Accelerate CPQ Package

Version 2.2.0

November 2022

Accelerate CPQ Package

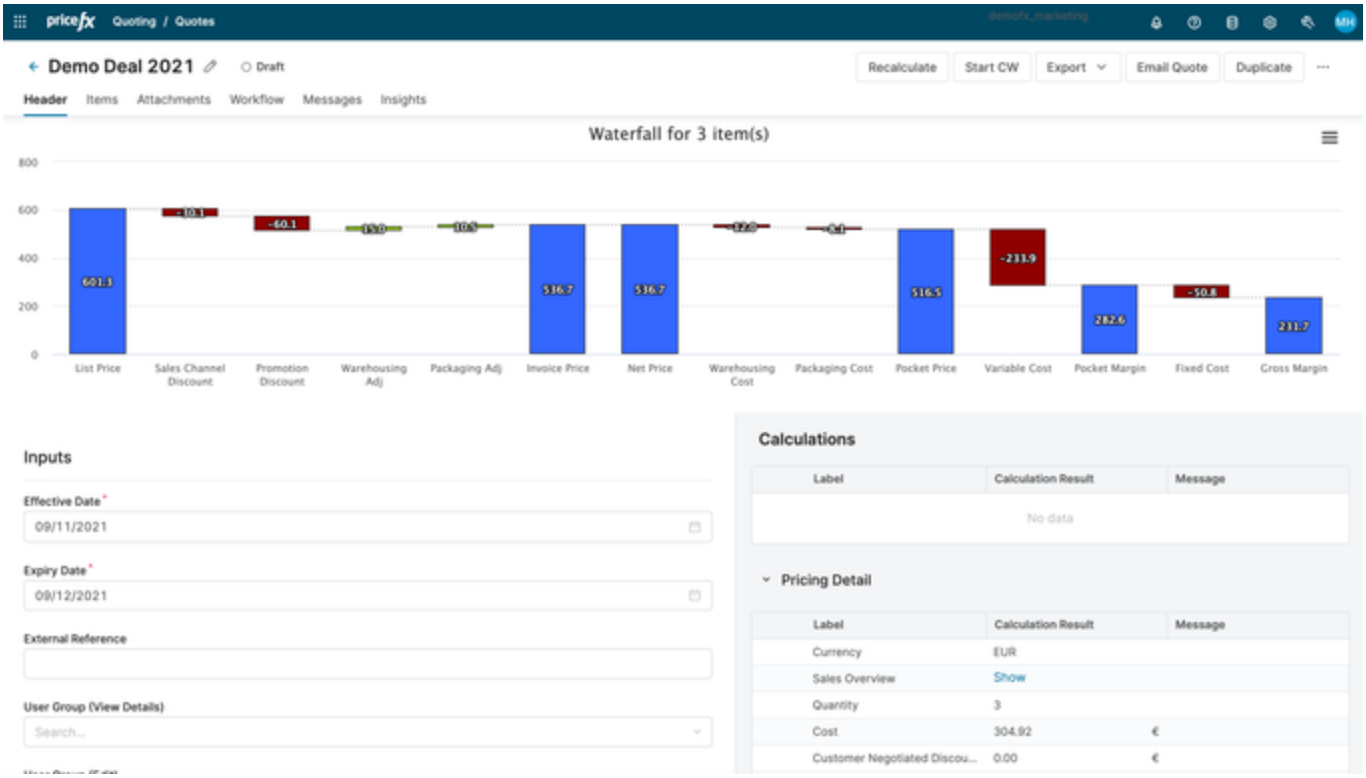
The CPQ Package consists of various components to support the whole process of negotiating a deal – selecting customers and products, pricing and negotiating the deal, approval workflow, quote output. The setup does not require heavy configuration, you only need to provide a few data tables and map transactional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

Product Info (CPQ)

Accelerate CPQ Package provides sales representatives with the latest product and customer details, discount guidelines, recommended target, stretch, and floor prices, and other relevant details to help them negotiate the right price with their customers.

This package includes:

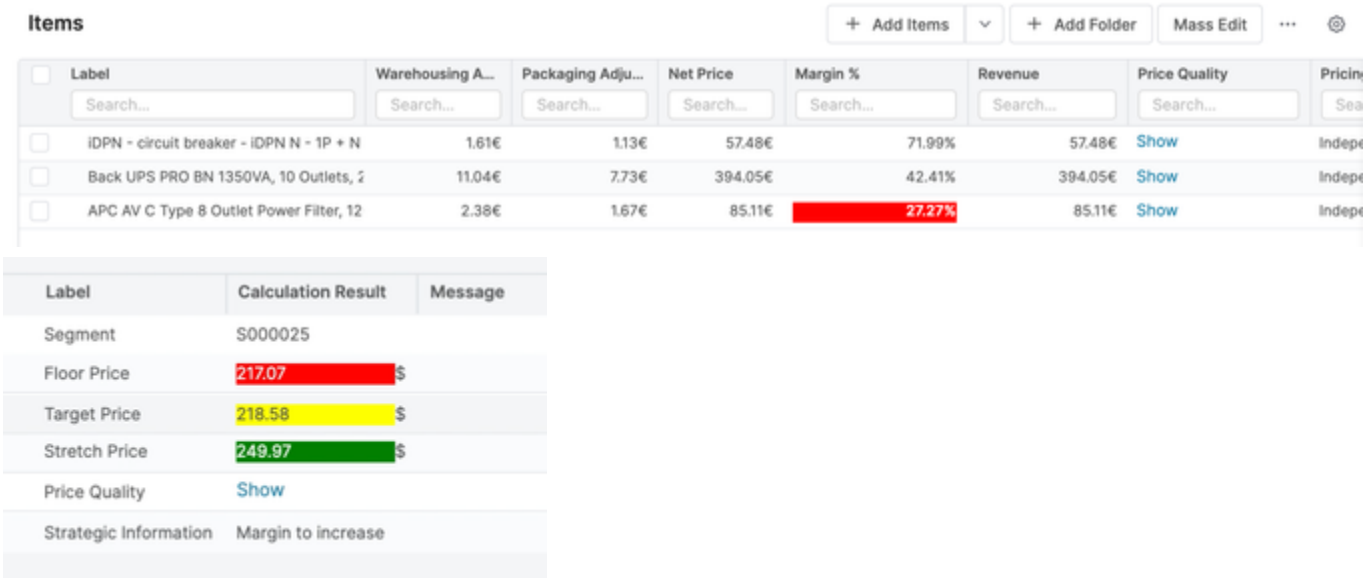
- Customer discounting at the customer and/or item level
- Product configuration
- Excel/Word/PDF based quote templates
- Email quote to customer capability
- Approval workflow through the Accelerate Approval Workflow Package



Key Deliverables

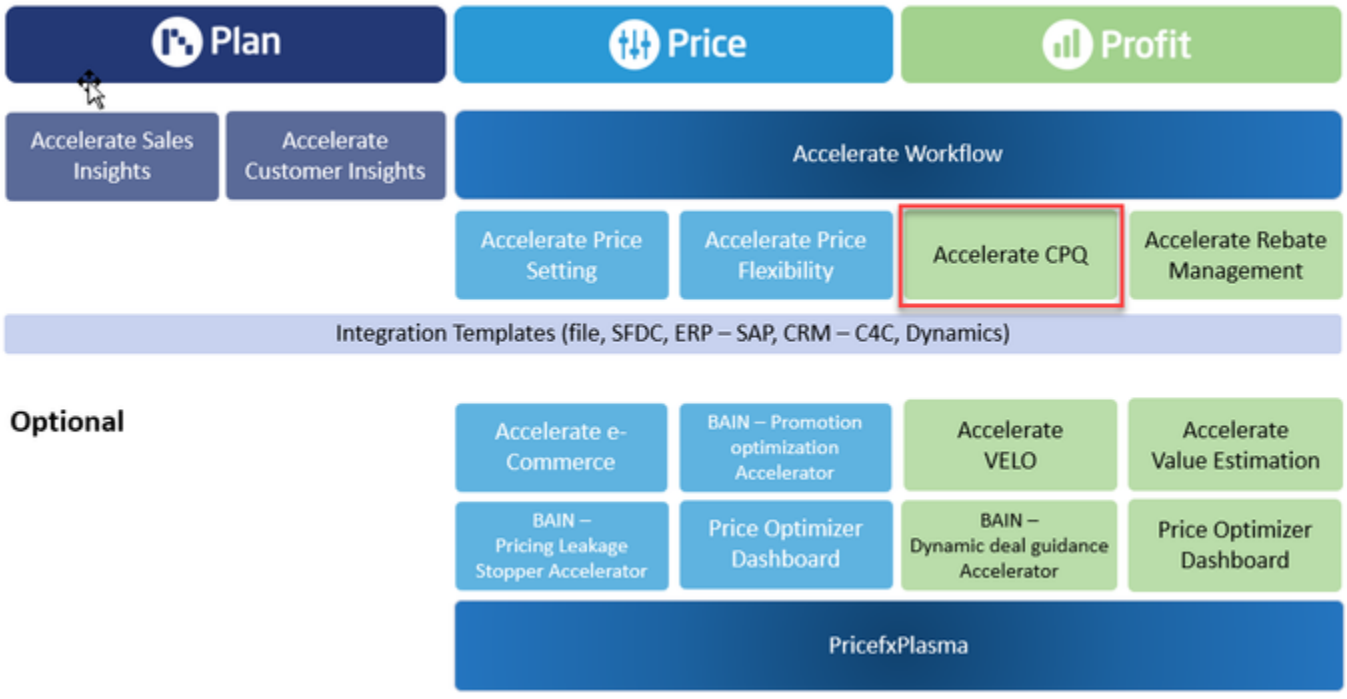
Enable your sales force to respond to sales inquiries with error-free quotes. Automate your overall CPQ process – selection of customer and products, pricing guidance and negotiation management for a deal, automated approval workflow, creation of quote document based on templates and handling of e-signature.

- Utilization of predefined discount models
- Product selection based on product group or sales organization
- Pricing calculation (revenue, cost, discount, margin, deal score, ...)
- Can be extended by pricing guidance based on AI-powered optimization
- Customer history (revenue, quantity, margin)
- Benchmarking with historical cases
- Competitive comparison
- Approval workflow initiation based on defined rules (provided with complimentary Accelerate Workflow)



Overview (CPQ)

The CPQ Accelerator is one of many pre-built solutions from Pricefx provide sales staff with the latest product and customer details, discount values, target/stretch/floor pricing, and additional relevant values to assist in negotiating the right price with customers.



Within the design of the Pricefx PLAN/PRICE/PROFIT structure, we can see exactly where the CPQ Accelerator will be included:

Business Overview (CPQ)

Premise

You need to respond to customer sales inquiries in a fast and efficient manner with quotes that are created in an automated fashion as part of a team from either Pricing, Financial, Sales, Sales Ops within your organization.

Desired Outcome

Enable your sales force to respond to sales inquiries with error-free quotes. Automate your overall CPQ process – selection of customers and products, pricing guidance and negotiation management for a deal, automated approval workflow, creation of quote document based on templates and handling of e-signature.

Context and Background

Your current CPQ process does not provide quotes in a manner that is timely, efficient, accurate, and with enough information and detail for sales staff to generate an error-free quote. There exists a need for improvement for price guidance, negotiations, approvals, competitive comparisons, and historical information.

Problem

The CPQ process is not an automated process and the salesforce is unable to respond to their customer's sales queries in a fast and efficient manner with a quoting system that provides pricing guidance, negotiation management, and an automated approval process.

Solution Capabilities

Once this Accelerator has been implemented and linked to your application, then these components will be available for immediate use:

- Ability to use predefined discount models

- Automated product selection
- Improved pricing calculations for revenue, cost, discount, etc.
- Automated approval workflows
- Benchmarking and competitive comparisons

Capabilities Summary (CPQ)

Capabilities

Out of the Box

- Support the whole process of negotiating a deal
  - Selecting customers and products
  - Pricing and negotiating the deal
  - Approval workflow – through the Approval Workflow accelerator
  - Quote output
- Publishing template
- Approval email message template
- Configuration wizard – allows setting many options through a guided process

Configurable

- Sources of data – which tables will be used for:
  - Currency conversion
  - List/base price
  - Competition data
  - etc.
- More Quote Types
- Price lookup in "matrix" price lists (more dimensions for lookup)
- Custom inputs on quote header and items
- Custom outputs on quote header and items
- Custom calculations

Audience

- **Technical Admin** – Installation, initial configuration, configuration of Quote Types
- **Business Admin** – Changes the accelerator behavior (likely with help of Technical Admin)
- **Business User** – Sales Representative (creates quotes), approvers

User Stories

Quote summary	View summary data for quoted items (i.e., quote currency, total revenue, net price, margin, margin %, quantity). Review the quote overview and understand profitability.
Quoted product sales history overview	View the sales history overview (based on historical data) with details of the Customer's buying behavior for the last period for quoted items. Understand the customer total spend over the selected time and offer a similar quote.
Customer buying history overview	See Customer's buying history overview (based on historical data) for the last months – total revenue, quantity, margin, margin %.

	Understand the customer total spend over the selected time and offer a similar quote.
Product quantity	Specify a quantity for a quoted Product.
	Provide a quotation matching the quantity needs and margin effect.
Product price/discount	Negotiate a price or discount % for a quoted Product.
	Provide a discount (additional, flat %, default %) to the customer.
Product price waterfall	View a price waterfall with the following elements – List Price, Discount, Discount %, Invoice Price, Rebate, Cash Discount %, Net Price, Cost, Margin, Margin %, Total Revenue of a Product SKU.
	Understand the product total profitability and revenue leakage and negotiate prices as needed.
Product sales history KPIs	View product sales (based on historical data) for the last months. Available KPIs are Revenue, Quantity, Margin %, Weighted Avg. Price, Avg. Revenue Per Invoice, Avg. Quantity per Invoice, Avg. Margin % per Invoice. It is possible to filter data by customer attributes, e.g., Segment, Customer ID, Region, Sales Person, etc.
	Understand the product financial performance and negotiate prices as needed.
Competitor comparison	View a comparison with competitors (if available). Available KPIs are Competitor Median Price, Top 5 Competitors, Lowest Competitor Price.
	Offer the right price relative to the competitors.
Price guidance	View Price Guidance (Stretch, Target, Floor Price) based on a data lookup from Product Extension for a product and customer.
	Adjust prices based on user's empowerment and understand the target price, price stretch, and price floor.
Historical prices	View a previous price (based on historical data) transacted by the customer. Available data: Unit Price, Margin %, Date.
	Validate the quote in line with historical behavior.
Standard discount lookup	Look up a standard discount (up to 6 product/customer hierarchical keys) or a discount exception (Product SKU, Customer ID).
	Review the standard discount structure.
Competition data lookup	Look up competition data from a standard Product Competition or Product Extension table.
	Review competitor prices at the product level.
Price data lookup	Look up price list data from a Price Setting price list or Product Extension table. Available tables are Prices per Customer, Prices per Segment, Prices per Country, Prices per Region.
	View all the product base prices and review, update, and delete them.

Business User Reference (CPQ)

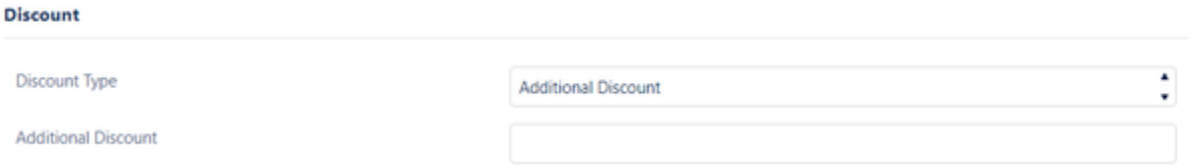
Quoting Tool

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 header level.

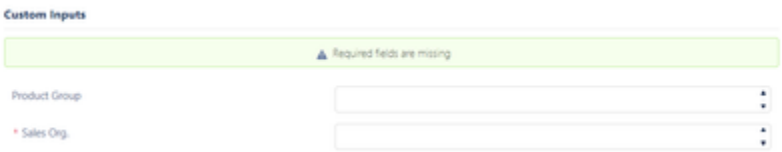


    2. **Default Discount %** - Allows to define a default discount % for line items; it can be overridden on the line item level.
    3. **Flat Discount %** - Allows to define a discount % which is applied to all line items, overriding discounts on the line item level.

Additional Input Fields

⛔ This feature is deprecated from v2.0 and will be disabled by default from v2.2.0.

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.



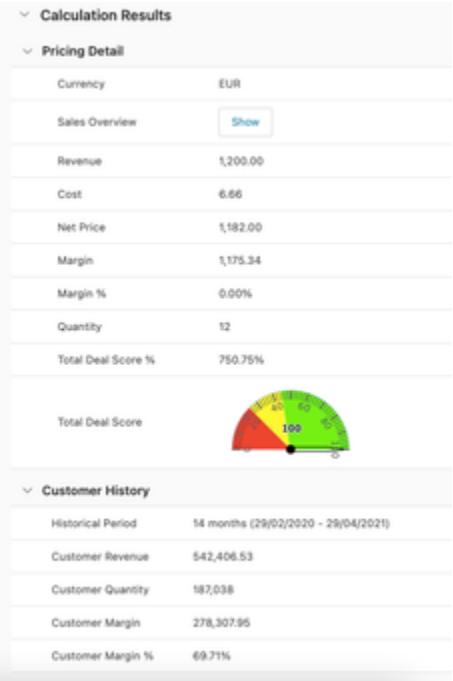
You can configure this manually in the CPQInputConfiguration PP or using CPQ Configuration Wizards with *Name, Label, Input Type, Source Table, Source Type, Source Field, Value Options* (if there is no source field) *, Default Value, Read Only, Required, Level (Header/Line), Customer Filter, Product 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 filterFormulaParam.

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.

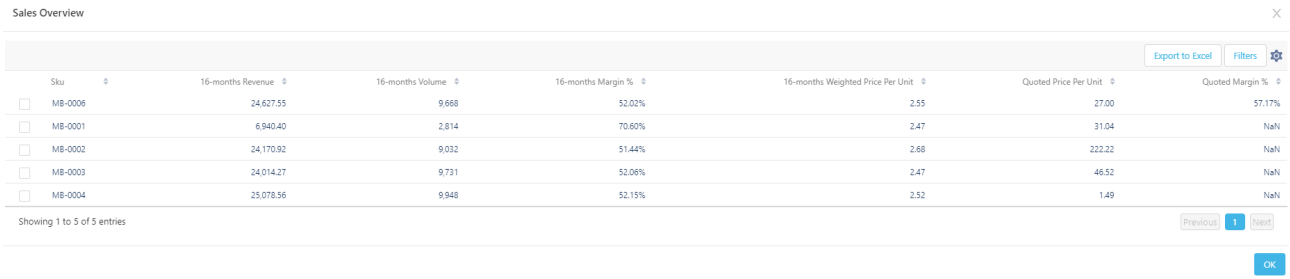
Calculated Results

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 you 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**

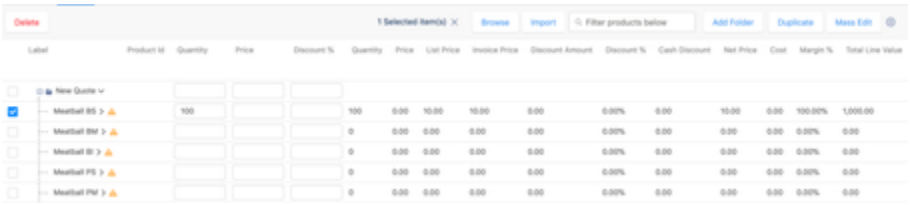
- **Cost**
- **Net Price**
- **Margin %**
- **Margin**
- **Quantity**
- **Total Deal Score %** = Sum (Invoice Price * Quantity) / Sum (Target Price * Quantity)
- **Total Deal Score**

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

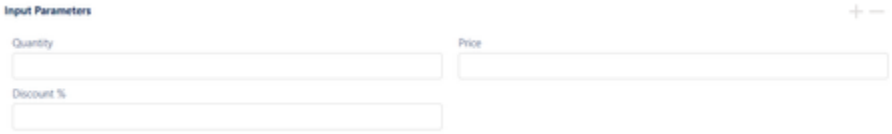


User Inputs

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.

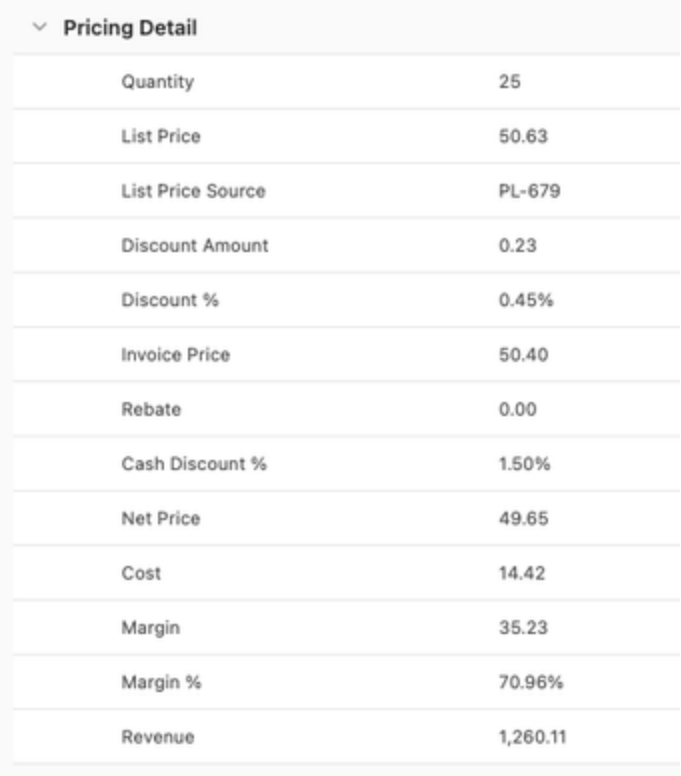


Calculated Results

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.

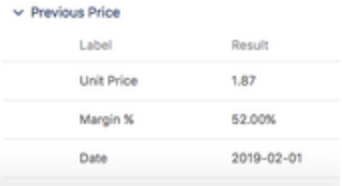
List Price is obtained from two sources in this order: Price Lists and Product Extensions.



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

Product price is queried from a price list (Price Setting 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. In case of the same priority and customer assigned PL, the first valid result price is used.

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.

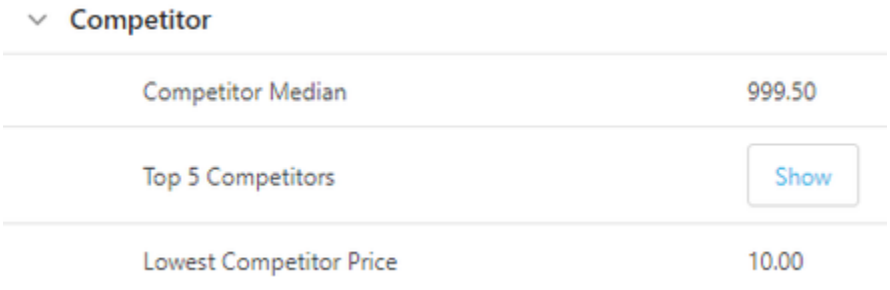
- **Invoice Price** - It is the Price input if set; otherwise it is the difference between List Price and Discount Amount.

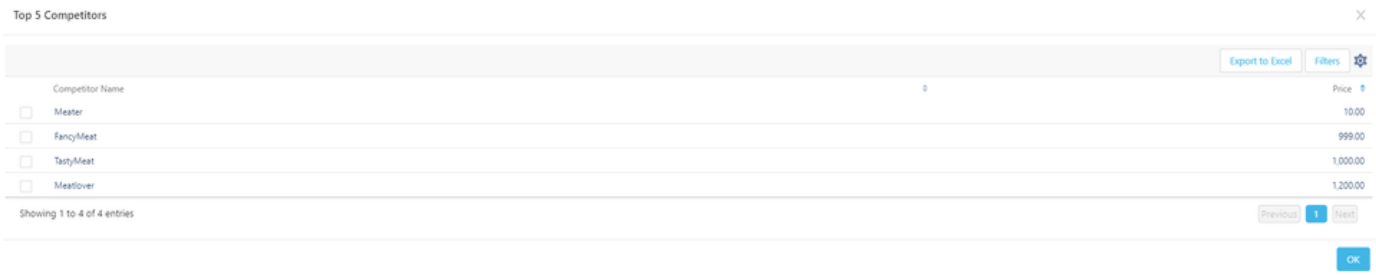
- **Rebate –** Rebate data for a customer and product from RebateManager. There are two calculation schemes: current or forecast (the maximum rebate is temporarily not be applied). By default: Forecast. The configuration is here.
  - To calculate a rebate, you need data from Rebate Records (RR) and you 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". (In case of customerId duplication, the first valid looked-up record is used.)
  ℹ Ignore if Cash Discount % is null, zero, negative.
- **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.

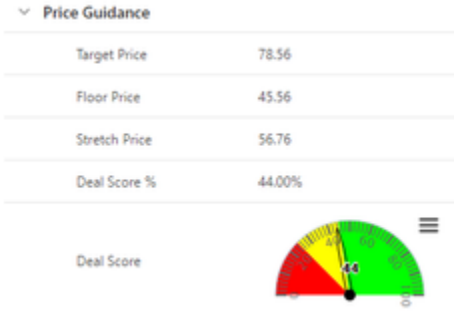
ℹ Ignore if the competitor price is zero, null, negative.

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.

Before v2.2.0, the guidance is an absolute value and shown directly as the price guidance.

From v2.2.0, the guidance is a percentage value, and the price guidance is calculated as "**cost * (1+ guidance)**".

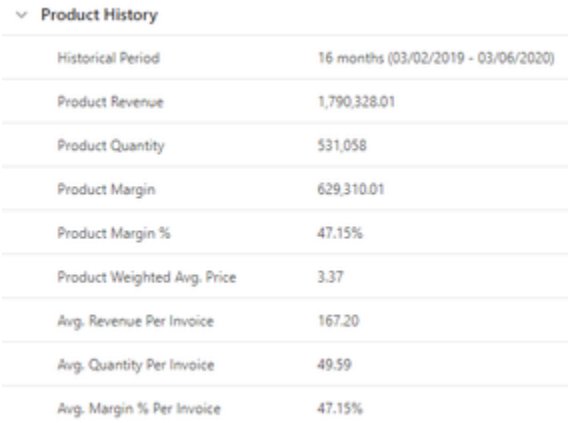


Deal Score % = Invoice Price / Target Price

ℹ 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.



- 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 the Price Setting module 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.

Admin User Reference (CPQ)

As a technical admin person, you can find here detailed descriptions of configuration and customizations of the CPQ accelerator.

Read on:

- Mandatory Data (CPQ)
- Installation (CPQ)
- Installation with Details (CPQ)
- Architecture Components (CPQ)
- CPQ Package Architecture

Mandatory Data (CPQ)

Type	Name	Fields	Mandatory
Master Data	Product		Core data
	Customer		Core data
Product Extension	ProductCost	Mandatory: • SKU • Cost - defaults to attribute1 Optional: • Currency • Additional dimensions fields	Core data

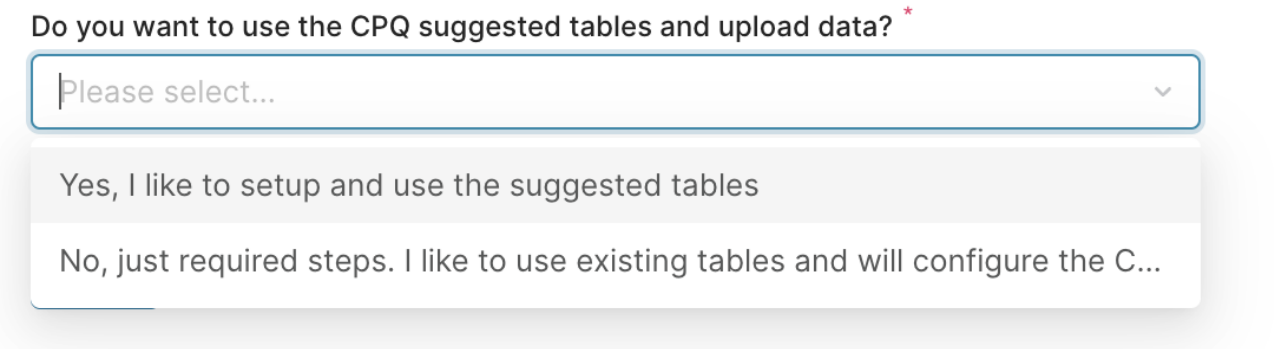
	Price List		Core data
Product Extension	• PriceByCustomer • PriceByCountry • PriceByRegion • PriceBySegment	Mandatory: • SKU • Attribute1 – customer mapping • Attribute2 – cost Optional: • Currency	Mandatory if the price of the product is not in the price list
Model Record	PriceOptimizer model	Mandatory: • Target price • Floor price • Stretch price Optional: • Currency	Optional
Product Extension	PriceGuidance	Mandatory: • Target price • Floor price • Stretch price Optional: • Currency	Optional
Product Extension	PriceCompetitors	Mandatory: • SKU • Competitor • Price Optional: • Currency	Optional
Customer Extension	CustomerCashDiscount	Mandatory: • Customer ID • Discount	Optional
Price Parameter	StandardDiscount	Mandatory: • Customer ID • Dimensional keys • Discount • Value Type Optional: • Currency	Optional
	ExceptionDiscount	Mandatory: • Customer (key1)	Optional

		• Product (key2) • Discount (attribute1) Optional: • Currency	
Transaction Data	Sales data	Mandatory: • SKU • Customer ID • Quantity • Invoice price • Net price • Margin • Pricing data Optional: • Unit price • Invoice ID	Optional

Installation (CPQ)

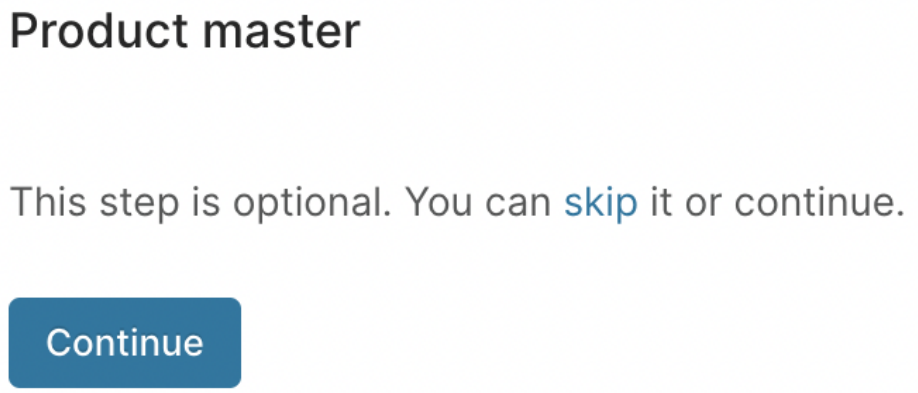
Steps to deploy the CPQ Package:

1. In PlatformManager, navigate to **Marketplace** > **Accelerator Packages** and find *CPQ Accelerator.*
2. Click **Deploy** and select a partition to which you want to deploy the package.
3. Click **Deploy**.
4. A warning dialogue will appear. After you read the warning text and you agree with the conditions, you can click **Continue**.
   If you need to leave the deployment process before it is finished, you can always come back later. The wizard will offer you to either start again, or continue in the previously started process.
5. Configure CPQ tables and data. You can get the suggested tables from Accelerators team or manually configure your existing tables.

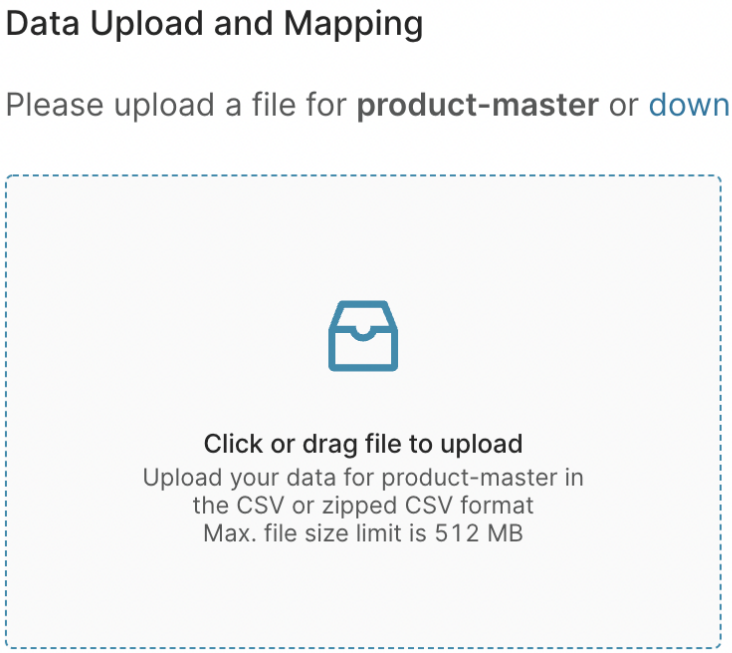


   If you choose **No, just required steps. I like to use existing tables...** you will skip some optional steps and start with the **Product Cost Mapping** step.
6. Configure optional steps. These are shown when you choose **Yes, I like to set up and use the suggested tables**.
   The optional steps are: **Product Master, Customer Master, Product Costs, Customer Level Price, Segment Level Price, Country Level Price, Region Level Price, Price Guidance, Customer Cash**

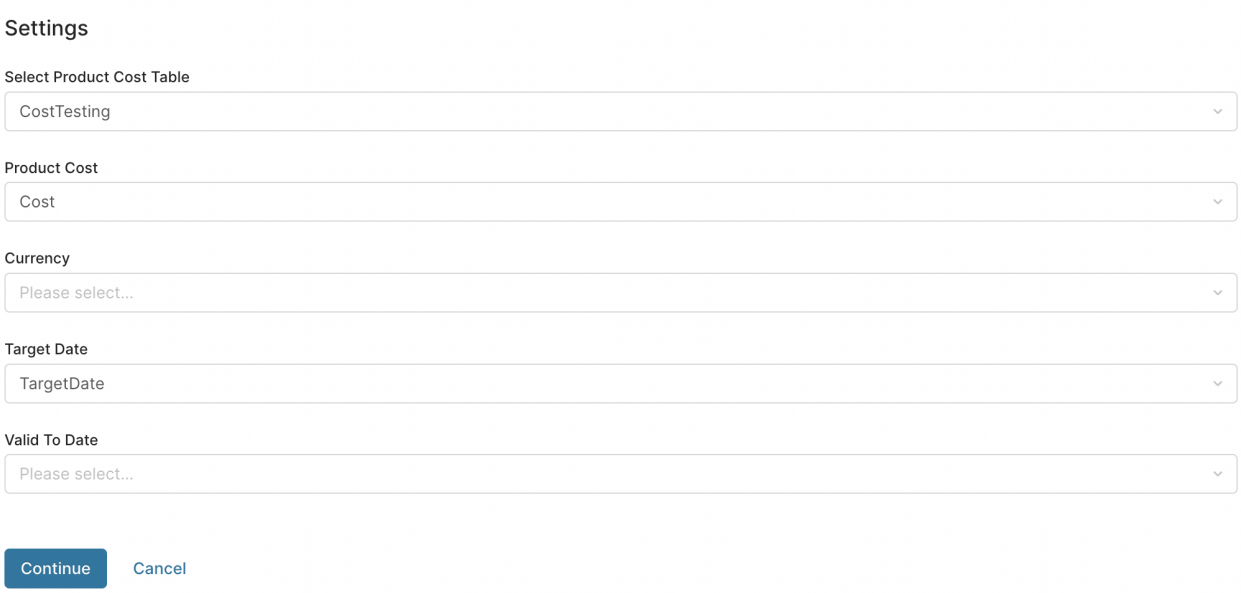
Discount, Competitor Data, Standard Discount, Exception discount. If you do not want to configure a particular step, just click **Skip**. You will move to another optional step.



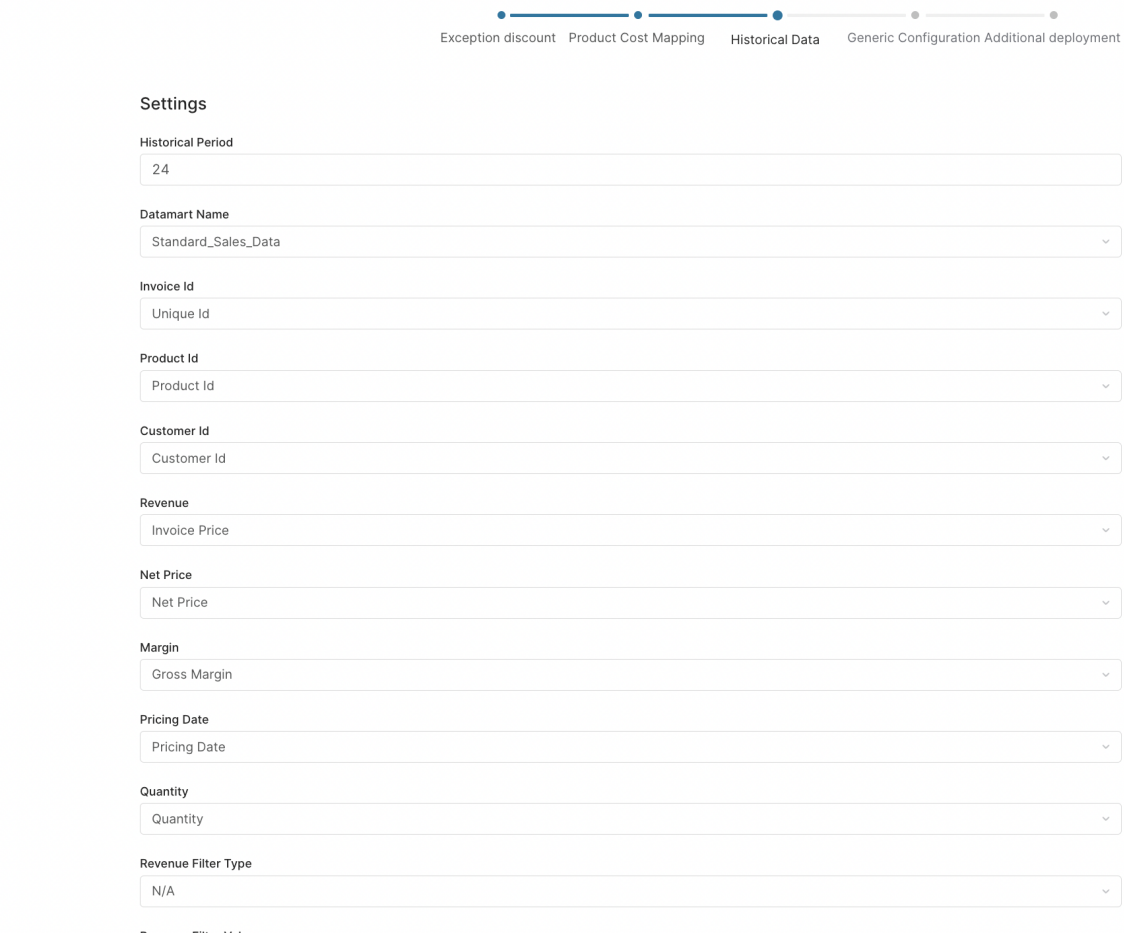
After clicking **Continue**, you need upload a CSV file for this step. You should download a sample file and fill data into it.



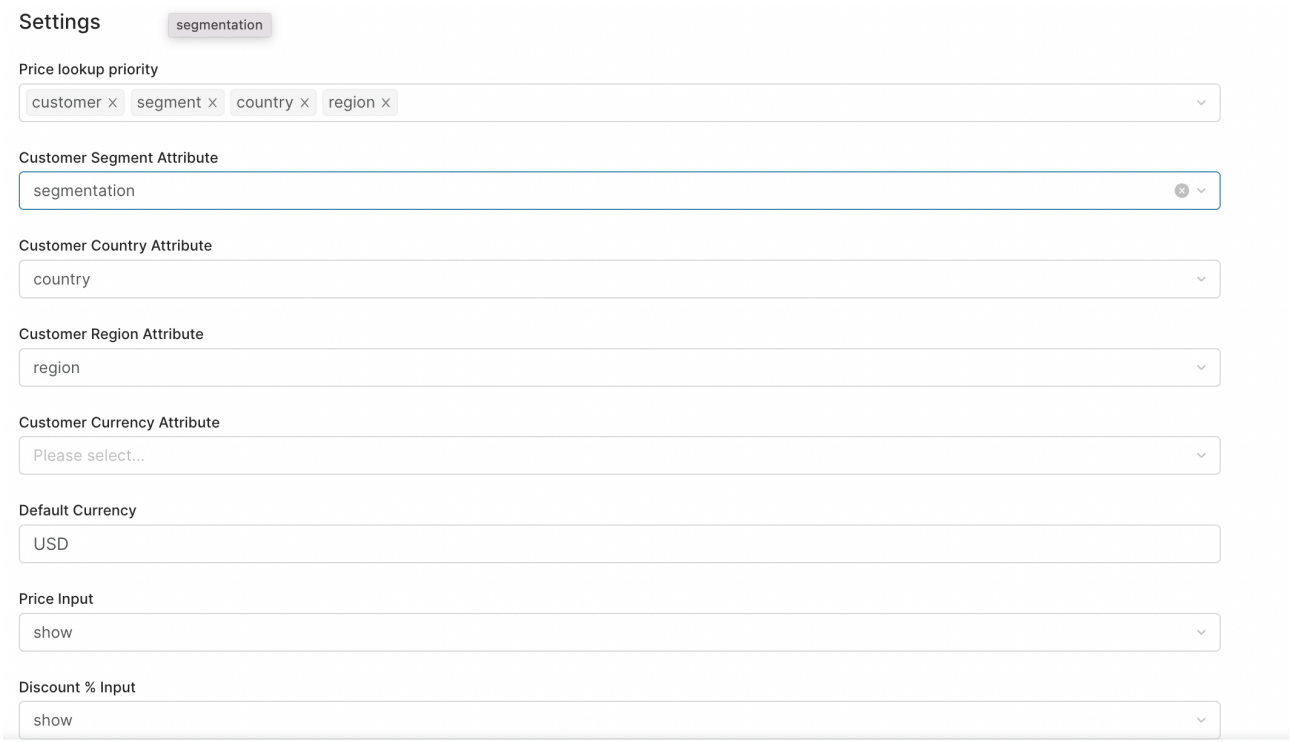
7. Product Cost Mapping. Here you map the cost table and some cost related columns.



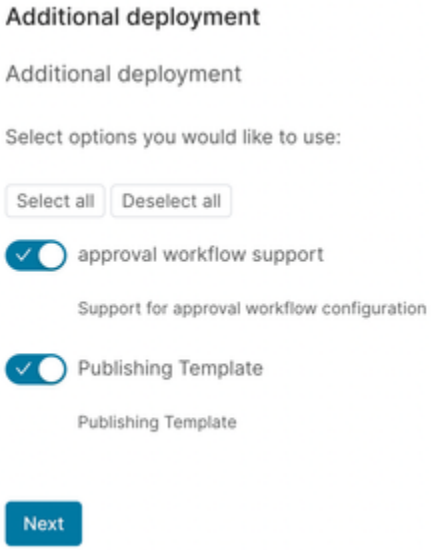
8. Historical Data. This is an optional step. You can skip it or map here some values.



9. Generic Configuration. Here you configure general fields and columns for CPQ.



10. Additional deployment. In CPQ, there are two additional packages: approval workflow and publishing template.



11. Wait for the deployment to finish.

Congratulations!

Your Accelerator / Accelerator Package was successfully deployed. Continue to see the result.

Finish | Go to partition

Installation with Details (CPQ)

Table of Contents

Prepare your own configuration files
Set up Approval Workflow

This installation tutorial will guide you though an installation and initial configuration steps of **CPQ Accelerator** version **2.0.2**.

Pre-requisites

Before you start, ensure that you have:

- Common accelerator installation pre-requisites
- Mandatory data for CPQ Accelerator
  - Master data for basic functionality of the accelerator
    - Tables
      - Products
      - Customers
      - Product Costs
      - Product prices – List prices can come either from a price list or from Product Extension tables. You can have the prices stores in a combination of 1 up to 4 tables – Customer Level Price, Segment Level Price, Country Level Price and Region Level Price. If you need to keep price data without distinction, you can place them e.g. into Region Level Price and keep the Region field empty.
    - These tables can be defined and data uploaded to the partition before or during or after installation.
    - If you plan to upload the CSV files during installation, you can find samples of CSV headers in the description od the installation steps below.
    - If you create some tables manually, then:
      - Make sure you created them with proper names and data structure.
      - You can do the configuration using the CPQ wizard after installation.
  - Historical Data – The assumption is that you already have the historical sales transactions uploaded in a Datamart.
- Optional data
  - Review the installation steps which other data can be uploaded already during installation process. You can skip their upload, if you do not use them in your pricing process. You can also add and configure them later.

Installation Steps

Initial Step

1. In Platform Manager, navigate to **Marketplace › Accelerator Packages**.
2. Find the CPQ Accelerator.
3. Click **Detail** and select your partition and version (e.g. 2.0.2) you want to install.
4. Click **Deploy**.
5. You will see a warning about possible risks. If you agree, go ahead.

> ⚠ The installation will override previous data tables with the same names as installed by the CPQ Accelerator and also previous settings.

Product Master

You can upload the Product Master data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Product Id,Product Name
```

The data will be uploaded to the table *Products*.

Customer Master

You can upload the Customer Master data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Customer Id,Customer Name
```

The data will be uploaded to the table *Customers*.

Product Costs

You can upload the Product Costs data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Product Id,Cost
```

The data will be uploaded to the Product Extension table *ProductCosts*.

Customer Level Price

You can upload the Customer Level Price data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
```

```
Product Id,CustomerId,Price
```

The data will be uploaded to the Product Extension table *PriceByCustomer*

Country Level Price

You can upload the Country Level Price data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Product Id,Country,Price
```

The data will be uploaded to the Product Extension table *PriceByCountry*.

Region Level Price

You can upload the Region Level Price data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Product Id,Region,Price
```

The data will be uploaded to the Product Extension table *PriceByRegion*.

Price Guidance

You can upload the Price Guidance data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Product Id,Target Price,Floor Price,Stretch
Price
```

The data will be uploaded to the Product Extension table *PriceGuidance*.

Customer Cash Discount

You can upload the Customer Cash Discount data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Customer Id,Discount %
```

The data will be uploaded to the Customer Extension table *CustomerCash Discount*.

Competitor Data

You can upload the Competitor Data data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Product Id,Competitor,Price
```

The data will be uploaded to the master data table *Competition Data*.

Standard Discount

You can upload the Standard Discount data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

```
Discount,Discount Value Type
```

The data will be uploaded to the Customer Parameter table *CPQStandardDiscount*.

Exception Discount

You can upload the Exception Discount data during the installation process if you want. If you already have them on the partition or plan to do it later, you can **skip** the step.

If you want to upload now, click **Continue** and you will be prompted to supply a CSV file with data in the format shown below. It shows only the mandatory fields, but your dataset can have also many other fields (e.g. Currency, Validity, etc.).

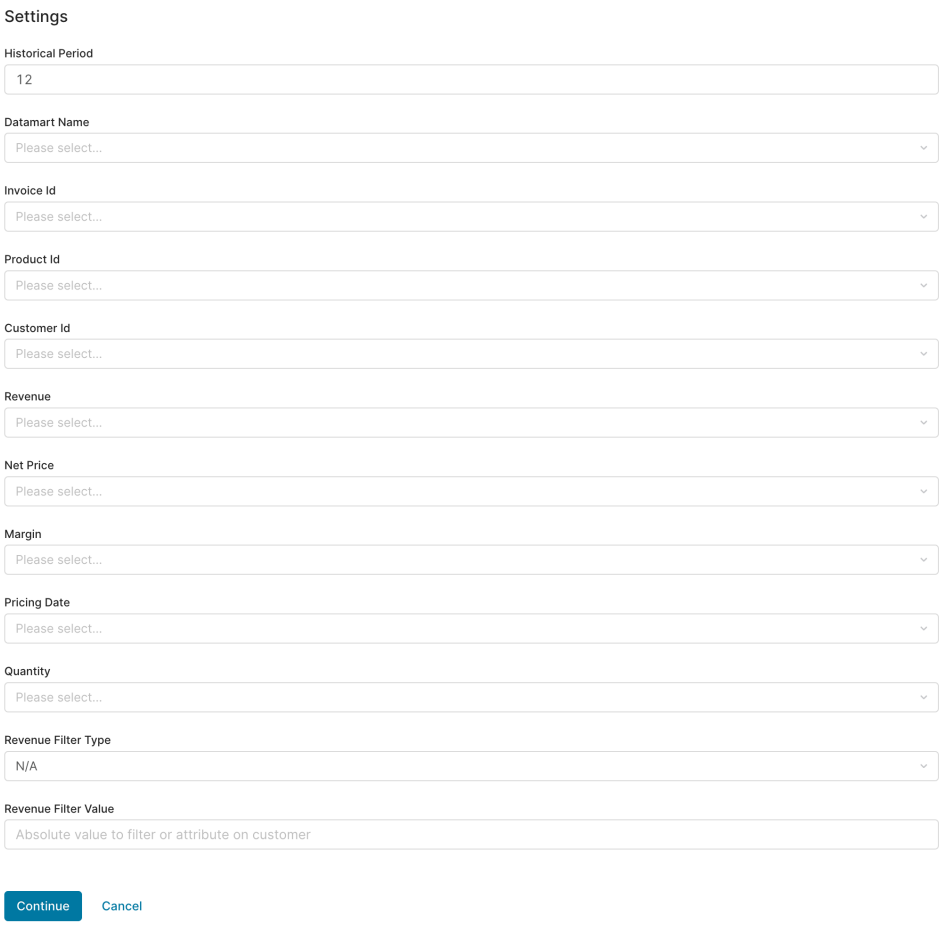
```
Discount,Discount Value Type
```

The data will be uploaded to the Customer Parameter table *CPQException Discount*.

Historical Data

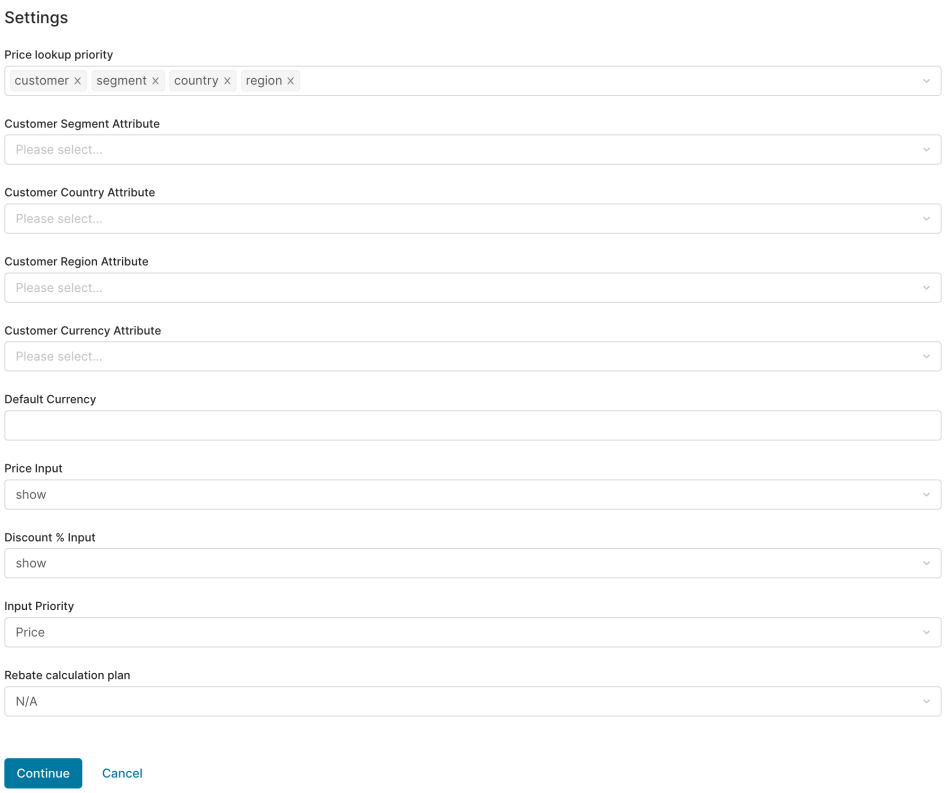
Provides information about in which Datamart you store your historical sales transactions.

Even though it is possible to skip mapping of the historical data, the default accelerator configuration contains a reference to a Datamart with certain pre-configured name, so if you do not provide your Datamart, you will get errors when calculating a quote.



The above settings will be stored in the Company Parameter *CPQConfiguration* (see Historical Data Configuration for details).

Generic Configuration



The above settings will be stored in the Company Parameter *CPQConfiguration*; for details see:

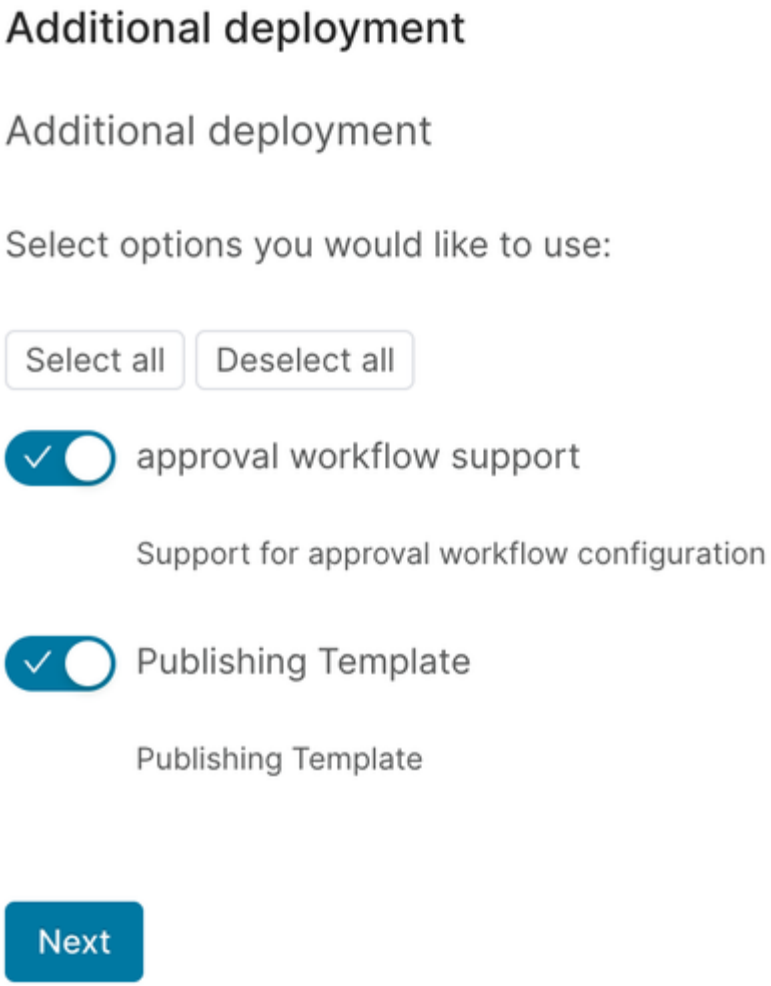
- Product Prices Configuration
- Customer Mapping Configuration
- Rebate Configuration
- Other Configurations - Inputs

Explanation of the settings:

- **Price lookup priority** – Order in which the prices will be searched.
- **Customer Segment Attribute –** Which attribute in the Customers data represents the segment. It will be used to look up prices by segments.
- **Customer Country Attribute –** Which attribute in the Customers data represents the country. It will be used to look up prices by country.
- **Customer Region Attribute –** Which attribute in the Customers data represents the region. It will be used to look up prices by region.
- **Customer Currency Attribute –** Which attribute in the Customers data represents the currency.
- **Default Currency –** Whenever it is not possible to find currency in the data, this currency will be used.
- **Price Input** (show/hide) – If the Sales Representative should be able to set the required product price on the quote line item during negotiation.
- **Discount % Input** (show/hide) – If the Sales Representative should be able to set the required discount % on the quote line item during negotiation.

- **Input Priority –** Price / Discount% – If both previous inputs are available on the quote line item and the sales rep enters both, which one should be used. This is stored in the configuration option *priceVsDiscountPctInputPriority.*
- **Rebate Calculation Plan –** N/A / forecast / current

Additional Deployment



- **Approval Workflow Support** – When selected, Approval Workflow Accelerator will be installed too. For a list of deployed components, see Architecture of the Approval Workflow.
  ⚠️ This will override any previous setting of approval workflow.
- Publishing Template – When selected, DOCX template for Quotes will be installed too. Then users can export quotes to DOCX or PDF. The name of the template is *MS Word template*.
  ⚠️ This action will override any previous quote publishing template with the same name.

Post-Installation Steps

Set up Quotes Calculation

1. Navigate to **Administration › Configuration › Quoting › Quoting General Setting**.
2. Set **Create Quote Option** to *Calculate new Quote immediately.*

Prepare your own configuration files

If you plan to use the same setting for all types of quotes, then you can skip this step. But if you want to have specific settings for various quote types, follow the steps below. If you want to get more details about the relations of the tables, see Configuration per Quote Type.

> We suggest keeping the naming patterns as mentioned below, just replace the text *MyOwn* with text which describes well your type of quote. Keep the name in camel-case and without spaces.

- In the Quote Types section, find an unused attribute field, and rename it: set the name to "configurationEntry", with the label "Configuration Entry".
- Create a new Quote Type named "MyOwnQuoteType".
  - As Pricing Logic select *CPQ Logic.*
  - As Header Logic select *CPQ Header Logic v2.0.*
  - In Configuration Entry put "CPQ_MyOwn_Configuration_Entry".
- Make a copy of the Company Parameter *CPQConfiguration.*
  - Rename the copy to "CPQ_MyOwn_Configuration".
  - Set its Valid After date to *2018-01-01*.
  - Set its Status to *Active.*
- Make a copy of the Company Parameter *CPQ_Default_Configuration_Entry.*
  - Rename the copy to "CPQ_MyOwn_Configuration_Entry".
  - Set its Valid After date to *2018-01-01.*
  - Set its Status to *Active.*
- In the Company Parameter *CPQ_MyOwn_Configuration_Entry* change the value of the key *Configurations* from *CPQConfiguration* to *CPQ_MyOwn_Configuration_Entry.*

Set up Approval Workflow

If you deployed support for approval workflow, you need to do its setup. For details see Approval Workflow configuration tables.

Architecture Components (CPQ)

Advanced Configuration Options

- cpq-default - JSON with configuration stored as key-value pairs.
- cpq-meta - JSON with meta-configuration. References many things, e.g. the cpq-default.

Calculation Logics

- CPQ
- CPQ_ConfigurationWizard
- CPQ_ConfigurationWizardCommonLib
- CPQ_ConfigurationWizardExecutor
- CPQ_ConfigurationWizardScreen
- CPQ_CustomerFilter
- CPQ_DefaultInputConfigurator
- CPQ_Header
- CPQ_Header_Utils

- CPQ_Input_Utils
- CPQ_Output_Utils
- CPQ_PriceList_Key_Providers
- CPQ_ProductFilter
- CPQ_Publishing_Template_Data_Provider
- CPQ_SharedLib
- CPQ_header_discounts_configurator
- HeaderInputs_Configurator
- LineInputs_Configurator

Configuration Wizard

- CPQ_Configuration_Wizard

Message Template

English and German templates for the approval workflow emails.

Data Tables

- Price Record fields definition
- Company Parameters
  - CPQConfiguration
  - CPQExceptionDiscount
  - CPQInputConfiguration
  - CPQStandard Discount
  - CPQ_Default_Configuration_Entry
  - CPQ_Default_Error_Configuration
  - CPQ_Default_Header_Configuration
  - CPQ_Default_Input_Output_Configuration

Publishing Templates

- MS Word template - for Quote document

Workflow Logics

- CPQ_Quote

Dependencies

This accelerator depends on other following accelerators, which will be deployed during the installation too.

- Shared Library
- Formula Evaluator Library
- Approval Workflow Library

CPQ Package Architecture

Library Dependency

- ApprovalWorkflow
- FormulaEvaluator

Data Structures

Mandatory Tables

Table Type	Table Name	Mandatory Fields	Optional Fields	Description
Product Extension	ProductCosts	• Product Id • Cost	• Currency • Target Date • Valid To Date • Additional dimensions filter	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 • Target date	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	• Target date	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 discount a value belongs. It can also be defined if the discount value is a percentage or absolute value.
Pricing Parameter	ExceptionDiscount	• Customer (key1) • Product (key2) • Discount (attribute1)	• 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.
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	CPQOutputCo	• Name	• User group	Contains configurations for outputs.

Parameter	nfiguration			

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 Logics**
  - 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.

Technical User Reference (CPQ)

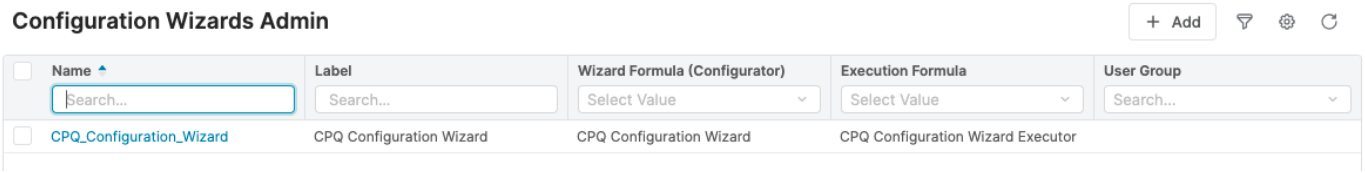
CPQ Configuration Wizard

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

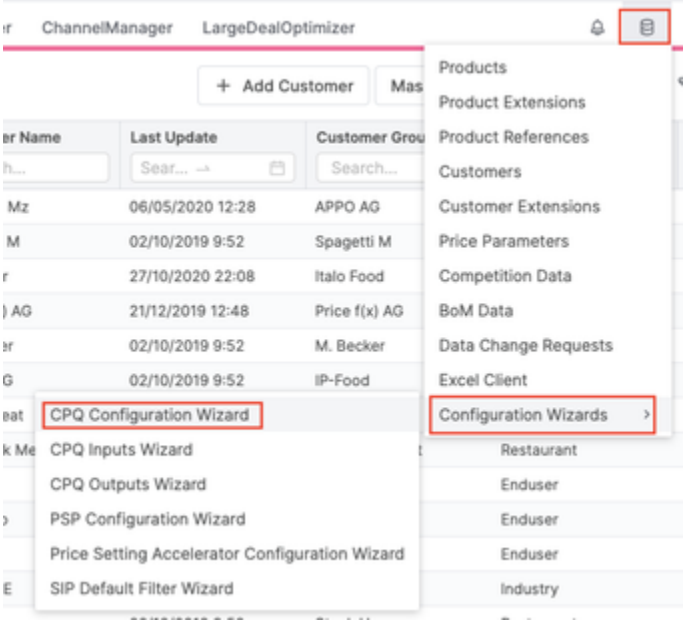
Configuration Wizards Admin

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

Go to **Configuration** > **Configuration Wizards** > **Configuration Wizards Admin**:

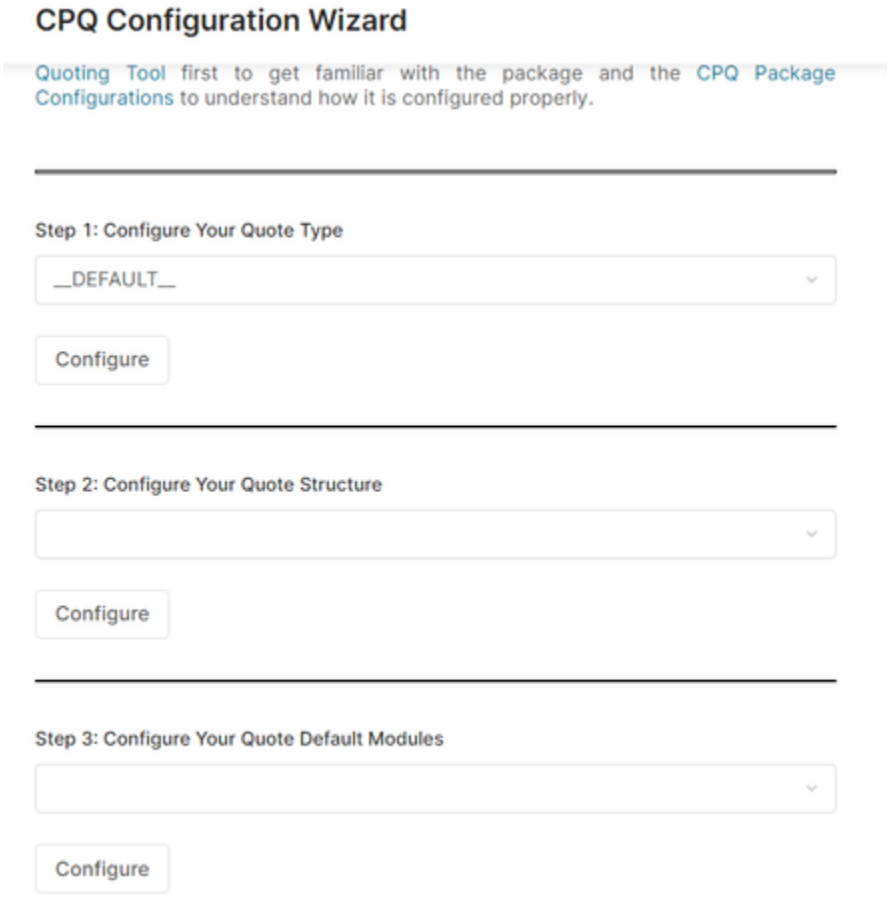


The Configuration Wizards Admin is already configured after deployment, you can use it for the configuration. Go to **Master Data** > **Configuration Wizards** > select **CPQ Configuration Wizard**.



Wizard Steps

The CPQ Configuration Wizard consists of three main steps which guide you through the configuration of each part of the CPQ.

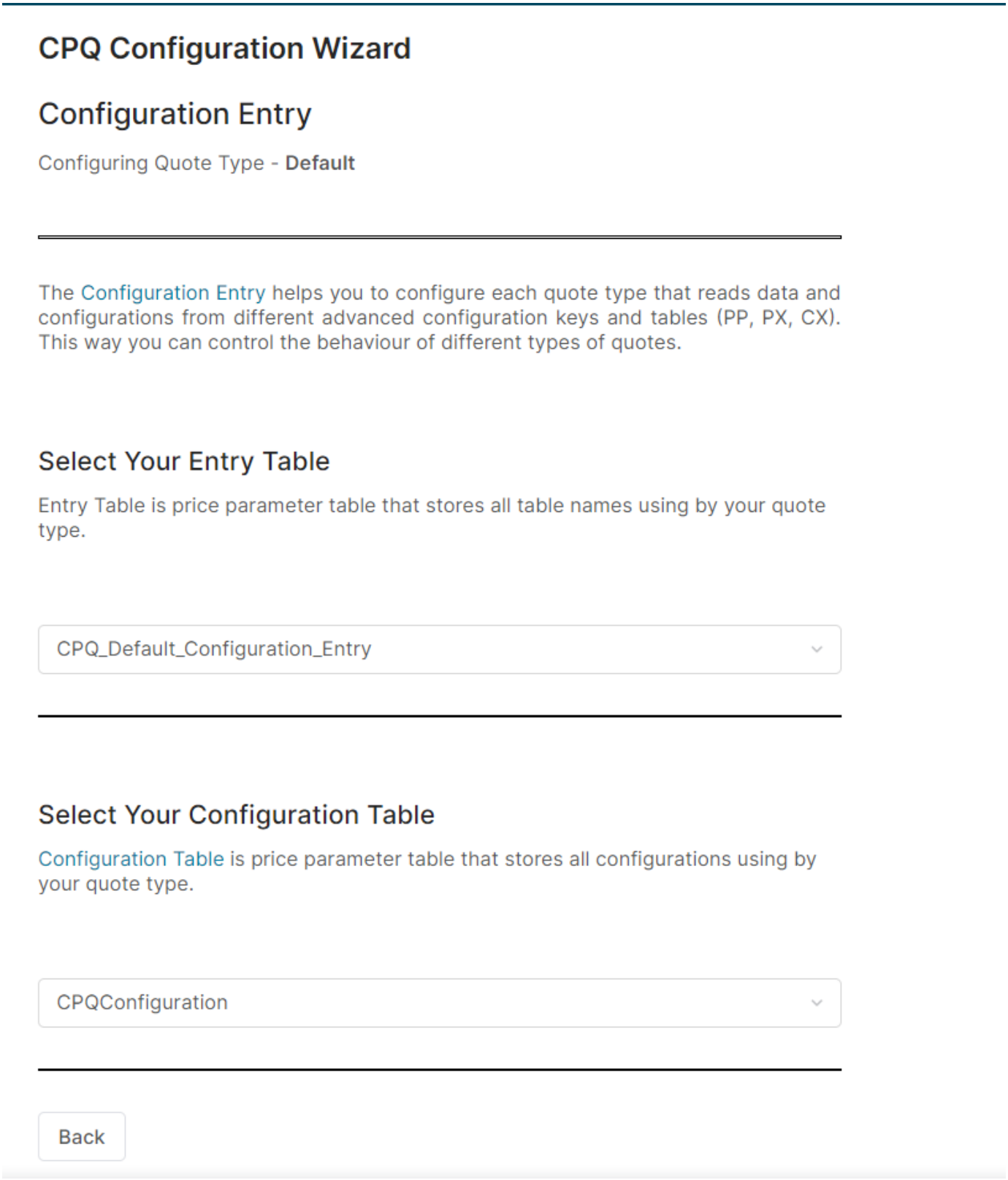


Step 1: Configure Your Quote Type

In this step, a list of quote types using the CPQ_Header logic is displayed to select from. The next configuration steps are applied to the selected quote type.

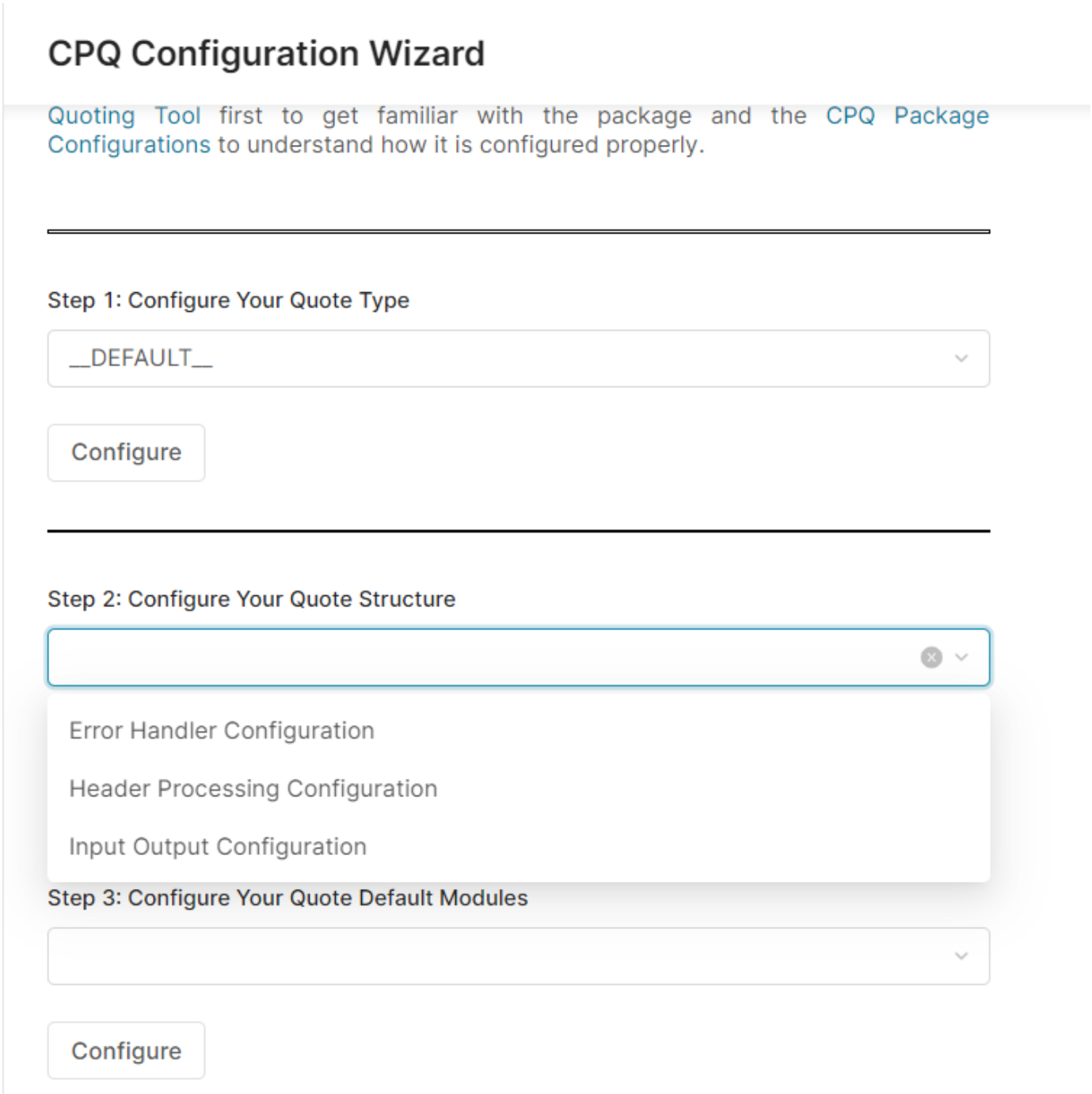
Then, you can specify the tables where the configurations/data should be used for the selected quote type by clicking the **Configure** button.

If you need a different table for each quote type, you need to prepare it yourself (the wizard does not support cloning/creating tables).



Step 2: Configure Your Quote Structure

Allows you to configure quote structures including input/outputs, header processing, and errors configuration.



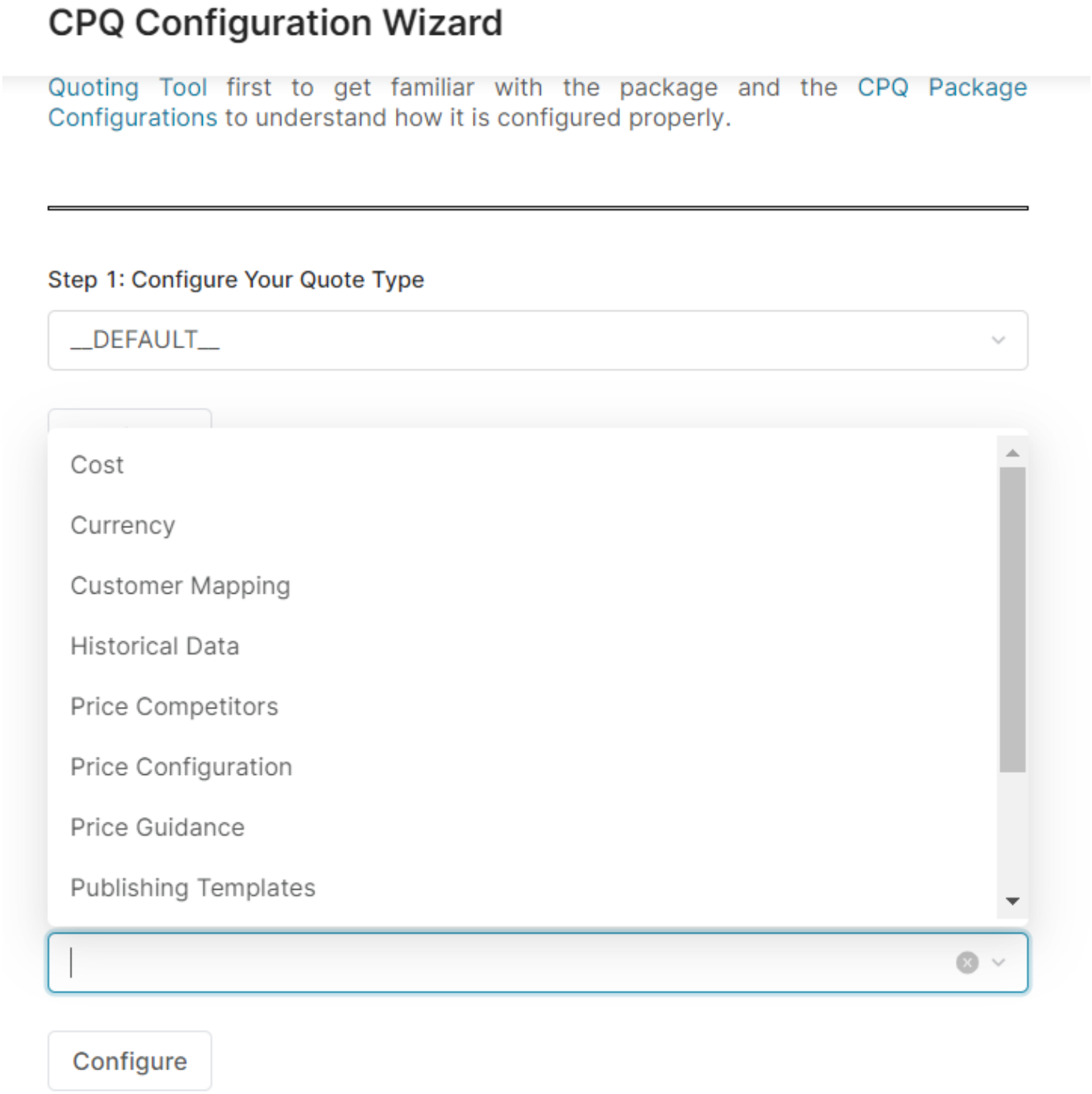
Step 3: Configure Your Quote Default Modules

This step allows you to configure the core modules of CPQ:

- Cost
- Currency
- Customer Mapping
- Historical Data
- Price Competitors
- Price Configuration
- Price Guidance
- Publishing Templates

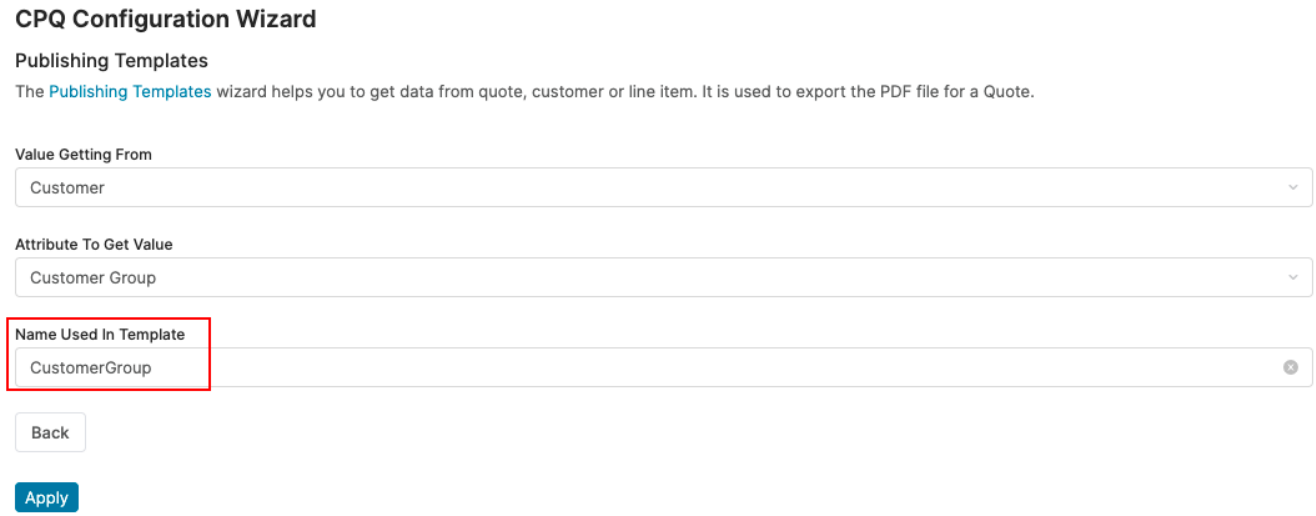
- Rebate
- Discounts
- Other Configurations

For details see How To.

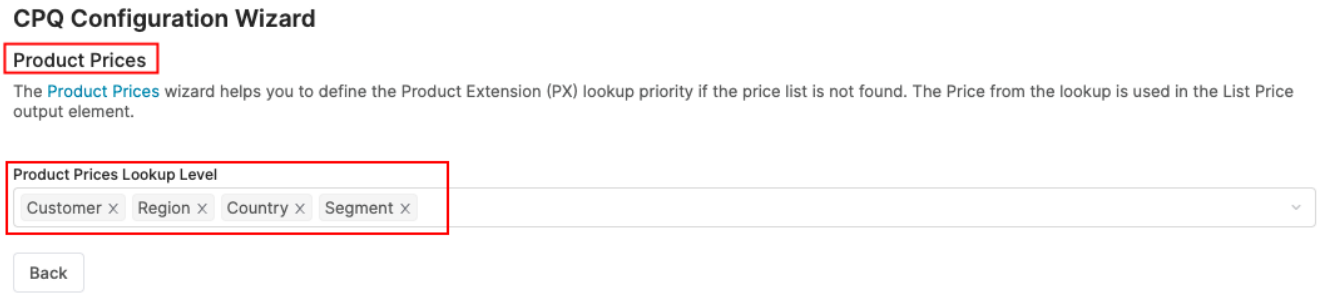


Note:

- There is a known issue when you input the value in the last row as below. You should press the Tab key or click outside this row, then the value is saved and applied.



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



- If you want to delete a configuration for the required fields, you need to delete it manually in Company Parameters (e.g. Cost in CPQConfiguration, Product_Cost_PX_Name in CPQ_Default_Configuration_Entry,..) where it is saved. For this case, the default one is used. See CPQ Package Architecture.

CPQ Configuration Wizard - Details on Modules

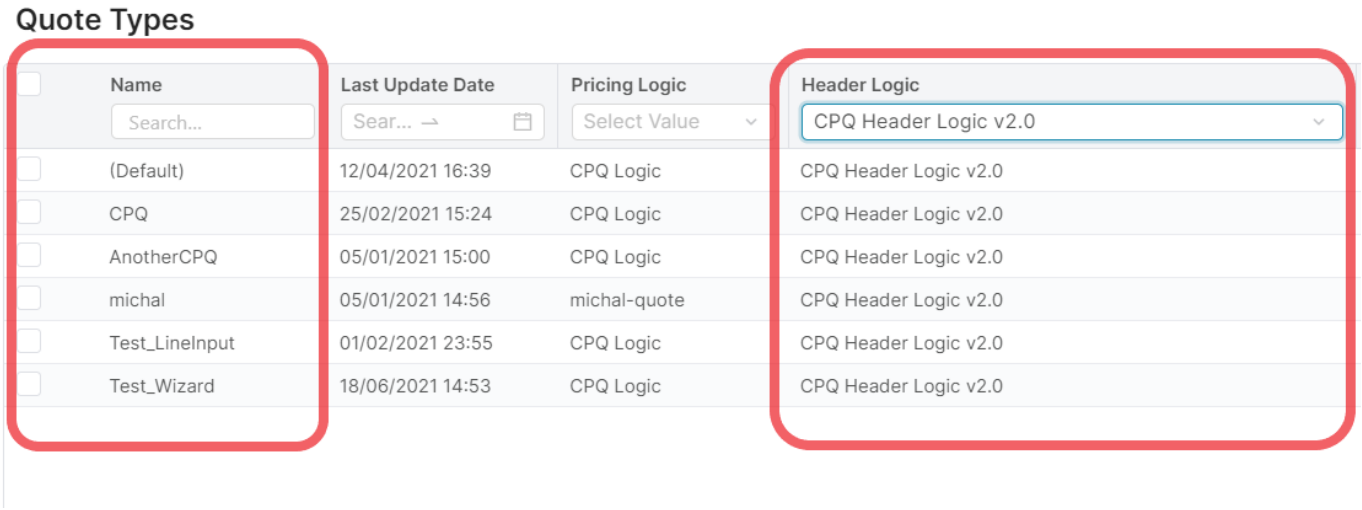
- Add Quote Type To Be Configured by Wizard
- Separate Configurations per Quote Type
- Configure Custom Library Running on Header
- Configure Custom Inputs and Outputs
- Configure Error Handler
- Configure Historical Data
- Configure Product Price Extension Lookup Order
- Configure Customer Mapping
- Configure Standard Discount
- Configure Rebate Integration Configuration
- Configure Cost Configuration
- Configure Price Competitors Configuration
- Configure Price Guidances Configuration
- Troubleshooting

Add Quote Type To Be Configured by Wizard

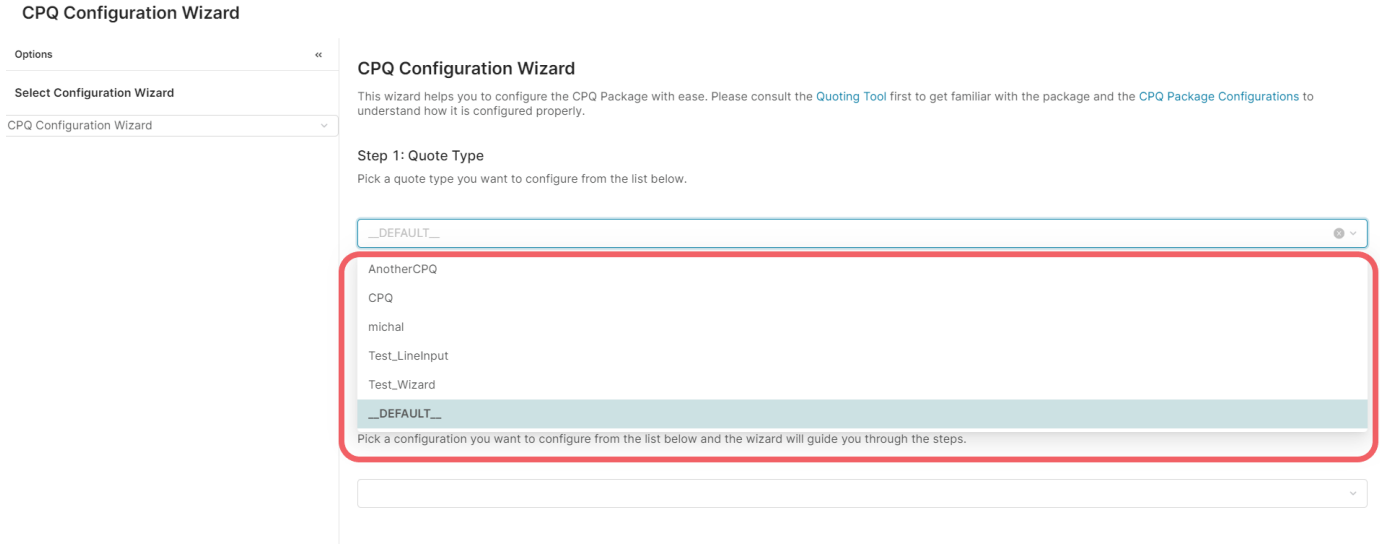
> Configuration Entry should be configured for each quote type, see Configuration Per Quote Type.

When CPQ Configuration Wizard is used for CPQ, it loads only Quote Types that use the CPQ_Header logic.

So, you need to select *CPQ_Header* logic as the **Header Logic** of your Quote Type.



Then it is ready to be configured by the wizard.



Separate Configurations per Quote Type

Each quote type can be configured to have different:

- Tables where the custom logics are configured for running on header/line items.
- Tables where data (such as cost, discount, configurations, etc.) are read from.

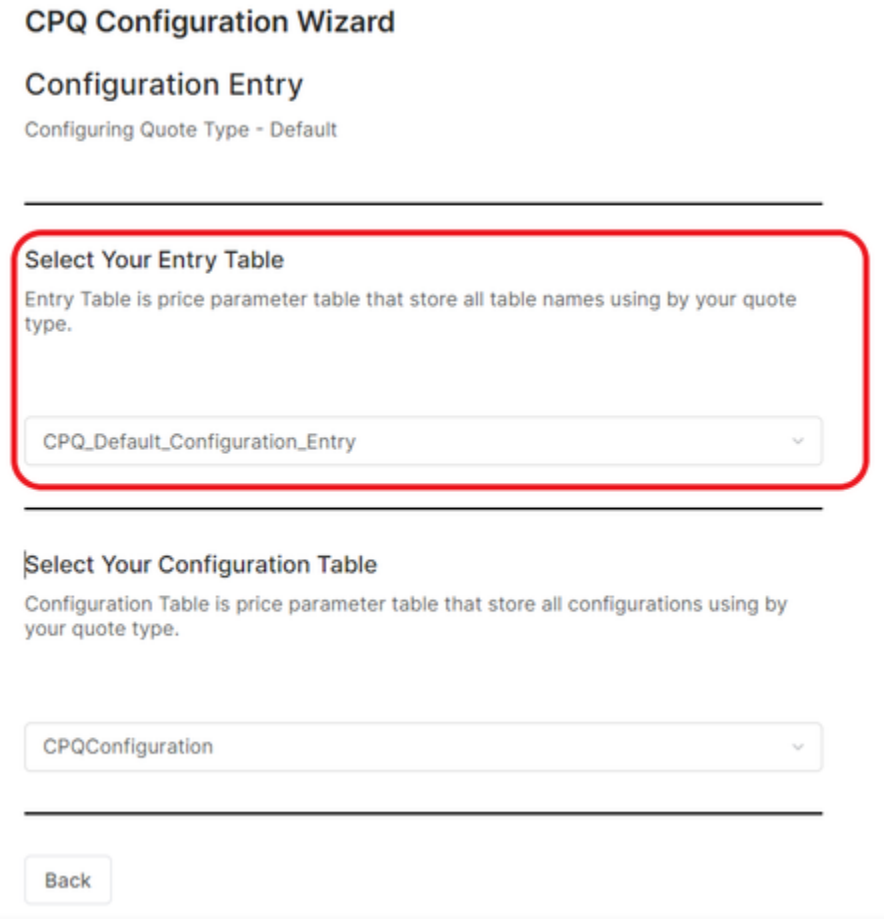
Pre-requisites

- The tables should be cloned/created manually before starting the wizard.
- Quote Type should be selected in Step 1.

- After the quote type is selected, click the **Configure** button, a new page will show and allow you to configure where to read the configuration and other data tables.

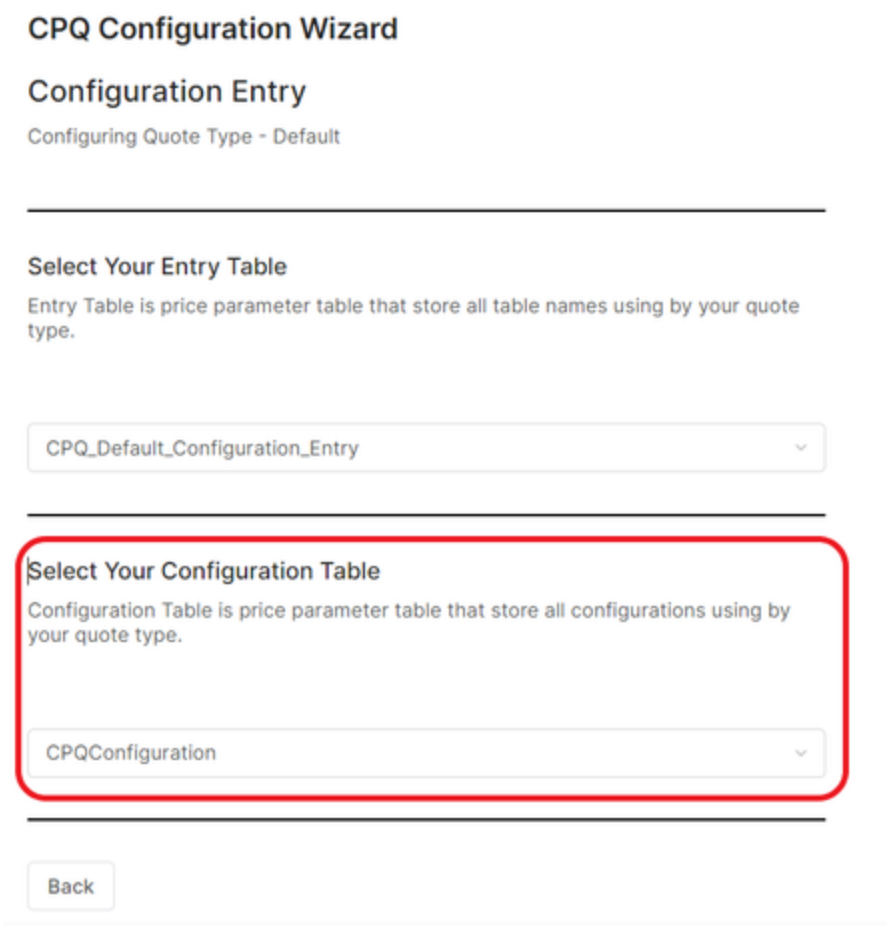
Separate Quote Configuration

Select the PP table that should store all other table names that are used by your quote.



Separate Data

Select the PP table that should store all default modules configurations that are used by your quote.



Apply Your Changes

Once you have all data selected, you need to click the **Apply** button to store your selections in the system.

If you click the **Back** button, you go back to the main screen and the current changes on the screen are ignored.

Configure Custom Library Running on Header

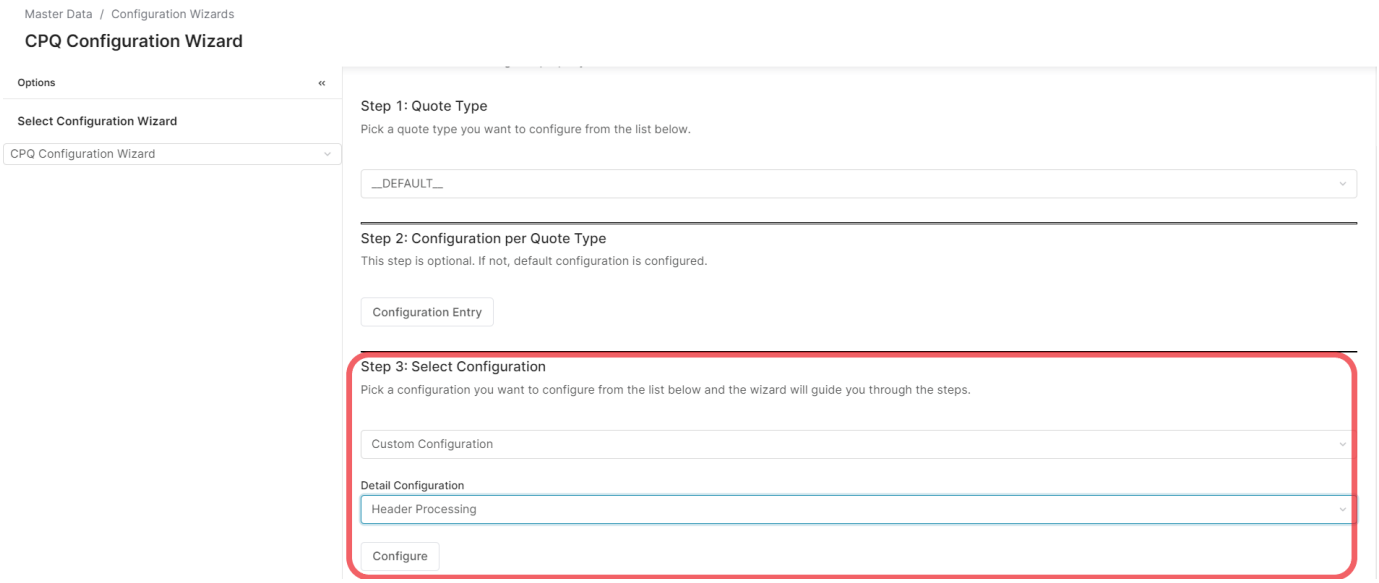
CPQ allows custom logics to be configured and run along with the default logics. The default logics can also be removed/replaced in the calculation. This can easily be done by using the CPQ Configuration Wizard.

Pre-requisites

- Custom logics are prepared in the form of a Pricefx Groovy library.
- Quote Type is selected in Step 1.
- Tables are configured in Step 2.

Select Configuration

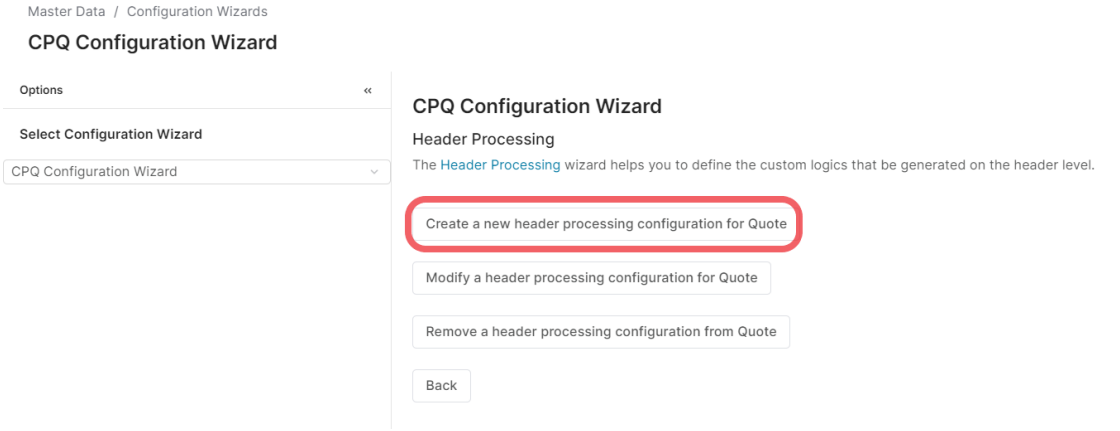
In Step 3, select **Custom Configuration** in the first drop-down input and **Header Processing** in the second input.



Then click the **Configure** button to enter the configuration screen.

Creating New Entry

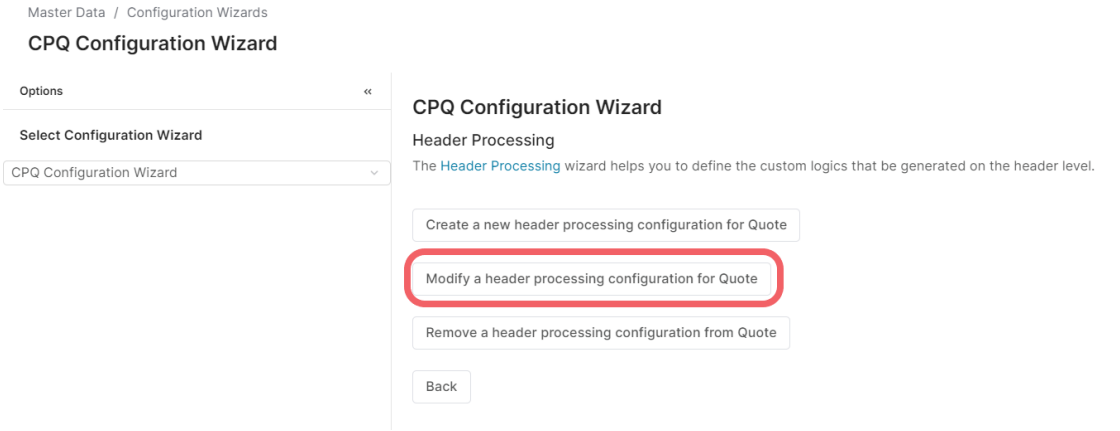
On the configuration screen, select **Create a new header processing configuration for Quote**.



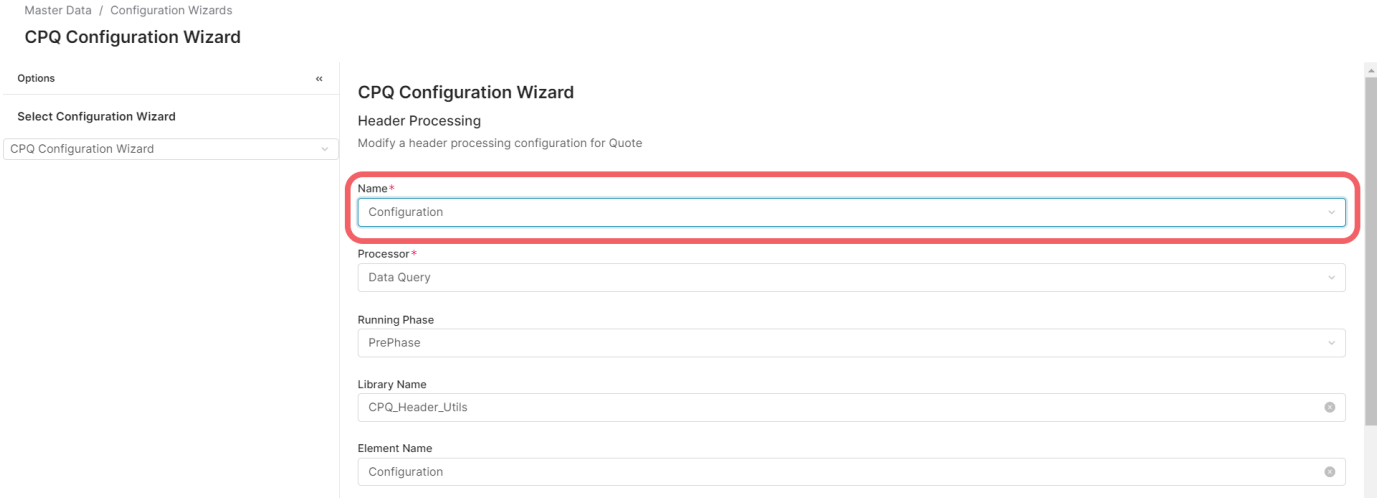
Fill in the inputs and click the **Apply** button to save the entry.

Modify an Entry

On the configuration screen, select **Modify a header processing configuration for Quote**.



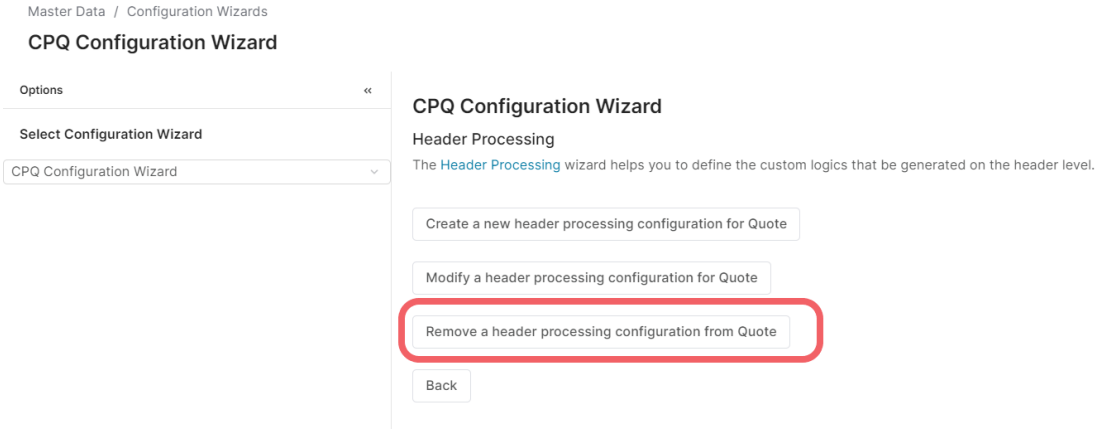
Select an existing entry from the **Name** input. The configuration for the selected entry is loaded and ready for modifying.



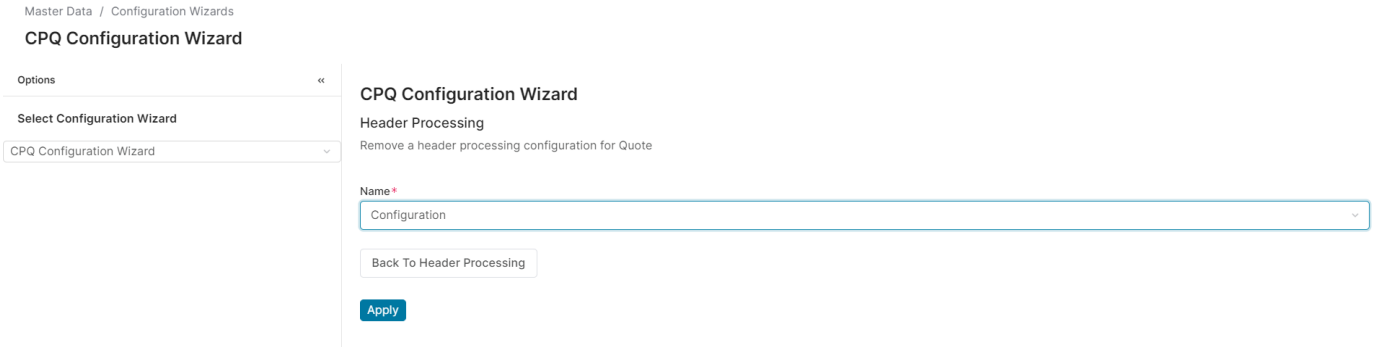
Click the **Apply** button to save the configuration when finished.

Remove an Entry

On the configuration screen, select **Remove a header processing configuration from Quote**.



Select an entry from the **Name** input.



Then click **Apply** button to remove the entry.

Configure Custom Inputs and Outputs

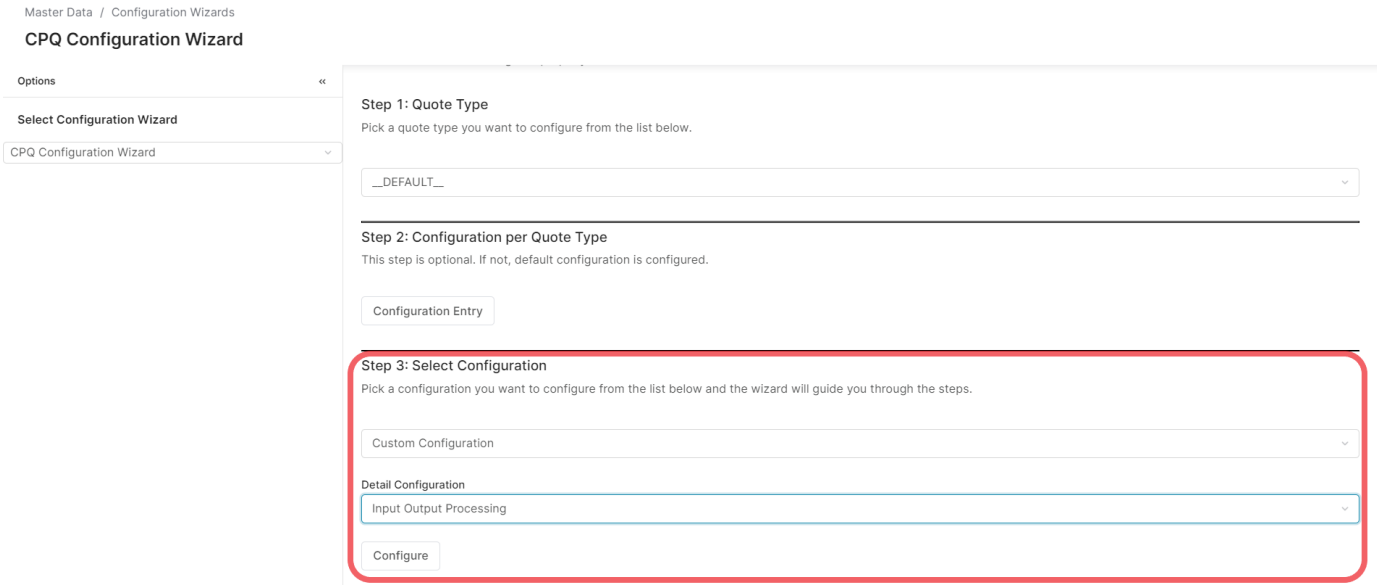
CPQ allows custom inputs/outputs to be configured and generated along with the default inputs/outputs. The default inputs/outputs can also be removed/replaced in the calculation. This can be easily done by using the CPQ Configuration Wizard.

Pre-requisites

- Custom logics are prepared in the form of a Pricefx Groovy library.
- Quote Type is selected in Step 1.
- Tables are configured in Step 2.

Select Configuration

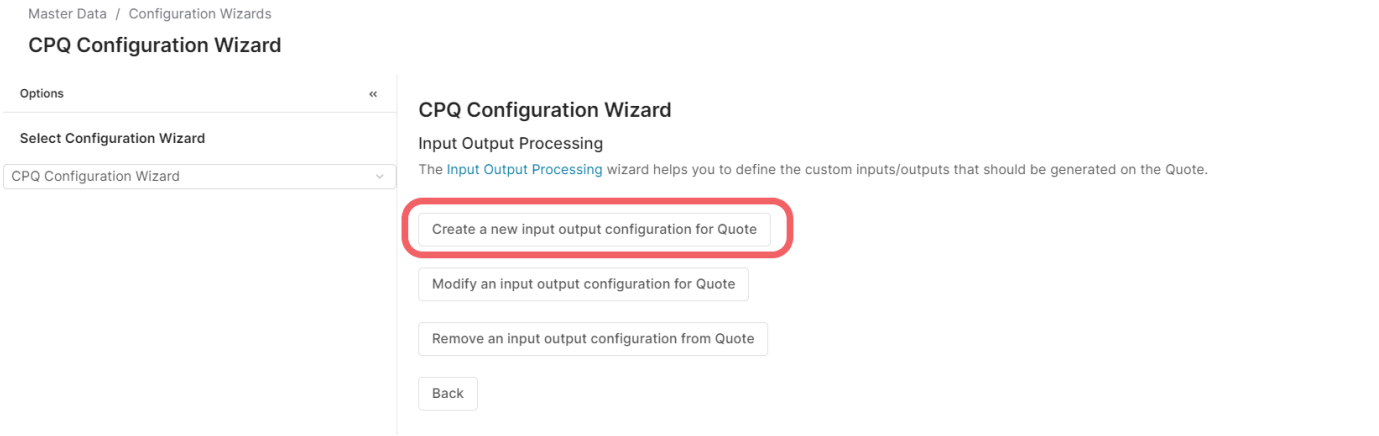
In Step 3, select **Custom Configuration** in the first drop-down input and **Input Output Processing** in the second input.



Then click the **Configure** button to enter the configuration screen.

Create New Entry

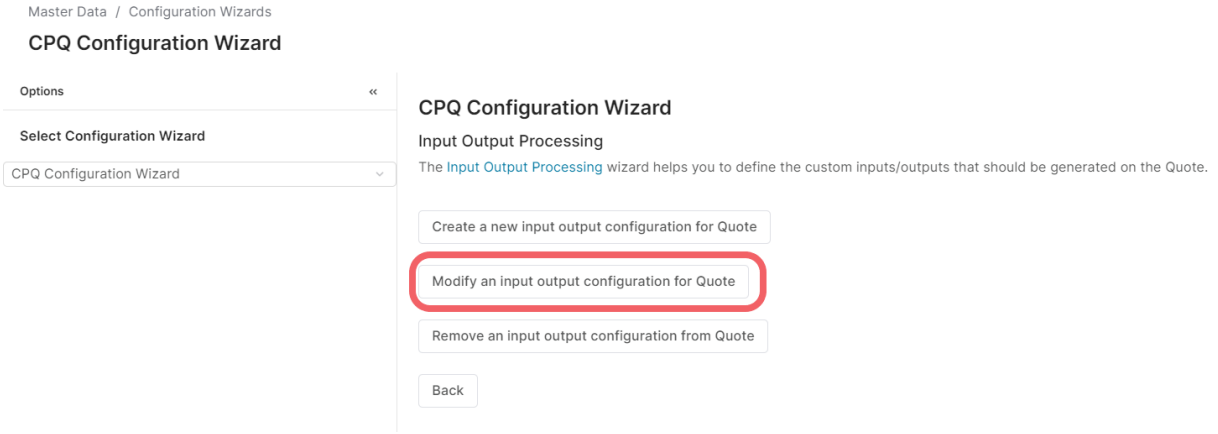
On the configuration screen, select **Create a new input output configuration for Quote** to enter the create new entry screen.



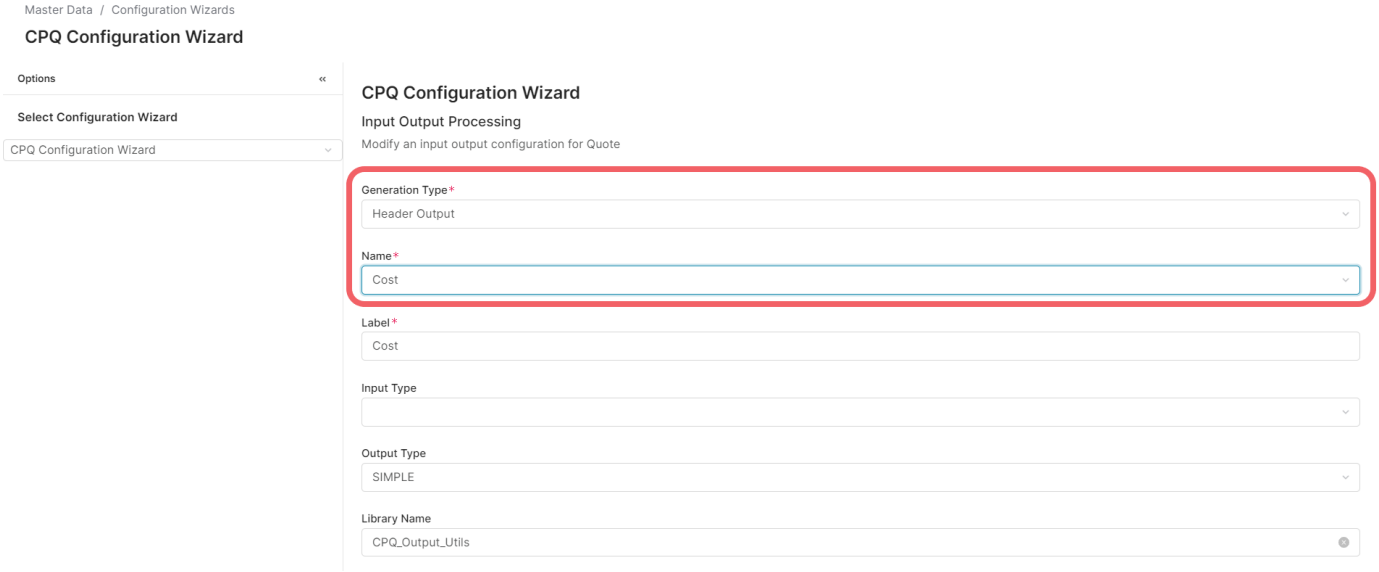
Fill in the inputs in the create new entry screen and click the **Apply** button to save the entry.

Modify Entry

On the configuration screen, select **Modify an input output configuration for Quote** to enter the modify existing entry screen.



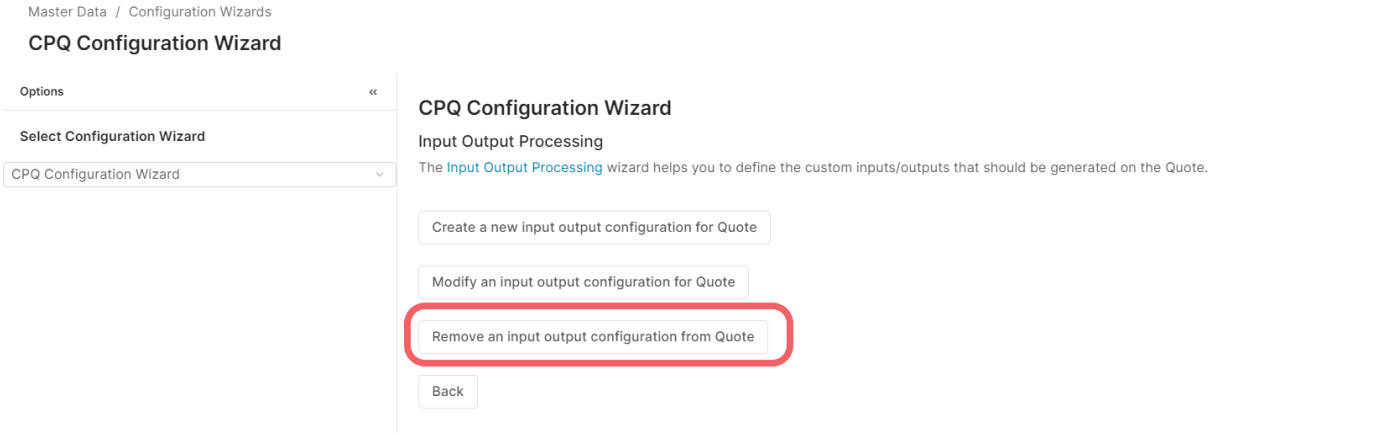
Select an existing entry from the **Generation Type** and **Name** inputs. The configuration for the selected entry is loaded and ready for modifying.



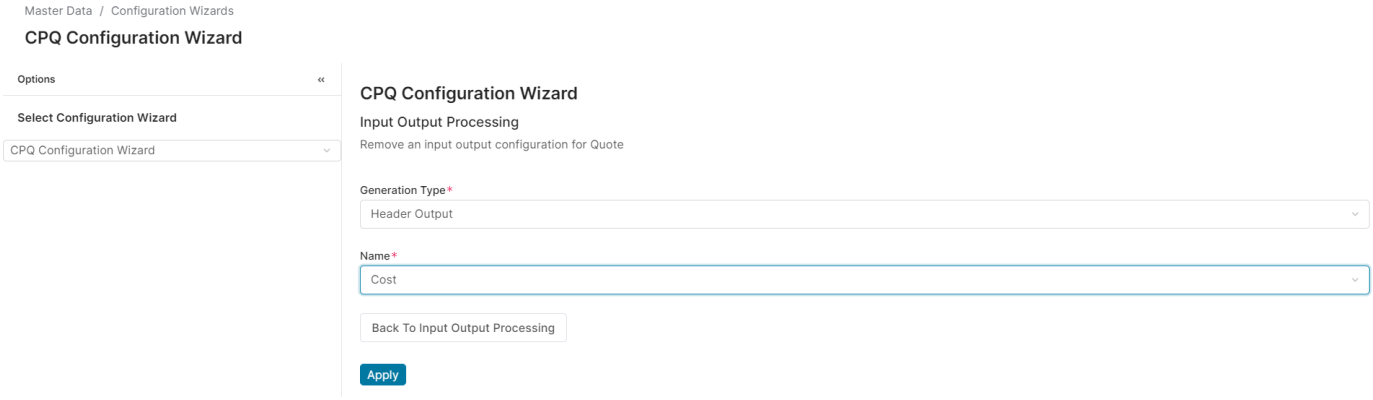
Click the **Apply** button to save the configuration when finished.

Remove Entry

On the configuration screen, select **Remove an input output configuration from Quote**.



Select an entry from the **Generation Type** and **Name** inputs.



Then click the **Apply** button to remove the entry.

Configure Error Handler

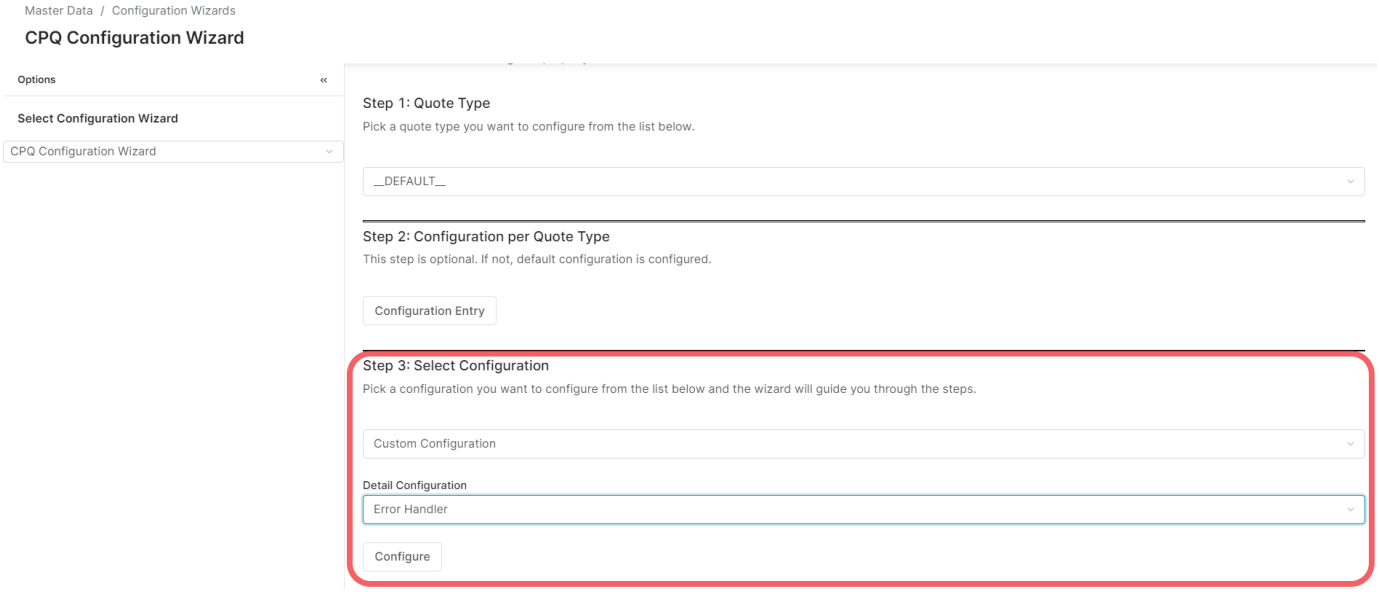
CPQ provides a tool to configure the error levels and messages. This configuration can easily be done using CPQ Configuration Wizard.

Pre-requisites

- Quote Type is selected in Step 1.
- Tables are configured in Step 2.

Select Configuration

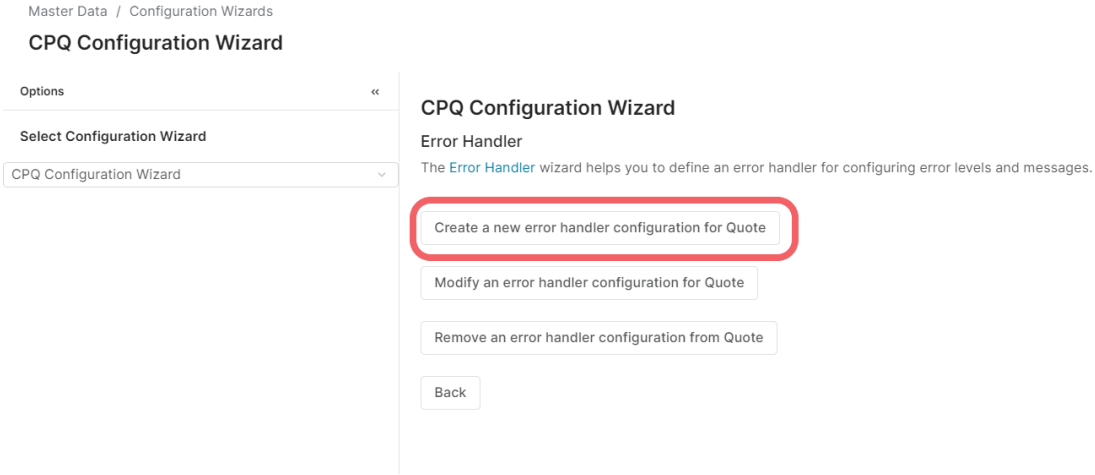
In Step 3, select **Custom Configuration** in the first drop-down input and **Error Handler** in the second input.



Then click the **Configure** button to enter the configuration screen.

Create New Entry

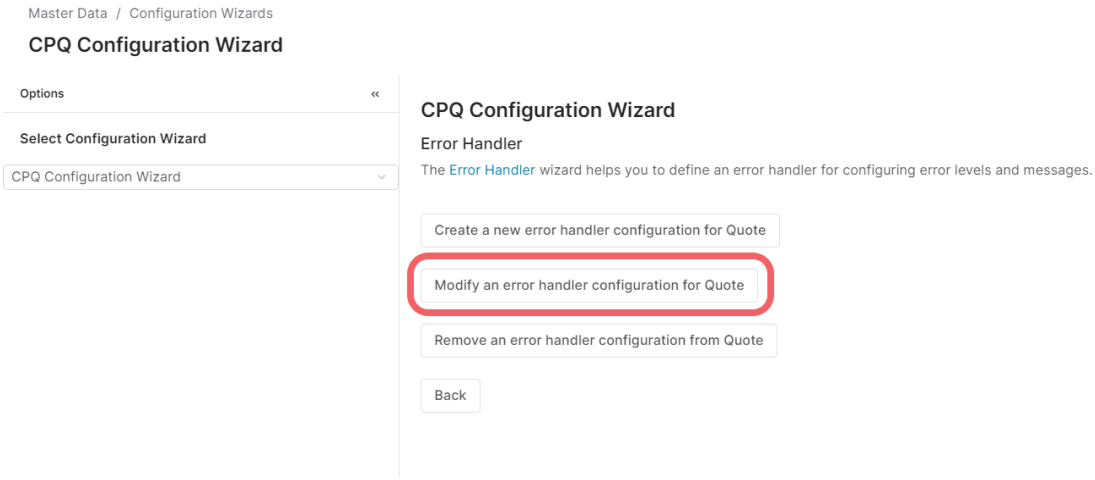
On the configuration screen, select **Create a new error handler configuration for Quote**.



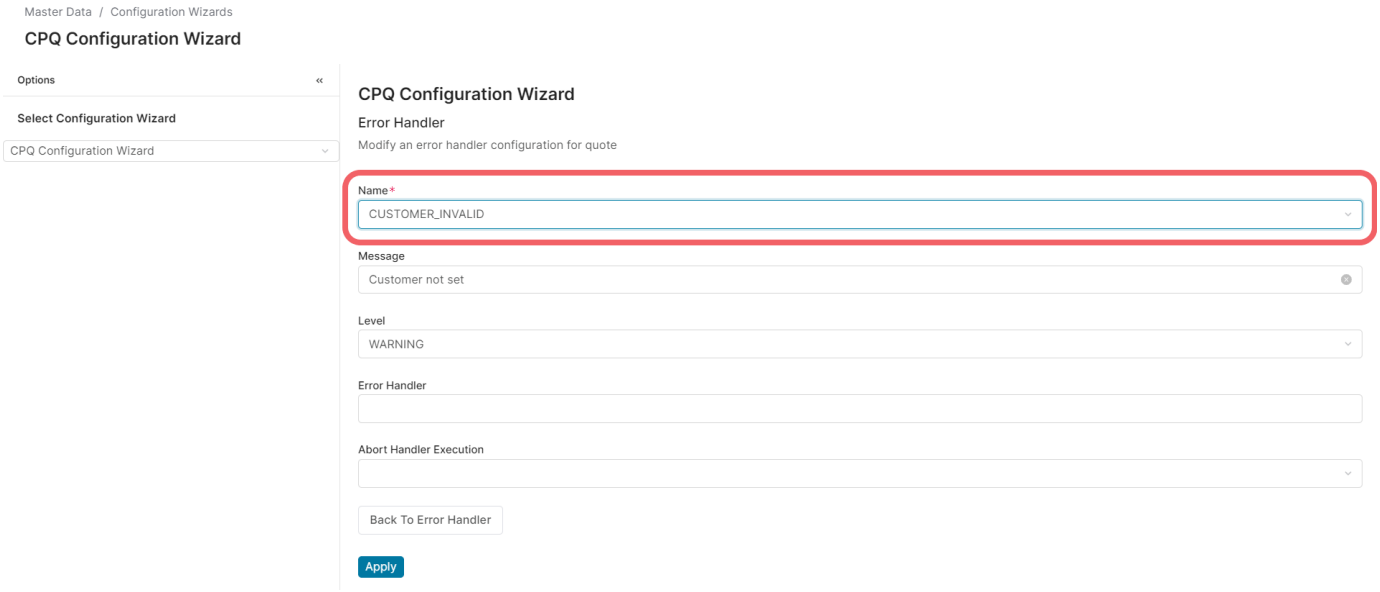
Fill in the inputs on the create new entry screen and click the **Apply** button to save the entry.

Modify Entry

On the configuration screen, select **Modify an error handler configuration for Quote**.



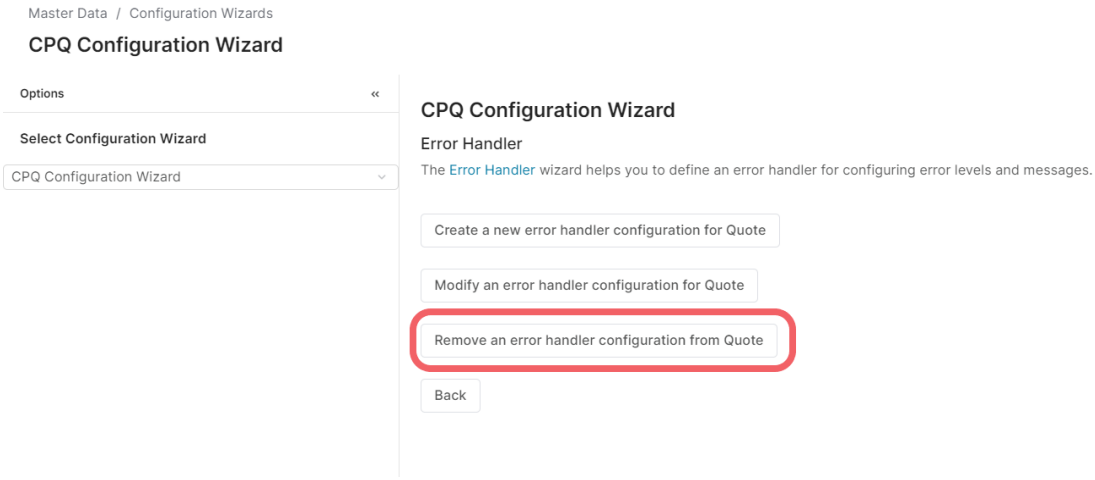
Select an existing entry from the **Name** input. The configuration for the selected entry is loaded and ready for modifying.



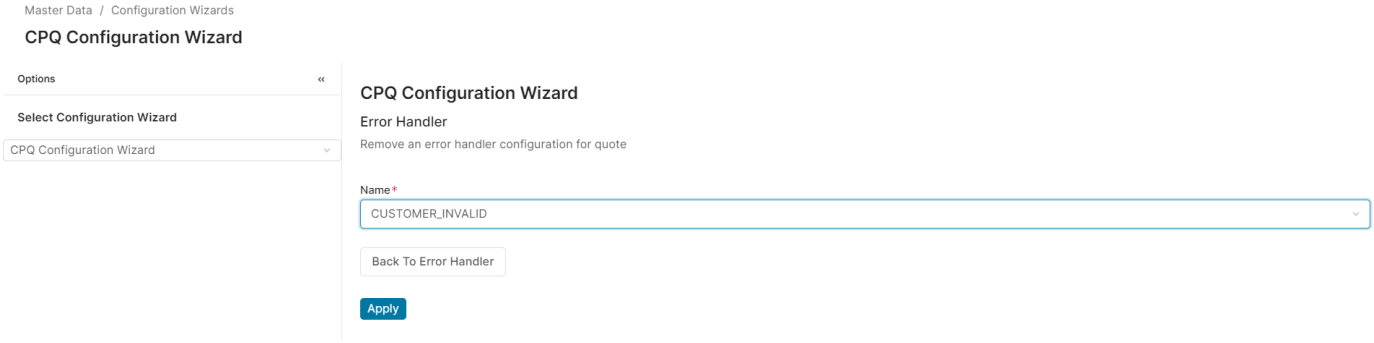
Click the **Apply** button to save the configuration when finished.

Remove Entry

On the configuration screen, select **Remove an error handler configuration from Quote**.



Select an entry from the **Name** inputs.



Then click the **Apply** button to remove the entry.

Configure Historical Data

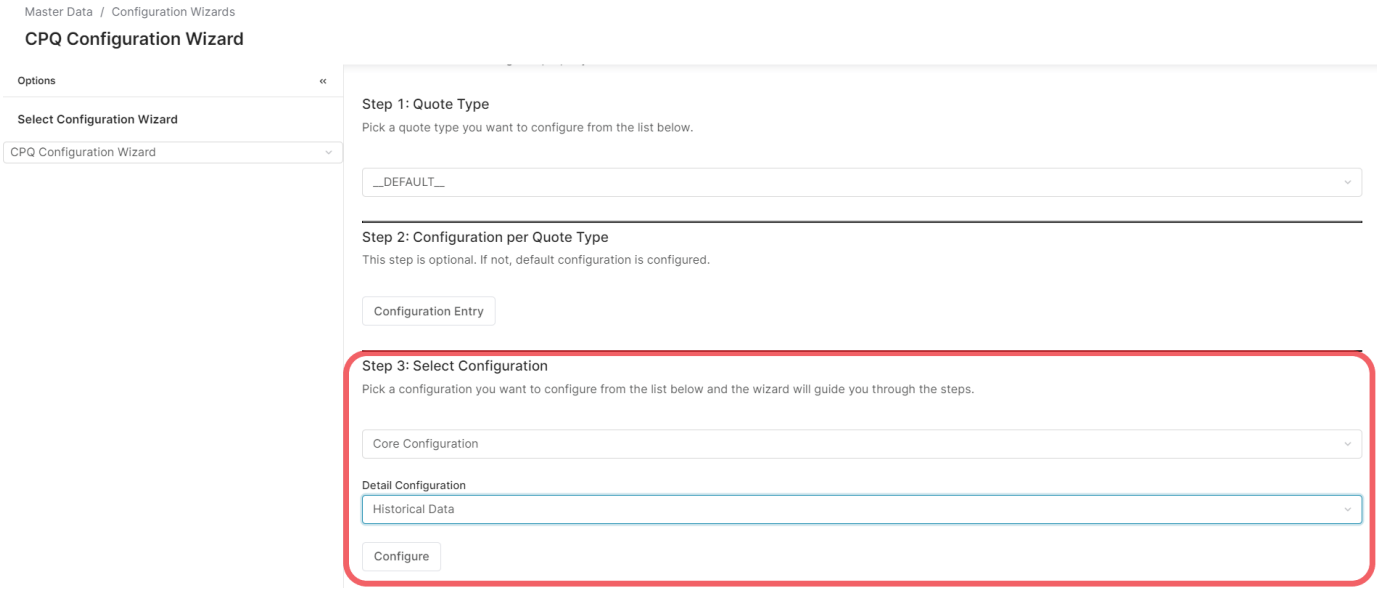
CPQ can show historical sales data for quoted customers and products. Having such data requires a connection between the sales Datamart and CPQ logic. This can be easily done by using CPQ Configuration Wizard.

Pre-requisites

- Custom logics are prepared in the form of a Pricefx Groovy library.
- Quote Type is selected in Step 1.
- Tables are configured in Step 2.

Select Configuration

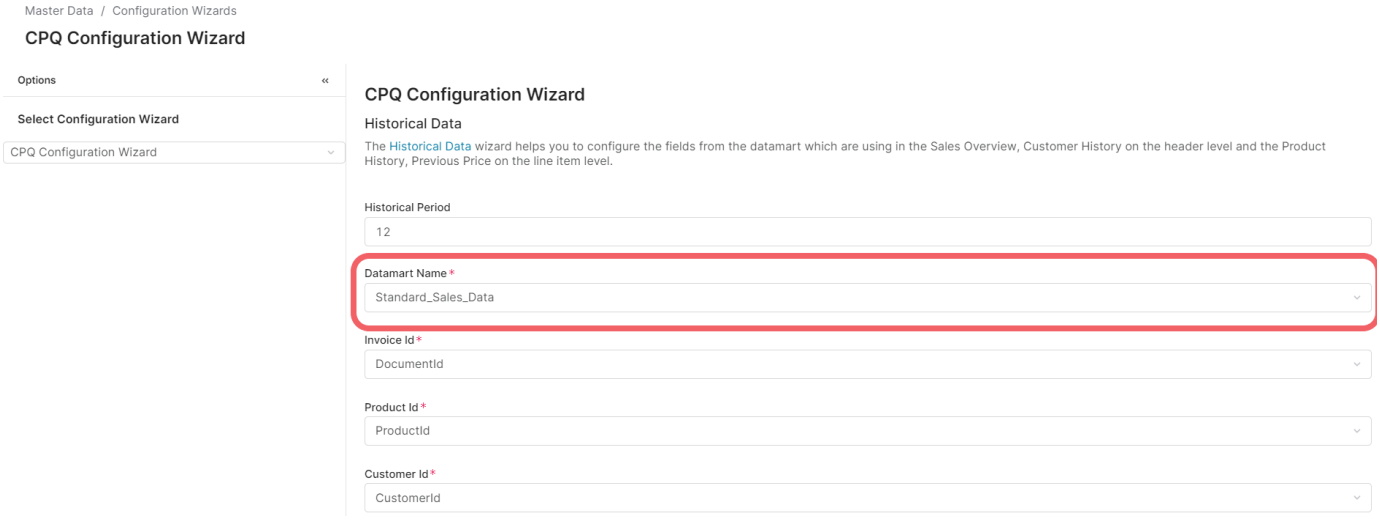
In Step 3, select **Core Configuration** in the first drop-down input and **Historical Data** in the second input.



Then click the **Configure** button to enter the configuration screen.

Historical Data Configuration

On the configuration screen, select the Datamart that contains the sales data. The Datamart fields will be fetched to select.



Map the Datamart fields and click the **Apply** button when finished.

Configure Product Price Extension Lookup Order

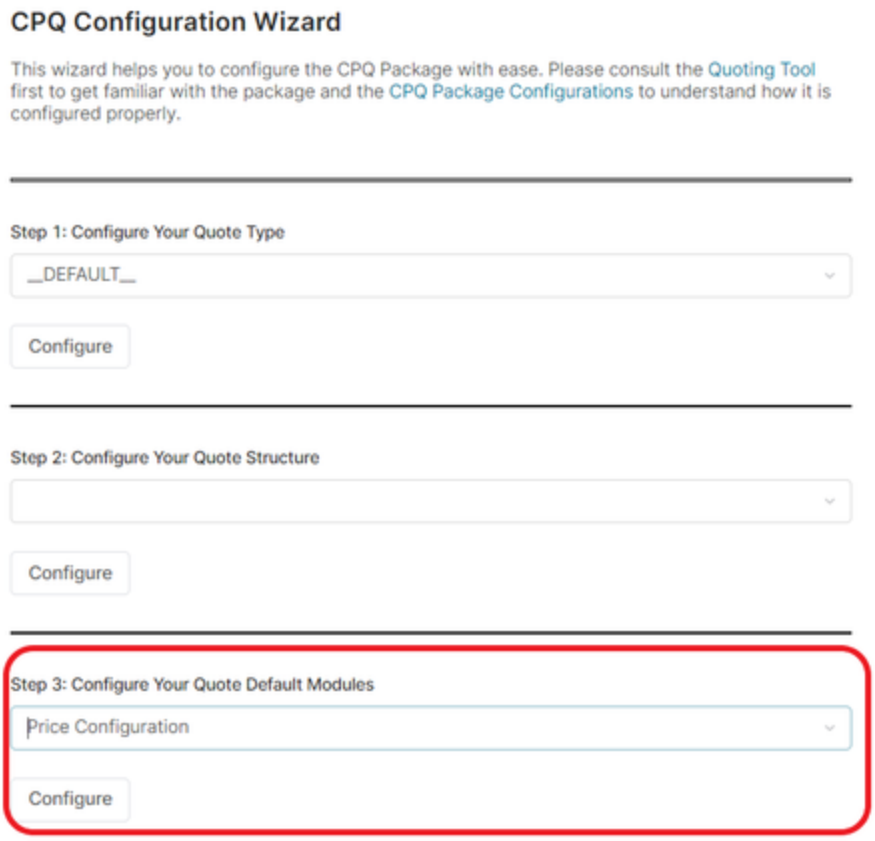
CPQ allows you to set the Product Price Extension lookup order. This can be easily done by using the CPQ Configuration Wizard.

Pre-requisites

- Quote Type is selected and configured in Step 1.

Select Configuration

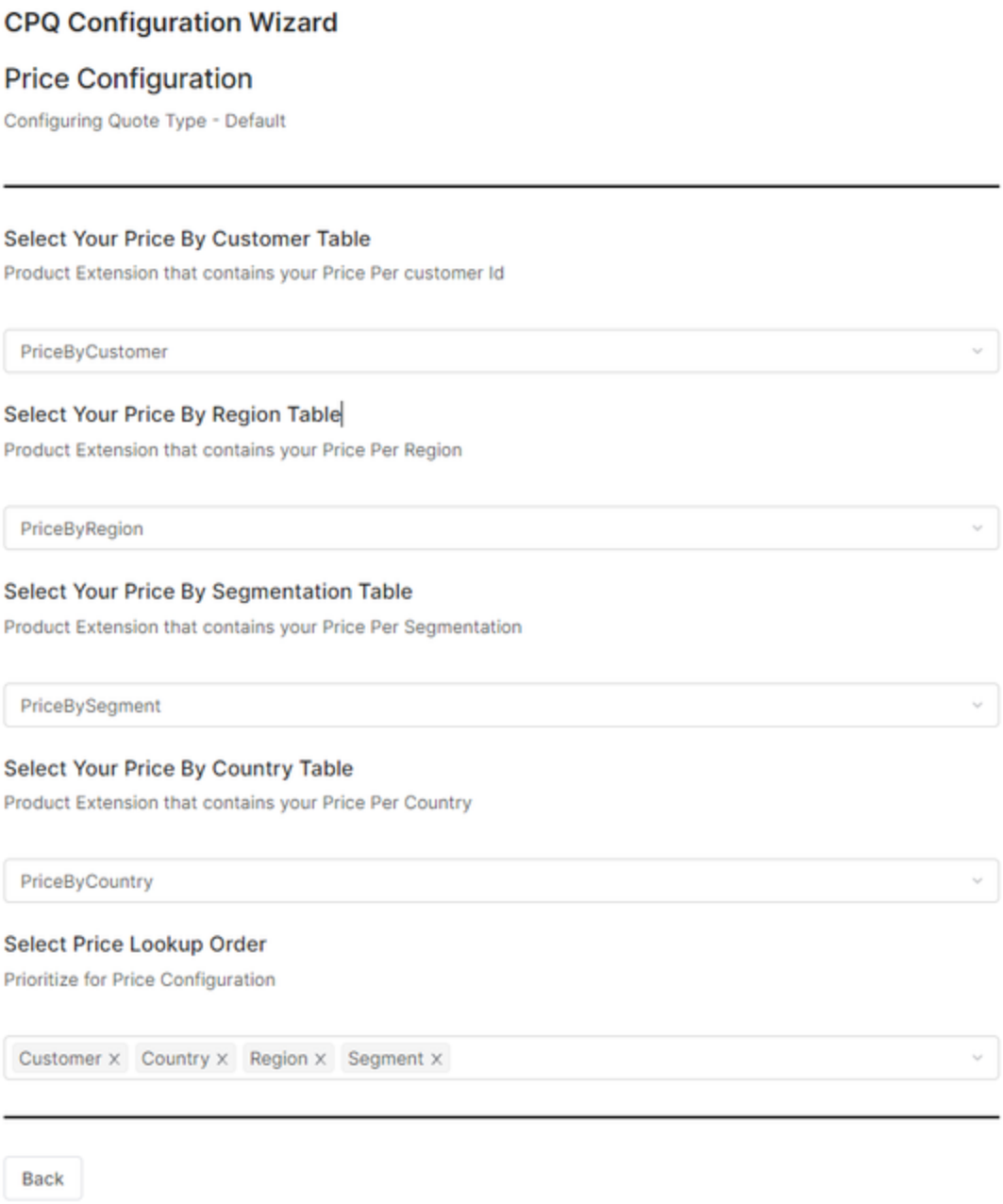
In Step 3, select **Prices Configuration** in the drop-down input.



Then click the **Configure** button to enter the configuration screen.

Price Lookup Configuration

This screen allows you to configure where and how the price data is stored in the Product Extension table.



Select your Product Extension table that contains your price data and fill the price lookup order.

Then click the **Apply** button.

Configure Customer Mapping

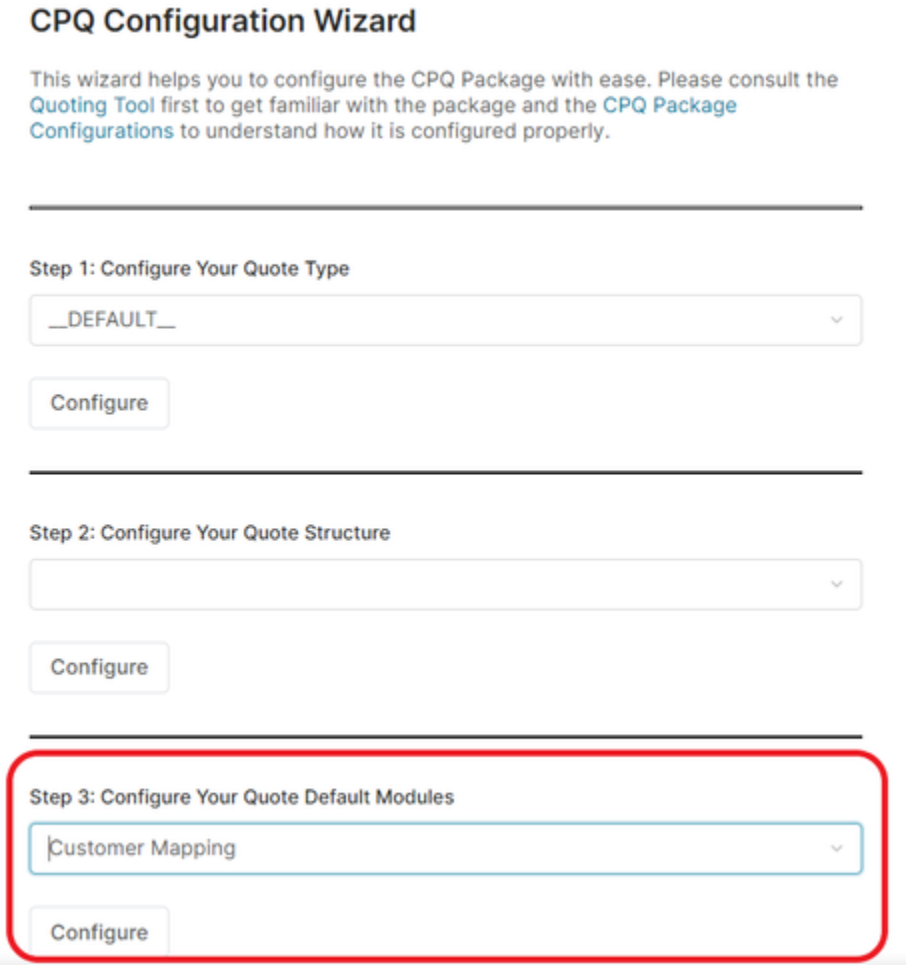
In CPQ, the Product Extension price lookup and currency conversion require mapping to the Customer Master. This can easily be done by using the CPQ Configuration Wizard.

Pre-requisites

- Quote Type is selected and configured in Step 1.

Select Configuration

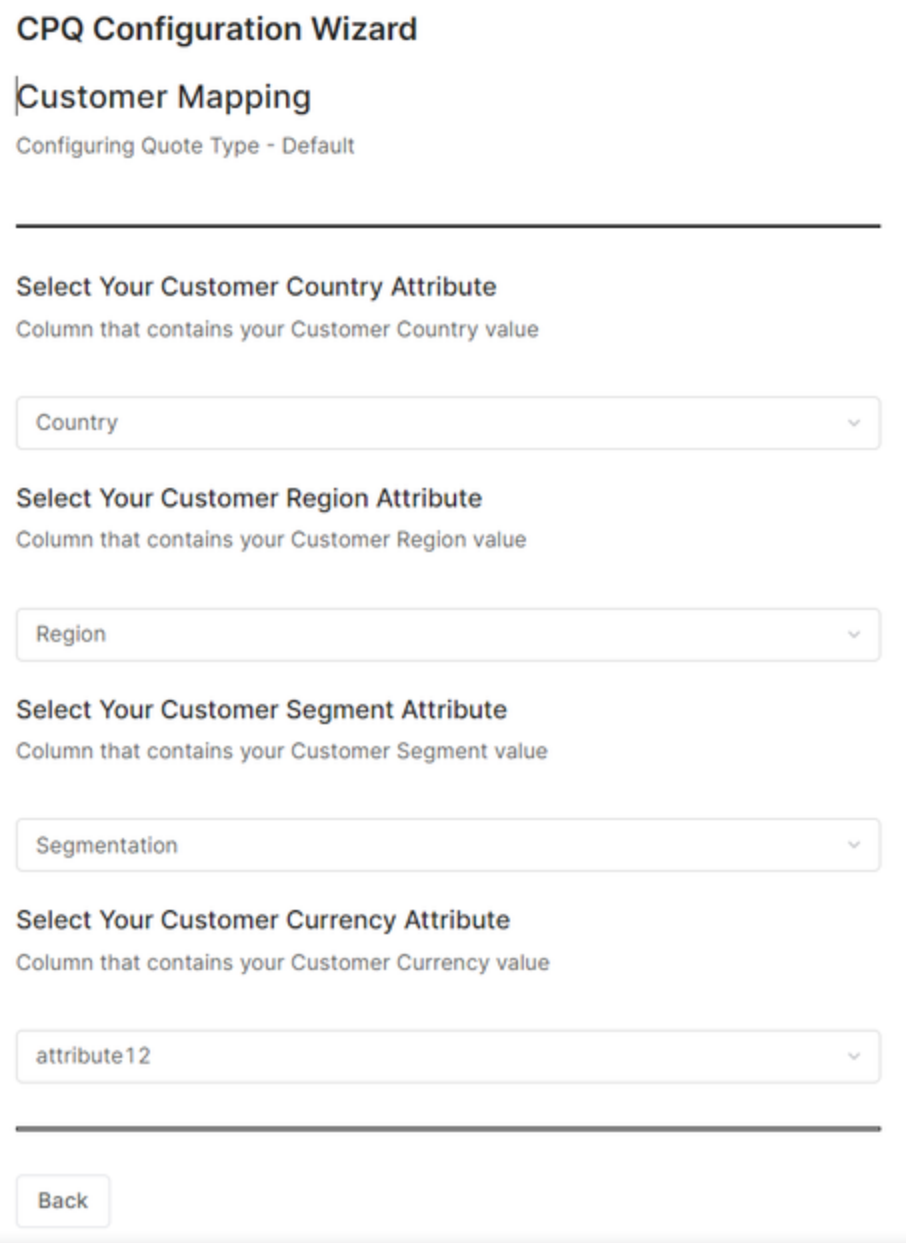
In Step 3, select **Customer Mapping** in the drop-down input.



Then click the **Configure** button to enter the configuration screen.

Customer Mapping Configuration

On the configuration screen, select the mapping from the Customer Master.



Then click the **Apply** button.

Configure Standard Discount

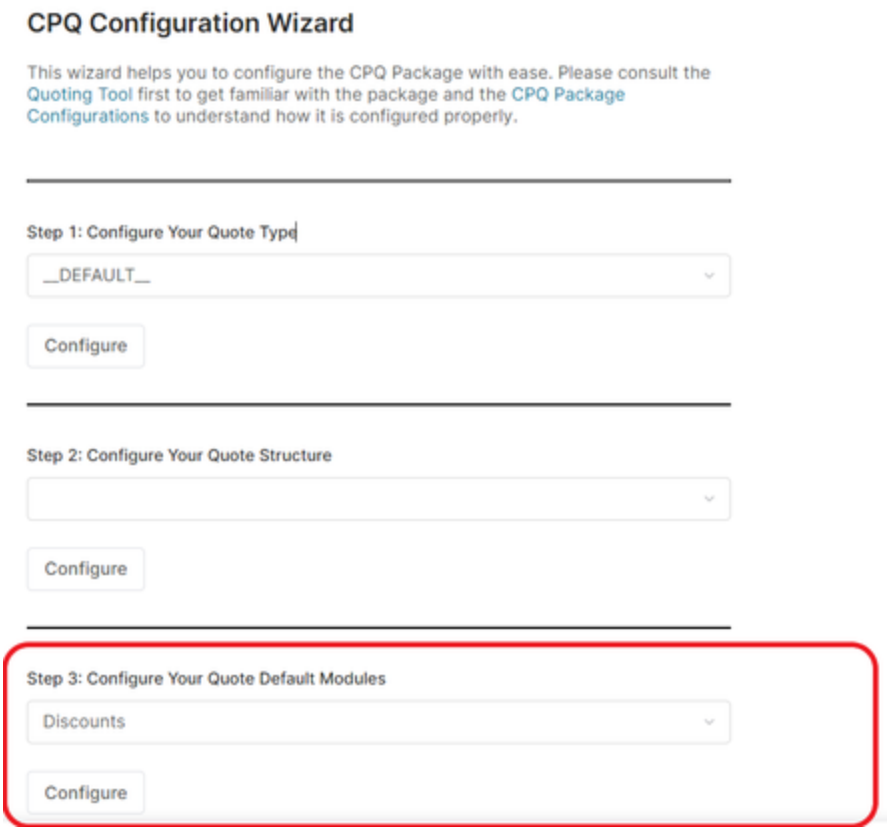
The Standard Discount table contains discount values for quoted products. The value can be configured in a combination of a maximum of 6 quote/product attributes. This can easily be done by using the CPQ Configuration Wizard.

Pre-requisites

- Quote Type is selected and configured in Step 1.

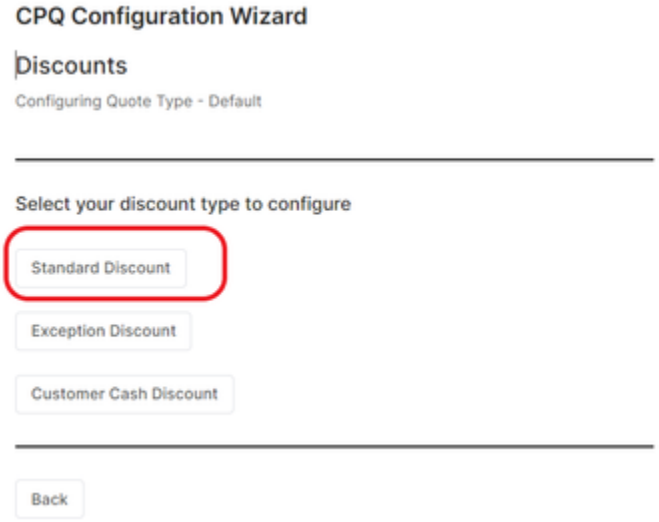
Select Configuration

In Step 3, select **Discounts** in the drop-down input.



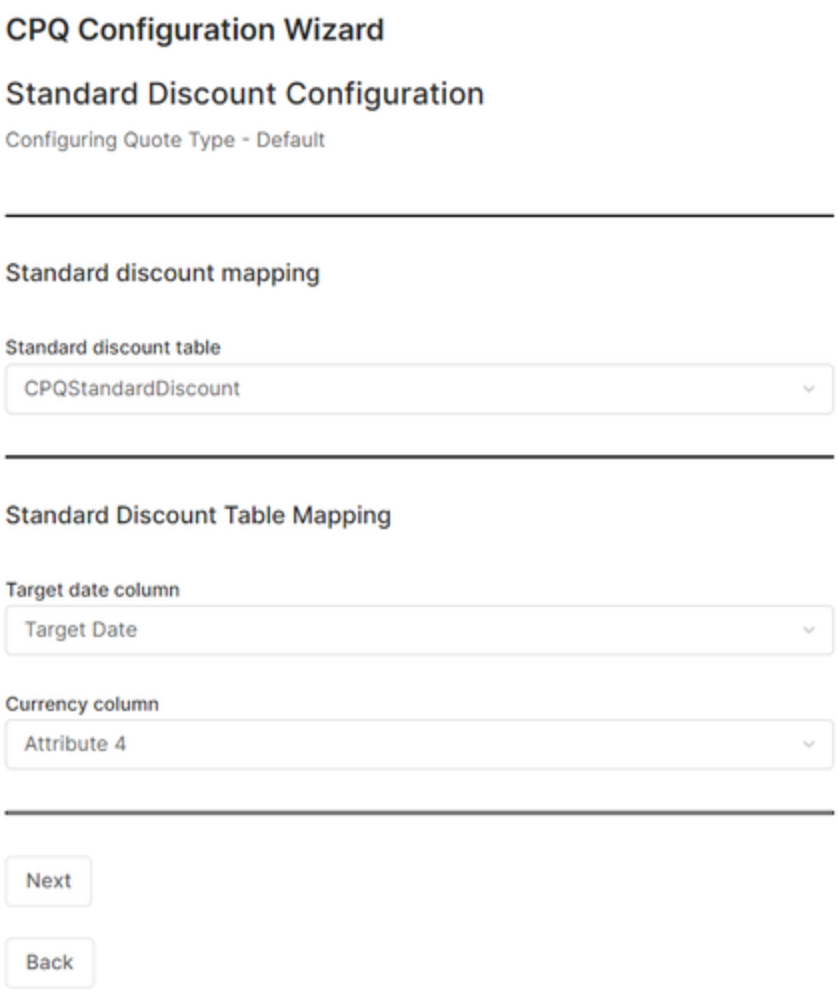
Click the **Configure** button to enter the discount type selection screen.

Then click the **Standard Discount** button to open the configuration screen.



Standard Discount Mapping Configuration

This screen allows you to configure where and how the discount data is stored in the Price Parameter table.



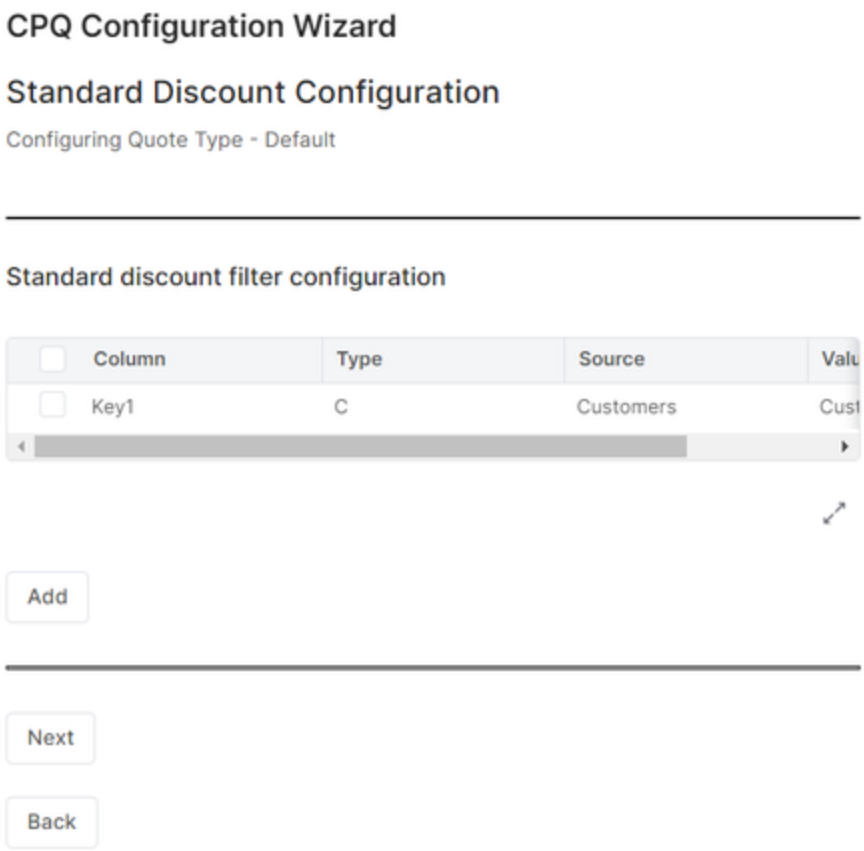
Select your Price Parameter table that contains the discount data and fill in the field mappings by selecting the available fields in the drop-down.

Then, click the **Next** button to move forward to the filter configuration screen.

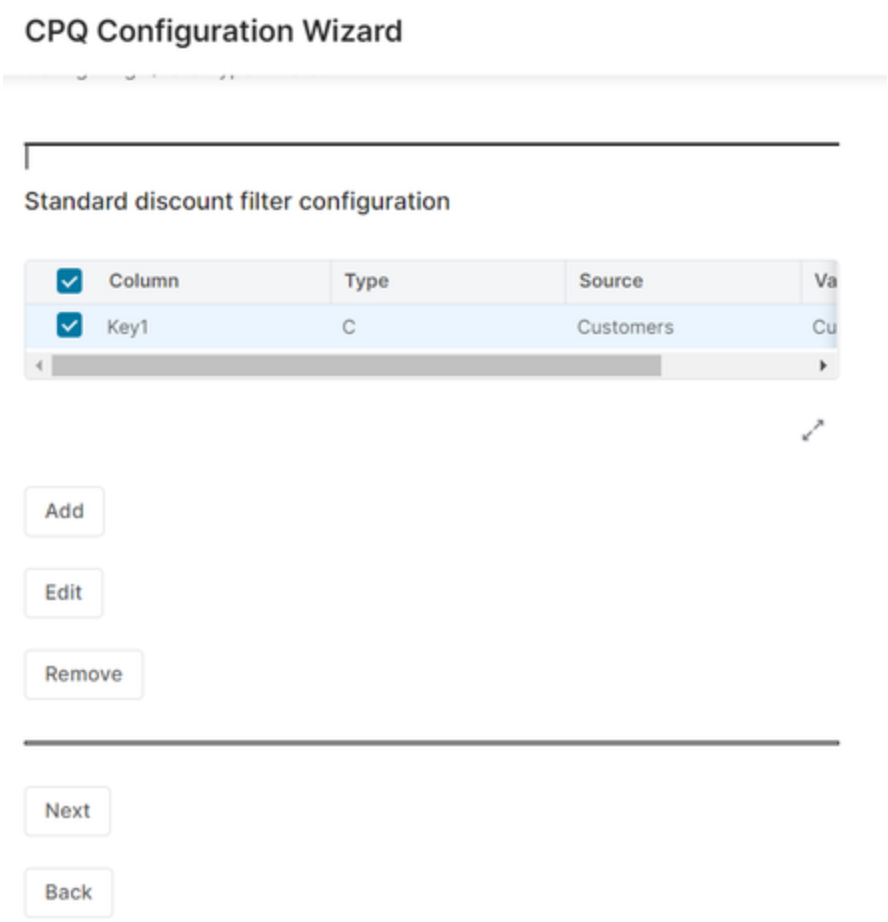
Standard Discount Filter Configuration

The screen allows you to configure how your discount data should be filtered.

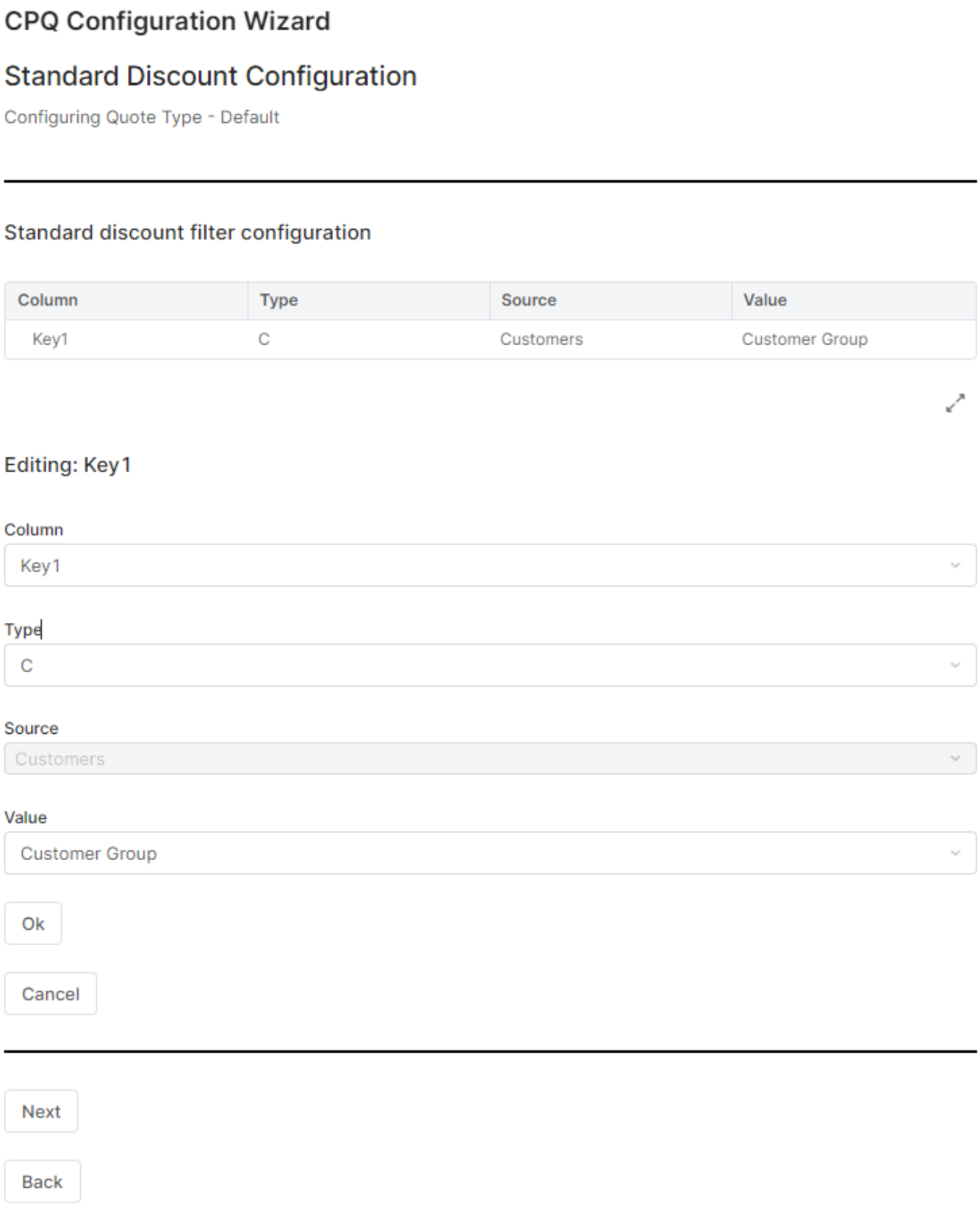
When loading for the first time, the screen shows the filters that you have already configured and saved in the system.



You can add, edit or remove the filter configuration by selecting the respective action button.



By clicking **Add** or **Edit**, a list of inputs will show for further configuration.



On the configuration screen for a specific key, describe the data that is used in the key column. This allows the CPQ logic to know where to get quoted product data to filter the standard discount value.

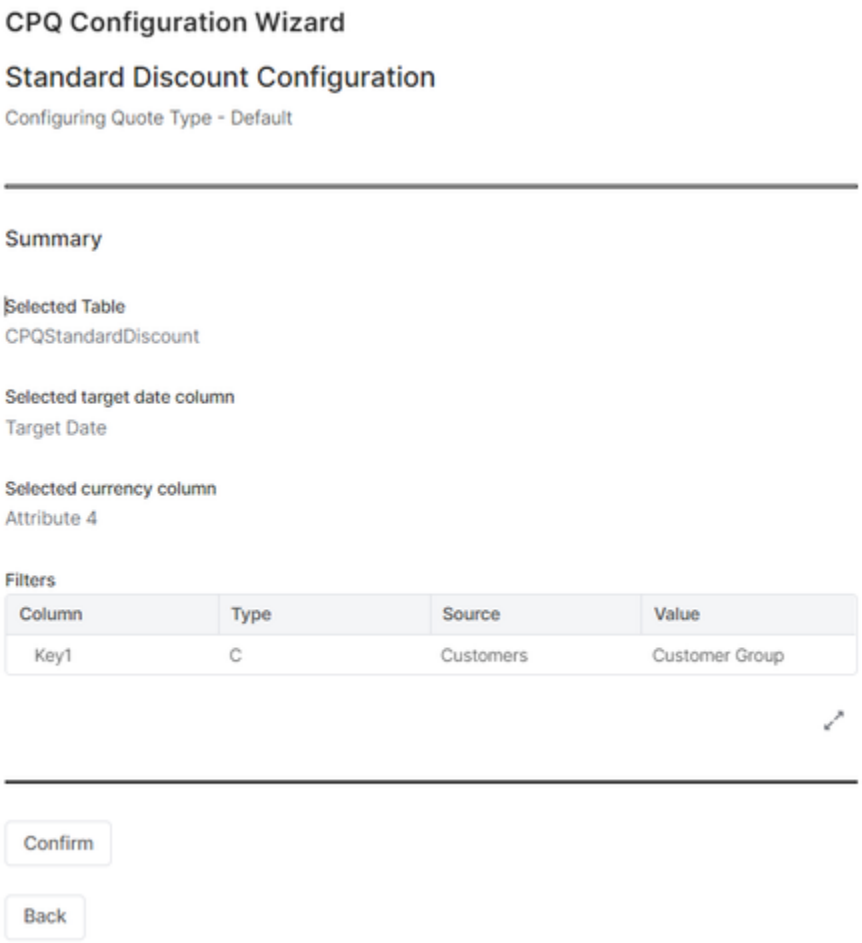
The available fields are:

- **Column** - Column in the product extension table. You can select from a list of columns.
- **Type** - The source where the value in the key column comes from. Available values are:
  - **C** - Customer Master
  - **P** - Product Master
  - **PX** - Product Extension
  - **CX** - Customer Extension
- **Source** - The table name where the filter value come from.
- **Value** - Column where the value in the key columns comes from. You can select from a list of column names.

When finished, click the **Next** button to see the summary.

Configuration Summary

The screen shows all of your configurations that have been done on the previous screens.



If you still want to modify your configuration, click the **Back** button.

When everything looks good, you need to click **Confirm** and then **Apply** to save all your configurations to the system.

Configure Rebate Integration Configuration

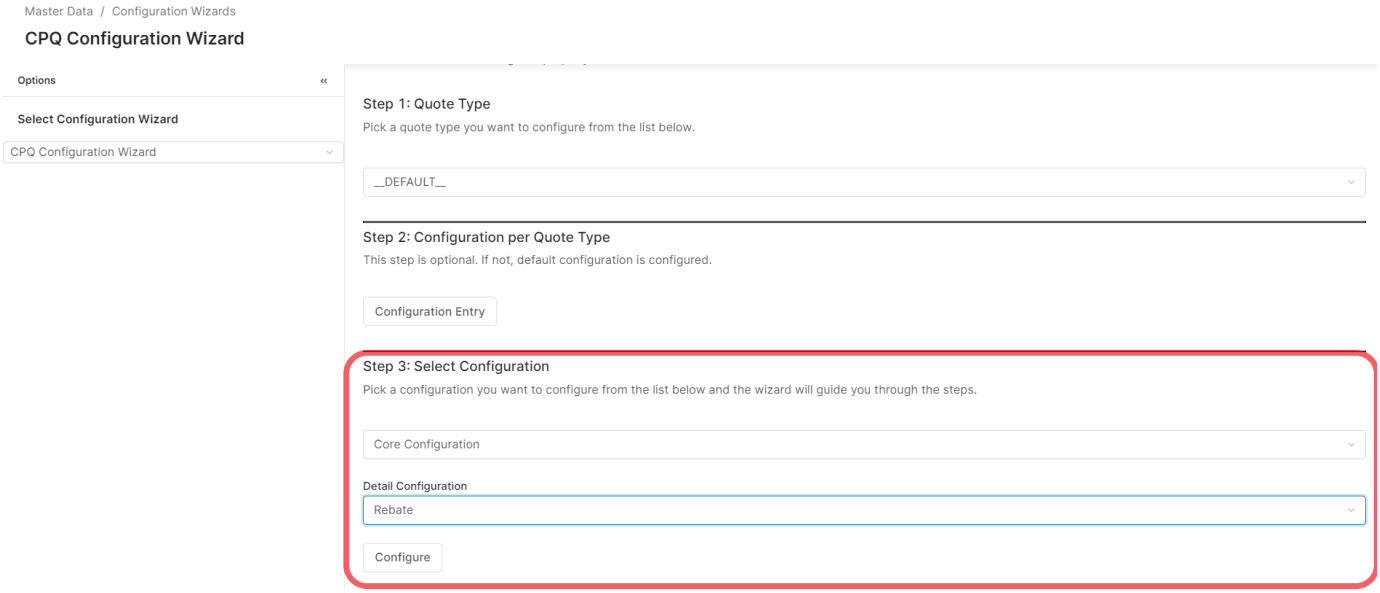
CPQ can access the rebate data generated by the Rebate Accelerator by predefined configurations. There is a Rebate Calculation Plan that can be configured using the CPQ Configuration Wizard.

Pre-requisites

- Custom logics are prepared in the form of a Pricefx Groovy library.
- Quote Type is selected in Step 1.
- Tables are configured in Step 2.

Select Configuration

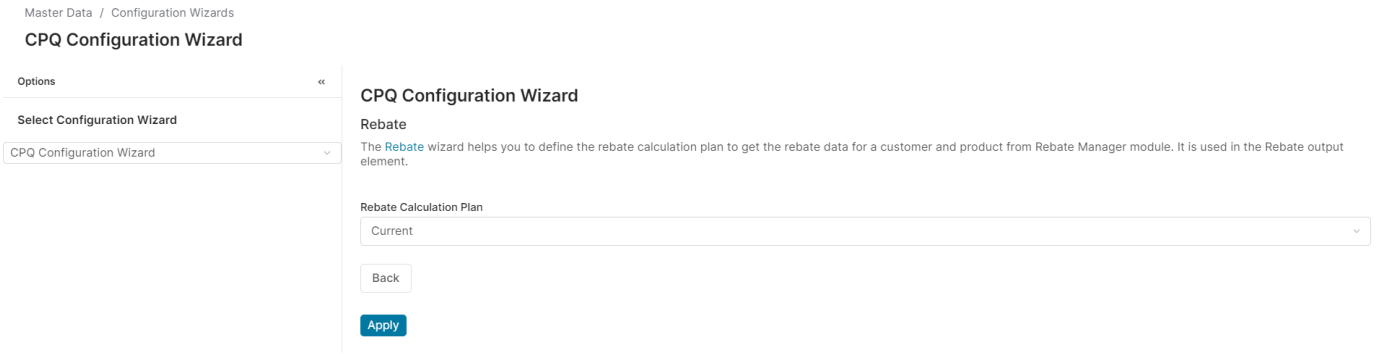
In Step 3, select **Core Configuration** in the first drop-down input and **Rebate** in the second input.



Then click the **Configure** button to enter the configuration screen.

Rebate Calculation Plan Calculation

On the configuration screen, select the desired **Rebate Calculation Plan**.



When finished, click the **Apply** button.

Configure Cost Configuration

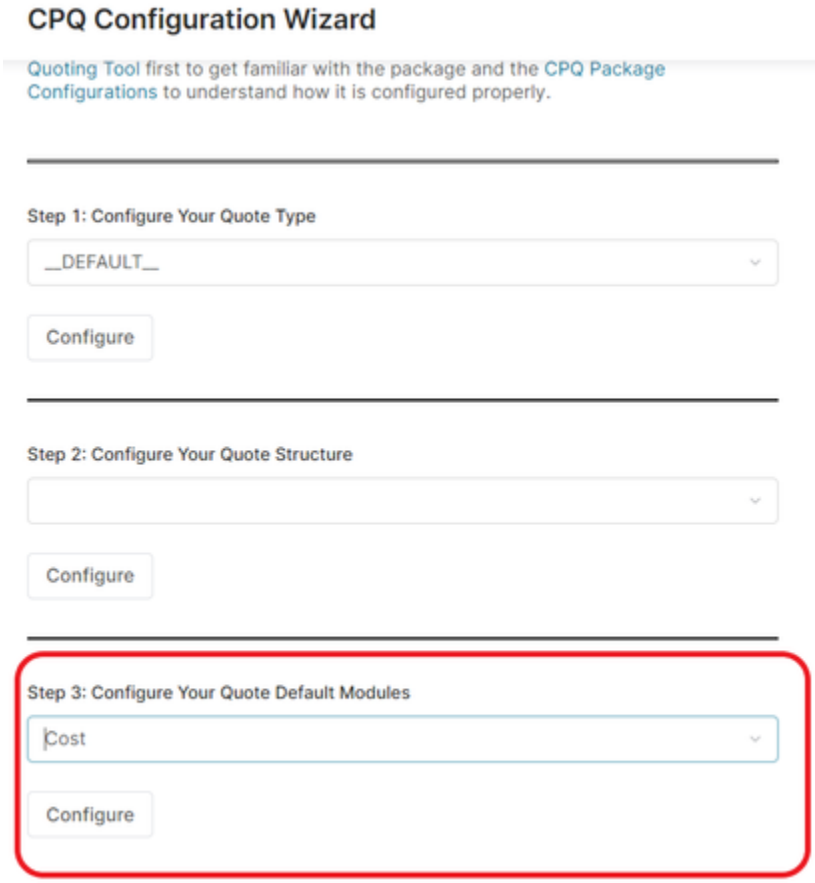
CPQ allows you to customize how the cost data is read from the Product Extension table. This can easily be done using the CPQ Configuration Wizard.

Pre-requisites

- Quote Type is selected and configured in Step 1.

Cost Module Configuration Selection

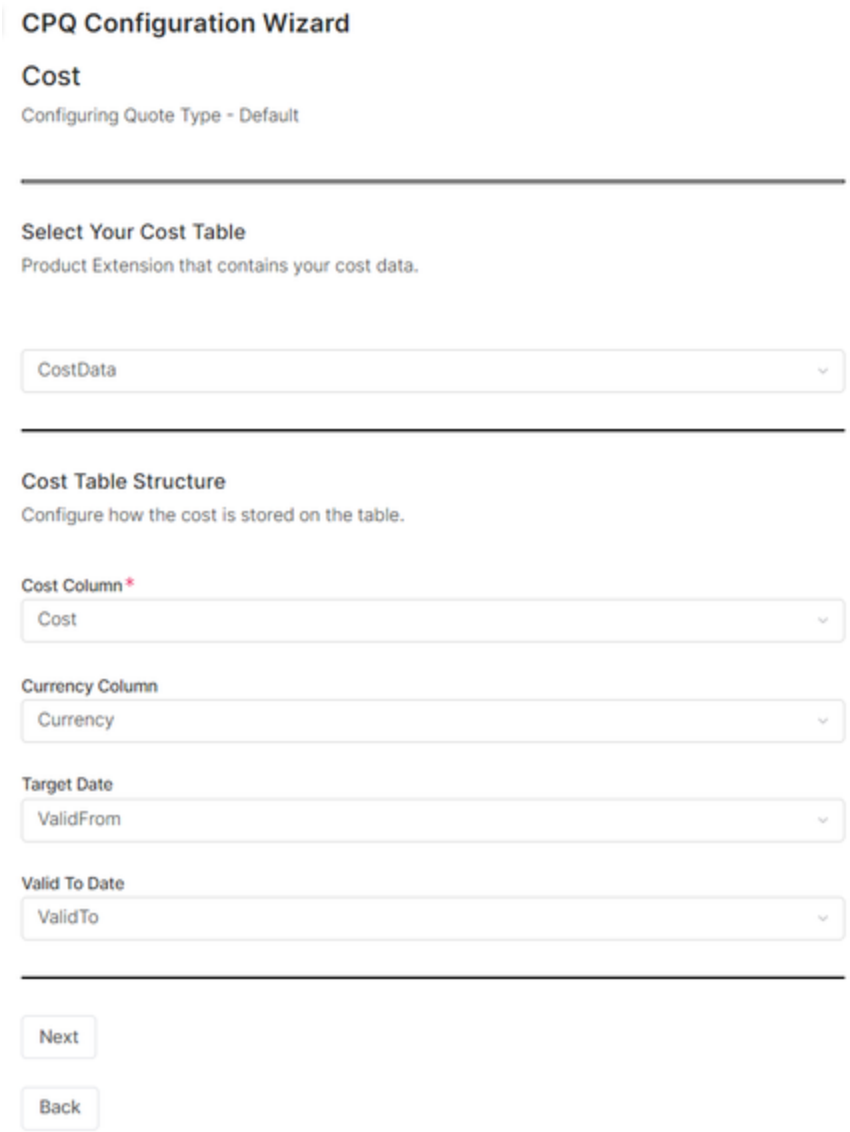
In Step 3, select **Cost** in the drop-down input.



Then click the **Configure** button to enter the Cost data configuration flow.

Cost Data Mapping Configurations

This screen allows you to configure where and how the cost data is stored in the Product Extension table.

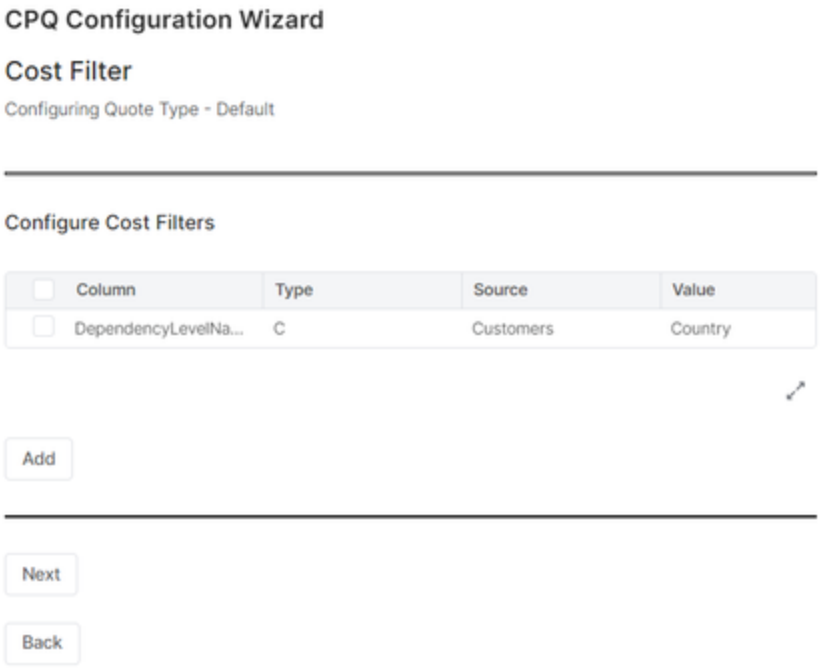


Select your Product Extension table that contains the cost data and fill in the field mappings by selecting the available fields in the drop-down menu.

Then, click the **Next** button to move forward to the Cost Filter configuration screen.

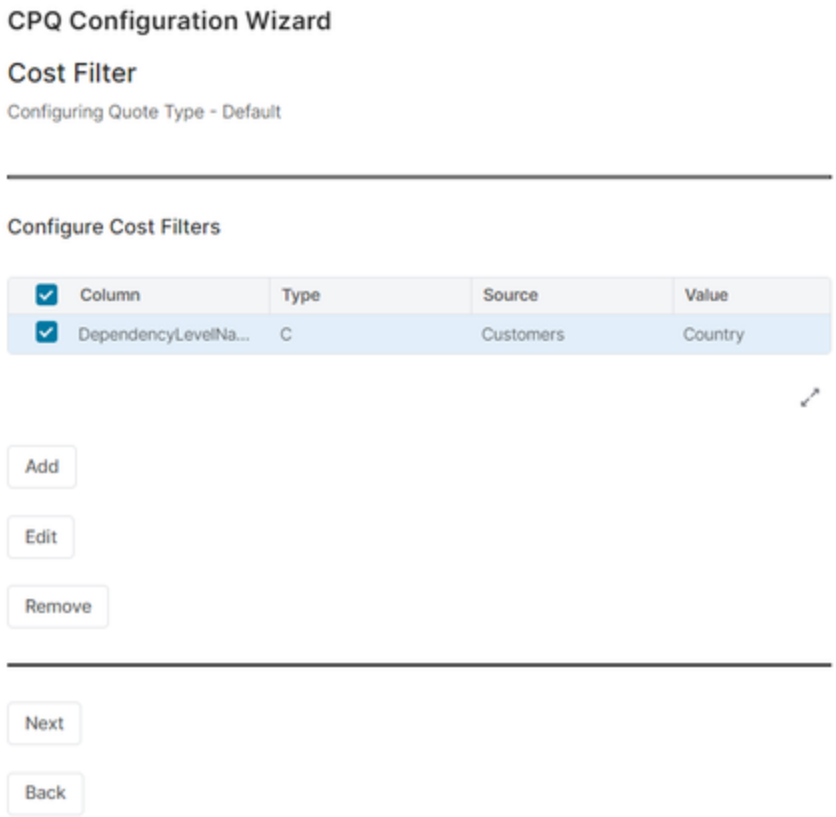
Cost Filtering

The next screen allows you to configure how your cost data should be filtered, such as the cost should be different for each customer country.

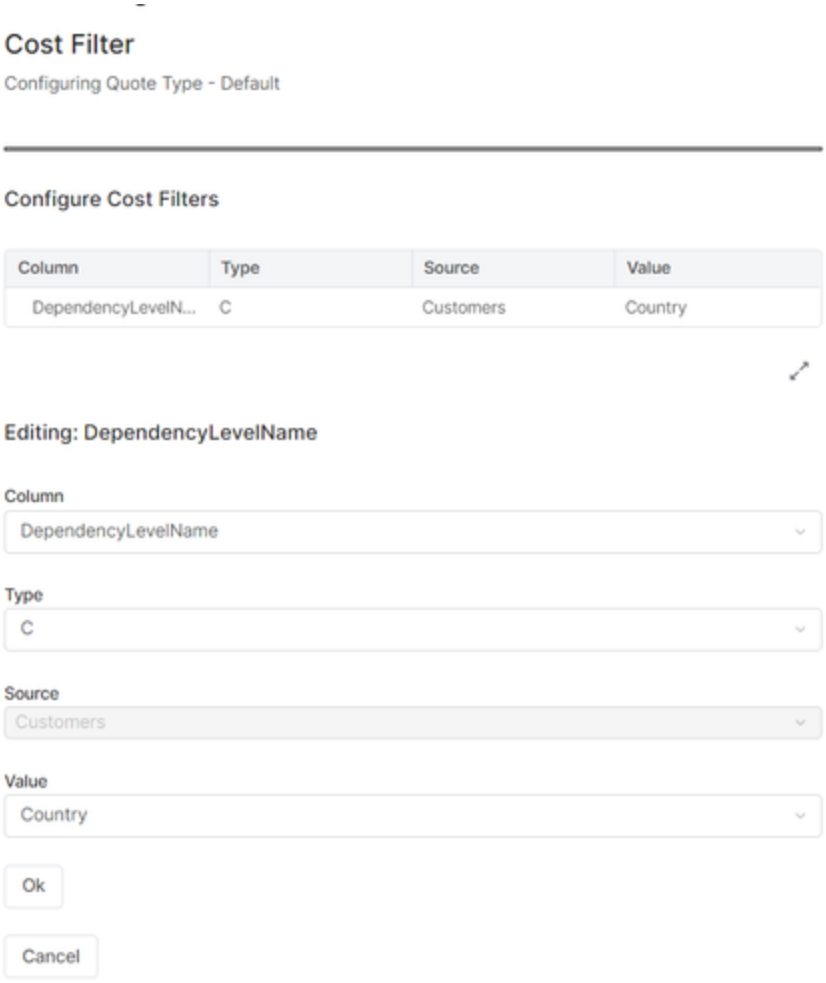


When loading for the first time, the screen shows the filters that you have already configured and saved in the system.

You can add, edit or remove the filter configuration by selecting the respective action button.



By clicking **Add** or **Edit**, a list of inputs will show for further configuration.



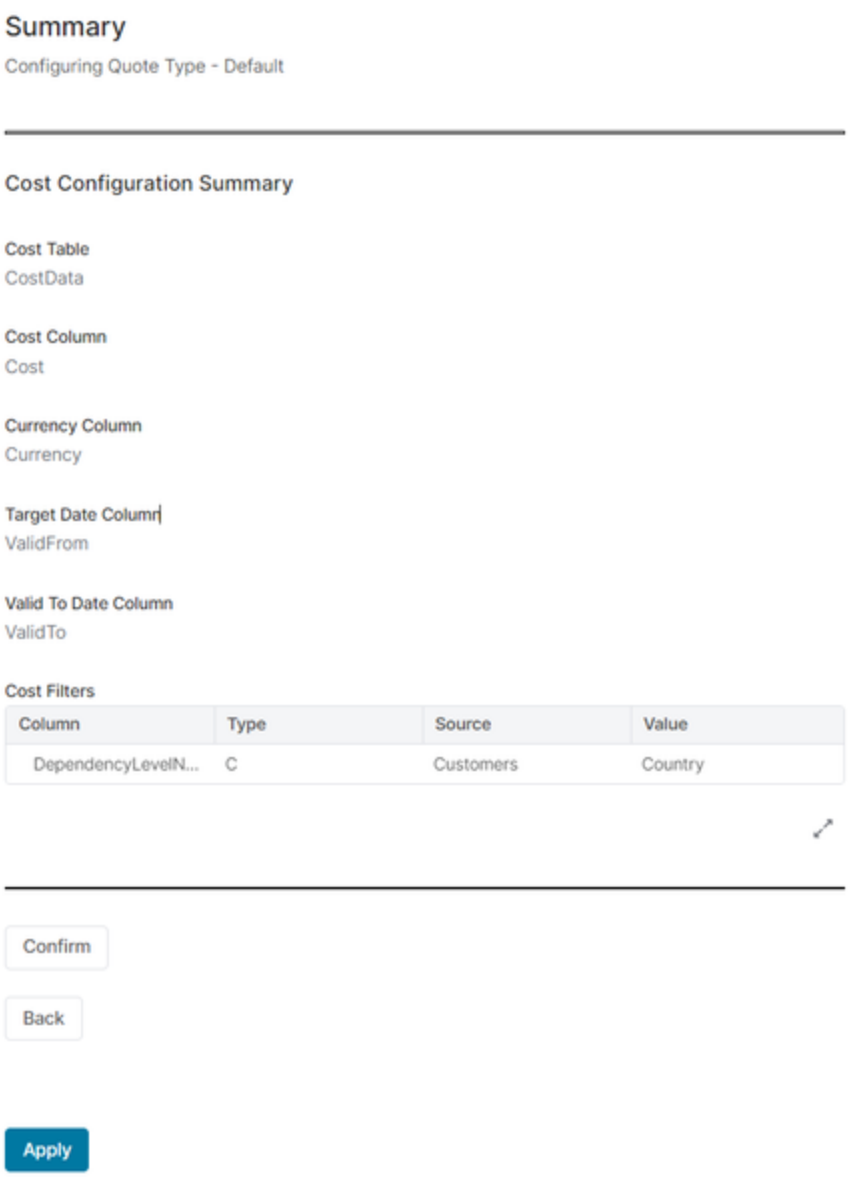
The available fields are:

- **Column** - Column in the product extension table. You can select from a list of columns.
- **Type** - The source where the value in the key column comes from. Available values are:
  - **C** - Customer Master
  - **P** - Product Master
  - **PX** - Product Extension
  - **CX** - Customer Extension
- **Value** - Column where the value in the key columns comes from. You can select from a list of column names.

When finished, click the **Next** button to see the summary.

Cost Configuration Summary

The screen shows all of your cost configurations that have been done on the previous screens.



To go back and modify your configuration, click the **Back** button.

When everything looks good, you need to click **Confirm** and then **Apply** to save all your configurations to the system.

Configure Price Competitors Configuration

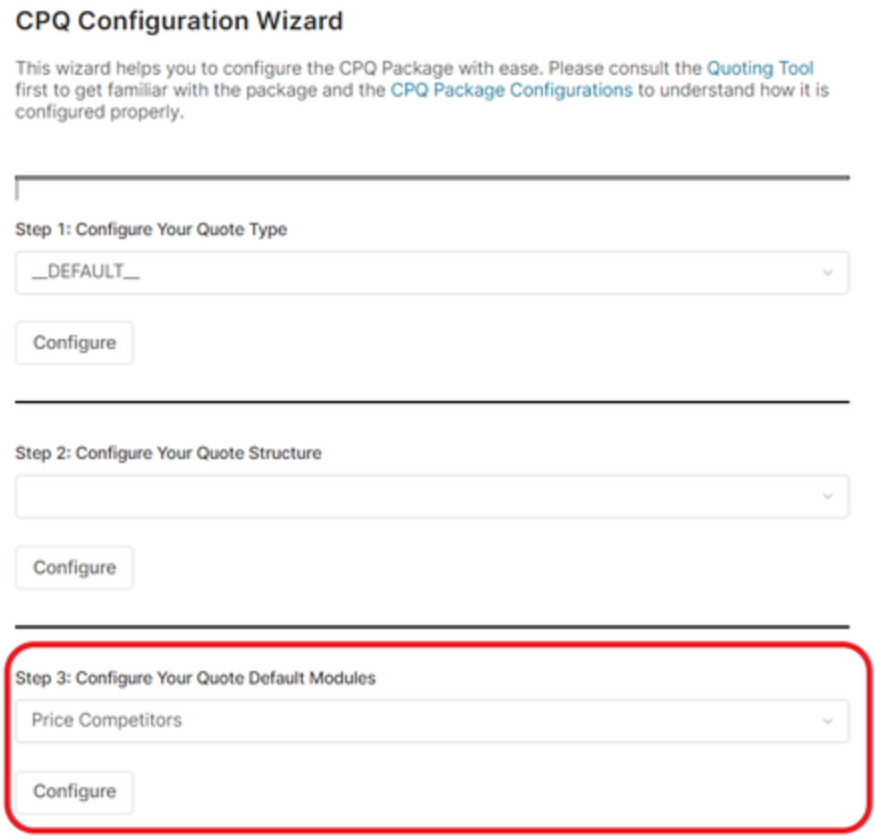
CPQ allows you to customize how the price competitor data is read from the Product Extension table. This can easily be done using the CPQ Configuration Wizard.

Pre-requisites

- Quote Type is selected and configured in Step 1.

Price Competitors Module Configuration Selection

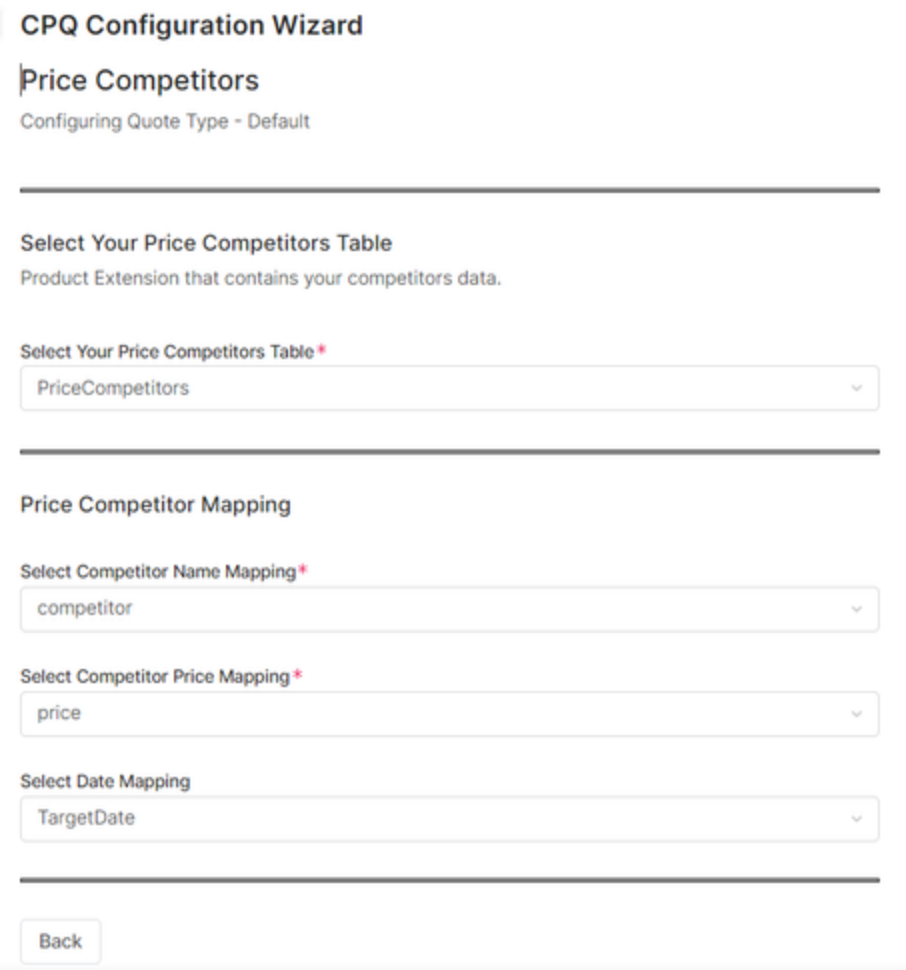
In Step 3, select **Price Competitors** in the drop-down input.



Then click the **Configure** button to enter the configuration mapping screen.

Price Competitors Mapping Configuration

This screen allows you to configure where and how the Price Competitors data is stored in the Product Extension table.



Select your Product Extension table that contains your competitor data and fill in the field mappings by selecting the available fields in the drop-down menu.

When finished, click the **Apply** button to save the configuration.

Configure Price Guidances Configuration

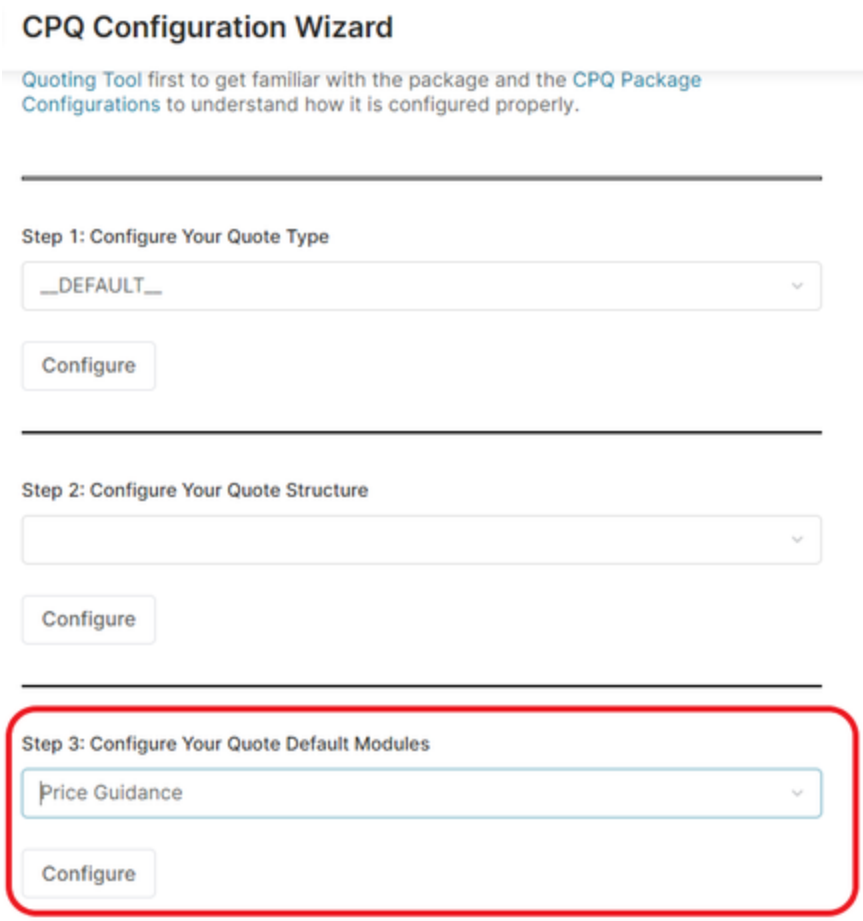
The CPQ Configuration Wizard makes it easier to integrate the PriceOptimizer data with the CPQ logic.

Pre-requisites

- Quote Type is selected and configured in Step 1.

Price Guidance Module Configuration Selection

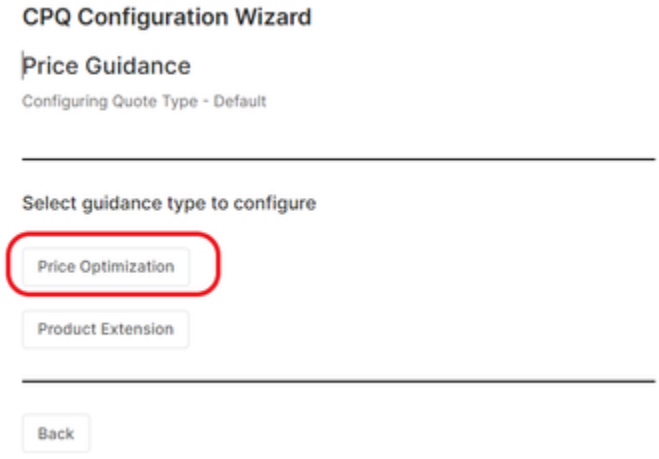
In Step 3, select **Price Guidance** in the drop-down input.



Then click the **Configure** button to enter the configuration selection screen.

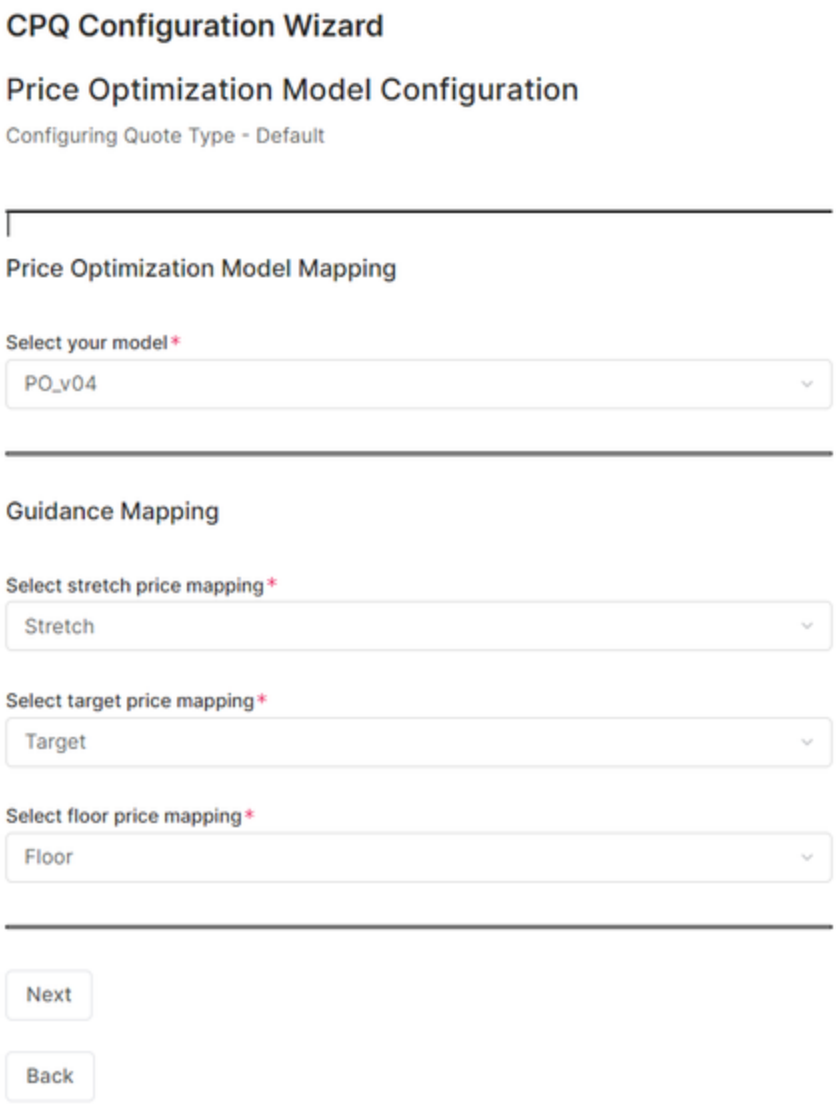
Price Optimization Configuration

To configure the Price Guidance using the Price Optimization, click on the **Price Optimization** button



Model Mapping Configuration

This screen allows you to configure where and how to read your Price Guidance from PriceOptimizer.



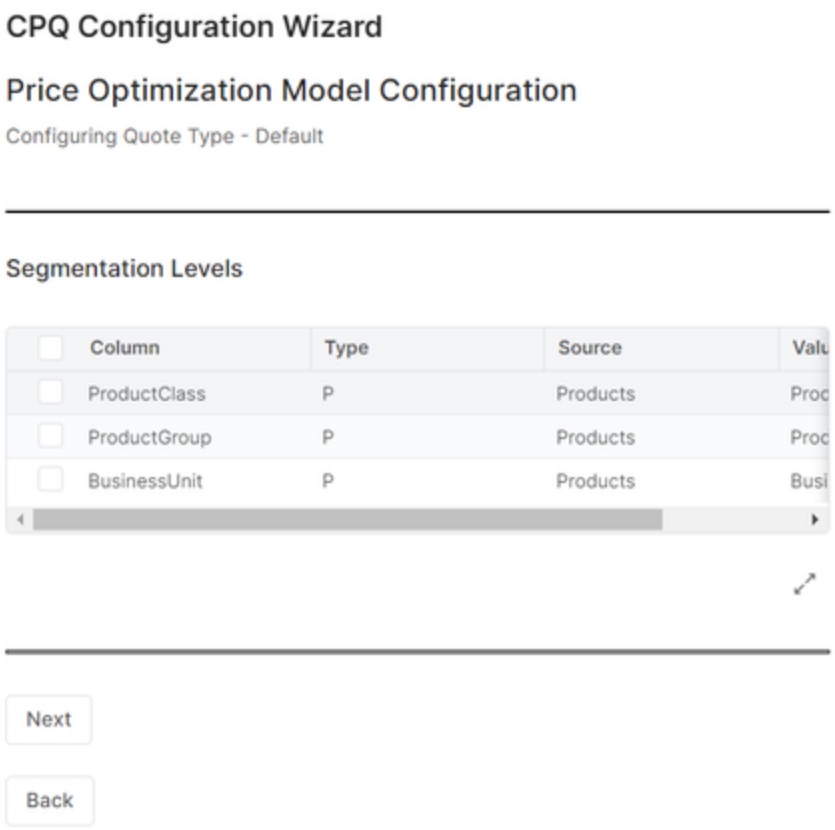
Select your model that contains the price guidance and select the mapping for stretch, target, and floor prices.

When finished, click **Next** to enter the Segmentations configuration screen.

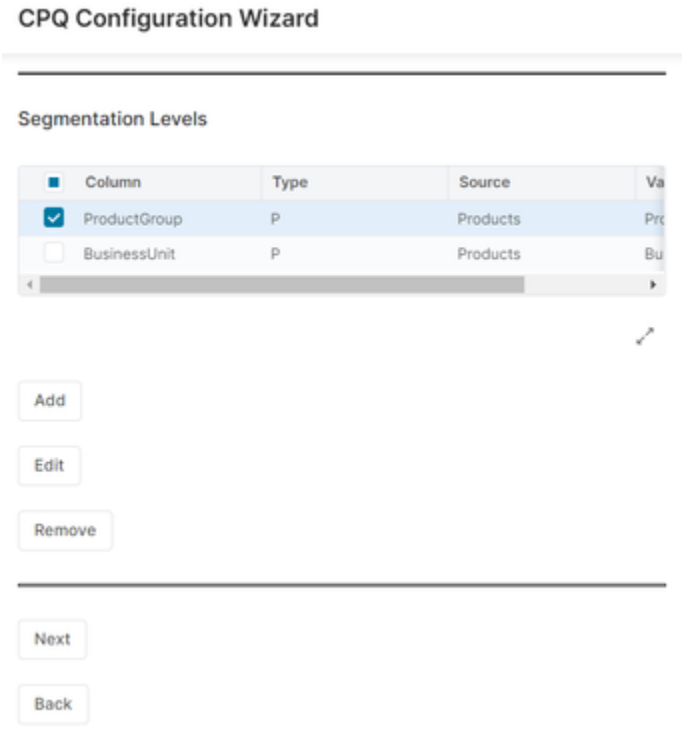
Segmentations Configuration

The next screen allows you to configure the segmentations of the PriceOptimizer Model.

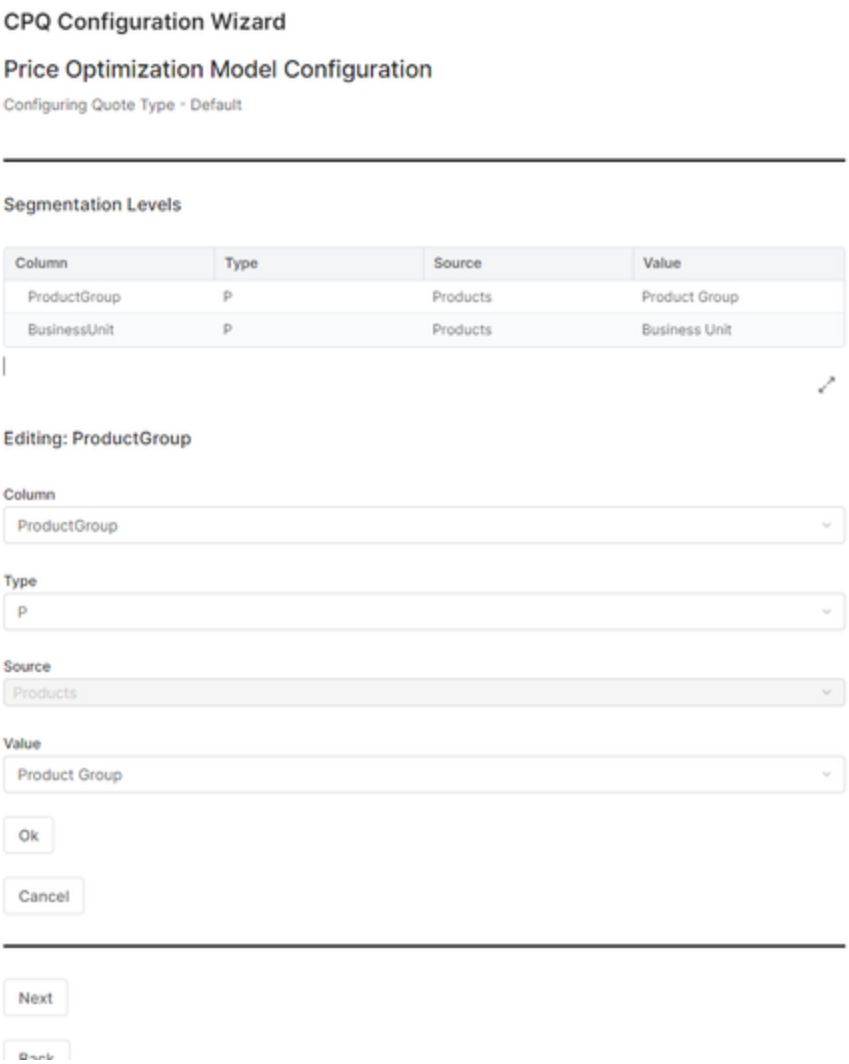
When loading for the first time, the screen shows the segmentations that you have already configured and saved in the system.



You can add, edit or remove the filter configuration by selecting the respective action button.



After clicking **Add** or **Edit**, a list of inputs will show for further configuration.



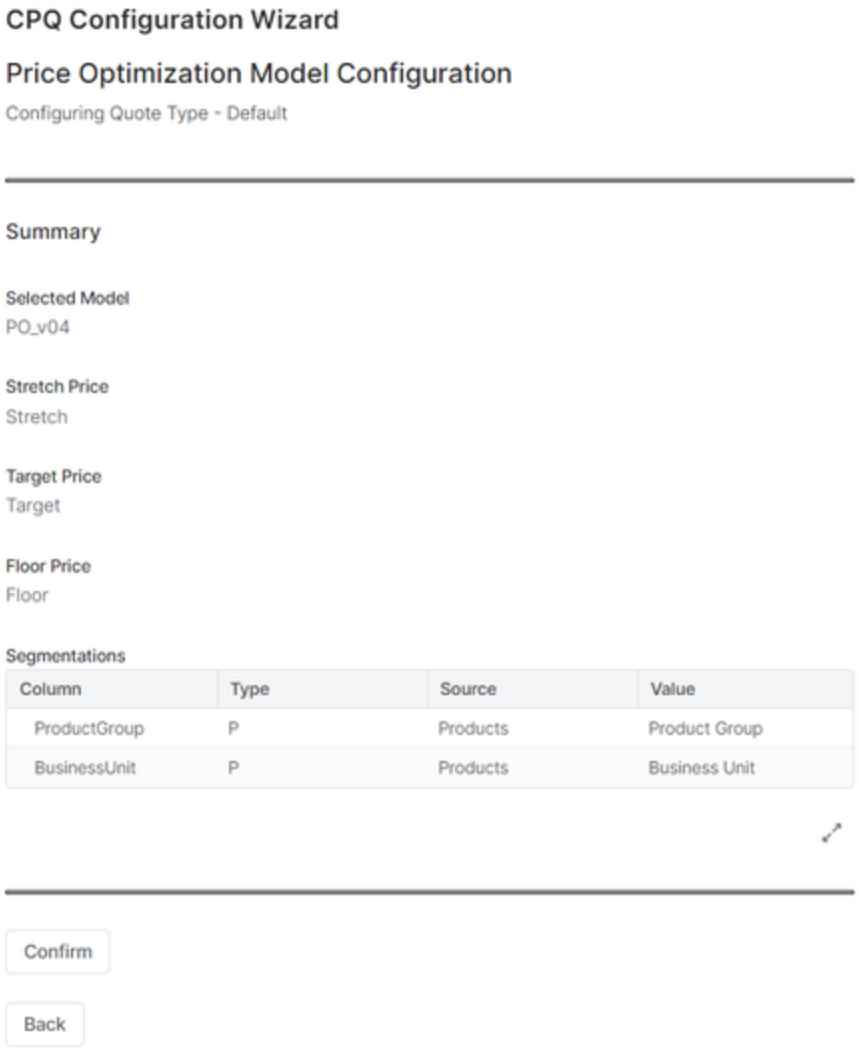
The available fields are:

- The available fields are:
  - **Column** - Column in the product extension table. You can select from a list of columns.
  - **Type** - The source where the value in the key column comes from. Available values are:
    - **C** - Customer Master
    - **P** - Product Master
    - **PX** - Product Extension
    - **CX** - Customer Extension
  - **Value** - Column where the value in the key columns comes from. You can select from a list of column names.

  When finished, click the **Next** button to see the summary.

Price Guidance Configuration Summary

The screen shows all of your price guidance configurations that have been done on the previous screens.



If you want still to modify your configuration, click the **Back** button.

When everything looks good, you need to click **Confirm** and then **Apply** to save all your configurations to the system.

Product Extension Configuration

To configure the Price Guidance using the Product Extension table, click on the **Product Extension** button

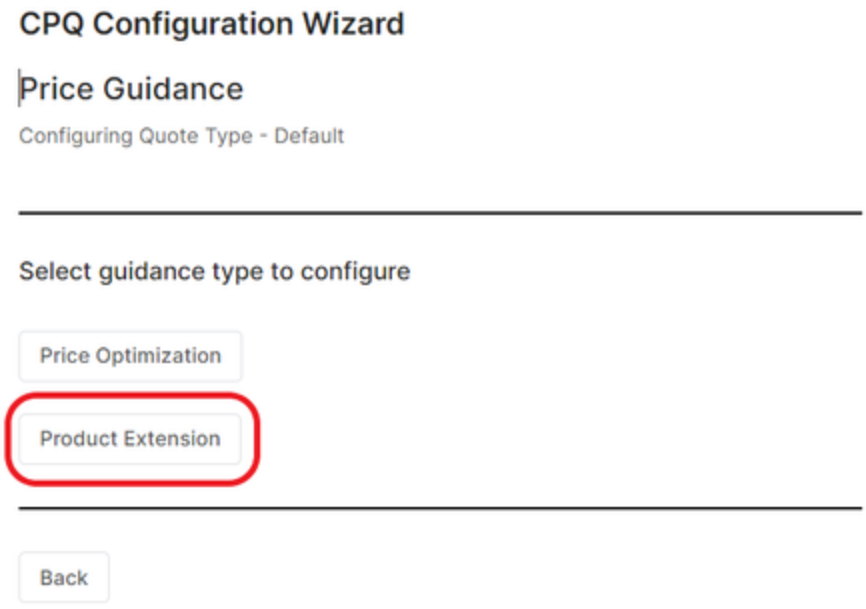


Table Mapping Configuration

This screen allows you to configure where and how the price guidance is stored on the Product Extension table.

CPQ Configuration Wizard

Price Guidance from Product Extension

Configuring Quote Type - Default

Product Extension Price Guidance Mapping

Select your table*

PriceGuidance

Guidance Mapping

Select stretch price mapping*

Stretch

Select target price mapping*

Target

Select floor price mapping*

Floor

Target date column

Target Date

Currency column

Attr. 5

Next

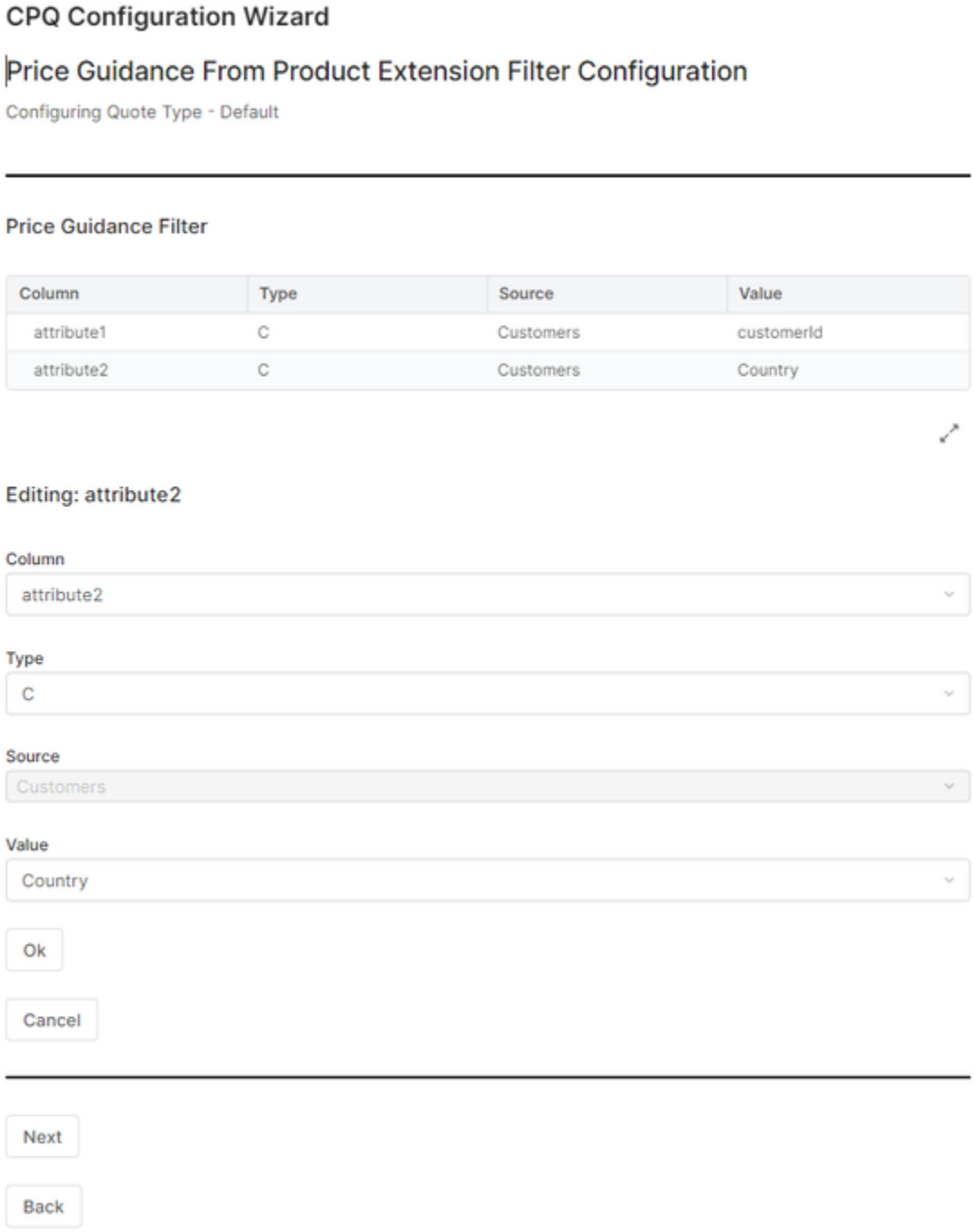
Back

Select your Product Extension table that contains the price guidance and select the mapping for stretch, target, and floor prices.

When finished, click **Next** to enter the Filter configuration screen.

Table Filter Configuration

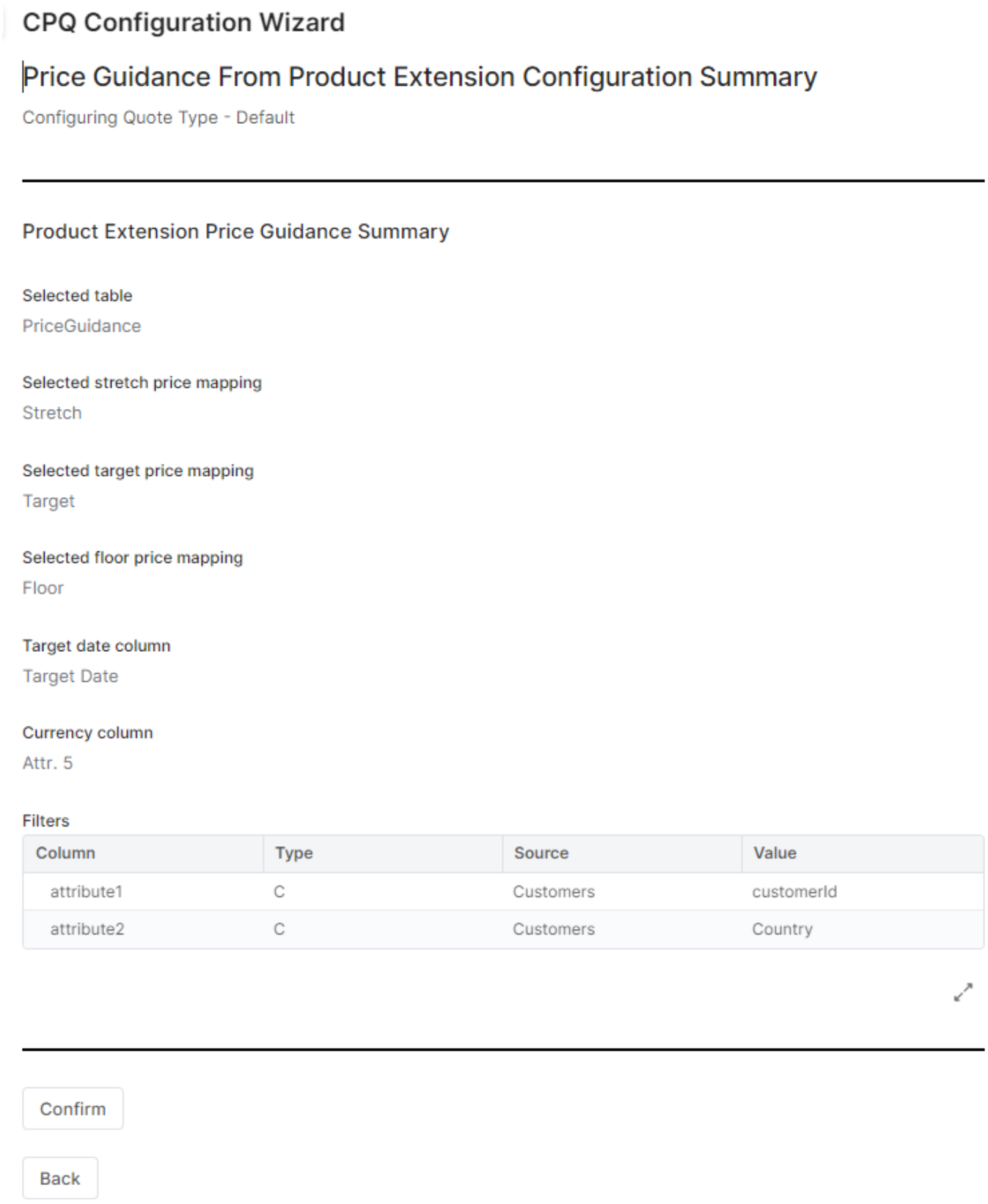
This screen allows you to configure how your price guidance should be filtered.



Click the **Next** button to see the summary.

Table Configuration summary

This screen show all configuration that you had done in previous steps.



If you want still to modify your configuration, click the **Back** button.

When everything looks good, you need to click **Confirm** and then **Apply** to save all your configurations to the system.

Troubleshooting

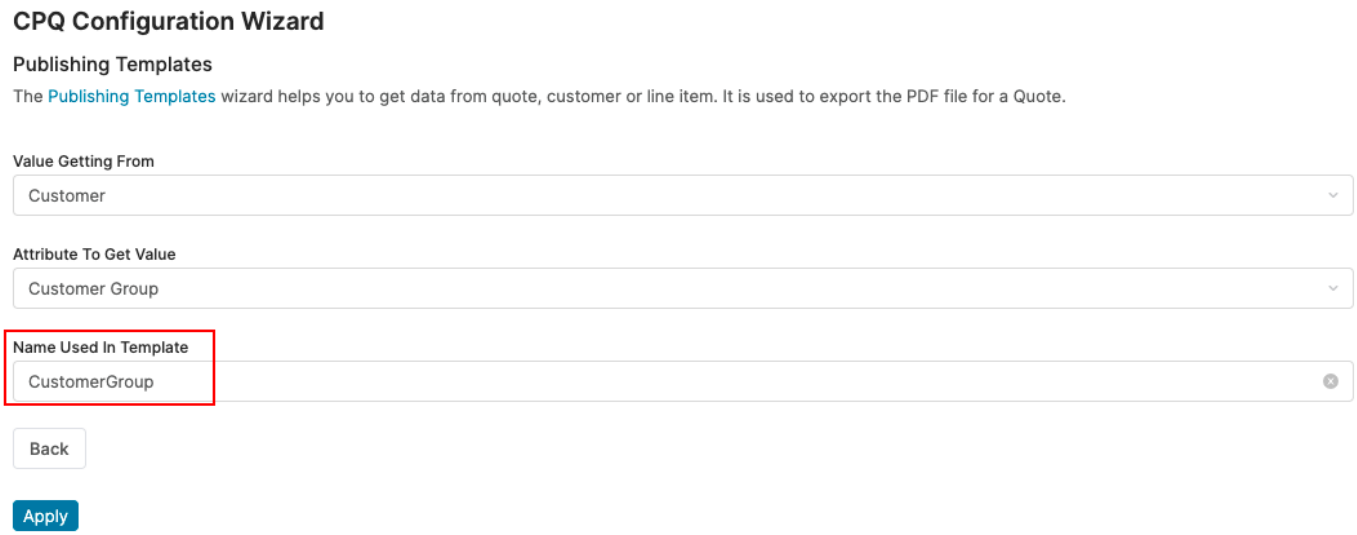
In this section:

- Configuration value is not updated into tables
- Quote Type is not displayed for selection

Configuration value is not updated into tables

Issue: When selecting a value and clicking the **Apply** button, the value is not updated in the table.

Solution: You might need to press the Tab key or click outside the input before clicking the **Apply** button.



Quote Type is not displayed for selection

Issue: Your expected Quote Type is not displayed in the wizard for selection.

Solution: Make sure the Quote Type uses the *CPQ_Header* logic.

CPQ Package Version 2.0

- Design
- Configuration Per Quote Type
- Custom Logics in CPQ
- Error Handler
- CPQ Package Version 1.x - OBSOLETE

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.

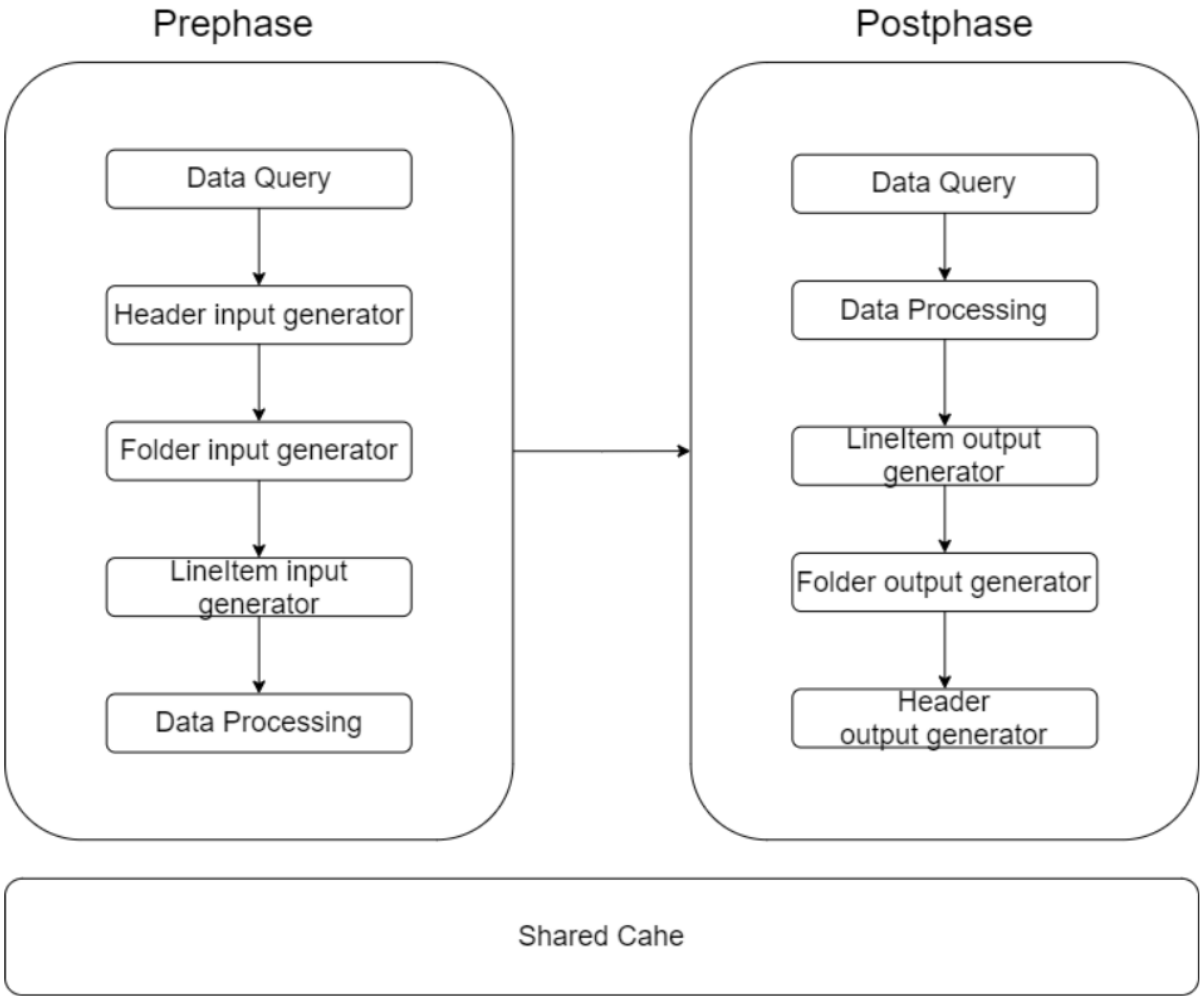
In this section:

- 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.



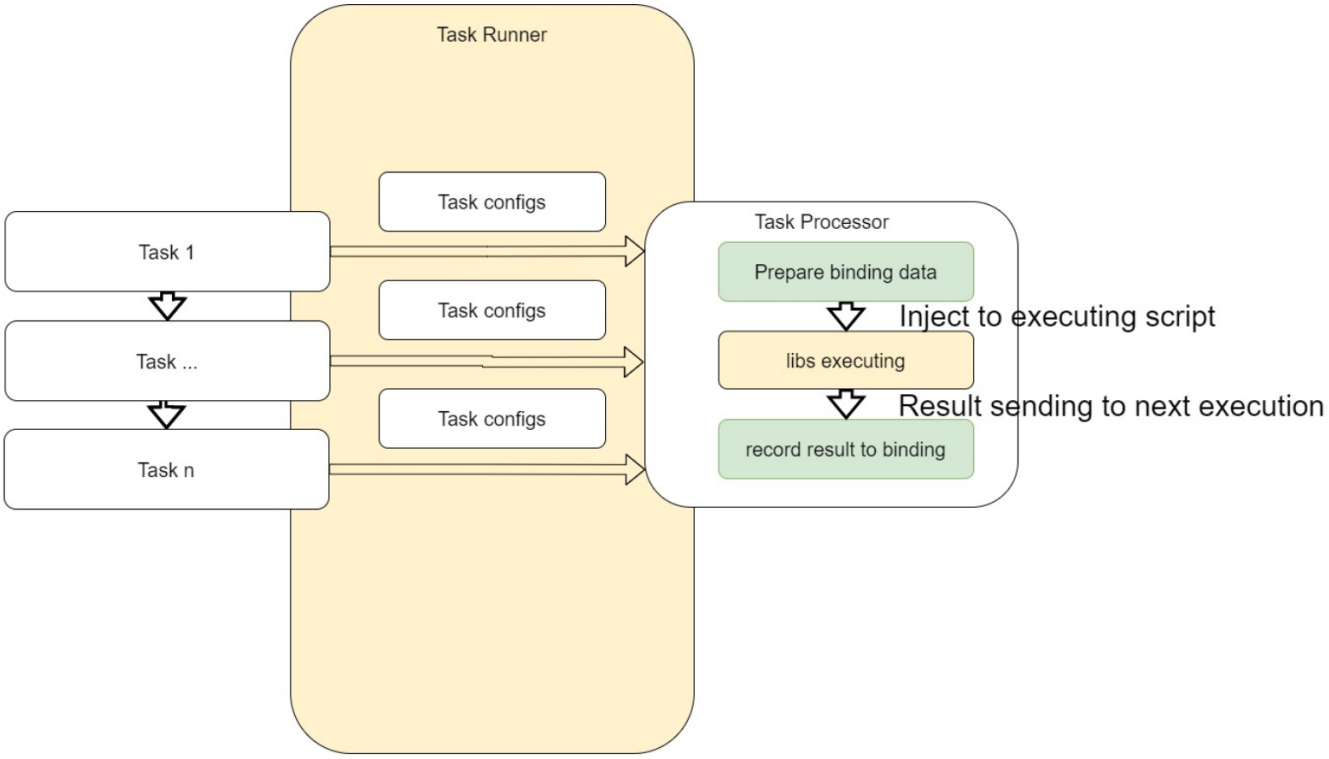
Per Quote Type Configuration

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.

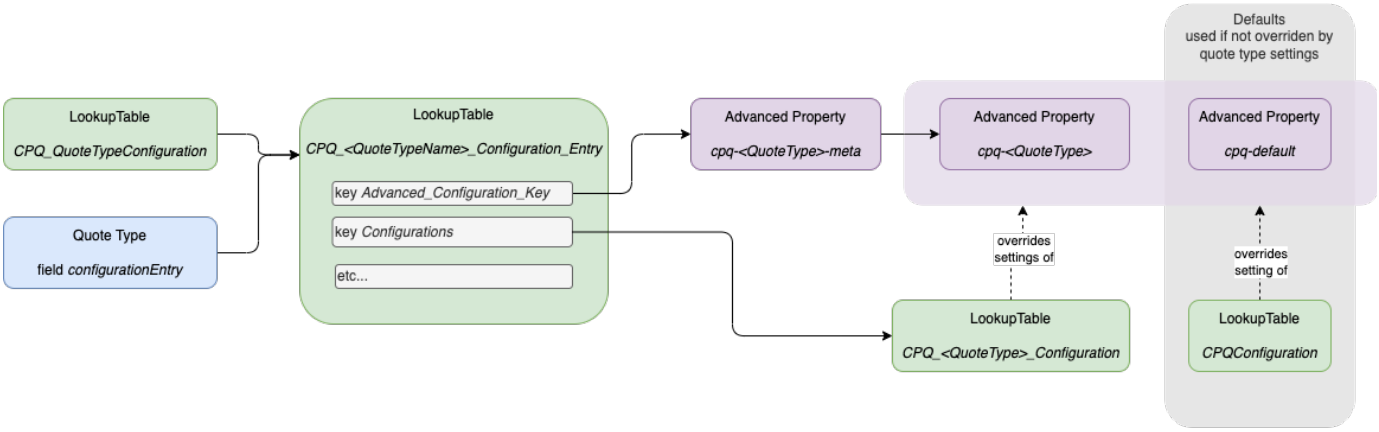
Task Processor is a library that is responsible for running the library defined in the task, recording the result, and sharing it with other tasks.

Data Sharing

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

Configuration Per Quote Type

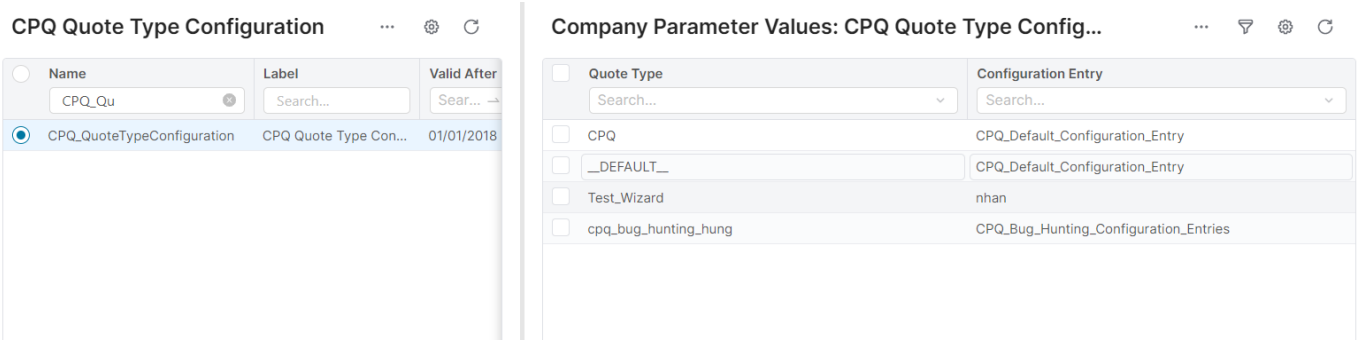
Configuration Schema



Price Parameter Configuration

There is a Price Parameter table *CPQ_QuoteTypeConfiguration* dedicated to the quote type configuration setting.

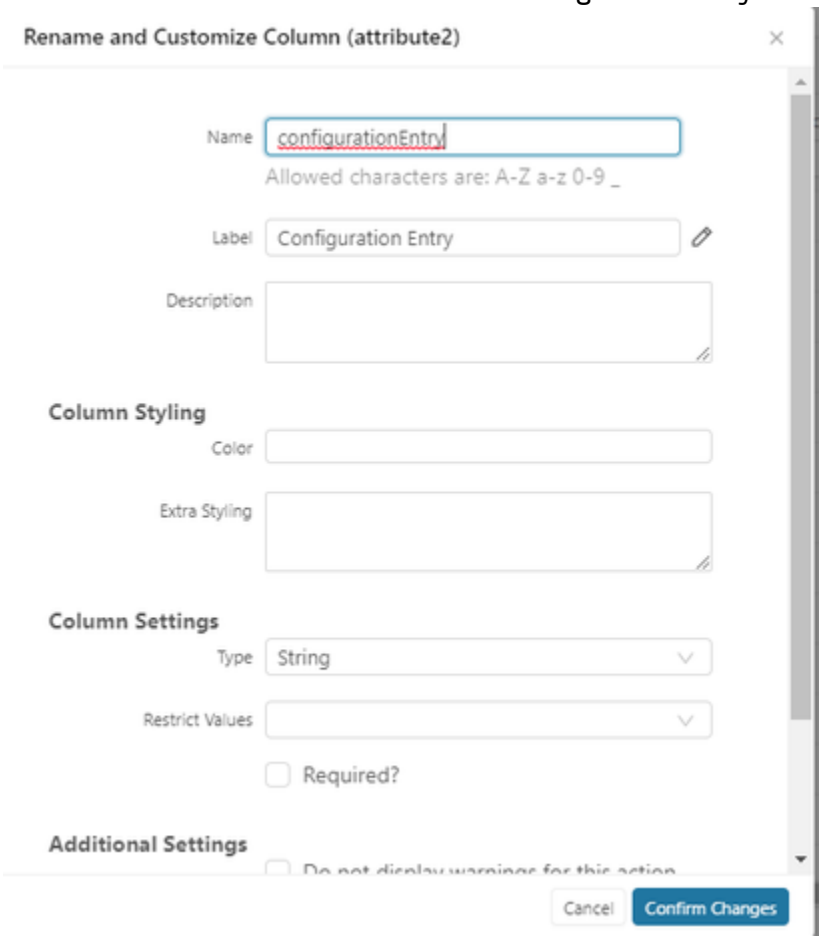
> The configurations in the price parameter take priority when both PP and Quote Types configurations in the Quoting module are present.



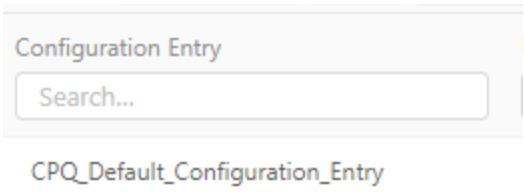
Quote Type Setting

To create a per quote type configuration:

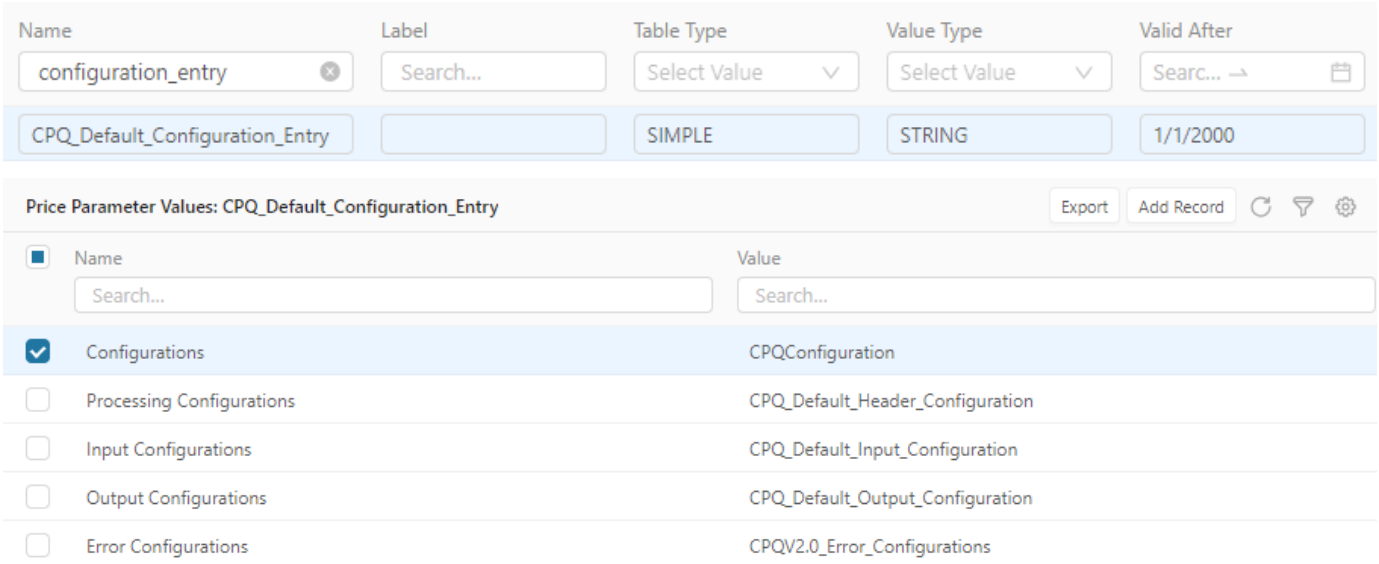
1. Open Quote Types.
2. Rename a column and set the name to “configurationEntry”.



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



Configuration Entry PP Structure



Configuration entry setting:

- **Advanced_Configuration_Key** – Advanced property (AP) key which defines AP keys from which to create the configuration.
- **Configurations** – PP table name which contains CPQ configurations.
- **Processing_Configurations** – Defines the PP table which contains configurations for data query and data processing.
- **Input_Output_Configurations** – Defines the PP table which contains configurations for inputs and outputs generator.
- **Error_Configurations** – PP table which contains configurations for errors.
- **Historical_Datamart_Name** – Name of a Datamart which contains historical data.
- **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.
- **Exception_Discount_PP_Name** – PP table which contains exception discount values.
- **Standard_Discount_PP_Name** – PP table which contains standard discount values.
- **Customer_Cash_Discount_CX_Name** – CX table which contains customer cash discount values.
- **Product_Cost_PX_Name** – PX table which contains product costs.
- **Competitor_PX_Name** – PX table which contains competitor prices.
- **Price_Guidance_PX_Name** – PX table which contains price by segmentation values.
- **Input_Configuration_PP_Name** – PP table which contains custom input configurations in version 1.0 (deprecated from version 2.0).

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

> If the configuration is not provided for the specific quote type, the default one is used. See CPQ Package Architecture.

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*

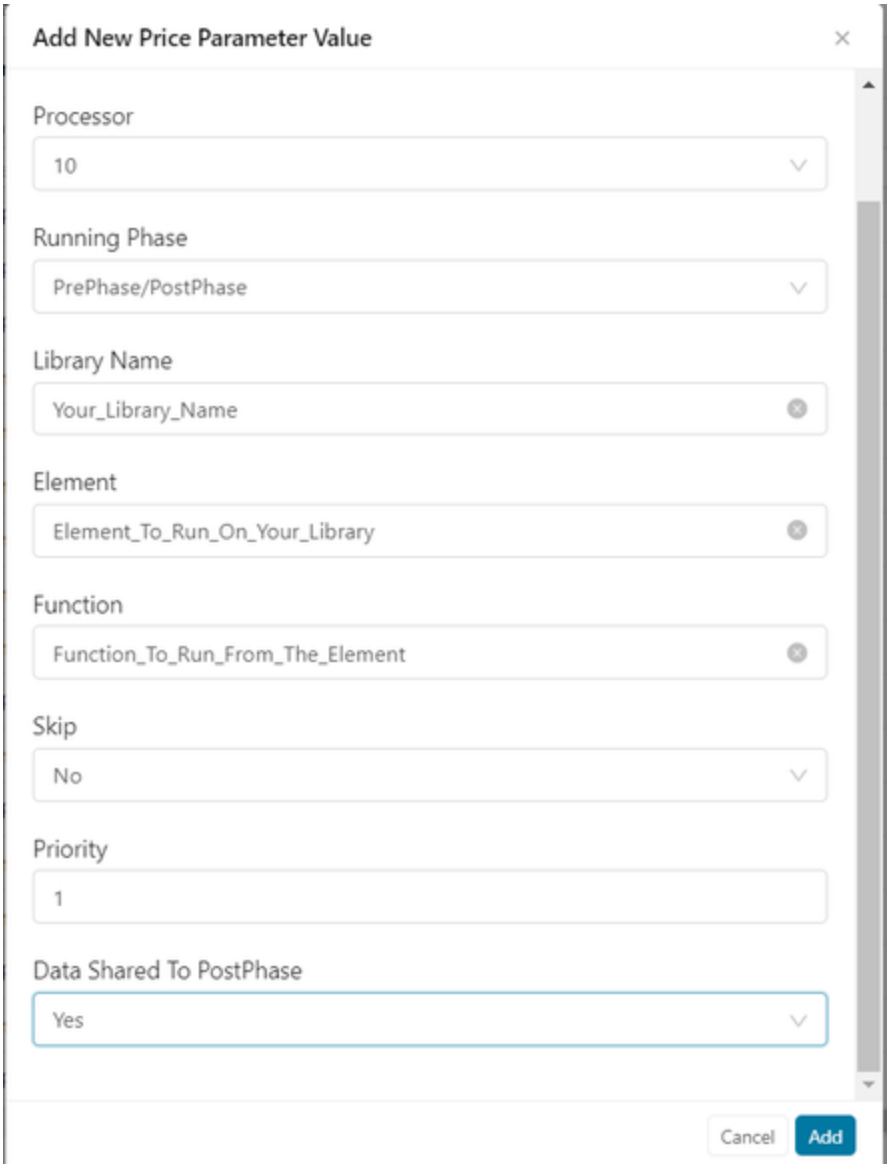
In this section:

- 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.

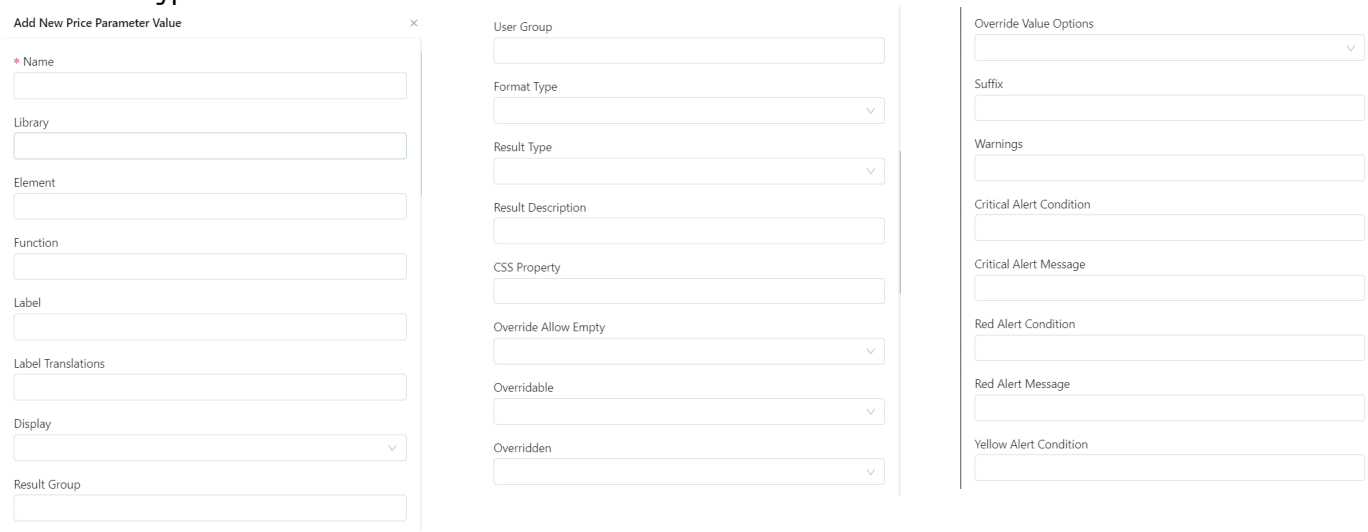
Default Price Parameter table: *CPQ_Default_Input_Output_Configuration*

In this section:

Configurations

You can set the following options:

- Generation Type - Available options are:
  - Header Input
  - Header Output
  - Custom Header
  - 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 logic.

- **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**

> From version 2.0.3, this is only available when `predefinedBindingAsScriptProperty` configuration is enabled.

```
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
}
```

The predefined variables are automatically injected into the custom library. This can slow down the process. In version 2.0.3, a new configuration named `predefinedBindingAsScriptProperty` is added and enabled as default. So you can decide whether to inject those variables automatically or not.

If it is disabled, you cannot use predefined bindings directly; a quick fix is to wrap the method implementation under `libs.CPQ_SharedLib.TaskProcessor.runWithPredefinedBindings`.

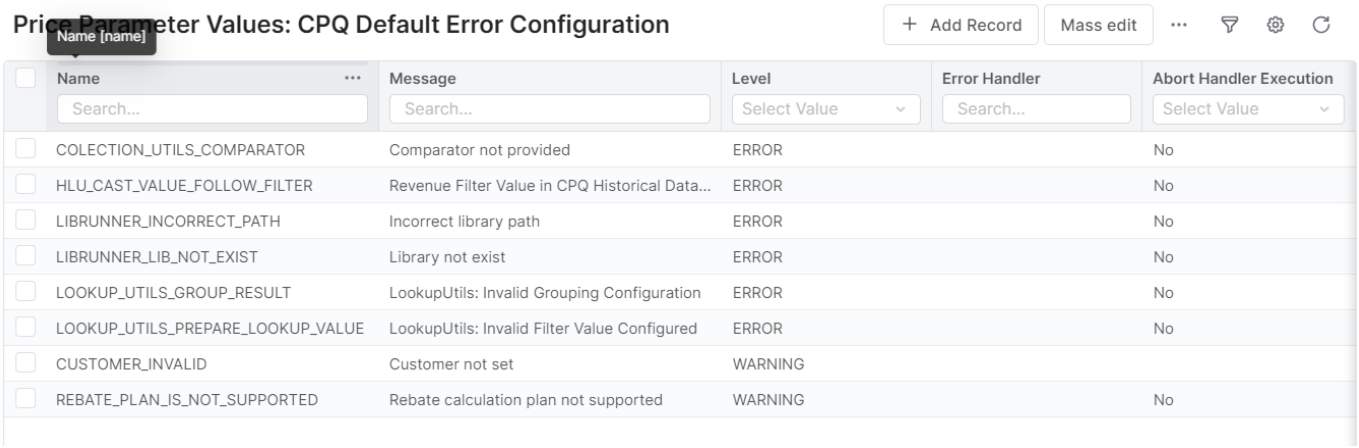
```
// When 'predefinedBindingAsScriptProperty' is enabled
BigDecimal execute() {
        boolean isPriceInput = libs.CPQ_Output_Utils.Common.
isPriceInput()
        // 'header' is predefined binding. You can access it directly
in custom library as it is injected automatically
        BigDecimal priceFromLookup = header.ProductPrices?.getAt
(lineItem.sku)?.resultPrice

        return isPriceInput ? BigDecimal.ZERO : priceFromLookup
}


// When 'predefinedBindingAsScriptProperty' is disabled
 BigDecimal execute() {
    // 'header' is predefined binding. You cannot access it directly in
custom library as it is not injected automatically.
    // A quick fix is to wrap the implementation under  "libs.
CPQ_SharedLib.TaskProcessor.runWithPredefinedBindings"
    return libs.CPQ_SharedLib.TaskProcessor.runWithPredefinedBindings
{
        boolean isPriceInput = libs.CPQ_Output_Utils.Common.
isPriceInput()
        BigDecimal priceFromLookup = header.ProductPrices?.getAt
(lineItem.sku)?.resultPrice

        return isPriceInput ? BigDecimal.ZERO : priceFromLookup
   }
 }
```

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 Package Version 1.x - OBSOLETE

⚠ The following sections are obsolete. In each section you will learn what replaces the configuration in CPQ Package Version 2.0.

- CPQ Inputs Configuration in Wizard - OBSOLETE
- CPQ Inputs Configuration 1.x - OBSOLETE
- CPQ Outputs Configuration - OBSOLETE
- Configurations Per Quote Type - OBSOLETE

CPQ Inputs Configuration in Wizard - OBSOLETE

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

- CPQ Inputs Wizard
  - Details on “Create”
  - Details on “Edit” and “Delete”
- CPQ Outputs Wizard

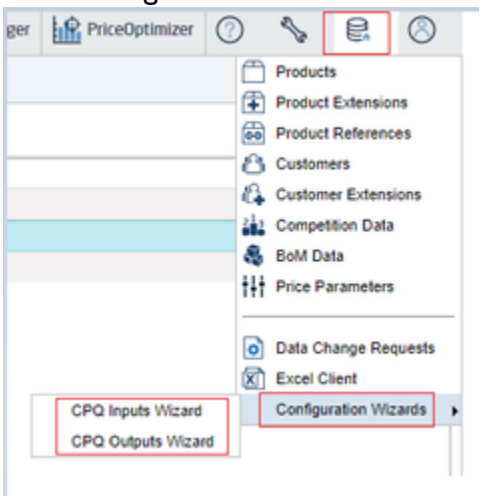
Price Parameters

- 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.

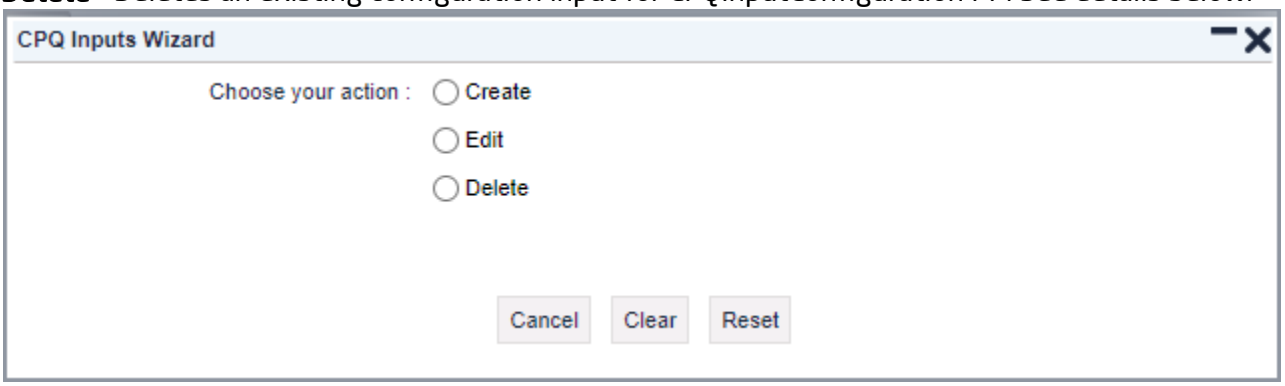


How It Works

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.

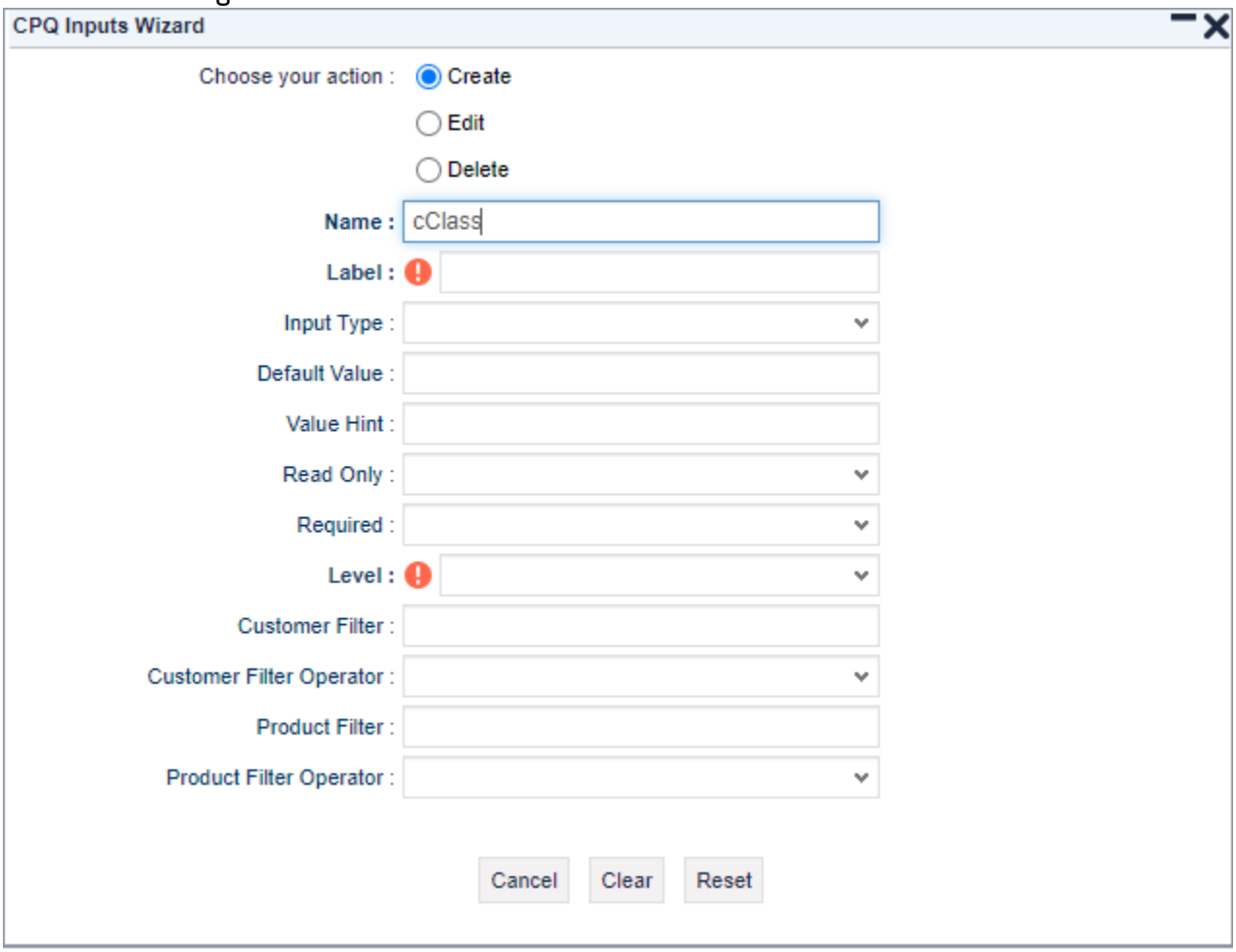


Details on "Create"

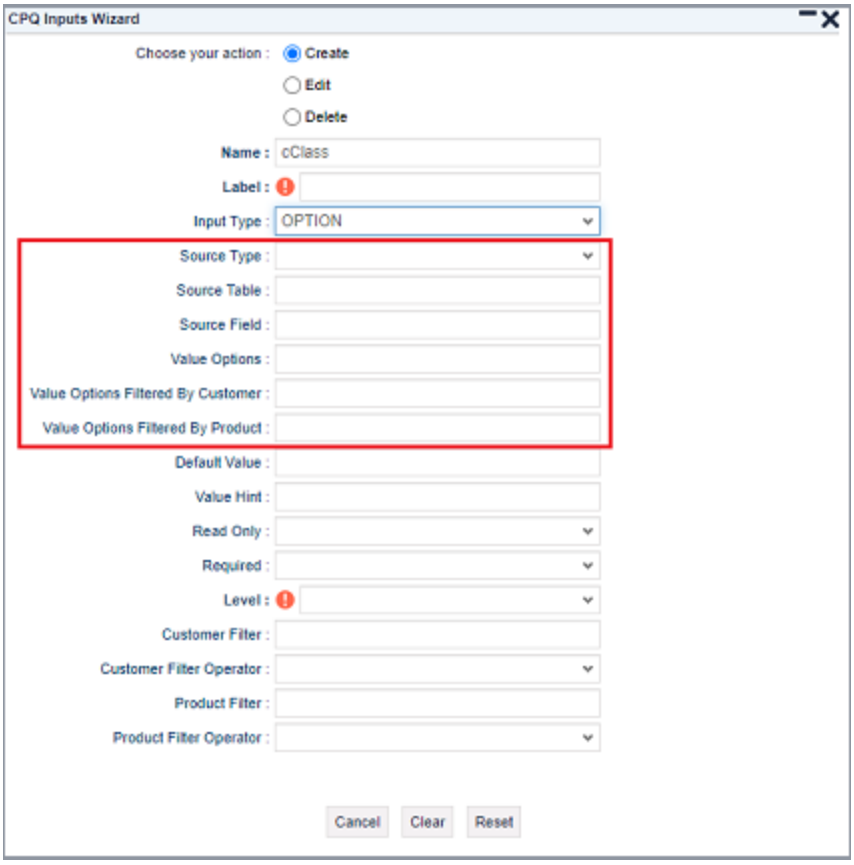
Enter a **Name** (preferably with no spaces, e.g. cClass, pGroup). Then press the **tab** key to apply the setting. If the name already exists, you will be notified.

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:



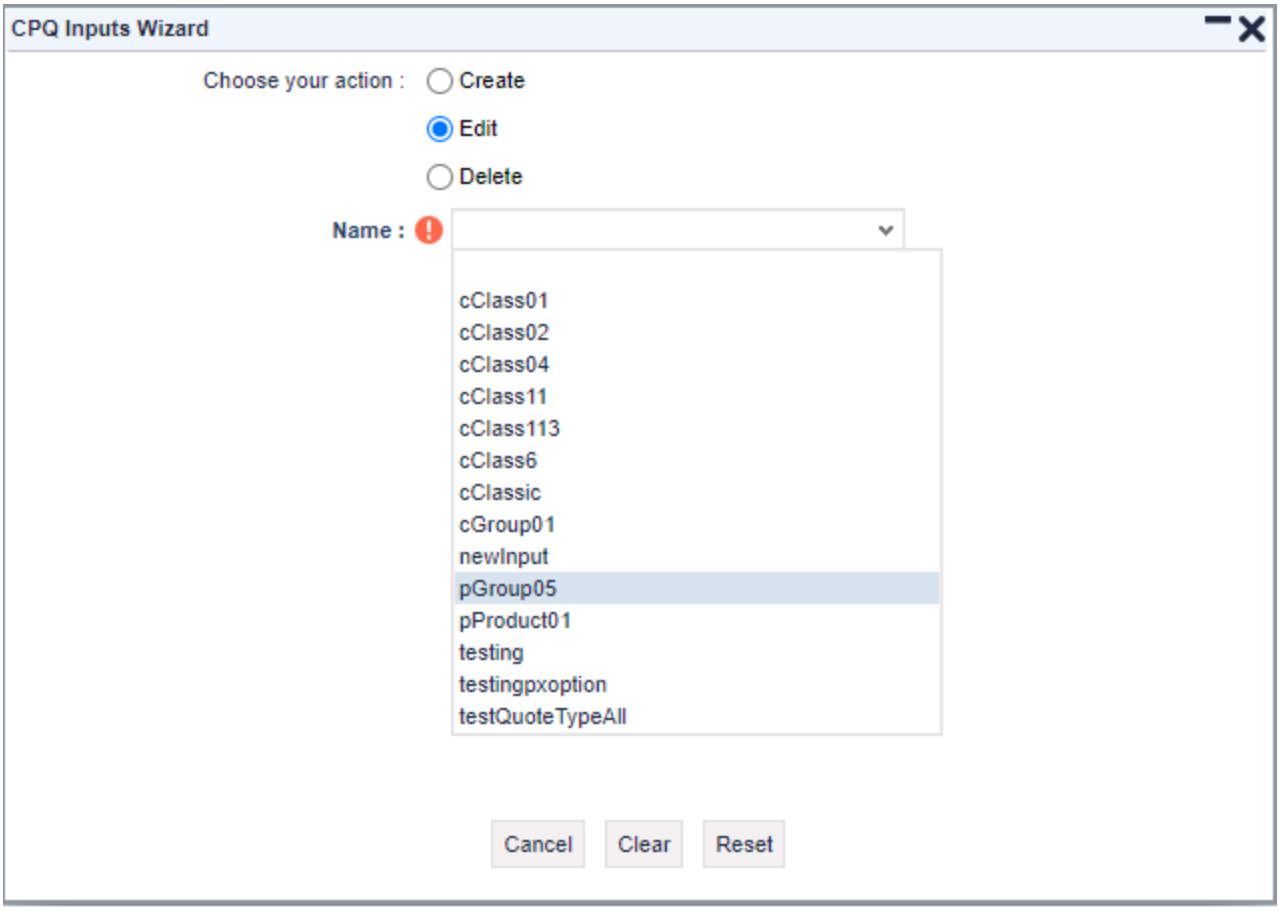
If you choose OPTION/ OPTIONS as Input Type, some hidden input fields will be displayed:



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):

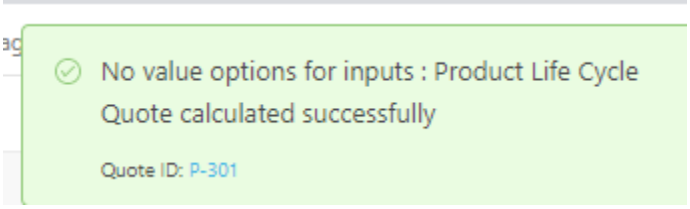
- OP_EQUAL (default for other supported input types)
- OP_NOT_EQUAL
- OP_LESS_THAN
- OP_GREATER_THAN
- OP_LESS_OR_EQUAL
- OP_GREATER_OR_EQUAL
- OP_LIKE
- OP_ILIKE
- OP_IN (default for input type is OPTIONS)
- OP_NOT_IN

+ 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:

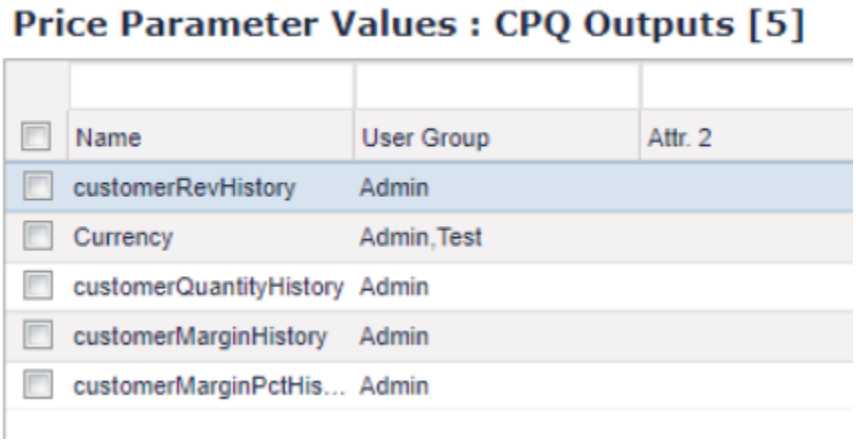


CPQ Outputs Wizard

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 Inputs Configuration 1.x - OBSOLETE

⚠ 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

CPQ Outputs Configuration - OBSOLETE

⚠ This CPQ Outputs Configuration is obsolete from version 2.0.

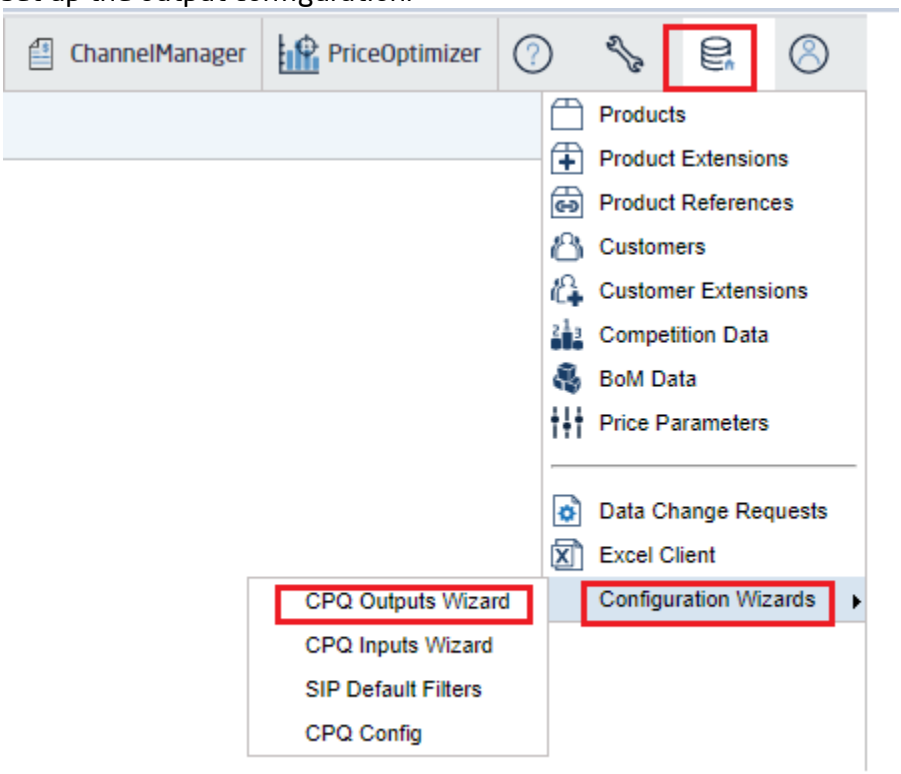
The CPQ Outputs Configuration is used to define configurations for a CPQ output.

Details

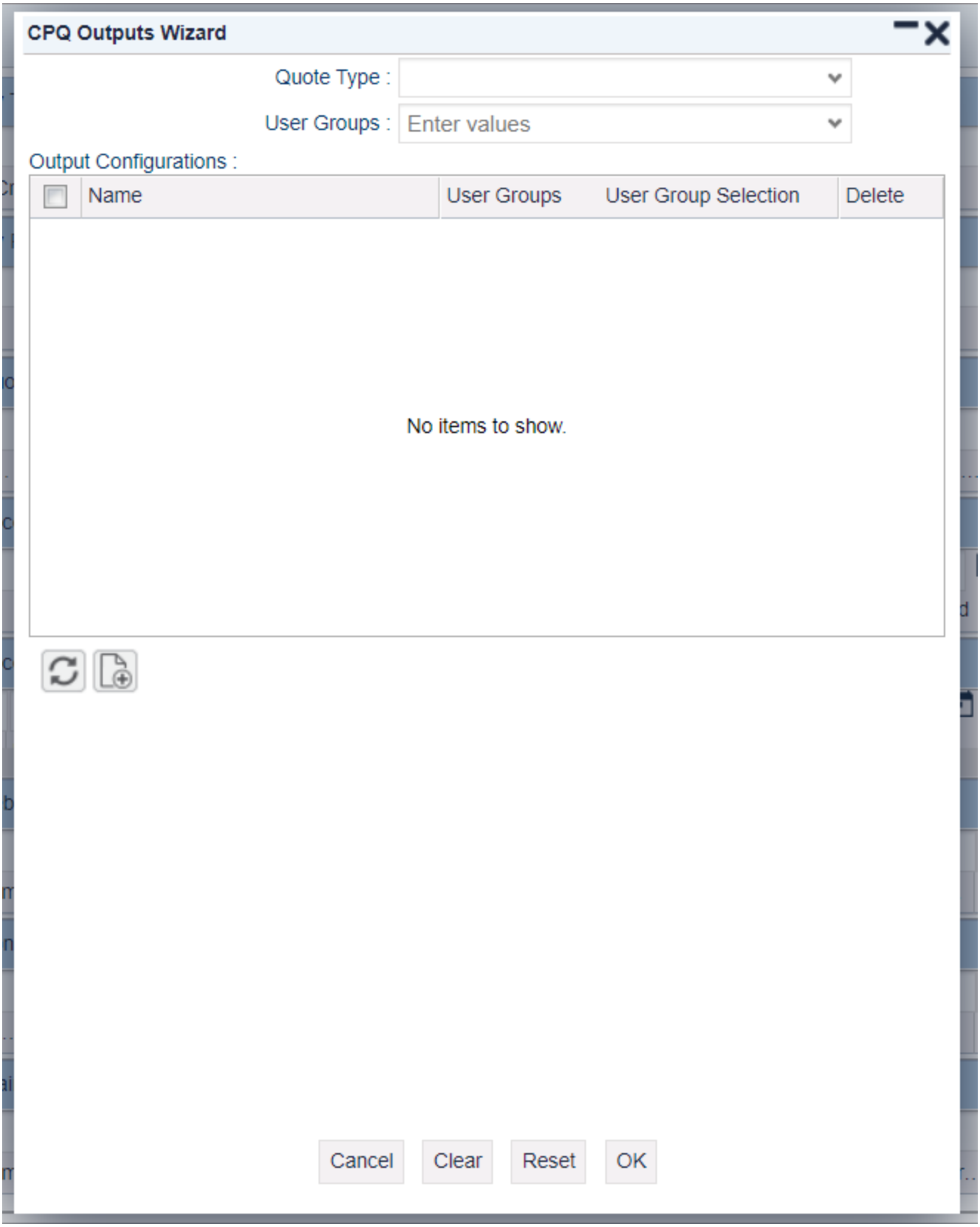
Name	Description
Name	Name of the output to configure.
User group	Defines a user group for the output.

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.

Configurations Per Quote Type - OBSOLETE

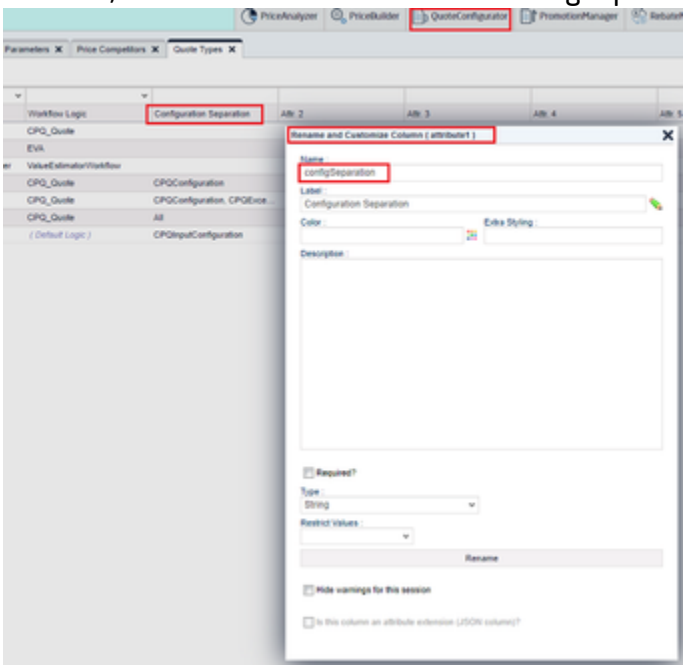
⚠ In version 2.0, this configuration is removed.

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

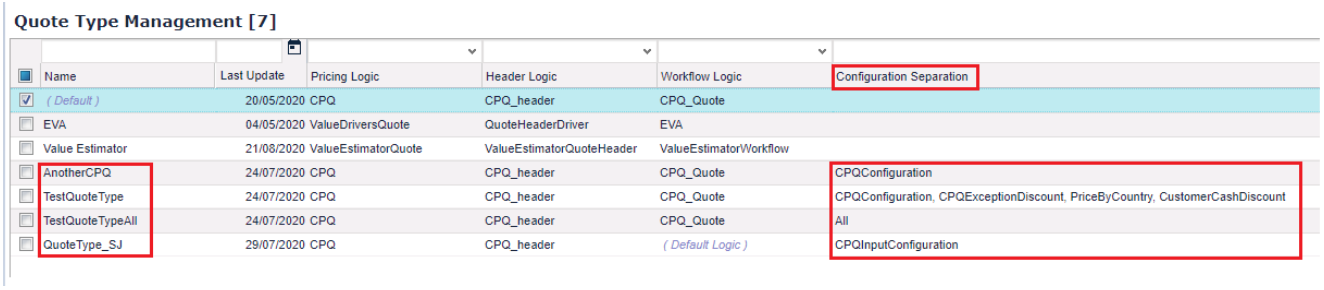
Set a Quote Type

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.

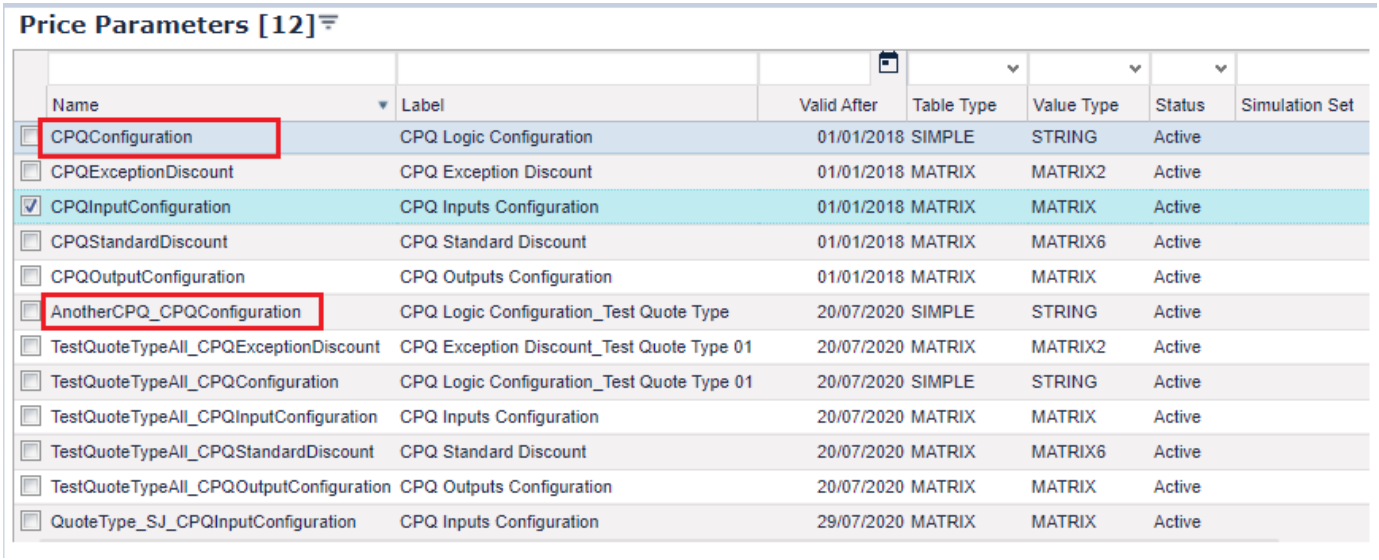


Quote Type Management [7]

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 Configurations

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

- Core Configuration
- Matrix Price List Lookup
- Integrations

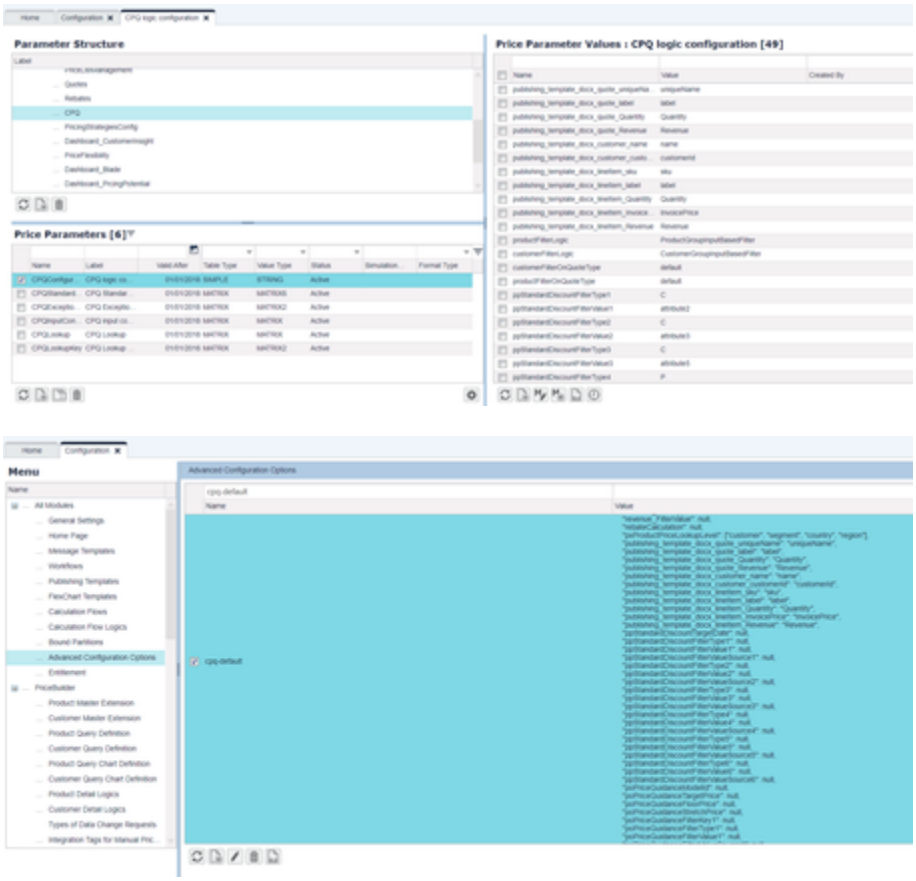
Core Configuration

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_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
pxCostTable	Required if the configuration at quote type level is not defined. The PX name that contains the product cost.
pxCost	Required. The column contains cost data. Default “attribute1”.
pxCostCurrency	Optional. The column contains currency data.

pxCostTargetDate	Optional. The column contains valid from date data.
pxCostValidToDate	Optional. The column contains valid to date data.
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.
pxCostFilterValueSource 1|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
pxCompetitorCurrency	Competitor currency column

Price Guidance Configuration

Price Guidance can be obtained from PX and PriceOptimizer.

PriceOptimizer Configurations

Name	Description
poPriceGuidanceModelName	PriceOptimizer model ID for the product price guidance
poPriceGuidanceTargetPrice	Target price column
poPriceGuidanceFloorPrice	Floor price column
poPriceGuidanceStretchPrice	Stretch price column
poPriceGuidanceCurrency	Currency column

poPriceGuidanceSegmentEntryLookupLevel1|2|3|4|5|6|...	Segmentation name for each level. You need to configure all segmentation levels that the selected model defines.
poPriceGuidanceSegmentEntryLookupTypeLevel1|2|3|4|5|6|...	The source data type for the segmentation. Supported types: • P - Product • C - Customer • PX - Product extension • CX - Customer extension The system will get the data from the source for specific product/customer and search for the segmentation.
poPriceGuidanceSegmentEntryLookupColumnLevel1|2|3|4|5|6|...	The attribute of the source type that is used to get data for each Segmentation.
poPriceGuidanceSegmentEntryLookupTableLevel1|2|3|4|5|6|...	Name of PX or CX to get the value.

Product Extension Configuration

Name	Description
pxGuidanceTargetPrice	Target price column
pxGuidanceFloorPrice	Floor price column
pxGuidanceStretchPrice	Stretch price column
pxGuidanceTargetDate	Target date column
pxGuidanceCurrency	Currency column
pxGuidanceFilterKey1|2|3|4|5|6|...	Optional. The column of price guidance product extension table that is used in the filter.
pxGuidanceFilterType1|2|3|4|5|6|...	Optional. The source data type. Supported types: • P - Product • C - Customer • PX - Product extension • CX - Customer extension
pxGuidanceFilterValue1|2|3|4|5|6|...	Optional. The attribute of the source type.
pxGuidanceFilterValueSource1|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
predefinedBindingAsScriptProperty	Auto-injected predefined binding as library elements variables. See Predefined Variables for more details.

Currency Conversion Configuration

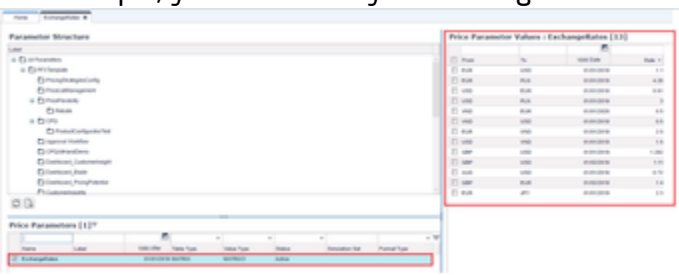
The Currency Conversion Configuration is used to convert currency based on the customer currency selection in a Quote header. If your data (price, cost, discount...) contains information about currency, it can be used in the CPQ Package. The currency conversion will be applied based on your provided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

Conversion Rate Lookup

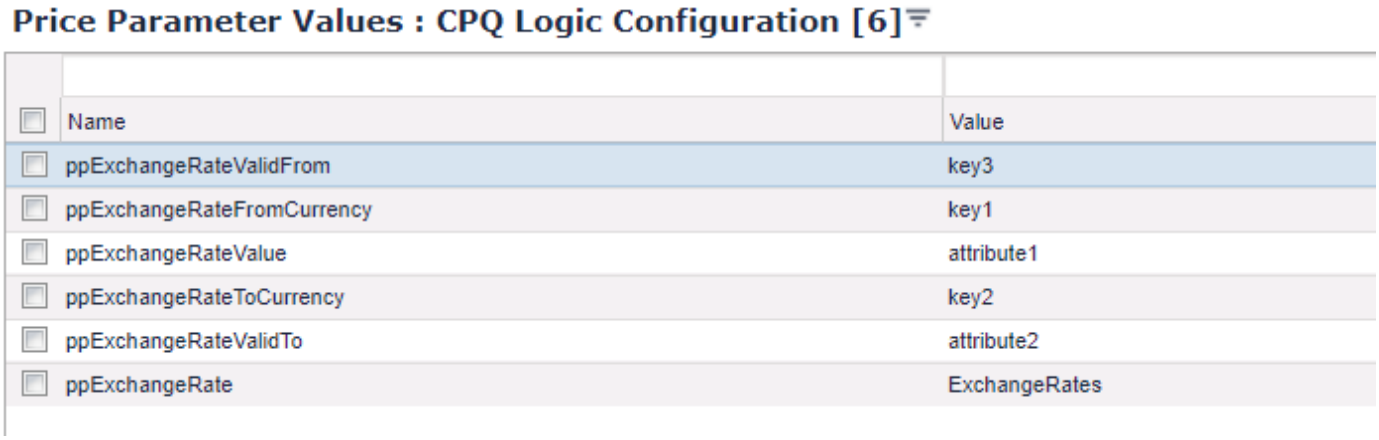
The ccy data source is the default source for your conversion rates. You can also define your custom conversion rates. CPQ supports fetching the conversion rate from a Price Parameters (PP) table. The configuration is:

Name	Description
ppExchangeRate	Name of a Price Parameters table which contains conversion rates
ppExchangeRateFromCurrency	From Currency column
ppExchangeRateToCurrency	To 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:



Lookup Details Configuration

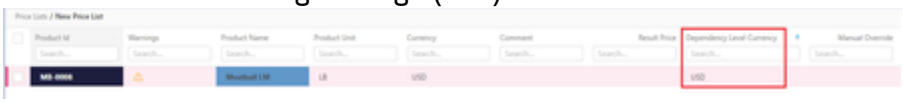
Lookup	Name	Description
Default Currency	defaultCurrency	Displays the default currency. It will be used if there is no defined currency.
Price List lookup	priceListItemCurrency	Lookup currency column in Price List
Product Prices lookup	priceByCustomerLevelCurrency	Lookup currency column in PX Price By 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	pxCompetitorCurrency	Lookup currency column in PX Price Competitors
Competitor Data	Currency	PCOMP - Currency read from a predefined column
Price Guidance PX	pxGuidanceCurrency	Lookup currency column in PX Price Guidance
	poPriceGuidanceCurrency	Lookup currency column in Price Optimization model

Price Guidance from Price Optimizer model		
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.

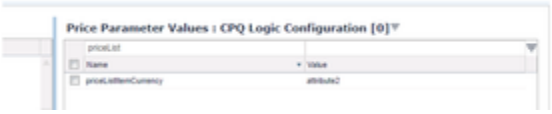
For Product Prices Lookup, it requires you to change the attribute size when adding the currency column. Please keep in mind that it may lose your data after each deployment because the original attribute size (3) will be deployed again. Please back up first in this case.

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.

Note:

- If we have defaultCurrency and do not map the Currency configuration (e.g. Cost, Price Guidance,..), the default currency is used.
- If we do not map both defaultCurrency and Currency configuration (e.g. Cost, Price Guidance), the original value is used without currency conversion.

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.

Integrations

- Sales Compensations

Sales Compensations

- Overview
- Configurations
  - Value Source
  - Value Configurations

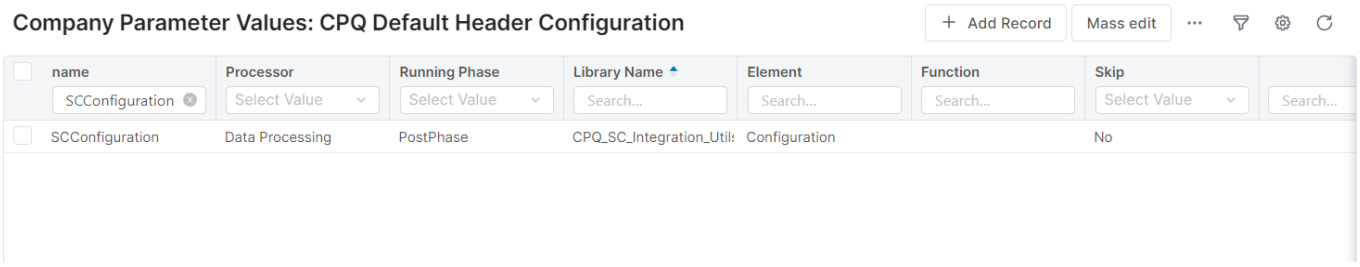
Overview

CPQ is configured to integrate with Sales Compensations by default.

A `SCConfiguration` task is configured on the `Data Processing` processor that provides the integration configuration.

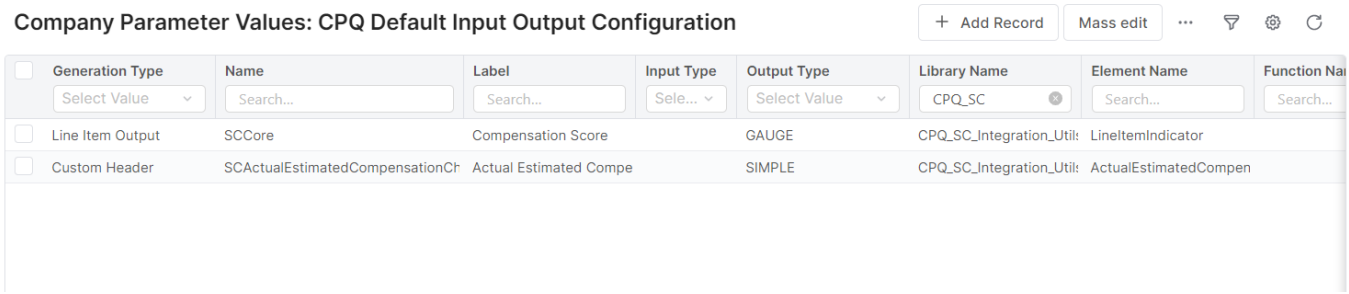
This is a required configuration so that the Sales Compensations Integration library can work with CPQ.

The task name `SCConfiguration` is a reserved name and should not be changed. This name is referenced by the outputs generator.

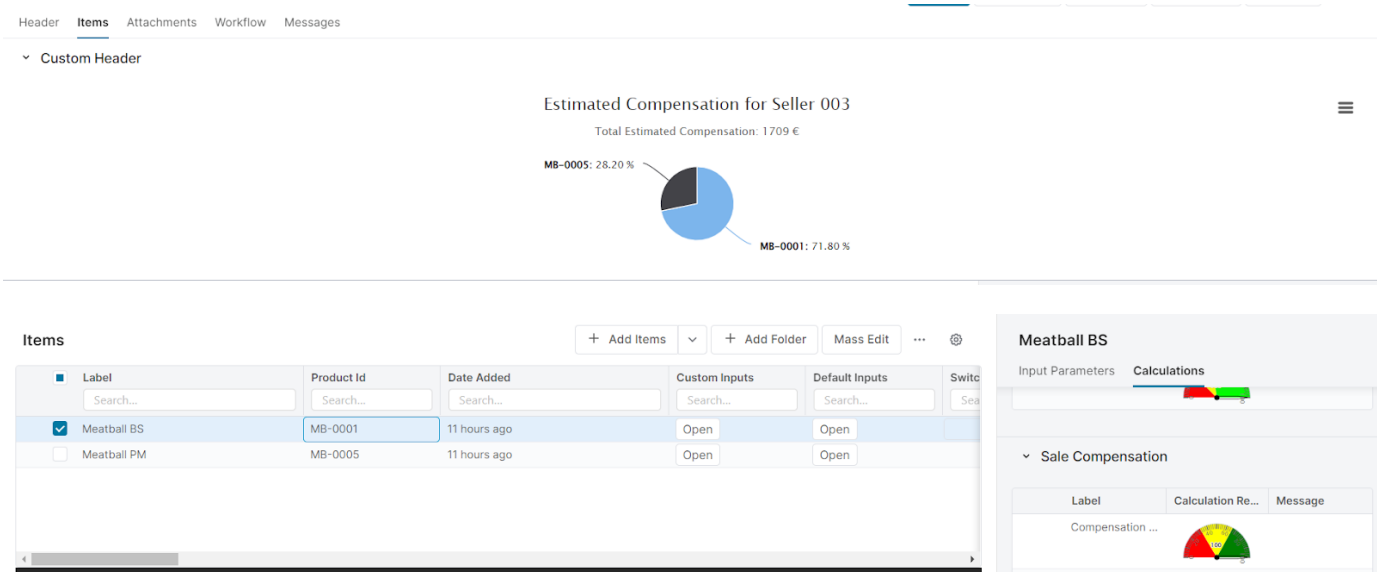


Outputs task is configured to provide the presentation of the Sales Compensations integration on the quote header and line item level.

The Customer Header output is based on the data of the Line Item output.



There will be charts added into line items and header.



Those tasks can be skipped by setting the `Skip` column to *Yes*.

Configurations

Sales Compensations require a few parameters to be calculated. The configurations define how to get data from CPQ to integrate with Sales Compensations.

Value Source

The source defines where to get the value in the CPQ. There are three value sources supported:

- Input - Value is taken from the line item input.
- Output - Value is taken from the line item output.
- Static - Configured value is used.

Value Configurations

Area	Configuration	Description
Quantity	`CPQ_SC_Integration_Quantity_Source`	Source of value. The default value is `Output`.
	`CPQ_SC_Integration_Quantity_Value`	The default value is `Quantity`.
Recommended price	`CPQ_SC_Integration_RecommendedPrice_Source`	Source of value. The default value is `Output`.
	`CPQ_SC_Integration_RecommendedPrice_Value`	The default value is `TargetPrice` (from Price Guidance).
Invoice price	`CPQ_SC_Integration_InvoicePrice_Source`	Source of value. The default value is `Output`
	`CPQ_SC_Integration_InvoicePrice_Value`	The default value is `InvoicePrice`.

Switch month	`CPQ_SC_Integration_SwitchMonth_Source`	Source of value. The default value is `static`.
	`CPQ_SC_Integration_SwitchMonth_Value`	The default value is `0`.

Working with QuoteStructure

QuoteStructure can be used to add line items to a quote using Groovy code. This helps in some use cases, such as importing quote line items from an Excel file.

From version 2.2.0, line items added using `QuoteStructure` can be handled by the CPQ Accelerator. This section will show you how to work with `QuoteStructure` in CPQ.

Utils

CPQ provides some public utils to make it easier to work with the `QuoteStructure`:

- `void addQuoteStructure(QuoteStructure quoteStructure)` - Loads the provided quoteStructure so that the CPQ knows which line items are going to be loaded and calls `quoteProcessor.addQuoteStructure(QuoteStructure quoteStructure)` to add quoteStructure into the system.
- `void addLineItem(String parentLineId, String key, List<Map<String, Object>> contextParameter)` -Loads the provided line item so that the CPQ knows which line item is going to be loaded and calls `quoteProcessor.addLineItem(String parentLineId, String key, List<Map<String, Object>> contextParameter)` to the add line item into the system.
- `void addLineItem(String key, List<Map<String, Object>> contextParameter)` - Loads the provided line item so that the CPQ knows which line item is going to be loaded and calls `quoteProcessor.addLineItem(String key, List<Map<String, Object>> contextParameter)` to add the line item into the system.

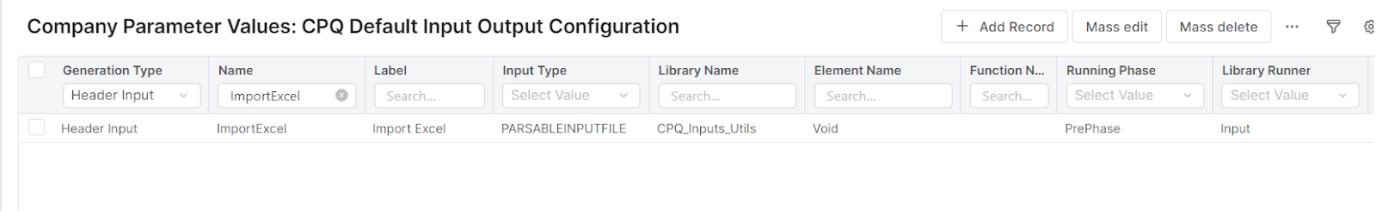
CPQ requires QuoteStructure / Line Items added using Groovy to be loaded in the pre-phase so that the inputs and outputs are generated correctly.

It is recommended in the Data Processing / Query with the running phase to set it to PrePhase.

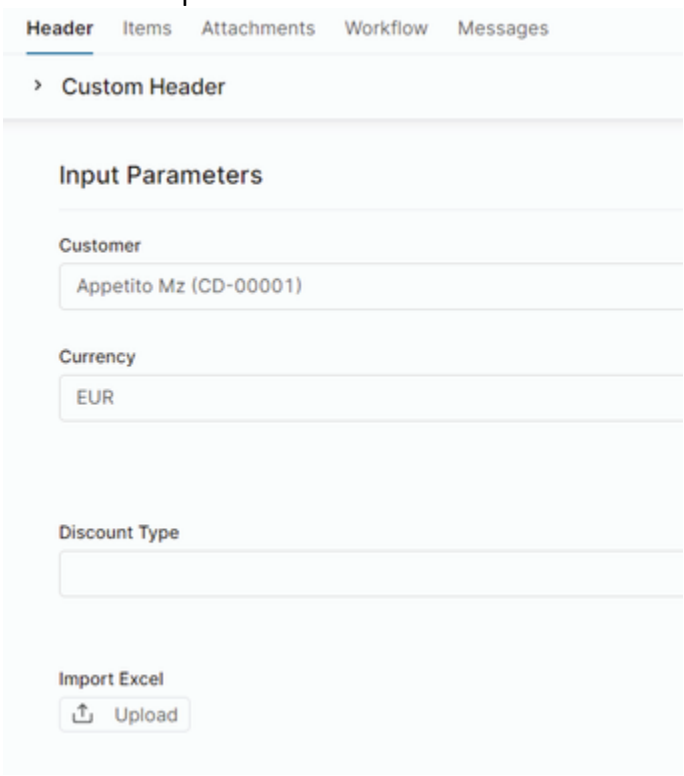
Samples

This example shows a use case where a user uploads an Excel file with line items and adds them to a quote.

You need to configure a file input on the header that allows the user to upload the Excel file. You need to load the Excel only once when the user uploads it, for that use the provided library `CPQ_Inputs_Utils.Void` which removes the user input with every recalculation.



Result in the quote:



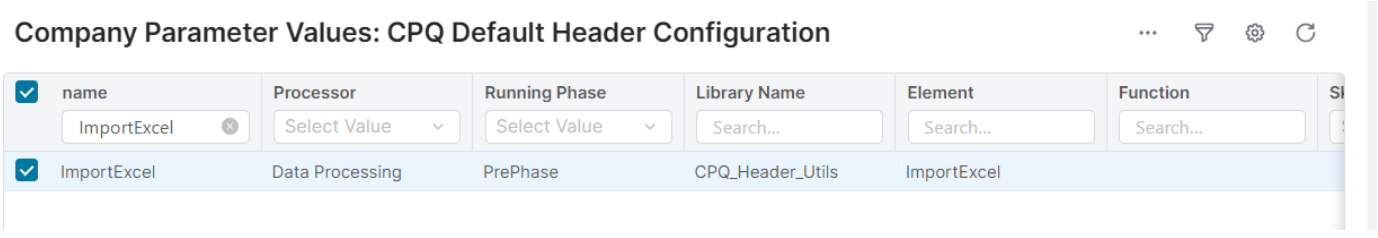
Then, you need to write a library to process the Excel file and add line items using QuoteStructure. Create a library and implement it, with the ImportExcel element in the `CPQ_Header_Utils` library. Of course, you are encouraged to separate CPQ core logic and custom logic.

```
import net.pricefx.common.apibuilder.quote.QuoteStructure

def execute() {
   List<List> rows = getExcelData()

   if (rows?.size() < 2) {
       return
   }

   QuoteStructure quoteStructure = buildQuoteStructure(rows)

   libs.CPQ_SharedLib.QuoteUtils.addQuoteStructure(quoteStructure)
}

protected List<List> getExcelData() {
```

```
    def quoteUtils = libs.CPQ_SharedLib.QuoteUtils
    String excelImportInputName = "ImportExcel" //Configured header
input name
    String importedFileHandler = quoteUtils.getLineItemInputMap("ROOT")
[excelImportInputName]
    def excelData = api.parsableInputFileDataFromHandle
(importedFileHandler)

    return excelData?.data?.Data
}

protected QuoteStructure buildQuoteStructure(List<List> rows) {
    int skuIndex = 0
    int colorIndex = 1
    def quoteStructure = new QuoteStructure()

    List rowLabels = rows[0]
    //Remove the label row
    rows = rows?.subList(1, rows?.size())

    String colorName = rowLabels[colorIndex]
    String colorLabel = colorName.capitalize()

    rows.each { List row ->
        String sku = row[skuIndex]
        String color = row[colorIndex]

        List inputs = [["name": colorName, "label": colorLabel, "value":
color]]
        quoteStructure.addPart(sku, inputs)
    }

    return quoteStructure
}
```

The above logic simply reads the Excel file from the input, reads its data to build `QuoteStructure`, then uses the quote utils to apply the `QuoteStructure`.

After the logic is created, you configure it to run the library in CPQ.



Now, you can upload the Excel file and recalculate it to see if the new line items from the file are added to the quote.

CPQ Element Names for Approval Workflow

Accelerate Approval Workflow Package can be used with CPQ package.

To learn what elements can be used in workflow step conditions see Values for Condition Expressions.

Configurator Package Integration

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

Product Configurator

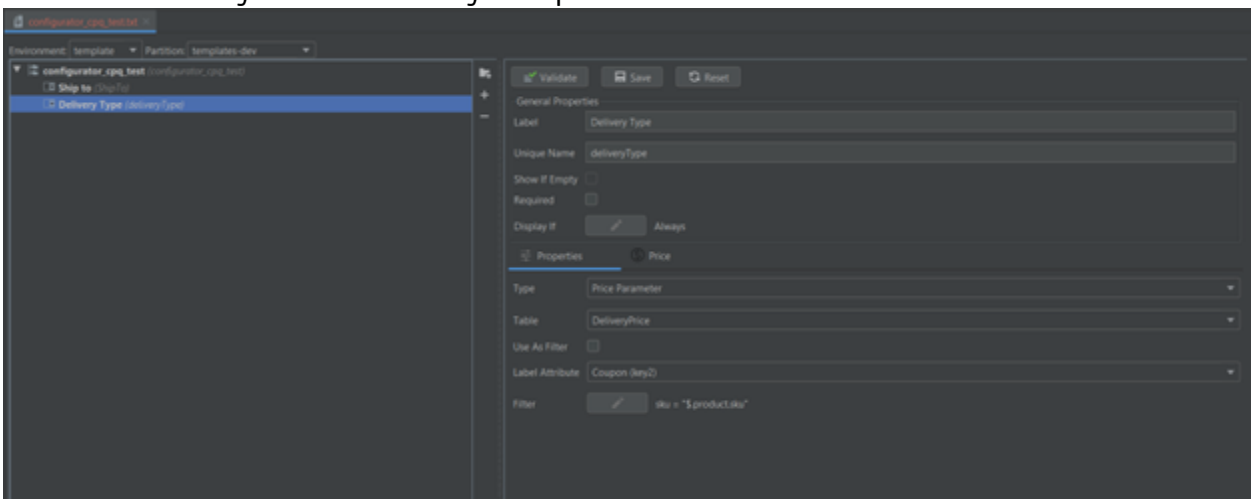
Integration with Configurator Package (https://gitlab.pricefx.eu/accelerators/configurator-package)

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

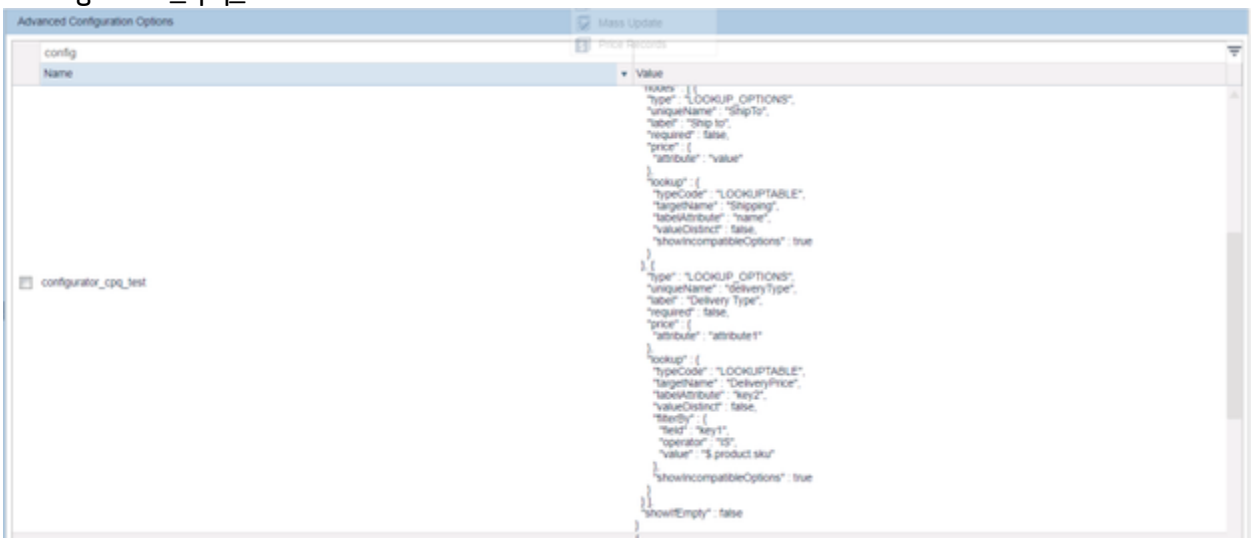
Usage

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:



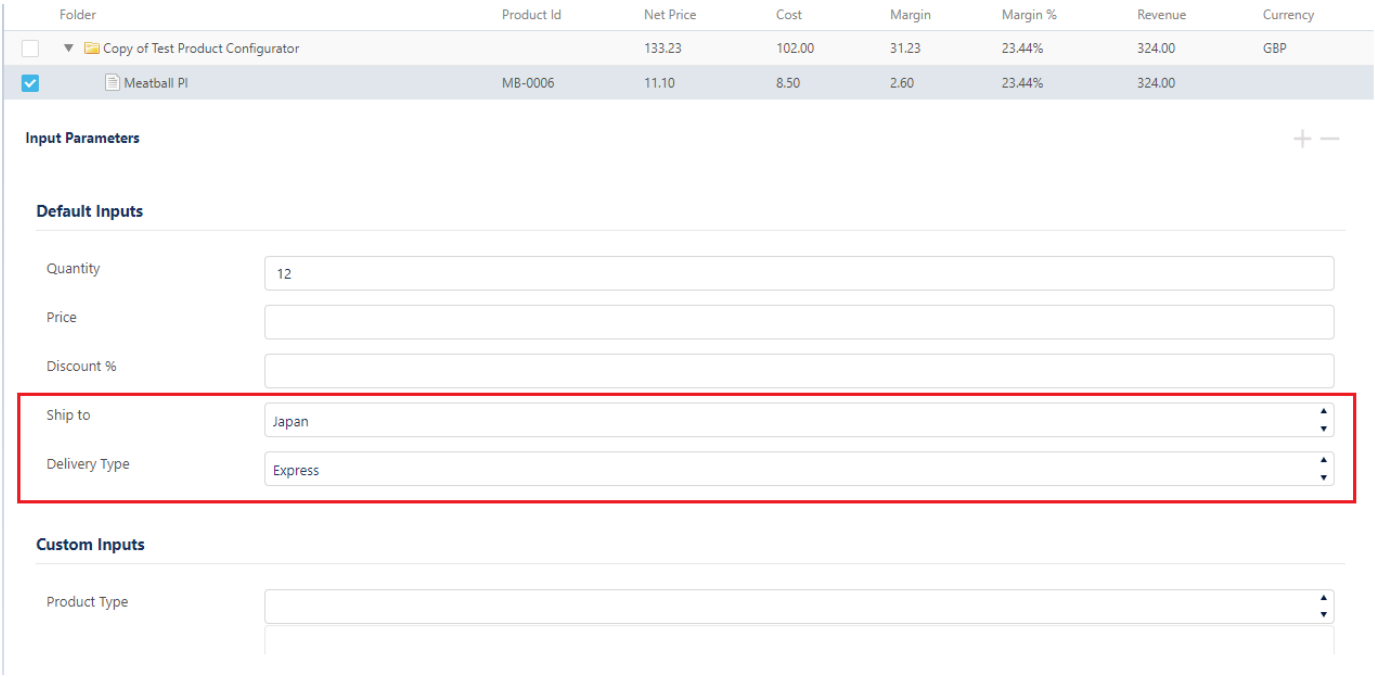
It is deployed now and its configuration is stored in AdvancedConfiguration under the name "configurator_cpq_test".



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

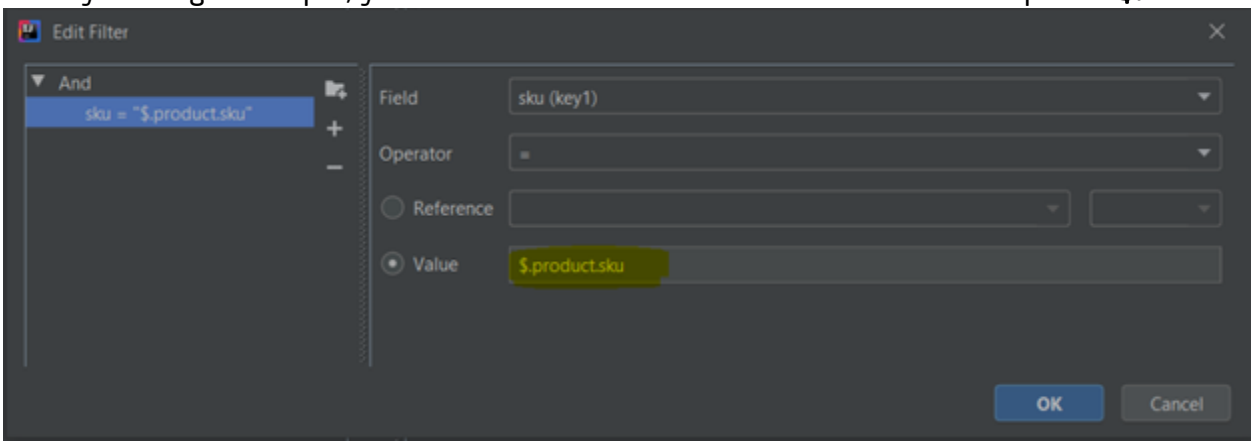
By default, the price will be the total of all input prices. However, you can calculate it using your formula.

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

Glossary (CPQ)

This glossary summarizes various terms used in CPQ Accelerator.

Sales Overview

Term	Description
X-months Revenue	X-months Invoice Price of a customer and a product
X-months Volume	X-months Volume of a customer and a product
X-months Margin %	X-months Margin / X-months Net Price of the same customer and product
X-months Weighted Price Per Unit	X-months Invoice Price / X-months Quantity of the same customer and product
Quoted Price Per Unit	Invoice Price of the product in a quote
Quoted Margin %	Margin % of the product in a quote

Customer History

Term	Description

Customer Revenue	Total Invoice Price of a customer based on a historical period
Customer Quantity	Total Quantity of a customer based on a historical period
Customer Margin	Total Margin of a customer based on a historical period
Customer Margin %	[Margin / Net Price] of a customer based on a historical period

Product History

Term	Description
Historical Period	Shows a historical period (month) to get data from (From - To)
Product Revenue	Total Invoice Price of a product based on a historical period
Product Quantity	Total Quantity of a product based on a historical period
Product Margin	Total Margin of a product based on a historical period
Product Margin %	[Margin / Net Price] of a product based on a historical period
Product Weighted Avg. Price	[Total Invoice Price / Total Quantity] of a product based on a historical period Note: Invoice Price is multiplied by Quantity for every transaction data
Avg. Revenue Per Invoice	[Total Invoice Price / Number of transaction] of a product based on a historical period
Avg. Quantity Per Invoice	[Total Quantity / Number of transaction] of a product based on a historical period
Avg. Margin % Per Invoice	[Average of Margin / Average of Net Price] of a product based on a historical period

Previous Price

Term	Description
Unit Price	Latest and previous transacted price per unit of the same customer and product Invoice Price / Quantity
Margin %	Latest and previous transacted Margin % of the same customer and product Margin / Net Price
Date	Latest and previous date of the same customer and product

Release Notes (CPQ)

CPQ Package 2.2.0

Stories

PFPCS-5397 Inputs/outpus of lineItems added by groovy should show after first calculation

PFPCS-5705 Input values of line items added by QuoteStructure should show in quote after recalculating

PFPCS-5719 Add QuoteStructure public utils

PFPCS-5720 Skip the configured input execution if a line item is added using Groovy

PFPCS-5721 Support for folder structure

PFPCS-5722 Predefined bindings values should contain the input values added by Groovy

PFPCS-5965 SC Quoting Plugin - Actual and Max compensation values on line item outputs

PFPCS-6053 SC Integration configuration wizard

PFPCS-5717 Review/remove AlertType configuration support

PFPCS-5990 Sale Compensation Integration plugin - support for min compensation

PFPCS-6052 Remove Custom Input generator by default

PFPCS-5563 CPQ - Simplify LibraryRunner

PFPCS-5564 CPQ - Simplify ErrorHandler util

PFPCS-5565 CPQ - optimize logic in TaskProcessor

PFPCS-5566 CPQ - optimize output generator

PFPCS-5567 CPQ - optimize rebate calculation on lineItem

PFPCS-5573 CPQ - remove deprecated element from output config

PFPCS-5625 Using api.find instead of api.stream

PFPCS-5681 Logics simplification

PFPCS-5701 VolumeBreakDown lookup executes too many times

PFPCS-5723 TaskProcessor instance should be initiated once time for each generator

PFPCS-5730 The PO guidance should be skipped when there is no model name config

PFPCS-5736 Performance - Review Data query and data processing tasks

PFPCS-5739 Update QuoteStructure shared cache to allow adding Line items multiple time without data overridden

PFPCS-5861 Output Utils - default empty input value

PFPCS-5954 Improve the QuoteUtils to support multi parents folder when using addLineItem method

PFPCS-6056 Library Runner - print library execution error to log

PFPCS-6324 PLI Currency configuration for different PL logics

Bugs

PFPCS-5581 Adjusting CPQ_SharedLib

PFPCS-5651 Missing RunningPhase column config in default input/output PP table

PFPCS-5658 Incorrect format type in default output configuration PP

PFPCS-5660 NPE throw when adding new product in quote

PFPCS-5674 Result type config does not take effect

PFPCS-5700 Error when calculate quote - Invalid Additional Discount

PFPCS-5702 Error "No signature of method" when configuring Publishing Template in CPQ Configuraiton Widzard

PFPCS-5704 "inputs" predefined binding does not contains header input values

PFPCS-5709 Invalid "Task result shared to Postphase"

PFPCS-5716 Square brackets automatically added every time configure Price Configuration in CPQ Configuration Wizard

PFPCS-5733 List Price displays 0.00 when input price in Input Parameters

PFPCS-5750 There should be a warning when entering a negative value for Historical period

PFPCS-5780 Wrong order for Price Guidance

PFPCS-5806 Customer Header - Cannot get property 'quoteCreator' on null object

PFPCS-5963 Price guidances show as percentage

PFPCS-6054 Configuration wizard - invalid "Input Priority" loading and saving

PFPCS-6087 Fix documentation link in CPQ package

PFPCS-6299 List Price Source is changed when changing Quote structure

PFPCS-6301 Estimated Compensation is not correct when adding folder structure

PFPCS-6305 Fix link inside SC Integration Configuration wizard

PFPCS-6316 List Price value equals 0 when List Price Source is a Dependent Price List

PFPCS-6366 Top 5 Competitors list only gets the first data found from Competition Data master/PX

PFPCS-6378 List Price from Matrix PriceList is incorrect

CPQ Package 2.1.1

New Features

PFPCS-5768 SC Integration - Header summary chart

PFPCS-5767 SC Integration - Line Item indicator

Bugs

PFPCS-5822 Quote throws error if Customer ID is not defined

PFPCS-5819 Line Item indicator - value is not correct if source of value is static

CPQ Package 2.1.0

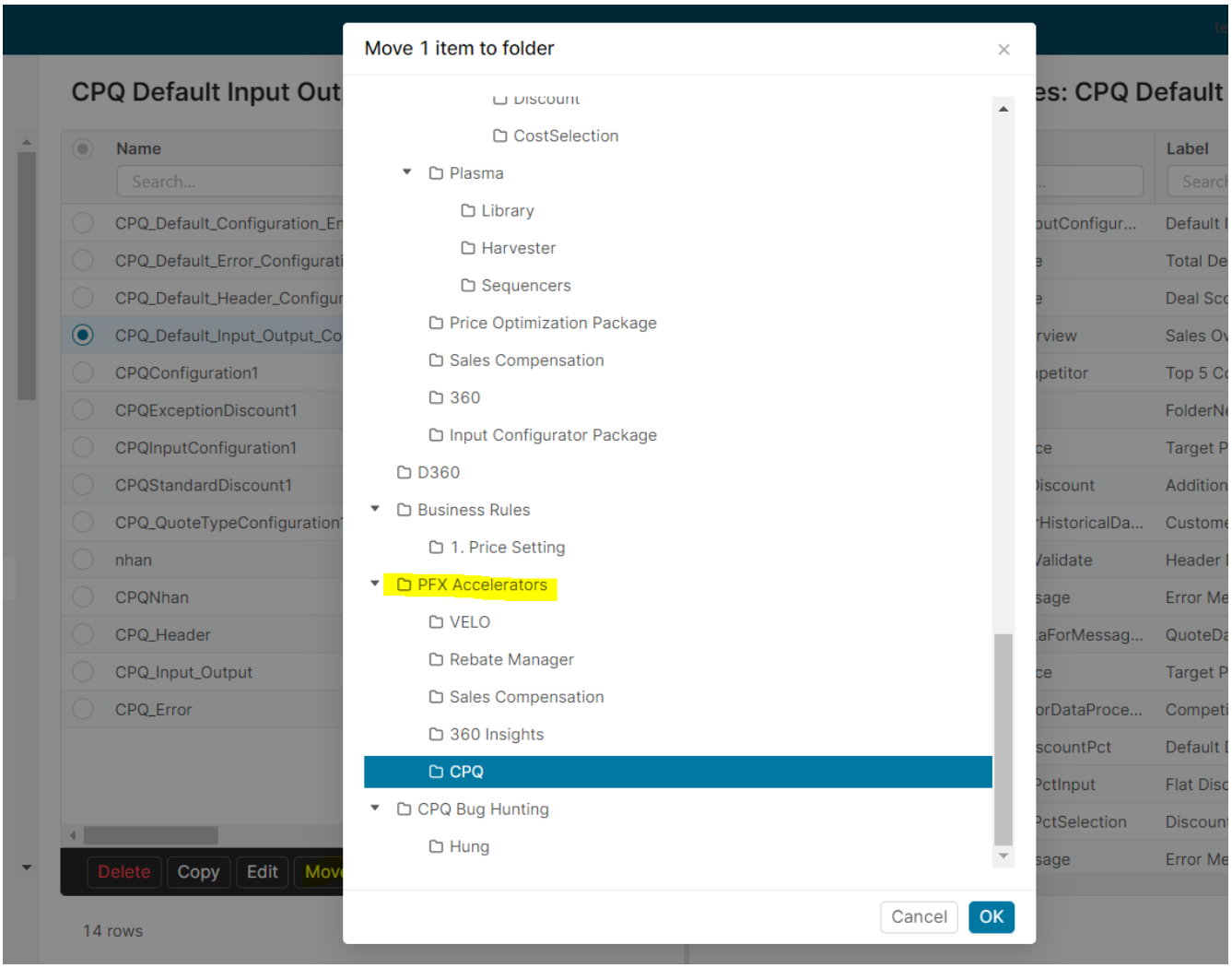
Please back up your existing CPQ Company Parameters tables before upgrading to CPQ Package 2.1.0.

Release Highlights

- New Configuration wizard UIs
- Improved Optimization module integration
- Improved performance
- Added support for more input/output configurations
- More configuration mapping
- Flexible lookup filter configuration

Upgrade Notes

- The default PP table's folder has changed to "PFX Accelerators/CPQ". So, you need to move all of the existing default PP tables to the new location



- The predefined variables auto-injection mechanism is now configurable – users can disable the injection to improve performance. See Predefined Variables for the details.

Change Requests

- PFPCS-5085 Data lookup using new BatchLookupUtils

Improvements

PFPCS-5463 Historical Datamart name configuration should be moved to ConfigurationEntry table

PFPCS-5436 Submit button is not greyed out and disabled on critical alert raising

PFPCS-5435 CPQ Accelerator doesn't support calculation result override feature

PFPCS-5366 Show Segment Id in Price Guidance calculation result in Quote

PFPCS-5354 Add Currency field mapping in Price Competitors wizard

PFPCS-5352 CPQ Wizard: Put a short introduction for each configuration back

PFPCS-4851 Change default output Quantity to NUMERIC

PFPCS-3572 Add ability to store Price Guidance by Product and Customer (CPQ Accelerator)

PFPCS-5502 Potential "Too many new instances created"

PFPCS-5499 Deployment step - Add Product Cost mapping

PFPCS-5498 Introduce new quote type named CPQ in Quote Types configuration

Stories

PFPCS-5467 Check mandatory fields in CPQ and fix them

PFPCS-5272 PX price guidance configuration wizard

PFPCS-5251 Reduce Duplication In Wizard Screen Code

PFPCS-5235 Recreate Historical Data Screen

PFPCS-5201 Recreate Publishing Template Screen

PFPCS-5189 Recreate Other Screen

PFPCS-5187 Recreate Rebate Screen

PFPCS-5179 Create Customer Mapping Screen

PFPCS-5161 Update Main config and remove unnecessary constant

PFPCS-5160 Price Competitor is not support for configuration

PFPCS-5106 CPQ quote type configuration in PP table

PFPCS-5097 Configuration Wizard - Currency Configuration Screen

PFPCS-4924 V2.x Performance review & improvement

PFPCS-4918 Show results from Deal Guidance (PO) in CPQ Accelerator

PFPCS-4917 Query Deal Guidance from CPQ Accelerator

PFPCS-4909 Configuration Wizard- Price Guidance Configuration

PFPCS-4908 Configuration Wizard - Discount Configuration Screen

PFPCS-4907 Configuration Wizard - Price Competitor Screen

PFPCS-4906 Configuration Wizard - Price Configuration Screen

PFPCS-4905 Configuration Wizard - Cost Configuration Screen

PFPCS-4904 Configuration Wizard - Main Screen

PFPCS-4903 Configuration Wizard - Quote Type Configuration Screen

PFPCS-4902 Configuration Wizard - Layout update

PFPCS-3601 PO lookup configuration support for new Model.

Bugs

PFPCS-5496 Library runner binding config - "No" value is not disabled the feature

PFPCS-5495 Restructure CPQ folder

PFPCS-5489 Error thrown in quotes when not mapping Datamart

PFPCS-5482 Configuration per quote type: Incorrect Historical Data values

PFPCS-5481 Configuration per quote type: Source (Segment Id/ Price Guidance PX table name) is not displayed

PFPCS-5480 Missing pre-defined values (Critical, Yellow, Red) in Alert Type

PFPCS-5479 CPQ Input Output wizard: Missing Overridable, Overridden, Alert Type, Alert Message

PFPCS-5470 Missing table name configuration for error, header processing, input output

PFPCS-5469 Configuration per quote type: Competitor does not get default entry table when having incorrect entry table structure

PFPCS-5465 Unused Names in CPQ_Default_Configuration_EntryPP

PFPCS-5464 Page Not Found when clicking a link in wizard

PFPCS-5462 PX Price Guidance shows 0 when no mapping Currency in wizard

PFPCS-5460 Price Guidance PO gets defaultCurrency configuration instead of base currency in DM

PFPCS-5457 Price Guidance is not converted correctly when looking up from PO

PFPCS-5455 Missing required inputs in publishing templates

PFPCS-5454 Issue when adding subfolder in quote

PFPCS-5449 All steps are optional during PlatformManager deployment

PFPCS-5447 Rebate shows data while no rebate calculation plan is configured

PFPCS-5443 Missing 0 before decimal places in publishing templates

PFPCS-5441 null is displayed in Currency input when default currency is not defined

PFPCS-5439 Price Guidance wizard: Able to go next step while required inputs are not filled

PFPCS-5438 PO Price Guidance wizard: No data is displayed in Guidance Mapping when leaving required inputs empty

PFPCS-5420 Existing PX table configured before is not displayed when going to wizard

PFPCS-5416 Missing Back button in Rebate wizard

PFPCS-5381 Cost: ValidTo in Cost table does not work if expiryDate in Quote < currentDate (calculationDate)

PFPCS-5380 PX Price Guidance filter config does not work with CX

PFPCS-5379 Customer Assigned PL priority not considered when performing Product Price Lookup

PFPCS-5378 Target Date in Price PX lookup does not work

PFPCS-5374 List Price gets from PX does not work

PFPCS-5370 Wizard: Both name and attribute are displaying together in dropdown list

PFPCS-5369 Wizard - Filter config screen: Error when change existing filter type config from P/C to PX/CX

PFPCS-5367 Incorrect price guidance returned when looking up from PO

PFPCS-5365 PriceGuidance PO wizard: Missing required input

PFPCS-5353 Error message: name is being displayed instead of Message in quote

PFPCS-5349 Historical Data wizard: Missing Revenue Filter Type & Revenue Filter Value

PFPCS-5348 Previous Price result: Showing Invalid Date in Date

PFPCS-5329 Configuration per quote type - default input configuration does not work for specific quote type

PFPCS-5318 Lookup config - add default table structure config

PFPCS-5304 PX Price Guidance lookup - data is not shown

PFPCS-5302 PX price guidance configuration wizard - cannot change PX table

PFPCS-5301 Customer cash discount table structure documentation - add target date

PFPCS-5299 Customer cash discount lookup configuration - Currency config is unnecessary

PFPCS-5297 CPQ_QuoteType configuration should be documented

PFPCS-5296 Documentation - default configurations should be mentioned in the Confluence document

PFPCS-5295 Configuration wizard: Filter builder is not initialized error when apply empty filter config

PFPCS-5294 Lookup Util - groupingProperties not work with generic value

PFPCS-5292 standard discount unused configurations in default data

PFPCS-5291 CPQ Configuration wizard - label for key column is not loaded for selection

PFPCS-5270 Cannot show all PP field in configuration wizard

PFPCS-5188 Old configuration do not removed when using cpq configuration wizard

PFPCS-4879 Lookup keys are not converted to technical name

PFPCS-4878 Configuration Wizard - PX field is not loaded to select when its columns is not renamed

PFPCS-4877 Price Guidance from PO failed to load when using configuration wizard to configure

PFPCS-4833 CPQ wizard - Duplicate configuration for Cost, Competitor table

PFPCS-4815 CPQ Configuration wizard - missing title and description

PFPCS-4814 CPQ Configuration wizard typo

PFPCS-3491 Fix documentation about PO configuration

PFPCS-2910 Configuration wizard - Not showing all columns of table in dropdown

CPQ Package 2.0.2

Release Highlights

- Improved CPQ Configuration Wizard layout with short instruction for each configuration
- Numerous bug fixes

Upgrade Notes

Before the deployment:

- First rename the logics of the existing version to prevent conflicts or overrides.
- Rename the label of existing CPQ Configuration Wizard in the Configuration Wizards Admin.

Improvements

PFPCS-4556 Hide Configure Filter Selected Key button when no configuration

PFPCS-4531 Allows to add more configuration entries per quote type

PFPCS-4527 Separate to smaller configuration selection

PFPCS-4482 Adjust some layouts in CPQ wizard

PFPCS-4454 Improve documentation for Additional Discount

PFPCS-4434 Display element name in the custom library that throws an error

PFPCS-4337 Duplicate behavior in output configuration: NONE and Skip

PFPCS-4136 Improve wizard for CPQ 2.0

PFPCS-4135 Add short introduction info in CPQ wizard

PFPCS-4067 Rearrange the order in Error matrix table

Change Requests

PFPCS-4435 CPQ Output Utils - should return unique data type for output

Tasks

PFPCS-4619 Update Calculation Logic valid date

PFPCS-4562 Update PP table valid after date

PFPCS-4222 Investigate about requiring to recalculate when adding new product

Sub-tasks

PFPCS-4573 Remove Unnecessary Code From Wizard V1

PFPCS-4445 Refactor Error Screen

PFPCS-4444 Refactor Header Screen

PFPCS-4440 Refactor Input Output Screen

PFPCS-4407 Refactor Generic Screen

PFPCS-4403 Refactor Currency Screen

PFPCS-4335 Refactor Publishing Templates Screen

PFPCS-4334 Refactor Price Guidance Screen

PFPCS-4333 Refactor Price Competitors Screen

PFPCS-4332 Refactor Cost Screen

PFPCS-4331 Refactor Rebate Screen

PFPCS-4283 Refactor Discount Screen

PFPCS-4278 Refactor Customer Screen

PFPCS-4271 Refactor Price Lookup Screen

PFPCS-4209 Refactor Historical Data for New Design

PFPCS-4193 Add Quote Type into Wizard Config

PFPCS-4160 Create Configuration Wizard Selection Page

PFPCS-4141 Create CPQ Configuration Entry Wizard

Bugs

PFPCS-4649 Update documentation link when page name is changed

PFPCS-4647 Publishing templates wizard - Mismatch returned value

PFPCS-4636 Issue with Error Handler CPQ

PFPCS-4633 Update the documentation link for CPQ in PlatformManager

PFPCS-4632 HIDDEN is not removed in Input Output wizard

PFPCS-4627 Publishing Templates - typeId is displayed instead of username if using CreatedBy

PFPCS-4625 Publishing Templates wizard- null is displayed in Name

PFPCS-4620 Issue with simple and matrix PL to display on quote

PFPCS-4612 Error thrown when adding product into folder

PFPCS-4600 CPQ wizard - Update documentation link

PFPCS-4572 Missing optional field TargetDate for CustomerCashDiscount CX

PFPCS-4442 Header Output Cannot Reset Predefined Elements

PFPCS-4297 Value displayed as name in CPQConfiguration PP does not work

PFPCS-4215 Missing output bindings

CPQ Package 2.0.1

Bug

- PFPCS-4138 History deployment step failed

CPQ Package 2.0.0

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

CPQ Package 1.2.2

- Improvements
- Bugs

Improvements

- [PFPCS-708] - Convert currency for List Price
  - **Support for converting currency for all data related to money**
    - It requires a definition of the currency column for some tables as needed. Please note that you might have to change the attribute size and your current data in that table may be affected too.
    - For more information see Currency Conversion Configuration.

Bugs

- [PFPCS-2745] - Incorrect warning message for currency when getting history data
- [PFPCS-2746] - Incorrect currency conversion for Cost
- [PFPCS-2749] - No currency conversion for history data
- [PFPCS-2753] - Incorrect currency conversion for PX, PP, Price List when not found exchange rates
- [PFPCS-2754] - Missing lookup currency configuration for PX Price Guidance
- [PFPCS-2755] - Incorrect currency conversion for Competition Data
- [PFPCS-2756] - Applied currency conversion for CX Customer Cash Discount when type is percentage
- [PFPCS-2758] - Change attribute size for PX
- [PFPCS-2772] - Incorrect warning message and conversion rate for Product Prices
- [PFPCS-2773] - Sort a list of currency in ascending (A-Z) alphabetical order
- [PFPCS-2814] - Incorrect currency conversion for historical data if getting from ccyDS
- [PFPCS-2824] - Only displays defaultCurrency when adding config dm_query_currency and ppExchangeRate
- [PFPCS-2834] - Display wrong currency in email templates
- [PFPCS-2839] - CPQ - Customer Assigned PL priority not considered when performing Product Price Lookup
- [PFPCS-2840] - QC- Incorrect product historical data values for the selected products
- [PFPCS-2841] - Incorrect conversion if ppExchangeRateValidFrom > current date
- [PFPCS-2843] - Incorrect conversion for List Price if not found exchange rates
- [PFPCS-2868] - Incorrect conversion if setting different validTo for same currency

CPQ Package 1.2.1

Task

- [PFPCS-2581] - Introduce automatic package version updating

CPQ Package 1.2.0

- Bugs
- Improvements
- New Features

Bugs

- [PFPCS-1866] - Available for use table "Customer" or "Product" in configuration
- [PFPCS-1867] - Warning "Configurator parameter not found" is shown for first added products
- [PFPCS-1868] - Missing Deal Score output value
- [PFPCS-1952] - CPQ Input Wizard - The previous filled fields is disappeared when choosing Source Type
- [PFPCS-1981] - Deal Score - Rounding 2 decimal points on Header level
- [PFPCS-1982] - Deal Score - Return empty if having invalid target price
- [PFPCS-2000] - Still keep old values in Source Table, Source Field when changing Source Type
- [PFPCS-2001] - CPQ Input Wizard - Sort a list in ascending (A-Z) alphabetical order
- [PFPCS-2010] - Show <empty> when using label in CPQ Input Configurations
- [PFPCS-2067] - Autodeploy failed due to incorrect publishing template structure
- [PFPCS-2082] - NPE when DM not set
- [PFPCS-2167] - Cannot invoke method startsWith() on null object
- [PFPCS-2203] - Change warning message when <name> existed in PP CPQ Inputs Config
- [PFPCS-2205] - Duplicate warning message if source field don't exist
- [PFPCS-2206] - Still keep old values in Source Type, Source Table when changing Input Type between OPTIONS and OPTION
- [PFPCS-2208] - Not work if inputting Yes/No in Customer/Product Filter in CPQ Input Configuration
- [PFPCS-2209] - Do not show drop-down list for existing Source Field when choosing Edit action
- [PFPCS-2216] - Wrong input field between CPQ Inputs Configuration PP and Inputs Wizard
- [PFPCS-2280] - Can't filter on Customer(s) field when selecting input type is OPTIONS, OPTION
- [PFPCS-2289] - Not work when configuring in Value Options Filtered By Customer/Product
- [PFPCS-2290] - Can't show the list in Quote when choosing Source Type is PP
- [PFPCS-2315] - Can't work if configuring attribute1=attribute5 in Value Options Filtered By Product
- [PFPCS-2324] - Can't work when level is Line and Source Type is P, PX
- [PFPCS-2325] - Disappeared existing Source Field when switching between Line and Header level in CPQ Inputs Wizard
- [PFPCS-2329] - Missing Sales Overview when using separate configuration for quote type

Improvements

- [PFPCS-1513] - Should notify user when not find the values filtered by quoted product/customer
- [PFPCS-1710] - Investigate where is the CPQ PP displayed within the Quote
- [PFPCS-1713] - Provide default implementation of customer/product filters
- [PFPCS-1945] - Handle case if one of Price Guidance PX is negative
- [PFPCS-1947] - Cosmetic changes for some PP labels
- [PFPCS-2204] - Set required fields in CPQ Inputs Configuration and Input Wizard
- [PFPCS-2210] - Should have drop-down list for Customer/Product Filter Operator in CPQ Input Wizard
- [PFPCS-2253] - Enhance warning message if having many non-existing source fields in CPQ Input Config
- [PFPCS-767] - CPQ Input Configuration enhancement
- [PFPCS-2084] - Support for using attribute name

New Features

- [PFPCS-2193] - Add product configurator. This required configurator-package library deployed
- [PFPCS-1667] - Multiple instances support
- [PFPCS-1716] - Add KPIs to Quote Line and Header

CPQ Package 1.1.0

Bugs

- [PFPCS-1053] - Historical period parameter is not optional
- [PFPCS-1122] - CPQ- history lookuputil: no property customerId defined.
- [PFPCS-1123] - HistoryLookupUtils - no such property dmCtx defined
- [PFPCS-1149] - Sale overview: Column Name error
- [PFPCS-1200] - Avg. Quantity Per Invoice should be empty when product don't have in DM
- [PFPCS-1202] - Error when input same competitor for both Competition Data and PX
- [PFPCS-1256] - Rebate value is not showed on CPQ
- [PFPCS-1491] - Documentation - Missing details for where to setup configuration: PP, AP
- [PFPCS-1492] - When creating a new quote, selecting a customer, and adding one new product, an error appears at the top of the screen.
- [PFPCS-1494] - SW error is displayed after new item is added to the new Quote:
- [PFPCS-1496] - Quote configurator - format of message
- [PFPCS-1497] - Quote configurator - non consistent errors
- [PFPCS-1505] - Not consistent when displaying the fields that are using ValueOptionsFilteredByCustomer and ValueOptionFilteredByProduct
- [PFPCS-1510] - CPQ - quantity is not working properly
- [PFPCS-1540] - Not possible to create Quote for USD customer and USD products
- [PFPCS-1567] - Incorrect Discount Type information on Confluence
- [PFPCS-1617] - Error (@0) null for Quote Summary
- [PFPCS-1642] - Unhandled error when using function getCurrencyExchangeRate in ConversionUtils
- [PFPCS-1656] - Remove Business Key from PX Product Cost that is used for Lightning
- [PFPCS-1015] - Sonar Bug - Code could use elvis operator
- [PFPCS-1016] - Sonar Bug - Violation in class None. The ABC score for method [defaultProductCostConfiguration] is [85.1]
- [PFPCS-1017] - Sonar Bug - Violation in class None. The ABC score for method [defaultStandardDiscountConfiguration] is [65.1]
- [PFPCS-1018] - Sonar Bug - Violation in class None. The ABC score for method [lookupProductHistory] is [87.2]
- [PFPCS-1019] - Sonar Bug - Violation in class None. The cyclomatic complexity for method [setConfiguratorParameter] is [28]
- [PFPCS-1020] - Sonar Bug - getLookupFilter
- [PFPCS-1021] - Sonar Bug - Violation in class None. The ABC score for method [defaultPriceGuidanceConfiguration] is [95.1]
- [PFPCS-1022] - Sonar Bug - Testing the negative condition first can make an if statement confusing
- [PFPCS-1023] - Sonar Bug - Violation in class None. The ABC score for method [defaultProductPriceConfiguration] is [66.4]
- [PFPCS-1024] - Sonar Bug - Violation in class None. The ABC score for method [getFolderSummary] is [61.9]
- [PFPCS-1025] - Sonar Bug - SalesOverview improvement
- [PFPCS-1026] - Sonar Bug - The [net.pricefx.server.dto.calculation.ContextParameter] import is never referenced
- [PFPCS-1027] - Sonar CodeSmell - Confusing declaration in class None. The variable 'records' is initialized to null
- [PFPCS-1028] - Sonar Code Duplication

Improvements

- [PFPCS-1148] - Change ProductCost PX table into new one "CostData"
- [PFPCS-1504] - Remove "Product" prefixes from PX names
- [PFPCS-1508] - Create cost data PX for Lightning
- [PFPCS-802] - Possibility to set default customer currency
- [PFPCS-388] - Price Guidance from Price Optimizer module
- [PFPCS-394] - Price record table mapping
- [PFPCS-723] - Top 5 Lowest competitor matrix column configuration
- [PFPCS-931] - CPQ Improvement
- [PFPCS-1111] - Enhance NumberUtils and use enhanced libs method in cpq
- [PFPCS-1248] - CPQ Accelerator - Input Configuration - Ability to apply customer/product filter when fetching values
- [PFPCS-1253] - Improve message when customer not set
- [PFPCS-1254] - Change message when currency is not defined in ccy DS
- [PFPCS-1384] - Ability to show/hide outputs in the CPQ Accelerator
- [PFPCS-1455] - CPQ Input Config. - Do not allow to Submit if set Required
- [PFPCS-1501] - Link from email notification leads to classic UI
- [PFPCS-1522] - Change "true/false" to "Yes/No" in CPQInputConfiguration P

Upgrade Notes

- [PFPCS-1504] - Remove "Product" prefixes from PX names: You need to migrate data from old PX tables to new tables:
  - ProductPriceByCustomer PriceByCustomer
  - ProductPriceByCountry PriceByCountry
  - ProductPriceByRegion PriceByRegion
  - ProductPriceBySegment PriceBySegment
  - ProductPriceCompetitors PriceCompetitors
  - ProductPriceGuidance PriceGuidance