CUSTOMER

Promotion Calculation Engine 3
Document Version: 3.2

Functional Guide – Promotion Calculation Engine

Contents

1 Document History

Version	Change	Date
3.2	Corrected topic Prorate the Benefit	07 Nov 2018
	Modified topic Apply a Simple Discount	
	Added terms to Definitions	
	Modified topic Configuration	
3.2	Modified topic Configuration	03 Jan 2019
	Added topic Apply Customer-Specific Prices.	
	Modified topics Use Thresholds, Intervals, and Limits in section THERE IS MORE.	
	Modified topics Manually Apply a Benefit in section THERE IS MORE.	
	Added new mapping for External Offer Identifier in topic Configuration.	
3.2	The following topics are added due to a fix: • 2019-10-29_13-35-47_Activate Price Derivation Rule Eligibilities with Respect to Sales and Returns • Handle Collisions with Respect to Sales and Returns • Choose Items with Respect to Sales and Returns • Compute Total Price Modification Methods with Respect to Sales and Returns • Mix and Match with Respect to Sales and Returns The following topics are modified due to a fix: • Handle Sales and Returns without the Original Transaction in section THERE IS MORE • Grant Loyalty Points in section THERE IS MORE • Select the Method to Discount • Divide and Prorate the Discount	25 Jan 2019
3.2	Modified description of rebateShareMethod SHARE to include missing details. The following pages were changed: • Prorate the Benefit • Configuration	07 Feb 2019
3.2	Modified description of the behavior when considerPreviousPromotionConditionFlag == false is set, including example • Set the Calculation Base Amount	15 May 2019

Version	Change	Date
3.2	Replaced the limitation of allowing simple discount RB only with a constraint to forbid mix&match • Use a Simple Product Group as a Trigger	14 Jun 2019
3.2	• 'Use an Item as Trigger': examples reworked. • 'Use a Merchandise Category as a Trigger' examples reworked • 'Combine Triggers' reworked	09 Jul 2019
3.2	Modified the description of calculation mode • Select a Calculation Mode	09 Jul 2019
3.2	Modified description for QUTI • Use Thresholds, Intervals, and Limits	09 Jul 2019
3.2	Modified the description for time restrictions • Recurrent Offer	22 Aug 2019
3.2	Modified the whole chapter, especially Brute Force and Greedy • Select a Promotion Price Derivation Rule in Case of a Collision	17 Sep 2019
3.2	Modified description of the interval handling • Use Thresholds, Intervals, and Limits	27 Sep 2019
3.2	Modified description for Mix and Match • Mix and Match	22 Oct 2019
3.2	The (in)validation of price derivation rule eligibilities with threshold type code QUTI or AMTI according to their interval quantity or amount value is added. The following page is changed: • Check the Validity of a Promotion	25 Oct 2019
3.2	UnitOfMeasureCode and QuantityInputMethod entries in the configuration section's tables were corrected. The following page is changed: • Configuration	29 Oct 2019
3.2	Merched the chapters Activate Price Derivation Rule Eligibilities with Respect to Sales and Returns to the chapter Handle Sales and Returns without the Original Transaction • Handle Sales and Returns without the Original Transaction	29 Oct 2019
3.2	Modified the subchapters of Round a Benefit by added decision trees: • Round the Benefit	07 Nov 2019
3.2	Modified the chapter Handle Sales and Returns without the Original Transaction by added decision trees: • Handle Sales and Returns without the Original Transaction	07 Nov 2019

Version	Change	Date
3.2	Added conditionLimit description below calculationTimeLimit description • Select a Promotion Price Derivation Rule in Case of a Collision	17 Feb 2020
3.2	Added the subchapter 'Interval-Based Transaction-Related Discounts' • Use Thresholds, Intervals, and Limits	18 Feb 2020
3.2	Added the subchapter 'Determine the Items Which Are Considered for the Calculation Base Amount' • Set the Calculation Base Amount	18 Feb 2020
3.2	Modified the subchapter 'Combine Rule Matching Items with a Logical OR' • Mix and Match	08 Apr 2020
3.2	Modified the chapter Round the Benefit with a disclaimer about the rounding rule. Modified the description of the multiple value. • Round the Benefit	08 Apr 2020
3.2	Modified the description of calculation time limit • Select a Promotion Price Derivation Rule in Case of a Collision	08 Apr 2020

2 Introduction

This guide provides an overview of the functionality of the promotion calculation engine (PCE). The PCE can be used to calculate effective sales prices, discounts, and loyalty points for a set of products or a shopping basket.

With omnichannel promotion pricing (OPP), SAP provides a promotion pricing service (PPS) for the calculation of effective sales prices which can be used in different sales channels. The PCE is the calculation core of this service.

This documentation describes the entire PCE functionality and contains examples for PCE use cases and their prerequisites.

Note that some use cases are currently not supported with OPP. For more information about offer types supported with OPP, see the application help on SAP Help Portal at https://help.sap.com/viewer/p/CARAB under *Application Help > SAP Customer Activity Repository > <Version> > Omnichannel Promotion Pricing > Offer Types with Omnichannel Promotion Pricing*.

2.1 References

For OPP and the PCE, more information is available in the SAP Help Portal under the section Development:

- Client API for Omnichannel Promotion Pricing: defines the Client interface of the PPS and accordingly of the PCE, too.
- Development and Extension Guide for Omnichannel Promotion Pricing: explains the necessary steps for extending the PPS.
- SDK Promotion Calculation Engine: defines the Software Development Kit (SDK) of the PCE.

2.2 Definitions

The table below compares the terminology of the PCE and the OPP/PPS. The terminology is the same if the respective field in the column OPP/PPS Terminology is blank. The PCE terminology is used within this documentation.

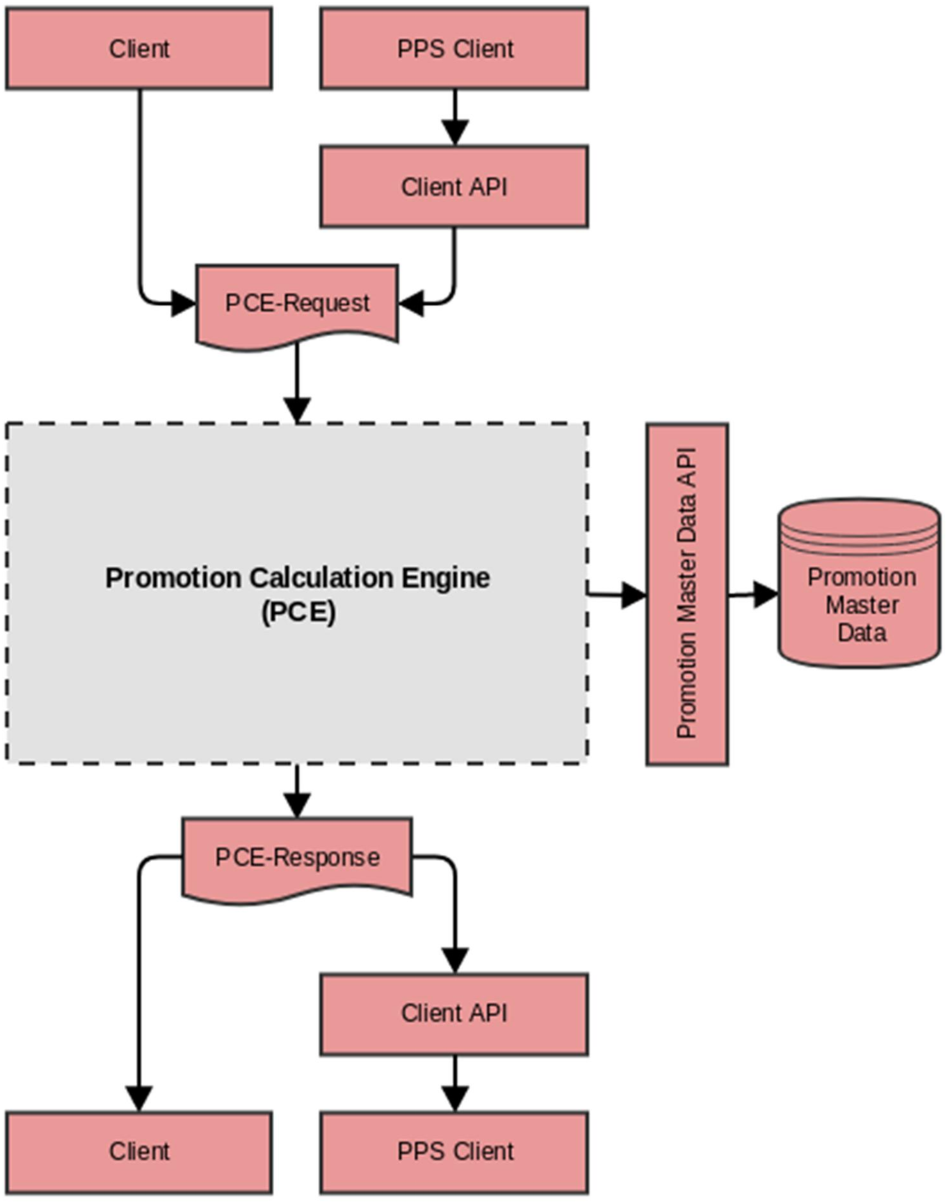
PCE Terminology	OPP/PPS Terminology	Description
Discount		A discount is a reduction of the sales price. This can be a direct price reduction or a voucher.
Discount Sales Price	**Discount Price**	Price of a line item before the computation of the PCE is finished and after a preceding promotion price derivation rule was applied to the line item.
Discount Sales Unit Price		Price of an item before the computation of the PCE is finished and after a preceding promotion price derivation rule was applied to the item.
Discount Total Amount		The total of discount sales prices in a transaction after a preceding promotion price derivation rule was applied.
Effective Sales Price		Price of a line item after the application of all applicable promotion price derivation rules.
Effective Sales Unit Price	**Effective Sales Price**	Price of an item after the application of all applicable promotion price derivation rules.
Effective Total Amount		The total of effective sales prices in a transaction after all promotion price derivation rules were applied.
Frequent Shopper Points		Like loyalty points. A bonus that a customer can earn when being subscribed to a loyalty program. The loyalty points are added to the loyalty program account balance for sales and are subtracted from it for returns.
Item	**Article**	Representation of a product, an article, a good, or a service that is bought or returned by a customer. An item is the smallest unit or customer pack that can be ordered independently and that cannot be split further into any smaller units.
Line Item	**Item**	An element of a transaction that contains information on goods or services delivered.
Loyalty Points		A bonus that a customer can earn when being subscribed to a loyalty program. The loyalty points are added to the loyalty program account balance for sales and are subtracted from it for returns.
Loyalty Program		A membership-based customer incentive initiative.
PCE-Request		Representation of a purchase before the PCE has processed anything.
PCE-Response		Representation of a purchase after the PCE has processed it.
Price Derivation Rule		A rule that provides the benefit that a customer is granted if the eligibility is met.
Price Derivation Rule Eligibility		Describes which preconditions the promotion price derivation rule has to fulfill in order to be applied.
Promotion	**OPP Promotion**	A collection of promotion price derivation rules. A promotion represents an offer in a format that complies with the format of the Association for Retail Technology Standards (ARTS).

PCE Terminology	OPP/PPS Terminology	Description
Promotion Master Data	**Price and Promotion Repository**	Repository that contains the regular sales prices and the promotion price derivation rules.
Promotion Master Data API	**Data Access API**	The interface that provides the promotions to the PCE.
Promotion Price Derivation Rule		A pair of one price derivation rule eligibility and one price derivation rule.
Regular Sales Price		The regular price for the line item before any discounts have been applied.
Regular Sales Unit Price	**Regular Price**	The regular or lookup per-unit price for the item before any discounts have been applied.
Regular Total Amount		The total of discount sales prices in a transaction before transaction-related promotion price derivation rules were applied.
Sales Price		Generic term for the price of a line item (regular, discount, effective sales price).
Sales Unit Price		Generic term for the price of an item (regular, discount, effective sales unit price).
Shopping Basket	**Shopping Cart**	A physical or virtual container for the items a customer is going to buy. The customer can add items to or remove items from it at any stage before he or she proceeds to check out and pays for the items.
Transaction		PCE internal representation of a shopping basket and additional related information about the purchase.

3 Business Process

This chapter provides you with an overview on how the PCE receives, processes, and hands back information.

The PCE uses a request/response mechanism for communication with the requesting Clients:



The PCE receives a request (hereafter referred to as PCE-request) from the Client. This request can be issued by a PPS Client via the Client API or directly by the client application (in the previous figure denoted by "Client"). The PCE-request contains the shopping basket with all required information about the purchase.

The PCE processes the request and applies monetary discounts and loyalty points to the given shopping basket. The calculation is based on the content of the shopping basket (items, regular sales unit prices, scanned coupons, assigned customer groups, and so on), the system configuration parameters, and the promotion master data.

How the promotion master data as well as the promotion master data API look like, depends on the different scenarios – i.e. the Client application scenario. Different scenarios are described in the document SDK Promotion Calculation Engine in detail.

Finally, the PCE issues a response (hereafter referred to as PCE-response). The PCE-response contains the purchase information with the added benefit and the effective sales prices.

The following chapters discuss the process in greater detail.

You find information on the following topics:

- Data Flow
- Processed Steps

3.1 Data Flow

The PCE needs information about the particular purchase and the promotion to calculate the benefit and apply it to the correct items. Within this section, an overview is given for accessing the promotion master data and the information regarding the purchase. Thus, it is answered where the necessary information comes from and how it is represented. Additionally, documents are noted for further reading on specific topics.

3.1.1 Promotion Master Data Access

For the calculation of effective sales prices the PCE needs the information about regular sales unit prices and promotion master data. The necessary information about the promotions needed by the PCE is saved in the promotion master data.

The locally deployed sales channels are regularly updated via the outbound layer. Thereby the data is updated with the help of the Idoc-format. The PCE accesses the promotion master data directly from the locally stored promotion master data.

The centrally deployed sales channels do not have to regularly update the promotion master data since these can directly access the SAP Central Price and Promotion Repository.

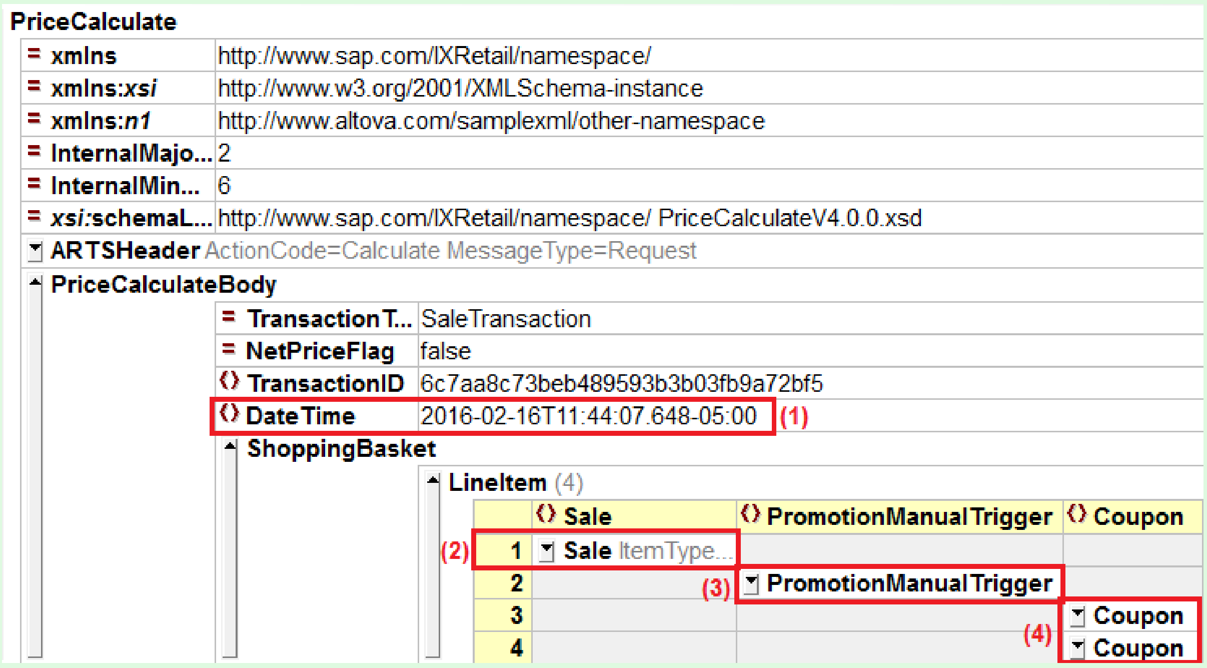
The SDK Promotion Calculation Engine documentation provides further information about accessing the promotion master data in the different scenarios.

3.1.2 PCE-Request

The PCE-request is received by the PCE and contains all necessary information about the purchase that is needed. It includes - among others - the date and time of the purchase, information on which items are in the shopping basket, and whether the customer has a coupon.

Exemplary PCE-Request

The following figure illustrates an exemplary PCE-request.



```
PriceCalculate
  = xmlns              http://www.sap.com/IXRetail/namespace/
  = xmlns:xsi          http://www.w3.org/2001/XMLSchema-instance
  = xmlns:n1           http://www.altova.com/samplexml/other-namespace
  = InternalMajo...    2
  = InternalMin...     6
  = xsi:schemaL...     http://www.sap.com/IXRetail/namespace/ PriceCalculateV4.0.0.xsd
  ARTSHeader ActionCode=Calculate MessageType=Request
  PriceCalculateBody
      = TransactionT...    SaleTransaction
      = NetPriceFlag       false
      <> TransactionID     6c7aa8c73beb489593b3b03fb9a72bf5
      <> DateTime          2016-02-16T11:44:07.648-05:00    (1)
      ShoppingBasket
          LineItem (4)
                  <> Sale              <> PromotionManualTrigger    <> Coupon
          (2) 1   Sale ItemType...
              2               (3)      PromotionManualTrigger
              3                                                     Coupon
              4                                               (4)   Coupon
```

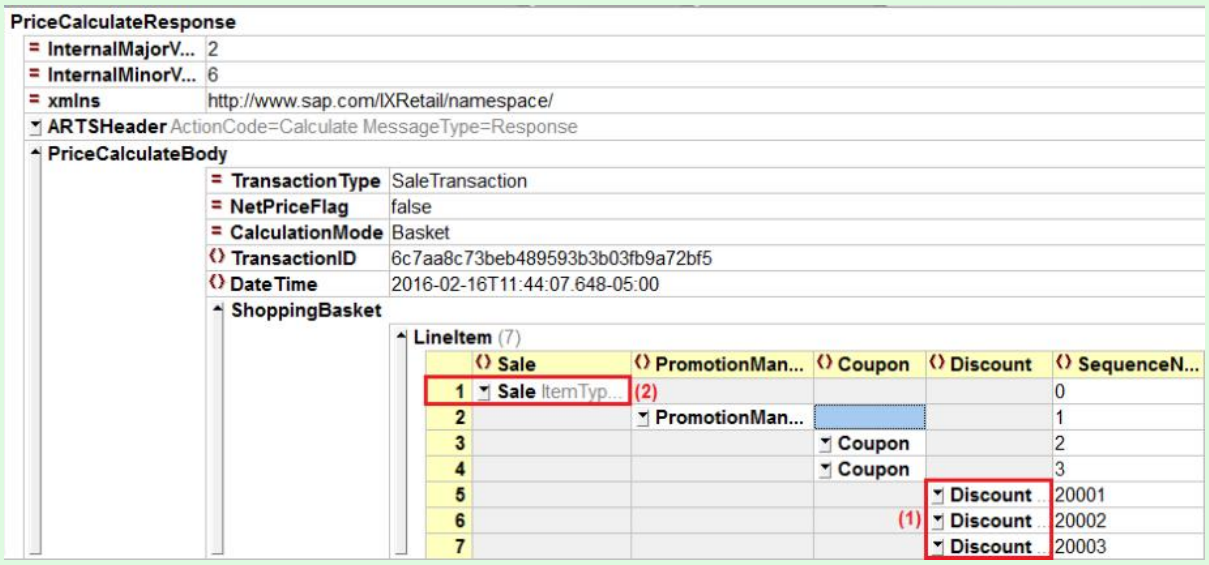
This PCE-request was created on 16 February 2016 (see (1) in preceding figure) and includes one sale (see (2) in preceding figure), a so-called manual trigger (see (3) in preceding figure), and two coupons (see (4) in preceding figure) in the shopping basket. A manual trigger is an action typically made by the cashier. For further information, see Manually Apply a Benefit. The content of a shopping basket is called line items.

3.1.3 PCE-Response

The PCE-response is generated by the PCE. The PCE-response contains all previously received information needed for the PPS-based price calculation. Additionally, it contains the modification of the regular sales unit prices. Information about the promotion price derivation rules that caused the modifications are part of the PCE-response, too. This results in the traceability of the changes made by the PCE. The added information is separated according to the two different main types of benefits, i.e. discounts and loyalty points.

Exemplary PCE-Response

The following figure illustrates an exemplary PCE-response after three promotion price derivation rules were applied to the exemplary PCE-request mentioned previously.



PriceCalculateResponse

= InternalMajorV...	2
= InternalMinorV...	6
= xmlns	http://www.sap.com/IXRetail/namespace/
ARTSHeader	ActionCode=Calculate MessageType=Response

PriceCalculateBody

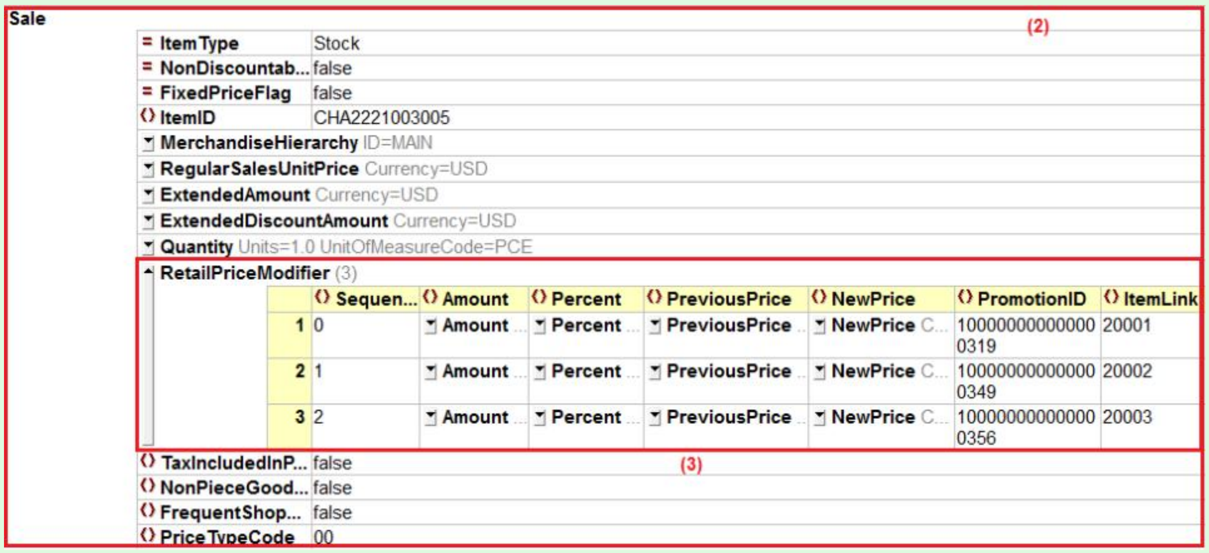
= TransactionType	SaleTransaction
= NetPriceFlag	false
= CalculationMode	Basket
TransactionID	6c7aa8c73beb489593b3b03fb9a72bf5
DateTime	2016-02-16T11:44:07.648-05:00

ShoppingBasket

LineItem (7)

	Sale	PromotionMan...	Coupon	Discount	SequenceN...
1	Sale ItemTyp... (2)				0
2		PromotionMan...			1
3			Coupon		2
4			Coupon		3
5				Discount ...	20001
6			(1)	Discount ...	20002
7				Discount ...	20003

The structure of the PCE-response is similar to the PCE-request except that three additional line items (see (1) in preceding figure) are added to it. This is due to the applied promotion price derivation rules that granted a discount for the whole PCE-request. In addition, the price modification of the sale (see (2) in preceding and succeeding figures) is listed according to the applied promotion price derivation rules. This is shown in the following figure whereby the price modification is listed in the retail price modifier (see (3) in succeeding figure).



Sale (2)

= ItemType	Stock
= NonDiscountab...	false
= FixedPriceFlag	false
ItemID	CHA2221003005
MerchandiseHierarchy	ID=MAIN
RegularSalesUnitPrice	Currency=USD
ExtendedAmount	Currency=USD
ExtendedDiscountAmount	Currency=USD
Quantity	Units=1.0 UnitOfMeasureCode=PCE

RetailPriceModifier (3)

	Sequen...	Amount	Percent	PreviousPrice	NewPrice	PromotionID	ItemLink
1	0	Amount ...	Percent ...	PreviousPrice ...	NewPrice C...	100000000000000319	20001
2	1	Amount ...	Percent ...	PreviousPrice ...	NewPrice C...	100000000000000349	20002
3	2	Amount ...	Percent ...	PreviousPrice ...	NewPrice C...	100000000000000356	20003

(3)

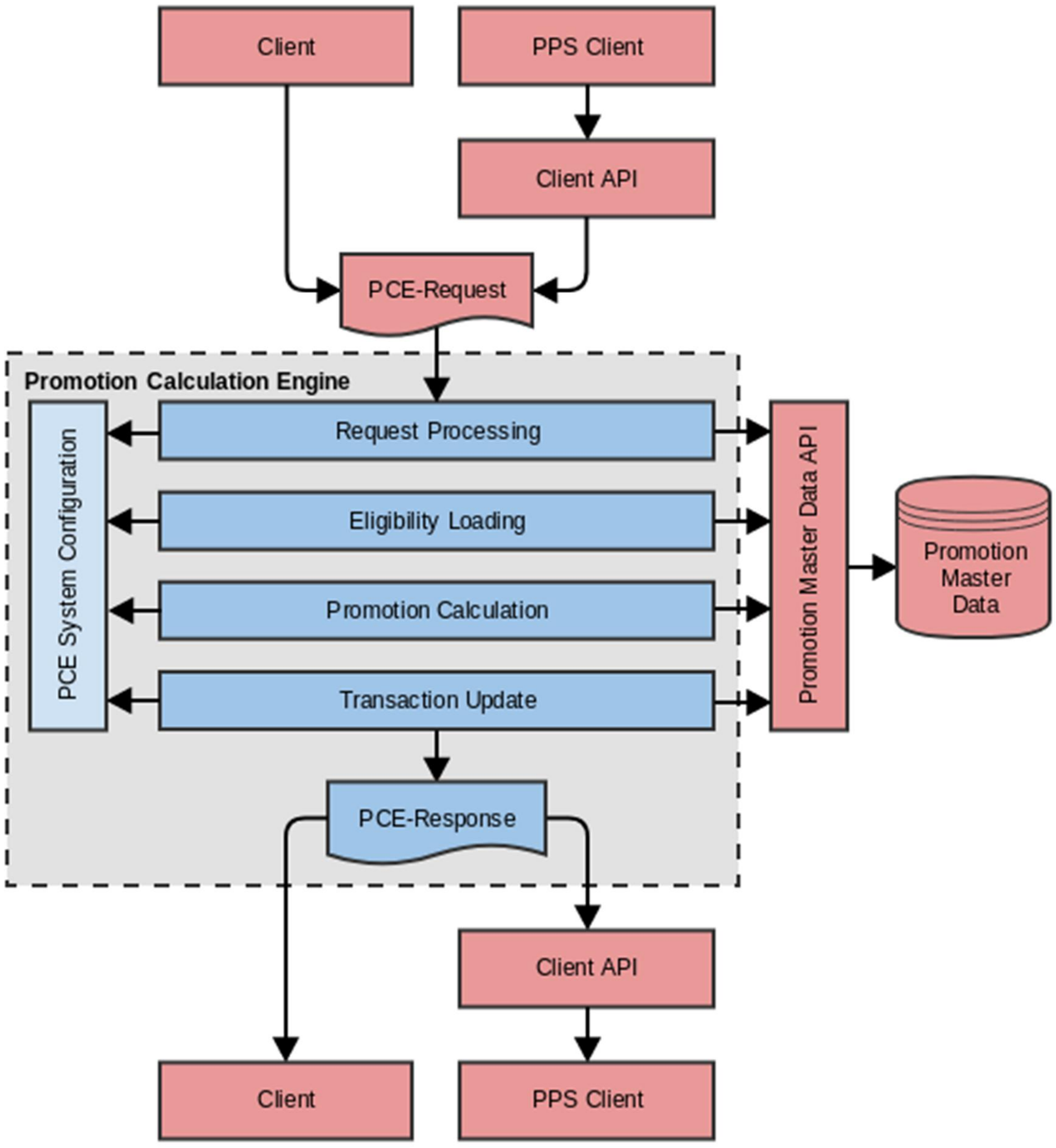
TaxIncludedInP...	false
NonPieceGood...	false
FrequentShop...	false
PriceTypeCode	00

The retail price modifier (see (3) in preceding figure) includes - amongst others - the so-called shares of the discount (the discount amount that results from a transaction-related promotion price derivation rule), the discount in percent, the previous price, the new price, and which promotion price derivation rule is applied to this particular line item.

3.2 Processed Steps

The PCE processes the PCE-request and the *promotion master data* always in the same way. This section includes a description of the execution steps. You learn how the received information is used and what modifications are made. Additionally, you find hints for further reading.

The following figure gives an overview of the processed steps:



3.2.1 Request Processing

In the first processing step, the received PCE-request is prepared for the calculation of the discounts and loyalty points.

HOW IT WORKS

The PCE receives the request via the Client API within the PPS. The PCE-request can be issued by a PPS Client or directly by the Client application (in the figure denoted by "Client"). It includes all required information about the purchase. A detailed description about the elements and the format of the PCE-request can be found in the document Client API for Omnichannel Promotion Pricing.

The PCE parses the PCE-request into an internal format, the *transaction*. The *transaction* represents the gradually modified PCE-request throughout the entire process.

Like the PCE-request, the *transaction* includes all necessary information about the purchase. One of the most important elements of a *transaction* is the *line item*. A *line item* represents an item of a shopping basket. The *line items* are later on used for triggering promotion price derivation rules.

After parsing the PCE-request, the PCE checks which calculation mode is active for the currently processed *transaction*. The calculation mode can be either the line item mode or the basket mode. When the line item mode is used, the *transaction* is split into parts that only contain one *line item*. When the basket mode is used, the *transaction* is handled entirely. In both cases, the transaction is prepared for the normalized calculation of line items.

THERE IS MORE

A detailed description of the impact on the promotion calculation by the request processing can be found in chapter Request Processing – Impact on the Promotion Calculation.

3.2.2 Eligibility Loading

The output of this step is a set of active price derivation rule eligibilities – that is, preconditions which are fulfilled by the currently processed transaction.

HOW IT WORKS

After the *transaction* is prepared, the *price derivation rule eligibility* is loaded via the *promotion master data* API. The *price derivation rule eligibility* describes the preconditions a *transaction* has to fulfill to apply the corresponding p*rice derivation rule*. It is one part of a *promotion price derivation rule*. The other part is the *price derivation rule*. You find a more detailed description of *promotion price derivation rules* in the next step.

There are several types of *price derivation rule eligibilities* which are activated by the appropriate triggers of the *transaction*. If all *price derivation rule eligibilities* are loaded, it is checked which of them are fulfilled. The fulfilled *price derivation rule eligibilities* are activated. For active *price derivation rule eligibilities*, it is (recursively) checked whether a *parent price derivation rule eligibility* exists and, if so, whether this *parent price derivation rule eligibility* can be activated as well.

THERE IS MORE

For further information about the different *price derivation rule eligibility* types, have a look at chapter Eligibility Loading – Triggers.

3.2.3 Promotion Calculation

In this step, the calculation of the discounts and loyalty points is performed. The output is the computed discounts and loyalty points for the currently processed transaction.

HOW IT WORKS

Based on the activated *price derivation rule eligibilities* and the content of the currently processed *transaction*, the benefits can be computed.

For this purpose, the *promotion price derivation rules* for the activated *price derivation rule eligibilities* are loaded via the *promotion master data* API. Afterwards, the PCE checks the validity of the promotion price derivation rules. The *promotion price derivation rules* are ordered according to a *sequence number*.

The PCE is capable to handle collisions, that is, cases when the *sequence* and *resolution* of two or more *promotion price derivation rules* is equivalent. The execution of the *price derivation rules* is handled separately on line item level and transaction level. The transaction-related promotion price derivation rules are calculated subsequently. They are always based on the line item-related promotion price derivation rules. There are various *price derivation rules*.

In summary, the following steps are executed within the step promotion calculation:

1. Load *promotion price derivation rules* for activated *price derivation rule eligibilities*.
2. Validate the promotion.
3. Execute the *price derivation rules* of the validated promotions
   a. for line item-related *promotion price derivation rules*,
   b. for transaction-related *promotion price derivation rules*.

THERE IS MORE

For further information about the preliminary functions of the promotion calculation, refer to chapter Promotion Calculation – General Behavior. The different types of *price derivation rules* are described in chapter Promotion Calculation – Price Derivation Rules.

3.2.4 Transaction Update

In this step, the computed discounts, loyalty points, and information for the Client application are added to the corresponding transaction elements. In addition, the PCE-response is created and transferred to the Client API.

HOW IT WORKS

The computed discounts and loyalty points of the previously executed steps are needed to update the currently processed transaction – that is the promotion calculation. In addition, the update also contains information of the applied *promotion price derivation rules* or the *promotion master data*, respectively.

If a benefit exists and several *line items* are affected, it is prorated to all related items in the *transaction*. Modifiers are prepared and added to the affected *line items*. Finally, the *transaction* is parsed into the PCE-response.

The PCE-response contains all information about the modification. For example, it contains the identifiers for the applied *promotion price derivation rules* and the calculated benefit. There are four modifiers: the retail price modifier or the price modification line item that include a monetary discount, and the frequent shopper points modifier or the loyalty reward line item that include loyalty points.

The *transaction* can also include the printout of a coupon, a gift certificate, and information for an action that the Client application performs afterwards.

A detailed description about the elements and the format of the PCE-response can be found in the document Client API for Omnichannel Promotion Pricing.

THERE IS MORE

For further information about behaviors that have an impact on the update of a *transaction*, refer to chapter Transaction Update – Finalize Promotion Calculation.

4 Request Processing – Impact on the Promotion Calculation

The first execution step is the request processing. The received PCE-request is prepared for the calculation of the discounts, loyalty points, and effective sales prices. For this purpose, the PCE-request is mapped to an internal format, the transaction. After parsing the PCE-request, the PCE checks which *calculation mode* is active for the currently processed transaction. The transaction is also prepared for the *normalized calculation* of line items during the execution.

In this chapter, the following topics are discussed:

- Select a Calculation Mode
- Calculate the Benefit with Normalized Quantities
- Load Ad Hoc Promotions

4.1 Select a Calculation Mode

The PCE supports two different *calculation modes*: the *basket* and the *line item mode*. Thereby, the PCE either considers the whole shopping basket or calculates the benefit independently for each line item.

HOW IT WORKS

When the PCE is called, the transaction includes the parameter *calculationMode*. This parameter can have the following values:

- BASKET: The PCE calculates the benefits for the entire shopping basket.
- LINE_ITEM: The PCE calculates the benefits per line item independently.

According to the parameter *calculationMode*, the PCE evaluates the promotion price derivation rules and applies the resulting benefits. Thereby, the default value is BASKET.

4.1.1 Set the Line Item Mode

The value of the parameter *calculationMode* is LINE_ITEM. The PCE computes the benefits for each line item in the transaction independently from each other. This means that promotion price derivation rules that affect multiple line items, for example, by containing eligibilities with several items, cannot be triggered. The transaction-related promotion price derivation rules can only be executed on line-items, which fulfill the eligibility of the transaction-related promotion price derivation rule on their own.

If the transaction does contain coupons, the PCE returns an error. The cause is that a coupon might be applicable only once but fits for multiple line items in the transaction. Thus, it would be considered for each of these line items but when using the *basket mode*, it might be consumed only once – depending on how the coupon eligibility is maintained. In addition, the transaction must not contain an external benefit.

4.1.2 Set the Basket Mode

The value of the parameter *calculationMode* is BASKET. The PCE computes the benefits for the entire shopping basket including promotion price derivation rules that require/affect multiple line items as well as transaction-related promotion price derivation rules. In addition, no specific restrictions apply.

4.1.3 THERE IS MORE

The *calculation mode* also controls whether any promotion can be applied or not. All promotion price derivation rules on client side contain the *calculation mode flag*, indicating whether the promotion is allowed in *line item mode* or not. The PCE forwards the configured *line item mode* to the *data access Layer (DAL)*, where it is evaluated. The PCE does not perform any additional validation on the promotions based on the *calculation mode*.

Additional information about the parameter can be also found in chapter Configuration.

Further information regarding the *DAL* can be found in the documentation SDK Promotion Calculation Engine > PCE Modules > The data access layer (DAL).

4.2 Calculate the Benefit with Normalized Quantities

The PCE calculates all benefits with normalized quantities. This results in the consistency of computed discounts and loyalty points. Thus, the calculation is independent of the capturing of items – separately versus summarized.

HOW IT WORKS

Each line item with a quantity n is split into n separate line items with quantity one. The line items with the quantity one are herein called *normalized line items*. After the splitting of line items, the PCE calculates the benefits. When the calculation is finished, the results are then aggregated to the real line item structure inside the transaction.

Example: Calculate the Benefit with Normalized Quantities
Let us assume that there exists an item chair that has a regular sales unit price of 89.95€.
Furthermore, there exists the following promotion price derivation rule: Buy several chairs with a regular sales price greater or equal to 100.00€ and get a discount of 3%. The maximum regular sales price is thereby 500.00€ for which a discount of 3% is applied.
Additionally assume that a customer buys six chairs in one purchase. These are summarized to one line item in the transaction. The regular sales price is then 539.70€.
The PCE calculates the discount of 15.01€ with normalized line items. Since the maximum regular sales price is exceeded, the PCE uses the 500.00€ as calculation base amount for the 3% discount.

Line Item	Calculation Base Amount	3% Discount of Calculation Base Amount	Regular Sales Price	Effective Sales Price
Chair with quantity 6	500.00€	15.01€	539.70€	524.69€

The discount would be 15.00€ if the PCE would not use the normalized calculation of line items, since 3% of 500.00€ is exactly 15.00€. If the same line item is captured six times separately, the request contains six chairs with quantity 1. Thereby, the PCE uses the fraction of the sales price of a line item as calculation base amount that is eligible for the promotion. In the current example this means the following:

	Line Item	Calculation Base Amount	3% Discount of Calculation Base Amount	Regular Sales Price	Effective Sales Price
	Chair with quantity 1	89.95€	2.70€	89.95€	87.25€
	Chair with quantity 1	89.95€	2.70€	89.95€	87.25€
	Chair with quantity 1	89.95€	2.70€	89.95€	87.25€
	Chair with quantity 1	89.95€	2.70€	89.95€	87.25€
	Chair with quantity 1	89.95€	2.70€	89.95€	87.25€
	Chair with quantity 1	50.25€	1.51€	89.95€	88.44€
Sum	Chair with quantity 6	500.00€	15.01€	539.70€	524.69€

The discount 1.51€ of the sixth chair is a result of the rounding and causes the 0.01€ difference to the exact discount of 3% of 500.00€.

This example is related to the *threshold type* AMT and the *simple discount* with the price modification method DISCOUNT_PERCENT. Refer to chapter Use Thresholds, Intervals, and Limits and Apply a Simple Discount for more information.

4.2.1 THERE IS MORE

The documentation SDK Promotion Calculation Engine describes the normalized calculation in greater detail. In addition, the discount amounts are rounded on the normalized item level as well. This can lead to different values for rounding than it may be expected for a line item.

4.3 Load Ad Hoc Promotions

The PCE is able to consume promotion price derivation rules which are applicable only to the currently processed transaction. These promotion price derivation rules are herein called *ad hoc promotions*. The *ad hoc promotions* are processed in addition to the promotion price derivation rules that are provided by the promotion master data. Thereby, these are handled by the PCE in the same way as the promotion price derivation rules originating from the promotion master data.

HOW IT WORKS

The processing and applying of *ad hoc promotions* consists of the following activities:

1. Retrieving the *ad hoc promotions*
2. Validating and activating *ad hoc promotions*
3. Registering *ad hoc promotions* to the PCE context
4. Applying the *ad hoc promotions* (Thereby exists no difference from applying promotions originating from the promotion master data.)

The *ad hoc promotions* are loaded by the PCE and are previously prepared by the Client application. Thereby, the structure of the *ad hoc promotions* is like the ones stored in the promotion master data. The *ad hoc promotions* are loaded before the loading of price derivation rule eligibilities in order to guarantee that these are processed like promotion price derivation rules originating from the promotion master data.

Ad hoc promotions can include only items, coupons, and/or merchandise categories as triggers as well as a combination of these. Any other trigger types are not supported for *ad hoc promotions*. All price derivation rule eligibilities described in an *ad hoc promotion* have to be fulfilled in order to apply its benefit to the currently processed transaction. An *ad hoc promotion* can include as a price derivation rule a simple discount, a mix and match, or applies loyalty points. The Client application prepares the *ad hoc promotions* and the PCE loads these for the processing of the transaction. The loading of other promotion price derivation rules from the promotion master data repository is not affected.

Common rules according to *sequence* (PromotionConditionSO.sequence) and *resolution* (PromotionConditionSO.resolution) apply for *ad hoc promotions*, too. This means that an *ad hoc promotion* may not be applied since another promotion price derivation rule with the same *sequence* and a higher *resolution* is as well valid for the currently processed transaction. Thus, the other promotion price derivation rule is applied and not the *ad hoc promotion*. Additionally, promotions with a smaller *sequence* apply before the *ad hoc promotion* with a greater *sequence*.

If mandatory data of the *ad hoc promotions* is missing or if the *ad hoc promotion* cannot be interpreted by the PCE, the *ad hoc promotion* is invalidated and thus, not applied to the transaction. Additionally, a warning is logged.

Example: Load Ad Hoc Promotions

Let us assume there exists the following items:

- Pair of socks: 10.00€
- Pair of winter shoes: 59.95€

Additionally assume that the following promotion price derivation rules are in the promotion master data:

1. Apply 5% discount on a pair of socks. (Sequence: 2)
2. Apply 20% discount on a pair of winter shoes. (Sequence: 4)

A customer buys a pair of winter shoes, a pair of socks, and provides a coupon. The Client application prepares an *ad hoc promotion* that grants 10% discount on the pair of socks if the customer shows a coupon and buys a pair of winter shoes (Sequence: 3 and a *calculation base sequence* (PromotionConditionRuleSO.calculationBaseSequence is -1).
The PCE receives the transaction from the Client application and retrieves the *ad hoc promotion*. It loads in addition the promotion price derivation rules in the promotion master data. These promotion price derivation rules and the *ad hoc promotions* are validated and activated if these are valid for the received transaction. Afterwards, the activated *ad hoc promotions* and promotion price derivation rules are applied by the PCE to the transaction.
This results in the following:

```
Pair of socks:                10.00€
- 5% discount:                 -0.50€
- 10% ad hoc discount:         -1.00€

Pair of winter shoes:         59.95€
- 20% discount:               -11.99€
-----------------------------------------------
Effective Total Amount:       56.46€
```

4.3.1 THERE IS MORE

Further information regarding the implementation and the extension possibilities are described in the documentation SDK Promotion Calculation Engine > How the PCE works > Request Processing >Load ad hoc promotions and SDK Promotion Calculation Engine > PCE Extension > Extension Points > Promotion data loading > Load ad hoc promotions, respectively.

5 Eligibility Loading – Triggers

The price derivation rule eligibility (*PromotionConditionEligibilitySO*) is very important for the selection of a benefit. One price derivation rule eligibility contains the precondition a transaction has to fulfill so that the corresponding price derivation rule is executed by the PCE. Thereby, the price derivation rule eligibility contains at least one so-called trigger. If the trigger is part of the currently processed transaction, the price derivation rule eligibility is activated. This comparison of the promotion master data – that is, the price derivation rule eligibility – with the current transaction is done within the step eligibility loading.

All triggered eligibilities as well as the corresponding combination eligibilities are validated for their status. The status code (*EligibilityStatusCode*) is expected to be STATUS_CODE_ACTIVE.

All price derivation rule eligibilities are maintained as trees in the promotion master data. This means that one price derivation rule eligibility can be the parent of several child price derivation rule eligibilities. The nodes of this tree contain the combination type of these price derivation rule eligibilities. The possible triggers and combination types are explained in the subsequent sections. It depends on the combination type, whether a parent eligibility is activated or not after the child eligibility was activated. A simple price derivation rule eligibility is maintained as a tree with one node in the promotion master data.

A special promotion price derivation rule is the manual promotion price derivation rule. It includes a manual price derivation rule eligibility and a manual price derivation rule. For a better understanding, both parts are described in chapter Manually Apply a Benefit that is not part of this chapter.

In this chapter, the following topics are discussed:

- Use an Item as a Trigger
- Use a Merchandise Category as a Trigger
- Use a Simple Product Group as a Trigger
- Use a Product Group as a Trigger
- Use Thresholds, Intervals, and Limits
- Use the Total of a Shopping Basket as a Trigger
- Use a Customer Group as a Trigger
- Use a Coupon as a Trigger
- Combine Triggers

THERE IS MORE

The activation of a price derivation rule eligibility in case of sold and/or returned items is described in Activate Price Derivation Rule Eligibilities without the Original Transaction. Especially, the behavior is explained with respect to the sale return type.

Related topics are:

- Activate Price Derivation Rule Eligibilities without the Original Transaction
- Handle Collisions with Respect to Sales and Returns
- Choose Items with Respect to Sales and Returns
- Compute Total Price Modification Methods with Respect to Sales and Returns
- Mix and Match with Respect to Sales and Returns
- Select the Method to Discount

5.1 Use an Item as a Trigger

The price derivation rule eligibility (*PromotionConditionEligibilitySO*) can contain a combination of different triggers. One of these triggers is the item. If a particular item is part of the transaction, the PCE applies the promotion price derivation rule stating the particular item in its price derivation rule eligibility. This price derivation rule eligibility is herein called item eligibility (*ItemPromotionConditionEligibilitySO*).

HOW IT WORKS

An item in the transaction is – among others – described by two attributes: the unique item identifier (*ItemPromotionConditionEligibilitySO.itemID*) and the unit of measure (*ItemPromotionConditionEligibilitySO.unitOfMeasureCode*).

This information is compared with the data of the active item eligibilities loaded via the promotion master data API. It is always called with both the item identifier and the unit of measure specified in the transaction for the given item when retrieving its price derivation rule eligibilities.

To apply a particular benefit to an item, the item eligibility has to be part of the promotion price derivation rule. The item identifier has to be stated in the item eligibility and the unit of measure can be stated in addition.

Example: Use an Item as a Trigger
Let us assume that there is an item "Arabica coffee". The item can be sold as packaged 500 g or as an individually scaled portion. In both cases, the Arabica coffee has the same identifier: "42".
However, the unit of measure is different. The packaged 500 g are sold by the piece which results in the unit of measure code "PCE". In contrast, the individual scaled portion of the Arabica coffee is sold by the kilogram which results in the unit of measure code "KG".

Let us assume that a customer shall get a benefit for buying Arabica coffee. For this purpose, a promotion price derivation rule has to be part of the master data that contains an item eligibility:

1. First possibility: The item eligibility includes only the item identifier "42" and the unit of measure code "_ALL".
2. Second possibility: The item eligibility includes the item identifier "42" and the unit of measure code "PCE".
3. Third possibility: The item eligibility includes the item identifier "42" and the unit of measure code "KG".

Let us assume that there is a promotion price derivation rule that contains an item eligibility for the item "Arabica coffee". In addition, we assume that a customer buys a 500 g package of Arabica coffee as a gift for a friend and a portion of Arabica coffee for himself.

Thus, the transaction contains two line items:

- The first line (the 500 g package) with item identifier "42" and unit of measure code "PCE".
- The second line item (the portion) with item identifier "42" and unit of measure code "KG".

The result would be as follows according to the different possibilities to define an item eligibility for the item:

1. First possibility: The item eligibility includes only the item identifier "42" and the unit of measure code "_ALL".
   - Both items trigger the promotion price derivation rule.
   - The customer gets a benefit for both items.
2. Second possibility: The item eligibility includes the item identifier "42" and the unit of measure code "PCE".
   - Only the 500 g package of Arabica coffee triggers the promotion price derivation rule.
   - The customer gets a benefit for the 500 g package of Arabica coffee he buys for his friend.
3. Third possibility: The item eligibility includes the item identifier "42" and the unit of measure code "KG".
   - Only the portion of Arabica coffee triggers the promotion price derivation rule.
   - The customer gets a benefit for the portion of Arabica coffee he buys for himself.

THERE IS MORE

The item as a trigger can be enhanced with values for a threshold, an interval, and/or a limit for the amount and/or the quantity of the corresponding item. See Use Thresholds, Intervals, and Limits for more information.

Further information about the objects and attributes can be found in chapter Configuration.

5.2 Use a Merchandise Category as a Trigger

A possible trigger type is the merchandise category. A merchandise category is a collection of items that have certain similar properties. If a particular item is part of a merchandise category and of the transaction, the PCE applies the promotion price derivation rule stating this particular merchandise category in its price derivation rule eligibility. This price derivation rule eligibility is herein called merchandise category eligibility (*MHGPromotionConditionEligibilitySO*).

The merchandise categories can be organized in a hierarchy. Thereby, these are linked with merchandise category hierarchy node. This means that an item that belongs to a merchandise category belongs also to the linked merchandise category node. The PCE does not differentiate between merchandise categories and merchandise category hierarchy nodes since these are all handled in the same way.

HOW IT WORKS

A merchandise category is defined by two attributes in the transaction: the merchandise category identifier (*SaleReturnLineItemMerchandiseHierarchyGroup.Key.MerchandiseHierarchyGroupID*) and the merchandise category qualifier (*SaleReturnLineItemMerchandiseHierarchyGroup.Key.MerchandiseHierarchyGroupIDQualifier*).

The merchandise category identifier is unique in the context of the same merchandise category qualifier. The merchandise category qualifier is used to differ between merchandise categories that belong to different hierarchies. This hierarchy includes merchandise category hierarchy nodes and merchandise categories which do not have any children.

The relation between the items and their merchandise category is transmitted as a part of the transaction. It is also compared with the data of the active price derivation rule eligibilities that are loaded via the promotion master data API.

Example: Use a Merchandise Category as a Trigger
Let us assume that there is a merchandise category with the identifier "chair", the merchandise category "SEAS" and a merchandise category with the identifier "furniture", the merchandise category "SEAS". The merchandise category "chair" is a subcategory of the merchandise category "furniture". This means that an item "office chair" that belongs to the merchandise category "chair" automatically belongs to the merchandise category "furniture", too. On the contrary, an item "desk" that belongs to the merchandise category "furniture" does not belong to the merchandise category "chair".

To apply a benefit that is triggered by a merchandise category, the merchandise category eligibility has to be part of the promotion price derivation rule. The merchandise category identifier can be stated in the price derivation rule eligibility as well as the merchandise category qualifier.

Example (cont.)

Let us assume that a customer shall get a benefit for buying an office chair and/or a desk. For this purpose, a promotion price derivation rule containing a merchandise category eligibility has to be part of the promotion master data.

1. First possibility: The merchandise category eligibility includes the merchandise category identifier "chair" and the merchandise category qualifier "SEAS".
2. Second possibility: The merchandise category eligibility includes the merchandise category identifier "table" and the merchandise category qualifier "SEAS".
3. Third possibility: The merchandise category eligibility includes the merchandise category identifier "furniture" and the merchandise category qualifier "SEAS".

A merchandise category eligibility is fulfilled if the item belongs to the indicated merchandise category in the transaction. All assignments of items to merchandise categories and merchandise category hierarchy nodes are part of the transaction data in the context of OPP.

Example (cont.)

Let us assume that a customer buys a kitchen chair and a desk for the family. Thus, the transaction contains two line items:

1. First line item: kitchen chair, merchandise category "chair", merchandise qualifier "SEAS".
2. Second line item: desk, merchandise category "table", merchandise qualifier "SEAS".

In the OPP context, the PPS is called and thus the PCE. The merchandise category "chair", the merchandise qualifier "SEAS" and the merchandise category "table", the merchandise qualifier "SEAS" are subcategories of the merchandise category "furniture", the merchandise qualifier "SEAS". The result would be as follows according to the different possibilities to define a merchandise category eligibility:

1. First possibility: The merchandise category eligibility includes the merchandise category identifier "chair" and the merchandise qualifier "SEAS".
   - Only the desk triggers the promotion price derivation rule.
   - The customer gets a benefit for desk item.
2. Second possibility: The merchandise category eligibility includes the merchandise category identifier "table" and the merchandise qualifier "SEAS".
   - Only the kitchen chair triggers the promotion price derivation rule.
   - The customer gets a benefit for the kitchen chair.
3. Third possibility: The merchandise category eligibility includes the merchandise category identifier "furniture" and the merchandise category qualifier "SEAS".
   - Both items triggers the promotion price derivation rule.
   - The customer gets a benefit for both items.
4. Forth possibility: The merchandise category eligibility includes the merchandise category identifier "furniture" and the merchandise category qualifier "MAIN".
   - No items triggers the promotion price derivation rule.
   - The customer does't get a benefit.

THERE IS MORE

It is possible to maintain a root merchandise category hierarchy node in the promotion master data. All merchandise categories are children of this root merchandise category hierarchy node. If this is done by the retailer, promotion price derivation rule eligibilities can be created for this root merchandise category hierarchy node – herein root merchandise category eligibility. The promotion price derivation rule is thereby on line item level. This promotion price derivation rule is in some aspects like a transaction-related one since the entire line items are considered for the calculation of the benefit. The advantage is that thereby not only a shopping basket threshold (*MarketBasketAmountEligibilitySO.marketBasketThresholdAmount*) can be defined but thresholds, intervals, and limits like described here. Another difference occurs with regard to the following flags:

- Discount flag (SaleReturnLineItem.discountFlag): A flag to indicate whether this line item can be discounted (false) or not (true).
- Eligible for loyalty points flag (SaleReturnLineItem.frequentShopperPointsEligibilityFlag): A flag to denote that the line item is eligible for loyalty points (false) or not (true).
- Not considered by the PCE flag (SaleReturnLineItem.notConsideredByLoyaltyEngineFlag): Determines whether the PCE should care about the line item as a trigger (false) or not (true).

Let us assume the following:

- A line item-related promotion price derivation rule is applied, whereby
- The not considered by the PCE flag is true, and
- Either the discount flag is true or the eligible for loyalty points flag is true.

The PCE does not apply a discount and does not grant loyalty points, respectively. If the promotion price derivation rule is on transaction level, the promotion price derivation rule is applied. This means that a root merchandise category eligibility in a line item-related promotion price derivation rule cannot replace a transaction-related one.

The merchandise category as a trigger can be enhanced with values for a threshold, an interval, or a limit for the amount or the quantity. See Use Thresholds, Intervals, and Limits for more details.

For further information about the attributes, see chapter Configuration.

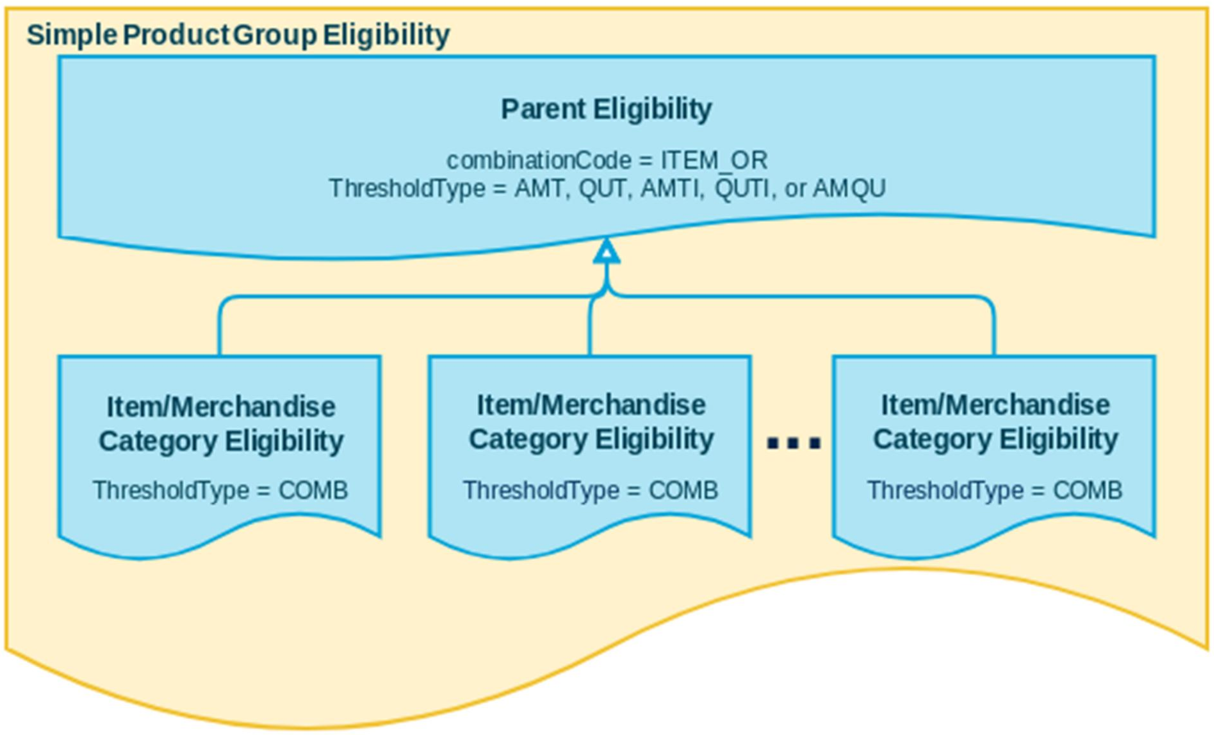
5.3 Use a Simple Product Group as a Trigger

A simple product group is a collection of items and merchandise categories with certain similar properties. If an item or merchandise category is part of a simple product group and of the transaction, the PCE applies the corresponding promotion price derivation rule.

The items and the merchandise categories are stated in several price derivation rule eligibilities. These price derivation rule eligibilities are herein called simple product group eligibility (*CombinationPromotionConditionEligibilitySO.combinationCode* = ITEM_OR).

HOW IT WORKS

To handle simple product groups as a trigger, several price derivation rule eligibilities are hierarchically organized like shown in the following figure:


Simple Product Group Eligibility
Parent Eligibility
combinationCode = ITEM_OR
ThresholdType = AMT, QUT, AMTI, QUTI, or AMQU
Item/Merchandise Category Eligibility
ThresholdType = COMB
Item/Merchandise Category Eligibility
ThresholdType = COMB
...
Item/Merchandise Category Eligibility
ThresholdType = COMB

All child item and/or merchandise category eligibilities contain a link to their parent price derivation rule eligibility that includes the elements of the simple product group. All other kinds of price derivation rule eligibilities cannot be a part of the simple product group eligibility. Thus, the parent eligibility is a special kind of combination eligibility. The combination code (*CombinationPromotionConditionEligibilitySO.combinationCode*) *ITEM_OR* indicates that this price derivation rule eligibility is forming a simple product group eligibility. Additionally, all item and merchandise category eligibilities have a special threshold type (*ThresholdType*) to indicate that they belong to a simple product group eligibility. The threshold type is *COMB*. This means that the parent eligibility includes a threshold type for the simple product group eligibility.

The transaction includes the items with their identifier, unit of measure, and the relation to their merchandise categories. If these items belong to a simple product group that shall get a benefit, the PCE activates the simple product group eligibility. This means that each item triggers an item eligibility and is thus activated. If this item eligibility has the threshold type *COMB,* it is a child eligibility that belongs to a simple product group eligibility. The PCE automatically checks whether the corresponding parent eligibility is fulfilled as well.

The same happens for an item that belongs to a merchandise category triggering a merchandise category eligibility with the threshold type *COMB*. The parent eligibility is activated if its thresholds, intervals, and limits are fulfilled by the transaction. This means that the PCE treats the simple product group like an item or a merchandise category – despite the fact that a simple product group eligibility is made of several item and/or merchandise category eligibilities. The threshold, the interval, and the limit quantities or amounts are maintained at the parent price derivation rule eligibility. The quantities or amounts of the items or merchandise categories are summarized. These sums are checked against the values in the parent price derivation rule eligibility.

Note that if the system parameter *timeValidationMethod* is ELIGIBILITY, the validity range is checked for all triggered item and merchandise category eligibilities and the corresponding combination eligibility.

Example: Use a Simple Product Group as a Trigger
Let us assume that a simple product group "office equipment" includes all items that belong to the merchandise category "stationary" and the item "office chair." In addition, there is a promotion price derivation rule with the following simple product group eligibility: A customer who buys three items of the simple product group "office equipment" gets a benefit.
If a customer buys three office chairs, he triggers the promotion price derivation rule with the previously mentioned simple product group eligibility. If a customer buys two items of the merchandise category "stationary" and one "office chair", the items of the transaction trigger the price derivation rule eligibility as well. The latter case is the main difference compared to the combination eligibility with the combination code OR.

THERE IS MORE

Since the simple product group eligibility is a special kind of combination eligibility, it is subject to the same terms like the other kinds of combination eligibilities (see chapter Combine Triggers for details).

It is also restricted that promotions with simple product groups as eligibility may not define a Mix and Match rule. All other rule types are allowed.

The parent eligibility of a simple product group eligibility can contain a threshold type and is thus capable to consume quantities or amounts. The impacts of the different threshold types are described in chapter Use Thresholds, Intervals, and Limits.

A simple product group may contain items and merchandise categories. For information on the main attributes of items or merchandise categories, see chapters Use an Item as a Trigger or Use a Merchandise Category as a Trigger, respectively.

For more information on promotion and eligibility validation, see chapter Check the Validity of a Promotion.

5.4 Use a Product Group as a Trigger

The product group is a possible trigger type for a price derivation rule eligibility. If an item of a product group is part of a transaction, the PCE executes the corresponding price derivation rule. The price derivation rule eligibility is herein called product group eligibility (*MerchandiseSetPromotionConditionEligibilitySO*).

HOW IT WORKS

The product group is defined by the attribute product group identifier (*MerchandiseSetPromotionConditionEligibilitySO.merchandiseSetID*). The product group identifier refers to all elements of the product group that are part of it. Like the simple product group, the product group is composed of items and merchandise categories.

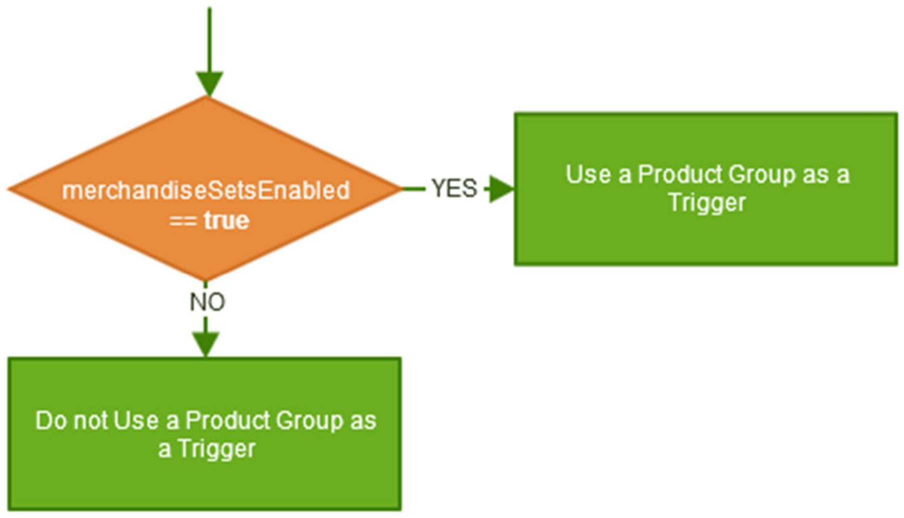
The main functional difference compared to a simple product group is that certain merchandise categories or items can be excluded from the product group. In addition, the product group eligibility is not realized via several combined price derivation rule eligibilities as simple product groups.

The system configuration flag *merchandiseSetsEnabled* activates the usage of the product groups (true) or it deactivates it (false). The system configuration flag *merchandiseSetsEnabled* is true by default.

To apply a benefit triggered by a product group, the product group eligibility has to be part of the promotion price derivation rule. The product group eligibility contains the product group identifier. If a promotion price derivation rule is created with a product group eligibility and the product group is changed afterwards, the promotion price derivation rule is still granted for all elements of the product group that were part of it at the creation time of the promotion price derivation rule.

A product group eligibility is fulfilled if an item in a transaction is part of it or if it is assigned to a merchandise category that is part of the product group. The PCE does not need the detailed structure of a product group. The promotion master data API provides the product group. This means that after the PCE processed the request, the price derivation rule eligibilities are loaded via the promotion master data API. If the system configuration flag *merchandiseSetsEnabled* is true, the PCE also loads the product group eligibilities for the line items in the received transaction.

The following figure illustrates, which value of the flag is resulting in which of the previously described scenarios.


merchandiseSetsEnabled == true
YES
Use a Product Group as a Trigger
NO
Do not Use a Product Group as a Trigger

THERE IS MORE

The product group works also with a mix and match price derivation rule since it is realized as an independent data object in the context of a promotion and not via a combination of eligibilities like the simple product group.

Information on the attributes of items or merchandise categories can be found in chapters Use an Item as a Trigger or Use a Merchandise Category as a Trigger, respectively.

The product group as a trigger can be enhanced with values for a threshold, an interval, and/or a limit for the amount and/or the quantity of the corresponding item. Have a look at Use Thresholds, Intervals, and Limits for details.

For further information about the system configuration flag *merchandiseSetsEnabled*, see chapter Configuration.

5.5 Use Thresholds, Intervals, and Limits

Some triggers of a price derivation rule eligibility can be described among others by quantities (*SaleReturnLineItem.quantity* times *SaleReturnLineItem.units*) and amounts according to the calculation base amount. It is possible to configure eligibilities in a way that predefined quantities and/or amounts – herein called threshold types (*ThresholdType*) – will trigger the particular price derivation rule eligibility – herein threshold eligibility (*ThresholdPromotionConditionEligibility*).

HOW IT WORKS

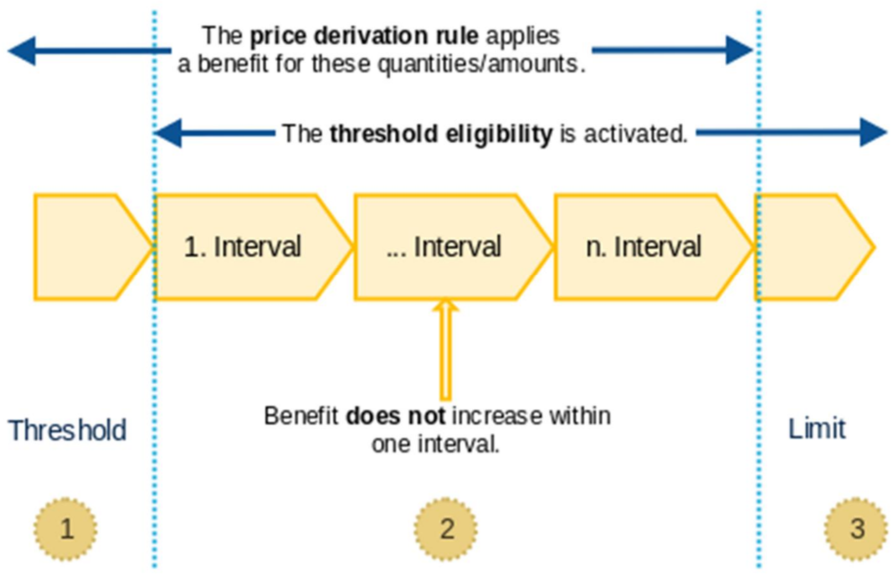
The amount of an item can be the regular sales price of the item (*SaleReturnLineItem.regularUnitPrice* times quantity) minus a previously applied discount.

The threshold types are selectable for the following price derivation rule eligibilities:

- Item eligibility
- Merchandise category eligibility
- Simple product group eligibility
- Product group eligibility
- Combination eligibility

A threshold type is described by a threshold, an interval, and/or a limit for quantities or amounts. The threshold describes the minimal quantity or amount a trigger type must have to get a benefit – herein called quantity threshold (*ThresholdPromotionConditionEligibility.thresholdQuantity*) or amount threshold (*ThresholdPromotionConditionEligibility.thresholdAmount*). The limit is the maximal quantity or amount a trigger gets a benefit for – herein called quantity limit (*ThresholdPromotionConditionEligibility.limitQuantity*) or amount limit (*ThresholdPromotionConditionEligibility.limitAmount*). Thus, the limit always has to be equal to or greater than the threshold of a price derivation rule eligibility. The interval defines ranges of quantities and/or amounts within the threshold and the limit – herein called quantity interval (*ThresholdPromotionConditionEligibility.intervalQuantity*) or amount interval (*ThresholdPromotionConditionEligibility.intervalAmount*). The benefit is the same for every quantity and/or amount within the same range or interval, respectively.

The following figure depicts the correlation between the threshold, the interval, and the limit.


The **price derivation rule** applies a benefit for these quantities/amounts.
The **threshold eligibility** is activated.
1. Interval
... Interval
n. Interval
Threshold
Benefit **does not** increase within one interval.
Limit
1
2
3

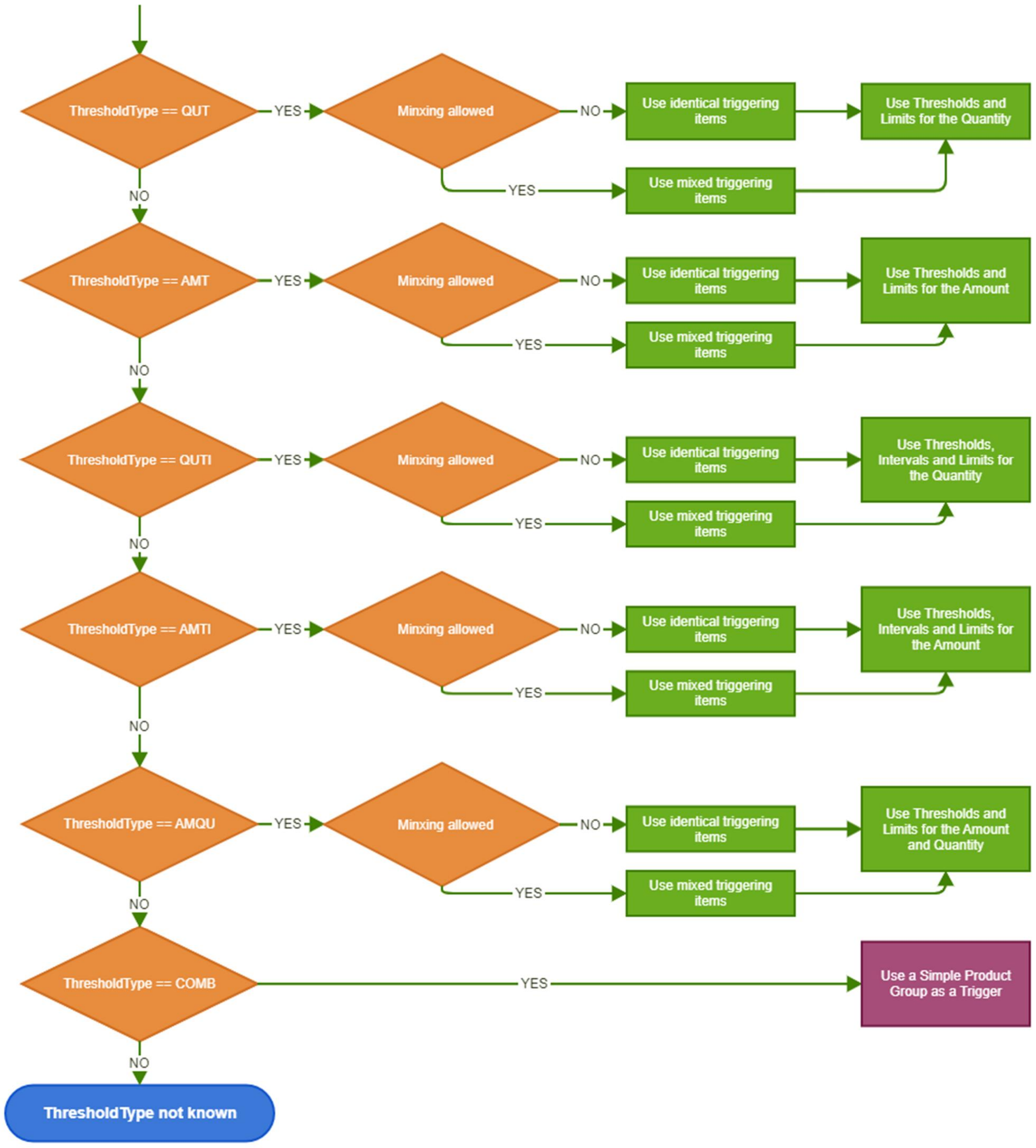
The threshold indicates the minimum quantity and/or amount of a certain trigger that has to be part of the transaction to activate a threshold eligibility. The limit indicates the maximum quantity and/or amount of a trigger that can get a benefit. The interval subdivides the range between the threshold and the limit. The interval itself is a range value. The applied benefit is the same for any quantity or amount of a trigger within an interval, respectively. This means that quantities and/or amounts that are less than the threshold do not activate the threshold eligibility – area 1 in the figure. In this case, no benefit is applied to the transaction. If a transaction includes quantities and/or amounts that are greater than or equal to the threshold – area 2 and 3 in the figure – the threshold eligibility is activated. This results in a benefit that is applied to the transaction. However, if the quantities and/or amounts exceed the limit – area 3 in the figure – the price derivation rule is executed and the benefit is computed only for the limit. Thus, a maximal possible benefit is configurable. For details, see the remarks in section THERE IS MORE.

Example: Use Thresholds, Intervals, and Limits
Let us assume that there are two items: the "Kitchen Chair" for 79.95€ and the "Office Chair" for 99.95€. Both items are part of the merchandise category "chair".
For every kind of threshold type, a threshold is defined. It can be, for instance, a threshold on the quantity – for example 2 kitchen chairs, 3 office chairs, or 4 items of merchandise category “chair”. The threshold can be on the amount, too. Kitchen chairs amounting to 150.00€, office chairs amounting to 200.00€, or items of the merchandise category “chair” amounting to 350.00€ are examples for an amount threshold.
The limit is also part of every threshold type. It follows the same principle as the threshold. Since the limit has to be greater than the threshold, it is not possible to have a limit of 300.00€ for chairs if the threshold is 350.00€.
The interval describes a range within the threshold and the limit. For a threshold of 150.00€ and a limit of 600.00€, the interval can be, for example, 5.00€, 250.00€, or 400.00€ but not 500€.

To consume quantities and/or amounts of an item or a merchandise category, the price derivation rule eligibilities have to contain the threshold type. This attribute defines how the PCE is interpreting the values for the threshold, the interval, and/or the limit. It can be set in the promotion master data with the following values:

- *QUT*: The threshold and the limit are set for the quantity.
- *AMT*: The threshold and the limit are set for the amount.
- *QUTI*: The threshold, the interval, and the limit are set for the quantity.
- *AMTI:* The threshold, the interval, and the limit are set for the amount.
- *AMQU*: The threshold and the limit are set for the quantity and for the amount.
- *COMB*: see chapter Use a Simple Product Group as a Trigger

The different attribute values and their meaning are roughly illustrated in the following decision tree:


ThresholdType == QUT
YES
Minxing allowed
NO
Use identical triggering items
Use Thresholds and Limits for the Quantity
YES
Use mixed triggering items
NO
ThresholdType == AMT
Use Thresholds and Limits for the Amount
ThresholdType == QUTI
Use Thresholds, Intervals and Limits for the Quantity
ThresholdType == AMTI
Use Thresholds, Intervals and Limits for the Amount
ThresholdType == AMQU
Use Thresholds and Limits for the Amount and Quantity
ThresholdType == COMB
Use a Simple Product Group as a Trigger
ThresholdType not known

5.5.1 Use Thresholds and Limits for the Quantity

To use thresholds and limits for the quantity, the threshold type has to be *QUT*. This threshold type is only triggered by quantities of items that are part of the possible triggers. Possible triggers are: items, merchandise categories, simple product groups, and product groups.

A quantity threshold and a quantity limit are set in the price derivation rule eligibility. To activate the price derivation rule eligibility, the transaction has to include the trigger with the correct quantities.

Example (cont.)

Let us assume there is a promotion price derivation rule that contains an eligibility with the threshold type QUT. In addition, we assume that the customer buys one kitchen chair and two office chairs for the family. Thus, the transaction contains two line items:

1. First line item: item identifier "kitchen chair", regular price 79.95€, quantity "1.0", merchandise category "chair"
2. Second line item: item identifier "office chair", regular price 99.95€, quantity "2.0", merchandise category "chair"

The result would be as follows:

- First possibility: The merchandise category eligibility includes the quantity threshold on chairs that is "2", and a quantity limit of "8" (*ThresholdType* = QUT).
  - Both line items trigger the merchandise category eligibility.
  - The quantity threshold of 2 is reached since three items of the merchandise category are part of the transaction.
  - The quantity limit is not reached.
  - The customer gets the benefit according to the corresponding price derivation rule for all three items.

5.5.2 Use Thresholds and Limits for the Amount

To use thresholds and limits for the amount, the threshold type has to be *AMT*. This threshold type is only triggered by amounts of items that are part of the possible triggers. Possible triggers are: items, merchandise categories, simple product groups, and product groups.

An amount threshold and an amount limit are set in the price derivation rule eligibility. The transaction is compared with the price derivation rule eligibility. To activate the price derivation rule eligibility, the transaction has to include the trigger with the correct amounts. For this purpose, the corresponding amount of the calculation base amount of the promotion price derivation rule (possibly reduced by the already granted monetary discount) is checked against these limits instead of the original amount.

Example (cont.)

Let us assume there is a promotion price derivation rule that contains an eligibility with the threshold type AMT. In addition, we assume that the customer buys one kitchen chair and two office chairs for the family. Thus, the transaction contains two line items:

1. First line item: item identifier "kitchen chair", regular price 79.95€, quantity "1.0", merchandise category "chair", amount 79.95€
2. Second line item: item identifier "office chair", regular price 99.95€, quantity "2.0", merchandise category "chair", amount 199.90€

The result would be as follows:

- Second possibility: The merchandise category eligibility includes the amount threshold on chairs that is "150.00€", and an amount limit of "500.00€" (*ThresholdType* = AMT).
  - Both line items trigger the merchandise category eligibility.
  - The amount threshold of 150.00€ is reached since three items of the merchandise category are part of the transaction and have together an amount of 279.85€.
  - The amount limit is not reached.
  - The customer gets the benefit according to the corresponding price derivation rule for all three items.

5.5.3 Use Thresholds, Intervals, and Limits for the Quantity

To use thresholds, intervals, and limits for the quantity, the threshold type has to be *QUTI*. This threshold type is only triggered by quantities of items that are part of the possible triggers. Possible triggers are: items, merchandise categories, simple product groups, and product groups. A quantity threshold, a quantity interval, and a quantity limit are set in the price derivation rule eligibility. The transaction is compared with the price derivation rule eligibility. To activate the price derivation rule eligibility, the transaction has to include the trigger with the correct quantities.

Example (cont.)

Let us assume there is a promotion price derivation rule that contains an eligibility with the threshold type QUTI. In addition, we assume that the customer buys one kitchen chair and two office chairs for the family. Thus, the transaction contains two line items:

1. First line item: item identifier "kitchen chair", regular price 79.95€, quantity "1.0", merchandise category "chair"
2. Second line item: item identifier "office chair", regular price 99.95€, quantity "2.0", merchandise category "chair"

The result would be as follows according:

- Third possibility: The merchandise category eligibility includes the quantity threshold on chairs that is "2", a quantity limit of "8", and additionally a quantity interval of "2" (*ThresholdType* = QUTI).
  - Both line items trigger the merchandise category eligibility.
  - The quantity threshold of 2 is reached since three items of the merchandise category are part of the transaction.
  - The quantity limit is not reached.
  - The quantity interval is 2, thus the customer gets the benefit for two quantities of the two line items.
  - The PCE chooses the line items that get the benefit (see Choose Items):
    - If the smallest achievable benefit shall be obtained, the customer gets a benefit for the kitchen chair and for one office chair.

- If the highest achievable benefit shall be obtained, the customer gets a benefit for the two office chairs.

The customer gets no additional benefit for the third chair even though the quantity limit is not reached. The reason is that the interval is "2". Thus, the customer would get an additional benefit for four chairs, for six chairs, and for eight chairs (the threshold plus a multiple of the interval). If the customer buys three, five, or seven chairs, the benefit would be the same as for two, four, or six chairs. If the customer buys more than eight chairs, the benefit will not increase any further. This means that the customer would get a benefit as if he had bought eight chairs.

In the following, the result for a simple discount of 2% is calculated. n is the number of kitchen chairs the customer buys. The customer does not buy any office chair.

1. *n < 2:* The discount is not applied for one kitchen chair (total discount = 0.00€).
2. *2 <= n <= 8:* The discount is applied to two, four, six, or eight kitchen chairs. See the solution table:

n	Applied discount
2	2 pieces * 2% * 79.95€ = **3.20€** (total discount; each chair gets a discount of 1.60€)
3	2 pieces * 2% * 79.95€ = **3.20€** (total discount; two chairs get a discount of 1.60€ each, while the first chair gets none)
4	4 pieces * 2% * 79.95€ = **6.40€** (total discount; each chair gets a discount of 1.60€)
5	4 pieces * 2% * 79.95€ = **6.40€** (total discount; four chairs get a discount of 1.60€ each, while the first chair gets none)
6	6 pieces * 2% * 79.95€ = **9.60€** (total discount; each chair gets a discount of 1.60€)
7	6 pieces * 2% * 79.95€ = **9.60€** (total discount; six chairs get a discount of 1.60€ each, while the first chair gets none)
8	8 pieces * 2% * 79.95€ = **12.80€** (total discount; each chair gets a discount of 1.60€)

3. *n > 8:* The discount is applied to eight kitchen chairs which results in a total discount of 12.80€ (eight chairs get a discount of 1.60€ each while any other chair gets none).

5.5.4 Use Thresholds, Intervals, and Limits for the Amount

To use thresholds, intervals, and limits for the amount, the threshold type has to be *AMTI.* This threshold type is only triggered by amounts of items that are part of the possible triggers. Possible triggers are: items, merchandise categories, simple product groups, and product groups. An amount threshold, an amount interval, and an amount limit are set in the price derivation rule eligibility. The transaction is compared with the price derivation rule eligibility. To activate the price derivation rule eligibility, the transaction has to include the trigger with the correct amounts. The corresponding amount of the calculation base amount of the promotion price derivation rule (possibly reduced by the already granted monetary discount) is checked against these limits instead of the original amount.

Example (cont.)

Let us assume there is a promotion price derivation rule that contains an eligibility with the threshold type AMTI. In addition, we assume that the customer buys one kitchen chair and two office chairs for the family. Thus, the transaction contains two line items:

1. First line item: item identifier "kitchen chair", regular price 79.95€, quantity "1.0", merchandise category "chair", amount 79.95€
2. Second line item: item identifier "office chair", regular price 99.95€, quantity "2.0", merchandise category "chair", amount 199.90€

The result would be as follows according to the fourth possibility:

- Fourth Possibility: The merchandise category eligibility includes the amount threshold on chairs that is "150.00€", an amount limit of "500.00€", and additionally an amount interval of "200.00€" (*ThresholdType* = AMTI).
    - Both line items trigger the merchandise category eligibility.
    - The amount threshold of 150.00€ is reached since three items of the merchandise category are part of the transaction and have an amount of 279.85€ together.
    - The amount limit is not reached.
    - The amount interval is 200.00€, thus the customer gets the benefit for 150.00€ (the threshold "150.00€" is less than the regular total amount of 279.85€ and is less than threshold plus interval 350€, thus the threshold is only considered for the computation).
    - The PCE chooses the line items that get the benefit (see Choose Items):
        - Since the benefit is the same in any case, the customer gets a benefit for 1.501 office chairs. The quantity 1.501 times the regular price of one office chair results in the amount threshold.

The customer gets no additional benefit for the third chair even though the amount limit is not reached. The reason is that the interval is "200.00€". Thus, the customer would get an additional benefit for an amount of 350.00€ and for 500.00€ (the threshold "150.00€" plus a multiple of the interval – 2 times "200.00" is 550.00€ – exceeds the limit, so that the limit of "500.00€" is used instead). If the customer buys chairs amounting to 400.00€, the benefit would be the same as for 350.00€. If the customer buys chairs amounting to more than 500.00€, the benefit will not increase any further. This means that the customer would get a benefit as if he had bought chairs amounting to 500.00€.

In the following, the result for a simple discount of 4% is calculated. n is the number of office chairs and Y€ is the amount of office chairs the customer buys. The customer does not buy any kitchen chair.

1. Y€ < 150€: The discount is not applied for one office chair since the threshold is not reached (total discount = 0.00€).

2. 150€ <= Y€ <= 500€: The discount is applied for 2, 3, 4 and 5 office chairs. See the Solution table:

n	Y€	Applied discount
2	199.90€	150€ are considered.

Line Item	Quantity	Considered Amount	Corresponding Quantity	Resulting Discount Amount
1	1	150€ - 99.95€ = 50.05€	50.05€ / 99.95€ = 0.501	50.05€ * 4% = 2€
2	1	99.95€	1	99.95€ * 4% = 4€
Total		150€	1.501	6€

3	299.85€	150€ are considered.

Line Item	Quantity	Considered Amount	Corresponding Quantity	Resulting Discount Amount
1	1	0	0	0€
2	1	150€ - 99.95€ = 50.05€	50.05€ / 99.95€ = 0.501	50.05€ * 4% = 2€
3	1	99.95€	1	99.95€ * 4% = 4€
Total		150€	1.501	6€

4	399.80€	150€ + 200€ = 350€ are considered.

Line Item	Quantity	Considered Amount	Corresponding Quantity	Resulting Discount Amount
1	1	350€ - 3 * 99.95€ = 50.15€	50.15€ / 99.95€ = 0.502	50.15€ * 4% = 2.01€
2	1	99.95€	1	99.95€ * 4% = 4€
3	1	99.95€	1	99.95€ * 4% = 4€
4	1	99.95€	1	99.95€ * 4% = 4€
Total		350€	3.502	14.01€

5	499.75€	150€ + 200€ = 350€ are considered.

Line Item	Quantity	Considered Amount	Corresponding Quantity	Resulting Discount Amount
1	1	0€	0	0€
2	1	350€ - 3 * 99.95€ = 50.15€	50.15€ / 99.95€ = 0.502	50.15€ * 4% = 2.01€
3	1	99.95€	1	99.95€ * 4% = 4€
4	1	99.95€	1	99.95€ * 4% = 4€
5	1	99.95€	1	99.95€ * 4% = 4€
Total		350€	3.502	14.01€

3. Y€ > 500€: the discount is applied to 3.502 pieces of office chairs which results in a total discount of 14.01€.

5.5.5 Use Thresholds and Limits for the Amount and the Quantity

To use thresholds and limits for the amount and the quantity, the threshold type has to be *AMQU*. This threshold type is triggered by amounts of items and quantities that are part of the possible triggers. Possible triggers are: items, merchandise categories, simple product groups, and product groups. An amount threshold, an amount limit, and a quantity threshold as well as a quantity limit are set in the price derivation rule eligibility. The transaction is compared with the price derivation rule eligibility. To activate the price derivation rule eligibility, the transaction has to include the trigger with the correct quantities and amounts. Thereby, the corresponding amount of the calculation base amount of the promotion price derivation rule (possibly reduced by the granted monetary discount) is checked against these limits instead of the original amount.

In the case of *AMQU*, both the amount and the quantity threshold value as well as the amount and quantity limit are checked. The check is made similar to *QUT* and *AMT*. *AMQU* is not an interval related threshold type, the interval amount and quantity will not be considered.

Note, that no threshold type exist which considers both quantity and amount related values and is interval related.

Example (cont.)
Let us assume there is a promotion price derivation rule that contains an item eligibility for the item "cup" with the threshold type AMQU, whit the following values set:

- ThresholdTypeCode = AMQU
- ThresholdQuantity = 2.0
- ThresholdAmount = 4.0$

- LimitQuantity = 3.0
- LimitAmount = 8.0$

Then the following behavior can be expected:

Scenario	Description	Behavior
1	Buy 1 piece of item "cup" for a unit price of 4.0$.	✗ Quantity thershold is not reached, promotion will not be applied.
2	Buy 2 pieces of item "cup" for a unit price of 1.0$.	✗ Amount thershold is not reached, promotion will not be applied.
3	Buy 4 pieces of item "cup" for a unit price of 1.5$.	✓ Qunatity limit was reached, promotion can be applied on **3 pieces** of item "cup" (which still reaches both thresholds)
4	Buy 3 pieces of item "cup" for a unit price of 3.2$	✓ Amount limit was reached, promotion can be applied on **2.5 pieces** of item "cup" (which still reaches both thresholds)

5.5.6 Not Applied Discount on Interval-Related Threshold Types

Note that the following behavior may occur when calculating a promotion which contains an eligibility with threshold type *QUTI* or *AMTI* and an interval > 0 with respect to the input transaction:

If the calculation of a price derivation rule for an interval results in an invalid result (e.g. price increase, negative price, zero discount but zero discounts are not allowed by configuration), the promotion price derivation cannot be applied on this interval and the PCE proceeds with the following intervals.

Example Zero Discount

A promotion price derivation rule P1 contains a merchandise category eligibility for merchandise category "sports apparel" with threshold type QUTI and an interval of 3. There also exists a corresponding price derivation rule R1 which sets the discount sales price of the summed up sales prices to 30€ (*FIX_PRICE_TOTAL/"PT"*).

The PCE is configured to regard the item with the lowest price first when calculating the promotion (*itemChooseMethod=LOWEST_FIRST*). Zero discounts are not allowed (*allowZeroRebate=false*).

We assume that the customer buys three caps with a regular sales price of 10€ each and four running shirts with a regular sales price of 15€ each.

Thus, the transaction contains the following line items:

1. First line item: item identifier "cap", regular price 10€, quantity "3.0", merchandise category "sports apparel"
2. Second line item: item identifier "shirt", regular price 15€, quantity "4.0", merchandise category "sports apparel"

The PCE executes the calculation as follows:
As the cheapest item is regarded first, the "cap" line item triggers promotion P1 for the first interval. Then the price derivation rule R1 is applied and the result is a new price of 30€. Since the total of the three caps is also 30€, the discount will be zero. As zero discounts are not allowed, the corresponding price derivation rule is not applied for the first interval and the PCE proceeds with the following interval. The "shirt" line item is triggering the promotion price derivation rule P1 with three of its quantity and is selected for the next interval. The discount is 15€ for the "shirt" line item. As result, the promotion price derivation rule applies only on the second line item with item identifier "shirt".

The general recommendation to avoid cases where intervals can be skipped is to set the system parameter *itemChooseMethod* and/or the *ChooseItemMethode* to *HIGHEST_FIRST.* More information about these parameters can be found in the chapters Choose Items and Configuration.

5.5.7 Interval-Based Transaction-Related Discounts

In the case of interval-based transaction-related discounts, PCE calculates and checks the intervals and sets according to the fulfilled intervals the Applied Count in the response. Depending on the transaction rebate method (TransactionRebateCalculationMethodCode), the discount is applied to all eligible single items (TRIGGER) or to all items of the entire transaction (TOTAL) regardless of the *threshold type*. Even in the case of an interval-related promotion price derivation rule, the discount is not applied per interval, but only applied once to the single eligible items (TRIGGER) and prorated between them or applied to all items in the transaction (TOTAL) and prorated between all items. For additional information about the *calculation base amount* see chapter Set the Calculation Base Amount.

Example: Interval-Based Transaction-Related Benefits

Let us assume there are the following items in the basket:

- Shirt (MHG "clothes"): quantity is 10 and unit price is 10.00€
- Sandals (MHG "shoes"): quantity is 10 and unit price is 10.00€

We assume that there is the following transaction-related promotion price derivation rule in the promotion master data:
Buy 5 items out of MHG "clothes" and get 5.00€ discount on the transaction. *Threshold eligibility* QUTI with *threshold quantity* 5, *interval* 5 and *limit* 99999. Apply a total discount of 5.00€.

Scene A: The PCE system parameter *transaction rebate method* is set to TRIGGER
Result:
The *interval* is fulfilled twice with 10 items of the MHG "clothes" and *Applied Count* is set to 2.

The discount of 5.00€ is applied once to the trigger items of MHG "clothes" and distributed over the 10 eligible items.
The total discount for the transaction is 5.00€, meaning every single item "Shirt" gets a prorated discount of 0.50€.

Scene B: The PCE system parameter *transaction rebate method* is set to TOTAL
Result:
The *interval* is fulfilled twice with 10 items of the MHG "clothes" and *Applied Count* is set to 2.
The discount of 5.00€ is applied once to the total basket and distributed over all 20 items.
The total discount for the transaction is 5.00€, meaning every single item in the basket gets a prorated discount of 0.25€.

5.5.8 Use Threshold Types for Single Items

If the threshold for single item flag (*MHGPromotionConditionEligibilitySO. thresholdForSingleItemFlag* or *MerchandiseSetPromotionConditionEligibilitySO.isMixingForbidden*) is set to false in a merchandise category or a product group, the quantity and/or the amount of all eligible items in the transaction are summarized and checked against the quantities/amounts indicated in the price derivation rule eligibility. Otherwise, the quantity and/or amount is summarized individually for each line item that triggers the corresponding price derivation rule eligibility. The result is checked against the quantities and/or amounts stated in this threshold eligibility.

The threshold for single item flag <u>is not</u> checked during eligibility loading and activation. Instead, the PCE takes care to only apply the promotion on the same line item during the promotion calculation.

Example
Let us assume that there are the items "red shirt" for 10 $ and "blue shirt" for 15 $, both belong to the merchandise category with the identifier "shirts".
Let us further assume that a promotion price derivation rule exists in the master data, which grants 10% discount for buying at least 3 items of the merchandise category "shirts"

- Threshold type AMT with threshold = 3
- The chooseItemMethod is set to HIGHEST_FIRST

<u>1.Scenario</u>
Let us assume, that the threshold for single item flag is set to false.
A customer buys 3 items of "red shirt" and 2 items of "blue shirt".
As a result, the PCE applies the promotion price derivation rule on all items, because the threshold of 3 items from the merchandise category is reached.

threshold for single item flag	applied	
false	3 x "red shirt" ✓	2 x "blue shirt" ✓

2.Scenario
Let us assume, that the threshold for single item flag is set to true.
A customer buys 3 items of "red shirt" and 2 items of "blue shirt".
As a result the promotion price derivation rule is appllied only on the 3 items of "red shirt", because it has to be the same line item.

threshold for single item flag	applied	
true	3 x "red shirt" ✓	2 x "blue shirt" ✗

3.Scenario
Let us assume, that the threshold for single item flag is set to true.
A customer buys 3 items of "red shirt" and 3 items of "blue shirt".
As a result the promotion price derivation rule is appllied only on the 3 items of "blue shirt", because the chooseItemMethod is set to HIGHEST_FIRST and the price of the item "blue shirt" is higher.

threshold for single item flag	applied	
true	3 x "red shirt" ✗	3 x "blue shirt" ✓

THERE IS MORE

The normalized calculation of every transaction is to be considered during the usage of thresholds, intervals, and limits. Since the normalized calculation rounds the benefit for each normalized line item individually (line item with quantity 1), the rounding difference is added for each additional quantity the promotion price derivation rule is applied to. This behavior results in a greater rounding difference the more quantities of line items get a benefit. In relation with thresholds, intervals, and limits this means that the greater the values are chosen the greater the rounding differences are – especially for quantity thresholds, quantity intervals, and quantity limits. For more information, refer to chapter Calculate the Benefit with Normalized Quantities.

The amount of an item for triggering a threshold type is typically the regular sales price of the line item times the quantity and minus the already applied monetary discounts. This base for the calculation can be changed with the attribute calculation base sequence

(*PromotionConditionRuleSO.calculationBaseSequence*) and is further explained in chapter Set the Calculation Base Amount. As a result, the amount can be:

- the regular sales price times the quantity
- the regular sales price times the quantity minus a part of the already applied monetary discounts
- the regular sales price times the quantity minus all already applied monetary discounts

There are certain combination eligibilities where the threshold for single item flag is restricted. This means that for some combinations of merchandise category eligibilities the flag has to be equivalent for all eligibilities that are part of the combination. Refer to chapter Combine Triggers for details.

There exists also an additional threshold type for simple product groups – that is *COMB*. The details are described in chapter Use a Simple Product Group as a Trigger.

In case a manual eligibility (*ManualPromotionConditionEligibilitySO*) is combined with a threshold eligibility, the manual trigger of the transaction is consumed for each fulfilled interval. There need to be two manual triggers in the transaction in order to apply a promotion price derivation rule for two fulfilled intervals of the threshold eligibility. Does there only exist one manual trigger in the transaction, the promotion price derivation rule is only applied once even though the interval of the threshold eligibility is fulfilled twice.

The threshold type and the threshold for single item flag define which quantities and/or amounts are triggering the price derivation rule eligibility. Further information about the attributes can be found in chapter Configuration.

5.6 Use the Total of a Shopping Basket as a Trigger

The regular/discount total amount of a shopping basket can be used to trigger a promotion price derivation rule. For this purpose, the price derivation rule eligibility contains a special threshold for the total amount of the shopping basket. This type of price derivation rule eligibility is herein called shopping basket eligibility (*MarketBasketAmountEligibilitySO*).

HOW IT WORKS

The shopping basket threshold (*MarketBasketAmountEligibilitySO.marketBasketThresholdAmount*) for the regular/discount total amount of a shopping basket is the minimal total amount a transaction has to contain to trigger a promotion price derivation rule. This value is part of a price derivation rule eligibility of the promotion master data. It can contain an arbitrary positive amount.

The PCE checks the regular/discount total amount of the shopping basket of the transaction. For this purpose, the amounts of all items are summed up and then compared with the shopping basket threshold. If the total amount is greater than or equal to it, the corresponding price derivation rule is applied. Thereby, the PCE calculates the discount total amount of the shopping basket based on the present information.

Example: Use the Total of a Shopping Cart as a Trigger
Let us assume that there is an item "Integrated Circuit" for a regular price of 1.99€ per piece.
We further assume that there is a promotion price derivation rule with a shopping basket eligibility that contains the shopping basket threshold of 5000.00€.
If a customer buys 3000 integrated circuits for the production of 3000 boards, the transaction has a regular total amount of 5970.00€. The shopping basket threshold is less than the regular total amount of the transaction and thus the corresponding price derivation rule is applied to it.
If a customer buys only 2000 integrated circuits, the regular total amount would be 3980.00€. The shopping basket threshold is greater than the regular total amount of the transaction and thus the corresponding price derivation rule is not applied.

THERE IS MORE

The shopping basket eligibility cannot be combined with line item-related promotion price derivation rules (*PromotionConditionRuleSO.transactionControlBreakCode* = "PO") if no additional type of price derivation rule eligibility is combined with it. If no additional price derivation rule eligibility exists, the shopping basket promotion price derivation rule is not applied.

For more information about the shopping basket threshold, see chapter Configuration.

5.7 Use a Customer Group as a Trigger

The price derivation rule eligibility can be comprised of several trigger types. One of these trigger types is the customer group. If a transaction contains a customer group (*RetailTransactionCustomerGroupAssignment*), the PCE applies the promotion price derivation rule stating the particular customer group. This price derivation rule eligibility is herein called customer group eligibility (*CustomerGroupPromotionConditionEligibilitySO*).

HOW IT WORKS

A customer group is defined by its customer group identifier (*CustomerGroupPromotionConditionEligibilitySO.customerGroupID*).

To apply a benefit triggered by a customer group, the customer group eligibility has to be part of the promotion price derivation rule. One customer group eligibility contains one customer group identifier.

A customer group eligibility is activated if the customer group identifier in the price derivation rule eligibility is part of the transaction. For this purpose, the customer group identifier is compared with the data of the loaded price derivation rule eligibilities. If the customer group eligibility is active, the corresponding price derivation rule is executed in order to apply the benefit.

THERE IS MORE

The customer group eligibility can be combined with line item-related promotion price derivation rules (*PromotionConditionRuleSO.transactionControlBreakCode* = "PO") if no additional type of price derivation rule eligibility is combined with it. If no additional price derivation rule eligibility exists, the price derivation rule is applied and considers all line items in the transaction.

More customer group eligibilities can be combined with each other to build a group of several customer groups. Refer to chapter Combine Triggers for details.

Note that the customer and the employee group as distinct trigger type are out of scope. These can be treated as customer groups.

5.8 Use a Coupon as a Trigger

The type of a price derivation rule eligibility can be coupon. If a transaction contains a coupon, the PCE applies the promotion price derivation rule stating the particular coupon. This price derivation rule eligibility is herein called coupon eligibility *(CouponPromotionConditionEligibilitySO).*

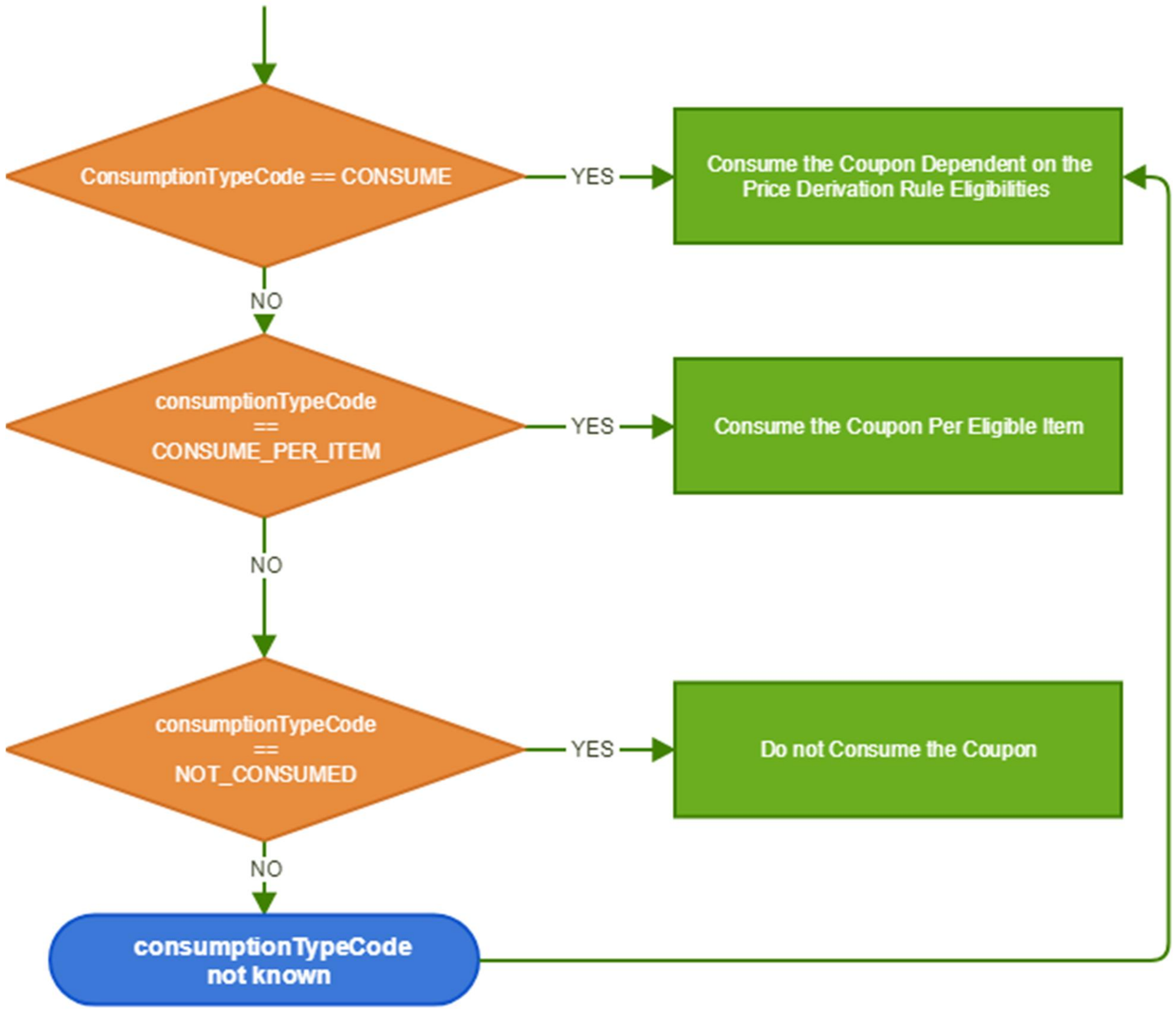
HOW IT WORKS

A coupon is defined by its coupon identifier (*CouponPromotionConditionEligibilitySO.couponNumber*). In addition, it has a consumption type *(CouponPromotionConditionEligibilitySO.consumptionTypeCode).*

Based on the consumption type, a coupon of the transaction (*RetailTransactionCouponSummary.Key.couponNumber*) is consumed or not. The attribute can be set in the promotion master data with the following values:

- *CONSUME/"00"*/null: The consumption of coupons depends on the price derivation rule eligibilities within the promotion price derivation rule.
- *CONSUME_PER_ITEM/"01"*: The coupon is consumed per eligible item.
- *NOT_CONSUMED/"02"*: The coupon is not consumed.

The different values and their meaning are roughly illustrated in the following decision tree:


ConsumptionTypeCode == CONSUME
YES
Consume the Coupon Dependent on the Price Derivation Rule Eligibilities
NO
consumptionTypeCode == CONSUME_PER_ITEM
YES
Consume the Coupon Per Eligible Item
NO
consumptionTypeCode == NOT_CONSUMED
YES
Do not Consume the Coupon
NO
consumptionTypeCode not known

The PCE consumes the coupons that are part of the transaction according to the consumption type. Thereby, the following outcomes are possible:

1. Consume the coupon dependent on the price derivation rule eligibilities.
2. Consume the coupon per eligible item.
3. Do not consume the coupon.

In addition, the PCE counts the quantity of coupons that were used/consumed during the calculation of the benefit. It is stored in the applied count of the coupon (*RetailTransactionCouponSummary.appliedCount*) in the transaction. The applied count cannot be greater than the input count of the coupons (*RetailTransactionCouponSummary.inputCount*) in the transaction. Thereby, the input count represents the quantity of registered coupons during a purchase.

5.8.1 Consume the Coupon Dependent on the Price Derivation Rule Eligibilities

The consumption type is *CONSUME*. This means that the consumption depends on the price derivation rule eligibilities included in the promotion price derivation rule. This results in an applied count that is increased by one each time the promotion price derivation rule is applied. If a threshold type is part of a price derivation rule eligibility of the same promotion price derivation rule, the applied count is increased dependent on the corresponding threshold type:

- quantity/amount threshold and quantity/amount limit are set (threshold type is *QUT/AMT*): A coupon is consumed once, thus the applied count is one.
- quantity/amount threshold, quantity/amount interval, and quantity/amount limit are set (threshold type is *QUTI/AMTI)*: A coupon is consumed once for each fulfilled interval, thus the applied count is the number of fulfilled intervals.

This is true if the transaction triggers a coupon eligibility and an item, merchandise category, simple product group, product group and/or combination eligibility. In all other cases, the coupon is consumed once per application of the promotion price derivation rule.

Example: Consume the Coupon Dependent on the Price Derivation Rule Eligibilities
Let us assume that there is the following item in the master data:

- Vase: 10.10€

We also assume that there is the following promotion price derivation rule in the promotion master data:

1. Get for every 2 vases 0.20€ discount for each vase if a coupon #1 is registered. The consumption type is CONSUME.

The customer buys 5 vases and one coupon #1 is registered. The PCE calculates a discount of 0.40€ since one coupon #1 is part of the transaction. The quantity interval (*ThresholdPromotionConditionEligibility.intervalQuantity*) is fulfilled two times but the input count of the coupon is 1 and thus, the promotion price derivation rule can only be once applied.
If two coupons #1 are registered, the PCE calculates a discount of 0.80€. The reason is that the quantity interval is fulfilled two times and the input count of the coupon is 2. As a result the promotion price derivation rule can be applied two times.
If three coupons #1 are registered, the PCE calculates a discount of 0.80€ since the quantity interval is only fulfilled two times.

5.8.2 Consume the Coupon per Eligible Item

The consumption type is *CONSUME_PER_ITEM*. The coupon is thus consumed for each eligible normalized line item in the transaction. Thereby, the applied count is equal to the quantity of the eligible line items.

Example: Consume the Coupon
Let us assume that there is the following item in the master data:

- Vase: 10.10€

We also assume that there is the following promotion price derivation rule in the promotion master data:

1. Get for every 2 vases 0.20€ discount for each vase if coupon #2 is registered. The consumption type is CONSUME_PER_ITEM.

The customer buys 5 vases and one coupon #2 is registered. The PCE does not apply the promotion price derivation rule since the consumption type is CONSUME_PER_ITEM and thus, at least 2 coupons #2 are needed to apply once the promotion price derivation rule.

If two coupons #2 are registered, the PCE calculates a discount of 0.40€. The reason is that the input count of the coupon is 2. As a result the promotion price derivation rule can be applied once.

If three coupons #2 are registered, the PCE calculates a discount of 0.40€ since the quantity interval is fulfilled two times but the input count of the coupon #2 is only 3. The consumption type CONSUME_PER_ITEM indicates that the input count of the coupon #2 needs to be 4 to apply the promotion price derivation rule twice.

5.8.3 Do not Consume the Coupon

The consumption type is *NOT_CONSUMED*. The coupon is consumed per transaction. It does not depend on how many promotion price derivation rules are triggered by the same coupon. This means, if one coupon is part of a transaction and triggers two promotion price derivation rules, the applied count is one.

Example: Do not Consume the Coupon

Let us assume that there is the following item in the master data:

- Vase: 10.10€

We also assume that there is the following promotion price derivation rule in the promotion master data:

1. Get for every 2 vases 0.20€ discount for each vase if coupon #3 is registered. The consumption type is NOT_CONSUMED.

The customer buys 5 vases and one coupon #3 is registered. The PCE calculates a discount of 0.80€ since the quantity interval is fulfilled two times and the consumption type is NOT_CONSUMED. This discount will not increase with the input count of the coupons #3 within the transaction.

If the customer buys 10 vases and one coupon #3 is part of the transaction. The PCE calculates a discount of 2.00€ since the quantity interval is fulfilled 5 times and the coupon #3 is once consumed per transaction.

5.8.4 Fall Back to the Default Consumption

If the consumption type is unknown or not set in the price derivation rule eligibility, the PCE assumes that the consumption type is *CONSUME.*

5.8.5 THERE IS MORE

The coupon eligibility can be combined with line item-related promotion price derivation rules (*PromotionConditionRuleSO.transactionControlBreakCode* = "*PO*") if no additional kind of price derivation rule eligibility is combined with it. If no additional price derivation rule eligibility exists, the price derivation rule is applied and considers all line items in the transaction in accordance to the consumption type.

It is also possible to grant a coupon as benefit. This should not be mixed up with the use case described in this chapter. For more details, see chapter Select the Method to Discount.

More information about the consumption type can be found in chapter Configuration.

5.9 Combine Triggers

Price derivation rule eligibilities can be combined with each other by different operators. Thereby, a parent price derivation rule eligibility – short parent eligibility – can have several child price derivation rule eligibilities – short child eligibilities (*CombinationPromotionConditionEligibilitySO.childEligibilityList*). This group of price derivation rule eligibilities is herein called combination eligibility (*CombinationPromotionConditionEligibilitySO*) – child and parent eligibilities.

HOW IT WORKS

A combination eligibility is composed of one parent and one child eligibility at least. The child eligibility refers to the parent eligibility and contains one of the following trigger types or their combination:

- Item
- Merchandise category
- Product group
- Shopping basket
- Customer group
- Coupon
- Manual trigger

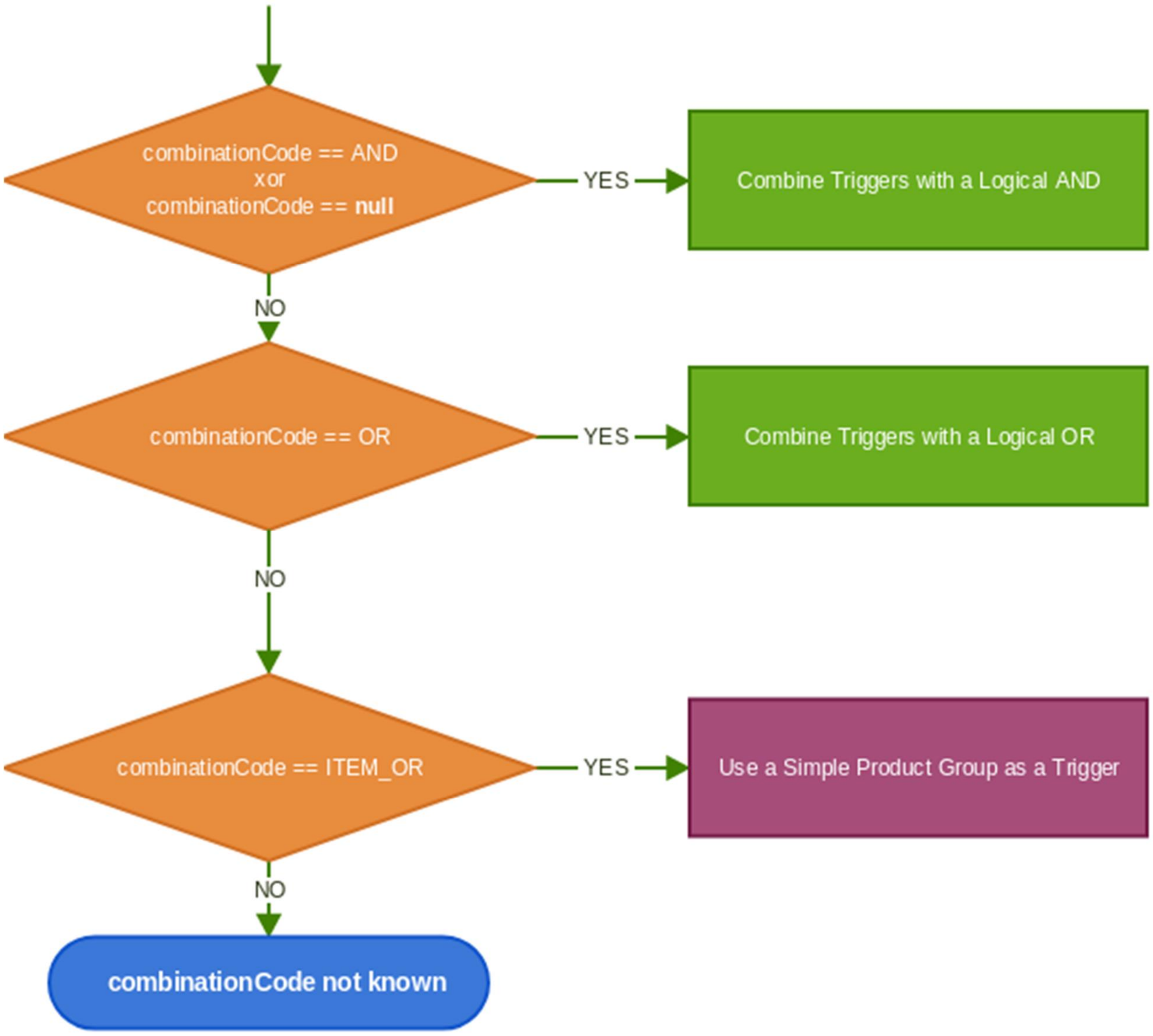
In addition, a threshold type can be part of the parent eligibility. In this case, this threshold type is ignored by the PCE. The only exception of this practice is the simple product group.

The combination code (*CombinationPromotionConditionEligibilitySO.combinationCode*) indicates how the price derivation rule eligibilities are combined with each other. The attribute can be set in the promotion master data of the parent eligibility with the following values:

- *AND:* combination via a logical AND operator
- *OR*: combination via a logical OR operator
- *ITEM_OR:* see Use a Simple Product Group as a Trigger.

If the combination code is not part of the parent eligibility, the PCE behaves as if the combination code is *AND.*

The different attribute values and their meaning are roughly illustrated in the following decision tree:


combinationCode == AND
xor
combinationCode == null
YES
Combine Triggers with a Logical AND
NO
combinationCode == OR
YES
Combine Triggers with a Logical OR
NO
combinationCode == ITEM_OR
YES
Use a Simple Product Group as a Trigger
NO
combinationCode not known

When activating a combination eligibility, the following outcomes are possible:

1. Combine triggers with a logical AND
2. Combine triggers with a logical OR

5.9.1 Combine Triggers with a Logical AND

The transaction has to include all triggers of child eligibilities of a parent eligibility with a logical AND operator as combination code. If only one trigger type is not part of the transaction, the PCE does not activate the parent eligibility. Thus, no further calculations are executed.

A combination eligibility with a logical AND operator is not allowed if it includes several child merchandise category eligibilities (*MHGPromotionConditionEligibilitySO*) with the threshold for single item flag (*MHGPromotionConditionEligibilitySO.thresholdForSingleItemFlag*) set to true. The corresponding promotion price derivation rule cannot be applied.

5.9.2 Combine Triggers with a Logical OR

The transaction has to include at least one trigger of a child eligibility of a parent eligibility with a logical OR operator as combination code. If no trigger type is part of the transaction, the PCE does not activate the parent eligibility. Thus, no further calculations are executed.

A price derivation rule eligibility which includes a logical OR operator is internally disassembled into several price derivation rule eligibilities without a logical OR operator.

5.9.3 THERE IS MORE

A special case of combination eligibility is the simple product group eligibility that is described in chapter Use a Simple Product Group as a Trigger.

A combination eligibility including several merchandise category eligibilities with different values for threshold for single item flag is not allowed. The corresponding promotion price derivation rule cannot be applied.

The combination code is part of the price derivation rule eligibility. Further information can be found in chapter Configuration.

6 Promotion Calculation – General Behavior

The step promotion calculation computes the discounts and loyalty points for a certain transaction. This is the core functionality of the PCE. In this step, the promotion price derivation rules are loaded and validated and the price derivation rules are executed. Thereby, the promotion price derivation rules are only loaded and validated if the corresponding price derivation rule eligibility is activated.

There are several limiting factors for the calculation. Some of these factors can be influenced by certain configuration parameters and others are given by the process itself. In the following chapters, these limiting factors and thus the general behavior of the PCE during the promotion calculation are explained.

In this chapter, the following topics are discussed:

6.1 Handle Line Item- or Transaction-Related Promotion Price Derivation Rules

The execution of the price derivation rules is handled separately on line item level and transaction level. The transaction-related promotion price derivation rules are calculated after the line item-related ones. The transaction-related promotion price derivation rules thus always base on the line item-related promotion price derivation rules.

HOW IT WORKS

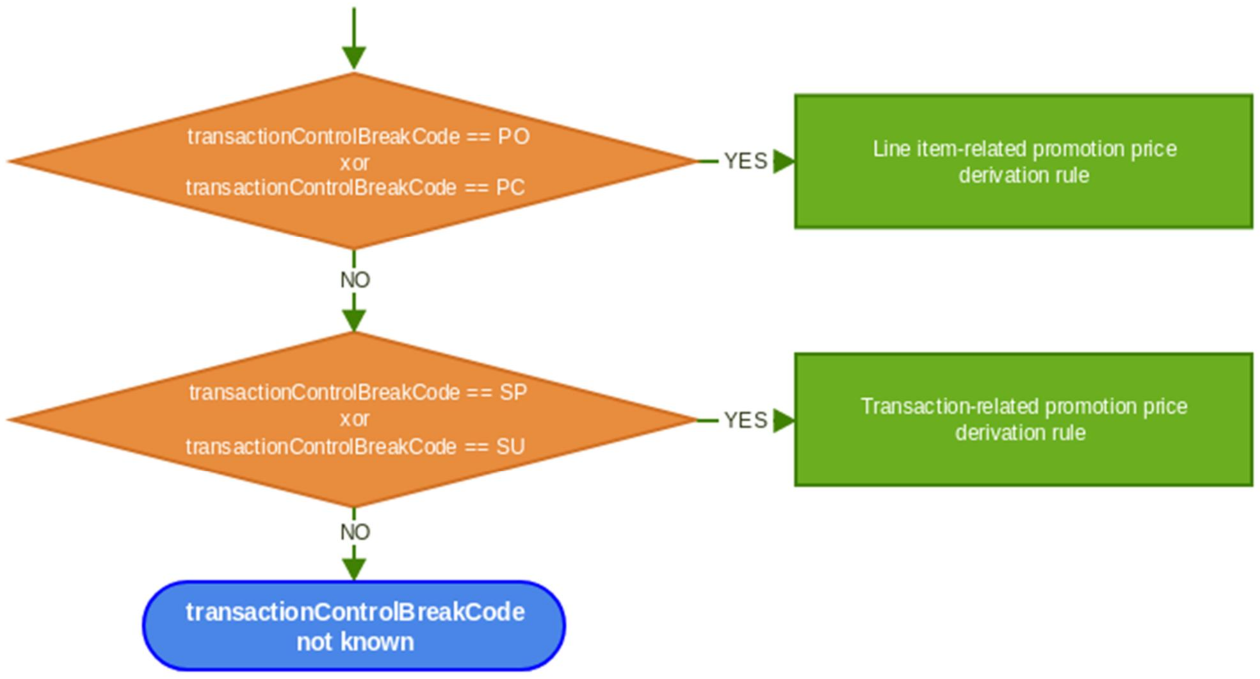
A transaction-related promotion price derivation rule calculates a benefit that is based on the regular/discount total amount of a transaction. A line-item-related promotion price derivation rule calculates a benefit that is based on the regular/discount sales price of eligible line items.

The attribute transaction control break code (*PromotionConditionRuleSO.transactionControlBreakCode*) specifies whether the promotion price derivation rule refers to

- One or several line items (transaction control break code is *PO/PC*) or
- The whole transaction (transaction control break code is *SU/SP*)

If the transaction control break code is null or unknown, *SU* (= transaction-related) is used by default.

The different attribute values and their meaning are roughly illustrated in the following decision tree.


transactionControlBreakCode == PO
xor
transactionControlBreakCode == PC
YES
Line item-related promotion price derivation rule
NO
transactionControlBreakCode == SP
xor
transactionControlBreakCode == SU
YES
Transaction-related promotion price derivation rule
NO
transactionControlBreakCode not known

As line item-related and transaction-related promotion price derivation rules are independent of each other, they do not compete for triggers. This means that a trigger which was consumed by a line item-related promotion price derivation rule is not available for another line item-related one with the same sequence (*PromotionConditionSO.sequence*). However, the trigger is still available for transaction-related promotion price derivation rules. The same line item (or part of the line item –

subset) may work as a trigger for several promotion price derivation rules with different sequence without consuming the corresponding trigger in the transaction.

Line item-related promotion price derivation rules with the transaction control break code either equal to *PO* or to *PC* are calculated after the PCE received the transaction and before transaction-related promotion price derivation rules. Transaction-related promotion price derivation rules with the transaction control break code either equal to *SP* or to *SU* are calculated after the PCE received the transaction.

Note that if the calculation mode of the PCE is set to LineItem, a transaction-related promotion price derivation rule with the transaction control break code *SU* or *SP* can only be applied to line-items, which fulfill the eligibility of the transaction-related promotion price derivation rule on their own, without the background of other line-items. To read more about calculation mode see Select a Calculation Mode.

THERE IS MORE

Further information about the attribute transaction control break code can be found in chapter Configuration.

Interval-based transaction-related discounts determine the items for the calculation base according to the system parameter *Transaction Rebate Method* differently than interval-based line item-related discounts, which do not consider this system parameter. For detailed information see Use Thresholds, Intervals, and Limits and Set the Calculation Base Amount.

6.2 Select a Promotion Price Derivation Rule in Case of a Collision

During the computation of the benefits for a certain transaction, different promotion price derivation rules can collide with each other. In case the colliding promotions also have a conflict, meaning that their calculation is not completely independent from each other, the PCE executes a conflict resolution known as "best price calculation". For selecting the set of promotion price derivation rules and their exact order in case of a conflict, it is possible to select one of two algorithms for the conflict resolution – the brute force algorithm and the greedy algorithm. To activate the brute force algorithm, the parameter *conflictHandlerStrategy* must have the value *BRUTE_FORCE.* To activate the greedy algorithm, the parameter must have the value *GREEDY*.

HOW IT WORKS

6.2.1 Calculation Order of Promotion Price Derivation Rules Based on Sequence and Resolution

If several promotion price derivation rules are applicable to the transaction, the PCE applies the promotion price derivation rules in a certain order. The order is defined based on the sequence (*PromotionConditionSO.sequence*) and the resolution (*PromotionConditionSO.resolution*) of the promotion price derivation rules. The PCE defines the order based on the following steps:

1. Ordering of the calculation of promotion price derivation rules based on the ascending sequence.
2. If the sequences of the promotion price derivation rules are equal, only the one with the highest resolution is applied to a single item. Following promotion price derivation rules (still same sequence, lower resolution) can only be applied to items not consumed this way.

The ordering of promotion price derivation rules is done separately for line item-related and transaction-related ones. The calculation is done first for line item-related promotion price derivation rules and subsequently for transaction-related ones.

Example 1: Ordering of line item-related and transaction-related promotion price derivation rules

The following tables illustrate cases of several promotion price derivation rules that are applicable and show the impact of sequence and resolution.

We assume that only one trigger item is available in the transaction and that all 4 promotion price derivation rules can be applied to this item.

🟩	line item-related promotions
🟦	transaction-related promotions

Case 1		
Promotion Price Derivation Rule	Sequence	Resolution
P1	0	0
P2	100	100
P3	100	100
P4	200	100

As a result, all promotions are applied in the order indicated in the table.

Case 2		
Promotion Price Derivation Rule	Sequence	Resolution
P1	200	0

P2	200	100
P3	200	200
P4	200	100

As a result, the following promotions are applied in the indicated order:

- P2 (as it has the higher resolution)
- P3 (as it has the higher resolution)

Example 2: Promotion applied only on a part of the line item

Let us assume that a retailer sells the item "green apple" and the item "banana". Both items belong to the merchandise hierarchy group "fruits".

Let us further assume that there are the two following line item-related promotion price derivation rules:

Name	Description	Sequence	Resolution	Threshold Type QUT for both	
				Threshold	**Limit**
promotion price derivation rule "apples"	Get 10% discount on maximum two items "green apple".	1	2	1	2
promotion price derivation rule "fruits"	Get 50% discount on every item of merchandise hierarchy group "fruits".	1	1	1	99999

We assume that the transaction of a customer looks like follows:

Line item	Quantity	Name
1	4	"green apple"
2	1	"banana"

The result would be as follows:

- Line item "green apple" triggers both promotion price derivation rules. Because of the higher resolution, the promotion price derivation rule "apple" is applied to two items "green apple" (because of limit=2).
- The promotion price derivation rule "fruits" is applied to the remaining two "green apples" and the item "banana".

6.2.2 Collision of Promotion Price Derivation Rules

A collision occurs when more than one applicable promotion price derivation rules have the same sequence and the same resolution. Since the ordering of promotion price derivation rules is done separately for line item-related and transaction-related promotion price derivation rules, no collisions are possible between a line item-related and a transaction-related promotion price derivation rule. Colliding promotion price derivation rules can be applied independently in any order, as long as they are not conflicting.

6.2.3 Conflict of Promotion Price Derivation Rules

A conflict occurs, when the colliding promotion price derivation rules fulfill one of the following criteria:

- The promotion price derivation rules intersect on their discounted items
- The promotion price derivation rules intersect on their matching normalized items for mix&match rules
- The promotion price derivation rules use the same coupons

If a promotion price derivation rule does not intersect in these areas with any other colliding promotion price derivation rules, then it is considered independent.

In case of a conflict, the order in which to apply the colliding promotion price derivation rules may influence the total discount granted to the customer. The PCE therefore executes the conflict resolution, trying to determine the optimal order of the conflicting promotion price derivation rules to maximize the granted total discount.

Warning: The conflict resolution is restricted by the values and execution algorithms of the promotion price derivation rules. Therefore, the conflict resolution can determine the highest possible total discount only with respect to these restrictions.

The conflict resolution applies each promotion price derivation rule according to the values and execution algorithm defined for the promotion price derivation rule, one after the other. Since the application of one promotion price derivation rule cannot be split up or interrupted, in some cases the PCE cannot find a possible better price. For example, the execution algorithm for interval-based promotion price derivation rules always applies as many intervals as possible based on the remaining triggering items. Applying only a part of the intervals, leaving some items for other promotion price derivation rules, might lead to an even better price, but would violate the specification of interval-based promotion price derivation rules (see chapter Use Thresholds, Intervals, and Limits for more information on intervals).

6.2.3.1 Calculation Time Limit

As mentioned above, the conflict resolution is based on trying out the optimal order of the conflicting promotion price derivation rules. In the process each promotion price derivation rule is applied several times. Depending on the number of conflicting promotion price derivation rules, the number of conflicting items and the conflict resolution algorithm, the conflict resolution can consume significant amounts of time. Therefore it is possible to limit the time for executing the conflict resolution algorithm to find the permutation of conflicting promotion price derivation rules with the best price. If there are multiple conflicts within a transaction, the time for each conflict resolution is restarted. The system parameter *calculationTimeLimit* defines the time limit in milliseconds for this calculation. The default value is 1000 milliseconds.

Warning: If the time limit is reached, the conflict resolution will be canceled and the permutation of promotion price derivation rules with the best price found so far will be applied. This might not match the possible best price.

Even though the parameter is configurable, a suggested minimum value should be taken into account, otherwise the calculation could run too fast into the time limit.

6.2.3.2 Condition Limit

The *condition limit* (conditionLimit) configuration parameter defines a limit for the number of applicable promotion price derivation rules having same *sequence* and *resolution* value. In case the number of applicable conflicting promotion price derivation rules exceeds this limit, a calculation limit exception is thrown before the best price calculation. Namely, if the *condition limit* is exceeded, best conflict resolution is not triggered. The default value of the *condition limit* is 100.

6.2.3.3 Select Promotion Price Derivation Rules in Case of a Conflict

The conflict resolution depends on the benefit type of the colliding promotion price derivation rules. Thus, the following scenarios are possible:

1. Select a promotion price derivation rule in case of multiple applicable monetary discounts.
2. Select a promotion price derivation rule in case of multiple applicable virtual discounts.
3. Select a promotion price derivation rule in case of multiple applicable loyalty points.
4. Select a promotion price derivation rule in case of multiple applicable benefits of different types.

For the calculation order, the PCE makes no distinction between promotion price derivation rules that apply a discount and those that apply loyalty points. They are treated equally, and all permutations are checked in case of a conflict. Only in case of equal reference values for different benefit types there exists a prioritization which is listed in the last possible outcome.

In addition, the computed effective sales price is compared with the regular sales price which results from applying the chosen promotion price derivation rule. If the regular sales price is smaller than the effective sales price – that is, a price increase – the promotion price derivation rule is not applied. This exception exists due to the best price principle.

6.2.3.4 Select a Promotion Price Derivation Rule in Case of Multiple Applicable Monetary Discounts

If all promotion price derivation rules apply a monetary discount, the following attributes have the following values:

- Bonus points flag (*PromotionConditionRuleSO.bonusPointsFlag*) is false.
- Discount method (*PromotionConditionRuleSO.discountMethodCode*) is either null, "00", or "03".

For computing the monetary discount, the following discounts are considered:

- Discount on line item level (*RetailPriceModifier.amount*)
- Discount on transaction level (*PriceModificationLineItem.amount*)

The promotion price derivation rule is selected that has the greatest discount.

6.2.3.5 Select a Promotion Price Derivation Rule in Case of Multiple Applicable Virtual Discounts

If all promotion price derivation rules apply a virtual discount, the following attributes have the following values:

- Bonus points flag is false.
- Discount method is not null, "00", or "03".

For computing the monetary discount, the following virtual discounts are considered:

- Virtual discount on line item level (*RetailPriceModifier.extraAmount*)
- Virtual discount on transaction level (*PriceModificationLineItem.extraAmount*)

The promotion price derivation rule is selected that has the greatest virtual discount.

6.2.3.6 Select a Promotion Price Derivation Rule in Case of Multiple Applicable Loyalty Points

If all promotion price derivation rules apply loyalty points, the following attribute has the following value:

- Bonus points flag is true.

For computing the loyalty points, the following loyalty points are considered:

- Loyalty points on line item level (*FrequentShopperPointsModifier.frequentShopperPointsEarnedAmount*)
- Loyalty points on transaction level (*LoyaltyRewardLineItem.frequentShopperPointsEarnedAmount*)

The promotion price derivation rule is selected that has the greatest value of loyalty points.

6.2.3.7 Select a Promotion Price Derivation Rule in Case of Multiple Applicable Benefits of Different Types

If the colliding promotion price derivation rules would apply different benefit types, the possible discounts and loyalty points are still compared with each other. For this purpose, the loyalty points are transformed into a value that is comparable with discounts. The result is then compared with the virtual discount/discount of the other modifiers. The PCE applies the promotion price derivation rule that has the greatest reference value at the end.

In case of equal reference values for different benefit types, the promotion price derivation rule is selected according to the following prioritization:

1. Monetary discount
2. Virtual discount
3. Loyalty points

Note for all four possible scenarios, if the benefit type is the same and the discount value is equal, the promotion price derivation rule with the smallest *InternalEligibilityID* will be selected.

6.2.4 THERE IS MORE

A detailed description of the brute force algorithm can be found here Conflict Resolution with the Brute Force Algorithm. For an explanation of the greedy algorithm, have a look at Conflict Resolution with the Greedy Algorithm. For a comparison of the different best price algorithms, have a look at Performance Comparison Between the Brute Force and the Greedy Algorithm.

6.2.4.1 Workaround for Scale Discount

For the creation of scale discounts, the general recommendation is to use the same sequence and a different resolution. If this behavior does not match the requirements, there is more help in the Promotion Calculation Engine SDK documentation.

Whether different resolutions can be used for scale discounts, depends on the other existing promotions. In case the client would like to define a competing promotion, independent from the promotions forming the scale promotion (for example, general customer group discount), all of these promotions must have the same sequence and resolution. Otherwise only the highest resolution is considered and the “Best Price Calculation” will not be used as expected. However, in the case of competing promotions with the scale discount, worse performance can be expected due to the “Best Price Calculation”, as well as reaching the calculation time limit in case of a high number of promotions participating in the scale discount.

6.2.5 Conflict Resolution with the Brute Force Algorithm

As one option for selecting the set of promotion price derivation rules and their exact order in case of a conflict, the PCE can use the brute force algorithm.

HOW IT WORKS

The brute force algorithm evaluates all combinations of colliding promotion price derivation rules one by one until the calculation time exceeds the parameter *calculationTimeLimit* (or is blocked by any other calculation limiter that can be integrated into the PCE in the future). The evaluation of each permutation of promotion price derivation rules involves recalculation of the promotion price derivation rules of the combination until the recalculation of one of the promotion price derivation rules fails. If none of the recalculations of the promotion price derivation rules of the combination fail, this combination of promotion price derivation rules is a candidate for the set of price derivation rules giving the best discount. The discount total amount granted by the promotion price derivation rules permutation is compared to find the highest benefit possible according to the algorithm.

The recalculation of a promotion price derivation rule by the PCE includes the decision of the triggering of the corresponding price derivation rule eligibilities for the transaction. It is a time-consuming operation when the overall execution time for the PCE for processing a given transaction is considered. The brute force algorithm for conflict resolution involves recalculation of promotion price derivation rules during the evaluation of each permutation of promotion price derivation rules. One promotion price derivation rule can be recalculated more than once also in the evaluation of different combinations.

The implementation of the brute force algorithm contains an optimization in order to eliminate the evaluation of all possible combinations of the colliding promotion price derivation rules. During the execution of the algorithm, if a combination x is not applicable as a whole (because there are not enough line items in the transaction), the other combinations containing all the promotion price derivation rules contained by the combination x are also not evaluated. Namely, if a combination is not applicable, the generation of further combinations from that combination is stopped.

An example illustrating the calculation of the brute force algorithm is given below:

Example: Calculation of Best Price by Brute Force Algorithm

Transaction and Promotions:

Let the transaction contain the following 3 line items:

- Item A with quantity 3 and the regular sales unit price 20€
- Item B with quantity 2 and the regular sales unit price 10€
- Item C with quantity 2 and the regular sales unit price 5€

Assume that we have the following four promotions. Each one containing a single promotion price derivation rule as indicated. All four promotions are not interval-related and can be applied once exactly on the number of items defined below.

- Promotion P1: Buy 3 item A and 1 item B with 9% discount
- Promotion P2: Buy 2 item A and 1 item B with 10% discount
- Promotion P3: Buy 2 item A, 1 item B, and 1 item C with 10% discount
- Promotion P4: Buy 1 item A, 1 item B, and 1 item C with 20% discount

The promotion price derivation rules for the above promotions have the same sequence and resolution values.

Calculation:

Each of the promotion price derivation rules above is applicable individually to the transaction. Since they have the same sequence and resolution values, the conflict resolution of the PCE is triggered. The brute force implementation calculates the discount generated by all the combinations of the four promotion price derivation rules as below (the promotion price derivation rules are indicated by their promotion identifier for the sake of simplicity):

Possible combinations				Calculation Result
P1	P2	P3	P4	Applied promotion: **P1** No item A is left after applying P1, the other 3 promotions cannot be applied after it. Discount = (20 * 0.09) * 3 + (10 * 0.09) = **6.3€**
		P4	P3	
	P3	P2	P4	
		P4	P2	
	P4	P2	P3	
		P3	P2	
P2	P1	P3	P4	Applied promotion: **P2** Not enough item A remains to apply P1 or P3 after P2. Discount = (20 * 0.1) * 2 + (10 * 0.1) = **5.0€**
		P4	P3	
	P3	P1	P4	
		P4	P1	
	P4	P1	P3	Applied promotions: **P2, P4** No item A is left after applying P2 and P4, the other 2 promotions cannot be applied after them. Discount = (20 * 0.1) * 2 + (10 * 0.1) + (20 * 0.2) + (10 * 0.2) + (5 * 0.2) = **12.0€**
		P3	P1	
P3	P1	P2	P4	Applied promotion: **P3** Not enough item A remains to apply P1 or P2 after P3. Discount = (20 * 0.1) * 2 + (10 * 0.1) + (5 * 0.1) = **5.5€**
		P4	P2	
	P2	P1	P4	
		P4	P1	
	P4	P1	P2	Applied promotions: **P3, P4** No item A is left after applying P3 and P4, the other 2 promotions cannot be applied after them. Discount = (20 * 0.1) * 2 + (10 * 0.1) + (5 * 0.1) + (20 * 0.2) + (10 * 0.2) + (5 * 0.2) = **12.5€** → **Best discount**
		P2	P1	
P4	P1	P2	P3	Applied promotion: **P4** Not enough item A remains to apply P1 after P4. Discount = (20 * 0.2) + (10 * 0.2) + (5 * 0.2) = **7.0€**
		P3	P2	

	P2	P1	P3	Applied promotions: **P4, P2** No item A is left after applying P4 and P2, the other 2 promotions cannot be applied after them. Discount = (20 * 0.2) + (10 * 0.2) + (5 * 0.2) + (20 * 0.1) * 2 + (10* 0.1) = **12.0€**
		P3	P1	
	P3	P1	P2	Applied promotions: **P4, P3** No item A is left after applying P4 and P3, the other 2 promotions cannot be applied after them. Discount = (20 * 0.2) + (10 * 0.2) + (5 * 0.2) + (20 * 0.1) * 2 + (10* 0.1) + (5* 0.1) = **12.5€** → Equals so far found best discount
		P2	P1	

The brute force algorithm results in 8 possible combinations of applicable colliding promotion price derivation rules in this example, compares the discounts generated by each combination, and generates a best discount of value 12.5€ with the combination (P3, P4).

6.2.5.1 Performance by Using the Brute Force Algorithm

When the number of conflicting promotion price derivation rules increases, the number of their combinations also increases. Therefore, the execution of the brute force algorithm starts to consume much more time. Also, since the brute force algorithm contains possibly a large number of recalculations of promotion price derivation rules when the number/quantity of line items in the transaction increases, the execution of the algorithm starts to consume much more time.

The calculation of *n* conflicting promotions can reach *n!* possible, applicable permutations at most, and in some scenarios this can lead to significantly increased calculation time.

Resolving into conflict resolution should be avoided wherever possible. It is only suitable for a very small number of colliding conditions. Otherwise, serious performance problems can occur, especially using the brute force algorithm!

The advantage of the brute force algorithm apparently comes into effect when the number of colliding promotion price derivation rules and line items in the transaction are small enough, so that the execution of the algorithm does not exceed the *calculationTimeLimit*. In these cases, all possible permutations are calculated to determine the highest possible total discount as a result, with respect to the restrictions defined by the values and execution algorithms of the promotion price derivation rules, of course.

6.2.5.2 THERE IS MORE

The *conflictHandlerStrategy* parameter for the PCE execution should have the value BRUTE_FORCE to activate the brute force algorithm for the best price calculation.

For an explanation of the other option for selecting the set of promotion price derivation rules and their exact order in case of a conflict, the greedy algorithm, have a look at Conflict Resolution with the Greedy Algorithm. For a comparison of the different best price algorithms, have a look at Performance Comparison Between the Brute Force and the Greedy Algorithm.

6.2.6 Conflict Resolution with the Greedy Algorithm

As another option for selecting the set of promotion price derivation rules and their exact order in case of a conflict, the PCE can use the greedy algorithm.

HOW IT WORKS

The greedy algorithm does not evaluate all combinations of colliding promotion price derivation rules one by one like the brute force algorithm. The PCE first sorts the colliding promotion price derivation rules according to the generated individual discounts. Then, the greedy algorithm fetches the promotion price derivation rule at the head of the sorted list and recalculates it. If the recalculation does not fail, the PCE adds the corresponding promotion price derivation rule to the ordered set of promotion price derivation rules that grants the best possible discount according to the greedy algorithm and restrictions by the promotion price derivation rules. Otherwise, this price derivation rule is discarded.

The greedy algorithm invokes one recalculation operation for each promotion price derivation rule during its execution. It prevents duplicate recalculations for a single price derivation rule, thereby it does not easily exceed the parameter *calculationTimeLimit* (at least for the default value). The greedy algorithm does not calculate every permutation, which means that it may not give the real best discount as a result in all cases. This is due to the greedy sorting approach that is followed and the lack of backtracking during the selection of promotion price derivation rules to recalculate.

An example illustrating the calculation of the greedy algorithm is given below:

Example: Conflict Resolution by Greedy Algorithm
Transaction and Promotions:
Let the transaction contain the following 3 line items:
Item A with quantity 3 and unit price 20€
Item B with quantity 2 and unit price 10€
Item C with quantity 2 and unit price 5€
Assume that we have the following four promotions, each containing a single promotion price derivation rule as indicated:
Promotion P1: Buy three item A and one item B with 9% discount.
Promotion P2: Buy two item A and one item B with 10% discount.
Promotion P3: Buy two item A, one item B and one item C with 10% discount.
Promotion P4: Buy one item A, one item B, and one item C with 20% discount.
The promotion price derivation rules for the above promotions have the same sequence and resolution values.
Calculation:
Each of the promotion price derivation rules above is individually applicable to the transaction. Since they have the same sequence and resolution values, the conflict resolution of the PCE is triggered. The greedy implementation first sorts the four promotion price derivation rules

according to the descending order of their individual discounts as below (the promotion price derivation rules are indicated by their promotion identifier for the sake of simplicity):

Promotion Price Derivation Rule	Discount
P4	(20 + 10 + 5) * 0.2 = 7€
P1	(3*20 + 10) * 0.09 = 6.3€
P3	(2*20 + 10 + 5) * 0.1 = 5.5€
P2	(2*20+10) * 0.1 = 5€

Then the greedy implementation selects (and removes) the first promotion, P4, from the head of the sorted promotion price derivation rules and calculates P4. The calculation of P4 rebates one item A, one item B, and one item C and gives 7€ discount. P4 is added to the set of promotion price derivation rules which gives the best discount and was initialized to the empty set. At this point, since the greedy implementation has not calculated the remaining promotion price derivation rules yet, it cannot know whether they are triggered for the remaining non-rebated line items or not.

Promotion Price Derivation Rule	Discount
P3	(2*20 + 10 + 5) * 0.1 = 5.5€
P2	(2*20+10) * 0.1 = 5€
P1	(2*20 + 10) * 0.09 = 4.5€

The greedy implementation selects (and removes) the first promotion price derivation rule from the list again, which is P3. It calculates P3 and it is triggered by the remaining non-rebated line items in the transaction (two item A, one item B, one item C). P3 is added to the set of promotion price derivation rules, which gives the best discount.
After triggering P3, there are no remaining non-rebated line items in the transaction and the greedy implementation stops. It returns the combination (P3, P4) as the combination of promotion price derivation rules that give the in this case best discount which is 7 + 5.5 = 12.5€ (same result as that of the brute force implementation for this example).

When the number of colliding promotion price derivation rules and/or the number/quantity of line items in the transaction increase, the execution of the greedy algorithm starts to take much time. But it is in almost all cases faster than the execution of the brute force algorithm. However, it may not give the real best discounts for specific scenarios. When these scenarios contain a small number of line items and colliding promotion price derivation rules, the execution of the brute force algorithm could generate the possible best discounts, based on restrictions defined by the promotion price derivation rule's values, without exceeding the *calculationTimeLimit*.

THERE IS MORE

The *conflictHandlerStrategy* configuration parameter for the PCE must be set to the value *GREEDY* in order to activate the greedy algorithm for the best price calculation.

A detailed description of the other option for selecting the set of promotion price derivation rules and their exact order in case of a conflict, the brute force algorithm, can be found here Conflict Resolution with the Brute Force Algorithm.

6.3 Check the Validity of a Promotion

Before the calculation of a benefit and after the price derivation rule eligibilities are activated, the PCE checks whether the promotion is valid. If the promotion is valid, the corresponding price derivation rule can be applied, otherwise not.

HOW IT WORKS

The validity of a promotion depends on several factors which are listed below:

- a validity period on promotion level
- a validity period on price derivation rule eligibility level
- interval quantity or amount values for price derivation rule eligibilities with threshold type code QUTI (quantity interval) or AMTI (amount interval)

The following attributes in the promotion master data are utilized during the validation of the promotion:

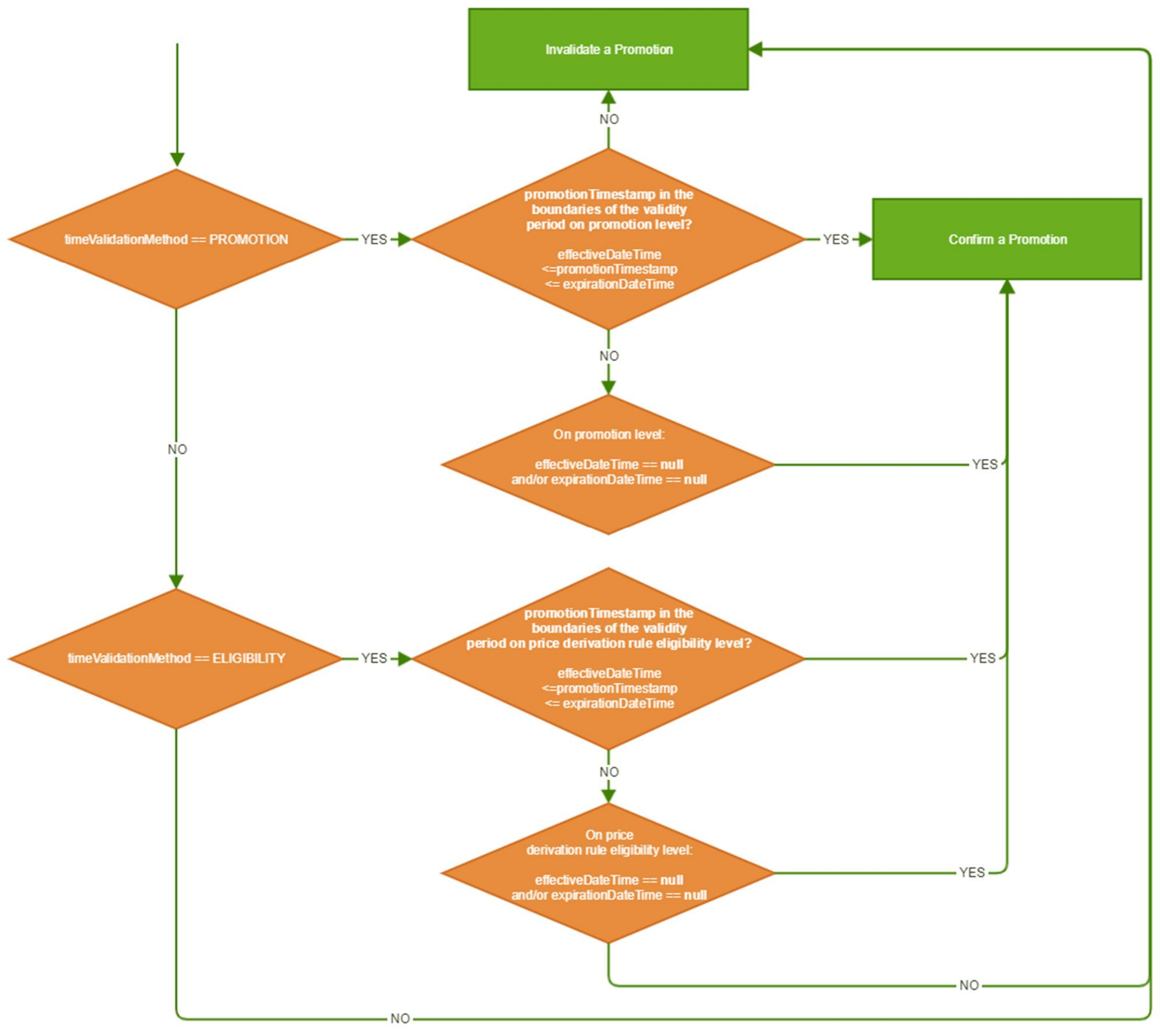
- Start time on promotion level (*PromotionSO.effectiveDateTime*) and end time on promotion level (*PromotionSO.expirationDateTime*)
- Start time on price derivation rule eligibility level (*PromotionConditionEligibilitySO.effectiveDateTime*) and end time on price derivation rule eligibility level (*PromotionConditionEligiblitySO.expirationDateTime*)
- Interval quantity (*ThresholdPromotionConditionEligibility.intervalQuantity*) and interval amount (*ThresholdPromotionConditionEligibility.intervalAmount)*

The start time is always the start point and the end time the end point of a validity period. The validity period is validated either on promotion level or on price derivation rule eligibility level. This is decided with the system parameter *timeValidationMethod*. This parameter can have the following two values:

- PROMOTION: validation is done on the promotion level
- ELIGIBILITY: validation is done on the price derivation rule eligibility level

The *timeValidationMethod* is PROMOTION by default.

The following figure gives an overview about the important attributes for the validation of a promotion with respect to the *timeValidationMethod* parameter. It is illustrated, which value of which attribute is resulting in which of the subsequently described scenarios.


Invalidate a Promotion
NO
timeValidationMethod == PROMOTION
YES
promotionTimestamp in the boundaries of the validity period on promotion level?
effectiveDateTime <=promotionTimestamp <= expirationDateTime
YES
Confirm a Promotion
NO
On promotion level:
effectiveDateTime == null and/or expirationDateTime == null
YES
NO
timeValidationMethod == ELIGIBILITY
YES
promotionTimestamp in the boundaries of the validity period on price derivation rule eligibility level?
effectiveDateTime <=promotionTimestamp <= expirationDateTime
YES
NO
On price derivation rule eligibility level:
effectiveDateTime == null and/or expirationDateTime == null
YES
NO
NO

The validation of price derivation rule eligibilities with threshold type code QUTI (quantity interval) or AMTI (amount interval) with respect to their interval quantity or amount value is performed as follows. If a price derivation rule eligibility of a promotion price derivation rule has the threshold type code QUTI (quantity interval) and the interval quantity value specified by the price derivation rule eligibility is lower than or equal to zero, then the price derivation rule eligibility and the corresponding promotion price derivation rule will be made invalid. Also, if a price derivation rule eligibility of a promotion price derivation rule has threshold type code AMTI (amount interval) and the interval amount value specified by the price derivation rule eligibility is lower than or equal to zero, then the price derivation rule eligibility and the corresponding promotion will be made invalid.

The PCE loads all promotion price derivation rules according to the activated price derivation rule eligibilities. These promotion price derivation rules are validated before the PCE executes the included price derivation rules. Thereby, the following outcomes are possible:

1. Confirm a promotion price derivation rules
2. Invalidate a promotion price derivation rules

6.3.1 Confirm a Promotion

To state that a promotion price derivation rules is valid, the validity period is compared with the time stamp of the transaction (*RetailTransaction.promotionTimestamp*).

If the *timeValidationMethod* is equal to PROMOTION, the start time on promotion level has to be smaller than or equal to the time stamp of the transaction. The end time on promotion level has to be greater than or equal to the time stamp of the transaction. Thus, the validity period encloses the time stamp of the transaction. The validity period of the promotion's eligibilities are not evaluated if the parameter *timeValidationMethod* is equal to PROMOTION.

If the system parameter *timeValidationMethod* is ELIGIBILITY, the start time and the end time are used that are stated in the price derivation rule eligibilities. Thereby, the time stamp of the transaction has to lie between these two values. The validity period of the promotion is not evaluated if the parameter *timeValidationMethod* is equal to ELIGIBILITY.

On both promotion and eligibility level validation, if the start time and the end time are both null, the promotion is confirmed regardless of the time stamp of the transaction.

The following applies additionally:

- If the price derivation rule eligibility that is not to be considered is a parent eligibility of a combination eligibility, all child eligibilities are also not taken into account.
- If the eligibility is a parent eligibility of a combination eligibility with the combination code (*CombinationPromotionConditionEligibilitySO.combinationCode*) equal to AND and this price derivation rule eligibility is confirmed, all child eligibilities must also be valid. Otherwise, this combination eligibility is not taken into account.

Note that a price derivation rule eligibility with an invalid validity period does not result in an invalid promotion. If at least one valid price derivation rule eligibility exists within a promotion, the promotion is valid for exactly this price derivation rule eligibility.

6.3.2 Invalidate a Promotion

To state that a promotion is invalid, the validity period is compared with the time stamp of the transaction.

If the *timeValidationMethod* is equal to PROMOTION, the start time on promotion level has to be not null and greater than the time stamp of the transaction. The end time on promotion level has to be not null and smaller than the time stamp of the transaction.

If the parameter *timeValidationMethod* is ELIGIBILITY, the start time and the end time are used that are stated in the price derivation rule eligibilities. Thereby, the time stamp of the transaction does not lie between these two values. A special case is when more than one price derivation rule eligibility or more than one combination eligibility exist. To state that the whole promotion is invalid, all contained price derivation rule eligibilities must have an start time on price derivation rule eligibility level and end time on price derivation rule eligibility level which are not null and do not enclose the time stamp of the transaction. As described previously, the promotion is confirmed otherwise but only the valid price derivation rule eligibilities are considered.

Additionally, invalidation of a price derivation rule eligibility with threshold type code QUTI or AMTI as a result having a value for interval quantity or amount in order, which is lower than or equal to zero, invalidates the corresponding promotion.

6.3.3 THERE IS MORE

Further information about the parameter and attributes can be found in chapter Configuration.

6.4 Recurrent Offer

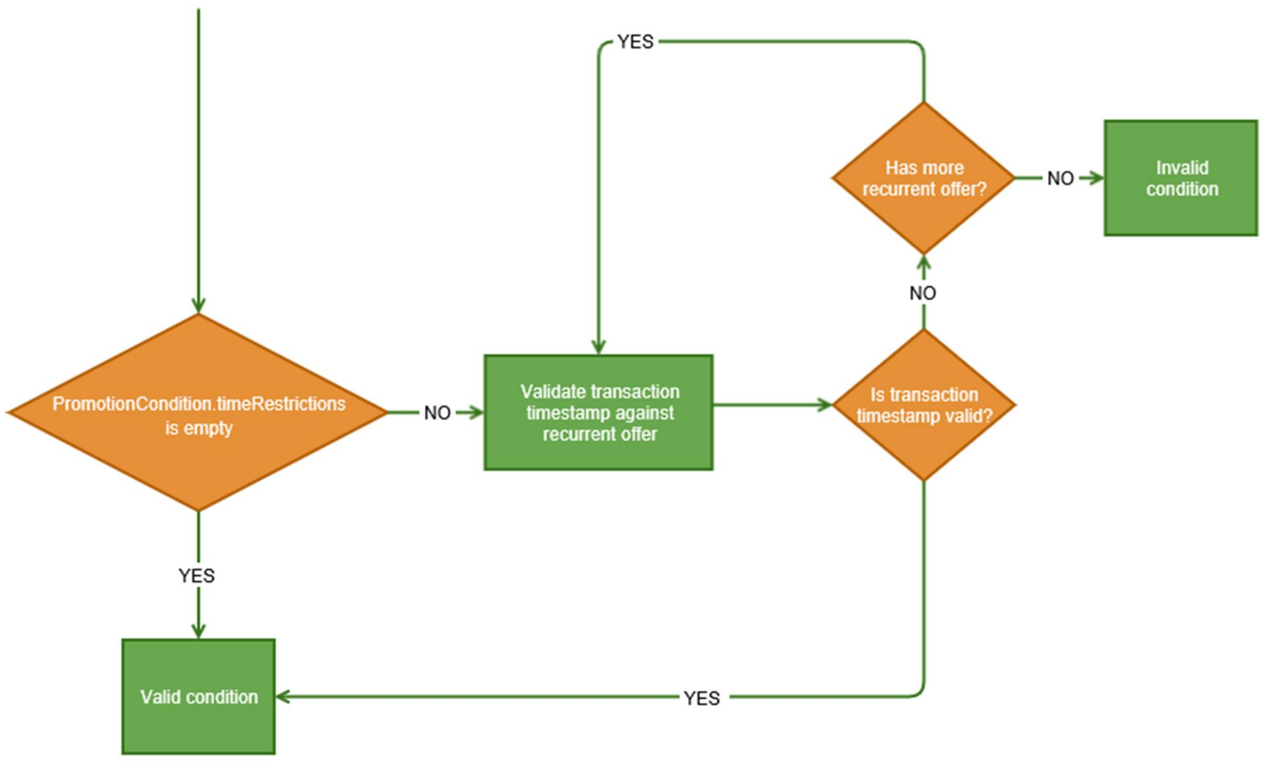
A promotion price derivation rule can be created for a certain validity period. Within this validity period, the promotion price derivation rule can be applied to a transaction if the transaction fulfills the corresponding eligibilities. It is also possible to define additional recurrent offers for a promotion derivation rule. If the transaction timestamp fulfills the recurrent offers of a validity period, the promotion price derivation rule can be applied to it.

Promotion price derivation rules can have one or more recurrent offers, these recurrent offers determine the validity period in which the promotion can be applied to the transaction.
The following rules must be considered:

- The *PromotionCondition* must have at least one *TimeRestriction* related to; otherwise, the promotion price derivation rule is always valid regardless of the current timestamp of the transaction.
- If the *PromotionCondition* has exactly one *TimeRestriction* and the *TimeRestriction.restricion* is empty, null, or has an invalid expression, the promotion price derivation rule is always not valid.
- If the *PromotionCondition* has at least two *TimeRestriction* and a *TimeRestriction.restricion* is empty, null, or has an invalid expression, this *TimeRestriction* is ignored and the remaining *TimeRestriction*s are evaluated to decide whether the promotion price derivation rule is valid.

HOW IT WORKS

A promotion price derivation rule can have a list of one or more defined recurrent offers which are based on Cron expressions. These expressions are processed in OR combinations and the transaction timestamp is validated against the recurrent offer. If the transaction timestamp is within the validity period defined by at least one of the recurrent offers, the promotion will be applied.


PromotionCondition.timeRestrictions is empty
NO
YES
Validate transaction timestamp against recurrent offer
Is transaction timestamp valid?
NO
YES
Has more recurrent offer?
YES
NO
Invalid condition
Valid condition

6.4.1 Validate recurrent offer against transaction timestamp with valid condition

Example 1

Let us assume that the promotion price derivation rule has only one recurrent offer with the following expression:

- ** * 7-9 ? * 2*

Which means that the promotion price derivation rule can be applied on each Monday morning between 7 o'clock and 10 o'clock.

Assuming that the transaction timestamp is *(2019-02-11T07:30:07.648+01:00)*, the condition is satisfied and the promotion can be applied.

Example 2

Let us assume that the promotion price derivation rule has two recurrent offers with the following expressions:

- ** * 7-9 ? * 2*

- ** * * ? * 3*

This means that the promotion can be applied every Monday between 7 and 10 or every Tuesday at any time.
Let us also assume that the transaction timestamp is *(2019-02-11T07:30:07.648+01:00)*. The transaction timestamp fulfills the first restriction, the promotion price derivation rule is valid, and the second restriction is not tested, the promotion can be applied.
In case the transaction timestamp(*2019-02-09T07:30:07.648+01:00*) does not fulfill either recurrent offers, because it is Saturday. As result, the promotion price derivation rule is invalidated and it is not applied.

6.4.2 Validate recurrent offer against transaction timestamp with invalid condition

Example 1
Let us assume that the promotion price derivation rule has only one recurrent offer with the following expression:

- ** * 7-9 ? * 2*

Which means that the promotion price derivation rule can be applied on each Monday morning between 7 o'clock and 10 o'clock.
Assuming the transaction timestamp falls on a Tuesday and the time is between 7 o'clock and 10 o'clock (*2019-02-12T07:30:07.648+01:00)*, the condition is no longer satisfied and the promotion cannot be applied. The time fits, but the promotion cannot be applied because the day Tuesday is wrong.

Example 2
Let us assume that the promotion price derivation rule has only one recurrent offer with the following expression:

- *null or empty*

Which means that no restriction was found. In that case, the condition is no longer satisfied and the promotion price derivation rule cannot be applied.

6.4.3 THERE IS MORE

TimePeriod.TimeRestriction uses Cron expressions as a pattern to provide the ability to specify complex time combinations including intervals, weekdays, and special cases. Further information about Cron expressions can be found in chapter Cron expression.

6.5 Set the Calculation Base Amount

It is possible to define which previously applied promotion price derivation rules shall be accounted in the ongoing promotion calculation process and which shall not. This is done by setting the calculation base sequence for each price derivation rule separately in the promotion master data. This means that the calculated benefit is applied to the line item or transaction, respectively, but not each benefit calculated before is taken into account in the ongoing calculation process.

HOW IT WORKS

The calculation base amount (*RetailPriceModifier.calculationBaseAmount*, *PriceModificationLineItem.calculationBaseAmount*, *FrequentShopperPointsModifier.computationBaseAmount*, or *LoyaltyRewardLineItem.computationBaseAmount*) is computed for each appliance of a benefit. It is the price basis used for computing the benefit. It also helps to retrace the computations of the PCE.

The main attribute is the calculation base sequence (*PromotionConditionRuleSO.calculationBaseSequence*) for controlling the calculation base amount. The calculation base sequence determines which of the previously computed discount sales prices are considered for obtaining the calculation base amount. It is used as follows:

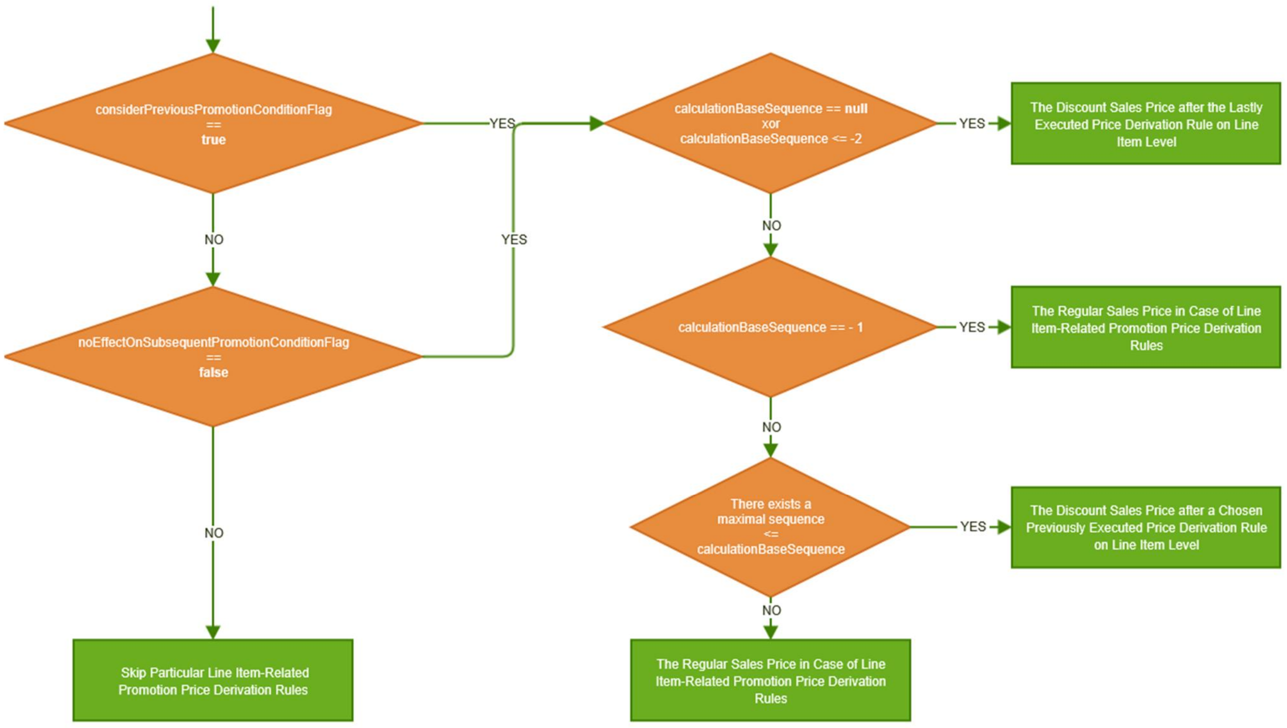
There are no previously executed price derivation rules or **the calculation base sequence is equal to -1**	Calculation base amount is the regular sales price (line item-related promotion price derivation rule) or the regular total amount (transaction-related promotion price derivation rule).
Calculation base sequence is null or less than/equal to -2	All previously executed price derivation rules influence the calculation base amount.
Otherwise **(Calculation base sequence is greater than/equal to zero)**	In case of a line item-related promotion price derivation rule, the calculation base amount of the current price derivation rule is equal to the discount sales price of a previously executed promotion price derivation rule that has the highest **sequence** (*PromotionConditionSO.sequence*) which is less than or equal to the calculation base sequence. In case of a transaction-related promotion price derivation rule, the calculation base amount of the current price derivation rule is equal to the discount total amount of the transaction-related promotion price derivation rule that has the highest sequence which is less than or equal to the calculation base sequence.

The consider predecessors flag (*PromotionConditionRuleSO.considerPreviousPromotionConditionFlag*) and the no effect on successors flag (*PromotionConditionRuleSO.noEffectOnSubsequentPromotionConditionFlag*) also influence the calculation base amount but only for line item-related promotion price derivation rules. The consider predecessors flag controls whether the concept behind the attribute no effect on successors flag false or the calculation base sequence concept true is to be considered. The attribute no effect on successors flag determines whether a promotion price derivation rule influences the calculation base amount of subsequent promotion price derivation rules false or not true.

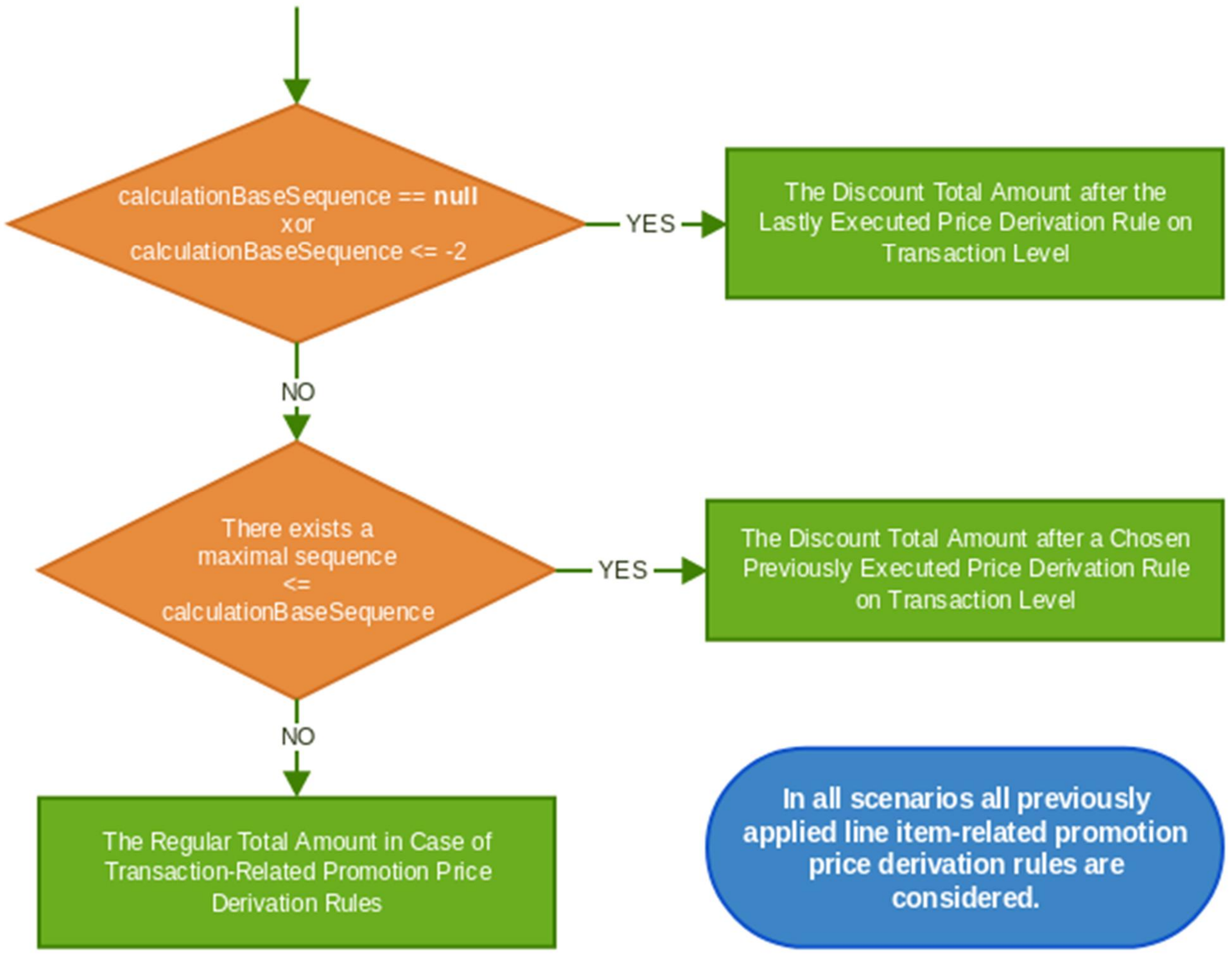
The no effect on successors flag (*PromotionConditionRuleSO.noEffectOnSubsequentPromotionConditionFlag*) of all previously applied conditions is only checked if the consider predecessors flag (PromotionConditionRuleSO.considerPreviousPromotionConditionFlag) is false on the currently processed condition. When the no effect on successors flag is false, then all previously applied conditions with no effect on successors flag false must be considered in the calculation base amount of the current condition. Otherwise, the calculation base amount of the current condition is set to the original price.

All these values are part of the promotion master data and obtained by the PCE through the promotion master data API.

The different attribute values and their meaning are roughly illustrated in the following decision tree for line item-related promotion price derivation rules.


considerPreviousPromotionConditionFlag == true
YES
NO
noEffectOnSubsequentPromotionConditionFlag == false
YES
NO
Skip Particular Line Item-Related Promotion Price Derivation Rules
calculationBaseSequence == null xor calculationBaseSequence <= -2
YES
The Discount Sales Price after the Lastly Executed Price Derivation Rule on Line Item Level
NO
calculationBaseSequence == - 1
YES
The Regular Sales Price in Case of Line Item-Related Promotion Price Derivation Rules
NO
There exists a maximal sequence <= calculationBaseSequence
YES
The Discount Sales Price after a Chosen Previously Executed Price Derivation Rule on Line Item Level
NO
The Regular Sales Price in Case of Line Item-Related Promotion Price Derivation Rules

The different attribute values and their meaning are roughly illustrated in the following decision tree for transaction-related promotion price derivation rules.


calculationBaseSequence == null
xor
calculationBaseSequence <= -2
YES
The Discount Total Amount after the Lastly Executed Price Derivation Rule on Transaction Level
NO
There exists a maximal sequence <= calculationBaseSequence
YES
The Discount Total Amount after a Chosen Previously Executed Price Derivation Rule on Transaction Level
NO
The Regular Total Amount in Case of Transaction-Related Promotion Price Derivation Rules
In all scenarios all previously applied line item-related promotion price derivation rules are considered.

Example: Set the Calculation Base Amount

Let us assume that there are the following items in the master data:

Item (Merchandise Category)	Regular Sales Unit Price
Desktop PC (electronic)	444.44€
Laptop (electronic)	555.55€
Shirt (clothing)	15.95€

We assume that the following promotion price derivation rules exist on line item level in the promotion master data:

1. First promotion: Items of the merchandise category electronic get a discount of 2%.

- Threshold type: QUTI
- Quantity threshold: 2
- Quantity interval: 2
- Quantity limit: 8

- sequence: 600
- considerPreviousPromotionConditionFlag true

2. Second promotion: Items of the merchandise category electronic get a discount of 25%.

- ThresholdType: QUTI
- Quantity threshold: 3
- Quantity interval: 3
- Quantity limit: 8
- sequence: 601
- considerPreviousPromotionConditionFlag true

3. Third promotion: Items of the merchandise category electronic get a discount of 50%.

- ThresholdType: QUTI
- Quantity threshold: 2
- Quantity interval: 2
- Quantity limit: 8
- sequence: 602
- considerPreviousPromotionConditionFlag true

We also assume that the following promotion price derivation rules exist on transaction level in the promotion master data:

4. Fourth promotion: Buy a basket for a minimum amount of 100€ and get 10€ discount with coupon #1.

- sequence 407891

5. Fifth promotion: Manually triggered discount is allowed when the value of the basket is greater than 100€. The price of the basket is manually set to 100.00€ then.

- sequence 407893

6. Sixth promotion: With coupon #2, get a discount of 20% for the basket.

- sequence 407895

The PCE calculates the benefit according to the three attributes:

- Consider predecessors flag
- No effect on successors flag
- Calculation base sequence

Thereby, the following outcomes are possible:

1. Use the regular sales price
2. Use the discount sales price after the lastly executed price derivation rule
3. Use the discount sales price after a chosen previously executed price derivation rule
4. Skip particular line item-related promotion price derivation rules

6.5.1 Determine the Items Which Are Considered for the Calculation Base Amount

The calculation base sequence in the promotion master data determines the prices of the items, which are used to compute the calculation base amount. But to determine the items which are considered for the calculation base amount, the PCE distinguishes line item-related discounts and transaction-related discounts.

A line item-related promotion price derivation rule uses the triggering items defined in the price derivation rule eligibility for computing the calculation base amount. If no triggering items are defined in the price derivation rule eligibility (e.g customer group or coupon as trigger) all items in the basket are used to compute the calculation base amount.

A transaction-related promotion price derivation rule determines the items to compute the calculation base amount by the system parameter *transaction rebate method*.

The *transaction rebate method* TRIGGER determines to use the triggering items to compute the calculation base amount. The *transaction rebate method* TOTAL determines to use all items of the transaction to compute the calculation base amount (for more information see chapters Prorate the Benefit and Use Thresholds, Intervals, and Limits).

The following table gives an overview of how the PCE determines the items to compute the calculation base amount,depending on the *transaction control break code* (PromotionConditionRuleSO.transactionControlBreakCode), the *Transaction Rebate Method* and the eligibility:

TransactionControlBreakCode	TransactionRebateMethode	Items Considered for the Calculation Base Amount	
		Eligibility with defined triggering items (e.g. item, MHG)	**Eligibility without defined triggering items** (e.g. coupon or customer group as trigger)
PO/PC (line item-related)	-	eligible items	all items in the transaction
SU/SP (transaction-related)	TRIGGER	eligible items	all items in the transaction
SU/SP (transaction related)	TOTAL	all items in the transaction	all items in the transaction

6.5.2 Use the Regular Sales Price

6.5.2.1 The Regular Sales Price in Case of Line Item-Related Promotion Price Derivation Rules

The consider predecessors flag has to be true. The calculation base sequence needs to be -1. If this is the case, the PCE uses the regular sales price of the line items as calculation base amount for the current computation of the benefit. If there is no previously computed and applied benefit, the PCE uses the regular sales price as the calculation base amount, too.

Example (cont.)

Let us assume that a customer buys the following items with the following regular sales price:

	Item (Merchandise Category)	Quantity	Regular Sales Price
1. line item	Desktop PC (electronic)	2	888.88€
2. line item	Laptop (electronic)	1	555.55€

The application of the previously mentioned line item-related promotion price derivation rules results in the following values if the cheapest items are chosen first. The column "Calculation Base Amount" shows the values of the calculation base amount if the calculation base sequence of all three promotion price derivation rules is -1.

Sequence	Promotion Price Derivation Rule	Calculation Base Sequence	Calculation Base Amount	Quantity	Monetary Discount	Previous Price	New Price
600	Discount 2% for every 2 quantities	-1	1. line item: 888.88€	2	1. line item: 17.78€	1. line item: 888.88€	1. line item: 871.10€
			2. line item: **null**	1	2. line item: **null**	2. line item: 555.55€	2. line item: 555.55€
601	Discount 25% for every 3 quantities	-1	1. line item: 888.88€	2	1. line item: 222.22€	1. line item: 871.10€	1. line item: 648.88€
			2. line item: 555.55€	1	2. line item: 138.89€	2. line item: 555.55€	2. line item: 416.66€
602	Discount 50% for every 2 quantities	-1	1. line item: 888.88€	2	1. line item: 444.44€	1. line item: 648.88€	1. line item: 204.44€
			2. line item: **null**	1	2. line item: **null**	2. line item: 416.66€	2. line item: 416.66€

The monetary discount is obtained according to the calculation base amount and the calculation base sequence. The application of the first promotion price derivation rule grants 2% discount of the regular sales price 888.88€ for the first line item. The monetary discount of the second promotion price derivation rule is 25% of 888.88€ for the first line item and of 555.55€ for the second line item. The PCE calculates for the lastly executed price derivation rule 50% of 888.88€ for the first line item.

6.5.2.2 The Regular Total Amount in Case of Transaction-Related Promotion Price Derivation Rules

The calculation of transaction-related promotion price derivation rules is executed after the computation of the line item-related ones. This means that for any transaction-related promotion price derivation rule all discount sales prices of the line items are used if such a promotion price derivation rule already was applied.

If the calculation base sequence is -1, the calculation base amount is the regular sales price of the line items for the computation of the benefit.

Example (cont.)

Let us assume that a customer buys the following items with the following regular sales price:

	Item (Merchandise Category)	Quantity	Regular Sales Price
1. line item	Shirt (clothing)	10	159.50€

Then the application of the previously mentioned transaction-related promotion price derivation rules results in the following values. The column "Calculation Base Amount" shows the values of the calculation base amount if the calculation base sequence of all three promotion price derivation rules is -1.

Sequence	Promotion Price Derivation Rule	Calculation Base Sequence	Calculation Base Amount	Quantity	Monetary Discount	Previous Price	New Price
407891	Buy a basket for a minimum amount of 100€ and get 10€ discount with coupon #1.	-1	1. line item: 159.50 €	10	1. line item: 10.00€	1. line item: 159.50€	1. line item: 149.50 €
407893	Manually triggered discount is allowed when the value of the basket is greater than 100€. The price of the basket is manually set to 100.00€ then.	-1	1. line item: 159.50€	10	1. line item: 59.50€	1. line item: 149.50€	1. line item: 90.00€
407895	With coupon #2, get a discount of 20% for the basket.	-1	1. line item: 159.50 €	10	1. line item: 31.90€	1. line item: 90.00€	1. line item: 58.10€

The monetary discount is obtained according to the calculation base amount and the calculation base sequence. The application of the first promotion price derivation rule grants 10.00€ discount if the basket amount is greater than 100€. Since the calculation base amount is greater than this threshold, the price derivation rule is executed. If a line item-related promotion price derivation rule sets the regular sales price of the ten shirts to 99.95€, the first and the second transaction-related promotion price derivation rules are not applicable. This is due to the calculation of transaction-related promotion price derivation rules on the basis of the line item-related ones. The monetary discount of the second promotion price derivation rule is 59.50€ since this is the difference between the calculation base amount of 159.50€ and the new discount total amount of 100.00€. The PCE calculates for the lastly executed price derivation rule 20% of 159.50€.

6.5.3 Use the Discount Sales Price after the Lastly Executed Price Derivation Rule

6.5.3.1 The Discount Sales Price after the Lastly Executed Price Derivation Rule on Line Item Level

The consider predecessors flag has to be true. The calculation base sequence has to be either null or less than or equal to -2. If this is the case, the PCE uses the discount sales price of the line items after the lastly executed price derivation rule as calculation base amount for the current computation of the benefit. If no previously computed and applied benefit exists, the PCE uses the regular sales price as the calculation base amount.

Example (cont.)

Let us assume that a customer buys the following items with the following regular sales price:

	Item (Merchandise Category)	Quantity	Regular Sales Price
1. line item	Desktop PC (electronic)	2	888.88€
2. line item	Laptop (electronic)	1	555.55€

The application of the previously mentioned promotion price derivation rules results in the following values if the cheapest items are chosen first. The column "Calculation Base Amount" shows the value of the calculation base amount if the calculation base sequence of all three promotion price derivation rules is -2.

Sequence	Promotion Price Derivation Rule	Calculation Base Sequence	Calculation Base Amount	Quantity	Monetary Discount	Previous Price	New Price
600	Discount 2% for every 2 quantities	-2	1. line item: 888.88€	2	1. line item: 17.78€	1. line item: 888.88€	1. line item: 871.10€
			2. line item: **null**	1	2. line item: **null**	2. line item: 555.55€	2. line item: 555.55€

601	Discount 25% for every 3 quantities	-2	1. line item: 871.10€	2	1. line item: 217.78€	1. line item: 871.10€	1. line item: 653.32€
			line item: 555.55€	1	line item: 138.89€	line item: 555.55€	line item: 416.66€
602	Discount 50% for every 2 quantities	-2	1. line item: 653.32€	2	1. line item: 326.66€	1. line item: 653.32€	1. line item: 326.66€
			2. line item: **null**	1	2. line item: **null**	2. line item: 416.66€	2. line item: 416.66€

The monetary discount is obtained according to the calculation base amount and the calculation base sequence. The application of the first promotion price derivation rule grants 2% discount of the lastly obtained discount sales price 871.10€ for the first line item. The monetary discount of the second promotion price derivation rule is 25% of 871.10€ for the first line item and of 555.55€ for the second line item. The PCE calculates for the lastly executed price derivation rule 50% of 653.32€ for the first line item.

6.5.3.2 The Discount Total Amount after the Lastly Executed Price Derivation Rule on Transaction Level

If the calculation base sequence is less than or equal to -2, the calculation base amount is the discount total amount after the lastly applied transaction-related promotion price derivation rule. The PCE calculates the benefit on the basis of the calculation base amount.

Example (cont.)

Let us assume that a customer buys the following items with the following regular sales price:

	Item (Merchandise Category)	Quantity	Regular Sales Price
1. line item	Shirt (clothing)	10	159.50€

The application of the previously mentioned transaction-related promotion price derivation rules results in the following values. The column "Calculation Base Amount" shows the values of the calculation base amount if the calculation base sequence of all three promotion price derivation rules is -2.

Sequence	Promotion Price Derivation Rule	Calculation Base Sequence	Calculation Base Amount	Quantity	Monetary Discount	Previous Price	New Price
407891	Buy a basket for a minimum amount of 100€ and get 10€ discount with coupon #1.	-2	1. line item: 159.50 €	10	1. line item: 10.00€	1. line item: 159.50€	1. line item: 149.50 €
407893	Manually triggered discount is allowed when the value of the basket is greater than 100€. The price of the basket is manually set to 100.00€ then.	-2	1. line item: 149.50 €	10	1. line item: 49.50€	1. line item: 149.50€	1. line item: 100.00 €
407895	With coupon #2, get a discount of 20% for the basket.	-2	1. line item: 100.00 €	10	1. line item: 20.00€	1. line item: 100.00€	1. line item: 80.00€

The monetary discount is obtained according to the calculation base amount and the calculation base sequence. The application of the first promotion price derivation rule grants 10.00€ discount if the basket amount is greater than 100€. Since the calculation base amount is greater than this threshold, the price derivation rule is executed. The monetary discount of the second promotion price derivation rule is 49.50€ since this is the difference between the calculation base amount of 149.50€ and the new discount total amount of 100.00€. The PCE calculates for the lastly executed price derivation rule 20% of 100.00€.

6.5.4 Use the Discount Sales Price after a Chosen Previously Executed Price Derivation Rule

6.5.4.1 The Discount Sales Price after a Chosen Previously Executed Price Derivation Rule on Line Item Level

The consider predecessors flag has to be true. The calculation base sequence has to be greater than or equal to zero. If this is the case, the PCE uses the discount sales price of the line items after a chosen executed price derivation rule as calculation base amount for the current computation of the benefit.

Thereby, the sequence of a promotion price derivation rule is important. The sequence of previously executed price derivation rules has to be less than or equal to the calculation base sequence. From all promotion price derivation rules that have a sequence that is smaller than or equal to the calculation base sequence, the discount sales price is chosen from the promotion price derivation rule with the highest sequence. This discount sales price is then used as the calculation base amount for the current computation of the benefit. If there is no previously computed and applied benefit, the PCE uses the regular sales price as the calculation base amount.

Example (cont.)

Let us assume that a customer buys the following items with the following regular sales prices:

	Item (Merchandise Category)	Quantity	Regular Sales Price
1. line item	Desktop PC (electronic)	2	888.88€
2. line item	Laptop (electronic)	1	555.55€

The application of the previously mentioned promotion price derivation rules results in the following values if the cheapest items are chosen first. The column "Calculation Base Amount" shows the value of the calculation base amount if the calculation base sequence of all three promotion price derivation rules is 600.

Sequence	Promotion Price Derivation Rule	Calculation Base Sequence	Calculation Base Amount	Quantity	Monetary Discount	Previous Price	New Price
600	Discount 2% for every 2 quantities	600	1. line item: 888.88€	2	1. line item: 17.78€	1. line item: 888.88€	1. line item: 871.10€
			2. line item: **null**	1	2. line item: **null**	2. line item: 555.55€	2. line item: 555.55€
601	Discount 25% for every 3 quantities	600	1. line item: 871.10€	2	1. line item: 217.78€	1. line item: 871.10€	1. line item: 653.32€
			2. line item: 555.55€	1	2. line item: 138.89€	2. line item: 555.55€	2. line item: 416.66€
602	Discount 50% for every 2 quantities	600	1. line item: 871.10€	2	1. line item: 435.56€	1. line item: 653.32€	1. line item: 217.76€
			line item: **null**	1	line item: **null**	line item: 416.66€	line item: 416.66€

The monetary discount is obtained according to the calculation base amount and the calculation base sequence. The application of the first promotion price derivation rule grants 2% discount of the regular sales price 888.88€ for the first line item. The monetary discount of the second promotion price derivation rule is 25% of 871.10€ for the first line item and of 555.55€ for the second line item. The PCE calculates for the lastly executed price derivation rule 50% of 871.10€ for the first line item.

6.5.4.2 The Discount Total Amount after a Chosen Previously Executed Price Derivation Rule on Transaction Level

If the calculation base sequence is greater than or equal to zero, the calculation base amount is the discount total amount after a chosen previously applied transaction-related promotion price derivation rule. The PCE calculates the benefit on the basis of the calculation base amount. Thereby, the sequence of a transaction-related promotion price derivation rule is important. The sequence of previously applied transaction-related promotion price derivation rules has to be less than or equal to the calculation base sequence. From all transaction-related promotion price derivation rules that have a sequence that is smaller than or equal to the calculation base sequence, the discount total amount is chosen from the transaction-related promotion price derivation rule with the highest sequence.

Example (cont.)

Let us assume that a customer buys the following items with the following regular sales price:

	Item (Merchandise Category)	Quantity	Regular Sales Price
1. line item	Shirt (clothing)	10	159.50€

The application of the previously mentioned transaction-related promotion price derivation rules results in the following values. The column "Calculation Base Amount" shows the values of the calculation base amount if the calculation base sequence of all three promotion price derivation rules is 407893.

Sequence	Promotion Price Derivation Rule	Calculation Base Sequence	Calculation Base Amount	Quantity	Monetary Discount	Previous Price	New Price
407891	Buy a basket for a minimum amount of 100€ and get 10€ discount with coupon #1.	407893	1. line item: 159.50 €	10	1. line item: 10.00€	1. line item: 159.50€	1. line item: 149.50 €

407893	Manually triggered discount is allowed when the value of the basket is greater than 100€. The price of the basket is manually set to 100.00€ then.	407893	1. line item: 149.50 €	10	1. line item: 49.50€	1. line item: 149.50€	1. line item: 100.00 €
407895	With coupon #2, get a discount of 20% for the basket.	407893	1. line item: 149.50 €	10	1. line item: 29.90€	1. line item: 100.00€	1. line item: 70.10€

The monetary discount is obtained according to the calculation base amount and the calculation base sequence. The application of the first promotion price derivation rule grants 10.00€ discount if the basket amount is greater than 100€. Since the calculation base amount is greater than this threshold, the price derivation rule is executed. The monetary discount of the second promotion price derivation rule is 49.50€ since this is the difference between the calculation base amount of 149.50€ and the new discount total amount of 100.00€. The PCE calculates for the lastly executed price derivation rule 20% of 149.50€.

6.5.5 Skip Particular Line Item-Related Promotion Price Derivation Rules

The skipping of particular promotion price derivation rules is only possible for line item-related ones. The consider predecessors flag has to be false. The no effect on successors flag can either be false or true. It depends on the no effect on successors flag of each executed price derivation rule how PCE computes the calculation base amount.

A promotion price derivation rule influences the calculation base amount of subsequent promotion price derivation rules if the no effect on successors flag is false and not if the flag is true. Thereby, the consider predecessors flag is evaluated from the currently processed promotion price derivation rule. If the flag is false, the no effect on successors flag is evaluated of all previously executed price derivation rules.

Example (cont.)

Let us assume that the previously mentioned promotion price derivation rules exist on line item level in the promotion master data. But this time, these are as follows:

1. First promotion: Items of the merchandise category electronic get a discount of 2%.
   - sequence: 600
   - considerPreviousPromotionConditionFlag false
   - noEffectOnSubsequentPromotionConditionFlag false

2. Second promotion: Items of the merchandise category electronic get a discount of 25%.
   - sequence: 601
   - considerPreviousPromotionConditionFlag false
   - noEffectOnSubsequentPromotionConditionFlag true
3. Third promotion: Items of the merchandise category electronic get a discount of 50%.
   - sequence: 602
   - considerPreviousPromotionConditionFlag false
   - noEffectOnSubsequentPromotionConditionFlag false

We assume that a customer buys the following items with the following regular sales prices:

	Item (Merchandise Category)	Quantity	Regular Sales Price
1. line item	Desktop PC (electronic)	2	888.88€
2. line item	Laptop (electronic)	1	555.55€

The application of the previously mentioned promotion price derivation rules results in the following values if the cheapest items are chosen first. The column "Calculation Base Amount" shows the value of the calculation base amount.

Sequence	Promotion Price Derivation Rule	Calculation Base Amount	Quantity	Monetary Discount	Previous Price	New Price
600	Discount 2% for every 2 quantities	1. line item: 888.88€	2	1. line item: 17.78€	1. line item: 888.88€	1. line item: 871.10€
		2. line item: **null**	1	2. line item: **null**	2. line item: 555.55€	2. line item: 555.55€
601	Discount 25% for every 3 quantities	1. line item: 871.10€	2	1. line item: 217.78€	1. line item: 871.10€	1. line item: 653.32€
		2. line item: 555.55€	1	2. line item: 138.89€	2. line item: 555.55€	2. line item: 416.66€
602	Discount 50% for every 2 quantities	1. line item: 871.10€	2	1. line item: 435.56€	1. line item: 653.32€	1. line item: 217.76€
		2. line item: **null**	1	2. line item: **null**	2. line item: 416.66€	2. line item: 416.66€

The monetary discount is obtained according to the calculation base amount and the no effect on successors flag. The application of the first promotion price derivation rule grants 2% discount of the regular sales price 888.88€ for the first line item. The monetary discount of the second promotion price derivation rule is 25% of 871.10€ for the first line item and of 555.55€ for the second line item. This is due to the fact that the first promotion price derivation rule shall influence the subsequent ones. The PCE calculates for the lastly executed price derivation rule 50% of 871.10€ for the first line item. This is due to the fact that the second promotion price derivation rule shall not influence the subsequent ones.

6.5.6 THERE IS MORE

Only monetary discounts are considered in the calculation base amount. This means, for example, if a promotion price derivation rule contains the print out of a coupon and applies a virtual discount of 0.50€, the calculation base amount is not affected—regardless of the configuration possibilities. This is the case for line item- and transaction-related promotion price derivation rules. Further explanations regarding the monetary and virtual discounts are given in chapter Select the Method to Discount.

The applicability of a promotion price derivation rule cannot be influenced by setting the calculation base amount if it only is valid for fully-priced items. Otherwise, the functionality of "fully-priced items only" could be circumvented.

Further information about the attributes can be found in chapter Configuration.

An item with a manually overridden price (see chapter Manually Override a Regular Sales Unit Price) is handled just like any other item. Thus in case of calculation base sequence -1, the manually overridden price is considered as the regular sales price.

6.6 Handle Sales and Returns without the Original Transaction

A customer brings back items to a store and wants to get back his/her money. Typically, the customer has to bring also the receipt from the corresponding purchase. If the original receipt is there, the POS retrieves the related transaction and calculates the money the customer gets back, the entire items in it can be returned or only a part of the transaction. The granted promotions are considered from the original receipt so that the paid price by the customer is given back to the customer. This process also assures that the previously gained money is transferred back to the account.

In some cases, the customer does not have any longer the receipt but wants to return items. The trader is then able to decide whether promotions are applied to returns, sales, or sales and returns for each promotion in the promotion master data. It depends on the goodwill of the trader if such a return scenario is possible or not. There is no possibility to know whether the customer has paid more or less in the previous purchase for the returned goods since the original receipt is missing. The trader bears the risk to lose money in this case.

In case of a return where the original transaction is missing, the Client application cannot handle the returned goods (herein also returns), but the PCE handles these. It is possible to define promotion price derivation rules in the promotion master data that are applicable to sold line items (herein also sales), returns, or both. This document is only related to this case where the original receipt is missing and no information is available except the returned goods.

The discounts and loyalty points are proportionally cleared based on the promotion price derivation rules applied to the original transaction. The Client application is handling this entire process.

HOW IT WORKS

The sale return type (*PromotionConditionSO.saleReturnTypeCode*) of the promotion price derivation rule decides whether the sales, the returns, or both are considered or not. This attribute can be set with one of the following values in the promotion master data:

- SALES_RETURNS/"00": Sales and returns are considered.
- SALES/"01": Sales are considered only.
- RETURNS/"02": Returns are considered only.

The PCE distinguishes the line items in a transaction based on the action code (*SaleReturnLineItem.actionCode*). The action code can have one of the following values:

- RETURN_ITEM: Return line item.
- SALE_ITEM: Sale line item.

If the action code of a line item is SALE_ITEM, the line item is treated as a sale by the PCE. If the action code is RETURN_ITEM, the line item is treated as a return.

The calculation base type (*PromotionConditionRuleSO.calculationBase*) is an additional attribute for transaction-related promotion price derivation rules that has to be taken into account. The subsequent values can be set for this attribute:

- CALCBASE_0/"00": Calculation base amount (*RetailPriceModifier.calculationBaseAmount*, *PriceModificationLineItem.calculationBaseAmount*, *FrequentShopperPointsModifier.computationBaseAmount*, or *LoyaltyRewardLineItem.computationBaseAmount*) is the sum of all sales in a transaction, sum over all returns in a return transaction.
- CALCBASE_01/"01": Calculation base amount is the sum of all sales minus the returns in a transaction.

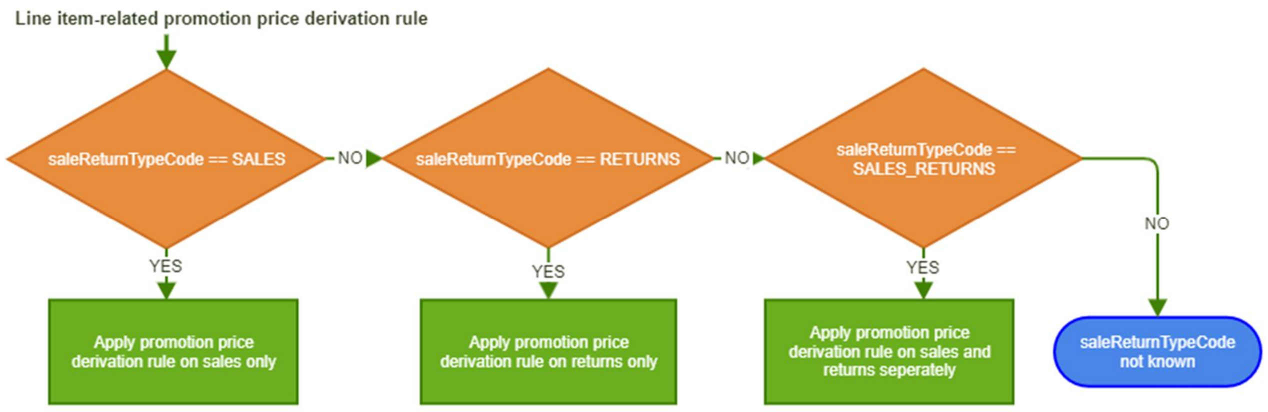
The behavior of the calculation base amount for sales returns is described in Set the Calculation Base Amount more detailed.

The following table summarizes the different combination possibilities of these attributes. Thereby, the columns "Line item-related promotion price derivation rules", "General transaction", and "Return transaction" states whether sales, returns, or both are considered during the calculation of the benefit. The two latter columns describe this for transaction-related promotion price derivation rules. The transaction category (*TransactionCategory.Key.transactionCategoryCode*) is provided in the PCE-request. Thereby, the return transaction (transaction category is "RETURN") includes only returns as line items. In contrast, the general transaction has any type of transaction category – except "RETURN".

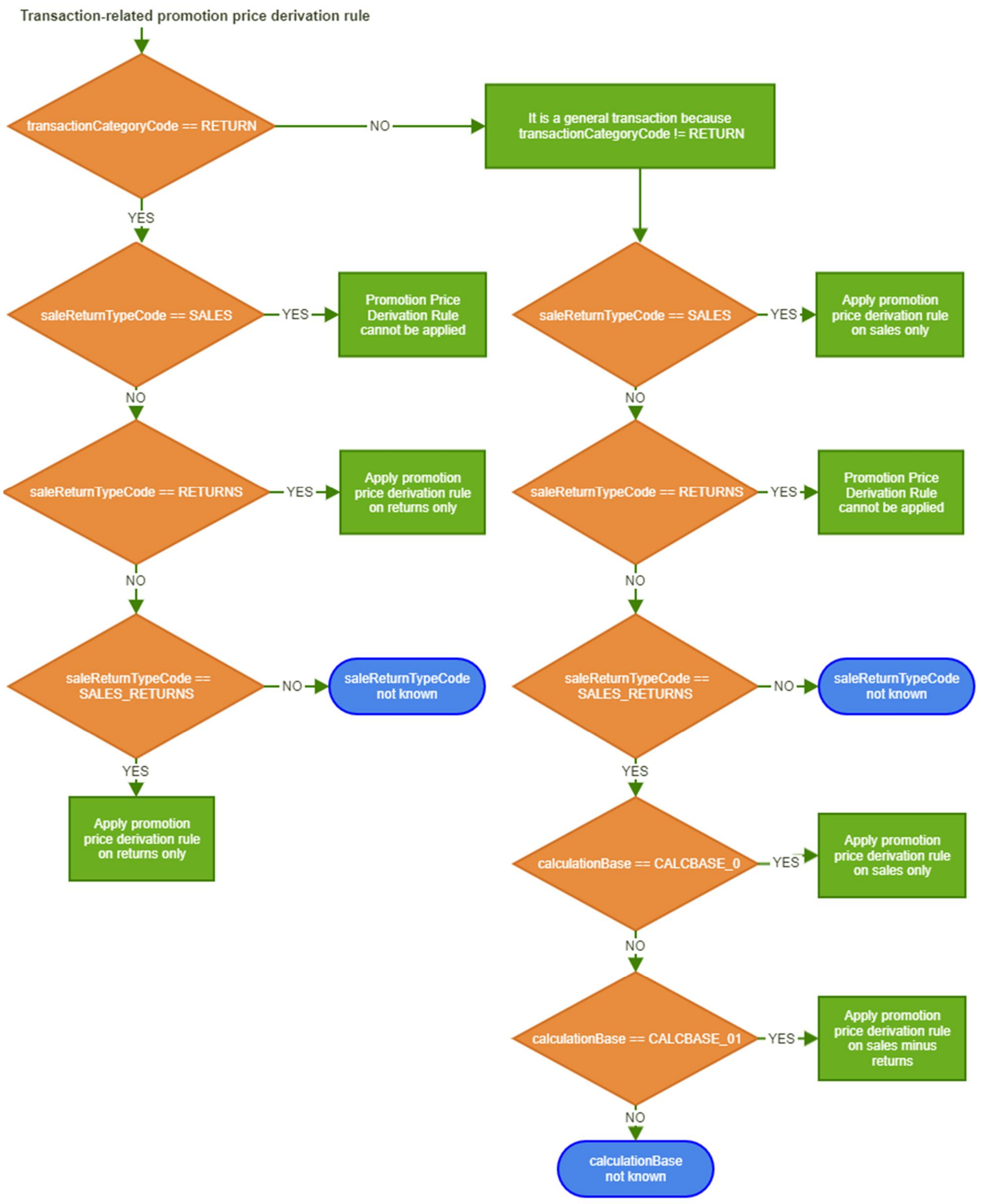
The yellow background color indicates that the calculation base amount for sales returns does not influence the behavior of the PCE.

Sale Return Type	**Calculation Base Type**	**Line Item-Related Promotion Price Derivation Rule**	**Transaction-Related Promotion Price Derivation Rules**	
			General Transaction	**Return Transaction**
SALES / "01"	CALCBASE_0/"00" or **null**	sold line items only (sales)	sales only	—
SALES / "01"	CALCBASE_01/"01"	sold line items only (sales)	sales only	—
RETURNS / "02"	CALCBASE_0/"00" or **null**	returned goods only (returns)	—	returns only
RETURNS / "02"	CALCBASE_01/"01"	returned goods only (returns)	—	returns only
SALES_RETURNS / "00" or **null**	CALCBASE_0/"00" or **null**	sales and returns separately	sales only	returns only
SALES_RETURNS / "00" or **null**	CALCBASE_01/"01"	sales and returns separately	sales minus returns	returns only

The following figure gives an overview about the behavior of the PCE in the case of a line item-related promotion price derivation rule. The calculation base type does not influence the behavior of the PCE, so it is not included in this decision tree.


Line item-related promotion price derivation rule
saleReturnTypeCode == SALES
NO
saleReturnTypeCode == RETURNS
NO
saleReturnTypeCode == SALES_RETURNS
NO
YES
YES
YES
Apply promotion price derivation rule on sales only
Apply promotion price derivation rule on returns only
Apply promotion price derivation rule on sales and returns seperately
saleReturnTypeCode not known

The following figure gives an overview about the behavior of the PCE in case of a transaction-related promotion price derivation rule.


Transaction-related promotion price derivation rule
transactionCategoryCode == RETURN
NO
It is a general transaction because transactionCategoryCode != RETURN
YES
saleReturnTypeCode == SALES
YES
Promotion Price Derivation Rule cannot be applied
NO
saleReturnTypeCode == RETURNS
YES
Apply promotion price derivation rule on returns only
NO
saleReturnTypeCode == SALES_RETURNS
NO
saleReturnTypeCode not known
YES
Apply promotion price derivation rule on returns only
saleReturnTypeCode == SALES
YES
Apply promotion price derivation rule on sales only
NO
saleReturnTypeCode == RETURNS
YES
Promotion Price Derivation Rule cannot be applied
NO
saleReturnTypeCode == SALES_RETURNS
NO
saleReturnTypeCode not known
YES
calculationBase == CALCBASE_0
YES
Apply promotion price derivation rule on sales only
NO
calculationBase == CALCBASE_01
YES
Apply promotion price derivation rule on sales minus returns
NO
calculationBase not known

THERE IS MORE

The promotion price derivation rules that contain a shopping basket eligibility are handled differently. This means that the shopping basket threshold (*MarketBasketAmountEligibilitySO.marketBasketThresholdAmount*) is always checked against the total amount of the transaction – sales minus returns. The case of sale return type SALES_RETURNS and the calculation base for sales returns CALCBASE_0 is computed either for sales only or for returns only, depending on the transaction control break code (*PromotionConditionRuleSO.transactionControlBreakCode*).

6.6.1 Activate Price Derivation Rule Eligibilities without the Original Transaction

It is crucial to separate the promotion calculation for sales and returns even though the promotion shall be applied to sales and returns. Especially, thresholds in the price derivation rule eligibility or required/limit quantities in the mix and match are of interest. The business case of returning goods always makes a retrograde calculation necessary for the trader's monetary gains and granted discounts. If the sales and returns are consolidated – unintentionally – during the promotion calculation within one purchase, this retrograde calculation would be wrong.

HOW IT WORKS

The price derivation rule eligibilities are evaluated separately for sales and returns. As a result, the price derivation rule eligibilities may be activated once by the sales, once by the returns, and once by sales minus returns in the transaction. The latter case is needed for transaction-related promotion price derivation rules where the calculation base amount shall be computed as sales minus returns. This is the only case where the retrograde calculation of a transaction and thus, a purchase, is not properly executed, because in this case the price derivation rule eligibilities are evaluated not separately, but for sales and returns together, to use the difference as the calculation base amount.

If several line items are needed to fulfill the promotion price derivation rule eligibility, it is possible that a promotion can be triggered by a set of items, for examples items belonging to:

- A merchandise hierarchy group,
- A simple product group, or
- A product group.

The sale return type of each promotion price derivation rule is not known by the PCE when the price derivation rule eligibilities are loaded and activated. This also applies to the transaction control break code(*PromotionConditionRuleSO.transactionControlBreakCode*), the calculation base type, the calculation base sequence (*PromotionConditionRuleSO.calculationBaseSequence*), the consider predecessors flag (*PromotionConditionRuleSO.considerPreviousPromotionConditionFlag*), and the no effect on successors flag (*PromotionConditionRuleSO.noEffectOnSubsequentPromotionConditionFlag*). This is due to the fact that these are part of the price derivation rule which is only loaded for the already activated price derivation rule eligibilities. As a result, the PCE activates the price derivation rule eligibility three times, once for the sales only, once for the returns only, and once for sales minus returns if possible.

The item, the merchandise category, the simple product group, and the product group eligibility are evaluated like described in Activate a Threshold Eligibility. The quantities and/or amounts are compared on the level of the parent eligibility for a simple product group eligibility only. In all other cases, the child eligibilities are activated first.

6.6.1.1 Activate a Threshold Eligibility

If a threshold type (*ThresholdType*) is also part of the price derivation rule eligibility, it may define thresholds, limits, and intervals for the quantity and/or amount of a line item and line item groups, respectively. A price derivation rule eligibility including a threshold type is called threshold eligibility (*ThresholdPromotionConditionEligibility*).

The activation of a threshold eligibility with a defined interval is handled in the same way as without the defined interval. This is due to the fact that the quantity interval (*ThresholdPromotionConditionEligibility.intervalQuantity*) or amount interval (*ThresholdPromotionConditionEligibility.intervalAmount*) has only an impact on the calculation of the benefit and controls the ranges between the threshold and the limit. Hence, the following subsection only describes the activation of the threshold eligibility without considering the intervals since the behavior of the PCE is equivalent in these cases.

6.6.1.1.1 Thresholds and Limits for the Quantity

The threshold type *QUT* is only triggered by quantities of items that are part of the possible triggers. Possible triggers are: items, merchandise categories, simple product groups, and product groups. A quantity threshold (*ThresholdPromotionConditionEligibility.thresholdQuantity*) and a quantity limit (*ThresholdPromotionConditionEligibility.limitQuantity*) are set in the price derivation rule eligibility. To activate the price derivation rule eligibility, the transaction has to include the trigger with the correct quantities.

The following three checks are executed by the PCE for the eligibility activation:

1. Checking whether the sales fulfill the threshold eligibility
2. Checking whether the returns fulfill the threshold eligibility
3. Checking whether the sales minus the returns fulfill the threshold eligibility

Example

Let us assume that there exists the item "Shirt" for 19.99€.

We further assume that there is a promotion price derivation rule with a threshold eligibility that has the quantity threshold 2 and the quantity limit 4.

If a customer buys a "Shirt" 3 times, the threshold eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✓ (3 + 0 = 3)

If a customer returns a "Shirt" 3 times, the threshold eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✓	✗ (0 + (-3) = (-3))

If a customer buys a "Shirt" 2 times and returns it once, the threshold eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✗ (2 + (-1) = 1)

If a customer buys a "Shirt" 7 times and returns it 2 times, the threshold eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✓ (7 + (-2) = 5)

If a customer buys a "Shirt" once and returns it once, the threshold eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✗	✗ (1 + (-1) = 0)

If a customer buys a "Shirt" 2 times and returns it 2 times, the threshold eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✗ (2 + (-2) = 0)

6.6.1.1.2 Thresholds and Limits for the Amount

To use thresholds and limits for the amount, the threshold type has to be AMT. This threshold type is only triggered by amounts of items that are part of the possible triggers. Possible triggers are: items, merchandise categories, simple product groups, and product groups.

An amount threshold (*ThresholdPromotionConditionEligibility.thresholdAmount*) and an amount limit (*ThresholdPromotionConditionEligibility.limitAmount*) are set in the price derivation rule eligibility. The transaction is compared with the price derivation rule eligibility. To activate the price derivation rule eligibility, the transaction has to include the trigger with the correct amounts. For this purpose, the corresponding amount of the calculation base amount of the promotion price derivation rule (possibly reduced by the already granted monetary discount) is checked against these limits instead of the original amount.

The following three checks are executed by the PCE for the eligibility activation:

1. Checking whether the sales fulfill the threshold eligibility

2. Checking whether the returns fulfill the threshold eligibility
3. Checking whether the sales minus the returns fulfill the threshold eligibility

Example

Let us assume that there is an item "Integrated Circuit" for a regular price of 1.99€ per piece. We further assume that there is a promotion price derivation rule with a threshold eligibility that contains the amount threshold of 2.00€ and the amount limit 30.00€.

If a customer buys 5 integrated circuits for the production of 5 boards, the transaction has a regular total amount of 9.95€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✓

If a customer returns 5 integrated circuits, the regular total amount would be -9.95€.

Active for:	Sales	Returns	Sales minus Returns
	✗	✓	✗

If a customer buys 2 integrated circuits and returns 1, the regular total amount would be 1.99€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✗

If a customer buys 20 integrated circuits and returns 18, the regular total amount would be 3.98€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✓

If a customer buys 1 integrated circuits and returns one, the regular total amount would be 0.0€.

Active for:	Sales	Returns	Sales minus Returns
	✗	✗	✗

If a customer buys 3 integrated circuits and returns 2, the regular total amount would be 1.99€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✗

6.6.1.1.3 Threshold Types for Single Items

The threshold for single item flag (*MHGPromotionConditionEligibilitySO.thresholdForSingleItemFlag* or *MerchandiseSetPromotionConditionEligibilitySO.isMixingForbidden*) is not checked during eligibility loading and activation.

Instead, the PCE takes care to only apply the promotion price derivation rule on the same line item during the promotion calculation.

Different Scenarios are explained in the chapter Use Thresholds, Intervals, and Limits.

6.6.1.2 Activate a Shopping Basket Eligibility

In order to activate the shopping basket eligibility (*MarketBasketAmountEligibilitySO*), the entire transaction is taken into account and separated into sales and returns. Thereby, the regular sales prices are used to determine the calculation base amount in advance for the activation of the price derivation rule eligibility. In a later step, the calculation base amount is recomputed and compared again with the shopping basket threshold. If the recomputed calculation base amount still satisfies the shopping basket eligibility, the PCE applies the benefit. Otherwise the benefit is not applied.

The following three checks are executed by the PCE for the eligibility activation:

1. Checking whether the sales fulfill the shopping basket eligibility
2. Checking whether the returns fulfill the shopping basket eligibility
3. Checking whether the sales minus the returns fulfill the shopping basket eligibility

Afterwards, the check is done again when all the influencing attributes and parameters are known for the calculation base amount determination.

Example

Let us assume that there is an item "Integrated Circuit" for a regular price of 1.99€ per piece.
We further assume that there is a promotion price derivation rule with a shopping basket eligibility that contains the shopping basket threshold of 5000.00€.
If a customer buys 3000 integrated circuits for the production of 3000 boards, the transaction has a regular total amount of 5970.00€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✓

If a customer returns 3000 integrated circuits, the regular total amount would be -5970.00€.

Active for:	Sales	Returns	Sales minus Returns
	✗	✓	✗

If a customer buys 2513 integrated circuits and returns 2, the regular total amount would be 4996.89€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✗

If a customer buys 5200 integrated circuits and returns 2513, the regular total amount would be 5347.13€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✓

If a customer buys 200 integrated circuits, the regular total amount would be 398.00€.

Active for:	Sales	Returns	Sales minus Returns
	✗	✗	✗

If a customer buys 2700 integrated circuits and returns 2600, the regular total amount would be 199.00€.

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✗

6.6.1.3 Activate a Customer Group Eligibility

A customer group eligibility is activated if the customer group identifier in the price derivation rule eligibility is part of the transaction. For this purpose, the customer group identifier is compared with the data of the loaded price derivation rule eligibilities. If the customer group eligibility is active, the corresponding price derivation rule is executed in order to apply the benefit.

The following three checks are executed by the PCE for the eligibility activation:

1. Checking whether the sales fulfill the customer group eligibility
2. Checking whether the returns fulfill the customer group eligibility
3. Checking whether the sales minus the returns fulfill the customer group eligibility

The customer group eligibility can be activated for sales, for returns, and for sales minus returns. This price derivation rule eligibility is activated for sales minus returns if it is active for the sales or the returns.

Example

Let us assume that there are two items: the "Kitchen Chair" for 79.95€ and the "Office Chair" for 99.95€.

We further assume that there is a promotion price derivation rule with a customer group eligibility that contains the customer group identifier "Office Worker".

If a customer buys a "Kitchen Chair" and an "Office Chair" and the customer group identifier "Office Worker" is registered, the customer group eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✓

If a customer returns a "Kitchen Chair" and returns an "Office Chair" and the customer group identifier "Office Worker" is registered, the customer group eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✓	✓

If a customer buys a "Kitchen Chair" and returns an "Office Chair" and the customer group identifier "Office Worker" is registered, the customer group eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✓

If a customer buys a "Kitchen Chair" and returns an "Office Chair" and the customer group identifier "Office Worker" is not registered, the customer group eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✗	✗

6.6.1.4 Activate a Coupon Eligibility

To apply a benefit triggered by a coupon, the coupon eligibility*(CouponPromotionConditionEligibilitySO)* has to be part of the promotion price derivation rule. The transaction includes a coupon with the corresponding coupon identifier (*RetailTransactionCouponSummary.Key.couponNumber*) and an input count (*RetailTransactionCouponSummary.inputCount*) that is greater than zero.

A coupon eligibility is activated if the corresponding coupon is part of the transaction. For this purpose, the coupon identifier is compared with the data of the loaded coupon eligibility. If the coupon eligibility is active, the corresponding price derivation rule is executed in order to apply the benefit.

The following three checks are executed by the PCE for the eligibility activation:

1. Checking whether the sales fulfill the coupon eligibility
2. Checking whether the returns fulfill the coupon eligibility
3. Checking whether the sales minus the returns fulfill the coupon eligibility

The coupon eligibility can be activated for sales, for returns, and for sales minus returns. It is possible to activate this price derivation rule eligibility for returns also even though a coupon is typically not brought back, too. The activation for the return is only done if the coupon is brought back in addition.

Example

Let us assume that there is the following item in the master data: "Vase" for 10.10€

We also assume that there is the following promotion price derivation rule in the promotion master data: get 0.20€ off if a coupon #1 is registered.

If a customer buys a "Vase" and the coupon #1 is registered, the coupon eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✓

If a customer returns a "Vase" and the coupon #1 is registered, the coupon eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✓	✓

If a customer returns a "Vase" and the coupon #1 is not registered, the coupon eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✗	✗

If a customer buys a "Vase" and returns a "Vase" and the coupon #1 is registered, the coupon eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✓

6.6.1.5 Activate a Combination Eligibility

A combination eligibility (*CombinationPromotionConditionEligibilitySO*) – parent and child eligibilities (*CombinationPromotionConditionEligibilitySO.childEligibilityList*) – is activated if the corresponding combination is part of the transaction. For this purpose, the child eligibilities are evaluated and activated first like described in this document.

The following three checks are executed by the PCE for the activation of the child eligibilities:

1. Checking whether the sales fulfill the price derivation rule eligibility
2. Checking whether the returns fulfill the price derivation rule eligibility
3. Checking whether the sales minus the returns fulfill the price derivation rule eligibility

It depends on the combination code (*CombinationPromotionConditionEligibilitySO.combinationCode*) and the activation of the child eligibility whether the parent eligibility is active for sales, for returns, or for sales minus returns.

6.6.1.5.1 Combine Triggers with a Logical AND

The transaction has to include all triggers of child eligibilities of a parent eligibility with a logical AND operator as combination code. If only one trigger type is not part of the transaction, the PCE does not activate the parent eligibility. Thus, no further calculations are executed.

The parent eligibility is activated like described in the following table:

Active for:	Sales for one child eligibility	Sales for another child eligibility	Sales for parent eligibility
	✓	✓	✓
	✓	✗	✗
	✗	✓	✗
	✗	✗	✗
	Returns for one child eligibility	Returns for another child eligibility	Returns for parent eligibility
	✓	✓	✓
	✓	✗	✗
	✗	✓	✗
	✗	✗	✗
	Sales minus returns for one child eligibility	Sales minus returns for another child eligibility	Sales minus returns for parent eligibility
	✓	✓	✓
	✓	✗	✗
	✗	✓	✗
	✗	✗	✗

6.6.1.5.2 Combine Triggers with a Logical OR

The transaction has to include at least one trigger of a child eligibility of a parent eligibility with a logical OR operator as combination code. If no trigger type is part of the transaction, the PCE does not activate the parent eligibility. Thus, no further calculations are executed.

A price derivation rule eligibility which includes a logical OR operator is internally disassembled into several price derivation rule eligibilities without a logical OR operator.

Active for:	Sales for one child eligibility	Sales for another child eligibility	Sales for parent eligibility
	✓	✓	✓
	✓	✗	✓
	✗	✓	✓
	✗	✗	✗
	Returns for one child eligibility	Returns for another child eligibility	Returns for parent eligibility
	✓	✓	✓
	✓	✗	✓
	✗	✓	✓
	✗	✗	✗
	Sales minus returns for one child eligibility	Sales minus returns for another child eligibility	Sales minus returns for parent eligibility
	✓	✓	✓
	✓	✗	✓
	✗	✓	✓
	✗	✗	✗

6.6.1.6 Activate a Manual Eligibility

The PCE gets a so called manual trigger (*ManualPromotionTrigger*) in the PCE-request. This manual trigger has to correspond to a price derivation rule eligibility in the promotion master data – herein manual eligibility (*ManualPromotionConditionEligibilitySO*). The manual eligibility always contains the trigger type (*ManualPromotionConditionEligibilitySO.triggerType*) and the trigger value (*ManualPromotionConditionEligibilitySO.triggerValue*). Both attributes can be set by the retailer since they are used to distinguish between different possible discounts.

To apply a manual benefit, the transaction includes a manual trigger with the following attributes:

- For line item-related promotion price derivation rules:
  - Item level trigger type *(SaleReturnLineItemPromotionTrigger.triggerType*)
  - Item level trigger value *(SaleReturnLineItemPromotionTrigger.triggerValue)*

  - Item level trigger sequence number*(SaleReturnLineItemPromotionTrigger.Key.triggerSequenceNumber)*
  - Item level trigger sequence addend*(SaleReturnLineItemPromotionTrigger.triggerSequenceAddend)*

- For transaction-related promotion price derivation rules:
  - Transaction level trigger type *(RetailTransactionPromotionTrigger.triggerType)*
  - Transaction level trigger value *(RetailTransactionPromotionTrigger.triggerValue)*
  - Transaction level trigger sequence number*(RetailTransactionPromotionTrigger.Key.triggerSequenceNumber)*
  - Transaction level trigger sequence addend*(RetailTransactionPromotionTrigger.triggerSequenceAddend)*

The following three checks are executed by the PCE for the eligibility activation:

1. Checking whether the sales fulfill the manual eligibility
2. Checking whether the returns fulfill the manual eligibility
3. Checking whether the sales minus the returns fulfill the manual eligibility

The manual eligibility can be activated for sales or for returns, but not for sales minus returns if the manual trigger is provided on line item level. The latter case cannot occur since the manual eligibility can only be part of line item-related promotion price derivation rules.

The manual eligibility can be activated for sales, for returns, or for sales minus returns if the manual trigger is provided on transaction level. This price derivation rule eligibility is activated for sales minus returns if it is active for the sales or the returns.

Example: Line Item Level

Let us assume that there exists the item "Shirt" for 19.99€.

We further assume that there is a promotion price derivation rule with a manual eligibility that has a trigger type and a trigger value. The trigger type could have any name and any chosen trigger value. In this example, we assume the name of the trigger type is "CO" and the trigger value is 1007.

If a customer buys a "Shirt" and a manual trigger is added on line item level with the trigger type "CO" and the trigger value 1007, the manual eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✗

If a customer returns a "Shirt" and a manual trigger is added on line item level with the trigger type "CO" and the trigger value 1007, the manual eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✓	✗

If a customer buys a "Shirt" and returns a "Shirt" and each has a manual trigger attached on line item level with the trigger type "CO" and the trigger value 1007, the manual eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✗

If a customer buys a "Shirt" and returns a "Shirt" and no manual trigger is added on line item level, the manual eligibility is not activated:

Active for:	Sales	Returns	Sales minus Returns
	✗	✗	✗

Example: Transaction Level

Let us assume that there exists the item "Shirt" for 19.99€.

We further assume that there is a promotion price derivation rule with a manual eligibility that has the trigger type "OC" and the trigger value 1009.

If a customer buys a "Shirt" and a manual trigger is added on transaction level with the trigger type "OC" and the trigger value 1009, the manual eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✗	✓

If a customer returns a "Shirt" and a manual trigger is added on transaction level with the trigger type "OC" and the trigger value 1009, the manual eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✗	✓	✓

If a customer buys a "Shirt" and returns a "Shirt" and a manual trigger is added on transaction level with the trigger type "OC" and the trigger value 1009, the manual eligibility is activated like follows:

Active for:	Sales	Returns	Sales minus Returns
	✓	✓	✓

If a customer buys a "Shirt" and returns a "Shirt" and no manual trigger is added on transaction level, the manual eligibility is not activated:

Active for:	Sales	Returns	Sales minus Returns
	✗	✗	✗

6.6.1.7 THERE IS MORE

The application of a benefit to sold and/or returned items is described in Apply a Benefit with Respect to Sales and Returns. Especially, the behavior is explained with respect to the sale return type.

Related topics are:

- Activate Price Derivation Rule Eligibilities without the Original Transaction
- Handle Collisions with Respect to Sales and Returns
- Choose Items with Respect to Sales and Returns
- Compute Total Price Modification Methods with Respect to Sales and Returns
- Mix and Match with Respect to Sales and Returns
- Select the Method to Discount

6.6.2 Apply a Benefit with Respect to Sales and Returns

The PCE applies benefits according to the sale return type and the calculation base type. Thereby, the subsequent outcomes are possible:

1. Apply a benefit to sold line items and goods returned
2. Apply a benefit to sold line items only
3. Apply a benefit to goods returned only

6.6.2.1 Apply a Benefit to Sold Line Items and Goods Returned

The sale return type is null or has the value SALES_RETURNS. The promotion price derivation rule is applied to both, i.e. sold line items and goods returned. In the case of line item-related promotion price derivation rules, the benefit is applied to all eligible sold and returned line items. In the case of transaction-related promotion price derivation rules, the calculation base type has to be checked first. The behavior after the check is described in Set the Calculation Base Amount. Please notice, that this attribute is only relevant for transaction-related promotion price derivation rules.

Example: Apply a Benefit to Sold Line Items and Goods Returned

Example of line item-related promotion price derivation rules:

A customer has bought a sweater of size 40 and for the regular sales unit price of 20.00€. Now, the customer wants to return the sweater of size 40 and buys the same sweater of size 42. There is a promotion valid and thus, is going to be applied to the transaction. This promotion applies a 10% discount to sold line items and to goods returned.

As a result, the customer has not to pay anything and the operator has not to pay out the customer.

	Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount**	**Effective Sales Price**	
		Sweater	1	20.00€	2.00€	18.00€	
	Sweater		1	20.00€	2.00€	18.00€	
Regular Total Amount							0.00€

Example of transaction-related promotion price derivation rules with calculation base type CALCBASE_0:

A customer has bought 5 sweaters of size 40. The regular sales unit price for one sweater is 27.27€. The customer paid 136.35€. Now, the customer wants to return 5 sweaters of size 40 and buy 2 sweaters of size 42. There is a promotion valid that grants a 10% discount on the transaction. The promotion has the sale return type SALES_RETURNS and the calculation base type is set to CALCBASE_0 (sales only).

As a result, the customer has not to pay anything and the operator has to pay out 87.26 € to the customer.

	Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount**	**Effective Sales Price**	
		Sweater	5	136.35€	0.00€	136.35€	
	Sweater		2	54.54€	5.45€	49.09€	
Regular Total Amount							-81.81€
Discount (subtracted from regular total amount)							5.45€
Effective Total Amount							-87.26€

Example of transaction-related promotion price derivation rules with calculation base type CALCBASE_01:

A customer has bought 5 sweaters of size 40. The regular sales unit price for one sweater is 27.27€. The customer paid 136.35€. Now, the customer wants to return 5 sweaters of size 40

and buy 2 sweaters of size 42. There is a promotion valid that grants a 10% discount on the transaction. The promotion has the sale return type SALES_RETURNS and the calculation base type is set to CALCBASE_01 (sales minus returns).
As a result, the customer has not to pay anything and the operator has to pay out 73.63€ to the customer.

	Sales	Return	Quantity	Regular Sales Price	Discount	Effective Sales Price	
		Sweater	5	136.35€	13.36€	122.72€	
	Sweater		2	54.54€	5.45€	49.09€	
Regular Total Amount							-81.81€
Discount (subtracted from regular total amount)							-8.18€
Effective Total Amount							-73.63€

6.6.2.2 Apply a Benefit to Sold Line Items Only

The sale return type is SALES. The promotion price derivation rule is applied only to line items that are sold and thus, have the action code SALE_ITEM. In the case of transaction-related promotion price derivation rules, the calculation base type has no relevance. Meaning that it proceeds as if the calculation base type s is null or CALCBASE_0. In addition, the transaction category is not "RETURN" or null.

Example: Apply a Benefit to Sold Line Items Only

Example of line item-related promotion price derivation rules:
A customer has bought a sweater of size 40 and for the regular sales unit price of 20.00€. Now, the customer wants to return the sweater of size 40 and buys the same sweater of size 42. There is a promotion valid and thus, is going to be applied to the transaction. This promotion applies a 10% discount to sold line items only.
As a result, the customer has not to pay anything and the operator has to pay out 2.00€ to the customer.

	Sales	Return	Quantity	Regular Sales Price	Discount	Effective Sales Price	
		Sweater	1	20.00€		20.00€	

	Sweater		1	20.00€	2.00€	18.00€	
Regular Total Amount							-2.00€

Example of transaction-related promotion price derivation rules:

A customer has bought 5 sweaters of size 40. The regular sales unit price for one sweater is 27.27€. The customer paid 136.35€. Now, the customer wants to return 5 sweaters of size 40 and buy 2 sweaters of size 42. There is a promotion valid that grants a 20% discount on the transaction. The promotion has the sale return type SALES. The transaction category is not "RETURN" or null.

As a result, the customer has not to pay anything and the operator has to pay out 92.72€ to the customer.

	Sales	Return	Quantity	Regular Sales Price	Discount	Effective Sales Price	
		Sweater	5	136.35€		136.35€	
	Sweater		2	54.54€	10.91€	43.63€	
Regular Total Amount							-81.81€
Discount (subtracted from regular total amount)							10.91€
Effective Total Amount							-92.72€

6.6.2.3 Apply a Benefit to Goods Returned Only

The sale return type is RETURNS. The promotion price derivation rule is applied only to goods returned and thus, have the action code RETURN_ITEM.

In the case of transaction-related promotion price derivation rules, the calculation base type has no relevance. Meaning that it proceeds as if the calculation base type is CALCBASE_01. In addition, the transaction category must be "RETURN" and the transaction does not include any sold line items.

Example: Apply a Benefit to Goods Returned Only

Example of line item-related promotion price derivation rules:

A customer has bought a sweater of size 40 and for the regular sales unit price of 20.00€. Now, the customer wants to return the sweater of size 40 and buys the same sweater of size 42. There is a promotion valid and thus, is going to be applied to the transaction. This promotion applies a 10% discount to goods returned only.

As a result, the customer has to pay 2.00€ and the operator has not to pay out the customer.

	Sales	Return	Quantity	Regular Sales Price	Discount	Effective Sales Price	
		Sweater	1	20.00€	2.00€	18.00€	
	Sweater		1	20.00€		20.00€	
Regular Total Amount							2.00€

<u>Example of transaction-related promotion price derivation rules:</u>

A customer has bought 5 sweaters of size 40. The regular sales unit price for one sweater is 27.27€. The customer paid 136.35€. Now, the customer wants to return 5 sweaters of size 40. There is a promotion valid that grants a 30% discount on the transaction. The promotion has the sale return type RETURNS. The transaction category must be "RETURN" and the transaction does not include any sold line items.

As a result, the customer has not to pay anything and the operator has to pay out 95.44€ to the customer.

	Sales	Return	Quantity	Regular Sales Price	Discount	Effective Sales Price	
		Sweater	5	136.35€	40.91€	95.44€	
Regular Total Amount							-136.35€
Discount (subtracted from regular total amount)							-40.91€
Effective Total Amount							-95.44€

6.6.2.4 THERE IS MORE

Related topics to the handling of the sales and the returns are:

- Activate Price Derivation Rule Eligibilities with Respect to Sales and Returns
- Handle Sales and Returns without the Original Transaction
- Handle Collisions with Respect to Sales and Returns
- Choose Items with Respect to Sales and Returns
- Compute Total Price Modification Methods with Respect to Sales and Returns
- Mix and Match with Respect to Sales and Returns
- Select the Method to Discount

The promotion price derivation rules that contain a shopping basket eligibility are not handled differently. This means that the shopping basket threshold

(*MarketBasketAmountEligibilitySO.marketBasketThresholdAmount*) is checked against sales or returns or sales minus returns.

6.6.3 Handle Collisions with Respect to Sales and Returns

Several promotion price derivation rules collide if the following conditions are fulfilled at the same time:

1. Equivalent transaction control (*PromotionConditionRuleSO.transactionControlBreakCode*),
2. Equivalent sequence (*PromotionConditionSO.sequence*),
3. Equivalent resolution (*PromotionConditionSO.resolution*), and
4. Applicable to the equivalent line items.

HOW IT WORKS

A line item can either be a return or a sales. The action code (*SaleReturnLineItem.actionCode*) decides whether a line item is a return or a sale. In case of a return the action code is RETURN_ITEM and for a sale the action code is SALE_ITEM.

If the colliding promotion price derivation rules are applicable to a collection of line items, this collection cannot include line items with different action codes. This is also the case for the sale return type (*PromotionConditionSO.saleReturnTypeCode*) SALES_RETURNS. As a result, the promotion price derivation rules still collide, but once for the sales and once for the returns. It is assumed that both types (sales, returns) are triggering the colliding promotion price derivation rules independently of each other. An exception exist for transaction-related promotion price derivation rules and the calculation base for sales returns (*PromotionConditionRuleSO.calculationBase*) CALCBASE_01. The sales and the returns are considered as one collection and thus, the sales are charged against the returns.

Examples are provided for the different behaviors of the PCE in the subsequent sections.

6.6.3.1 Colliding Promotions Only for Sales

Example: Line Item-Related Promotion Price Derivation Rules

A customer has bought a sweater of size 40 and for the regular sales unit price of 20.00€. Now, the customer wants to return the sweater of size 40 and buys the same sweater of size 42. There are two promotions valid and thus, are going to be applied to the transaction. The first promotion price derivation rule applies a 5% discount to sold line items and to goods returned. The second promotion price derivation rule grants a 10% discount to sold line items only. Both have the same sequence and resolution. The two promotion price derivation rules are handled as if these are three separate ones in order to avoid charging the returned line items against the sold line items. This is due to the fact that the first promotion price derivation rule is applicable to sales and returns. The collision handling is executed once for the sold line items and not for the goods returned. In case of returns, no collision occurs.

1. First promotion price derivation rule is applied on sold line items ⚠ and collides with the third option (see 3. below).
2. First promotion price derivation rule is applied on returned line items ✓ and does not collide with any other promotion price derivation rule.
3. Second promotion price derivation rule is applied on sold line items ⚠ and collides with the first option (see 1. above).

As a result, the customer does not have to pay anything and the operator has to pay out 1.00€ to the customer.

Sales	Return	Quantity	Regular Sales Price	Discount of 1st Promotion	Discount of 2nd Promotion	Effective Sales Price	
	Sweater	1	20.00€	1.00€ (5% discount)		19.00€	
Sweater		1	20.00€		2.00€ (10% discount as best price)	18.00€	
Regular Total Amount							-1.00€

Example: Transaction-Related Promotion Price Derivation Rules

A customer has bought 2 sweaters of size 40. The regular sales unit price of one sweater is 27.27€. The customer paid 54.54€. Now, the customer wants to return 2 sweaters of size 40 and buy 3 sweaters of size 42. The transaction is a general transaction, the transaction category (*TransactionCategory.Key.transactionCategoryCode*) is not "RETURN". There are promotion price derivation rules valid and thus, are going to be applied to the transaction.

1. Promotion: grants 15% discount for the receipt. The price derivation rule has the sale return type SALES. The transaction category must not be "RETURN".
2. Promotion: grants 20% discount for the receipt. The price derivation rule has the sale return type SALES_RETURNS. The calculation base for sales returns is CALCBASE_01 (sales minus returns).

The two promotion price derivation rules are handled as if these are two ones since the first one only applies on sold items and the second one has the calculation base for sales returns CALCBASE_01 which does not require sold and returned line items to be handled separately. The promotion price derivation rules collide due to both being applied on the sold items and 1sr one win.

As a result, the customer has to pay 15.00€.

	Sales	Return	Quantity	Regular Sales Price	Discount of 1st Promotion	Discount of 2nd Promotion	Effective Sales Price	
		Sweater	2	54.54€			54.54€	

	Sweater		3	81.81€	12.27€		69.54€	
Regular Total Amount								27.27€
Discount of 1st Promotion (subtracted from regular total amount)								12.27€
Effective Total Amount								15.00€

6.6.3.2 Colliding Promotions Only for Returns

Example: Line Item-Related Promotion Price Derivation Rules

A customer has bought a sweater of size 40 and for the regular sales unit price of 20.00€. Now, the customer wants to return the sweater of size 40 and buys the same sweater of size 42. There are two promotions valid and thus, are going to be applied to the transaction. The first promotion price derivation rule applies a 5% discount to sold line items and to goods returned. The second promotion price derivation rule grants a 10% discount to returned line items only. Both have the same sequence and resolution. The two promotion price derivation rules are handled as if these are three separate ones in order to avoid charging the returned line items against the sold line items. This is due to the fact that the first promotion price derivation rule is applicable to sales and returns. The collision handling is executed once for the goods returned and not for the sales. In case of sales, no collision occurs.

1. First promotion price derivation rule is applied on sold line items ✓ and no collision occurs.
2. First promotion price derivation rule is applied on returned line items ⚠ and collides with the third option (see below).
3. Second promotion price derivation rule applied on returned line items ⚠ and collides with the second option (see above).

As a result, the customer has to pay 1.00€ for the transaction.

Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount of 1st Promotion**	**Discount of 2nd Promotion**	**Effective Sales Price**	
	Sweater	1	20.00€		2.00€ (10% discount as best price)	18.00€	
Sweater		1	20.00€	1.00€ (5% discount)		19.00€	
Regular Total Amount							1.00€

Example: Transaction-Related Promotion Price Derivation Rules

A customer has bought 2 sweaters of size 40. The regular sales unit price of one sweater is 27.27€. The customer paid 54.54€. Now, the customer wants to return 2 sweaters of size 40 and buy 3 sweaters of size 42. The transaction is a return transaction, the transaction category is "RETURN". There are promotions valid and thus, are going to be applied to the transaction.

1. promotion: grants 15% discount for the receipt. The promotion has the sale return type RETURNS. The transaction category must be "RETURN".
2. promotion: grants 20% discount for the receipt. The promotion has the sale return type SALES_RETURNS. The calculation base for sales returns is CALCBASE_0 (returns only in return transaction).

The two promotion price derivation rules are handled as if these are two ones since both apply only on returned items. The promotion price derivation rules are going to collide and 2nd one win.

As a result, the customer has to pay 38.18€.

	Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount of 1st Promotion**	**Discount of 2nd Promotion**	**Effective Sales Price**	
		Sweater	2	54.54€		10.91€	43.63€	
	Sweater		3	81.81€			81.81€	
Regular Total Amount								27.27€
Discount of 2nd Promotion (subtracted from regular total amount)								-10.91€
Effective Total Amount								38.18€

6.6.3.3 Colliding Promotions for Sales and Returns

Example: Line Item-Related Promotion Price Derivation Rules

A customer has bought a sweater of size 40 and for the regular sales unit price of 20.00€. Now, the customer wants to return the sweater of size 40 and buys the same sweater of size 42. There are three promotions valid and thus, are going to be applied to the transaction.

- The first promotion price derivation rule applies a 5% discount to sold line items and to goods returned.
- The second promotion price derivation rule grants a 10% discount to returned line items only.
- The third promotion price derivation rule grants a 15% discount to sold line items only.

All three have the same sequence and resolution. The three promotion price derivation rules are handled as if these are four separate ones in order to avoid charging the returned line items against the sold line items. This is due to the fact that the first promotion price derivation rule is applicable to sales and returns. The collision handling is executed once for the sold line items and once for the goods returned.

1. First promotion price derivation rule is applied on sold line items ⚠ and collides with the fourth option (see below).
2. First promotion price derivation rule is applied on returned line items ⚠ and collides wit the third option (see below).
3. Second promotion price derivation rule is applied on returned line items ⚠ and collides with the second option (see above).
4. Third promotion price derivation rule is applied on sold line items ⚠ and collides with the first option (see above).

As a result, the customer does not have to pay anything and the operator has to pay out 1.00€ to the customer.

Sales	Return	Quantity	Regular Sales Price	Discount of 1st Promotion	Discount of 2nd Promotion	Discount of 3rd Promotion	Effective Sales Price	
	Sweater	1	20.00€		2.00€ (10% discount as best price)		18.00€	
Sweater		1	20.00€			3.00€ (15% discount as best price)	17.00€	
Regular Total Amount								-1.00€

Example: Transaction-Related Promotion Price Derivation Rules

A customer has bought 2 sweaters of size 40. The regular sales unit price of one sweater is 27.27€. The customer paid 54.54€. Now, the customer wants to return 2 sweaters of size 40 and buy 3 sweaters of size 42. The transaction is a general transaction, the transaction category <u>is not</u> "RETURN". There are promotions valid and thus, are going to be applied to the transaction.

1. Promotion: grants 5% discount for the receipt. The promotion has the sale return type SALES_RETURN . The calculation base for sales returns is CALCBASE_0 (sales only in a general transaction).
2. Promotion: grants 20% discount for the receipt. The promotion has the sale return type SALES_RETURN. The calculation base for sales returns is CALCBASE_01 (sales minus returns).

The two promotion price derivation rules are handled as if these are two ones since the first is only applied on sold items while the second has the calculation base for sales returns CALCBASE_01 which does not require sold and returned items to be handled separately. The promotion price derivation rules collide due to both being applied on the sold items and 2nd one win.
As a result, the customer has to pay 21.82€.

	Sales	Return	Quantity	Regular Sales Price	Discount of 1st Promotion	Discount of 2nd Promotion	Effective Sales Price	
		Sweater	2	54.54€		10.91€	43.63€	
	Sweater		3	81.81€		16.36€	65.45€	
Regular Total Amount								27.27€
Discount of 2nd Promotion (subtracted from regular total amount)								5.45€
Effective Total Amount								21.82€

6.6.3.4 THERE IS MORE

A detailed description of the collision handling can be read in chapter Select a Promotion Price Derivation Rule in Case of a Collision.

Related topics to the handling of the sales and the returns are:

- 2019-10-29_13-35-47_Activate Price Derivation Rule Eligibilities with Respect to Sales and Returns
- Handle Sales and Returns without the Original Transaction
- Choose Items with Respect to Sales and Returns
- Compute Total Price Modification Methods with Respect to Sales and Returns
- Mix and Match with Respect to Sales and Returns
- Select the Method to Discount

6.6.4 Choose Items with Respect to Sales and Returns

The PCE chooses line items out of several eligible line items for a promotion if not all eligible ones shall get a benefit.

HOW IT WORKS

The line items are chosen with respect to the achievable benefit. The choose item method *(PromotionConditionRuleSO.chooseItemMethod)* LOWEST_FIRST leads to the smallest achievable benefit whereas the choose item method HIGHEST_FIRST leads to the highest achievable benefit. In case of sales and returns, the benefit is either subtracted from the sales price (sale) or added to the sales price (return). This influences the meaning of smallest or highest achievable benefit, too. The achievable benefit corresponds to the absolute value that is granted as benefit.

The PCE computes the benefit for the sales and for the returns separately and thus, chooses the line items separately as well. The sold line items are thereby considered first and afterwards the returned line items. An exception exist for transaction-related promotion price derivation rules and the calculation base for sales returns (*PromotionConditionRuleSO.calculationBase*) CALCBASE_01. The sales and the returns are considered as one collection and thus, the sales are charged against the returns. Thereby, the absolute values of the benefits are compared with each other to achieve the "highest" or "smallest" benefit, respectively.

Examples are provided for the different behaviors of the PCE in the subsequent sections.

6.6.4.1 Resulting in the Smallest Achievable Benefit

Example: Line Item-Related Promotion Price Derivation Rule

A customer has bought a sweater for the regular sales unit price of 20.00€. Now, the customer wants to return the sweater and buys a skirt for the regular sales unit price of 18.00€ and a T-shirt for the regular sales unit price of 15.00€. There is one promotion valid and thus, is going to be applied to the transaction.

- The promotion price derivation rule applies a 10% discount to sold and returned items of the merchandise category "clothes", to which all line items in the transaction belong.
- The promotion price derivation rule may apply to maximal 1 line item.
- The promotion has the sale return type (*PromotionConditionSO.saleReturnTypeCode*) SALES_RETURNS and the choose item method LOWEST_FIRST.

The promotion price derivation rule is applied as two separate promotion price derivation rules in order to avoid charging the sold line items against the returned line items:

1. Applied on sold line items only.
2. Applied on returned line items only.

As a result, the customer has to pay 13.50€.

Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount of Promotion**	**Effective Sales Price**	
	Sweater	1	20.00€	2.00€ (Promotion price derivation rule applied secondly for returns only and to max. 1 line item)	18.00€	
Skirt		1	18.00€		18.00€	

T-Shirt		1	15.00€	1.50€ (Promotion price derivation rule applied firstly for sales only and to max. 1 line item)	13.50€	
Regular Total Amount						13.50€

Example: Transaction-Related Promotion Price Derivation Rule

A customer has bought a sweater for the regular sales unit price 27.27€. Now, the customer wants to return the sweater and buy a T-shirt with a regular sales unit price of 30.00€ and a pair of jeans for 50.00€ . The transaction is a general transaction, the transaction category (*TransactionCategory.Key.transactionCategoryCode*) <u>is not</u> "RETURN". There is a promotion valid and thus, is going to be applied to the transaction.

- The promotion price derivation rule applies a 10% discount to sold and returned line items of the merchandise category "clothes", to which all line items in the transaction belong.
- The promotion price derivation rule has the sale return type SALES_RETURNS, the calculation base for sales returns is CALCBASE_0 (sales only in general transaction), and the choose item method LOWEST_FIRST.

The promotion price derivation rule is applied once only, since it can only apply on sold line items in the general transaction.

As a result, the customer has to pay 44.73€.

	Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount of the Promotion**	**Effective Sales Price**	
		Sweater	1	27.27€		27.27€	
	T-shirt		1	30.00€	3.00€	27.00€	
	Jeans		1	50.00€	5.00€	45.00€	
Regular Total Amount							52.73€
Discount (subtracted from regular total amount)							8.00€
Effective Total Amount							44.73€

6.6.4.2 Resulting in the Highest Achievable Benefit

Example: Line Item-Related Promotion Price Derivation Rule

A customer has bought a sweater for the regular sales unit price of 20.00€ and a T-shirt for the regular sales unit price of 19.00€. Now, the customer wants to return the T-shirt and the sweater and buys a skirt for the regular sales unit price of 18.00€. There is one promotion valid and thus, is going to be applied to the transaction.

- The promotion price derivation rule applies a 10% discount to sold and returned line items of merchandise category "clothes", to which all line items in the transaction belong.
- The promotion price derivation rule may apply to maximal 1 item.
- The promotion price derivation rule has the sale return type SALES_RETURNS and the choose item method HIGHEST_FIRST.

The promotion price derivation rule is applied as two separate ones in order to avoid charging the sold against the returned line items:

1. Applied on sold line items only.
2. Applied on returned line items only.

As a result, the customer does not have to pay anything and the operator has to pay out 20.80€ to the customer.

	Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount of Promotion**	**Effective Sales Price**	
		Sweater	1	20.00€	2.00€	18.00€	
		T-Shirt	1	19.00€		19.00€	
	Skirt		1	18.00€	1.80€	16.20€	
Regular Total Amount							-20.80€

Example: Transaction-Related Promotion Price Derivation Rule

A customer has bought a sweater for the regular sales unit price 27.27€. Now, the customer wants to return the sweater and buy a T-shirt with a regular sales unit price of 15.00€ and a pair of jeans for 50.00€ . There is a promotion valid and thus, is going to be applied to the transaction.

- The promotion price derivation rule applies a 10% discount to sold and returned line items of merchandise category "clothes", to which all line items in the transaction belong.
- According to the eligibility's limit, the promotion price derivation rule may apply to maximal 2 line items.

- The promotion price derivation rule has the sale return type SALES_RETURNS, the calculation base for sales returns CALCBASE_01 (sales minus returns), and the choose item method HIGHEST_FIRST.

The promotion price derivation rule is applied once only, since the calculation base for sales returns is CALCBASE_01. It does not require sold and returned line items to be handled separately. Because the promotion price derivation rule is applied to two line items, the absolute values of the sales unit prices are compared with each other to achieve the highest benefit. The jeans and the returned sweater are discounted, not the T-shirt.
As a result, the customer has to pay 35.46€.

	Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount of the Promotion**	**Effective Sales Price**	
		Sweater	1	27.27€	2.72€	24.55€	
	T-shirt		1	15.00€		15.00€	
	Jeans		1	50.00€	4.99€	45.01€	
Regular Total Amount							37.73€
Discount (subtracted from regular total amount)							2.27€
Effective Total Amount							35.46€

6.6.4.3 THERE IS MORE

A detailed description of the selection of line items can be read in chapter Select Triggers and Discountable Line Items and in chapter Choose Items.

Related topics to the handling of the sales and the returns are:

- Activate Price Derivation Rule Eligibilities with Respect to Sales and Returns
- Handle Sales and Returns without the Original Transaction
- Handle Collisions with Respect to Sales and Returns
- Compute Total Price Modification Methods with Respect to Sales and Returns
- Mix and Match with Respect to Sales and Returns
- Select the Method to Discount

6.6.5 Compute Total Price Modification Methods with Respect to Sales and Returns

In case of simple discounts with a total price modification method *(RebatePromotionConditionRuleSO.priceModificationMethodCode)*, the benefit is computed on the summed up sales prices of all eligible line items and then proportionally distributed to each one.

HOW IT WORKS

The following price modification methods are counted to the total price modification methods:

- DISCOUNT_TOTAL/"RT": reduction of the summed up sales prices by an amount
- FIX_PRICE_TOTAL/"PT": setting the discount sales price of the summed up sales prices
- SET_PRICE_TOTAL/"ST": see Package an Offer
- DISCOUNT_PERCENT_TOTAL/"TP": reduction of the summed up sales prices by a percentage
- DISCOUNT_PERCENT_TOTAL2/"T2": reduction of the summed up sales prices by a percentage with different rounding

The sold line items are not charged against the returned line items in case of such promotion price derivation rules either – except for transaction-related promotion price derivation rules with the sale return type (*PromotionConditionSO.saleReturnTypeCode*) SALES_RETURNS and the calculation base for sales returns (*PromotionConditionRuleSO.calculationBase*) CALCBASE_01, where the calculation base (*RetailPriceModifier.calculationBaseAmount*, *PriceModificationLineItem.calculationBaseAmount*, *FrequentShopperPointsModifier.computationBaseAmount*, or *LoyaltyRewardLineItem.computationBaseAmount*) is calculated as sales minus returns.

Examples are provided for the different behaviors of the PCE in the subsequent sections. These examples focus on the three discounts' representatives the PCE supports.

6.6.5.1 Reduction by an Amount

Example
There exists a line item-related promotion price derivation rule with the price modification method DISCOUNT_TOTAL, which grants a 10.0€ off for all items "A". An item "A" has the regular sales unit price of 20.00€. The sale return type of the price derivation rule is SALES_RETURNS.

- *There are 3 sold line items "A" in the transaction.*
  - The promotion price derivation rule is only calculated once for the sales.
  - The discount is 10.00€. The customer has to pay 50.00€.
  - Two of the line items have the proportionally distributed discount of 3.33€, whereas the third line item receives a discount of 3.34€, so the sum matches the defined discount of 10.00€ in the price derivation rule.
- *There is 2 returned line items "A" in the transaction.*
  - The promotion price derivation rule is only calculated once for the returns.
  - The discount is 10.00€. The cashier has to pay out 30.00€ to the customer.

- The line items have the proportionally distributed discount of 5.00€ each.

- *There is one sold and two returned line items "A" in the transaction.*
  - The promotion price derivation rule is calculated once for the sales and once for the returns.
  - The discount is 10.00€ for the sold items and 10.00€ for the returned items. The cashier has to pay out 20.00€ to the customer.
  - The two returned line items "A" have the proportionally distributed discount of 5.00€ each. The sold line item "A" has the discount 10.00€.

6.6.5.2 Reduction by a Percentage

Example

We have a transaction-related promotion price derivation rule with the price modification method DISCOUNT_PERCENT_TOTAL, which grants 10% discount for all items "A". An item "A" has the regular sales unit price of 20.00€. The sale return type SALES_RETURNS, the calculation base for sales returns is CALCBASE_01 (sales minus returns).

- *There are 3 sold line items "A" in the transaction.*
  - The promotion price derivation rule is only calculated once for sales minus returns.
  - The discount is 6.00€. The calculation base is (60.00 € - 0.00€). The customer has to pay 54.00€.
  - All line items have the proportionally distributed discount of 2.00€ each.
- *There is one returned line item "A" in the transaction.*
  - The promotion price derivation rule is only calculated once for sales minus returns.
  - The discount is 2.00€. The calculation base is (0.00 € - 20.00€). The cashier has to pay out 18.00€ to the customer.
  - Line item A has the proportionally distributed discount of 2.0€.
- *There is one sold and two returned line items "A" in the transaction.*
  - The promotion price derivation rule is only calculated once for sales minus returns.
  - The discount is 2.00€. The calculation base is (20.00 € - 40.00€). The cashier has to pay out 18.00€ to the customer.
  - The two returned line items "A" have the proportionally distributed discount of 2.00€ each. The sold line item "A" has the discount 2.00€.

6.6.5.3 Set a New Price

Example

There exists a line item-related promotion price derivation rule with the price modification method SET_PRICE_TOTAL, which grants a new price of 10.0€ for all line items "A" together. An item "A" has the regular sales unit price of 20.00€. The sale return type SALES_RETURNS

- *There are 3 sold line items "A" in the transaction.*
  - The promotion price derivation rule is only calculated once for the sales.
  - The discount is 50.00€. The customer has to pay 10.00€.
  - Two of the line items have the proportionally distributed discount of 16.67€, whereas the third line item receives a discount of 16.66€, so the sum matches the defined discount of 50.00€.
- *There is one returned item "A" in the transaction.*
  - The promotion price derivation rule is only calculated once for the returns.
  - The discount is 10.00€. The cashier has to pay out 10.00€ to the customer.
  - Line item A has the proportionally distributed discount of 10.00€.
- *There is one sold and two returned line items "A" in the transaction.*
  - The promotion price derivation rule is calculated once for the sales and once for the returns.
  - The discount is 10.00€ for the sold line items and -30.00€ for the returned line items. Neither the customer, nor the cashier has to pay anything.
  - The two returned line items "A" have the proportionally distributed discount of 15.00€. The sold line item "A" has the discount 10.00€.

6.6.5.4 THERE IS MORE

Further information about simple discounts are described in Apply a Simple Discount.

Related topics to the handling of the sales and the returns are:

- 2019-10-29_13-35-47_Activate Price Derivation Rule Eligibilities with Respect to Sales and Returns
- Handle Sales and Returns without the Original Transaction
- Handle Collisions with Respect to Sales and Returns
- Choose Items with Respect to Sales and Returns
- Mix and Match with Respect to Sales and Returns
- Select the Method to Discount

6.6.6 Mix and Match with Respect to Sales and Returns

It is possible that triggers in the transaction do not get the benefit, but the items, merchandise categories, and product groups which are defined in the price derivation rule – the rule matching items (*MatchingItemSO*). This price derivation rule is herein called mix and match (*MixAndMatchPromotionConditionRuleSO*). As a result, the transaction has to contain the triggers for the price derivation rule eligibility and also the transaction matching items. The transaction matching items (*SaleReturnLineItem)* correspond to the rule matching items in the price derivation rule.

HOW IT WORKS

The sale return type (*PromotionConditionSO.saleReturnTypeCode*) selects the rule matching items of a mix and match as well. In case of the computation of sales only, the promotion is only applied if the triggers and the discountable line items are sales. If sales and returns are considered together, the triggers indicates which type the discountable line items need to have in order to apply a promotion price derivation rule. This is due to the fact that sales are not charged against returns. This also means, that if several triggers are necessary for triggering a promotion price derivation rule, all triggers and discountable line items need to have the same action code (*SaleReturnLineItem.actionCode*) – either SALE_ITEM or RETURN_ITEM.

The mix and match limit count *(MixAndMatchPromotionConditionRuleSO.limitCount*) is only considered for the mix and match combination code *(MixAndMatchPromotionConditionRuleSO.combinationCode*) OR. If the limit count is set to n (n is an element of the natural numbers – positive integer), up to n counts of the normalized discountable line items receive a discount in one calculation, regardless whether the calculation is performed only on sold or only on returned line items.

The required quantity (*MatchingItemSO.requiredQuantity*) is only considered for mix and match combination code AND and OR_QUANTITY. If the required quantity of a specific rule matching item is set to n , exactly n counts of the normalized transaction matching items receive a discount, regardless of these being either sold or returned line items.

Examples are provided for the different behaviors of the PCE in the subsequent sections.

6.6.6.1 Mix and Match for Sales Only

Example

Assume there exists a promotion price derivation rule in the promotion master data that grants for one table (eligible item), one sofa or one chair (rule matching items) for free. The mix and match combination code is OR in addition and the mix and match limit count of the rule is set to 1.0. The price derivation rule eligibility contains only the table, with threshold type (*ThresholdType*) QUT and the quantity threshold (*ThresholdPromotionConditionEligibility.thresholdQuantity*) 1.0.

If the sale return type is SALES, the PCE calculates the following:

1. Buy a table and buy two sofas. The promotion price derivation rule is calculated once for the sales. The benefit is applied to one of the sofas. ✓
2. Buy a table and two chairs, return an other table. The promotion price derivation rule is calculated once for the sales. The benefit is applied to one of the bought chairs. ✓
3. Buy a table and return a chair and a sofa. The promotion price derivation rule is calculated once for the sales. The benefit is not applied at all. ✗

4. Return a table, a sofa and a chair. The promotion price derivation rule is not triggered at all and thus, not calculated. ✗

6.6.6.2 Mix and Match for Returns Only

Example

Assume there exists a promotion price derivation rule in the promotion master data that grants for one table (eligible item), one sofa or one chair (rule matching items) for free. The mix and match combination code is OR in addition and the mix and match limit count of the rule is set to 1.0. The price derivation rule eligibility contains only the table, with threshold type QUT and the quantity threshold 1.0.

If the sale return type is RETURNS, the PCE calculates the following:

1. Buy a table and buy two sofas. The promotion price derivation rule is not triggered at all and thus, not calculated. ✗
2. Return a table and two chairs, buy an other table. The promotion price derivation rule is calculated once for the returns. The benefit is applied to one of the returned chairs. ✓
3. Buy a table and return a chair and a sofa. The promotion price derivation rule is not triggered at all and thus, not calculated. ✗
4. Return a table, a sofa and a chair. The promotion price derivation rule is calculated once for the returns. The benefit is applied to either the sofa or the chair due to the mix and match limit count 1.0. ✓

6.6.6.3 Mix and Match for Sales and Returns

Example

Assume there exists a promotion price derivation rule in the promotion master data that grants for one table (eligible item), one sofa or one chair (rule matching items) for free. Then mix and match combination code is OR and the mix and match limit count of the rule is set to 1.0. The price derivation rule eligibility contains only the table, with threshold type QUT and the quantity threshold 1.0.

If the sale return type is SALES_RETURNS, the PCE calculates the following:

1. Buy a table and buy two sofas. The promotion price derivation rule is calculated once for the sales. The benefit is applied to one of the sofas. ✓
2. Buy a table and two chairs, return an other table. The promotion price derivation rule is calculated once for the sales and once for the returns. The benefit is applied to one of the bought chairs only. ✓
3. Return a table and two chairs, buy an other table. The promotion price derivation rule is calculated once for the sales and once for the returns. The benefit is applied to one of the returned chairs only. ✓

4. Buy a table and return a chair and a sofa. The promotion price derivation rule is calculated once for the sales. The promotion price derivation rule does not apply at all. ✗
5. Return a table, a sofa and a chair. The promotion price derivation rule is calculated once for the returns. The benefit is applied to either the sofa or the chair due to the mix and match limit count 1.0. ✓

6.6.6.4 THERE IS MORE

A detailed description of the mix and match can be read in chapter Mix and Match.

Related topics to the handling of the sales and the returns are:

- 2019-10-29_13-35-47_Activate Price Derivation Rule Eligibilities with Respect to Sales and Returns
- Handle Sales and Returns without the Original Transaction
- Handle Collisions with Respect to Sales and Returns
- Choose Items with Respect to Sales and Returns
- Compute Total Price Modification Methods with Respect to Sales and Returns
- Select the Method to Discount

6.7 Select Triggers and Discountable Line Items

The PCE activates price derivation rule eligibilities that are fulfilled by the currently processed transaction. It selects normalized line items from the transaction – so-called triggers – that fulfill the price derivation rule eligibilities. In general, normalized line items are line items that have the quantity one.

The PCE orders the possible triggers in a way that allows the fulfillment of a maximum number of price derivation rule eligibilities within one promotion. To achieve this the items which trigger a lesser number of price derivation rule eligibilities of the promotion will be chosen first. This way the trigger items which have a higher probability to fulfill the further, remaining price derivation rule eligibilities are going to be processed afterwards and the number of fulfilled eligibilities is maximized. During the processing of a price derivation rule eligibility, the triggering normalized line items are used according to this order.

HOW IT WORKS

A so-called working bucket is generated for each activated price derivation rule eligibility of one promotion. The working bucket of a price derivation rule eligibility includes all the normalized line items that are triggers for this particular price derivation rule eligibility. The normalized line items are ordered according to the ascending number of triggered price derivation rule eligibilities of one promotion. This is done for each working bucket. The normalized line item on top of the working bucket thus fulfills the least number of price derivation rule eligibilities. This may result in more applied benefit for the customer.

During the processing of a price derivation rule eligibility, the normalized line item on top of its working bucket is selected as the first trigger for the corresponding price derivation rule eligibility. Then, the second normalized line item in this working bucket is selected. This process continues until the corresponding price derivation rule eligibility is satisfied. After that, the next price derivation rule eligibility is processed and so on.

The same procedure is followed if a mix and match rule (*MixAndMatchPromotionConditionRuleSO*) is computed by the PCE. In this case, a list of normalized line items that correspond to a rule matching item (*MatchingItemSO*) in the mix and match – so-called transaction matching items – form the working bucket for the rule matching item. The transaction matching item on top of the working bucket is selected since it hits the least number of rule matching items. This may result in more applied benefit for the customer.

It can happen that one normalized/transaction matching item fulfills as many price derivation rule eligibilities or rule matching items as another normalized/transaction matching item. In this case, these normalized/transaction matching items are sorted according to the choose item method *(PromotionConditionRuleSO.chooseItemMethod)* for the promotion price derivation rule that is currently processed. If the normalized/transaction matching items produce also equal values during sorting according to the choose item method, then the line item that got the greatest item identifier is selected first. If these identifiers are equal, too, the normalized/transaction matching item with the greatest sequence in the transaction is selected first.

In summary, the first priority is given to the normalized line item triggering as many price derivation rule eligibilities, or rule matching items in case of mix and match calculation, as possible for the currently processed promotion price derivation rule. By also accounting the choose item method during sorting (as the second criterion), the second priority is given to the generation of the lowest/highest possible discount according to the choose item method, after satisfying as much price derivation rule eligibility as possible.

Example: Select Transaction Matching Items
The promotion master data contains a mix and match with two rule matching items and with the mix and match combination code *(MixAndMatchPromotionConditionRuleSO.combinationCode*) AND. Both rule matching items contain product group identifiers (*MerchandiseSetPromotionConditionEligibilitySO.merchandiseSetID*) to identify rule matching items: 472021 for the rule matching item #1 and 472023 for rule matching item #2. Both rule matching items require one transaction matching item that is part of the product

group in order to be applied. The transaction contains two line items with equal regular sales prices:

- Line item: "noodles"
- Line item: "pasta sauce"

Product group 472021 (rule matching item #1) includes both line items of the transaction.
Product group 472023 (rule matching item #2) includes only line item "pasta sauce" of the transaction.
Rule matching item #1 is processed first by the PCE. The transaction is as follows:

- Line item "noodles"
  - o *sequence = 0*
  - o *itemID = 000*
- Line item "pasta sauce"
  - o *sequence = 1*
  - o *itemID = 001*

The working bucket of the rule matching item #1 includes line item with sequence = 0 and line item with sequence = 1. The line items in this working bucket are sorted according to the ascending number of rule matching items they correspond to. After sorting, the contents of the working bucket of the rule matching item #1 are [Line item with sequence = 0, Line item with sequence = 1]. Line item with sequence = 0 hits one rule matching item and line item with sequence = 1 hits two rule matching items. So, line item with sequence = 0, which is on top of the working bucket, is selected as the transaction matching item for the rule matching item #1.
The processing continues with the rule matching item #2. The working bucket of the rule matching item #2 includes only line item with sequence = 1. There is no need to sort this working bucket. So, line item with sequence = 1 is selected as the transaction matching item for the rule matching item #2. Both rule matching items are satisfied, so the corresponding promotion price derivation rule will be applied.

THERE IS MORE

Example: Select Transaction Matching Items in Case Selection Based on the Number of Fulfilled Eligibilities or Rule Matching Items Is Not Performed
The following example shows the possible inconsistent behavior in case the selection based on the number of fulfilled eligibilities or rule matching Items of transaction matching items is not performed by the PCE.
The promotion master data defined contains a mix and match rule with two rule matching items and with the mix and match combination code AND. Both rule matching items contain product group identifiers to identify rule matching items: 472021 for the rule matching item #1 and 472023 for rule matching item #2. Both rule matching items require one transaction

matching item that is part of the product group in order to be applied. The transaction contains two line items with equal regular sales prices:

- Line item: "noodles"
- Line item: "pasta sauce"

Product group 472021 (rule matching item #1) includes both line items of the transaction.
Product group 472023 (rule matching item #2) includes only line item "pasta sauce" of the transaction.
Rule matching item #1 is processed first by the PCE. The transaction is as follows:

- Line item "noodles"
    - o *sequence = 0*
    - o *itemID = 000*
- Line item "pasta sauce"
    - o *sequence = 1*
    - o *itemID = 001*

The line item with sequence = 1 will be used for the calculation, as both line items are transaction matching items for rule matching item #1, the regular sales prices are the same for both line items. In this case, the line item with the highest item ID in the transaction would be taken (see Choose Items). This leaves rule matching item #2 with zero transaction matching items, the promotion price derivation rule will not be applied.
However if rule matching item #2 was processed first, then the promotion could have been successfully applied on the same transaction. Rule matching item #2 would discount the "pasta sauce", while rule matching item #1 would discount the remaining item "noodles".
Conclusion:
The PCE would behave inconsistent based on the promotion masterdata structure if the transaction matching items were not sorted considering the number of fulfilled rule matching items. The possible benefit for the customer would depend on the order the rule matching items are processed – which depends on their rule matching item ID. Thus, the PCE selects the transaction matching items according to the number of rule matching items these correspond to.

6.8 Choose Items

The choose item method *(PromotionConditionRuleSO.chooseItemMethod)* provides the possibility to configure the sorting of the items when applying the benefit. As the case may be, the result of the promotion calculation could be a smaller or higher achievable benefit for the customer.

HOW IT WORKS

A transaction may contain more than one line items eligible for a promotion so that the PCE has to choose which line item gets the benefit and which does not. The following box describes the basic information of the used items and the promotion price derivation rule for the following examples.

Example: Choose Items

Let us assume that there is an item "bread maker" and the item "bread mix" with the possible flavors rye, spelt, and wheat. These have the following prices:

- Bread maker: 59.00€
- Bread mix "rye": 1.80€
- Bread mix "spelt": 2.30€
- Bread mix "wheat": 1.50€

We also assume that the following promotion price derivation rule exists in the promotion master data: For buying a bread maker, the customer gets for two bread mixes a discount of 50%. This promotion price derivation rule is applicable to all flavors of a bread mix. This example is a so-called Mix and Match. But this is not the only price derivation rule where this choice of items is used.

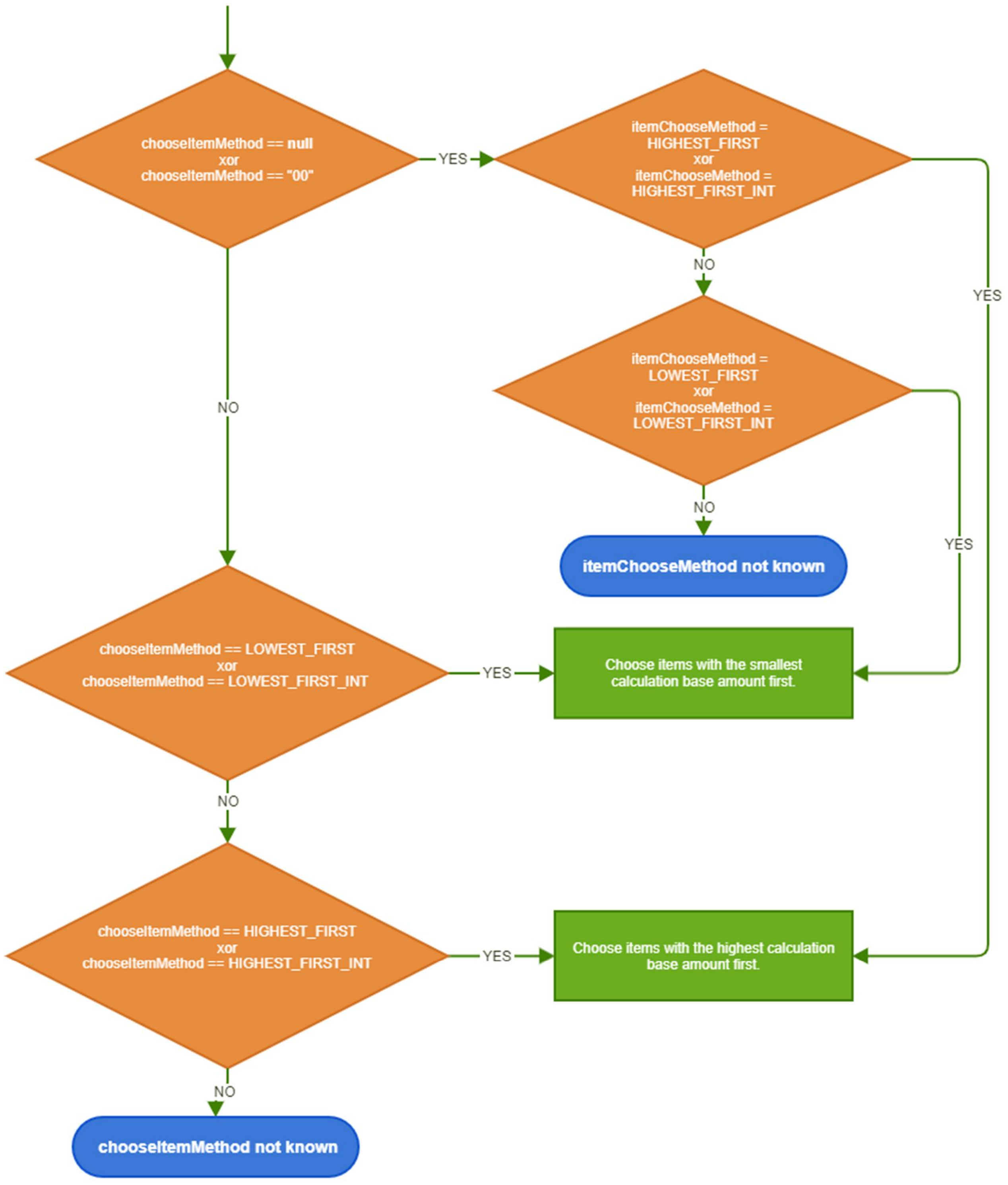
Based on the choose item method, the eligible line items of a particular transaction are chosen. For this purpose, it can be set in the promotion master data with the following values:

- LOWEST_FIRST or LOWEST_FIRST_INT: The eligible items are sorted based on their calculation base amount ascending. There is no difference in the handling of values LOWEST_FIRST and LOWEST_FIRST_INT.
- HIGHEST_FIRST or HIGHEST_FIRST_INT: The eligible items are sorted based on their calculation base amount descending. There is no difference in the handling of values HIGHEST_FIRST or HIGHEST_FIRST_INT.

It is also possible that the choose item method is not configured and thus empty (null) or "00". In this case, the PCE behaves according to the system configuration parameter *itemChooseMethod.* It can have the same values as the choose item method.

The following figure gives an overview about the behavior. It is illustrated, which value is resulting in which of the previously described scenarios.

Note that the handling of equal regular prices is independent from the choose item method.


chooseItemMethod == null
xor
chooseItemMethod == "00"
YES
itemChooseMethod =
HIGHEST_FIRST
xor
itemChooseMethod =
HIGHEST_FIRST_INT
NO
YES
itemChooseMethod =
LOWEST_FIRST
xor
itemChooseMethod =
LOWEST_FIRST_INT
NO
itemChooseMethod not known
YES
NO
chooseItemMethod == LOWEST_FIRST
xor
chooseItemMethod == LOWEST_FIRST_INT
YES
Choose items with the smallest calculation base amount first.
NO
chooseItemMethod == HIGHEST_FIRST
xor
chooseItemMethod == HIGHEST_FIRST_INT
YES
Choose items with the highest calculation base amount first.
NO
chooseItemMethod not known

The following section describes the process of choosing items in greater detail.

6.8.1 Choose Item Methods

Example (cont.)

Let us assume that a customer buys a bread maker for 59.00€. Additionally, the customer buys one or more of the following bread mixes:

1. First possibility: one bread mix "rye", one bread mix "spelt", and one bread mix "wheat"
2. Second possibility: three bread mixes "spelt"

The PCE calculates the benefit according to the value of the choose item method and *itemChooseMethod*. Thereby, the following four outcomes are possible:

1. Choose items with the smallest calculation base amount.
2. Choose items with the highest calculation base amount.
3. Choose items in case of equal prices.
4. Fall back to the default method.

6.8.1.1 Choose Items with the Smallest Calculation Base Amount First

The line items, which are eligible for the price derivation rule eligibility, are sorted in ascending order based on their calculation base amount. The first eligible line item (or line items, depending on the promotion price derivation rule) in the ordered transaction gets the benefit. The order of the line items in the transaction itself is not changed.

Example (cont.)

Let us assume that the choose item method is either equal to LOWEST_FIRST or to LOWEST_FIRST_INT. The PCE calculates the following monetary discounts for the previously mentioned possibilities:

- First possibility: One bread mix "rye", one bread mix "spelt", one bread mix "wheat", and a bread maker
  - The ordered transaction looks as follows:

i.	Bread mix "wheat":	1.50€
ii.	Bread mix "rye":	1.80€
iii.	Bread mix "spelt":	2.30€
iv.	Bread maker:	59.00€

  - The customer gets a 50% discount on one bread mix "wheat" and one bread mix "rye" each. The total discount amounts to 1.65€.

6.8.1.2 Choose Items with the Highest Calculation Base Amount First

The line items, which are eligible for the price derivation rule eligibility, are sorted in descending order based on their calculation base amount. The first eligible line item (or line items, depending on the promotion price derivation rule) in the ordered transaction gets the benefit. The order of the line items in the transaction itself is not changed.

Example (cont.)

Let us assume that the choose item method is either equal to HIGHEST_FIRST or to HIGHEST_FIRST_INT. The PCE calculates the following monetary discounts for the previously mentioned possibilities:

- First possibility: one bread mix "rye", one bread mix "spelt", one bread mix "wheat", and a bread maker
  - The ordered transaction looks as follows:

i.	Bread maker:	59.00€
ii.	Bread mix "spelt":	2.30€
iii.	Bread mix "rye":	1.80€
iv.	Bread mix "wheat":	1.50€

  - The customer gets a 50% discount on one bread mix "spelt" and one bread mix "rye" each. The total discount amounts to 2.05€.

6.8.1.3 Choose Items in Case of Equal Prices

In the case of equal prices, the line items are sorted in descending order according to their sequence in the transaction. This means that the last line items get the discount. The order of the line items in the transaction itself is not changed.

Example (cont.)

Let us assume that the choose item method is either equal to "00", LOWEST_FIRST, HIGHEST_FIRST, LOWEST_FIRST_INT, or to HIGHEST_FIRST_INT. The PCE calculates the following monetary discounts for the previously mentioned possibilities since the bread mixes have all the same regular price:

- Second possibility: three bread mixes "spelt" and a bread maker
  - The ordered transaction looks as follows:

i.	Bread maker:	59.00€
ii.	Bread mix "spelt":	2.30€
iii.	Bread mix "spelt":	2.30€
iv.	Bread mix "spelt":	2.30€

  - The customer gets a 50% discount on two bread mixes "spelt" each. The total discount amounts to 2.30€.
  - The bread mixes at position 3. and 4. get 1.15€ each as a discount.

6.8.2 Fall Back to the Default Method

If the choose item method is not set in the price derivation rule or has the value "00", the PCE falls back to the default method: It checks the *itemChooseMethod* in the system configuration.

Depending on the value of the parameter, the items are sorted based on their calculation base amount.

6.8.3 THERE IS MORE

The selection of items for the application of a promotion price derivation rule is not to be mixed up with the best price calculation in case of the collision of several promotion price derivation rules. In the latter case, a promotion price derivation rule is selected and not the item on which the promotion price derivation rule shall be applied.

Further information about the choose item method and *itemChooseMethod* can be found in chapter Configuration.

6.9 Apply a Benefit to Fully-Priced Items Only

A fully-priced item is an eligible item in the transaction that is not discounted by any previously applied promotion price derivation rule.

HOW IT WORKS

Based on the no previous monetary discount flag *(PromotionConditionRuleSO.noPreviousMonetaryDiscountAllowedFlag)* and the item in the transaction, the promotion price derivation rule is applied to the item or not. For this purpose, the no previous monetary discount flag can be set in the promotion master data with the following values:

- True: The promotion price derivation rule is only applied to a given discountable item if no other monetary discounts have been granted to the item before.
- False: The promotion price derivation rule can be applied to all discountable items.

It is possible that the no previous monetary discount flag is not configured. In this case, the PCE behaves as if the flag were false.

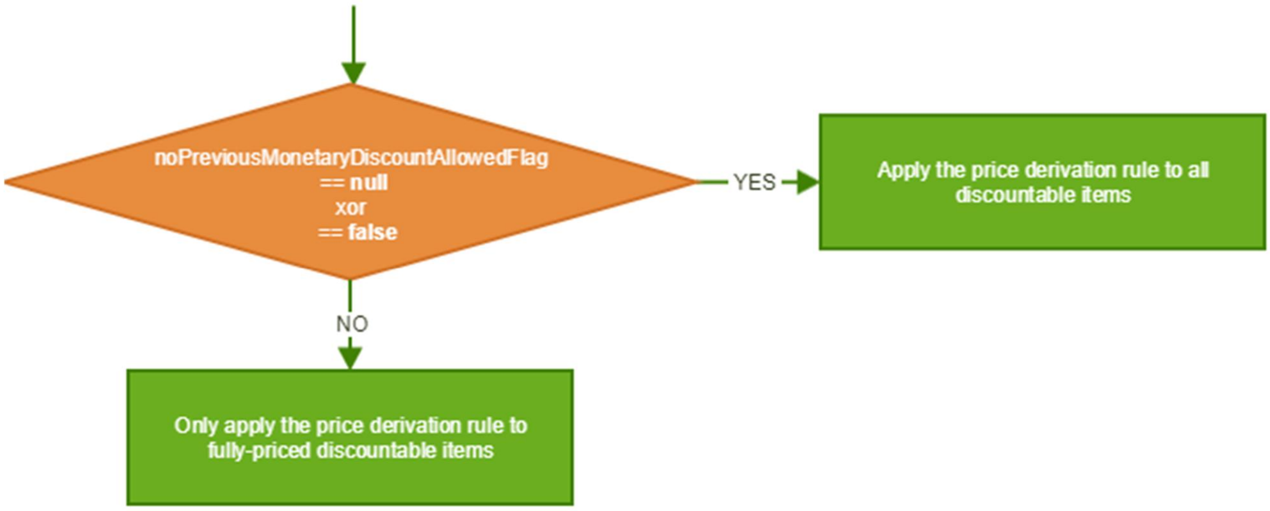
A discount computed by the PCE which reduces the price of a line item or the transaction according to the price derivation rules is called monetary discount. Any other promotion which grants bonus points, virtual discounts or otherwise does not change the price of items or the transaction is considered a non-monetary discount.

If the flag is true:The no previous monetary discount flag influences the application of the so maintained promotion price derivation rule and does not prevent any subsequent promotion price derivation rules. If a transaction includes a discountable item, the PCE behaves according to the no previous monetary discount flag. This would result in the following outcomes:

- If the flag is true:

  - The PCE checks whether a monetary discount was granted for the discountable item in advance. If a monetary discount was granted previously, the price derivation rule is not applied. Otherwise, the PCE executes the price derivation rule.
- If the flag is false:
  - The PCE executes the corresponding price derivation rule.

The subsequent figure gives an overview about the behavior.


noPreviousMonetaryDiscountAllowedFlag
== null
xor
== false
YES
Apply the price derivation rule to all discountable items
NO
Only apply the price derivation rule to fully-priced discountable items

Example: Apply a Benefit to Fully-Priced Items Only

Let us assume that there exist two promotion price derivation rules with an item eligibility for "Arabica coffee" each. The item "Arabica coffee" is part of the master data and costs 7.00€ per 500 g package.

One promotion price derivation rule grants 3.00€ discount for two packages of Arabica coffee and is not restricted to fully-priced items, whereas the other grants 10% discount and is restricted to fully-priced items. This means the following promotion price derivation rules exist in the promotion master data:

1. First promotion price derivation rule: The item eligibility includes Arabica coffee with a quantity threshold of 2 and a quantity limit of 2, grants 3.00€ monetary discount, and the no previous monetary discount flag is false. The promotion price derivation rule's sequence is 1.
2. Second promotion price derivation rule: The item eligibility includes Arabica coffee, grants 10% monetary discount, and the no previous monetary discount flag is true. The promotion price derivation rule's sequence is 2.

We assume that a customer buys five packages Arabica coffee. Then the customer gets the following benefits for a transaction with one line item and the quantity 5:

- 5x Arabica Coffee with a regular price:

- The customer gets a benefit of 3.00€ from the first promotion price derivation rule for two packages of Arabica coffee.
- The customer gets an additional 10% discount for the 3 other packages of Arabica coffee.
- This results in an effective price of: 29.90€.

If the no previous monetary discount flag of the second promotion price derivation rule is false, the customer gets 10% discount for all 5 packages of Arabica coffee and not only for three. This results in an effective price of 28.80€.

THERE IS MORE

The promotion price derivation rules for the fully-priced items do not consider the calculation base sequence (*PromotionConditionRuleSO.calculationBaseSequence*), the no effect on successors flag (*PromotionConditionRuleSO.noEffectOnSubsequentPromotionConditionFlag*), and the consider predecessors flag (PromotionConditionRuleSO.considerPreviousPromotionConditionFlag). This means that the calculation base amount cannot be changed for these promotion price derivation rules.

In addition, a zero discount is not changing the price total of the corresponding item and thus, the item with a zero discount is still considered fully-priced. This results in the outcome that a promotion price derivation rule that has a no previous monetary discount flag which is set to true is still applicable to an item with a zero discount.

Further information about the no previous monetary discount flag can be found in chapter Configuration.

An item with a manually overridden price (see chapter Manually Override a Regular Sales Unit Price) is handled just like any other fully-priced item. If the no previous monetary discount flag is set to true, promotion price derivation rules are thus also applicable to line items with an overridden price if no price derivation rule was applied previously.

7 Promotion Calculation – Price Derivation Rules

The step promotion calculation computes the discounts and loyalty points for a certain transaction. This is the core functionality of the PCE. Within this step, the promotion price derivation rules are loaded and validated and the price derivation rules are executed. Thereby, only the promotion price derivation rules are loaded and validated if the corresponding price derivation rule eligibility is activated.

There are several types of benefits that can be calculated during this step. The PCE chooses the benefit type according to the information stored in the price derivation rule. In addition, it performs the calculation of the benefit dependent on the price derivation rule. This means it depends on the information stored in the price derivation rule and thus the benefit type which calculation steps are performed by the PCE and which are not. The different price derivation rules are described in the following chapters.

In this chapter, the following topics are discussed:

- Apply a Simple Discount
- Package an Offer
- Apply a Zero Discount
- Grant Loyalty Points
- Manually Apply a Benefit
- Apply Customer-Specific Prices
- Mix and Match
- Trigger an Action of the Client Application
- Handle Externally Applied Benefits

7.1 Apply a Simple Discount

The PCE can compute simple discounts like a discount amount, a discount in percent, or a new price.

HOW IT WORKS

The bonus points flag *(PromotionConditionRuleSO.bonusPointsFlag)* controls whether a discount or loyalty points are granted as a benefit:

- Bonus points flag is false results in a discount.
- Bonus points flag is true results in loyalty points.

To apply a simple discount, the bonus points flag has to be set to false in the corresponding price derivation rule.

Based on the price modification method *(RebatePromotionConditionRuleSO.priceModificationMethodCode)*, the PCE computes the different simple discounts. The price modification method can be set in the promotion master data with the following values. Each value represents a simple discount and defines its calculation:

- DISCOUNT_SINGLE/"RS": reduction of the sales unit price by an amount
- DISCOUNT_PERCENT/"RP": reduction of the sales unit price by a percentage
- FIXED_PRICE/"PS": setting the discount sales unit price
- DISCOUNT_TOTAL/"RT": reduction of the summed up sales prices by an amount
- FIX_PRICE_TOTAL/"PT": setting the discount sales price of the summed up sales prices
- SET_PRICE_TOTAL/"ST": see Package an Offer
- DISCOUNT_PERCENT_TOTAL/"TP": reduction of the summed up sales prices by a percentage
- DISCOUNT_PERCENT_TOTAL2/"T2": reduction of the summed up sales prices by a percentage with different rounding

When computing the discount amounts of the current transaction, the following outcomes are possible:

1. Apply a discount as amount
2. Apply a discount in percent
3. Set a new price

7.1.1 Apply a Discount as Amount

It is possible to reduce the regular sales unit price of an item and the summed up regular sales prices of several line items by an amount, respectively.

The price modification method must be DISCOUNT_SINGLE in order to reduce the regular sales unit price of an item. The calculation base amount is the regular sales unit price if this regular sales unit price is not reduced by a previously executed price derivation rule. The price derivation rule states the amount by which the regular sales unit price shall be reduced. If several quantities of a line item or several line items are eligible, the PCE applies the discount amount for each quantity and line item (normalized line item which means a line item with the quantity one). This results in the multiplication of the discount.

Note that the behavior is different for transaction-related promotion price derivation rules. In such cases, the DISCOUNT_SINGLE is computed in the same way as the price modification method DISCOUNT_TOTAL.

If the price modification method is DISCOUNT_TOTAL, the summed up sales prices of several eligible line items are reduced by an amount. The calculation base amount is the sum over all regular sales prices of eligible line items in the transaction. The calculation base amount is then reduced by the amount that is denoted in the corresponding price derivation rule. If different line items are eligible for the promotion price derivation rule, the overall discount is prorated to the line items.

7.1.2 Apply a Discount in Percent

It is possible to apply a discount in percent.

The price modification method must be DISCOUNT_PERCENT in order to reduce the regular sales unit price of an item by a percentage. The calculation base amount is the regular sales unit price if it is not reduced by a previously executed price derivation rule. The price derivation rule states the percentage by which the calculation base amount shall be reduced. This percentage is computed for each normalized eligible line item's calculation base amount and thus the PCE computes the discount amount for each one. This results in the addition of the discount amounts if several quantities or line items are eligible.

If the price modification method is DISCOUNT_PERCENT_TOTAL or DISCOUNT_PERCENT_TOTAL2, the summed up regular sales prices of several eligible line items are reduced by a percentage. The calculation base amount is the sum over all regular sales prices of eligible line items in the transaction. The calculation base amount is then reduced by the percentage denoted in the price derivation rule. The percentage is the corresponding discount amount over all eligible line items. The overall discount amount is prorated to the line items.

7.1.3 Set a New Price

It is possible to set the discount sales price of an item and the summed up discount sales prices of several line items, respectively. The PCE afterwards calculates the granted discount amounts.

The price modification method must be FIXED_PRICE in order to set the discount sales unit price of an item. The calculation base amount is the regular sales unit price if this regular sales unit price is not reduced by a previously executed price derivation rule. The price derivation rule states the new discount sales unit price of an item. The PCE calculates the difference between the calculation base amount and the new discount sales unit price as discount amount. If several quantities of a line item or several line items are eligible, the PCE applies the previously obtained discount amount for each normalized item. This results in the multiplication of the discount.

Note that the behavior is different for transaction-related promotion price derivation rules. In such cases, the FIXED_PRICE is computed in the same way as the price modification method FIX_PRICE_TOTAL.

If the price modification method is FIX_PRICE_TOTAL, the summed up regular sales prices of several eligible line items are set to a new discount sales price. The calculation base amount is the sum over all regular sales prices of eligible line items in the transaction. The PCE calculates the difference between the calculation base amount and the new discount sales price for the set of items. This difference is the discount amount. If different line items are eligible for the promotion price derivation rule, the discount amount is prorated to the line items.

7.1.4 THERE IS MORE

The computation of the discount and the setting of new prices depend on the calculation base amount. Chapter Set the Calculation Base Amount describes how to adjust the calculation base amount for the price derivation rule execution.

The discount total amount in total cannot be greater than the regular total amount of the transaction. This also applies to the discount shares, meaning that the discount share for each line item cannot exceed the regular sales price of that line item. Refer to chapter Prorate the Benefit for more details.

Note that for the price derivation rule type "Mix and Match" only discounts are allowed. Thus, the bonus points flag is not analyzed for this price derivation rule type.

7.2 Package an Offer

The packaged offer is a type of promotion price derivation rule. It sets the discount price for a set of items.

HOW IT WORKS

The price modification method (*RebatePromotionConditionRuleSO.priceModificationMethodCode*) is set to SET_PRICE_TOTAL in the price derivation rule. It is used to indicate that the PCE shall apply a packaged offer. The calculation of the benefit is done as if price modification method is FIX_PRICE_TOTAL.

Example

Let us assume there is a packaged offer in the promotion master data. It states that when buying a coffee maker and two coffee pads, you get them together for 59.00€. In addition, the regular sales unit price of the coffee maker is 79.00€ and of the coffee pad it is 5.00€.
If a customer buys exactly one coffee maker and two coffee pads, the regular total amount is 89.00€. The PCE applies the packaged offer which results in a discount of 30.00€ in total. That discount is prorated to the involved line items according to their regular sales prices and involved quantities. Thus, the discount share of the two coffee pads is 3.38€ and the discount share of the coffee maker is 26.62€ (the difference to 30.00€) if the cheapest items are chosen first.

THERE IS MORE

The discount price in total cannot be greater than the regular total amount of the transaction. This also applies to the discount shares, meaning that the discount share for each line item cannot exceed the regular sales price of that line item. Refer to chapter Prorate the Benefit for more details.

You can look up the handling of FIX_PRICE_TOTAL in chapter Apply a Simple Discount.

7.3 Apply a Zero Discount

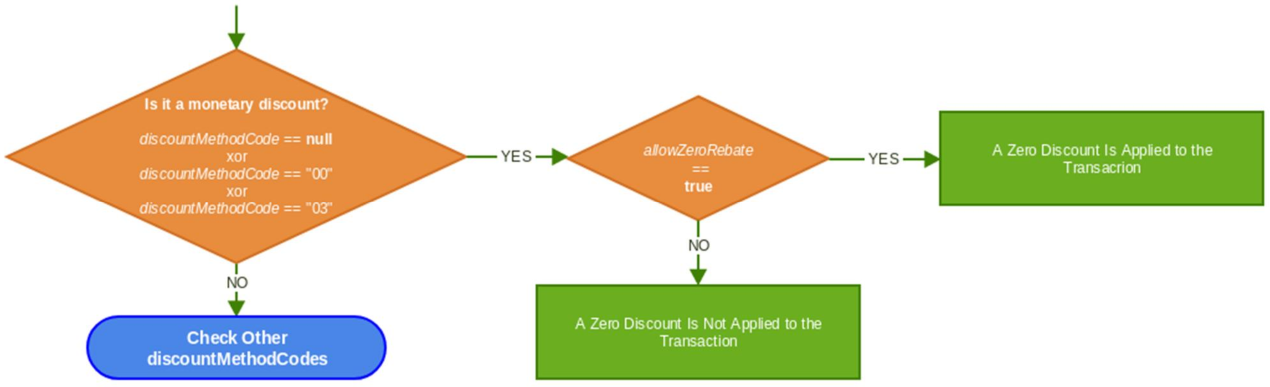
It is possible to allow or deny the application of a promotion price derivation rule resulting in a monetary discount of zero. Thereby, the discount can be zero because the price derivation rule includes a monetary discount of zero. A promotion price derivation rule which typically includes a non-zero discount can also result in a discount of zero for particular transactions.

HOW IT WORKS

A monetary discount is typically calculated by the PCE. It reduces the total of a line item or a transaction.

Based on the value of the system flag *allowZeroRebate*, the PCE applies a zero discount or not. If *allowZeroRebate* is false, the PCE generally does not apply monetary discounts of zero. If the system flag *allowZeroRebate* is true, the PCE does apply a monetary discount of zero. This results in the attachment of a retail price modifier (*RetailPriceModifier*) or a price modification line item (*PriceModificationLineItem*) with a monetary discount of zero to the transaction.

The following figure gives an overview about the behavior according to the parameter:


Is it a monetary discount?
discountMethodCode == null
xor
discountMethodCode == "00"
xor
discountMethodCode == "03"
YES
allowZeroRebate
==
true
YES
A Zero Discount Is Applied to the Transacrion
NO
A Zero Discount Is Not Applied to the Transaction
NO
Check Other discountMethodCodes

The PCE is invoked by the input of a transaction. The transaction contains line items triggering a promotion price derivation rule that shall reduce the regular/discount total amount. The PCE calculates the discount. If the monetary discount is non-zero, the transaction is updated in any case afterwards. If the monetary discount is zero, the system flag *allowZeroRebate* has to be true so that the transaction is updated. Otherwise, the transaction will not be updated.

Example: Apply a Zero Discount

Let us assume that there is an item "Arabica coffee" which costs 9.99€ for a 500 g package.

We further assume that a promotion price derivation rule is part of the promotion master data. This promotion price derivation rule sets the total to 19.98€, if the transaction includes two or more and less than four Arabica coffees.
Assume that one customer A buys two quantities of Arabica coffee. The customer A gets a benefit of 0.0€ which is a zero discount. Since in the case the two quantities of Arabica coffee get a zero discount, it depends on the system flag *allowZeroRebate* how the PCE behaves: If the flag is true, the transaction is updated, otherwise not.
Assume that another customer B buys three quantities of Arabica coffee. The customer B gets a benefit of 9.99€, because the promotion price derivation rule sets the total from 29,97 to 19.98€. This case is not a zero discount and the transaction update is not dependent on the system flag *allowZeroRebate*.

7.3.1 THERE IS MORE

There are additional possibilities that do not result in the reduction of the total of a line item or a transaction at all. These possibilities are described in chapter Select the Method to Discount. It contains further information, for example, about the printing of coupons.

Further information about the flag *allowZeroRebate* can be found in chapter Configuration.

7.3.1.1 Zero Discount and Retail Price Modifier Quantity

The Retail Price Modifier Quantity (*RetailPriceModifier.quantity)* in the response can be different depending on the setting of the *allowZeroRebate* flag. Due to the proration of the benefit, a part of the transaction may not receive a discount. If *allowZeroRebate*=true , the quantity of items receiving zero discount is included in the *RetailPriceModifier.quantity*. If *allowZeroRebate*=false , it is not allowed to receive a zero discount, and the quantity of items receiving zero discount will not be included in the *RetailPriceModifier.quantity*.

Example: Different RetailPriceModifier.quantity
Let us assume that there is an item "lip balm" with a regular sales price of 1.19€ .
We further assume that a promotion price derivation rule is part of the promotion master data. This promotion price derivation rule applies 20% position discount, if the transaction includes minimum amount of 150.00€ for "lip balm".

- The promotion is triggered by a quantity of 126.05 pieces of "lip balm" (150.00 € divided by 1.19€ =126.05 items).
- 20% discount of 150.00€ are 30.00€, what leads after rounding to 0.24€ discount per item (30.00€ divided by 126.05 pieces).
- Because of rounding the discount of 30.00 € is reached after applying 0.24€ discount to 125 items.
- Last 1.05 items get zero discount.

Behavior, when the system flag *allowZeroRebate* is true: *RetailPriceModifier.quantity*=126.05 in the response, because Zero Discount is allowed.
Behavior, when the system flag *allowZeroRebate* is false: *RetailPriceModifier.quantity*=125.00 in the response, because Zero Discount is not allowed and the last 1.05 items are excluded.

7.4 Grant Loyalty Points

The PCE can compute loyalty points based on the current transaction.

HOW IT WORKS

To grant loyalty points, the bonus points flag *(PromotionConditionRuleSO.bonusPointsFlag)* has to be true in the corresponding price derivation rule.

The PCE computes the loyalty points based on the price modification method (*RebatePromotionConditionRuleSO.priceModificationMethodCode*). The price modification method can be set in the promotion master data with the following values. Each value represents a different way of calculating loyalty points:

- DISCOUNT_SINGLE/"RS": Set the amount of loyalty points per normalized item (the line items with the quantity one).
- DISCOUNT_PERCENT/"RP": Calculate the discount in percent per normalized item (the line items with the quantity one) and convert it into loyalty points.
- DISCOUNT_TOTAL/"RT": Set the amount of loyalty points.
- DISCOUNT_PERCENT_TOTAL/"TP": Calculate the discount in percent and convert it into loyalty points.
- DISCOUNT_PERCENT_TOTAL2/"T2": Calculate the discount in percent and convert it into loyalty points with different rounding.

When computing the loyalty points, the following outcomes are possible:

1. Grant an amount of loyalty points
2. Grant loyalty points as a percentage

7.4.1 Grant an Amount of Loyalty Points

An amount of loyalty points can be granted as reward for the purchase in two ways - DISCOUNT_SINGLE (for each eligible normalized line item) and DISCOUNT_TOTAL (for total quantity or amount). The amount of loyalty points is always denoted in the price modification amount (*RebatePromotionConditionRuleSO.priceModificationAmount*).

If the price modification method is DISCOUNT_SINGLE, the PCE grants an amount of loyalty points for each eligible normalized line item. If several quantities of a line item or several line items are eligible, the PCE grants the amount of loyalty points for each quantity and line item that is eligible. This results in the multiplication of the amount of loyalty points. The amount of loyalty points is rounded afterwards according to the rounding rule (*RoundingRuleDO*).

The price modification method must be DISCOUNT_TOTAL to grant an amount of loyalty points for the total quantity or amount defined by the price derivation rule eligibility. The price modification amount is rounded first according to the rounding rule and afterwards the amount of loyalty points is applied. If different normalized line items are eligible for the promotion price derivation rule, the amount of loyalty points is prorated to these normalized line items.

Example: Grant an Amount of Loyalty Points

Let us assume that there is the following item in the master data:

- Vase: 10.10€

We also assume that there are the two following promotion price derivation rules in the promotion master data:

1. Get for every purchase vase 100 loyalty points in interval of two. The price modification method is DISCOUNT_SINGLE.
2. Get for every two vases 100 loyalty points. The price modification method is DISCOUNT_TOTAL.

The customer buys 5 vases. The PCE grants 400 loyalty points as result of the first promotion price derivation rule. The first price derivation rule eligibility is fulfilled two times with 4 vases. Since only for every two vases loyalty points are granted, the customer does not get additional loyalty points for the 5th vase.

In addition, the customer gets 200 loyalty points as result of the second promotion price derivation rule. The second price derivation rule eligibility is fulfilled two times with 4 vases. Thus, the 100 loyalty points are doubled.

7.4.2 Grant Loyalty Points as a Percentage

It is possible to grant loyalty points as a percentage of the sales price. The percentage is always part of the attribute price modification percent (*RebatePromotionConditionRuleSO.priceModificationPercent*). The system parameter *percentualPointsMethod* defines thereby when the rounding of the loyalty points is done – either at the end of the calculation (*percentualPointsMethod* is AFTER) or at the beginning of the calculation (*percentualPointsMethod* is BEFORE). This system parameter is not considered for the price modification method DISCOUNT_PERCENT_TOTAL2.

The following three text parts describe the different order of the same calculation steps for the three price modification method values:

1. DISCOUNT_PERCENT/"RP"

2. DISCOUNT_PERCENT_TOTAL/"TP"
3. DISCOUNT_PERCENT_TOTAL2/"T2"

The table below summarizes the order of calculation steps for these three price modification methods. Each background color emphasizes thereby the same calculation step.

1. The price modification method must be DISCOUNT_PERCENT to grant loyalty points as a percentage of a normalized line item. First of all, the discount amount for the percentage is calculated. Subsequently, the discount amount is converted into loyalty points. The calculation base amount (*LoyaltyRewardLineItem.computationBaseAmount* or *FrequentShopperPointsModifier.computationBaseAmount*) is the regular sales unit price if it is not reduced by a previously executed price derivation rule. The price modification percent states the percentage of the calculation base amount that is considered for the conversion into loyalty points. This percentage is computed for each normalized eligible line item's calculation base amount and thus, the PCE computes the discount amount for each one. It depends on the *percentualPointsMethod* when the discount amount is converted into loyalty points and when these are rounded according to the rounding rule. If the *percentualPointsMethod* is AFTER, the discount amount is computed and multiplied with the system parameter *pointsFactor*. The result is rounded afterwards. If the *percentualPointsMethod* is BEFORE, the calculation base amount is multiplied with the system parameter *pointsFactor* and then rounded according to the rounding rule. The calculation base amount is thus already converted into loyalty points. The percentage for the eligible normalized line items is calculated subsequently and rounded again afterwards.

2. If the price modification method is DISCOUNT_PERCENT_TOTAL, the loyalty points are computed as a percentage of the summed up sales prices of several eligible line items. Like in case of DISCOUNT_PERCENT, the calculation depends on the system parameter *percentualPointsMethod*. The overall calculation base amount and discount amount, respectively, is multiplied with the value of the system parameter *pointsFactor*. The difference is that these computed loyalty points are the sum that is granted for all eligible line items. The computed amount of loyalty points is prorated to the eligible normalized line items afterwards.

3. The price modification method DISCOUNT_PERCENT_TOTAL2 depends not on the system parameter *percentualPointsMethod*. The calculation base amount is thereby first rounded according to the rounding rule. Afterwards, the calculation base amount is multiplied by the price modification percent and divided by 100. These results in the overall loyalty points which are finally multiplied with the *pointsFactor*, rounded again and prorated to the eligible normalized line items.

The following table highlights the different order of the calculation steps between these three price modification methods for granting loyalty points as a percentage.

Order of Calculation Steps	**1. DISCOUNT_PERCENT**		**2. DISCOUNT_PERCENT_TOTAL**		**3. DISCOUNT_PERCENT_TOTAL2**
1	*percentualPointsMethod* == AFTER	*percentualPointsMethod* == BEFORE	*percentualPointsMethod* == AFTER	*percentualPointsMethod* == BEFORE	–
2	Calculation base amount determination for normalized line items.	Calculation base amount determination for normalized line items.	Calculation base amount determination for line items.	Calculation base amount determination for line items.	Calculation base amount determination.
3	Rounded calculation base amount * price modification percent / 100.	Multiplied with *pointsFactor.*	Rounded calculation base amount * price modification percent / 100.	Multiplied with *pointsFactor.*	Rounding according to rounding rule.
4	Multiplied with *pointsFactor.*	Rounding according to rounding rule.	Multiplied with *pointsFactor.*	Rounding according to rounding rule.	Rounded calculation base amount * price modification percent / 100.
5	Rounding according to rounding rule.	Rounded calculation base amount loyalty points * price modification percent / 100.	Rounding according to rounding rule.	Rounded calculation base amount loyalty points * price modification percent / 100.	Multiplied with *pointsFactor.*
6	–	Rounding according to rounding rule.	Proration of loyalty points.	Rounding according to rounding rule.	Rounding according to rounding rule.
7	–	–	–	Proration of loyalty points.	Proration of loyalty points.

Example: Grant Loyalty Points as a Percentage

Let us assume that there is the following item in the master data:

- Vase: 10.10€
- Bunch of flowers: 12.54€

We also assume that there is the following promotion price derivation rule in the promotion master data:

1. Get for every two vases 10% loyalty points. The price modification method is DISCOUNT_PERCENT_TOTAL. The *pointsFactor* is 1.0 and the *percentualPointsMethod* is AFTER. The rounding rule states to round the amount commercially to an integer.

The customer buys three vases and two bunches of flowers. The PCE then determines the calculation base amount which is 30.30€ and multiplies it with 0.1 which results in 3.03. This is then multiplied with the *pointsFactor* of 1.0. Then the amount is rounded to 3 loyalty points in accordance to the rounding rule. As a result, the customer has to pay 55.38€ and is granted 3 loyalty points.

7.4.3 THERE IS MORE

It is possible to maintain base loyalty points via a transaction-related promotion price derivation rule with the price modification method DISCOUNT_PERCENT and the price modification percent 100%. In order to multiply these (base loyalty points multiplication), another transaction-related promotion price derivation rule can be maintained in the promotion master data. This promotion price derivation rule has also the price modification method DISCOUNT_PERCENT. The price modification percent is then the multiplication factor minus one times 100. This approach does not support multiple loyalty programs.

A promotion price derivation rule granting loyalty points may be applicable to returned line items as well.

The computation of the loyalty points is based on the calculation base amount. Chapter Set the Calculation Base Amount describes how to adjust the calculation base amount for the price derivation rule execution.

Note that the price modifications described in this chapter are not applicable for the price derivation rule type "Mix and Match".

7.5 Manually Apply a Benefit

An employee of the retail company grants a special/manual discount to manually apply a benefit. This discount can either be granted for one or more items or for the entire purchase. Thereby, the information about the discount is given by the request (manual price derivation rule) or by the promotion master data (predefined price derivation rule).

HOW IT WORKS

The PCE gets a so called manual trigger (*ManualPromotionTrigger*) in the PCE-request. This manual trigger has to correspond to a price derivation rule eligibility in the promotion master data – herein manual eligibility (*ManualPromotionConditionEligibilitySO*). The manual eligibility always contains the trigger type (*ManualPromotionConditionEligibilitySO.triggerType*) and the trigger value (*ManualPromotionConditionEligibilitySO.triggerValue*). Both attributes can be set by the retailer since they are used to distinguish between different possible discounts.

The way how the PCE calculates the benefit depends on the price derivation rule. This results in the following possibilities:

1. Manually trigger a predefined price derivation rule.

2. Manually trigger and set a price derivation rule.
3. Apply a manual trigger several times.

7.5.1 Manually Trigger a Predefined Price Derivation Rule

To apply a manual benefit, the transaction includes a manual trigger with the following attributes:

- For line item-related promotion price derivation rules:
  - Item level trigger type *(SaleReturnLineItemPromotionTrigger.triggerType*)
  - Item level trigger value *(SaleReturnLineItemPromotionTrigger.triggerValue)*
  - Item level trigger sequence number *(SaleReturnLineItemPromotionTrigger.Key.triggerSequenceNumber)*
  - Item level trigger sequence addend *(SaleReturnLineItemPromotionTrigger.triggerSequenceAddend)*
- For transaction-related promotion price derivation rules:
  - Transaction level trigger type *(RetailTransactionPromotionTrigger.triggerType)*
  - Transaction level trigger value *(RetailTransactionPromotionTrigger.triggerValue)*
  - Transaction level trigger sequence number *(RetailTransactionPromotionTrigger.Key.triggerSequenceNumber)*
  - Transaction level trigger sequence addend *(RetailTransactionPromotionTrigger.triggerSequenceAddend)*

The trigger type and trigger value have to be equivalent to the trigger type and trigger value in the manual eligibility. If there is no further information in the transaction as described previously, the price derivation rule has to be defined in the promotion master data. The price derivation rule can be chosen in advance and with respect to the price derivation rule types that are supported by the PCE – for example, simple discounts or loyalty points. The PCE then calculates the benefit according to the price derivation rule in the promotion master data.

In the OPP scenario, the item level and the transaction level privilege type and the privilege value have to be part of the transaction (see the additional attributes for Manually Trigger and Set a Price Derivation Rule and their functionality). In this case, the privilege type needs to be AM to indicate that the price derivation rule is defined in the promotion master data. The content of the privilege value can be an arbitrary number since the PCE does not take it into account.

Example: Manual trigger triggered by a client

Let us assume that there is an item "Sun Lotion" for 15.00€.

Let us also assume that there is a line item-related promotion price derivation rule that contains a manual eligibility with trigger type "CO" and trigger value "333" which grants a 30% discount.
In case when a customer purchases a sun lotion on a hot day (more than 30^{o} Celsius outside), the client grants the customer an additional 30% discount on this item. For this purpose, the client provides a manual trigger with privilege type "AM" (trigger type and trigger value have to be "CO" and "333" respectively) on the item.
Then the line item-related promotion price derivation rule is triggered manually by the client, 4.50€ are granted, and the customer has to pay only 10.50€.

7.5.2 Manually Trigger and Set a Price Derivation Rule

The price derivation rule type (*PromotionConditionRuleSO.typeCode*) is "MA" for this manual benefit in the promotion master data. This means that the PCE expects the price derivation rule information in the transaction. To apply a manual benefit and to manually set the price derivation rule, the transaction includes a manual trigger with the additional attributes:

- For line item-related promotion price derivation rules:
  - Item level privilege type *(SaleReturnLineItemPromotionTrigger.privilegeType)*
  - Item level privilege value *(SaleReturnLineItemPromotionTrigger.privilegeValue)*
- For transaction-related promotion price derivation rules:
  - Transaction level privilege type *(RetailTransactionPromotionTrigger.privilegeType)*
  - Transaction level privilege value *(RetailTransactionPromotionTrigger.privilegeValue)*

As in the previously described scenario, the trigger type and trigger value have to be equivalent with the trigger type and trigger value in the manual eligibility. The price derivation rule is defined by the privilege type and privilege value of the manual trigger in the transaction. Thereby, the privilege type can have the following values:

- RP: reduction of the sales unit price by a percentage
- RS: reduction of the sales unit price by an amount
- PS: setting the discount price of the sales unit price

These values correspond to the values of the price modification method for the simple discounts and are computed likewise.

Note that it is not possible to manually grant loyalty points without a predefined price derivation rule. The amount, the percentage, or the discount price are stated in the privilege value of the transaction.

Example: Manual trigger triggered by a sales person
Let us assume that there is an item "T-Shirt" for 20.00€.
Let us also assume that there is a line item-related promotion price derivation rule that contains a manual eligibility with trigger type "CO" and trigger value "123" which grants a manual benefit.
In case a customer purchases a T-Shirt which has a make-up stain on it, the sales person can grant the customer an additional 5.00€ discount on this item. For this purpose, the sales person has to give a manual discount with privilege type "RS" and privilege value "5.00" (trigger type and trigger value have to be "CO" and "123" respectively) on the item.
Then the line item-related promotion price derivation rule is triggered manually by the sales person, 5.00€ is granted, and the customer has to pay only 15.00€.

7.5.3 Apply a Manual Benefit Several Times

It is possible that the same manual benefit is granted several times by the employee of the retail company. The corresponding transaction then includes several manual triggers with the same trigger type and trigger value that trigger the same promotion price derivation rule with the same sequence and resolution. The promotion price derivation rules would collide. In this case, the PCE takes the trigger sequence addend of the manual trigger into account to resolve the collision of the corresponding promotion price derivation rule. For this purpose, the trigger sequence addend includes unique values in the context of a transaction. This value is then added to the sequence of the promotion price derivation rule. This means, the higher the trigger sequence addend is, the later this manual trigger is consumed.

Example: Manual trigger several times
Let us assume that there is an item "Table" for 200.00€.
Let us also assume that there is a line item-related promotion price derivation rule that contains a manual eligibility with trigger type "CO" and trigger value "123", which grants a manual benefit. This promotion price derivation rule has the sequence 100.
In case a table has a scratch on it, the sales person can grant the customer an additional 5% discount on the table. Beside this, in case when some small piece of furniture like a screw is missing, the sales person can grant the customer an additional 3% discount.
Let us assume that a customer purchases such a table, the sales person gives a manual discount with privilege type "RP", privilege value "5.00", and manual trigger sequence addend "1" because of a scratch on a table. The sales person also gives a manual discount with privilege type "RP", privilege value "3.00", and manual trigger sequence addend "2" because of a missing screw. Trigger types and trigger values are "CO" and "123" respectively for both manual triggers.
The line item-related promotion price derivation rule is triggered twice manually by the sales person. To resolve the collision of these two promotion price derivation rules, the PCE

calculates the sequences as sum of the promotion price derivation rule sequence and the trigger sequence addend. Thereby, the sequence of the first promotion price derivation rule is considered as 101 and sequence of the second one as 102. This means that the customer gets at first a 5% discount and then a 3% discount on the purchased table.
As a result, 10.00€ and 6.00€ discounts are granted and the customer has to pay 184.00€ instead of 200.00€.

Note that manually-triggered promotion price derivation rules can collide with other, not-manually-triggered promotion price derivation rules as well. If this is not desired, avoiding it is possible by assigning the sequences of manually-triggered promotion price derivation rules from a sequence range reserved for them. It also should be considered that increasing the sequences from this range with the manual trigger sequence addend of the manually-triggered promotion price derivation rules still should not overlap with the sequence range assigned to not-manually-triggered promotion price derivation rules.

7.5.4 Manually Trigger a Promotion Price Derivation Rule at the Wrong Level

The manual trigger can either trigger a line item-related promotion price derivation rule or a transaction-related one. It depends on the position of the manual trigger in the transaction, which manual eligibility is activated. If the manual trigger is on line item level – at *SaleReturnLineItemPromotionTrigger* –, the promotion price derivation rule has to be a line item-related one (transaction control (*PromotionConditionRuleSO.transactionControlBreakCode*) is PO/PC). Otherwise, the manual trigger at line item level is ignored by the PCE. This means that the trigger will be part of the PCE-response, the manually triggered promotion price derivation rule is not applied, and the calculation of other promotions continues.

The same happens if the manual trigger on transaction level is part of the PCE-request – at *RetailTransactionPromotionTrigger* – and the corresponding promotion price derivation rule is line item-related. The manual trigger is ignored by the PCE. Thus, a manual trigger that is on transaction level can only trigger a transaction-related promotion price derivation rule (transaction control is SU/SP).

7.5.5 THERE IS MORE

A combination eligibility with the combination code (*CombinationPromotionConditionEligibilitySO.combinationCode*) AND that includes several manual eligibilities is not allowed. The corresponding promotion price derivation rule cannot be applied.

In case a manual eligibility is combined with a threshold eligibility (*ThresholdPromotionConditionEligibility*), the manual trigger of the transaction is consumed for each fulfilled interval. There have to be two manual triggers in the transaction in order to apply a promotion price derivation rule for two fulfilled intervals of the threshold eligibility. If there only exists one manual trigger in the transaction, the promotion price derivation rule is only applied once even though the interval of the threshold eligibility is fulfilled twice.

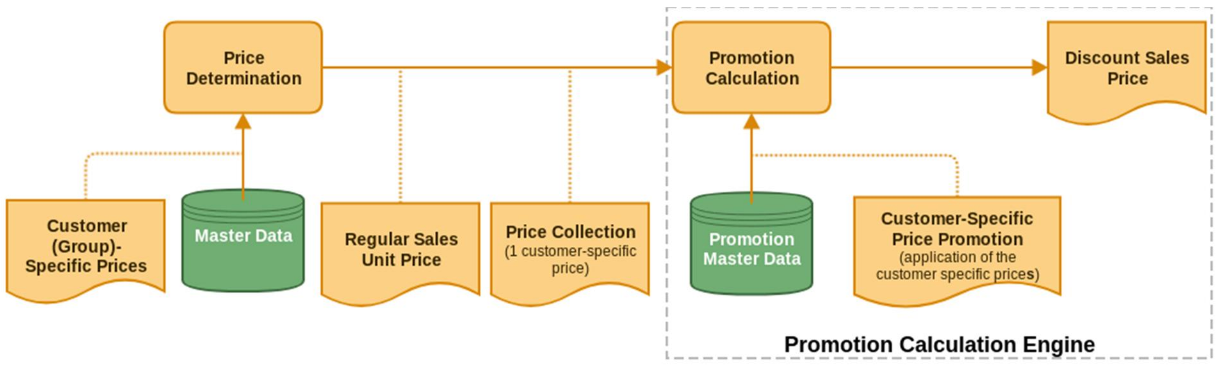
The PCE does not support the case that a manual trigger triggers a promotion price derivation rule in the promotion master data that contains a mix and match. This means that such a promotion price derivation rule is not applied.

Further information about the attributes can be found in chapter Configuration.

7.6 Apply Customer-Specific Prices

A wholesaler has agreed on particular prices with its customers – customer (group)-specific prices. The wholesaler has maintained these prices in the master data. In addition, one promotion for all customer specific prices is maintained in the promotion master data to apply these prices. This promotion enables the wholesaler to decide how and when the customer (group)-specific price shall be applied. The PCE does not differentiate between customer-specific and customer group-specific prices, these are all customer-specific prices.

The diagram below depicts the process of applying customer-specific prices.


Price Determination
Promotion Calculation
Discount Sales Price
Customer (Group)-Specific Prices
Master Data
Regular Sales Unit Price
Price Collection (1 customer-specific price)
Promotion Master Data
Customer-Specific Price Promotion (application of the customer specific prices)
Promotion Calculation Engine

Before the PCE is called, the calling instance determines which of the possible customer (group)-specific prices may be applied to the transaction. In the diagram above called "Price Determination". This determination results in one customer-specific price in addition to the regular sales unit price of an item. If there exist several customer-specific prices on one line item in the transaction, the PCE does not apply any of these. This additional price is provided to the PCE via a price collection and together with the regular sales unit price which is not part of the price collection. Each line item, which shall "get" a customer-specific price, has a price collection attached to it. A promotion is calculated to apply the customer-specific price to an item – herein called customer-specific price promotion . This step is labelled with "Promotion Calculation" in the diagram above. As a result, the customer-specific price is applied to a line item and attached to the transaction like a discount sales price.

The following wording is used in the subsequent sections:

Term	Description
Price	Additional price value – like the value of the customer specific price; **e.g. 5.00€** (*SaleReturnLineItem.SaleReturnLineItemAdditionalPriceList.SaleReturnLineItemAdditionalPrice.PriceAmount*)
Price Type Trigger	Additional price type as **part of the transaction** – e.g. indicator that it is a customer specific price; **e.g. "CS"** (*SaleReturnLineItem.SaleReturnLineItemAdditionalPriceList.SaleReturnLineItemAdditionalPrice.PriceTypeCode*)
Customer -Specific Price	The pairing of the price and the price type (trigger); **e.g. 5.00€ and "CS"**
Price Collection	A collection/list of particular prices – like the customer-specific prices – in the transaction
Price Type	Additional price type as **part of the promotion** – e.g. indicator that it is a customer specific price; **e.g. "CS"** (*AdditionalPriceTypePromotionConditionEligibilitySO.additionalPriceTypeCode*)

HOW IT WORKS

A customer-specific price is defined by its amount – price – and by a price type trigger that is CUSTOMER_SPECIFIC/"CS". The calling instance of the PCE attaches to the price the corresponding price type trigger in the price collection. It determines the customer-specific prices that satisfy the subsequent statements:

1. There exists at most one customer-specific price for one line item and one line item in the transaction. The price collection is added to each line item that has a customer-specific price even though the line items include the same item.
2. The determined price has the same currency than the regular sales unit price of the related item.

In the promotion master data, the customer-specific price promotion has to exist. Otherwise, no customer-specific price is applied by the PCE. This promotion refers to the price derivation rule eligibility which has the price type CUSTOMER_SPECIFIC/"CS" – in the following called customer-specific price eligibility. If a transaction includes a customer-specific price as part of the price collection, the customer-specific price promotion is triggered and the corresponding price derivation rule is applied. It is necessary that the price derivation rule is of rule type (*PromotionConditionRuleSO.priceType*) "AP" – herein price type rule. Thereby, the calculation is similar to the calculation of a price derivation rule containing the price modification method (*RebatePromotionConditionRuleSO.priceModificationMethodCode*) with the following value, even though this value is not set in the price type rule:

- FIXED_PRICE/"PS": defining the discount sales unit price

The utilization of a customer-specific price promotion enables the wholesaler to use the flexibility one promotion provides. Since some promotion attributes will never be related to the usage of a customer-specific price, the following restrictions for the customer-specific price promotion exist:

- Only usable as line item-related promotion price derivation rule (transaction control (*PromotionConditionRuleSO.transactionControlBreakCode*) is PO always)
- Not applicable in the context of loyalty points (bonus points flag *(PromotionConditionRuleSO.bonusPointsFlag)* is false always)
- Customer-specific price eligibility cannot be combined with any other kind of price derivation rule eligibilities
- Customer-specific price eligibility is only activated with a price type rule as part of the promotion
- Price type rule can only change the total amount of the transaction (discount methods (*PromotionConditionRuleSO.discountMethodCode*) is "00" always)

If the customer-specific price promotion does not meet the restrictions and the price type trigger CUSTOMER_SPECIFIC exists, the PCE does not activate the related eligibility and does not apply the price.

The PCE provides a retail price modifier (*RetailPriceModifier*) in the response to the calling instance. This retail price modifier can be used to evaluate: how often a customer-specific price was applied and/or how much the benefit for its consumers was. The promotion price derivation rule information in the retail price modifier is enhanced with the applied price type – here CUSTOMER_SPECIFIC.

All other content of the retail price modifier and included information about applied promotion price derivation rules are filled as usual – like for any other applied promotion price derivation rule.

Example

- Regular sales unit price for product “Washing Powder” is 20€
- Customer A is eligible for getting a customer-specific price for product “Washing Powder” of 15.00€
  - Customer-specific price promotion:
    - Sequence of the promotion is 200
    - Calculation base sequence: -2
    - Discount Method Code is 00
    - Transaction Control Break Code is PO
- Result: The customer A has to pay the customer-specific price and customer B has to pay regular sales unit price.

Customer A			
	Sequence	**Price/ Discount**	
1 package “Washing Powder”		20.00€	Regular sales price
Customer-specific price	200	15.00€/ - 5.00€	Customer-specific price applied
Effective sales price		15.00€	

Customer B			
	Sequence	**Price/ Discount**	
1 package “Washing Powder”		20.00€	Regular sales price
Customer-specific price	200	-	Customer-specific price not applied
Effective sales price		20.00€	

Subsequently, the following scenarios are described in more detail and can be used as a starting point for customization. This means, that the PCE supports these cases without any further adjustments only with the correctly set system parameters and promotion master data:

1. Apply the customer-specific prices in any case
2. User either the customer-specific prices or apply promotion price derivation rules
3. Use customer-specific prices in place of regular sales unit prices
4. Calculate the discount of customer-specific prices independently from the order of the promotion price derivation rules

7.6.1 Apply the Customer-Specific Prices in any Case

A wholesaler may want that the customer-specific price is paid in any case if customer (group)-specific prices are negotiated. There exist two approaches to achieve this. One approach is based on the application of the customer-specific prices at the beginning of the promotion calculation and the other way is based on the application of these prices at the end. The subsequent sections describe in more detail which parameters and attributes of the customer-specific price promotion need to be set to achieve this scenario.

To achieve the applying of the customer-specific price promotion in any case, the system parameter *allowZeroRebate* set to true. This parameter decides whether the customer-specific prices are applied to a line item at all (true) or not (false) in case there is no change in the sales price. The value needs to be explicitly changed since the default value is false.

The same applies to the increase of price allowed flag (*PromotionConditionRuleSO.increaseOfPriceAllowed*). This attribute enables or disables an increase in the sales unit price by a customer-specific price. This scenario requires to enable the price increase and thus, the increase of price allowed flag needs to be true which is not the default value.

Note that the increase of price allowed flag is only evaluated by the PCE in case of a price type rule.

This scenario is only possible if there do not exist any collisions with the customer-specific price promotion – regardless of the chosen approach. The sequence (*PromotionConditionSO.sequence*) of the customer-specific price promotion has to be unique compared to the sequence of all other promotion price derivation rules in the promotion master data.

There exist one case where it is possible that the customer-specific price promotion is not applied to a line item. This case cannot be prohibited by neither configuration of the customer-specific price promotion nor by the adjustment of system parameters. This case occurs whenever a manual price is applied. There exist three ways to apply a manual price:

- The line item has attached a manual trigger (*SaleReturnLineItemPromotionTrigger*) that sets a price derivation rule. If the manual trigger has the item level privilege type PS (setting the discount price of the sales unit price; *SaleReturnLineItemPromotionTrigger.privilegeType*), the customer-specific price is not applied to the line item.
- The cashier decides to apply to an item a new price due to a missing price, or a mark or something else. The line item includes already a retail price modifier (*RetailPriceModifier*) with the calculation method MAPO (*RetailPriceModifier.calculationMethodCode*) before the PCE is called.
- The line item has attached a manual trigger that triggers a predefined price derivation rule. If the price derivation rule has one of the following price modification methods, the customer-specific price is not applied to the line item:

    o FIXED_PRICE/"PS": defining the discount sales unit price

    o FIX_PRICE_TOTAL/"PT": defining the discount sales price

    o SET_PRICE_TOTAL/"ST": see Package an Offer

Example: Manual Promotion Granted By Sales Person

- Manual promotion at POS: discount price 80€ for closet “Berlin” as it has a scratch on it

- Customer specific price: customer A pays for product “Closet Berlin” 100€ (regular sales price 150€)
- Promotion: 10% off on all closets in the store “Heidelberg”
- The wholesaler has decided that first the manual price, then the customer specific price, and then the promotion are applied
- Expectation: the manual price gets applied and the promotion. The customer specific price doesn’t get applied.

Customer A comes to the store “Heidelberg”		
	Price/ Discount	
1 closet “Berlin”	150€	Regular sales price
Manual price	*80€/ - 70.00€*	*Manual price gets applied (manual price promotion)*
Customer specific price	100€/ 20.00€	Customer specific price – not applied
Promotion – 10 %	*72.00€/ - 8.00€*	*Promotion applied*
Effective sales price	72€	

Example: Manual Price Override (MAPO) and Customer Specific Price

- Manual price override at POS: regular sales price 120€ for closet “Berlin” typed in by the sales person as the price is missing in the POS
- Customer specific price: customer A pays for product “Closet Berlin” 100€
- Promotion: 10% off on all closets in the store “Heidelberg”
- Expectation: the manual overridden price gets applied and the promotion. The customer specific price doesn’t get applied.

Customer A comes to the store “Heidelberg”		
	Price/ Discount	
1 closet “Berlin”	–	
Manual Price Override	120€	Regular sales price

Customer specific price	100€/ - 20.00€	Customer specific price – not applied
Promotion – 10 %	*108.00€/ - 12.00€*	*Promotion applied*
Effective sales price	108€	

Example: Manually Triggering a Promotion That Sets the Price by a Sales Person

- Manually triggered promotion at POS: discount price 80€ for closet "Berlin"
- Customer specific price: customer A pays for product "Closet Berlin" 100€
- Promotion: 10% off on all closets in the store "Heidelberg"
- Expectation: the manual price and the promotion gets applied. The customer specific price doesn't get applied.

Customer A comes to the store "Heidelberg"		
	Price/ Discount	
1 closet "Berlin"	150€	
Customer specific price	100€/ - 50.00€	Customer specific price – not applied
Manually triggered promotion	*80€/ - 70.00€*	
Promotion – 10 %	*72.00€/ - 8.00€*	*Promotion applied*
Effective sales price	72€	

7.6.1.1 Use Customer-Specific Prices at the Beginning of the Promotion Calculation

Customer-specific price is applied at first, but exclusively. No further promotion price derivation rules will be calculated for the line item after the customer-specific price has been applied to it. The sequence of the customer-specific price promotion needs to be the smallest sequence available in the promotion master data. In addition, the exclusive flag (*PromotionConditionSO.exclusiveFlag*) must be true to prevent that another promotion price derivation rule is applied to the line item.

The subsequent attributes of the customer-specific price promotion should be set accordingly to achieve this. Please, have a look into the chapter Configuration for a description of these. Attributes are not mentioned in this table which are not relevant for this scenario:

Attribute	Value	Note
allowZeroRebate	**true**	Need to be set explicitly.
Sequence	Smallest value compared to all other sequences in the promotion master data. (Please, also consider ad hoc promotions in this case, like US manufacturer coupons)	
Increase of Price Allowed Flag	**true**	
Exclusive Flag	**true**	
Sale Return Type (*PromotionConditionSO.saleReturnTypeCode*)	SALES_RETURNS/"00"	Do not need to be set explicitly.
Prohibit Transaction-Related Promotions Flag (*PromotionConditionRuleSO.prohibitTransactionRelatedPromotionConditionFlag*)	**false**	
No Effect on Successors Flag (*PromotionConditionRuleSO.noEffectOnSubsequentPromotionConditionFlag*)	**false**	
Consider Predecessors Flag (*PromotionConditionRuleSO.considerPreviousPromotionConditionFlag*)	**true** (recommended)	
No Previous Monetary Discount Flag (*PromotionConditionRuleSO.noPreviousMonetaryDiscountAllowedFlag*)	**false**	
Calculation Base Sequence (*PromotionConditionRuleSO.calculationBaseSequence*)	-2 (recommended)	
Bonus Points Flag	**false**	Set to pass the restrictions for the customer-specific price promotion.
Transaction Control	PO	
Discount Method	"00"	

Example

- Regular sales unit price for product “Tooth Paste Super White” is 5.00€
- Customer A is eligible for getting a customer-specific price for product “Tooth Paste Super White” of 4.00€
    - o *allowZeroRebate*: true

- Customer-specific price promotion:
    - Smallest sequence
    - Increase of Price Allowed Flag: true
    - Exclusive flag: true
- Promotion: Get 10% off on merchandise category “Tooth Paste”
- The wholesaler has decided that the customer-specific price is applied at first and the promotion price derivation rule afterwards
- Result: For customer A, the customer-specific price is applied and the promotion price derivation rule is not. The customer A has to pay the customer-specific price.

Customer A			
	Sequence	**Price/ Discount**	
1 package “ Tooth Paste Super White ”		5.00€	Regular sales price
Customer-specific price	-200	4.00€/ - 1.00€	Customer-specific price applied
Promotion – 10 %	1	*3.60€/ - 0.40€*	*Promotion – not applied since the customer-specific price promotion is exclusive.*
Effective sales price		4.00€	
Customer B			
1 package “ Tooth Paste Super White ”		5.00€	Regular sales price
Promotion – 10 %	1	*4.50€/ - 0.50€*	*Promotion applied*
Effective sales price		4.50€	

7.6.1.2 Use Customer-Specific Prices at the End of the Promotion Calculation

The customer-specific prices are applied at last. Other promotion price derivation rules can be applied before. The customer-specific price promotion is not allowed to be only applicable to fully-priced items. From this follows that the no previous monetary discount flag has to be false of the customer-specific price promotion. In addition, no other promotion price derivation rule is allowed to be exclusive.

The subsequent attributes of the customer-specific price promotion should be set accordingly to achieve this. Please, have a look into the chapter Configuration for a description of these attributes. The attributes are not mentioned in this table which are not relevant for this scenario:

Attribute	Value	Note
allowZeroRebate	**true**	Need to be set explicitly.
Sequence	Greatest value compared to all other sequences in the promotion master data. (Please, also consider ad hoc promotions in this case, like US manufacturer coupons)	
Increase of Price Allowed Flag	**true**	
No Previous Monetary Discount Flag	**false**	
Calculation Base Sequence	-2	
Consider Predecessors Flag	**true**	
Sale Return Type	SALES_RETURNS/"00"	Do not need to be set explicitly.
Prohibit Transaction-Related Promotions Flag	**false**	
No Effect on Successors Flag	**false**	
Exclusive Flag	**false**	
Bonus Points Flag	**false**	Set to pass the restrictions for the customer-specific price promotion.
Transaction Control	PO	
Discount Method	"00"	

Example

- Regular sales unit price for product "Washing Powder" is 20€
- Customer A is eligible for getting a customer-specific price for product "Washing Powder" of 15.00€
  - *allowZeroRebate*: true
  - Customer-specific price promotion:
    - Greatest sequence
    - Increase of Price Allowed Flag: true
    - No previous monetary discount flag: false
    - Calculation base sequence: -2
    - Consider predecessors flag: true
- Promotion #1 applicable: Get 10% off on product "Washing Powder"

- Promotion #2 applicable: Get 50% off on product “Washing Powder”
- The wholesaler has decided that first promotion #1, then the promotion #2, and at least the customer-specific price are applied
- Result: The promotion #1, promotion #2, and the customer-specific price are applied. The customer A has to pay the customer-specific price even though it is a price increase.

Customer A			
	Sequence	**Price/ Discount**	
1 package “Washing Powder”		20.00€	Regular sales price
Promotion #1 – 10%	0	18.00€/ - 2.00€	*Promotion applied*
Promotion #2 – 50 %	1	*9.00€/* *- 9.00€*	*Promotion applied*
Customer-specific price	200	15.00€/ **+** 6.00€	Customer-specific price applied
Effective sales price		15.00€	

7.6.2 Use either the Customer-Specific Prices or Apply Promotion Price Derivation Rules

The customer-specific price promotion has to be exclusive and only applicable to fully-priced items. This means that the no previous monetary discount flag and the exclusive flag are true. The behavior is independent of the calculation order of the customer-specific prices and promotion price derivation rules. The promotion price derivation rules do not need to be exclusive to prevent the application of a customer-specific price. These promotion price derivation rules also do not need to be only applicable to fully-priced items. The wholesaler can define with the calculation order in relation to the sequence of the customer-specific price promotion whether either the promotions are applied or the customer-specific price.

The no previous monetary discount flag makes sure that whenever a promotion price derivation rule is applied to a line item before the computation of the customer-specific price, the customer-specific price is not applied.

The exclusive flag ensures that whenever a customer-specific price is applied to a line item, no further promotion price derivation rules can be applied to this particular line item.

The subsequent attributes of the customer-specific price promotion should be set accordingly to achieve this. Please, have a look into the chapter Configuration for a description of these attributes. The attributes are not mentioned in this table which are not relevant for this scenario:

Attribute	Value	Note
No Previous Monetary Discount Flag	**true**	Need to be set explicitly.
Exclusive Flag	**true**	
allowZeroRebate	**false**	Do not need to be set explicitly.
Sequence	Greatest/smallest value or something in between compared to all other sequences in the promotion master data. (Please, also consider ad hoc promotions in this case, like US manufacturer coupons)	
Increase of Price Allowed Flag	**false**	
Calculation Base Sequence	-2 (recommended)	
Consider Predecessors Flag	**true** (recommended)	
Sale Return Type	SALES_RETURNS/"00"	
Prohibit Transaction-Related Promotions Flag	**false**	
No Effect on Successors Flag	**false**	
Bonus Points Flag	**false**	Set to pass the restrictions for the customer-specific price promotion.
Transaction Control	PO	
Discount Method	"00"	

Example

- Regular sales unit price for product “Tooth Paste Super White” is 5.00€
- Customer A is eligible for getting a customer specific price for product “Tooth Paste Super White” of 4.00€
  - Customer-specific price promotion:
    - Exclusive flag: true
    - No previous monetary discount flag: true
- Promotion #1 available: Get 10% off on merchandise category “Tooth Paste” only.
- Promotion #2 available: Get 25% off on merchandise category “Tooth Paste”
- The wholesaler has decided that first promotion #1, then the customer-specific price, and finally promotion #2 are applied

- Result: the promotion #1 is applied, but not the customer-specific price, and promotion #2 is applied. In this case only the promotion price derivation rules are applied.

Customer A			
	Sequence	**Price/ Discount**	
1 package “Tooth Paste Super White”		5.00€	Regular sales price
Promotion #1 – 10 %	0	*4.50€/ - 0.50€*	*Promotion #1 applied*
Customer-specific price	1	4.00€/ - 0.50€	*Customer-specific price – not applied since it is only applicable to fully-priced line item and 1 package "Tooth Paste Super White" is reduced by promotion #1 (no longer fully-priced).*
Promotion #2 – 25%	2	*3.37€/ - 1.13€*	*Promotion #2 applied*
Effective sales price		3.37€	

If we would assume the same scenario except that promotion #1 is not valid, the calculation result would look like the following:

- Result: the customer-specific price is applied and the promotion #2 is not applied. In this case only the customer-specific price is used.

Customer A			
	Sequence	**Price/ Discount**	
1 package “Tooth Paste Super White”		5.00€	Regular sales price
Customer-specific price	1	4.00€/ - 1.00€	*Customer-specific price applied*
Promotion #2 – 25%	2	*3.00€/ - 1.00€*	*Promotion #2 not applied since the customer-specific price promotion is exclusive.*
Effective sales price		4.00€	

7.6.3 Use Customer-Specific Prices in Place of Regular Sales Unit Prices

The calculation base sequence of the price type rule itself is of no relevance since the customer-specific price promotion needs to be calculated by the PCE before any other promotion. In contrast, this attribute is significant for all other promotion price derivation rules in the promotion master data. The calculation base sequence should not be -1 for any one. Promotion price derivation rules with a calculation base sequence of -1 use the regular sales unit prices as basis for the benefit calculation and thus, not the customer-specific price. Please refer to Set the Calculation Base Amount to see how the calculation base sequence can be set and what the transaction can look like.

The subsequent attributes of the customer-specific price promotion should be set accordingly to achieve this. Please, have a look into the chapter Configuration for a description of these attributes. The attributes are not mentioned in this table which are not relevant for this scenario:

Attribute	**Value**	**Note**
Sequence	Smallest value compared to all other sequences in the promotion master data. (Please, also consider ad hoc promotions in this case, like US manufacturer coupons)	Need to be set explicitly.
No Previous Monetary Discount Flag	**false**	Do not need to be set explicitly.
Exclusive Flag	**false**	
allowZeroRebate	**false**	
Increase of Price Allowed Flag	**false**	
Calculation Base Sequence	-2 (recommended)	
Consider Predecessors Flag	**true** (recommended)	
Sale Return Type	SALES_RETURNS/"00"	
Prohibit Transaction-Related Promotions Flag	**false**	
No Effect on Successors Flag	**false**	
Bonus Points Flag	**false**	Set to pass the restrictions for the customer-specific price promotion.
Transaction Control	PO	
Discount Method	"00"	

The subsequent attributes of the other promotion price derivation rules should be set accordingly to achieve this. Please, have a look into the chapter Configuration for a description of these attributes. The attributes are not mentioned in this table which are not relevant for this scenario:

Attribute	Value	Note
Sequence	Greater than the sequence of the customer-specific price promotion.	Need to be set explicitly.
Calculation Base Sequence	<> -1 (shall not be -1)	
Consider Predecessors Flag	**true** (recommended)	

Example

- Regular sales unit price for product “Tooth Paste Super White” is 5.00€
- Customer A is eligible for getting a customer specific price for product “Tooth Paste Super White” of 4.00€:
  - Customer-specific price promotion:
    - Smallest sequence
- Promotion #1 available: Get 10% off on merchandise category “Tooth Paste”
  - Calculation base sequence: -2 (lastly computed discount sales price)
  - Consider predecessors flag: true
- Promotion #2 available: Get 25% off on merchandise category “Tooth Paste”
  - Calculation base sequence: -2
  - Consider predecessors flag: true
- The wholesaler has decided that first the customer-specific price, then promotion #1, and at least promotion #2 are applied.
- Result: The customer-specific price is applied at first and afterwards all promotion price derivation rules. The calculation of the discounts base on the previously computed discount sales prices.

Customer A				
	Order	**Calculation Base Sequence**	**Price/ Discount**	
1 package “Tooth Paste Super White”			5.00€	Regular sales price
Customer-specific price	0	-2	4.00€/ - 1.00€	*Customer-specific price applied*
Promotion – 10 %	1	-2	*3.60€/ - 0.40€*	*Promotion #1 applied*
Promotion – 25%	42	-2	*2.70€/ - 0.90€*	*Promotion #2 applied*
Effective sales price			2.70€	

If the wholesaler wants that the aforementioned promotion price derivation rules use the customer-specific price as calculation base amount, regardless of their order, the calculation base sequence has to be the sequence of the customer-specific price promotion. In addition, the customer-specific price promotions sequence should not be a negative value:

- Promotion #1 and promotion #2: calculation base sequence is the smallest sequence

If we would assume the same scenario except the changes mentioned directly above, the calculation result would look like the following:

Customer A				
	Sequence	**Calculation Base Sequence**	**Price/ Discount**	
1 package "Tooth Paste Super White"			5.00€	Regular sales price
Customer-specific price	0 (should not be negative)	-2	4.00€/ - 1.00€	*Customer-specific price applied*
Promotion – 10 %	1	0 (use the result of the promotion price derivation rule with the sequence 0; in this case the customer-specific price promotion)	*3.60€/ - 0.40€*	*Promotion #1 applied (10% of the customer-specific price)*
Promotion – 25%	42	0 (use the result of the promotion price derivation rule with the sequence 0; in this case the customer-specific price promotion)	*2.60€/ - 1.00€*	*Promotion #2 applied (25% of the customer-specific price)*
Effective sales price			2.60€	

7.6.4 Calculate the Differences of Customer-Specific Prices Independently from the Order of the Promotion Calculation

The calculation of promotions is based on the sequences of each promotion price derivation rule in the promotion master data. The promotion price derivation rule with the smallest sequence is executed by the PCE first and the one with the greatest sequence is computed at last. As a conclusion, the later a promotion price derivation rule is executed, the more its result is influenced by the previously executed promotions. This also affects the customer-specific price promotion.

It is possible to circumvent this with the help of the calculation base sequence attribute. In this case, the difference is the difference between the regular sales unit price and the customer-specific price multiplied by the line items quantity – independently, whether other promotions were applied before or not. Therefore, the calculation base sequence of the customer-specific price promotion should be -1. The customer-specific prices could be applied at the beginning of the promotion calculation, somewhere in between, or at the end, it always results in the same difference.

In addition, the difference is subtracted from the previously computed discount sales price and thus, the result may not be the defined customer-specific price but something less or greater – depending whether the price increases or not. This may even lead to negative prices for line items, as long as the difference is greater than the previously calculated discount sales price and less than the regular sales price.

Attribute	**Value**	**Note**
Calculation Base Sequence	-1	Need to be set explicitly.
Consider Predecessors Flag	**true**	
allowZeroRebate	**false**	Do not need to be set explicitly.
Sequence	Greatest/smallest value or something in between compared to all other sequences in the promotion master data. (Please, also consider ad hoc promotions in this case, like US manufacturer coupons)	
Increase of Price Allowed Flag	**false**	
No Previous Monetary Discount Flag	**false**	
Sale Return Type	SALES_RETURNS/"00"	
Prohibit Transaction-Related Promotions Flag	**false**	
No Effect on Successors Flag	**false**	
Exclusive Flag	**false**	
Bonus Points Flag	**false**	Set to pass the restrictions for the customer-specific price promotion.
Transaction Control	PO	
Discount Method	"00"	

Example

- Regular sales unit price for product "Washing Powder" is 20€
- Customer A is eligible for getting a customer-specific price for product "Washing Powder" of 15.00€

- Customer-specific price promotion:
  - Calculation base sequence: -1
  - Consider predecessors flag: true
- Promotion #1 applicable: Get 10% off on product “Washing Powder”
- Promotion #2 applicable: Get 50% off on product “Washing Powder”
- The wholesaler has decided that first promotion #1, then the promotion #2, and at least the customer-specific price are applied
- Result: The promotion #1, promotion #2, and the customer-specific price are applied. The computed difference for the customer-specific price is independent of the execution order of it. The prices could be applied at the beginning of the promotion calculation, somewhere in between, or at the end, it always results in a difference of - 5.00€.

Customer A				
	Sequence	**Calculation Base Sequence**	**Price/ Discount**	
1 package “Washing Powder”			20.00€	Regular sales price
Promotion #1 – 10%	0	-2 (use latest computed discount sales price)	18.00€/ - 2.00€	*Promotion applied*
Promotion #2 – 50 %	100	-2	*9.00€/ - 9.00€*	*Promotion applied*
Customer-specific price	200	-1 (use regular sales price)	4.00€/ **- 5.00€**	Customer-specific price applied • *5.00€ difference (regular sales price 20.00€ and customer-specific price 15.00€ results in 5.00€ discount)*
Effective sales price			4.00€	

7.6.5 THERE IS MORE

The application of the customer-specific prices depends on the configuration of the customer-specific price promotion. This leads to some use cases where the customer-specific prices are not applicable to the corresponding line item. The following list states the possible exceptions:

- A manual price is applied to the line item, regardless of the execution order.
- The customer-specific price promotion is not applied after a promotion price derivation rule was applied that has the exclusive flag set to true.

- The customer-specific price promotion collides with another promotion price derivation rule that grants a better benefit. Please refer to Select a Promotion Price Derivation Rule in Case of a Collision.
- The customer-specific price promotion does not result in a benefit and the system flag *allowZeroRebate* is false. Please, refer to Apply a Zero Discount.
- The customer-specific price promotion results in a price increase and the attribute increase of price allowed flag is false.

It can also happen that a promotion price derivation rule is not applied since the customer-specific price promotion was processed before.

- Promotion price derivation rules applicable only to fully-priced items are not applicable to line items with a customer-specific price.
- The customer-specific price promotion has the exclusive flag true and no further promotion price derivation rules can be applied to the line items with a customer-specific price.

7.7 Mix and Match

Typically, the PCE applies a benefit to the eligible line items that trigger the corresponding price derivation rule eligibility. The price derivation rule type mix and match is an exception: It is possible that triggers in the transaction do not get the benefit, but the items, merchandise categories, and product groups which are defined in the price derivation rule – the rule matching items (*MatchingItemSO*). This price derivation rule is herein called mix and match (*MixAndMatchPromotionConditionRuleSO*). As a result, the transaction has to contain the triggers for the price derivation rule eligibility and also the transaction matching items. The transaction matching items (*SaleReturnLineItem)* correspond to the rule matching items in the price derivation rule.

HOW IT WORKS

The rule matching items are defined in the mix and match in the promotion master data. A rule matching item is defined by the matching item identifier (*MatchingItemSO.matchingItemID*). A rule matching item can be the following:

- An item
- A merchandise category
- A product group

What the rule matching item includes exactly, is defined in the matching item relation (*MatchingItemRelationSO*). In addition, a rule matching item contains the mix and match price modification *(MatchingItemSO.reductionMethodCode*) that indicates how the discount is computed for the corresponding transaction matching item. It is restricted to the following values:

- REBATE_SINGLE/"RS": reduction of the sales unit price by an amount

- REBATE_PERCENT/"RP": reduction of the sales unit price by a percentage
- PRICE_SINGLE/"PS": setting the discount sales unit price of the regular sales unit price

Thereby, the PCE calculates and prorates the discount like described in Apply a Simple Discount and Prorate the Benefit. Since the mix and match price modification is related to a rule matching item, the mix and match price modification can be different for each one. The mix and match price modification percent (*MatchingItemSO.priceModificationPercent*), mix and match price modification amount (*MatchingItemSO.priceModificationAmount*), or the mix and match new price (*MatchingItemSO.newPriceAmount*) are filled with a value in accordance to the mix and match price modification by the PCE.

In the OPP/PPS scenario, the PCE maps this value related to the actual mix and match price modification from the promotion master data to the mix and match price modification value (*MatchingItemSO.PriceReductionMethode*) and one of the following attributes:

- Mix and match price modification amount,
- mix and match price modification percent, and
- mix and match new price.

So the same value will be mapped to two different attributes of the PCE internal model.

The combination of several rule matching items is also necessarily part of the price derivation rule. It is defined by the mix and match combination code (*MixAndMatchPromotionConditionRuleSO.combinationCode*). This code has the following possible values:

- OR: combination via a logical OR operator
- AND: combination via a logical AND operator
- OR_QUANTITY: combination via an exclusive OR (XOR) operator with a certain threshold quantity

The evaluation of transaction matching items is done likewise to the comparison of triggers with the eligible line items. It is decided based on the same attributes and procedure whether a line item matches a rule matching item and is thus a transaction matching item. The PCE sorts the rule matching items according to their matching item identifier. The rule matching item with the smallest matching item identifier is processed first.

Example: Select Transaction Matching Items in Case Sorting Is Not Performed
The following example shows the possible inconsistent behavior if sorting of rule matching items would not be performed by the PCE.
The promotion master data defined for the test case contain a mix and match with two rule matching items and with the mix and match combination code AND. Both rule matching items contain product group

identifiers (*MerchandiseSetPromotionConditionEligibilitySO.merchandiseSetID*) to identify rule matching items: 472021 for the rule matching item #1 and 472023 for rule matching item #2. Both rule matching items require one transaction matching item that is part of the product group in order to be applied. The transaction contains two line items with equal regular sales prices:

- Line item #11
- Line item #12

Product group 472021 (rule matching item #1) includes both line items of the transaction. Product group 472023 (rule matching item #2) includes only line item #12 of the transaction.
Scenario 1:
Rule matching item #1 is processed first.
The line item #12 will be used for the calculation, as both line items are transaction matching items for rule matching item #1 and the regular sales prices are the same for both items. In this case, the line item with the highest sequence in the transaction is taken (see Choose Items). This leaves rule matching item #2 with zero transaction matching items, the promotion price derivation rule *will not be applied*.
Scenario 2:
Rule matching item #2 is processed first.
The line item #12 will be used for the calculation, as only this line item is a transaction matching item for rule matching item #2. This leaves rule matching item #1 with 1 transaction matching item – line item #11. The promotion price derivation rule *will be applied*.
Conclusion:
The PCE would behave inconsistently if the rule matching items were not sorted before: The possible benefit for the customer would depend on the order in which the rule matching items are processed and this order can differ. Thus, the PCE sorts the rule matching items according to their matching item identifier.

There also exist different thresholds and limits for the rule matching items. The subsequent table summarizes the different mix and match combination codes and the possible limits and thresholds that are checked by the PCE.

Mix and Match combination code	Threshold Quantity per Rule Matching Item	Limit Quantity per Rule Matching Item
OR	1	**Mix and match limit count** (*MixAndMatchPromotionConditionRuleSO.limitCount*)
AND	**Required quantity** (*MatchingItemSO.requiredQuantity*) or 1 if the required quantity is empty	**Required quantity** or 1 if the required quantity is empty

Mix and Match combination code	Threshold Quantity per Rule Matching Item	Limit Quantity per Rule Matching Item
OR_QUANTITY	**Required quantity** or 1 if the required quantity is empty	**Required quantity** or 1 if the required quantity is empty

Thereby, for each mix and match promotion price derivation rule the mix and match combination code has to be defined with corresponding attributes which have to be considered by calculation.

The combination of the rule matching items can result in the following behaviors of the PCE:

1. Combine rule matching items with a logical OR,
2. combine rule matching items with a logical AND,
3. combine a certain threshold quantity of rule matching items with an exclusive OR.

7.7.1 Combine Rule Matching Items with a Logical OR

The mix and match combination code is OR. The benefit can be applied if one or several rule matching items are part of the processed transaction. The discount is computed according to the mix and match price modification. The PCE chooses at least one rule matching item of the mix and match.

The sorting order of the rule matching items in the promotion master data depends on the matching item identifier. The rule matching item with the smallest matching item identifier is chosen first. The sorting order in which the transaction matching items are considered depends on the choose item method *(PromotionConditionRuleSO.chooseItemMethod) for selecting a normalized item.*

Therefore, if the transaction includes several transaction matching items, the PCE selects the transaction matching items according to the corresponding rule matching item with the smallest matching item identifier first. If several transaction matching items can still be selected afterward, the choose item method *(PromotionConditionRuleSO.chooseItemMethod)* is considered for selecting a normalized item.

Note that all the applicable mix and match price modifications refer to the regular/discount sales unit price of an item. The mix and match limit count defines the maximum quantities of transaction matching items that can get a discount. The PCE only checks this value for this particular mix and match combination code. If the limit count is not reached with the application of the first processed rule matching item, the PCE tries to apply the following rule matching items until the limit count is reached.

Example: Combine Rule Matching Items with a Logical OR
If the price derivation rule includes several rule matching items (item "pasta sauce" and item "basil") with the mix and match combination code OR, the behavior is as follows: The promotion states, for example, that the pasta sauce gets 20% off or the basil gets 50% off if

the customer buys one package of noodles. If the transaction includes the subsequent items and we assume that the mix and limit count of the mix and match rule is 10:

- Line item "pasta sauce", "basil", and "noodles" with a quantity 1 for each:
  - Item "pasta sauce" gets 20% off and item "basil" gets 50% off.
  - The limit count has not been reached but no further transaction matching items are available.
- Line item "pasta sauce" and "noodles" with a quantity 1 for each:
  - Item "pasta sauce" gets 20% off.
- Line item "basil" and "noodles" with a quantity 1 for each:
  - Item "basil" gets 50% off.

If the mix and match limit count is set, for example, to 1 and the transaction contains the following items, the subsequent discount is computed:

- Line item "pasta sauce", "basil", and "noodles" with a quantity 1 for each:
  - Either item "pasta sauce" gets 20% off if item "pasta sauce" has a smaller matching item identifier in the price derivation rule than item "basil"
  - or item "basil" gets 50% off if item "basil" has a smaller matching item identifier in the price derivation rule than item "pasta sauce".
  - The discount is not applied to both transaction matching items because the mix and match limit count states 1 as a maximum.

- Line item "pasta sauce" and "noodles" with a quantity 2 for the "pasta sauce" and a quantity 1 for the "noodles":
  - Line item "pasta sauce" gets 20% off for the quantity 1.
- Line item "basil" and "noodles" with a quantity 3 for the "basil" and a quantity 1 for the "noodles":
  - Line item "basil" gets 50% off for the quantity 1.

7.7.2 Combine Rule Matching Items with a Logical AND

The mix and match combination code is AND. The benefit can only be applied if all transaction matching items are part of the currently processed transaction and correspond to all rule matching items in the mix and match. The discount is computed according to the mix and match price modification. The mix and match limit count is not considered. A threshold quantity of rule matching items is checked. This threshold quantity is defined by the required quantity (*MatchingItemSO.requiredQuantity*). If the required quantity is equal to null, 1 is used instead. No less and no more transaction matching items than defined by the required quantity receive the discount by a mix and match. In case that enough transaction matching items and triggers are available to fulfill the preconditions multiple times, the mix and match is applied multiple times accordingly.

Example: Combine Rule Matching Items with a Logical AND

If the price derivation rule includes several rule matching items (item "pasta sauce" and item "basil") with the mix and match combination code AND, the behavior is as follows: The promotion states, for example, that the pasta sauce gets 20% off and the basil gets 50% off if one package of noodles is bought. If the transaction includes the subsequent items:

- Line item "pasta sauce", "basil", and "noodles" with a quantity 1 for each:
  - Line item "pasta sauce" gets 20% and
  - line item "basil" gets 50%.
- Line item "pasta sauce" and "noodles" with a quantity 1 for each:
  - The PCE does not compute any discount.
- Line item "basil" and "noodles" with a quantity 1 for each:
  - The PCE does not compute any discount.

If the required quantity is set, for example, to 2 for the "pasta sauce" and the transaction contains the following line items, the subsequent discount is computed:

- Line item "pasta sauce", "basil", and "noodles" with a quantity 1 for each:
  - No discount is computed by the PCE since the "pasta sauce" is not in the transaction with at least quantity 2.

- Line item "pasta sauce" with quantity 3 and "basil" as well as "noodles" with a quantity 1 for each:
  - Line item "pasta sauce" gets 20% for the quantity 2.
  - One quantity of line item "pasta sauce" is not discounted and
  - line item "basil" gets 50%

7.7.3 Combine a Certain Threshold Quantity of Rule Matching Items with an Exclusive OR

The mix and match combination code is OR_QUANTITY. The benefit can be applied if one or several rule matching items are part of the processed transaction. The discount is computed according to the mix and match price modification. The PCE chooses only one rule matching item of the mix and match. Which rule matching item is chosen depends on the transaction and on the matching item identifier. If the transaction includes several transaction matching items, the PCE selects the transaction matching items that correspond to the rule matching item with the smallest matching item identifier. If several transaction matching items can still be selected afterwards, the choose item method *(PromotionConditionRuleSO.chooseItemMethod)* is considered for selecting a normalized item.

Note that all the applicable mix and match price modifications refer to the regular/discount sales unit price of an item.

There exist two differences compared to the mix and match combination code OR. First, the mix and match limit count is not considered. Second, a threshold quantity of rule matching items is checked.

This threshold quantity is defined by the required quantity. If the required quantity is equal to null, 1 is used instead. No less and no more transaction matching items receive the discount by a mix and match than what is defined by the required quantity. In case that enough transaction matching items and triggers are available to fulfill the preconditions multiple times, the mix and match is applied multiple times accordingly.

Example: Combine a Certain Threshold Quantity of Rule Matching Items with an Exclusive OR

If the price derivation rule includes several rule matching items (item "pasta sauce" and item "basil") with the mix and match combination code OR_QUANTITY, the behavior is as follows. The promotion states, for example, that the pasta sauce gets 20% off or the basil gets 50% off if the customer buys one package of noodles. If the transaction includes the subsequent items:

- Line item "pasta sauce", "basil", and "noodles" with a quantity 1 for each:
  - Either item "pasta sauce" gets 20% off if item "pasta sauce" has a smaller matching item identifier in the price derivation rule than item "basil"
  - or item "basil" gets 50% off if item "basil" has a smaller matching item identifier in the price derivation rule than item "pasta sauce".
  - Only one of the transaction matching items gets the discount.
- Line item "pasta sauce" and "noodles" with a quantity 1 for each:
  - Item "pasta sauce" gets 20% off.
- Line item "basil" and "noodles" with a quantity 1 for each:
  - Item "basil" gets 50% off.

If the required quantity is set, for example, to 2 for the "pasta sauce" and the transaction contains the following line items, the subsequent discount is computed:

- Line item "pasta sauce", "basil", and "noodles" with a quantity 1 for each:
  - Item "basil" gets 50% off since the required quantity for the "pasta sauce" is not reached.

- Line item "pasta sauce" with quantity 3 and "basil" as well as "noodles" with a quantity 1 for each:
  - Either item "pasta sauce" gets 20% off for the quantity 2 and not for quantity 3 if item "pasta sauce" has a smaller matching item identifier in the price derivation rule than item "basil"
  - or item "basil" gets 50% off if item "basil" has a smaller matching item identifier in the price derivation rule than item "pasta sauce".
  - Only one of the transaction matching items gets the discount.

7.7.4 THERE IS MORE

The discount flag (*SaleReturnLineItem.discountFlag*) must be true for transaction matching items. This ensures that no discounts are granted to non-discountable line items (the discount flag is false), but otherwise non-discountable line items can cause a discount of other transaction matching items. This means that line items which are not discountable are possible triggers.

If the no previous monetary discount flag *(PromotionConditionRuleSO.noPreviousMonetaryDiscountAllowedFlag)* is set to true in the mix and match, only fully-priced items are considered as transaction matching items. The eligible line items should not be affected by the flag. A not fully-priced item still can trigger a discount in case the no previous monetary discount flag is set to true.

Additionally, the mix and match is only applied if one of the following eligibilities is part of a line item-related promotion price derivation rule:

- Item eligibility
- Merchandise category eligibility
- Product group eligibility

This also means that mix and match has to be a line item-related promotion price derivation rule.

For more information on how the line items corresponding to a rule matching item are chosen, see chapter Choose Items.

The handling of sales and returns without the original transaction affects the selection of line items for a mix and match, too. In case of sales only, the promotion price derivation rule is applied if the triggers and the transaction matching items are sold line items in the transaction. The same is true for returns only, both triggers and transaction matching items must be returns. If sales and returns are considered, the triggers indicate of which type the transaction matching item should be to apply a price derivation rule.

The discount methods (*PromotionConditionRuleSO.discountMethodCode*) are also considered during the calculation of the discount for the transaction matching items.

Please note that loyalty points are not allowed for matching items. Further information about loyalty points can be found in chapter Grant Loyalty Points.

Further information about the attributes can be found in chapter Configuration.

Note for a Better Performance
The mix and match makes it possible to define promotion price derivation rules that grant discounts for a different set of line items than the line items triggering it.
Assume that we have the following promotion: *Buy 2 packages of noodle and get 1 can of pasta sauce for free.*

In this case, the noodles are the eligible line items and the pasta sauce is the rule matching item. This promotion price derivation rule can be modeled in several possible ways:

	Price Derivation Rule Eligibility	Price Derivation Rule	Notes
Promotion Price Derivation Rule #1	**Item eligibility** (*ItemPromotionConditionEligibilitySO*) for item "noodles" (**Threshold type**(*ThresholdType*) QUTI and **quantity interval** (*ThresholdPromotionConditionEligibility.intervalQuantity*) 2)	Mix and match with 1 rule matching item 1. "pasta sauce" (Mix and match price modification is DISCOUNT_PERCENT with 100%)	The price derivation rule eligibility is only created for the eligible line items. The mix and match only contains information regarding the rule matching items that can receive the discount. If there are no transaction matching items but enough triggers, the calculation is still performed even though the promotion price derivation rule cannot be applied.
Promotion Price Derivation Rule #2	**Combination eligibility** (*CombinationPromotionConditionEligibilitySO*) with **combination code** (*CombinationPromotionConditionEligibilitySO.combinationCode*) AND and 2 child eligibilities 1. Item eligibility for item "noodles" (Threshold type QUTI and quantity interval 2), 2. Item eligibility for item "pasta sauce" (Threshold type QUTI and quantity interval 1)	Mix and match with 2 rule matching items 1. "noodles" (Mix and match price modification is DISCOUNT_PERCENT with 0%) 2. "pasta sauce" (Mix and match price modification is DISCOUNT_PERCENT with 100%)	The price derivation rule eligibility as well as the mix and match contain both items as eligible line items and as rule matching items. If there are no transaction matching items but enough triggers, the calculation is not performed at all, as the price derivation rule eligibility cannot be activated. In case of a transaction that activates the price derivation rule eligibility, the calculation is performed for both triggers and transaction matching items. Based on the price derivation rule, the triggers always receive a zero discount.

Promotion Price Derivation Rule #3	Combination eligibility with combination code AND and 2 child eligibilities 1. Item eligibility for "noodles" (Threshold type QUTI and quantity interval 2), 2. Item eligibility for "pasta sauce" (Threshold type QUTI and quantity interval 1)	Mix and match with 1 rule matching item 1. "pasta sauce" (Mix and match price modification is DISCOUNT_PERCENT with 100%)	The price derivation rule eligibility is created for both the eligible line items and the rule matching items. The mix and match only contains information regarding the rule matching items that will receive the discount. If there are no transaction matching items but enough triggers, the calculation is not performed at all, as the price derivation rule eligibility cannot be activated. In case of a transaction that activates the price derivation rule eligibility, the calculation is performed for the transaction matching items only.

All three possibilities result in the same applied discount, however, there is a difference in how the PCE processes them. To achieve the best performance, the promotion price derivation rule #3 is recommended to be used when defining a mix and match. If there are not enough transaction matching items to match the required quantity of the rule, this is recognized during the eligibility loading and the calculation is not performed by contrast with promotion price derivation rule #1. Furthermore, there are no unnecessary zero discount calculations involved by contrast with promotion price derivation rule #2, which also provides better performance.

7.8 Trigger an Action of the Client Application

An action shall be performed by the Client application which invokes the PCE in advance. The corresponding price derivation rule includes the information about the external action the Client application has to perform. The PCE takes over the external action data from the promotion master data to the transaction. This kind of price derivation rule is herein called external action price derivation rule (*ExternalActionPromotionConditionRuleSO*).

HOW IT WORKS

A promotion price derivation rule includes an external action price derivation rule. The promotion master data also includes an arbitrary number of external action texts (*ExternalActionTextSO*) and external action parameters (*ExternalActionParameterSO*) with an identifier and a value for the link in the external action price derivation rule. These external action texts and external action parameters are needed by the Client application for further processing of the PCE-response. Thus, the PCE does not calculate and apply a benefit to the transaction, but forwards the external action texts and external action parameters from the promotion master data to the Client application.

Additionally, it adds a discount amounting to 0.0 or 0.0 loyalty points to the transaction in the process step transaction update. Thereby, the price modification (*RetailTransactionPromotionPriceDerivationRule.priceModificationMethodCode*) is set to "EX" in the PCE-response. This price modification indicates that the Client application has to perform some kind of action in accordance to the added external action texts and external action parameters in the promotion master data.

Example: Trigger an Action of the Client

Let us assume there exists the following external action price derivation rule in the promotion master data:

1. Promotion: Buy for more than 100€ items and include the external action text with "Give-away Candy" and the external action parameter that includes the item identifier of the candy.

If a customer buys now 3 jeans for 69.95€, the PCE applies the promotion with the external action price derivation rule. The PCE-response includes then the external action text "Give-away Candy" and the item identifier of the candy. The issuing Client application or PPS Client can now use these data from the PCE-response. For example these can show a message box to the cashier. The cashier then gives the candy to the customer.

7.9 Handle Externally Applied Benefits

The PCE is capable to process benefits externally applied to a shopping basket. This means that the benefit applied to the shopping basket had been computed before the PCE was called and it will not be changed by the PCE. As a result, the PCE does not know what price derivation rule was executed and what triggered the externally applied benefit. Further internal promotion price derivation rules will be applied by the PCE if possible.

In contrast to a manual price derivation rule, the externally applied benefit is already computed and the result is already attached to the transaction before the PCE is called. The PCE does not calculate a benefit as it does if a manual price derivation rule is triggered by the transaction.

HOW IT WORKS

The handling of externally applied benefits is triggered by an external system by issuing a PCE-request that already includes applied benefits. The external originator flag *(externalSystemOriginatorFlag)* labels these benefits. The external originator flag determines whether a specific benefit was created by an external system (true) or not (false). If the flag is missing in the request, false is assumed. The PCE will neither remove nor change externally applied benefits and apply further promotion price derivation rules on top, when possible. As a result, a PCE-response is issued back to the calling external system containing the unchanged external benefit and possible internally calculated ones.

In order to apply an external benefit either a retail price modifier *(RetailPriceModifier)*, a price modification line item *(PriceModificationLineItem)*, a frequent shopper points modifier *(FrequentShopperPointsModifier)*, or a loyalty reward line item *(LoyaltyRewardLineItem)* needs to be part of the PCE-request. Furthermore in order for the PCE to recognize that the benefit was applied externally, a flag in the four different modifiers is set – that is, the external originator flag. The externally generated modifiers of the PCE-request are herein called external modifiers (external originator flag is true). Thereby, the schemas of the PCE-request are not violated. The following information is included:

- The PCE-request must include the following attributes and entities:
  - Actual unit price (*SaleReturnLineItem.actualUnitPrice*) of the line item(s) in the transaction
  - Quantity (*SaleReturnLineItem.quantity*) of the line item(s) in the transaction
  - Modifier (a retail price modifier, a price modification line item, a frequent shopper points modifier, or a loyalty reward line item)
- Modifier must include at least the following attributes to change the behavior of the PCE:
  - External originator flag *(externalSystemOriginatorFlag)* is true
  - Discount (*RetailPriceModifier.amount* or *PriceModificationLineItem.amount*)
- Modifier can include the following attributes in addition:
  - Sequence (*RetailTransactionPromotionPriceDerivationRule.promotionPriceDerivationRuleSequence*) the promotion price derivation rule sequence with which the external promotion was calculated
  - Applied quantity (*RetailPriceModifier.appliedQuantity* and *FrequentShopperPointsModifier.appliedQuantity*) then the PCE can check promotion price derivation rules that only apply to fully-priced items, too.
  - Trigger sequence number (*RetailPriceModifier.triggerSequenceNumber, FrequentShopperPointsModifier.triggerSequenceNumber, PriceModificationLineItem.triggerSequenceNumber or LoyaltyRewardLineItem.triggerSequenceNumber*) indicating the manual trigger's sequence number, which was used to trigger the external benefit

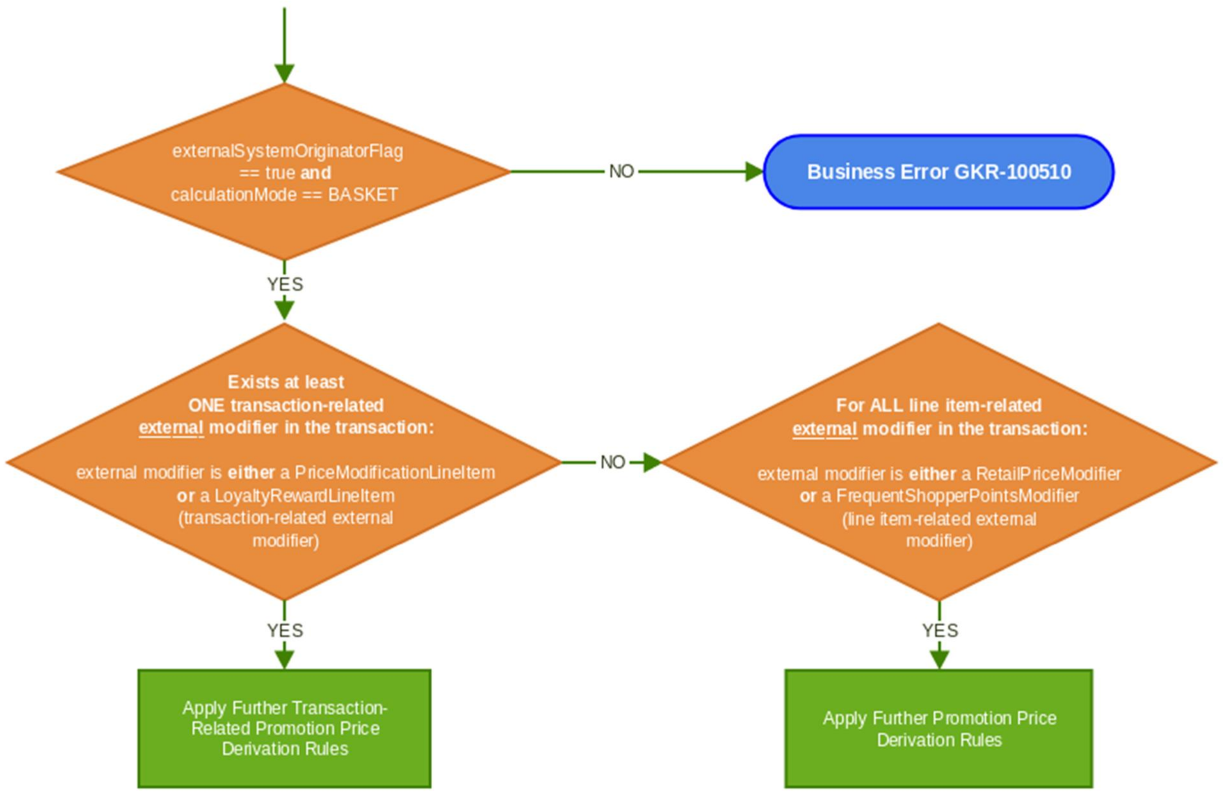
Promotion price derivation rules with a higher sequence (*PromotionConditionSO.sequence*) than the highest sequence of the externally applied benefits can be applied by the PCE in addition. The sequence is checked separately for promotion price derivation rules granting line item-related benefits and transaction-related ones.

Note, that if a transaction-related promotion price derivation rule is externally applied, all line items in the shopping basket may be effected. As a result, the line items in the shopping basket cannot be eligible for another promotion price derivation rule that applies to fully-priced items only. It is in addition assumed that transaction-related externally applied benefits are already prorated correctly to the line items in the corresponding transaction. Besides this, it is expected that a manual price override of a line items regular sales unit price is effective before the externally applied benefit is calculated. This means, that the externally applied benefit has the overridden price as calculation base amount.

In addition, the calculation mode (*calculationMode*) of the transaction is expected to be BASKET, otherwise a business error is thrown (GKR-100510 see SDK Promotion Calculation Engine > *Troubleshooting* > *Business Error Codes*).

If an external modifier is attached to a line item and one of the following flags is true, the PCE applies further promotion price derivation rules according to the definition of these flags. The external modifier and its impact on the transaction is not changed.

- Discount flag *(SaleReturnLineItem.discountFlag)*: A flag to indicate whether this line item can be discounted (false) or not (true).
- Eligible for loyalty points flag (*SaleReturnLineItem.frequentShopperPointsEligibilityFlag*): A flag to denote that the line item is eligible for loyalty points (false) or not (true).
- Not considered by the PCE flag (*SaleReturnLineItem.notConsideredByLoyaltyEngineFlag*): Determines whether the PCE should care about the line item as a trigger (false) or not (true).


externalSystemOriginatorFlag == true and calculationMode == BASKET
NO
Business Error GKR-100510
YES
Exists at least ONE transaction-related external modifier in the transaction:
external modifier is either a PriceModificationLineItem or a LoyaltyRewardLineItem (transaction-related external modifier)
NO
For ALL line item-related external modifier in the transaction:
external modifier is either a RetailPriceModifier or a FrequentShopperPointsModifier (line item-related external modifier)
YES
Apply Further Transaction-Related Promotion Price Derivation Rules
YES
Apply Further Promotion Price Derivation Rules

7.9.1 Apply Further Transaction-Related Promotion Price Derivation Rules

It is possible that the PCE applies further transaction-related promotion price derivation rules and no line item-related promotion price derivation rules. In order to reach this, the PCE-request must include at least one transaction-related external modifier and/or line item-related external modifiers. The PCE applies then further transaction-related promotion price derivation rules with a higher sequence. Therefore it is expected, that transaction-related external modifiers include the sequence.

If at least one of these external modifiers applies a non-zero discount (discount is greater than zero), then the PCE applies then further transaction-related promotion price derivation rules with a higher sequence but further promotion price derivation rules that are only applicable to fully-priced items can not be applied.

If the PCE-request includes at least one transaction-related external modifier and the sequence of one transaction-related external modifier is null, further promotion price derivation rules must not be applied.

In case a transaction level *manual trigger* exists in the transaction with the same *trigger sequence number*, as the external modifier's *trigger sequence number*, then the PCE considers this trigger already used for triggering the externally applied benefit. As a result the *manual trigger* will not be used to trigger any further internal promotion price derivation rules.

Example: Handle Externally Applied Benefits

Assume that the following promotion price derivation rules exist in the promotion master data:

1. Promotion: Granting 10% discount on the entire transaction if the customer has a coupon. (Sequence 3)
2. Promotion: Granting 2% discount on the entire transaction if the regular total amount is greater than 250€. (Sequence 5)

In addition, assume that a consumer buys 10 kg pork (1 kg pork costs 9.95€) and 15 kg beef (1 kg beef costs 12.99€) at the meat counter. The scale applies an external benefit with the sequence 4 to the transaction and grants 100 loyalty points. Thus, the scale receipt with a barcode looks like:

```
10 kg pork          99.50€
15 kg beef         194.85€
--------------------------------
- ext. benefit:    100 loyalty points
Total              294.35€
```

The consumer takes the scale receipt. At the Client application, the consumer shows the scale receipt and the coupon. The Client application retrieves the corresponding transaction by scanning the barcode of it.

The PCE is receiving the transaction with the already applied 100 loyalty points by the scale. The corresponding loyalty reward line item has set the external originator flag to true.

The PCE does not apply the 1. promotion that grants 10% discount since the loyalty reward line item includes the sequence of 4. The 2. promotion is applied by the PCE since the promotion price derivation rule has a higher sequence than the externally applied benefit and the regular total amount of the shopping basket is greater than 250€.

The receipt looks afterwards like:

```
10 kg pork          99.50€
15 kg beef         194.85€
--------------------------------
- ext. benefit:    100 loyalty points
Subtotal           294.35€
- discount 2:        5.89€
Total:             288.46€
```

The PCE prorates the transaction-related discount. The consumer has to pay 288.46€ for the purchase.

7.9.2 Apply Further Promotion Price Derivation Rules

It is possible that the PCE applies further promotion price derivation rules regardless of the level these are applied to – that is, line item and/or transaction level. For this to be possible it is expected that the PCE-request only includes line item-related external modifiers. All external modifiers are expected to include a sequence and an applied quantity.

The sequence is necessary to decide which further promotion price derivation rules can be applied. The PCE applies any possible promotion price derivation rule if the eligible normalized item does not have attached an external benefit (the transaction matching item in case of a mix and match). The sequence of the common promotion price derivation rule (promotion price derivation rule part of the promotion master data) does not play a role in this case. If a normalized item that has an external benefit, then only promotion price derivation rules with a higher sequence may be applied to this normalized item, thus the external benefit's sequence is considered.

The PCE needs the applied quantity of the external benefit in order to decide whether a promotion applicable to fully-priced items can be applied to the eligible normalized items or not. If the promotion has the no monetary discount before flag *(PromotionConditionRuleSO.noPreviousMonetaryDiscountAllowedFlag)* set to true if it is only applicable to fully-priced items. If the applied quantity is missing in one external modifier and a discount is applied that is greater than zero, further promotion price derivation rules are applied to the transaction but no ones that are applicable to fully-priced items only.

The externally applied benefits are independent of the allow zero discounts flag *(allowZeroRebate)*. An externally applied benefit can be a zero discount even though the allow zero discounts flag is false. In this case, the PCE does not apply any further retail price modifiers or price modification line items with a zero discount. The external modifier is still part of the PCE-response.

A manual trigger in the transaction is not handled differently - in regards to the sequence - than other promotion price derivation rules. This means, for example, the following:

- If the line item-related external modifier has no sequence: the manual trigger is not processed if it is attached on the same line item with the externally applied benefit, otherwise the corresponding price derivation rule can be executed.
- If the transaction-related external modifier has no sequence: the manual trigger is not processed.
- etc.

In case an item level *manual trigger* exists on an item with the same *trigger sequence number*, as the external modifier's *trigger sequence number* on the same item, then the PCE considers this trigger already used for triggering the externally applied benefit. As a result the *manual trigger* will not be used to trigger any further internal promotion price derivation rules.

If there exists an external modifier that does not include a sequence and it applies a zero discount, further promotion price derivation rules can be applied regardless of the first applicable promotion price derivation rule's sequence. This means, that the PCE calculates benefits as if there is no external modifier part of the corresponding line item.

If one line item-related external modifier does not include a sequence and one applies a non-zero discount, the PCE applies further transaction-related promotion price derivation rules. It is not applying any further promotion price derivation rules that are only applicable to fully-priced items.

Example: Apply Further Promotion Price Derivation Ruels

Assume that the following promotion price derivation rules exist in the promotion master data:

1. Promotion: Buy pork and get 3% on beef. (Sequence 3)
2. Promotion: Buy beef and get 3% on pork. (Sequence 2)

In addition, assume that a consumer buys 10 kg pork (1 kg pork costs 9.95€) and 15 kg beef (1 kg beef costs 12.99€) at the meat counter. The scale applies an external benefit with the sequence 4 and the applied quantity of 10 to the beef and grants for each kilogram beef 2.00€ discount up to 10 kg. Thus, the scale receipt with a barcode looks like:

```
10kg pork          99.50€
15 kg beef        194.85€
- ext. discount:   20.00€
--------------------------------
Total             274.35€
```

The consumer takes the scale receipt. At the Client application, the consumer shows the scale receipt and the Client application retrieves the transaction by scanning the barcode of it.

The PCE receives the transaction with the already applied 20.00€ discount by the scale. The corresponding retail price modifier has set the external originator flag to true.

The PCE does not apply the first promotion since the benefit should be applied on the beef (the rule matching item) and the sequence of the first promotion is less than the sequence of the externally applied benefit. The second promotion is applied by the PCE since the rule matching item – the pork – has no external modifier.

The receipt looks afterwards like:

```
10 kg pork         99.50€
- discount 2:       3.00€
15 kg beef        194.85€
 - ext. discount:  20.00€
--------------------------------
Total             271.35€
```

The consumer has to pay 271.35€ for the purchase.

7.9.3 THERE IS MORE

It cannot be guaranteed that the same promotion identifier (*RetailTransactionPromotionPriceDerivationRule.Key.promotionID*) and the same sequence in the external modifier indicate that the externally applied benefit is the result of an "internally" corresponding common promotion with the equivalent promotion identifier and sequence. This is the reason why the PCE only applies further promotion price derivation rules that have a greater sequence than the externally applied benefit.

Additionally, there is no special handling of external modifiers in combination with the calculation base sequence (*PromotionConditionRuleSO.calculationBaseSequence*).

8 Transaction Update – Finalize Promotion Calculation

The computed discounts and loyalty points of the previously executed steps are required to update the currently processed transaction. If a benefit exists and several line items are affected, it is prorated to all related items in the transaction. Modifiers are prepared and added to the affected line items. Finally, the transaction is mapped into the PCE-response. The PCE-response contains all information about the modification. For example, it contains the identifier for the applied promotion price derivation rules and the calculated benefit. There are four modifiers: the retail price modifier (*RetailPriceModifier*) or the price modification line item (*PriceModificationLineItem*) that include a monetary discount, and the frequent shopper points modifier (*FrequentShopperPointsModifier*) or the loyalty reward line item (*LoyaltyRewardLinteItem*) that include loyalty points.

The bonus points flag (*PromotionConditionRuleSO.bonusPointsFlag)* and the transaction control break code (*PromotionConditionRuleSO.transactionControlBreakCode)* decide which modifier is attached to the transaction. If the bonus points flag is true, either the frequent shopper points modifier or the loyalty reward line item is added to the transaction. Thereby, the transaction control break code is either "PO" or "PC", so that a frequent shopper points modifier is added to the corresponding eligible line item. If the transaction control break code is either "SP" or "SU", the loyalty reward line item is attached to the transaction as separate line item. If the bonus points flag is false, either the retail price modifier or the price modification line item is added. Thereby, the transaction control break code is either "PO" or "PC", so that a retail price modifier is added to the corresponding eligible line item. If the transaction control break code is either "SP" or "SU", the price modification line item is attached to the transaction as separate line item.

In this chapter, the following topics are discussed:

- Round the Benefit
- Select the Method to Discount
- Prorate the Benefit

8.1 Round the Benefit

There are two ways in which the PCE rounds the calculated benefits:

- The PCE rounds the calculated benefits according to rounding rules (*RoundingRuleDO*). These rounding rules are linked via the round rules (*PromotionConditionRuleSO.roundingRuleID*) set in price derivation rules in the promotion master data.
- If there is an invalid round rule or no round rule set in the price derivation rule, the rounding is processed according to the settings in the price derivation rule or according to default values, that are introduced in the following explanations.

Rounding rules consist of a value and a direction. The value is the *multiple* (*RoundingRuleDO.multiple*), whereas the direction is encoded in the round method (*RoundingRuleDO.roundMethodCode*). Rounding always is done on the benefit and not on the effective sales unit price.

The round method can have the following values:

- NO_ROUNDING: no rounding
- HALF_UP: round half up, is referred to in the entire document as COMMERCIAL (round up for greater than or equal to the half of the multiple, else round down)
- DOWN: round down
- UP: round up
- HALF_DOWN: round half down (round up for greater than the half of the multiple, else round down)

The *multiple* value describes the rounding target. It defines the number decimal places of the rounded discount and whether the last number is a multiple of 0, 1 or 5. The *multiple* value can be composed of a number that consists of several zeros and can contain a 1 or a 5 at the end. However, the *multiple* value can contain either a 1 or a 5, but not both (e.g. 0.01, 0.5).

The price derivation rule can contain three different kinds of rounding rules which result in the following possibilities:

1. Round benefits
2. Round benefit shares
3. Round the monetary equivalent of loyalty points

Each of these possibilities can be defined by a separate rounding rule.

8.1.1 Round Benefits

The round rule (*PromotionConditionRuleSO.roundingRuleID*) is set in the price derivation rule. If it is valid, it applies for the computed benefits – that is, discounts and loyalty points – according to the price derivation rule. Otherwise, the attributes are set according to the following possibilities:

1. Round discounts
2. Round loyalty points

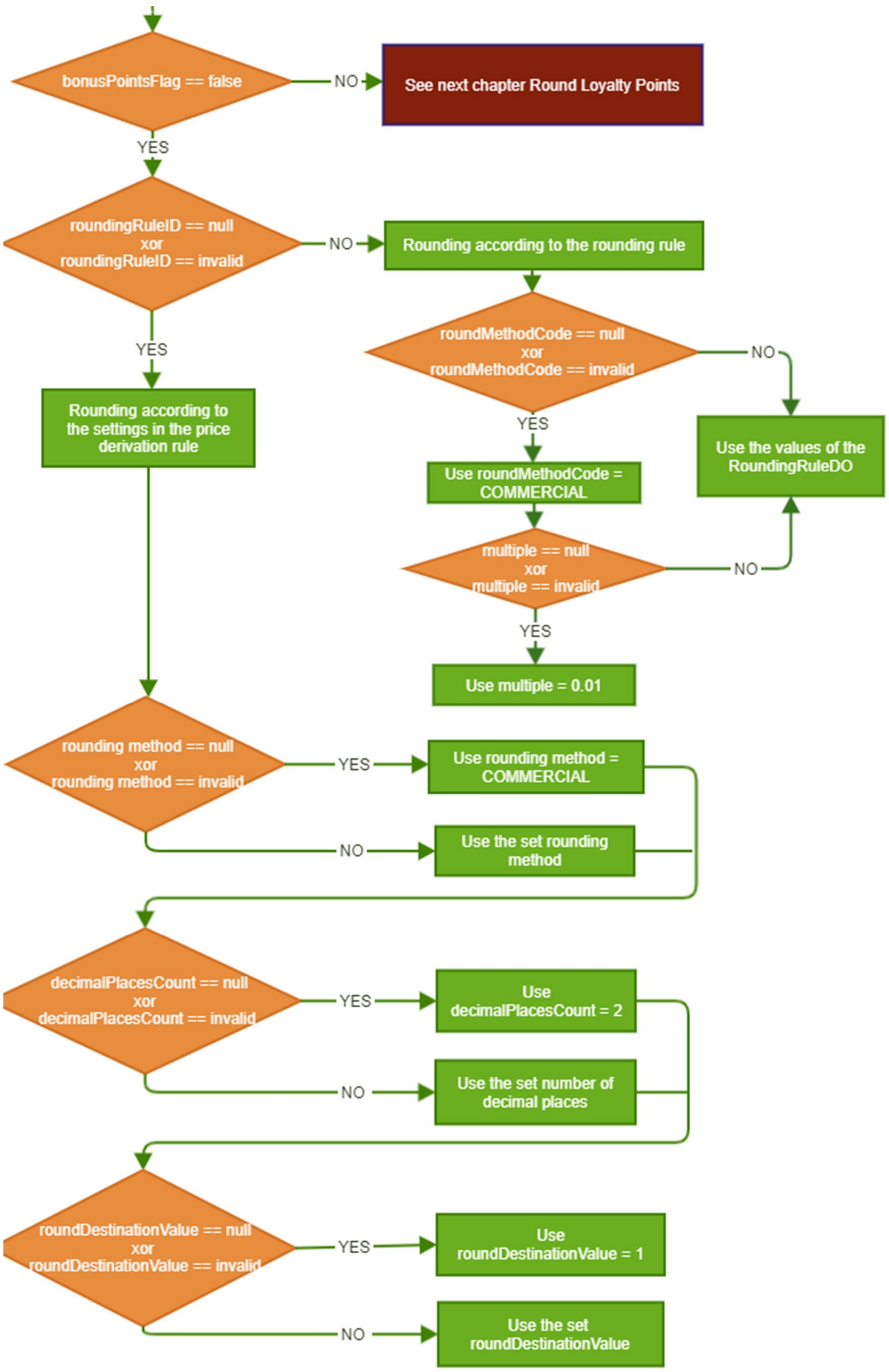
8.1.1.1 Round Discounts

If the round rule is null or does not reference a valid rounding rule, the bonus points flag (*PromotionConditionRuleSO.bonusPointsFlag*) decides how the PCE proceeds. If the bonus points flag is false, the discounts are rounded according to the following possibilities in the price derivation rule that are:

- Rounding method (*PromotionConditionRuleSO.roundingMethodCode*) with the following possible values:
  - o COMMERCIAL_ROUNDING/"00": commercial rounding, round half up, is referred to in the entire document as a COMMERCIAL
  - o DOWN/"01": round down
  - o UP/"02": round up
- Decimal places count (*PromotionConditionRuleSO.decimalPlacesCount*) can be:
  - o null: that is mapped to "2"
  - o a value that represents the decimal places the benefit can maximally have
- Round destination value *(PromotionConditionRuleSO.roundDestinationValue)* can have the following values:
  - o "1"/null: no additional treatment of the rounded result necessary
  - o "5": the rounded result is rounded again to the next respectively to the previous multiple of 5

If the bonus points flag is false and the rounding method is null or is invalid, it is set to COMMERCIAL_ROUNDING. If the decimal places count is null, the default value of 2 is used. If the round destination value is null, it is set to 1.

If the round rule is valid, the bonus points flag is false and the round method is null or invalid, the value COMMERCIAL is used instead. If the multiple is null or invalid, the default value of 0.01 is used.

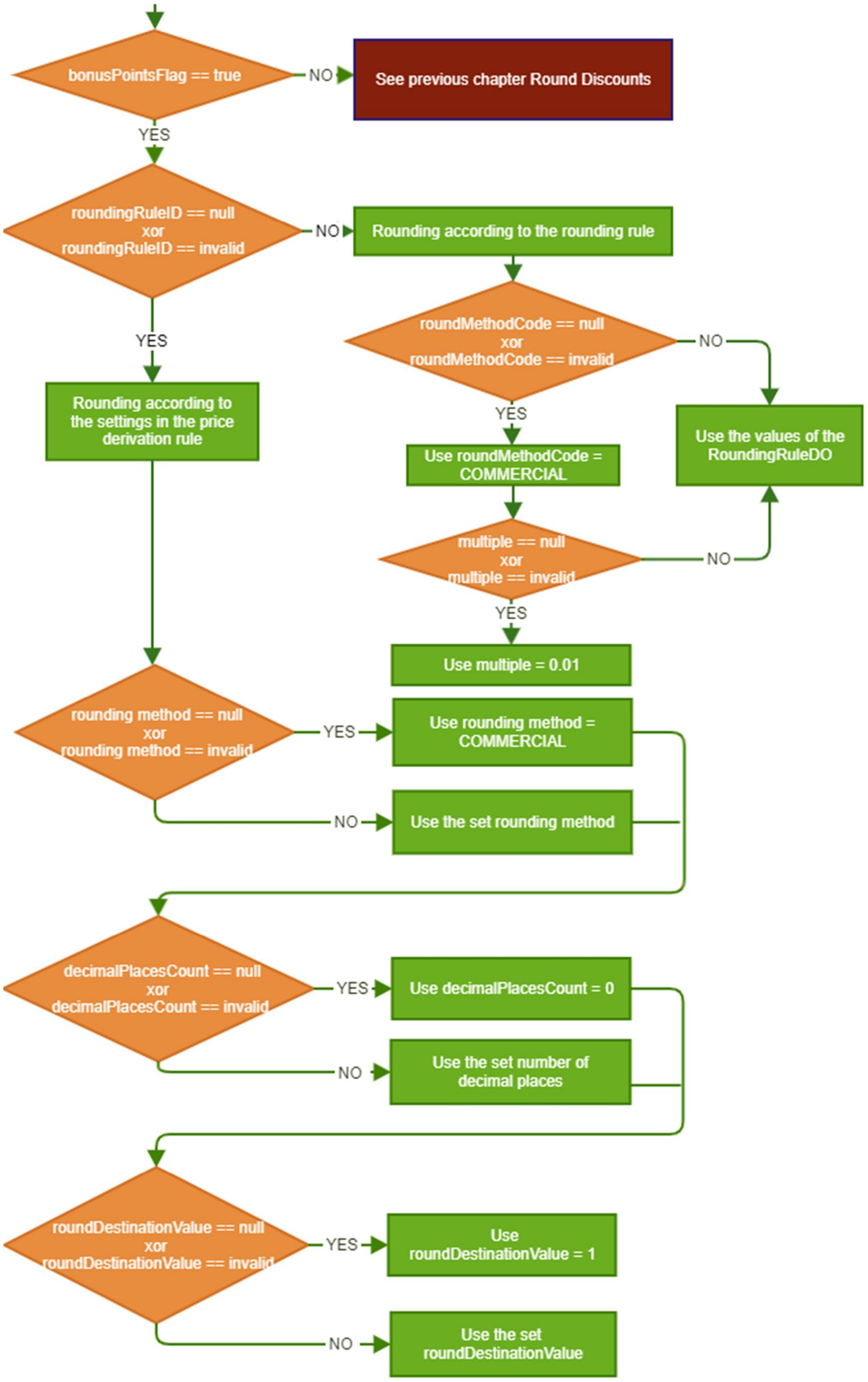

bonusPointsFlag == false
NO
See next chapter Round Loyalty Points
YES
roundingRuleID == null
xor
roundingRuleID == invalid
NO
Rounding according to the rounding rule
roundMethodCode == null
xor
roundMethodCode == invalid
NO
Use the values of the RoundingRuleDO
YES
Use roundMethodCode = COMMERCIAL
multiple == null
xor
multiple == invalid
NO
YES
Use multiple = 0.01
YES
Rounding according to the settings in the price derivation rule
rounding method == null
xor
rounding method == invalid
YES
Use rounding method = COMMERCIAL
NO
Use the set rounding method
decimalPlacesCount == null
xor
decimalPlacesCount == invalid
YES
Use decimalPlacesCount = 2
NO
Use the set number of decimal places
roundDestinationValue == null
xor
roundDestinationValue == invalid
YES
Use roundDestinationValue = 1
NO
Use the set roundDestinationValue

8.1.1.2 Round Loyalty Points

If the round rule is null or does not reference a valid rounding rule, the flag bonus points flag decides how the PCE proceeds. If the bonus points flag is true, the loyalty points are rounded according to the attributes in the price derivation rule.

If the bonus points flag is true and the rounding method is null or is invalid, it is set to COMMERCIAL_ROUNDING. If the decimal places count is null, the default value of 0 is used. If the round destination value is null, it is set to 1.

If the round rule is valid, the bonus points flag is true and the round method is null or invalid, the value COMMERCIAL is used instead. If the multiple is null or invalid, the default value of 0.01 is used.


bonusPointsFlag == true
NO
See previous chapter Round Discounts
YES
roundingRuleID == null xor roundingRuleID == invalid
NO
Rounding according to the rounding rule
YES
Rounding according to the settings in the price derivation rule
roundMethodCode == null xor roundMethodCode == invalid
NO
YES
Use roundMethodCode = COMMERCIAL
Use the values of the RoundingRuleDO
multiple == null xor multiple == invalid
NO
YES
Use multiple = 0.01
rounding method == null xor rounding method == invalid
YES
Use rounding method = COMMERCIAL
NO
Use the set rounding method
decimalPlacesCount == null xor decimalPlacesCount == invalid
YES
Use decimalPlacesCount = 0
NO
Use the set number of decimal places
roundDestinationValue == null xor roundDestinationValue == invalid
YES
Use roundDestinationValue = 1
NO
Use the set roundDestinationValue

8.1.2 Round Benefit Shares

The share rounding rule (*PromotionConditionRuleSO.shareRoundingRuleID*) is set in the price derivation rule. If it is valid, it is used for the benefit shares that are computed during the proration of a benefit that results from a transaction-related promotion price derivation rule. Otherwise, the attributes are set according to the following possibilities:

1. Round discount shares
2. Round loyalty point shares

8.1.2.1 Round Discount Shares

If the share rounding rule is null or does not reference a valid rounding rule, the bonus points flag decides how the PCE proceeds. If the bonus points flag is false, the discounts are rounded according to the corresponding system parameters.

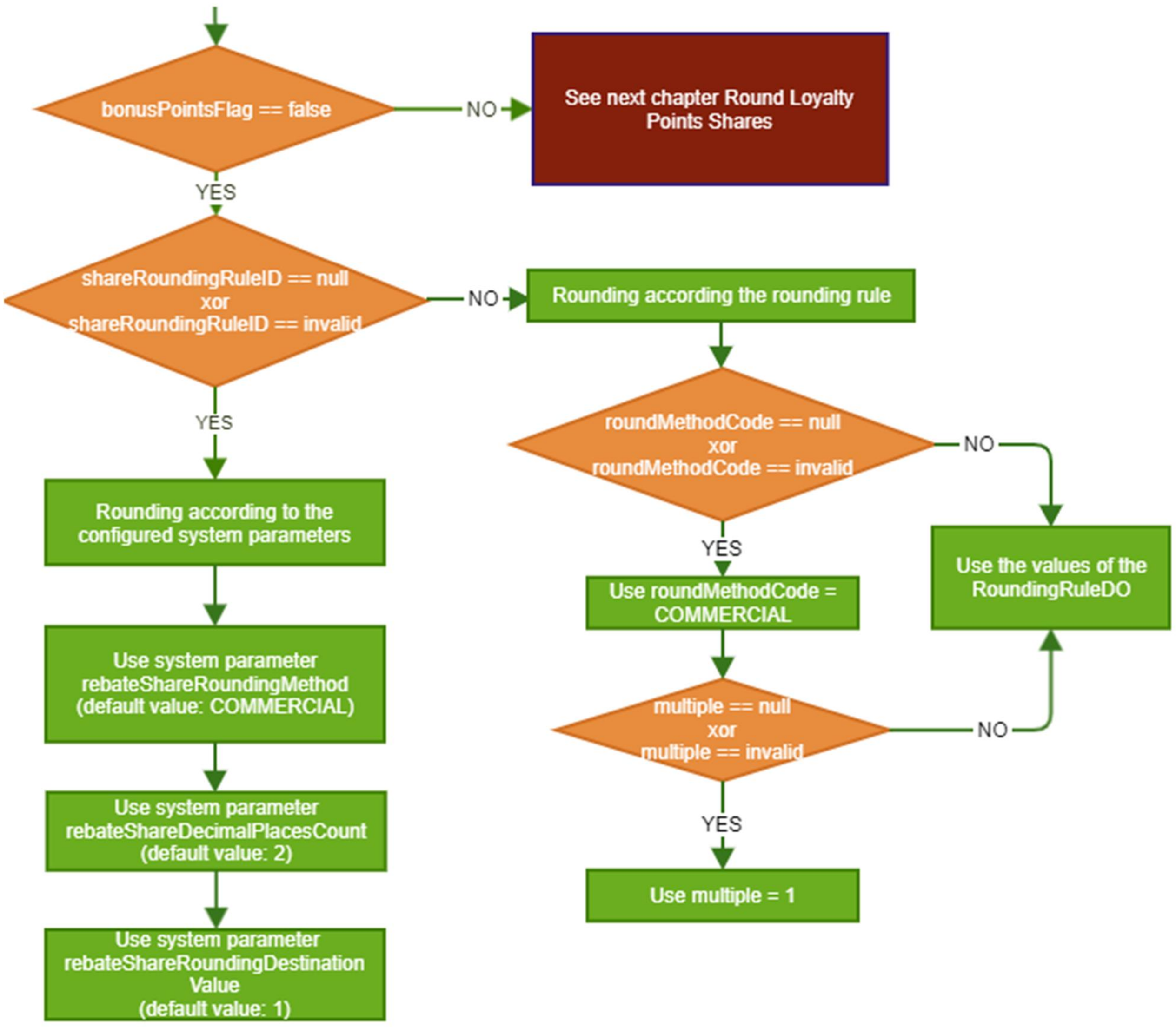
The system parameter *rebateShareRoundingMethod* is used. This parameter can have the following values:

- NO_ROUNDING: no rounding
- HALF_UP: round half up, is referred to in the entire document as a COMMERCIAL
- DOWN: round down
- UP: round up
- HALF_DOWN: round half down

If the bonus points flag is false, the corresponding system parameter *rebateShareDecimalPlacesCount* is used. This parameter can be a natural number between zero and four. The default value is thereby two.

If the bonus points flag is false, the corresponding system parameter *rebateShareRoundingDestinationValue* is used. This parameter can be either 1 or 5. The default value is thereby 1.

If the share rounding rule is valid, the bonus points flag is false and the round method is null or invalid, the value COMMERCIAL is used. If the multiple is null or invalid, the default value of 1 is used.


bonusPointsFlag == false
NO
See next chapter Round Loyalty Points Shares
YES
shareRoundingRuleID == null xor shareRoundingRuleID == invalid
NO
Rounding according the rounding rule
YES
Rounding according to the configured system parameters
roundMethodCode == null xor roundMethodCode == invalid
NO
YES
Use roundMethodCode = COMMERCIAL
Use the values of the RoundingRuleDO
Use system parameter rebateShareRoundingMethod (default value: COMMERCIAL)
multiple == null xor multiple == invalid
NO
YES
Use system parameter rebateShareDecimalPlacesCount (default value: 2)
Use multiple = 1
Use system parameter rebateShareRoundingDestination Value (default value: 1)

8.1.2.2 Round Loyalty Point Shares

If the share rounding rule is null or does not reference a valid rounding rule, the bonus points flag decides how the PCE proceeds. If the bonus points flag is true, the loyalty points are rounded according to the corresponding system parameters.

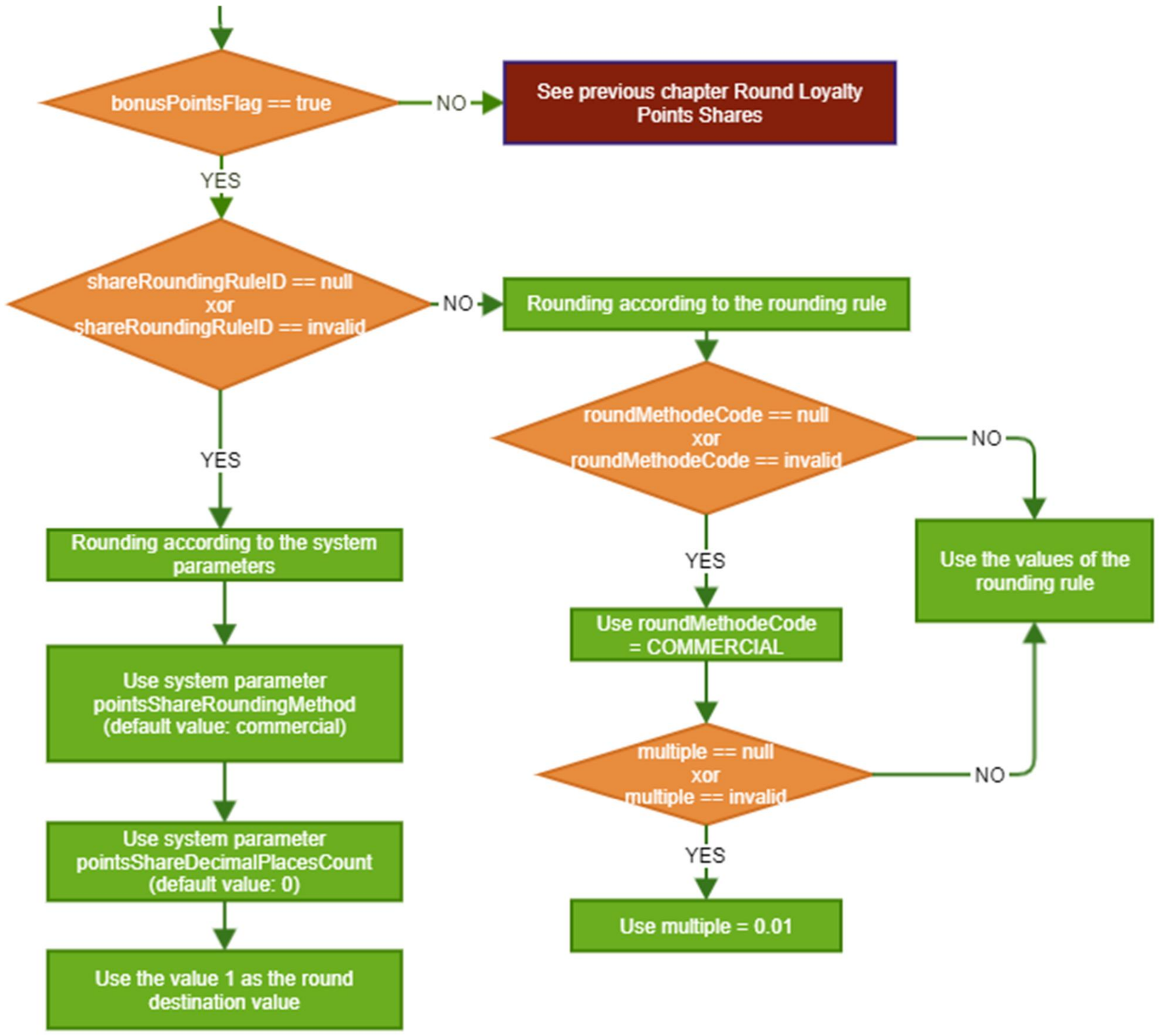
If the bonus points flag is true the system parameter *pointsShareRoundingMethod* is used. This parameter can have the following values:

- "Down": truncation of the result
- "Up": rounding up
- "Commercial": round half up (this is the default value), is referred to in the entire document as a COMMERCIAL

If the bonus points flag is true the corresponding system parameter *pointsShareDecimalPlacesCount* is used. This parameter can be a natural number between zero and four. The default value is thereby zero.

If the bonus points flag is true the round destination value 1 is used.

If the share rounding rule is valid, the bonus points flag is true and the round method is null or invalid, the value COMMERCIAL is used. For the multiple the default value of 0.01 is used.


bonusPointsFlag == true
NO
See previous chapter Round Loyalty Points Shares
YES
shareRoundingRuleID == null xor shareRoundingRuleID == invalid
NO
Rounding according to the rounding rule
YES
roundMethodeCode == null xor roundMethodeCode == invalid
NO
Rounding according to the system parameters
YES
Use the values of the rounding rule
Use roundMethodeCode = COMMERCIAL
Use system parameter pointsShareRoundingMethod (default value: commercial)
multiple == null xor multiple == invalid
NO
YES
Use system parameter pointsShareDecimalPlacesCount (default value: 0)
Use multiple = 0.01
Use the value 1 as the round destination value

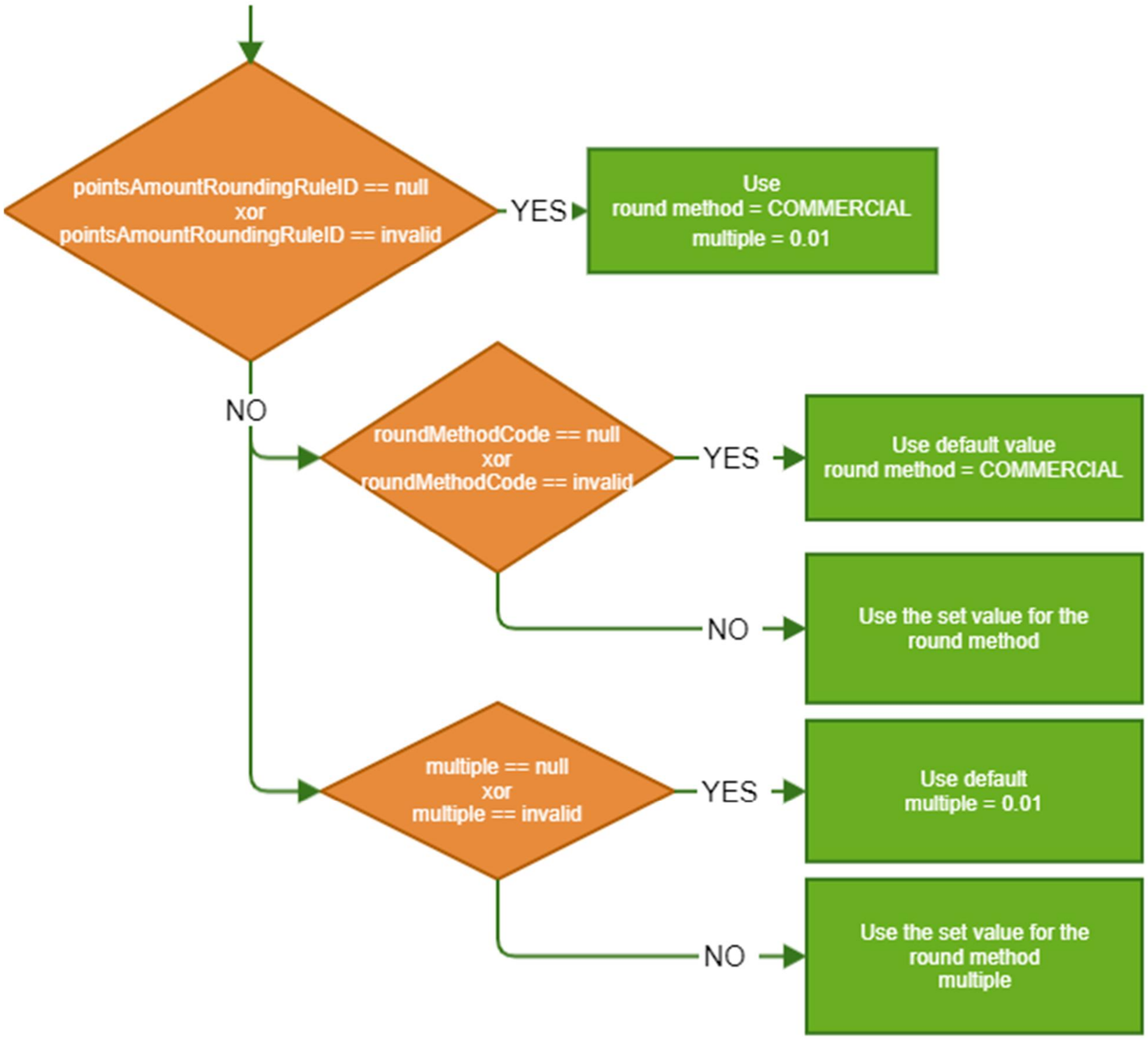
8.1.3 Round the Monetary Equivalent of Loyalty Points

The points amount rounding rule (*PromotionConditionRuleSO.pointsAmountRoundingRuleID*) is set in the price derivation rule. It is used to round the monetary equivalent of loyalty points. Thus, this type of rounding rule is not applicable to price derivation rules that apply discounts.

If the points amount rounding rule is null or does not reference a valid rounding rule, the monetary equivalent is rounded according to the following values:

- Round method is COMMERCIAL
- Multiple is 0.01

If only the round method is invalid or null, the default value is used instead. The default value is COMMERCIAL. On the other hand, if only the multiple of the rounding rule is invalid or null, the attribute is set to 0.01.



pointsAmountRoundingRuleID == null
xor
pointsAmountRoundingRuleID == invalid
YES
Use
round method = COMMERCIAL
multiple = 0.01
NO
roundMethodCode == null
xor
roundMethodCode == invalid
YES
Use default value
round method = COMMERCIAL
NO
Use the set value for the
round method
multiple == null
xor
multiple == invalid
YES
Use default
multiple = 0.01
NO
Use the set value for the
round method
multiple

8.1.4 THERE IS MORE

Due to the normalized calculation, the discount amounts are rounded on the normalized item level as well. This can lead to different values for rounding than it may be expected for a line item.

8.2 Select the Method to Discount

It is possible to decide how to credit a discount. The PCE supports the following methods:

- A discount that reduces the price
- The creation of a gift certificate

- A discount as tender
- The creation of a coupon

These different ways to credit a discount – herein called discount methods (*PromotionConditionRuleSO.discountMethodCode*) – indirectly influence the money exchange between a customer and a retailer. In addition, a discount method can create an incentive for a customer to come back to redeem the previously received coupon or gift certificate.

HOW IT WORKS

The discount method of the price derivation rule controls the method of applying a discount for any type of price derivation rule. It indicates how the discount is applied to the transaction. The discount method is provided with the promotion master data. The following values are possible:

- "00": The PCE reduces the total amount of the transaction (see Use the Discount to Reduce the Price).
- "01": The PCE does not change the total amount of the transaction and prepares the creation of a gift certificate with the value of the discount (see Create a Gift Certificate).
- "02": The discount is used as tender for the current transaction, i.e. it reduces the amount the customer still has to pay (see Use the Discount as Tender).
- "03": The PCE reduces the total amount of the transaction by a gift certificate amount and provides the information about this gift certificate (see Reduce the Total Amount as Gift Certificate).
- "04": The PCE does not change the total amount of the transaction and prepares the creation of a coupon (see Create a Coupon).

If the discount method is not set in the promotion master data, the PCE reduces the total price of the transaction. This behavior is the same as if the discount method had the value "00".

8.2.1 Use the Discount to Reduce the Price

The discount computed by the PCE reduces the price of the line item or the transaction according to the applicable price derivation rules. This type of discount is called monetary discount.

8.2.2 Create a Gift Certificate

The PCE prepares the transaction in a way that a gift certificate is created for the customer. The price derivation rule indicates the value of the gift certificate and the time period in which it is valid. This means that the computed discount is the value of the gift certificate. This discount does not reduce the price of the corresponding line items or the transaction and is called virtual discount.

Example

Let us assume that there is the following promotion price derivation rule in the promotion master data: For every purchased cocktail dress get a gift certificate of value 50.00€. We also assume that the discount method is "01". The sale return type (*PromotionConditionSO.saleReturnTypeCode*) is SALES_RETURNS, thus, the promotion price derivation rule may apply to both returned and sold line items.
The customer bought a cocktail dress in blue color for the regular sales unit price 424.24€. Now she would like to return the dress and buy two of the same dress in different colors for a sum amount of 848.48€.
As a result, the customer receives no monetary discount and has to pay 424.24€. The customer also receives a 50.00€ value gift certificate.

Sales	Return	Quantity	Regular Sales Price	Monetary Discount	Virtual Discount	Effective Sales Price	
	Cocktail dress	1	424.24€	0.00€	50.00€	424.24€	
Cocktail dress		2	848.48€	0.00€	100.00€	848.48€	
Regular Total Amount							424.24€
Gift Certificate							50.00€

8.2.3 Use the Discount as Tender

If the discount method is "02", it is a discount granted as a tender. The price derivation rule defines how this discount is computed. The computed discount does not reduce the price of the corresponding line item in the transaction. Thus, it is a virtual discount. The Client recognizes this discount method in the transaction and utilizes the virtual discount as tender for the purchase, after the PCE has attached the calculated virtual discount to the transaction. The PCE does not check whether the corresponding tender exists in the master data. If it does not exist, the promotion price derivation rule is applicable, but the Client cannot create the corresponding tender line item.

The PCE does not take into account tenders for foreign currencies that would lead to a conversion of the exchange rate. The PCE considers all entries as entered in home currency.

Example
Let us assume that there is the following promotion price derivation rule in the promotion master data: For every purchased cocktail dress get a 50.00€ tender. We also assume that the discount method is "02" for a given line item related promotion. The SaleReturnTypeCode is "00" (sales and returns), thus the promotion may apply to both returned and sold items.

The customer bought a cocktail dress in blue color for the regular sales unit price 424.24€. Now she would like to return the dress and buy two of the same dress in different colors for a sum amount of 848.48€.
As a result, the customer gets no monetary discount and has to pay 424.24€. The customer also receives a 50.00€ value tender, which may reduce the sum of the payed amount, if valid.

Sales	Return	Quantity	Regular Sales Price	Monetary Discount	Virtual Discount	Effective Sales Price	
	Cocktail dress	1	424.24€	0.00€	50.00€	424.24€	
Cocktail dress		2	848.48€	0.00€	100.00€	848.48€	
Regular Total Amount							424.24€
Tender							50.00€

8.2.4 Reduce the Total Amount as Gift Certificate

If the discount method is "03", the customer pays the full amount and receives a gift certificate which can be redeemed at a later time. The price derivation rule indicates which value the gift certificate has and in which time period the certificate is valid.

In contrast to discountMethodCode "01", the amount of the gift certificate is deducted directly and added again as a special position type which is not subject to VAT. Thus, VAT is charged for the reduced amount, analogous to a monetary discount. The virtual discount is set to zero.

Example
Let us assume that there is the following promotion price derivation rule in the promotion master data: For every purchased cocktail dress get a 50.00€ discount as gift certificate with the expiration date 2020-01-01 14:14:26. We also assume that the discount method is "03". The sale return type is SALES_RETURNS, thus, the promotion price derivation rule may apply to both returned and sold line items.
The customer bought a cocktail dress in blue color for the regular sales unit price 424.24€. Now she would like to return the dress and buy two of the same dress in different colors for a sum amount of 848.48€.
As a result, the customer has to pay 374.24€. Expiration date 2020-01-01 14:14:26 of the gift certificate is provided by PCE.

Sales	Return	Quantity	Regular Sales Price	Monetary Discount	Virtual Discount	Effective Sales Price	
	Cocktail dress	1	424.24€	50.00€	0.00€	374.24€	
Cocktail dress		2	848.48€	100.00€	0.00€	748.48€	
Regular Total Amount							374.24€

8.2.5 Create a Coupon

The PCE prepares the transaction in a way that a coupon is created for the customer. The price derivation rule indicates that a coupon should be printed. The computed discount does not reduce the price of the corresponding line items or the transaction, respectively. Thus, this is a virtual discount. In addition, there is another promotion price derivation rule in the promotion master data that has the printed coupon as a trigger. The latter promotion price derivation rule defines the effect of the coupon.

Example

Let us assume that there is the following promotion price derivation rule in the promotion master data: For every purchased cocktail dress get the coupon "Absolute Reduction of 50€" with coupon ID 123456. We also assume that the discount method is "04". The sale return type is SALES_RETURNS, thus, the promotion price derivation rule may apply to both returned and sold line items. Beside this, there is another promotion price derivation rule in the promotion master data, which grants 50€ discount, when coupon with coupon ID 123456 is scanned.

The customer bought a cocktail dress in blue color for the regular sales unit price 424.24€. Now she would like to return the dress and buy two of the same dress in different colors for a sum amount of 848.48€.

As a result, the customer gets no monetary discount and has to pay 424.24€. The customer also receives 3 coupons with coupon ID 123456, each of this coupons can grant discount of 50.00€ for a next purchase.

Sales	Return	Quantity	Regular Sales Price	Monetary Discount	Virtual Discount	Effective Sales Price	
	Cocktail dress	1	424.24€	0.00€	50.00€	424.24€	
Cocktail dress		2	848.48€	0.00€	100.00€	848.48€	

Regular Total Amount		424.24€
Coupon		3 x 50€ (2 for the sold cocktail dresses, 1 for the returned cocktail dress)

8.2.6 THERE IS MORE

The discount method is also considered in case of returning items without the original transaction.

The discount method is part of the price derivation rule. Further information can be found in chapter Configuration.

8.3 Prorate the Benefit

The resulting benefits from the step promotion calculation are prorated if several line items are eligible.

The benefit type is crucial for the proportional distribution of these in case of transaction-related promotion price derivation rules. The different benefit types are:

- Loyalty points (bonus points flag (*PromotionConditionRuleSO.bonusPointsFlag*) is true)
- Monetary discounts (the bonus points flag is false and the discount method (*PromotionConditionRuleSO.discountMethodCode*) is either "00" or "03")
- Virtual discounts (the bonus points flag is false and the discount method is either "01", "02", or "04")

HOW IT WORKS

8.3.1 Prorate Line Item-Related Benefits

It is possible that a line item-related promotion price derivation rule affects multiple line items in a transaction. The corresponding benefit – discount or loyalty points – is prorated after the calculation is finished. In case of a threshold eligibility (*ThresholdPromotionConditionEligibility*) with an amount or quantity interval (*ThresholdPromotionConditionEligibility.intervalAmount* or *ThresholdPromotionConditionEligibility.intervalQuantity*) the proportional distribution is done after each applied interval.

Per definition of the price modification methods *(RebatePromotionConditionRuleSO.priceModificationMethodCode*) and the mix and match price modifications *(MatchingItemSO.reductionMethodCode)*, a proportional distribution of a line item-related benefit does not happen if these attributes have one of the following values:

- DISCOUNT_SINGLE/"RS": reduction of the sales unit price by an amount
- DISCOUNT_PERCENT/"RP": reduction of the sales unit price by a percentage
- FIXED_PRICE/"PS": setting the discount sales unit price

The reason is that the benefit is defined per sales unit price and not per the summed up sales prices. As a conclusion, a computation result of a Mix and Match (*MixAndMatchPromotionConditionRuleSO*) is not prorated to several line items since only the aforementioned price modification methods are supported in this case.

At first, the eligible items are collected and the calculation base amount is computed as well as the benefit. Then these items are sorted based on choose item method (for more information see Choose Items) and then for the proration of the benefit they are sorted in ascending order based on their calculation base. This means that previously applied discounts can be considered in the proportional distribution, depend on the calculation base sequence. The last expensive item is used first for the computation of the benefit share. Thereby, the calculation base amount of the discount and the calculation base amount of the item are used in order to compute the ratio of the benefit share to the benefit. The least item and thus, the most expensive one gets the residual of the benefit.

Example: Prorate Line Item-Related Benefits

Let us assume there are the following items which belong to the merchandise category "clothes":

- Pants: 40.50€
- Shirt: 25.00€

We also assume that there is the following promotion price derivation rule in the promotion master data:

1. Buy "clothes" and get 5% off on all bought "clothes". (price modification method DISCOUNT_PERCENT_TOTAL/"TP")

The customer buys a shirt and five pants. The calculation base amount of the eligible items is 227.50€, and the PCE calculates a total discount of 11.38€. This discount is subsequently prorated to the eligible items (at first for shirt, then for four pants, and the rest of the benefit is granted for the fifth pant) as follows:

Item	Regular Sales Price	Quantity	Discount	Calculation Base Amount of the Normalized Items	Calculation Base Amount	Discount on Normalized Items	Effective Sales Price
Shirt	25.00€	1		25.00€		25.00€ * 11.38€ / 227.50€ = 1.25€	23.75€

Pant	40.50€	1	11.38€	40.50€	227.50€	40.50€ * 11.38€ / 227.50€ = 2.03€	38.47€
Pant	40.50€	1		40.50€		40.50€ * 11.38€ / 227.50€ = 2.03€	38.47€
Pant	40.50€	1		40.50€		40.50€ * 11.38€ / 227.50€ = 2.03€	38.47€
Pant	40.50€	1		40.50€		40.50€ * 11.38€ / 227.50€ = 2.03€	38.47€
Pant	40.50€	1		40.50€		11.38€ - 1.25€ - 4 * 2.03€ = 2.01€	38.49€

8.3.2 Prorate Transaction-Related Benefits

The benefit of the transaction-related promotion price derivation rule is stated either in a price modification line item *(PriceModificationLineItem*) or in a loyalty reward line item *(LoyaltyRewardLineItem*) of the currently processed transaction. The benefit is prorated to the eligible and discountable (line item have *NonDiscountableFlag* false) items. This prorated benefit is herein referred to as benefit share, discount share, and loyalty points share, respectively. For each benefit share either a retail price modifier *(RetailPriceModifier)* or a frequent shopper points modifier *(FrequentShopperPointsModifier*) is attached to the corresponding eligible item. The following steps are executed in order to obtain the benefit share for each item for the processed transaction-related promotion price derivation rule:

1. Calculation of the discount total amount
2. Calculation of the benefit shares for all line items except the last line item
3. Calculation of the residual for the last line item

Thereby, the system parameter *rebateShareMethod* is taken into account for discount shares but not for loyalty points. It defines the method for proportionally distributing the discount among the different items (for transaction-related discounts):

- SHARE: Shares for discounts in percent are calculated directly from sales unit price of an item, items are sorted according to their sales unit prices, starting with the item that has the smallest sales unit price. In case of equal sales unit prices the items will be sorted descending based on their line item sequence number, meaning the item registered last will be used to calculate the rebate share first.
- STANDARD: Standard calculation, line items sorted in order as they were registered (starting with the line item that has the smallest sequence number (*RetailTransactionLineItem.Key.retailTransactionLineItemSequenceNumber*) in the transaction).

The PCE checks the bonus points flag (*PromotionConditionRuleSO.bonusPointsFlag*) to decide if the system parameter *rebateShareMethod* is used or not.

8.3.2.1 Calculate Monetary Discount Shares

The discount total amount is the sum of the sales prices of all line items in the processed transaction. This means that all previously applied promotion price derivation rules are taken into account to calculate the discount shares for the processed transaction-related promotion price derivation rule. Note that the discount total amount must not be equal to 0, as a division by it can take place. If this is not the case, the proportional distribution will not be carried out.

If the system parameter *rebateShareMethod* is equal to STANDARD or the price modification percent *(RebatePromotionConditionRuleSO.priceModificationPercent*) is 0.0 or null, the discount total amount is used to obtain the ratio between it and the discount sales unit price of the items. This ratio corresponds to the discount share. Thus, the PCE uses the previously calculated discount and multiplies it with the discount sales price of the line item. This result is then divided by the discount total amount in order to get the discount share. Thereby, the ordering of the line items is according to their registration.

If the system parameter *rebateShareMethod* is equal to SHARE and the price modification percent is not 0.0 and is not null, a percentage is calculated in order to obtain the discount share. Thereby, the price modification percent is multiplied with the discount sales unit price of the item in order to compute the discount share for the corresponding item. An exception is the manually applied benefit, whereby no information about the discount type is given in the price derivation rule. In order that the percentage is calculated to obtain the discount share, the privilege type (*ManualPromotionTrigger.PrivilegeType*) is "RP". The discount share is then the multiple of the privilege value (*ManualPromotionTrigger.PrivilegeValue*) and the total discount. Otherwise, the discount share would be calculated according to the case where the price modification percent is 0.0 or null. Thereby, the line items that get a discount share are sorted in ascending order according to their current discount sales prices.

The resulting discount share is rounded according to the share rounding rule (*PromotionConditionRuleSO.shareRoundingRuleID*). It is then added to the retail price modifier of the corresponding line item.

The last line item that is processed within the calculation of the discount shares is handled differently by the PCE. The discount share of the last line item is the difference between the total discount and the sum over all already computed discount shares for the processed promotion price derivation rule.

Example: Calculate Monetary Discount Shares

Let us assume there are the following items:

- Pants: 40.50€

- Shirt: 25.00€

We also assume that there is the following promotion price derivation rule in the promotion master data:

1. Buy items amounting to 200€ and get 15% off on the transaction. Assume the sale return type (*PromotionConditionSO.saleReturnTypeCode*) is SALES_RETURNS and the calculation base type (*PromotionConditionRuleSO.calculationBase*) is CALCBASE_01 (sales minus returns).

The system parameter *rebateShareMethod* is equal to STANDARD:

The customer buys a shirt and five pants. Thereby, the shirt are registered at first and afterwards the pants.

The PCE computes the total discount of 34.13€. In order to obtain the discount shares, the PCE first orders the two line items according to their sequence numbers in the transaction. Thus, the discount share is firstly computed for the shirt and afterwards for the pants. The promotion price derivation rule has the price modification percent 15. Thus, the following discount shares are calculated:

- Shirt:
  - 25.00€ (discount sales unit price of the item) * 15% (price modification percent) = 3.75€
  - Prorated 3.75€ discount from 1. promotion price derivation rule
- Pants:
  - 40.50€ (discount sales unit price of the item) * 15% (price modification percent) = 6.08€
  - Prorated 6.08€ discount from 1. promotion price derivation rule to four pants (in total: 24.32€)
  - Last item (fifth pant) gets the residual: 34.13€ (discount) - 3.75€ (discount shares of the shirt) - 24.32€ (discount shares of the four pants) = 6.06€

	Sales	Return	Quantity	Regular Sales Price	Discount on the line items	Shared discount on the normalized items	Effective Sales Price	
	Shirt		1	25.00€	3.75€	3.75€	21.25€	
	Pants		5	202.50€	30.38€	6.08€	172.12€	
						6.08€		
						6.08€		
						6.08€		
						6.06€		
Regular Total Amount								227.50€

Discount of the promotion (subtracted from regular total amount)		34.13€
Effective Total Amount		193.37€

If we assume that the customer returns a shirt, buys a shirt and five pants, the total discount would be 30.38€. The proportionally distribution would then look like follows:

- Shirt (returned):
  - 25.00€ (discount sales unit price of the item) * 15% (price modification percent) = 3.75€
  - Prorated 3.75€ discount from 1. promotion price derivation rule
- Shirt (sold):
  - 25.00€ (discount sales unit price of the item) * 15% (price modification percent) = 3.75€
  - Prorated 3.75€ discount from 1. promotion price derivation rule
- Pants:
  - 40.50€ (discount sales unit price of the item) * 15% (price modification percent) = 6.08€
  - Prorated 6.08€ discount from 1. promotion price derivation rule to four pants (in total: 24.32€)
  - Last item (fifth pant) gets the residual: 30.38€ (discount) - 3.75€ (discount shares of the returned shirt) + 3.75€ (discount shares of the sold shirt) - 24.32€ (discount shares of the four pants) = 6.06€

	Sales	**Return**	**Quantity**	**Regular Sales Price**	**Discount on the line items**	**Shared discount on the normalized items**	**Effective Sales Price**	
		Shirt	1	25.00€	3.75€	3.75€	21.25€	
	Shirt		1	25.00€	3.75€	3.75€	21.25€	
	Pants		5	202.50€	30.38€	6.08€	172.12€	
						6.08€		
						6.08€		
						6.08€		
						6.08€		

Regular Total Amount		202.50€
Discount of the promotion (subtracted from regular total amount)		30.38€
Effective Total Amount		172.12€

<u>The system parameter *rebateShareMethod* is equal to SHARE:</u>
The customer buys a shirt and five pants. Thereby, the pants are registered at first and afterwards the shirt.
The PCE computes the total discount of 34.13€. In order to obtain the discount shares, the PCE first orders the two line items according to their sales prices. Thus, the discount share is firstly computed for the shirt and afterwards for the pants. The promotion price derivation rule has the price modification percent 10. Thus, the following discount shares are calculated:

- Shirt:
  - 25.00€ (discount sales unit price of the item) * 15% (price modification percent) = 3.75€
  - Prorated 3.75€ discount from 1. promotion price derivation rule
- Pants:
  - 40.50€ (discount sales unit price of the item) * 15% (price modification percent) = 6.08€
  - Prorated 6.08€ discount from 1. promotion price derivation rule to four pants (in total: 24.32€)
  - Last item (fifth pant) gets the residual: 34.13€ (discount) - 3.75€ (discount shares of the shirt) - 24.32€ (discount shares of the four pants) = 6.06€

	Sales	Return	Quantity	Regular Sales Price	Discount on the line items	Shared discount on the normalized items	Effective Sales Price	
	Pants		5	202.50€	30.38€	6.08€	172.12€	
						6.08€		
						6.08€		
						6.08€		
						6.06€		
	Shirt		1	25.00€	3.75€	3.75€	21.25€	

Regular Total Amount		227.50€
Discount of the promotion (subtracted from regular total amount)		34.13€
Effective Total Amount		193.37€

8.3.2.2 Calculate Virtual Discount Shares

The calculation of the virtual discount shares is done like the calculation of the monetary discount shares.

8.3.2.3 Calculate Loyalty Points shares

All previously applied promotion price derivation rules are taken into account to calculate the loyalty points shares for the processed transaction-related promotion price derivation rule. Note that the discount total amount must be unequal to 0, as a division by it can take place. If this is not the case, the proportional distribution will not be carried out.

The discount total amount is used to obtain the ratio between it and the discount sales unit price of the items. This ratio corresponds to the ratio between the loyalty points and the loyalty points share. Thus, the PCE uses the previously calculated loyalty points and multiplies them with the discount sales price of the line item. This result is then divided by the discount total amount in order to get the loyalty points share. Thereby, the order of the line items is according to their registration.

The resulting loyalty points share is rounded according to the share rounding rule. It is then added to the frequent shopper points modifier of the corresponding line item.

The last line item that is processed within the calculation of the loyalty points shares is handled differently by the PCE. The loyalty points share of the last line item is the difference between the loyalty points and the sum over all already computed loyalty points shares for the processed promotion price derivation rule.

Example: Calculate Loyalty Point Shares

Let us assume there are the following items:

- Pants: 40.50€
- Shirt: 25.00€

We also assume that there is the following promotion price derivation rule in the promotion master data:

1. Buy items amounting to 200€ at least and get 100 loyalty points on the transaction.

The customer buys a shirt and five pants. Thereby, the pants are registered at first and afterwards the shirt.

The PCE grants 100 loyalty points. In order to obtain the loyalty points shares, the PCE first orders the two line items according to their sequence numbers in the transaction. Thus, the loyalty points share is firstly computed for the pants and afterwards for the shirt. Thus, the following loyalty points shares are calculated:

- Pants:
  - 100 loyalty points * 40.50€ (discount sales unit price of the items) / 227.50€ (discount total amount) = 17.8 loyalty points share
  - Prorated 17.8 loyalty points from 1. promotion price derivation rule to each pants (in total: 89 loyalty points)
- Shirt:
  - Last item gets the residual: 100 (loyalty points) - 89 (loyalty points shares of the five pants) = 11 loyalty points share
  - Prorated 11.1 loyalty points from 1. promotion price derivation rule

Line item	Quantity	Regular Sales Price	Loyalty points on the line items	Shared loyalty points on the normalized items	Loyalty points in total
Pants	5	202.50€	89	17.8	100
				17.8	
				17.8	
				17.8	
				17.8	
Shirt	1	25.00€	11	11	

8.3.2.4 Prorate Transaction-Related Benefits Triggered by Items, Merchandise Categories, and Product Groups

The transaction-related benefit can be prorated differently to the line items in the transaction. This proration depends on the system parameter *TransactionRebateMethod*, which specifies how to determine the calculation base amount of the transaction-related discounts:

- TOTAL - total amount of the transaction is used for calculation base, amount of each line item is considered in the calculation.
- TRIGGER - only amounts of the eligible line items is used for calculation base, amounts of eligible line items in the transaction are considered in the calculation.

A special handling of transaction-related benefit happens if the benefit is triggered by an item, a merchandise category, or a product group.

In case when transactionRebateMethod is Trigger, the discount is not prorated to all line items in the transaction, the discount is prorated only to the eligible line items.

Example: Prorate Transaction-Related Benefits

Let us assume there are the following items:

- Shirt: 25.00€
- Pants: 40.50€

We also assume that system parameter rebateShareMethod is STANDARD and there are the following promotion price derivation rules in the promotion master data:

1. Buy a shirt and get 5.00€ off on the transaction.
2. Buy a shirt and pants and get 40.00€ off on the transaction.

The system parameter *transactionRebateMethod* is equal to TRIGGER:

The customer buys a shirt and pants. The PCE calculates the following discount shares:

- 1. Promotion price derivation rule

  Shirt:

  5.00€ (transaction discount) * 25.00€ (calculation base amount of the line item) / 25.00€ (discount total amount) = 5.00€

  Pants:

  0€ (transaction discount)

- 2. Promotion price derivation rule

  Shirt:

  40€ (transaction discount) * 20€ (calculation base amount of the line item) / 60.50€ (discount total amount) = 13.22€

  Pants:

  40€ (transaction discount) -13.22€ (discount shares of the shirt) = 26.78€

Sales Quantity		Regular Sales Price	Total discount from 1st promotion	Shared discoun from 1st promotion	Price of the line item after 1st promotion	Total discount from 2st promotion	Shared discount from 2st promotion	Effective Sales Price	Total amount
Shirt	1	25.00€	5.00€	5.00€	20.00€	40.00€	13.22€	6.78€	20.5€
Pants	1	40.50€		-	40.50€		26.78€	13.72€	

The system parameter *transactionRebateMethod* is equal to TOTAL:

The customer buys a shirt and pants. The PCE calculates the following discount shares:

- 1. Promotion price derivation rule

  Shirt:

  5.00€ (transaction discount) * 25.00€ (calculation base amount of the line item) / 65.50€ (discount total amount) = 1.91€

Pants:

5.00€ (transaction discount) - 1.91€ (discount shares of the shirt) = 3.09€

- 2. Promotion price derivation rule

Shirt:

40€ (transaction discount) * 23.09€ (calculation base amount of the line item) / 60.50€ (discount total amount) = 15.27€

Pants:

40€ (transaction discount) -15.27€ (discount shares of the shirt) = 24.73€

Sales	Quantity	Regular Sales Price	Total discount from 1st promotion	Shared discoun from 1st promotion	Price of the line item after 1st promotion	Total discount from 2st promotion	Shared discount from 2st promotion	Effective Sales Price	Total amount
Shirt	1	25.00€	5.00€	1.91€	23.09€	40.00€	15.27€	7.82€	20.5€
Pants	1	40.50€		3.09€	37.41€		24.73€	12.68€	

8.3.3 THERE IS MORE

The PCE distributes the benefit to the eligible line items and triggering line items, respectively, in case of line item-related promotion price derivation rules. As a result, the benefit is proportionally distributed to sales if the price derivation rule eligibility is active for sales. If the price derivation rule eligibility is active for the returns, the benefit is proportionally distributed to the returns in the transaction. In case that the price derivation rule eligibility is active for sales and for returns, the benefit is computed twice and distributed twice. The aforementioned attributes and system parameters are taken into account for the proportional distribution to the sales and to the returns.

The distribution of the benefit may depend on the trigger and eligible line items, respectively, in case of transaction-related promotion price derivation rules. In case the price derivation rule eligibility is an item, a merchandise category, and or a product group eligibility, the benefit is proportionally distributed to the triggering line items of the promotion price derivation rule. Otherwise, the entire transaction is taken into account meaning that the benefit is proportionally distribute to all line items in the transaction.

For the rounding behavior of the PCE, refer to chapter Round the Benefit.

Further information about the parameters and attributes can be found in chapter Configuration.

9 Configuration

9.1 System Parameters of the PCE

Name	Type	Default Value	Description
adHocPromotionID	long	-10000	The first value to be used as promotion ID for ad hoc promotions mapped by the PCE. Further ad hoc promotion objects mapped will get the next sequence number in decreasing order. test
allowZeroRebate	boolean	false	Decides whether a promotion price derivation rule which provides a zero discount will be used (**true**) or not (**false**).
cacheSize	long	5000	Defines the size of all local master data caches. This value is the count per type of element (that is, item eligibility, merchandise category eligibility, promotion). Each type has a separate cache that can keep up to 5000 different objects in the default setting.
calculationMode	CalculationMode	Basket	The calculation mode of the PCE. It can be provided via an element of the PCE-request. Possible values: • **Basket**: mapped to enum field BASKET. The PCE calculates the benefits for the entire shopping basket. • **LineItem**: mapped to enum field LINE_ITEM. The PCE calculates the benefits per line item independently. • **0**: the default value will be used.

Name	Type	Default Value	Description
calculationQuantityLimit	int	50000	Defines the overall calculation quantity limit for one transaction.
calculationTimeLimit	long	1000	Defines the time limit in ms for the best price algorithm.
charset	Charset	UTF-8	Defines the default Charset to be used. If value **0** is set, then the default value will be used.
checkApiVersion	boolean	false	Defines whether the request Client API version should be checked (**true**) or not (**false**).
conditionLimit	long	100	Defines the number of maximum applicable promotion price derivation rules having same sequence and resolution values.
conflictHandlerStrategy	ConflictHandlerStrategy	Greedy	Defines the conflict handler strategy to be used for best price calculation in case of collisions. Possible values: • **BruteForce**: mapped to enum field BRUTE_FORCE. In case of collision try all possible combinations of conditions during best price calculation in order to find the condition set giving the best price. • **Greedy**: mapped to enum value GREEDY. Apply a greedy approach to find the condition set giving the best price during best price calculation.
defaultBusinessUnitGroupId	long	1000000000000000 14	Gets the default business unit group identifier.

Name	Type	Default Value	Description
discountVectorForItemEligibilities	boolean	false	Specifies whether the discount control vector check is used for promotion price derivation rules with item eligibilities (**true**) or not (**false**).
discountVectorMethod	DiscountVectorMethodCode	Common	Defines which positions are checked against the discount vector. Possible values: • **All**: mapped to enum field ALL. Promotion validity level. • **Common**: mapped to enum field COMMON. Price derivation rule eligibility validity level. • **0**: the default value will be used.
enableExternalActionEligibilities	boolean	false	Enable handling of external action eligibilities. With this enabled external action promotion will be evaluate triggers and items in the same way as other promotions. It will add PriceModifier only on triggering items. With disabled setting the triggers are only checked initially if they are fullfilled and then the promotion is applied to complete transaction.
enabled	boolean	true	Enabling of the promotion service (**true**) or not (**false**).

Name	Type	Default Value	Description
incrementalCalculationMode	IncrementalCalculationMode	RecalculateAll	Enum for controlling the behavior of incremental calculation mode. It is not used in server embedded mode. This value controls the reuse of session (context) information if they are passed with a calculation request. Possible values: <ul><li>**Disabled**: No incremental calculation at all, always a fresh context is used (not supported by the OPP or PPS, respectively).</li><li>**RecalculateAll**: Keep context, but recalculate all promotions completely.</li><li>**Incremental**: If a set of changes is passed, this will be used to determine what can be kept in the context and what needs to be recalculated (not supported by the OPP or PPS, respectively).</li><li>**0**: the default value will be used (not supported by the OPP or PPS, respectively).</li></ul>

Name	Type	Default Value	Description
itemChooseMethod	ChooseItemMethod	Lowest	Determines in which order items are considered for the calculation. Possible values: <ul><li>**Lowest**: mapped to enum field LOWEST_FIRST/ "01". The cheapest item is chosen first.</li><li>**Highest**: mapped to enum field HIGHEST_FIRST/ "02". The most expensive item is chosen first.</li><li>**LowestPerInterval**: mapped to enum field LOWEST_FIRST_INT/"03". Handled like value LOWEST_FIRST.</li><li>**HighestPerInterval**: mapped to enum field HIGHEST_FIRST_INT/"04". Handled like value HIGHEST_FIRST.</li><li>**0**: the default value will be used.</li></ul>
itemTypesWithoutDiscounts	List<String>	Empty list	List of line item types generally excluded from discount. The values are expected to be passed as a list with delimiter ";".
itemTypesWithoutReceiptRelatedDiscounts	List<String>	SALES_ORDER("SO")	List of line item types which are not allowed for transaction-related monetary discounts (see PositionType). The values are expected to be passed as a list with delimiter ";".
itemTypesWithoutReceiptRelatedPoints	List<String>	SALES_ORDER("SO")	List of line item types which are not allowed for transaction-related loyalty points (see PositionType). The values are expected to be passed as a list with delimiter ";".

Name	Type	Default Value	Description
localMasterdataCacheEnabled	boolean	true	Defines whether the local master data caching is enabled or not.
manufacturerCouponGracePeriod	Integer	0	Defines the grace period in days with which the manufacturer coupon validity period must be extended.
manufacturerCouponPromotionSequence	long	-10000	Defines the condition sequence to be used for manufacturer coupon related ad hoc promotions.
manufacturerCouponPromotionTenderTypeCode	String	"ZTPR"	The rule tender type code to map in case of manufacturer coupon related ad hoc promotions.
merchandiseSetsEnabled	boolean	true	Decides whether a product group is adaptable by the PCE (**true**) or not (**false**).
minimumQuantityForSingleNormalizedItemCreation	int	30000	Defines a minimum quantity above which the line item normalization should produce a single normalized item with quantity > 1 for performance improvement.
nonPieceItemPerPosition	boolean	false	Specifies whether the percent discount for non-piece items (QuantityInputMethod is not PIECE) should be calculated per unit of measure (**true**) or per PIECE (**false**).

Name	Type	Default Value	Description
percentualPointsMethod	RoundingPercentualPointsMethodCode	BEFORE	Defines the method which is used to calculate percentual loyalty points promotion price derivation rules values. Possible values (case insensitive): • **AFTER**: Round at the end of the calculation for unit of measure. • **BEFORE**: Round before calculation for unit of measure. • **0**: the default value will be used.
pointsFactor	BigDecimal	1.0	Factor by which the discount percentage amount is multiplied to grant loyalty points.
pointsRating	BigDecimal	0.01	Factor by which the loyalty points are multiplied to compare discounts with loyalty points.
pointsShareDecimalPlacesCount	int	DEFAULT_POINTS_SCALE = 0 (Default value for scale used for points rounding; constant field values)	Defines the number of decimal places (0 - 4) for loyalty points splitting.
pointsShareRoundingMethod	RoundingMethod	Commercial	Defines the rounding method for loyalty points shares: The following values are possible (case insensitive): • **Commercial**: Maps to enum field HALF_UP. Round half up, commercial. • **Down**: Maps to enum field DOWN. Round down. • **Up**: Maps to enum field UP. Round up. • **0**: the default value will be used.

Name	Type	Default Value	Description
priceTypeCodesWithoutDiscounts	List<String>	Empty list	List of price types (SaleReturnLineItem.priceTypeCode) generally not considered by the PCE. The values are expected to be passed as a list with delimiter ";".
processingMode	RequestResponseProcessingMode	Service	Defines the PCE processing mode. Possible values: • **Service**: if the PPS is used • **POS**: used for direct communication with POS systems. • **0**: the default value will be used.
rebateShareDecimalPlacesCount	int	2	Defines the number of decimal places (0 - 4) for discount splitting.

Name	Type	Default Value	Description
rebateShareMethod	RebateShareCalculationMethodCode	SHARE	Defines the method for proportionally prorating the transaction-related discount among the different line items (for transaction-related monetary discounts, not used for loyalty points). • **SHARE**: Shares for discounts in percent are calculated directly from sales price of a line item, line items are sorted according to their sales prices, starting with the line item that has the smallest sales price. In case of equal prices the items will be sorted descending based on their line item sequence number, meaning the item registered last will be used to calculate the rebate share first. • **STANDARD**: Standard calculation, line items sorted in order as they were registered. • **0**: the default value will be used.

Name	Type	Default Value	Description
rebateShareRoundingDestinationValue	int	1	Defines the rounding destination value for discount shares, which are used to round the value to the proper denomination (not used for loyalty points). Additional rounding information: • "1"/**null**: No additional treatment of the rounded result necessary. • "5": The rounded result is rounded again to the next, respectively, to the previous multiple of 5.
rebateShareRoundingMethod	RoundingMethod	Commercial	Defines the rounding method for discount shares (not used for loyalty points). • **Commercial**: Maps to enum field HALF_UP. Round half up, commercial. • **Down**: Maps to enum field DOWN. Round down. • **Up**: Maps to enum field UP. Round up. • **0**: the default value will be used.

Name	Type	Default Value	Description
saleReturnTypeCodesForReturns	List<SaleReturnTypeCode>	02;00	Sets the sale return type codes for returns. Possible values: • **00**: Mapps to enum field SALES_RETURNS. Sales and returns are considered. • **01**: Mapps to enum field SALES. Sales are considered only. • **02**: Mapps to enum filed RETURNS. Returns are considered only. The values are expected to be passed as a list with delimiter ";".
saleReturnTypeCodesForSales	List<SaleReturnTypeCode>	01;00	Sets the sale return type codes for sales. Possible values: • **00**: Mapps to enum field SALES_RETURNS. Sales and returns are considered. • **01**: Mapps to enum field SALES. Sales are considered only. • **02**: Mapps to enum filed RETURNS. Returns are considered only. The values are expected to be passed as a list with delimiter ";".

Name	Type	Default Value	Description
timeValidationMethod	TimeValidationMethodCode	PROMOTION	Method to determine the validity level of a condition. Possible values (case insensitive): • **PROMOTION**: Promotion validity level. • **ELIGIBILITIES**: Price derivation rule eligibility validity level. • **0**: the default value will be used.
transactionRebateMethod	TransactionRebateCalculationMethodCode	TRIGGER	Specifies how to determine the calculation base amount of the transaction-related discounts. Possible values (case insensitive): • **TOTAL** • **TRIGGER** • **0**: the default value will be used.
userActionProhibited	boolean	true	Specifies whether it is forbidden that the user selects or enters a promotion price derivation rule value.

9.2 Promotion Master Data Codes

Name	Description
AdjustmentMethodCode	A mnemonic code denoting what kind of adjustment is being made to the regular sales price of the item. • INCREASE: Increase amount with calculated value. • DECREASE: Decrease amount with calculated value.
AmendmentTypeCode	Determines whether the promotion price derivation rule can be used only for sales, only for amendments, or for both. • SALES_AMENDMENTS/"00": For sales as well as for amendments. • SALES/"01": Only for sales. • AMENDMENTS/"02": Only for amendments.
CalculationBaseType	The code is only influencing the calculation base amount for transaction-related promotion price derivation rules. It defines how the calculation base amount is obtained on the basis of the sold and returned line items in the context of a transaction. • CALCBASE_00/"00": Calculation base amount is the sum of all sales in a transaction. • CALCBASE_01/"01": Calculation base amount is the sum of all sales minus the returns in a transaction.

Name	Description
ChooseItemMethod	Defines the order in which items are used in promotion calculation. • LOWEST_FIRST/"01": Cheapest item is chosen first. • HIGHEST_FIRST/"02": Most expensive item is chosen first. • LOWEST_FIRST_INT/"03": Handled like value LOWEST_FIRST. • HIGHEST_FIRST_INT/"04": Handled like value HIGHEST_FIRST.
CombinationCode	Code denoting the style of combination that is to be applied across the child eligibilities. • AND: Combination via a logical AND operator. • OR: Combination via a logical OR operator. • INTERSECTION: Combination via an intersection. • ITEM_OR: Combination for simple product groups.
CouponConsumptionCode	Defines the consumption of coupons. • CONSUME/"00"/**null**: Consumption of coupons depends on the other eligibilities inside the condition. • CONSUME_PER_ITEM/"01": Coupon is consumed (one coupon per item). • NOT_CONSUMED/"02": Coupon is not consumed (one coupon for all items).
CouponPrintoutRule	The typecode of the printout rule. • SEPARATE_RECEIPT/"00": Print as extra receipt. • PRINT_AT_END/"01": Print at the end of the transaction.
DiscountMethodCode	The type code of the printout rule. • NORMAL_REBATE/"00": Normal rebate • GIFT_CERTIFICATE/"01": Rebate as gift certificate • TENDER/"02": Rebate as tender • GIFT_CERTIFICATE_CHARGED/"03": Rebate as gift certificate charged • COUPON/"04": Rebate as coupon • MANUFACTURER_COUPON_TENDER/"99": Rebate as manufacturer coupon related tender
EligibilityStatusCode	Defines the price derivation rule eligibility status codes values. • STATUS_CODE_ACTIVE: Status active. • STATUS_CODE_INACTIVE: Status inactive.
EligibilityType	Price derivation rule eligibility type interface. Each price derivation rule eligibility has a distinct typecode. • COMB: Combination eligibility. • EGRP: Employee eligibility. • CGRP: Customer group eligibility. • CUST: Customer eligibility. • MSTR: Merchandise category eligibility. • ITEM: Item eligibility. • TOTL: Shopping basket eligibility. • COUP: Coupon eligibility. • MANU: Manual trigger eligibility. • POST: Position type eligibility. • EXTV: External trigger eligibility. • MSET: Product group eligibility. • MACP: Manufacturer coupon eligibility. • APRT: Additional price type eligibility.

Name	Description
MerchandiseSetTypeCode	Defines the type of a product group element. • MRHRC: Merchandise category. • ITM: Item. • OPR: Set operation.
MixAndMatchCombinationType	Determines how the rule matching items belonging to the price derivation rule can be combined. • OR: Combination via a logical OR operator. • AND: Combination via a logical AND operator. • OR_QUANTITY: Combination via a logical OR operator with a certain threshold quantity. • INTERSECTION: Combination of merchandise categories via an intersection. • INTERSECTION_QUANTITY: Combination of merchandise categories via an intersection with a certain threshold quantity. • ST: Set a discount price for all rule matching items. • XO: Exclusive OR combination. Only allowed for manufacturer coupon related promotions.
PositionType	Type safe, extensible enumeration in a PositionTypePromotionConditionEligibilitySO. • COMMON/"CO": Common sale return line item. • DEPOSIT_ITEM/"DI": Deposit line item. • DONATION/"DO": Invoice payment line item. • DOWNPAYMENT/"DP": Down payment line item. • DOWNPAYMENT_CLEARING/"DC": Down payment clearing line item. • EMPTIES_RETURN/"ER": Empties return line item. • GIFT_CERTIFICATE/"GC": Gift certificate line item. • INVOICE_PAYMENT/"IP": Invoice payment line item. • ITEM_INFORMATION/"II": Invoice payment line item. • PAY_IN/"PI": Pay in line item. • PAY_OUT/"PO": Pay out line item. • PREPAID/"PR": Prepaid line item. • PROMOTION_GIFT_CERTIFICATE/"PG": Promotion gift certificate line item. • SALES_ORDER/"SO": Sales order line item. • SALES_ORDER_PICK_UP/"PU": Sales order pickup line item. • SCALE_RECEIPT/"SR": Scale receipt line item.
PriceModificationMethod	A code denoting the method of modifying the price or loyalty points that are being applied to the transaction. • DISCOUNT_SINGLE/"RS": Reduction of the sales unit price by an amount. • DISCOUNT_PERCENT/"RP": Reduction of the sales unit price by a percentage. • FIXED_PRICE/"PS": Setting the discount price of the sales unit price. • DISCOUNT_TOTAL/"RT": Reduction of the summed up sales prices by an amount. • FIX_PRICE_TOTAL/"PT": Setting the discount of the summed up sales prices. • SET_PRICE_TOTAL/"ST": Set price total. • DISCOUNT_PERCENT_TOTAL/"TP": Reduction of the summed up sales prices by a percentage. • DISCOUNT_PERCENT_TOTAL2/"T2": Reduction of the summed up sales prices by a percentage with different rounding. • DISCOUNT_INTERVAL_TOTAL/"PI": New total price per interval.

Name	Description
PrivilegeType	For mapping privilege type values for manual benefits. • RS: Reduction of the sales unit price by an amount. • RP: Reduction of the sales unit price by a percentage. • PS: Setting the discount price of the sales unit price. • RT: Reduction of the summed up sales prices by an amount. • PT: Setting the discount of the summed up sales prices. • TP: Reduction of the summed up sales prices by a percentage. • T2: Reduction of the summed up sales prices by a percentage with different rounding. • ST: Set price total.
ReductionMethodMixAndMatch	A code denoting the method of modifying the price that is being applied in case of a mix and match. • REBATE_SINGLE/"RS": Reduction of the sales unit price by an amount. • REBATE_PERCENT/"RP": Reduction of the sales unit price by a percentage. • PRICE_SINGLE/"PS": Setting the discount price of the sales unit price. • ALLOCATION_PERCENT/"MP": Set a discount price for all rule matching items.
RoundingMethod	Rounding methods. • NO_ROUNDING: No rounding. • HALF_UP: Round half up, commercial (round up for greater than or equal to the half of the *multiple*, else round down). • DOWN: Round down. • UP: Round up. • HALF_DOWN: Round half down (round up for greater than the half of the *multiple*, else round down).
RoundingMethodCode	Determines how the calculated benefit is to be rounded. • COMMERCIAL_ROUNDING/"00": Commercial rounding, round half up. • DOWN/"01": Round down. • UP/"02": Round up.
RuleControlType	Determines whether the price derivation rule is to be applied during the entry of the transaction. • PO: Line item-related price derivation rule, calculated after each line item. • SU: Transaction-related price derivation rule, calculated after subtotal. • PC: Line item-related price derivation rule, calculated after subtotal. • SP: Transaction-related price derivation rule, calculated after each line item. If there is another than the above-mentioned supported value in the request, error ID GKR-100500 is sent back. If this value is missing in the request, the default value is set to SU.
RuleType	Defines the type of price derivation rule. • RB: Either simple discount or loyalty points. • MM: Mix and match. • NO: No discount. • GP: Buy one get one. • MA: Manual price derivation rule. • EX: External action price derivation rule. • AP: Additional price type rule.

Name	Description
SaleReturnTypeCode	Determines whether the promotion price derivation rule can be used only for sales, only for returns, or for both. • SALES_RETURNS/"00": Sales and returns are considered. • SALES/"01": Sales are considered only. • RETURNS/"02": Returns are considered only.
ThresholdType	A code for the type of threshold which applies to a price derivation rule eligibility. • NONE: No threshold type. • QUT: The threshold and the limit are set for the quantity. • QUTI: The threshold, the interval, and the limit are set for the quantity. • AMT: The threshold and the limit are set for the amount. • AMTI: The threshold, the interval, and the limit are set for the amount. • AMTS: The threshold, the interval, and the limit are set for the regular sales price. • AMQU: The threshold is set for the quantity and the amount. • COMB: See chapter Use a Simple Product Group as a Trigger.
TriggerValueTypeCode	The type of the TriggerValue for ExternalTriggerValuePromotionConditionEligibilitySO. • TURNOVER/"00": Turnover of the (current) year. • REBATE_AMOUNT/"01": Discount amount which was granted during the (current) year.

9.3 Transaction Data Codes

Name	Description
AdditionalPriceTypeCode	Additional price type code. • CS: Customer specific price.
CalculationMethodCode	A mnemonic code denoting how the price modification calculation was performed. • MANUAL_PRICE_OVERRIDE: Manual price override calculation method.
PriceModificationMethodTx	PriceModification codes for transaction data mapping. Combination of RuleType and PriceModificationMethod. • RS: Reduction of the sales unit price by an amount. • RP: Reduction of the sales unit price by a percentage. • PS: Setting the discount price of the sales unit price. • RT: Reduction of the summed up sales prices by an amount. • PT: Setting the discount of the summed up sales prices. • ST: Set price total. • TP: Reduction of the summed up sales prices by a percentage. • T2: Reduction of the summed up sales prices by a percentage with different rounding. • PI: New total price per interval. • RM: Manual price derivation rule. • NO: No discount. • MM: Mix and match. • GP: Buy one get one. • EX: External action price derivation rule.

Name	Description
RetailTransactionLineItemTypeCode	A code to denote the type of retail transaction line item. • CHILD_ITEM: Child item. • LOYALTY_REWARD_LINE_ITEM: Loyalty reward line item. • PRICE_MODIFICATION_LINE_ITEM: Price modification line item. • ROUNDING_LINE_ITEM: Rounding line item. • SALE_RETURN_LINE_ITEM: Sale return line item. • TAX_LINE_ITEM: Tax line item. • TENDER_LINE_ITEM: Tender line item. • VOIDS_LINE_ITEM: Voids line item. • WORKER_DISCOUNT_LINE_ITEM: Worker discount line item.
SaleReturnLineItemActionCode	A code denoting how the item is being treated in the line item. • RETURN_ITEM: Return item. • SALE_ITEM: Sale item.
SalesOrderDeliveryTypeCode	Type-safe extensible enumeration for mapping the delivery typecode of the related sales order. • PICKUP_OWN_STORE: Pick up in own store. • DELIVERY: Delivery. • IMMEDIATE_PICKUP: Pick up immediately. • PICKUP_FOREIGN_STORE: Pick up in foreign store.
QuantityInputMethod	Sale return line items quantity input methods. • PIECE: Solid pieces (single quantity). • LENGTH: Length. • AREA: Area (length and width). • VOLUME: Length and width and height. • WEIGHT: Weight. • MEASUREMENT: General measurement (for arbitrary units of measure). • PARTITION: Divisible pieces (quantity with decimal places).

9.4 Objects and Attributes from the Master Data

Name	Default Value	Data type, range	Mandatory	Description
AdditionalPriceTypePromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				
additionalPriceTypeCode	-	AdditionalPriceTypeCode	Yes	This attribute contains the price type of a particular price that is triggered by an additional price in the price collection.
AdditionalPriceTypePromotionConditionRuleSO extends PromotionConditionRuleSO				
additionalPriceTypeCode	-	AdditionalPriceTypeCode	No	The price type of the promotion defined by the connected eligibility.
CombinationPromotionConditionEligibilitySO extends ThresholdPromotionConditionEligibility				

Name	Default Value	Data type, range	Manda tory	Description
combinationCode	-	CombinationCode	Yes	Code denoting the style of combination that is to be applied across the child eligibilities.
childEligibilityList	-	List<PromotionConditionEligibilitySO>	No	List of all child eligibilities.
thresholdForSingleItemFlag	false	Boolean	No	If set to **false** in a merchandise category, the quantity and/or the amount of all eligible items in the transaction are summarized and checked against the quantities/amounts indicated in the price derivation rule eligibility. Otherwise, the quantity and/or amount are summarized individually for each item that triggers the corresponding price derivation rule eligibility. Not supported by OPP/PPS.
CouponPromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				
couponNumber	-	String	Yes	Identifier of a coupon.
consumptionTypeCode	-	CouponConsumptionCode	Yes	Defines the consumption of coupons.
CurrencyRoundingRuleDO				
roundMethodCode	-	RoundingMethod	No	Rounding methods defined in configuration.
roundDecimalCount	-	Integer	No	Count of decimals which the result should have.
roundDestinationValue	-	Integer	No	Defines the rounding destination value, which is used to round the value to the proper denomination.
CustomerGroupPromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				
customerGroupID	-	String	No	Identifier of a customer group.
CustomerPromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				
customerID	-	String	No	Identifier of a customer.
ExternalActionParameterSO				

Name	Default Value	Data type, range	Mandatory	Description
parameterID	-	String	No	Identifier of a parameter that is needed by an external action.
parameterValue	-	String	No	Value of a parameter that is needed by an external action.
ExternalActionPromotionConditionRuleSO extends PromotionConditionRuleSO				
externalActionID	-	String	No	Identifier of an external action.
externalActionDescription	-	String	No	Description of an external action.
externalActionParameterList	-	List<? extends ExternalActionParameterSO >	No	List of external action parameters and their values.
externalActionTextList	-	List<? extends ExternalActionTextSO>	No	List of external action texts.
ExternalActionTextSO				
textID	-	String	No	Identifier for a text that is needed by an external action.
text	-	String	No	The text that is needed by an external action.
language	-	String	No	Language of the text.
ExternalTriggerValuePromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				
triggerValueTypeCode	-	TriggerValueTypeCode	No	The type of the trigger value.
triggerValue	-	BigDecimal	No	The value or amount, respectively.
freeTriggerValueTypeCode	-	String	No	The external trigger value typecode.
Get3Pay2PromotionConditionRuleSO extends PromotionConditionRuleSO				
toBePaidQuantity	-	BigDecimal	No	Number of items which is to be paid.
ItemPromotionConditionEligibilitySO extends ThresholdPromotionConditionEligibility				
itemID	-	String	Yes	Unique item identifier.
unitOfMeasureCode	-	String	Yes	Unit of measure of an item.

Name	Default Value	Data type, range	Mandatory	Description
ManualPromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				
triggerType	-	String	Yes	Reference to the Clients. Arbitrary value.
triggerValue	-	String	Yes	Reference to the Clients. Arbitrary value.
thresholdValue	0.0	BigDecimal	No	Start value (lower bound) for choosing. Not supported by OPP/PPS. Data Model in GK Environment • The default value is null.
intervalValue	0.0	BigDecimal	No	Interval value for choosing. Not supported by OPP/PPS. Data Model in GK Environment • The default value is null.
limitValue	0.0	BigDecimal	No	Limit value(upper bound) for choosing. Not supported by OPP/PPS. Data Model in GK Environment • The default value is null.
privilegeType	-	PrivilegeType	No	Setting the modification of the discount. Not supported by OPP/PPS.

Name	Default Value	Data type, range	Manda tory	Description
userActionRequiredFlag	false	Boolean	No	Determines whether additionally it is necessary that the user presses a button in order that the price derivation rule eligibility is fulfilled.
ManufacturerCouponPromotionConditionEligibilitySO extends ThresholdPromotionConditionEligibility Only available in the PCE internal model, not part of the promotion model.				
companyCode	-	String	Yes	The company code to match.
familyCode	-	String	Yes	The family code to match.
parseEligibilityFlag	false	boolean	Yes	The parse eligibility flag. In case of value true, the eligibility is considered during eligibility activation and rule application both. In case of value false the eligibility is only considered for activating the eligibility, but not for rule calculation.
unitOfMeasureCode	-	String	Yes	The unit of measure code to match.
ManufacturerCouponQualifyingPurchaseSO Only available in the PCE internal model, not part of the promotion model.				
companyCode	-	String	Yes	The company code of the purchase requirement. It is a globally unique number assigned to a company by a GS1 Member Organization used to identify the manufacturer or organization making this coupon offer.
familyCode	-	String	Yes	The family code of the purchase requirement, which is assigned by the manufacturer to identify the qualifying family of products to be purchased.

Name	Default Value	Data type, range	Mandatory	Description
purchaseRequirementCode	-	String	Yes	The purchase requirement code which controls how the purchase requirement should be considered. The supported values are: • 0: the threshold number of units to purchase • 1: the threshold amount of the accumulative total of the qualifying purchase items (2 decimals) • 2: the threshold cash value of the total transaction (2 decimals) • 3: the threshold number of pounds (2 decimals) • 4: the threshold number of kilograms (3 decimals)
purchaseRequirement	0.0	BigDecimal	Yes	
ManufacturerCouponSO Only available in the PCE internal model, not part of the promotion model.				
additionalPurchaseRulesCode	-	String	No	The Additional Purchase Rules Code specifies which items must be purchased to qualify. Supported values are: • 0 - Primary item OR 2nd item OR 3rd item is required to satisfy their own purchase requirement, • 1 - Primary item AND 2nd item AND 3rd item is required to satisfy their own purchase requirement, • 2 - Primary item AND (2nd item OR 3rd item) is required to satisfy their own purchase requirement, • 3 - 2nd OR 3rd item is needed to satisfy the primary purchase requirement.
appliedIntervalCount	-	BigDecimal	No	The applied interval count.

Name	Default Value	Data type, range	Mandatory	Description
barcode	-	String	Yes	The GS1 barcode.
doNotMultiplyFlag	false	boolean	No	If = 1 then this offer must not be multiplied (default is 0), mapped to a boolean value.
expirationDate	-	LocalDate	No	The expiration date of the offer and must match the human readable expiration date on the coupon. It is in the format of YYMMDD.
itemsUsedInCalculation	null	List<RetailTransactionLineItem.Key>	No	The list of items used in the calculation.
offerCode	-	String	Yes	The Offer Code is a 6-digit number that identifies this offer.
primaryQualifyingPurchase	-	ManufacturerCouponQualifyingPurchaseSO	Yes	The primary qualifying purchase.
saveValue	-	BigDecimal	Yes	The amount of discount to be granted. The Save Value Code defines its format. If the Save Value Code is missing, then it must be considered as cents off.
saveValueAppliesTo	-	String	No	This data element only applies when multiple purchase requirements are present and indicates which qualifying item the savings apply to: • 0: is the Primary Qualifying Item (default) • 1: is the 2nd Qualifying Item • 2: is the 3rd Qualifying Item

Name	Default Value	Data type, range	Mandatory	Description
saveValueCode	-	String	No	The save value code which controls how the save value should be considered. The supported values are: • 0 / null: cents off qualifying purchase items, • 1: if Save Value = 0 one qualifying purchase item unit is free. If Save Value > 0 the one qualifying purchase item unit is free up to maximum amount in Save Value, • 2: the number of qualifying purchase item units that are free, • 5: percent off qualifying purchase item, • 6: cents off final transaction amount
saveValueStringRepresentation	-	String	Yes	The amount of discount to be granted in its String representation.
secondQualifyingPurchase	-	ManufacturerCouponQualifyingPurchaseSO	No	The 2nd qualifying purchase.
startDate	-	LocalDate	No	The start date for the offer and must match the human readable start date on the coupon. It must be earlier than or equal to the expiration date and is in the format YYMMDD.
storeCouponFlag	-	String	No	In case of value 0 or null the manufacturer coupon is not a store coupon. In case of values >0 the coupon is a store coupon and will not be processed by the PCE.
thirdQualifyingPurchase	-	ManufacturerCouponQualifyingPurchaseSO	No	The 3rd qualifying purchase.
totalDiscountAmount	-	BigDecimal	No	The total discount amount granted by the manufacturer coupon.
MarketBasketAmountEligibilitySO extends PromotionConditionEligibilitySO				

Name	Default Value	Data type, range	Mandatory	Description
marketBasketThresholdAmount	-	BigDecimal	Yes	The extended regular sales price required to cause the application of the price derivation rule - before application of the corresponding price derivation rule.
internalMarketBasketTriggerID	-	Long	No	Internal identifier. Not supported by OPP/PPS.
MatchingItemRelationSO				
itemID	-	String	Yes	Identifier of an item.
uomCode	-	String	Yes	Unit of measure of an item.
internalPromotionalProductID	-	Long	No	Internal generated identifier of promotional product.
internalVirtualMhgID	-	String	No	Internal generated identifier of a merchandise category.
MatchingItemSO				
internalRuleID	-	Long	No	An identifier for a specific price derivation rule.
priceModificationPercent	0.0	BigDecimal	No	The percentage modification each time this price derivation rule is applied. Data Model in GK Environment • The default value is null.

Name	Default Value	Data type, range	Mandatory	Description
priceModificationAmount	0.0	BigDecimal	No	The monetary discount of the modification each time this price derivation rule is applied. Data Model in GK Environment • The default value is null.
newPriceAmount	0.0	BigDecimal	No	A price modification expressed as a new price. Data Model in GK Environment • The default value is null.
priceModificationAllocationPercent	0.0	BigDecimal	No	The percentage of the total modification from the promotion that is to be applied. Data Model in GK Environment • The default value is null.
reductionMethodCode	-	ReductionMethodMixAndMatch	Yes	A code denoting the method of modifying the price that is being applied.
adjustmentMethodCode	-	AdjustmentMethodCode	No	A mnemonic code denoting what kind of adjustment is being made to the regular sales price of the item.

Name	Default Value	Data type, range	Mandatory	Description
value	0.0	BigDecimal	No	The percentage modification, modification amount, or new price amount according to the reductionMethodCode. Data Model in GK Environment • The default value is null.
requiredQuantity	0.0	BigDecimal	No	The count of the item which is needed in the context of the mix and match. Data Model in GK Environment • The default value is null.
itemList	-	List<? extends MatchingItemRelationSO>	No	Promotional product: Items.
mhgList	-	List<? extends MatchingMhgRelationSO>	No	Promotional product: Merchandise categories.
merchandiseSet	-	MerchandiseSetSO	No	Promotional product: Product groups.
matchingItemID	-	Long	Yes	Identifier of a rule matching item.
manufacturerCoupon	-	MatchingManufacturerCouponRelationSO	No	Promotional product: manufacturer coupon. Only available in the PCE internal model, not part of the promotion model.

Name	Default Value	Data type, range	Mandatory	Description
MatchingManufacturerCouponRelationSO				
Only available in the PCE internal model, not part of the promotion model.				
companyCode	-	String	Yes	The company code to match.
familyCode	-	String	Yes	The company code to match
unitOfMeasureCode	-	String	Yes	The unit of measure code to match.
MatchingMhgRelationSO				
mhgID	-	String	Yes	Identifier of a merchandise category.
mhgQualifier	-	String	Yes	Qualifier of a merchandise category.
internalPromotionalProductID	-	Long	No	Internal generated identifier of promotional product.
internalVirtualMhgID	-	String	No	Internal generated identifier of a merchandise category.
MerchandiseSetPromotionConditionEligibilitySO extends ThresholdPromotionConditionEligibility				
merchandiseSetID	-	String	Yes	Identifier of a product group.
isMixingForbidden	-	boolean	Yes	Indicates that it is not possible to mix different items of the specified merchandise set in order to reach the required threshold.
merchandiseSet	-	MerchandiseSetSO	No	Deprecated.
MerchandiseSetRelationSO				
childMerchandiseSetID	-	Long	No	Identifier of a child product group.
parentMerchandiseSetID	-	Long	No	Identifier of a parent product group.
rootMerchandiseSetID	-	Long	No	Identifier of the root product group.
merchandiseSetTypeCode	-	MerchandiseSetTypeCode	No	Defines type of a product group element.

Name	Default Value	Data type, range	Mandatory	Description
merchandiseSetValue	-	String	No	The content of an element (code of the set operation, item identifier, or merchandise category identifier).
merchandiseSetValue2	-	String	No	Further optional information regarding to a set element (unit of measure, merchandise category qualifier).
MerchandiseSetSO				
internalMerchandiseSetGroupID	-	Long	No	Identifier of a product group.
internalPromotionalProductID	-	Long	No	Identifier of a promotional product.
merchandiseSetID	-	Long	Yes	Identifier for the product group element.
merchandiseSetElementList	-	List<? extends MerchandiseSetRelationSO >	No	List of the elements of a product group.
MHGPromotionConditionEligibilitySO extends ThresholdPromotionConditionEligibility				
merchandiseHierarchyGroupIDQualifier	-	String	Yes	Qualifier of the merchandise category.
merchandiseHierarchyGroupID	-	String	Yes	Identifier of the merchandise category.
thresholdForSingleItemFlag	-	Boolean	Yes	If set to **false** in a merchandise category, the quantity and/or the amount of all eligible items in the transaction are summarized and checked against the quantities/amounts indicated in the price derivation rule eligibility. Otherwise, the quantity and/or amount is summarized individually for each item that triggers the corresponding price derivation rule eligibility.
MixAndMatchPromotionConditionRuleSO extends PromotionConditionRuleSO				
limitCount	-	BigDecimal	Yes	The maximum number of rule matching items that may be purchased along with the item that triggered the price derivation rule eligibility.

Name	Default Value	Data type, range	Mandatory	Description
combinationCode	-	MixAndMatchCombinationType	Yes	Determines how the rule matching items belonging to the price derivation rule can be combined.
matchingItemList	-	List<? extends MatchingItemSO>	No	List of rule matching items.
newSetPriceAmount	-	BigDecimal	No	New price of the set which is defined by the rule matching items of the promotion price derivation rule.
PositionTypePromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				
positionType	-	PositionType	No	Type of an item.
PromotionConditionEligibilitySO				
InternalEligibilityID	-	Long	Yes	Internal identifier of a price derivation rule eligibility.
typeCode	-	EligibilityType	Yes	A code that indicates the type of the price derivation rule eligibility.
rootEligibilityID	-	Long	No	Reference to the root price derivation rule eligibility of the combination eligibility.
parentEligibilityID	-	Long	Yes	Reference to the parent eligibility.
levelID	-	Short	No	Level in the combination eligibility the price derivation rule eligibility belongs to. Not supported by OPP/PPS.
effectiveDateTime	-	Timestamp	Yes	Time stamp when this price derivation rule eligibility starts to be effective.
expirationDateTime	-	Timestamp	Yes	Time stamp when this price derivation rule stops to be effective.
statusCode	-	String	Yes	Defines the current status of the price derivation rule eligibility.

Name	Default Value	Data type, range	Manda tory	Description
negationFlag	false	Boolean	No	Determines whether the price derivation rule eligibility must be fulfilled (**false**) or not (**true**) in order that the price derivation rule can be applied. Not supported by OPP/PPS.
PromotionConditionRuleSO				
internalRuleID	-	Long	Yes	Internal identifier of the price derivation rule.
name	-	String	No	Name of price derivation rule.
description	-	String	No	Description of price derivation rule.
transactionControlBreakCode	-	RuleControlType	Yes	This is a special code that determines when, during the entry of a sale transaction, this price derivation rule is applied.
statusCode	-	String	Yes	Defines the current status for the price derivation rule.
typeCode	-	RuleType	Yes	Defines the type of price derivation rule.
ruleID	-	String	No	External identifier of the price derivation rule.
bonusPointsFlag	-	Boolean	Yes	Determines whether a discount or loyalty points are awarded by the price derivation rule.
roundingMethodCode	-	RoundingMethodCode	No	Determines how the calculated benefit is to be rounded.
decimalPlacesCount	-	Integer	No	Requested number of decimal places of the calculated benefit.
roundDestinationValue	-	Integer	No	Additional rounding information: • "1"/**null**: No additional treatment of the rounded result necessary. • "5": The rounded result is rounded again to the next respectively to the previous multiple of 5.

Name	Default Value	Data type, range	Manda tory	Description
discountMethodCode	-	String	Yes	Determines how the benefit influences the transaction. • "00"/null: The discount reduces the regular/discount total amount. • "01": The discount does not influence the regular/discount total amount or the amount the customer has to pay, but the customer gets a gift certificate about the discount which can be used for payment the next time. • "02": The discount is used as tender for the current transaction, i.e. it reduces the amount the customer still has to pay. • "03": The discount reduces the regular/discount total amount but it will be counterbalanced by a gift certificate sale. • "04": A coupon will be given to the customer instead of a discount. Regular/discount total amount will not be reduced. • "99": Rebate as manufacturer coupon related tender
giftCertificateExpirationDate	-	Date	No	End of the date range in which the gift certificate is valid.
tenderTypeCode	-	String	No	A code which uniquely identifies the type of tender, for example, cash, check, credit card, etc.
prohibitTransactionRelatedPromotionConditionFlag	false	Boolean	No	Determines whether applying this promotion price derivation rule influences the calculation base amount of subsequent transaction-related promotion price derivation rules (**false**) or not (**true**). It is relevant for line item-related monetary promotion price derivation rules only.

Name	Default Value	Data type, range	Mandatory	Description
chooseItemMethod	null	ChooseItemMethod	No	Determines in which order items are considered for the calculation.
noEffectOnSubsequentPromotionConditionFlag	-	Boolean	Yes	Determines whether a promotion price derivation rule influences the calculation base amount of subsequent promotion price derivation rules (**false**) or not (**true**).
calculationBase	-	CalculationBaseType	No	Only influencing the calculation base amount for transaction-related promotion price derivation rules. Defines how the calculation base amount is obtained on the basis of the sold and returned line items in the context of a transaction.
couponPrintoutID	-	String	No	Identifier of the printout coupon.
couponPrintoutRule	-	CouponPrintoutRule	No	The typecode of the printout rule.
couponPrintoutText	-	Object	No	Formatted text of the printout coupon.
considerPreviousPromotionConditionFlag	-	boolean	Yes	Controls whether the concept behind the parameter noEffectOnSubsequentPriceDerivationRulesFlag (**false**) or the calculationBaseSequence concept (**true**) is to be considered.
printoutValidityPeriod	-	BigDecimal	No	Validity period for printout coupons or gift certificates.
tid	-	Long	No	Translation identifier - used for the unique identification of translations in the common translation table.
externalConditionRuleID	-	String	No	External identifier of the promotion price derivation rule.

Name	Default Value	Data type, range	Mandatory	Description
calculationBaseSequence	-	Long	Yes	Determines which of the previously applied modifiers is to be considered as calculation base amount for the current modification. In detail, the calculation base amount for the current price derivation rule is to be determined as following: • In case that no price derivation rules were applied before, it is the regular sales price or regular total amount. • <= -2 / **null**: All price derivation rules are to be considered which were applied before the current price derivation rule, that is, the calculation base amount for the current price derivation rule equals to the discount sales price/discount total amount of the price derivation rule which was applied just before it – at latest. • Otherwise, the calculation base amount for the current price derivation rule equals to the discount sales price/discount total amount of that price derivation rule which was applied one or more steps before it, having the highest sequence <= calculationBaseSequence (or the regular sales price/regular total amount if no such price derivation rule was applied).
roundingRuleID	-	Long	No	The system-generated identifier of the rounding rule which applies for discounts/loyalty point counts resulting from applying the price derivation rule.

Name	Default Value	Data type, range	Mandatory	Description
shareRoundingRuleID	-	Long	No	The system-generated identifier of the rounding rule which applies for shares resulting from prorating transaction-related discounts or loyalty points resulting from applying the price derivation rule.
pointsAmountRoundingRuleID	-	Long	No	The system-generated identifier of the rounding rule applies for monetary equivalents to loyalty point counts resulting from applying the price derivation rule.
noPreviousMonetaryDiscountAllowedFlag	-	boolean	Yes	Indicates whether the price derivation rule is applicable on items with previously applied monetary discounts (**false**) or not (**true**).
increaseOfPriceAllowed	false	boolean	Yes	This flag decides whether a promotion price derivation rule or a particular price is allowed to increase the price in comparison to the latest calculated discount sales price.
PromotionConditionSO				
promotionID	-	Long	Yes	Identifier of a promotion.
conditionID	-	Long	Yes	Identifier of a promotion price derivation rule.
internalRuleID	-	Long	Yes	Identifier of a specific price derivation rule.
internalEligibilityID	-	Long	Yes	Identifier for a price derivation rule eligibility. It is the identifier of the root eligibility of the promotion price derivation rule.
timeGroup	-	PromotionConditionTimeGroupSO	No	Time restrictions.
typeCode	-	String	No	Code of the promotion price derivation rule type.
rule	-	PromotionConditionRuleSO	Yes	Price derivation rule data.
eligibility	-	PromotionConditionEligibilitySO	No	Price derivation rule eligibility data.

Name	Default Value	Data type, range	Mandatory	Description
sequence	-	Long	Yes	Sequence of the promotion for collision handling.
resolution	-	Long	Yes	Resolution of the promotion for collision handling.
notShowingFlag	-	Boolean	No	Determines whether the result of the applied promotion price derivation rule is to be suppressed on displays and on the receipt (**true**) or not (**false**). For example, line item-related loyalty points are not printed after each line item, but only summarized at the end of the receipt.
description	-	String	No	Description of the promotion.
receiptPrinterName	-	String	No	Transaction text (overwrites the transaction text of the promotion).
operatorDisplayName	-	String	No	Operator display text (overwrites the operator display text of the promotion).
customerDisplayName	-	String	No	Customer display text (overwrites the customer display text of the promotion).
itemDiscountControlVector	-	String	No	Vector to control which items can get that benefit.
saleReturnTypeCode	SALES_RETURNS	SaleReturnTypeCode	No	Determines whether the promotion price derivation rule can be used only for sales, only for returns, or for both.
amendmentTypeCode	SALES_AMENDMENTS	AmendmentTypeCode	No	Determines whether the promotion price derivation can be used only for sales, only for amendments, or for both.
exclusiveFlag	-	Boolean	Yes	Determines whether this promotion price derivation rule is an exclusive one (**true**) or not (**false**).
iconID	-	String	No	Identifier of the icon that should be displayed as sales information.

Name	Default Value	Data type, range	Mandatory	Description
concurrenceControlVector	-	String	No	Determines how this promotion price derivation rule works in relation to other applicable promotion price derivation rules.
tid	-	Long	No	Translation identifier – used for the unique identification of translations in the common translation table.
PromotionConditionTimeGroupSO				
internalTimeGroupID	-	Long	Yes	Identifier of a time group.
PromotionMhgFilterSO				
merchandiseHierarchyGroupID	-	String	Yes	Identifier of a merchandise category.
merchandiseHierarchyGroupIDQualifier	-	String	Yes	Qualifier of a merchandise category.
PromotionSO				
businessUnitGroupID	-	Long	No	Identifier of the business unit group.
promotionID	-	String	Yes	External identifier of the promotion.
effectiveDateTime	-	Timestamp	Yes	Time stamp when this price derivation rule eligibility starts to be effective.
expirationDateTime	-	Timestamp	Yes	Time stamp when this price derivation rule stops to be effective.
externalOfferID	-	String	No	External offer identifier
operatorDisplayName	-	String	No	Text to be displayed to the operator.
customerDisplayName	-	String	No	Text to be displayed to the customer.
receiptPrinterName	-	String	No	Text which is to be printed on the receipt.
origin	-	String	No	Origin of the promotion.
description	-	String	No	Description of the promotion.

Name	Default Value	Data type, range	Mandatory	Description
promotionTypeID	-	String	No	Identifier of the promotion type.
conditionList	-	List<? extends PromotionConditionSO>	No	List of promotion price derivation rules.
internalPromotionID	-	Long	Yes	Internal identifier of a promotion.
promotionMultiLanguageTexts	-	List<PromotionTextSO>	No	List with translations of texts.
PromotionTextSO				
description	-	String	No	Description of a promotion.
languageCode	-	String	No	Code of the corresponding language.
name	-	String	No	Name of the promotion.
receiptText	-	String	No	Text to be printed on the receipt.
customerDisplayText	-	String	No	Text to be displayed to the customer.
operatorDisplayText	-	String	No	Text to be displayed to the operator.
RebatePromotionConditionRuleSO extends PromotionConditionRuleSO				
priceModificationMethodCode	-	PriceModificationMethod	Yes	A code denoting the method of modifying the price that is being applied to the transaction.
priceModificationAmount	-	BigDecimal	No	The modification to be applied to the line item, expressed as a monetary discount.
priceModificationPercent	-	BigDecimal	No	The modification to be applied to the line item, expressed as a percentage of the regular/discount sales price.
newPriceAmount	-	BigDecimal	No	The reduction to be applied to the line item, expressed as the new price for the line item.
adjustmentMethodCode	DECREASE	AdjustmentMethodCode	No	A mnemonic code denoting what kind of adjustment is being made to the regular/discount sales price of the line item.
RoundingRuleDO				

Name	Default Value	Data type, range	Mandatory	Description
Key.businessUnitGroupID	-	Long	No	Identifier of the business unit group.
Key.roundingRuleID	-	Long	Yes	The system-generated identifier of the rounding rule.
roundMethodCode	-	RoundingMethod	Yes	Defines how the decimal number is to be rounded.
multiple	-	BigDecimal	Yes	The absolute difference between the calculated discount before rounding and after rounding. It can contain the digit 0 and either 1 or 5. Thereby, the digits 1 and 5 do appear at most once in the *multiple* whereas the digit 0 can appear more often.
ThresholdPromotionConditionEligibility extends PromotionConditionEligibilitySO				
thresholdTypeCode	-	ThresholdType	Yes	A code for the type of threshold which applies to a price derivation rule eligibility.
thresholdQuantity	-	BigDecimal	No	A quantity of a line item required to be purchased to trigger this price derivation rule eligibility.
thresholdAmount	-	BigDecimal	No	An amount of a line item required to be purchased to trigger this price derivation rule eligibility.
intervalQuantity	-	BigDecimal	No	An interval quantity for which this price derivation rule eligibility is triggered.
intervalAmount	-	BigDecimal	No	An interval amount for which this price derivation rule eligibility is triggered.
limitQuantity	-	BigDecimal	No	The maximum quantity of a line item for which this price derivation rule eligibility is triggered.
limitAmount	-	BigDecimal	No	The maximum amount of a line item for which this price derivation rule eligibility is triggered.
WorkerDiscountGroupPromotionConditionEligibilitySO extends PromotionConditionEligibilitySO				

Name	Default Value	Data type, range	Mandatory	Description
employeeDiscountGroupID	-	Long	No	Identifier for a specific employee discount group.

9.5 Objects and Attributes from the Transaction

Name	Default Value	Data type, range	Mandatory	Description
FrequentShopperPointsModifier				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
Key.frequentShopperPointsModifierSequenceNumber	-	Short	Yes	The sequence number for this FrequentShopperPointsModifier allowing more than one loyalty points' modification to occur on each line item.
prorateFrom	-	Short	No	The sequence number of the LoyaltyRewardLineItem. Only filled for loyalty point shares of transaction-related loyalty points.
priceDerivationRuleEligibilityID	-	Long	Yes	Identifier of the root price derivation rule eligibility.
priceDerivationRuleID	-	Long	Yes	Identifier of the price derivation rule.
computationBaseAmount	0.0	BigDecimal	No	Calculation base amount for the computation of the loyalty points.

Name	Default Value	Data type, range	Mandatory	Description
externalSystemOriginatorFlag	false	Boolean	No	Determines whether this entry was created by an external system. If **true**, it must not be changed, but other price modifications with higher sequence may be applied.
eligibilityTypeCode	-	String	Yes	Typecode of the root promotion price derivation rule eligibility.
frequentShopperPointsEarnedCount	0.0	BigDecimal	No	The number of loyalty points earned.
frequentShopperPointsEarnedAmount	0.0	BigDecimal	No	The money equivalent (expressed in local currency) to the number of loyalty points earned.
appliedQuantity	0.0	BigDecimal	Yes	The quantity the loyalty points' modifier applies to.
pointsPercentage	-	BigDecimal	No	The loyalty points price derivation rule to be applied, expressed as a percentage of the calculation base amount.
promotionID	-	Long	No	Identifier of a promotion.
triggerSequenceNumber	-	Short	No	The identifier of the used manual trigger, because one line item can have more than one manual triggers.
LoyaltyRewardLineItem				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
promotionID	-	Long	Yes	Identifier of a promotion.
priceDerivationRuleID	-	Long	No	Identifier of a price derivation rule.
priceDerivationRuleEligibilityID	-	Long	No	Identifier of the root price derivation rule eligibility.

Name	Default Value	Data type, range	Mandatory	Description
computationBaseAmount	0.0	BigDecimal	No	Calculation base amount for the computation of the loyalty points.
externalSystemOriginatorFlag	false	Boolean	No	Determines whether this entry was created by an external system. If **true**, it must not be changed, but other price modifications with higher sequence may be applied.
couponNumber	-	String	No	Identifier of the coupon the customer gets.
couponAmount	0.0	BigDecimal	No	The discount of the coupon the customer gets.
giftCertificateFaceValueAmount	0.0	BigDecimal	No	The discount printed or embossed on the gift certificate.
pointsPercentage	-	BigDecimal	No	The loyalty points price derivation rule to be applied, expressed as a percentage of the calculation base amount.
pointsAwardedCount	-	BigDecimal	No	The number of loyalty points earned.
loyaltyRewardTypeCode	-	String	No	Describes the kind of loyalty reward. Possible values include loyalty points, coupons, and gift certificates.
frequentShopperPointsEarnedAmount	0.0	BigDecimal	No	The money equivalent (expressed in local currency) to the number of loyalty points earned.
triggerSequenceNumber	-	Short	No	The identifier of the used manual trigger, because the transaction can include more than one manual triggers.
ManualPromotionTrigger				
triggerType	-	String	Yes	Reference to the Clients. Arbitrary value.
triggerValue	-	String	No	Reference to the Clients. Arbitrary value.
privilegeType	-	String	Yes	Defines the PriceModificationMethodTx.

Name	Default Value	Data type, range	Mandatory	Description
privilegeValue	-	BigDecimal	No	Reduction amount, reduction percent, or new price amount.
reasonCode	-	String	No	Reason code for the manual discount.
reasonDescription	-	String	No	A narrative description describing the manual discount reason.
reference	-	String	No	An additional information concerning the trigger.
triggerSequenceAddend	-	Long	No	A value which is to be added to the sequence of the promotion price derivation rule (which comes from the promotion master data).
PriceModificationLineItem				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
priceDerivationRuleID	-	Long	No	Identifier of a price derivation rule.
priceDerivationRuleEligibilityID	-	Long	No	Identifier of the root price derivation rule eligibility.
externalSystemOriginatorFlag	false	Boolean	No	Determines whether this entry was created by an external system. If **true**, it must not be changed, but other price modifications with higher sequence may be applied.
proRatedFlag	false	Boolean	No	Indicates whether this price modification was prorated across all of the line items in the RetailTransaction. Always true.
roundingAmount	-	BigDecimal	No	The discount that the price modification was adjusted by (after calculation from the percentage).

Name	Default Value	Data type, range	Mandatory	Description
calculationBaseAmount	0.0	BigDecimal	No	Calculation base amount for the computation of the discounts.
promotionID	-	Long	No	Identifier of a promotion.
amount	0.0	BigDecimal	No	The monetary value of the price modification that was computed.
extendedAmountBeforeModification	0.0	BigDecimal	No	The regular/discount total amount before applying the current price modification.
extendedAmountAfterModification	0.0	BigDecimal	No	The regular/discount total amount after applying the current price modification.
extraAmount	0.0	BigDecimal	No	A discount which results from applying a price modification that does not affect the regular/discount total amount.
percentage	-	BigDecimal	No	The percentage value in case of percentage price modifiers.
triggerSequenceNumber	-	Short	No	The identifier of the used manual trigger, because the transaction can include more than one manual triggers.
RetailPriceModifier				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
Key.retailPriceModifierSequenceNumber	-	Short	Yes	The sequence number for this RetailPriceModifier allowing more than one price modification to occur on each line item.
priceDerivationRuleID	-	Long	No	Identifier of a price derivation rule.
priceDerivationRuleEligibilityID	-	Long	No	Identifier of the root price derivation rule eligibility.

Name	Default Value	Data type, range	Mandatory	Description
extraAmount	0.0	BigDecimal	No	A discount which results from applying a price modification that does not affect the regular/discount sales price.
roundingAmount	0.0	BigDecimal	No	The monetary amount that the price modifier was adjusted by the rounding.
externalSystemOriginatorFlag	false	Boolean	No	Determines whether this entry was created by an external system. If **true**, it must not be changed, but other price modifications with higher sequence may be applied.
prorateFrom	-	Short	No	The sequence number of the PriceModificationLineItem. Only filled for discount shares of transaction-related discounts.
calculationBaseAmount	0.0	BigDecimal	No	Calculation base amount for the computation of the discounts.
reasonCode	-	String	No	The reason code for manual price overrides.
previousPrice	-	BigDecimal	No	The regular sales unit price that was valid before manual price override.
percent	0.0	BigDecimal	No	The percentage value in case of percentage price modifiers
amount	0.0	BigDecimal	No	The discount of this RetailPriceModifier.
calculationMethodCode	-	String	No	A typecode used to identity manual price overrides.
adjustmentMethodCode	-	String	No	A mnemonic code denoting what kind of adjustment is being made to the regular/discount sales price of the line item (INCREASE, DECREASE).

Name	Default Value	Data type, range	Mandatory	Description
newPrice	-	BigDecimal	No	Discount sales price of a line item. This attribute is only filled in case of the price modification method is: • FIXED_PRICE/"PS": defining the discount sales unit pricet • FIX_PRICE_TOTAL/"PT": defining the discount sales price of the summed up sales prices • SET_PRICE_TOTAL/"ST": see Package an Offer
eligibilityTypeCode	-	String	No	The typecode of the root price derivation rule eligibility.
extendedAmountBeforeModification	0.0	BigDecimal	No	The regular/discount sales price before applying the current price modification.
extendedAmountAfterModification	0.0	BigDecimal	No	The regular/discount sales price after applying the current price modification.
appliedQuantity	0.0	BigDecimal	Yes	The quantity the price modifier applies to.
promotionID	-	Long	No	Identifier of a promotion.
triggerSequenceNumber	-	Short	No	The identifier of the used manual trigger, because one line item can have more than one manual triggers.
reasonDescription	-	String	No	A narrative description describing the manual discount reason.
promotionSequence	-	Long	No	Sequence of a promotion price derivation rule for internal usage.
RetailTransaction				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.

Name	Default Value	Data type, range	Mandatory	Description
retailTransactionCouponSummaryList	-	List<RetailTransactionCouponSummary>	No	List of RetailTransactionCouponSummary.
retailTransactionPromotionPriceDerivationRuleList	-	List<RetailTransactionPromotionPriceDerivationRule>	No	List of RetailTransactionPromotionPriceDerivationRule.
promotionManualTriggerList	-	List<RetailTransactionPromotionTrigger>	No	List of RetailTransactionPromotionTrigger.
retailTransactionLineItemList	-	List<RetailTransactionLineItem>	No	List of RetailTransactionLineItem.
retailTransactionTotalList	-	List<RetailTransactionTotal>	No	List of RetailTransactionTotal.
retailTransactionCustomerList	-	List<RetailTransactionCustomer>	No	List of RetailTransactionCustomer.
retailTransactionModifierCouponList	-	List<RetailTransactionModifierCoupon>	No	List of RetailTransactionModifierCoupon.
retailTransactionExternalTriggerList	-	List<RetailTransactionExternalTrigger>	No	List of RetailTransactionExternalTrigger.
retailTransactionManufacturerCouponSummaryList	-	List<RetailTransactionManufacturerCouponSummary>	No	List of RetailTransactionManufacturerCouponSummary.
promotionTimestamp	-	Timestamp	No	The time stamp used for promotion validity checks and calculation.
RetailTransactionCouponSummary				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.couponNumber	-	String	Yes	Identifier of a coupon.
appliedCount	-	BigDecimal	No	Number of used coupons.
inputCount	0.0	BigDecimal	No	Count of registered coupons.

Name	Default Value	Data type, range	Mandatory	Description
privilegeType	-	String	No	Determines how the regular/discount sales price (resp. regular/discount total amount) is to be calculated in case that the PrivilegeValue is given.
privilegeValue	-	BigDecimal	No	Reduction amount, reduction percent or new price amount – depending on the chosen PrivilegeType.
RetailTransactionCustomer				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.customerID	-	String	Yes	Identifier of a customer.
Key.addressTypeCode	-	String	Yes	The customer typecode.
firstName	-	String	No	A person's first name.
lastName	-	String	No	A person's surname.
genderType	-	String	No	A code for specifying a person's gender.
birthDayNumber	-	Integer	No	A number in the range 1-31 denoting the day of the month part of the person's date of birth.
birthMonthNumber	-	Integer	No	A number in the range 1-12 denoting the month part of the person's date of birth.
birthYearNumber	-	Integer	No	The year part of the person's date of birth.
addressTypeDescription	-	String	No	The description of the customer typecode.
retailTransactionCustomerGroupAssignmentList	-	List<RetailTransactionCustomerGroupAssignment>	No	List of customer groups.
RetailTransactionCustomerGroupAssignment				

Name	Default Value	Data type, range	Mandatory	Description
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.customerID	-	String	Yes	Identifier of a customer.
Key.addressTypeCode	"00"	String	Yes	The customer typecode.
Key.customerGroupID	-	String	Yes	Identifier of a customer group.
RetailTransactionExternalTrigger				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.triggerTypeCode	-	String	Yes	The type of the TriggerValue.
triggerAmount	-	BigDecimal	Yes	The value (amount). Its type is determined by TriggerTypeCode.
RetailTransactionLineItem				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
saleReturnLineItem	-	SaleReturnLineItem	No	Item.
priceModificationLineItemList	-	List<PriceModificationLineItem>	No	List of transaction-related discounts.
loyaltyRewardLineItemList	-	List<LoyaltyRewardLineItem>	No	List of transaction-related loyalty points.
retailTransactionLineItemAssociationList	-	List<RetailTransactionLineItemAssociation>	No	List of associative entities recording relationships between line items within the same RetailTransaction.

Name	Default Value	Data type, range	Mandatory	Description
retailTransactionLineItemTypeCode	-	String	Yes	A code to denote the type of retail transaction line item, such as sale/return, void, tender.
voidFlag	false	Boolean	No	A boolean indicator that tells if this line item is voided (**true**) or not (**false**).
beginDateTimestamp	-	Timestamp	No	The start time of the RetailTransactionLineItem.
entryMethodCode	-	String	No	A retailer assigned code to denote how the RetailTransactionLineItem was entered at the workstation.
workstationID	-	String	No	Identifier of the workstation where the line item was created.
workstationTypeCode	-	String	No	Type of the workstation where the line item was created.
keyedOfflineCode	-	String	No	A code that indicates the online/offline state when the transaction was completed.
RetailTransactionLineItemAssociation				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.lineItemSequenceNumber	-	Short	Yes	The sequence number of line item.
Key.toLineItemSequenceNumber	-	Short	Yes	The sequence number of the linked line item.
Key.lineItemAssociationTypeCode	-	String	Yes	A retailer assigned code denoting the relationship between the two items.
Key.toBusinessUnitGroupID	-	Long	Yes	Identifier for the business unit group of the linked transaction.
toTransactionID	-	String	No	The transaction identifier of the linked transaction.
RetailTransactionManufacturerCouponSummary				

Name	Default Value	Data type, range	Mandatory	Description
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.manufacturerCouponScanCode	-	String	Yes	The manufacturer coupon barcode.
appliedCount	-	BigDecimal	No	The number how often the manufacturer coupon was applied.
appliedIntervalCount	-	BigDecimal	No	The number of intervals for which the manufacturer coupon was applied.
inputCount	-	BigDecimal	No	The number how often the manufacturer coupon was registered.
keyEnteredFlag	false	Boolean	Yes	A flag to denote whether the manufacturer coupon was key entered or scanned.
manufacturerCompanyCode	-	String	No	The 6 to 12 digit GS1 Company Prefix of the manufacturer that is funding this offer. This information is derived from the ManufacturerCouponScanCode in case that the manufacturer coupon was scanned, or manually entered otherwise.
manufacturerOfferCode	-	String	No	A 6-digit number (assigned by the holder of the ManufacturerCompanyCode) that identifies this offer. This information is derived from the ManufacturerCouponScanCode in case that the manufacturer coupon was scanned, or manually entered otherwise.
saleLineItemValidationList	-	List<RetailTransactionManufacturerCouponValidation>	No	List of sale line item references where the promotion provided by this coupon was applied.
tenderLineItemSequenceNumber	-	Short	No	The reference to the corresponding tender line item.
totalAmount	-	BigDecimal	No	The total discount amount that is granted by applying the manufacturer coupon according to its AppliedCount.

Name	Default Value	Data type, range	Mandatory	Description
RetailTransactionManufacturerCouponValidation				
Key.businessUnitGroupID	-	Long	Yes	A unique system assigned identifier for a group of BusinessUnits.
Key.transactionID	-	String	Yes	A universally unique identifier (UUID) for the Transaction. This may be assembled from alternate key members.
Key.manufacturerCouponScanCode	-	String	Yes	The manufacturer coupon barcode.
Key.saleLineItemSequenceNumber	-	Short	Yes	Reference to the sale line item the manufacturer coupon was applied to
RetailTransactionModifierCoupon				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.promotionID	-	Long	Yes	Identifier of a promotion.
Key.priceDerivationRuleID	-	Long	Yes	Identifier of a price derivation rule.
Key.priceDerivationRuleEligibilityID	-	Long	Yes	Identifier of the root price derivation rule eligibility.
Key.couponSequenceNumber	-	Short	Yes	The sequence number of a coupon entry (starting with 1), in case there was more than one coupon registered for one price modifier.
couponNumber	-	String	No	Identifier of a coupon.
RetailTransactionPromotionExternalActionParameter				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.promotionID	-	Long	Yes	Identifier of a promotion.

Name	Default Value	Data type, range	Mandatory	Description
Key.priceDerivationRuleID	-	Long	Yes	Identifier of a price derivation rule.
Key.priceDerivationRuleEligibilityID	-	Long	Yes	Identifier of the root price derivation rule eligibility.
Key.parameterID	-	String	Yes	Identifier of an external action parameter.
parameterValue	-	String	Yes	The value of the parameter.
RetailTransactionPromotionExternalActionText				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.promotionID	-	Long	Yes	Identifier of a promotion.
Key.priceDerivationRuleID	-	Long	Yes	Identifier of a price derivation rule.
Key.priceDerivationRuleEligibilityID	-	Long	Yes	Identifier of the root price derivation rule eligibility.
Key.textID	-	String	Yes	Identifier of an external action text.
text	-	String	Yes	Text for an external action.
language	-	String	No	Language of the external action text.
RetailTransactionPromotionPriceDerivationRule				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.promotionID	-	Long	Yes	Identifier of a promotion.
Key.priceDerivationRuleID	-	Long	Yes	Identifier of a price derivation rule.
Key.priceDerivationRuleEligibilityID	-	Long	Yes	Identifier of the root price derivation rule eligibility.
promotionDescription	-	String	No	Description of a promotion.

Name	Default Value	Data type, range	Mandatory	Description
receiptPrinterName	-	String	No	Text to be printed on the receipt.
promotionPriceDerivationRuleSequence	-	Long	No	Sequence - for collision handling.
couponPrintoutText	-	Object	No	The formatted text of the printout coupon.
promotionPriceDerivationRuleResolution	-	Long	No	Resolution - for collision handling.
promotionPriceDerivationRuleTypeCode	-	String	No	The typecode of the promotion price derivation rule.
transactionControlBreakCode	-	String	No	This typecode defines the base for applying the promotion price derivation rule and it defines the calculation time.
priceModificationMethodCode	-	String	No	A code denoting the method of modifying the price that is being applied to the transaction resp. line item (see PriceModificationMethodTx).
priceDerivationRuleDescription	-	String	No	Business description for this price derivation rule.
promotionOriginatorTypeCode	-	String	No	The typecode of the originator of the promotion.
externalPromotionID	-	String	No	The external identifier of the promotion.
externalPriceDerivationRuleID	-	String	No	The external identifier of the price derivation rule.
externalOfferID	-	String	No	The external offer identifier of the promotion.
triggerQuantity	0.0	BigDecimal	No	The quantity of items fulfilling the price derivation rule eligibility which is required to be purchased to trigger it. Is only filled for N for M discount (PriceDerivationRule.TypeCode = “GP”).
giftCertificateExpirationDate	-	Timestamp	No	End of the date range in which the gift certificate is valid.

Name	Default Value	Data type, range	Mandatory	Description
discountMethodCode	-	String	No	Determines how the discount influences the transaction. • "00"/null: The discount reduces the regular/discount total amount. • "01": The discount does not influence the regular/discount total amount or the amount the customer has to pay, but the customer gets a gift certificate about the discount which can be used for payment the next time. • "02": The discount is used as tender for the current transaction, that is, it reduces the amount the customer still has to pay. • "03": The discount reduces the regular/discount total amount but it will be counterbalanced by a gift certificate sale. • "04": A coupon will be given to the customer instead of a discount. Regular/discount total amount will not be reduced. • "99": Rebate as manufacturer coupon related tender
frequentShopperPointsFlag	false	Boolean	No	Determines whether a discount or loyalty points are awarded by the price derivation rule (see bonusPointsFlag).
customerGroupLoyaltyPointsDefaultQuantity	0.0	BigDecimal	No	The count of default loyalty points assigned to the customer group.

Name	Default Value	Data type, range	Mandatory	Description
prohibitPrintFlag	false	Boolean	No	Determines whether the result of the applied promotion price derivation rule is to be suppressed on displays and on the receipt (**true**) or not (**false**). For example, line item-related loyalty points are not printed after each line item, but only summarized at the end of the receipt (see notShowingFlag).
tenderTypeCode	-	String	No	A code which uniquely identifies the tender in case of discount as tender.
promotionTypeName	-	String	No	Name of promotion type.
calculationBase	-	String	No	Only influences the calculation base amount for transaction-related promotion price derivation rules. Defines how the calculation base amount is obtained on the basis of the sold and returned line items in the context of a transaction.
pointsConversionAmount	0.0	BigDecimal	No	Exchange rate for the conversion from loyalty points into local currency.
noEffectOnSubsequentPriceDerivationRulesFlag	false	Boolean	No	Determines whether applying this price derivation rule influences the calculation base amount of subsequent price derivation rules (**false**) or not (**true**).
prohibitTransactionRelatedPriceDerivationRulesFlag	false	Boolean	No	Determines whether applying this price derivation rule influences the calculation base amount of subsequent transaction-related price derivation rules (**false**) or not (**true**). It is relevant for line item-related monetary promotion price derivation rules only.
couponPrintoutID	-	String	No	The identifier of the printout coupon.
couponPrintoutRule	-	String	No	The typecode of the printout rule.

Name	Default Value	Data type, range	Mandatory	Description
exclusiveFlag	false	Boolean	No	Determines whether this price derivation rule is an exclusive one (**true**) or not (**false**).
requireUserInteractionFlag	false	Boolean	No	Determines whether an additional user interaction (pressing a button) is necessary in order to fulfill the eligibility for applying the price derivation rule (**true**) or not (**false**).
considerPreviousPriceDerivationRulesFlag	false	Boolean	No	Controls whether the concept behind the parameter noEffectOnSubsequentPriceDerivationRulesFlag (**false**) or the calculationBaseSequence concept (**true**) is to be considered.
concurrenceControlVector	-	String	No	Determines how this promotion price derivation rule works in relation to other applicable promotion price derivation rules.
appliedCount	0.0	BigDecimal	No	Describes how often the current price derivation rule was applied.
printoutValidityPeriod	0.0	BigDecimal	No	Describes how long (how many days) the printed coupon/gift certificate is valid.
externalActionID	-	String	No	The identifier of the action which is to be performed by the Client application.
externalActionDescription	-	String	No	The description of the action which is to be performed by the Client application.
externalActionTextList	-	List<RetailTransactionPromotionExternalActionText>	No	The texts for the action which is to be performed by the Client application.
externalActionParameterList	-	List<RetailTransactionPromotionExternalActionParameter>	No	The parameters for the action which is to be performed by the Client application.
promotionMultiLanguageTexts	-	List<PromotionTextSO>	No	List of translated texts.

Name	Default Value	Data type, range	Mandatory	Description
calculationBaseSequence	-	Long	No	Determines which one of the previously applied modifiers is to be considered as calculation base amount for the current modification. In detail, the calculation base amount for the current price derivation rule is to be determined as following: • In case that no price derivation rules were applied before, it is the regular sales price or regular total amount. • <= -2 / **null**: All price derivation rules are to be considered which were applied before the current price derivation rule, that is, the calculation base amount for the current price derivation rule equals to the discount sales price/discount total amount of the price derivation rule which was applied just before it – at latest. • Otherwise, the calculation base amount for the current price derivation rule equals to the discount sales price/discount total amount of that price derivation rule which was applied one or more steps before it, having the highest sequence <= calculationBaseSequence (or the regular sales price/regular total amount if no such price derivation rule was applied).
additionalPriceTypeCode	-	AdditionalPriceTypeCode	No	This attribute contains the price type of the price type eligiblity.
RetailTransactionPromotionTrigger extends ManualPromotionTrigger				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.

Name	Default Value	Data type, range	Mandatory	Description
Key.triggerSequenceNumber	-	Short	Yes	The identifier of the trigger because a transaction can have more than one trigger.
RetailTransactionTotal				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.transactionTotalTypeCode	-	String	Yes	A unique assigned mnemonic identifier that identifies the TransactionTotalType.
amount	-	BigDecimal	No	The monetary value of the effective total amount.
ReturnLineItem				
SaleReturnLineItem				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
frequentShopperPointsModifierList	-	List<FrequentShopperPointsModifier>	No	List of FrequentShopperPointsModifier.
saleReturnLineItemModifierReferenceList	-	List<SaleReturnLineItemModifierReference>	No	List of SaleReturnLineItemModifierReference.
retailPriceModifierList	-	List<RetailPriceModifier>	No	List of RetailPriceModifier.
saleReturnLineItemMerchandiseHierarchyGroupList	-	List<SaleReturnLineItemMerchandiseHierarchyGroup>	No	List of SaleReturnLineItemMerchandiseHierarchyGroup.
saleReturnLineItemSalesOrder	-	SaleReturnLineItemSalesOrder	No	Information about a customer order related to this line item.
saleReturnLineItemPromotionTriggerList	-	List<SaleReturnLineItemPromotionTrigger>	No	List of SaleReturnLineItemPromotionTrigger.

Name	Default Value	Data type, range	Mandatory	Description
saleReturnLineItemModifierCouponList	-	List<SaleReturnLineItemModifierCoupon>	No	List of SaleReturnLineItemModifierCoupon.
itemID	-	String	No	Identifier of an item.
quantity	0	Integer	Yes	Quantity of a line item.
extendedAmount	0.0	BigDecimal	No	The product of the quantity times the discount sales price of an item.
extendedDiscountAmount	-	BigDecimal	No	The monetary discount of all line item-related discounts that were applied to this line item.
actionCode	-	String	Yes	A code denoting the sign of the line item.
itemType	-	String	Yes	The typecode of the line item.
units	-	BigDecimal	No	The number of units sold.
notConsideredByLoyaltyEngineFlag	false	Boolean	No	Determines whether the PCE should care about the line item as a trigger (**false**) or not (**true**).
priceTypeCode	-	String	No	The price typecode.
discountFlag	false	Boolean	No	A flag to indicate whether this line item can be discounted (**true**) or not (**false**).
grandExtendedAmount	0.0	BigDecimal	No	The line item total including taxes and discounts.
discountTypeCode	-	String	Yes	A code to control which items can get that benefit.
frequentShopperPointsEligibilityFlag	false	Boolean	No	A flag to denote that the line item is eligible for loyalty points (**true**) or not (**false**).
quantityInputMethod	-	String	No	Determines how the input of quantity was done on the Client application (see QuantityInputMethod).

Name	Default Value	Data type, range	Mandatory	Description
mainMerchandiseHierarchyGroupIDQualifier	-	String	No	A qualifier which is used to differ between merchandise categories belonging to different functions but having the same identifier.
mainMerchandiseHierarchyGroupID	-	String	No	Unique system assigned identifier for the merchandise category.
unitOfMeasureCode	-	String	Yes	The code used to specify the unit of measure for the item.
regularUnitPrice	0.0	BigDecimal	No	The regular sales unit price of the line item.
actualUnitPrice	-	BigDecimal	Yes	The current discount sales unit price of the line item.
isReturnLineItem	false	boolean	Yes	Indicates if the saleReturnLineItem is returned (true) or sold (false). The field if filled by the pricing engine based on the saleReturnLineItem's actionCode.
companyCode	-	String	No	The company code of the item's manufacturer.
familyCode	-	String	No	The family code of the item assigned by the retailer.
saleReturnLineItemAdditionalPriceList	-	List<SaleReturnLineItemAdditionalPrice>	No	The list of additional prices
SaleReturnLineItemAdditionalPrice				
Key.businessUnitGroupID	-	Long	Yes	A unique system assigned identifier f or a group of BusinessUnits.
Key.transactionID	-	String	Yes	The transaction ID.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
Key.priceTypeCode	-	AdditionalPriceTypeCode	Yes	The price type code.
priceAmount	-	BigDecimal	Yes	Additional price amount of the item

Name	Default Value	Data type, range	Mandatory	Description
SaleReturnLineItemMerchandiseHierarchyGroup				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
Key.merchandiseHierarchyGroupID	-	String	Yes	Identifier of a merchandise category.
Key.merchandiseHierarchyGroupIDQualifier	-	String	Yes	Qualifier of a merchandise category.
timeStampEffective	-	Timestamp	No	The time stamp when this assignment becomes active.
timeStampExpiration	-	Timestamp	No	The timestamp when this assignment becomes inactive.
statusCode	-	String	No	Defines the current status for the assignment. Valid codes include: • AC/**null**: Active. • IA: Inactive.
SaleReturnLineItemModifierCoupon				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
Key.modifierSequenceNumber	-	Short	Yes	The sequence number of the related modifiers.
Key.promotionID	-	Long	Yes	Identifier of a promotion.
Key.priceDerivationRuleID	-	Long	Yes	Identifier of a price derivation rule.
Key.priceDerivationRuleEligibilityID	-	Long	Yes	Identifier of the root price derivation rule eligibility.

Name	Default Value	Data type, range	Mandatory	Description
Key.couponSequenceNumber	-	Short	Yes	The sequence number of the coupon entry (starting with 1), in case there was more than one coupon registered for one modifier.
couponNumber	-	String	Yes	Identifier of a coupon.
SaleReturnLineItemModifierReference				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.promotionID	-	Long	Yes	Identifier of a promotion.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
Key.priceDerivationRuleID	-	Long	Yes	Identifier of a price derivation rule.
Key.priceDerivationRuleEligibilityID	-	Long	Yes	Identifier of the root price derivation rule eligibility.
referenceQuantity	0.0	BigDecimal	Yes	The quantity share of that line item which was used for fulfilling the price derivation rule eligibilities.
SaleReturnLineItemPromotionTrigger extends ManualPromotionTrigger				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
Key.triggerSequenceNumber	-	Short	Yes	The identifier of the trigger, because one line item can have more than one trigger.
SaleReturnLineItemSalesOrder				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.

Name	Default Value	Data type, range	Mandatory	Description
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.retailTransactionLineItemSequenceNumber	-	Short	Yes	The sequence number of line item within the context of this RetailTransaction.
requestedDeliveryDate	-	Timestamp	No	The delivery date that was requested by the customer.
salesOrderTypeCode	-	String	No	The typecode of the related sales order.
salesOrderDeliveryTypeCode	-	String	Yes	The delivery typecode of the related sales order.
Transaction				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
retailTransaction	-	RetailTransaction	Yes	Retrieve the related RetailTransaction.
transactionTypeCode	-	String	No	A code to denote the type of transaction.
transactionCategoryList	-	List<TransactionCategory>	No	List of transaction categories.
beginDateTimestamp	-	Timestamp	No	The time and date a transaction is initiated.
workstationID	-	String	No	Identifier of the workstation where the transaction was created.
TransactionCategory				
Key.businessUnitGroupID	-	Long	Yes	Identifier of the business unit group.
Key.transactionID	-	String	Yes	Identifier of the transaction.
Key.transactionCategoryCode	-	String	Yes	The code representing the transaction category.

10 Appendix

10.1 Performance Comparison Between the Brute Force and the Greedy Algorithm

Various experiments are performed to compare the performance and accuracy (generated discounts) of the brute force (short BF) and greedy algorithms (short greedy) in the scope of the best price calculation. The experiments are based on three variables: the number of line items in the transaction, the quantity of each line item and the number of colliding promotion price derivation rules. There are three sets of experiments. In each set of experiments, the values of two of the above variables are set to a constant value and the value of the remaining variable is changed.

Each experiment is repeated ten times. In each experiment, the price of each line item in the transaction is 100€ and all the colliding promotion price derivation rules have price modification percent of value 2. Calculation time limit in each experiment is the time limit for the execution of the brute force or greedy implementation (not for the whole PCE). When the execution of a brute force or greedy implementation for a set of colliding promotion price derivation rules exceeds this time limit, the execution of the brute force or greedy implementation is stopped by the PCE before the PCE continues its normal execution. The results of each set of experiments are described next.

10.1.1 Varying the Number of Line Items in the Transaction

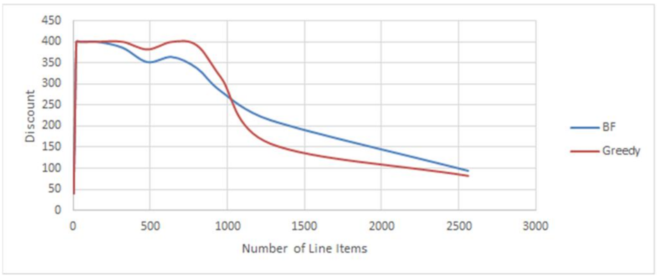
In the first set of experiments, the quantity of each line item is set to 10 and the number of colliding promotion price derivation rules is set to 20. The number of line items in the transaction is varied. There are two subsets of experiments in this scope. Across the two subsets of the experiments, only the value for the calculation time limit differs. The results of the first subset of experiments are shown below:

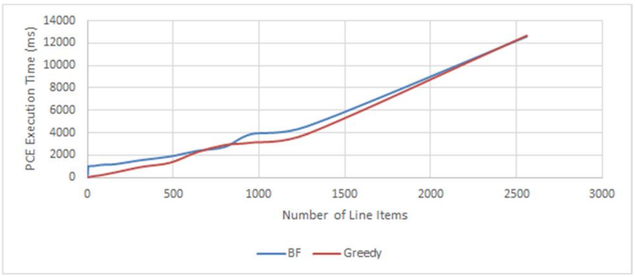
CALCULATION_TIME_LIMIT = 1000 ms AND LINE_ITEM_QUANTITY = 10 AND PROMOTION_COUNT = 20

NUMBER_OF_LINE_ITEMS	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
2	40	40	240	36
5	100	100	1011	46
10	200	200	1014	56
20	400	400	1028	85
40	400	400	1041	130
80	400	400	1129	213

NUMBER_OF_LINE_ITEMS	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
160	400	400	1192	453
320	386	400	1571	960
480	352	382	1873	1315
640	364	400	2387	2266
800	338	392	2730	2905
960	282	314	3903	3126
1280	214	158	4575	3891
2560	94	82	12592	12676

The graphical representations of these data are shown in the two figures below:





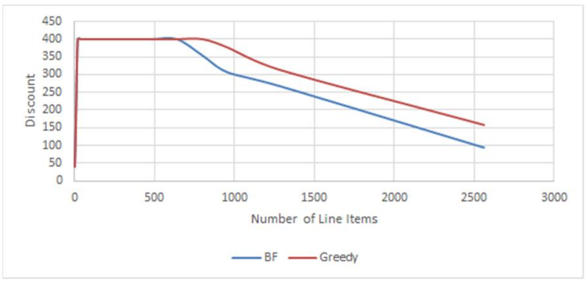
The results of the second subset of experiments are shown below:

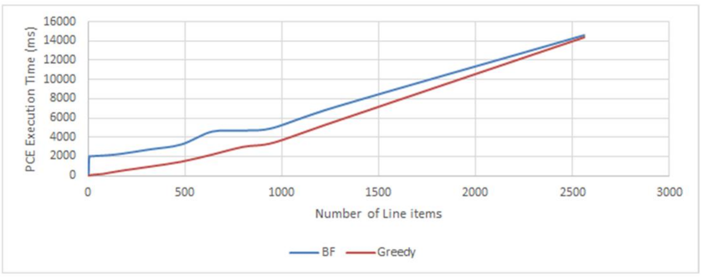
CALCULATION_TIME_LIMIT = 2000 ms AND LINE_ITEM_QUANTITY = 10 AND PROMOTION_COUNT = 20

NUMBER_OF_LINE_ITEMS	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
2	40	40	234	36
5	100	100	2011	45
10	200	200	2014	58
20	400	400	2027	86
40	400	400	2059	128
80	400	400	2109	208
160	400	400	2256	487

NUMBER_OF_LINE_ITEMS	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
320	400	400	2742	957
480	400	400	3247	1458
640	400	400	4590	2200
800	354	400	4694	3002
960	306	376	5003	3458
1280	268	314	7174	5676
2560	94	158	14605	14385

The graphical representations of these data are shown in the two figures below:





10.1.2 Varying the Quantity of a Line Item

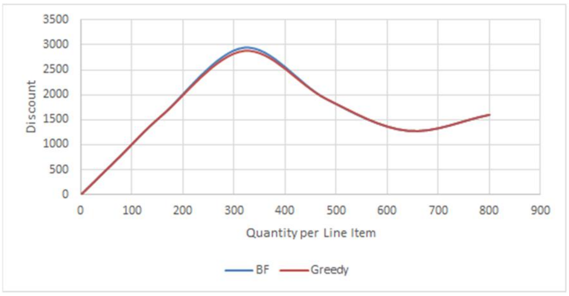
In the second set of experiments, the number of line items is set to 5 and the number of colliding promotion price derivation rules is set to 20. The quantity for each line item is varied. There are two subsets of experiments in this scope. Across the two subsets of the experiments, only the value for the calculation time limit differs. The results of the first subset of experiments are shown below:

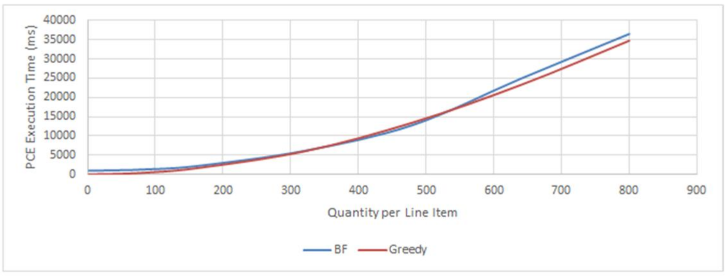
CALCULATION_TIME_LIMIT = 1000 ms AND NUMBER_OF_LINE_ITEMS = 5 AND PROMOTION_COUNT = 20

LINE_ITEM_QUANTITY	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
2	20	20	1005	29
5	50	50	1007	33
10	100	100	1011	45
20	200	200	1025	73
40	400	400	1078	146
80	800	800	1290	425

LINE_ITEM_QUANTITY	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
160	1600	1600	2186	1627
320	2944	2880	6140	6013
480	1920	1920	12833	13482
640	1280	1280	24897	23256
800	1600	1600	36490	34753

The graphical representations of these data are shown in the 2 figures below:





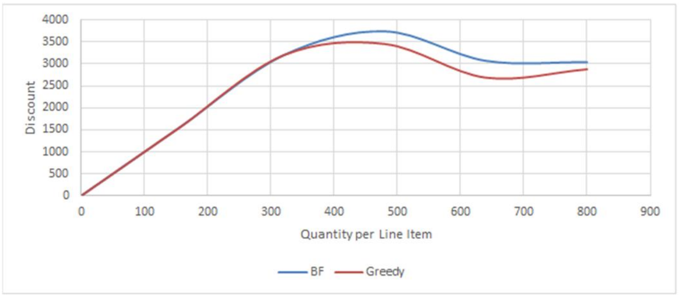
The results of the second subset of experiments are shown below:

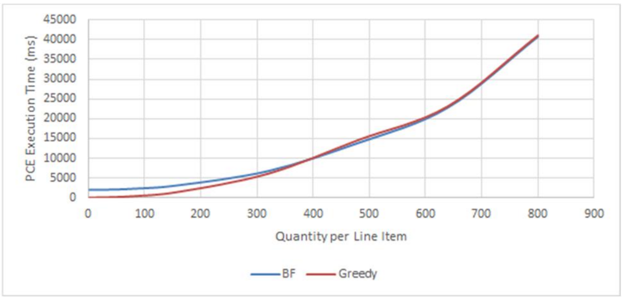
CALCULATION_TIME_LIMIT = 2000 ms AND NUMBER_OF_LINE_ITEMS = 5 AND PROMOTION_COUNT = 20

LINE_ITEM_QUANTITY	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
2	20	20	2006	25
5	50	50	2009	33
10	100	100	2010	46
20	200	200	2028	78
40	400	400	2079	150
80	800	800	2288	404
160	1600	1600	3162	1497
320	3200	3200	6779	6109
480	3744	3456	13879	14608

LINE_ITEM_QUANTITY	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
640	3072	2688	22843	23173
800	3040	2880	40806	41088

The graphical representations of these data are shown in the 2 figures below:





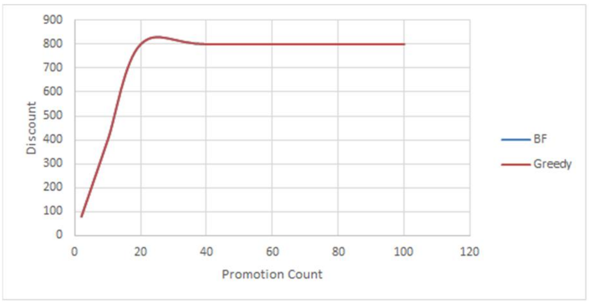
10.1.3 Varying the Number of Colliding Promotion Price Derivation Rules

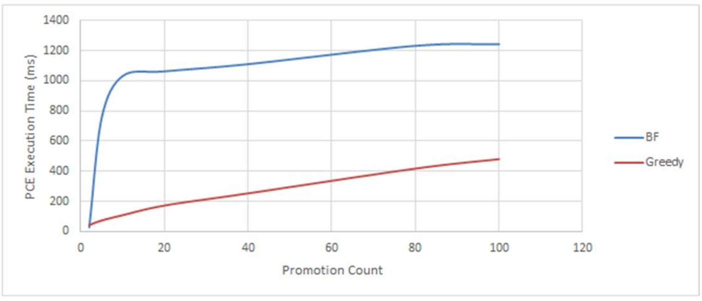
In the third set of experiments, the number of line items is set to 20 and the quantity of each line item is set to 20. The number of colliding promotion price derivation rules is varied. There are two subsets of experiments in this scope. Across the two subsets of the experiments, only the value for the calculation time limit differs. The results of the first subset of experiments are shown below:

CALCULATION_TIME_LIMIT = 1000 ms AND NUMBER_OF_LINE_ITEMS = 20 AND LINE_ITEM_QUANTITY = 20

PROMOTION_COUNT	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
2	80	80	29	44
5	200	200	759	74
10	400	400	1032	109
20	800	800	1063	173
40	800	800	1111	254
80	800	800	1232	418
100	800	800	1243	481

The graphical representations of these data are shown in the 2 figures below:



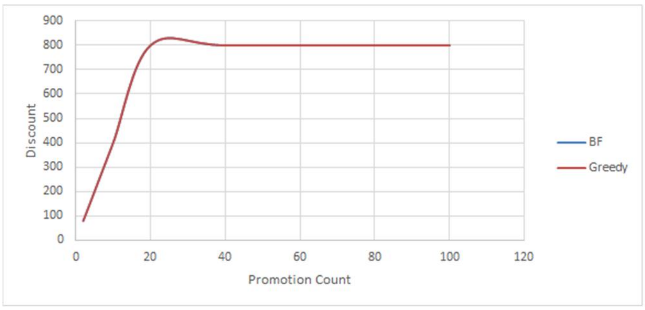


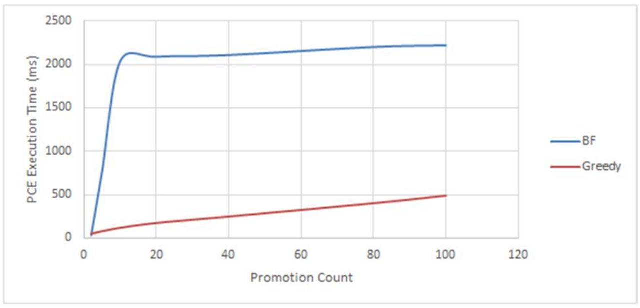
The results of the second subset of experiments are shown below:

CALCULATION_TIME_LIMIT = 2000 ms AND NUMBER_OF_LINE_ITEMS = 20 AND LINE_ITEM_QUANTITY = 20

PROMOTION_COUNT	BF DISCOUNT	GREEDY DISCOUNT	BF PCE EXECUTION TIME (ms)	GREEDY PCE EXECUTION TIME (ms)
2	80	80	35	51
5	200	200	766	80
10	400	400	2034	119
20	800	800	2088	175
40	800	800	2108	249
80	800	800	2200	403
100	800	800	2221	489

The graphical representations of these data are shown in the 2 figures below:





10.1.4 Results of the Experiments

The following results can be derived roughly from the above experiments:

- When the number of line items in the transaction is below 1000, the greedy algorithm can give acceptable discounts and better performance in terms of execution time. But if the number of line items is too small (<3), the brute force algorithm can be selected when we consider that the greedy algorithm may generate worse discount amounts for some specific scenarios.

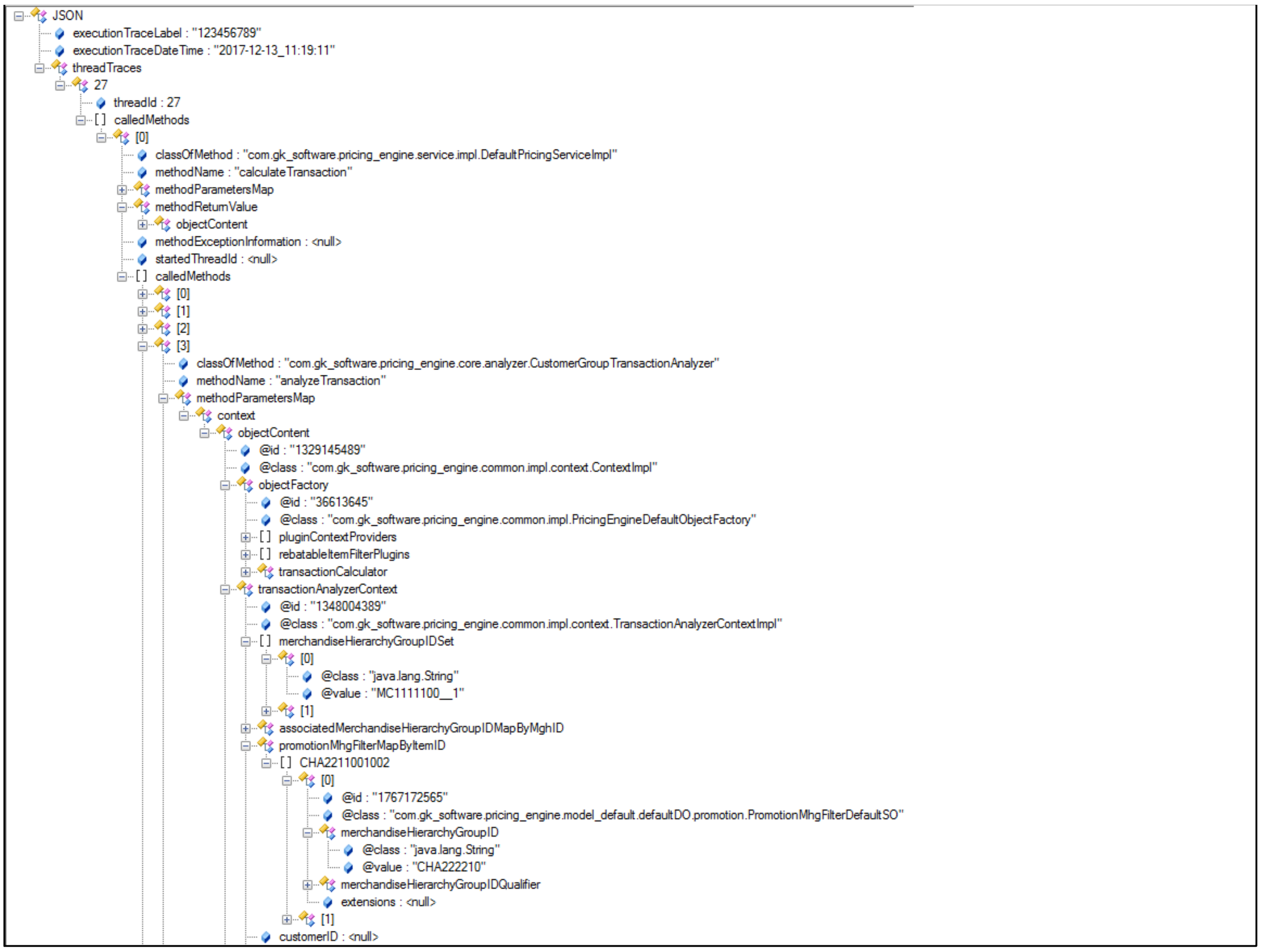
- However, when the quantity of each line item goes beyond 300, the brute force algorithm starts to give better discounts.
- Also, when the colliding promotion price derivation rule count is very small (<3), the brute force algorithm outperforms the greedy algorithm (as can be seen on the last set of experiments with varying promotion price derivation rule counts).
- When the number of colliding promotion price derivation rules gets bigger (>5), the greedy algorithm outperforms the brute force.

10.2 Log the Calculation of the Benefit

10.2.1 Overview of the Implementation

An Aspect Oriented Programming (AOP) -based solution to the PCE calculation logic logging problem is implemented. The implementation utilizes Spring AOP support (library) to log the methods called during the processing of a PCE request. The names and values of the parameters and the return values of the methods are logged with information about the system hash values and types of them. In essence, the unique system hash value for each object (parameter, return object or an object contained by them, whatever it is) is logged in order to eliminate the duplicate logging of a specific object in a specific scope. This scope is the parameter or return object scope. Namely, in the log of a specific parameter or return object, there is no duplicate log of the same object contained by this parameter or return object. If a parameter or return object x has more than one references (containment) of an object y, then only the first reference of y is logged with the full content of object y. Other references to object y within object x are logged only with the unique system hash value of object y.

The classes in which the methods to be logged are found include the default classes and the custom classes that are specified by the user via an external configuration property. The classes of the logged methods are also logged. The output is a JSON file storing the logged parameter and return objects including their values and types, and exceptions raised, if any. A sample output is shown in the below figure.



```
JSON
  executionTraceLabel : "123456789"
  executionTraceDateTime : "2017-12-13_11:19:11"
  threadTraces
    27
      threadId : 27
      [ ] calledMethods
        [0]
          classOfMethod : "com.gk_software.pricing_engine.service.impl.DefaultPricingServiceImpl"
          methodName : "calculateTransaction"
          methodParametersMap
          methodReturnValue
            objectContent
          methodExceptionInformation : <null>
          startedThreadId : <null>
          [ ] calledMethods
            [0]
            [1]
            [2]
            [3]
              classOfMethod : "com.gk_software.pricing_engine.core.analyzer.CustomerGroupTransactionAnalyzer"
              methodName : "analyzeTransaction"
              methodParametersMap
                context
                  objectContent
                    @id : "1329145489"
                    @class : "com.gk_software.pricing_engine.common.impl.context.ContextImpl"
                    objectFactory
                      @id : "36613645"
                      @class : "com.gk_software.pricing_engine.common.impl.PricingEngineDefaultObjectFactory"
                      [ ] pluginContextProviders
                      [ ] rebatableItemFilterPlugins
                      transactionCalculator
                    transactionAnalyzerContext
                      @id : "1348004389"
                      @class : "com.gk_software.pricing_engine.common.impl.context.TransactionAnalyzerContextImpl"
                      [ ] merchandiseHierarchyGroupIDSet
                        [0]
                          @class : "java.lang.String"
                          @value : "MC1111100__1"
                        [1]
                      associatedMerchandiseHierarchyGroupIDMapByMghID
                      promotionMhgFilterMapByItemID
                        [ ] CHA2211001002
                          [0]
                            @id : "1767172565"
                            @class : "com.gk_software.pricing_engine.model_default.defaultDO.promotion.PromotionMhgFilterDefaultSO"
                            merchandiseHierarchyGroupID
                              @class : "java.lang.String"
                              @value : "CHA222210"
                            merchandiseHierarchyGroupIDQualifier
                            extensions : <null>
                          [1]
                      customerID : <null>
```

The output JSON object describes an execution trace that can contain traces for multiple threads. The label of the execution trace is determined as the message identifier of the corresponding PCE request. Each thread has its own trace in the overall execution trace, which includes the methods called in this thread. The log of a method contains the logs of the methods called by this method also. The identifiers and classes of the logged objects in the scope of called methods are specified by @id and @class keys in order. The identifier of an object is its unique system hash value.

HOW IT WORKS

10.2.2 Configuration of Calculation Logic Logging Functionality

As stated above, there are classes whose methods are logged by default by the calculation logic logging implementation. These classes include the following with their package names:

- com.gk_software.pricing_engine.service.impl.DefaultPricingServiceImpl
- com.gk_software.pricing_engine.api.eligibility.EligibilityLoader+
- com.gk_software.pricing_engine.api.engine.ConditionLoader+
- com.gk_software.pricing_engine.api.rules.RuleLoader+
- com.gk_software.pricing_engine.api.analyzer.TransactionAnalyzer+
- com.gk_software.pricing_engine.common.engine.PromotionCalculator+
- com.gk_software.pricing_engine.common.result.SummaryResultHandler+

- com.gk_software.pricing_engine.common.result.PartialResultCalculator+
- com.gk_software.pricing_engine.common.calculator_service.CalculatorService+
- com.gk_software.pricing_engine.common.engine.ConflictHandler+
- com.gk_software.pricing_engine.api.rules.EligibilityRuleHandler+
- com.gk_software.pricing_engine.api.rules.RebateDistributionProcessor+
- com.gk_software.pricing_engine.api.rules.CalculationRuleProcessor+
- com.gk_software.pricing_engine.common.transaction.TransactionUpdateHandler+

The '+' sign at the end of a class canonical name above indicates that the logged classes include both the specified class (or interface) and the classes inheriting from this class (or interface). For instance, from the second item above it can be deduced that the methods of all the classes implementing the EligibilityLoader interface are logged by default.

In addition to the default classes, the classes in which the methods to be logged can also be specified by the users via an external configuration property named 'calculation.logging.custom.entities'. The value of this property is a list of package prefixes and class canonical names (that can be ended with '+' sign) separated by comma. An example value for this property is shown below:

```
calculation.logging.custom.entities=com.gk_software.pricing_engine.api.eligibility.Eligibility
Activator+,com.gk_software.pricing_engine.api.eligibility.EligibilityValidator+
```

In case a package is included in the property 'calculation.logging.custom.entities', the calls of the methods of all the classes in all the packages whose name start with this prefix value will be logged.

Another configuration property that can be specified also externally is named 'calculation.logging.logFilesLocation', which determines the location of the directory in which the output JSON file is stored.

No file separator should be placed at the end of the specified location.
The 'calculation.logging.custom.entities' and 'calculation.logging.logFilesLocation' properties should be stored in a file named 'application-calculationLogging.properties' stored in the directory that the user runs the PCE application.

10.2.3 Activating the Calculation Logic Logging Functionality

The name of Spring profile that the user should activate before running PCE in order to activate the calculation logic logging functionality is 'calculationLogging'. After creating and customizing the 'application-calculationLogging.properties' file as stated above in the directory where the user will run the PCE application, she/he can run the PCE application as in the following example code block with activation of the calculation logic logging functionality:

```
java -jar <PCE-application-name>.jar --spring.profiles.active="calculationLogging"
```

THERE IS MORE

10.2.4 Limitations of the Calculation Logic Logging Functionality

One of the important limitations of the implementation of the calculation logic functionality is that the implementation does not detect changes in objects across the method calls. This results in duplicate logging of the content of the same object across method calls. For instance, even if the context object does not change across two method calls of which it is a parameter, the context object's content is logged twice as parameter object for each of these methods. This may cause the size of the output file bigger than that should be and also decrease the performance. In order to eliminate this, another aspect could be implemented, which can detect changes in objects.

Second important limitation is that the methods of only the objects which are Spring beans can be logged. It is a limitation of Spring AOP.

Third limitation is that there is no special handling for loops now. A special handling can be implemented for loops in the future as described in the solution proposal. But in this respect, the fact that mostly private methods are called inside loops should be considered, as private methods are not logged.

10.2.5 Related Topics

The calculation logic logging functionality can be useful for both consultants and developers. For consultants, it can give insights into the calculation of the benefit by PCE. For developers, it can serve as a call flow and memory profiler and help them improve the quality and performance of the code. One special related case can be the large number of references to the same object inside another object. Most of these references can be removed by developers via coordination with designers, and as a result, the performance and quality of the PCE implementation can be increased.

In essence, the calculation logic functionality is an instance of call graph generation according to the execution of the application. The generated call graph represents not the all possible execution paths that can be taken by the application, instead it represents only a part of the execution path followed by the application in response to a specific request. There are other tools that can be used for call graph generation for Java programs. The links to some of them are listed below:

https://sable.github.io/soot/

https://github.com/gousiosg/java-callgraph

https://projects.eclipse.org/projects/tptp.platform

10.3.1 GS1 Value Conversions

The GS1 Purchase Requirement field determines threshold requirements of the price derivation rule eligibilities to be used. The format of this field is controlled by the Purchase Requirement Code field based on which the conversion must be implemented. The following conversion rules are to be followed when interpreting the Purchase Requirement as a BigDecimal.

Purchase requirement code	Conversion	Example
0	The purchase requirement defines the number of units to purchase, thus it must be considered as an integer.	"1" → 1.0 "123" → 123.0
1 / 2	The purchase requirement defines a cache value, thus it must be interpreted as cents, the last two digits are always decimal values.	"1" → 0.01 "123" → 1.23
3	The purchase requirement defines a weight value in pounds, the last two digits are always decimal values.	"1" → 0.01 "123" → 1.23
4	The purchase requirement defines a weight value in kilograms, the last three digits are always decimal values.	"1" → 0.001 "123" → 0.123
9	Not supported	

The GS1 Save Value field determines the face amount granted by the coupon. The format of this field is defined by the Save Value Code field based on which the conversion must be implemented. The following conversion rules are to be followed when interpreting the Save value as a BigDecimal.

Save value code	Conversion	Example
null / 0 / 6	The save value defines the discount amount in cents, thus it must be interpreted as cents, the last two digits are always decimal values.	"1" → 0.01 "123" → 1.23
1	The save value defines the limit amount of the free item, thus it must be interpreted as cents, the last two digits are always decimal values.	"1" → 0.01 "123" → 1.23
2	The save value defines the number of qualifying purchase item units, thus it must be considered as an integer.	"1" → 1.0 "123" → 123.0

Save value code	Conversion	Example
5	The save value defines the reduction percent, thus it must be considered as an integer.	"1" → 1.0 "123" → 123.0

10.3.2 Mapping the Ad Hoc Promotion from a Manufacturer Coupon

The GS1 formatted barcode of US Manufacturer Coupons consists of several control fields determining what type of a promotion should be applied. The following tables clarify what price derivation rule eligibilities and price derivation rules must be mapped in case of which GS1 value.

The price derivation rule eligibility to use is determined based on the *Purchase Requirement Code*. The primary, 2nd, and 3rd qualifying purchase's purchase requirement code must all be handled accordingly. Note that the "*Save Value Applies to which Item*" value is only considered in case the "*Additional Purchase Rules Code*" is 1 or 2, meaning that multiple qualifying items are available because of the AND combination eligibility.

Purchase requirement code	Price derivation rule eligibility to map	
0	**ManufacturerCouponPromotionConditionEligibilitySO**	
	Eligibility Type	MACP
	Threshold Type Code	QUT
	Company Code	Company Prefix of the Qualifying Purchase Item
	Family Code	Purchase Family Code of the Qualifying Purchase Item
	UOM Code	PCE
	Threshold Quantity	Purchase Requirement of the item Qualifying Purchase Item
	Parse Eligibility Flag	True

Purchase requirement code	Price derivation rule eligibility to map	
1	**ManufacturerCouponPromotionConditionEligibilitySO**	
	Eligibility Type	MACP
	Threshold Type Code	AMT
	Company Code	Company Prefix of the Qualifying Purchase Item
	Family Code	Purchase Family Code of the Qualifying Purchase Item
	UOM Code	_ALL
	Threshold Amount	Purchase Requirement of the item Qualifying Purchase Item
	Limit Amount	Purchase Requirement of the item Qualifying Purchase Item
	Parse Eligibility Flag	True
2	**ManufacturerCouponPromotionConditionEligibilitySO**	
	Eligibility Type	MACP
	Threshold Type Code	AMT
	Company Code	_ALL
	Family Code	000
	UOM Code	_ALL
	Threshold Amount	Purchase Requirement of the item Qualifying Purchase Item
	Limit Amount	999999.0
	Parse Eligibility Flag	True

Purchase requirement code	Price derivation rule eligibility to map	
3	**ManufacturerCouponPromotionConditionEligibilitySO**	
	Eligibility Type	MACP
	Threshold Type Code	QUT
	Company Code	Company Prefix of the Qualifying Purchase Item
	Family Code	Purchase Family Code of the Qualifying Purchase Item
	UOM Code	LBS
	Threshold Quantity	Purchase Requirement of the item Qualifying Purchase Item
	Parse Eligibility Flag	True
4	**ManufacturerCouponPromotionConditionEligibilitySO**	
	Eligibility Type	MACP
	Threshold Type Code	QUT
	Company Code	Company Prefix of the Qualifying Purchase Item
	Family Code	Purchase Family Code of the Qualifying Purchase Item
	UOM Code	KG
	Threshold Quantity	Purchase Requirement of the item Qualifying Purchase Item
	Parse Eligibility Flag	True
9	Not supported	

In case of multiple qualifying purchase items are defined in the manufacturer coupon, the price derivation rule eligibility must be constructed according to the Additional Purchase Rules Code field as follows. If the Additional Purchase Rules Code field is not provided in the barcode, then the Primary Qualifying Purchase must be mapped to a promotion as defined in the above table, no combination of price derivation rule eligibilities is necessary.

Additional purchase rules code (if available)	Price derivation rule eligibility to map
0	**OR** combination eligibility of the following child eligibilities • Price derivation rule eligibility for the *Primary* Item according to the "*Primary* purchase requirement code" • Price derivation rule eligibility for the *2nd* Item according to the "*2nd* purchase requirement code" • Price derivation rule eligibility for the *3rd* Item according to the "*3rd* purchase requirement code", if available
1	**AND** combination eligibility of the following child eligibilities • Price derivation rule eligibility for the *Primary* Item according to the "*Primary* purchase requirement code" • Price derivation rule eligibility for the *2nd* Item according to the "*2nd* purchase requirement code" • Price derivation rule eligibility for the *3rd* Item according to the "*3rd* purchase requirement code", if available
2	**AND** combination eligibility of the following child eligibilities • Price derivation rule eligibility for the *Primary* Item according to the "*Primary* purchase requirement code" • **OR** combination eligibility of o Price derivation rule eligibility for the *2nd* Item according to the "*2nd* purchase requirement code" o Price derivation rule eligibility for the *3rd* Item according to the "*3rd* purchase requirement code"
3	**OR** combination eligibility of the following child eligibilities • Price derivation rule eligibility for the *2nd* Item according to the "*Primary* purchase requirement code" • Price derivation rule eligibility for the *3rd* Item according to the "*Primary* purchase requirement code"

The price derivation rule to map is defined as follows.

Save Value Code	Save Value	The price derivation rule to map		
null / 0	Any value	*MixAndMatchPromotionConditionRuleSO*		

		Mix and Match Combination Code		IF Additional Purchase Rules Code is 0
				THEN OR_QUANTITY
				ELSE XO

		Limit Count (for combination XO)		IF Purchase Requirement Code is 0 OR 3 OR 4
				THEN Purchase Requirement
				ELSE 999999

		Matching Item	Mix and Match Price Modification	RS

			Mix and Match Price Modification Amount	IF Purchase Requirement Code is 0 OR 3 OR 4
				THEN Save Value / Purchase Requirement
				ELSE IF Purchase Requirement Code is 2
				THEN Save Value / Available Qualifying Item Count
				ELSE Save Value

			Matching Manufacturer Coupon Relation	As defined in the mapping table below

			Required Quantity (for combination OR_QUANTITY)	IF Purchase Requirement Code is 0 OR 3 OR 4
				THEN Purchase Requirement
				ELSE 999999

1	0	*MixAndMatchPromotionConditionRuleSO*		

		Mix and Match Combination Code		IF Additional Purchase Rules Code is 0
				THEN OR_QUANTITY
				ELSE XO

		Limit Count (for combination XO)		1
		Matching Item	Mix and Match Price Modification	RP
			Mix and Match Price Modification Percent	100
			Matching Manufacturer Coupon Relation	As defined in the mapping table below
			Required Quantity (for combination OR_QUANTITY)	1

Save Value Code	Save Value	The price derivation rule to map		
	> 0	*MixAndMatchPromotionConditionRuleSO*		

		Mix and Match Combination Code		IF Additional Purchase Rules Code is 0
				THEN OR_QUANTITY
				ELSE XO

		Limit Count (for combination XO)		1
		Matching Item 1	Mix and Match Price Modification	RS
			Mix and Match Price Modification Amount	Save Value
			Matching Manufacturer Coupon Relation	As defined in the mapping table below
			Matching Item ID	< Matching Item 2's Matching Item ID
			Required Quantity (for combination OR_QUANTITY)	1
		Matching Item 2	Mix and Match Price Modification	RP
			Mix and Match Price Modification Percent	100
			Matching Manufacturer Coupon Relation	As defined in the mapping table below
			Matching Item ID	> Matching Item 1's Matching Item ID
			Required Quantity (for combination OR_QUANTITY)	1

Save Value Code	Save Value	The price derivation rule to map		
2	Any value	*MixAndMatchPromotionConditionRuleSO*		

		Mix and Match Combination Code		IF Additional Purchase Rules Code is 0
				THEN OR_QUANTITY
				ELSE XO

		Limit Count (for combination XO)		Save Value
		Matching Item	Mix and Match Price Modification	RP
			Mix and Match Price Modification Percent	100
			Matching Manufacturer Coupon Relation	As defined in the mapping table below
			Required Quantity (for combination OR_QUANTITY)	Save Value
5	Any value	MixAndMatchPromotionConditionRuleSO		

		Mix and Match Combination Code		IF Additional Purchase Rules Code is 0
				THEN OR_QUANTITY
				ELSE XO

		Limit Count (for combination XO)		IF Purchase Requirement Code is 0 OR 3 OR 4
				THEN Purchase Requirement
				ELSE 999999

		Matching Item	Mix and Match Price Modification	RP
			Mix and Match Price Modification Percent	Save Value
			Matching Manufacturer Coupon Relation	As defined in the mapping table below

			Required Quantity (for combination OR_QUANTITY)	IF Purchase Requirement Code is 0 OR 3 OR 4
				THEN Purchase Requirement
				ELSE 999999

Save Value Code	Save Value	The price derivation rule to map
6	Any value	**The price derivation rule eligibility to map** **AND** Combination eligibility of the following child eligibilities. • The first child eligibility is created as defined above in mapping table to satisfy the manufacturer coupon's purchase requirements but with the following change: If the qualifying item is not available, then the price derivation rule eligibility can not be activated due to combination AND, however if the price derivation rule eligibility was activated, then it would not be used for the calculation as defined by the parseEligibilityFlag.

ParseEligibilityFlag	false

- The second child eligibility would be created with the following values, thus its working bucket contains all items from the transaction: *ManufacturerCouponPromotionConditionEligibilitySO*

Eligibility Type	MACP
Threshold Type Code	AMT
Company Code	_ALL
Family Code	000
UOM Code	_ALL
Threshold Amount	Save Value
Parse Eligibility Flag	true

The price derivation rule to map

RebatePromotionConditionRuleSO

Price Modification Type Code	RT
Price Modification Amount	Save Value

MatchingManufacturerCouponRelationSO mapping for the mix and match should be done as follows. Note that the "Save Value Applies to which Item" value is only considered in case the "Additional Purchase Rules Code" is 1 or 2, meaning that multiple qualifying items are available because of the AND combination eligibility.

Save Value Applies to which Item	Additional Purchase Rules Code	Values to use	
Not considered	0	Matching items must be created for all qualifying items found in the coupon with the values defined above. The *MatchingManufacturerCouponRelationSO* attributes for each must be mapped from the qualifying purchase item the Matching Item was created for.	
		Company Code	nth Qualifying Item's Company Code
		Family Code	nth Qualifying Item's Family Code
		UOM Code	nth Qualifying Item's UOM Code based on the Purchase Requirement Code
	3	Two Matching items must be created for the price derivation rule with the mapped values defined above. The *MatchingManufacturerCouponRelationSO* attributes are mapped for the 1st one from the 2nd Qualifying Purchase Item and for the 2nd one from the 3rd Qualifying Purchase Item.	
		Company Code	nth Qualifying Item's Company Code
		Family Code	nth Qualifying Item's Family Code
		UOM Code	nth Qualifying Item's UOM Code based on the Purchase Requirement Code
0 / null	1 / 2	*MatchingManufacturerCouponRelationSO*	
		Company Code	Primary Qualifying Item's Company Code
		Family Code	Primary Qualifying Item's Family Code
		UOM Code	Primary Qualifying Item's UOM Code based on the Purchase Requirement Code
1	1 / 2	*MatchingManufacturerCouponRelationSO*	
		Company Code	2nd Qualifying Item's Company Code
		Family Code	2nd Qualifying Item's Family Code
		UOM Code	2nd Qualifying Item's UOM Code based on the Purchase Requirement Code

Save Value Applies to which Item	Additional Purchase Rules Code	Values to use	
2	1 / 2	*MatchingManufacturerCouponRelationSO*	
		Company Code	3rd Qualifying Item's Company Code
		Family Code	3rd Qualifying Item's Family Code
		UOM Code	3rd Qualifying Item's UOM Code based on the Purchase Requirement Code

10.3.3 Default promotion values to map

In case the mapping tables above did not specify the value to map for a specific attribute of the promotion, then the following default values must be mapped.

The identifiers of the different promotion objects will be assigned by the PCE during the mapping of the ad hoc promotion. The first ad-hoc-promotion-related promotion model object mapped by the PCE will get the ID defined in the PCE manufacturerCouponPromotionID configuration parameter. Further mapped promotion objects will always be assigned the next ID in descending order.

10.3.3.1 PromotionSO

Attribute	Type	Default value to map
BusinessUnitGroupID	Long	context.getTransaction().getKey().getBusinessUnitGroupID()
ConditionList	List<? extends PromotionConditionSO>	Each promotion must have one condition. See mapping of PromotionConditionSO.
CustomerDisplayName	String	"Manufacturer Coupon Discount " + Offer Code
Description	String	"Manufacturer Coupon Discount " + Offer Code
EffectiveDateTime	Timestamp	null
ExpirationDateTime	Timestamp	null
InternalPromotionID	Long	ID generated by PCE according to rules above.
OperatorDisplayName	String	"Manufacturer Coupon Discount " + Offer Code
Origin	String	"03" (external)
PromotionID	String	InternalPromotionID
PromotionMultiLanguageTexts	List<PromotionTextSO>	null
PromotionTypeID	String	null

Attribute	Type	Default value to map
ReceiptPrinterName	String	"Manufacturer Coupon Discount " + Offer Code

10.3.3.2 PromotionConditionSO

Attribute	Type	Default value to map
AmendmentTypeCode	AmendmentTypeCode	SALES_AMENDMENTS("00")
ConcurrenceControlVector	String	null
ConditionID	Long	ID generated by PCE according to rules above.
CustomerDisplayName	String	null
Description	String	"Manufacturer Coupon Discount " + Offer Code
Eligibility	PromotionConditionEligibilitySO	See mapping of PromotionConditionEligibilitySO
ExclusiveFlag	Boolean	false
IconID	String	null
InternalEligibilityID	Long	ID generated by PCE according to rules above.
InternalRuleID	Long	ID generated by PCE according to rules above.
ItemDiscountControlVector	String	null
NotShowingFlag	Boolean	false
OperatorDisplayName	String	null
PromotionID	Long	Promotion's InternalPromotionID
ReceiptPrinterName	String	null
Resolution	Long	0
SaleReturnTypeCode	SaleReturnTypeCode	SALES("01")
Sequence	Long	manufacturerCouponPromotionSequence
Tid	Long	null
TimeGroup	PromotionConditionTimeGroupSO	null
TimeRestrictions	List<TimeRestrictionSO>	null
TypeCode	String	"MACP"

10.3.3.3 PromotionConditionRuleSO

Attribute	Type	Default value to map
BonusPointsFlag	Boolean	false
CalculationBase	CalculationBaseType	null
CalculationBaseSequence	Long	null
ChooseItemMethod	ChooseItemMethod	Requirement FuReMC-025 • If the price modification method defines an absolute discount amount, then LOWEST_FIRST("01") • If the price modification method defines a percentage discount, then HIGHEST_FIRST("02")
ConsiderPreviousPromotionConditionFlag	Boolean	true
CouponPrintoutID	String	null
CouponPrintoutRule	CouponPrintoutRule	null
CouponPrintoutText	Object	null
DecimalPlacesCount	Integer	2
Description	String	null
DiscountMethodCode	String	"99" (discount as manufacturer coupon tender)
ExternalConditionRuleID	String	ConditionID
GiftCertificateExpirationDate	Date	null
InternalRuleID	Long	RuleID
Name	String	null
NoEffectOnSubsequentPromotionConditionFlag	Boolean	false
NoPreviousMonetaryDiscountAllowedFlag	Boolean	false
PointsAmountRoundingRuleID	Long	null
PrintoutValidityPeriod	BigDecimal	0
ProhibitTransactionRelatedPromotionConditionFlag	Boolean	false
RoundDestinationValue	Integer	1

Attribute	Type	Default value to map
RoundingMethodCode	RoundingMethodCode	COMMERCIAL_ROUNDING ("00")
RoundingRuleID	Long	null
RuleID	String	ID generated by PCE according to rules above.
ShareRoundingRuleID	Long	null
StatusCode	String	"AC"
TenderTypeCode	String	The configured tender type code.
Tid	Long	null
TransactionControlBreakCode	RuleControlType	PO

10.3.3.4 RebatePromotionConditionRuleSO extends PromotionConditionRuleSO

Attribute	Type	Default value to map
PromotionConditionRuleSO.TypeCode	RuleType	RuleTypeEnum.RB
AdjustmentMethodCode	AdjustmentMethodCode	"DC"
NewPriceAmount	BigDecimal	null
PriceModificationAmount	BigDecimal	null
PriceModificationPercent	BigDecimal	null

10.3.3.5 MixAndMatchPromotionConditionRuleSO extends PromotionConditionRuleSO

Attribute	Type	Default value to map
PromotionConditionRuleSO.TypeCode	RuleType	RuleTypeEnum.MM
NewSetPriceAmount	BigDecimal	null

10.3.3.6 MatchingItemSO

Attribute	Type	Default value to map
AdjustmentMethodCode	AdjustmentMethodCode	"DC"
InternalRuleID	Long	PromotionConditionRuleSO.InternalRuleID
ItemList	List<? extends MatchingItemRelationSO>	null

Attribute	Type	Default value to map
MatchingItemID	Long	ID generated by PCE according to the rules above.
MerchandiseSet	MerchandiseSetSO	null
MhgList	List<? extends MatchingMhgRelationSO>	null
NewPriceAmount	BigDecimal	0.0
PriceModificationAllocationPercent	BigDecimal	0.0
PriceModificationAmount	BigDecimal	null
PriceModificationPercent	BigDecimal	null
RequiredQuantity	BigDecimal	0.0
Value	BigDecimal	0.0

10.3.3.7 PromotionConditionEligibilitySO

Attribute	Type	Default value to map
EffectiveDateTime	Timestamp	null
ExpirationDateTime	Timestamp	null
InternalEligibilityID	Long	ID generated by PCE according to rules above.
LevelID	Short	null
NegationFlag	Boolean	false
ParentEligibilityID	Long	If the GS1 "Additional purchase rules code" is available, then the parent eligibility's ID, or the internalEligibilityID if this is the root eligibility. If the GS1 "Additional purchase rules code" is not available, then the internalEligibilityID.
RootEligibilityID	Long	If the GS1 "Additional purchase rules code" is available, then the root eligibility's ID, or the internalEligibilityID if this is the root eligibility. If the GS1 "Additional purchase rules code" is not available, then the internalEligibilityID.
StatusCode	String	"AC"

10.3.3.8 ThresholdPromotionConditionEligibility extends PromotionConditionEligibilitySO

Attribute	Type	Default value to map
IntervalAmount	BigDecimal	0.0

Attribute	Type	Default value to map
IntervalQuantity	BigDecimal	0.0
LimitAmount	BigDecimal	ThresholdAmount
LimitQuantity	BigDecimal	ThresholdQuantity
ThresholdAmount	BigDecimal	0.0
ThresholdQuantity	BigDecimal	0.0
ThresholdTypeCode	ThresholdType	null

10.3.3.9 ManufacturerCouponPromotionConditionEligibilitySO extends ThresholdPromotionConditionEligibility

Attribute	Type	Default value to map
PromotionConditionEligibilitySO.TypeCode	EligibilityType	MACP

10.3.3.10 CombinationPromotionConditionEligibilitySO extends ThresholdPromotionConditionEligibility

Attribute	Type	Default value to map
PromotionConditionEligibilitySO.TypeCode	EligibilityType	COMB
ThresholdForSingleItemFlag	Boolean	false

10.4 Cron expression

Format

Field Name	Allowed Values	Allowed Special Characters
Seconds	0-59	, - * /
Minutes	0-59	, - * /
Hours	0-23	, - * /
Day-of-month	1-31	, - * ? / L W
Month	1-12 or JAN-DEC	, - * /
Day-of-Week	1-7 or SUN-SAT	, - * ? / L #
Year (Optional)	empty, 1970-2199	, - * /

For each field, a number of special characters are also available to extend the meaning of the given field. The special characters which must be used to fulfill the requirements are listed below with a short description on how to use them.

- ',' is used as a delimiter for lists. For example, "MON,WED,FRI" in the day-of-week field means "the days Monday, Wednesday, and Friday".
- '-' is used to specify ranges. For example, "10-12" in the hour field means "the hours 10, 11 and 12".
- '?' is allowed for the day-of-month and day-of-week fields. It is used to specify 'no specific value'. This is useful when you need to specify something in one of the two fields, but not the other.
- '*' is used to specify all values. For example, "*" in the minute field means "every minute".
- '/' is used to specify increments. For example, "0/15" in the seconds field means "the seconds 0, 15, 30, and 45". Specifying '*' before the '/' is equivalent to specifying 0 is the value to start with. The "/" character simply helps you turn on every "nth" value in the given field's value set. Thus, "7/6" in the month field only turns on month "7", it does NOT mean every 6th month.
- '#' is allowed for the day-of-week field. This character is used to specify "the n-th" XXX day of the month. For example, the value of "6#3" in the day-of-week field means the third Friday of the month.
- 'L' is allowed for the day-of-month and day-of-week fields. This character is short-hand for "last", but it has a different meaning in each of the two fields. For example, the value "L" in the day-of-month field means "the last day of the month". If used as "L" in the day-of-week field, it simply means "7". But if used in the day-of-week field after another value, it means "the last xxx day of the month". For example, "6L" means "the last Friday of the month". You can also specify an offset from the last day of the month, such as "L-3" which would mean the third-to-last day of the calendar month. When using the 'L' option, it is important not to specify lists, or ranges of values, as you will get confusing/unexpected results.