Accelerate CPQ Package

Version 2.0.0

May 2021

Accelerate CPQ Package

The CPQ Package consists of various components to support the whole process of negotiating a deal – selecting customers and products, pricing and negotiating the deal, approval workflow, quote output. The setup does not require heavy configuration, you only need to provide a few data tables and map transactional data to provide sales insight during the negotiation.

- Quoting Tool
- CPQ Package Configurations
- CPQ Package Version 2.0
- CPQ Configuration Wizard
- CPQ Package Architecture
- Configurator Package Integration
- Configurations Per Quote Type
- CPQ Package Release Notes

Quoting Tool

- Header Level View
- Line Item Level
- Data Requirements
- Setup

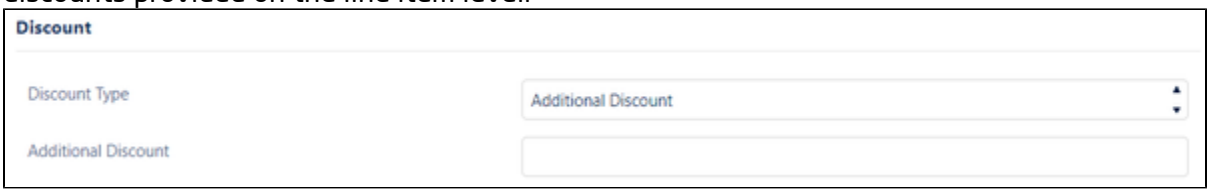
Header Level View

On the header level, the package provides standard input entries, summarized KPIs of quoted items, historical purchases of the customer for quoted items and some general KPIs.

User Inputs

- Customer – Allows the user to choose a customer for the quote; the package allows to set a filter which can limit availability of the customers for selection.
- Currency – Allows the user to choose a customer currency that is defined in the ccy Data Source or from Price Parameters table (version 1.2.2).
- Discount % – Optional user input which can enable one of the predefined discount models to make negotiating easier.

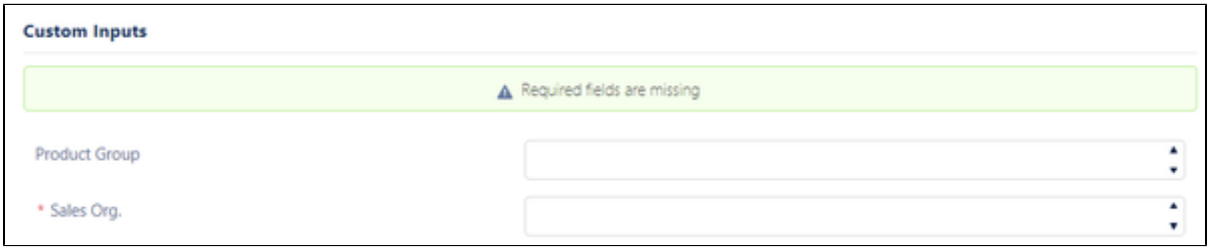
1. Additional Discount – Allows to define an additional discount (absolute value) on top of discounts provided on the line item level.



2. Default Discount % – Allows to define a default discount % for line items; it can be overridden on the line item level.
3. Flat Discount % – Allows to define a discount % which is applied to all line items, overriding discounts on the line item level.

Additional Input Fields

Input fields for the quote header level can be configured without changing the code. These fields can be also applied as parameters for filters used in the customer/product selection screens.

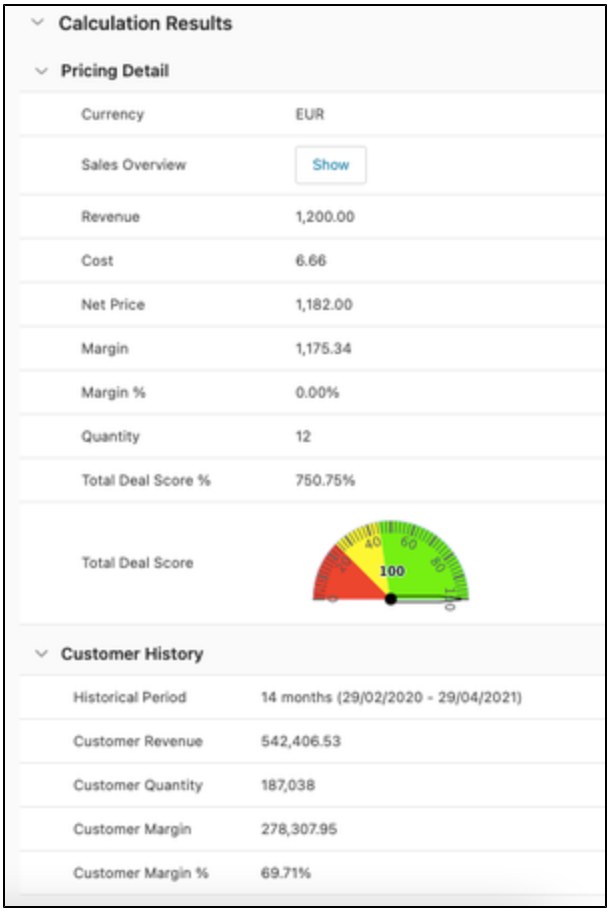


You can configure this manually in the CPQInputConfiguration PP or using CPQ Configuration Wizards with *Name, Label, Input Type, Source Table, Source Type, Source Field, Value Options* (if there is no source field) *, Default Value, Read Only, Required, Level (Header/Line), Customer Filter, Product Filter, Value Options Filtered By Customer, Value Options Filtered By Product*. If the input values are set to be applied to the customer/product filter, the logic will get the value of these inputs and pass them to the filter logic using f ilterFormulaParam.

You need to create your own filter logic for the customer/product filter and build your filter based on the custom input value provided from filterFormulaParam. From version 1.2, we provide the default implementation of customer/product filters for Source Type = P/C and Input Type = OPTION/OPTIONS.

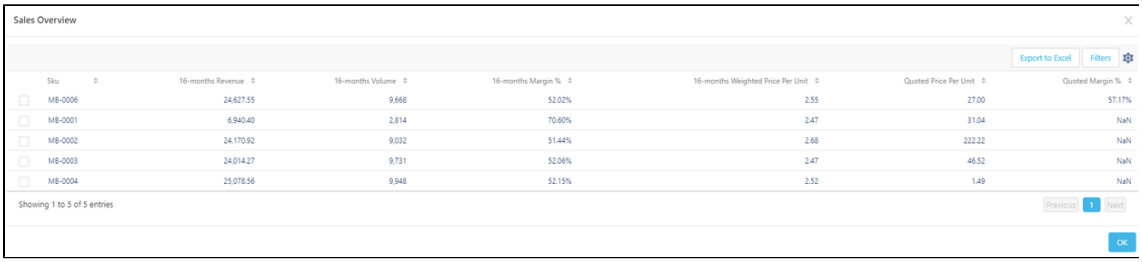
Calculated Results

In the Calculation Results, there are two sections: Pricing Detail and Customer History.



Pricing Detail

- Currency - Displays customer currency (the default is based on *defaultCurrency* in the CPQConfiguration PP)
- Sales Overview
  - If some products were purchased by the customer in last x months, the table can show the following KPIs and allows to compare the currently quoted price and margin:
    - X-months Revenue
    - X-months Volume
    - X-months Margin %
    - X-months Weighted Price Per Unit = sum (Invoice Price) / sum (Quantity) (if Invoice Price is multiplied by Quantity)
    - Quoted Price Per Unit (from the current quote)
    - Quoted Margin % (from the current quote)



- Revenue
- Cost

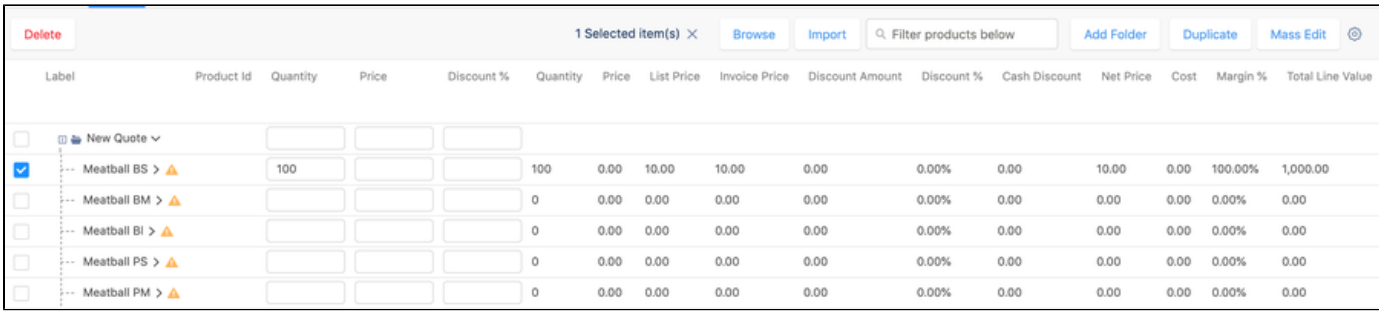
- Net Price
- Margin %
- Margin
- Quantity
- Total Deal Score % = Sum (Invoice Price * Quantity) / Sum (Target Price * Quantity)
- Total Deal Score

Customer History

If the historical sales data are available and you configure *historicalPeriod* in the CPQConfiguration PP (the default is 12 months), it shows summary data for the time period defined during the installation process. The following KPIs are available:

- Historical Period
- Customer Revenue
- Customer Quantity
- Customer Margin
- Customer Margin %

Line Item Level



User Inputs

For quoted products the user can define the following inputs:

- Quantity
- Price
- Discount %

> If there are values in both fields (Price and Discount %), the priority level depends on your configuration *priceVsDiscountPctInputPriority* in the CPQConfiguration PP.

Calculated Results

Pricing Details

This section provides detailed information about pricing of the item: it starts with a List Price and shows the different adjustments to pricing leading to the Net Price and Margin values.

List Price is obtained from two sources in this order: Price Lists and Product Extensions.

Pricing Detail

Quantity	25
List Price	50.63
List Price Source	PL-679
Discount Amount	0.23
Discount %	0.45%
Invoice Price	50.40
Rebate	0.00
Cash Discount %	1.50%
Net Price	49.65
Cost	14.42
Margin	35.23
Margin %	70.96%
Revenue	1,260.11

Previous Price provides information about the last transaction done by the customer and product for the quoted item.

Previous Price

Label	Result
Unit Price	1.87
Margin %	52.00%
Date	2019-02-01

A product price is queried from a price list (PriceBuilder module) which:

- Has the Approved status (approvalState == APPROVED).
- Is assigned to the quoted customer or not assigned to any customer.
- Has a target date <= quote target date.

If there are multiple price lists found, the one with the latest target date takes priority. A price list assigned to the quoted customer will be used for the lookup first and if the product price is not found, the system will continue to a price list not assigned to any customer. When multiple PLs are assigned to a customer, the Quote will check the priority order.

From version 2.0, CPQ supports getting product prices from the matrix type. The product price can differ based on secondary keys. By default, CPQ uses volume breakdown configured by Price Setting Accelerator (Volume Breakdown) to look up the price. For details see Matrix Price List Lookup.

If the price list is not found, the product price will do a lookup from a product extension (PX) (4 levels):

- Price By Country
- Price By Customer
- Price By Region
- Price By Segment

PX records should meet these conditions:

- The target date is not set or should be <= quote target date.
- The price is set.

If there are multiple records found, the one with the latest target date takes priority. Level keys on supported must be available on customer master attributes.

From version 2.0, the List Price Source element is added in Pricing Details. It helps you to know which PL-ID or product extensions PX is used to get a value from and display in Quote. You can turn it on or off in CPQ_Default_Input_Output_Configuration PP.

Discount

Discount Type	Level	Priority	Description
Additional Discount	Header level		Applies to the overall quote invoice price
Flat Discount %	Header level	1	
Discount %	Line Item level	2	
Default Discount %	Header level	3	

Exception Discount and Standard Discount

If we have Exception Discount and Standard Discount:

- Exception Discount takes priority over Standard Discount.
- Will apply together with Flat Discount %, Discount %, Default Discount %.

Exception Discount

It is a predefined PP table named "CPQExceptionDiscount" created to configure an exception discount for quoted products with combination of customer ID and SKU.

Standard Discount

It is a predefined PP table named "CPQStandardDiscount" and there are six keys to configure the standard discount for a product. It is possible to define that a record in the table is valid for any key by using a generic key alias (the default is *****) as the key value. The value can be set as an amount discount or % discount.

ℹ The lookup will ignore any discount less than 0.

Discount Amount = List Price – Invoice Price

or

Discount Amount = Discount % * List Price

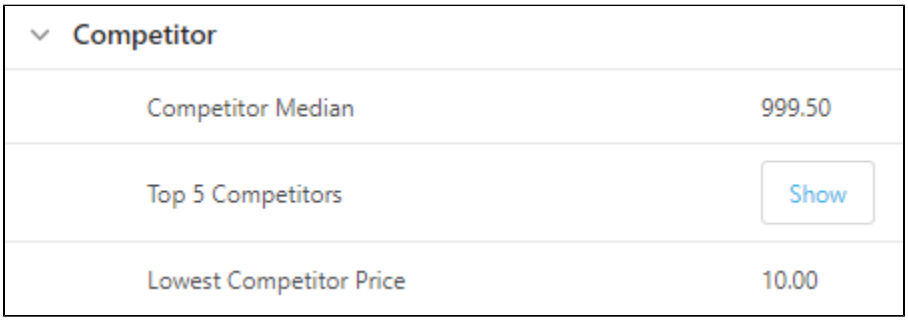
If Discount Amount > List Price, it will not be applied and then a warning message for this "Discount Amount must be less than List Price" should be shown.

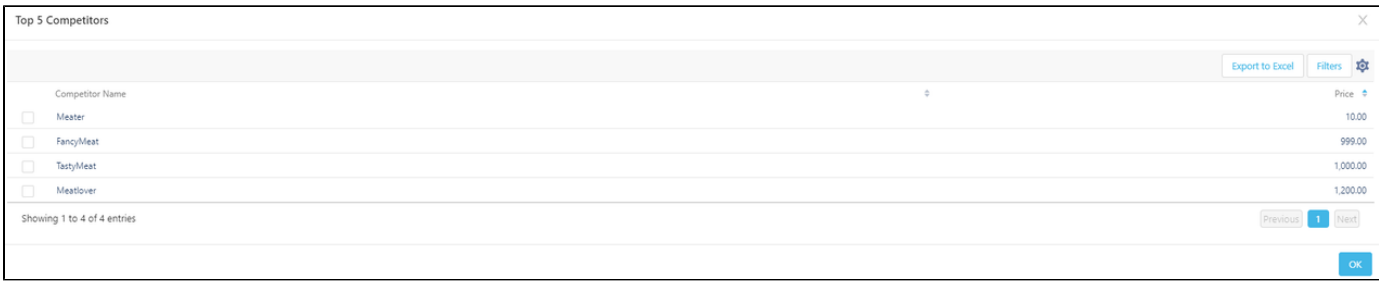
- **Invoice Price** – It is the Price input if set; otherwise it is the difference between List Price and Discount Amount.
- **Rebate –** Rebate data for a customer and product from RebateManager. There are two calculation schemes: current or forecast (the maximum rebate is temporarily not be applied). The configuration is here.
  - To calculate a rebate, you need data from Rebate Records (RR) and you need to meet these requirements:
    - Agreement Status == Approved
    - Start Date (RR) <= Effective Date (Quote)
    - End Date (RR) >= Expiry Date (Quote)
    - Applies to Customer Grouping, Customer and Product Grouping, Product
  - Formula:
    current = **List Price * CurrentRebate / CurrentBaselineValue**
    forecast = **List Price * Forecast Rebate / ForecastBaselineValue**
  - If we have many rebate types applied to the specific combination of Customer and Product, we will sum up all these rebate values.
- **Cash Discount %** – Value from the customer extension "CustomerCashDiscount" (duplicate customer IDs are not allowed).
  ℹ Ignore if Cash Discount % is null, zero, negative and do not use Target Date to compare.
- **Net Price** = Invoice Price – Rebate – Cash Discount % * Invoice Price
- **Cost** – Value from the product extension "ProductCost". Cost records should have:
  - Target Date attribute not set.
    or
  - Target Date <= quoted Target Date
  - Cost should be greater than 0.

  If there are multiple records found, the one with the latest target date takes priority.
  ℹ Show the red alert and message "Invalid Cost" if the cost is not provided (null, zero, negative).
- **Margin** = Net Price – Cost (do not show a value if no cost is provided)
- **Margin %** = Margin / Net Price (do not show a value if no cost is provided)
- **Revenue** = Invoice Price * Quantity

Competitor

If available, this section can provide the user with the information about competitors and theirs pricing. It shows Median Competitor Price, Top 5 Competitors and their prices and the lowest Competitor Price.



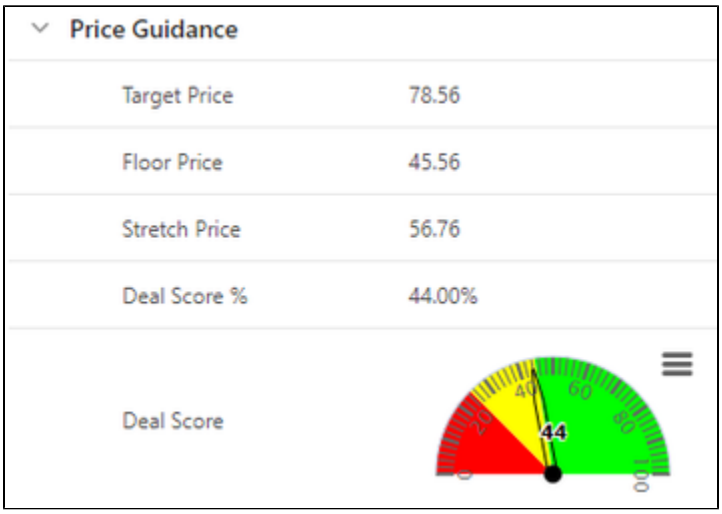


You can get the value from Competition Data, Price Competitors PX or both. Target date (Info date) will be used only if we have the same competitor for a given product, then we get a value based on the latest target date.

Ignore if the competitor price is zero, null, negative.

Price Guidance

If there are data for price guidance from a PX named "PriceGuidance" (can be also populated by PriceOptimizer), this section provides guidance to the user about Target, Floor and Stretch Prices.

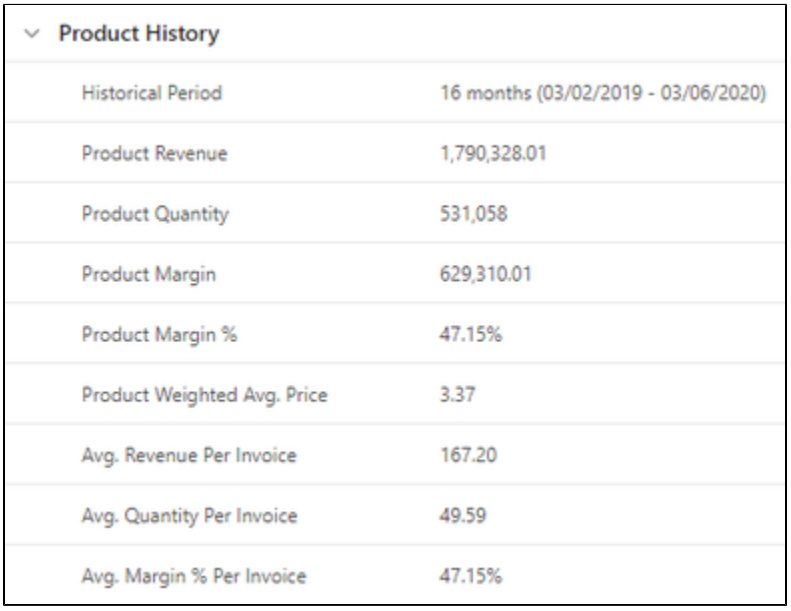


Deal Score % = Invoice Price / Target Price

Ignore if the one of the values is zero, null, negative.

History

If transactional data is available, this section provides an insight about the size of historical purchases of the product.



Product History	
Historical Period	16 months (03/02/2019 - 03/06/2020)
Product Revenue	1,790,328.01
Product Quantity	531,058
Product Margin	629,310.01
Product Margin %	47.15%
Product Weighted Avg. Price	3.37
Avg. Revenue Per Invoice	167.20
Avg. Quantity Per Invoice	49.59
Avg. Margin % Per Invoice	47.15%

- Historical Period (based on *historicalPeriod* in the CPQConfiguration PP, the default is 12 months)
- Product Revenue
- Product Quantity
- Product Margin
- Product Margin %
- Product Weighted Avg. Price = Sum of Invoice Price / Sum of Quantity (if Invoice Price is multiplied by Quantity)
- Avg. Revenue Per Invoice
- Avg. Quantity Per Invoice
- Avg. Margin % Per Invoice = Sum of Margin / Sum of Net Price

Data Requirements

- Mandatory
  - Price list / Base price (can be linked to the price list in PFX PriceBuilder or it can use the data from a Product Extension)
  - Costs
- Optional
  - Competition Data
  - Customer Cash Discount
  - Transactional Data
  - Price Guidance Data (Target, Floor, Stretch margins)

Setup

You can set up this package manually or through PlatformManager.

CPQ Package Configurations

These configurations can be used to configure how the Quoting Tool works or gets data (e.g. prices, cost, competitors, inputs, outputs, etc.):

- Basic Configuration
- Inputs Configuration
- Outputs Configuration
- Currency Conversion Configuration
- Matrix Price List Lookup

Basic Configuration

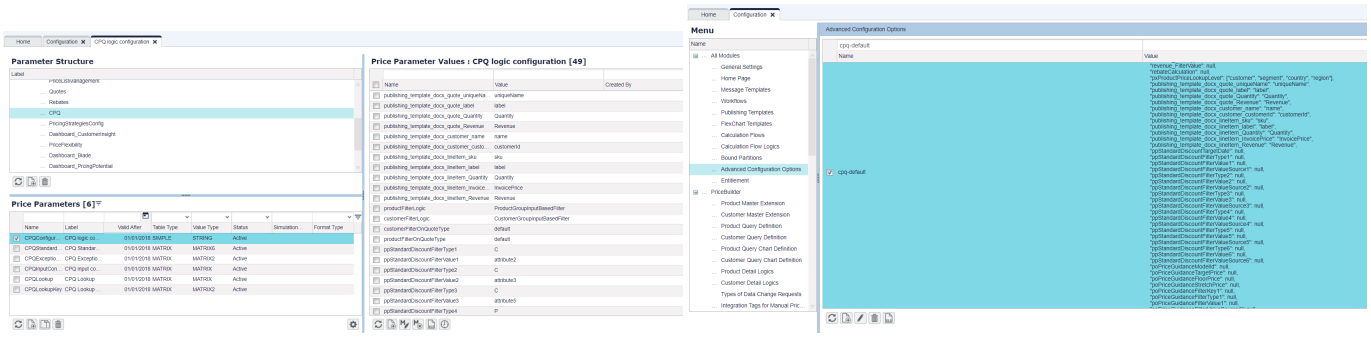
- Package Configuration Source
- Historical Data Configuration
- Product Prices Configuration
- Customer Mapping Configuration
- Cost Configuration
- Price Competitors Configuration
- Price Guidance Configuration
- Standard Discount Configuration
- Rebate Configuration
- Publishing Template Data Configuration
- Other Configurations

Package Configuration Source

CPQ Package reads configurations from:

- Advanced Configuration with the key named "cpq-default"
- Price Parameter table named "CPQConfiguration"

Those can be different per quote type, see Configuration Per Quote Type for more details.



Configurations defined in the Price Parameter table will override those in Advanced Configuration.

Historical Data Configuration

Name	Description
historicalPeriod	Optional. Historical period (the default is 12 months).
dm_name	Datamart name
dm_invoice_id_column	Invoice ID column name
dm_invoice_price_column	Invoice price column name
dm_net_price_column	Net price column name
dm_margin_column	Margin column name
dm_pricing_date_column	Pricing date column name
dm_product_id_column	Product ID column name
dm_quantity_column	Quantity column name
dm_per_unit_price_column	Price per unit column. The default is: invoice_price_column / quantity_column.
dm_revenue_column	Revenue column name
dm_customer_id_column	Customer ID column name
revenue_FilterField	Revenue filter column It applies to Product Weighted Avg. Price in Quote (Product History result).
revenue_FilterType	Revenue filter type. Supported types: • C - customer • ABS - absolute value It applies to Product Weighted Avg. Price in Quote (Product History result).
revenue_FilterValue	Revenue filter value. It can be a Customer attribute to get data or absolute data to filter. It applies to Product Weighted Avg. Price in Quote (Product History result).

Product Prices Configuration

Name	Description
pxProductPriceLookupLevel	Product price lookup level. There are four levels supported: • Customer • Region • Country • Segment Multiple levels can be used - separate them by a comma and give priority by their order.

Customer Mapping Configuration

Customer attribute mapping is required to get a product price for each level.

Name	Description
customerCountryAttribute	Customer country attribute
customerRegionAttribute	Customer region attribute
customerSegmentAttribute	Customer segment attribute
customerCurrency	Customer currency

Cost Configuration

Name	Description
pxCostFilterKey1|2|3|4|5|6	Optional. Additional dimension to configure the cost. There are maximum six dimensions allowed.
pxCostFilterType1|2|3|4|5|6	Optional. Value type for each dimension. Supported types: • P - Product • C - Customer • PX - Product extension • CX - Customer extension
pxCostFilterValue1|2|3|4|5|6	Optional. Value attribute for each dimension.
pxCostFilterValueSource1|2|3|4|5|6	Optional. Name of PX or CX to get the value.

Price Competitors Configuration

Price Competitors can be obtained from both PX and Competition Data.

Name	Description
pxCompetitorDate	Target Date column
pxCompetitorName	Competitor Name column
pxCompetitorPrice	Competitor Price column
pxCompetitorTable	Name of PX to get the value from

Price Guidance Configuration

Price Guidance can be obtained from PX and PriceOptimizer.

Name	Description
poPriceGuidanceModelId	PriceOptimizer model ID for the product price guidance
poPriceGuidanceTargetPrice	Target price column
poPriceGuidanceFloorPrice	Floor price column
poPriceGuidanceStretchPrice	Stretch price column
poPriceGuidanceFilterKey1|2|3|4|5|6	Optional. Additional dimension for filtering. There are maximum six dimensions allowed.
poPriceGuidanceFilterType1|2|3|4|5|6	Optional. Value type for each dimension. Supported types: • P – Product • C – Customer • PX – Product extension • CX – Customer extension
poPriceGuidanceFilterValue1|2|3|4|5|6	Optional. Value attribute for each dimension.
poPriceGuidanceFilterValueSource1|2|3|4|5|6	Optional. Name of PX or CX to get the value.

Standard Discount Configuration

Name	Description
ppStandardDiscountFilterType1|2|3|4|5|6	Value type for each dimension. Supported types: • P – Product • C – Customer • PX – Product extension • CX – Customer extension
ppStandardDiscountFilterValue1|2|3|4|5|6	Attribute to get the value using a filter.

ppStandardDiscountFilterValueSource1|2|3|4|5|6	Name of PX or CX to get the value.

Rebate Configuration

Name	Description
rebateCalculation	Rebate calculation plan. Supported values are: • Forecast • Current

Publishing Template Data Configuration

Name	Description
publishing_template_docx_quote_ {name_use_in_template}	Gets data from the quote object. The value of the config should be the attribute you want to get. E.g. publishing_template_docx_quote_quoteName: uniqueName
publishing_template_docx_customer_ {name_use_in_template}	Gets data from the customer object.
publishing_template_docx_lineItem_ {name_use_in_template}	Gets data from the lineItem object. The value on the lineItem level can be an input name or output name.

Other Configurations

Name	Description
productFilterLogic	Product filter logic name used to the filter quote products.
customerFilterLogic	Customer logic name used to filter the customers.
customerFilterOnQuoteType	Optional. Default value: default.
productFilterOnQuoteType	Optional. Default value: default.
displayPriceInput	Shows/hides the price input on each line item. Possible values: true | false
displayDiscountPercentageInput	Shows/hides the discount % input on each line item. Possible values: true | false
priceVsDiscountPctInputPriority	Defines priority of the price input and discount % input. Possible values: priceInputPriority, discountPercentageInputPriority |

	discountPercentageInputPriority, priceInputPriority

Inputs Configuration

The CPQ Inputs Configuration is deprecated from version 2.0. It is recommended to use CPQ_Default_Input_Output_Configuration PP instead.

The CPQ Inputs Configuration is used to create a custom input in the CPQ Package. Custom inputs can be displayed on the header or line item level.

Custom inputs are generated under the input configurator:

- LineItemInputConfigurator - Contains inputs for each line item.
- HeaderInputConfigurator - Contains inputs for the header level.

Details

Name	Description
Name	Name of the input
Label	Input label to display on a quote
Input Type	Type of the input. Supported values are: • USERENTRY - normal user input • STRINGUSERENTRY - string user input • INTEGERUSERENTRY - integer user input • BOOLEANUSERENTRY - Boolean user input • DATEUSERENTRY - date user input • TIMEUSERENTRY - time user input • DATETIMEUSERENTRY - datetime user input • OPTION - single option selection • OPTIONS - multiple options selection • RADIO - radio input • BOOLEAN - Boolean input
Source Table	Used if the input type is OPTION or OPTIONS. Defines a table name where to get the values options.
Source Type	Used if the input type is OPTION or OPTIONS. Defines a table type (P, C, PX, CX, PP) where to get the values options.
Source Field	Used if the input type is OPTION or OPTIONS. Defines a column where to get the values options.

Value Options Filter By Customer	Defines if the selected customer will be used to filter value options from the source query.
Value Options Filter By Product	Defines if the selected product will be used to filter value options from the source query.
Value Options	Used if the input type is OPTION or OPTIONS and the source is not defined. Defines the value options, separated by comma.
Default Value	Defines an input default value.
Value Hint	Defines an input value hint.
Read Only	Defines if the input is read only.
Required	Defines if the input is required.
Level	Defines where to display the input. Supported values are: Header, Line.
Customer Filter	Defines if the input value can be passed to a customer filter logic as a parameter.
Product Filter	Defines if the input value can be passed to a product filter logic as a parameter.
Customer Filter Operator	Provides the operator for Customer Filter. Supported values are: • OP_EQUAL (default for other supported input types) • OP_NOT_EQUAL • OP_LESS_THAN • OP_GREATER_THAN • OP_LESS_OR_EQUAL • OP_GREATER_OR_EQUAL • OP_LIKE • OP_ILIKE • OP_IN (default for input type is OPTIONS) • OP_NOT_IN
Product Filter Operator	Supports the operator for Product Filter. Supported values are: • OP_EQUAL (default for other supported input types) • OP_NOT_EQUAL • OP_LESS_THAN • OP_GREATER_THAN • OP_LESS_OR_EQUAL • OP_GREATER_OR_EQUAL • OP_LIKE • OP_ILIKE • OP_IN (default for input type is OPTIONS) • OP_NOT_IN

Outputs Configuration

This CPQ Outputs Configuration is obsolete from version 2.0.

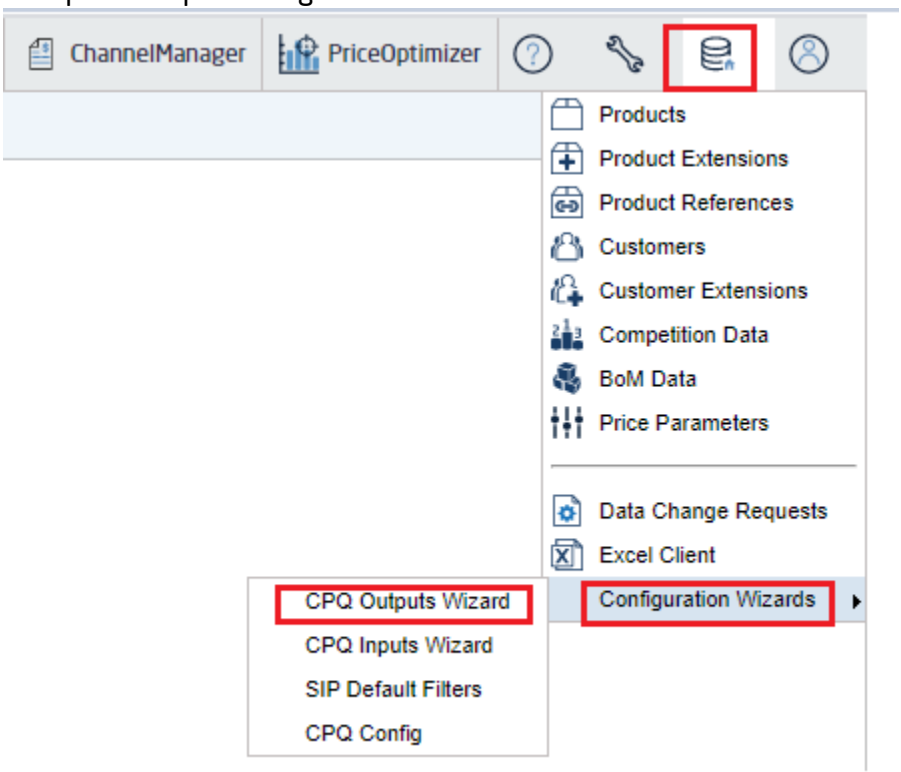
The CPQ Outputs Configuration is used to define configurations for a CPQ output.

Details

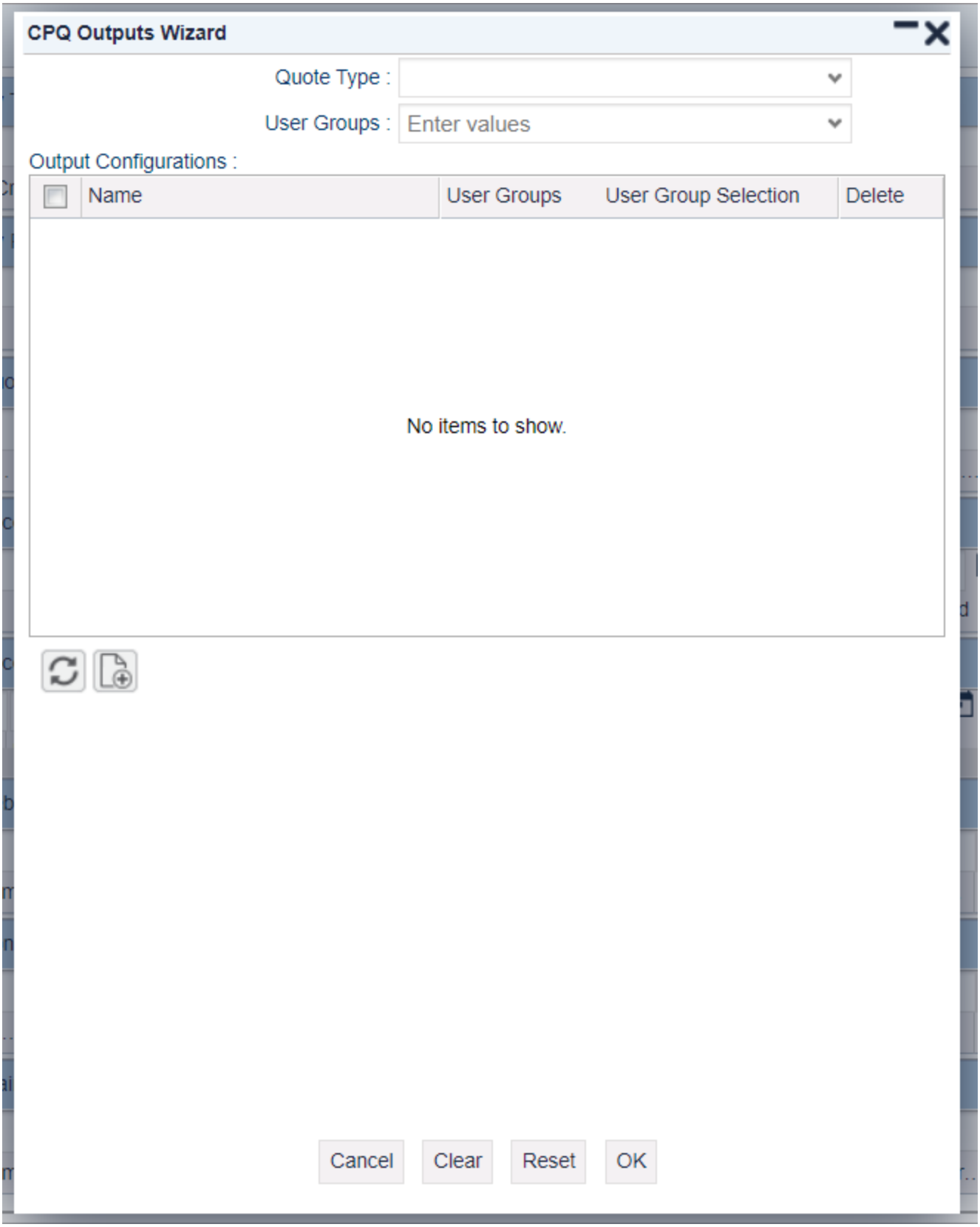
Name	Description
Name	Name of the output to configure.
User group	Defines a user group for the output.

Configuration Wizard

Configuration wizard makes the output configuration easier by providing a user interface for selecting an output and configurations. The CPQ Package defines a logic for the configuration wizard and configuration wizard executor. You need to define the configuration wizard in Administration > Configuration Wizards Admin where you add a new wizard. After the wizard is created, you can use it to set up the output configuration:



The wizard interface helps create/update/delete output configurations:



Options:

- **Quote Type** – If the CPQ logic is provided under another quote type rather than the default one, you need to select the correct quote type.

- **User Group** - Select a user group to apply.
- **Output Configurations** - List of configured outputs.
  - **Name** - Name of the CPQ output (a dropdown where you can select).
  - **User Group** - If you need to define a user group for that particular output.
  - **User Group Selection** - Defines if you want to merge or override the user group selected above.
    - **Merge** - Merges the user groups from the User Group input and the one defined in the row.
    - **Override** - Skips the user group defined in the row and uses the values from the User Group input.
    - **Leave empty** - Does nothing, uses the value in the row.
  - **Delete** - Defines if you want to delete the configurations.

Currency Conversion Configuration

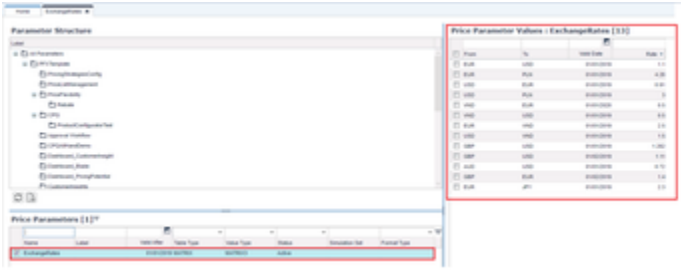
The Currency Conversion Configuration is used to convert currency based on the customer currency selection in a Quote header. If your data (price, cost, discount...) contains information about currency, it can be used in the CPQ Package. The currency conversion will be applied based on your provided conversion rate.

Conversion Rate Lookup

The ccy data source is the default source for your conversion rates. You can also define your custom conversion rates. CPQ supports fetching the conversion rate from a Price Parameters (PP) table. The configuration is:

Name	Description
ppExchangeRate	Name of a Price Parameters table which contains conversion rates
ppExchangeRateFromCurrency	From Currency column
ppExchangeRateToCurrency	Target Currency column
ppExchangeRateValidFrom	Valid From column
ppExchangeRateValidTo	Valid To column
ppExchangeRateValue	Column contains rate value

For example, you can define you exchange rates in Price Parameters like this:



Then you can configure to read your rates this way:

Price Parameter Values : CPQ Logic Configuration [6]

Name	Value
ppExchangeRateValidFrom	key3
ppExchangeRateFromCurrency	key1
ppExchangeRateValue	attribute1
ppExchangeRateToCurrency	key2
ppExchangeRateValidTo	attribute2
ppExchangeRate	ExchangeRates

Lookup Details Configuration

Lookup	Name	Description
Default Currency	defaultCurrency	Displays the default currency. It will be used if there is no defined currency.
Price List lookup	priceListItemCurrency	Lookup currency column in Price List
Product Prices lookup	priceByCustomerLevelCurrency	Lookup currency column in PX Price By Customer
	priceByCountryLevelCurrency	Lookup currency column in PX Price By Country
	priceByRegionLevelCurrency	Lookup currency column in PX Price By Region
	priceBySegmentLevelCurrency	Lookup currency column in PX Price By Segment
Exception Discount PP	Attribute4	Currency read from predefined column
Standard Discount PP	ppStandardDiscountCurrency	Lookup currency column in PP CPQStandardDiscount
Cost lookup	pxCostCurrency	Lookup currency column in PX cost data
Price Competitors PX	Attribute4	Currency read from a predefined column
Competitor Data	Currency	PCOMP - Currency read from a predefined column
Price Guidance PX	Attribute5	Currency read from a predefined column
Price Guidance from Price Optimizer model	poPriceGuidanceCurrency	Lookup currency column in PriceOptimizer for Price Guidance
Query Currency in DM	dm_query_currency	By default, when using a conversion rate from the ccy data source, currency

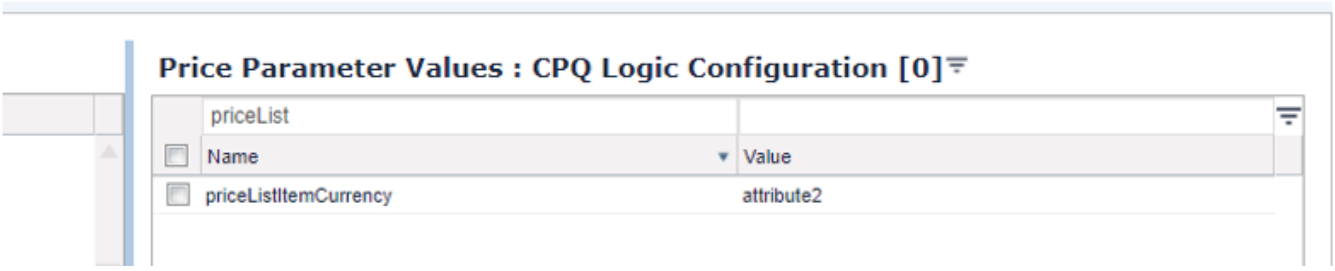
		conversion for historical data will be done by setting the currency option for the query. If your Datamart is not set up for currency, the conversion will not be applied. If you set this configuration to 'false', the conversion will always be applied in the Groovy code, so the problem can be fixed.

For Product Prices Lookup, it requires you to change the attribute size when adding the currency column. Please keep in mind that it may lose your data after each deployment because the original attribute size (3) will be deployed again. Please back up first in this case.

How It Works

In the price list item, there is a specified currency, for example, when using a price list that was created by Accelerate Price Setting Package (PSP):

Then you can define the currency of your product's price in CPQ using the 'priceListItemCurrency' configuration:



We use the same definition for other money data used in CPQ, such as Standard Discount, Exception Discount, Price Guidance and Price Competitors, etc.

Sample configuration:

Price Parameter Values : CPQ Logic Configuration [11]

Name	Value
customerCurrency	attribute12
defaultCurrency	GBP
ppExchangeRateFromCurrency	key1
ppExchangeRateToCurrency	key2
priceListItemCurrency	attribute1
pxCostCurrency	attribute4
priceByCustomerLevelCurrency	Currency
priceByCountryLevelCurrency	attribute4
priceByRegionLevelCurrency	attribute4
priceBySegmentLevelCurrency	attribute4
ppStandardDiscountCurrency	attribute4

If there is no conversion rate found, the value 0 will be displayed for historical data, Price Guidance and <empty> for others.

Matrix Price List Lookup

In a Matrix Price List, the product price can differ based on secondary keys.

You need to define a logic (we call it "keys provider") that specifies which secondary keys you want to use for price lookup. Your logic can be configured by setting 'matrixPriceListSecondaryKeyProvider' in CPQ configurations.

Your keys provider must be a Groovy library and accept a list of product IDs as parameter. We expect it to return a map of desired secondary keys per product.

You can also access the current quote view in your provider by using 'quote' variable, e.g. quote. uniqueName.

```
Map mySecondaryKeys(List skus){
  def mySku = skus[0]
  api.logInfo("quote-$quote.uniqueName", mySku)
  return [(mySku): ["key1", "key2"]]
  }
```

Keys provider runs in the quote pre-phase context.

If you use Price Setting Package to create the matrix price list, there is a predefined volumes breakdown key provider that can be used.

CPQ Package Version 2.0

- Design
- Configuration Per Quote Type
- Custom Logics in CPQ
- Error Handler

Design

The main requirement of the CPQ Package version 2.0 is to allow custom code from the library to be executed by CPQ logics.

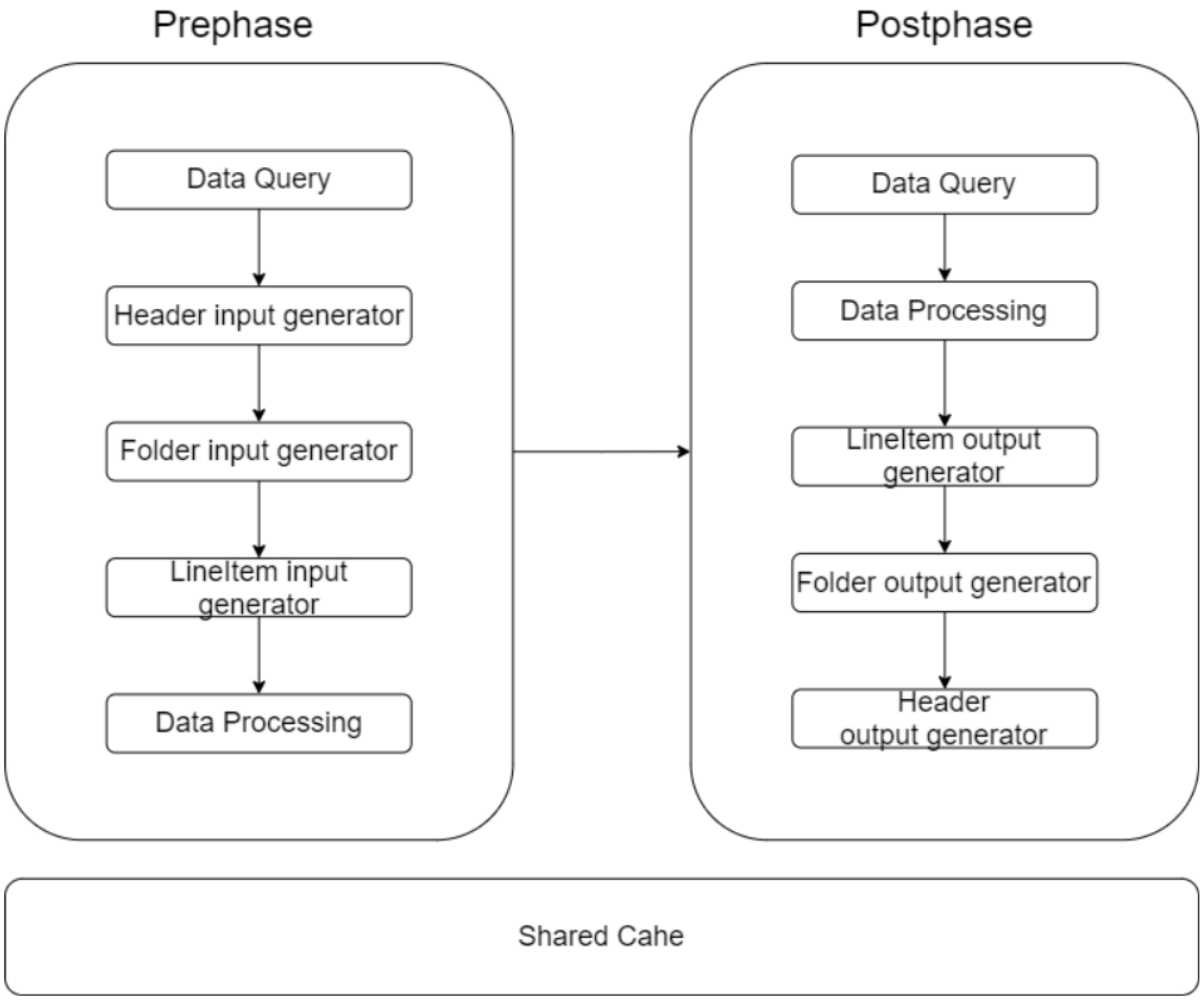
In this section:

- Overview
- Per Quote Type Configuration
- Task Runner
- Data Sharing

Overview

CPQ Package version 2.0 has the following functionality:

- Allows you to run custom libraries on the quote header. The libraries can run both in pre-phase and post-phase.
- There are **task runners** for data query, data processing, input, and output generator
- Allows you to set the priority of tasks for each runner. In addition, some tasks can depend on the results of other tasks.
- Existing logics are split into multiple modules and plugins as CPQ default configurations.



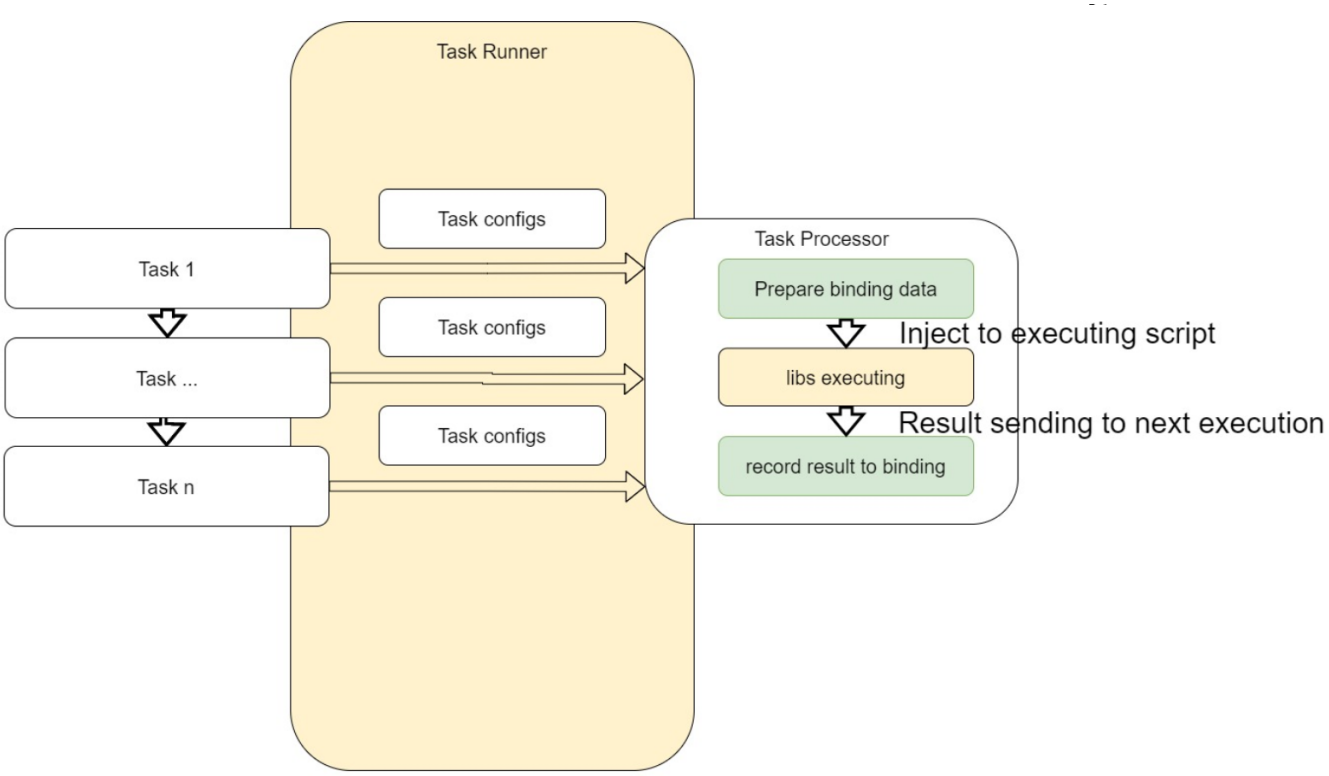
Per Quote Type Configuration

Each quote type defines a name of the PP table which is the main entry for other configurations; the table contains other configurations:

- **Configurations** - Define the PP table which contains core configurations used by CPQ.
- **Processing_Configurations**- Define the PP table which contains configurations for data query and data processing.
- **Input_Output_Configurations** - Define the PP table which contains configurations for inputs and outputs generator.

Task Runner

Task Runner is an element in the header and line item logic that allows running the custom library and handles data sharing between a task and other Task Runners.



Tasks define the library path that contains the logic for the execution.

Task Processor is a library that is responsible for running the library defined in the task, recording the result, and sharing it with other tasks.

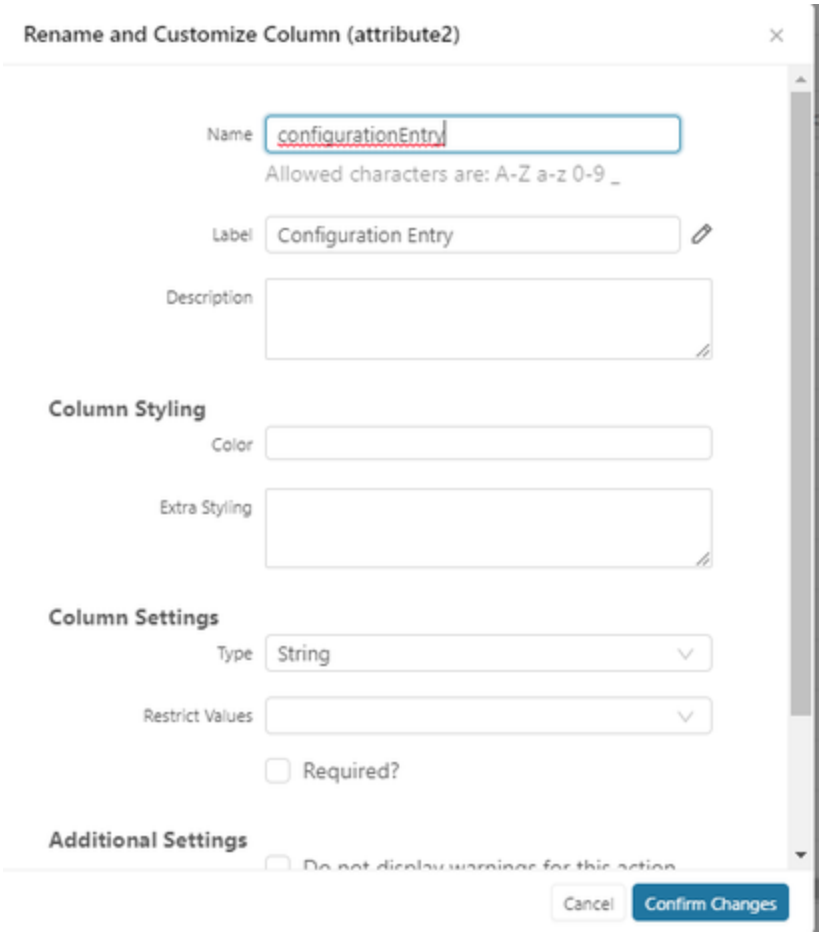
Data Sharing

Data are shared from pre-phase to post-phase using the shared cache.

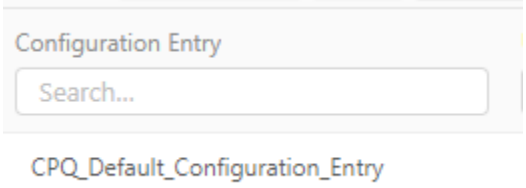
Configuration Per Quote Type

To create a per quote type configuration:

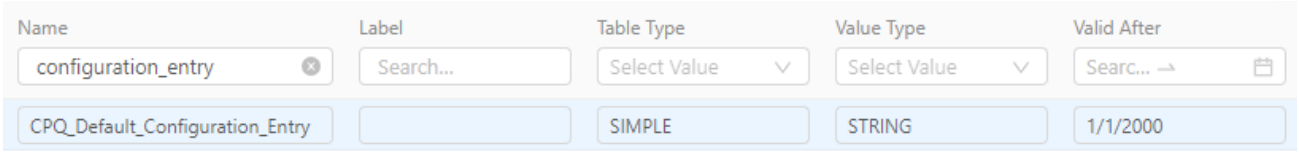
1. Open Quote Types.
2. Rename a column and set the name to “configurationEntry”.

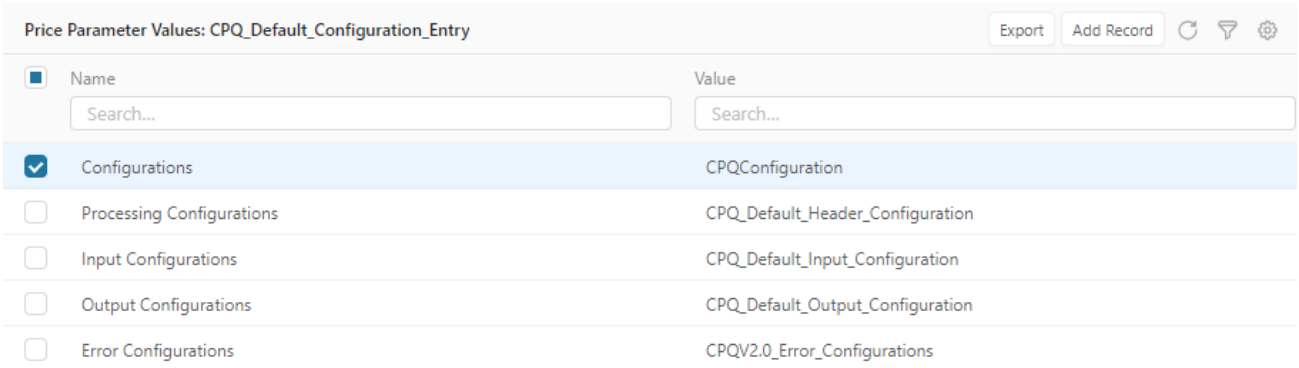


3. Set the name of the PP table which contains the configuration entries.



4. Configuration entry PP structure:





- Configuration entry setting:
  - **Configurations** - PP table name which contains CPQ configurations.
  - **Processing_Configurations** - Define the PP table which contains configurations for data query and data processing.
  - **Input_Output_Configurations** - Define the PP table which contains configurations for inputs and outputs generator.
  - **Error_Configurations** - PP table which contains configurations for errors.
  - **Competitor_PX_Name** - PX table which contains competitor prices
  - **Customer_Cash_Discount_CX_Name** - CX table which contains customer cash discount values
  - **Exception_Discount_PP_Name** - PP table which contains exception discount values
  - **Input_Configuration_PP_Name** - PP table which contains custom input configurations in version 1 (deprecated from version 2.0)
  - **Price_By_Country_PX_Name** - PX table which contains price by country values
  - **Price_By_Customer_PX_Name** - PX table which contains price by customer value
  - **Price_By_Region_PX_Name** - PX table which contains price by region values
  - **Price_By_Segment_PX_Name** - PX table which contains price by segmentation values
  - **Price_Guidance_PX_Name** - PX table which contains price by segmentation values
  - **Product_Cost_PX_Name** - PX table which contains product costs
  - **Standard_Discount_PP_Name** - PP table which contains standard discount values
  - **Advance_Configuration_Key** - Advanced property (AP) key which defines AP keys to create configuration from.

From now, the system knows where (from which tables) to get configurations and data for your Quote Type.

Custom Logics in CPQ

You can add your custom calculations (at header and line item level) to CPQ without modifying the core logic/library.

There are two PP tables defining the structure of the quote logic which can be configured in the configuration entry Price Parameter table:

- **Processing_Configurations** – PP table which contains header elements configurations. The default table name is **CPQ_Default_Header_Configuration**.
- **Input_Output_Configurations** – PP table which contains line item elements configurations. The default table name is **CPQ_Default_Input_Output_Configuration**.

Your custom logic should be written as a library so that it can be configured and run by CPQ.

If you want to use a custom PP table, ensure that its structure is the same as defined in default PP tables.

Further in this section:
- Custom Header Logic
- Custom Input & Output logic
- Predefined Variables

Custom Header Logic

This Price Parameter table defines custom logics that can be run on the header level.

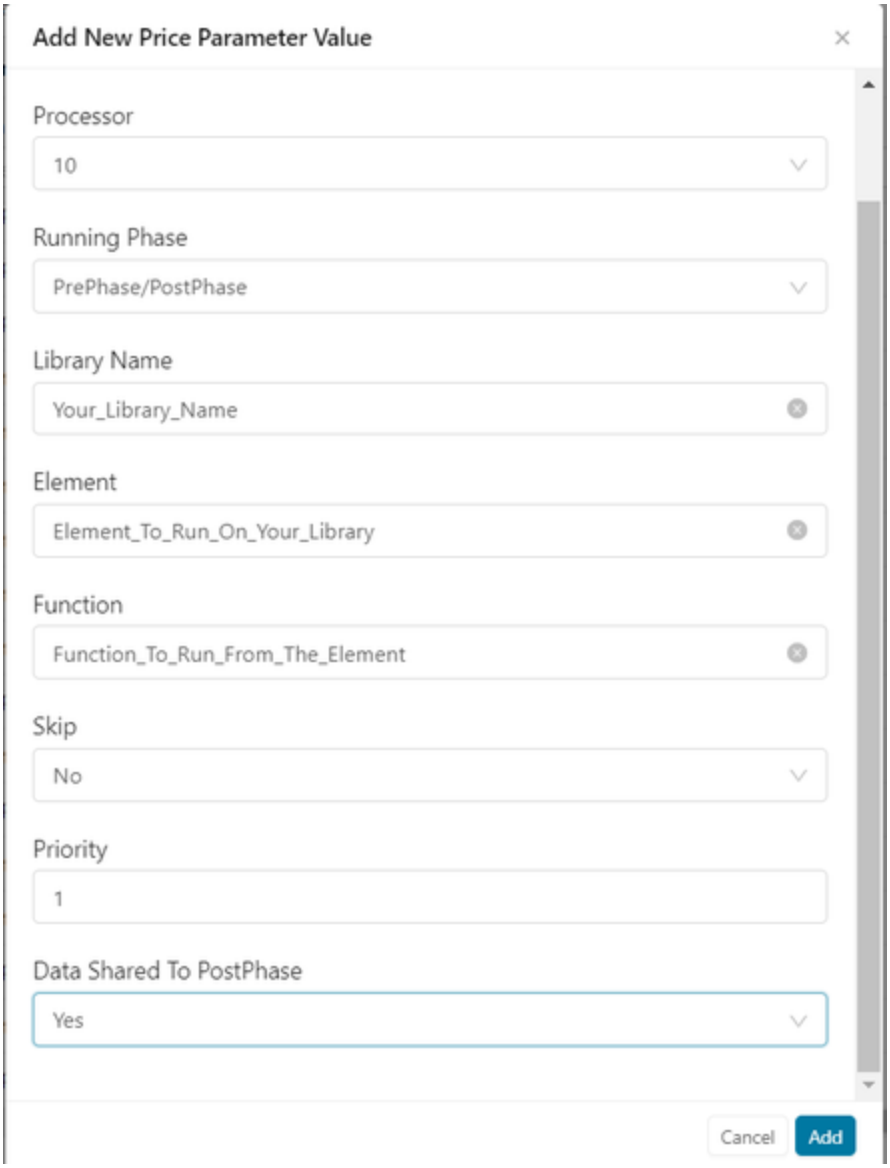
Default Price Parameter table: **CPQ_Default_Header_Configuration**

In this section:
- Configurations
- Custom Logic

Configurations

You can set the following options:

- **Name** – Name of your element. This can be used to access returned data from another element.
- **Processor** – Defines an element in the CPQ header to run your element. There are 2 processors available: **Data Query** and **Data Processing**.
- **Running Phase** – Select here the phase in the header to run your element. Can be pre-phase, post-phase or both.
- **Library Name** – Name of the library which contains the calculation for your element.
- **Element** – Element from the library which contains the calculation for your element.
- **Function** – Function name from the library element which contains the calculation for your element.
- **Skip** – Skips executing your element. Can be Yes/No.
- **Priority** – Defines priority of running within the processor.
- **Data Shared to Post-phase** – Defines if your element result can be accessed from the post-phase. This can be shared only when your running phase is pre-phase. Data is shared between phases using the shared cache.



Custom Logic

Your logics should be defined in the Pricefx Groovy library.

You need to provide the “library name”, “element name”, and “function name” of your library.

Your function will run in the quote header context.

There are predefined objects that you can use in your function:

- **elements** - Map of processed elements value.
- **quoteProcessor** - Pricefx predefined object, this is auto-binding to your function.

Your function should not take any arguments and should return the value if you need to access the result from other elements.

Custom Input & Output logic

This Price Parameter table defines custom logics that can be run on the line item level.

Default Price Parameter table: **CPQ_Default_Input_Output_Configuration**

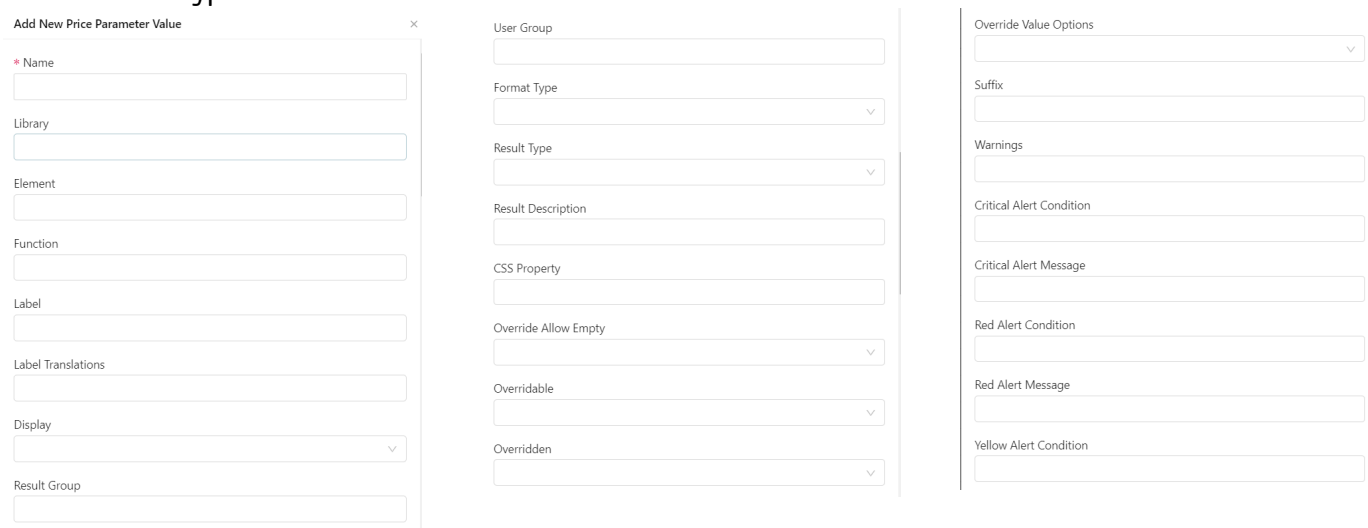
In this section:

- Configurations
- Custom Logic

Configurations

You can set the following options:

- Generation Type - Available options are:
  - Header Input
  - Header Output
  - Line Item Input
  - Line Item Output
  - Folder Input
  - Folder Output
- Name - Element name, will be used as the name of the generated output.
- Label - Element label, will be used as the label of the generated output.
- Input Type - The type of input to generate.
- Output Type - The type of output to generate.
- Processor - Select here an element on the CPQ line item level to run your element. There are 30 elements out of the box that can be configured to run your logic.
- Library Name - Name of the library which contains the calculation for your element.
- Element - Element from the library which contains the calculation for your element.
- Function - Function from the library element which contains the calculation for your element.
- Library runner - Defines in which phase the library should be run. The options are: Input, Output, or Both.
- Skip
- Priority
- Result group
- Format type



Custom Logic

Your logics should be defined in the Pricefx Groovy library.

You need to provide the “library name”, “element name”, and “function name” of your library.

There are predefined objects that you can use in your function:

- elements - Map of processed values.
- header - Map of processed values in header processing.

Your function should not take any arguments and should return the value of your output.

Predefined Variables

There are some predefined variables that are already initialized and shared across custom logics.

- **quoteProcessor** - Pricefx predefined processor available in the quote header context and already bound to your custom library as a predefined variable. It can be accessed from all contexts.
- **elements** - Map containing previous logic results. It can be accessed from all contexts.
- **header** - Map containing header processed results. This variable can be accessed in the inputs and outputs generation context.
- **lineItem** - Current processing line item. This variable can be accessed in the inputs and outputs generation context.
- **inputs** - Current processing line item input values. This variable can be accessed in the inputs and outputs generation context.
- **inputConfig** - Current processing input configuration. We can use this to update the current input config. This variable can be accessed in the inputs generation context.
- **outputs** - Current processing line item outputs values. This variable can be accessed in the inputs and outputs generation context.
- **outputConfig** - Current processing output configuration. We can use this to update the current output config. This variable can be accessed in the outputs generation context.

The predefined variable can be accessed either by its name or TaskProcessor.

- name

```
function myElement(){
    // previous element name from PP table is “myCustomQuantity”
   def quantity = elements.myCustomQuantity

    return quantity
}
```

- TaskProcessor

```
function myElement(){
    // previous element name from PP table is “myCustomQuantity”
   Map elements = libs.CPQ_SharedLib.TaskProcessor.
```

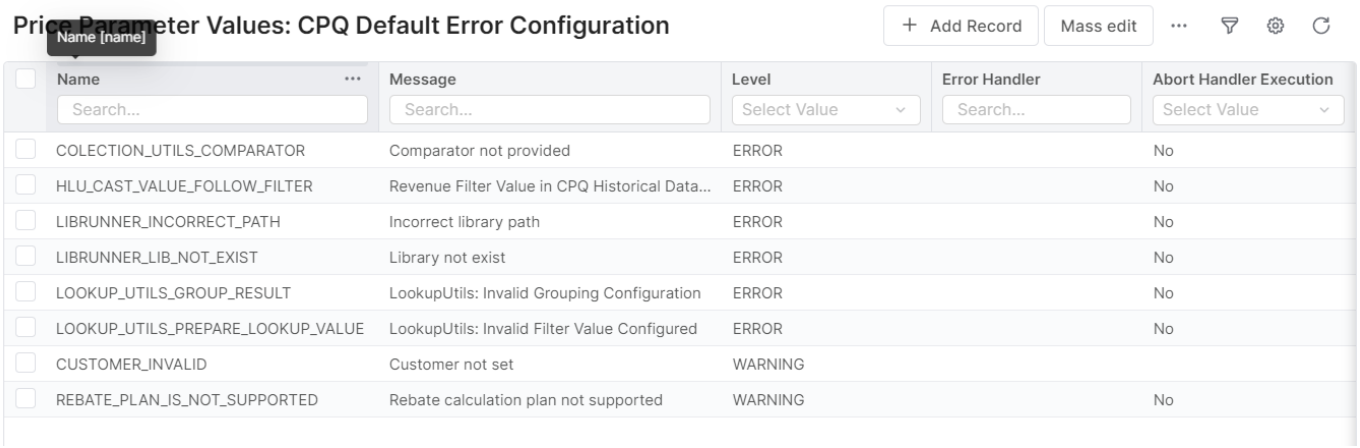
```
getProcessingBinding("elements")
   def quantity = elements.myCustomQuantity

    return quantity
}
```

Error Handler

Starting with version 2.0, there is a simple error handler for configuring error levels and messages.

The error configurations can be configured in a PP table defined in Configuration Per Quote Type.

The default PP table for error handler named **CPQ_Default_Error_Configuration** will be used if the configuration is not specified on the quote type level.



Usage

In your logic, you just need to throw an exception with the key configured in the PP table.

Then you can use the PP table to configure the level, error message, etc.

```
def yourLogic(){
  try{
    // your logics
  }catch(e){
    api.throwException("ERROR_CONFIG_KEY")
  }
}
```

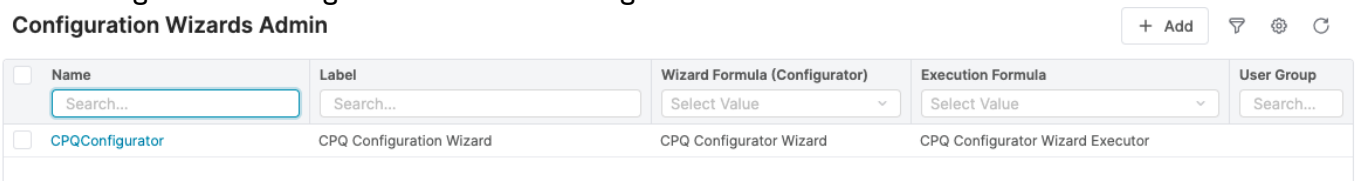
CPQ Configuration Wizard

The CPQ Configuration Wizard makes the package configuration easier by providing a user-friendly interface for the setup (instead of manual changes in Price Parameters).

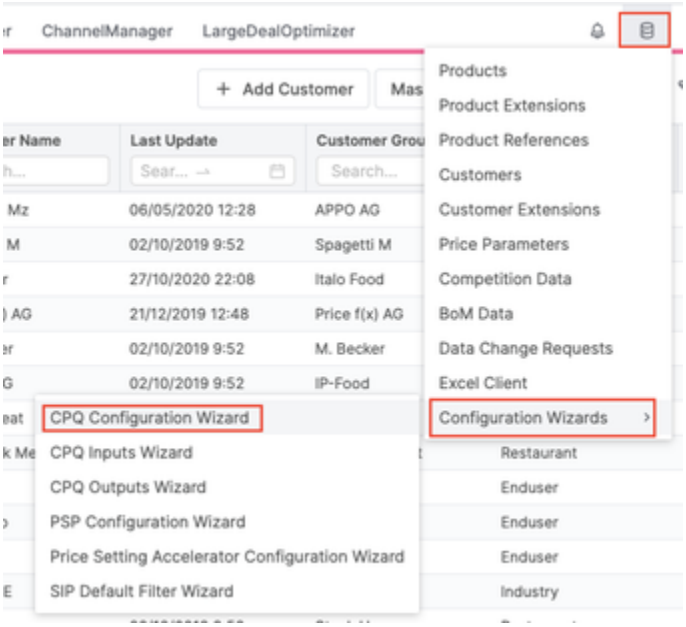
Configuration Wizards Admin

In the CPQ Package, we defined a default logic for the configuration wizard and wizard executor.

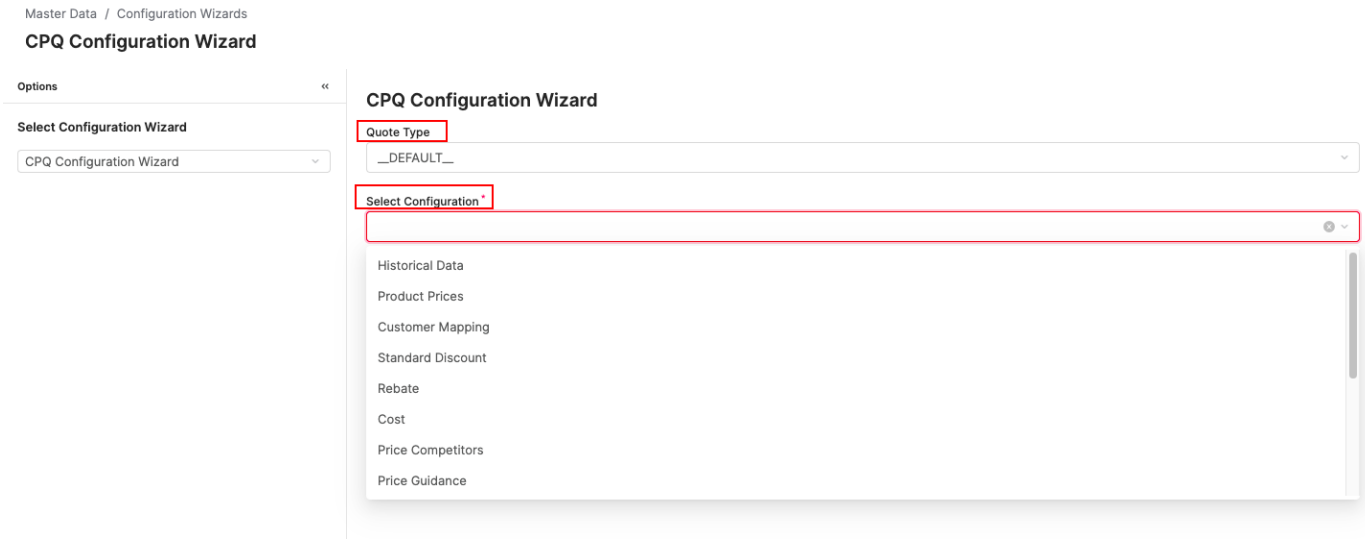
Go to Configuration Configuration Wizards Configuration Wizards Admin:



The Configuration Wizards Admin is already configured after deployment, you can use it for the configuration. Go to Master Data Configuration Wizards select CPQ Configuration Wizard.



Detail



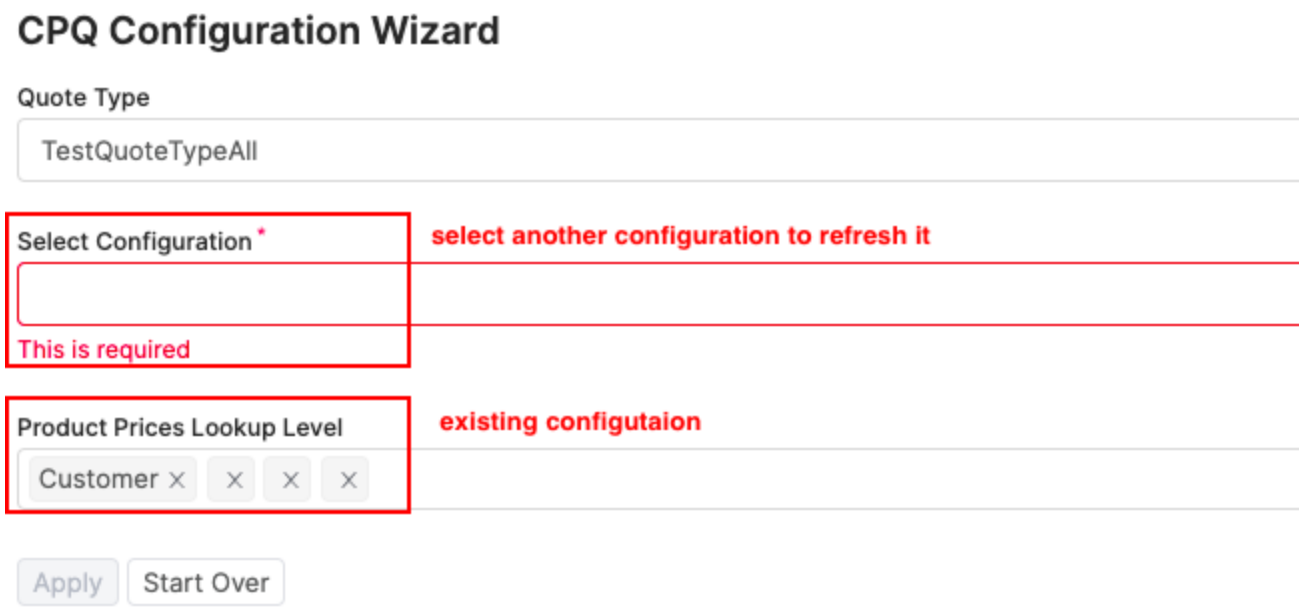
- **Quote Type** - Allows you to select which quote type is configured.
- **Select Configuration** - List of configurations:
  - Historical Data
  - Product Prices
  - Customer Mapping
  - Standard Discount
  - Rebate
  - Cost
  - Price Competitors
  - Price Guidance
  - Publishing Templates
  - Currency
  - Header Processing (version 2.0)
  - Inputs (Deprecated)
  - Inputs / Outputs (version 2.0)
  - Error (version 2.0)
  - Others

For details see:

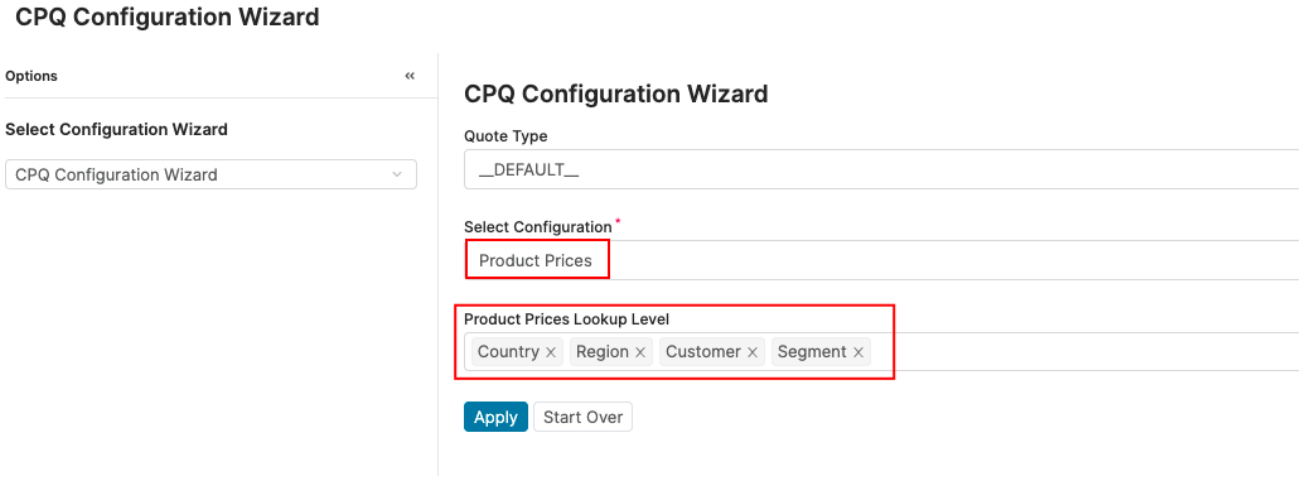
- CPQ Package Configurations
- Version 2.0

Note:

- Quote Type
  There is a known issue when you select another quote type in the wizard, the existing configuration of previous quote type is still displayed. You need to select another configuration in <Select Configuration> to refresh the wizard.



- Product Prices configuration wizard



In Product Prices Lookup Level, it will apply the priority by the order you selected.

CPQ Inputs Configuration - CPQ Outputs Configuration in Wizard

- CPQ Inputs Wizard
  - Details on "Create"
  - Details on "Edit" and "Delete"
- CPQ Outputs Wizard

Price Parameters

⚠️ They are both obsolete from version 2.0, as they were replaced by CPQ Configuration Wizard.

- CPQInputConfiguration

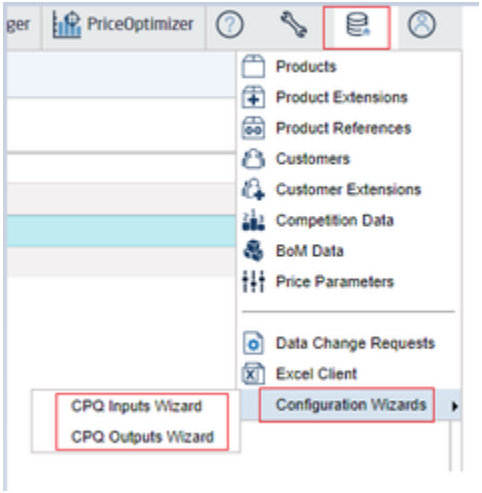
- CPQOutputConfiguration

Setup

1. In the CPQ Package you need to define a logic for the configuration wizard and wizard executor first. Go to Administration Configuration Wizard Admin and define Name, Label, Wizard Formula (Configurator), Execution Formula.

   To provide a better guidance, we created an example label with "CPQ Inputs Wizard" and "CPQ Outputs Wizard".

2. After the Configuration Wizard Admin is created, you can use it for the configuration. Go to Master Data Configuration Wizards choose which wizard you want to use.

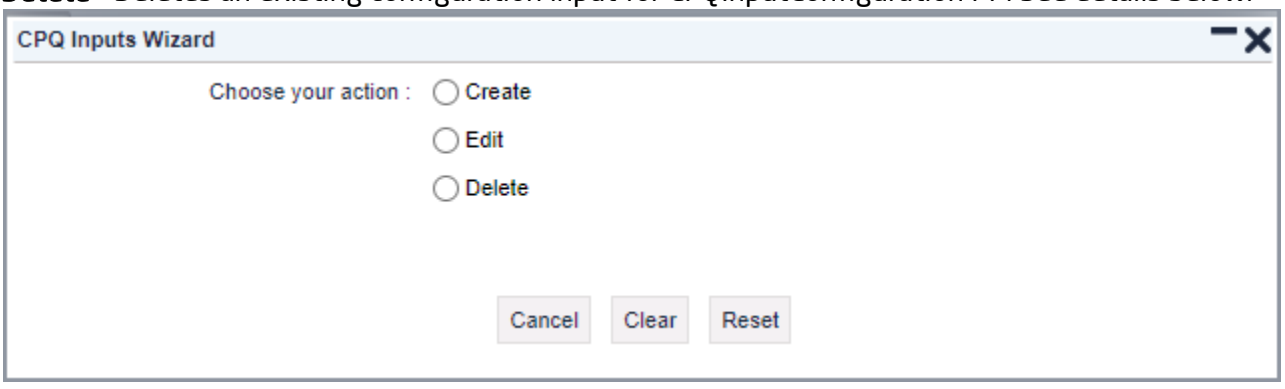


How It Works

CPQ Inputs Wizard

The CPQ Inputs Configuration defines input fields which should be available in the Quote Header and defines if the field is also used in the Customer or Product filter. The CPQ Inputs Wizard provides you three options when you open it:

- **Create** - Creates a new configuration input for CPQInputConfiguration PP. See details below.
- **Edit** - Edits an existing configuration input for CPQInputConfiguration PP. See details below.
- **Detete** - Deletes an existing configuration input for CPQInputConfiguration PP. See details below.

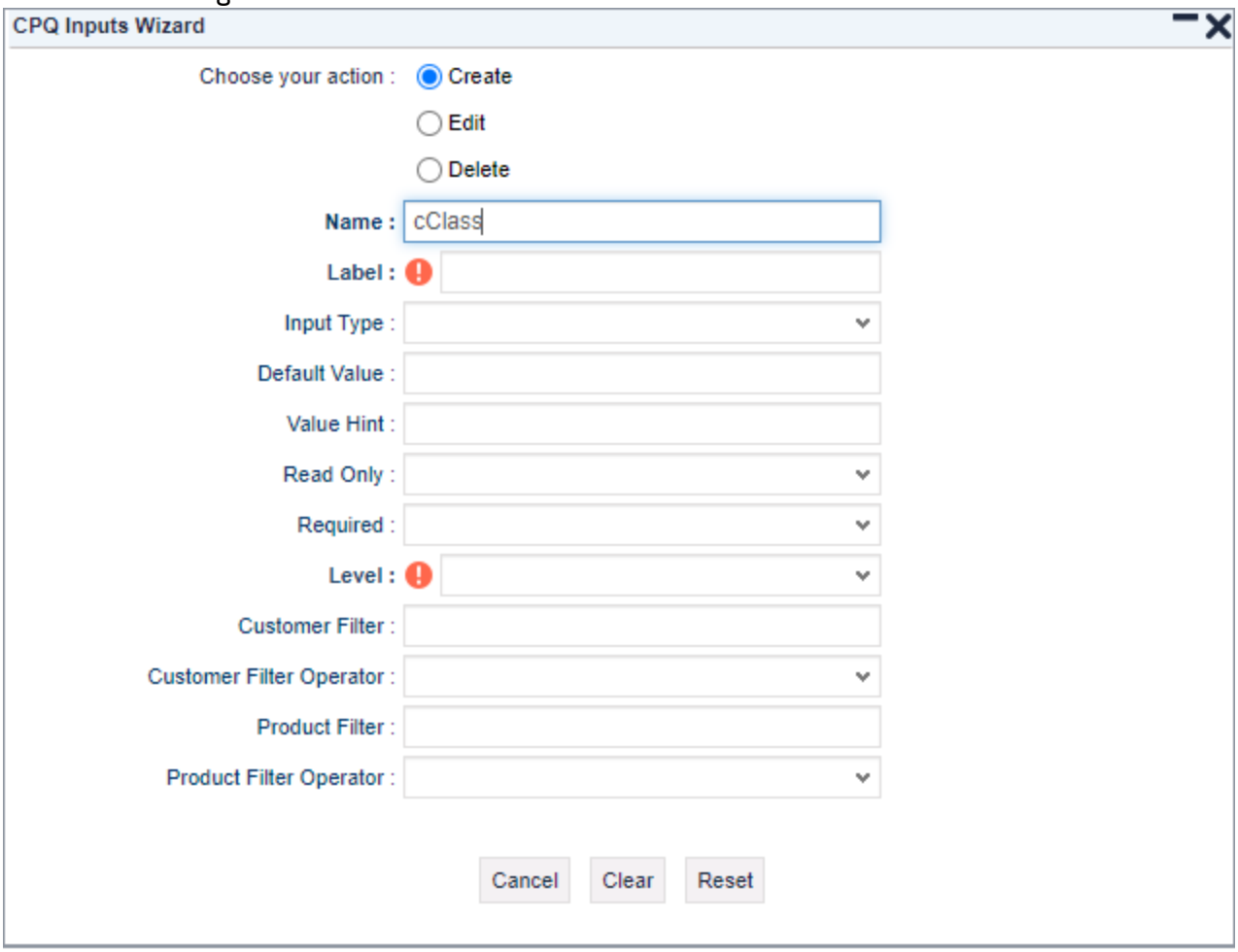


Details on "Create"

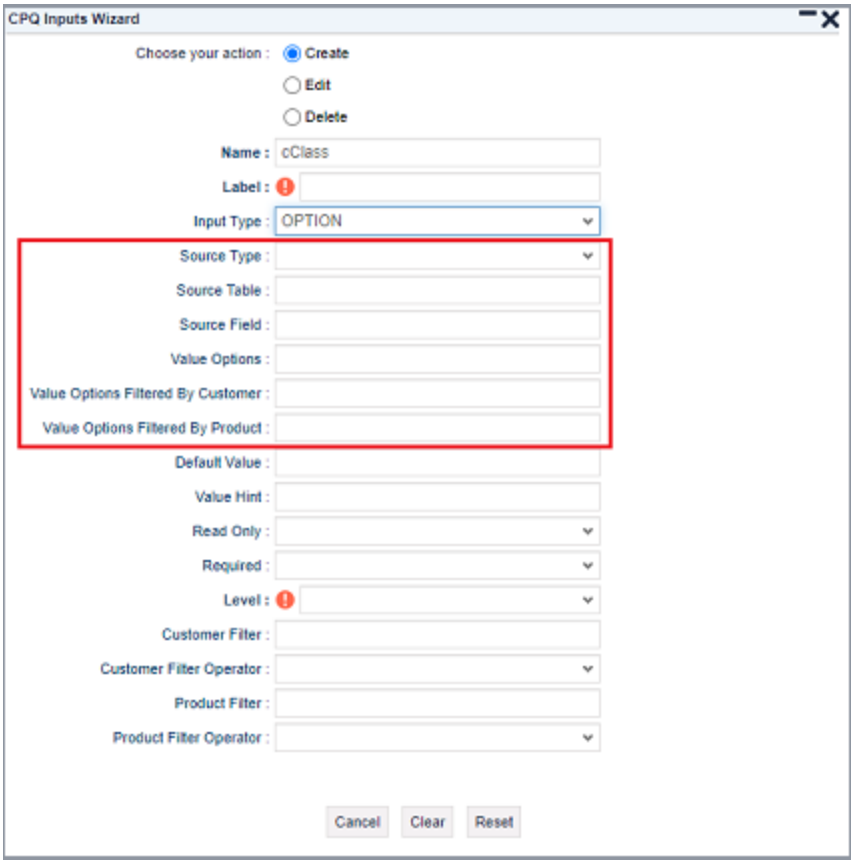
Enter a **Name** (preferably with no spaces, e.g. cClass, pGroup). Then press the **tab** key to apply the setting. If the name already exists, you will be notified.

After the name check, the next configuration is shown (after pressing the tab key): **Required fields**. You need to fill in Name, Label, Level (if you do not fill all required fields, the OK button will not be displayed).

The default configuration:



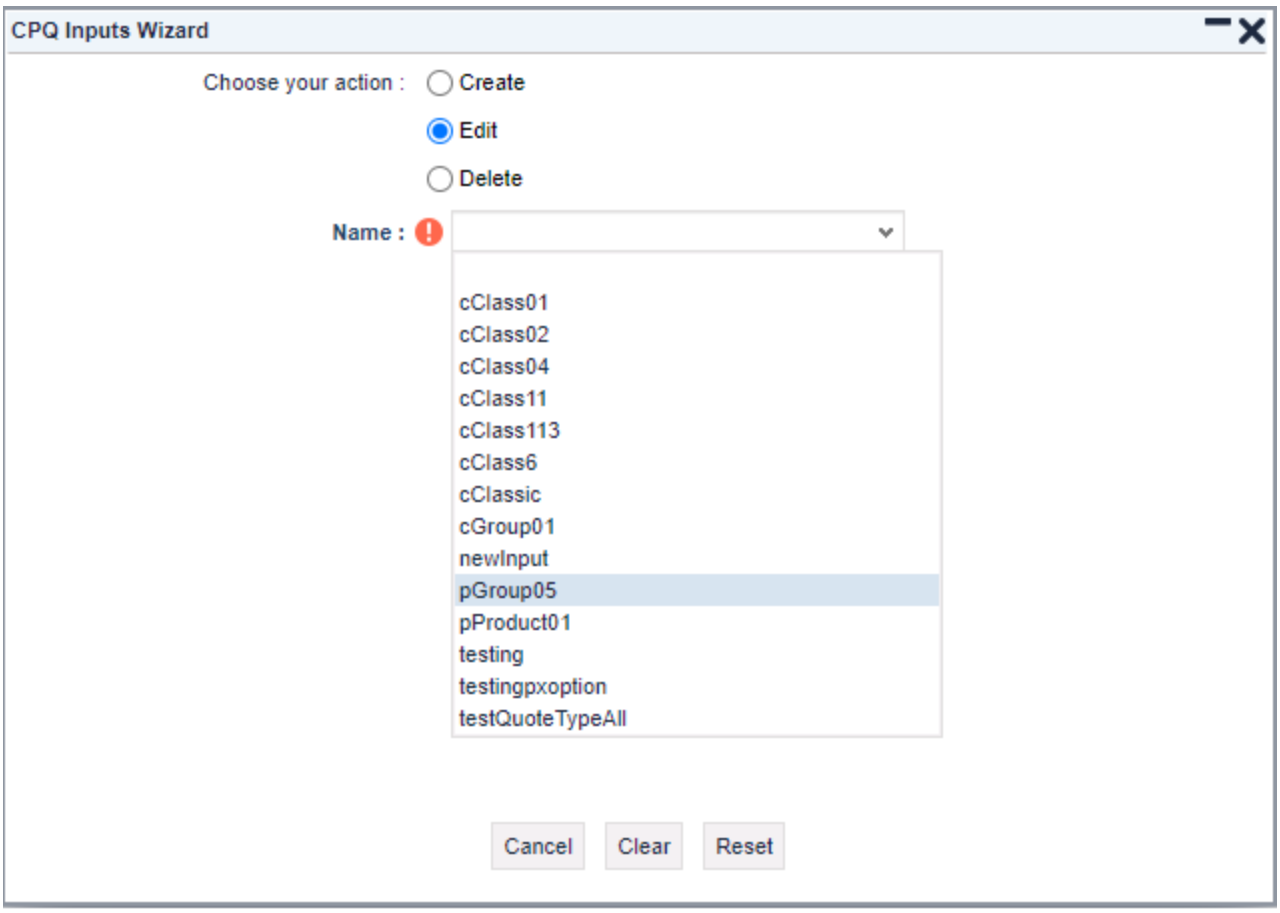
If you choose OPTION/ OPTIONS as Input Type, some hidden input fields will be displayed:



The Source Table (with Source Type: P, C, PX, CX, PP) and Source Field are sorted in the ascending (A-Z) alphabetical order. The displayed lists in the Source Table and Source Field will be 'name', not 'label'. If you do not define a label for fields, the list of Source Fields will not be displayed.

Details on "Edit" and "Delete"

Select the item to edit or delete from the **Name** drop-down list.



After you make your changes and click the OK button, this input configuration will be added automatically into the CPQInputConfiguration PP. You need to click Refresh in the PP table and then the Recalculate button in the Quote to apply it.

Example configurations:

Price Parameter Values : CPQ Inputs Configuration [8]

Name	Label	Input Type	Source Table	Source ...	Source Field	Value Options	Default Value	Value Hint	Read Only	Required	Level	Customer Filter	Product Filter	Value Options Filtered By Customer	Value Options Filtered By Product	Customer Filter Operator	Product F
cClass04	Customer Type	OPTION		C	Customer Type				No	Yes	Header	Yes				OP_NOT_EQUAL	
valueOption01	Sales Org	OPTION		C	Sales Org						Header			customerId			
cClass6	Customer Group	OPTIONS		C	Customer Type						Header	Yes				OP_IN	
valueOption03	Customer Extension	OPTION	ValueOptionsFilteredByCustomer	CX	Attr 2						Header			attribute1 = attribute3			
pGroup01	Product Group	OPTION		P	Product Group						Header		Yes				
pType	Product Class	OPTIONS		P	Product Class						Line				sku		
pPP	Product Type	OPTIONS	TestFolderQuoteApproval	PP	folderBeverages						Line				name = attribute4		
pLifeCycle	Product Life Cycle	OPTION	PriceByCountry	PX	Price						Header			attribute1=attribute5			

For more details on configuration see CPQ Package Configurations. You can use the name and label instead of the technical name.

+ ***Customer Filter / Product Filter***: When you use this filter, the logic will get the value of these inputs and pass them to the filter logic. Then it will be applied to the Customer field on the header level or to the Product on the line item level.

How to configure:

- Yes/No
- or Customer/Product attribute based on your input Source Field

We provide the default implementation of customer/product filters for Source Type = P/C and Input Type = OPTION/OPTIONS. But you still need to build your own filter logic for the remaining source types and

input types to work.

Customer/ Product Filter Operator with supported values (drop-down):

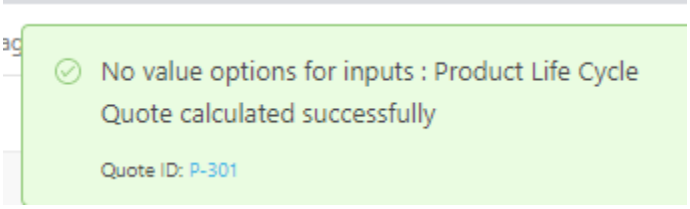
- OP_EQUAL (default for other supported input types)
- OP_NOT_EQUAL
- OP_LESS_THAN
- OP_GREATER_THAN
- OP_LESS_OR_EQUAL
- OP_GREATER_OR_EQUAL
- OP_LIKE
- OP_ILIKE
- OP_IN (default for input type is OPTIONS)
- OP_NOT_IN

+ Value Options Filtered By Customer/ Product (only applies when Input Type is OPTION, OPTIONS): When you select the Customer field on the header level or the Product on the line item level, we will show input values based on that. It is opposed to the Customer / Product Filter.

How to configure:

- If Source Type is P/C you only need to input customerId or sku
- Other Source Types, e.g: attribute of Customer Class (CX) = attribute of Customer Class (C)

There will be the following warning message if the value options are not found:

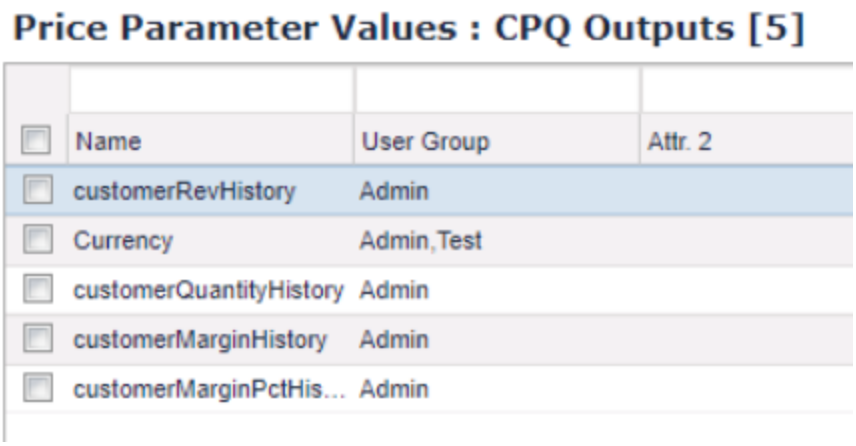


CPQ Outputs Wizard

The CPQ Outputs Configuration can show/hide outputs in the CPQ Accelerators. It allows business users to set what fields are visible in quoting and for which user groups those fields are available. Up to now this had to be set up in the logic itself.

For more details on configuration see CPQ Package Configurations.

Use the CPQOutputConfiguration PP or go to Master Data Configuration Wizards CPQ Outputs Wizard.



This is how you set up the access: in the PP table above, only user belonging to the given user group can see the items.

Note: When you change, update/delete something in the PP table, please remember to recalculate and save your quote first, then the other user group will not see it when they open it. Both the Header and Line Item level will be hidden. If a user has full admin roles, this configuration will not work.

CPQ Package Architecture

- Library Dependency
- Data Structures
- Components

Library Dependency

- ApprovalWorkflow
- FormulaEvaluator

Data Structures

Mandatory Tables

Table Type	Table Name	Mandatory Fields	Optional Fields
Product Extension	ProductCosts	• Product Id • Cost	Contains cost per product which is used in the CPQ cost output element.
Product		• Product Id (SKU) • Label	
Customer		• Customer Id	

		• Name	

Optional Tables

Table Type	Table Name	Mandatory Fields	Optional Fields	Description
Product Extension	PriceGuidance	• Product Id • Target • Floor • Stretch	• Additional dimensions	Contains price guidance per product, included target price, floor price and stretch price.
Product Extension	PriceCompetitors	• Product Id • Competitor • Price	• Target date	Contains a competitor price per product.
Customer Extension	CustomerCashDiscount	• Customer Id • Discount		Contains discount information per customer (customer ID).
Pricing Parameter	StandardDiscount	• Key 1 - Key6 • Discount • Value Type	• Target date	Standard discount configuration. There are six keys to configure to what a discount value belongs. We can also define if the discount value is a percentage or absolute value.
Pricing Parameter	ExceptionDiscount	• Customer (key1) • Product (key2)	• Target date	Exception for the discount configuration. The exception discount is defined for customer/product pairs. Values for the customer and product can be an attribute in the customer/product data or in PX/CX.

		• Discount (attribute1)		
Pricing Parameter	CPQInputConfiguration	• Name • Label • Input Type	• Source table • Source type • Source field • Value options • Default value • Value hint • Required • Level • Customer Filter • Product Filter • Value options filtered by customer • Value options filtered by product	Contains configurations for custom inputs
Pricing Parameter	CPQOutputConfiguration	• Name	• User group	Contains configurations for outputs

Product Extensions

- PriceByCustomer - Contains a price per product per customer which is considered to be a list price of the product.
  - Product Id (SKU)
  - Customer (attribute1): customer ID/number
  - Price (attribute2)
- PriceByCountry - Contains a price per product per country which is considered to be a list price of the product.
  - Product Id (SKU)
  - Country (attribute1)
  - Price (attribute2)

- PriceByRegion - Contains a price per product per region which is considered to be a list price of the product.
    - Product Id (SKU)
    - Region (attribute1)
    - Price (attribute2)
- PriceBySegment - Contains a price per product per segmentation which is considered to be a list price of the product.
    - Product Id (SKU)
    - Segment (attribute1)
    - Price (attribute2)

Components

- Calculation Logic
    - CPQ - CPQ quoting logic
    - CPQ_SharedLib
    - CPQ_Publishing_Template_Data_Provider
    - CPQ_header - CPQ header logic
    - CPQ_header_discounts_configurator - Configurator logic for the discount type input
    - CPQ_CustomerFilter
    - CPQ_ProductFilter
    - HeaderInputs_Configurator
    - LineInputs_Configurator
    - CPQ_Output_Configuration_Wizard
    - CPQ_Output_Configuration_Wizard_Executor
- Publishing Template
    - publishingTemplates_Q_CPQTemplate.docx
- Message Template
    - Quote_approvalRequired_en.html
- Workflow Formula
    - CPQ_Quote - Workflow formula based on the ApprovalWorkflow library, with the data map extended for the quote summary.
- Quote Type
    - CPQ - Dedicated quote type for CPQ, with logic, header logic and workflow logic setup.

Configurator Package Integration

- Product Configurator
- Usage
- Price Settings
- Customer and Product Filter

Product Configurator

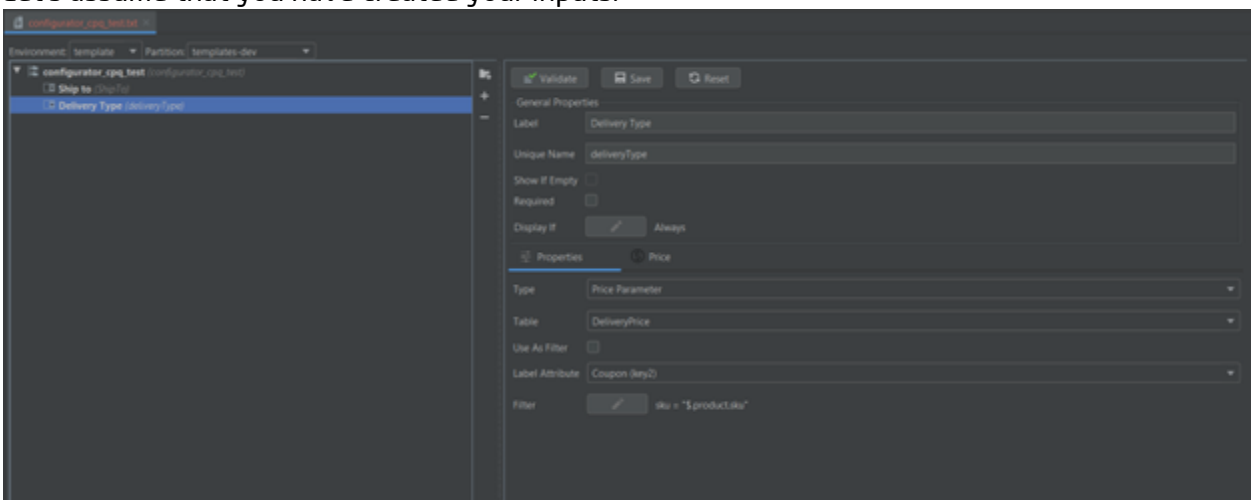
Integration with Configurator Package (https://gitlab.pricefx.eu/accelerators/configurator-package)

Note: It is planned to consolidate it with the existing CPQ Input configuration in the next release.

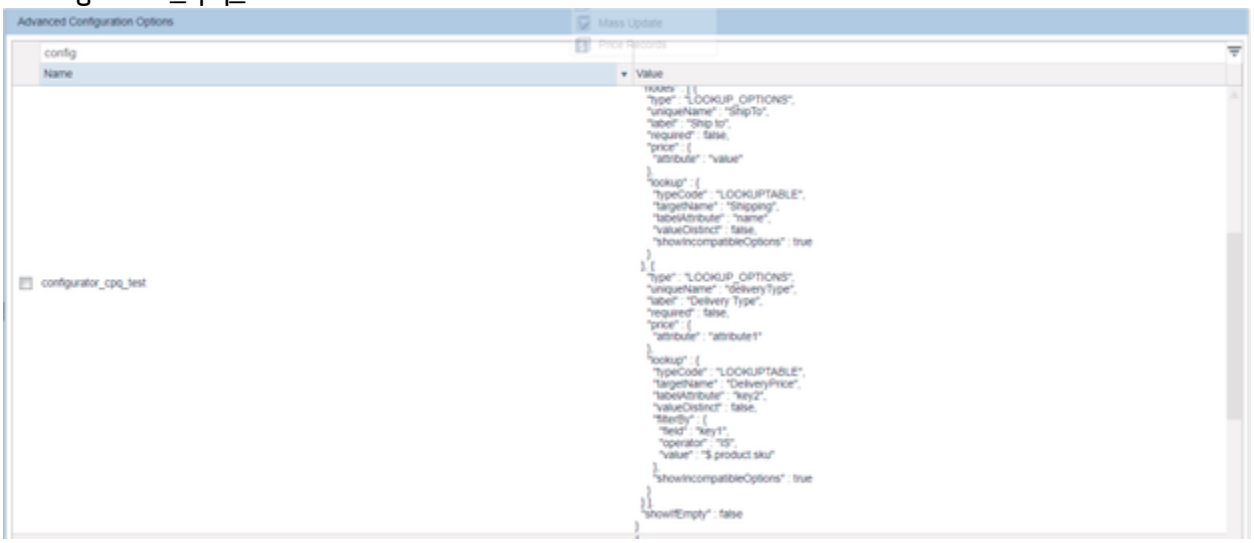
Usage

You can design your input configurators using ConfiguratorDesigner (https://pricefx.atlassian.net/wiki/spaces/KB/pages/2344190596/Configurator+Designer) and use it in CPQ.

Let's assume that you have created your inputs:



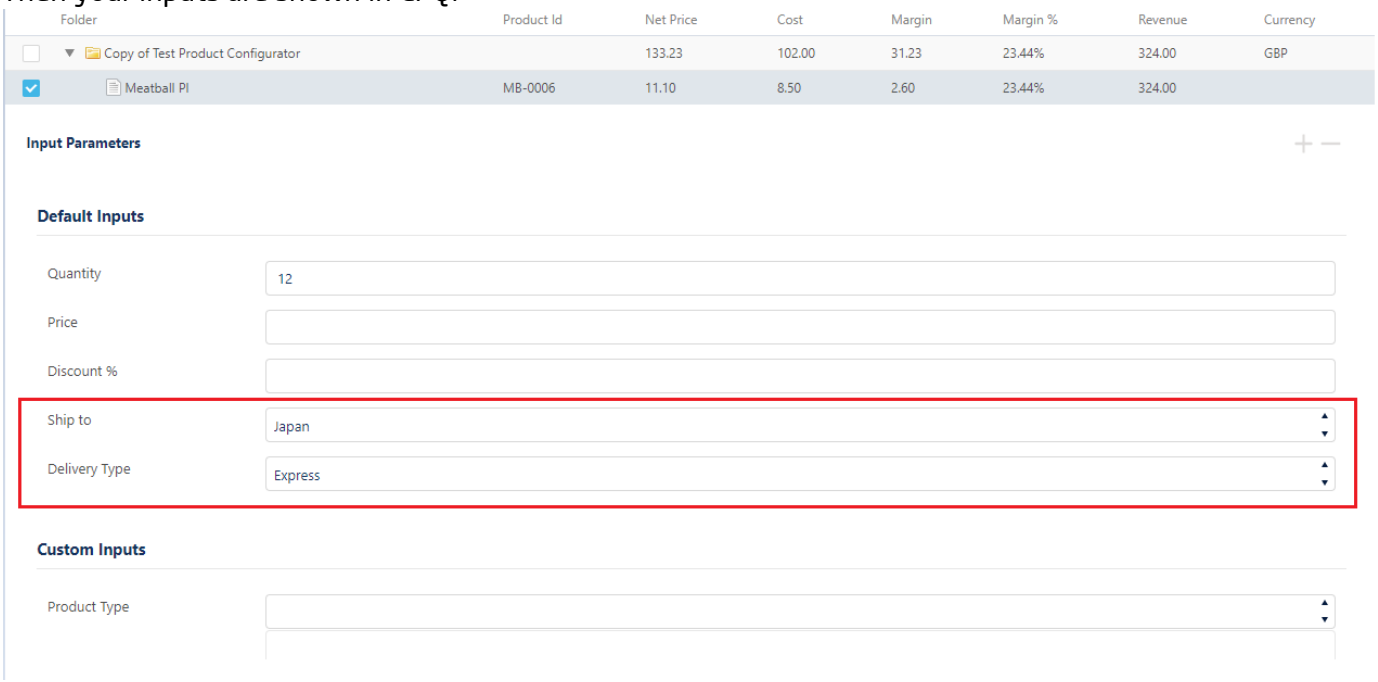
It is deployed now and its configuration is stored in AdvancedConfiguration under the name "configurator_cpq_test".



To be able to use your inputs in CPQ, all you need is to add a configuration entry to CPQConfiguration PP table or AdvancedConfiguration. For details see CPQ Package Basic Configurations.

Configuration Name	Value
productConfigurator	configurator_cpq_test

Then your inputs are shown in CPQ:



Price Settings

Configurator package allows you to set a price for each input. You can use these input prices in CPQ as the product price.

To do so, you need to set "productConfiguratorInputPriority" in "priceVsDiscountPctInputPriority" configuration:

Configuration Name	Value
priceVsDiscountPctInputPriority	**productConfiguratorInputPriority**, discountPercentageInputPriority,priceInputPriority

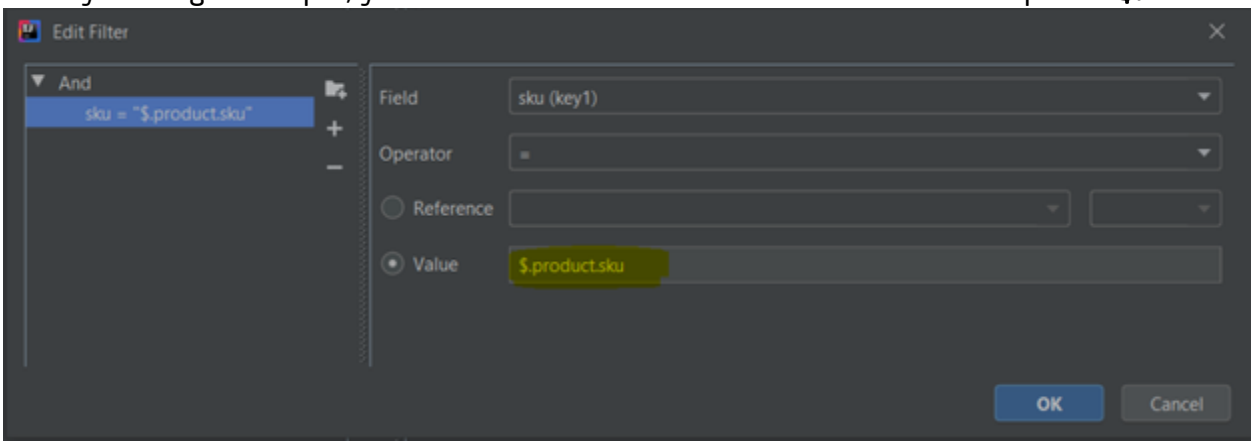
By default, the price will be the total of all input prices. However, you can calculate it using your formula.

Configuration Name	Value
productConfigurator_priceFormula	Your Formula. Ex. "**deliveryType * (1 + ShipTo)"** Where **deliveryType** and **ShipTo** are the input name

Customer and Product Filter

While integrating with CPQ, you can specify whether your input is filtered by the quoted customer or product.

When you design the input, you need to add a filter and add a value to it with a prefix "**$.**"



The supported values are:

Name	Description	Example
sku	Current product Id	$.sku
customerId	Quoted customer Id	$.customerId
product.[product attribute]	Attribute on of product	$.product.attribute1
customer.[customer attribute]	Quoted customer attribute	$.customer.attribute12

Configurations Per Quote Type

In version 2.0, this configuration is removed.

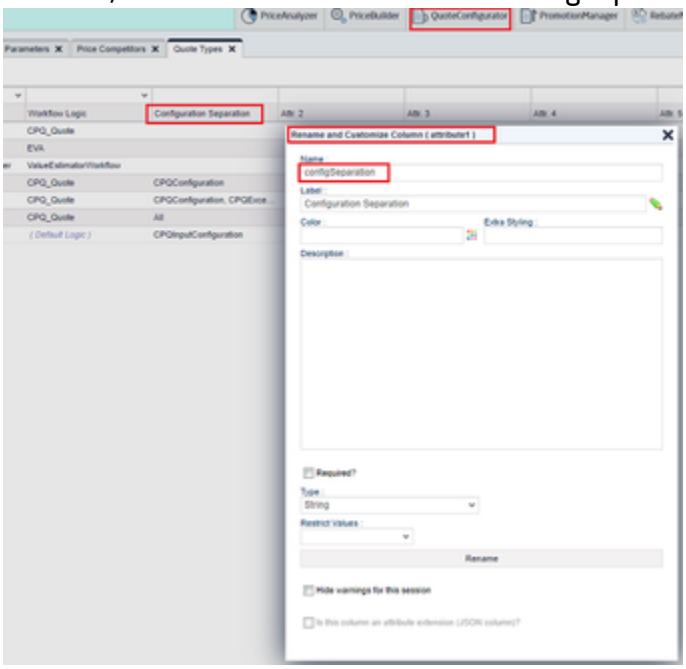
You can configure that each quote type reads data and configurations from difference advanced configuration keys and tables (PP, PX, CX). This way you can control behavior of different types of quotes.

Source Type	Name	Default Type	Quote Type (e.g.: AnotherCPQ)
Advanced Configuration	cpq-default	cpq-default	AnotherCPQ_cpq-default
	cpq-historical-data-default	cpq-historical-data-default	AnotherCPQ_cpq-historical-data-default
	cpq-generic-default	cpq-generic-default	AnotherCPQ_cpq-generic-default
Customer Extension (CX)	CustomerCashDiscount	CustomerCashDiscount	AnotherCPQ_CustomerCashDiscount
Product Extension (PX)	ProductCosts	ProductCosts	AnotherCPQ_ProductCosts
	PriceByCustomer	PriceByCustomer	AnotherCPQ_PriceByCustomer
	PriceByCountry	PriceByCountry	AnotherCPQ_PriceByCountry

	PriceBySegment	PriceBySegment	AnotherCPQ_PriceBySegment
	PriceByRegion	PriceByRegion	AnotherCPQ_PriceByRegion
	PriceGuidance	PriceGuidance	AnotherCPQ_PriceGuidance
	PriceCompetitors	PriceCompetitors	AnotherCPQ_PriceCompetitors
Price Parameter (PP)	CPQConfiguration	CPQConfiguration	AnotherCPQ_CPQConfiguration
	CPQStandardDiscount	CPQStandardDiscount	AnotherCPQ_CPQStandardDiscount
	CPQExceptionDiscount	CPQExceptionDiscount	AnotherCPQ_CPQExceptionDiscount
	CPQInputConfiguration	CPQInputConfiguration	AnotherCPQ_CPQInputConfiguration
	CPQOutputConfiguration	CPQOutputConfiguration	AnotherCPQ_CPQOutputConfiguration

Set a Quote Type

Before setting a quote type, you need to specify which quote type attribute will store the configuration. To do so, edit the column and name it "configSeparation".



From now on, you can configure which source you need to separate for the specific quote type:

- **Leave empty** - No need to separate, use the original tables.
- **All** - All tables, advanced configurations need to be cloned per a quote type. Add "QuoteTypeName_" as a prefix, e.g. AnotherCPQ_CPQConfiguration.
- **List of table names separated by comma** - Specify which default table / advanced configurations need to be cloned for each quote type.

Quote Type Management [7]

Name	Last Update	Pricing Logic	Header Logic	Workflow Logic	Configuration Separation
(Default)	20/05/2020	CPQ	CPQ_header	CPQ_Quote	
EVA	04/05/2020	ValueDriversQuote	QuoteHeaderDriver	EVA	
Value Estimator	21/08/2020	ValueEstimatorQuote	ValueEstimatorQuoteHeader	ValueEstimatorWorkflow	
AnotherCPQ	24/07/2020	CPQ	CPQ_header	CPQ_Quote	CPQConfiguration
TestQuoteType	24/07/2020	CPQ	CPQ_header	CPQ_Quote	CPQConfiguration, CPQExceptionDiscount, PriceByCountry, CustomerCashDiscount
TestQuoteTypeAll	24/07/2020	CPQ	CPQ_header	CPQ_Quote	All
QuoteType_SJ	29/07/2020	CPQ	CPQ_header	(Default Logic)	CPQInputConfiguration

Prepare Source

For each quote type, we need to clone the default configured tables / advanced configuration and derive from it our own configurations for the selected quote type. That way logics can read the correct configurations for the given quote type.

Price Parameters [12]

Name	Label	Valid After	Table Type	Value Type	Status	Simulation Set
CPQConfiguration	CPQ Logic Configuration	01/01/2018	SIMPLE	STRING	Active	
CPQExceptionDiscount	CPQ Exception Discount	01/01/2018	MATRIX	MATRIX2	Active	
CPQInputConfiguration	CPQ Inputs Configuration	01/01/2018	MATRIX	MATRIX	Active	
CPQStandardDiscount	CPQ Standard Discount	01/01/2018	MATRIX	MATRIX6	Active	
CPQOutputConfiguration	CPQ Outputs Configuration	01/01/2018	MATRIX	MATRIX	Active	
AnotherCPQ_CPQConfiguration	CPQ Logic Configuration_Test Quote Type	20/07/2020	SIMPLE	STRING	Active	
TestQuoteTypeAll_CPQExceptionDiscount	CPQ Exception Discount_Test Quote Type 01	20/07/2020	MATRIX	MATRIX2	Active	
TestQuoteTypeAll_CPQConfiguration	CPQ Logic Configuration_Test Quote Type 01	20/07/2020	SIMPLE	STRING	Active	
TestQuoteTypeAll_CPQInputConfiguration	CPQ Inputs Configuration	20/07/2020	MATRIX	MATRIX	Active	
TestQuoteTypeAll_CPQStandardDiscount	CPQ Standard Discount	20/07/2020	MATRIX	MATRIX6	Active	
TestQuoteTypeAll_CPQOutputConfiguration	CPQ Outputs Configuration	20/07/2020	MATRIX	MATRIX	Active	
QuoteType_SJ_CPQInputConfiguration	CPQ Inputs Configuration	29/07/2020	MATRIX	MATRIX	Active	

In the image above, we have the default and "AnotherCPQ" quote type.

We also separate data for both quote types in "CPQConfiguration" table.

Then we clone/copy this table and rename them with the "AnotherCPQ_" prefix.

Now, each quote type will have a different master and input configurations and the remaining configurations will read from the original source, e.g. "CPQStandardDiscount", "CPQOutputConfiguration", etc.

CPQ Package 2.0.0

- Release Highlights
  - CPQ Configuration Wizard
  - Get Price List From Matrix Type
  - Change Attribute Size of Product Prices Lookup from PX
- New Features and Fixed Issues
- Stories
  - Improvements
  - Tasks
  - Bugs

Release Highlights

The main purpose of the CPQ Package version 2.0 is to allow custom code from the library to be executed by CPQ logics.

New functionality:

- You can run custom libraries on the quote header. The libraries can run both in pre-phase and post-phase.
- There are **task runners** for data query, data processing, input, and output generator.
- You can set the priority of tasks for each runner. In addition, some tasks can depend on the results of other tasks.
- Existing logics are split into multiple modules and plugins as CPQ default configurations.

New Price Parameters introduced:

- CPQ_Default_Configuration_Entry
- CPQ_Default_Header_Configuration
- CPQ_Default_Input_Output_Configuration
- CPQ_Default_Error_Configuration

For more information, please see Version 2.0.

Upgrade notes:

- In Unity QuoteConfigurator select Default quote type change "**cpq header**" (label) to "**CPQ Header Logic"** (label) in the Header Logic column.
- Inactive "**cpq header**" logic in Header Logics.

CPQ Configuration Wizard

The CPQ Configuration Wizard has been introduced. It allows you to configure all CPQ configurations easier by providing a user-friendly interface for the setup (instead of manual changes in Price Parameters).

For more information, please see CPQ Configuration Wizard.

Get Price List From Matrix Type

Allows you to get product price from the Matrix type in the PriceList module.

Change Attribute Size of Product Prices Lookup from PX

In the previous version 1.2.2, it was required that you changed the attribute size when adding the currency column. It may have lead to losing your data after each deployment because the original attribute size (3) was to be deployed again.

From version 2.0, the attribute size of Price By Customer, Price By Country, Price By Region, Price By Segment PX is changed from 3 to 6.

Note: Please back up your data first before deployment.

New Features and Fixed Issues

Stories

PFPCS-2982 CPQ v2.0

PFPCS-3625 Simplify the "Task Runner" approach in CPQ Header Logic

PFPCS-3624 Remove LineItem Logic in CPQ 2.0

PFPCS-3936 Create wizard for CPQ 2.0

PFPCS-3541 Ability to overwrite output config at code level

PFPCS-3540 Ability for adding alert in lineItem at code level

PFPCS-2438 Enhance add null value in OPTION and OPTIONS in CPQ input config

PFPCS-2376 Enhance Using userName/Label instead of technical name

PFPCS-1187 Enhance: Move Explanation widget content in to PP

Improvements

PFPCS-3477 Enhance performance in CPQ_v2

PFPCS-3475 Add label for new CPQ PP tables

PFPCS-3389 Set default value in CPQ_Default_Configuration_Entry PP

PFPCS-3361 Should move other configs per quote type into CPQ_Default_Configuration_Entry

PFPCS-3328 Remove logic not used

PFPCS-2914 CPQConfig wizard - Should not display if user does not configure some fields in wizard

PFPCS-2705 Support to get price from PL if using matrix logic

PFPCS-2571 Display in Quote which PL ID is used to get value from

- **Upgrade note:**
  - The List Price Source element is added in Pricing Details. It helps you to know which PL-ID or product extensions PX is used to get value from and display in Quote. You can turn it on or off in CPQ_Default_Input_Output_Configuration PP.

PFPCS-2545 Change NaN in Sales Overview to empty value

PFPCS-2538 Add Historical Period in Customer History group on Header level

PFPCS-2348 Enhance warning message when not found value in ValueOptionFilter or non-existing source field in CPQInputConfiguration PP

PFPCS-2346 Add quote type in CPQ Inputs Wizard

Tasks

PFPCS-3875 Add documentation link to CPQ package description

PFPCS-3599 Set folder for CPQ_Default_Error_Configuration & Change label

PFPCS-3559 Add data to Error PP

PFPCS-3346 CPQ Naming and Convention

PFPCS-3345 Apply Error handler to existing CPQ version

PFPCS-3344 CPQ Logics naming & refactoring

PFPCS-3343 CPQ Default tables

PFPCS-3051 Change attribute size of PX Price By Customer, Country, Region, Segment

Bugs

PFPCS-4056 Wizard lib can't read quote type config based on CPQ2.0

PFPCS-4051 Quote - undefined displayed in publishing templates

PFPCS-4050 Remove obsolete CPQOutputConfiguration PP

PFPCS-4035 Recheck default data in CPQConfiguration PP

- **Upgrade note:**
  - Removed pxCostTable and pxCompetitorTable in CPQConfiguration PP after deployment from PlatformManager.
  - Price Competitor PX is not supported for deployment from PlatformManager by CPQ script. We support only Pricefx Competitor Data upload. In case you want to have Price Competitors PX, you need to create this PX manually after deployment.

PFPCS-3995 Quote - Rebate output value does not display correctly

PFPCS-3976 Target Date is not applied for Product Costs PX

- **Upgrade note:**
  - Target Date (Valid From) is applied for Product Costs now. You need to configure which Target Date field is used in CPQ Configuration Wizard.
  - CPQ supports Valid From only.

PFPCS-3962 Quote - Error timed out if adding more than 4 items

PFPCS-3950 Can't open Sales Overview when creating new quote

PFPCS-3918 Quote - Inconsistent in displaying empty or 0 for output results

- **Upgrade note:**
  - All output values are displayed as 0.00 if value is null/ empty/ 0.

PFPCS-3587 List Price is still displayed when inputting value in Price input (line item)

PFPCS-3539 Input not generated when 2 quote calculated at the same time by 1 user

PFPCS-3501 ProductConfigurator is not working properly when adding new product

PFPCS-3466 Folder show 0 value in Quote

PFPCS-3443 List Price is still displayed in quote if Price input = 0

PFPCS-3442 Price Guidance should be displayed null if no data

PFPCS-3406 Values on Header are displaying 0 in Quote

PFPCS-2977 CPQ Inputs Config_N/A is not displayed correctly if Input Type is OPTIONS

PFPCS-2809 Displays 100% in Sales Overview in case we don't have Cost and Margin

PFPCS-2409 No data available in Product filter when Input Type is Options

PFPCS-2403 CPQ Input Config - Can't work if Input Type is OPTIONS and filter operator is OP_NOT_EQUAL

PFPCS-2345 Product Configurator - Must recalculate 2 times to get correct Invoice Price