



USER GUIDE | PUBLIC

Document Version: 1.0 – 2020-06-24

# SAP Profitability and Performance Management

# Content

- 1 SAP Profitability and Performance Management. . . . . 4**
- 1.1 Applications for Business Users. . . . . 13
  - Administration. . . . . 14
  - Modeling. . . . . 18
  - Execution. . . . . 27
  - System Reports. . . . . 49
  - Tools. . . . . 53
- 1.2 Concepts for Key Users. . . . . 58
  - Financial and Business Modeling Entities. . . . . 60
  - Function Building Blocks and Reusable Templates. . . . . 60
  - Information Models for Master Data and Lookup. . . . . 66
  - Parallelization and Partitioning. . . . . 68
  - Roles and Authorizations. . . . . 71
  - Integration with BW, BPC and Analysis for Office. . . . . 73
  - Integration with SAP ERP and SAP S/4HANA. . . . . 74
  - Integration with SAP Analytics Cloud and SAP Digital Boardroom. . . . . 75
  - Integration with SAP S/4HANA for Financial Products Subledger. . . . . 76
  - Activation of Functions, Process Templates and Environments. . . . . 78
- 1.3 Functions. . . . . 79
  - Environment. . . . . 81
  - Calculation Unit. . . . . 83
  - Description. . . . . 91
  - Allocation. . . . . 92
  - Conversion. . . . . 102
  - Derivation. . . . . 105
  - Join. . . . . 108
  - Funds Transfer Pricing. . . . . 120
  - Valuation. . . . . 132
  - Flow Modeling. . . . . 136
  - File Adapter. . . . . 196
  - Remote Function Adapter. . . . . 199
  - Calculation. . . . . 206
  - Transfer Structure. . . . . 214
  - Model BW. . . . . 217
  - Model Table. . . . . 219
  - Model RDL. . . . . 221

	Model View. . . . .	223
	Query. . . . .	225
	Writer. . . . .	230
	View. . . . .	232
	Machine Learning. . . . .	236
1.4	How-to Guide. . . . .	243

# 1 SAP Profitability and Performance Management

With SAP Profitability and Performance Management, SAP provides a new generation of integrated performance management applications that do not require their own data model but can use and reuse existing data and information models from other SAP and non-SAP applications in the cloud or on-premise.

SAP Profitability and Performance Management is built on the in-memory platform SAP HANA. Using the advanced potential of SAP HANA, SAP Profitability and Performance Management is designed for business and provides an instant insight by using a single source of truth, real-time processes, and agile financial and business modeling capabilities. Thanks to the principles of SAP Fiori user experience, it is designed to run simply and comfortably for business users.

## Implementation Considerations

SAP Profitability and Performance Management can be deployed both in the cloud and on-premise and covers various integration scenarios.

### Deployment Options



**Cloud**

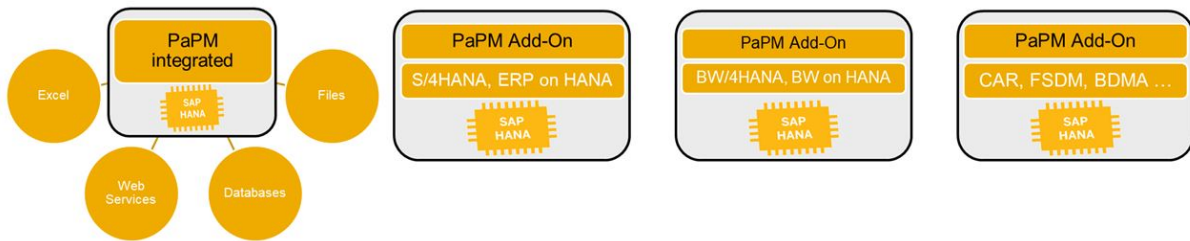


**On-Premise**

SAP Profitability and Performance Management can be used on a separate instance and can still be integrated with other SAP and non-SAP components. We recommend that you implement SAP Profitability and Performance Management as closely as possible to the source data. If other applications that contain relevant data are already installed on SAP HANA, we recommend that you use SAP Profitability and Performance Management on the same SAP HANA platform or even on the same instance to ensure optimal performance and the maximum reuse of existing data and metadata, such as hierarchies, master data, and so on.

## Integration

The business data aggregation capabilities of SAP Profitability and Performance Management (abbreviated as "PaPM" in the diagram below) enable the integration of operational systems and data warehouses at high speed with little or no data replication.



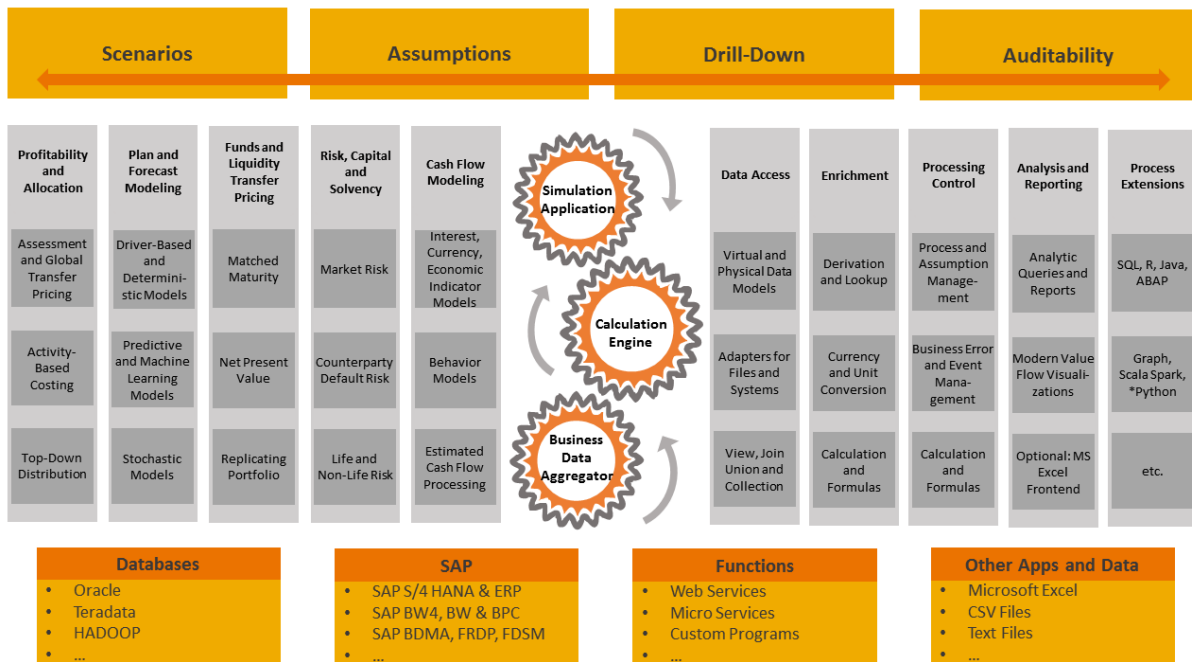
The business data aggregation capabilities of SAP Profitability and Performance Management enable the integration of operational systems and data warehouses at high speed with little or no data replication.

SAP Profitability and Performance Management uses the official application interfaces from the SAP or non-SAP applications for data read access, for example CDS views, from SAP HANA or SAP S/4HANA, open ODS views from SAP BW or BW/4HANA, calculation views from CAR, FSDM and BDMA – either locally or remotely via smart data access. If this redundancy-free approach is not feasible, SAP Profitability and Performance Management uses other official application interfaces, such as SAP BAPIs and Web services, or classic file imports of various formats.

SAP Profitability and Performance Management uses the official application interfaces from the SAP or non-SAP application for data write access, for example SAP HANA-based write interfaces like HAP to BW or BW/4HANA, SAP HANA-based PAK functions to BPC and AMDP interfaces to the Results Data component. If this redundancy-free approach is not feasible, SAP Profitability and Performance Management uses other official application interfaces, such as SAP BAPIs and Web services, or classic file exports of various formats.

## Features

The simulation application capabilities of SAP Profitability and Performance Management enable the execution of what-if scenarios for business users and the management of assumptions and drivers. Based on the granularity of the financial model, it allows drill-down from high-level to very detailed results and provides transparency by offering traceability and auditability information. In addition, it allows non-SAP and SAP BI tools, like SAP Analysis for Microsoft Office, to access the information or even trigger further calculations.



**Features Overview**

The calculation engine of SAP Profitability and Performance Management allows business users to design and execute financial and business models by configuring and combining functions across the following areas:

1. Profitability and allocation
  1. Global transfer pricing
  2. Assessments
  3. Activity-based costing
  4. Top-down distribution
2. Plan and forecast modeling
  1. Driver-based and deterministic models
  2. Predictive and machine learning models
  3. Stochastic models
3. Funds and liquidity transfer pricing
  1. Matched maturity approach
  2. Net present value approach
  3. Replication portfolio approach
  4. Further volume and account-based methods
4. Risk, capital, and solvency
  1. Market risk calculations
  2. Counterparty default risk calculations
  3. Life and non-life risk calculations
5. Cash flow modeling
  1. Interest, currency, and economic indicator models
  2. Behavior models
  3. Estimated cash flow processing
6. Data access

1. Access to local and remote virtual and physical data models
2. Adapters for files and selected systems
3. Views, joins, unions and collections
7. Enrichment
  1. Derivations and lookups
  2. Currency and unit conversions
  3. Calculations and formulas
8. Processing control
  1. Process and assumption management
  2. Business error and event management
  3. Report management
9. Analysis and reporting
  1. Analytic queries and reports
  2. Item variance and reconciliation reports
  3. Optional SAP Analysis for Microsoft Office/Excel frontend
10. Process extensions
  1. SQL, R, Graph Script, Scala Spark
  2. Java and ABAP
  3. Further custom programs via industry standard interfaces like Web services

## Main Use Cases and Sample Content

SAP Profitability and Performance Management covers the following main use cases:

### Cross Industry

1. Agile Plan and Forecast Modeling
 

In order for any company to plan or predict financial results, this Sample Content examines two opportunities, traditional or modern approach.

The traditional planning approach comprises different calculations and assumptions, from multiplication operations at the granular level to summarizing the total cost.

The modern statistical planning approach emphasizes predictive, stochastic and deep learning strategy. Combining all the previous results the system performs various additional calculations at the level of different functional areas.
2. Direct Tax Calculation
 

Sample Content for Direct Tax Calculation covers tax reporting and planning with aim to optimize calculation of current and deferred tax. It helps an entity to recognize the current and future tax consequences of transactions and other events that have been recognized in the financial statements.
3. IT Cost Management
 

IT Cost Management Sample Content provides insight into the sophisticated and flexible ways business users can gain IT cost transparency in order to perform successful financial management in IT. We start from the lower end, which comprises cost pools such as internal labor, hardware, and external labor. In the final allocation step, costs are allocated from services to the business units.
4. Process Mining on SAP S/4HANA
 

Process Mining extracts information from SAP ERP event logs to gain a detailed understanding of workflows in an organization, which can then be used to build a solid foundation for process improvements.

This Sample Content covers end-to-end examples applicable to processes common to different industries with a focus on KPIs like throughput times, level of automation, compliance to standard processes, etc. It is split into the following documents:

- Process Mining on SAP S/4HANA – Order to Cash
  - Process Mining on SAP S/4HANA – Accounts Payable
  - Process Mining on SAP S/4HANA – Accounts Receivable
  - Process Mining on SAP S/4HANA – Purchase to Pay
  - Process Mining on SAP S/4HANA – Production Planning
  - Process Mining on SAP S/4HANA – Warehouse Management
5. Product and Service Costing (with Operational Transfer Pricing)  
This Sample Content covers intercompany recharging process, markup and VAT calculation, tracing Global Transfer Pricing during the production process and applying step-based calculations with markup adjustments. It uses actual and planning data, comprising both top-down allocation approach, for revenue and cost allocation to products and services level, and bottom-up approach for examining manufacturing process that uses a bill of materials to calculate unit costs. After the fiscal year ends, tax compliance results are used to calculate credit and debit note adjustments in generating P&L for tax purposes.
  6. Profitability and Cost Management  
Business users have the ability to manage and analyze enterprise profitability and cost in one central solution. This completes the processing and provides the detailed profitability results from activity-based costing model. It can easily be customized to other standards and client-specific requirements.
  7. Simple Cost Allocation Management  
Profitability optimization with granular revenue and cost information at a product or customer level, using the standardized approach with SAP S/4HANA transaction financial data (S/4 ACDOCA table). You can dynamically change allocation drivers to adjust the logic of distributing revenues and costs from General Ledger data to further business dimensions.
  8. Value Chain Sustainability Management  
This sample content describes a project accelerator, helpful ideas and best practices for modeling an end-to-end Value Chain Sustainability calculation model that is used to calculate and simulate the ecological footprint for a plan production accounting use case.  
The value chain sustainability process comprises a bottom-up and top-down calculation of costs, profitability and transfer prices as well as carbon emissions, energy consumption and water consumption. Finally, also average company wages are set into the relation of living wages.

## Industry-specific

### Consumer Industries

1. Consumer Products Profitability and Cost Management  
Consumer Products Sample Content covers one fictive company's actual General Ledger data combined with planned (forecasted) data on a quarterly basis in the fiscal year from Consumer Industry that focuses on producing fast-moving consumer goods with beverages and non-food products and profitability of production process.
2. Fashion Profitability and Cost Management  
Fashion Sample Content leverages the profitability of a company that does business in the clothing industry. Allocations to products, channel, store and customer level show profitability on the lowest granularity level with actual and forecasted data and concrete business drivers, depending on collections, age or gender.
3. Life Sciences Profitability and Cost Management



The life sciences sector and its precursor, the pharmaceutical industry, has a long and rich history. Life Sciences industry-specific dimensions emphasize the importance of legal aspects and quality control resources and activities for pharmaceutical products, medical devices and consumer products.

#### 4. Retail Profitability and Cost Management

Retail industry-specific dimensions emphasize the omni-channel approach, bringing detailed information about whether the sales channel is a store (discount stores, hypermarkets or supermarkets) or online channel (web store or marketplace). Users from the retail industry can apply multiple rule segments to allocate to article-level directly or partially via channels, whether that is the store or online channel, down to the customer level. A unique feature of this Sample Content is a supply chain management perspective, where the retailer can examine the purchasing costs per supplier based on current store inventory levels.

### Discrete Industries

#### 1. Aerospace and Defense Profitability and Cost Management

Delivering up to half of overall revenue, data-related services will help A&D companies reduce time to market for innovations without compromising the financial or operational safety while simulating innovation scenarios along a complex value chain.

#### 2. Automotive Profitability and Cost Management

Automotive Profitability and Cost Management Sample Content focuses on production process of parts sold both individually and used in autos, motorcycles and electric vehicles, as well as providing financial services and fleet management services. Data is split into actuals and planned data with easily customizable drivers. Simulation process provides insight into how business drivers influence the expected outcome of a company from automotive industry.

#### 3. High Tech Profitability and Cost Management

High tech industries play an important role in the modern economy because technological innovation is emphasized in both economic and industrial sectors. The Sample Content considers the importance of research and development processes, comprising resources and activities for high tech products, where the user can simulate business decisions. There has been a shift in high tech employment from manufacturing to services in recent years.

#### 4. Industrial Machinery and Components Profitability and Cost Management

The Sample Content for Industrial Machinery and Components focuses on the production process of heavy industrial machines, with the implementation of activity-based costing. Allocation of costs and revenues to resources, activities, products, projects, and customers is shown, with a market-applicable what-if scenario analysis.

### Energy and Natural Resources

#### 1. Chemicals Profitability and Cost Management

Users from the chemical industry can apply multiple rule segments to allocate to the customer level directly and partially via channels, down to the customer level. Chemical industry-specific dimensions emphasize the importance of legal aspects and quality control resources and activities for chemical products.

#### 2. Mill Products Profitability and Cost Management

Mill products companies must deliver profitable growth without over-exploiting the environment by running collaborative supply chains, optimizing manufacturing, and offering best-in-class customer experiences. With respect to total costs, a company from this industry can plan future revenues and find out the best strategy to win on any market, selling various products, such as paper, plastics, recyclables, etc.

#### 3. Mining Profitability and Cost Management

The Sample Content for companies from the mining industry provides deep insights on granular revenue and cost information at the product, channel or customer level to optimize profitability and minimize costs.

Mining industry-specific dimensions capitalize on the importance of HR, CSR and Rehabilitation Fund taking consideration resources and activities.

4. Oil and Gas Profitability and Cost Management

The Oil and Gas Sample Content gives an overview of profitability for both segments in one Oil and Gas Company - Upstream that focuses on extraction of crude oil and natural gas, and on the refining, trading and distribution in downstream business units, using standard activity-based costing methodology.

5. Utilities Profitability and Cost Management

Main benefits gained from this Sample Content are coming from activity-based costing along with user-defined what-if simulation and the impact of different parameters (electricity, salaries, research and development). Main business drivers on the production process in companies providing different types of services from the utility industry, including electricity, natural energy, water and wastewater.

### Financial Services Industries

1. Banking Profitability and Cost Management

The Banking Profitability and Cost Management Sample Content covers an end-to-end example of an activity-based Costing Model applicable for both actual and planning data. The goal of this model is to optimize profitability while minimizing costs such as compensations, rent or IT cost, with banking-specific dimensions emphasizing the importance of each back office or front office activities for products and services according to the business division and region.

2. Funds and Liquidity Transfer Pricing

Funds and liquidity transfer pricing covers an end-to-end example of a matched maturity model applicable to both the assets and liabilities of banks, focusing on specific illustrated products. This is an important task for banks and aims to manage funds and liquidity in a consistent way, optimizing liquidity and minimizing related risks and costs by gaining insights into granular revenue, risk and cost information at the product, channel or customer level.

3. Insurance Profitability and Cost Management

Insurance Profitability and Cost Management is an important task for every enterprise from the insurance industry, as an end-to-end example of a Cost and Revenue Allocation Model. It provides profitability insights in the area of related products and services with the goal to optimize profitability and to minimize cost by gaining deep insights on granular revenue and cost information at line of business, department or policy level. The core of the calculation model comprises of Global Transfer Pricing between entities in an insurance group, stepladder allocation between different cost centers, and optionally advanced calculations for allocating profits to P&L dimensions.

### Public Services Industries

1. Defense and Security Profitability and Cost Management

Defense and security organizations will face greater risk, complexity, and diversity, as well as dynamic changes in economics and budgets. This shift will create network-centric operations driven by data and analytics. This Sample Content can provide data analytics technologies that extend intelligence and maximize resource utilization, with dashboards driven by analytical insights to empower executives about their budget planning, considering resources, activities and products and services at the lowest level of granularity.

2. Future Cities Profitability and Cost Management

Profitability and Performance Management with detailed financial data and business related drivers and assumptions empowers the smart city of the future to solve tough challenges faster and optimize livability and prosperity by enabling digital transformation and providing accurate insights from IoT and artificial intelligence products and services.

3. Healthcare Profitability and Cost Management

Healthcare Sample Content covers one fictive hospital's actual general ledger data combined with planned (forecasted) data on a quarterly basis in the fiscal year that focuses on providing healthcare services such as hospitalization, medical screening, laboratory analysis, surgery and pharmacy.

4. Higher Education and Research Profitability and Cost Management  
Higher Education and Research providers (universities, research centers or institutes) with SAP Profitability and Performance Management can better understand where there are genuine cost differences (for example online versus on-campus delivery, better support of regional campuses and providers with academically disadvantaged students). Activity-based costing allocations allow greater transparency of teaching and research spending (including verification of appropriate use of funding).
5. Public Sector Profitability and Cost Management  
Every private or public enterprise providing services wants to have granular budget information, observed as revenue and cost information at back or front office department, service or activity level. In order to examine public service profitability the best, this Sample Content gives traceability information about funding sources and cost breakdown for both operational and investment expenses.

## Services Industries

1. Airline Profitability and Cost Management  
This Sample Content provides deep insights into granular revenue and cost information, at the route level to optimize profitability and minimize costs. The airline industry is extremely sensitive to costs such as fuel, labor and maintenance costs. Attention-grabbing airline dimensions emphasize legs, routes and network-based approach instead of static route profitability.
2. Engineering, Construction and Operations Profitability and Cost Management  
Engineering, Construction and Operations industry-specific dimensions emphasize the importance of project-based planning for different types of construction, including project management and supervision of residential, commercial, industrial or civil (heavy) construction and engineering.
3. Media Profitability and Cost Management  
Technology will continue enabling content creation, content consumption and monetization, as well as targeted audience engagement and exceptional customer experiences, in media and entertainment industry. SAP Profitability and Performance Management with this Sample Content can provide clients with better plan strategies for Consent-Based Marketing, Content Monetization considering different sales channels and analyzing profitability via omni-channel Commerce Management.
4. Postal Services Profitability and Cost Management  
Every enterprise from industries in the area of postal or package delivery industry and related services in the whole supply chain for logistics, e-commerce or freight transport can optimize profitability and minimize costs by gaining deep insights on granular revenue and cost information at process, service, channel or customer level.
5. Professional Services Profitability and Cost Management  
The Professional Services Profitability and Cost Management Sample content will help in changing service delivery and workforce management to improve outcome-based business models, from consulting, analytics, business intelligence, HR or any other service providing company. The goal of this model is to optimize profitability while minimizing costs to provide successful project and managed services business, aligning financial data with business development and talent planning, etc.
6. Railways, Travel and Transportation Profitability and Cost Management  
Railways, Travel and Transportation Sample Content leverages profitability of a company that does business in travel and transportation industry. Allocations to vehicles, routes, and legs show profitability on the lowest granularity level with actual and forecasted data and concrete business drivers. Travel and Transportation specific dimensions emphasize legs, routes and network-based approach instead of static route profitability.
7. Sport and Entertainment Profitability and Cost Management

Sports club, organization, private fund or any other partner from the sport and entertainment industry can have detailed insights into profitability analysis with ticketing systems, for both actual and planning data. Entertainment parks and integrated resorts can win the innovation game by adapting to disruptive change and connecting their physical and digital operations. Success depends mainly on excellence in the area of effective operations management. This Sample Content can help the clients in cutting the cost of ticket sales and boost profits by analyzing the data of visitors, subscribers, and fans, tapping new revenue sources and streamlining financial and IT landscape.

#### 8. Telecommunication Profitability and Cost Management

Telecommunications Sample Content focuses on the profitability of a fictive company from the telecommunications industry that provides mobile, internet and TV services. Data split into actuals and planned can be customized, with allocations on service, customer and channel level. The defined process uses a parameter that shows the impact of any resource or activity driver on this specific industry profitability.

#### **i** Note

For the above use case a Sample Content is available which describes SAP best practices. The administration guide describes the installation and activation of the Sample Content.

SAP Profitability and Performance Management can also be used as a tool for other use cases which are not listed here.

## Related Information

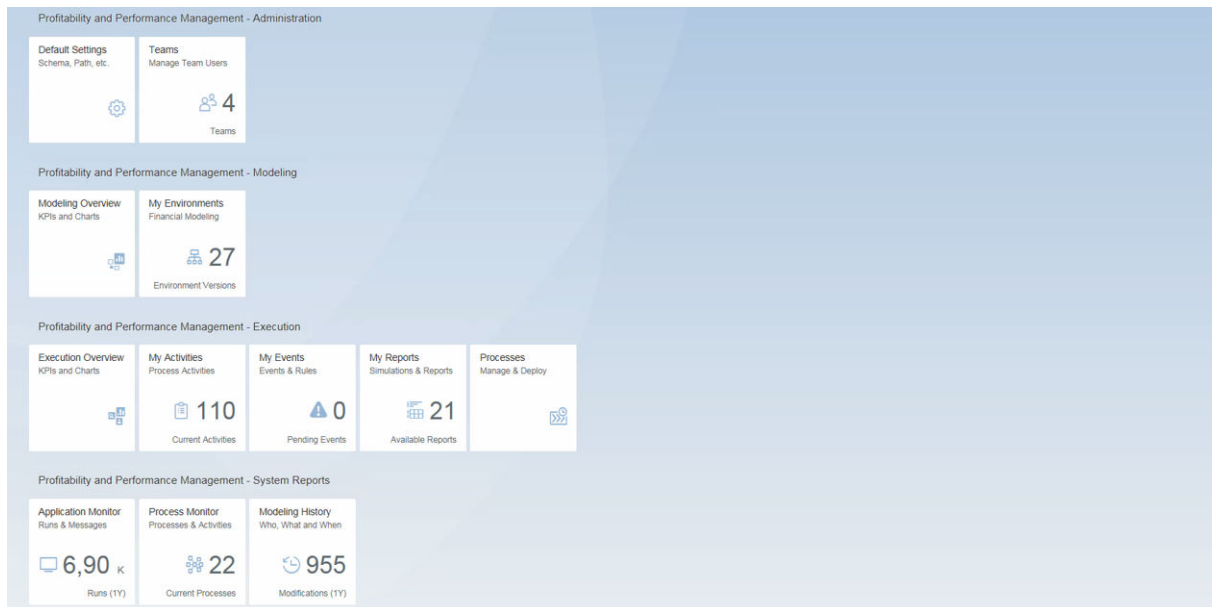
See also [Applications for Business Users \[page 13\]](#)

For more information about the available functions in SAP Profitability and Performance Management, see [Modeling Environment \[page 24\]](#).

## 1.1 Applications for Business Users

General applications are available .

The following following applications are available for business users to help them streamline their work processes:



Application Overview

### Administration

1. [Default Settings \[page 14\]](#)  
This application allows you to enter specific settings for new environments, such as schema or path..
2. [Teams \[page 16\]](#)  
This application allows you to manage the teams that are the basis for assigning activities, events and reports to specific user groups.

### Modeling

1. [Modeling Overview \[page 18\]](#)  
This application displays predefined statistics and KPIs relating to the usage of the modeling environment during design time.
2. [My Environments \[page 19\]](#)  
This application provides you with access to your modeling environments.

### Execution

1. [Execution Overview \[page 27\]](#)  
This application displays predefined statistics and KPIs relating to the usage and behavior of the execution environment during runtime.
2. [My Activities \[page 28\]](#)  
This application provides you with a central access point for your processes and activities.
3. [My Events \[page 30\]](#)

This application provides you with a central access point for your business events. You can also access any errors that may occur when processes and activities are executed. This application allows manual repairs as well as the configuration of automated situation handling rules.

4. [My Reports \[page 33\]](#)  
This application provides you with a central access point for your reports and what-if simulations.
5. [Processes \[page 47\]](#)  
This application allows key users to apply processes to user groups, including the setting of deadlines.

## System Reports

1. [Application Monitor \[page 49\]](#)  
This application displays the detailed logs of all user and batch operations.
2. [Process Monitor \[page 51\]](#)  
This application provides an overview of the currently deployed processes.
3. [Modeling History \[page 52\]](#)  
This application displays the change history of all environments and allows users to retrieve historic versions.

## Tools

1. [Activate Function \[page 53\]](#)  
This tool allows you to activate a function without having to navigate to the modeling environment.
2. [Run Function \[page 54\]](#)  
This tool provides support with running functions without having to navigate to the modeling environment.
3. [Delete Temporary Data \[page 57\]](#)  
This tool allows you to delete SAP Profitability and Performance Management Y tables, such as data produced by functions and the model table.

## Related Information

See also [Concepts for Key Users \[page 58\]](#)

For more information about the available functions in SAP Profitability and Performance Management, see [Modeling Environment \[page 24\]](#).

## 1.1.1 Administration

### 1.1.1.1 Default Settings

Default settings are applied to every new environment.

The default settings are defined in a default environment with the name "Environment Template for Default Settings" and the ID "SAP" with version "1". When a new environment is created, the default environment setting or configuration is copied to the new environment.

Typical default settings include the environment database connection. Other settings, such as fields or functions, can be defined in the default settings and are also copied to every new environment.

You can use this function if your organization has a set of formats that needs to be followed or considered. If you change or add to the default settings, your required settings and formatting standards will then automatically be applied when modelers create a new environment.

## Procedure

1. In the client where SAP Profitability and Performance Management is installed, choose [SAP Menu](#) > [Profitability and Performance Management](#) > [Administration](#) > [Manage Default Settings](#) .

### i Note

The system displays the same environment as the one defined in [SAP Menu](#) > [Profitability and Performance Management](#) > [Modeling](#) > [Start My Environments](#) > [Environment Template for Default Settings \(Environment SAP, version 1\)](#) .

2. In *Edit* mode, you can add nodes, functions, fields, and settings that you want to be considered a “template”.
3. Once you have finished configuring your template, choose *Save*.

### i Note

Everything that you add here and save will then be part of all new environments that are created in the same system.

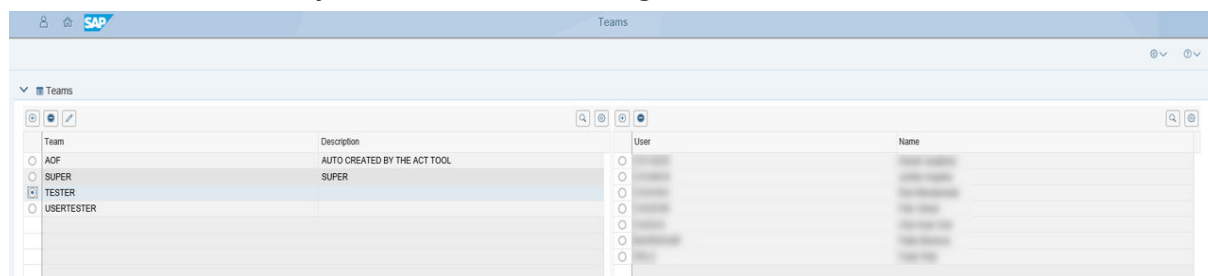
## Related Information

For more information about modeling entities, see [Financial and Business Modeling Entities \[page 60\]](#).

For more information about the available functions, see [Functions \[page 79\]](#).

## 1.1.1.2 Teams

Teams are groups of users that work together on processes, business events, and reports. Multiple teams are typically used in decentralized processes, where different activities have to be executed by different groups of users. Each user can see only those activities that are assigned to their team.



Teams and User Groups

Application managers can control the teams and the assignment of users to a team.

The following key features are available:

Key Feature	Use
Team Management	<p>Teams (user groups) can be created, edited, and removed. Teams are available across all environments and even for other applications in the same client, such as SAP Business Workflow.</p> <p>Specific users can be added to a team and removed from a team.</p> <p>Users are created, edited, and deleted centrally by SAP Net-Weaver administrators and not in this application.</p>
Assignment of Teams	<p>Teams can be assigned to activities during the deployment of processes, so that these activities can be performed by the users that belong to that team.</p>

## Procedure

- In the client where SAP Profitability and Performance Management is installed, choose ► [SAP Menu](#) ► [Profitability and Performance Management](#) ► [Administration](#) ► [Manage Teams](#) ►.
- The system opens a browser window that has two screen sections:
 

**Left section: team creation window**

  - Choose the [Add](#) button and enter a name and a description to create a team.
  - Choose the [Remove](#) button to remove or delete any existing teams. If the team is being used in workflows or processes by SAP Profitability and Performance Management, the system does not allow you to delete them.



### i Note

If the team is being used by process activities, the system displays a *Where-used* screen.

An administrator can only delete a team after removing the assignment in all process activities shown in the *Where-used* list.

3. If you want to change the description, choose the *Edit* button, and rename the description of the team.

#### **Right section: assigning users to teams window**

### i Note

Users must have already been created. This must be done by user administrators.

1. On the left-hand side of the screen, choose the team for which you want to add or remove a user.
2. Choose the *Add* button to add a user to the team that you have selected.
3. Choose the *Remove* button to remove a user from the team you have selected.

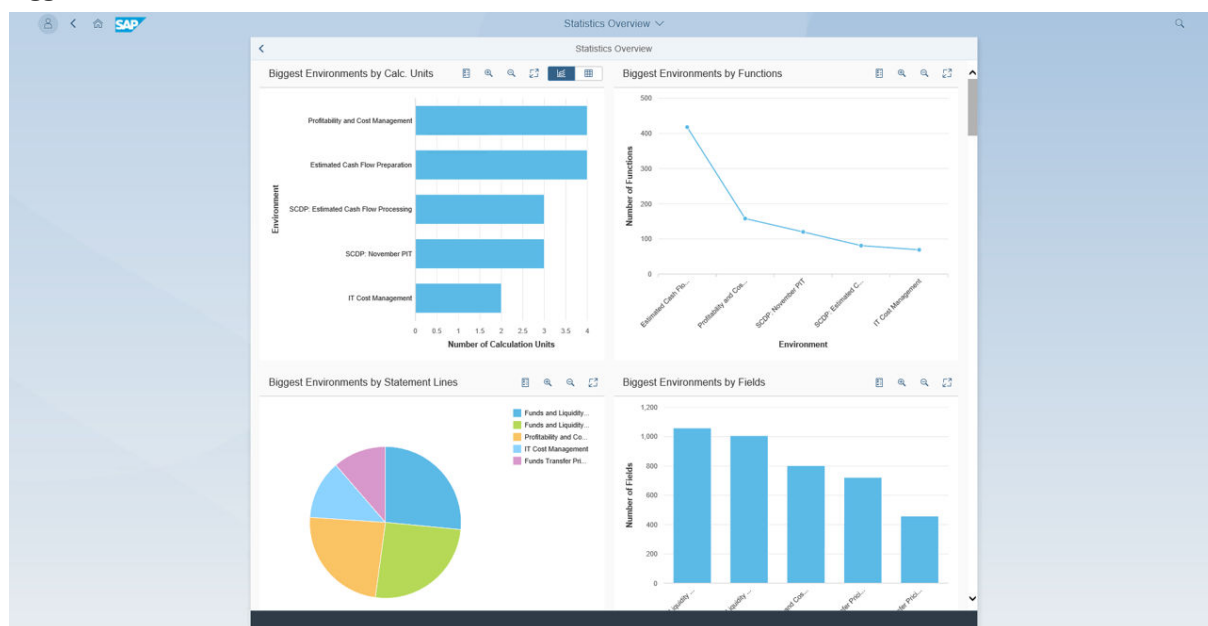
## **Related Information**

For more information about the use of teams in processes, see [Manage and Deploy Processes \[page 47\]](#).

## 1.1.2 Modeling

### 1.1.2.1 Modeling Overview

The modeling overview displays various key performance indicators that are relevant for modeling, such as biggest and smallest environments and activation times.



Key Performance Indicators in Modeling Overview

The following key features are available:

Key Feature	Use
Modeling Key Performance Indicators	The modeling overview provides users with an overview of all their environments, including their size, frequency of change, if function templates for reuse are available, and other useful information.
Key Performance Indicator Graphs	All key performance indicators are displayed in a graphical format.  Each graphic is interactive and the user can navigate from the elements to the corresponding environment, the application monitor, the process monitor, and the modeling history.  Graphics can also be displayed in a tabular format.

To use the full functionality of *Modeling Overview*, we highly recommend to launch SAP Profitability and Performance Management via Fiori Launchpad.

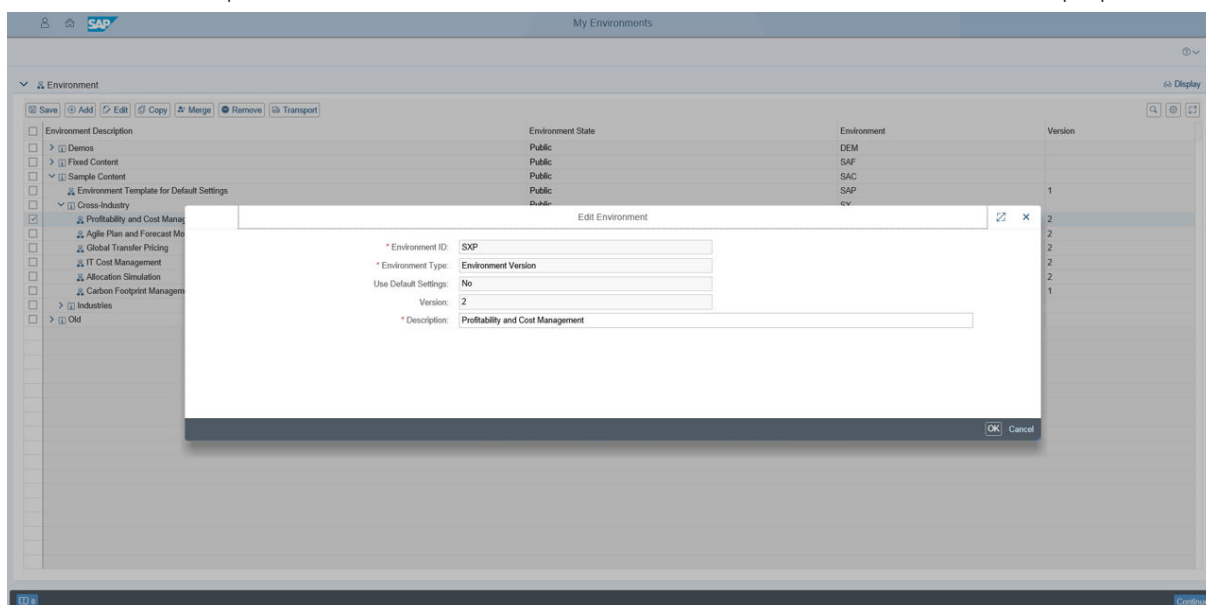
## Related Information

For more information about financial and business modeling, see [Financial and Business Modeling Entities \[page 60\]](#).

For more information about the modeling environment, see [Modeling Environment \[page 24\]](#).

### 1.1.2.2 My Environments

You can define multiple environments, and can use nodes to structure environments for different purposes.



My Environments

An environment is a versioned group of shared metadata, functions, and information that comprises a financial and business model. It can be managed in the system landscape without affecting other environments.

The following key features are available:

Key Feature	Use
Environment Management	Environments and their versions can be added, edited, copied, merged, removed, and transported.  Changes made in an environment are not only saved but also historized with information about who made the change and when. This information is available in the <i>Modeling History</i> application. For more information see <a href="#">Modeling History [page 52]</a> .

Key Feature	Use
Nodes Management	<p>Nodes can be used to structure multiple environments in a hierarchy and, like directories, can contain other nodes.</p> <p>Nodes can be created, edited, removed, and transported but, unlike environments, do not have a version.</p>
Authorizations	<p>Authorizations can be attached to environments and nodes so that they can be viewed or edited only by selected modeling users. While SAP Profitability and Performance Management provides a dedicated application for team management, authorizations are managed centrally by the SAP system administrator.</p>
Modeling Environment	<p>When you select an environment, the system opens the modeling environment application where you can use the <a href="#">Continue</a> button to maintain the environment.</p>

## Procedure

In the client where SAP Profitability and Performance Management is installed, choose [SAP Menu](#) [Profitability and Performance Management](#) [Modeling](#) [Start My Environments](#). The system opens a browser window where you can process the following activities in *Edit* mode.

### Add

You can use the [Add](#) button to create a new node or environment:

1. Choose an entry from [My Environment](#) to be the starting point of the node or environment that you are creating.

#### i Note

If you don't choose a starting point, the environment or node you create will be added at the first level in the structure.

2. Choose [Add \(+\)](#) to open the [Add Environment](#) window.
3. Make entries in all of the fields as follows:
  1. **Add Level**

You can choose one of the following options:

    - "Same Level": This option means that the environment or node that you create is structured or created at the same level as the selected entry. If you have not selected an entry before choosing [Add \(+\)](#), the system automatically creates the environment or node at the highest level of the structure.
    - "One Level Below": You can only use this option if you have selected a node before choosing [Add \(+\)](#). You can only add nodes or environments to nodes or directories.

If you selected an environment before choosing [Add \(+\)](#), the system automatically sets the [Add Level](#) field to "Same Level".

## 2. Environment ID

This is a 3-digit alphanumeric ID that is permanently assigned to the node or environment. Once it has been created, it is not possible to edit the environment ID.

### **i** Note

In SAP Profitability and Performance Management, do not use an environment ID that starts with the letter "S" (for example "SEN"). This letter is reserved exclusively for use in the default template environment and for sample content included in SAP Profitability and Performance Management releases and support packages.

## 3. Environment Type

You can choose one of the following options:

- "Environment Version": You can use this option if you intend to create an environment where configuration can take place
- "Node": This option is used to structure the environments and entries on the *My Environment* screen.

## 4. Use Default Settings

You can choose one of the following options:

- "Yes": If you select this option, you can reuse and adapt the configuration made in the *Default Settings Environment* (SAP version 1)
- "No": If you select this option, the system creates an environment that is completely blank and you need to adjust the full environment, including the assignment of the database connection.

## 5. Version

This is only a required field for environments (not for nodes). The version is a 4-digit alphanumeric ID that is permanently assigned to the environment. Once the system has created it, it is not possible to edit it.

## 6. Description

Enter a description here to name the environment or node. You can adjust this description later, if necessary, by choosing *Edit* on the *My Environment* screen.

4. Choose *Save* for the changes to take effect.

## Edit

You can use the *Edit* button to change the description of the node or environment:

1. Select the node or environment that you want to edit.
2. Choose *Edit*.
3. A popup window appears where you can change the description.
4. Choose *OK*.
5. Choose *Save* for the changes to take effect.

## Copy

You can use the *Copy* button to copy an environment (never a node). The system automatically copies all the configuration carried out in the environment. It also activates some functions, such as *Model Table* with the option "Transport Yes" or functions that are added as activities of a process template.

Follow these steps:

1. Choose an environment that you want to copy.
2. Choose *Copy*.

3. The system displays a popup window where you need to provide the following information:

1. **Environment ID**

This is a 3-digit alphanumeric ID that is permanently assigned to the environment. Once the system has created it, you can no longer edit it.

**i Note**

In SAP Profitability and Performance Management, do not use an environment ID that starts with the letter "S" (for example "SEN"). This letter is reserved explicitly for use in the default template environment and for sample content included in SAP Profitability and Performance Management releases and support packages.

2. **Version**

The version is a 4-digit alphanumeric ID that is permanently assigned to the environment. Once the system has created it, you can no longer edit it.

3. **Description**

Enter a description here to name the environment or node. You can change this later, if necessary, by choosing *Edit* on the *My Environment* screen.

4. Choose *OK*.

5. Choose *Save* for the changes to take effect.

## Merge

You can use the *Merge* button to merge the configuration of one environment with the configuration of another environment (never a node).

The merge function not only adds or merges functions from one environment to another, it also merges functions, formulas, and rule types for example.

**❁ Example**

If a function called "ZPER" exists in both Environment 1 and in Environment 2, all configuration settings for both ZPER functions are merged, including the formula and rules used.

Follow the steps below:

1. Choose an environment. This is the environment that will be merged with the other environment, meaning all configuration settings from this environment will be transferred to the other environment.
2. Choose *Merge*.
3. The system displays a popup window. Select the environment that you want to merge the environment from step 1 with.
4. Choose *OK*.
5. The system copies or merges all functions and rules from the environment of step 1 to the environment from step 3.
6. Choose *Save* for the changes to take effect.

## Remove

You can use the *Remove* button to remove an environment and/or node, and then transport this cleanup process or deletion to other systems using a transport request.

Follow the steps below:

1. Choose the environment or node that you want to delete.

2. Choose *Remove*.
3. The system displays a popup window where you can enter the following information:
  1. Comment section  
You can use this optional field to document why deletion is necessary, for example.
  2. Transport request selection
    - “Available Transport Request”: Displays all Customizing requests assigned to the user deleting an environment.
    - “Create New Transport Request”: Creates a new Customizing transport for the user deleting an environment.
    - “No Transport”: Does not save the changes to a transport. This means the same environment will not automatically be deleted in the next system.
4. Choose *Confirm* to automatically remove the environment or node (plus everything underneath it). In this case, you do not need to choose *Save* for the changes to take effect.

## Transport

To transport an environment from one client to another, or from one system to another, SAP Profitability and Performance Management uses the standard transport management provided by SAP.

An environment that needs to be transported is saved to a Customizing transport that can then be imported from one system to another. This means that all configuration settings in the environment are copied automatically to another system by means of TMS (Transport Management System).

To transport an environment, follow the steps below:

1. Select an environment or set of environments that need to be transported.

### i Note

Do not select the node since it will automatically be included in the transport. Select only the environment when you create a transport.

2. Choose the *Transport* button.
3. A popup window with the following options appears:
  - “Available Transport Request”: This displays all Customizing requests assigned to the user.
  - “Create New Transport Request”: This option creates a new Customizing transport for the user.
4. Choose *OK* to automatically add the environment settings and configuration settings to the transport. This can then be imported to the next system once it is released.

### i Note

- For the development system (DEV), the settings should not cause a conflict with the number range for the quality system (QA) and the productive system (PRD).

### Example

System	From	To
DEV	000000	199999
QA	200000	399999

System	From	To
PROD	400000	599999

- The RFC connection user must be assigned the authorization `/NXI/P1_MODELING_USER_ALL` since it will generate the procedure during transport. For more information, see “Create RFC Destination” in the Administration Guide for SAP Profitability and Performance Management.

The system also activates the following functions:

- [Model Table](#) with the option “Transport Yes”, [Model View](#), [Model BW](#), [File Adapter](#) and [Model RDL](#)
- Functions that are added as activities of a process template
- Functions with the processing type “Executable”
- Functions that are a remote adapter (only the RFA types HANA R Script and HANA Stored Procedure are not activated)
- Functions that are an input of a view using an iterative type (loop).

The system also transports all the contents of the specific tables to the environments selected. To check all the tables included in the transports, follow the steps below:

1. In the SAP NetWeaver system, launch transaction code `/n/nxi/p1_mf`.
2. Choose [Function Table Info](#).

#### i Note

The [Function Table Info](#) overview screen appears. The listed table names are included once a transport is performed in the Modeling Environment.

If you are using SAP Profitability and Performance Management 3.0 SP07 or below, select [Meta Function Table Info](#).

3. The [Function Table Info](#) overview screen appears. The listed table names are included once a transport is performed in the Modeling Environment.

## Related Information

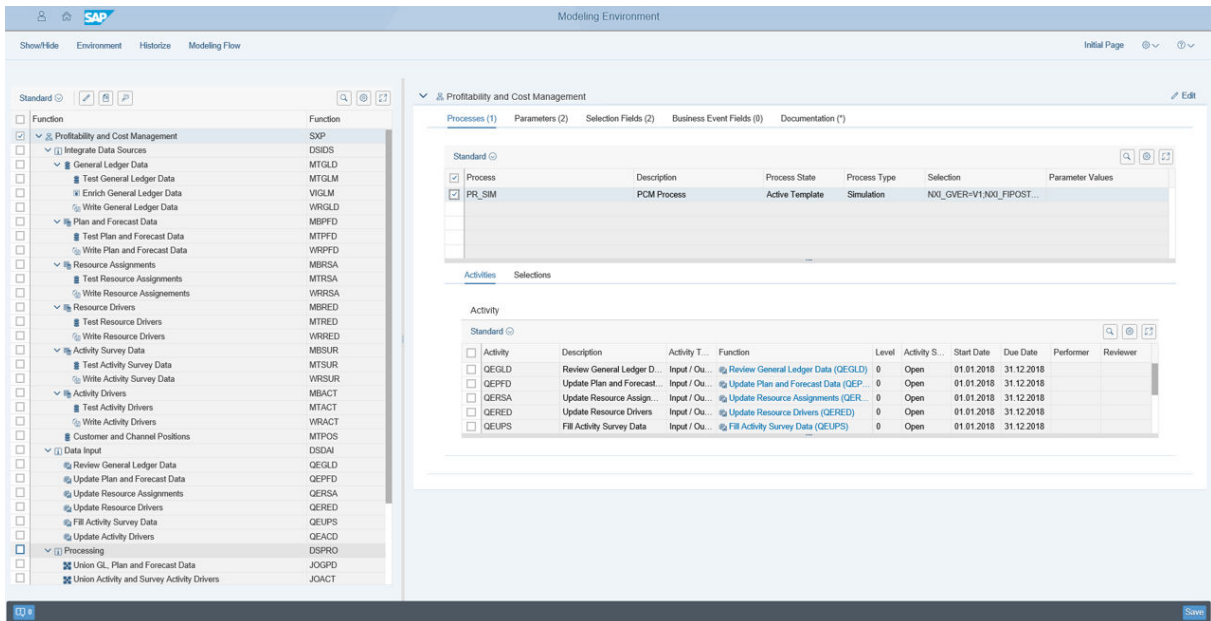
For more information about the modeling environment, see [Modeling Environment \[page 24\]](#).

### 1.1.2.2.1 Modeling Environment

The modeling environment is used by modeling users to set up and change financial and business models. This is where all model design, changes, and enhancements are made to meet the requirements of specific use



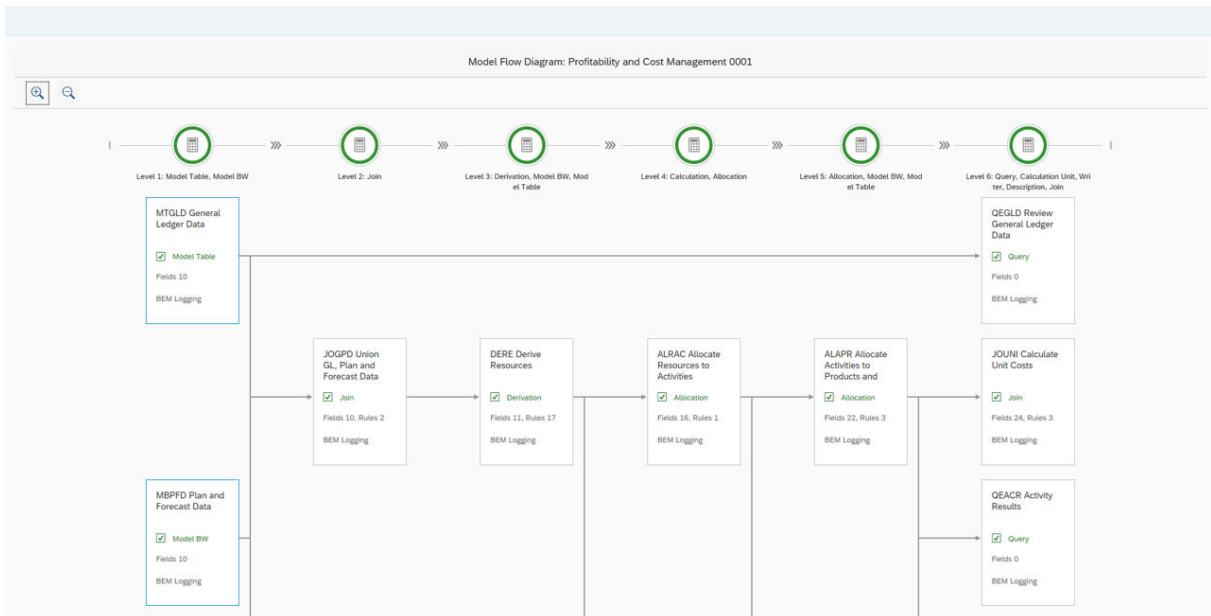
cases. A model can be set up from scratch or from a copy of one of the sample content models that is then adjusted to meet specific needs.



**Modeling Environment**

The modeling user role has the necessary authorizations to design a model and the process template activities on top, which the execution users will be allowed to run once the model and its processes are deployed.

Next to the hierarchy function display on the left, all functions and their dependencies can also be displayed in an interactive model flow diagram.



**Model Flow Diagram**

This model flow diagram shows the input-output relationships between functions and also allows various context-sensitive actions.

The following key features are available:

Key Feature	Use
Function Hierarchy	<p>The modeling environment allows the construction and maintenance of a model by adding and connecting multiple functions into a common network. The output of a function can be the input of other functions and thus contribute to the logic of the model.</p> <p>These functions can be arranged optionally in a function hierarchy, which is displayed on the left. This hierarchy has no effect on the logic of the model and simply serves for better readability.</p> <p>Functions can be added, removed, changed, and copied.</p> <p>In change mode, the <code>hierarchy</code> function is locked against changes from other users and changes are made persistent during save. Other users can see these changes once they refresh the <code>hierarchy</code> function or switch to change mode themselves.</p> <p>Where-used lists and a network diagram can visualize the logical dependencies of the output input relationships.</p>
Function Details	<p>When a function is selected in the <code>hierarchy</code> function, the function details are displayed on the right.</p> <p>Depending on the function type, certain functions are available in display mode to run a function or to analyze or show a result, for example.</p> <p>In edit mode, the function is locked against changes from other users and changes are made persistent during save. Other users can see these changes once they display the function details or switch to edit mode themselves.</p>
Environment Details	<p>The <i>Environment</i> button in the screen header opens the details of the environment.</p>
Environment Historization	<p>The <i>Historize</i> button in the screen header takes a snapshot of the current saved status of the whole environment configuration, including all field and function details, and saves this snapshot in the modeling history. We recommend you do this before you make bigger changes to an environment because it allows you to restore the snapshot later if needed.</p>

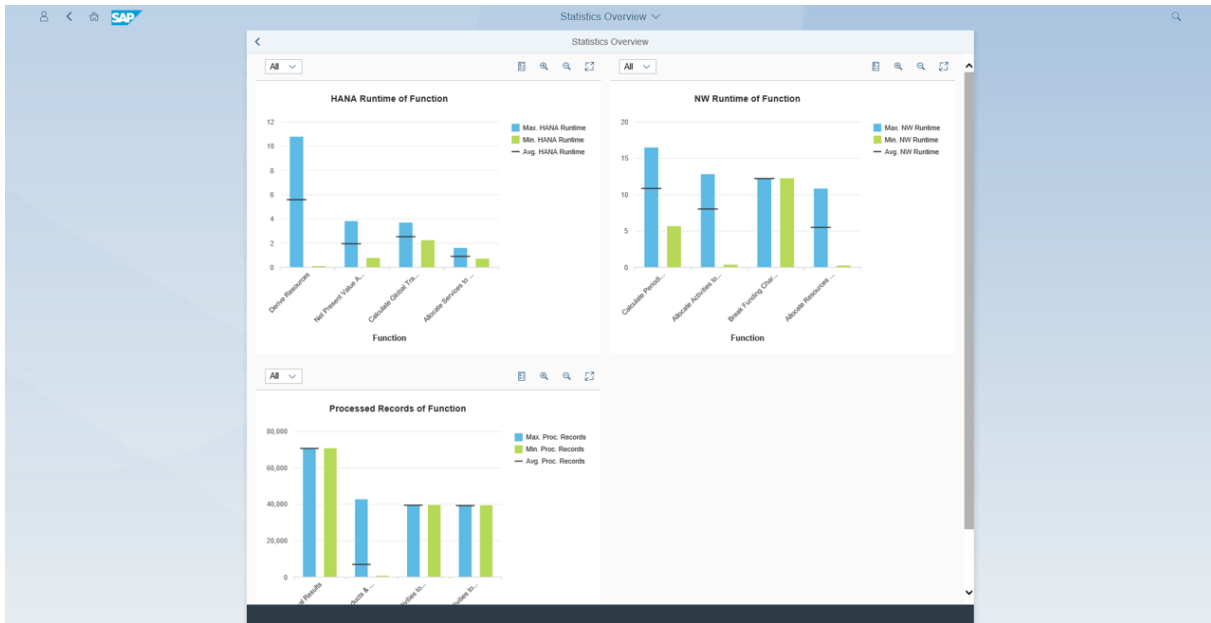
## Related Information

For more information about the available functions, see [Functions \[page 79\]](#).

## 1.1.3 Execution

### 1.1.3.1 Execution Overview

The execution overview displays various key performance indicators that are relevant for execution, such as biggest and smallest runtimes and processed data volumes.



Key Performance Indicators in Execution Overview

The following key features are available:

Key Feature	Use
Execution Key Performance Indicators	The execution overview provides users with an overview of the runtime, data volumes, and other useful information about the execution of models.
Key Performance Indicator Graphs	<p>All key performance indicators are displayed in a graphical format.</p> <p>Each graphic is interactive and the user can navigate from the elements to the corresponding environment, the application monitor, the process monitor, and the modeling history.</p> <p>Graphics can also be displayed in a tabular format.</p>

To use the full functionality of [Execution Overview](#), we highly recommend to launch SAP Profitability and Performance Management via Fiori Launchpad.

## Related Information

For more information about environments, see [Financial and Business Modeling Entities \[page 60\]](#).

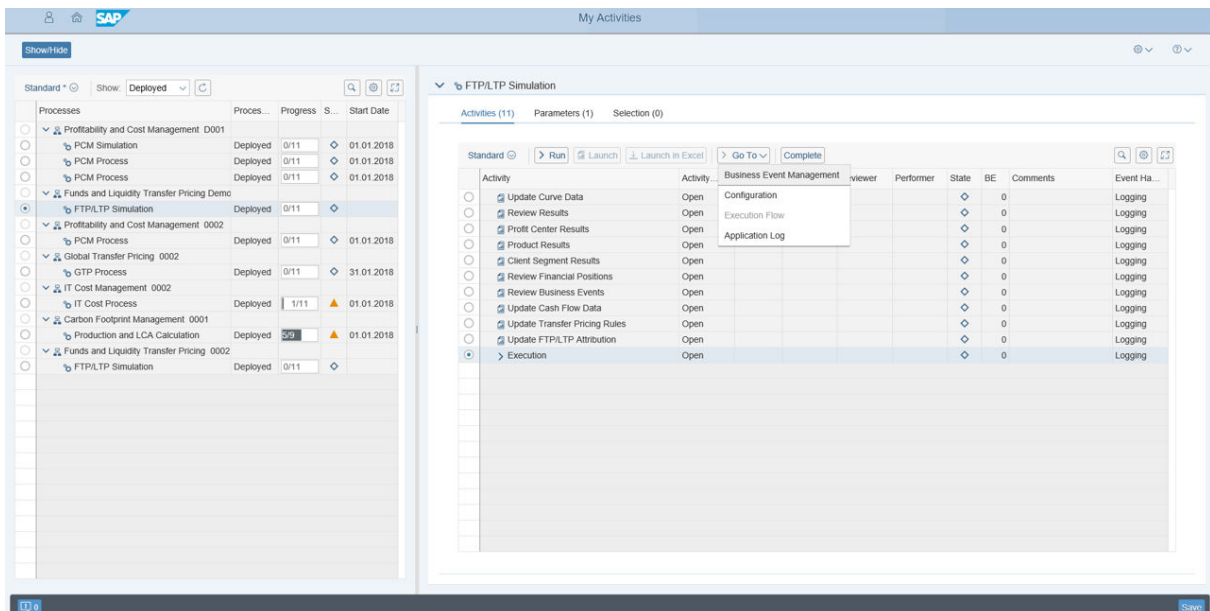
For more information about how to define processes, activities, parameters, and selection fields, see [Calculation Unit \[page 83\]](#).

For more information about how to deploy processes, see [Manage and Deploy Processes \[page 47\]](#).

For more information about how to monitor processes, see [Process Monitor \[page 51\]](#).

### 1.1.3.2 My Activities

The user can access the current activities of their team that need to be processed.



My Activities

This application allows you to execute process activities, change the *Activity State* (complete, submit, approve, reject), change parameters and selections (for simulation run type only) and change comments. Various actions are available in the *GoTo* menu: You can choose *Application Log*, *Business Event Management* or *Modeling* for the selected activity.

The application does not display process instances and their activities that are assigned to other teams and that are not relevant for the user. The system displays only deployed process instances.


#### i Note

You use the *My Activities* application for execution, and the *Manage and Deploy Processes* application for process instance management.

The following key features are available:

Key Feature	Use
Processes	<p>Processes are displayed in a hierarchy on the left together with the environment to which they belong. By default, only the current processes that need attention from the user are displayed. All processes can be displayed, including finished processes.</p> <p>A progress indicator shows how many of the activities are already finished.</p>
Activities	<p>If a process is selected, the relevant activities for the user are displayed on the right.</p> <p>Activities can require two types of attention:</p> <ol style="list-style-type: none"> <li data-bbox="804 835 1394 1003">1. Input/output This type of activity requires manual user interaction because they either display data for review or allow data input. In both cases, users launch an analytic report to access the data.</li> <li data-bbox="804 1014 1394 1137">2. Execution This type of activity triggers automatic logic and calculations. In this case, users run a function to produce interim or final results.</li> </ol> <p>By combining both types of activities, complex decentralized processes are structured that can involve multiple teams and various manual and automatic steps in parallel or in sequence, including an optional business workflow with the principle of dual control.</p>
Parameters	<p>All the parameters that are relevant for the execution of the process activities are listed here with their values.</p> <p>If the process type is "Simulation", the parameters can be changed at any time during the execution of activities. If not, they are fixed during the deployment of the process.</p>
Selections	<p>All the selections that are relevant for the execution of the process activities are listed here.</p> <p>If the process type is "Simulation", the selections can be changed at any time during the execution of activities. If not, they are fixed during the deployment of the process.</p>

The following authorization checks are implemented:

- Display  
A user without this authorization can see all the activities in *My Activities*. However, if he chooses the button , he will not be able to see the function ID (FID) in *Modeling*.

## i Note

The authorization check is performed in *Modeling*.

- Run  
A user without this authorization is not allowed to run executable activities. Those activities are displayed in the list but cannot be run.
- Launch  
A user without this authorization is not allowed to perform the actions *Launch* and *Launch in Excel* for specific activities. Those activities are displayed in the list but cannot be launched in *Analyze* screen or Excel.

## Related Information

For more information about displaying and editing data, see [Analytics Component \[page 35\]](#).

For more information about how to define processes, activities, parameters, and selection fields, see [Calculation Unit \[page 83\]](#).

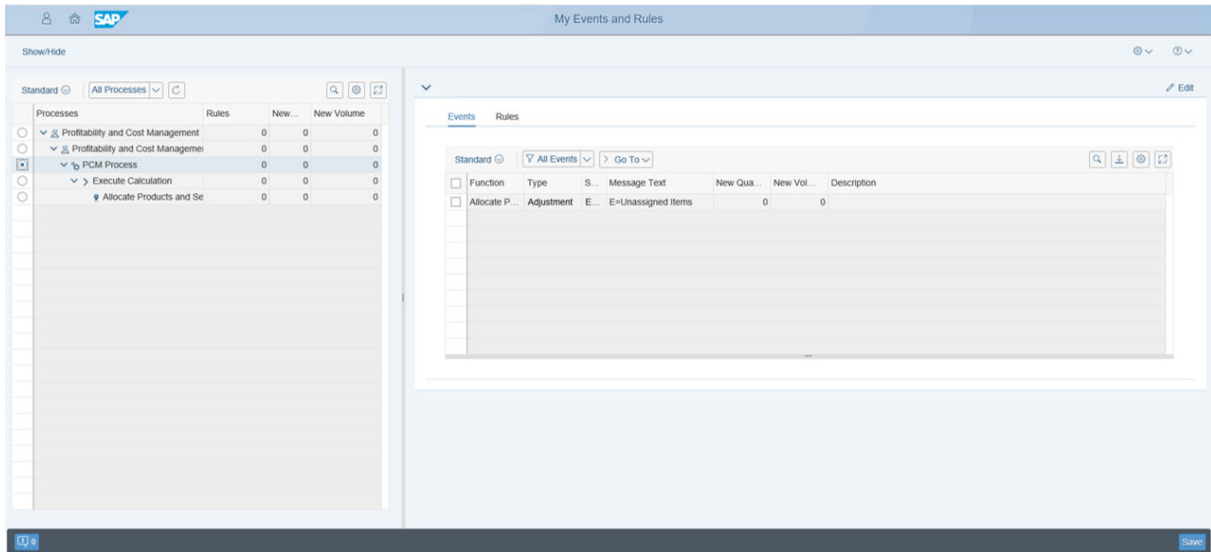
For more information about how to deploy processes, see [Manage and Deploy Processes \[page 47\]](#).

For more information about how to monitor processes, see [Process Monitor \[page 51\]](#).

### 1.1.3.3 My Events

The user can access the current exceptional business events that occurred during the execution of activities and need to be processed.

The situation handling of exceptional business events can be done manually or automatically. In the latter case, the execution user defines an automatic resolution rule that is then applied every time such a business event occurs so that no manual interaction is required.



### My Events

The application does not display other business events where the processes are assigned to other teams and are not relevant for the user.

The following key features are available:

Key Feature	Use
Processes	<p>Processes are displayed in a hierarchy on the left together with the environment to which they belong. By default, only the current processes that need attention from the user are displayed. All processes can be displayed, including finished processes.</p> <p>The user can expand the hierarchy to drill down further to see the activity and the function of the activity in which the business event occurred.</p> <p>Additional indicators show if and how many automatic rules have already been defined, how many records are affected, and what volume (the sum of the key figures of these records). Both quantity and volume give a first indication of how material the business events are.</p>

Key Feature	Use
Events	<p>The business events are displayed on the right in a list with additional information about the state of the event, the message text, the affected quantity of records, and volume.</p> <p>The user can select an event to view the detailed data and decide what steps to take to resolve the situation:</p> <ol style="list-style-type: none"> <li>1. Event The event will not be handled and left in an open status.</li> <li>2. Adjustment The user can adapt and correct the underlying data for this event and run the corresponding activity again. This step can be repeated until the situation has been resolved and the quantity and volume shows 0. Technically, the business event handling does not change the data of a data source. Instead it applies a one-time rule on the input of a function to adjust the data accordingly.</li> <li>3. Transmit The erroneous record will be moved directly to the result without adjustment.</li> <li>4. Ignore If the event is not material or otherwise important, the event can be ignored. No partial restart of an activity is necessary in this case.</li> </ol>
Rules	<p>Automatic business event rules are managed in the same way as events. The only difference is that an adjustment rule is permanent and applied automatically each time in the future when a corresponding event occurs.</p>

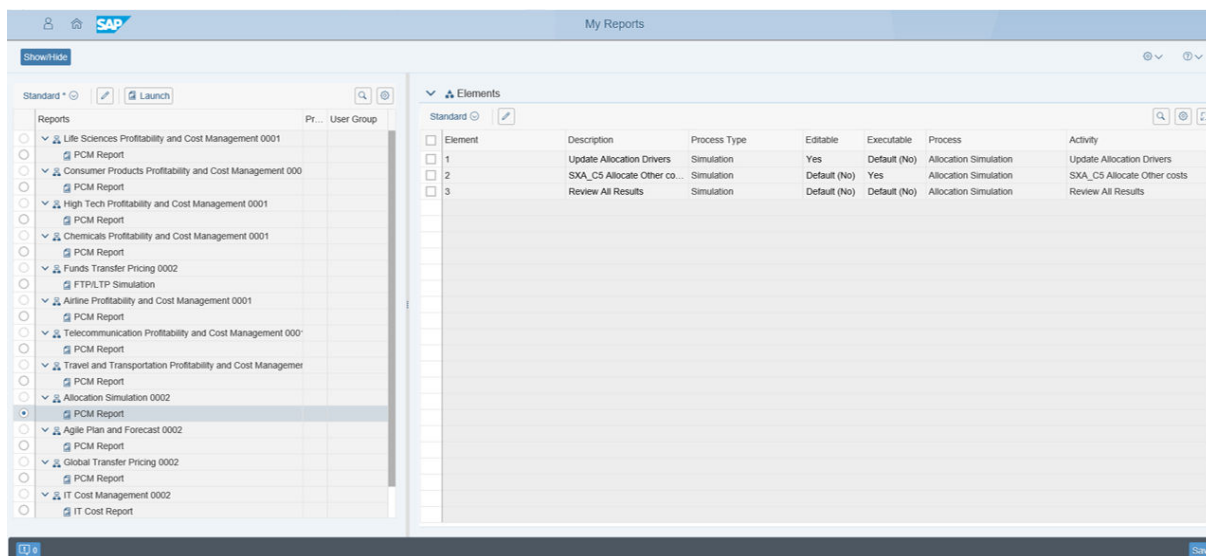
## Related Information

For more information about the definition of business event fields, see [Calculation Unit \[page 83\]](#).



## 1.1.3.4 My Reports

The user can access the current reports that are defined on top of processes and can also create new reports.



My Reports

The application does not display other business events where the processes are assigned to other teams and are not relevant for the user.

The following key features are available:

Key Feature	Use
Report Management	<p>Reports are displayed on the left in a hierarchy with the environment to which they belong.</p> <p>Reports can be either private for a user or accessible for a team.</p> <p>The main purpose of reports is to provide dynamic reports and what-if simulations that can cover multiple processes and activities in an environment.</p> <p>Users can select and launch a report, which will open the simulation and reporting application.</p>

## Key Feature

## Use

Elements of a Report

Reports consist of one or more elements, where each element refers to a process.

Reports and elements inherit all their settings from the underlying process and activities like default layouts, teams, and authorizations.

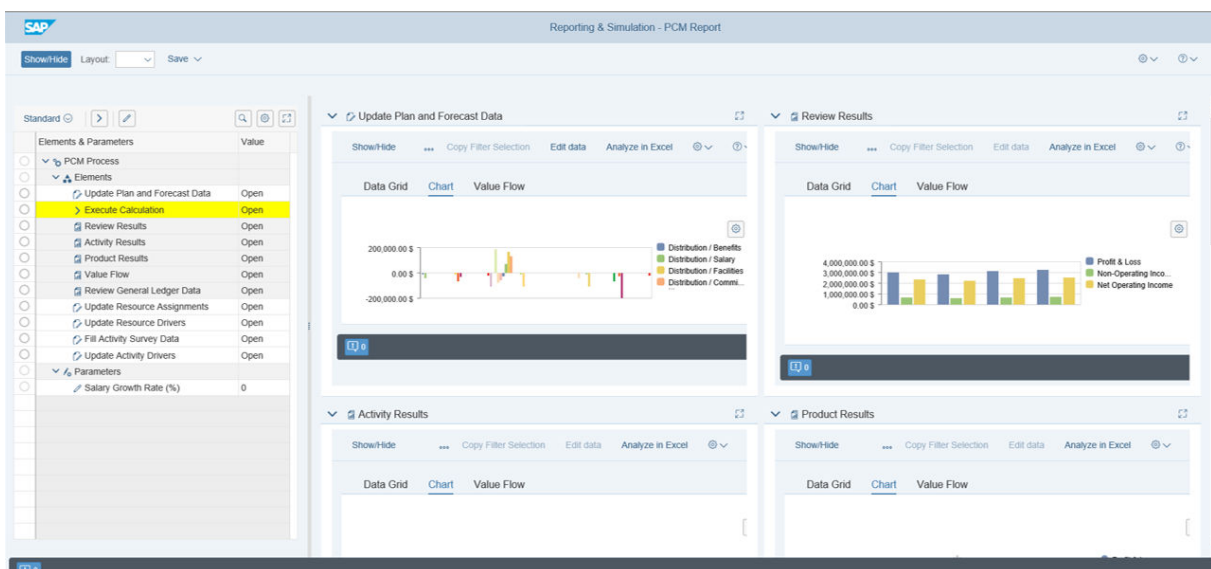
If processes are included and the process has the type "Simulation", the launched report can be used for a what-if simulation as all parameters are available for changes and activities with the type "Execution" can be triggered to run.

## Related Information

For more information, see [Simulation and Reporting \[page 34\]](#).

### 1.1.3.4.1 Simulation and Reporting

The application runs reports for execution users and gives them access to all the information for the report elements. By default, dynamic reporting capabilities are included to execute drill-downs and adapt the layouts of all the elements of the report.



Simulation and Reporting

If simulation in the underlying processes is also enabled, what-if simulation is available in the report as well.

The following table explains the key features available.

## Key Feature

## Use

Elements and Parameters

A list of all element titles and parameters available in the report is shown on the left-hand side of the screen.

If what-if simulation is enabled, parameters can be changed and the execution of activities is also possible.

Charts and Tables

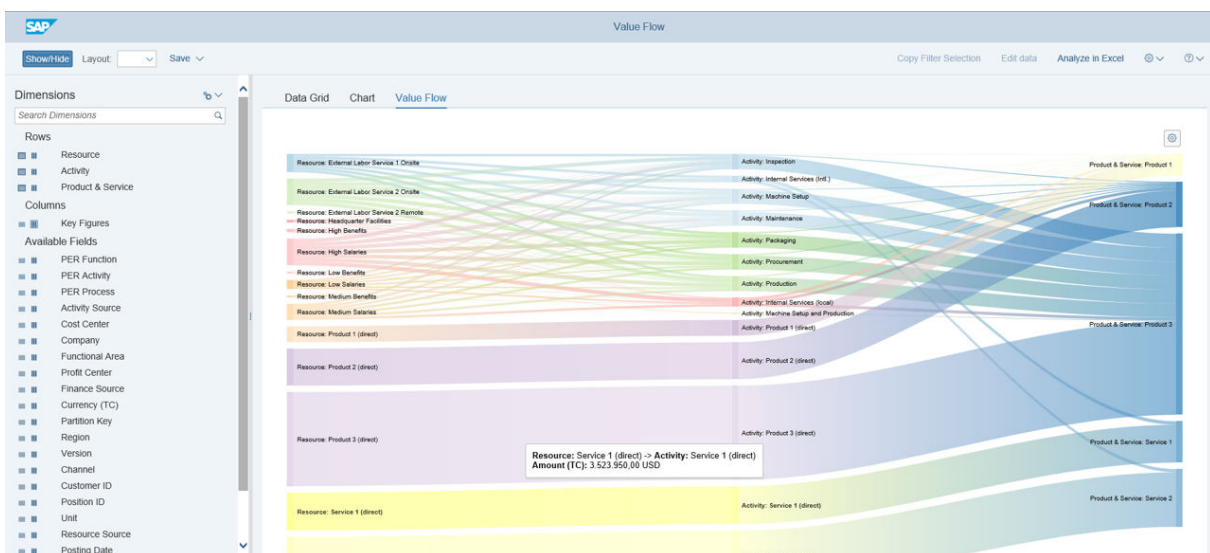
All input/output activities are visualized in chart or table format on the right-hand side of the screen. This visualization uses the standard analytics component application so that all of its features are available for each report element.

## Related Information

For more information, see [Analytics Component \[page 35\]](#).

### 1.1.3.4.1.1 Analytics Component

The analytics component is the standard application to visualize data. It allows interactive self-service reporting, where users can display data in data grids and charts.



Analytics Component

If the underlying data model and the query function enables data editing, users can also modify and input data.

Show/Hide Layout:  Save

Data Grid Chart Value Flow

Resource	Activity	Product & Service	Amount (TC)	Absolute Amount
External Labor Service 1 Onsite	Inspection	Product 1	\$ -7.454,54	\$ 7.454,54
		Product 2	\$ -14.909,08	\$ 14.909,08
		Product 3	\$ -223.636,38	\$ 223.636,38
	Internal Services (Intl.)	Service 1	\$ -126.514,26	\$ 126.514,26
		Service 2	\$ -119.485,74	\$ 119.485,74
	Machine Setup	Product 1	\$ -7.454,54	\$ 7.454,54
		Product 2	\$ -14.909,08	\$ 14.909,08
		Product 3	\$ -223.636,38	\$ 223.636,38
	Maintenance	Product 1	\$ -7.454,54	\$ 7.454,54
		Product 2	\$ -14.909,08	\$ 14.909,08
		Product 3	\$ -223.636,38	\$ 223.636,38
	Packaging	Product 1	\$ -7.454,54	\$ 7.454,54
		Product 2	\$ -14.909,08	\$ 14.909,08
		Product 3	\$ -223.636,38	\$ 223.636,38
	Procurement	Product 1	\$ -7.454,54	\$ 7.454,54
		Product 2	\$ -14.909,08	\$ 14.909,08
		Product 3	\$ -223.636,38	\$ 223.636,38
	Production	Product 1	\$ -7.454,54	\$ 7.454,54
		Product 2	\$ -14.909,08	\$ 14.909,08
		Product 3	\$ -223.636,38	\$ 223.636,38
	Inspection	Product 1	\$ -10.649,36	\$ 10.649,36
		Product 2	\$ -21.298,68	\$ 21.298,68

Data Editing using Query Function

The following table explains the key features available.

Key Feature	Use
Dimensions for Navigation	<p>The list of dimensions for navigation can be shown or hidden. The user can decide which dimensions appear on the row and column axes. The following additional options for manipulating each characteristic are available in the context menu:</p> <ul style="list-style-type: none"> <li>• Sorting</li> <li>• Filtering</li> <li>• Use of master data hierarchies</li> <li>• Display of IDs or texts for characteristic data</li> </ul>

Key Feature	Use
Data Grid	<p>The data grid can be manipulated using the options in the context menu, various toolbar buttons and the collapse/expand icons of the hierarchy nodes.</p> <p>If the underlying query is input-enabled, data can also be edited and saved.</p>
Chart	<p>The chart component provides a large selection of different and highly configurable graphs that provide visual representations of business data. The chart component also provides an out-of-the-box drill-down feature for interactive analysis.</p>
Value Flow	<p>The value flow diagram provides modern visualizations, especially to display the flow of values and money between dimensions. The value flow diagram also provides an interactive drill-down feature.</p>

## Related Information

For more information about the analytics component, see [Analytics Component](#).

### 1.1.3.5 Studio

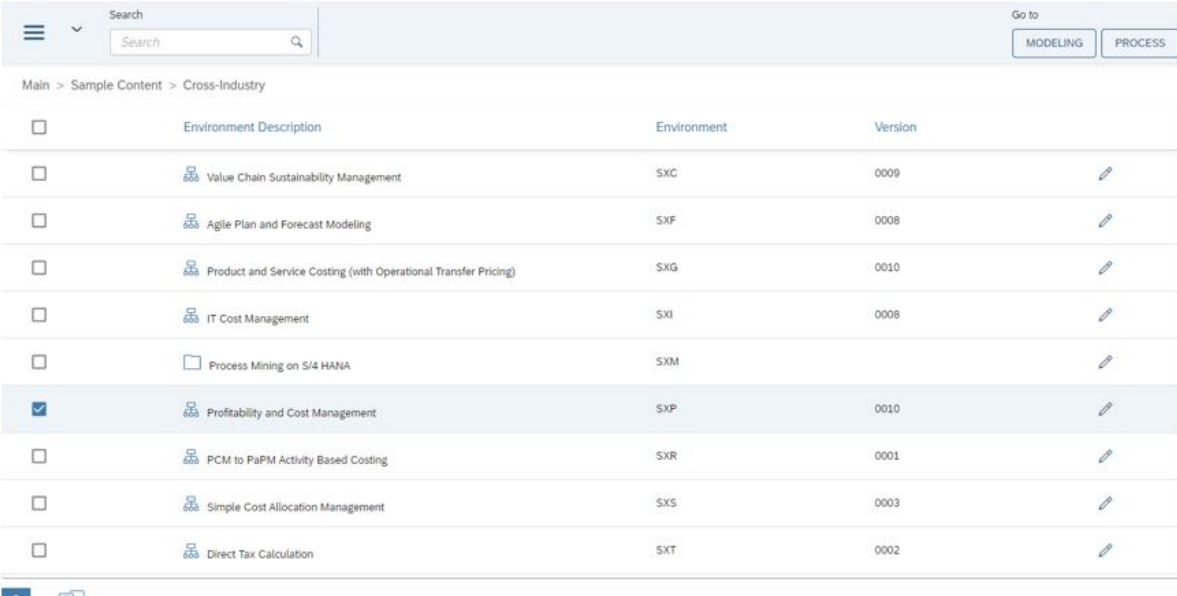
SAP Profitability and Performance Management comes with an optional, visual user interface, which can be accessed via the *Studio* tile in the SAP Fiori Launchpad.





















From SAP Profitability and Performance Management Studio, you can reach the applications for environments, modeling, process management and report management.

## 1.1.3.5.1 Environments

The *Environments* application gives access to the list of available environments. You can select an environment version to navigate to the *Modeling* and *Process Management* application.

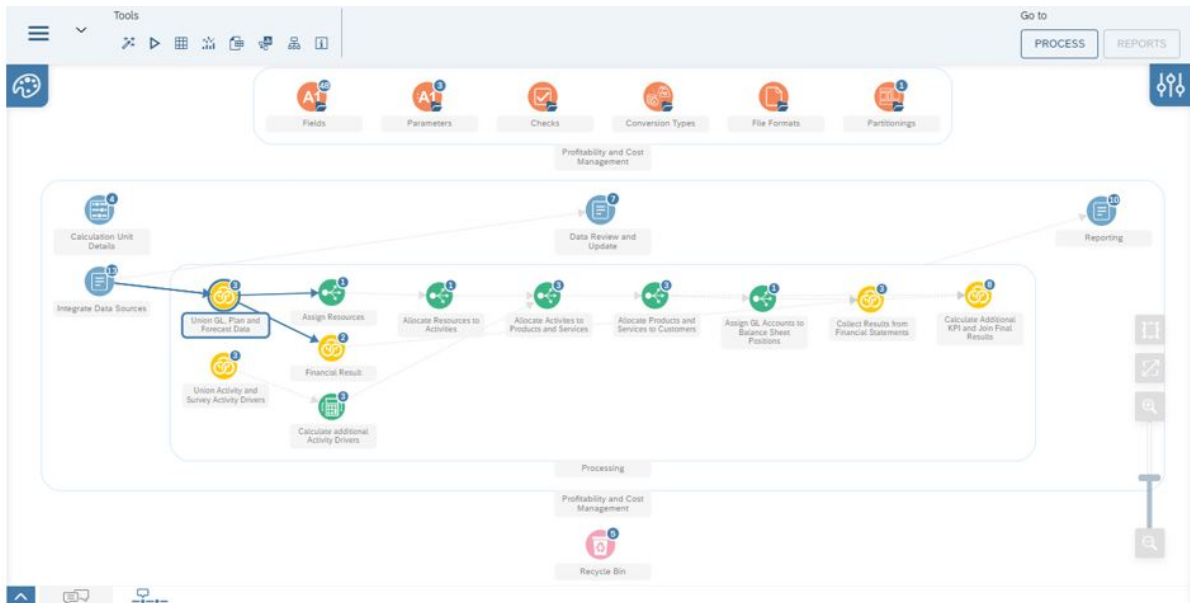


The screenshot shows the SAP Environments application interface. At the top, there is a search bar and a 'Go to' section with buttons for 'MODELING' and 'PROCESS'. Below the search bar, the breadcrumb navigation reads 'Main > Sample Content > Cross-Industry'. The main content is a table with the following columns: 'Environment Description', 'Environment', and 'Version'. The table lists several environment versions, with 'Profitability and Cost Management' (SXP, 0010) selected. Each row has a checkbox on the left and an edit icon on the right.

<input type="checkbox"/>	Environment Description	Environment	Version	
<input type="checkbox"/>	 Value Chain Sustainability Management	SXC	0009	
<input type="checkbox"/>	 Agile Plan and Forecast Modeling	SXF	0008	
<input type="checkbox"/>	 Product and Service Costing (with Operational Transfer Pricing)	SXG	0010	
<input type="checkbox"/>	 IT Cost Management	SXI	0008	
<input type="checkbox"/>	 Process Mining on S/4 HANA	SXM		
<input checked="" type="checkbox"/>	 Profitability and Cost Management	SXP	0010	
<input type="checkbox"/>	 PCM to PaPM Activity Based Costing	SXR	0001	
<input type="checkbox"/>	 Simple Cost Allocation Management	SXS	0003	
<input type="checkbox"/>	 Direct Tax Calculation	SXT	0002	

## 1.1.3.5.2 Modeling

The *Modeling* application shows a visual representation of an environment version in the form of a directed graph, where you can see the dependencies between business functions. You can see all settings of business functions here. You can also use the *Properties Panel* on the right side and trigger specific tools from the menu header.



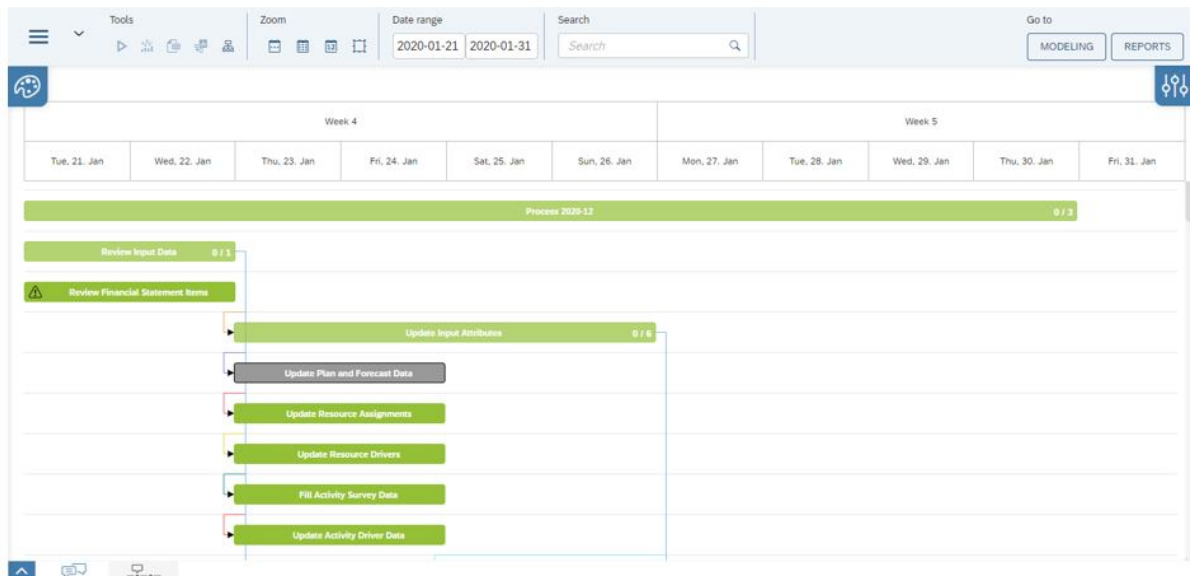
The following table explains the key features available in the *Tools* and *Go to* menu:

Key Feature	Use
Activate	Activates a selected business function, so that it is ready to run afterwards.
Run	Executes a selected business function, so that the results are available for display, analysis and visualization afterwards.
Show	Shows the results of a selected business function in a tabular form, where the data can be filtered, sorted and also technical information per record can be displayed.
Analyze	Analyzes the results of a selected business function in a pivot table or pivot chart. See also <i>Analytic Component</i> .
Analyze in Excel	If the optional SAP Analysis for Microsoft Office AddOn is installed, you can analyze the results of a selected business function in Microsoft Excel in a pivot table or pivot chart.
Visualize	Visualizes the results of a selected business function in various chart and diagram types. See also <i>Report Element / Visualize Application [page 43]</i> .
Configuration	Allows the navigation to the <i>Modeling Environment</i> , where you can edit the environment version.
Messages	Allows the navigation to the <i>Application Monitor</i> , where you can examine the messages across all environment versions.
Process	Allows the navigation to the <i>Process Management</i> application, where process instances are visualized in the form of a GANTT diagram.

Key Feature	Use
Reports	If you select a process template in ► <a href="#">Calculation Unit Details</a> ► <a href="#">Process Templates</a> ►, you can navigate to the <a href="#">Report Management</a> application, where you can define reports based on that process template.

### 1.1.3.5.3 Process Management

The [Process Management](#) application shows a visual representation of the process instances in the form of a GANTT diagram. You can inspect all settings of the process instances with their process parameters and selections here. You can also see the activities with their activity state, their assigned start and end date as well as their performer and reviewer team.



The following key features available in the [Tools](#) and [Go to](#) menu:

Key Feature	Use
Run	If the process instance is in the deployed process state, you can execute a selected business function, so that the results are available for display, analysis and visualization afterwards.
Show	If the process instance is in the deployed process state, you can show the results of a selected business function in a tabular form, where the data can be filtered, sorted and also technical information per record can be displayed.
Analyze	If the process instance is in the deployed process state, you can analyze the results of a selected business function in a pivot table or pivot chart. See also <a href="#">Analytic Component</a> .



Key Feature	Use
Analyze in Excel	If the process instance is in the deployed process state and the optional SAP Analysis for Microsoft Office AddOn is installed, you can analyze the results of a selected business function in Microsoft Excel in a pivot table or pivot chart.
Visualize	If the process instance is in the deployed process state, you can visualize the results of a selected business function in various chart and diagram types. See also <a href="#">Report Element / Visualize Application [page 43]</a> .
Configuration	Allows the navigation to the <i>Modeling Environment</i> , where you can edit the environment version.
Modeling	Allows the navigation to the <i>Modeling</i> application, where process environment versions are visualized in the form of a directed graph.
Reports	If you select a process template, you can navigate to the <i>Report Management</i> application, where the qualitative reporting and simulation is available.

## 1.1.3.5.4 Report Management

The *Report Management* application allows the creation and consumption of reports as well as the execution of simulations in case the underlying process template respectively process instance is set to process type "Simulation".

**Our Profitability and Cost Management focus**

Our Profitability and Cost Management focus is an outgrowth of our purpose to help the world run better. This holistic view requires, that our Profitability and Cost Analysis is developed within our performance management framework including Planning and Reporting. Consisting of these three core components our organization can efficiently and effectively analyse business costs, income and profitability at multiple levels to make informed business decisions.

**Profitability KPIs**

The KPI dashboard provided here presents some of the profitability KPIs such as Revenue which comprises sales revenue generated from core business, Gross Profit and Net Profit which help to analyse the profitability and financial performance of the business are also available. Owner's equity which is an indicator of how much of Asset is funded by own sources. Total Asset indicator presents overall asset of one company and it's useful to compare it with Owners' equity and different categorize of profit in order calculate different ratios.

Revenue (mil USD)	Net Profit (mil USD)	Gross Profit (mil USD)	Owner's Equity (mil USD)	Total Asset (mil USD)
18,50 m	11,23 m	14,23 m	56,80 m	108,31 m

**Product Results**

Product Result chart visualize which products and services have made impact on result individually. Further details about channels and customer for each product or service are possible to see using drill down capability. All of the results are organized based on geographic region.

**Profitability Analysis by Geographic Region**

The screenshot shows a world map with a pie chart overlaid on the Americas region, indicating profitability analysis by geographic region.

Static qualitative content like texts, images and videos can be combined with interactive visualizations, so-called report elements, based on input/output activities into a report. You can arrange multiple reports using separate tabs.

Dependent on the underlying process settings, the following edit restrictions apply:

**1. Reports on process templates**

Reports are created and prepared based on process templates by the modeling user. In this case no edit restrictions in the report apply. When a process instance is created based on a process template, these reports are made available to execution users automatically.

**2. Reports on process instances of type “Simulation”**




If the report refers to a process instance of type “Simulation”, no edit restrictions to the execution user apply.

**3. Reports on process instances of type “Run”**

If the report refers to a process instance of type “Run”, the editing of the report is not restricted, as long as the process state is “Open”. As soon as the process state is set to “Deployed”, the editing is not allowed anymore, just story filters can be changed and the drill-down as well as drill-through in report elements is allowed.

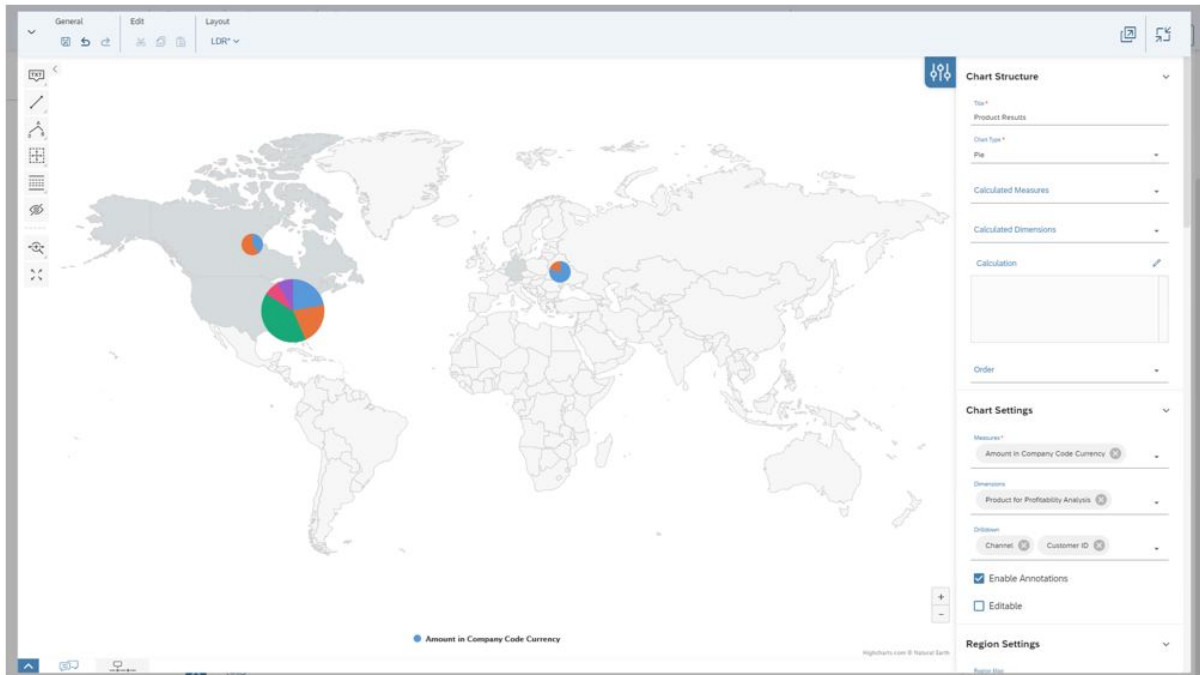
Dependent on the edit restriction, the following key features are available in the [General](#), [Edit](#), [Tools](#), [Editor](#) and [Go to](#) menu:

Key Feature	Use
Palette	Shows the execution and input/output activities of the current process.  From here you can drag the input/output activities as interactive report elements into the report to visualize them.  You can also select the execution activities from here and trigger the (re)execution of an activity.
Save	Saves the report. The layouts of any embedded report element will not be affected by this, because they have to be saved separately.
Undo	Performs an undo of the last editing operation in the report. The undo/redo-stack will be cleared on <a href="#">Save</a> .
Redo	Performs a redo of the last undone editing operation in the report. The undo/redo-stack will be cleared on <a href="#">Save</a> .
Cut	Cuts out the selected content of a report, so that it is available for paste.
Copy	Copies the selected content of a report, so that it is available for paste.
Paste	Pastes previously cut or copied content at the current cursor position into the report.
Simulate	Triggers the client-side simulation of a report, based on the maintained simulation script. After the simulation script is finished, the system updates the report elements accordingly.

Key Feature	Use
Run	Triggers the server-side (re)execution of an activity if an execution activity is selected in the <i>Palette</i> , respectively if only one execution activity is available in the process.
Configuration	Allows the navigation to the <i>Modeling Environment</i> , where you can edit the environment version.
Messages	Allows the navigation to the <i>Application Monitor</i> , where you can examine the messages across all environment versions.
Editor	In the <i>Editor</i> menu all report editing features for the static content reside like paragraph style, font family, font size, formatting, text alignment, font color, font background color, numbered list, bulleted list, todo list, block quote, decrease indent, increase indent and inserting of images, tables and media like YouTube videos.
Modeling	Allows the navigation to the <i>Modeling</i> application, where process environment versions are visualized in the form of a directed graph.
Process	Allows the navigation to the <i>Process Management</i> application, where process instances are visualized in the form of a GANTT diagram.
Properties	<p>The <i>Properties Panel</i> on the right side shows all process parameters and process selections of the underlying process template respectively process instance.</p> <p>Story filters comprise all fields, which are in use by the embedded report elements and allow to filter all report elements at the same time. The client-side simulation can contain a script, which can be executed via  <i>Tools</i> </p> <p><i>Simulate</i> .</p>

### 1.1.3.5.5 Report Element / Visualize Application

You can trigger the *Visualize* application stand-alone from *Modeling* as well as from *Process Management*. In this case the system opens it in a separate browser tab. It can also be used for embedded report elements in *Report Management*.



## Key Features

The basic set of features is in both cases identical:

Key Feature	Use
Annotations Palette	If in the <i>Properties Panel</i> the checkbox <i>Enable Annotations</i> is set for the chart, the <i>Annotations Palette</i> is available on the left side to comment and annotate the data in the chart.
Save	Saves the visualization. In case the visualization is an embedded report element, the report itself will not be affected by this, because it has to be saved separately.
Undo	Performs an undo of the last editing operation in the visualization. The undo/redo-stack will be cleared on <i>Save</i> .
Redo	Performs a redo of the last undone editing operation in the visualization. The undo/redo-stack will be cleared on <i>Save</i> .
Cut	Cuts out the selected content of a visualization, so that it is available for paste.
Copy	Copies the selected content of a visualization, so that it is available for paste.
Layout	You can manage layout variants here, where always one layout is marked as default and the others are available optionally.

Key Feature	Use
Properties Panel	<p>The properties on the right side contain a chart structure panel to customize the visualization of the chart types bar/column, stacked bar/column, stacked area, line, time series, heat map, numeric point, donut, pie, sanke, process and relationship.</p> <p>On top of the data coming from the underlying input/output activity, additional calculated measures and calculated dimensions can be defined and filled using scripted calculations. These calculated measures and dimensions are then available and included in all further settings.</p> <p>The data can be ordered by multiple fields ascending and descending if needed.</p> <p>Under <i>Chart Settings</i> you can pick the measures and dimensions relevant for visualization. In addition you can define drilldown fields, which allow an easy and interactive way to dig into details.</p> <p>For certain chart types you can enable annotations. This allows the individual commenting of data points and the drawing of lines. Especially for time series also a list of predefined financial and mathematical annotations are available like trends and regressions.</p> <p>You can switch some chart types to "editable" which allows graphical changes of data. If data on aggregated level is changed (like a bar for example), the delta change is automatically distributed to all underlying data records. These data changes cannot be saved permanently, but are available for client-side simulations.</p> <p>Under <i>Field Settings</i> you can maintain for each dimension a value selection. Based on the chart type you can also define a pattern, color and icon. For measures you can also maintain aggregations.</p> <p>Under <i>Styling</i> you can optionally set the background color and a background image, title position, axis labels and legend positions.</p>

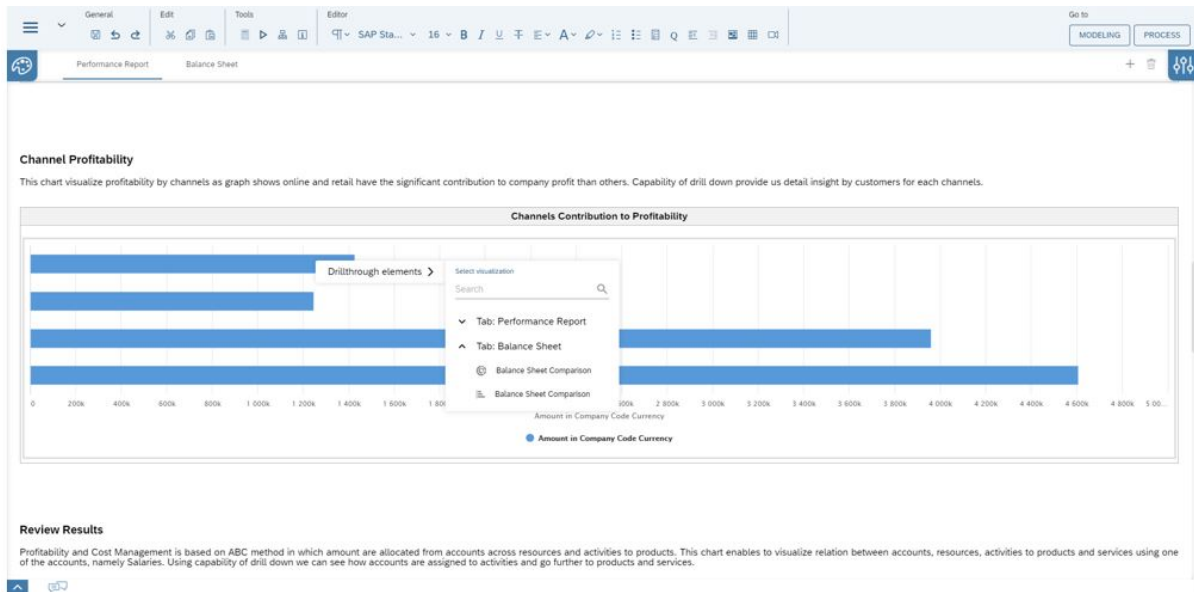
## Advanced Features

The following advanced features are only available for embedded report elements and specific chart types.

### Drill-Through

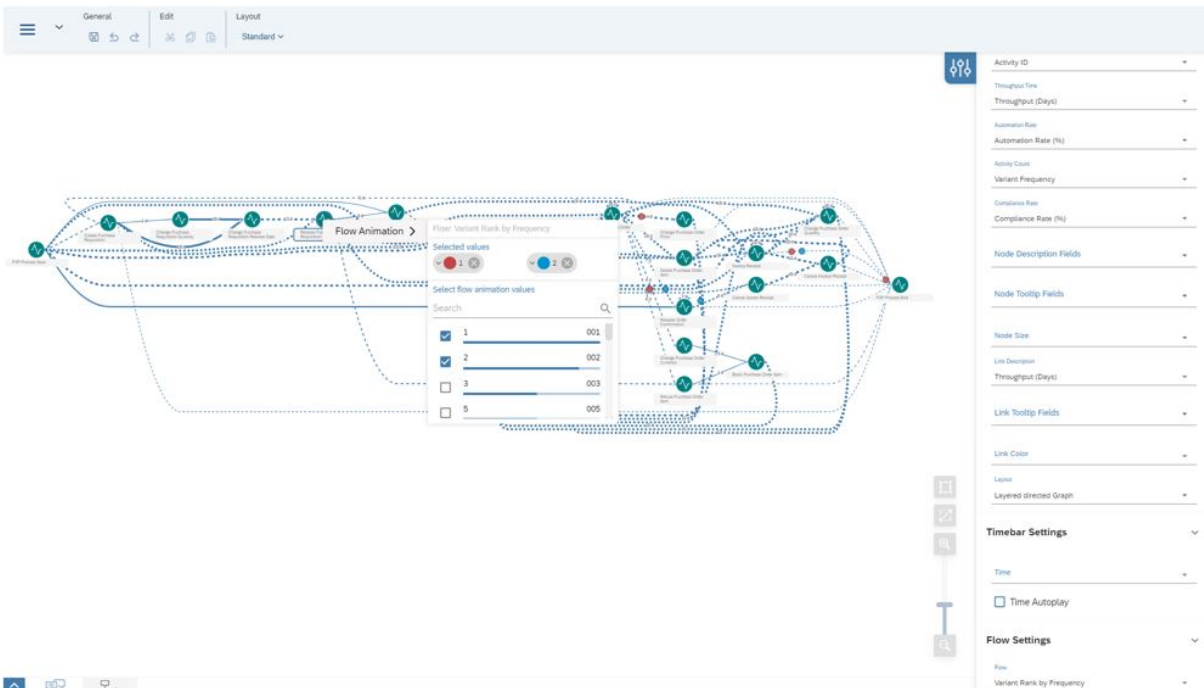
Drill-Through is available for all report elements of a report and can be reached by a right mouse click on a data point or node in the chart (on tables long touch). It offers all other report elements and their layout variants as

a target. All filters, story filters, visualization field setting filters as well as the dimension values of the selected data point or node are handed over to the target chart.



## Flow Animation

The flow settings for *Flow Animation* are available for process and relationship diagrams. If in the properties a *Flow* field is selected, its values are offered in the diagram for flow animation. It is especially helpful to animate the flow through activities in Process Mining Analysis.



## 1.1.3.6 Manage and Deploy Processes

This application allows you to manage process instances. .

Process instances are based on process templates from the modeling application. Process instance management comprises the creation and deletion of process instances as well as the changing of process states.

The default state after creation is "Open". Only processes in the state "Deployed" are visible in the *My Activities* application.

The *Manage and Deploy Processes* application allows you to change the following attributes:

- *Activity Description*
- *Start Date*
- *Due Date*
- Performer/Reviewer groups
- *Comments*.

You can also change parameters and selections. However, only if the *Run Type* is "Open" or "Suspended". One of the most important features is the change of the activity state, especially the *Reset State* pushbutton, which resets the activity state to the initial value "Open".

You cannot execute activities directly from the Manage and Deploy Processes application, but can manage the settings listed above. You can execute activities from the *My Activities* application. You can execute activities from the *My Activities* application by choosing the *My Activities* pushbutton in the header of the *Manage and Deploy Processes* application".

Activity	Start D...	Due Date	Reviewer	Performer	State	BE	Event H...	Comments
Review General Ledger Data	01.01.2...	31.12.2018			Open	0	Logging	
Update Plan and Forecast Data	01.01.2...	31.12.2018			Open	0	Logging	
Update Resource Assignments	01.01.2...	31.12.2018			Open	0	Logging	
Update Resource Drivers	01.01.2...	31.12.2018			Open	0	Logging	
Fill Activity Survey Data	01.01.2...	31.12.2018			Open	0	Logging	
Update Activity Drivers	01.01.2...	31.12.2018			Open	0	Logging	
Execute Calculation	01.01.2...	31.12.2018			Open	0	Manage...	
Review Results	01.01.2...	31.12.2018			Open	0	Logging	
Activity Results	01.01.2...	31.12.2018			Open	0	Logging	
Product Results	01.01.2...	31.12.2018			Open	0	Logging	
Value Flow	01.01.2...	31.12.2018			Open	0	Logging	

### Processes Application

The application manager runs processes and assigns activities to teams.

The following table explains the key features available.

Key Feature	Use
Processes	<p>Processes are displayed in a hierarchy together with the environment to which they belong on the left-hand side of the screen. Processes can have various states:</p> <ul style="list-style-type: none"> <li>• <i>Open</i> Open processes can be changed and settings like start dates, due dates, performer and reviewer team, parameters and selections can be maintained. "Open" can be deployed so that the execution teams can start working on the processes.</li> <li>• <i>Deployed</i> Deployed processes are visible to the execution teams who can work on the activities in the My Activities application. Deployed processes can be suspended if there are problems or completed if everything goes well.</li> <li>• <i>Suspended</i> Suspended processes are not visible to the execution team. In the same way as for open processes, changes can be applied to the settings like due dates, parameters or selections. Afterwards, the state can be set to <i>Deployed</i>, <i>Aborted</i> or <i>Completed</i>.</li> <li>• <i>Completed</i> If the activities of a deployed process are finished, the process can be set to <i>Completed</i>.</li> <li>• <i>Aborted</i> If a process needs to be terminated without success, it can be set to <i>Aborted</i>.</li> </ul>
Activities	<p>If a process is selected, the activities are displayed on the right-hand side of the screen.</p> <p>Only if the process state is <i>Open</i> or <i>Suspended</i>, can changes be applied to the activity state, start date, due date, reviewer and performer team as well as to the parameters and selections.</p> <p>The activity can have various states:</p> <ul style="list-style-type: none"> <li>• <i>Open</i> The activity is open for execution.</li> <li>• <i>Pending</i> The activity is not open for execution yet because preceding activities are not finished yet.</li> <li>• <i>In Approval</i> A dual control principle workflow is attached to the activity and this is not finished yet.</li> <li>• <i>Completed</i> The activity is completed.</li> </ul>



Key Feature	Use
Parameters	<p>All parameters that are relevant for the execution of the process activities are listed here with their values.</p> <p>Parameters can be changed only if the process state is <i>Open</i> or <i>Suspended</i>.</p>
Selections	<p>All selections that are relevant for the execution of the process activities are listed here with their values.</p> <p>The selections can be changed only if the process state is <i>Open</i> or <i>Suspended</i>.</p>

## Related Information

For more information about the application manager, see the Administration Guide for SAP Profitability and Performance Management.

For more information about the definition of processes and activities, see [Calculation Unit \[page 83\]](#).

## 1.1.4 System Reports

### 1.1.4.1 Application Monitor

The application enables the user to inspect the messages that have been logged during activations and runs for every function within an environment. This helps users to find out if warnings or errors occurred and when.

The screenshot displays the SAP Application Monitor interface. It features a top navigation bar with the SAP logo and the title 'Application Monitor'. Below this, there are two main sections: 'Execution Log' and 'Message Log'. Both sections are currently expanded to show their respective data tables.

**Execution Log Table:**

Status	Run ID	Environment	Version	Calculation Unit ID	Process	Activity	Main Function ID	Package	Business Event	Run Type	User	Main Time Sta...
✓	FA163EA271D71EE89...	SXG	2	SXG	P1	A0005	JOEXE			Run	C5270785	2018-06-04 19:22...
✓	FA163EA271D71EE89...	SXG	2	SXG	P1	A0005	JOEXE			Run	C5270785	2018-06-04 19:21...
✓	FA163EA271D71EE89...	SXG	2	SXG			QESEP			Activation of Fields	C5270785	2018-06-04 19:19...
✓	FA163EA271D71EE89...	SSA	1	SSA	P1	6	JOFIN			Run	C5270785	2018-06-04 19:09...
✓	FA163EA271D71EE89...	SSA	1	SSA	P1	6	JOFIN			Run	C5270785	2018-06-04 19:07...
✓	FA163EA271D71EE89...	SSA	1	SSA	P1	6	JOFIN			Run	C5270785	2018-06-04 19:06...
✓	FA163EA271D71EE89...	SSA	1	SSA	P1	6	JOFIN			Run	C5270785	2018-06-04 19:05...
✓	FA163EA271D71EE89...	SXG	2	SXG	P1	A0005	JOEXE			Run	C5270785	2018-06-04 18:59...

**Message Log Table:**

Status	Main Function ID	Function	Message Text	Time Stamp
✓	JOEXE	JOEXE	Execution of Function=JOEXE finished in AppServer in 3.05...	2018-06-04 19:22:46.4876680
✓	JOEXE	JOEXE	Result records written to DB in 0.092000 seconds (running 1...	2018-06-04 19:22:46.4700000
✓	JOEXE	JOEXE	Function=JOEXE Execute Preparation and GTP Calculation...	2018-06-04 19:22:46.3700000
✓	JOEXE	JOEXE	Records processed with Status OK=9490, Abort=0, Error=0...	2018-06-04 19:22:46.3700000
✓	JOEXE	JOEXE	Processing Message "OK" for Volume=0.0 and Quantity=94...	2018-06-04 19:22:46.3700000
✓	JOEXE	JOEXE	Input WRCOS selected 745 records	2018-06-04 19:22:46.2240000
✓	JOEXE	VIGTP	Function=VIGTP Calculate Global Transfer Pricing (step do...	2018-06-04 19:22:46.1950000
✓	JOEXE	VIGTP	Records processed with Status OK=8745, Abort=0, Error=0...	2018-06-04 19:22:46.1950000

Application Monitor

The search, filtering and sorting of messages is also supported.

The following table explains the key features available.

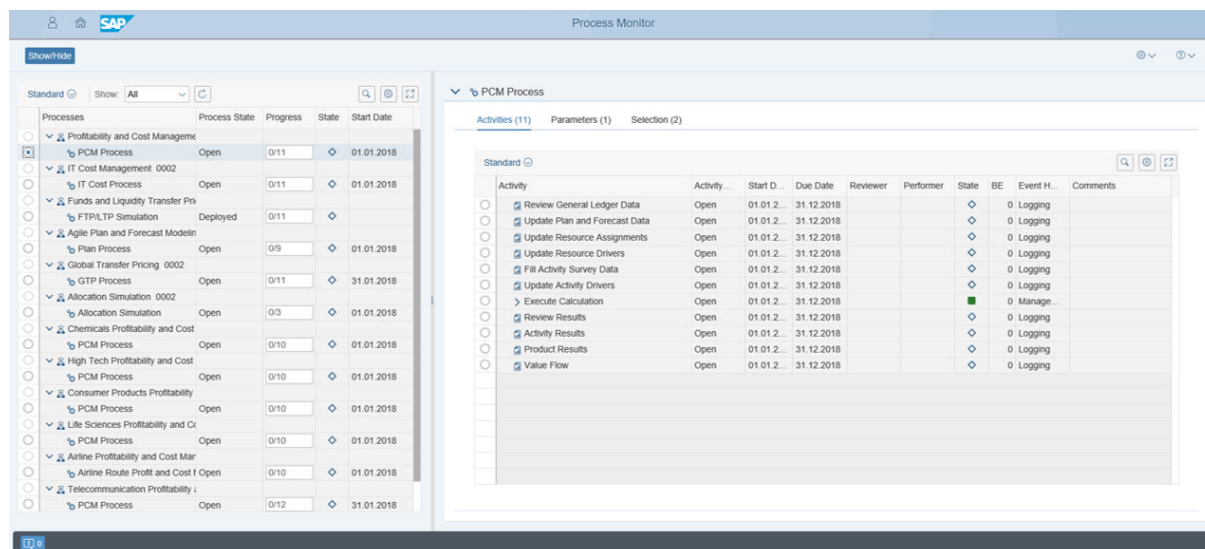
Key Feature	Use
Run Log	<p>The application creates a unique log entry each time an individual function is generated and run. The log contains the following information:</p> <ul style="list-style-type: none"><li>• A unique run ID</li><li>• A status</li><li>• A run set ID</li><li>• An input set ID</li><li>• A timestamp</li><li>• The name of the user who executed the run or generated the function</li></ul>
Message Log	<p>This contains the list of messages that are associated with every execution. The list of messages usually contains the following information:</p> <ul style="list-style-type: none"><li>• The status and results of a run</li><li>• Function-specific messages that are associated with a run (for example, unassigned items for allocation, records that were not transferred for the transfer structure/derivation)</li><li>• The results of a generation</li></ul>

## Related Information

For more information about the definition of custom specific checks that are logged in the application monitor, see [Environment \[page 81\]](#).

## 1.1.4.2 Process Monitor

The application enables the user to examine all currently active and past processes.



Process Monitor

Search, filter and sorting of processes and activities is supported as well.

The following table explains the key features available.

Key Feature	Use
Processes	Processes are displayed in a hierarchy together with a Progress indicator shows, how many of the included activities are finished.
Activities	If a process is selected, then on the right side the activities of the process are displayed.
Parameters	All parameters, which are relevant for the execution of the process activities are listed here together with their values.
Selections	All selections, which are relevant for the execution of the process activities are listed here together.

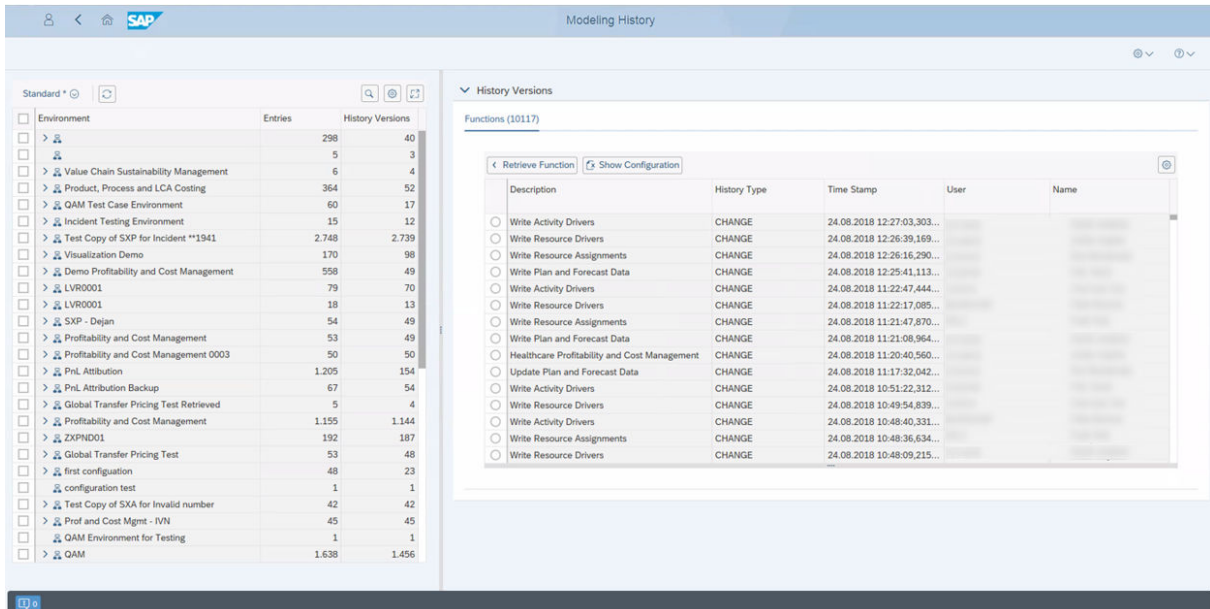
## Related Information

For more information on how to define processes, activities, parameters and selection fields, see [Calculation Unit \[page 83\]](#).

For more information on how to deploy processes, see [Processes \[page 47\]](#).

## 1.1.4.3 Modeling History

The application enables the user to trace and inspect the configuration changes to a model within an environment. This helps the user to trace and audit, who did what and when. Depending on the user authorizations, even historic versions of environments and functions can be restored.



Modeling History

The following table explains the key features available.

Key Feature	Use
Environment List	All current and historic environments are listed on the left-hand side of the screen.
History Versions	Once an environment is selected, a detailed list of all changes to the environment, functions and fields are displayed on the right-hand side of the screen.  Old versions of an environment, function or field can be selected and restored.

## Related Information

For more information about modeling, see [Modeling Environment \[page 24\]](#).

## 1.1.5 Tools

### 1.1.5.1 Activate Function

You can use this tool to activate a function without having to access the modeling environment.

The following fields are available:

Fields	Description
Environment	This is a 3-digit alphanumeric ID that is permanently assigned to the environment. Use the <b>F4</b> help to display all environments configured in the system and client. This field is used as a filter to specify an environment
Version	This is a 4-digit alphanumeric ID that is permanently assigned to the environment. You can use this field as a filter to reduce the number of records so that the system checks only within the specified version of the environment.
Function	The identifier of the function where the data to be activated is located. Use the <b>F4</b> help to display all the functions configured in the specified environment and version.
Run Type	Allows you to select which run type to perform. You can choose the following options: <ul style="list-style-type: none"><li>• Activation (ACT) Generates the procedure of the environment/function in real time</li><li>• Activation Simulation (ACT_SIMU) Runs the tool in test mode. The system issues configuration errors upfront without generating its procedure.</li></ul>
Show Model Flow Diagram	Allows you to view the Model Flow Diagram of the environment.
Active necessary func. only	Activates all executable function, data sources and process templates. This can only be used when you activate an environment.

## Procedure

In the client where SAP Profitability and Performance Management is installed, choose **SAP Menu** > **Profitability and Performance Management** > **Tools** > **Activate Function** or launch transaction code `/NXI/P1_FW_ACTIVATE`.

1. The **Performance Management Activate** window appears.
2. Make entries in or choose the following required (\*) and optional fields:
  - \*Environment
  - \*Version

- Function
  - Run Type
3. Choose one of the following options:
    - Show Model Flow Diagram
    - Activate necessary func. only

After you have made the relevant selections, the program can be executed immediately, either in the background or as a scheduled task. Continue with the following steps.

### Execute Immediate (F8)

1. Choose *Execute* or **F8**.
2. The *Display Logs* window opens.

### Execute in Background – Immediate

1. Choose **► Program ► Execute in Background ►** on the main menu.
2. On the *Background Print Parameters* screen, choose *Continue*.
3. On the *Start Time* screen, choose *Immediate* and *Save* consecutively to run the program in the background.
4. To see the progress of the background run, launch transaction code **SM37**.

### Execute in Background – Scheduled

1. Choose **► Program ► Execute in Background ►** on the main menu.
2. On the *Background Print Parameters* screen, choose *Continue*.
3. On the *Start Time* screen, choose *Date/Time*.
4. The *Date/Time* section appears, and you can specify the date and time that you want to schedule for the run.
5. To see the progress of the background run, launch transaction code **SM37**.

## 1.1.5.2 Run Function

This tool helps you to run functions without having to access the modeling environment.

The following fields are available:

Field	Description
Environment	This is a 3-digit alphanumeric ID that is permanently assigned to the environment. Use the <b>F4</b> help to display all the environments configured in the system and client. This field is used as a filter to specify an environment.
Version	This is a 4-digit alphanumeric ID that is permanently assigned to the environment. You can use this field as a filter to reduce the number of records that the system checks to within a specified version of the environment.

Field	Description
Calculation Unit	This is the ID of the calculation unit where the function relevant for execution is located. Choose <input type="text" value="F4"/> to display all the calculation units configured in the specified environment.
Process	Calculation-unit-wide unique ID of a process template. This can be referred to in process management to instantiate processes.
Activity	Process-template-wide unique ID of an activity. This can be referred to in report elements.
Function	The ID of the function relevant for execution. Choose <input type="text" value="F4"/> to display all functions configured in the specified environment and version entered above.
Run ID	The ID of the run relevant for execution.
Business Event	The ID of the business event relevant for execution.
Run Type	Allows you to select which run type to perform. You can choose from the following options: <ul style="list-style-type: none"> <li>• Run (RUN) Executes the run of the chosen function in real time</li> <li>• Run Simulation (RUN_SIMU) Executes the run of function in test mode</li> </ul>
Package	The ID of the package of a function relevant for execution.
Package Parameter	Specifies a finite value for the parameter to be incorporated in function execution.
Package Selection	Specifies a field filter (selection) to be incorporated in function execution.
Run Mode	Specifies the mode that a function is run in.  See <a href="#">Parallelization and Partitioning [page 68]</a> .
Synchronous Execution	Defines whether the run is carried out in synchronous or asynchronous mode: <ul style="list-style-type: none"> <li>• When set to "Yes" (checked): Unless the execution of the run is completed, you will not be able to move or do anything on the user interface. In modeling, after synchronous execution you will see the application log.</li> <li>• When set to "No" (unchecked): Asynchronous mode is activated. Once you start execution, the run is done in a separate process. You do not have to wait until the run is completed before starting another activity. In modeling, the run is always asynchronous and once you start it, you can move to another function to run or edit it.</li> </ul>

Field	Description
Show Result	<p>Indicates whether the result of the run is displayed:</p> <ul style="list-style-type: none"> <li>• When set to “Yes” (checked): The system displays the results of the run in the SAP NetWeaver system.</li> <li>• When set to “No” (unchecked): Results do not appear in SAP NetWeaver system after execution</li> </ul>

## Procedure

In the client where SAP Profitability and Performance Management is installed, choose **SAP Menu** **Profitability and Performance Management** **Tools** **Run Function** or launch transaction code `/NFI/P1_FW_RUN`.

1. The *Performance Management Run* window appears.
2. Make entries in or selections for the following required (\*) and optional fields:
  - \*Environment
  - \*Version
  - Calculation Unit
  - Process
  - Activity
  - Function
  - Run ID
  - Business Event
  - Run Type
  - Package
  - Package Parameter
  - Package Selection
  - Run Mode
3. Choose one of the following options:
  - Synchronous Execution
  - Show Result

After you have made the relevant selections, the program can be executed immediately, either in the background or as a scheduled task. Continue with the following steps.

### Execute Immediate (F8)

1. Choose *Execute* or **F8**.
2. The *Display Logs* window appears.

### Execute in Background – Immediate

1. Choose **More** **Program** **Execute in Background**.



2. On the *Background Print Parameters* screen, choose *Continue*.
3. On the *Start Time* screen, choose *Immediate* and *Save* consecutively to run the program in the background.
4. To see the progress of the background run, launch transaction code **SM37**.

### Execute in Background – Scheduled

1. Choose **► Program > Execute in Background >** on the main menu.
2. On the *Background Print Parameters* screen, choose *Continue*.
3. On the *Start Time* screen, choose *Date/Time*.
4. The *Date/Time* section appears, where you can specify the date and time that you want to schedule for the run.
5. To see the progress of the background run, launch transaction code **SM37**.

## 1.1.5.3 Delete Temporary Data

This tool deletes Y-tables, including the data produced by functions in SAP Profitability and Performance Management.

The following fields are available:

Fields	Description
Environment	This is a 3-digit alphanumeric ID that is permanently assigned to the environment. Use the <b>F4</b> help to display all the environments configured in the system and client. This field is used as a filter to specify an environment.
Version	This is a 4-digit alphanumeric ID that is permanently assigned to the environment. You can use this field as a filter to reduce the number of records that the system checks to within the specified version of the environment.
Calculation Unit	The ID of the calculation unit where the data to be deleted is located.
Function	The ID of the function where the data to be deleted is located. Choose <b>F4</b> to display all the functions configured in the environment and version specified above.
Run In Simulation	If you choose this option, the system executes the program without actually deleting the data. Otherwise, the system executes the deletion.

## Procedure

In the client where SAP Profitability and Performance Management is installed, choose ► *SAP Menu* ► *Profitability and Performance Management* ► *Tools* ► *Delete Temporary Data* or launch transaction code `/NXI/P1_FW_DEL_TDATA`.

1. The *Delete Temporary Data* window appears.
2. In the *Select Function* section, make entries in the following optional fields:
3. In the *Simulation* section, you can choose whether the program is run in actual mode or in simulation mode.

Once you have made the relevant selections, the program can be executed immediately, in the background or as a scheduled task. Continue with the following steps.

### Execute Immediate (F8)

1. Choose *Execute* or `F8`.
2. The system displays a pop-up message advising you that the data for the selected function will be deleted. It asks you for confirmation to proceed:
  - If you choose *Yes*, the data is deleted.
  - If you choose *No*, the system cancels the program.

### Execute in Background – Immediate

1. Choose ► *Program* ► *Execute in Background* on the main menu.
2. On the *Background Print Parameters* screen, choose *Continue*.
3. On the *Start Time* screen, choose *Immediate* and *Save* consecutively to run the program in the background.
4. To view the progress of the background run, launch transaction code **SM37**.

### Execute in Background – Scheduled

1. Choose ► *Program* ► *Execute in Background* on the main menu.
2. On the *Background Print Parameters* screen, choose *Continue*.
3. On the *Start Time* screen, choose *Date/Time*.
4. The *Date/Time* section appears, where you can specify the date and time that you want to schedule for the run.
5. To view the progress of the background run, launch transaction code **SM37**.

## 1.2 Concepts for Key Users

Get an overview of the general concepts and integration capabilities on which SAP Profitability and Performance Management is built.

The following concepts are relevant for key users to help them understand how SAP Profitability and Performance Management works and how it can be used.

1. **Financial and Business Modeling Entities**  
SAP Profitability and Performance Management uses entities like `Environments`, `Calculation Units` and `Functions` to structure and simplify the design of financial and business models, irrespective of the specific purpose and across business areas such as controlling, finance or risk.
2. **Function Building Blocks and reusable Templates**  
The functions use a common building block approach so that they can be plugged together to work in a common financial model. Each of these functional building blocks are systematically designed to be available and visible for use in every function only as necessary. In a general context, these function building blocks comprise header, input, lookup, signature, rules, checks and documentation.
3. **Information Models for Business Entity Master Data and Lookup**  
Data model functions can be used to make central master data information available to all functions via lookup formulas.
4. **Parallelization and Partitioning**  
By default, SAP Profitability and Performance Management takes care of runtime optimization automatically. For high-end computing requirements, you can make manual parallelization and partitioning settings to optimize the runtime further.
5. **Roles and Authorizations**  
SAP Profitability and Performance Management allows you to manage authorizations based on applications and functions. You can also set up characteristic-based authorizations for data to restrict the visibility of data.
6. **Integration with non-SAP Systems and File Import/Export**  
You can integrate SAP Profitability and Performance Management with non-SAP systems using various industry standards.
7. **Integration with BW, BPC and AfO**  
SAP Profitability and Performance Management allows integration with SAP Business Warehouse, SAP Business Planning and Consolidation and SAP Analysis for Office, including redundancy-free reuse of data, master data and hierarchies.
8. **Integration with ERP and S/4HANA**  
SAP Profitability and Performance Management allows integration with ERP and SAP S/4HANA, including redundancy-free reuse of data, master data and hierarchies as well as allocation rules.
9. **Integration with SAP Analytics Cloud and SAP Digital Boardroom**  
SAP Profitability and Performance Management allows integration with SAP Digital Boardroom and SAP Analytics Cloud using live data and imported data connections.
10. **Integration with FRDP**  
SAP Profitability and Performance Management allows integration with Finance and Risk Data Platform (FRDP), including redundancy-free reuse of data, master data and hierarchies, as well as the CVPM process orchestration of SAP Profitability and Performance Management functions and fast Results Data storage.

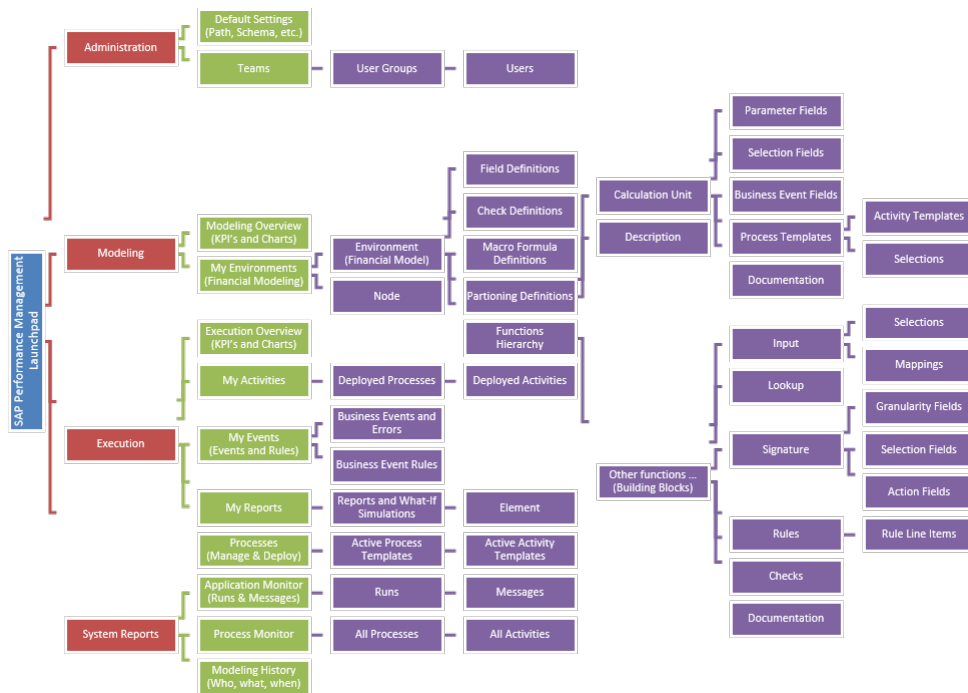
## Related Information

For more information about SAP Profitability and Performance Management functions see [Functions \[page 79\]](#).

## 1.2.1 Financial and Business Modeling Entities

Get an overview of the most important entities in SAP Profitability and Performance Management and where to find them.

SAP Profitability and Performance Management uses entities to structure, harmonize and simplify the design of a financial and business model irrespective of its purpose (for example, controlling, finance or risk).



Financial and Business Modeling Entities

The picture gives an overview of these entities, how they relate to each other and where to find them on the user interface of the respective application.

Users can view or edit all or parts of the above entities depending on the authorizations and roles they have been assigned.

## Related Information

For more information about SAP Profitability and Performance Management functions see [Functions \[page 79\]](#).

For more information about entities see [Applications for Business Users \[page 13\]](#).

## 1.2.2 Function Building Blocks and Reusable Templates

Function building blocks are the basis on which SAP Profitability and Performance Management functions are built. This allows functions to be connected to each other to design comprehensive financial and business

models, and to fulfill complex activities in end-to-end processes. It is also the basis for incorporating reusable function templates which can reduce the configuration effort.

The following function categories are available:

1. Information functions

These functions define the data and information model in an environment. Technically, they act as a proxy that contains the details required to read and – if allowed – write data from and to this data model. Functionally, they define or display the available fields from that model. The following functions are provided:

- [Model BW \[page 217\]](#)
- [Model RDL \[page 221\]](#)
- [Model Table \[page 219\]](#)
- [File Adapter \[page 196\]](#) (of type “Import”)

2. Processing functions

These functions process data from information functions and produce an output. Processing functions can be connected so that the output of a function is used as input for subsequent functions. Most functions belong to this category, such as the following:

- [Calculation \[page 206\]](#)
- [Allocation \[page 92\]](#)
- [Funds Transfer Pricing \[page 120\]](#)
- [Valuation \[page 132\]](#)
- [Flow Modeling \[page 136\]](#)
- [Join \[page 108\]](#)
- [View \[page 232\]](#)
- [Transfer Structure \[page 214\]](#)
- [Derivation \[page 105\]](#)

3. Write and Adapter functions

These functions can store data or hand over data to external systems for further processing. They also provide output to subsequent functions. Functions belonging to this category include the following:

- [Writer \[page 230\]](#)
- [Remote Function Adapter \[page 199\]](#)
- [File Adapter \[page 196\]](#) (of type “Export”)

4. Query function

The *Query* function defines whether the data display for a user is read-only or editable. For the editable query, the input function has to be a *Model BW* function. A query function does not provide output to subsequent functions because its purpose is data input (when editable) and reporting (when read-only and editable).

- [Query \[page 225\]](#)

5. Calculation Unit function

The *Calculation Unit* function helps to structure larger environments into multiple parts that can define processes and activities independently of each other. A typical example is to use it to structure a decentralized month-end closing process into separate business units and/or closing units (= calculation units), where each closing unit has its own processes and activities. Only the results are stored in one central place at the end.

- [Calculation Unit \[page 83\]](#)

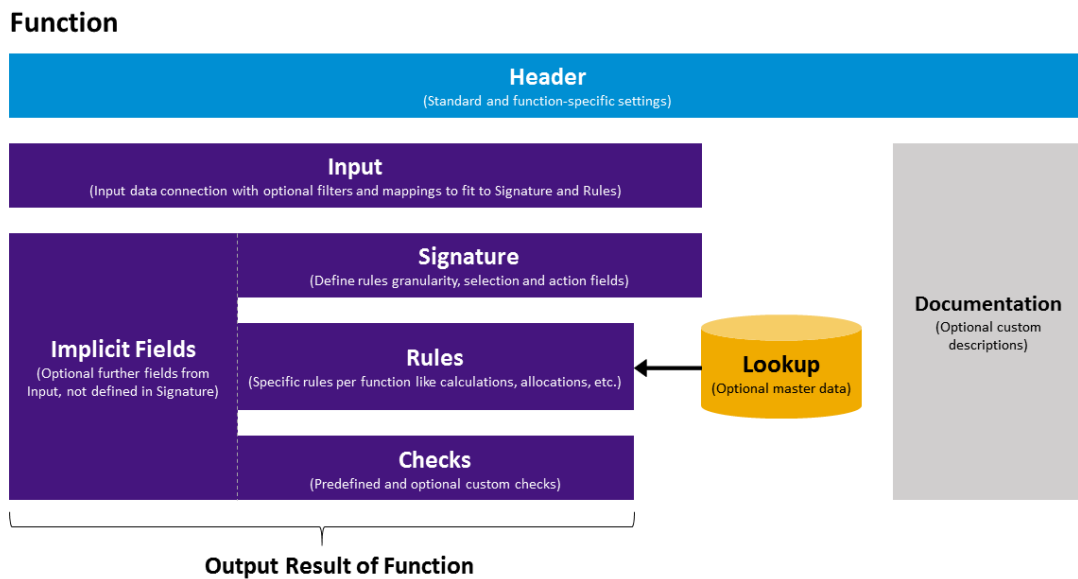
6. Description function

The *Description* function helps to structure and document the model. It does not affect the result but improves the readability of the model.

- [Description \[page 91\]](#)

Depending on the categories (as described above), some or all of the building blocks are relevant for a given function.

The diagram below provides a general overview of the function building blocks that appear as tabs on the function details user interface.



### Function Building Blocks

The following function building blocks are available:

- [Header \[page 63\]](#)
- [Input \[page 64\]](#)
- [Lookup \[page 64\]](#)
- [Signature \[page 64\]](#)
- [Rules \[page 66\]](#)
- [Checks \[page 66\]](#)

### Related Information

For more information about SAP Profitability and Performance Management functions see [Functions \[page 79\]](#).

## 1.2.2.1 Header

The header belongs to the individual part of a function. Wherever functionally feasible, it contains the following common standard settings:

1. Include original Input Data  
If you select “Yes”, the original input data is added to the output data along with the produced results. This makes it easier to model requirements, in cases where one scenario is built on top of another and the original scenario results therefore need to be kept and addition results need to be added to them.
2. Suppress initial Results  
If you select “Yes”, results that contain only their initial values in their action fields (for example, key figures 0 and characteristics “ ”) are excluded from the output. If initial result records have no effect on the result, it can reduce the data volume and processing of unnecessary records.
3. Result Model Table  
If no model table is assigned, the function is directly executed in the modeling environment and the results are stored in a function-specific temporary table. Otherwise, the results of the function are stored in the model table.

The following scenarios apply:

- Fields are in the results of a function but not included in the result model table, these fields are not automatically included or disregarded. During *Run* or *Show*, the function result adapts the fields and the data available in the result model table.
- A function uses an input function with the result model table, the function consumes the data stored in the result model table and uses all the fields.
- A function uses an input function with the result model table and the processing type is set to “sub-function”, the data is not written in the result model table unless the function is assigned as an activity under a process template or a run has been initiated.
- A function uses an input function with the result model table and the processing type is set to “executable”, the data is written in the result model table when the function is executed.

#### 4. Result Handling

This setting offers various options:

1. Include enriched data  
This setting includes only data records in the results to which a rule was applied. At the end of the function, if there are data records for which the system was unable to apply a rule, it writes a warning to the message log.
2. Include all data  
This setting includes all data records in the results, irrespective of whether a rule was applied or not.
3. Error on non-enriched data  
This setting works in the same way as for include enriched data. However, for non-enriched data an error message is prepared and further processing is done based on the function event type setting:
  1. Logging  
The error is written to the message log.
  2. Management  
The error is written to the message log and a business event is registered so that the business user can deal with the exceptional situation and fix it.
4. Abort on non-enriched data  
This setting works in the same way as errors in non-enriched data, but instead of an error message the system writes an abort message to the log, and the function is terminated.

## 1.2.2.2 Input

All processing functions, `Write` and `Adapter` functions, and `Query` functions have an input. The input connects the function to a preceding function.

You can configure specific selections to restrict the data transferred from the `Input` function, and can configure mappings to adapt the data to the required signature of the function. The latter also helps if the function rules are based on a function template and different data with different fields needs to be processed based on common rules.

In contrast to the data that comes from lookup, the data that comes from the input is the basis for the business event and error management, including partial restart capabilities.

## 1.2.2.3 Lookup

In the `Calculation`, `FTP` and `Valuation` functions, you can use lookup data models, which you can access in formulas to look up central master data settings.

To be able to use lookup data models in formulas, you first need to ensure they are registered on the [Lookup](#) tab.

## 1.2.2.4 Signature

All processing functions have a signature, which can produce a result for subsequent functions. The signature defines the minimum number of relevant fields of a function. There can also be further implicit fields from the input. These simply pass through the function without any change or any effect on the logic, and also appear in the output if no aggregation within the function is defined. If you add or remove fields from the data model this implicit field handling ensures the following:

- Data model changes do not affect the calculation model as long as no signature field is removed. If signature fields are missing, the input needs to be adapted to provide another field or mapping or a formula to substitute the original field. Alternatively, the rules of the function need to be adapted.
- Data model changes are propagated through all subsequent functions of the model automatically. The only exception is if a function uses explicit field handling to explicitly define the fields of the output, for example, in a view or if a rule includes aggregation (grouping).
- Data model changes are automatically propagated to queries for reporting. If a field is added, it is available for reporting. If a field is removed, it is no longer available for reporting. The latter can have an effect on predefined layouts, which then look different and might need to be adjusted.

The only functions that offer explicit field handling are views and joins. Aggregations can be run based on specific configuration settings and rules in the `Allocation`, `Valuation`, `FTP`, `Flow Modeling`, `Join` and `Transfer Structure` functions.

The signature is the interface of a function and defines a simple pivot table, in which calculations and logic can be applied. The signature is structured into three groups of fields, on which computations and modifications can occur:



- Header fields that describe the granularity characteristics.
- Row fields that describe the selection characteristics.
- Value fields that describe the action key figures and characteristics.

Granularity Fields		VERSION	Actual		
Selection Fields	COST_CENTER	COST_ELEMENT	AMOUNT	QUANTITY	Action Fields
	Cost Center 1	Cost Element 1	100	1	
	Cost Center 2	Cost Element 2	200	2	
	Cost Center 3	Cost Element 3	300	3	
	Cost Center 4	Cost Element 4	400	4	
	Cost Center 5	Cost Element 5	500	5	
	Cost Center 1	Cost Element 6	600	6	
	Cost Center 2	Cost Element 7	700	7	
	Cost Center 3	Cost Element 8	800	8	
	Cost Center 4	Cost Element 9	900	9	
	Cost Center 5	Cost Element 10	1000	10	

### Signature

The figure contains an example where the granularity fields contain the functional area, the selection fields contain the cost center plus cost element and the action fields contain the amount and quantity.

More details regarding these three groups of fields in a signature are described below:

#### 1. Granularity Fields

Granularity fields define the minimum granularity of a function. They cannot occur as selection or action fields in the same function, which means rules or modifications on the granularity fields are not allowed. Instead, the granularity fields always stay stable from input through processing until output of the function. In data warehouses they are also known as block characteristics. Formulas and formula functions, like aggregations including SAP HANA window functions, are not allowed across values of these characteristics. Granularity fields can be used for horizontal package parallel processing, because they ensure that the overall result is always the same, irrespective of whether all the data is processed in one or multiple packages when you use granularity fields for grouping. A typical example is the granularity field "VERSION" in a calculation function, which ensures that all calculations are run for each version and not across all versions.

#### 2. Selection Fields

Selection fields can be used as a condition within the rules of a function. A typical example is the selection field "COST\_ELEMENT" in a calculation, which allows the rules to be applied to selected cost elements only.

#### 3. Action Fields

Action fields can be used for calculation formulas and assignments within the rules of a function because their values can be changed in the function. A typical example is the action field "AMOUNT QUANTITY" in an allocation, which can then be allocated and distributed.

Selection and action fields can also overlap in certain functions. For example, a cost center can be used in a derivation both as a selection and an action field to fill in a default cost center value if the original value is empty.

## 1.2.2.5 Rules

Rules contain the individual part of most of the functions. They contain the following common fields:

1. Rule ID  
The rule ID has to be unique in a function. If (interim) results of a function are persisted, the rule ID is stored to enable you to trace which rule of a function was applied to each data record.
2. Rule State  
The rule state can be active or inactive. Inactive rules are not executed. For example, you can set a rule to inactive if you temporarily do not want the system to apply it. You do not need to delete the rule and reenter it again later.
3. Rule Level  
You can use the rule level to define hierarchical rules.
4. Rule Description  
You can use the rule description to enter a user-defined text and comments.

## 1.2.2.6 Checks

You can run custom checks on the results data of all processing functions, and of write and adapter functions.

Checks are defined at environment level and can be registered in one or more functions. When a function is executed, these checks are applied to the result of the function. If the check condition is satisfied, an appropriate message is written to the application log.

If the business event and error management is activated for the function and the message type is either "error" or "abort", business events are also created. You can deal with these business events in the `My Events` application.

## 1.2.3 Information Models for Master Data and Lookup

The term *Master Data* is used in the following two ways:

1. Master Data of a Field  
Field master data defines the values permitted for a field like `InfoObjects` and data elements. For `InfoObjects`, you can also define hierarchies on top to structure the permitted values further for calculation and reporting.
2. Master Data of a Business Entity  
Business entity master data defines a table or a set of a tables used to define records with combinations of characteristic and key figure values according to the business requirements, like product master data, financial instrument master data and so on. Business entity master data is rarely changed and is reused by many functions to control calculation (for example, how the funds transfer price of a retail loan is calculated).  
Business entity master data can reside in any model function.

Both kinds of master data are supported by SAP Profitability and Performance Management and can be used for lookup.

## Usage and Lookup

The usage and lookup of master data happens in two steps.

1. The respective model functions needs to be registered on the *Lookup* tab. These model functions then contain the master data. To use the master data for further processing, a lookup ID has to be defined.
2. The lookup of data can then be included in a formula. The format for lookup consists of the lookup ID followed by the field to be looked up and then square brackets, in which the selections are defined.

If multiple records fulfill the lookup criteria, the default aggregation is used to return exactly one value.

## Example

The following master data is available under lookup ID `MY_DATA`.

Example of Master Data

<code>COST_CENTER</code>	<code>COST_ELEMENT</code>	<code>AMOUNT</code>	<code>QUANTITY</code>
Cost Center 1	Cost Element 1	100	1
Cost Center 2	Cost Element 2	200	2
Cost Center 3	Cost Element 3	300	3
Cost Center 4	Cost Element 4	400	4
Cost Center 5	Cost Element 5	500	5
Cost Center 1	Cost Element 6	600	6
Cost Center 2	Cost Element 7	700	7
Cost Center 3	Cost Element 8	800	8
Cost Center 4	Cost Element 9	900	9
Cost Center 5	Cost Element 10	1000	10

The lookup statement `MY_DATA.AMOUNT[COST_ELEMENT='Cost Element 1']` would return the amount 100.

The lookup statement `MY_DATA.QUANTITY[COST_ELEMENT='Cost Element 1']` would return the quantity 1.

The lookup statement `MY_DATA.COST_ELEMENT[AMOUNT=500]` would return the cost element `Cost Element 5`.

If the default aggregation for the field `Amount` is summation, the lookup statement `MY_DATA.AMOUNT[COST_CENTER='Cost Center 1']` would return the amount 700, because the value "Cost Center 1" is not unique and the amount is therefore added up to  $100+600 \Rightarrow 700$  automatically.

If the default aggregation for the field `Cost Element` is maximum, the lookup statement `MY_DATA.COST_ELEMENT[COST_CENTER='Cost Center 2']` would return the cost element `Cost Element 7`, because the value "Cost Center 2" is not unique and the cost element maximum is therefore taken automatically ( "Cost Element 7").

The lookup statement `MY_DATA.AMOUNT [COST_ELEMENT='ABC' ]` would return the amount 0, which is the initial value of the field `Amount` because there is no cost element "ABC" in the master data.

The lookup statement `MY_DATA.COST_CENTER [COST_ELEMENT='ABC' ]` would return the cost center " ", which is the initial value of the field `Cost Element` because there is no cost element "ABC" in the master data.

## Related Information

For more information about SAP Profitability and Performance Management functions see [Functions \[page 79\]](#).

## 1.2.4 Parallelization and Partitioning

For high-end scenarios, you need to explicitly configure parallelization and partitioning in the modeling environment to enable you to do the following:

1. Handle datasets with more than 2 billion records  
If the data volume of a function exceeds 2 billion records, partitioning and parallelization must be set up so that the volume of each partition is below 2 billion records.
2. Actively manage RAM and CPU usage  
If the usage of RAM and CPU resources during execution needs to be restricted, you can set up partitioning and parallelization so that only a subset of data is processed at the same time.

In both scenarios, the dataset has to be logically separated into parts that can be processed independently of other parts.

## Partitioning Setup

You set up partitioning in the following two steps:

1. Register a field on the environment *Partitioning* tab. This field must be available in the input data being processed, which is then suitable for the logical separation of datasets into independent parts.
2. Enter separate values for each partition to identify and select the data in the partition.

A typical example is a version field, where the first partition is identified by the value "ACTUAL", the second partition by the value "PLAN", the third partition by the value "FORECAST", and so on.

You can define parallelization on top of a partitioning configuration by defining numeric level values for each partition value. By default, all partition values use the level value "1", which means that all partitions are calculated in parallel during execution of level 1. If you change the level for single partition values, you can enforce sequential execution.

## Example

<i>Partitioning Field</i>	<b>VERSION</b>	
<i>Partition Ranges</i>	<b>Field Value</b>	<b>Level</b>
	<b>ACTUAL</b>	<b>1</b>
	<b>PLAN</b>	<b>2</b>
	<b>FORECAST</b>	<b>2</b>
	<b>SCENARIO 1</b>	<b>3</b>
	<b>SCENARIO 2</b>	<b>4</b>

Example of Partitioning

In the above example, 5 partition ranges for the field `VERSION` are set up. Based on the level, it is defined that the actual version is executed first, then the plan and forecast versions are executed in parallel on level 2. After that, scenario 1 is executed, and finally scenario 2.

## Run Mode

The run mode defines the system's behavior when a run of a function is triggered, respectively when a Model BW or Model Table with source environment is activated. The default run mode is specified in the partitioning setup.

The following settings are available to determine the run mode:

1. Parallel (P) or Sequential (S):
  1. "Parallel" means that the control returns immediately to the caller and does not wait for the function execution to be finished. The success of the execution is noted in the application log.
  2. "Sequential" means that the control returns to the caller only after the function execution is finished.
2. Packaged (P) or Unpackaged (U):
  1. "Packaged" means that the ranges of the partitioning are used to trigger multiple instances of the function executions, each of them restricted to the field value defined in the range.
  2. "Unpackaged" means that one instance of the function execution is triggered without restriction to a range field value.
3. Batch (B), Dialog (D) or Process like Caller (X):
  1. "Batch" means that a new background job is opened, the execution of the function is submitted to this background job and the job definition is closed afterwards.
  2. "Dialog" means that a new task is opened in dialog mode, where the execution of the function is triggered.

3. "Process Like Caller" means that the execution of the function is triggered directly in the process of the caller (which can be either in dialog or background mode)
4. Partitioned (P):
  1. "Partitioned" means that the environment managed Model Table or Model BW is activated in such a way that the partitioning range information is applied on the database. This is especially helpful in scale-out environments.

## i Note

For model tables, a change from non-partitioned to partitioned or vice versa updates the database immediately. For Model BWs, this change request is only recognized, and the BW administrator has to trigger the execution in the BW administration application.

## Example

Partitioning Field	VERSION
Field Value	Level
ACTUAL	0
PLAN	0

Run Mode: PPB Parallel, Packaged, Batch Process

### Sample Dataset

a) Packaged: System will package the dataset based on the partitioning field, in this case VERSION

VERSION	PRODUCT_ID	QUANTITY
ACTUAL	P001	40
ACTUAL	P002	20
ACTUAL	P003	10
ACTUAL	P004	50
ACTUAL	P005	30
PLAN	P001	60
PLAN	P002	20
PLAN	P003	50
PLAN	P004	80
PLAN	P005	10

b) Parallel: System will process the packaged dataset in parallel

VERSION	PRODUCT_ID	QUANTITY
ACTUAL	P001	40
ACTUAL	P002	20
ACTUAL	P003	10
ACTUAL	P004	50
ACTUAL	P005	30
PLAN	P001	60
PLAN	P002	20
PLAN	P003	50
PLAN	P004	80
PLAN	P005	10

PROCESS 1 →

PROCESS 2 →

c) Batch Process: The processes will be assigned to the batch job (background job)  
Note: You will see the actual process assigned in transaction code SM51

VERSION	PRODUCT_ID	QUANTITY
ACTUAL	P001	40
ACTUAL	P002	20
ACTUAL	P003	10
ACTUAL	P004	50
ACTUAL	P005	30
PLAN	P001	60
PLAN	P002	20
PLAN	P003	50
PLAN	P004	80
PLAN	P005	10

PROCESS 1 → BTC

PROCESS 2 → BTC

Partitioning Field	VERSION
Field Value	Level
ACTUAL	0
PLAN	0

Run Mode: SUX Sequential, Unpackaged, Process Like Caller

### Sample Dataset

a) Unpackaged: System will disregard the partition range and will not package the dataset

VERSION	PRODUCT_ID	QUANTITY
ACTUAL	P001	40
ACTUAL	P002	20
ACTUAL	P003	10
ACTUAL	P004	50
ACTUAL	P005	30
PLAN	P001	60
PLAN	P002	20
PLAN	P003	50
PLAN	P004	80
PLAN	P005	10

b) Sequential: System will process the unpackaged dataset by assigning one process (sequential)

VERSION	PRODUCT_ID	QUANTITY
ACTUAL	P001	40
ACTUAL	P002	20
ACTUAL	P003	10
ACTUAL	P004	50
ACTUAL	P005	30
PLAN	P001	60
PLAN	P002	20
PLAN	P003	50
PLAN	P004	80
PLAN	P005	10

PROCESS 1 →

c) Process Like Caller: The processes will be assigned to either dialog job or batch job (background job) depending on the caller  
Example: In the modeling UI, the process will be assigned in a dialog job, whereas in a CPM process with "Run in Background" it will be assigned in a batch job.

VERSION	PRODUCT_ID	QUANTITY
ACTUAL	P001	40
ACTUAL	P002	20
ACTUAL	P003	10
ACTUAL	P004	50
ACTUAL	P005	30
PLAN	P001	60
PLAN	P002	20
PLAN	P003	50
PLAN	P004	80
PLAN	P005	10

PROCESS 1 → BTC / DIA

In the SUX example, the data records will be processed using one dialog or batch job sequentially.

## Comparison of Different Run Modes

If no partitioning is assigned to a function, a trigger for execution always uses the default run mode SUX, which has the settings Sequential, Unpackaged and Process Like Caller. If you have assigned the partitioning ID to a function, it is used if this function is triggered for execution in the following applications:

- In the Modeling Environment  
The modeling user can overrule the standard described above in the *Advanced* tab of the run dialog.
- In the *My Activities* application
- In the *My Reports* application

## Related Information

For more information about functions, see SAP Profitability and Performance Management [Functions](#) [page 79].

## 1.2.5 Roles and Authorizations

SAP Profitability and Performance Management is targeted toward the business user. It is designed to enable the business department (for example, accounting, controlling and risk) to operate modeling, execution and analysis of data with minimal IT involvement. The solution is delivered with preconfigured user roles and provides each of them with a specialized working environment optimized to support them in their main area of responsibility.

The solution comes with the following predefined roles:

1. Administration Role /NXI/P1\_ADMIN\_USER\_ALL  
Users assigned to this role can run the following transactions:
  - Default Settings
  - Teams
2. Modeling Role /NXI/P1\_MODELING\_USER\_ALL  
Users assigned to this role can run the following transactions:
  - Modeling Overview
  - My Environments
3. Execution Role /NXI/P1\_EXECUTION\_USER\_ALL  
Users assigned to this role can run the following transactions:
  - Execution Overview
  - My Activities
  - My Events
  - My Reports
4. Execution Role /NXI/P1\_EXECUTION\_MAN\_ALL  
Users assigned to this role can run the transaction Processes.
5. Management Role /NXI/P1\_SYSTEM\_USER\_ALL  
Users assigned to this role can run the following transactions:
  - Application Monitor
  - Process Monitor
  - Modeling History

By default, this role provides only display rights. To retrieve historic versions, the authorizations “Overwrite” and “Copy” are required. For more information, see below.

## Granular Authorizations

In addition, you can maintain granular authorizations with the authorization object /NXI/P1F using the following fields:

1. /NXI/P1ENV  
This attribute defines the environment for which the authorization is maintained.
2. /NXI/P1VER  
This attribute defines the environment version for which the authorization is maintained.
3. /NXI/P1PCU  
This attribute defines the calculation unit for which the authorization is maintained.
4. /NXI/P1FTY  
This attribute defines the function type for which the authorization is maintained.
5. /NXI/P1FID  
This attribute defines the function ID for which the authorization is maintained.
6. /NXI/P1ACT  
This attribute defines for which action the authorization is maintained. The following values are allowed:
  - "Create"
  - "Display"
  - "Delete"
  - "Activate"
  - "Execute"
  - "Transport"
  - "Edit"
  - "Merge"
  - "Analysis"
  - "Remove"
  - "Copy"
  - "Overwrite"

You can use "\*" as a placeholder for each authorization attribute to cover all the possible values of the attribute.

## Related Information

For more information about SAP Profitability and Performance Management functions, see [Functions \[page 79\]](#).



## 1.2.6 Integration with BW, BPC and Analysis for Office

SAP Profitability and Performance Management allows the convenient integration with SAP Business Warehouse, SAP Business Planning and Consolidation and SAP Analysis for Microsoft Office, including redundancy-free reuse of data, master data and hierarchies.

### SAP Business Warehouse Integration

The solution uses SAP Business Warehouse capabilities as an underlying Tool-BW in standalone scenarios. This includes relevant applications for management and maintenance of the BW and reusing Business Warehouse objects in integrated scenarios:

1. InfoObjects with master data and hierarchies  
The `Environment` function allows you to maintain managed InfoObjects and fields referring to BW managed InfoObjects.
2. Data Store Objects (Advanced)  
The `Model BW` function allows you to maintain managed ADSOs and to refer to BW managed DSOs and ADSOs.
3. InfoCubes  
The `Model BW` function allows you to refer to BW managed InfoCubes.
4. BW Queries  
The `Query` function allows you to maintain managed queries and to refer to BW managed queries.
5. Process Chains  
Process chains are generated automatically to control the vertical parallelization of functions involved in an activity.
6. Open ODS Views  
Open ODS views are generated automatically on top of nearly all functions to enable the analytic report screen.
7. Data Transfer Processes  
The Data Transfer Process can be used to store data in BW objects.
8. Characteristics-based Authorization  
BW characteristics-based authorization is used to secure and restrict access to data (for example, by legal entity or product group).

### SAP Business Planning and Consolidation Integration

The solution reuses the following SAP Business Planning and Consolidation objects, including the relevant applications for managing SAP Business Planning and Consolidation objects:

1. Planning Application Kit (PAK)  
The `Model Writer` function allows you to hand results to the SAP HANA-based planning engine buffer. These results can then be reviewed by a user before deciding to save the data. For more information, see [Analytics Component \[page 35\]](#).  
The `Model Writer` function also allows you to store results using the SAP HANA-based planning engine directly in a BW Object. For more information, see [Model Writer \[page 230\]](#).

## SAP Analysis for Microsoft Office Integration

Since SAP Profitability and Performance Management uses a lot of BW capabilities, it also uses all interfaces for a comprehensive SAP Analysis for Microsoft Office integration.

In the same way as in the web-based Reporting & Simulation application, it is possible to trigger writer functions of the BW write type “Planning” directly from SAP Analysis for Microsoft Office, and it is also possible to report and input data from Analysis for Office. For more information, see [Analytics Component \[page 35\]](#).

### Related Information

For more information about SAP Profitability and Performance Management functions, see [Functions \[page 79\]](#).

For more information about InfoAreas, see [Environment \[page 81\]](#).

For more information about InfoObjects, see [Environment \[page 81\]](#).

For more information about Data Store Objects (Advanced), see [Model BW \[page 217\]](#).

For more information about InfoCubes, see [Model BW \[page 217\]](#).

For more information about BW queries, see [Query \[page 225\]](#).

For more information about process chains, see [Parallelization and Partitioning \[page 68\]](#).

For more information about open ODS views, see [Analytics Component \[page 35\]](#).

For more information about HAP-based BW data transfer processes, see [Model Writer \[page 230\]](#).

For more information about characteristics-based authorization, see [Roles and Authorizations \[page 71\]](#).

For more information about BPC environments, see [Environment \[page 81\]](#).

## 1.2.7 Integration with SAP ERP and SAP S/4HANA

SAP Profitability and Performance Management allows integration with SAP ERP and SAP S/4HANA, including redundancy-free reuse of data, master data and hierarchies.

The solution uses SAP ERP and SAP S/4HANA capabilities to access accounting data in integrated scenarios:

1. Local Scenario

In this scenario, the solution is installed on the same NetWeaver client. It directly uses the following:

1. Reading of Master Data  
Master data attached to data elements and InfoObjects is reused.
2. Reading of Hierarchy Data  
Hierarchy data attached to InfoObjects is reused.
3. Reading Accounting and Controlling Data  
Accounting and controlling data available as SAP HANA-based CDS view interfaces is reused.

4. Posting of Accounting and Controlling Data  
The official BAPI is used for posting via the `Remote Function Adapter`.
  5. Other use cases  
Further read or write access use cases can be customized using the `Model View`, `Model Table` and `Remote Function Adapter` functions.
2. Remote Scenario
- In this scenario, the solution is installed on a separate NetWeaver client or instance. It can remotely reuse the following:
1. Reading of Master and Hierarchy Data  
Master and hierarchy data can be accessed remotely based on SAP HANA-based CDS view interfaces, but only during runtime.  
If this data needs to be accessed during design time in the modeling environment, the replication of the corresponding fields to local InfoObjects in the SAP Profitability and Performance Management instance has to be set up on the remote instance. For more information, see the SAP ERP and SAP S/4HANA documentation. Once this replication is set up, from an SAP Profitability and Performance Management perspective the master data and hierarchy data behaves as it does in the local scenario.
  2. Reading of Accounting and Controlling Data  
Accounting and controlling data available as SAP HANA-based CDS view interfaces from the remote SAP ERP and SAP S/4HANA instance can be reused.
  3. Posting of Accounting and Controlling Data  
For posting, the official BAPI is used. You need to specify the remote RFC destination on the *Advanced* tab for the environment.
  4. Other use cases  
Further read or write access use cases can be customized using the SAP Profitability and Performance Management functions `Model View`, `Model Table` and `Remote Function Adapter`.

## Related Information

For more information about SAP Profitability and Performance Management functions, see [Functions \[page 79\]](#).

## 1.2.8 Integration with SAP Analytics Cloud and SAP Digital Boardroom

SAP Profitability and Performance Management allows easy integration with SAP Analytics Cloud and SAP Digital Boardroom. It also reuses the integration capabilities of BW and SAP HANA, and supports live data connections and import data connections with SAP Analytics Cloud.

SAP Analytics Cloud can access data using the following artifacts in integrated scenarios:

1. Query functions  
SAP Analytics Cloud can access data using BW queries. The name of the BW query is visible in the function details header. This is the standard recommended design, because users can see in the solution exactly the same data as in SAP Analytics Cloud.

## 2. Information functions

SAP Analytics Cloud can also read data from various information functions . This is only necessary if the data needs to be processed further in SAP Analytics Cloud before the final results are presented to users.

### 1. Model View

Since model views do not hold data, but refer to a data source, SAP Analytics Cloud has to be configured to refer to the same data source as well.

### 2. Model Table

Access by SAP Analytics Cloud has to be configured according to the source type. The following options are available:

#### 1. Environment

The data is managed by the solution and can be accessed via a SAP S/4HANA view. The name of the SAP S/4HANA view is displayed in the function header as the table name.

#### 2. All other source types

All other model table source types refer to a data source. SAP Analytics Cloud has to be configured to refer to the same data source as well.

### 3. Model BW

Access by SAP Analytics Cloud has to be configured according to the source type. The following options are available:

#### 1. Environment

The data is managed by the solution and can be accessed via a SAP S/4HANA view. The name of the SAP S/4HANA view is displayed in the function header as the view name.

#### 2. All other source types

All other model BW source types refer to a data source. SAP Analytics Cloud has to be configured to refer to the same data source as well.

### 4. Model Results Data

Since model Results Data components do not hold data, but refer to a data source, SAP Analytics Cloud has to be configured to refer to the same data source as well.

## Related Information

For more information about SAP Profitability and Performance Management functions, see [Functions \[page 79\]](#).

## 1.2.9 Integration with SAP S/4HANA for Financial Products Subledger

SAP Profitability and Performance Management comes with predefined content called Estimated Cashflow Preparation (ECP).

Unlike the product's other traditional sample contents, the ECP content is integrated with SAP S/4HANA for financial products subledger 1812 through several data sources and triggered by CVPM processes.

The following is an overview of the integration points to help you to maximize the functions of the fixed content for Estimated Cash Flow Preparation with SAP S/4HANA for financial products subledger 1812.

## 1. Model Assignment (Actuarial granularity)

### 1. Input Tables

#### Contract Header

Table Name	Description
/1BC/AC<Client><RDA>_<RT>	Reinsurance Contract
/1BC/BR<Client><RDA>_<RT>	Contract Coverage
SRINS, S_PAPI	Pattern Assignment Portfolio Item
SRINS, S_ANAN	Analytical Attributes

#### Model Approach L&H

Table Name	Description
SRINS, S_AMS	Actuarial Model Stream
SRINS, S_BVOL	Business Volume

#### Model Approach P&C

Table Name	Description
SRINS, S_ULT	Ultimate
SRINS, S_FP	Factor Pattern
SRINS, S_RPE	Exposure Development Pattern
SRINSS_RPS,	Seasonality Pattern

Table Name	Description
SRINS, S_LFP	Lag Factor Pattern
SAFI, S_SCT_TVR	Reported Actual
/1BC/DABT_<Client>_FLAT	Business Transaction
SRINS, S_BEFCM	Manual Upload BECF
SRINS, S_EPSM	Manual Upload EPS

### 2. Processing

Model Assignment Processing is integrated with a CVPM process that can be executed using transaction code /BA1/FJ\_MODEL\_ASSIGN

### 3. Output Table

Table Name	Description
SRINS, S_ACG	Actuarial Granularity

## 2. Best Estimate Cash Flow Calculation

### 1. Input Table

Table Name	Description
/1BC/DAMD<Client>_BA1_F4_FXRATE_F	Forward Exchange Rates
SRINS, S_ACG	Actuarial Granularity
SRINS, S_BECECF	Best Estimate Cash Flow

## 2. Processing

Best Estimate Cash Flow processing is integrated with a CVPM process that can be executed using transaction code /BA1/FJ\_ECP.

## 3. Output Table

Table Name	Description
SAFI, S_BECECF	Best Estimate Cash Flow
SAFI, S_EPS	Exposure Period Split
SAFI, S_SCT_CDA	Change Driver Results
SAFI, S_SCT_VEC	Derived Cash Flow

## 3. Additional Functions (Simulation)

### 1. Input and Output Tables

The Simulation scenario uses additional input and output tables that resemble the input and output tables of the non-simulation process for Model Assignment and Best Estimate Cash flow Calculation business processes.

Table S\_ANANS (Analytical Attribute Simulation) is an example of a simulation table that resembles the non-simulation table S\_ANAN (Analytical Attribute).

### 2. Processing

ECP Functions are integrated with CVPM You can trigger a simulation process using the following transaction codes:

- /BA1/FJ\_MA\_SIMUL – CVPM transaction for Actuarial Granularity Simulation
- /BA1/FJ\_ECP\_SIM – CVPM transaction for Best Estimate Cashflow Simulation

## Related Information

For more information about SAP S/4HANA for financial products subledger ([https://help.sap.com/viewer/product/S4HANA\\_FIN\\_PROD\\_SUBLEDGER/latest/en-US](https://help.sap.com/viewer/product/S4HANA_FIN_PROD_SUBLEDGER/latest/en-US)), see *Sample Content for Estimated Cash Flow Preparation*.

For further preparatory steps, see “Preparatory Processing”.

## 1.2.10 Activation of Functions, Process Templates and Environments

SAP Profitability and Performance Management clearly separates the design of a model from its execution. A model is designed in the modeling environment application. This is sometimes also referred to as Customizing.

The *Activate* button in the modeling environment is used by the modeling user to trigger the generation of all the required artifacts once a function or a process is designed and ready. This activation is a mandatory step that ensures that the function or process template is ready to be executed.

The following *Activate* buttons exist:

1. *Activate* button in the Calculation Unit function  
This activation goes through the entire environment and activates everything that is required. This means that afterwards in the processes application, new processes with activities can be deployed and execution users can work on these processes and activities. If you set up a new process template or change an existing process template configuration and its underlying activities, this activation has to be triggered.
2. *Activate* button for individual function  
This activation activates an individual function, including any required sub-functions and underlying data model functions. The main purpose here is to allow the modeling user to test and run the function directly from within the modeling environment by choosing the *Run* button for that particular function.
3. *Activate* button in function hierarchy  
If the modeling user has selected multiple functions in the function hierarchy, this *Activate* button calls the activation for every function that has been selected. The main purpose is the same as for the individual function.

## Related Information

For more information about SAP Profitability and Performance Management, see [Functions \[page 79\]](#).

## 1.3 Functions

Financial and business models consist of functions that are connected to each other by means of input-output relationships.

The output of one function can be the input of multiple other functions, and in this way complex calculations and logic can be modeled in a comfortable way.

The following functions are available:

Key Feature	Use
Allocation	Performs direct and indirect allocations
Calculation	Performs mathematical formulas
Calculation Unit	Encapsulates a group of functions and makes them reusable
Conversion	Performs currency and unit conversions
Derivation	Performs if-then-else enrichments of data

<b>Key Feature</b>	<b>Use</b>
Description	Describes processes and topics used for the documentation of models
Environment	Registers all required fields and the connection to the database
File Adapter	Provides automated access to files
Flow Modeling	Provides calculation for the best-estimate cash flow (BECF)
Funds Transfer Pricing	Performs funds and liquidity transfer pricing calculations
Join	Performs collections, joins, unions and lookups for separate data
Machine Learning	Provides a rule type to train and uses a time-series forecast model based on input data
Model BW	Provides read and write access to a local BW InfoSource like Advanced DSOs
Model RDL	Provides read and write access to a local FRDP Results Data layer
Model Table	Provides read and write access to a local or remote data table
Model View	Provides read access to a local or remote data table or view
Query	Allows the output and input of data
Remote Function Adapter	Performs an ABAP-based remote function call (for example, a call to a remote FI-GL posting BAPI)
Transfer Structure	Performs a transfer from accounting-based data to costing-based data (also called denormalization)
Valuation	Performs comprehensive calculations with different valuation methods (for example, discounting)
View	Projects or aggregates data, including filtering options and formulas
Writer	Stores data in a model table, Model RDL or model BW



## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.1 Environment

An environment comprises the information and calculation details for a financial and business model. A modeling environment in SAP Profitability and Performance Management can have multiple environments and also an environment can have multiple (unique) versions.

#### Key Features

##### Environment Details

Environment details can be reached from the modeling environment and contain settings that are valid for all functions in the environment.

##### Fields

Fields are the basis for every function and can be divided into two different groups of field types.

In the first group, the fields are owned by, created and managed within an environment. The complete definition of a field, including data types, master data and hierarchies, is maintained by the modeling users. This includes the transportation through the system landscape and ensures that these fields are available in transport target systems as well.

In the second group, the fields are owned and managed by an external non-SAP or SAP application. The complete definition, including data type, master data and hierarchies, is therefore done "outside", and the environment with all its functions reuses these metadata definitions without being able to influence them. This is key when models are integrated with other applications and ensures consistency.

The following field types are available:

- **Environment InfoObjects**  
Data types, master data and hierarchies are maintained by the modeling user. Fields are visible to other environments in all clients of the same system. Fields can refer to other InfoObjects that share metadata.
- **Environment Fields**  
Data types are maintained by the modeling user. Fields are visible only in the environment in which the field is defined.  
Usage of virtual hierarchies is not supported by SAP S/4HANA.
- **BW InfoObjects**  
Data types, master data and hierarchies are maintained by an external application and are used in the environment as part of the model. The fields are therefore registered in the environment, and refer to their original source.  
Usage of virtual hierarchies is not supported by SAP S/4HANA.

- **BW Fields**  
These are similar to BW InfoObjects, but no master data or hierarchies are available in the source.
- **DDIC Fields**  
Data types and master data are maintained by an external application and are used in the environment as part of the model. The fields are therefore registered in the environment, and refer to their original source.
- **HANA Fields**  
Similar to DDIC fields, but no master data is available from the source.

Across the different field types, there are the following field categories:

1. **Key Figures**  
Key figures are used for calculations and can contain natural numbers, integers, decimals or floating points. In formulas, mathematical operations can be applied to key figures.
2. **Characteristics**  
Characteristics are used to identify key figures and contain texts, codes, dates or numerical characteristic values. In formulas, data-type-specific operations can be applied (for example datetime functions, text functions).
3. **Unit**  
Units are required to give meaning to the values for the key figures. Key figures of the type “amount” are always assigned a currency key, and key figures of the type “quantity” are also assigned a unit of measurement.

Fields can be used in the following ways:

1. **As parameters:**  
Parameters are used to steer processes and calculations. Therefore, they cannot be part of a data model, but can be used in formulas and calls of certain functions to influence the logic and operations applied there. A typical example of a parameter is a flag, which allows a calculation to be skipped or executed in a process.
2. **As fields:**  
Fields can be used in all functions to work on data. A typical example of a field is a financial period, which identifies for which month the data is valid.

## Checks

Each function includes built-in system checks to detect inconsistencies in the result data during an execution run.

Modeling users can also define custom checks here and register them in one or more functions later so that they are applied during an execution run on the results.

Custom checks use selection conditions to detect specific records in the result data and append a message text and a message type to the application log.

## File Formats

File format definitions are centrally maintained in the environment and are referred to by File Adapter functions for data import and export.

## Conversion Types

Conversion type definitions are needed by the conversion function for currency and unit conversions.

## Partitionings

Partitionings can be used to enable and define the package parallel processing of data.

## Advanced Settings

The advanced settings allow you to define standard integration scenarios:

1. DB Connection Name  
Here, you need to register the NetWeaver database connection to the underlying SAP HANA database. Since a user and password is always attached to a DB connection, it indirectly specifies the authorizations and therefore which data and views are available. By default, this is the standard DBCON connection.
2. RFC Destination  
This allows you to connect an SAP ERP or SAP S/4HANA system to SAP Finance and Controlling. If this is set, the allocation function can read and reuse the allocation rule Customizing from the (remote) system (for example, to simulate allocations on general ledger data and rules).

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

## 1.3.2 Calculation Unit

A calculation unit represents a financial or business unit on which calculations and analysis like financial closing can be performed independently of other calculation units.

A calculation unit is a container function that defines the functions relevant for execution using process templates and activities. It also specifies the parameters and selection fields that are required for execution.

From a modeling perspective, it is a collection of objects, such as fields and functions.

## Key Features

### Process Templates and Activities

Process templates are used to specify the set of functions that are relevant for execution. A process template is structured by one or many activities which have to be executed to finish the process.

Process templates and activities are defined by the following information:

1. Process Template ID  
Calculation unit-wide unique ID of a process template that can be referred to in process management to instantiate processes.
2. Description  
Short description of the purpose of a process template.
3. Process Template State  
The process template state can be set to either inactive or active. Inactive process templates are not ready to be deployed. Active process templates can be deployed and instantiated as a process in process management, and can therefore be used to run processes in production.

4. Process Type  
Can be set to either "Simulation" or "Run". If the process type is set to "Simulation", all parameters and field selections can be changed at any time to allow what-if simulation. If the process type is set to "Run", all parameters and field selections have to be fixed during the deployment and cannot be changed during what-if simulation.
5. Activity ID  
Process template-wide unique ID of an activity that can be referred to in report elements.
6. Description  
Short description of the purpose of an activity.
7. Activity Type  
The activity type can be set to either "Input/Output" or "Execution". "Input/Output" allows you to look at the data of a function, typically a query function. "Execution" allows you to trigger the execution of a function.
8. Level  
The level defines which activities depend on each other.  
You can set up activities in hierarchical form by using the *Same Level* or *One Level Below* options when you add them on the *Activities* tab.  
For example, the activity *Review Actual Data* with level 1 can be worked on immediately after process deployment, but the activity *Execution Calculation* underneath it with level 2 will remain pending until the activity *Review Actual Data* is finished.
9. Start Date and End Date  
In both fields, you need to define default values. These can be overwritten during process deployment.
10. Performer and Reviewer  
The performer defines a team (group of users) that can work on an activity. The reviewer can also define a team that has to review the activity in a workflow using the *Dual Control [page 88]* principle and can either approve or reject it.

## Parameters

Parameters are defined in the environment and can be registered here so that they are available for use in process templates.

Parameters can influence the behavior of functions below the calculation unit at runtime. For example, you want to analyze the profitability of an organizational unit every quarter using an assumed sector growth rate %. For this, the modeling user can design a business model on the basis of a *Period* parameter. The financial analyst (execution user) can specify this parameter value during the deployment of a process.

## Selection Fields

Fields are defined in the environment and can be registered here as selection fields so that they are available for use in process templates. Selection fields can filter the input data of functions below the calculation unit at runtime. For example, you want to analyze the profitability of an organizational unit every quarter. For this, the modeling user can design a business model on the basis of a *Period* parameter. The financial analyst (execution user) can specify this parameter value during the deployment of a process.

## Business Event Fields

Fields are defined in the environment and can be registered here as business event fields so that the event and error handling for all functions in the calculation unit is done at that common level. If no business event fields are registered, error and event handling is done for the individual fields of each function.

## Documentation

User-specific inline documentation can be entered here to describe which settings were made and why.

## Procedures

In the client where SAP Profitability and Performance Management is installed, choose **SAP Menu** > **Profitability and Performance Management** > **Modeling** > **Start My Environments**.

The *Environment* screen appears in a separate browser window. Here you can create or choose an existing environment and continue. Once you are in the environment, choose the calculation unit and switch to *Edit* mode, which allows you to perform the following activities:

### Process Template

#### Adding a Process Template

1. Choose *Add (+)*,
2. The *Add Details* window appears, and you can enter the attributes of the process template that you want to create.
3. Choose *OK* and the created process template is displayed.

#### Removing a Process Template

1. Select an existing process template that you want to delete.  
Choose *Remove (-)*.  
Choose *Save* after the deletion to finalize the changes you have made.

#### Copying a Process Template

You can copy an existing process template and thereby change the process template ID, description and target position. The system also copies all activities and selections present in the source template .

1. Select the existing process template you want to copy.
2. Choose *Copy*.
3. The *Copy Details* window appears.
4. Enter the new process template ID, description and target position.,
5. Choose *OK*.
6. Choose *Save* once you have copied your entries to finalize the changes you have made.

### Activities Inside Process Templates

Once you have selected the process template in edit mode, you can perform the following activities on the *Activities* tab:

#### Adding an Activity

1. Choose *Add (+)*,
2. The *Add Details* window appears.
3. When you have made your entries for the fields, choose *OK*.
4. Choose *Save* to complete the changes you have made.

#### Removing an Activity

1. Choose an existing activity that you want to delete.
2. Choose *Remove (-)*.
3. Choose *Save* after you have deleted to save the changes you have made.

### Copying an Activity

You can also copy an existing activity. Parameters and Selections from the copied activity are then transferred to the new activity.

1. Select an existing activity that you want to copy.
2. Choose *Copy*
3. The *Copy Details* window appears.
4. Once you have entered the new activity ID, description and target position, choose *OK*.
5. Choose *Save* to save the changes you have made.

### Selections Inside Process Template

Within the process template, you can process the following activities on the *Selections* tab in edit mode:

- Parameters
  - **Add**
    1. Choose *Add (+)*.
    2. The *Fields* window appears.
    3. The system lists the available parameters. Select the desired parameter and choose *OK*.
    4. Choose *Save* to complete the changes made.
  - **Remove**
    1. Choose a parameter that you want to delete.
    2. Choose *Remove (-)*
    3. Choose *Save* to save the changes you have made.
  - **Formula**

Use this function to set the desired parameter value.

    1. Select a parameter and choose *(F)*.
    2. The *Formula* window appears.
    3. Enter the required value for the parameter and choose *OK*.
    4. Choose *Save* to save the changes you have made.
- Selection Conditions
  - **Add**
    1. Choose *Add (+)*.
    2. The *Fields* window appears.
    3. Select the required selection field and choose *OK*.
    4. Choose *Save* to save the changes you have made.
  - **Remove**
    1. Select a field that you want to delete.
    2. Choose *Remove (-)*.
    3. Choose *Save* to save the changes you have made.
  - **Filter**
    1. Select a field to configure the selection.
    2. Choose *Filter*.
    3. The *Selection Condition* window appears.

4. Enter the required selection value for the field using operators.
  5. Choose *OK*.
  6. Choose *Save* to save the changes you have made.
- **Paste Filter Selection**  
You can use this function to copy a filter selection from the reporting application.
    1. Select an existing field.
    2. Choose *Paste Filter Selection*.
    3. The system copies the selection filter to the selected field.

## Parameters

### Adding a Parameter

1. Choose *Add (+)*.
2. The system displays the parameter fields available in the environment.

#### i Note

The parameters declared here are listed in the *Activity Selection* section.

3. Select the required parameter field and choose *OK*.

### Removing a Parameter

1. Select the parameter that you want to delete.
2. Choose *Remove (-)*.
3. Choose *Save* to save the changes you have made.

## Selection

### Adding a Selection

1. Choose *Add (+)*.
2. The system displays the fields available in the environment.

#### i Note

Fields declared here are listed in the *Activity Selection Condition* section.

3. Select the required field and choose *OK*.

### Removing a Selection

1. Select the field you want to delete.
2. Choose *Remove (-)*.
3. Choose *Save* to save the changes you have made.

## Business Event Fields

### Adding a Field

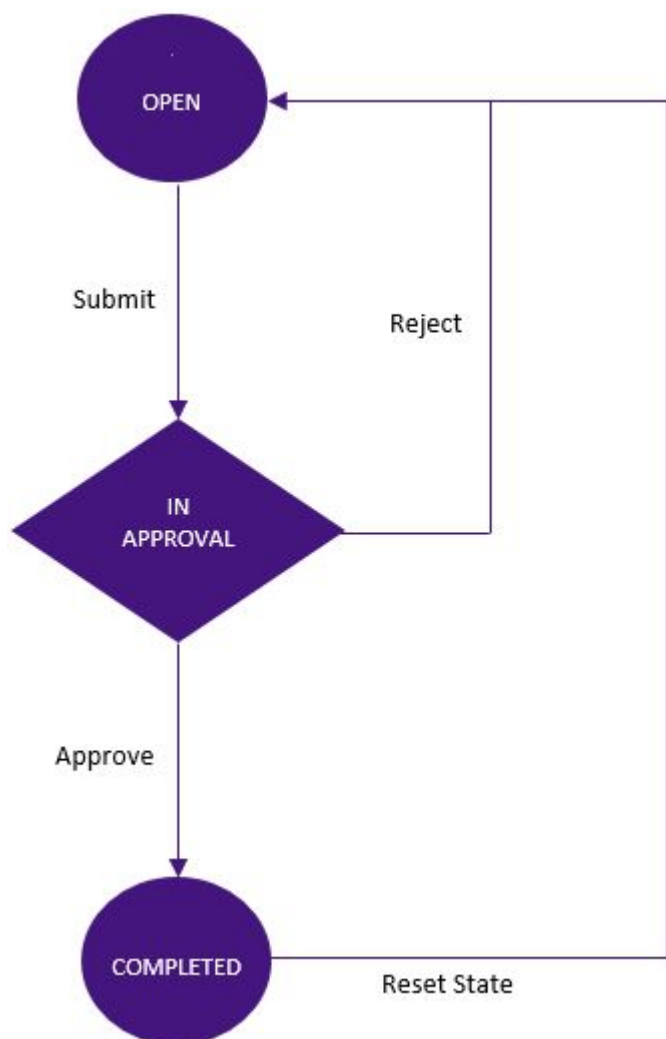
1. Choose *Add (+)*.
2. The *Add Fields* window appears. It displays the fields available in the environment.
3. Select the required field and choose *OK*.

### Removing a Field

1. Select the field that you want to delete.
2. Choose *Remove (-)*.
3. Choose *Save* to save the changes you have made.

### 1.3.2.1 Dual Control

Dual control enables two user groups (performer and reviewer) to execute the actions *Submit*, *Approve*, *Reject* and *Reset State* for activities according to the Dual Control Workflow shown below:



#### i Note

Once an activity is rejected, the system resets the status to “Open”, and you need to change it before you can submit the activity again.



If an activity has a defined previous activity, the system sets the parent activity state to “Pending” until the previous activities are completed. Afterwards, the system follows the flowchart. The example below gives a step-by-step overview of how the status of the activities changes:

Initial State

Activity	Status	Previous Activity
A0001	Pending	A0003
A0002	Pending	A0003
A0003	Pending	A0004
A0004	Open	

When the performer submits activity A0004 for review, the status changes from “Open” to “In Approval”:

Activity	Status	Previous Activity
A0001	Pending	A0003
A0002	Pending	A0003
A0003	Pending	A0004
<b>A0004</b>	<b>In Approval</b>	

As soon as activity A0004 has been approved, the status changes from “In Approval” to “Completed”. The status of activity A0003 is now “Open”:

Activity	Status	Previous Activity
A0001	Pending	A0003
A0002	Pending	A0003
<b>A0003</b>	<b>Open</b>	A0004
A0004	Completed	

When the performer submits activity A0003 for review, the status changes from “Open” to “In Approval”:

Activity	Status	Previous Activity
A0001	Pending	A0003
A0002	Pending	A0003
<b>A0003</b>	<b>In Approval</b>	A0004
A0004	Completed	

As soon as Activity A0003 has been approved, the status changes from “In Approval” to “Completed”. The status of the activities A0001 and A0002 is now “Open”:

Activity	Status	Previous Activity
A0001	Open	A0003
A0002	Open	A0003

Activity	Status	Previous Activity
<b>A0003</b>	<b>Completed</b>	A0004
A0004	Completed	

When the performer submits activity A0001 for review, the status changes from “Open” to “In Approval”:

Activity	Status	Previous Activity
<b>A0001</b>	<b>In Approval</b>	A0003
A0002	Open	A0003
A0003	Completed	A0004
A0004	Completed	

When the performer submits activity A0002 for review, the status changes from “Open” to “In Approval”:

Activity	Status	Previous Activity
A0001	In Approval	A0003
<b>A0002</b>	<b>In Approval</b>	A0003
A0003	Completed	A0004
A0004	Completed	

As soon as activity A0001 has been approved, the status changes from “In Approval” to “Completed”:

Activity	Status	Previous Activity
<b>A0001</b>	<b>Completed</b>	A0003
A0002	In Approval	A0003
A0003	Completed	A0004
A0004	Completed	

As soon as activity A0002 has been approved, the status changes from “In Approval” to “Completed”:

Activity	Status	Previous Activity
A0001	Completed	A0003
<b>A0002</b>	<b>Completed</b>	A0003
A0003	Completed	A0004
A0004	Completed	

### i Note

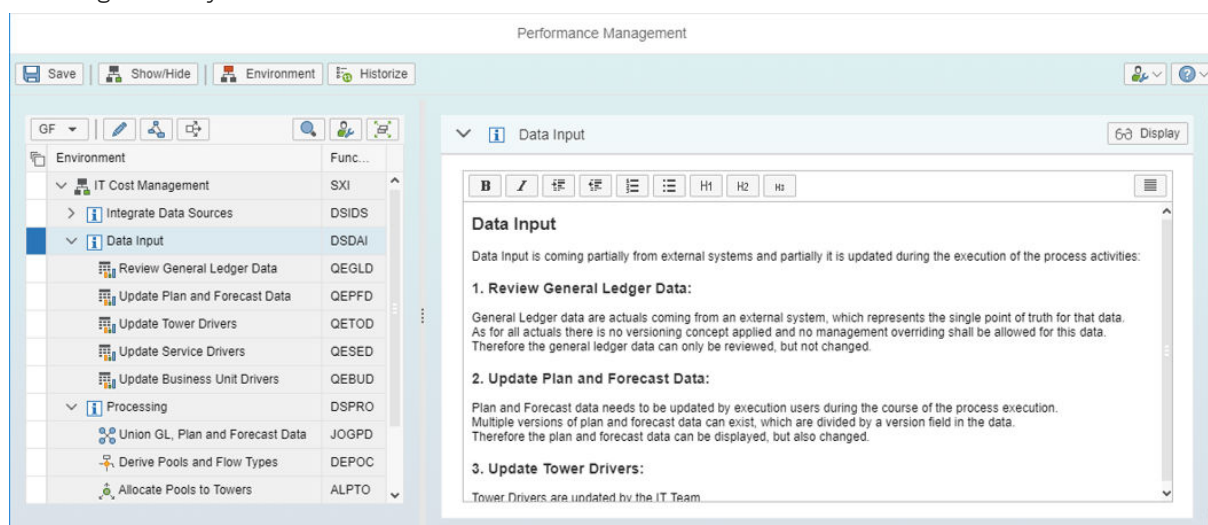
SAP Profitability and Performance Management sends email notifications in the following cases:

- Activity started, receiver is performer team
- Activity sent for approval, receiver is reviewer team

- Activity approved, receiver is both performer and reviewer team
- Activity rejected, receiver is both performer and reviewer team.

## 1.3.3 Description

A function that provides information, usually used to explain or provide details about a function or hierarchy of functions. It is used for the inline documentation of models as well as to structure other functions in the modeling hierarchy.



In the example above, several description functions are used to structure the IT Cost Management model into Integrate Data Sources, Data Input, Processing and so on. For the Data Input description function, modeling users maintained detailed documentation about the purpose of the functions underneath.

### Note

The *Description* function is also a part of each function. You can find it on the *Documentation* tab.

## Documentation Editor

You can use the editor and its simple formatting options to provide descriptions and documentation about the function configuration and why things were modeled this way.

You can also use the following buttons to format the texts in the documentation editor:

Button	Use
Bold	Darkens the text to emphasize it.
Italic	Formats the text in a slightly slanted way to set it apart.

Button	Use
Increase Indent	Increases space between the current paragraph and the left page margin.
Decrease Indent	Decreases space between the current paragraph and the left page margin.
Numbering	Creates an ordered list for steps or checklists.
Bullets	Creates an unordered list to denote significance.
Heading 1, Heading 2 and Heading 3	Formats text as a (section) title.

## Related Information

For more details about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.4 Allocation

The *Allocation* function is used to distribute key figures from one entity to another using a distribution base.

The entity from which key figures are distributed is known as the sender. The sender key figures represent the values to be allocated by the allocation function.

The entity that receives the distributed key figures is known as the receiver. One or more key figures from the receiver constitute the distribution base or bases.

The following table explains the key features available:

## Key Features

### Header

In the header, you define the principal behavior of the allocation.

- Allocation Type

In allocation, you can use the following allocation types according to your needs:

1. Allocation

Key figures of the sender entity are distributed to a receiver entity, and these distribution records are the result of the allocation. The key figures of the sender entity are not affected. This type of allocation is typically used in top-down distribution allocations, activity-based costing allocations and other allocations where iterations or postings of allocation results are not necessary.

2. Allocation with Offset Records

Key figures of the sender entity are distributed to a receiver entity, and these distribution records are the result of the allocation. In addition, offset records are created at the granularity of the sender entity but with opposite signs of the key figures. This type of allocation is typically used in assessment

allocations and can be iterative, where the sender needs to be reduced by the allocated key figures and the receiver is enhanced, so that the overall sum of the key figure values stays the same.

### 3. Allocation with Detailed Offset Records

This allocation type is similar to "Allocation with Offset Records". However, here the offset records are created using the sender and receiver entity dimensions to produce more details. This means that information about which fraction of the sender entity key figures were distributed to which receiver record is stored in the results. This type of allocation is typically used in distribution allocation to provide traceability from sender to receiver. It can also be used for iterations.

- Iterative:

The *Iterative* indicator is available when you choose the allocation types "Allocation with Offset Records" or "Allocation with Detailed Offset Records". It is not available for the allocation type "Allocation".

If you set the *Iterative* indicator to "Yes", the allocation process is executed iteratively. This action also enables the *Iterative Processing* function on the *Advance* tab of allocation.

If the modeler selects iterative processing, the system repeats the allocation process, using allocation results from the previous iteration as the sender for the next iteration. Iterations are repeated until there are no senders to be allocated or the exit condition defined in the advanced allocation settings is fulfilled. The modeler defines the exit condition using the early exit check and/or the cycle maximum value. Iteration is repeated until one of these conditions are fulfilled.

- Periodic

The modeler uses the *Periodic* indicator to specify whether the allocation process is executed based on a defined period or time interval.

If the modeler selects periodic processing, the system runs the allocation process using the period/time interval defined in the allocation settings in the *Advanced* tab of allocation.

In the advanced allocation settings, the modeler can specify the fiscal year and period intervals. The modeler can also choose whether periodic processing is cumulative.

If the *Periodic* option is not meant to be used, the modeler must set it to "No".

- Value Adjustment

During the allocation process, there can be a difference in the sender amount and total receiver amount (to which the given sender amount was distributed) as a result of rounding behavior. These differences in value can be compensated using one of the following financial value adjustment options:

1. No Adjustment

The system does not adjust the difference in value. This is typically used in planning-only scenarios with high values, where "a missing cent" is not relevant.

2. Last Row

The system adds the value difference to the last receiver (corresponding to the specified sender).

3. Biggest Value Row

The difference in value is added to the receiver (corresponding to the specified sender) with the highest allocated amount.

4. Absolute Biggest Value Row

The difference in value is added to the receiver (corresponding to the specified sender) with the highest allocated amount, not taking the +/- sign into account.

All other header options such as *Include original Input Data*, *Result Handling*, *Suppress initial Results* and *Result Model Table* are called "Functional Building Blocks" and are not specific to allocation. For more information about these options, see [Header \[page 63\]](#).

## Sender, Receiver and Reference

On the *Sender* and *Receiver* tab, you define the input for the allocation function with an option to enrich the data before the allocation process. Typically, the sender points to G/L data and the receiver points to driver

data. The *Reference* tab is optional and reuses the receiver input, but allows you to define separate selection criteria. For example, the receiver data from the current period can be considered during the allocation process, while the assigned driver value is disregarded, and the driver value from the previous period is considering.

To add a selection criteria on the *Reference* tab, proceed as follows:

1. In edit mode, go to the *Reference* tab of your allocation function, then choose *Add Field*.
2. The *Add Fields* pop up window appears. Choose the characteristic field that you want to use to add to the selection, then choose *OK*. The characteristic field should be available in the receiver table.
3. Choose the selection condition and define the selection criteria.

## Rules

Each allocation rule line defines one segment of an allocation. For each allocation rule, the sender and receiver rules need to be specified.

To do this, proceed as follows:

1. On the *Rule* tab of your allocation function, choose *Add*.
2. An *Add Detail* popup window appears. Enter the following information:
  1. Add Level  
If you choose the “Same Level” option , the rule is processed immediately.  
If you choose the “One Level Below” option, the allocation result from the parent rule is used as the sender by the child rule(s). The final allocation result contains the allocation result derived from the lowest-level child rules.

### i Note

You can define allocation rules hierarchically by choosing the “One Level Below” option.

2. Rule  
This is a unique technical ID that must be provided by the modeler. Do not use blank spaces. For example, `ALLOC_COST` is a good rule ID whereas `ALLOC COST` is not. Once you have created the rule, you can no longer edit the entry.
3. Description  
Add a description to enable you to distinguish between rules. You can edit the description later if required.
3. Choose a Rule Type:
  1. Direct Rule Type  
Allocates records from sender to receiver with the same characteristics.
  2. Indirect Rule Type  
Allocates records from sender to receiver according to distribution criteria.
  3. Indirect Detailed Rule Type  
Allocates records from sender to receiver according to distribution criteria comprehensively.
4. Adjust the *Sender Rule* tab:
  1. Sender Rule  
Define the values or amounts to be allocated based on the setting configured. The default option, *Posted Amount*, uses the sender input or the result of an allocation rule at a lower level.
  2. Sender Share  
Defines the percentage of the value that is allocated from the sender. Usually this is 100% so that the full value from the sender is allocated.

3. Sender Value Fields:

Defines the value fields that have to be allocated from the sender to the receiver.

4. Mapping Method:

When senders are allocated to receivers in direct allocation, receivers are matched based on one of the following mapping methods:

○ Empty as value:

Empty characteristics from the sender are matched only to empty characteristics of the receiver.

○ Empty as any value:

Empty characteristics from the sender are matched to any value in the characteristics of the receiver. Characteristic values from the receiver are ignored if the characteristics are empty in the sender.

In other words, the receiver characteristic value is not taken into consideration when the system allocates amounts from the sender if the sender's characteristics are empty.

5. Subview:

You can apply further selections, formulas and groupings here if needed.

5. Adjust the *Receiver Rule* tab:

1. Receiver Rule:

Define how the values from the sender are to be allocated or distributed by choosing one of the following settings:

○ Variable Portions

The receiver input is used and the distribution base acts as a driver.

○ Variable Percentages

The receiver input is used and the distribution base acts as an allocation percentage.

○ Variable Factors

The receiver input is used and the distribution base acts as an allocation factor.

○ Variable Even

The receiver input is used and no distribution base is needed because the sender is evenly distributed.

2. Scale:

○ No scaling:

The distribution base is not scaled before it is applied.

○ Standard scaling:

- If the sum of receiver tracing factors is greater than or equal to zero, the largest negative tracing factor is set to zero.

The other tracing factors are increased accordingly.

- If the sum of the receiver tracing factors is zero, the largest positive tracing factor is set to zero.

The other tracing factors are decreased correspondingly.

○ Absolute value:

With negative receiver tracing factors the +/- sign is reversed. All the receiver tracing factors are therefore positive.

○ Negative tracing factors to zero:

Negative tracing factors are set to zero.

○ Smallest negative tracing factor to zero:

The smallest negative tracing factor is set to zero. All other tracing factors are increased correspondingly.

○ Smallest negative tracing factor to zero, but zero = zero:

The smallest negative tracing factor is set to zero. All other tracing factors are increased correspondingly. Receivers that used tracing factor "0" before scaling retain the zero.

3. Distribution Base  
Is the basis for the allocation. The specific treatment is dependent on the receiver rule (see above).
4. Driver Result  
If you make an entry here, the allocation calculates the percentage portion based on the distribution base value and retains the driver percentage in the allocation result. These percentage portions are often easier to read by business users than distribution bases, which sometimes have quite small or large values.
5. Subview:  
You can apply further selections, formulas and groupings if needed.

Further options include the scaling of driver values, the distribution base definition as a field or formula and the option of assigning a driver result field to retain the driver percentage in the allocation result.

## Advanced

The following settings are relevant for periodic processing:

1. Iterative Processing  
If iterative allocation is set to "Yes" in the header, you need to make additional settings on the [Advanced](#) tab.  
The following settings are relevant for iterations:
  - Iteration Counter  
If you register a field in this optional setting, the result shows which records have been allocated in which iteration cycle. As a prerequisite, the field has been defined as an action field on the [Signature](#) tab of the allocation.
  - Cycle Maximum Value  
The number of iterations is limited to the value entered (for example, 100). You can also use a formula, field or parameter to have a more flexible value set.
  - Early Exit Check  
If you have registered a check in this optional setting, it is applied after each iteration cycle, and if the check conditions are fulfilled, the iteration stops. This is helpful if you want to apply a threshold, like Amount < 10 USD, below which the iteration cycle stops.
2. Periodic Processing  
If periodic allocation is set to "Yes" in the header, you need to make additional settings on the [Advanced](#) tab. The following settings are relevant for periodic processing:
  - Periodic Counter  
This field in the output provides information about the period for which a particular allocation record is created in periodic processing.  
As a prerequisite, the field has been defined as an action field in the [Signature](#) tab of the allocation.
  - Fiscal Year  
This field contains financial year information in the sender and/or receiver data.  
As a prerequisite, the field has been defined as a selection field in the [Signature](#) tab of the allocation.
  - Fiscal Year Value  
This is the financial year for which the allocation is executed. The value you enter here is used to restrict sender and receiver data for a given financial year. You can also use a formula, field or parameter to have a more flexible value set.
  - Period  
This field contains term or timeframe information in the sender and/or receiver data.  
As a prerequisite, the field has been defined as a selection field in the signature of the allocation.



- First Period Value  
Period signifying the start of the analysis or processing timeframe.  
The *First* period field contains the first period for which the allocation can be executed. You can also use a formula, field or parameter to have a more flexible value set.
- Last Period Value  
Period signifying the end of the analysis or processing timeframe.  
The *Last* period field contains the last period for which allocation can be performed.
- Specific Periodic Processing  
When you carry out periodic allocation, you can choose the following options for special processing:
  - None  
In periodic processing, senders from a given period are allocated to receivers from the same period.
  - Cumulation Indicator  
You use this to specify whether the cumulation effect applies to the periodic allocation process. If this indicator is set, the application allocates the sender amounts to receivers posted up to and including the current period. This is based on allocated tracing factors accumulated from the first period onward. The application also accumulates the allocation amounts it has determined and posts them in the current period, minus the amounts allocated in the prior periods.
  - Last Periods  
The system first allocates the senders from a given period to suitable receivers from the same period. If no receivers are found in the same period for some of the senders, the system tries to find suitable receivers for those senders from the preceding period.  
If the system still finds no receivers in that period for some of the senders, it tries to find suitable receivers for the senders from two periods before.  
This process is continued until all senders are allocated or until the maximum number of periods defined for the last periods processing is reached.

### 3. Offset Mapping

Offset mapping allows you to define characteristics or settings for offset records generated during the allocation process.

If allocation is relevant for offset data (for example, the allocation type is "Allocation with Offset Records" or "Allocation with detailed Offset Records"), offset data is added to the allocation result. Offset mapping is also used in iterative allocation to define the field mapping to determine sender data from the allocation result of the previous iteration. For example, the receiver cost center in the allocation result of the previous iteration is mapped as the sender cost center in the sender data for the next iteration.

You can specify the following types of offset mapping:

1. Offset:  
Field mapping. In other words, the field and relevant offset field is defined.
2. Debit/Credit  
The field that contains debit/credit information is selected. Values for the debit and credit sign are also set.

## 1.3.4.1 Example: Indirect Allocation with Offset Records

Sender

Cost Center	Amount
CC01	200
CC02	300
CC03	400

Receiver

Contract	Product	Distribution Rate
DD01	A100	20
DD05	A100	40
DD03	B200	10
DD04	A100	60
DD02	B200	30

Result

Amount	Cost Center	Contract	Distribution Rate	Product	Portion
25	CC01	DD01	20	A100	0.125
50	CC01	DD05	40	A100	0.25
12.5	CC01	DD03	10	B200	0.062
75	CC01	DD04	60	A100	0.375
37.5	CC01	DD02	30	B200	0.187
37.5	CC02	DD01	20	A100	0.125
75	CC02	DD05	40	A100	0.25
18.75	CC02	DD03	10	B200	0.062
112.5	CC02	DD04	60	A100	0.375
56.25	CC02	DD02	30	B200	0.187
50	CC03	DD01	20	A100	0.125
100	CC03	DD05	40	A100	0.25
25	CC03	DD03	10	B200	0.062
150	CC03	DD04	60	A100	0.375
75	CC03	DD02	30	B200	0.187
-200	CC01		0		0

Amount	Cost Center	Contract	Distribution Rate	Product	Portion
-300	CC02		0		0
-400	CC03		0		0

1. Amount in sender will be allocated proportionally using *Distribution Rate* in receiver as the distribution percentage.
2. In order to get the portion percentage:
  1. Get the total of *Distribution Rate* from the receiver (*Distribution Rate* = 160).
  2. *Portion* is the quotient when you divide *Distribution Rate* by the total of distribution rates (for example, 20 / 160 = 0.1250).
  3. Allocate *Amount* to the receiver by multiplying *Amount* with *Portion*. (200 \* 0.1250).
3. The negative entries in the *Amount* column are the allocated amounts that came from the sender.

### 1.3.4.2 Example: Direct Allocation with Unassigned Items

Sender

Product	Channel	Customer	Amount	Financial Period
238	92H2	AA	300	1
224	92H2	DD	200	2
238	92H2	AA	400	3
224	92H2	DD	400	4
239	92H3	CC	1,000.00	5

Receiver

Product	Coverage	Channel	Customer	Distribution Rate
224	6981	92H2	DD	60
224	6982	92H2	DD	40
238	6985	CXH0	DD	55
238	6986	CXH0	DD	45
238	6989	92H2	AA	20
238	6990	92H2	AA	80

Result

Channel	Coverage	Customer	Distribution Rate	Financial Period	Product	Amount	Portion
92H2	6989	AA	20	1	238	60	0.2
92H2	6990	AA	80	1	238	240	0.8
92H2	6982	DD	40	2	224	80	0.4

Channel	Coverage	Customer	Distribution Rate	Financial Period	Product	Amount	Portion
92H2	6981	DD	60	2	224	120	0.6
92H2	6989	AA	20	3	238	80	0.2
92H2	6990	AA	80	3	238	320	0.8
92H2	6982	DD	40	4	224	160	0.4
92H2	6982	DD	60	4	224	240	0.6
Unassigned Item							
239	92H3		CC		1,000.00	5	

1. Amount in sender will be allocated proportionally using *Distribution Rate* in receiver as the distribution percentage.
2. In order to get the portion percentage:
  1. Group *Customer*, *Channel* and *Product*. These are the fields that have the same characteristics from our sender and receiver.
  2. Get the total of *Distribution Rate* of the grouped *Customer*, *Channel* and *Product* (*Distribution Rate* = 160).
  3. Divide *Distribution Rate* by the total of distribution rates (for example  $20 / 100 = 0.2$ ).
  4. Allocate *Amount* to the receiver by multiplying *Amount* with *Portion*. (for example  $300 * .2 = 60$ ).
3. The 4 entries will be allocated directly but the 5th entry will not be allocated thereby producing an unassigned item.
4. The following error message will appear: "Processing Message "Unassigned Items" for Volume=1000 and Quantity=1".

### 1.3.4.3 Example: Simple Indirect Allocation (Indirect Allocation Using the "Reference" Tab)

Sender

Cost Center	Amount
CC01	200
CC02	300
CC03	400

Receiver

Contract	Product	Distribution Rate	Version
DD01	A100	20	1
DD05	A100	40	1
DD03	B200	10	1

Contract	Product	Distribution Rate	Version
DD04	A100	60	1
DD02	B200	30	1
DD01	A100	10	2
DD05	A100	20	2
DD03	B200	30	2
DD04	A100	40	2
D002	B200	50	2

Reference Tab

Field	Description	Selection
ZE_VER	Version	2

Result

Amount	Contract	Product	Distribution Rate	Portion
60	DD01	A100	10	0.07
120	DD05	A100	20	0.13
180	DD03	B200	30	0.2
240	DD04	A100	40	0.27
300	DD02	B200	50	0.33
-900			0	0

1. *Amount* in sender will be allocated proportionally using *Distribution Rate* in receiver as the distribution percentage.
2. In order to get the portion percentage:
  1. Get the total of *Distribution Rate* from the receiver (Distribution Rate = 150).
  2. *Portion* is the quotient when you divide *Distribution Rate* by the total of distribution rate (for example  $10 / 150 = 0.0666$ ).
  3. Allocate *Amount* to the receiver by multiplying *Amount* with *Portion* ( $900 * 0.0666$ ).
3. The negative entry in *Amount* column is the allocated amount that came from the sender.

### Note

You will notice that we used the distribution rate of Version 2 as we assigned the field *Version* and selected "2" on the *Settings* tab; thereby the allocation function used the data for Version 2 as its parameters.

## 1.3.5 Conversion

The conversion function covers two main types of conversion: currency conversion and unit of measurement conversion.

The *Currency Conversion* function can convert one currency into another country's currency. Based on current exchange rates, the value of the source amount may increase or decrease after the currency has been converted.

The *Unit Conversion* function can convert between different units of measurement for the same quantity based on the conversion factors. Conversion factors are used to change the unit of a measured quantity without changing its value.

### Key Features

#### Headers

All header options, for example *Include Original Input Data*, *Result Handling*, *Suppress Initial Results*, and *Result Model Table* are functional building blocks and are not specific to conversion. For more information about these options, see [Header \[page 63\]](#).

#### Rules

You can use two types of conversion - Unit Conversion and Currency Conversion. You can define the type of conversion by defining the line type within the rule.

#### Unit Conversion

Input Fields:

- *Conversion Type*: Defines the conversion of unit using a predefined conversion table (T006)
- *Value*: Amount which will be converted
- *Unit*: Source unit or source currency
- *Target Unit*: Defines the target currency

Output Fields

- *Flow Value*: Converted amount
- *Flow Unit*: Target unit or target currency

#### Currency Conversion

Input Fields

- *Conversion Type*: Defines the conversion rate, for example the middle rate, or the bid & ask rate for currency conversion
- *Value*: Amount which will be converted
- *Unit*: Source unit or source currency
- *Date*: Determines the date of conversion
- *Target Unit*: Defines the target currency
- *Timestamp*: Specifies the time stamp of processing

Output Fields

- *Flow Value*: Converted amount
- *Flow Unit*: Target unit or target currency

## Conversion Type Definitions

You need to define which references the system will use to run the conversion function for currency and unit conversions.

For example, category (unit or currency), tables (T006 or TCURR...), from which client, schema, conversion methods, rates, and market relevant data.

1. In the *Environment* section, choose *Conversion Types*.
2. Choose *Edit* in the upper right corner of the *Environment* section.
3. Choose *Add*, make entries in all the the required fields and choose *OK*.

Fields	Description
Conversion Types	This is a unique ID that must be provided by the modeler, for example "CONV0001".
Description	To distinguish between rules, enter a description. Once the description has been created, editing is still possible.
Category	Specify the conversion category (unit and currency).

The conversion function uses SAP S/4HANA Conversion, for more information about the other fields available in the Conversion definition, see:

- [CONVERT\\_CURRENCY Function](#)
- [CONVERT\\_UNIT Function](#)

The *Conversion* function uses exchange rates/conversion factors from standard SAP tables which are available on the Netweaver instance where SAP Profitability and Performance Management is deployed. It uses the following tables:

- **Currency**
  - TCURR: Used to store exchange rate data
  - TCURV: Used to store exchange rate types for currency translation
  - TCURX: Used to store decimal places in currency data
  - TCURN: Used to store quotation data
- **Unit**
  - TCURF: Used to store conversion factor data
  - T006: Used to store units of measurement data
  - T006D: Used to store dimension data

### 1.3.5.1 Example: Currency Conversion

The source currency will be converted from source currency (USD) to target currency(GBP)?.

Input Data

Company Code	Value Data	Source Currency	Unit	Target Currency	Unit
CC01	20010101	100.00	USD	0	GBP

Table TCURR

Client	ExRt	From	To	Valid From	Exchange Rate
800	M	USD	GBP	20010101	2.00-

Given the exchange rate (USD to GBP) = 2.000- the amount in the target currency is calculated as follows:  
(which basically has a (-) negative notation):

$$\text{Target Currency} = \text{Source Currency} / 2.0 = 1.00 / 2.000 = 0.5$$

Therefore, USD 1 = GBP 0.500 and USD 100.00 = GBP 50.00.

Result

Company Code	Value Data	Source Currency	Unit	Target Currency	Unit
CC01	20010101	100.00	USD	50.00	GBP

## 1.3.5.2 Example: Unit Conversion

The distance from point A to point B is 519 km and will be converted from unit "KM" to target unit "CM".

Input Data

Point A	Point B	Distance	Unit	Target Distance	Target Unit
AMSTERDAM	WALLDORF	519	KM	0	CM

Cl.	MU	3	6	DcR	C	V	1	2	Dimen.	Numerator	Denominat.	Exp	Add.const.	fpe	Dec	ISO	P
<input type="checkbox"/>	800	CM	X	X	3	X			LENGTH	1	100	0	0.000000	0	0	CMT	X
<input type="checkbox"/>	800	KM	X	X	0	X			LENGTH	1.000	1	0	0.000000	0	0	KMT	X

Table T006

Based on the values above, the function converts the unit from KM to CM as follows:

$$\text{Target Distance} = \text{Distance} * \text{Numerator} * \text{Denominator} = 519 * 1000 = 519000$$

Result

Point A	Point B	Distance	Unit	Target Distance	Target Unit
AMSTERDAM	WALLDORF	510	KM	519000	CM



## 1.3.6 Derivation

*Derivation* is a data enrichment function that can be used to enhance the data in a dataset with calculated attributes based on predefined rules at runtime. The enriched data can then be used for consumption in downstream processes such as allocation. If the data to be derived is already available in the source data, the derived data is only overwritten if the condition values are met. Otherwise, the source values are retained.

### Key Features

#### Header

In the header, you define the principal behavior of the derivation.

You can use the *Ensure Distinct Result* option with the following settings:

- Yes: Only the first successful derivation of overlapping derivation rules is included in the result and all subsequent matching derivations are excluded.
- No: All matching derivations of overlapping derivation rules are included in the result. This can lead to more result records than originally input.

Other header options, for example *Include Original Input Data*, *Result Handling*, *Suppress Initial Results* and *Result Model Table* are functional building blocks and are not specific to conversion. For more information about these options, see [Header \[page 63\]](#).

#### Rules

Each derivation rule semantically defines an if-then-statement. The if-part is maintained in the *Selection* section of a rule and the then-part in the *Action* section of a rule. In the if-part (Selection section), you specify which subset of the input data the rule applies to. In the then-part (Action section), all fields specified are then filled with configured or set values.

### Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

See also:

- [Example: Simple Condition \[page 106\]](#)
- [Example: Ensure Distinct Result \[page 106\]](#)

## 1.3.6.1 Example: Simple Condition

Input

Contract	Product 1	Origination Date	Amount	Premium
SUNSHINE	LIFE1	2016-01-01	600	0
SUNSHINE	LIFE2	2016-01-10	400	0
SUNSHINE	LIFE3	2016-01-15	500	0
MOONLIGHT	NONLIFE1	2016-01-04	200	0
MOONLIGHT	NONLIFE2	2016-01-21	300	0
MOONLIGHT	NONLIFE3	2016-01-29	100	0

Applying the following rule, we will get the result from the table below:

```
If Contract = SUNSHINE and Product 1 = LIFE1 then Premium = Amount
```

Result

Contract	Product 1	Origination Date	Amount	Premium
SUNSHINE	LIFE1	2016-01-01	600	600

## 1.3.6.2 Example: Ensure Distinct Result

Input: Customer-Branch Table

Branch	Customer	Customer Type	Deposit Amount	Interest Rate
B1	C1	New	5,000	0.01
B2	C2	Regular	10,000	0.015
B3	C3	Loyal	15,000	0.02
B4	C4	Regular	50,000	0.015
B1	C5	Regular	45,000	0.015
B1	C6	Loyal	120,000	0.02
B2	C7	Loyal	56,000	0.02
B3	C8	Loyal	70,000	0.02
B2	C9	New	105,000	0.01
B4	C10	New	80,000	0.01
B4	C11	Regular	60,000	0.015
B1	C12	Loyal	80,000	0.02

The system applies the following rule to derive the loyal customer from B1, and adds additional interest respectively to the derived values:

```
If Branch = B1 and Customer Type = LOYAL then Additional Interest = 0.0500
```

The deposit with interest is computed for the interim result:

Interim Result: First Rule

Branch	Customer	Customer Type	Deposit Amount	Interest Rate	Additional Interest	Deposit with Interest
B1	C6	Loyal	120,000	0.02	0.5	123,000
B1	C12	Loyal	80,000	0.02	0.5	82,000

Afterwards, the system applies the following rule:

```
If Customer Type = LOYAL and Deposit Amount > 50,000 then Additional Interest = 0.0250
```

Since B1 has already been derived, the system no longer includes it in the derivation. The system adds the additional interest respectively to the derived values and computes the deposit with interest:

Interim Result: Second Rule

Branch	Customer	Customer Type	Deposit Amount	Interest Rate	Additional Interest	Deposit with Interest
B2	C7	Loyal	56,000	0.02	0.25	57,260
B3	C8	Loyal	70,000	0.02	0.25	71,575

After the system has applied both rules, the final result is as follows:

Result

Branch	Customer	Customer Type	Deposit Amount	Interest Rate	Additional Interest	Deposit with Interest
B1	C6	Loyal	120,000	0.02	0.5	123,000
B1	C12	Loyal	80,000	0.02	0.5	82,000
B2	C7	Loyal	56,000	0.02	0.25	57,260
B3	C8	Loyal	70,000	0.02	0.25	71,575

## 1.3.7 Join

*Join* is a data access function that brings together the results of two or more other functions based on defined rules.

### Key Features

#### Header

In the header, you define the principal behavior of the join.

You can choose between different join types:

- **Implicit Fields**  
Fields coming from inputs are automatically considered during the join procedure. If the field is visible on multiple join rules (or inputs), you can use the “Auto Filling” function.
- **Explicit Fields**  
Fields of inputs are manually defined and maintained by the configurator. These fields will then be considered during processing.

#### Example

In the “Union All” join type, the modeler has to add all fields required for the output even if they are not part of the input data. The fields must be added on each subview of the rules to be unioned before activation.

The modeler can control the join results with the “Auto Filling” option. The following settings are possible:

- **No:** If a field is defined in a join rule in multiple inputs, the field content is taken from the first input that contains the field.
- **If Null then First to Last:** The first non-null value is taken and if all values are null, the initial value is returned for that field.
- **If Null/Initial then First to Last:** The first non-null and non-initial value is taken and if all values are null or initial, an initial value is returned for that field.

#### Note

##### **Null vs Initial**

In SAP Profitability and Performance Management UI, both NULL and INITIAL are represented by an EMPTY cell for the characteristic and “0” for the key figure. The modeler must take care when processing input data with these values, because the response is different:

- **NULL (?)** does not have any value and does not occupy memory. This can be achieved if a field was not assigned with values initially.
- **INITIAL (' ')** has a value and so occupies memory. In SAP Profitability and Performance Management a cell can have an INITIAL value if it was assigned with an empty record.

### Rules

Each join rule semantically defines the reading of a specific input.

Hierarchical join rules are also supported by assigning higher levels. The hierarchy of levels is resolved starting with the highest level, feeding as input to the lower levels and ending with level 0.

The following rule types are available:

1. *From*: This is always the first rule of a level.
2. *Left Outer Join*: This join type returns all rows from the rule above, and the columns and rows from this rule, where the predicates match.
3. *Inner Join*: This join type returns all rows when there is at least one predicate match in the rule above and this rule.
4. *Full Outer Join*: This join type returns all (matched or unmatched) rows from both the rule above and this rule.
5. *Cross Join*: This join type returns the cartesian product of the rule above and this rule.
6. *Union All*: This join behaves in the same way as a union, but duplicate records are not removed.
7. *Lookup*: Looks up fields and fills them in the first non-lookup rule above where the predicates match. At least one field needs to be defined as a lookup field.
8. *Lookup Auto Predicate*: Looks up fields and fills them in the first non-lookup rule where all common fields match. At least one field needs to be defined as a lookup field.

## Sub View

You can define further selections, formulas, aggregations and sorting orders for each rule.

## Complex Selections

If required, you can define complex selections using formulas and SQL functions.

## Join Predicates

You can define the predicate conditions for the matching for join and lookup rules here.

## Complex Predicates

If required, you can enter complex on-predicates for join and lookup rules here using formulas and SQL functions.

### Note

These settings are only relevant for multiple join rules in which either non-null or non-initial values of the same field are considered and returned for that field. If there is only one rule, the Auto Filling options are not relevant.

Usage of HANA HINT in complex selections could help in scenarios where a join function execution runs into an out of memory dump or is very slow. Refer to the HANA help for HINT to evaluate which HINT could help in your implementation-specific scenario (rules at the same level or multi-level rules which lead to subqueries, formula/selection being used).

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

See also:

- [Example: Full Outer Join – Autofilling Set to No \[page 110\]](#)
- [Example: If Null/Initial Then First to Last \[page 111\]](#)
- [Example: If Null Then First to Last \[page 113\]](#)
- [Example: Inner Join \[page 114\]](#)
- [Example: Left Outer Join \[page 115\]](#)
- [Example: Cross Join Implicit \[page 116\]](#)
- [Example: Union All \[page 117\]](#)
- [Example: Lookup Auto Predicates \[page 118\]](#)

### 1.3.7.1 Example: Full Outer Join – Autofilling Set to No

Input Tables

Product - Material Table

Product Code	Material Code	Request Order
P0001	M1011	5
P0001	M1010	2
P0002	M1009	1
P0002	M1011	3
P0005	M1012	10
P0005	M1011	30

JO - Product Table

Product Code	Product	Price	Unit
P0001	Shoe	75	EUR
P0002	Watch	300	EUR
P0003	Shirt	80	EUR
P0004	Shorts	20	EUR

This scenario executes a Full Outer Join that is based on the join predicate.

In this case, *Product Code* is the predicate used for both rules and all items with product code “P0001” and “P0002” are added in the final results.

Interim Result ( Product - Material Table and JO - Product Table)

Product - Material Table

Product Code	Material Code	Request Order
P0001	M1011	5
P0001	M1010	2
P0002	M1009	1
P0002	M1011	3

JO - Product Table

Product Code	Product	Price	Unit
P0001	Shoe	75	EUR
P0002	Watch	300	EUR

The following table shows parts of the Full Outer Join table. However, since they contain “?” or null values, they are not included in the final results.

Product Code	Material Code	Request Order	Product Code	Product	Price	Unit
P0005	M1012	10				
P0005	M1011	30		?	?	?
	?	?		?	?	?
	?	?	P0003	Shirt	80	EUR
			P0004	Shorts	20	EUR

The system does not return the rest of the non-null and/or non-initial values unless the auto filling setting is set to "If Null/Initial then First to Last".

The null values are caught by an error handler that informs you if there are null values for the join of the first rule with the fields *Material Code*, *Request Order*, *Price* and *Unit* for the product codes that are not included in the output.

Expected Result

Material Codw	Request Order	Product Code	Product	Price	Unit
M1011	5	P0001	Shoe	75	EUR
M1010	2	P0001	Shoe	75	EUR
M1009	1	P0002	Watch	300	EUR
M1011	3	P0002	Watch	300	EUR

Returns values for matching rows ("P0001" and "P0002") based on the join predicates set for the field *Product Code* (PROD\_CODE).

## 1.3.7.2 Full Outer Join Special Scenarios for Auto Filling field

### 1.3.7.2.1 Example: If Null/Initial Then First to Last

Input Tables

Legend: " is considered as an initial or empty input

JO – Product / Customer in US

Product	Customer	Amount
PROD01	US_CUST01	200
PROD04	US-CUST04	100
PROD06	US_CUST06	300
PROD07	"	100
PROD08	"	200
PROD09	US_CUST09	300

JO – Product / Customer in DE

Product	Customer	Price
PROD01	DE_CUST01	120
PROD04	DE_CUST04	60
PROD05	DE_CUST05	180
PROD07	DE_CUST07	60
PROD08	DE_CUST08	120
PROD10	DE_CUST10	180

The system takes the first non-null and non-initial value and if all values are null or initial, it returns an initialized value, empty or blank for a Character (CHAR) field and "0" for a *Key Figure* field.

Interim Results

Product	Customer	Amount	Price
PROD01	US_CUST01	200	120
PROD04	US_CUST04	100	60
PROD05	DE_CUST05	?	180
PROD06	US_CUST06	300	?
PROD07	"	100	60
PROD08	"	200	120
PROD09	US_CUST09	300	?
PROD10	DE_CUST10	?	180

In PROD05 row, we had our first null value (?). Since we only have two tables we won't be able to look further for another initial value. Since all values are null for the *Amount* field, the system returns an initialized value. For *Key Figure*, it will be "0". The same scenario will be encountered for PROD06, PROD09 and PROD10.

In PROD07 row, we had our first initial value ("), as you can see we look at the next table for the same *Customer* field. Since the next value for Customer is "DE\_CUST07", it will be the value for *Customer* field at PROD07. The same is true for PROD08 scenario, which will have a value of "DE\_CUST08".

Expected Result

Product	Customer	Amount	Price
PROD01	US_CUST01	200	120
PROD04	US_CUST04	100	60
PROD05	DE_CUST05	0	180
PROD06	US_CUST06	300	0
PROD07	DE_CUST07	100	60
PROD08	DE_CUST08	200	120
PROD09	US_CUST09	300	0



Product	Customer	Amount	Price
PROD10	DE_CUST10	0	180

### 1.3.7.2.2 Example: If Null Then First to Last

Input Tables

Legend: " is considered as an initial or empty input

JO – Product / Customer in US

Product	Customer	Amount
PROD01	US_CUST01	200
PROD04	US-CUST04	100
PROD06	US_CUST06	300
PROD07	"	100
PROD08	"	200
PROD09	US_CUST09	300

JO – Product / Customer in DE

Product	Customer	Price
PROD01	DE_CUST01	120
PROD04	DE_CUST04	60
PROD05	DE_CUST05	180
PROD07	DE_CUST07	60
PROD08	DE_CUST08	120
PROD10	DE_CUST10	180

The system takes the first non-null value and if all values are null, it returns the initial value for that field.

Interim Results

Product	Customer	Amount	Price
PROD01	US_CUST01	200	120
PROD04	US_CUST04	100	60
PROD05	DE_CUST05	?	180
PROD06	US_CUST06	300	?
PROD07	"	100	60
PROD08	"	200	120
PROD09	US_CUST09	300	?
PROD10	DE_CUST10	?	180

In the row PROD05, we have our first null value (?). Since all values are null for the *Amount* field, the system returns an initialized value. For *Key Figure*, the value is 0. The same scenario applies to PROD06, PROD09 and PROD10.

In the row PROD07, we have our first initial value (""). Since this initial value ("") is a non-null value, it is used as the result. The same is true for the scenario PROD08, which has a value of empty or blank because the type is "Character".

Expected Result

Product	Customer	Amount	Price
PROD01	US_CUST01	200	120
PROD04	US_CUST04	100	60
PROD05	DE_CUST05	0	180
PROD06	US_CUST06	300	0
PROD07		100	60
PROD08		200	120
PROD09	US_CUST09	300	0
PROD10	DE_CUST10	0	180

### 1.3.7.3 Example: Inner Join

Input Tables

Product - Material Table

Product Code	Material Code	Request Order
P0001	M1011	5
P0001	M1010	2
P0002	M1009	1
P0002	M1011	3
P0005	M1012	10
P0005	M1011	30

Product Table

Product Code	Product	Price	Currency
P0001	Shoe	75	EUR
P0002	Watch	300	EUR
P0003	Shirt	80	EUR
P0004	Shorts	20	EUR

Returns all values with corresponding matches based on the join predicates for product code "P0001" and "P0002".

*Product*, *Price* and *Currency* which correspond to each *Product Code* will be distributed to each product code resulting in the below table.

Interim Results (Product - Material Table and Product Table)

Product Code	Material Code	Request Order	Product	Price	Currency
P0001	M1011	5	Shoe	75	EUR
P0001	M1010	2	Shoe	75	EUR
P0002	M1009	1	Watch	300	EUR
P0002	M1011	3	Watch	300	EUR

Since a formula has been declared in the subview of RULE2, SAP Profitability and Performance Management will perform it after the Inner Join has been run.

The system adds the field *Payment Amount* (Payment Amount = Request Order \* Price) along with its value to the final output.

Expected Result

Product Code	Material Code	Request Order	Product	Price	Currency	Payment Amount
P0001	M1011	5	Shoe	75	EUR	375
P0001	M1010	2	Shoe	75	EUR	150
P0002	M1009	1	Watch	300	EUR	300
P0002	M1011	3	Watch	300	EUR	900

### 1.3.7.4 Example: Left Outer Join

Input Tables

Product - Material Table

Product Code	Material Code	Request Order
P0001	M1011	5
P0001	M1010	2
P0002	M1009	1
P0002	M1011	3
P0005	M1012	10
P0005	M1011	30

Product Table

Product Code	Product	Price	Currency
P0001	Shoe	75	EUR
P0002	Watch	300	EUR
P0003	Shirt	80	EUR
P0004	Shorts	20	EUR

Returns all values with corresponding matches based on the join predicates for product codes P0001 and P0002.

The product codes P0005, P0003 and P004 have no matches in tables. They will be dropped in the result.

Expected Result

Product Code	Material Code	Request Order	Product	Price	Currency
P0001	M1011	5	Shoe	75	EUR
P0001	M1010	2	Shoe	75	EUR
P0002	M1009	1	Watch	300	EUR
P0002	M1011	3	Watch	300	EUR

Returns all values from the left table and all corresponding matches based on the join predicates for product codes P0001 and P0002

## 1.3.7.5 Example: Cross Join Implicit

Input Tables

Material Table				Product Table		Order Table	
Material Code	Material	Cost per Order	Unit	Product Code	Product	Branch	Order
M1011	Leather	10	USD	P0001	Shoe	BR001	10
M1012	Thread	5	USD	P0002	Watch	BR002	20

Interim Result of Material Table and Product Table (MatPro Table)

Material Code	Material	Cost per Order	Unit	Product Code	Product
M1011	Leather	10	USD	P0001	Shoe
M1012	Thread	5	USD	P0001	Shoe
M1011	Leather	10	USD	P0002	Watch
M1012	Thread	5	USD	P0002	Watch

Returns material code M1011 and M1012 for product code P0001.

Returns material code M1011 and M1012 for product code P0002.

The Cross Join produces an interim result set which is the number of rows in the first table multiplied by the number of rows in the second table.

Expected Result

Material Code	Material	Cost per Order	Unit	Product Code	Product	Branch	Order
M1011	Leather	10	USD	P0001	Shoe	BR001	10
M1011	Leather	10	USD	P0001	Shoe	BR002	20
M1011	Leather	10	USD	P0002	Watch	BR001	10
M1011	Leather	10	USD	P0002	Watch	BR002	20
M1012	Thread	5	USD	P0001	Shoe	BR001	10
M1012	Thread	5	USD	P0001	Shoe	BR002	20
M1012	Thread	5	USD	P0001	Watch	BR001	10
M1012	Thread	5	USD	P0002	Watch	BR002	20

The Cross Join produces a result set which is the number of rows in the interim table result multiplied by the number of rows on the third table.

In this result, we are identifying the number of orders, per product and per branch by cross referencing from [Material Table](#) to [Product Table](#) and [Order Table](#).

## 1.3.7.6 Example: Union All

Input Tables

JO - Material Table

Material Code	Material	Cost per Order	Currency
M1001	Paper	3	USD
M1002	Plastic	3	USD
M1003	Wax	4	USD
M1006	Botton	1	USD
M1007	Cotton	10	USD
M1009	Glass	50	USD
M1010	Lace	5	USD
M1011	Leather	10	USD
M1012	Thread	5	USD

JO - Product Table

Product Code	Product	Price	Currency
P0001	Shoe	75	EUR
P0002	Watch	300	EUR
P0003	Shirt	80	EUR
P0004	Shorts	20	EUR

Filter the tables first so that you have only the required entries:

- For the Material Table, we set the condition to M1001, M1002 and M1003.
- For the Product Table, we set the condition to P0001, P0002 and P0003.

Based on the selection conditions we set in the subview of each rule, we will have the following tables:

Interim Result

Material Table

Material Code	Material	Cost per Order	Currency
M1001	Paper	3.00	USD
M1002	Plastic	3.00	USD
M1003	Wax	4.00	USD

Product Table

Product Code	Product	Price	Currency
P0001	Shoe	75	EUR
P0002	Watch	300	EUR
P0003	Shirt	80	EUR

If we are using an explicit type of join, the fields that we define in the subview of each rule will be the output. However, we need to specify all the fields that we need in the subview of each rule because we need to have the same fields across the subview for the explicit view to work.

Union All is used to combine the result sets of two or more tables. It does not remove duplicate rows and all rows are returned.

Expected Result

Material Code	Material	Product Code	Product	Cost per Order	Price	Currency
M1001	Paper			3	0	USD

Material Code	Material	Product Code	Product	Cost per Order	Price	Currency
M1002	Plastic			3	0	USD
M1003	Wax			4	0	USD
		P0001	Shoe	0	75	EUR
		P0002	Watch	0	300	EUR
		P0003	Shirt	0	80	EUR

### 1.3.7.7 Example: Lookup Auto Predicates

#### Level 1 Processing

Product - Material Table will be enriched as the corresponding *Material* will be retrieved and displayed for every matching entry in the field *Material Code* (MAT\_CODE).

#### i Note

The system resolves the hierarchy of levels starting with the highest level, feeding as an input to the lower levels and ending with level 0.

Product - Material Table (Level 1, From)

Product Code	Material Code	# Request Order
P0001	M1011	5
P0001	M1010	2
P0002	M1009	1
P0002	M1011	3
P0005	M1012	10
P0006	M1011	30

Material Table (Level 1, Lookup Auto Predicate)

Material Code	Material	Cost per Order	Unit
M1001	Paper	3	USD
M1002	Plastic	3	USD
M1003	Wax	4	USD
M1006	Botton	1	USD
M1007	Cotton	10	USD
M1009	Glass	50	USD
M1010	Lace	5	USD
M1011	Leather	10	USD
M1012	Thread	5	USD

Interim Result (Level 1)

Product Code	Materials Code	# Request Order	Material
P0001	M1011	5	Leather

Product Code	Materials Code	# Request Order	Material
P0001	M1010	2	Lace
P0002	M1009	1	Glass
P0002	M1011	3	Leather
P0005	M1012	10	Thread
P0006	M1011	30	Leather

Level 0 Processing

## Level 0 Processing

The product table declared in the first rule ("From") will now perform a Left Outer Join (for every matched product code (`PROD_CODE`) entry) with the Level 1 result (enhanced Product - Material Table) since result processed from a higher level will be considered as an input for the lower level.

### Note

Setting the Product - Material Table as an input function for the second rule will not affect the results of the join since the system automatically detects that the input will be coming from the enhanced Product - Material Table.

Product Table (Level 0, From)

Product Code	Product	Price	Unit
P0001	Shoe	75	EUR
P0002	Watch	300	EUR
P0003	Shirt	80	EUR
P0004	Shorts	20	EUR

Interim Result (Level 1)

Product Code	Materials Code	# Request Order	Material
P0001	M1011	5	Leather
P0001	M1010	2	Lace
P0002	M1009	1	Glass
P0002	M1011	3	Leather
P0005	M1012	10	Thread
P0006	M1011	30	Leather

Expected Result

Product Code	Product	Price	Unit	Material Code	Request Order	Material
P0001	Shoe	75	EUR	M1011	5	Leather
P0001	Shoe	75	EUR	M1010	2	Lace
P0002	Watch	300	EUR	M1009	1	Glass
P0002	Watch	300	EUR	M1011	3	Leather
P0003	Shirt	80	EUR		0	

Product Code	Product	Price	Unit	Material Code	Request Order	Material
P0004	Shorts	20	EUR		0	
P0005				M1012	10	Thread
P0006				M1011	30	Leather

## 1.3.8 Funds Transfer Pricing

The funds transfer pricing (FTP) methodology determines the cost of funds associated with the lending and borrowing from a financial institution (for example, a bank) while considering liquidity, interest rate and currency risks.

This mechanism is therefore also a part of the internal process that sets interest rates on the retail and commercial products of a bank. FTP measures the performance of the bank's deposit-raising (borrowing) and funds-advancing (lending) business units by constructing and evaluating deposits and loans profitability indicators (for example, FTP rate, costs of funding, net interest margin (NIM), and net interest income (NII)). The bank's treasury departments usually determine the following different FTP rate types:

1. Rates charged for providing funds to loan-giving business units
2. Compensation rates for funds received from deposit-raising business units

To meet the regulatory and functional needs for a sound FTP system, financial institutions can implement the *Funds Transfer Pricing* tool integrated in SAP Profitability and Performance Management that enables you to do the following:

- Generate various cash flows on both fixed and variable rate instruments using *Flow Generation* and *Rate Modeling* rule types
- Construct FTP cost of funds yield curve using multiple interpolation and smoothing methods
- Calculate various durations on selected instruments
- Apply cash-flow-based FTP rate calculation methods (term-weighted matched maturity, NPV approach, and so on)
- Apply non-cash-flow-based FTP rate calculation methods (caterpillar or strip funding approach, weighted average rates, pool rate assignment, and so on)
- Use or construct other fund and liquidity transfer pricing approaches

Various rule and line types of the FTP function enable the implementation of the intricate methodology outlined above.

### Rule Types

The following rule types are available:

- **Durations**  
Calculates three different types of duration. Their outputs are of similar numerical value, but theoretical concepts behind calculations and the interpretation of results differ as do the practical applications.
- **Flow Generation**



As an umbrella for six different line types, this rule type generates principal cash flows according to amortization pattern or flexible individual cash flow for disbursement, except principals by *Single Flow* line type. Note that interest cash flows are handled by the *Rate Modeling* rule type. Therefore, by combining different line types of *Flow Generation* and *Rate Modeling*, SAP Profitability and Performance Management is able to generate various types of future cash flows in real business scenarios.

- **Flow Merge**

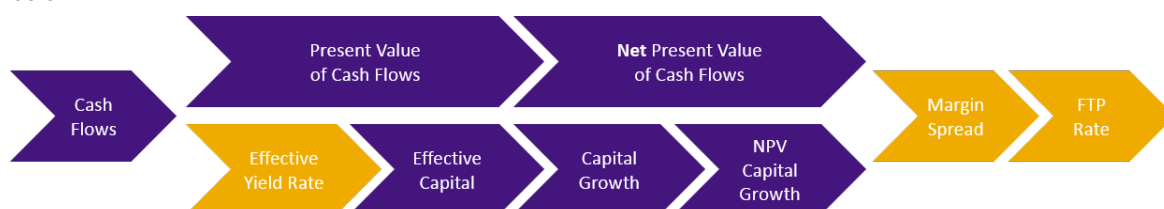
Joins financial position master data from one function with relevant transaction or business event data from another function that indicates exceptional cash flows.

- **Formula**

Applies formulas and HANA SQL functions that return numeric and string values to output fields.

- **Market Interest Rate**

Determines the FTP/LTP rate (the rate at which a bank extends or accepts loans to or from its internal departments) with several calculation steps in the logic sequence of the *Net Present Value Approach* shown below.



- **Matched Maturity**

Generally, as an extension of the multiple pools FTP methodology, this approach involves the coordination of a financial institution's cash inflows with its cash outflows based on the maturities of its assets and liabilities. Specifically, it calculates the FTP rate by matching due dates of (asset & liability instruments') principal cash flows to the marginal cost of funds curve (also referred to as the FTP curve) rates of corresponding maturity.

- **Rate Modeling**

As an umbrella for six different line types, this rule type covers various areas: it generates interest cash flows, interpolates rates on a yield curve, creates forward rates and determines daycount. Since the *Flow Generation* rule type generates principal cash flows in combination with *Rate Modeling*, SAP Profitability and Performance Management can generate various types of future cash flows in real business scenarios.

- **Running Total**

This rule type sequentially sums a selected field and updates this sum (also referred to as the "running total") for each row by adding it to the previous running total. It is used in finance to calculate, for example, loan principal outstanding that is the basis for interest calculations.

- **Series Generation**

This rule type generates series data by providing several parameters such as *Step Size*, *Series Type*, *Period From* and *Period To*.

- **Strip Funding**

Divides the net present value of principal payments by the sum of the term-accrued period-end principal balances, while discounting both with matching funding rates to calculate the FTP rate.

- **Weighted Average Rate**

This approach centers on the logic used to split a financial position into components. The chosen logic mandates matching of each part to different factors (weights) and funding rates – the only variables in the simple FTP weighted average rate calculation (WAR).

## Rule Lines

### Durations

The following rule lines are available:

- **Macaulay Duration** is the weighted average maturity of future cash flows of a financial instrument. The weight of each cash flow's maturity is determined by dividing the present value of the cash flow by the net present value of the observed instrument. Under the assumption of yearly compounding, Macaulay duration is given by

$$\frac{\sum T_i \times \frac{CF_i}{(1+r)^{T_i}}}{\sum \frac{CF_i}{(1+r)^{T_i}}}$$

where "CF<sub>i</sub>" is the (absolute) amount of the i-th cash flow, "T<sub>i</sub>" is the respective maturity of the i-th cash flow and "r" is the yield to maturity of this financial instrument.

- **Modified Duration** is a measure of price sensitivity, defined as the percentage derivative of price with respect to yield to maturity and compounding method. Under the same assumption, modified duration is given by

$$\frac{\sum T_i \times \frac{CF_i}{(1+r)^{T_i}}}{\sum \frac{CF_i}{(1+r)^{T_i}}} \times \frac{1}{(1+r)}$$

- **Fisher-Weil Duration**: If rates from a zero coupon yield curve are used instead of the yield to maturity, the Fisher-Weil duration is calculated with the same formula instead of Macaulay duration.

### Flow Generation

The following rule lines are available:

- **Periodic Fixed Amount Flow**  
Calculates principal payments that, in combination with interest payments, would compose a fixed periodic total of equal value in all periods (but with constantly changing principal and interest values).  
Principal calculation is done in two steps:

First, the amortizing factor for n-th principal payment (A<sub>n</sub>) is found:

$$A_n = \frac{1 - \left(1 + \frac{r}{f}\right)^{t_n - T}}{1 - \left(1 + \frac{r}{f}\right)^{-T}}$$

Where "r" is the nominal interest rate, "f" is the payment frequency (number of principal payments in a year), "t<sub>n</sub>" is the term of the n-th principal payment, and "T" is the term of the last principal payment determined by the maturity.

Then, the amortizing factors are used to calculate the periodic principal payments (P<sub>n</sub>):

$$P_n = N(A_{n-1} - A_n)$$

- **Periodic Fixed Even Flow**

Calculates equal principal payments (hence “even flow”) by dividing the total outstanding amount on the maturity date with the total number of payment periods.

- **Periodic Fixed Interest Rate Flow**

Calculates principal cash flows as the simple product of a selected rate and selected start value, except for the last payment, which is the difference between start value and sum of previous principal payments.

- **Periodic Fixed Rate Flow**

This line type’s output, principal cash flow, is determined by subtracting (1) interest payments from (2) total cash flows, except for the last payment which equals the remaining principal of the previous period:

1. Interest is the product of the input field *Rate* and the remaining principal of the previous period.
2. Total cash flow is the product of the input fields *Repayment Rate* and *Start Value*, with the exception of the last payment, which is the sum of the remaining principal and the last interest payment.

- **Periodic Fixed Value Flow**

This line type’s periodic cash flows are equal to the amount set in the input field *Value*. The exception to this is the last payment, which is the difference between the amount set in the input field *Start Value*, the sum of all previous periodic cash flows, and the amount set in the input field *End Value*.

- **Single Flow**

Line types described above generate multiple output rows as principal and interest calculations across all periods. The *Single Flow* line type generates only one output row, which makes it suitable for special or atypical cash flows like balloon/bullet payments, administration fees, disbursements, and so on.

## Flow Merge

The following line type is available:

- **Include Events to Flows**

Joins an instrument’s master data (for example, origination amount and date, currency) from one function with its tripartite event data (cash flow amount/type/date) from another function into one data record (hence “flow merge”).

## Market Interest Rate

### i Note

For easier understanding, line type descriptions are given in the order of the calculation steps (they are not listed in alphabetic order).

Some line types use continuous compounding to depict continuous and infinite reinvestment of interest, rather than monthly, quarterly, or annual compounding. Equations with continuous compounding use natural logarithms (for example,  $y = \ln(x)$ ) or inverse of natural log ( $x = \exp(y)$ ).

The following line types are available:

- **Effective Yield Rate**

Calculates continuous effective interest rate with respect to a given set of future cash flows associated to a financial position in two steps. In the first step, it calculates the internal rRate of return (IRR) as the root of the following equation:

$$P_0 = \sum_{i=1}^n \frac{P_i}{(1+r)^{\frac{T_i}{d}}}$$

where:

- “ $P_i$ ” is the  $i$ -th (re)payment including principal and interest
- “ $T_i$ ” is the term of the  $i$ -th payment counted in days, for example, the days from starting date to  $i$ -th payment counted regarding a given interest calculation method
- “ $d$ ” is the total number of days in a year regarding the given day count convention. For example,  $d = 360$  for Act/360, 30/360;  $d = 365$  for Act/365, and so on.

In the second and last step, the system calculates and displays the continuous effective interest rate “ $r_{eff}^c$ ”, based on the assumption of continuous compounding, as determined by the following formula:

$$(r_{eff}^c) = \ln(1 + r_{eff})$$

where “ $\ln$ ” is the natural logarithm.

- **Effective Capital over Time**

Calculates the effective capital as the difference between the continually compounded initial capital (inflow) and the sum of subsequent repayments (outflows). Each period's effective capital is the difference between the previous period's effective capital (that is continually compounded) and the current period's repayment. The following equation resembles this relationship:

$$ECOT_i = ECOT_{i-1} \times e^{(r_{eff}^c \times \frac{T_i - T_{i-1}}{d})} - P_i$$

where  $i = 1, \dots, n$  and “ $ECOT_0$ ” is the initial amount invested (for example, the total amount of a loan).

- **Capital Growth**

Capital growth for a given period equals the ratio of (i) the product of the previous period's effective capital and the continuous effective rate for the elapsed period (hereinafter numerator) and (ii) the continuous effective interest rate (hereinafter denominator). The calculation is therefore similar to that of perpetual annuity: periodic income (numerator: product of effective capital and effective yield rate for the elapsed period) is divided by the effective yield rate (denominator).

$$CG_i = ECOT_{i-1} \times \frac{e^{(r_{eff}^c \times \frac{T_i - T_{i-1}}{d})} - 1}{r_{eff}^c}$$

where  $i = 1, \dots, n$ . By convention  $CG_1 = 0$ .

- **Capital Growth Net Present Value**

Returns the simple product of an already calculated discount factor and capital growth to generate present values of capital growth for each period.

- **Net Present Value**

Returns the simple product of an already calculated discount factor and cash flows to generate the present value of future cash flows.

- **Net Present Value Sum**

Returns the sum of the present value of cash flows, which is the NPV of cash flows.

- **Margin Spread**

Determines margin spread as the ratio of cash flow NPV and capital growth NPV. Margin spread is the rate that is usually charged above FTP rate to ensure that each investment (for example, an extended loan) generates positive returns. From a financial institutions's perspective, margin spread is the difference between the borrowing and lending rates in deposits or the difference between the cost of borrowing and return from lending.

- **FTP Rate**

Determines the funds transfer pricing rate as the difference between continuous effective interest rate and margin spread. FTP is the rate at which a bank extends (or accepts) loans to (or from) its internal departments.

- **Market Interest Rate**

This line type contains all of the Net Present Value approach's outputs as described above.

## Matched Maturity

**Matched Maturity**, also known as "Term Weighted Matched Maturity", calculates the FTP rate by matching principal repayments' due dates to the rates on the marginal cost of funds curve (or FTP curve) of same maturities, while treating the product of corresponding terms and principal repayments as weights:

$$\frac{\sum_{i=1}^n P_i \times T_i \times r_i}{\sum_{i=1}^n P_i \times T_i}$$

where "P<sub>i</sub>" is the i-th principal payment amount relating to the i-th term "T<sub>i</sub>" and "r<sub>i</sub>" is the prevailing interest rate relating to "T<sub>i</sub>" retrieved from the specified FTP rate curve, and "n" is the total number of principal payments.

### i Note

Note on terminology: *Cost of Funds Curve* (usually an index like the LIBOR swap, FHLB funding, or local national government treasury curve) contains reference rates at which financial institutions presumably can raise debt. *Marginal Cost of Funds Curve* (or FTP Curve) modifies reference rates to account for incremental costs (banks bear) to get new funding because of their credit rating, liquidity (borrowing's size/term), and so on.

Use "Rate Modeling" or "Interpolation" rule types to generate the FTP curve, the main prerequisite of the Matched Maturity approach.

## Rate Modeling

The following rule lines are available:

- **Daycount**

Calculates the number of days between two dates: starting date(s) of input field *Date* and ending date(s) of input field *Curve Date*. The calculation also depends on daycount basis, like actual/actual, 30/360, and so on.

- **Forward Rate**

Derives forward rates from spot rates of a selected yield curve by non-arbitrage principle using the equation below:

$$\frac{(1 + r_{x+y})^{t_{x+y}}}{(1 + r_x)^{t_x}} - 1$$

where

- "r<sub>x+y</sub>" is the spot rate of a zero-coupon bond of longer maturity term "t<sub>x+y</sub>"
- "r<sub>x</sub>" is the spot rate of a zero-coupon bond of shorter maturity term "t<sub>x</sub>".

The following table compares derivation and application of spot and forward rates:

Spot Rates	Forward Rates
Zero-coupon treasury yields	Derived from spot rates
Today's price (or interest rate) of immediate transaction	Today's price (or interest rate) of transaction to occur in the future
Discounts a future cash flow to the present date	Discounts a distant future cash flow to a closer future date
Example: 3-month and 12-month zero-coupon rates	Example: 9-month rate expected 3 months from now (implied by 3-month and 12-month zero-coupon rates)

Non-arbitrage principle means that the forward rate shall equal the future spot rate. For example, a longer-term spot rate (for example 12 months) shall equal compounded return of the shorter-term spot rate (for example 3 months) and the remaining-term forward rate (for example 9 months).



- **Lookup Rate by Interpolation**

Linearly interpolates rates of a yield curve (of the *Lookup* tab) on dates from the input field *Date* (of the *Input* function) and assigns them to the Input function's instrument(s) of matching characteristic values for curve ID, validity date, and currency.

- **Periodic Fixed Interest**

Produces periodic interest ordinarily using a set of cash outflows (for example disbursements) and cash inflows (principal repayments) as input. Specifically, the calculation applies the **fixed** periodic interest rate to the instrument's remaining value (outstanding balance) – usually a difference between initial disbursement and the sum of subsequent principal repayments.

- **Periodic Variable Interest**

Produces periodic interest, ordinarily using a set of cash outflows (for example disbursements) and cash inflows (principal repayments) as input. Specifically, the calculation applies the **variable** periodic interest rates looked up from a certain interest rate curve to the instrument's remaining value (outstanding balance) – usually a difference between initial disbursement and the sum of subsequent principal repayments.

## Strip Funding

**Strip Funding** calculates the FTP rate as the ratio of (i) NPV of principal payments discounted using maturity matching funding rates by annual compound method and (ii) the sum of discounted and term-accrued principal balances remaining at each period.

$$\frac{B_0 - \sum_{i=1}^N P_i \times d_i}{\sum_{i=1}^N B_{i-1} \times \Lambda_{i-1,i} \times d_i}$$

where “ $B_0$ ” is the initial balance on FTP pricing date, “ $B_i$ ” is the outstanding closing balance of the payment period  $i$ . “ $P_i$ ” is the principal payment amount of this payment period. “ $\Lambda_{i-1,i}$ ” is the accrual factor in year for the whole period  $i$ . And “ $d_i$ ” is the discount factor derived from a given FTP reference curve by annual compound method.

## i Note

Note on terminology: “Strip Funding” is a widely accepted method of separately valuating each principal cash flow (in case of maturing products) or each component (in case of non-maturing products) to determine the FTP rate. Each principal cash flow or component is, “stripped” from the rest and assigned a funding rate, and so on. The rule types *Matching Maturity*, *Strip Funding* and *Weighted Average Rate* apply this general concept using different equations to generate FTP rates.

### Weighted Average Rate

This approach entails four steps:

1. Splits a financial position into components according to a certain logic (like differing behavioral tendencies of position's parts). The chosen logic mandates the remaining steps.
2. Assigns weighting factors (percentages or amounts) to each part.
3. Selects a suitable marginal cost of funds curve and assigns relevant funding rates to each component.
4. Calculates FTP (WAR) as the ratio of
  - product sum of factors ( $f_i$ ) and corresponding funding rates ( $r_i$ ) and
  - sum of factors ( $f_i$ ) as shown in the equation below.

## i Note

Only step four is solely set up and executed in this line type; the first three steps are executed in other rule types in SAP Profitability and Performance Management (for example, *Join* or *Interpolation*).

$$\frac{\sum f_i \times r_i}{\sum f_i}$$

This rule line type calculates the average funding rate from an FTP curve weighted by term – a special variant of the term-weighted matched maturity method. It is also a non-cash-flow transfer pricing method for non-maturity positions like checking, savings, money market, and credit card accounts. These kinds of non-contractual cash flow balances can behave both as long and short maturities, and can be split accordingly and assigned a respective long-term and short-term interest rate.

## 1.3.8.1 Example: Series Generation

### Input Data

Position ID	Increment	Minimum	Maximum	Element Number
001	1	1	500	0

The function generates a series of values starting from “1” with an increment equal to “1” until the element number reaches the maximum, which is set to “500”.

## Result

Position ID	Increment	Minimum	Maximum	Element Number
001	1	1	500	1
001	1	1	500	2
001	1	1	500	3
001	1	1	500	4
001	1	1	500	5
001	1	1	500	6
001	1	1	500	7
001	1	1	500	8
001	1	1	500	9
001	1	1	500	...
001	1	1	500	500

### 1.3.8.2 Example: Formula

#### Input Data

Position ID	Term (m)	Cash Flow Principal	Currency	Term Unit	Interest Calculation Method	Interest Rate (%)	Discount Factor
L04000000	0.000	10.000.000	USD	Day	360	0.000	1.000
L04000000	31.000	302.000	USD	Day	360	1.893	0.998
L04000000	59.000	302.000	USD	Day	360	2.033	0.997
L04000000	90.000	302.000	USD	Day	360	2.134	0.995
L04000000	120.000	302.000	USD	Day	360	2.226	0.993
L04000000	151.000	302.000	USD	Day	360	2.319	0.990
L04000000	181.000	302.000	USD	Day	360	2.408	0.988



Position ID	Term (m)	Cash Flow Principal	Currency	Term Unit	Interest Calculation Method	Interest Rate (%)	Discount Factor
L04000000 0	212.000	302.000	USD	Day	360	2.499	0.986
L04000000 0	243.000	302.000	USD	Day	360	2.590	0.983
L04000000 0	273.000	302.000	USD	Day	360	2.678	0.980
L04000000 0	304.000	302.000	USD	Day	360	2.769	0.977
L04000000 0	334.000	302.000	USD	Day	360	2.856	0.974
L04000000 0	365.000	302.000	USD	Day	360	2.947	0.971
L04000000 0	396.000	302.000	USD	Day	360	2.971	0.968
L04000000 0	425.000	302.000	USD	Day	360	2.991	0.966
L04000000 0	456.000	302.000	USD	Day	360	3.012	0.963
L04000000 0	486.000	302.000	USD	Day	360	3.033	0.960
L04000000 0	517.000	302.000	USD	Day	360	3.054	0.958
L04000000 0	547.000	302.000	USD	Day	360	3.074	0.955
L04000000 0	578.000	302.000	USD	Day	360	3.096	0.952

## Configuration

Rule	Description	Rule Type	State	Rule Grouping Fields	Rule Ordering Fields	Selection
R001	Key Figure Formula	Key Figure Formula	Active			

Line	Description	Formula	Result
L001	Key Figure Formula	Cash Flow Principal * 0.5	(Net) Present Value

Based on the configuration, the key figure that the formula applies is "Cash Flow Principal". The system applies the formula "Cash Flow Principal \* 0.5" for each data entry and stores it in the additional column *(Net) Present Value*.

## Expected Result

Position ID	Term (m)	Cash Flow Principal	Currency	Term Unit	Interest Calculation Method	Interest Rate (%)	Discount Factor	(Net) Present Value
L04000000	0.000	10,000.000	USD	Day	360	0.000	1.000	5,000.00
L04000000	31.000	302.000	USD	Day	360	1.893	0.998	151.00
L04000000	59.000	302.000	USD	Day	360	2.033	0.997	151.00
L04000000	90.000	302.000	USD	Day	360	2.134	0.995	151.00
L04000000	120.000	302.000	USD	Day	360	2.226	0.993	151.00
L04000000	151.000	302.000	USD	Day	360	2.319	0.990	151.00
L04000000	181.000	302.000	USD	Day	360	2.408	0.988	151.00
L04000000	212.000	302.000	USD	Day	360	2.499	0.986	151.00
L04000000	243.000	302.000	USD	Day	360	2.590	0.983	151.00
L04000000	273.000	302.000	USD	Day	360	2.678	0.980	151.00
L04000000	304.000	302.000	USD	Day	360	2.769	0.977	151.00
L04000000	334.000	302.000	USD	Day	360	2.856	0.974	151.00
L04000000	365.000	302.000	USD	Day	360	2.947	0.971	151.00
L04000000	396.000	302.000	USD	Day	360	2.971	0.968	151.00
L04000000	425.000	302.000	USD	Day	360	2.991	0.966	151.00
L04000000	456.000	302.000	USD	Day	360	3.012	0.963	151.00
L04000000	486.000	302.000	USD	Day	360	3.033	0.960	151.00
L04000000	517.000	302.000	USD	Day	360	3.054	0.958	151.00
L04000000	547.000	302.000	USD	Day	360	3.074	0.955	151.00

Position ID	Term (m)	Cash Flow Principal	Currency	Term Unit	Interest Calculation Method	Interest Rate (%)	Discount Factor	(Net) Present Value
L0400000 00	578.000	302.000	USD	Day	360	3.096	0.952	151.00

### 1.3.8.3 Example: Running Total

#### Input Data

Yield Curve Type	Period	Interpolated Yield	Amount
Yield Curve	0	-0.0001537	100.0000
Yield Curve	1	0.0043374	100.4337
Yield Curve	2	0.0088285	101.7735
Yield Curve	3	0.0096286	102.9165
Yield Curve	4	0.0104287	104.2372
Yield Curve	5	0.011203	105.7284
Yield Curve	6	0.0119945	107.4160
Yield Curve	7	0.012786	109.3009
Yield Curve	8	0.0135775	111.3924
Yield Curve	9	0.0143518	113.6835
Yield Curve	10	0.0151261	116.1983
Yield Curve	11	0.0159004	118.9495
Yield Curve	12	0.0166747	121.9507
Yield Curve	13	0.017449	125.2173

#### Configuration

Rule	Description	Rule Type	State	Rule Grouping Fields	Rule Ordering Fields	Selection
R001	Running Total	Running Total	Active			

Line	Description	Value	Granularity	Term	Running Total
L001	Running Total	Amount	Yield Curve Type	Period	Running Total

Based on the configuration, the key figure to be aggregated for the running total is *Amount*. The system computes the total of all the data and stores it in the additional column *Running Total*.

## Expected Result

Yield Curve Type	Period	Interpolated Yield	Amount	Running Total
Yield Curve	0	-0.0001537	100.0000	100.0000
Yield Curve	1	0.0043374	100.4337	200.4337
Yield Curve	2	0.0088285	101.7735	302.2072
Yield Curve	3	0.0096286	102.9165	405.1237
Yield Curve	4	0.0104287	104.2372	509.3609
Yield Curve	5	0.011203	105.7284	615.0893
Yield Curve	6	0.0119945	107.4160	722.5053
Yield Curve	7	0.012786	109.3009	831.8062
Yield Curve	8	0.0135775	111.3924	943.1987
Yield Curve	9	0.0143518	113.6835	1056.8822
Yield Curve	10	0.0151261	116.1983	1173.0805
Yield Curve	11	0.0159004	118.9495	1292.0300
Yield Curve	12	0.0166747	121.9507	1413.9806
Yield Curve	13	0.017449	125.2173	1539.1980

## 1.3.9 Valuation

In financial accounting, valuation is the process of determining the values of financial instruments or other resources. The *Valuation* function provides a variety of different rule and line types to calculate financial or statistical measures, such as discounted value of money, as well as median, variance, and so on, which can be used in the process.

Valuation usually consists of several product-specific or service-specific steps. In SAP Profitability and Performance Management each rule or line of the valuation function represents one of these steps in the configuration. For example, for commercial insurance contracts discounting may be carried out at cash flow level and value aggregation used to calculate statistical properties or sum up discounted cash flows to present values of the observed contracts. Therefore, you can use hierarchical rules to trigger the sequential valuation steps.

## Key Features

### Rule Types

**Characteristic Formula:** Applies formulas and SAP HANA SQL functions that return a string to characteristics output field.

**Discounting:** Calculates discounted or (in investment finance) “present” value of future money, such as expected payments.

**Duration:** Calculates three different types of durations. Their outputs are of similar numerical value, but the theoretical concepts behind the calculations and the interpretation of results differ significantly, as do the practical applications.

**Interpolation:** Produces a new series by linear approximating values between (interpolation) and beyond (extrapolation) known-value time points. It does so by replacing user-specified initial values with linear approximated values. It is used in finance to determine interest rates for missing time points along the yield curve (i.e. term structure of the curve).

**Key Figure Formula:** Applies formulas and SAP HANA SQL functions that returns numeric values to key figure output field.

**Line Item Valuations:** As an umbrella for eight different line types, this rule type enables the following data manipulations:

- Output can be calculated separately for each change in selected granularity (*Balance Granularity Fields*)
- Input can be multiplied by a number, formula or selected field (*Factor*)
- Input can be filtered for certain *Selection* conditions.

**Running Total:** Like the line type “Running Balance” of rule type “Line Item Valuation”, this rule type sequentially sums a selected field and updates this sum (aka “running total”) for each row by adding it to the previous running total. Unlike the line type “Running Balance” of rule type “Line Item Valuation”, this rule type takes the current row’s value in the calculation of the running total. It is used in finance to calculate the amount of outstanding loan principal that forms the basis for interest calculation, for example.

**Value Aggregation:** Output can be calculated separately for each change in selected granularity and by limiting the range of rows from the current row (Lower / Upper Row Offset).

### Rule Lines

#### Duration

- **Macaulay Duration** is the weighted average maturity of future cash flows of a financial instrument. The weight of each cash flow’s maturity is determined by dividing the present value of the cash flow by the net present value of the observed instrument. Under the assumption of yearly compounding, Macaulay Duration is given by the following formula, where “CF<sub>i</sub>” is the (absolute) amount of the i-th cash flow, T is the respective maturity of i-th cash flow, “r” is the yield of maturity of this financial instrument:

$$\frac{\sum T_i \times \frac{CF_i}{(1+r)^{T_i}}}{\sum \frac{CF_i}{(1+r)^{T_i}}}$$

- **Modified Duration** is a measure of price sensitivity, defined as the percentage derivative of price with respect to yield to maturity and compounding method. Under the same assumption modified duration is given by

$$\frac{\sum T_i \times \frac{CF_i}{(1+r)^{T_i}}}{\sum \frac{CF_i}{(1+r)^{T_i}}} \times \frac{1}{(1+r)}$$

- **Fisher-Weil Duration:** When rates from zero-coupon yield curve are used instead of the yield to maturity, the so-called Fisher-Weil duration is calculated with the same formula instead of Macaulay duration.

### Interpolation

- **Extrapolation None:** Leading and trailing initial values are not extrapolated (replaced). For example, an input of [null, null, 1, 2, null, null, 5, null, null] returns [null, null, 1, 2, 3, 4, 5, null, null].
- **Extrapolation Linear:** Leading and trailing initial values are replaced by the values calculated by the linear line extended first or last two known values respectively.
- **Extrapolation Constant:** Extends the first and last known value to leading and trailing initial values respectively.

### Line Item Valuations

- **Balance:** Returns the sum of an expression that is maintained in the *Value* input field of rows from the flow lookup function that matches the current row of input function by certain characters. The sum can then be multiplied with the returned value of an expression if this expression is maintained in input field *Factor*.
- **Formula:** Applies certain formulas and SQL functions to key figures and/or characteristics.
- **Lag:** Uses SAP HANA SQL *LAG* function to return the value of a previous row where the position is specified by the offset value in the same granularity set. The offset value should be positive, the default is "1". Its widespread usage entails cases where preceding data needs to be compared with or applied to current data. For example, the previous period's interest rate needs to be applied to current period's interest payment calculation.
- **Lead:** Uses SAP HANA SQL *LEAD* function to return the value of a following row whose position is specified by the offset value in the same granularity set. The offset value should be positive, the default is "1". Its widespread usage entails cases where subsequent data needs to be compared with or applied to current data. For example, a running total of all future payments needs to be calculated on a data set containing outflows.
- **Register:** Just like the line type "Formula", this line type applies formulas and SQL functions to key figures and/or characteristics. Unlike the rule type "Formula", this line type does not have options "Flow Lookup Function" and "Lookup Result".
- **Running Balance or Running "Opening" Balance:** Similar to the rule type "Running Total", this rule type sums the values of a selected field of rows before the current row so that each following line's value is added to the previous running balance. Unlike the rule type "Running Total", this rule type considers the preceding row's value in the calculation – hence the term "opening" balance.
- **Scaled Weighted Average:** Calculates weighted mean by dividing the sum of weights times values with the sum of the weights, according to the following formula:

$$\frac{\sum_{i=1}^n W_i \times X_i}{\sum_{i=1}^n W_i}$$

- **Running Scaled Weighted Average:** Calculates a "running" weighted mean for rows before the current row by dividing the sum of weights times values with the sum of the weights.

### Value Aggregation

- **Number of Rows:** Returns the number of rows of a selected field.
- **Minimum Value:** Returns the minimum value of a selected field.
- **Statistical Median:** Returns the median of a selected field.
- **Maximum Value:** Returns the maximum value of a selected field.
- **Sum Value:** Returns the sum of a selected field.
- **Arithmetical Mean:** Returns the average of a selected field.
- **Standard Deviation Square Root of Population Variance:** Returns standard deviation of a selected field by taking the square root of the population variance.
- **Standard Deviation Square Root of Sample Variance:** Returns standard deviation of a selected field by taking the square root of the sample variance.
- **Population Variance Value:** Returns the population variance of an expression as the sum of squares of the difference of <expression> from the mean of <expression>, divided by the number of rows remaining.
- **Sample Variance Value:** Returns the population variance of an expression as the sum of squares of the difference of <expression> from the mean of <expression>, divided by the number of rows remaining minus 1.

### 1.3.9.1 Example: Running Total

#### Input Data

Yield Curve Type	Period	Interpolated Yield	Amount
Yield Curve	0	-0.0001537	100.0000
Yield Curve	1	0.0043374	100.4337
Yield Curve	2	0.0088285	101.7735
Yield Curve	3	0.0096286	102.9165
Yield Curve	4	0.0104287	104.2372
Yield Curve	5	0.011203	105.7284
Yield Curve	6	0.0119945	107.4160

In order to aggregate the value, the system uses the following formula:

$$RT = \sum_i^n Principal_i$$

Based on the configuration, the key figure to be aggregated for the running total is "Amount". The system computes the total of all the data and stores it in the additional column "Running Total".

## Result

Yield Curve Type	Period	Interpolated Yield	Amount	Running Total
Yield Curve	0	-0.0001537	100.0000	100.0000
Yield Curve	1	0.0043374	100.4337	200.4337
Yield Curve	2	0.0088285	101.7735	302.2072
Yield Curve	3	0.0096286	102.9165	405.1237
Yield Curve	4	0.0104287	104.2372	509.3609
Yield Curve	5	0.011203	105.7284	615.0893
Yield Curve	6	0.0119945	107.4160	722.5053

### 1.3.10 Flow Modeling

The *Flow Modeling* function offers a set of rule types that provide different calculation logic to process different kinds of business requirements. Each rule type represents an independent and reusable logic for enriching amounts, factors and dates of cash flows consumed, for example. You can add several rule types to the same *Flow Modeling* function, which can run in parallel or sequentially. In sequential processing the successor rule type consumes the results from the predecessor rule type. You can also add parallel-processed line items for some rule types. The key features of each configuration are explained below.

## Key Features

### Rules

Flow Modeling consists of several rule types where each rule type represents an encapsulated and reusable logic for the calculation of data.

The following rule types and corresponding examples are available:

1. [Characteristic Formula \[page 139\]](#):  
Applies formulas and SQL functions to characteristics.
2. [Key Figure Formula \[page 140\]](#):  
Applies formulas and SQL functions to key figures.
3. [Flow Cut-off \[page 141\]](#):  
Applies a cut-off to cash flow data according to a given reference day. The rule type deletes all cash flow items prior to the cut-off period.
4. [Series Generation \[page 144\]](#):  
Generates series data by providing several parameters such as *Step Size*, *Series Type*, *Period From* and *Period To*.
5. [Term Conversion \[page 147\]](#):  
Converts terms of different periodicities into a common basis of days. The rule type offers the option to choose different day count conventions (30/360 German, ACT/ACT).



6. [Term Selection \[page 149\]](#):  
Selects a specific term on a month-end basis from a cash flow taking into consideration which period type (monthly, quarterly, yearly) and day count convention (30/360 German, ACT/ACT) is set in the configuration.
7. [Term Target \[page 152\]](#):  
Enriches a given set of cash flow data (based on a daily periodicity) and returns a cash flow structure that uses a consistent periodicity of months, quarters or years, and that can be based on different sorts of day count conventions (30/360 German, ACT/ACT). It also interpolates the values of the terms added to the original pattern structure.
8. [Term To Date \[page 155\]](#):  
Converts a given set of terms of cash flows into dates referencing a given start date. The configuration allows you to choose between a default approach and an approach that applies different logic to distinguish between pattern items which are of balance type (cumulative factor/amount values) or movement type (delta factors/amounts).
9. [Value Conversion \[page 159\]](#):  
Comprises two different calculation methods that can be used either to sum up cash flow items over a given set of terms (running total), or to calculate the delta values between a given set of cash flow items (balance = cumulative values, movement = delta values).
10. [Incremental Value Calculation \[page 162\]](#):  
Combines two factor patterns and computes a mutual pattern.
11. [Redistribution \[page 166\]](#):  
Calculates estimate values prior to the Reference Date for Redistribution (RDR) and redistributes them to the future periods after this date.
12. [Scale Factor \[page 170\]](#):  
Ratio of two corresponding values with similar field/data types (division).
13. [Scaling \[page 171\]](#):  
Applies a scale factor to the actuarial model stream (multiplication).
14. [Acknowledge Actuals \[page 173\]](#) (Acknowledgement of Cedent Data):  
Enriches the actuals by determining the missing earlier life cycle date information by applying matching logic.  
The matching of the actuals to the estimates can be carried out in the following ways:
  - If the CF calculation is "01", the system matches the actual to the estimate based on the business date of the actual. In other words, *Settled Date* for settled transactions, *Due Date* for due transactions, and *Reported Date* for reported actuals.
  - If the CF calculation is "02", the system matches the actual to the estimates based only on the Secondary Risk Incurred Date. This is applied in L&H business where the missing dates in the actuals are predetermined by a reverse life cycle conversion based on the models.
15. [Life Cycle Conversion \[page 176\]](#):  
Applies the lags and lag factors delivered by the actuarial input in the form of a lag factor pattern to the cashflow stream to determine the amounts and date for the lifecycle stage.
16. [Modulation In \[page 179\]](#):
  - New contract:  
Modulation In for contract T must be equal to Modulation Out for contract T multiplied by -1.
  - Renewed contract:  
Modulation In for contract T must be equal to Modulation Out for contract T-1 multiplied by -1.
  - Incurred date of Modulation In cash flows should be set to be in the first period of contract T.
17. [Modulation Out \[page 181\]](#):
  - Selects amounts in the basis cash flow where the exposure date > coverage end date.

- If Modulation Out for contract T is updated and contract T+1 exists, Modulation In for contract T+1 must be (re-)calculated.
18. [Item Number Generation \[page 182\]](#):  
Separates each partition by creating a number for each one. To do this, the system needs a *Granularity* field (which separates the partitions from each other) and an *Item Number* field (which is filled by this rule type).
19. [Cashflow Regime \[page 184\]](#):  
Adjusts the cashflow stream based on the regime into which each of the cashflows falls:
- Follow Actuals (01)  
This regime comprises only the effect of actuals. Therefore the system removes any estimate item that falls in this regime from the final cashflow.
  - Reflect Actuals (02)  
This regime comprises only the effect of actuals. Model-based estimates do not have any effect in this regime. However, more actuals may be expected to be reported in this period. The system therefore calculates an additional incurred estimate as a factor of the actuals. These additional Incurred estimates will have the same date information as that of the actuals but the amounts will be calculated as a factor of the actuals and will apply the formula  $(\text{Amount} * \text{Factor} / (1 - \text{Factor}))$ .
  - Follow Estimates (03)  
In this regime, the model-based estimates are expected to be effective. Therefore, for every actual that falls into this regime an additional negated estimate is introduced into the cashflow with the same date information as the actuals.
  - Estimated Future (04)  
In this regime, no actual information is expected.
20. [Clear Actual Dates \[page 189\]](#)  
Clears the actual date information in the cashflows arising from actuals.
21. [Incurred to Reported Factor Calculation \[page 191\]](#)  
Calculates the *Incurred to Reported* lags in cases where these lags are not delivered directly, using the more granular *Policy Holder to Primary Insurer* lag and *Primary Insurer to Reinsurer* lag.
22. [Factor for Additional Incurred \[page 194\]](#)  
Calculates the factors to be applied in the Reflect Actual Regime based on the delivered *Policy Holder to Primary Insurer* lags. The factors are calculated as a difference between 1 and the cumulated sum of the policy holder to primary insurer lag factors for a certain granularity. The number of periods is determined by the Reflect Actuals attachment point.

Hierarchical rules are supported by assigning higher levels. In this case, the hierarchy of the levels is resolved starting with the lowest level, which is fed as input to the higher levels, and ending with the highest level.

## Sub View

You can define further selections, aggregations and sorting orders for each rule.

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

## 1.3.10.1 Example: Characteristics Formula

Applies formulas and SQL functions to characteristics.

In the example, we apply a simple SQL formula to compare two fields containing dates and we mark a third field according to that formula.

### Input Data

Contract ID	Date_1	Date_2	Total Premium
A	2019-01-01	2019-02-01	300
B	2019-04-01	2019-03-01	400

### Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
CHARF	Characteristic Formula	Characteristic Formula	Active			

Rule Line

Line	Formula	Result
L001	<pre> CASE WHEN DATE_1 &gt; DATE 2 THEN 'X' ELSE " END </pre>	DATE_IND

When *DATE\_1* is greater than *DATE\_2*, the field *DATE\_IND* (Date Indicator) is marked with "X".

### Key Configuration Description

The table below explains the meaning of the options required to run this configuration.

Field	Meaning	Notes
Line	Identifier of the line	Must be unique. A single characteristic formula rule can run multiple lines.

Field	Meaning	Notes
Formula	SQL formula that has to be applied to categorical variables	Mandatory input
Result	Field where the result of the rule is stored	Mandatory output. Must be defined in the <i>Action</i> section of the <i>Signature</i> tag.

## Expected Result

Contract ID	Date_1	Date_2	Total Premium	Date Indicator
A	2019-01-01	2019-02-01	300	
B	2019-04-01	2019-03-01	400	X

## 1.3.10.2 Example: Key Figure Formula

Applies formulas and SQL functions to key figures.

In this example, we apply a very simple SQL formula to compute the weighted premium (`WEIGHTED_PREMIUM`) based on the fields `TOTAL_PREMIUM` and `WEIGHT`.

## Input Data

Contract_ID	Date_1	TOTAL_PREMIUM	WEIGHT
A	2019-01-01	300	0.4
B	2019-04-01	400	0.6

## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
CHARF	Key Figure Formula	Key Figure Formula	Active			

Rule Line

Line	Formula	Result
L001	TOTAL_PREMIUM * WEIGHT	WEIGHTED_PREMIUM

## Key Configuration Description

The table below explains the meaning of the options required to run this configuration.

Field	Meaning	Notes
Line	Identifier of the line	Must be unique. A single Key Figure Formula rule can run multiple lines.
Formula	SQL formula that has to be applied to numerical variables	Mandatory input
Result	Field where the result of the rule is stored	Mandatory output. Must be defined in the <i>Action</i> section of the <i>Signature</i> tag.

## Expected Output

Contract ID	Date_1	Total Premium	Weight	Weighted Premium
A	2019-01-01	300	0.4	120
B	2019-04-01	400	0.6	240

### 1.3.10.3 Example: Flow Cut-Off

Applies a cut-off to cash flow data according to a given reference day. The rule type deletes all cash flow items prior to the cut-off period.

This function gives two different results depending on the type of input values: for delta values (MOV values), the pattern after the cut-off starts from period one; for cumulative values (BAL values), the resulting pattern starts from zero. The function can handle the day count conventions 30/360 German and ACT/ACT. Two different examples are shown below.

## Input Data

Start Date	Contract ID	Pattern KF Type	Period Unit	Period To	Result Value	Cut-Off Period
2019-01-01	A	MOV	4	30	EUR 30	60
2019-01-01	A	MOV	4	60	EUR 60	60
2019-01-01	A	MOV	4	90	EUR 90	60
2018-01-01	A	MOV	4	120	EUR 120	60
2019-01-01	B	MOV	4	30	EUR 30	30
2019-01-01	B	MOV	4	60	EUR 60	30
2019-01-01	B	MOV	4	90	EUR 90	30
2019-01-01	C	MOV	4	30	EUR 30	30
2019-01-01	C	MOV	4	60	EUR 60	30
2019-01-01	C	MOV	4	90	EUR 90	30
2019-01-01	C	MOV	4	120	EUR 120	30

In the table above, there are three patterns for three contracts. The field `PERIOD_UNIT` is not used in the configuration. It refers to `PERIOD_UNIT` and `PERIOD_TO` and indicates that they are days.

## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
FCF	Flow Cut-Off	Flow Cut-Off	Active			

Rule Line

Line	Line Granularity	Day Count Convention	Start Date	Value Type	Period To	Cut-Off Comparison	Period To Result	Period Unit
L001	CONTRACT_ID	German 30/360	START_DATE	PATTERN_KF_TYPE	PERIOD_TO	CUT_OFF_PERIOD	PERIOD_TO_RESULT	PERIOD_UNIT

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line.	Must be unique.

Field	Meaning	Notes
<b>Line Granularity</b>	Determines the size of one partition of data.	Mandatory input that has to be inserted in the <i>Granularity</i> section of the <i>Signature</i> tag.
<b>Day Count Convention</b>	Day count convention used to determine the result periods.	Mandatory input.  The conventions managed by the function are: <ul style="list-style-type: none"> <li>• Actual/Actual</li> <li>• German 30/360</li> </ul>
<b>Start Date</b>	Starting date to compute the day count.	Mandatory input.
<b>Value Type</b>	You can select the value "MOV" or "BAL" to determine which value type the cash flow item is made of.	Mandatory input.  The user has to make sure that the field contains these two values only.
<b>Period To</b>	Number of days of the period.	Mandatory input
<b>Cut-Off Comparison</b>	Key field to identify which cash flow items need to be deleted.	Mandatory input  Rule in case of delta items: Cut all periods <= cut-off period. If the balance is not zero, the system treats the cut-off period as a new period.
<b>Period To Result</b>	Output field to indicate the specific period of pattern of the cut-off.	Mandatory output
<b>Period Unit Result</b>	Value type of the period result.	Mandatory output.  The value "6" stands for "month"

## Expected Output

Start Date	Contract ID	Pattern KF Type	Period To	Result Value	Cut-Off Period	Period To Result	Period Unit Result
2019-01-01	A	MOV	90	EUR 90	60	1	6
2019-01-01	A	MOV	120	EUR 120	60	2	6
2019-01-01	B	MOV	60	EUR 60	30	1	6
2019-01-01	B	MOV	90	EUR 90	30	2	6
2019-01-01	C	MOV	60	EUR 60	30	1	6
2019-01-01	C	MOV	90	EUR 90	30	2	6
2019-01-01	C	MOV	120	EUR 120	30	3	6

All cash flow items prior to or equal to the cut-off period have been deleted.

## Example with BAL Data

The configuration does not change with respect to the previous example. What is different now is the value of the field `PATTERN_KF_TYPE` (moreover, the field `PERIOD_UNIT` has been deleted).

Input Data

Start Date	Contract ID	Pattern KF Type	Period To	Result Value	Cut-Off Period
2019-01-01	A	BAL	30	EUR 30	60
2019-01-01	A	BAL	60	EUR 60	60
2019-01-01	A	BAL	90	EUR 90	60
2019-01-01	A	BAL	120	EUR 120	60

Output Data

Start Date	Contract ID	Pattern KF Type	Period To	Result Value	Cut-Off Period	Period To Result	Period Unit Result
2019-01-01	A	BAL	60	EUR 60	60	0	6
2019-01-01	A	BAL	90	EUR 90	60	1	6
2019-01-01	A	BAL	120	EUR 120	60	2	6

## 1.3.10.4 Example: Series Generation

Generates series data by providing several parameters, like *Step Size*, *Series Type*, *Period From* and *Period To*.

### Input Data

Contract ID	Period From	Period To
A	1	3
A	4	4

Contract ID	Period From	Period To
B	1	2
B	3	8

Contract ID	Period From	Period To
A	1	3

Contract ID	Period From	Period To
A	1	2



Contract ID	Period From	Period To
A	3	8

## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
GS	Generate Series	Generate Series	Active			CON-TRACT="A"

Rule Line

Line	Incremented By	Minimum	Maximum	Series Type	Element Number	Fraction	Period From	Period To Result
1	1	PERIOD_FROM	PERIOD_TO	Integer	PERIOD_NUMBER			

### Note

Condition = Filter options that select only cash flow items containing filter option values. In this example the rule is computing only for cash flow items which have a value of "A" in column CONTRACT.

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique.
Increment by	Field or single value that defines the number added to the value of the field <i>Minimum</i> .	Mandatory input
Minimum	Starting value of the series	Mandatory input
Maximum	Ending value of the series	Mandatory input
Series Type	Defines the type of input and affects the resulting output.	Mandatory input.  Possible values: <ul style="list-style-type: none"> <li>Integer</li> <li>Date</li> <li>Decimal</li> </ul>

Field	Meaning	Notes
Element Number	The system writes the main result of the function to this field.	Mandatory output.  You need to insert the target field in the <i>Action</i> tag of the <i>Signature</i> section.
Fraction	This field can be used to save an additional result of the function.	The following calculation is computed: Minimum value of the range/maximum value of the range.
Period From Result	This field can be used to save an additional result of the function.	Period From + Increment by
Period To Result	This field can be used to save an additional result of the function.	Period From Result + Increment by

## Expected Result

Contract ID	Period From	Period To	Period Number
A	1	3	1
A	1	3	2
A	1	3	3
A	4	4	1

Contract ID	Period From	Period To	Period Number
A	1	3	1
A	1	3	2
A	1	3	3

Contract ID	Period From	Period To	Period Number
A	1	2	1
A	1	2	2
A	3	8	1
A	3	8	2
A	3	8	3
A	3	8	4
A	3	8	5
A	3	8	6

The three patterns are shown separately for clarity of exposition.

## 1.3.10.5 Example: Term Conversion

Converts terms of different periodicities into a common basis of days. The rule type offers the option to choose different day count conventions (30/360 German, ACT/ACT).

In this example, we show two patterns in the input data: a pattern for contract A and a pattern for contract B. We filter for contract A using a subview. The day conversion is performed starting from the key date on the field inserted in *Period To* on the *Configuration* tab. The result is a range in days.

### Input Data

The fields *Pattern Key Figure Type* and *Value* are not relevant for the conversion.

Start Date	Contract ID	Pattern KF Type	Period To	Cal FreqCode	Value
2019-01-01	A	MOV	3	6	EUR 90
2019-01-01	A	MOV	4	6	EUR 120
2019-01-01	A	MOV	5	6	EUR 150
2019-01-01	A	MOV	6	6	EUR 180

Start Date	Contract ID	Pattern KF Type	Period To	Cal FreqCode	Value
2019-01-16	B	MOV	2	6	EUR 50
2019-01-16	B	MOV	8	6	EUR 150

### Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TC	Term Conversion	Term Conversion	Active			CONTRACT = "A"

Rule Line

Line	Line Granularity	Start Date	Day Count Convention	Period To	Period Unit	Period To Conv	Period From Conv	Period Unit Conv
1	Contract ID	Start Date	German 30/360	Period	Period Unit	Period to Converted	Period from Converted	Period Unit Converted

#### i Note

Condition = Filter out options that select only cash flow items containing filter option values. In this example, the rule computes only for cash flow items with an "A" entry in the *Contract ID* column.

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique.
Line Granularity	Determines the size of one partition of data	Mandatory input that has to be inserted in the <i>Granularity</i> tab of the <i>Signature</i> section.
Start Date	Starting date to compute the day count	Mandatory input
Day Count Convention	Day count convention used to determine the result period	Mandatory input.  The conventions managed by the function are: <ul style="list-style-type: none"> <li>• US 30/360</li> <li>• Actual/Actual</li> <li>• EU 30/360</li> <li>• German 30/360</li> </ul>
Period To	Parameter to indicate the period to be added to <i>Start Date</i>	Mandatory input
Period Unit	Period type of input data (for example, monthly or quarterly)	Mandatory input.  The possible values are the following: <ul style="list-style-type: none"> <li>• 4 if <i>Period To</i> is in days</li> <li>• 5 if <i>Period To</i> is in weeks</li> <li>• 6 if <i>Period To</i> is in months</li> <li>• 11 if <i>Period To</i> indicates quarters</li> <li>• 12 if <i>Period To</i> indicates half a year</li> <li>• 7 if <i>Period To</i> indicates years</li> </ul>
Period From Conv	Exact day from which the period count starts.	Mandatory output
Period To Conv	Number of days of the period	Mandatory output
Period Unit Conv	The system writes the period unit of the result to this field.	Mandatory output.  The function performs the conversion from the number in the field inserted in <i>Period Unit</i> to days (value equal to 4).

## Expected Result

Start Date	Contract ID	Pattern KF Type	Period To	Cal Freq Code	Value	Period From Conv	Period To Conv	Period Unit Converted
2019-01-01	A	MOV	3	6	EUR 90	1	90	4
2019-01-01	A	MOV	4	6	EUR 120	91	120	4
2019-01-01	A	MOV	5	6	EUR 150	121	150	4
2019-01-01	A	MOV	6	6	EUR 180	151	180	4

### 1.3.10.6 Example: Term Selection

Selects a specific term on a month-end basis from a cash flow taking into consideration which period type (monthly, quarterly, yearly) and day count convention (30/360 German, ACT/ACT) is set in the configuration.

In the example for this type of configuration, we run the function for three different patterns and for two types of target conversion (from days to months and from days to quarters). The output is a new pattern for the three contract IDs with a new periodicity.

### Input Data

Start Date	Contract ID	Pattern KF Type	Period Unit	Period To	Result Value
2019-01-01	A	MOV (Delta)	3	30	EUR 30
2019-01-01	A	MOV (Delta)	4	60	EUR 60
2019-01-01	A	MOV (Delta)	2	90	EUR 90
2019-01-01	A	MOV (Delta)	8	120	EUR 120

Start Date	Contract ID	Pattern KF Type	Period Unit	Period To	Result Value
2019-01-01	B	MOV (Delta)	4	30	EUR 30
2019-01-01	B	MOV (Delta)	4	60	EUR 60
2019-01-01	B	MOV (Delta)	4	90	EUR 90

Start Date	Contract ID	Pattern KF Type	Period Unit	Period To	Result Value
2019-01-01	C	MOV (Delta)	4	30	EUR 30
2019-01-01	C	MOV (Delta)	4	60	EUR 60
2019-01-01	C	MOV (Delta)	4	90	EUR 90
2019-01-01	C	MOV (Delta)	4	120	EUR 120

## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TS	Term Selection	Term Selection	Active			

Rule Line

Line	Line Granularity	Day Count Convention	Start Date	Period Type	Value Type	Period To	Period To Result	Period Unit Result
1	Contract ID	German 30/60	Start Date	Monthly/ Quarterly		Period To	Period To Result	Period Unit Result

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique
Line Granularity	Determines the size of one partition of data	Mandatory input that has to be inserted in the <i>Granularity</i> tab of the <i>Signature</i> section
Day Count Convention	Day count convention used to determine the result period	Mandatory input.  The function manages the following conventions : <ul style="list-style-type: none"> <li>• US 30/360</li> <li>• Actual/Actual</li> <li>• EU 30/360</li> <li>• German 30/360</li> </ul>
Start Date	Starting date to compute the day count	Mandatory input
Period Type	Determines the final pattern structure (for example, monthly or quarterly).	Mandatory input
Period To	Day count of the starting pattern for the cash flow	
Period To Result	The system writes the period type of the output to this field.	Mandatory output

Field	Meaning	Notes
Period Unit Result	Period type of output data (for example, monthly or quarterly )	<p>Mandatory input depending on the value that has been defined under <i>Period Type</i>.</p> <p>The following values are possible:</p> <ul style="list-style-type: none"> <li>• 6 if <i>Period To</i> is in months</li> <li>• 11 if <i>Period To</i> indicates quarters</li> <li>• 7 if <i>Period To</i> indicates years</li> </ul>

## Expected Output

### Expected Output for Period Type = Monthly

Start Date	Contract ID	Pattern KF Type	Period To	Period Unit	Period To Result
2019-01-01	A	MOV (Delta)	30	6	1
2019-01-01	A	MOV (Delta)	60	6	2
2019-01-01	A	MOV (Delta)	90	6	3
2019-01-01	A	MOV (Delta)	120	6	4

Start Date	Contract ID	Pattern KF Type	Period To	Period Unit	Period To Result
2019-01-01	B	MOV (Delta)	30	6	1
2019-01-01	B	MOV (Delta)	60	6	2
2019-01-01	B	MOV (Delta)	90	6	3

Start Date	Contract ID	Pattern KF Type	Period To	Period Unit	Period To Result
2019-01-01	C	MOV (Delta)	30	6	1
2019-01-01	C	MOV (Delta)	60	6	2
2019-01-01	C	MOV (Delta)	90	6	3
2019-01-01	C	MOV (Delta)	120	6	4

### Expected Output for Period Type = Quarterly

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
2019-01-01	A	MOV (Delta)	90	11	1
2018-01-01	A	MOV (Delta)	120	11	2

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
2019-01-01	B	MOV (Delta)	90	11	1

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
2019-01-01	C	MOV (Delta)	90	11	1
2019-01-01	C	MOV (Delta)	120	11	2

### 1.3.10.7 Example: Term Target

Enriches a given set of cash flow data (based on a daily periodicity) and returns a cash flow structure that uses a consistent periodicity of months, quarters or years, and that can be based on different sorts of day count conventions (30/360 German, ACT/ACT). It also interpolates the values of the terms added to the original pattern structure.

In the example, we present a pattern for MOV values (delta values) that are broken into smaller periods. The function proportionally spreads the amount between the resulting ranges.

#### Input Data

Contract ID	Start Date	Pattern Key Figure Type	Value	Period From	Period To	Period Unit
A	2019-01-01	MOV (Delta)	EUR 90	1	90	4
A	2019-01-01	MOV (Delta)	EUR 120	91	120	4

Contract ID	Start Date	Pattern Key Figure Type	Value	Period From	Period To	Period Unit
B	2019-01-01	MOV (Delta)	EUR 90	1	90	4

Contract ID	Start Date	Pattern Key Figure Type	Value	Period From	Period To	Period Unit
C	2019-01-01	MOV (Delta)	EUR -	1	90	4
C	2019-01-01	MOV (Delta)	EUR 120	91	120	4



## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TT	Term Target	Term Target	Active			

Rule Line

Line	*Line Granularity	*Start Date	Cut-Off Comparison	*Value Type	*Day Count Convention	*Period Type	*Period From	*Period To	*Value (Result)	*Period To Result	*Period Unit Result	Period Cut-Off
1	Contract ID	Start Date	-	Pattern KF Type	German 30/60	Monthly	Period From	Period To	Value	Period To Result	Period Unit Result	

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique.
Line Granularity	Determines the size of one partition of data	Mandatory input that has to be inserted in the <i>Granularity</i> section of the <i>Signature</i> tab
Start Date	Starting date to compute the day count	Mandatory input
Cut-Off Comparison	Cut-off date for a preparational step	All items prior to or equal to this date are deleted later if the rule type flow cut-off is also applied.
Value Type	This field can have the value "MOV" or "BAL" to determine which value type the cash flow item is made of	Mandatory input. Ensure that the field contains the values "MOV" or "BAL" only. These values differ in the way that the starts of the period are managed.
Day Count Convention	Day count convention used to determine the result period	Mandatory input. The function manages the following conventions : <ul style="list-style-type: none"> <li>• US 30/360</li> <li>• Actual/Actual</li> <li>• EU 30/360</li> <li>• German 30/360</li> </ul>

Field	Meaning	Notes
<b>Period Type</b>	Parameter to indicate the pattern structure of the output	Mandatory input. The possible values are: <ul style="list-style-type: none"> <li>• Monthly</li> <li>• Quarterly</li> <li>• Yearly</li> </ul>
<b>Period From</b>	Start of the range	Mandatory input
<b>Value (Result)</b>	This field is both the input field for the amount to be spread and the field used to store the result.	Mandatory input/output
<b>Period To Result</b>	New pattern structure	Mandatory output
<b>Period Unit Result</b>	Period unit of the pattern	Mandatory output depending on <i>Value Type</i>

## Expected Output

Start Date	Contract ID	Pattern KF Type	Period From	Period To	Period Unit	Period Unit Result	Period To Result	Result Value
2019-01-01	A	MOV (Delta)	91	120	4	6	0	EUR 0
2019-01-01	A	MOV (Delta)	1	90	4	6	30	EUR 30
2019-01-01	A	MOV (Delta)	1	90	4	6	60	EUR 60
2019-01-01	A	MOV (Delta)	1	90	4	6	90	EUR 90
2019-01-01	A	MOV (Delta)	91	120	4	4	120	EUR 120

Start Date	Contract ID	Pattern KF Type	Period From	Period To	Period Unit	Period Unit Result	Period To Result	Result Value
2019-01-01	B	MOV (Delta)	1	90	4	6	0	EUR 0
2019-01-01	B	MOV (Delta)	1	90	4	6	30	EUR 30
2019-01-01	B	MOV (Delta)	1	90	4	6	60	EUR 60
2019-01-01	B	MOV (Delta)	1	90	4	6	90	EUR 90

Start Date	Contract ID	Pattern KF Type	Period From	Period To	Period Unit	Period Unit Result	Period To Result	Result Value
2019-01-01	C	MOV (Delta)	91	120	4	6	0	EUR 0
2019-01-01	C	MOV (Delta)	1	90	4	6	30	EUR 30
2019-01-01	C	MOV (Delta)	1	90	4	6	60	EUR 60
2019-01-01	C	MOV (Delta)	1	90	4	6	90	EUR 90
2019-01-01	C	MOV (Delta)	91	120	4	4	120	EUR 120

If we change the *Period Type* from “Monthly” to “Quarterly” the result for contract A is as follows:

Contract ID	Start Date	Pattern KF Type	Period From	Period To	Period Unit	Value	Period To Result	Period Unit Result
A	2019-01-01	MOV	91	120	4	0	0	11
A	2019-01-01	MOV	1	9	4	90	90	11
A	2019-01-01	MOV	91	120	4	120	120	11

### 1.3.10.8 Example: Term To Date

Converts a given set of terms of cash flows into dates referencing a given start date.

The configuration allows you to choose between a default approach and an approach that applies different logic to distinguish between pattern items that are of either balance type (cumulative factor/amount values) or movement type (delta factors/amounts).

The examples below are both for patterns of type MOV (delta) and of type BAL (cumulative).

#### Input Data with Pattern MOV (Delta) Values

Start Date	Contract ID	Pattern KF Type	Period Unit	Period To
2019-01-01	A	MOV (Delta)	6	1
2019-01-01	A	MOV (Delta)	6	2

Start Date	Contract ID	Pattern KF Type	Period Unit	Period To
2019-01-06	B	MOV (Delta)	6	1
2019-01-06	B	MOV (Delta)	6	2

Start Date	Contract ID	Pattern KF Type	Period Unit	Period To
2019-01-01	C	MOV (Delta)	6	1
2019-01-01	C	MOV (Delta)	6	2
2019-01-01	C	MOV (Delta)	6	3
2019-01-01	C	MOV (Delta)	6	4

## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TTD	Term To Date	Term To Date	Active			

Rule Line

Line	*Line Granularity	*Period Type	*Start Date	*Date Determinant	*Day of Month	*Period	*Value Type	*Result Date
1	Contract ID	Monthly/Day of Period	Start Date	End of Period	BLANK/25	Period To	Pattern Key Figure Type	Result Date

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique
Line Granularity	Determines the size of each partition of data	Mandatory input that has to be inserted in the <i>Granularity</i> section of the <i>Signature</i> tab
Period Type	Determines the final pattern structure (for example, monthly or quarterly)	Mandatory input
Start Date	Starting date to compute the day count	Mandatory input

Field	Meaning	Notes
<b>Date Determinant</b>	Specifies how the date is determined	Mandatory input.  Codes: <ul style="list-style-type: none"> <li>• 1 = Start of period</li> <li>• 2 = Middle of period</li> <li>• 3 = End of period</li> <li>• 4 = Actual day of period</li> <li>• 5 = Day of period</li> </ul>
<div style="border: 1px solid #ccc; background-color: #f0f0f0; padding: 5px;"> <p><b>i Note</b></p> <p>If you use code "5", the <i>Day of Month</i> field is mandatory.</p> </div>		
<b>Day of Month</b>	Determines the result date if <i>Date Determinant</i> is set to <i>Day of Month</i>	
<b>Period To</b>	Day count of the starting pattern for the cash flow	Mandatory input
<b>Value Type</b>	Can have either the value "MOV" or "BAL", and determines which value type the cash flow item is comprised of.	Ensure that the field contains these two values only. This is important because different logic is applied for movements / balance types on the first date of the resulting pattern.
<b>Result Date</b>	The function writes the new date to this field.	Mandatory output

## Expected Output

### Expected Output for Input MOV (Delta Values) – Period Type = Monthly

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
2019-01-01	A	MOV (Delta)	1	6	2019-01-31
2019-01-01	A	MOV (Delta)	2	6	2019-02-28

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
2019-01-01	B	MOV (Delta)	1	6	2019-06-30
2019-01-01	B	MOV (Delta)	2	6	2019-07-31

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
2019-01-01	C	MOV (Delta)	1	6	2019-01-31

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
2019-01-01	C	MOV (Delta)	2	6	2019-02-28
2019-01-01	C	MOV (Delta)	3	6	2019-03-31
2019-01-01	C	MOV (Delta)	4	6	2019-04-30

### Expected Output for Input MOV (Delta Values) – Period Type = Day of Period

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
2019-01-01	A	MOV (Delta)	1	6	2019-01-25
2019-01-01	A	MOV (Delta)	2	6	2019-02-25

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
2019-01-01	B	MOV (Delta)	1	6	2019-06-25
2019-01-01	B	MOV (Delta)	2	6	2019-07-25

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
2019-01-01	C	MOV (Delta)	1	6	2019-06-25
2019-01-01	C	MOV (Delta)	2	6	2019-07-25
2019-01-01	C	MOV (Delta)	3	6	2019-03-25
2019-01-01	C	MOV (Delta)	4	6	2019-04-25

### Input Data with Pattern BAL (Cumulative) Values

Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To
2019-01-01	D	BAL (cumulative)	7	0
2019-01-01	D	BAL (cumulative)	7	1
2019-02-01	E	BAL (cumulative)	11	0
2019-02-01	E	BAL (cumulative)	11	1
2019-02-01	E	BAL (cumulative)	11	2
2019-02-01	E	BAL (cumulative)	11	3

## Flow Modeling Configuration

Rule Line

Line	*Line Gran-ularity	*Period Type	*Start Date	*Date De-terminant	*Day of Month	*Period	*Value Type	*Result Date
1	Contract ID	Quarterly	Start Date	End of Pe-riod		Period To	Pattern Key Figure Type	Result Date

### Expected Output for Input BAL (Cumulative Values) – Period Type = Quarterly

Start Date	Contract ID	Pattern Key Fig-ure Type	Period Unit	Period To	Result Date
2019-01-01	D	BAL (cumulative)	7	0	2019-01-01
2019-01-01	D	BAL (cumulative)	7	1	2019-01-31
2019-02-01	E	BAL (cumulative)	11	0	2019-02-01
2019-02-01	E	BAL (cumulative)	11	1	2019-03-31
2019-02-01	E	BAL (cumulative)	11	2	2019-06-30
2019-02-01	E	BAL (cumulative)	11	3	2019-09-30

### 1.3.10.9 Example: Value Conversion

Comprises two different calculation methods that can be used either to sum up cash flow items over a given set of terms (running total), or to calculate the delta values between a given set of cash flow items (balance = cumulative values, movement = delta values).

This example shows one of the methods. Starting with MOV data (delta values), we produce BAL data (cumulative values) for the cash flows.

With the configuration presented in the example, only the cash flow items of key figure type "Movement" are converted into balance values. Pattern items, which are of the type balance, will not be converted.

### Input Data

Start Date	Contract ID	Pattern Key Figure Type	Period To	Value
2019-01-01	A	MOV (Delta)	1	USD 90

Start Date	Contract ID	Pattern Key Figure Type	Period To	Value
2019-01-01	A	MOV (Delta)	2	USD 120
2019-01-01	B	MOV (Delta)	1	USD 90
2019-01-01	C	MOV (Delta)	1	USD 0
2019-01-01	C	MOV (Delta)	2	USD 120

## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
VC	Value Conversion	Value Conversion	Active			

Rule Line

Line	*Line Granularity	*Conversion Type	*Value Type Target	*Value Type Field	*Period To	*Value	*Value Result Field
1	Contract ID	Mov to Bal	Movement	Pattern Key Figure Type	Period To	Value	Result Amount

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique
Line Granularity	Determines the size of one partition of data.	Mandatory input that has to be inserted in the <i>Granularity</i> tab of the <i>Signature</i> section.
Conversion Type	Specifies the calculation method used.	Mandatory input. "Mov to Bal" computes the running total, whereas "Bal to Mov" calculates delta values.
Value Type Target	Determines which key figure types (movement or balance) are converted	Mandatory input
Value Type Field	Can have the value "MOV" or "BAL", and determines which value type the cash flow item is comprised of.	Mandatory input Ensure that the field contains these two values only.
Period To	Determines the structure of the pattern.	Mandatory input



Field	Meaning	Notes
Value Result Field	The system writes the resulting amount to this field.	Mandatory output

## Expected Output

In this example, the rule computes the running total of all cash flow items that have the value "MOV" in the *Pattern Key Figure Type* column.

Start Date	Contract ID	Pattern KF Type	Period To	Value	Result Amount
2019-01-01	A	MOV	1	USD 90	USD 90
2019-01-01	A	MOV	2	USD 120	USD 210
2019-01-01	B	MOV	1	USD 90	USD 90
2019-01-01	C	MOV	1	USD 0	USD 0
2019-01-01	C	MOV	2	USD 120	USD 120

Input Data BAL

Start Date	Contract ID	Pattern KF Type	Period To	Value
2019-01-01	A	BAL	1	USD 90
2019-01-01	A	BAL	2	USD 120
2019-01-01	B	BAL	1	USD 90
2019-01-01	C	BAL	1	USD 0
2019-01-01	C	BAL	2	USD 120

The Flow Modeling configuration is the same; however, the *Conversion Type* is now "Bal to Mov" and *Value Type Target* is now "Balance".

Output Data BAL

Start Date	Contract ID	Pattern KF Type	Period To	Value	Result Amount
2019-01-01	A	BAL	1	USD 90	USD 90
2019-01-01	A	BAL	2	USD 120	USD 30
2019-01-01	B	BAL	1	USD 90	USD 90
2019-01-01	C	BAL	1	USD 0	USD 0
2019-01-01	C	BAL	2	USD 120	USD 120

## 1.3.10.10 Example: Incremental Value Calculation

The *Incremental Value Calculation* configuration, based on a vector of dates with two patterns (factors) assigned, combines the two factors and computes a mutual pattern.

In the example, we use a set of dates and their two factor patterns for a single contract as basis. The function does the following:

- It derives a new structure for the dates from the date of the pattern (start date in the example).
- It calculates a mutual pattern. Using the wording of the example, the mutual pattern is computed to retrieve the sum of the mutual pattern equal to the incurred pattern per incurred date. Moreover, the sum of the mutual pattern for a due date is equal to the sum of the due pattern for that due date (shown in the example).

### Input Data

Contract ID	Date	Incurred Pattern	Due Pattern
A	2017-01-31	0.44	0.09
A	2017-02-28	0.36	0.37
A	2017-03-31	0.14	0.00
A	2017-04-30	0.06	0.29
A	2017-05-31	0.00	0.25

### Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping Fields	Rule Ordering Fields	Selection
IVC	Incremental Value Calculation	Incremental Value Calculation	Active			

Rule Line

*Granularity Fields	*Period	*First Value Field	*Second Value Field	*First Period Field	*Second Period Field	*Value
Contract ID	Start Date	Incurred Pattern	Due Pattern	Incurred Date	Due Date	Mutual Pattern

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique.
Granularity Fields	Determines the size of one partition of data	Mandatory input that has to be inserted in the <i>Granularity</i> section of the <i>Signature</i> tab.
Period	Term structure of the pattern	Mandatory input
First Value Field	First pattern value	Mandatory input
Second Value Field	Second pattern value	Mandatory input
First Period Field	Main date of the new term structure	Mandatory output
Second Period Field	Secondary date of the new term structure	Mandatory output
Value	Resulting mutual pattern	Mandatory output

## Expected Output and Explanation

The following table shows the expected output:

Contract ID	Incurred Date	Due Date	Mutual Pattern
A	2017-01-31	2017-01-31	0.09
A	2017-01-31	2017-02-28	0.18
A	2017-01-31	2017-03-31	0.00
A	2017-01-31	2017-04-30	0.09
A	2017-01-31	2017-05-31	0.08
A	2017-02-28	2017-02-28	0.19
A	2017-02-28	2017-03-31	0.00
A	2017-02-28	2017-04-30	0.09
A	2017-02-28	2017-05-31	0.08
A	2017-03-31	2017-03-31	0.00
A	2017-03-31	2017-04-30	0.08
A	2017-03-31	2017-05-31	0.06
A	2017-04-30	2017-04-30	0.03
A	2017-04-30	2017-05-31	0.03
A	2017-05-31	2017-05-31	0.00

Interim results explained below will be calculated by the system in all datasets based on dates from 2017-01-31 until 2017-05-31.

The following table is the input table with 5 unique records.

Contract ID	Date	Incurred Pattern	Due Pattern	
A	2017-01-31	0.44	0.09	Record 1
A	2017-02-28	0.36	0.37	Record 2
A	2017-03-31	0.14	0.00	Record 3
A	2017-04-30	0.06	0.29	Record 4
A	2017-05-31	0.00	0.25	Record 5

### A) Explaining Output Field: First Period Field – in this example in the “Incurred Date” column

*First Period Field* starts with the date of the record from the input data. It produces 5 unique sets of data records.

Contract ID	Incurred Date	
A	2017-01-31	Taken from date of record 1
A	2017-02-28	Taken from date of record 2
A	2017-03-31	Taken from date of record 3
A	2017-04-30	Taken from date of record 4
A	2017-05-31	Taken from date of record 5

### B) Explaining Output Field: Second Period Field – in this example in the “Due Date” column

*Second Period Field* produces records based on dates taken from records 1 to 5, starting from the date registered in the *First Period Field* until the date of the last record. For example, for record 1, it starts with due date 2017-01-31 and continues until the date of record 5, which is 2017-05-31.

Contract ID	Incurred Date	Due Date
A	2017-01-31	2017-01-31
		2017-02-28
		2017-03-31
		2017-04-30
		2017-05-31
A	2017-02-28	2017-02-28
		2017-03-31
		2017-04-30
		2017-05-31
A	2017-03-31	2017-03-31
		2017-04-30
		2017-05-31

Contract ID	Incurred Date	Due Date
A	2017-04-30	2017-04-30
		2017-05-31
A	2017-05-31	2017-05-31

At this point the new term structure is created.

### C) Explaining Output Field: Value – in this example “Mutual Pattern”

Based on the *Primary Period Field (Incurred Date)*, the sum of the mutual pattern for that incurred date that is equal to the incurred pattern of that incurred date (it is indicated only for the first incurred date, but it is true also for the other dates).

The following table shows the pattern of the input data again:

Contract ID	Start Date	Incurred Pattern	Due Pattern	Record
A	2017-01-31	0.44	0.09	Record 1
A	2017-02-28	0.36	0.37	Record 2
A	2017-03-31	0.14	0.00	Record 3
A	2017-04-30	0.06	0.29	Record 4
A	2017-05-31	0.00	0.25	Record 5

Contract ID	Incurred Date	Due Date	Mutual Pattern
A	2017-01-31	2017-01-31	0.09
A	2017-01-31	2017-02-28	0.18
A	2017-01-31	2017-03-31	0.00
A	2017-01-31	2017-04-30	0.09
A	2017-01-31	2017-05-31	0.08
A	2017-02-28	2017-02-28	0.19
A	2017-02-28	2017-03-31	0.00
A	2017-02-28	2017-04-30	0.09
A	2017-02-28	2017-05-31	0.08
A	2017-03-31	2017-03-31	0.00
A	2017-03-31	2017-04-30	0.08
A	2017-03-31	2017-05-31	0.06
A	2017-04-30	2017-04-30	0.03
A	2017-04-30	2017-05-31	0.03
A	2017-05-31	2017-05-31	0.00

The sum of the first five mutual patterns is 0.44. The same as the incurred pattern for the corresponding start date.

## D) Explaining Output Field: Value – in this example “Mutual Pattern”

Based on the Secondary Period Field (*Due Date*), the sum of the mutual pattern for a due date is equal to the due pattern of that due date (it is indicated only for the second due date, but it is true also for the other dates).

Contract ID	Start Date	Incurred Pattern	Due Pattern	
A	2017-01-31	0.44	0.09	Record 1
A	<b>2017-02-28</b>	0.36	<b>0.37</b>	Record 2
A	2017-03-31	0.14	0.00	Record 3
A	2017-04-30	0.06	0.29	Record 4
A	2017-05-31	0.00	0.25	Record 5

Contract ID	Incurred Date	Due Date	Mutual Pattern
A	2017-01-31	2017-01-31	0.09
A	2017-01-31	<b>2017-02-28</b>	<b>0.18</b>
A	2017-01-31	2017-03-31	0.00
A	2017-01-31	2017-04-30	0.09
A	2017-01-31	2017-05-31	0.08
A	2017-02-28	<b>2017-02-28</b>	<b>0.19</b>
A	2017-02-28	2017-03-31	0.00
A	2017-02-28	2017-04-30	0.09
A	2017-02-28	2017-05-31	0.08
A	2017-03-31	2017-03-31	0.00
A	2017-03-31	2017-04-30	0.08
A	2017-03-31	2017-05-31	0.06
A	2017-04-30	2017-04-30	0.03
A	2017-04-30	2017-05-31	0.03
A	2017-05-31	2017-05-31	0.00

In the same way as for the *Second Value Field* for record 2, the field *Due Pattern* is also 0.37.

### 1.3.10.11 Example: Redistribution

This function calculates estimate values before the Reference Date for Redistribution (RDR) and redistributes them to the future periods after this date.

There are two possible methods: *Snow Canon* (SC) and *Snow Plough* (SP).

In the Snow Plough method the values are redistributed to the first value date field after the RDR.

In the Snow Canon method the values are redistributed proportionally to all the value date fields after the RDR. The following two examples show the differences between the two methods.

## Input Data for Snow Canon

Contract ID	Incurred Date	Reported Date	Due Date	Reference Date of Redistribution	Redistribution Method	Amount
A	2017-01-01	2017-02-01	2017-03-01	2017-02-23	SC	200
A	2017-01-01	2017-02-01	2017-04-01	2017-02-23	SC	400
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SC	300
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SC	100

## Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
RED	Redistribution	Redistribution	Active			

Input Fields

*Redistribution Method	*Reference Date of Redistribution	*Granularity Fields	*Date Determinant	Day of Month
Redistribution Method	Redistribution Reference Date	Incurred Date	Start Day of Period	

Changing Fields

*Value Date Field	*Value	Cleanup Fields
Reported Date	Amount	Due Date

## Key Configuration Description

Field	Meaning	Notes
Redistribution Method	Indicates whether the redistribution method is "Snow Canon" or "Snow Plough".	Mandatory input. Values: <ul style="list-style-type: none"> <li>• SC: Snow Canon</li> <li>• SP: Snow Plough</li> </ul>
Reference Date for Redistribution	Represents the threshold for the redistribution of the amounts.	Mandatory input
Granularity Field	Determines the size on one partition of data.	Mandatory input that has to be added to the <i>Granularity</i> section in the <i>Signature</i> tab.

Field	Meaning	Notes
Date Determinant	Specifies how the date should be determined.	Mandatory input Possible values: <ul style="list-style-type: none"> <li>• Start date of the period</li> <li>• Mid date of the period</li> <li>• End date of the period</li> <li>• Actual date of period</li> <li>• Day of period</li> </ul>
Day of Month	Indicates the day of the month where <i>Date Determinant</i> is set equal to <i>Day of Period</i> .	
Value Date Field	Date that is compared to <i>Reference Date for Redistribution</i>	Mandatory input/output. Values in this field where the date is before the <i>Reference Date for Redistribution</i> will be redistributed (depending on the method).
Value	Amount for the redistribution	Mandatory input/output
Cleanup Field	This date is adjusted to perform the redistribution.	

## Redistribution

**Redistributed Records:** The system selects the data with a *Reported Date* before the "Redistribution Reference Date".

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	2017-01-01	2017-02-01	2017-03-01	2017-02-23	SC	200
A	2017-01-01	2017-02-01	2017-04-01	2017-02-23	SC	400

**Records to be Redistributed with Allocation Factors:** The system calculates the allocation factors to redistribute the amounts.

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SC	300
						75%
						(= 300/400)



Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount	
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SC	100	25%
							(=100/400)

The amount of the new record is calculated as follows:

- $200 \times 0.75 = 150$
- $200 \times 0.25 = 50$
- $400 \times 0.75 = 300$
- $400 \times 0.25 = 100$

## Exected Result for Snow Canon

Contract ID	Incurred Date	Reported Date	Due Date	Reference Date of Redistribution	Redistribution Method	Amount
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SC	150
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SC	300
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SC	300
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SC	50
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SC	100
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SC	100

## Input Data for Snow Plough

Contract ID	Incurred Date	Reported Date	Due Date	Reference Date of Redistribution	Redistribution Method	Amount
A	2017-01-01	2017-02-01	2017-03-01	2017-02-23	SP	200
A	2017-01-01	2017-02-01	2017-04-01	2017-02-23	SP	400
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SP	300
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SP	100
A	2017-01-01	2017-05-01	2017-06-01	2017-02-23	SP	500

The flow modeling configuration is the same as in the previous example. The only difference is the value in the field *Redistribution Method* ("SP" stands for the Snow Plough method).

## Expected Output for Snow Plough

Contract ID	Incurred Date	Reported Date	Due Date	Reference Date of Redistribution	Redistribution Method	Amount
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SP	150
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SP	300
A	2017-01-01	2017-04-01	2017-05-01	2017-02-23	SP	300
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SP	50
A	2017-01-01	2017-05-01	2017-06-01	2017-02-23	SP	0
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SP	100
A	2017-01-01	2017-05-01	2017-06-01	2017-02-23	SP	0
A	2017-01-01	2017-04-01	2017-06-01	2017-02-23	SP	100
A	2017-01-01	2017-05-01	2017-06-01	2017-02-23	SP	500

### 1.3.10.12 Example: Scale Factor

Ratio of two corresponding values with similar field/data types (division).

In the example, we calculate the weighted claim (claim/total premium) for a pattern for three terms for a contract.

#### Input Data

Contract ID	Date	Total Premium	Claim
A	2019-01-01	1000	100
A	2019-02-01	1000	200
A	2019-03-01	1000	300

#### Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
SF	Scale Factor	Scale Factor	Active			

Line	*Numerator Value	*Denominator Value	Scale Factor
1	Claim	Total Premium	Weighted Claim

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique.
Numerator Value	Numerator of the ratio	Mandatory input
Denominator Value	Denominator of the ratio	Mandatory input
Scale Factor	Fields where the output is written	Mandatory output.  The user has to add the field in the <i>Action</i> section of the <i>Signature</i> tab.

## Expected Output

Contract ID	Date	Total Premium	Claim	Weighted Claim
A	2019-01-01	1000	100	0.1
A	2019-02-01	1000	200	0.2
A	2019-03-01	1000	300	0.3

### 1.3.10.13 Example: Scaling

Applies a scale factor through multiplication.

## Input Data

Contract ID	Date	Weight	Ultimate
A	2019-01-01	0.33	1000
A	2019-02-01	0.33	1000
A	2019-03-01	0.33	1000

## Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
SC	Scaling	Scaling	Active			

Line	*Value	*Scale Factor	Granularity Fields	*Scaled
1	Ultimate	Weight	Contract ID	Scale Value

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique
Value	Multiplicand	Mandatory input
Scale Factor	Multiplier	Mandatory input
Granularity Fields	Determines the size on one partition of data	Mandatory input. The user has to add the field in the <a href="#">Granularity</a> section of the <a href="#">Signature</a> tab.
Scaled	Field where the output is written	Mandatory output. The user has to add the field in the <a href="#">Action</a> section of the <a href="#">Signature</a> tab.

Interim Result before Value Adjustment

Contract ID	Date	Weight	Ultimate	Scale Value
A	2019-01-01	0.33	1000	330
A	2019-02-01	0.33	1000	330
A	2019-03-01	0.33	1000	330

### i Note

1000 - 330 - 330 - 330 =10 added to last record.

## Expected Data

Contract ID	Date	Weight	Ultimate	Scale Value
A	2019-01-01	0.33	1000	330

Contract ID	Date	Weight	Ultimate	Scale Value
A	2019-02-01	0.33	1000	330
A	2019-03-01	0.33	1000	340

### 1.3.10.14 Example: Acknowledgment of Actuals

Enriches actuals by applying matching logic to determine date information missing earlier in the life cycle, and also determines the regime for every cashflow item.

The matching of actuals to estimates can be carried out in the following ways:

- If the CF calculation is "01", the system matches the actual to the estimate based on the business date of the actual. This means, *Settled Date* for settled transactions, *Due Date* for due transactions and *Reported Date* for reported actuals.
- If the CF calculation is "02", the system matches the actual to the estimates based only on the Secondary Risk Incurred Date. This is used in L&H business where the missing dates in the actuals are predetermined by a reverse life cycle conversion based on models.

In this example the CF calculation is "01" and the dates for an actual (Cf Indicator = 03) are enriched starting from a pattern of estimates (Cf Indicator = 01). The tables below contain the most important values and parameters.

#### CF\_CALC

01	Used for P&C
02	Used for L&H

#### CF\_INDICATOR

01	Estimate
02	Reported Actual
03	Due Business Transaction
04	Settled Business Transaction

#### REGIME

01	Follow Actuals
02	Reflect Actuals
03	Follow Estimates
04	Estimated Future

#### PERIOD TYPE

Monthly
Quarterly
Annual

## Input Data

Cf Calc	Contract	Coverage	Category	Cf Indicator	Pr In-curred Date	Pr Re-ported Date	In-curred Date	Re-ported Date	Due Date	Set-tled Date	Set-tled Amount	Currency	BT ID	Hold Back Date	RA HBD	Key Date	Re-gime
01	RIC_Q101 DT	COV_1DT	1010	1	2018-01-01	2018-02-25	2018-01-05	2018-02-25	2018-03-25	2018-04-25	100	EUR		2018-01-01	2018-01-01	2018-05-05	
01	RIC_Q101 DT	COV_1DT	1010	1	2018-01-01	2018-03-25	2018-01-05	2018-03-25	2018-04-25	2018-05-25	200	EUR		2018-01-01	2018-01-01	2018-05-05	
01	RIC_Q101 DT	COV_1DT	1010	1	2018-01-01	2018-04-25	2018-03-25	2018-04-25	2018-05-25	2018-06-25	300	EUR		2018-01-01	2018-01-01	2018-05-05	
01	RIC_Q101 DT	COV_1DT	1010	3					2018-05-04	2018-06-25	150	EUR	DUE_01	2018-01-01	2018-01-01	2018-05-05	
01	RIC_Q101 DT	COV_1DT	1010	1	2018-01-01	2018-05-25	2018-04-25	2018-05-25	2018-06-25	2018-07-25	400	EUR		2018-01-01	2018-01-01	2018-05-05	

## Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
AOA	Acknowledgement of Actuals Same Period	Acknowledge of Actuals	Active			

### Input Fields

*Actuarial Granularity Fields	Contract, Cost Category, Coverage
*BT Granularity Fields	BT ID
*First Hold Back Date Field	Hold Back Date
*Second Hold Back Date Field	Ra HBD
*Business Date Field	Key Date
*Period Type	Monthly
*Life Cycle Step Field	Cf Indicator
CF Calculation Field	CF Calc
Match Basis	Same Period Match

## Changing Fields

*Amount Field	Settled Amount
Prim. Risk Incurred Date Field	Incurred Date
*Sec. Risk Incurred Date Field	Pr. Incurred Date
Prim. Risk Reported Date Field	Reported Date
Sec. Risk Reported Date Field	Pr. Reported Date
Due Date Field	Due Date
*Settled Date Field	Settled Date
Regime Field	Regime

## Key Configuration Description

Field	Meaning	Notes
<b>Actuarial Granularity Fields</b>	List of fields that uniquely identify a set of cash flows as a group	Mandatory input.  The fields have to be inserted into the <i>Granularity</i> section of the <i>Signature</i> tab.
<b>BT Granularity Fields</b>	List of fields that uniquely identify a set of cash flows as a group	Mandatory input
<b>First Hold Back Date Field</b>	Defines the date after which the model-based cashflows take precedence.	
<b>Second Hold Back Date Field</b>	Defines the date from which additional incurred date has to be calculated as a factor of actuals to be accounted as estimates.	
<b>Business Date Field</b>	Defines the key date for which the estimated cashflows are projected.	
<b>Period Type</b>	Defines the periodicity used as a basis for matching actuals to estimates.	
<b>Life Cycle Step Field</b>	Defines the definition of cashflow: estimate and various actual types.	Mandatory input
<b>CF Calculation Field</b>	Distinguishes between Life & Health and Property & Casualty Businesses.	
<b>Match Basis</b>	Determines the matching logic.	Possible Values: <ul style="list-style-type: none"> <li>• Same period match</li> <li>• Past period match</li> <li>• Future period match</li> <li>• All periods match</li> </ul>
<b>Regime Field</b>	Determines the regime to which the cashflow item belongs to.	

## Output Result

Cf Calc	Contract	Coverage	Cost Category	Cf Indicator	Pr Incurred Date	Pr Reported Date	In-curred Date	Re-ported Date	Due Date	Set-tled Date	BT ID	Hold Back Date	Key Date	RA HBD	Re-gime	Set-tled Amount	Cur-rency
01	RIC_Q101 DT	COV_1DT	1010	03	2018-01-01	2018-04-25	2018-03-25	2018-04-25	2018-05-04	2018-06-25	DUE_01	2018-01-01	2018-05-05	2018-01-01	3	150	EUR
01	RIC_Q101 DT	COV_1DT	1010	01	2018-01-01	2018-02-25	2018-01-05	2018-02-25	2018-03-25	2018-04-25		2018-01-01	2018-05-05	2018-01-01	3	100	EUR
01	RIC_Q101 DT	COV_1DT	1010	01	2018-01-01	2018-03-25	2018-02-25	2018-03-25	2018-04-25	2018-05-25		2018-01-01	2018-05-05	2018-01-01	3	200	EUR
01	RIC_Q101 DT	COV_1DT	1010	01	2018-01-01	2018-04-25	2018-03-25	2018-04-25	2018-05-04	2018-06-25		2018-01-01	2018-05-05	2018-01-01	3	300	EUR
01	RIC_Q101 DT	COV_1DT	1010	01	2018-01-01	2018-05-25	2018-04-25	2018-05-25	2018-06-25	2018-07-25		2018-01-01	2018-05-05	2018-01-01	1	400	EUR

### 1.3.10.15 Example: Life Cycle Conversion

Applies the lags and lag factors delivered by the actuarial input to the cash flow stream in the form of a lag factor pattern, to determine the amounts and date for the lifecycle stage.

The example below shows a cash flow pattern. In this pattern, the reported date is one month after the incurred date. An amount is shown for each line of the pattern. What we want to do is to determine a new date (due date), based on the other dates, using the factor to spread the reported amount.

## Input Data

Contract	Coverage	Cost Category	Cf Indicator	Incurred Date	Re-ported Date	Due Date	Cal Freq Code	Re-ported Amount	Currency	Factor
RIC_Q101 DT	COV_Q101 DT	1010	01	2019-02-25	2019-03-25		6	200	EUR	0.4
RIC_Q101 DT	COV_Q101 DT	1010	01	2019-02-25	2019-03-25		6	200	EUR	0.6



Contract	Coverage	Cost Category	Cf Indicator	Incurred Date	Re-reported Date	Due Date	Cal Freq Code	Re-reported Amount	Currency	Factor
RIC_Q10 1DT	COV_Q10 1DT	1010	01	2019-03-25	2019-04-25		6	300	EUR	0.4
RIC_Q10 1DT	COV_Q10 1DT	1010	01	2019-03-25	2019-04-25		6	300	EUR	0.6
RIC_Q10 1DT	COV_Q10 1DT	1010	01	2019-04-25	2019-05-25		6	400	EUR	0.4
RIC_Q10 1DT	COV_Q10 1DT	1010	01	2019-04-25	2019-05-25		6	400	EUR	0.6

## Flow Modeling Configuration

### Rules

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
LFC	R2D	Life Cycle Conversion	Active			

### Input Fields

Life Cycle Reversed	<b>Life Cycle Conversion</b>
Client Reporting Frequency	
Date Determinant	Day of Period
Date Field	Reported Date
Date of Month	25
Lag Factor Value	Factor
Lag Factor Frequency	Cal Freq Code
LCC Frequency Field	Contract, Cost Category, Coverage, Incurred Date, Reported Date
Period	Period
Value	Reported Amount

### Output Fields

Life Cycle Reversed	<b>Life Cycle Conversion</b>
Life Cycled Amount	Due Amount
Life Cycled Date	Due Date

## Key Configuration Description

Fields	Meaning	Notes
Life Cycle Reversed	Defines whether the life cycle process determines a date in the future or in the past.	Mandatory input
Client Reporting Frequency	Only applies in the incurred to reported life cycle convention step.	The value is delivered as part of the master data.
Date Determinant	Defines how the date should be determined.	Mandatory input.  Codes: <ul style="list-style-type: none"> <li>• 1, Start of the period</li> <li>• 2, Mid of the period</li> <li>• 3, End of the period</li> <li>• 4, Actual day of the period</li> </ul>
Date Field	Defines the date which will be used as the input for the Life Cycle Conversion Step.	Mandatory input
Date of Month	Defines the day of the result date according to the configuration.	
Lag Factor Value	Defines the factors to be applied for determining the amounts per life cycled date.	Mandatory input
Lag Factor Frequency	Defines the periodicity of the lag factors (monthly, quarterly, annual etc...)	Mandatory input
LCC Granularity Field	A list of fields that uniquely identify a set of cash flows as a group.	Mandatory input
Period	A list of fields that uniquely identify a set of cash flows as a group.	Mandatory input
Value	Defines the number of periods to be applied for the date determination.	Mandatory input
Life Cycled Amount	Field where the resulting amount is written.	Mandatory output
Life Cycled Date	Field where the resulting date is written.	Mandatory output

## Expected Output

Contract	Coverage	Cost Category	Cf Indicator	Incur-red Date	Re-ported Date	Due Date	Re-ported Amount	Period	Factor	Due Amount	Currency	Cal Freq Code
RIC_Q1 01DT	COV_Q 101DT	1010	01	2019-0 2-25	2019-0 3-25	2019-0 3-25	200	00000 0	0.4	80	EUR	6
RIC_Q1 01DT	COV_Q 101DT	1010	01	2019-0 2-25	2019-0 3-25	2019-0 4-25	200	00000 1	0.6	120	EUR	6
RIC_Q1 01DT	COV_Q 101DT	1010	01	2019-0 3-25	2019-0 4-25	2019-0 4-25	300	00000 0	0.4	120	EUR	6
RIC_Q1 01DT	COV_Q 101DT	1010	01	2019-0 3-25	2019-0 4-25	2019-0 5-25	300	00000 1	0.6	180	EUR	6
RIC_Q1 01DT	COV_Q 101DT	1010	01	2019-0 4-25	2019-0 5-25	2019-0 5-25	400	00000 0	0.4	160	EUR	6
RIC_Q1 01DT	COV_Q 101DT	1010	01	2019-0 4-25	2019-0 5-25	2019-0 6-25	400	00000 1	0.6	240	EUR	6

### 1.3.10.16 Example: Modulation In

The table below explains the general concept of Modulation In.

Scenarios	Previous Contract	Following Contract	Rule
First scenario	No	Yes	ModIn totals ModOut multiplied by -1 with <i>Incurred Date</i> = "Contract Start Date"
Second scenario	No	Yes	ModIn totals ModOut multiplied by -1 with <i>Incurred Date</i> = "Contract Start Date"
			ModIn of following contract multiplied by -1 with <i>Incurred Date</i> = "Start Date of following Contract"
Third scenario	Yes	Yes	ModIn should be calculated
			ModIn of following contract multiplied by -1 with <i>Incurred Date</i> = "Start Date of following Contract"

The following example shows how this configuration is calculated in practice.

## Input Data

Contract	Coverage	Old Quota Share R.	Quota Share R.	Value Mod
CONTRACT_A	COVERAGE_A	10	10	100
CONTRACT_A	COVERAGE_B	10	20	100
CONTRACT_A	COVERAGE_C	10	5	100
CONTRACT_B	COVERAGE_A	10	10	100
CONTRACT_B	COVERAGE_B	10	20	200
CONTRACT_B	COVERAGE_C	0.1	1	1

## Flow Modeling Configuration

Rules

Rule	Description	Rule Type	State	Rule Grouping Fields	Rule Ordering Fields	Selection
MDI	Modulation In	Modulation In	Active			

Input Fields

Old Share Field	Contract End Date
New Share Field	Exposure Date
Granularity Field	Contract Coverage

Changing Fields

Value	Value Mod
-------	-----------

Calculation logic: (New Share Field/Old Share Field)\*(-Value)

## Expected Output

Contract	Coverage	Old Quota Share R.	Quota Share R.	Value Mod
CONTRACT_A	COVERAGE_A	10	10	-100
CONTRACT_A	COVERAGE_B	10	20	-200
CONTRACT_A	COVERAGE_C	10	5	-50
CONTRACT_B	COVERAGE_A	10	10	-100
CONTRACT_B	COVERAGE_B	10	20	-400
CONTRACT_B	COVERAGE_C	0.1	1	-10

## 1.3.10.17 Example: Modulation Out

Selects amounts in the basis cash flow where the exposure date is greater than the coverage end date. If Modulation Out for contract T is updated and contract T+1 exists, Modulation In for contract T+1 must be (re-)calculated.

Scenarios	Previous Contract	Following Contract	Rule
First scenario	No	No	ModOut is based on UnMod
Second scenario	No	Yes	ModOut is based on UnMod
Third scenario	Yes	Yes	ModOut is based on UnMod

Here is an example to explain the computation that this configuration does in practice.

### Input Data

Contract	Contract End Date	Exposure Date	Value Mod
CONTRACT_D	2019-01-01	2019-01-01	100
CONTRACT_A	2019-01-01	2019-01-19	200
CONTRACT_B	2019-01-01	2019-01-15	300
CONTRACT_C	2019-01-01	2019-01-02	400

If the exposure date is greater than the contract end date, the system calculates the days between the two dates, calculates a factor based on the day counts (depending on the *Day Count Convention*) and multiplies the value for that factor with a negative sign.

For the second line of the input data (the first has Contract End Date=Exposure Date and is excluded from the computation) we will get:

- 18 days between the two dates
- 30 days for January (in the convention German 30/360 all the months have 30 days).

So, the result will be  $(18/30)*(-200) = -120$

### Flow Modeling Configuration

Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping Fields	Rule Ordering Fields	Selection
MDO	Modulation	Modulation Out	Active			

#### Input Fields

<b>Contract End Date</b>	Contract End Date
<b>Exposure Date</b>	Exposure Date
<b>Day Count Convention</b>	German 30/360

#### Changing Fields

<b>Value</b>	Value Mod
--------------	-----------

## Key Configuration Description

Field	Meaning	Notes
<b>Contract End Date</b>	End date of the contract	Mandatory input
<b>Exposure Date</b>	Exposure date	Mandatory input. If Exposure Date > Contract End Date, Modulation Out is performed.
<b>Day Count Convention</b>	Convention to compute the day count	Mandatory input
<b>Value</b>	Value field	Mandatory output

## Expected Output

Contract	Contract End Date	Exposure Date	Value Mod
CONTRACT_A	2019-01-01	2019-01-19	-120
CONTRACT_B	2019-01-01	2019-01-15	-140
CONTRACT_C	2019-01-01	2019-01-02	-400

### 1.3.10.18 Example: Item Number Generation

Separates each partition by creating a number for each one. To do this, the system needs a *Granularity* field (which separates the partitions from each other) and an *Item Number* field (which is filled by this rule type).

In the example, we mark the cash flow pattern according to the version number: when the version number increases, we add one to the *Item Number*. Only the *Granularity* field (Contract ID) is relevant for the calculation.

## Input Data

Version ID	Start Date	Contract ID	Period From	Period To
1	2019-01-01	A	1	3
1	2019-01-01	A	4	4
2	2020-01-16	B	1	2
2	2020-01-16	B	3	8
3	2019-01-01	C	1	3
4	2019-01-01	D	1	2
4	2019-01-01	D	3	8

## Flow Modeling Configuration

Rules

Rule	Description	Rule Type	State	Rule Grouping Fields	Rule Ordering Fields	Selection
IN	CF Item Number	Item Number Generation	Active			

Rule Line

Line	*Item Number Granularity Fields	*Item Number
L1	Contract ID	Item ID

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique. A single Key Figure Formula Rule can run more lines.
Item Number Granularity Field	Determines the size of one partition of data	Mandatory input. You have to add the field in the <i>Granularity</i> section of the <i>Signature</i> tab.
Item Number	Fields where the output is written	Mandatory output. You have to add the field in the <i>Action</i> section of the <i>Signature</i> tab.

## Expected Output

Version ID	Start Date	Contract ID	Item ID
1	2019-01-01	A	1
1	2019-01-01	A	2
2	2020-01-16	B	1
2	2020-01-16	B	2
3	2019-01-01	C	1
4	2019-01-01	D	1
4	2019-01-01	D	2

### 1.3.10.19 Example: Cash Flow Regime

This configuration is helpful to manage cash flows that follow the distinction between Estimates, Reported Actuals, Due Business Transaction and Settled Business Transaction.

This function adjusts the cashflow stream based on the regime into which each of the cashflows falls. In the context of ECP the regimes that are managed are:

- Follow Actuals (O1)  
This regime comprises only the effect of actuals. Therefore the system removes any estimate item that falls in this regime from the final cashflow.
- Reflect Actuals (O2)  
This regime comprises only the effect of actuals. Model-based estimates do not have any effect in this regime. However, more actuals may be expected to be reported in this period. The system therefore calculates an additional incurred estimate as a factor of the actuals. These additional incurred estimates will have the same date information as that of the actuals but the amounts will be calculated as a factor of the actuals and will apply the formula  $(\text{Amount} * \text{Factor} / (1 - \text{Factor}))$ .
- Follow Estimates (O3)  
In this regime, the model-based estimates are expected to be effective. Therefore, for every actual that falls into this regime an additional negated estimate is introduced into the cashflow with the same date information as the actuals.
- Estimated Future (O4)  
In this regime, no actual information is expected.

We show three simple examples for each of the three regimes (the first three require configuration because different behaviors are expected). The key fields and parameters of the function are explained below.



## Values for Key Fields of the Examples

### CF\_CAL

01	P&C
02	L&H

### CF\_INDICATOR

01	Estimate
02	Reported Actual
03	Due Business Transaction
04	Settled Business Transaction

### REGIME

01	Follow Actuals
02	Reflect Actuals
03	Follow Estimate
04	Estimated Future

## Example: Follow Actuals

### Input Data

Cf Calc	Con tract	Cov- erage	Cost Cate- gory	Cf Indi- cator	Pr In- cur- red Dat e	Pr Re- port Dat e	In- cur- red Dat e	Re- port Dat e	Due Dat e	Set- tled Dat e	Set- tled Amo unt	Cur- renc y	BT_I D	Hold Bac k Dat e	RA HBD	Key Dat e	Re- gim e	Fac- tor
02	RIC_	COV	1010	01	2017	2017	2017	2017	201	201	50	EUR		201	2017	201	01	0.00
	Q10	_Q1			-11-0	-12-	-11-2	-12-	8-01	8-02				8-01	-11-3	8-05		0
	1DT	01D			1	25	5	25	-25	-25				-31	0	-15		
		T																
02	RIC_	COV	1010	03	201	201	201	201	201	201	150	EUR		201	2017	201	01	0.00
	Q10	_Q1			8-01	8-04	8-03	8-04	8-05	8-06				_01	8-01	-11-3	8-05	0
	1DT	01D			-01	-25	-25	-25	-14	-25				-31	0	-15		
		T																

## Flow Modeling Configuration

Rules (the same for all regimes)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
CFR	CF Regime	Cash Flow Re- gime	Active			

Rule Line

Line	Line Type	Description
L001	CF Regime: Follow Actuals	FA

\*Input Fields

Life Cycle Step Field	Cf Indicator
Regime	Regime

### Key Configuration Description

Field	Meaning	Notes
Life Cycle Step Field	Indicates the nature of the entry	Mandatory input
Regime	Indicates the regime that has to be applied	Mandatory input

### Expected Output

Cf Calc	Contract	Coverage	Category	Indicator	Pr In-curred Date	Pr Reported Date	In-curred Date	Re-reported Date	Due Date	Settled Date	Settled Amount	Currency	BT ID	Hold Back Date	RA HBD	Key Date	Regime	Factor
02	RIC_Q10	COV_Q1	1010	03	2018-01-01	2018-04-25	2018-03-25	2018-04-25	2018-05-14	2018-06-25	150.000	EUR	DUE_01	2018-01-31	2017-11-30	2018-05-15	01	0.000
		1DT		T														

### Example: Reflect Actuals

#### Input Data

Cf Calc	Contract	Coverage	Category	Indicator	Pr In-curred Date	Pr Reported Date	In-curred Date	Re-reported Date	Due Date	Settled Date	Settled Amount	Currency	BT ID	Hold Back Date	RA HBD	Key Date	Regime	Factor
02	RIC_Q10	COV_Q1	1010	03	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-15	2018-04-25	75.000	EUR	DUE_02	2018-01-31	2017-11-30	2018-05-15	02	0.700
		1DT		T														
02	RIC_Q10	COV_Q1	1010	03	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-15	2018-04-25	100.000	EUR		2018-01-31	2017-11-30	2018-05-15	03	0.700
		1DT		T														

Rule Line

Line	Line Type	Description
L001	CF Regime: Reflect Actuals	FA

\*Input Fields

BT Granularity Fields	BT_ID
Amount	Settled Amount
Factor	Factor
Life Cycle Step Field	Cf Indicator
Regime	Regime

### Key Configuration Description

Field	Meaning	Notes
BT Granularity Fields	Fields that define the partition of data	Mandatory input
Amount	Field that contains the amounts	Mandatory input
Factor	Factor that has to be applied	Mandatory input
Life Cycle Step Field	Indicates the nature of the entry	Mandatory input
Regime	Indicates the regime that has to be applied	Mandatory input

### Expected Output

Cf Calc	Contract	Coverage	Category	Cf Indicator	Pr In-curred	Pr Report Date	In-curred Date	Re-report Date	Due Date	Settled Date	BT ID	Currency	Factor	Hold Back Date	Key Date	RA HDB	Regime	Settled Amount
02	RIC_Q10	COV_Q1	1010	03	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-15	2018-04-25	DUE_Q102	EUR	0.70	2018-01-31	2018-05-15	2017-11-30	02	75.00
02	RIC_Q10	COV_Q1	1010	01	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-25	2018-04-25		EUR	0.70	2018-01-31	2018-05-15	2017-11-30	03	100.00
02	RIC_Q10	COV_Q1	1010	01	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-15	2018-04-25		EUR	0.70	2018-01-31	2018-05-15	2017-11-30	02	175.00

The added amount 175 is computed according to the formula (Amount\*Factor/(1-Factor)) as (75\*0.7/(1-0.7)).

## Example: Follow Estimates

### Input Data

Cf Calc	Contract	Coverage	Category	Cost Cf Indicator	Pr In-curred Date	Pr Reported Date	In-curred Date	Re-reported Date	Due Date	Settled Date	Settled Amount	Currency	BT ID	Hold Back Date	RA HDB	Key Date	Re-gime	Factor
02	RIC_Q10	COV_Q1	1010	03	2018-01-01	2018-02-01	2018-01-25	2018-02-25	2018-03-15	2018-04-25	75.00	EUR	DUE_02	2018-01-31	2017-11-30	2018-05-15	02	0.70
02	RIC_Q10	COV_Q1	1010	01	2018-01-01	2018-02-01	2018-01-25	2018-02-25	2018-03-15	2018-04-25	100.00	EUR		2018-01-31	2017-11-30	2018-05-15	01	0.70

Rule Line

Line	Line Type	Description
L001	CF Regime: Follow Estimates	FA

\*Input Fields

BT Granularity Fields	BT_ID
Amount	Settled Amount
Life Cycle Step Field	Cf Indicator
Regime	Regime

### Key Configuration Description

Field	Meaning	Notes
BT Granularity Fields	Fields that define the partition of data	Mandatory input
Amount	Field that contains the amounts	Mandatory input
Life Cycle Step Field	Indicates the nature of the entry	Mandatory input
Regime	Indicates the regime that has to be applied	Mandatory input

## Expected Output

Cf Calc	Contract	Coverage	Cost Category	Cf Indicator	Pr Incurred	Pr Reported Date	In-curred Date	Re-reported Date	Due Date	BT ID	Cur-rency	Factor	Hold Back Date	Key Date	RA HDB	Re-gim e	Set-tled Amo unt	Set-tled Dat e
02	RIC_Q10	COV_Q1	1010	03	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-15		EUR	0.70	2018-01-31	2018-05-15	2017-11-03	03	-75.00	2018-04-25
02	RIC_Q10	COV_Q1	1010	01	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-15	DUE_02	EUR	0.70	2018-01-31	2018-05-15	2017-11-03	03	75.00	2018-04-25
02	RIC_Q10	COV_Q1	1010	01	2018-01-01	2018-02-25	2018-01-25	2018-02-25	2018-03-15		EUR	0.70	2018-01-31	2018-05-15	2017-11-03	03	100.00	2018-04-25

The function added a new entry with a negative amount.

## 1.3.10.20 Example: Clear Actual Dates

Clears the actual date information in the cashflows arising from actuals.

We show below a simplified pattern. The function clears the dates according to a predefined logic that depends on the values in the field inserted in *Life Cycle Step* field. The result is the same pattern but with the dates cleared.

## Input Data

CF Indicator	Contract	Coverage	Cost Category	PR Incurred Date	PR Reported Date	Incurred Date	Re-reported Date	Due Date	Settled Date	Settled Amount
03	RIC_Q10DT	COV_Q10	1010	2017-01-1	2017-12-2	2017-11-2	2017-12-2	2018-01-25	2018-02-25	50.000
01	RIC_Q102DT	COV_Q10	1010	2017-01-1	2017-12-2	2017-11-2	2017-12-2	2018-01-25	2018-02-25	50.000
02	RIC_Q103DT	COV_Q10	1010	2017-01-1	2017-12-2	2017-11-2	2017-12-2	2018-01-25	2018-02-25	50.000

## Flow Modeling Configuration

Rules

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
CAD	Clear Actual Dates	Clear Actual Dates	Active			

Rule Line

Line	*Life Cycle Step Field	*Pr. Risk Incurred Date	*Sec. Risk Incurred Date	*Pr. Risk Reported Date	*Sec. Risk Reported Date	*Due Date	*Settled Date
L1	CF Indicator	Pr. Incurred Date	Incurred Date	Pr. Reported Date	Reported Date	Due Date	Settled Date

## Key Configuration Description

Field	Meaning	Notes
Line	Identifier of the line	Must be unique. A single Key Figure Formula Rule can run multiple lines.
Life Cycle Step Field	Defines the logic for cancellation of the dates	<p>Mandatory input.</p> <p>Four codes are possible:</p> <ul style="list-style-type: none"> <li>• 1: The dates are not deleted</li> <li>• 2: All the dates are deleted except <i>Due Date</i> and <i>Settled Date</i></li> <li>• 3: All the dates are deleted but <i>Settled Date</i></li> <li>• 4: All the dates are deleted.</li> </ul>
Pr. Risk Incurred	Date of the pattern	Mandatory input and output
Sec. Risk Incurred Date	Date of the pattern	Mandatory input and output
Pr. Risk Reported Date	Date of the pattern	Mandatory input and output
Sec. Risk Reported Date	Date of the pattern	Mandatory input and output
Due Date	Date of the pattern	Mandatory input and output
Settled Date	Date of the pattern	Mandatory input and output

## Expected Output

Cf Indica- tor	Contract	Coverage	Cost Cat- egory	Pr Incur- red Date	Pr Re- ported Date	Incurred Date	Reported Date	Due Date	Settled Date
01	RIC_Q101 DT	COV_Q101 DT	1010	2017-01-11	2017-12-2 5	2017-11-25	2017-12-2 5	2018-01-2 5	2018-02-2 5
02	RIC_Q101 DT	COV_Q101 DT	1010					2018-01-2 5	2018-02-2 5
03	RIC_Q101 DT	COV_Q101 DT	1010						2018-02-2 5
04	RIC_Q101 DT	COV_Q101 DT	1010						

### 1.3.10.21 Example: Incurred to Reported Factor Calculation

Calculates the *Incurred to Reported* lags in cases where these lags are not delivered directly, using the more granular *Policy Holder to Primary Insurer* and *Primary Insurer to Reinsurer* lags.

In brief, depending on the selections, the system computes a combined factor and a new structure for the cash flows.

## Input

Input Data

Contract	Coverage	Cost Cate- gory	RAAP	LFP Type	LFP SUB- CAT	Period	Cal Freq Code	Factor
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	000000	11	0.200
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	000001	11	0.100
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	000002	11	0.700
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PI2RI	000001	11	0.600
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PI2RI	000002	11	0.400

## Flow Modeling Configuration

Rules

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
I2R	Incurred to Reported Factor Calculation	Incurred to Reported Factor Calculation	Active			LFP_SUBCAT=(LFP_SUBCAT! = ' ')

Input Fields

Input Fields	Value
*Lag Factor Pattern Granularity Fields	Cal Freq Code, Contract, Cost Category, Coverage, LFP Subcat, LFP Type
*Lag Factor Type Field	LFP Type
*LFP Sub Category	LFP Subcat
Insured to Insurer Lag Factor Type	Policy Holder to Primary Insurer
Insurer to Reinsurer Lag Factor Type	Primary Insurer to Reinsurer
*Lag Factor Type	Incurred to Reported

Changing Fields

Changing Fields	Value
*Lag Factor Value Field	Factor
*Period	Period

## Key Configuration Description

Field	Meaning	Notes
*Lag Factor Pattern Granularity Fields	Determines the size of one partition of data.	Mandatory input. You have to add the field in the <a href="#">Granularity</a> section of the <a href="#">Signature</a> tab.
*Lag Factor Type Field	Indicates the life cycle of the cash flow.	Mandatory input. The field must contains the following values: <ul style="list-style-type: none"> <li>• IN2RE (Incurred to Reported)</li> <li>• RE2DU (Reported to Due)</li> <li>• DU2SE (Due to Settled)</li> </ul>



Field	Meaning	Notes
*LFP Sub Category	Indicates the subcategory of the pattern.	Mandatory input.  The field must contain the following values: <ul style="list-style-type: none"> <li>• PH2PI (Policy Holder to Primary Insurer)</li> <li>• PI2RI (Primary Insurance to Reinsurer)</li> </ul>
Insured to Insurer Lag Factor Type	Manages how to combine the data for the calculation (see the example).	
Insurer to Reinsurer Lag Factor Type	Manages how to combine the data for the calculation (see the example).	
*Lag Factor Type	Acts as a filter on the field <i>Lag Factor Type Field</i>	Mandatory input
*Lag Factor Value Field	Where the combined factor is written.	Mandatory input/output.  You have to add the field in the <i>Action</i> section of the <i>Signature</i> tab.
*Period	Where the new period structure is written.	Mandatory input/output.  You have to add the field in the <i>Action</i> section of the <i>Signature</i> tab.

## Expected Output

Contract	Coverage	Cost Category	RAAP	LFP Type	LFP SUB-CAT	Period	Cal Freq Code	Factor
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	1	11	0.120
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	2	11	0.140
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	3	11	0.460
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	4	11	0.280

## Additional Information on Calculation Logic

PERIOD						
LFP_SUBCAT	000000	000001	000002	000003	000004	
PH2PI	0.2	0.1	0.7			
PI2RI	0	0.6	0.4			
Calculation for PH2PI = 0.2	$0.2*0=0$	$0.2*0=0$	$0.2*0.4=0.08$			
Calculation for PH2PI = 0.1		$0.1*0=0$	$0.1*0.6=0.06$	$0.1*0.4=0.04$		
Calculation for PH2PI = 0.7			$0.7*0=0$	$0.7*0.6=0.42$	$0.7*0.4=0.28$	
IN2RE	0	$0.12+0=0.12$	$0.08+0.06+0=0.14$	$0.04+0.42=0.46$	0.28	New Period Structure

### 1.3.10.22 Example: Factors for Additional Incurred

In the context of the Estimated Cash Flow Preparation, this function is usually used to calculate the factors to be applied in the *Reflect Actual Regime* based on the delivered *Policy Holder to Primary Insurer* lags.

Factors are calculated as a difference between 1 and the cumulated sum of the policy holder to primary insurer lag factors for a certain granularity. The number of periods is determined by the *Reflect Actuals* attachment point.

## Input

Input Data

Contract	Coverage	Cost Category	RAAP	LFP Type	LFP SUB-CAT	Period	Cal Freq Code	Factor
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	000000	11	0.200
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	000001	11	0.100
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PH2PI	000002	11	0.700
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PI2RI	000001	11	0.600
RIC_Q101D T	COV_Q101D T	1010	000001	IN2RE	PI2RI	000002	11	0.400

### Input Fields

Lag Factor Pattern Granularity Fields	Contract, Cost Category, Coverage
Lag Factor Frequency Field	Cal Freq Code
Date Determinant	Actual Date of Period
Period Field	Period
RA Attachment Pt. Field	RAAP
Hold Back Date Field	Hold Back Date

## Flow Modeling Configuration

### Rules (Tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
I2R	Factors for Additional Incurred Cash Flows	Factors for Additional Incurred Cash Flows	Active			<pre>LFP_TYPE = 'IN2RE'; LFP_SUBC AT = 'PH2PI'</pre>

## Output

### Output Fields

Lag Factor Pattern Granularity Fields	Factor
Lag Factor Frequency Field	Period Start Date
Date Determinant	Period End Date

### Output Data

Con-tract	Cover-age	Cost Cate-gory	RAAP	LFP Type	LFP SUB-CAT	Period	Cal Freq Code	Factor	Hold Back Date	Period End Date	Period Start Date
RIC_Q1 01DT	COV_Q1 01DT	1010	000000	IN2RE	PH2PI	1	11	0.70	2018-01-31	2018-01-31	2018-01-01
RIC_Q1 01DT	COV_Q1 01DT	1010	000000	IN2RE	PH2PI	2	11	0.00	2018-01-31	2017-12-31	2017-12-01

## 1.3.11 File Adapter

The *File Adapter* function provides automated access to files so that file content can be imported as input for calculations and results can be exported as file content.

### Key Features

#### Header

In the header, you define the principal behavior of the *File Adapter* function. The following fields are available:

- *File IO Type*:
  - Import: The purpose of the File Adapter function is to import data from a server file.
  - Export: The purpose of the File Adapter function is to export data into a server file.
- *File Format*: Refers to the definition of a file format, which is maintained centrally for the environment.
- *File Name*: Specifies the name of the file on the server that is used.
- *Header Row*: Defines the row number in which the header columns are available. The value 0 means that there is no header row.
- *Number of Threads*: This is only relevant for import. Multiple threads can reduce the import time. The maximum permitted value is 256.
- *Batch Size*: Specifies the number of records to be inserted in each commit.
- *Table Lock*: If this is set, the data for column store tables is loaded faster.
- *No Type Check*: Specifies that the records are inserted without checking the each field type.
- *Fail on Invalid Data*: Specifies that the import fails unless all the entries are imported without errors.

#### Server Files

This is a helper tab that has no influence on the runtime of the function.

The *Refresh Directory List* button shows a list of files that are currently available on the server. The content of these files can also be viewed here. Use the *Select File* button to register it in the header as the file name to be used.

Small files can also be uploaded and downloaded but we recommend that you use server-side IT-driven mechanisms to manage files in the server directory.

#### Preview

This is a helper tab that has no influence on the runtime of the function

Once the file name is set in the header, the Preview b allows you to preview the file.

#### Stage

This is a helper tab that has no influence on the runtime of the function.

Once the file name is set in the header, the *Stage* tab allows you to stage the file in a temporary table separating the data into columns. This makes it easier to analyze data, including filtering, sorting, and checking.

## Mapping

The file columns can be mapped to existing fields in the environment. The [Field Mapping Proposal](#) button helps you to match columns to field names.

Optionally, formulas can be defined to convert data.

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.11.1 Example: Importing CSV Files

In this example, we import a CSV file from a file server to SAP Profitability and Performance Management.

#### Note

It is important that you set the file IO type to "IMPORT".

## Input Data

ZE_CUST (Customer)	ZE_PROD (Product)	ZX_QTY (Quantity)	ZE_AMT (Amount)
CN002	PROD01	40	23
CN001	PROD01	20	20
CN004	PROD03	30	33
CN003	PROD03	30	25
CN001	PROD02	33	40
CN004	PROD01	25	28
CN002	PROD02	25	30

## Procedure

### Uploading a CSV to the file server

Convert the input data above into a CSV file format and upload it to the server `/usr/sap/trans70/hana` (the whitelist path).

You can configure the whitelist path under [Environment > File Format Tab](#).

## Server File Tab

1. In the function details, while in edit mode, choose [Upload](#).
2. Choose the input data that you want to upload. Select the file name of the CSV file.

### i Note

This is the CSV file created from the data in the *Input Data* section.

## Use the CSV file in the function

1. Once you have uploaded the file to the server, choose the [Refresh Directory List](#) button.
2. Select .csv file, then choose the [Select File](#) button.
3. The system displays the file name in the header section as “ <File name>.csv”.

## Mapping of Fields

1. You can map the file columns to existing fields in the environment. In edit mode, choose the [Field Mapping Proposal](#) button. This button helps to match columns to field names.
2. In the *File* column, there should be a field named ZX\_QTY (Quantity).
3. Map ZX\_QTY to ZE\_QTY by assigning ZE\_QTY in the *Field* column.

## Expected Result

### i Note

[File Adapter](#) is an integration function to consume text file (CSV) records that are used as input for SAP Profitability and Performance Management. The system does not carry out any additional processing in this scenario except for reading data from the Excel file and preparing it for consumption by the processing functions, like [Allocation](#) or [Join](#).

The result provided by the function can be used either as input for other functions or for calculation. In this example, the result must be structured as shown in the table below.

### i Note

Now ZE\_QTY is stated instead of ZX\_QTY since this is the configuration done on the mapping tab.

ZE_CUST (Customer)	ZE_PROD (Product)	ZE_QTY (Quantity)	ZE_AMT (Amount)
CN002	PROD01	40	23
CN001	PROD01	20	20
CN004	PROD03	30	33
CN003	PROD03	30	25
CN001	PROD02	33	40
CN004	PROD01	25	28

ZE_CUST (Customer)	ZE_PROD (Product)	ZE_QTY (Quantity)	ZE_AMT (Amount)
CN002	PROD02	25	30

## 1.3.12 Remote Function Adapter

The *Remote Function Adapter* function provides automated communication capabilities to other applications and systems so that they can be included in calculations and processes.

### Key Features

#### Header

- Function Types:
  - Finance Account Payable
 

This function type enables you to post an accounts payable entry to SAP ERP or SAP S/4HANA. Each record is mapped to one posting, a GL account on one side and a vendor account on the other. If an RFC destination is defined at the environment level, the entries are posted in that system.

For FI-GL integration scenarios with SAP ERP or SAP S/4HANA, this functionality will enable posting of general ledger documents from SAP Profitability and Performance Management.
  - Finance Accounts Payable/Receivable:
 

This function type enables you to post either an accounts payable or accounts receivable entry. Each record is mapped to one posting, a GL account on one side and a customer or a vendor account on the other. If an RFC destination is defined at the environment level, the entries are posted in that system.

For FI-GL integration scenarios with SAP ERP or SAP S/4HANA, this functionality will enable posting of general ledger documents from SAP Profitability and Performance Management.
  - Finance Accounts Receivable
 

This function type enables you to post an accounts receivable entry to SAP ERP or SAP S/4HANA. Each record is mapped to one posting, a customer account on one side and a GL account on the other. If an RFC destination is defined at the environment level, the entries are posted in that system.

For FI-GL integration scenarios with SAP ERP or SAP S/4HANA, this functionality will enable posting of general ledger documents from SAP Profitability and Performance Management.
  - Finance Extended
 

This is a combination of the function types “Finance General Ledger” and “Finance Accounts Payable/Receivable”. If an RFC destination is defined at the environment level, the entries are posted in that system.

For FI-GL integration scenarios with SAP ERP or SAP S/4HANA, this functionality will enable posting of general ledger documents from SAP Profitability and Performance Management.
  - Finance General Ledger
 

This function type enables you to post a general ledger entry to SAP ERP or SAP S/4HANA. Each record is mapped to one posting, a debit and a credit on each side. If an RFC destination is defined at the environment level, the entries are posted in that system.

For FI-GL integration scenarios with SAP ERP or SAP S/4HANA, this functionality will enable posting of general ledger documents from SAP Profitability and Performance Management.

- Finance General Ledger Items  
This function type enables you to post a general ledger entry to SAP ERP or SAP S/4HANA, with multiple lines per document. A grouping field determines which lines are to be grouped into one document. Debit amounts (positive) and credit amounts (negative) should add up to zero.
- HANA R Script  
An external R script procedure is called. The expected interface of the external R script is displayed on the *Rules* tab. The R script can be entered directly on the *Rules* tab.
- HANA Stored Procedure  
An external SAP HANA stored procedure is called. The expected interface of the external SAP HANA stored procedure is displayed on the *Rules* tab. You need to define the authoring schema that the system uses to store the procedure together with the procedure name.
- External Function  
An external function is called, like a remote NetWeaver function or a Web service. The expected interface of the external function is displayed on the *Rules* tab if you are calling the function remotely. You need to define the RFC destination that the system uses to call the function .
- Replicate CO Master Data  
Replicates master data and hierarchies for specific and main controlling module fields like *Cost Center*, *Cost Element*, *Profit Center* etc. from an integrated SAP S/4HANA or SAP ERP system into SAP Profitability and Performance Management environment fields.
- Sales and Distribution  
For integration scenarios with SAP ERP or SAP S/4HANA, this functionality enables creation of sales order, inquiry or quotation in SAP ERP or SAP S/4HANA Sales and Distribution module from SAP Profitability and Performance Management.  
It is mass processing enabled. Multiple and different documents can be created at once. It makes use of the Sales and Distribution BAPIs `BAPI_SALESORDER_CREATEFROMDAT2`, `BAPI_INQUIRY_CREATEFROMDATA2` and `BAPI_QUOTATION_CREATEFROMDATA2`.
- RFC Destination  
The RFC destination defined in this field is used to post documents, call external reports, get master data and hierarchies. The RFC destination must exist in SM59 before use. If no value has been specified, the local client and system is used.
- Field Length Extension  
The information comes from the RFC target system. This is a read-only field in the header that provides information to the modeler.
- Authoring Schema  
This is mapped in the physical schema into access and deploy transported or stored objects using the SAP HANA database.

### **i** Note

You need to map the physical schema that the stored procedure is located in to the authoring schema. For more information, see *Maintain Schema Mapping for NXI Schema* (part of the Administration Guide for SAP Profitability and Performance Management).

- Stored Procedure  
This is the name of the stored procedure to be executed.

All other header options, such as *Include Original Input Data*, *Result Handling*, *Suppress Initial Results* and *Result Model Table* are referred to as “Functional Building Blocks” and are not specific to the Remote Function Adapter function. For more information about these options, see [Header \[page 63\]](#).



## Input

For more information about modeling input, see [Input \[page 64\]](#).

## Rules

- Rule format: Table Form  
This is available for the following remote function types: Finance, Replicate CO Master Data and Sales Distribution
  - Component  
Corresponding component item relevant for the specific remote function adapter
  - Name  
Component description
  - Declaration Type  
Input: Entered field or value that is sent to the corresponding component in the target system defined in the RFC destination. If the RFC destination is not defined, the input will be posted or updated within the source system.  
Output: The field specified here receives the corresponding component value from the target system defined in the RFC destination, and is included in the run result. If the RFC destination is not defined, no value is shown for the field.
  - Is Mandatory  
Green: *Is Mandatory* field is set to "Yes". This indicates that an entry has to be made for this component.  
Orange: *Is Mandatory* field is set to "Yes". This indicates that an entry has not been made for this component.  
Grey: *Is Mandatory* field is set to "No"- This indicates that an entry is not required for this component. You need to make entries in the list for all mandatory fields.
  - Description  
Field description

### i Note

To check which fields have been defined for each function type, proceed as follows :

1. Go to transaction `/n/nxi/pl_mf`
2. From here go to **► Rule Types ► Remote Function Adapter Mapping ►**.
3. The system gives you the option to select an RFA type (for example, Finance General Ledger).
4. This provides you with an overview of the fields defined for the specific function.
5. You can make certain changes to the mapping list. Open a customer incident to report a change request.

- Rule format: Template and Statement  
This is available for the following remote function types: HANA Stored Procedure and External Function
  - Template  
Is used as the template guide for creating the procedure that is called in the remote function adapter – HANA Stored Procedure and External Function.
  - Statement  
The system displays the expected interface, provided you have defined the procedure in the header of the function.

This is available for remote function type: HANA R Script

  - Template

Is used as the template guide for creating scripts that are executed by the remote function adapter – HANA R Script.

- Statement:  
You can enter script directly on the user interface of the remote function adapter.

## RFC Destination

For FI-GL and FI-GLI types, there are two ways of adding the RFC destination:

- Environment level
- Function level

If the RFC destination is maintained on both an environment level and function level, the RFC destination to be followed gets maintained on the function level.

### ❖ Example

Given the following setup:

1. Environment Level – RFC Destination: ABC
2. RFA – FI-GLI – RFC Destination: DEF

the RFA function will post the document in the DEF system.

For SAP Profitability and Performance Management 3.0 SP07 or below, you cannot maintain an RFC destination inside these function types, but you can for SP08 and above. See SAP Note [2902513](#) for further details.

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.12.1 Example of a SAP HANA Stored Procedure

#### Prerequisite

The remote function adapter requires input data which is then processed by the SAP HANA Stored Procedure.

In this example, simple input data is used in the form of a model table, to show the connection and logic of the remote function adapter SAP HANA stored procedure type.

1. Create a model table.
2. Name it "RFA Input", for example.
3. On the right-hand side of the screen, configure the model table by entering the following:
  - *Model Table Source*: Environment
  - *Transport Data*: Default (No)
4. Add a field.

#### ❖ Example

In this example, the field `JB_PROD` (characteristic, length = 50) has been added to show the relationship between input data and the remote function adapter.

5. Choose *Maintain Data* and add an entry.

#### ❖ Example

In this example, the added data is `PRD01`.

##### Product field (JB\_PROD)

Product field (JB_PROD)
PRD01

6. Now the input has been defined and ready to be used.

## Remote Function Adapter

The *Remote Function Adapter* function provides automated communication capabilities to other applications and systems so that they can be included in calculations and processes.

One of these is **HANA STORED PROCEDURE**, where an external SAP HANA stored procedure can be called by the function to process the input function in step 6. .

### How to set up the remote function adapter

1. Create a remote function adapter.
2. Name it "RFA Function", for example.
3. On the right-hand side of the screen, configure the header of the remote function adapter by making the following entries:
  - *Function Type*: **HANA STORED PROCEDURE**
  - *Supress Initial Result*: Default
  - *Result Model Table*: Empty (by default)
  - *Authoring Schema*: <Where your procedure is located>
  - *Stored Procedure*: <The stored procedure created that can process the input data>
  - *Include Original Input Data*: Default
  - *Result Handling*: Default
4. On the right-hand side of the screen, configure the tabs of the remote function adapter.
  - *Input Function*: <Enter the input function that the SAP HANA stored procedure needs to process>  
For the purposes of this example, it is the prerequisite model table "RFA Input" that was created earlier .

- The *Rule* tab has two sections:

1. *Template* Section (Read Only)

This is a generated procedure based on the input function that can be used as a template for the called SAP HANA stored procedure.

The configurator must be aware of the fact that this template is a mandatory format, and that it must be closely checked against the called SAP HANA stored procedure

### ❁ Example

#### ☰ Sample Code

```
CREATE PROCEDURE "<SCHEMA>". "<HANAPROCEDURE>"
(IN it_al "<DEFAULT SCHEMA>". "/NXI/TP1AL",
IN it_input TABLE(JB_PROD NVARCHAR(50)),
OUT ot_result TABLE(JB_PROD NVARCHAR(50), FS_PER_MSG_TEXT_
NVARCHAR(5000), FS_PER_FORMULA_ NVARCHAR(5000)),
OUT ot_msg TABLE(MSGTY NVARCHAR(1), MSG_TEXT NVARCHAR(5000))
) LANGUAGE SQLSCRIPT SQL SECURITY DEFINER AS
BEGIN
END;
```

2. *Statement* Section (Read Only)

This reflects the SQL statement(s) of the called SAP HANA stored procedure.

The parameter section of the SAP HANA stored procedure must be completely filled using the respective template format based on the input function.

In this example, it looks something like this:

#### ☰ Sample Code

```
CREATE PROCEDURE "<SCHEMA>". "<HANAPROCEDURE>"
(IN it_al "<DEFAULT SCHEMA>". "/NXI/TP1AL",
IN it_input TABLE(JB_PROD NVARCHAR(50)),
OUT ot_result TABLE(JB_PROD NVARCHAR(50), FS_PER_MSG_TEXT_
NVARCHAR(5000), FS_PER_FORMULA_ NVARCHAR(5000)),
OUT ot_msg TABLE(MSGTY NVARCHAR(1), MSG_TEXT NVARCHAR(5000))
) LANGUAGE SQLSCRIPT SQL SECURITY DEFINER AS
BEGIN
ot_result = select 'ABC' as JB_PROD, 'MSG_TEXT' as FS_PER_MSG_TEXT_,
'MS_FORMULA' as FS_PER_FORMULA_ from dummy;
ot_msg = select 'I' as MSGTY, 'SUCCESS' as MSG_TEXT from dummy;
END;
```

### i Note

#### Explanation

- *ot\_result* = This is the section where input processing will happen with the help of the SAP HANA stored procedure.  
In the above example, the procedure states that the value "ABC" is assigned to the field *JB\_PROD*, assign "MSG\_TEXT" to field *FS\_PERMSG\_TEXT\_*, and add "MS\_FORMULA" as a value for *FS\_PER\_FORMULA\_* field.  
Since the last two fields are technical fields, the scenario can focus on *JB\_PROD* now getting a value "ABC".
- *ot\_msg* = This is the section where run information can be manipulated and set by the SAP HANA stored procedure.

In the above example, the procedure states that the value "I" or "Information" is assigned as MSGTY (msgtype), and have a word "SUCCESS" as MSG\_TEXT.

Additional Note: This is an actual example of a SAP HANA procedure that SAP Profitability and Performance Management can call in the remote function adapter (this is the same as with the *Statement* section in the *Rule* tab above).

The screenshot shows the SAP HANA Studio interface. On the left, a tree view displays the database catalog, including 'Public Synonyms', 'C5144919', 'Column Views', 'EPM Models', 'EPM Query Sources', 'Functions', 'Indexes', 'Procedures', 'Table Types', 'Sequences', 'Synonyms', and 'Tables'. The 'SAMPLE\_HANA\_SP' procedure is selected. The main editor displays the following SQLScript code:

```
CREATE PROCEDURE "C5144919"."SAMPLE_HANA_SP"( IN it_a1 "SAPPEU"."/NXI/TP1AL",
IN it_input TABLE(JB_PROD NVARCHAR(50)),
OUT ot_result TABLE(JB_PROD NVARCHAR(50),
FS_PER_MSG_TEXT_ NVARCHAR(5000),
FS_PER_FORMULA_ NVARCHAR(5000)),
OUT ot_msg TABLE(MSGTY NVARCHAR(1),
MSG_TEXT NVARCHAR(5000)) ) LANGUAGE SQLSCRIPT SQL SECURITY DEFINER AS
BEGIN -- code
ot_result = select
'ABC' as JB_PROD,
'MSG_TEXT' as FS_PER_MSG_TEXT_,
'MS_FORMULA' as FS_PER_FORMULA_
from dummy
;
ot_msg = select
'I' as MSGTY,
'SUCCESS' as MSG_TEXT
from dummy
;
end
```

3. **Check:** Making entries on this tab is not mandatory, but if a check is required after processing, you can enter proper check conditions on this tab.
4. **Documentation:** Making entries on this tab is not mandatory, but you can use it to document the scenario or provide documentation or instructions about the function being modeled.
5. Once you have completed the set-up, choose *Activate*.
6. Then choose *Run*.

## Example

Here is the result of the remote function adapter function after calling a SAP HANA stored procedure to process the input function:

The screenshot shows the SAP RFA Function (23121) interface. It includes a filter section and a result list section. The result list shows a table with the following data:

Product field ( JB_PROD )	FS_PER_FORMULA_	FS_PER_MSG_TEXT_	MS_FORMULA	MSG_TEXT
ABC	MS_FORMULA	MSG_TEXT		

Here is the application log of the remote function adapter function after calling a SAP HANA stored procedure to process the input function:

Status	Function	Message Text
✓	RFA Function	Run started for Environment=915, Version=JBA, Function=23121, Run Type=RUN
✓	RFA Function	Run Attributes Process=, Activity=, Run=02E0EC2E788F1ED996DB52E2800CF163...
✓	RFA Function	Run Parameters Package=, Package Parameter=, Package Selection=
✓	RFA Function	Input 23113 selected 1 records
✓	RFA Function	..... SUCCESS .....

## 1.3.13 Calculation

*Calculation* is a data enrichment function that can be used to enhance the data in a dataset with calculated attributes based on predefined rules at runtime.

The enriched data can then be used for consumption in downstream processes such as allocation. If the data to be calculated is already available in the source data, the calculated data is only overwritten if the condition values are met. Otherwise, the source values are retained.

### Key Features

#### Header

The calculation function includes a parser to detect dependencies between fields used in formulas and to ensure that rules are executed in the correct order internally. Circular dependencies are not allowed.

In the header, you define the principal behavior of the calculation.

You can use the calculation type as a specific header option:

- **Relative:** The complete input data is run through and each calculation rule is applied where the selected conditions are met. This method is similar to the one used for derivations, but respects dependencies in formulas.
- **Absolute:** The selected conditions define a subset of the input data and the calculations are applied to this subset where the selected conditions are met. This is typically used in planning calculations, where calculations need to be applied to selected line items.

#### i Note

In the calculation type *Absolute*, the fields *Granularity* and *Selection* are used not only for defining the signature but are also used as fields for grouping and ordering. The result is grouped and ordered based on the fields defined in the granularity and selection.

The system applies the calculation formula to the grouping result.

## Rules

Each calculation rule semantically defines an if-then statement. The if part specifies for which records of the input data the rule is relevant. The then part is an action and contains a list of fields and formulas that have to be calculated.

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.13.1 Example: Calculation with Lookup (Relative)

## Input

This is the table that will be used as an input function:

Input 1: CA - Data Table 1

Channel	Account	Customer	Product	Quantity	Amount
90AH5	1	CN007	PROD01	80	17.9
90AH4	3	CN008	PROD02	100	14.6
90AH2	2	CN002	PROD01	40	23
90AH5	1	CN002	PROD02	25	30
90AH1	1	CN001	PROD01	20	20
90AH3	3	CN003	PROD03	30	25
90AH3	3	CN005	PROD01	20	24.5
90AH4	1	CN001	PROD02	33	40
90AH2	1	CN004	PROD01	25	28
90AH4	3	CN006	PROD01	40	21.2
90AH1	2	CN004	PROD03	30	33
90AH5	1	CN009	PROD02	75	11.3

This is the table that will be used as the lookup input:

Input 2: CA - Data Table 2

Channel	Account	Customer	Product	Quantity	Amount
90AH3	3	CN003	PROD03	15	62.5

Channel	Account	Customer	Product	Quantity	Amount
90AH4	1	CN001	PROD02	89	250
90AH1	2	CN004	PROD03	99	206.25
90AH4	3	CN006	PROD01	112	132.5
90AH5	3	CN010	PROD03	55	20
90AH1	1	CN001	PROD01	76	125
90AH2	2	CN002	PROD01	80	143.75
90AH1	1	CN004	PROD01	103	175
90AH2	1	CN009	PROD02	126	70.63
90AH4	1	CN001	PROD02	17	100
90AH5	1	CN002	PROD02	13	75
90AH5	1	CN007	PROD01	40	44.75
90AH5	2	CN002	PROD01	20	57.5
90AH3	3	CN005	PROD01	108	153.13
90AH5	1	CN007	PROD01	117	111.88
90AH1	2	CN004	PROD03	15	82.5
90AH3	3	CN005	PROD01	10	61.25
90AH5	1	CN009	PROD02	75	28.25
90AH2	1	CN004	PROD01	13	70
90AH5	1	CN002	PROD02	94	187.5
90AH5	3	CN010	PROD03	130	50
90AH1	1	CN001	PROD01	10	50
90AH4	3	CN006	PROD01	20	53
90AH4	3	CN008	PROD02	50	36.5
90AH3	3	CN003	PROD03	85	156.25
90AH4	3	CN008	PROD02	121	91.25

## Calculation

1. Collect all entries containing "90AH5" on the *Channel* field of Data Table 1.

CA - Data Table 1

Channel	Account	Customer	Product	Quantity	Amount
90AH5	1	CN007	PROD01	80	17.9
90AH5	1	CN002	PROD02	25	30



Channel	Account	Customer	Product	Quantity	Amount
90AH5	1	CN009	PROD02	75	11.3

2. Collect all entries containing "90AH5" on the *Channel* field of Data Table 2 and sum up the "Amount" field.

CA - Data Table 2

Channel	Account	Customer	Product	Quantity	Amount
90AH5	3	CN010	PROD03	55	20
90AH5	1	CN009	PROD02	126	70.63
90AH5	1	CN002	PROD02	13	75
90AH5	1	CN007	PROD01	40	44.75
90AH5	1	CN007	PROD01	117	111.88
90AH5	1	CN009	PROD02	75	28.25
90AH5	1	CN002	PROD02	94	187.5
90AH5	3	CN010	PROD03	130	50
				Aggregated Amount of 90AH5	588.01

3. The *Amount* field will be populated with the result of the assigned formula on the *Rules* tab ( $\text{Quantity} * \text{MTCA2.Amount} [\text{Channel}=90\text{AH5}] / 2$ ), where  $\text{Quantity} = \text{Quantity from Data Table 1 and MTCA2./ZQA/ZMT/}[\text{Channel}=90\text{AH5}] = \text{Aggregated Amount of 90AH5 from Data Table 2}$ .

Channel	Account	Customer	Product	Quantity	Amount
90AH5	1	CN007	PROD01	80	23,520.40

The *Amount* field will then have this formula:

$\text{Quantity} * \text{MTCA2.Amount} [\text{Channel}=90\text{AH5}] / 2$

When we compute the following values for each field, we will have the following output of amount:

$80 * 588,010 / 2$ , whereby 588,010 is the aggregated amount of 90AH5 of Data Table 2.

Channel	Account	Customer	Product	Quantity	Amount
90AH5	1	CN002	PROD02	25	7,350.13

The *Amount* field will then have this formula:

$\text{Quantity} * \text{MTCA2.Amount} [\text{Channel}=90\text{AH5}] / 2$

When we compute the following values for each field, we will have the following output of amount:

$25 * 588,010 / 2$ , whereby 588,010 is the aggregated amount of 90AH5 of Data Table 2.

Channel	Account	Customer	Product	Quantity	Amount
90AH5	1	CN009	PROD02	75	22,050.38

The *Amount* field will then have this formula:

$\text{Quantity} * \text{MTCA2.Amount} [\text{Channel}=90\text{AH5}] / 2$

When we compute the following values for each field, we will have the following output of amount:  
 $25 * 588,010/2$ , whereby 588,010 is the aggregated amount of 90AH5 of Data Table 2.

Final Output

Channel	Account	Customer	Product	Quantity	Amount
90AH5	1	CN007	PROD01	80	23,520.40
90AH5	1	CN002	PROD02	25	7,350.13
90AH5	1	CN009	PROD02	75	22,050.38

### 1.3.13.2 Example: Calculation Scenario with Condition - Absolute

#### Input

This is the table that will be used as an input function:

CA - Absolute and Relative Table

Channel	Account	Customer	Product	Quantity	Amount
90AH3	3	CN003	PROD03	15	62.5
90AH04	1	CN001	PROD02	89	250
90AH01	2	CN004	PROD03	99	206.25
90AH04	3	CN006	PROD01	112	132.5
90AH05	3	CN010	PROD03	55	20
90AH01	1	CN001	PROD01	76	125
90AH02	2	CN002	PROD01	80	143.75
90AH02	1	CN004	PROD01	103	175
90AH05	1	CN009	PROD02	126	70.63
90AH04	1	CN001	PROD02	17	100
90AH05	1	CN002	PROD02	13	75
90AH05	1	CN007	PROD01	40	44.75
90AH02	2	CN002	PROD01	20	57.5
90AH03	3	CN005	PROD01	108	153.13
90AH05	1	CN007	PROD01	117	111.88
90AH01	2	CN004	PROD03	15	82.5
90AH03	3	CN005	PROD01	10	61.25

Channel	Account	Customer	Product	Quantity	Amount
90AH05	1	CN009	PROD02	75	28.25
90AH02	1	CN004	PROD01	13	70
90AH05	1	CN002	PROD02	94	187.5
90AH05	3	CN010	PROD03	130	50
90AH01	1	CN001	PROD01	10	50
90AH04	3	CN006	PROD01	20	53
90AH04	3	CN008	PROD02	50	36.5
90AH03	3	CN003	PROD03	85	156.25
90AH04	3	CN008	PROD02	121	91.25
90AH05	1	CN010	PROD03	130	50
90AH01	3	CN001	PROD01	10	50
90AH04	3	CN006	PROD01	20	53
90AH04	3	CN008	PROD02	50	36.5
90AH03	3	CN003	PROD03	85	156.25
90AH04	3	CN008	PROD02	121	91.25

The scenario will filter out values where the selection conditions are met for both rules.

CA - Absolute and Relative Table

Chanel	Account	Customer	Product	Quantity	Amount	Premium
90AH3	3	CN003	PROD03	15	62.5	
90AH4	1	CN001	PROD02	89	250	937.5
90AH1	2	CN004	PROD03	99	206.25	
90AH4	3	CN006	PROD01	112	132.5	1,100
90AH5	3	CN010	PROD03	55	20	
90AH1	1	CN001	PROD01	76	125	11,500
90AH2	2	CN002	PROD01	80	143.75	18,025
90AH2	1	CN004	PROD01	103	175	8,899.38
90AH5	1	CN009	PROD02	126	70.63	18,025
90AH4	1	CN001	PROD02	17	100	8,899.38
90AH5	1	CN002	PROD02	13	75	1,700
90AH5	1	CN007	PROD01	40	44.75	1,150
90AH2	2	CN002	PROD01	20	57.5	16,538.04
90AH3	3	CN005	PROD01	108	153.13	13,089.96
90AH5	1	CN007	PROD01	117	111.88	1,237.5
90AH1	2	CN004	PROD03	15	82.5	

Chanel	Account	Customer	Product	Quantity	Amount	Premium
90AH3	3	CN005	PROD01	10	61.25	2,118.75
90AH5	1	CN009	PROD02	75	28.25	612.5
90AH2	1	CN004	PROD01	13	70	17,624
90AH5	1	CN002	PROD02	94	187.5	910
90AH5	3	CN010	PROD03	130	50	
90AH1	1	CN001	PROD01	10	50	1,060
90AH4	3	CN006	PROD01	20	53	1,825
90AH4	3	CN08	PROD02	50	36.5	1,060
90AH3	3	CN003	PROD03	85	156.25	
90AH4	3	CN008	PROD02	121	91.25	13,281.25
90AH5	3	CN010	PROD03	130	50	
90AH1	1	CN001	PROD01	10	50	1,060
90AH4	3	CN006	PROD01	20	53	1,825
90AH4	3	CN008	PROD02	50	36.5	1,060
90AH3	3	CN003	PROD03	85	156.25	
90AH4	3	CN008	PROD02	121	91.25	13,281.25

Rule 1: We set the condition for *Product* as "PROD01" and use the following formula to calculate the premium: Quantity[1] \* Amount[1].

Rule 2: We set the condition for *Product* as "PROD02" and use the following formula to calculate the premium: Quantity[-1] \* Amount[-1].

### Note

[1] means that referencing the selection condition of the rule Product = PROD01, the system will calculate the value of the *Premium* of the next absolute data (any succeeding data which satisfies the selection condition).

[-1] means that referencing the selection condition of the rule Product = PROD02, the system will calculate the value of the *Premium* of the previous absolute data (any preceding data which satisfies the selection condition).

## Logic and Computation

### Example 1

- Rule 1  
Example is PROD01 at Product row 4, the next absolute data is PROD03 at Product row 5. Therefore, value of Premium row 4 = Quantity row 5 \* Amount row 5, so 55 \* 20 = 1100.
- Rule 2

Example is PROD02 at Product row 2, the previous absolute data is PROD03 at Product row 1. Therefore the value of Premium is the product of Quantity row 1 \* Amount row 1. Hence, the value of Premium at Cell H241 is 937,5.

## Example 2

- Rule 1  
Example is PROD01 at Product row 8, the next absolute data is PROD02 at Product row 9. Therefore, value of Premium for row 8 = Quantity of row 9 \* Amount of row 9, so  $126 * 70,63 = 8899,83$ .
- Rule 2  
Example is PROD02 at Product row 26, the previous absolute data is PROD03 at Product row 25. Therefore, value of Premium for Cell E265 = Quantity of row 25 \* Amount of row 25. Hence, value of Premium at Cell H844 =  $85 * 156,25 = 13.281,25$ .

Final Output

Channel	Account	Customer	Product	Quantity	Amount	Premium
90AH4	1	CN001	PROD02	89	250	937,5
90AH4	3	CN006	PROD01	112	132,5	1,100
90AH1	1	CN001	PROD01	76	125	11,500
90AH2	2	CN002	PROD01	80	143,75	18,025
90AH2	1	CN004	PROD01	103	175	8,899,38
90AH5	1	CN009	PROD02	126	70,63	18,025
90AH4	1	CN001	PROD02	17	100	8,899,38
90AH5	1	CN002	PROD02	13	75	1,700
90AH5	1	CN007	PROD01	40	44,75	1,150
90AH2	2	CN002	PROD01	20	57,5	16,538,04
90AH3	3	CN005	PROD01	108	153,13	13,089,96
90AH5	1	CN007	PROD01	117	111,88	1,237,5
90AH3	3	CN005	PROD01	10	61,25	2,118,75
90AH5	1	CN009	PROD02	75	28,25	612,5
90AH2	1	CN004	PROD01	13	70	17,625
90AH5	1	CN002	PROD02	94	187,5	910
90AH1	1	CN001	PROD01	10	50	1,060
90AH4	3	CN006	PROD01	20	53	1,825
90AH4	3	CN008	PROD02	50	36,5	1,060
90AH4	3	CN008	PROD02	121	91,25	13,281,25
90AH1	1	CN001	PROD01	10	50	1,060
90AH4	3	CN006	PROD01	20	53	1,825
90AH4	3	CN008	PROD02	50	36,5	1,060
90AH4	3	CN008	PROD02	121	91,25	13,281,25

## 1.3.14 Transfer Structure

*Transfer Structure* is a data enrichment function that can be used to transpose data according to predefined condition fields and settings. If those conditions are not met, the Transfer Structure function retains the source data. The function provides a pivot and an unpivot option.

### Key Features

#### Header

In the header, you define the principal behavior of the *Transfer Structure* function.

Transfer Structure Type:

- Transfer Structure: Transfers values to columns, sometimes also called pivoting of data. This is typically used to turn account-based data into costing-based data.
- Reverse Transfer Structure: Transfers columns to values, sometimes also called unpivoting of data. This is typically used to turn costing-based data into account-based data.

The *Retain Fields* option is relevant for the Transfer Structure type:

- All Fields: All fields are retained.
- All Fields except Action -> Source Fields: Selection condition fields are retained, but action source fields are excluded from the result.

The *Aggregate Result* option is relevant for the Transfer Structure type:

- Group Characteristics: Key figures are automatically aggregated using all input characteristics as grouping fields.
- Group Characteristics and Key Figures: Key figures are automatically aggregated using all input characteristics and key figures as grouping fields.
- No Grouping: No aggregation takes place.

#### Rules

Each transfer structure rule semantically defines an if-then statement. The if part selects which subset of the input data the rule is relevant to. The then part is an action and contains a list of fields and values that need to be assigned.

### Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

## 1.3.14.1 Example: Transfer Structure

Input

Contract	Product	Value Date	Amount	Premium
C1	P10	2016-01-01	100	50
C2	P20	2016-01-07	200	250
C3	P3	2016-01-12	400	500
C1	P10	2016-01-16	50	300
C1	P10	2016-01-01	1,000	400
C2	P20	2016-02-10	150	700

This is the input data of the function, in which it will be enriched by the *Transfer Structure* function.

In the input tab, there is a formula where the following is computed: Amount = Amount/2.5

The formula will then be executed first.

Contract	Product	Value Date	Amount	Premium
C1	P10	2016-01-01	40	50
C2	P20	2016-01-07	80	250
C3	P3	2016-01-12	160	500
C1	P10	2016-01-16	20	300
C1	P10	2016-01-01	400	400
C2	P20	2016-02-10	60	700

In the header portion, it is stated that characteristics are grouped. That is why in the input, the characteristics that are the same will be grouped. You can see that rows 1 and 5 are the same so they will be grouped. Then the key figures will be summed up.

Contract	Product	Value Date	Amount	Premium
C1	P10	2016-01-01	440	450
C2	P20	2016-01-07	80	250
C3	P3	2016-01-12	160	500
C1	P10	2016-01-16	20	300
C2	P20	2016-02-10	60	700

The aggregated characteristics with the key figures being totaled is already shown in row 1.

In our first rule, we have a selection of contract C1 and P10 (rows 1 and 4) and it has a source field in the action tab where the *Amount* field is selected.

In our second rule, we have a selection of contract C2 and P20 (rows 2 and 5) and it has a source field in the action tab where the *Amount* field is selected.

Contract	Product	Value Date	Amount	Premium
C1	P10	2016-01-16	20	300
C1	P10	2016-01-01	440	450
C2	P20	2016-01-07	80	250
C2	P20	2016-02-10	60	700

Here the rules tab will kick in and will perform the following formulas where it is mentioned that in the first rule the field *Premium 1* will now be equivalent to *Amount*.

In the second rule the field *Premium 2* will be equivalent to *Amount*.

Contract	Product	Value Date	Amount	Premium	Premium 1	Premium 2
C1	P10	2016-01-16	20	300	20	0
C1	P10	2016-01-01	440	450	440	0
C2	P20	2016-01-07	80	250	0	80
C2	P20	2016-02-10	60	700	0	0

In the header portion, it is stated that fields that will be retained will be the following: All fields except Selection & Action->Source Fields.

This means that fields within the selection and the source fields in the action will be excluded from the output (columns *Contract*, *Product* and *Amount* will be excluded in the output).

Final Output

Value Date	Premium	Premium 1	Premium 2
2016-01-16	300	20	0
2016-01-01	450	400	0
2016-01-07	250	800	80
2016-02-10	700	0	60

## 1.3.14.2 Example: Reverse Transfer Structure

Input

Product	Customer	Premium 1	Premium 2
PROD01	CUST01	10	0
PROD02	CUST02	0	20



In our rule types, it is mentioned that there would be additional fields in where we will transfer the field values from the original one to the new fields:

Product	Customer	Premium 1	Premium 2	Premium Type	Premium
PROD01	CUST01	10	0		
PROD02	CUST02	0	20		

In the first rule it is indicated that in the line which contains PROD01 and CUST01, the value of *Premium 1* would be transferred to *Premium*, while the value PREMIUM\_1 would be entered in the field *Premium Type*:

Product	Customer	Premium 1	Premium 2	Premium Type	Premium
PROD01	CUST01	10	0	PREMIUM_1	10
PROD02	CUST02	0	20		

In the first rule it is indicated that in the line which contains PROD02 and CUST02, the value of *Premium 1* would be transferred to *Premium*, while the value PREMIUM\_2 would be entered in the field *Premium Type*:

Product	Customer	Premium 1	Premium 2	Premium Type	Premium
PROD01	CUST01	10	0	PREMIUM_1	10
PROD02	CUST02	0	20	PREMIUM_2	20

In the header portion, it is stated that fields that will be retained will be the following: All fields except Selection & Action->Source Fields.

Final Output

Premium Type	Premium
PREMIUM_1	10
PREMIUM_2	20

## 1.3.15 Model BW

*Model BW* is a data model function that allows you to define and access BW InfoProviders.

### Key Features

#### Header

In the header, you define the Model BW source:

- Environment: The data model is managed in the Environment. You can use the *Editable* option to specify whether manual data input is permitted or not.
- Business Warehouse: The data model is managed externally and it is referenced within the *Model BW* function to make it available in the environment.

Furthermore, you can select *Editable* to define the data displayed for a user by the *Query* function.

## Fields

If the Model BW source is Environment, you can enter the fields of the Model BW on the *Fields* tab as a list. You can also include navigational attributes or read access and mark characteristics to be treated as key figures.

If the Model BW source is Business Warehouse, the fields of the InfoProvider are listed on the *Fields* tab. You can also exclude fields from read access.

## Parameters

If the referenced view has parameters, they are listed on this tab. For each parameter, you need to assign either a constant value or an environment parameter.

## Procedure: Using the Business Warehouse as Model BW Source

Follow the steps below to create a *Model BW* function in the environment:

1. In edit mode, choose the *Add* button.  
The *Function* window appears.
2. Fill in the required fields:
  - Function
  - Description
  - Function Type = "Model BW"
3. Fill in optional fields as necessary:
  - Event Handling
  - Logging
  - Processing Type
  - Partitioning
4. Choose *Ok* to save your changes.
5. In edit mode, fill in the following function details:
  - Model BW Source = "Business Warehouse"
  - BW InfoProvider

### **i** Note

Select the model object from the BW client, for example "ADSO".

6. On the *Fields* tab, choose *Synchronize*  
The fields of the InfoProvider are listed.
7. Activate the function.

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#) and SAP note [2921584](#).

## 1.3.16 Model Table

*Model Table* is a data model function that allows you to define and access local and remote database tables.

### Key Features

#### Header

In the header, you define the Model Table source:

- *Environment*: The data model is managed in the Environment. You can use the *Transport Data* option to decide if the data in the model table is transported together with the environment through the system landscape or not.
- *Data Dictionary*: The data model is managed externally and it is referenced within the *Model Table* function to make it available in the environment. For this, you need to enter the table name. Field information is synchronized into the environment fields.
- *HANA*: The data model is managed externally and it is referenced within the *Model Table* function to make it available in the environment. You need to enter the authoring schema and table name. Field information is synchronized into the environment fields.
- *SDA*: The data model is managed externally in a remote system and it is referenced within the *Model Table* function to make it available in the environment. You need to enter the remote source name, remote database, remote schema and remote table name. Field information is synchronized into the environment fields.

#### Fields

If the Model Table source is Environment, you can enter the fields of the model table as a list in the *Fields* tab.

If the Model Table source is Data Dictionary, SAP HANA or SDA, the fields of the table are listed on the *Fields* tab. You can also exclude fields from read access.

### Further Details

If a DDIC table that is used as the source for a model table has a client field (field with DDIC data type `CLNT`), the system selects data differently, depending on whether or not you have selected the *Exclude* option. If you do not select *Exclude*, the system filters source data, and selects only data for the current system client. If you select *Exclude*, the system selects all data from the source object, and does not filter by client.

If a model table that uses a DDIC table as a source is used as the target for a writer function, the system always populates the client field with the value of the current client, irrespective of whether or not you have selected the *Exclude* option for the client field.

### Procedure: Using the Environment as Model Table Source

Follow the steps below to create a *Model Table* function in the environment:

1. In edit mode, choose the *Add* button.  
The *Function* window appears.
2. Fill in the required fields:
  - Function
  - Description
  - Function Type = "Model Table"
3. Fill in optional fields as necessary:
  - Event Handling
  - Logging
  - Processing Type
  - Partitioning
4. Choose *Ok* and save your changes.
5. In edit mode, fill in the following function details:
  - Model Table Source = "Environment"
  - Transport Data
6. On the *Fields* tab, add the fields needed for the table:
  1. Choose *Add*.  
The *Add Fields* window appears.
  2. Select all the fields needed for the table from the list.
  3. Choose *Ok*.
  4. The fields are now available in the *Fields* tab.
7. Activate the function.
8. In edit mode on function details, choose *Maintain Data*  
The *Data Editor* appears.  
This allows you to load the data in the model table. To maintain the data follow the steps below:
  1. Choose *Edit*.
  2. Input all the data and save.

### **i** Note

The data can be maintained in two ways:

- Writing directly on the fields (Copy and Paste)
- Importing using an excel file.

## **Related Information**

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

## 1.3.17 Model RDL

*Model RDL* is a data model function that allows you to access SAP HANA-optimized Results Data types of the Financial Products subledger datamodel.

### Key Features

#### Header

Header fields are important to be filled correctly by the modeler in order to connect to an existing and activated data model. To perform this, the modeler must then define the results data area and the result type.

Example is Result Data Area: `SRINS`; Result Type: `S_ACG`

#### Fields

While on edit mode, choose the *Synchronize* button. This will then show a popup window allowing the modeler to exclude fields by setting the *Exclude* checkbox to marked (checked). All excluded fields will not be considered during processing in SAP Profitability and Performance Management.

### Simple Reading Mechanism, Selection using Fixed Value Model RDL (for example View, Join)

SAP Profitability and Performance Management reads data records from a specific Result Type through active and generated SQL or HANA Views. Automatically the system will consume all data records written in the HANA view.

In order to filter results with fixed selection, you have to create a SAP Profitability and Performance Management processing function such as the *View* or *Join* function on top of the Model RDL.

To perform this, follow below steps:

1. Create a processing function such as *View* with the Model RDL as the input function.
2. Select this *View* function and on the right side of the screen and choose *Edit*.
3. Add a selection by choosing the *Selection* button and provide a selection criteria, for example `SOURCESYSTEM = XX01`.
4. Once fully set, activate the *View* function – this function then must be used as input during processing instead of using the Model RDL function directly.

### Dynamic Reading Mechanism, Selection using Parameter or Field Model RDL (Join)

SAP Profitability and Performance Management reads data records from a specific result type through active and generated SQL or HANA Views. Automatically the system will consume all data records written in the HANA view.

In order to filter results with dynamic selection, you have to create a SAP Profitability and Performance Management *Join* function on top of the Model RDL.

### ❖ Example

`SOURCESYSTEM = I_LEGAL_PARAM` where `SOURCESYSTEM` is a field and `I_LEGAL_PARAM` is a parameter or `SOURCESYSTEM = LEGAL_FIELD` where `SOURCESYSTEM` is a field and `LEGAL_FIELD` is a field.

To perform this, follow below steps:

1. Create a *Join* function.
2. Select this join function and on the right side of the screen choose *Edit*.
3. Go to the *Rules* tab and add a rule type "FROM" with the Model RDL as its input.
4. Add a selection by going to the Complex Selection of the rule and use below format to select

```
WHERE SOURCESYSTEM = I_LEGAL_PARAM
```

5. Once fully set, activate the join function – it then must be used as the input during processing instead of using the Model RDL function directly.

## Reading while using Parallelization Model RDL (View, Join)

In order to have a more optimal parallelization setup while consuming data from RDL, you can use a SAP Profitability and Performance Management processing function such as *View* or *Join* function on top with a Package Selection entry.

To perform this, follow below steps:

1. Create a processing function such as View with the Model RDL as the input function.
2. On the left side of the screen choose *Edit*.
3. Choose *Function Attributes* button (button on the left of the *Activation* button).
4. On the *Advanced* tab under *Package Selection*, add at least one field that must be selected to avoid reading all data from the table.
5. Once fully set, activate the *View* function – it then must be used as the input during processing instead of using the Model RDL function directly.

## Writing Mechanism

SAP Profitability and Performance Management writes data records to a specific RDL Result Type using a *Writer* function with a configured *Output* function pointing to a Model RDL. This model RDL function must then be configured to point to specific Result Data Area and Result Type.


The system writes to these result types by calling AMDP classes which have active HANA procedures. There are specific result categories where AMDP classes automatically generate HANA procedures. So as a

prerequisite, in order for being able to write to the Result Type, these result types need to be assigned to one of below result categories:

HKRIC, HSRIC, HKCFR, HKEPS, HSEPS, HKACG, HSACG, HKAMS, HSAMS, HKANA, HSANA, HKRMC, HSRMC, HKVOL, HSVOL, HKMES, HSMES, HKPAT, HSPAT, HKLSE, HSLSE, HKLFP, HSLFP, HKPAP, HSPAP, HKPAI, HSPAI, HKRPE, HSRPE, HKRPS, HSRPS, HKULT, HSULT, HKVEC, HSVEC, HKCFO, HKCDA.

If you connect it to FPSL RDLs then this will be visible in the customizing activity ► [Financial Products Subledger](#) ► [Data Model](#) ► [Edit Result Data Area and Data Structures](#) ►. Choose ► [Result Data Area](#) ► [Result Types](#) ►. You can see if the Result Type then is mapped to the accepted result category (ResCat).

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#) and SAP note [2935308](#) .

## 1.3.18 Model View

[Model View](#) is a data model function that allows read access to local and remote database tables and views.

## Key Features

### Header

In the header, you define the Model View source:

- [Data Dictionary Table](#): The table is managed externally and is referenced within the [Model View](#) function to make it available in the environment. You need to enter the table name. Field information is synchronized into the environment fields.
- [Data Dictionary View](#): The view is managed externally and is referenced within the [Model View](#) function to make it available in the environment. You need to enter the view name. Field information is synchronized into the environment fields.
- [HANA Table](#): The table is managed externally and it is referenced within the [Model View](#) function to make it available in the environment. You need to enter the authoring schema and table name. Field information is synchronized into the environment fields.
- [HANA View](#): The view is managed externally and is referenced within the [Model View](#) function to make it available in the environment. You need to enter the authoring schema and view name. Field information is synchronized into the environment fields.
- [SDA](#): The table or view is managed externally in a remote system and is referenced within the [Model View](#) function to make it available in the environment. You need to enter the remote source name, remote database, remote schema and remote table name. Field information is synchronized into the environment fields.

- **CDS View:** The view is defined for existing database tables and any other views or CDS views in the ABAP Dictionary using the statement `DEFINE VIEW` in the CDS DDL in ABAP Core Data Services (CDS). This is done in the CDS source code of a CDS data definition in the ABAP Development Tools.

## Fields

The fields of the table or view are listed on the *Fields* tab. You can also exclude fields from read access.

## Parameters

If the referenced table or view has parameters, they are listed on this tab. For each parameter, you need to assign either a constant value or an environment parameter.

## Further Details

If a DDIC table overview that is used as source for a model view has a client field (field with DDIC data type `CLNT`), the system selects data differently, depending on whether or not you have selected the *Exclude* option. If you do not select the *Exclude* option, the system filters source data, and selects only data for the current system client. If you select the *Exclude* option, the system selects all data from the source object, and does not filter by client.

## Procedure: Using the Data Dictionary Table as Model View Source

Follow the steps below to create a *Model View* function in the environment:

1. In edit mode, choose the *Add* button.  
The *Function* window will appear.
2. Fill in the required fields:
  - Function
  - Description
  - Function Type = "Model View"
3. Fill in any optional fields as necessary:
  - Event Handling
  - Logging
  - Processing Type
  - Partitioning
4. Choose *Ok* to save your changes.
5. In edit mode, fill in the following function details:
  - Model View Source = "Business Warehouse"
  - Table Name
6. On the *Fields* tab, choose *Synchronize*.  
The fields of the Data Dictionary (DDIC) table are listed.
7. Activate the function.



## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.19 Query

*Query* is a reporting function that allows the output and input of data.

## Key Features

### Header

In the header, you define the query source:

- *Environment*: The query is managed in the Environment. You can use the input function to define which function data the system accesses. If the source is an editable model BW, you can use the *Editable* option to specify whether manual data input is permitted. Optionally, a key date can be specified, which is then propagated to InfoObjects with time-dependent master data. Technically, this uses NetWeaver and BEx Query technology.
- *Business Warehouse*: The query is managed externally and is referenced within the Query function to make it available in the environment. You need to enter the external query name.
- *Analysis for Office*: The Analysis for Office work is managed externally and is referenced within the Query function to make it available in the environment. You need to enter the workbook name.
- *Environment CDS*: The query is managed in the Environment. You can use the input function to define which function data the system accesses. If the source model is editable, you can use the *Editable* option to specify whether manual data input is permitted.

### Filter

In the *Filter* tab, you define general fixed and default values:

- Fixed Values: These values cannot be changed by the user when the report is run.
- Default Values: The report starts with these values, but the user can change them when the report is run.

The detailed general, key figure and hierarchy settings for each field are explained below in the "Sheet Definition" key feature.

### Sheet Definition

If the query source is Environment, the following additional options are available:

Fields can be arranged in rows, columns and as free fields (which are available for users but are not part of the report by default).

In addition to the mandatory key figure structure, a further structure can be defined which allows you to arrange key figures and characteristics in a hierarchical structure.

For each field, the following general settings are available:

- Description: Allows you to modify the description of the field in the report.
- Selection: Allows you to define a selection for the field.
- Formula: Allows you to define a formula for the field.
- Access Type for Result Values:
  - Posted Values
  - Characteristic Relationships
  - Master Data
- Show Result Rows:
  - Always
  - Never
  - Only if more than one child

For each key figure, the following settings are available:

- Editable: Yes or no
- Aggregation Behavior:
  - Default: The aggregation behavior is taken from the environment field definition.
  - Maximum: In case of aggregation, the key figure maximum value is displayed.
  - Minimum: In case of aggregation, the key figure minimum value is displayed.
  - Summation: In case of aggregation, the key figure is summed for display.

For each characteristic, the following hierarchy settings are available:

- Hierarchy Name: If the field has active hierarchies, one can be chosen as a default for the report. The user can choose another hierarchy during runtime.
- Hierarchy Date: If the hierarchies are time-dependent, a constant or parameter has to be specified here.
- Hierarchy Version: If the hierarchies are versioned, a constant or parameter has to be specified here.

## Procedure

Note that unlike the *Show* function available during modeling, the *Query* function does not provide any technical fields from the model, like message text, function ID, rule ID, or formula, because this is not relevant for the user later on during execution of processes and reports. If you want to display query data separately by process instance, you need to do the following:

1. Modeling
  1. Define an InfoObject field in BW with the properties *Master Data* and *Authorization-Relevant* switched on, and then register this field in the Environment.
  2. Define a parameter in the Environment referencing the same InfoObject (for example, `I_VERSION`).
  3. Register the parameter at calculation unit level so that it is accessible for the process.
  4. Assign the `VERSION` field to the value of `I_VERSION` using a formula in a function so that the value of the parameter is persisted with the data.
  5. In the Query functions, define a selection for the version field using a variable that represents "Single Value".
2. Process Management
  1. When a process instance is created, the parameter needs to be set to a value (for example, `I_VERSION = V001`).

2. Deploy the process instance so that execution users can use it.

When execution users run the Query functions, they need to enter a value for the `VERSION` field and the system checks whether they are allowed to see the data.

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.19.1 Example: Editable Query Using BW ADSO as Source

#### Input Data

Customer	Product	Country	Amount	Quantity
C001	P1	US	500.00	50
C001	P1	DE	1,000.00	100
C001	P2	US	500.00	200
C001	P2	DE	1,500.00	300
C002	P3	US	2,000.00	200
C002	P2	DE	1,500.00	250
C002	P1	US	1,000.00	150
C002	P2	DE	1,500.00	150
C003	P3	US	2,000.00	200
C003	P1	US	2,500.00	100
C003	P2	US	1,000.00	150
C003	P3	DE	1,500.00	200
C004	P2	DE	1,000.00	250
C004	P1	DE	2,000.00	150
C004	P3	US	2,500.00	50
C004	P2	DE	1,500.00	150

In the *Query* function, *Country*, *Product* and *Customer* are defined as rows respectively and *Amount* is editable and defined as a column.

Given the input above, the function will arrange the table by country, then by product and then by customer.

After being arranged by country, the table from the *Input Data* section looks as follows:

Country	Customer	Product	Amount
DE	C001	P1	1,000.00
DE	C004	P1	2,000.00
DE	C001	P2	1,500.00
DE	C002	P2	1,500.00
DE	C002	P2	1,500.00
DE	C004	P2	1,000.00
DE	C004	P2	1,500.00
DE	C003	P3	1,500.00
US	C001	P1	500.00
US	C002	P1	1,000.00
US	C003	P1	2,500.00
US	C001	P2	500.00
US	C003	P2	1,000.00
US	C002	P3	2,000.00
US	C003	P3	2,000.00
US	C004	P3	2,500.00

Afterwards, the table is arranged by product and then by customer:

Country	Product	Customer	Amount
DE	P1	C001	1,000.00
DE	P1	C004	2,000.00
DE	P2	C001	1,500.00
DE	P2	C002	1,500.00
DE	P2	C002	1,500.00
DE	P2	C004	1,000.00
DE	P2	C004	1,500.00
DE	P3	C003	1,500.00
US	P1	C001	500.00
US	P1	C002	1,000.00
US	P1	C003	2,500.00
US	P2	C001	500.00
US	P2	C003	1,000.00
US	P3	C002	2,000.00

Country	Product	Customer	Amount
US	P3	C003	2,000.00
US	P3	C004	2,500.00

## Result

The data will appear on the *Analyze* screen (data grid) as below. The amounts are aggregated based on the characteristics fields.

Country	Product	Customer	Amount
DE	P1	C001	1,000.00
		C004	2,000.00
	P2	C001	1,500.00
		C002	3,000.00
		C004	2,500.00
	P3	C003	1,500.00
US	P1	C001	500.00
		C002	1,000.00
		C003	2,500.00
	P2	C001	500.00
		C003	1,000.00
	P3	C002	2,000.00
		C003	2,000.00
		C004	2,500.00

## Editing the Values from the Query

When the query is editable, you can change the key figure value directly through data grid view. The query will only be available for editing and when there are no fields in the *Free* section. This is the case when all hidden characteristic fields are shown.

To edit the data, follow below steps:

1. Choose *Analyze* on the upper right corner of the screen.
2. In edit mode, select the key figure you need to change. (e.g. "Amount").
3. Choose *Save*.

### i Note

You can also add a new entry. Choose *New Line* and add the values or information accordingly.

## 1.3.20 Writer

*Writer* is a processing function that allows you to write in model tables, model BWs and model RDL components. Therefore, the *Writer* function covers all the technical complexity of the different access modes.

### Key Features

#### Header

You first need to define the output function. For this you can use all the model tables, model BWs and model RDL components of the environment.

The function automatically detects the type of output function and offers dependent choices.

BW Write Type:

- Planning: Data is written using the Planning Engine and users who are editing data can continue to work.
- Loading: Data is written using the SAP S/4HANA-based data transfer process, and users who are editing data cannot continue to work during this time.

Writer Type:

- Insert: Data is inserted in addition to existing data.
- Modify: Data with the same characteristic values is overwritten and new data is inserted.
- Delete and Insert: Existing data is deleted first, and new data is then inserted.

#### Output

On this tab, you can apply any required mapping to output fields, including selections, formulas, grouping and sorting.

If the output function type is "Model BW", the selections are also applied to the deletion of data.

### Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

## 1.3.20.1 Example: Write to Model Table

### Input Data

Input Table

Customer	Product	Quantity	Amount
CN001	PROD01	20	20.00
CN002	PROD01	40	23.00
CN003	PROD03	30	25.00
CN001	PROD02	33	40.00
CN002	PROD02	25	30.00
CN004	PROD03	30	33.00
CN004	PROD01	25	28.00

Output Table

Customer	Product	Quantity	Amount
----------	---------	----------	--------

A writer simply writes the result of the input function, which can be a data source, enrichment or process function. You can prepare this input on the writer function's *Input* tab and *Output* tab before the system writes it to the internal or external table through data source functions such as *Model Table*, *Model RDL* or *Model BW*. In this example scenario, both the input and the output is an Environment Model Table.

### Result

You can check the result by choosing *Maintain Data* in the *Model Table* function.

Customer	Product	Quantity	Amount
CN001	PROD01	20	20.00
CN002	PROD01	40	23.00
CN003	PROD03	30	25.00
CN001	PROD02	33	40.00
CN002	PROD02	25	30.00
CN004	PROD03	30	33.00
CN004	PROD01	25	28.00

## 1.3.21 View

*View* is a data access function that can be used to select data and provide projections and aggregations on top of it. This view on the data can then be used for consumption in other functions like allocation. In addition, a view has several options for fine tuning data consumption. For example, it can use a table sample or fraction of the input data to select a specific time version of data in history tables or it can only provide input data if a run parameter precondition is met (for example, if `MY_READ_PARAM_FLAG = "X"`). A view can also run iterations of input function calls, including early exit checks.

### Key Features

#### Header

In the header, you define the principal behavior of the view.

View Type:

- **Implicit Fields:** The view adopts all the fields from the input. Only the fields explicitly named in the output are used only if you have defined an aggregation on the *Output* tab using field grouping .
- **Explicit Fields:** The view adopts only the fields explicitly named on the *Output* tab; all other fields are excluded.

#### Output

You can enter additional field details (such as select conditions, formulas, group aggregations and sort orders) on the *Output* tab. This is optional for the view type "Implicit Fields", but mandatory for the type "Explicit Fields".

#### Advanced

The following options are available on the *Advanced* tab:

- **Top:** You can enter a constant number or parameter in this field to restrict the data reading to a given absolute number of records. For example, `Top = 100` reads only the first 100 records of the input.
- **Run Parameter Precondition:** If you enter a parameter condition here, the view only provides an output if this precondition is met. Otherwise, the output is 0 records.
- **Default Type:** If you choose "Default output, if input empty", the view populates and displays results based on the assigned field value in the Output tab. This gives a modeler the option of producing one record table with self-assigned values.
- **Iteration Type:**
  - None: No Iteration
  - For Loop: The Input function is called multiple times in a loop using the Low and the High fields as boundaries.
- **Iteration Parameter:** In the *Iteration Parameter* field, you need to enter a parameter that contains the current loop number and thus makes it available for the input function as well.
- **Early Exit Check:** You can register an early exit check from the environment checks. This is applied to the view result and if the check is successful, the iteration is exited early.
- **Iteration Result:**
  - All Iterations: The result of all iterative calls of the view input is collected and provided as output.



- Last Iteration: Only the output of the last iterative call is provided.

## Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see [Concepts for Key Users \[page 58\]](#).

### 1.3.21.1 Example: Aggregation

Input: VI - Order Table 1

Customer	Product	Quantity	Amount
CN001	PROD01	40	23
CN001	PROD02	20	20
CN002	PROD01	30	33
CN002	PROD02	30	25
CN003	PROD03	33	40
CN004	PROD03	25	28
CN004	PROD01	25	30

1. Group customers that have the same characteristics (from the field *Group*).

Customer	Product	Quantity	Amount
CN001	PROD01	20	20
	PROD02	33	40
CN002	PROD01	40	23
	PROD02	25	30
CN003	PROD03	30	25
CN004	PROD03	30	33
	PROD01	25	28

2. Add up the *Amount* per grouped *Customer* (from the formula field `SUM (AMOUNT)` ).

Customer	Product	Quantity	Amount
CN001	PROD01	20	60 = 20 + 40
	PROD02	33	
CN002	PROD01	40	53 = 23 + 30

Customer	Product	Quantity	Amount
	PROD02	25	
CN003	PROD03	30	25
CN004	PROD03	30	61 = 33 + 28
	PROD01	25	

3. Count how many products there are in each grouped *Customer* (from the formula `COUNT (PRODUCT)` ).

Customer	Product	Quantity	Amount
CN001	2 Products (PROD01 & PROD02)	20 33	60
CN002	2 Products (PROD01 & PROD02)	40 25	53
CN003	1 Product	30	25
CN004	2 Products (PROD03 & PROD01)	30 25	61

4. Add up the *Quantity* per grouped *Customer* (from the formula `SUM (QUANTITY)` ).

Customer	Product	Quantity	Amount
CN001	2	53 = 20 + 33	60
CN002	2	65 = 25 + 40	53
CN003	1	30	25
CN004	2	55 = 30 + 25	61

5. Result: Aggregate View

Customer	Product	Quantity	Amount
CN001	2	53	60
CN002	2	65	53
CN003	1	30	25
CN004	2	55	61

## 1.3.21.2 Example: Loop

Input: VI - Result Table

Input Number	Divisible by 3
1	NO

When a view is set to loop, it calls the input function and runs it multiple times.

In a normal modeling scenario, the iteration is handled by calling a chain of functions ending with a writer. A simple scenario has been created to highlight the looping mechanism provided by the [View](#) function.

Prerequisite: The model table has one input record.

Initial Input Table

Input Number	Divisible
1	NO
2	NO
3	YES

1. The model table acts as a repository for both input (join) and output (writer), which will then be used in the loop.

Assuming that there are already three records in the model table:

2. The Join acts as the process that must be run on the records before the result is written back to the model table. In this scenario, the join has been set up to perform the following three steps:
  1. Step 1: All records from the model table are collected and the maximum record is marked. Marking is done by performing `Iteration Counter = MAX (Input Number) OVER ()`, giving the result on the left.

Input Number	Divisible by 3	Iteration Counter
3	YES	3

2. Step 2: Select only the record that has the same iteration counter and input number. This can be configured by using complex selections tab: `WHERE Iteration Counter = Input Number`. In this case, the result of step 2 is on the left.

Input Number	Divisible by 3
4	NO

3. Step 3:
  1. `ZQA_INPNO` is incremented by 1, the formula is `Input Number = Input Number + 1`, so `Input Number = 3+1`.
  2. The record should be assessed if divisible by 3. The formula for this is as follows:

```
DIVISIBLE = CASE WHEN MOD ((Input Number +1),3) = 0 THEN "YES"
ELSE "NO"
END
```

SO

```
DIVISIBLE = CASE WHEN MOD ((3+1),3) = 0 THEN "YES"  
ELSE "NO"  
END
```

In this case, the condition statement provides a "NO" since  $4 \text{ MOD } 3 = 1$ .

3. The iteration counter must also be set to "Not Used" to prepare the writing to the model table. The result of step 3 is on the left.

Input Number	Divisible by 3
1	NO
2	NO
3	YES
4	NO

3. The final result from step 2 is inserted in the model table using a writer. If the model table is checked, it must have a result like the one on the left after the iteration.

If the process needs to be run multiple times, it can be done via view.

4. A view with active looping functions must be used to call the writer multiple times. The main steps 1 to 3 are executed multiple times based on what has been defined on the *Advance* tab of the view.

### i Note

If the iteration is set to loop from 1 to 50, the writer is called 51 times; once from the input tab of the view and 50 more times from the loop set (1-50). Do not assess the result using the view result, instead use the model table where you can see all the records that have been written since the last iteration.

## 1.3.22 Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems with the ability to automatically learn and improve from experience without being explicitly programmed.

The *Machine Learning* function allows you to get knowledge and insights and learn from your own data within SAP Profitability and Performance Management. By completing a few simple configuration steps, you can train and deploy best-in-class predictive models that currently deal with classification, clustering, regression and time-series forecasting scenarios. You can leverage these functionalities powered by SAP automated predictive technology to integrate trustworthy predictions into your business process models built in or upon SAP Profitability and Performance Management, so that you augment their business intelligence capabilities by learning from data.

## Key Features

### Header

All header options such as [Result Handling](#), [Include Original Input Data](#), [Suppress Initial Results](#) and [Result Model Table](#) are called “Functional Building Blocks” and are not specific to Machine Learning. For more information about these options, see [Header \[page 63\]](#).

### Rules

Machine Learning offers the following rule types that deal with time-series forecast, clustering, regression and classification:

#### Forecast Rule Type

The Machine Learning function provides the rule type [Forecast](#) to train and use time-series predictive models based on input data. It runs several models (for example, linear regression, or exponential smoothing) on historical data to determine the best model trained from the input dataset. It also predicts future values with this model for a specific measure. The forecasted values can be used later in other functions. The prediction is only applied to the specific selected measure. However, input data is replaced in the output if you choose to run a forecast over a field that contains historical data.

The following rule line input fields are available for configuration:

- [Date Field](#): Specifies the date field for the time series
- [End Date](#): Specifies the end date as a constant or parameter of the historical data
- [Signal Field](#): Specifies the field from which the values are used for the forecast
- [Excluded Fields](#): List of fields that are not relevant for the forecast, or those fields whose impact should be excluded from the forecast
- [Forecast Period](#): Period for which the forecast is executed
- [Forecast Unit](#): Period unit for the number of periods (for example, year, quarter, or month)
- [Positive Forecast](#): Defines whether only positive forecasted values are generated
- [Segmented By](#): Defines which fields are segmented based on the input dataset in case of multiple independent forecasts

The following rule output fields are available for configuration:

- [Forecast Field](#): The field in which the system writes the forecast result from the signal field
- [Model](#): An optional field containing the trained or used model ID that is automatically assigned by the forecast algorithm. It is a characteristic field type and we recommend you use a length of 30 characters.

#### Clustering Rule Type

The [Machine Learning](#) function provides rule type [Clustering](#) to train a clustering model based on input data. The goal of a clustering model is to find underlying structures in the input data, for example segmenting the input data into multiple clusters, where each cluster contains data points that are similar to each other with respect to observed features.

The clustering function runs a k-means algorithm on the input data to determine a partition into clusters. This function then calculates which cluster each data point belongs to. You can choose explicitly which fields (features) of the input data are to be considered by the algorithm. The model that is calculated is saved and stored under a unique model ID for each segment.

The following rule input fields are available for configuration:

- *Minimal Number of Cluster*: Specifies the minimal number of clusters
- *Maximal Number of Cluster*: Specifies the maximum number of clusters
- *Clustering Fields*: Specifies the fields (features) according to which the input data is to be clustered
- *Segmented By*: Specifies the fields according to which the whole input data is to be segmented. An independent clustering model is trained for each segment. This means, an independent segmentation into clusters is found. If this list is empty, then the whole input dataset is considered as one segment by default.

The model starts with a cluster number specified by *Minimal Number of Clusters*. It then finds the optimal number of clusters for each input data segment by iterating over different numbers of clusters within the range given by *Minimal Number of Clusters* and *Maximal Number of Clusters* while optimizing certain distance measures within each segment. If the user leaves these two input fields empty, the model will start with default number of 10 clusters.

The following rule output fields are available for configuration:

- *Cluster ID*: The field returns the cluster to which an input data point is assigned to by a model. It is an integer by default.
- *Model*: The field returns the unique ID of the trained model for each segment that is being used for predicting. It is a characteristic field type and we recommend you use a length of 30 characters.

### Regression Rule Type

The Machine Learning function provides rule type *Regression* to train and use a regression model for prediction. The idea of a regression model is to define the relationship between input data and target field using training data and the specific functional form learned depends on the choice of model.

The goal of a regression model is to learn to predict an output based on an input set of features. Taking advantage of gradient boosting technique used by SAP HANA automated predictive library (APL), a regression predicts a target field based on influenced fields. Gradient boosting is a machine learning algorithm to find the shortcomings in the previous predictions and combines base learners by sequentially minimizing the difference between the actual and predicted values. It mainly deals with large volumes of data to make a prediction with high prediction power.

The following rule input fields are available for configuration:

- *Target Field*: Specifies the field that stores target values of the prediction
- *Influence Fields*: Specifies the model input fields (features) that are used to find the assumed relationship to the target field
- *Segmented By*: Specifies the fields according to which the whole input data is to be segmented. An independent regression model is trained for each segment. This means, an independent segmentation into regression is found. If this list is empty, then the whole input dataset is considered as one segment by default.
- *Order by Fields*: Specifies the fields according to that the segmented datasets are to be sorted.

The following rule output fields are available for configuration:

- *Predicted Value*: Specifies a field that stores the predicted values. It must be a key figure of numeric type.
- *Model*: The field returns the unique ID of the trained model for each segment that is being used for predicting. It is a characteristic field type and we recommend you use a length of 30 characters.

### Classification Rule Type

The *Machine Learning* function provides a rule type *Classification* to train and use a binary or multinomial classification model for prediction. The idea of a classification model is to interpret a relationship between a set

of descriptive attributes (features) and a nominal target attribute with two or more than two classes using training data. The specific functional form learned depends on the choice of model.

The goal of a trained classification model is to predict a nominal output based on new input of descriptive attributes. The classification takes advantage of gradient boosting technique empowered by SAP HANA automated predictive library (APL). Gradient boosting is a machine learning algorithm to find the shortcomings in the previous predictions. It combines base learners by sequentially minimizing difference between the actual and predicted values. It mainly deals with large volumes of data to make a prediction with high prediction power.

The following rule input fields are available for configuration:

- **Classification Type:** Offers three options – binary, multinomial and autonomous. Default is autonomous. When it is chosen, the algorithm will automatically inspect the classes of the target field in the input data and choose the right model for training.
- **Target Field:** Specifies the field that stores the target attribute of the prediction. The target attribute should be integer-valued.
- **Influence Field:** Specifies the model input fields (features) that are used to find the assumed relationship to the target field.
- **Segmented By:** Specifies the fields according to which the whole input data is to be segmented. An independent regression model is trained for each segment, for example an independent segmentation into regression is found. If this list is empty, then the whole input dataset is considered as one segment by default.
- **Order by Fields:** Specifies the fields according to which the segmented datasets are to be sorted.

The following rule output fields are available for configuration:

- **Predicted Value:** Specifies a field that stores the predicted values
- **Model:** The field returns the unique ID of the trained model for each segment that is being used for predicting. It is a characteristic field type and we recommend you use a length of 30 characters.

## 1.3.22.1 Example: Forecast Rule Type

### Input Data

Posting Date	Account	Area	Cost Center	Amount	End Date	Period
2018-01-31	BEN	ADM	HR	638677	2018-12-31	12
2018-02-28	BEN	ADM	HR	585426	2018-12-31	12
2018-03-31	BEN	ADM	HR	514609	2018-12-31	12
2018-04-30	BEN	ADM	HR	934270	2018-12-31	12
2018-05-31	BEN	ADM	HR	894057	2018-12-31	12
2018-06-30	BEN	ADM	HR	449385	2018-12-31	12

Posting Date	Account	Area	Cost Center	Amount	End Date	Period
2018-07-31	BEN	ADM	HR	835734	2018-12-31	12
2018-08-31	BEN	ADM	HR	798317	2018-12-31	12
2018-09-30	BEN	ADM	HR	840057	2018-12-31	12
2018-10-31	BEN	ADM	HR	771350	2018-12-31	12
2018-11-30	BEN	ADM	HR	556637	2018-12-31	12
2018-12-31	BEN	ADM	HR	813597	2018-12-31	12
2018-01-31	COM	SAL	MED	610763	2018-12-31	12
2018-02-28	COM	SAL	MED	558743	2018-12-31	12
2018-03-31	COM	SAL	MED	627313	2018-12-31	12
2018-04-30	COM	SAL	MED	614808	2018-12-31	12
2018-05-31	COM	SAL	MED	675383	2018-12-31	12
2018-06-30	COM	SAL	MED	552520	2018-12-31	12
2018-07-31	COM	SAL	MED	581324	2018-12-31	12
2018-08-31	COM	SAL	MED	560999	2018-12-31	12
2018-09-30	COM	SAL	MED	626623	2018-12-31	12
2018-10-31	COM	SAL	MED	559184	2018-12-31	12
2018-11-30	COM	SAL	MED	584711	2018-12-31	12
2018-12-31	COM	SAL	MED	595020	2018-12-31	12

The forecast period is set to 12. This is why there are 12 months of forecasted amounts for future dates per account.

The function calculates the values in the *Forecast Amount* column based on the algorithm of the Automated Predictive Learning applied.

From these values, the function generates forecasted amounts for the next forecasted periods.

## Result

Posting Date	Account	Amount	Forecast Amount	Forecast ID
2018-01-31	BEN	638 677.00	638 677.00	Y800QS5GRPAML0100001
2018-02-28	BEN	585426.00	585 426.00	Y800QS5GRPAML0100001
2018-03-31	BEN	514 609.00	514 609.00	Y800QS5GRPAML0100001
2018-04-30	BEN	934 270.00	934 270.00	Y800QS5GRPAML0100001



Posting Date	Account	Amount	Forecast Amount	Forecast ID
2018-05-31	BEN	894 057.00	894 057.00	Y800QS5GRPAML010 0001
2018-06-30	BEN	449 385.00	449 385.00	Y800QS5GRPAML010 0001
2018-07-31	BEN	835 734.00	835 734.00	Y800QS5GRPAML010 0001
2018-08-31	BEN	798 317.00	798 317.00	Y800QS5GRPAML010 0001
2018-09-30	BEN	840 057.00	840 057.00	Y800QS5GRPAML010 0001
2018-10-31	BEN	771 350.00	771 350.00	Y800QS5GRPAML010 0001
2018-11-30	BEN	556 637.00	556 637.00	Y800QS5GRPAML010 0001
2018-12-31	BEN	813 597.00	813 597.00	Y800QS5GRPAML010 0001
2019-01-31	BEN	0.00	844 209.58	Y800QS5GRPAML010 0001
2019-02-28	BEN	0.00	734 238.07	Y800QS5GRPAML010 0001
2019-03-31	BEN	0.00	663 676.74	Y800QS5GRPAML010 0001
2019-04-30	BEN	0.00	789 127.24	Y800QS5GRPAML010 0001
2019-05-31	BEN	0.00	685 265.97	Y800QS5GRPAML010 0001
2019-06-30	BEN	0.00	618 404.41	Y800QS5GRPAML010 0001
2019-07-31	BEN	0.00	734 044.91	Y800QS5GRPAML010 0001
2019-08-31	BEN	0.00	636 293.88	Y800QS5GRPAML010 0001
2019-09-30	BEN	0.00	573 132.08	Y800QS5GRPAML010 0001
2019-10-31	BEN	0.00	678 962.57	Y800QS5GRPAML010 0001
2019-11-30	BEN	0.00	587 321.78	Y800QS5GRPAML010 0001
2019-12-31	BEN	0.00	527 859.75	Y800QS5GRPAML010 0001
2018-01-31	COM	610 763.00	610 763.00	Y800QS5GRPAML010 0002

Posting Date	Account	Amount	Forecast Amount	Forecast ID
2018-02-28	COM	558 743.00	558 743.00	Y800QS5GRPAML010 0002
2018-03-31	COM	627 313.00	627 313.00	Y800QS5GRPAML010 0002
2018-04-30	COM	614 808.00	614 808.00	Y800QS5GRPAML010 0002
2018-05-31	COM	675 383.00	675 383.00	Y800QS5GRPAML010 0002
2018-06-30	COM	552 520.00	552 520.00	Y800QS5GRPAML010 0002
2018-07-31	COM	581 324.00	581 324.00	Y800QS5GRPAML010 0002
2018-08-31	COM	560 999.00	560 999.00	Y800QS5GRPAML010 0002
2018-09-30	COM	626 623.00	626 623.00	Y800QS5GRPAML010 0002
2018-10-31	COM	559 184.00	559 184.00	Y800QS5GRPAML010 0002
2018-11-30	COM	584 711.00	584 711.00	Y800QS5GRPAML010 0002
2018-12-31	COM	595 020.00	595 020.00	Y800QS5GRPAML010 0002
2019-01-31	COM	0.00	572 812.99	Y800QS5GRPAML010 0002
2019-02-28	COM	0.00	570 716.78	Y800QS5GRPAML010 0002
2019-03-31	COM	0.00	571 710.49	Y800QS5GRPAML010 0002
2019-04-30	COM	0.00	557 375.26	Y800QS5GRPAML010 0002
2019-05-31	COM	0.00	555 196.11	Y800QS5GRPAML010 0002
2019-06-30	COM	0.00	556 020.56	Y800QS5GRPAML010 0002
2019-07-31	COM	0.00	541 937.52	Y800QS5GRPAML010 0002
2019-08-31	COM	0.00	539 675.44	Y800QS5GRPAML010 0002
2019-09-30	COM	0.00	540 330.64	Y800QS5GRPAML010 0002
2019-10-31	COM	0.00	526 499.78	Y800QS5GRPAML010 0002

Posting Date	Account	Amount	Forecast Amount	Forecast ID
2019-11-30	COM	0.00	524 154.76	Y800QS5GRPAML010 0002
2019-12-31	COM	0.00	524 640.70	Y800QS5GRPAML010 0002

Machine Learning also shows the input data as designed for the accounts "BEN" and "COM". It forecasted an upward trend for account "BEN" and a positive trend for account "COM". Since the forecast period was specified as 12 months, Machine Learning projected the amount for the next 12 months.

## Related Information

[SAP HANA Automated Predictive Library Reference Guide. Version 1902 or later.](#)

## 1.4 How-to Guide

Get an easy start in financial and business modeling by following this How-to Guide.

The following How-to Guide aims to help support business users in their first steps in configuring their own financial models.

1. Modeling and execution of a simple financial and business model
  1. Administration
    1. Check the default settings  
Make sure that default settings have already been defined for Schema, Path, and so on.
    2. Create a team  
Create a team and assign users who will be allowed to later execute the processes and run the simulations.
  2. Modeling
    1. Create an environment  
Set up a non-private environment using the default settings.
    2. Create an information model  
Define fields with master data and hierarchies as well as Model BW functions, which will contain the data during execution.
    3. Create input queries on top of the information model  
Define input-ready queries, which allow data to be entered during execution in a secure way.
    4. Create a calculation model  
Define and connect the Join, Derivation, Calculation and Allocation functions, which define the logic of the calculation model.
    5. Create report queries on top of the calculation model  
Define read-only queries to visualize and review the results.
    6. Define production and simulation process templates with activity templates  
Define the orchestration of the manual and calculation activities.

### 3. Execution

1. Deploy production and what-if simulation processes  
Use the prepared templates to deploy a production and simulation process and assign the prepared team.
2. Execute production process activities  
Run through the production process activities.
3. Assemble a report, including what-if simulation  
To make the what-if simulation process even more interactive, assemble a report from the simulation process.
4. Execute the what-if simulation report  
Launch the what-if simulation report, modify data and run the simulation.

## Related Information

For more information about financial and business modeling entities, see [Financial and Business Modeling Entities \[page 60\]](#).



For more information about common aspects of SAP Profitability and Performance Management functions, see [Modeling Environment \[page 24\]](#).

# Important Disclaimers and Legal Information

## Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

## Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.

The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

## Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

## Gender-Related Language

We try not to use gender-specific word forms and formulations. As appropriate for context and readability, SAP may use masculine word forms to refer to all genders.

## Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

© 2020 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.