineItem.CommonCommodityCode" nameTableRelationName="CommodityCodeGLMap"/>
        </inField>
</inClass>
</extension>
```

Best practice is to use ConditionBasedGenericRelationField instead of GenericRelationField going forward