USER GUIDE | PUBLIC

Document Version: 1.0 – 2019-09-24

SAP Profitability and Performance Management

THE BEST RUN SAP

Content

1 SAP Profitability and Performance Management

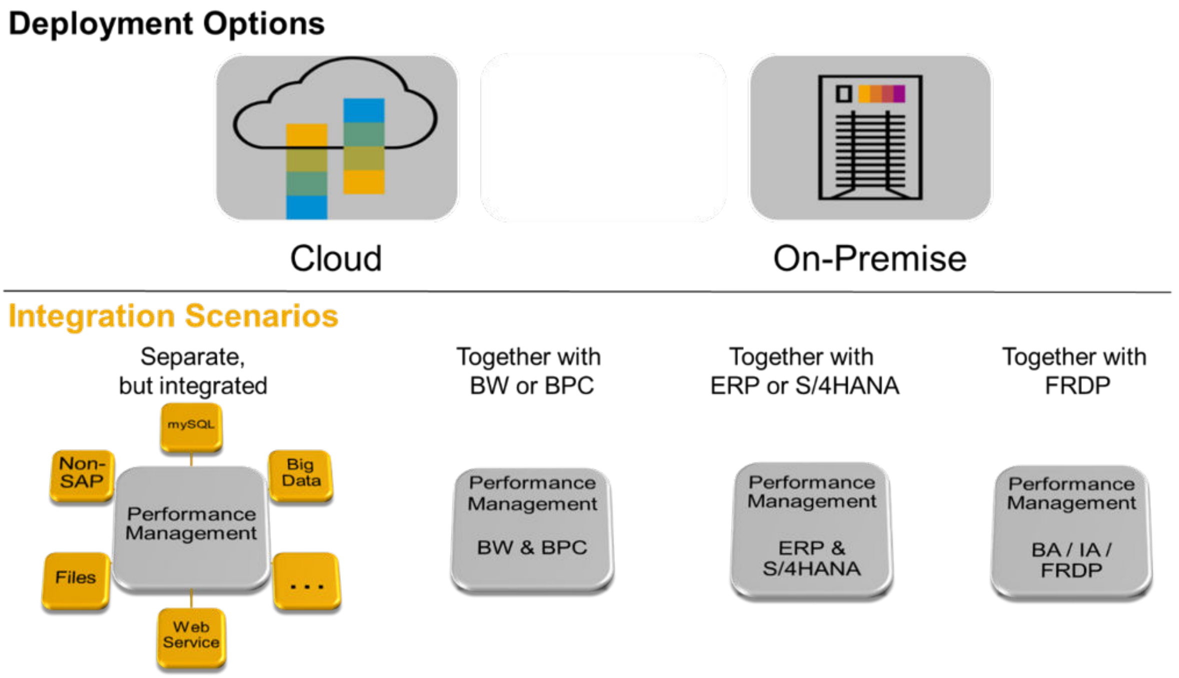
With SAP Profitability and Performance Management, SAP provides a new generation of integrated performance management applications that do not require their own data model but can use and reuse existing data and information models from other SAP and non-SAP applications in the cloud or on-premise.

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 and Integration Scenarios



SAP Profitability and Performance Management can be used on a separate instance but integrated with other SAP and non-SAP components. We recommend that you implement SAP Profitability and Performance Management as closely as possible to the relevant 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. In this case, other typical deployment scenarios are on the same instance as BW and BPC, ERP, or the finance and risk data platform (FRDP).

Integration

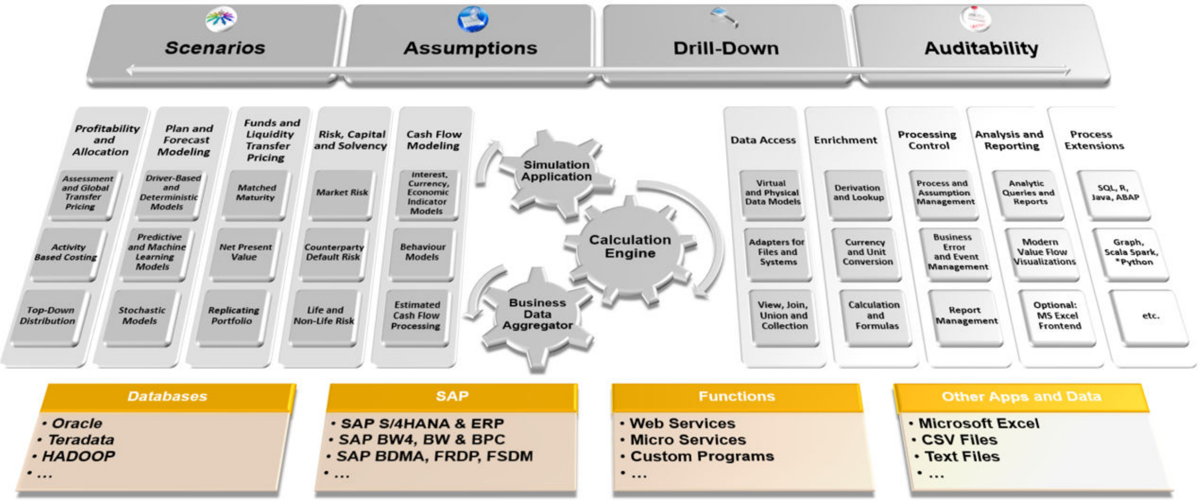
The business data aggregation capabilities of SAP Profitability and Performance Management enable the integration of operational systems and data warehouses at high speed with little or no data replication.

SAP Profitability and Performance Management uses the official application interfaces from the SAP or non-SAP application for data read access, for example CDS views from SAP S/4HANA, open ODS views from BW or calculation views from the FRDP – 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 Business Warehouse, SAP HANA-based PAK functions to BPC and AMDP interfaces to RDL. 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:

1. Profitability and cost management
   Profitability and cost management process for actuals and planning data including what-if simulation and reporting.
2. Agile plan and forecast modeling
   Process that uses actuals as the foundation for the application of strategic and operational drivers in various deterministic, predictive, stochastic, and deep learning models to determine forecast and planning results.
3. IT cost management
   IT cost management process for actuals and planning data using standardized technology business management activity-based costing rules.
4. Global transfer pricing
   Global transfer pricing and recharges for actuals and planning data using allocations, markup, and tax calculation rules.
5. Allocation simulation
   Assessment and distribution of actuals for GL-level data using iterative allocation rules including traceability of original cost center and element.
6. Carbon footprint management
   Calculates the usage and efficiency of carbon (CO2) along a process with activities including what-if simulation and reporting.
7. Funds and liquidity transfer pricing
   Net interest income calculation for actuals and planning data using funds and liquidity transfer pricing methods for different types of instruments on single contract, portfolio, and GL level.
8. Risk, capital, and solvency management
   Minimum and required capital calculations based on selected risk and solvency rules.
9. Service industries airline profitability and cost management
   Route profitability process for actuals and planning data using activity-based costing rules.
10. Service industries travel and transportation profitability
    Route and waybill profitability process for actuals and planning data using activity-based costing rules.
11. Discrete industries high tech profitability and cost management
    Product, channel, and customer profitability for actuals and planning data using activity-based costing rules.
12. Consumer products profitability and cost management
    Product, channel, and customer profitability for actuals and planning data using activity-based costing rules.
13. Telecommunications profitability and cost management
    Product, channel, customer and subscription profitability for actuals and planning data using activity-based costing rules.
14. Chemicals profitability and cost management
    Product, channel, and customer profitability for actuals and planning data using activity-based costing rules.
15. Life sciences products profitability and cost management
    Product, channel, and customer profitability for actuals and planning data using activity-based costing rules.
16. Business and financial modeling

High-speed processing and analysis of big data volumes providing traceability and what-if simulation capabilities.

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

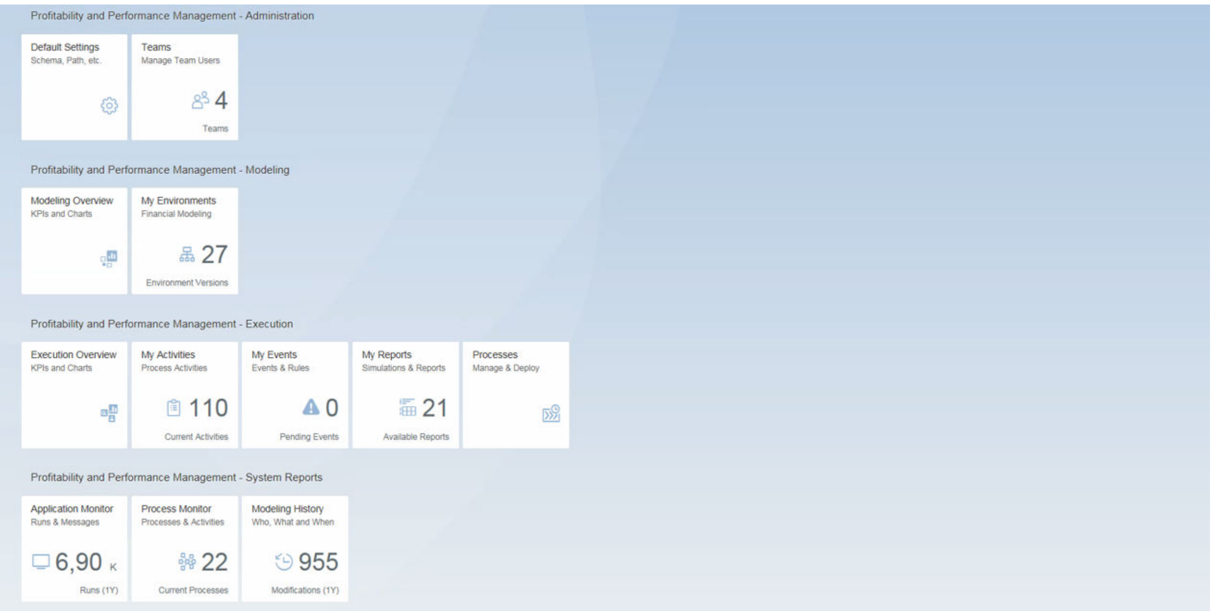
See also Applications for Business Users [page 8]

For more information about the available functions in SAP Profitability and Performance Management, see Modeling Environment [page 13].

1.1 Applications for Business Users

General applications are available for business users to help them streamline their work processes.

The following applications are available.



Application Overview

Administration

1. Default Settings
   This application allows you to enter specific settings for new environments, such as schema, path, and so on.
2. Teams
   This application allows you to manage the teams that are the basis for the assignment of activities, events, and reports to specific user groups.

Modeling

1. Modeling Overview
   This application displays predefined statistics and KPIs relating to the usage of the modeling environment during design time.
2. My Environments
   This application provides the user with access to their modeling environments.

Execution

1. Execution Overview
   This application displays predefined statistics and KPIs relating to the usage and behavior of the execution environment during runtime.
2. My Activities
   This application provides the user with one central place to access their processes and activities.
3. My Events
   This application provides the user with one central place to access their business events. The user can also access any errors that may occur during the execution of processes and activities. This application allows manual repairs as well as the configuration of automated situation handling rules.
4. My Reports
   This application provides the user with one central place to access their reports and what-if simulations.
5. Processes
   This application allows key users to deploy and un-deploy processes to user groups, including the setting of deadlines.

System Reports

1. Application Monitor
   This application displays the detailed logs of all user and batch operations.
2. Process Monitor
   This application provides an overview of the currently deployed processes.
3. Modeling History
   This application displays the change history of all environments and allows the user to retrieve historic versions.

Related Information

See also Concepts for Key Users [page 32]

For more information about the available functions in SAP Profitability and Performance Management, see Modeling Environment [page 13].

1.1.1 Default Settings

Default settings are applied to every new environment.

The default settings are maintained in a default environment with the name “Default Environment Settings” and the ID “SAP”. When a new environment is created, the default environment is copied to the new environment.

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

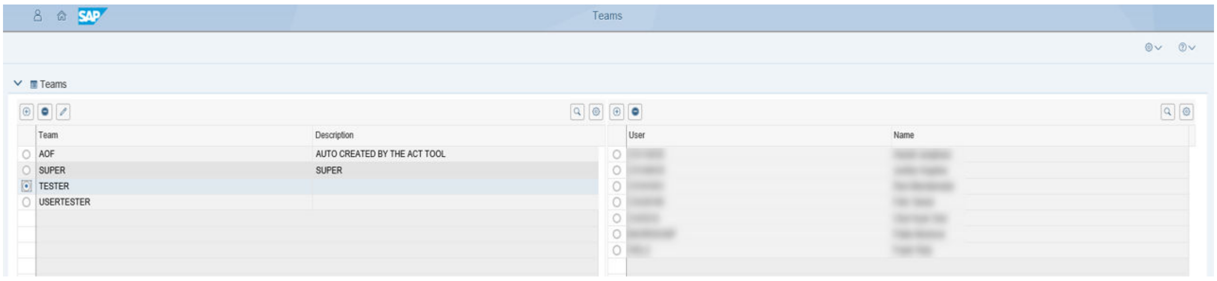
Related Information

For more information about modeling entities, see Financial and Business Modeling Entities [page 33].

For more information about the available functions, see Functions [page 52].

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.



	Team	Description
○	AOF	AUTO CREATED BY THE ACT TOOL
○	SUPER	SUPER
●	TESTER	
○	USERTESTER	

	User	Name
○	[redacted]	[redacted]
○	[redacted]	[redacted]
○	[redacted]	[redacted]
○	[redacted]	[redacted]
○	[redacted]	[redacted]
○	[redacted]	[redacted]
○	[redacted]	[redacted]

Teams and User Groups

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

The following key features are available:

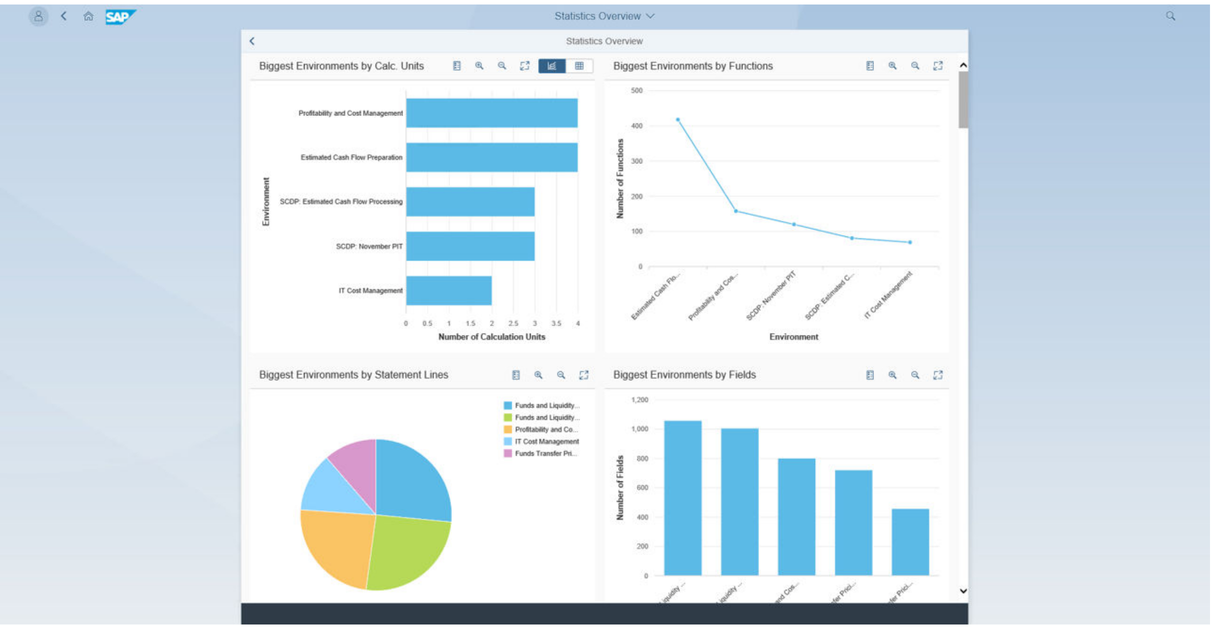
Key Feature	Use
Team Management	Teams (user groups) can be created, edited, and removed. Teams are available across all environments and even for other applications in the same client, such as SAP Business Workflow. Specific users can be added to a team and removed from a team. Users are created, edited, and deleted centrally by SAP NetWeaver administrators and not in this application.
Assignment of Teams	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.

Related Information

For more information about the use of teams in processes, see Manage and Deploy Processes [page 25].

1.1.3 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.

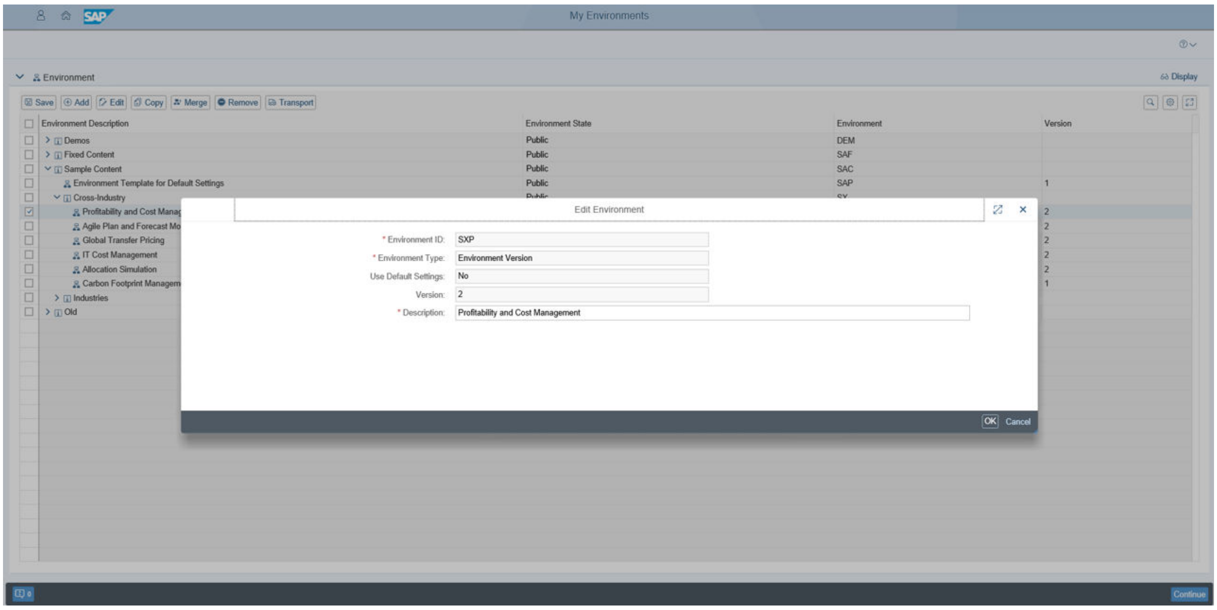
Related Information

For more information about financial and business modeling, see Financial and Business Modeling Entities [page 33].

For more information about the modeling environment, see Modeling Environment [page 13].

1.1.4 My Environments

The user can maintain multiple environments and can use nodes to structure environments for different purposes.



☐	Environment Description	Environment State	Environment	Version
☐	Demos	Public	DEM	
☐	Fixed Content	Public	SAF	
☐	Sample Content	Public	SAC	
☐	Environment Template for Default Settings	Public	SAP	1
☐	Cross-Industry	Public	SX	
☑	Profitability and Cost Manag			2
☐	Agile Plan and Forecast Mo			2
☐	Global Transfer Pricing			2
☐	IT Cost Management			2
☐	Allocation Simulation			2
☐	Carbon Footprint Managem			1
☐	Industries			
☐	Old			

* Environment ID:	SXP
* Environment Type:	Environment Version
Use Default Settings:	No
Version:	2
* Description:	Profitability and Cost Management

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, removed, and transported. Every change that is made to an environment is not only saved but is also archived with information about who made the change and when.

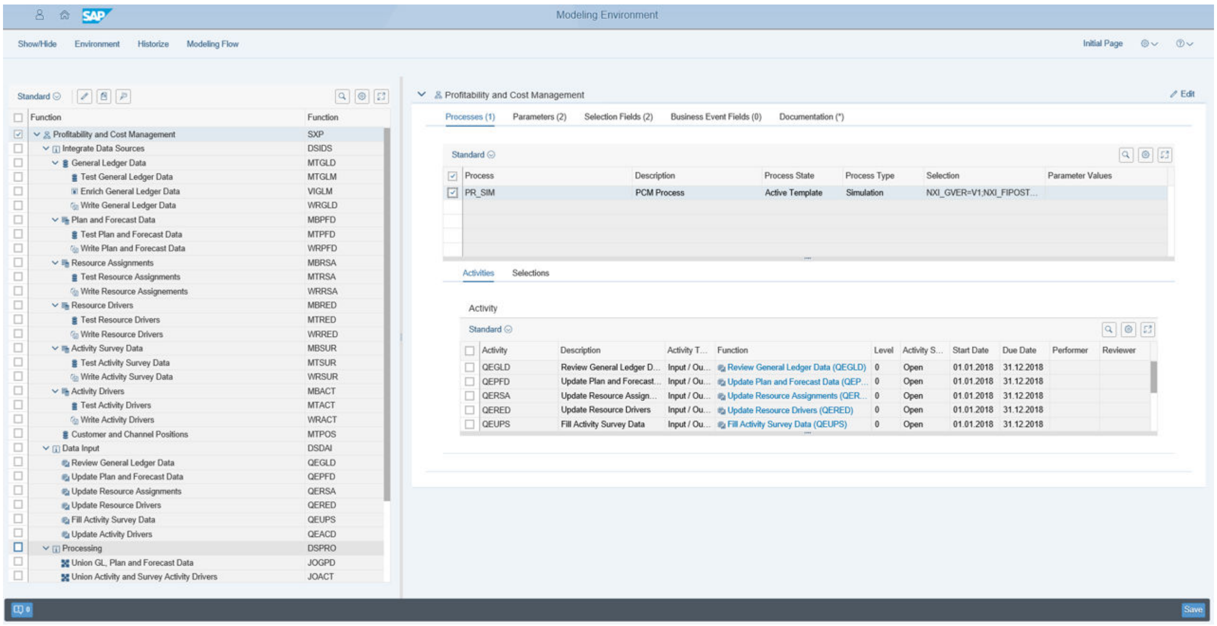
Key Feature	Use
Nodes Management	Nodes can be used to structure multiple environments in a hierarchy and, like directories, can contain other nodes. Nodes can be created, edited, removed, and transported and, unlike environments, do not have a version.
Authorizations	Authorizations can be attached to 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.
Modeling Environment	An environment can be selected. The modeling environment application is started using the Continue button and the user is able to maintain the environment.

Related Information

For more information about the modeling environment, see Modeling Environment [page 13].

1.1.5 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.



☐	Function	Function
☑	Profitability and Cost Management	SXP
☐	Integrate Data Sources	DSIDS
☐	General Ledger Data	MTGLD
☐	Test General Ledger Data	MTGLM
☐	Enrich General Ledger Data	VIGLM
☐	Write General Ledger Data	WRGLD
☐	Plan and Forecast Data	MBPFD
☐	Test Plan and Forecast Data	MTPFD
☐	Write Plan and Forecast Data	WRPFD
☐	Resource Assignments	MBRSA
☐	Test Resource Assignments	MTRSA
☐	Write Resource Assignements	WRRSA
☐	Resource Drivers	MBRED
☐	Test Resource Drivers	MTRED
☐	Write Resource Drivers	WRRED
☐	Activity Survey Data	MBSUR
☐	Test Activity Survey Data	MTSUR
☐	Write Activity Survey Data	WRSUR
☐	Activity Drivers	MBACT
☐	Test Activity Drivers	MTACT
☐	Write Activity Drivers	WRACT
☐	Customer and Channel Positions	MTPOS
☐	Data Input	DSDAI
☐	Review General Ledger Data	QEGLD
☐	Update Plan and Forecast Data	QEPFD
☐	Update Resource Assignments	QERSA
☐	Update Resource Drivers	QERED
☐	Fill Activity Survey Data	QEUPS
☐	Update Activity Drivers	QEACD
☐	Processing	DSPRO
☐	Union GL, Plan and Forecast Data	JOGPD
☐	Union Activity and Survey Activity Drivers	JOACT

☑	Process	Description	Process State	Process Type	Selection	Parameter Values
☑	PR_SIM	PCM Process	Active Template	Simulation	NXI_GVER=V1;NXI_FIPOST...	

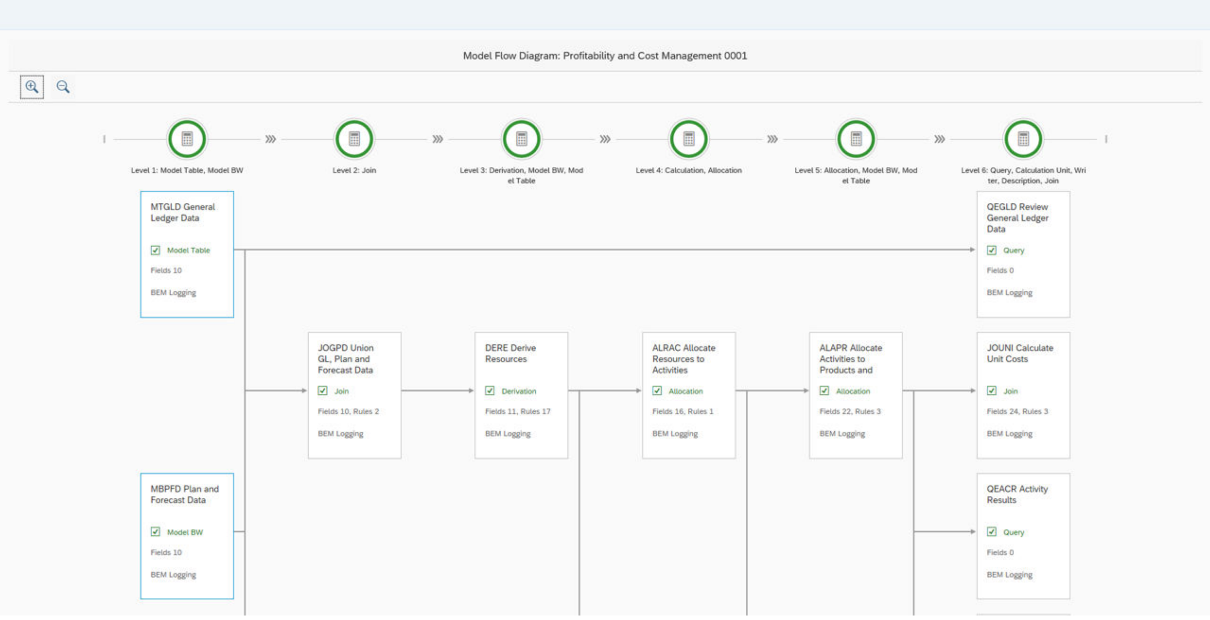
Activity

☐	Activity	Description	Activity T...	Function	Level	Activity S...	Start Date	Due Date	Performer	Reviewer
☐	QEGLD	Review General Ledger D...	Input / Ou...	Review General Ledger Data (QEGLD)	0	Open	01.01.2018	31.12.2018		
☐	QEPFD	Update Plan and Forecast...	Input / Ou...	Update Plan and Forecast Data (QEP...	0	Open	01.01.2018	31.12.2018		
☐	QERSA	Update Resource Assign...	Input / Ou...	Update Resource Assignments (QER...	0	Open	01.01.2018	31.12.2018		
☐	QERED	Update Resource Drivers	Input / Ou...	Update Resource Drivers (QERED)	0	Open	01.01.2018	31.12.2018		
☐	QEUPS	Fill Activity Survey Data	Input / Ou...	Fill Activity Survey Data (QEUPS)	0	Open	01.01.2018	31.12.2018		

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: Profitability and Cost Management 0001
Level 1: Model Table, Model BW
Level 2: Join
Level 3: Derivation, Model BW, Mod el Table
Level 4: Calculation, Allocation
Level 5: Allocation, Model BW, Mod el Table
Level 6: Query, Calculation Unit, Wri ter, Description, Join
MTGLD General Ledger Data
☑ Model Table
Fields 10
BEM Logging
MBPFD Plan and Forecast Data
☑ Model BW
Fields 10
BEM Logging
JOGPD Union GL, Plan and Forecast Data
☑ Join
Fields 10, Rules 2
BEM Logging
DERE Derive Resources
☑ Derivation
Fields 11, Rules 17
BEM Logging
ALRAC Allocate Resources to Activities
☑ Allocation
Fields 16, Rules 1
BEM Logging
ALAPR Allocate Activities to Products and
☑ Allocation
Fields 22, Rules 3
BEM Logging
QEGLD Review General Ledger Data
☑ Query
Fields 0
BEM Logging
JOUNI Calculate Unit Costs
☑ Join
Fields 24, Rules 3
BEM Logging
QEACR Activity Results
☑ Query
Fields 0
BEM Logging

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:

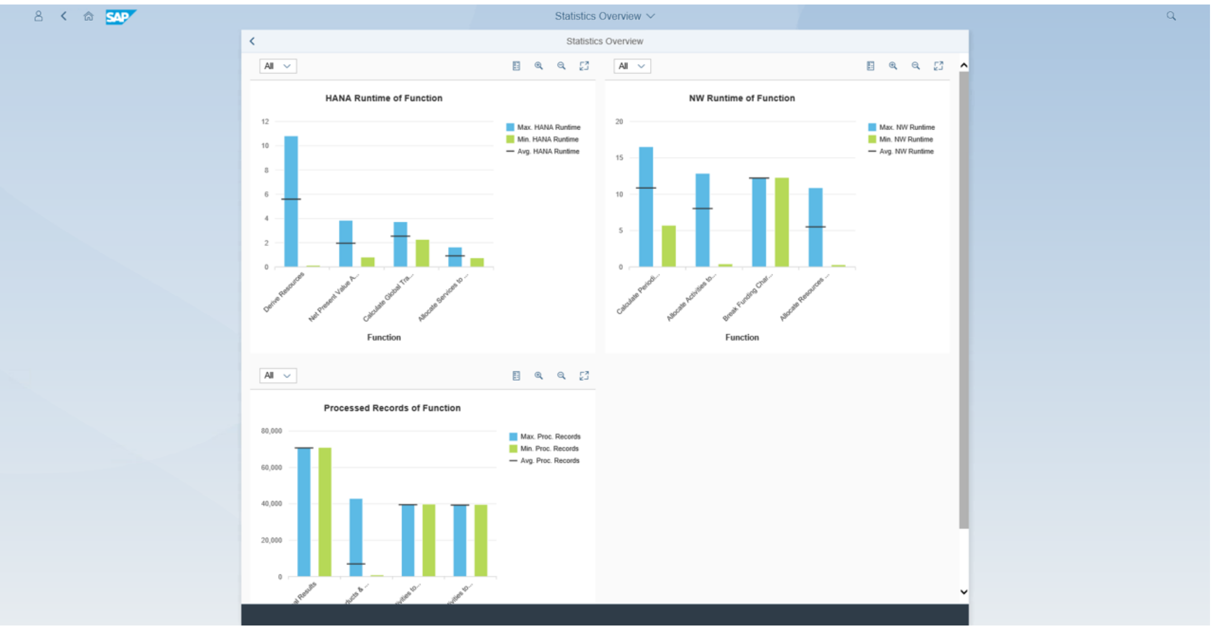
Key Feature	Use
Function Hierarchy	The modeling environment allows the construction and maintenance of a model by adding and connecting multiple functions into a common network. The output of a function can be the input of other functions and thus contribute to the logic of the model. These functions 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. Functions can be added, removed, changed, and copied. In change mode, the `hierarchy` 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 `hierarchy` function or switch to change mode themselves. Where-used lists and a network diagram can visualize the logical dependencies of the output input relationships.
Function Details	When a function is selected in the `hierarchy` function, the function details are displayed on the right. 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. 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.
Environment Details	The *Environment* button in the screen header opens the details of the environment.
Environment Historization	The *Historize* 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.

Related Information

For more information about the available functions, see Functions [page 52].

1.1.6 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	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.

Related Information

For more information about environments, see Financial and Business Modeling Entities [page 33].

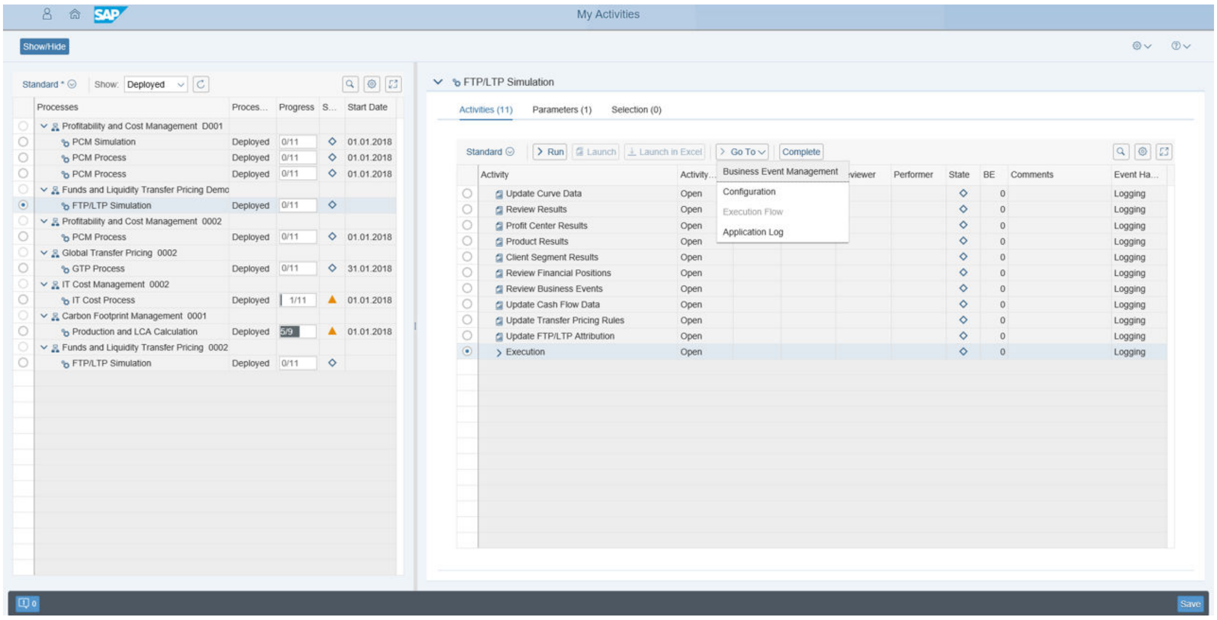
For more information about how to define processes, activities, parameters, and selection fields, see Calculation Unit [page 57].

For more information about how to deploy processes, see Manage and Deploy Processes [page 25].

For more information about how to monitor processes, see Process Monitor [page 30].

1.1.7 My Activities

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



	Processes	Proces...	Progress	S...	Start Date
○	Profitability and Cost Management D001				
○	PCM Simulation	Deployed	0/11	◇	01.01.2018
○	PCM Process	Deployed	0/11	◇	01.01.2018
○	PCM Process	Deployed	0/11	◇	01.01.2018
○	Funds and Liquidity Transfer Pricing Demo				
●	FTP/LTP Simulation	Deployed	0/11	◇	
○	Profitability and Cost Management 0002				
○	PCM Process	Deployed	0/11	◇	01.01.2018
○	Global Transfer Pricing 0002				
○	GTP Process	Deployed	0/11	◇	31.01.2018
○	IT Cost Management 0002				
○	IT Cost Process	Deployed	1/11	▲	01.01.2018
○	Carbon Footprint Management 0001				
○	Production and LCA Calculation	Deployed	5/9	▲	01.01.2018
○	Funds and Liquidity Transfer Pricing 0002				
○	FTP/LTP Simulation	Deployed	0/11	◇	

	Activity	Activity...			viewer	Performer	State	BE	Comments	Event Ha...
○	Update Curve Data	Open					◇	0		Logging
○	Review Results	Open					◇	0		Logging
○	Profit Center Results	Open					◇	0		Logging
○	Product Results	Open					◇	0		Logging
○	Client Segment Results	Open					◇	0		Logging
○	Review Financial Positions	Open					◇	0		Logging
○	Review Business Events	Open					◇	0		Logging
○	Update Cash Flow Data	Open					◇	0		Logging
○	Update Transfer Pricing Rules	Open					◇	0		Logging
○	Update FTP/LTP Attribution	Open					◇	0		Logging
●	Execution	Open					◇	0		Logging

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	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. A progress indicator shows how many of the activities are already finished.
Activities	If a process is selected, the relevant activities for the user are displayed on the right. Activities can require two types of attention: 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. 2. Execution This type of activity triggers automatic logic and calculations. In this case, users run a function to produce interim or final results. 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.
Parameters	All the parameters that are relevant for the execution of the process activities are listed here with their values. 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.
Selections	All the selections that are relevant for the execution of the process activities are listed here. 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.

Related Information

For more information about displaying and editing data, see Analytics Component [page 23].

For more information about how to define processes, activities, parameters, and selection fields, see Calculation Unit [page 57].

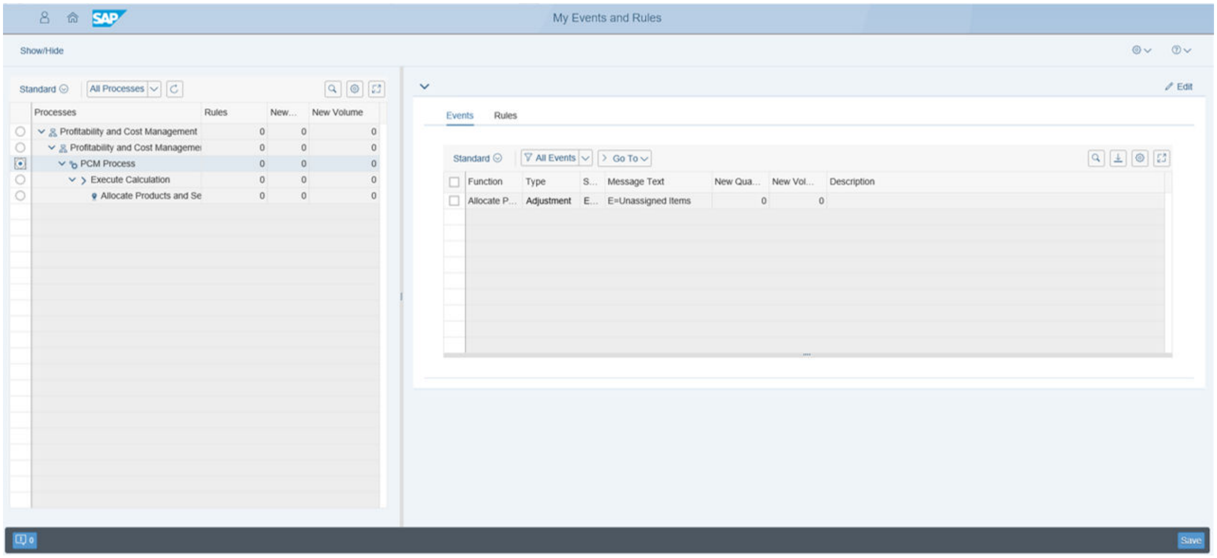
For more information about how to deploy processes, see Manage and Deploy Processes [page 25].

For more information about how to monitor processes, see Process Monitor [page 30].

1.1.8 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.



	Processes	Rules	New...	New Volume
○	Profitability and Cost Management	0	0	0
○	Profitability and Cost Manageme	0	0	0
●	PCM Process	0	0	0
○	Execute Calculation	0	0	0
○	Allocate Products and Se	0	0	0

□	Function	Type	S...	Message Text	New Qua...	New Vol...	Description
□	Allocate P...	Adjustment	E...	E=Unassigned Items	0	0	

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:

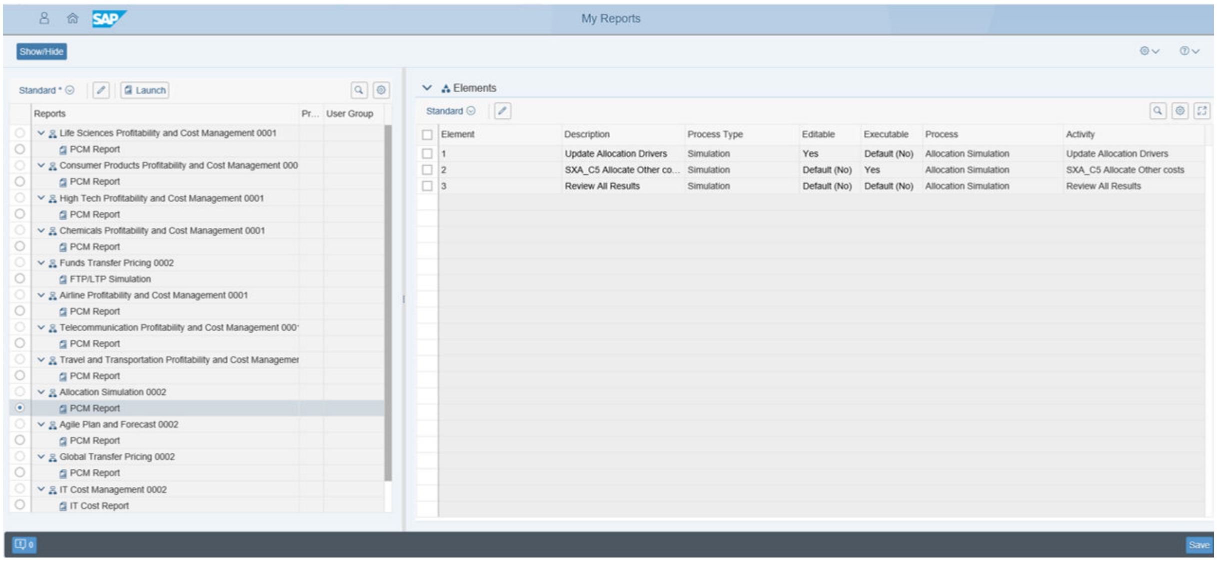
Key Feature	Use
Processes	Processes are displayed in a hierarchy on the left together with the environment to which they belong. By default, only the current processes that need attention from the user are displayed. All processes can be displayed, including finished processes. The user can expand the hierarchy to drill down further to see the activity and the function of the activity in which the business event occurred. 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.
Events	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. The user can select an event to view the detailed data and decide what steps to take to resolve the situation: 1. Event The event will not be handled and left in an open status. 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. 3. Transmit The erroneous record will be moved directly to the result without adjustment. 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.
Rules	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.

Related Information

For more information about the definition of business event fields, see Calculation Unit [page 57].

1.1.9 My Reports

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



Reports	Pr...	User Group
○ Life Sciences Profitability and Cost Management 0001		
○ PCM Report		
○ Consumer Products Profitability and Cost Management 000		
○ PCM Report		
○ High Tech Profitability and Cost Management 0001		
○ PCM Report		
○ Chemicals Profitability and Cost Management 0001		
○ PCM Report		
○ Funds Transfer Pricing 0002		
○ FTP/LTP Simulation		
○ Airline Profitability and Cost Management 0001		
○ PCM Report		
○ Telecommunication Profitability and Cost Management 000		
○ PCM Report		
○ Travel and Transportation Profitability and Cost Managemer		
○ PCM Report		
○ Allocation Simulation 0002		
● PCM Report		
○ Agile Plan and Forecast 0002		
○ PCM Report		
○ Global Transfer Pricing 0002		
○ PCM Report		
○ IT Cost Management 0002		
○ IT Cost Report		

□ Element	Description	Process Type	Editable	Executable	Process	Activity
□ 1	Update Allocation Drivers	Simulation	Yes	Default (No)	Allocation Simulation	Update Allocation Drivers
□ 2	SXA_C5 Allocate Other co...	Simulation	Default (No)	Yes	Allocation Simulation	SXA_C5 Allocate Other costs
□ 3	Review All Results	Simulation	Default (No)	Default (No)	Allocation Simulation	Review All Results

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

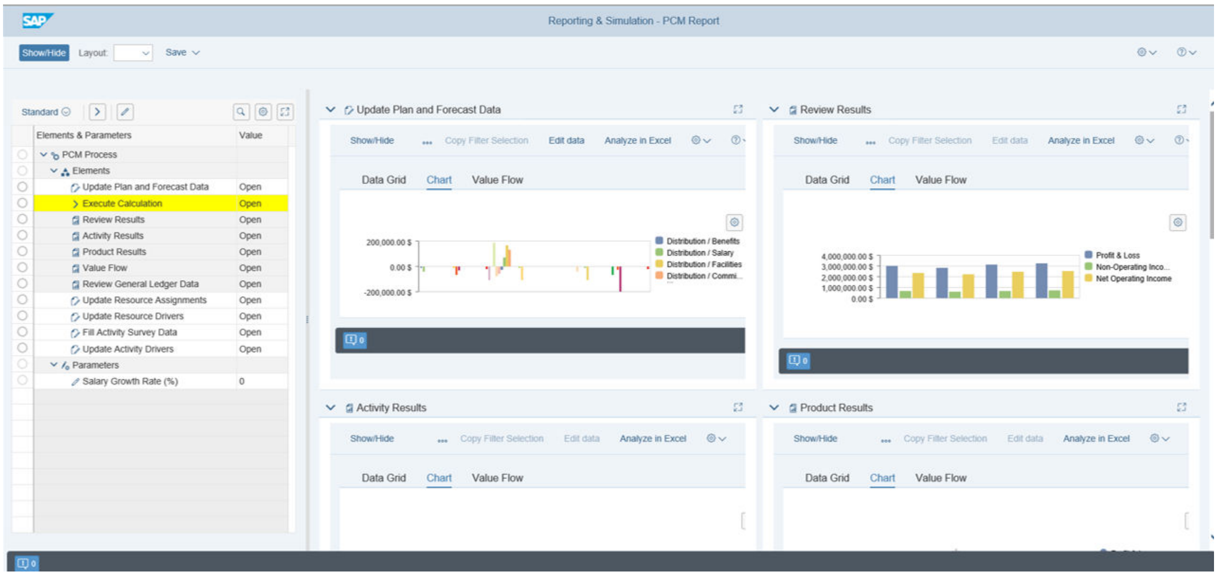
Key Feature	Use
Elements of a Report	Reports consist of one or more elements, where each element refers to a process. Reports and elements inherit all their settings from the underlying process and activities like default layouts, teams, and authorizations. If processes are included and the process has 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 22].

1.1.10 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.



Elements & Parameters	Value
PCM Process	
Elements	
Update Plan and Forecast Data	Open
Execute Calculation	Open
Review Results	Open
Activity Results	Open
Product Results	Open
Value Flow	Open
Review General Ledger Data	Open
Update Resource Assignments	Open
Update Resource Drivers	Open
Fill Activity Survey Data	Open
Update Activity Drivers	Open
Parameters	
Salary Growth Rate (%)	0

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.

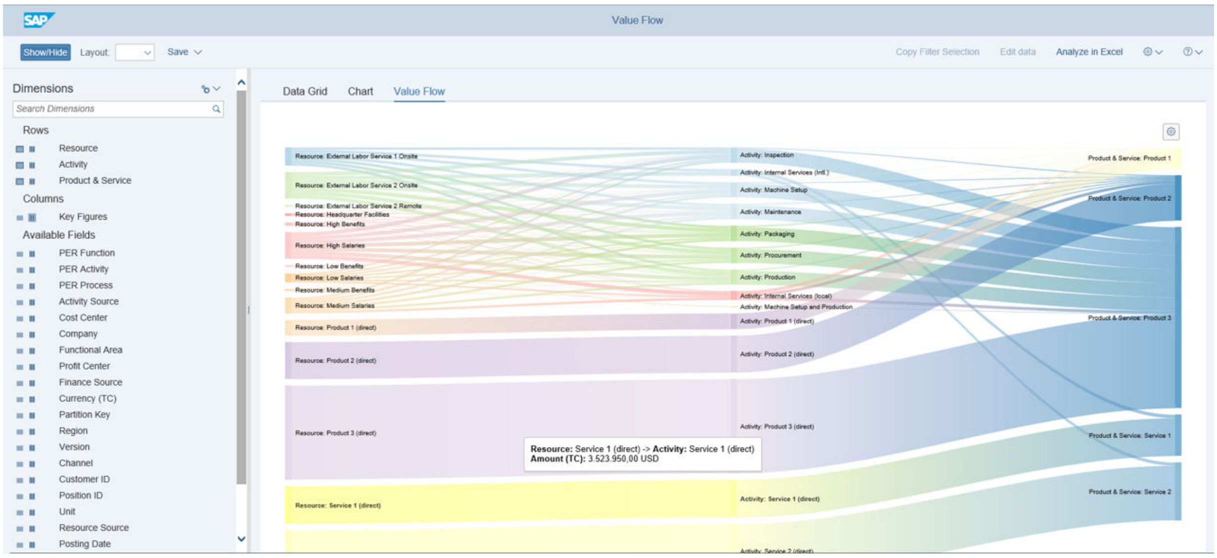
Key Feature	Use
Elements and Parameters	A list of all element titles and parameters available in the report is shown on the left-hand side of the screen. If what-if simulation is enabled, parameters can be changed and the execution of activities is also possible.
Charts and Tables	All input/output activities 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 23].

1.1.10.1 Analytics Component

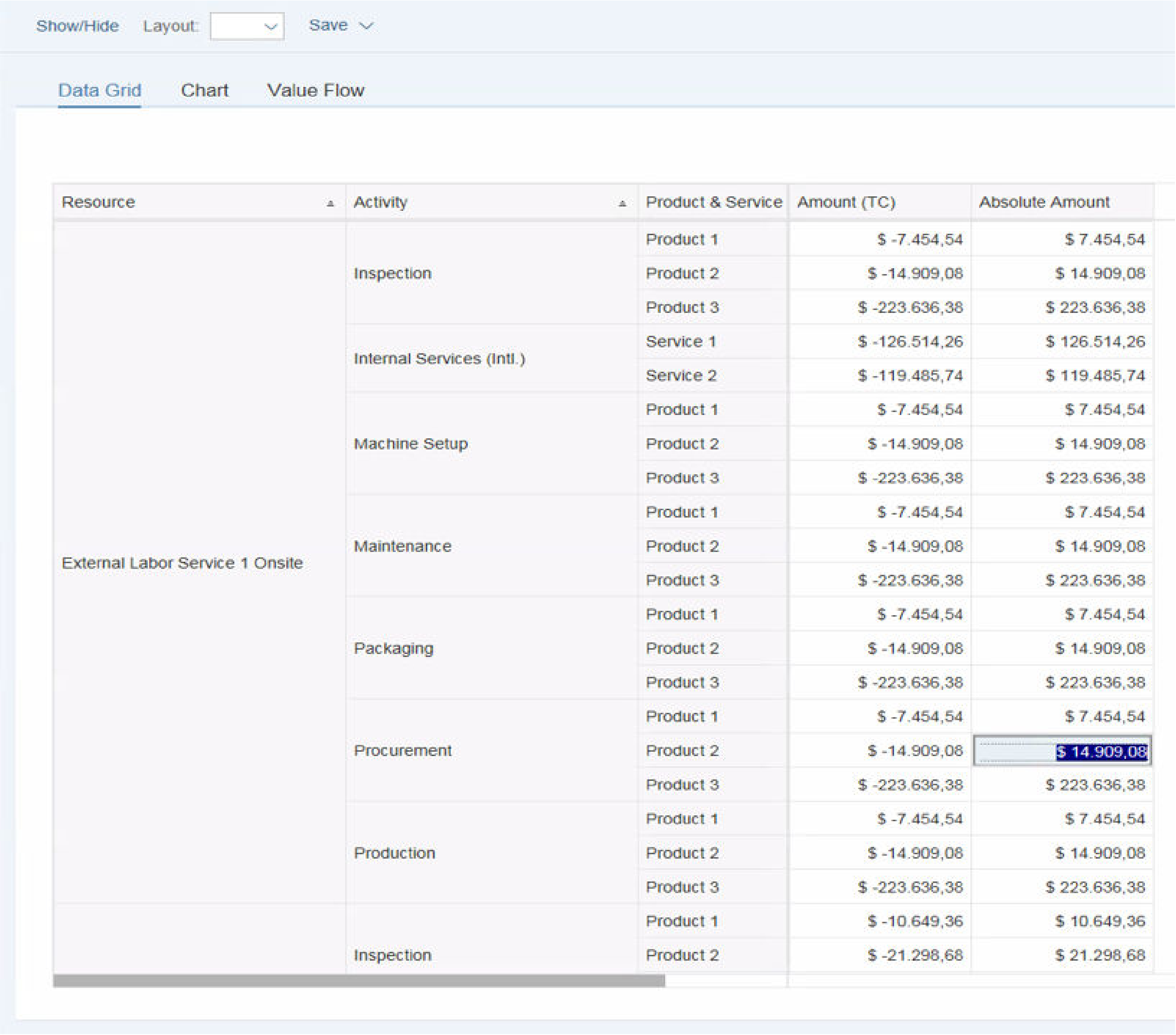
The analytics component is the standard application to visualize data. It allows interactive self-service reporting, where users can display data in data grids and charts.


SAP
Value Flow
Show/Hide
Layout:
Save
Copy Filter Selection
Edit data
Analyze in Excel
Dimensions
Search Dimensions
Rows
Resource
Activity
Product & Service
Columns
Key Figures
Available Fields
PER Function
PER Activity
PER Process
Activity Source
Cost Center
Company
Functional Area
Profit Center
Finance Source
Currency (TC)
Partition Key
Region
Version
Channel
Customer ID
Position ID
Unit
Resource Source
Posting Date
Data Grid
Chart
Value Flow
Resource: External Labor Service 1 Onsite
Resource: External Labor Service 2 Onsite
Resource: External Labor Service 2 Remote
Resource: Headquarter Facilities
Resource: High Benefits
Resource: High Salaries
Resource: Low Benefits
Resource: Low Salaries
Resource: Medium Benefits
Resource: Medium Salaries
Resource: Product 1 (direct)
Resource: Product 2 (direct)
Resource: Product 3 (direct)
Resource: Service 1 (direct)
Activity: Inspection
Activity: Internal Services (Intl.)
Activity: Machine Setup
Activity: Maintenance
Activity: Packaging
Activity: Procurement
Activity: Production
Activity: Internal Services (local)
Activity: Machine Setup and Production
Activity: Product 1 (direct)
Activity: Product 2 (direct)
Activity: Product 3 (direct)
Activity: Service 1 (direct)
Activity: Service 2 (direct)
Product & Service: Product 1
Product & Service: Product 2
Product & Service: Product 3
Product & Service: Service 1
Product & Service: Service 2

Resource: Service 1 (direct) -> **Activity:** Service 1 (direct)
Amount (TC): 3.523.950,00 USD

Analytics Component

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



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	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: - Sorting - Filtering - Use of master data hierarchies - Display of IDs or texts for characteristic data

Key Feature	Use
Data Grid	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. If the underlying query is input-enabled, data can also be edited and saved.
Chart	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.
Value Flow	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.

Related Information

For more information about the analytics component, see Analytics Component.

1.1.11 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.

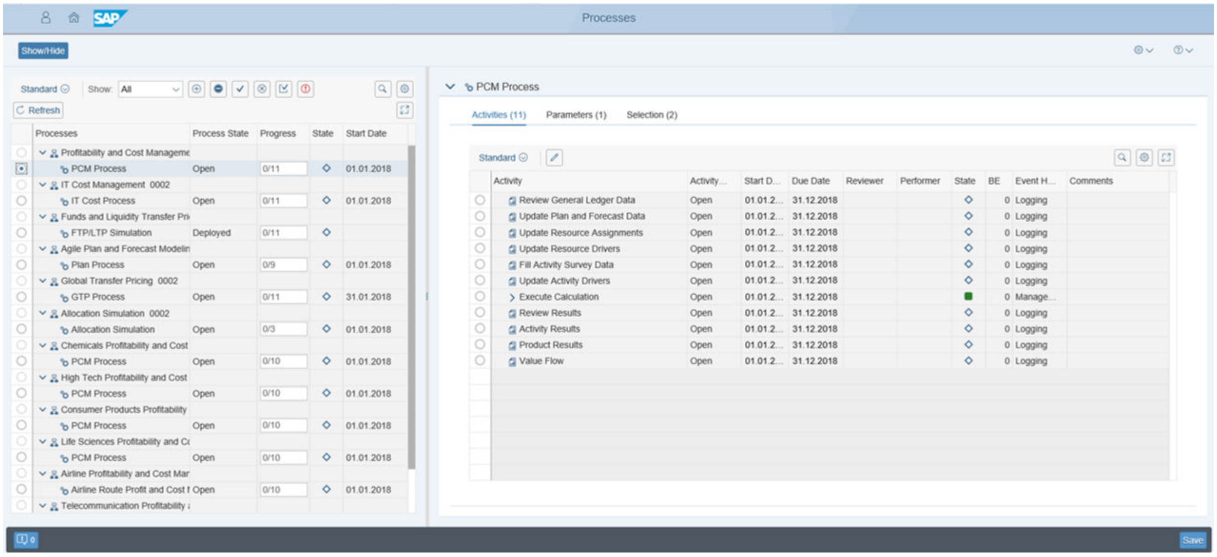
The default state after creation is “Open”. Only processes in the state “Deployed” are visible in the *My Activities* application.

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 activites from the *My Activities* application by choosing the *My Activities* pushbutton in the header of the *Manage and Deploy Processes* application".



Processes	Process State	Progress	State	Start Date
Profitability and Cost Manageme				
PCM Process	Open	0/11	◇	01.01.2018
IT Cost Management 0002				
IT Cost Process	Open	0/11	◇	01.01.2018
Funds and Liquidity Transfer Pri				
FTP/LTP Simulation	Deployed	0/11	◇	
Agile Plan and Forecast Modelin				
Plan Process	Open	0/9	◇	01.01.2018
Global Transfer Pricing 0002				
GTP Process	Open	0/11	◇	31.01.2018
Allocation Simulation 0002				
Allocation Simulation	Open	0/3	◇	01.01.2018
Chemicals Profitability and Cost				
PCM Process	Open	0/10	◇	01.01.2018
High Tech Profitability and Cost				
PCM Process	Open	0/10	◇	01.01.2018
Consumer Products Profitability				
PCM Process	Open	0/10	◇	01.01.2018
Life Sciences Profitability and Cc				
PCM Process	Open	0/10	◇	01.01.2018
Airline Profitability and Cost Mar				
Airline Route Profit and Cost I	Open	0/10	◇	01.01.2018
Telecommunication Profitability a				

Activity	Activity...	Start D...	Due Date	Reviewer	Performer	State	BE	Event H...	Comments
Review General Ledger Data	Open	01.01.2...	31.12.2018			◇	0	Logging	
Update Plan and Forecast Data	Open	01.01.2...	31.12.2018			◇	0	Logging	
Update Resource Assignments	Open	01.01.2...	31.12.2018			◇	0	Logging	
Update Resource Drivers	Open	01.01.2...	31.12.2018			◇	0	Logging	
Fill Activity Survey Data	Open	01.01.2...	31.12.2018			◇	0	Logging	
Update Activity Drivers	Open	01.01.2...	31.12.2018			◇	0	Logging	
Execute Calculation	Open	01.01.2...	31.12.2018			🟩	0	Manage...	
Review Results	Open	01.01.2...	31.12.2018			◇	0	Logging	
Activity Results	Open	01.01.2...	31.12.2018			◇	0	Logging	
Product Results	Open	01.01.2...	31.12.2018			◇	0	Logging	
Value Flow	Open	01.01.2...	31.12.2018			◇	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	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: • *Open* 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. • *Deployed* 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. • *Suspended* 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 *Deployed*, *Aborted* or *Completed*. • *Completed* If the activities of a deployed process are finished, the process can be set to *Completed*. • *Aborted* If a process needs to be terminated without success, it can be set to *Aborted*.
Activities	If a process is selected, the activities are displayed on the right-hand side of the screen. Only if the process state is *Open* or *Suspended*, can changes be applied to the activity state, start date, due date, reviewer and performer team as well as to the parameters and selections. The activity can have various states: • *Open* The activity is open for execution. • *Pending* The activity is not open for execution yet because preceding activities are not finished yet. • *In Approval* A dual control principle workflow is attached to the activity and this is not finished yet. • *Completed* The activity is completed.

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

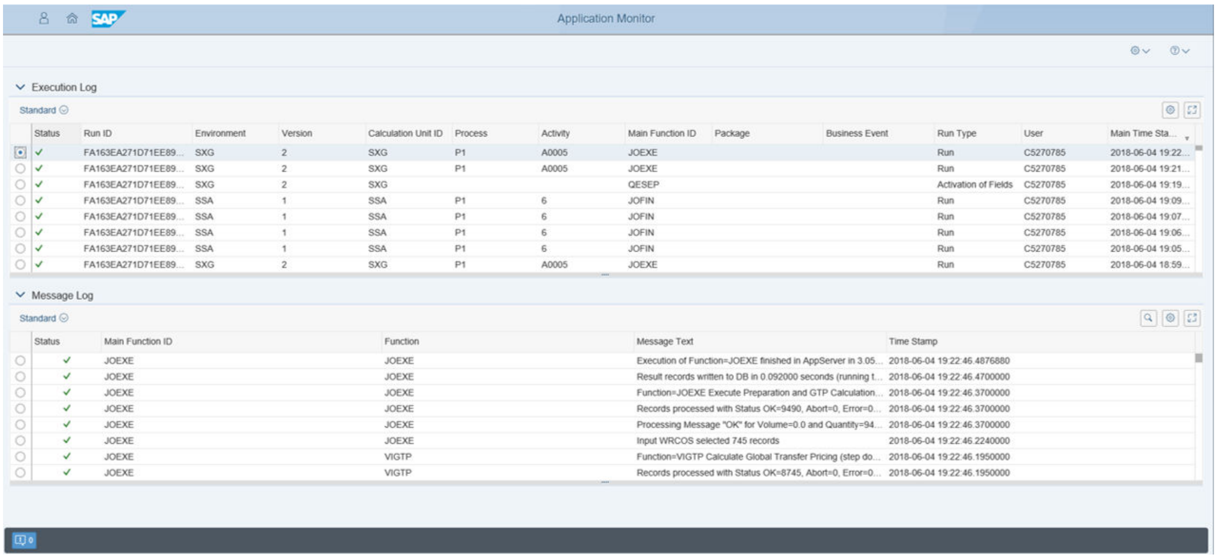
Related Information

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 57].

1.1.12 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.



Execution Log

Standard

	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

Standard

	Status	Main Function ID	Function	Message Text	Time Stamp
○	✓	JOEXE	JOEXE	Execution of Function=JOEXE finished in AppServer in 3.05...	2018-06-04 19:22:46.4876880
○	✓	JOEXE	JOEXE	Result records written to DB in 0.092000 seconds (running t...	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.

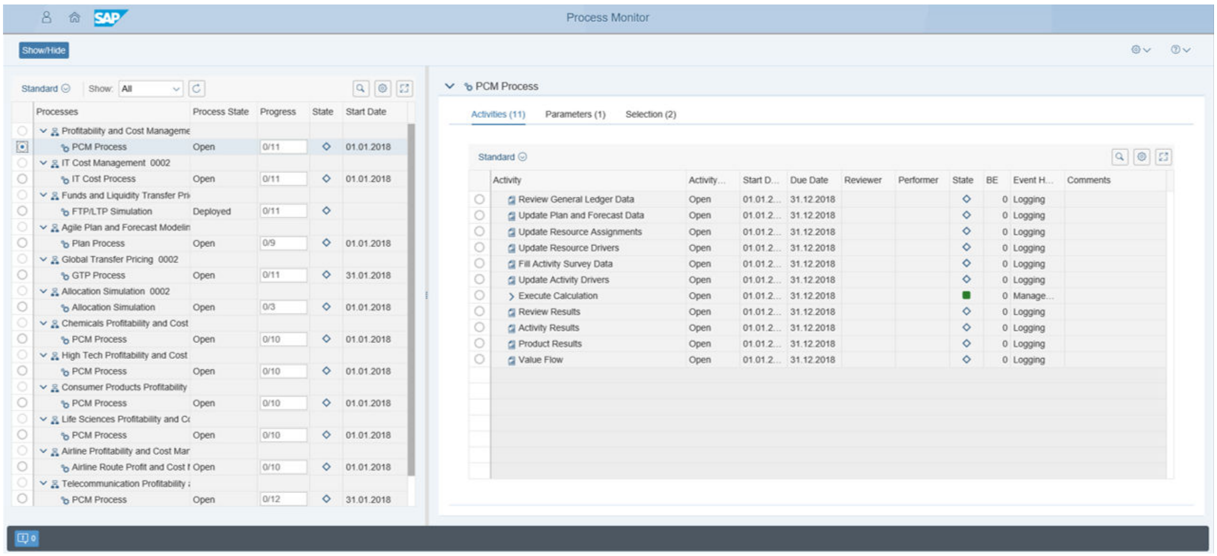
Key Feature	Use
Run Log	The application creates a unique log entry each time an individual function is generated and run. The log contains the following information: ● A unique run ID ● A status ● A run set ID ● An input set ID ● A timestamp ● The name of the user who executed the run or generated the function
Message Log	This contains the list of messages that are associated with every execution. The list of messages usually contains the following information: ● The status and results of a run ● 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) ● The results of a generation

Related Information

For more information about the definition of custom specific checks that are logged in the application monitor, see Environment [page 54].

1.1.13 Process Monitor

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



Processes	Process State	Progress	State	Start Date
Profitability and Cost Manageme				
PCM Process	Open	0/11	◇	01.01.2018
IT Cost Management 0002				
IT Cost Process	Open	0/11	◇	01.01.2018
Funds and Liquidity Transfer Pri				
FTP/LTP Simulation	Deployed	0/11	◇	
Agile Plan and Forecast Modelin				
Plan Process	Open	0/9	◇	01.01.2018
Global Transfer Pricing 0002				
GTP Process	Open	0/11	◇	31.01.2018
Allocation Simulation 0002				
Allocation Simulation	Open	0/3	◇	01.01.2018
Chemicals Profitability and Cost				
PCM Process	Open	0/10	◇	01.01.2018
High Tech Profitability and Cost				
PCM Process	Open	0/10	◇	01.01.2018
Consumer Products Profitability				
PCM Process	Open	0/10	◇	01.01.2018
Life Sciences Profitability and Co				
PCM Process	Open	0/10	◇	01.01.2018
Airline Profitability and Cost Mar				
Airline Route Profit and Cost I	Open	0/10	◇	01.01.2018
Telecommunication Profitability a				
PCM Process	Open	0/12	◇	31.01.2018

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

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.

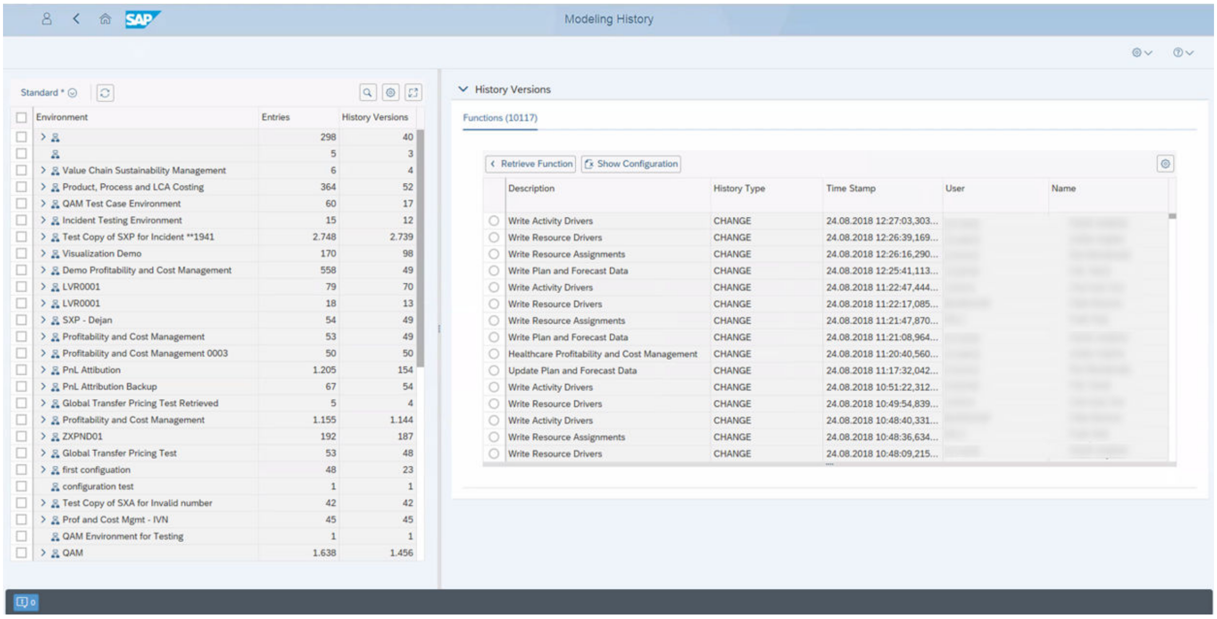
Related Information

For more information on how to define processes, activities, parameters and selection fields, see Calculation Unit [page 57].

For more information on how to deploy processes, see Processes [page 25].

1.1.14 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.



	Environment	Entries	History Versions
□	>	298	40
□		5	3
□	> Value Chain Sustainability Management	6	4
□	> Product, Process and LCA Costing	364	52
□	> QAM Test Case Environment	60	17
□	> Incident Testing Environment	15	12
□	> Test Copy of SXP for Incident **1941	2.748	2.739
□	> Visualization Demo	170	98
□	> Demo Profitability and Cost Management	558	49
□	> LVR0001	79	70
□	> LVR0001	18	13
□	> SXP - Dejan	54	49
□	> Profitability and Cost Management	53	49
□	> Profitability and Cost Management 0003	50	50
□	> PnL Attibution	1.205	154
□	> PnL Attribution Backup	67	54
□	> Global Transfer Pricing Test Retrieved	5	4
□	> Profitability and Cost Management	1.155	1.144
□	> ZXPND01	192	187
□	> Global Transfer Pricing Test	53	48
□	> first configuation	48	23
□	configuration test	1	1
□	> Test Copy of SXA for Invalid number	42	42
□	> Prof and Cost Mgmt - IVN	45	45
□	QAM Environment for Testing	1	1
□	> QAM	1.638	1.456

	Description	History Type	Time Stamp	User	Name
○	Write Activity Drivers	CHANGE	24.08.2018 12:27:03,303...	[redacted]	[redacted]
○	Write Resource Drivers	CHANGE	24.08.2018 12:26:39,169...	[redacted]	[redacted]
○	Write Resource Assignments	CHANGE	24.08.2018 12:26:16,290...	[redacted]	[redacted]
○	Write Plan and Forecast Data	CHANGE	24.08.2018 12:25:41,113...	[redacted]	[redacted]
○	Write Activity Drivers	CHANGE	24.08.2018 11:22:47,444...	[redacted]	[redacted]
○	Write Resource Drivers	CHANGE	24.08.2018 11:22:17,085...	[redacted]	[redacted]
○	Write Resource Assignments	CHANGE	24.08.2018 11:21:47,870...	[redacted]	[redacted]
○	Write Plan and Forecast Data	CHANGE	24.08.2018 11:21:08,964...	[redacted]	[redacted]
○	Healthcare Profitability and Cost Management	CHANGE	24.08.2018 11:20:40,560...	[redacted]	[redacted]
○	Update Plan and Forecast Data	CHANGE	24.08.2018 11:17:32,042...	[redacted]	[redacted]
○	Write Activity Drivers	CHANGE	24.08.2018 10:51:22,312...	[redacted]	[redacted]
○	Write Resource Drivers	CHANGE	24.08.2018 10:49:54,839...	[redacted]	[redacted]
○	Write Activity Drivers	CHANGE	24.08.2018 10:48:40,331...	[redacted]	[redacted]
○	Write Resource Assignments	CHANGE	24.08.2018 10:48:36,634...	[redacted]	[redacted]
○	Write Resource Drivers	CHANGE	24.08.2018 10:48:09,215...	[redacted]	[redacted]

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 13].

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 RDL data storage.

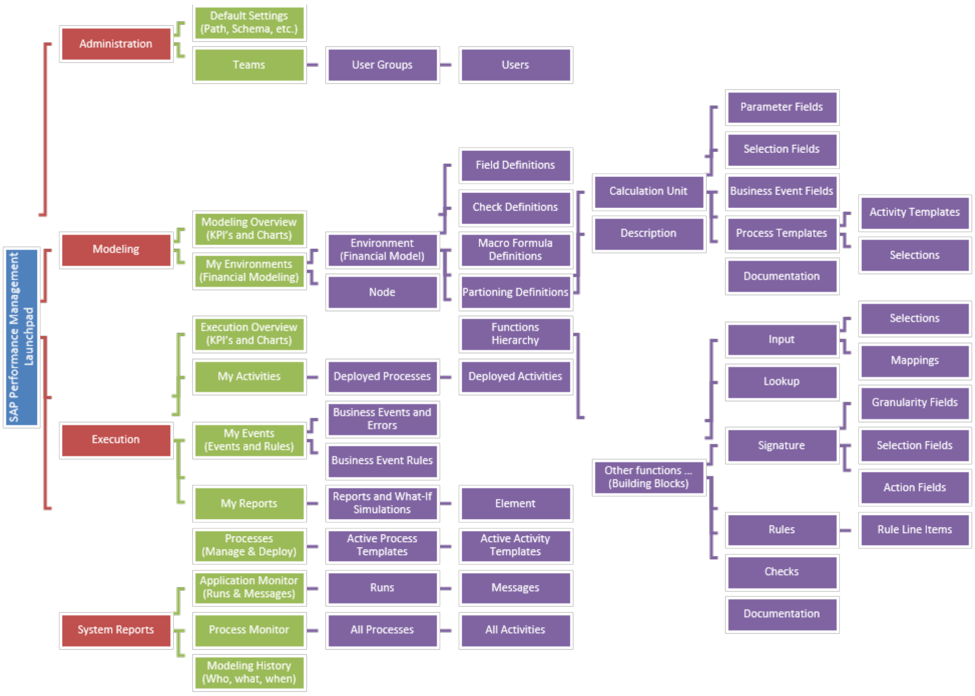
Related Information

For more information about SAP Profitability and Performance Management functions see Functions [page 52].

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 52].

For more information about entities see:

- Default Settings [page 9]
- Teams [page 10]
- Modeling Overview [page 11]
- My Environments [page 12]
- Modeling Environment [page 13]
- Execution Overview [page 16]
- My Activities [page 17]
- My Events [page 19]
- My Reports [page 21]
- Manage and Deploy Processes [page 25]
- Application Monitor [page 28]
- Process Monitor [page 30]
- Modeling History [page 31]

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.

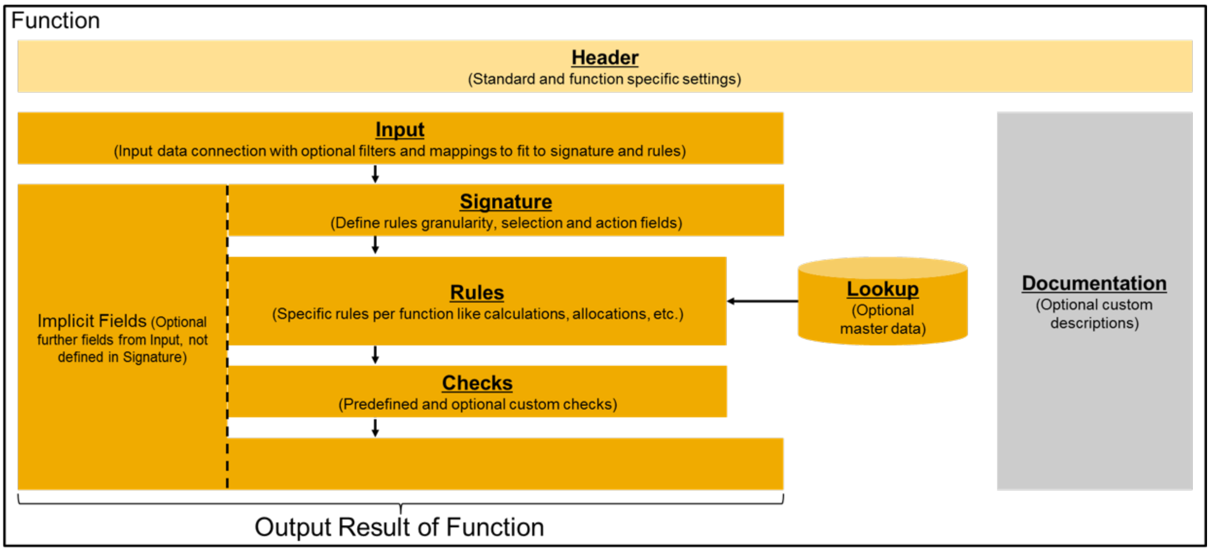
The following function categories are available:

1. Information functions
   These functions define the data and information model in an environment and comprise `Model BW`, `Model RDL`, `Model Table` and `Model View`. 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.
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 (for example, the `Calculation`, `Allocation`, `Valuation` and `Derivation` functions).
3. Write and Adapter functions
   This category comprises the `Write` function, the `Remote Function Adapter` and the `File Adapter` function of type export. These functions can store data or hand over data to external systems for further processing. They also provide output to subsequent functions.
4. Query function
   The `Query` function defines whether the data display for a user is read-only or editable. For the latter, 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 and reporting.
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 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.

6. Description function
   The `Description` function helps to structure and document the model. It has no effect on the result but improves the readability of the model.

Depending on the categories explained above, some or all of the building blocks are relevant for a function.

The figure below shows a general overview of the function building blocks that appear as tabs in the function details user interface.


Function
Header
(Standard and function specific settings)
Input
(Input data connection with optional filters and mappings to fit to signature and rules)
Signature
(Define rules granularity, selection and action fields)
Rules
(Specific rules per function like calculations, allocations, etc.)
Checks
(Predefined and optional custom checks)
Implicit Fields (Optional further fields from Input, not defined in Signature)
Lookup
(Optional master data)
Documentation
(Optional custom descriptions)
Output Result of Function

Function Building Blocks

The following function building blocks are available:

- Header [page 35]
- Input [page 36]
- Lookup [page 37]
- Signature [page 37]
- Rules [page 39]
- Checks [page 39]

Related Information

For more information about SAP Profitability and Performance Management functions see Functions [page 52].

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. Central Result Model Table
   If a model table is assigned here, the (interim) results of the function are stored in the model table. Fields that are in the results of a function, but not included in the model table are not automatically included or disregarded. If no model table is assigned, the following scenarios apply:
   1. If the processing type of the function is set to “executable” or the function is directly executed in the modeling environment for testing, the (interim) results are stored in a function-specific temporary table.
   2. Otherwise, the (interim) results are not stored and are passed on to a subsequent function.
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

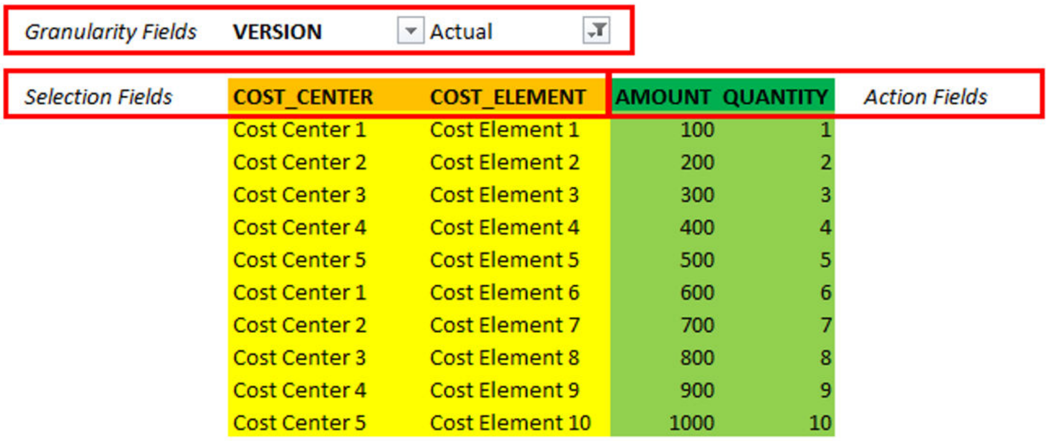
All processing functions have a signature, which can produce a result for subsequent functions. The signature defines the minimum number of relevant fields of a function. There can also be further implicit fields from the input. These simply pass through the function without any change or any effect 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

COST_CENTER	COST_ELEMENT	AMOUNT	QUANTITY
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 => 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 52].

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

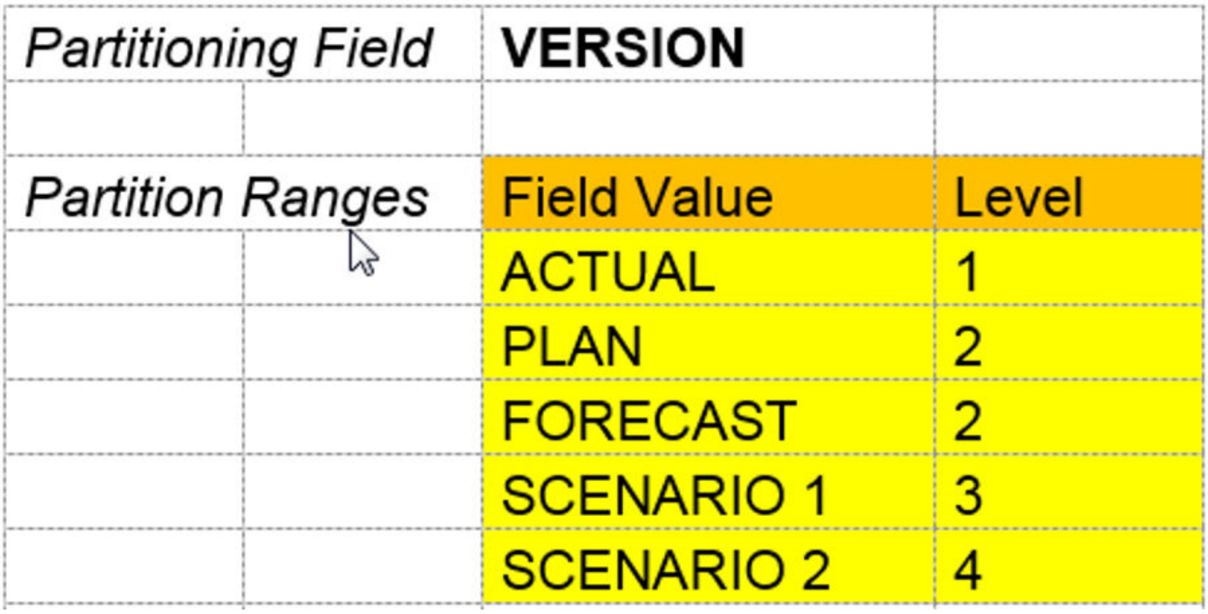
You set up partitioning in the following two steps:

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



Partitioning Field	VERSION	
Partition Ranges	Field Value	Level
	ACTUAL	1
	PLAN	2
	FORECAST	2
	SCENARIO 1	3
	SCENARIO 2	4

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:

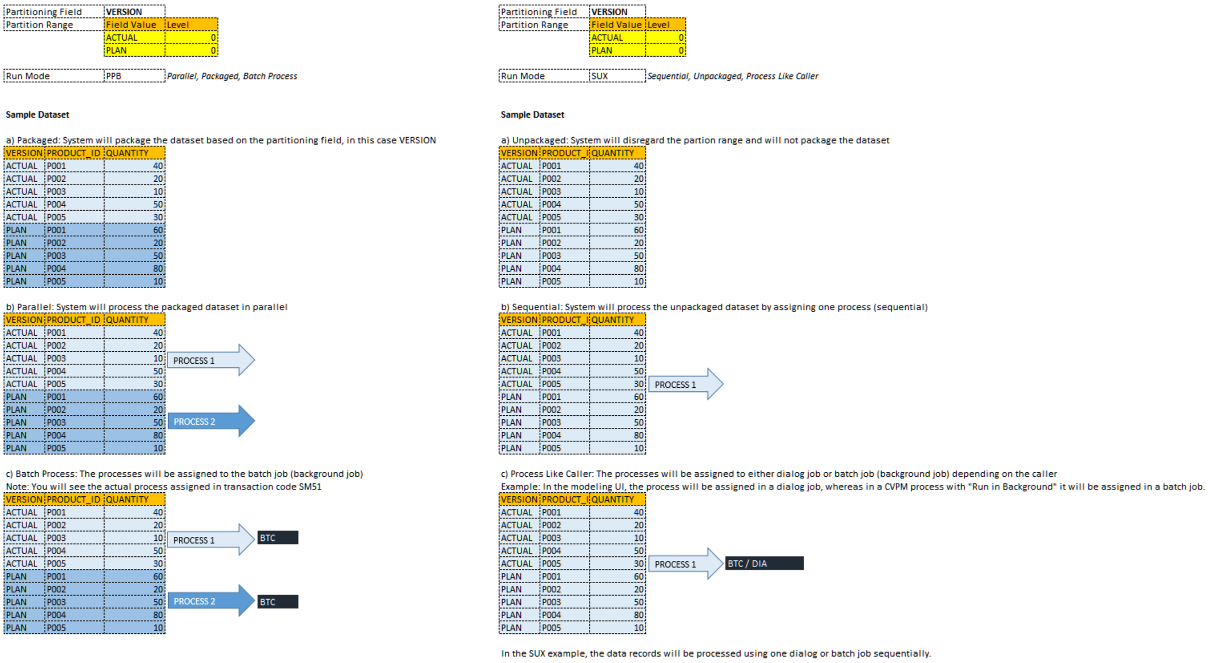
1. Parallel (P) or Sequential (S):
    1. “Parallel” means that the control returns immediately to the caller and does not wait for the function execution to be finished. The success of the execution is noted in the application log.
    2. “Sequential” means that the control returns to the caller only 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**	
Partition Range	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

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

Partitioning Field	**VERSION**	
Partition Range	Field Value	Level
	ACTUAL	0
	PLAN	0

Run Mode	SUX	*Sequential, Unpackaged, Process Like Caller*

Sample Dataset

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

VERSION	PRODUCT_I	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_I	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

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 CVPM process with "Run in Background" it will be assigned in a batch job.

VERSION	PRODUCT_I	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

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 ManagementFunctions [page 52].

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 52].

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 Transfert 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 23].
   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 155].

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 23].

Related Information

For more information about SAP Profitability and Performance Management functions, see Functions [page 52].

For more information about InfoAreas, see Environment [page 54].

For more information about InfoObjects, see Environment [page 54].

For more information about Data Store Objects (Advanced), see Model BW [page 147].

For more information about InfoCubes, see Model BW [page 147].

For more information about BW queries, see Query [page 152].

For more information about process chains, see Parallelization and Partitioning [page 41].

For more information about open ODS views, see Analytics Component [page 23].

For more information about HAP-based BW data transfer processes, see Model Writer [page 155].

For more information about characteristics-based authorization, see Roles and Authorizations [page 44].

For more information about BPC environments, see Environment [page 54].

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 52].

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 HANA view. The name of the SAP HANA 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 HANA view. The name of the SAP HANA 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 RDL
       Since model RDLs 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 52].

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_BECFM`	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_BECF	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_BECF	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/1812.001/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 52].

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	Function to perform direct and indirect allocations
Calculation	Function to perform mathematical formulas
Calculation Unit	Function to encapsulate a group of functions and make them reusable
Conversion	Function to perform currency and unit conversions

Key Feature	Use
Derivation	Function to perform if-then-else enrichments of data
Description	Function to describe processes and topics used for the documentation of models
Environment	Function to register all required fields and the connection to the database
File Adapter	Function to provide automated access to files
Funds Transfer Pricing	Function to perform funds and liquidity transfer pricing calculations
Join	Function to perform collections, joins, unions and lookups for separate data
Model Table	Function to provide read and write access to a local or remote data table
Model View	Function to provide read access to a local or remote data table or view
Model RDL	Function to provide read and write access to a local FRDP Results Data Layer
Model BW	Function to provide read and write access to a local BW InfoSource like Advanced DSOs
Writer	Function to store data in a model table, model RDL or model BW
Transfer Structure	Function to perform a transfer from accounting-based data to costing-based data (also called denormalization)
View	Function to project or aggregate data, including filtering options and formulas
Remote Function Adapter	Function to perform an ABAP-based remote function call (for example, a call to a remote FI-GL posting BAPI)
Valuation	Function to perform comprehensive calculations with different valuation methods (for example, discounting)
Flow Modeling	Function to provide calculation for the best-estimate cash flow (BECF).

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.1 Environment

An environment comprises the information and calculation details for a financial and business model. An environment is versioned and multiple versions of the same environment as well as multiple environments can be in production in parallel.

The following table explains the key features available:

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

Key Feature	Use
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 tpes 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

Key Feature	Use
	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.

Key Feature	Use
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 32].

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. It specifies which process templates with activities are provided for execution as well as which parameters and selection fields are relevant.

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

The following table explains the key features available:

Key Feature	Use
Process Templates and Activities	Process templates define what execution users can do to run models. 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 which 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 thus used to run processes in production. 4. Process Type Can be set to either “Simulation” or Production. 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 “Production”, 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 which 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

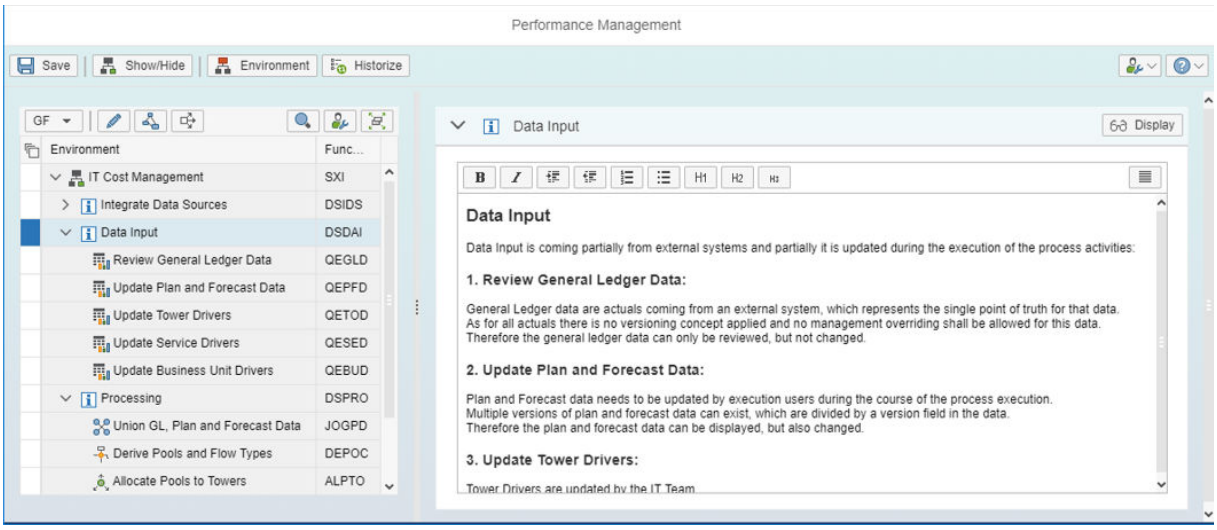
Key Feature	Use
	it with level 2 will remain pending until the activity *Review Actual Data* is finished. 9. Start Date and End Date In both fields, you need to define default values. These can be overwritten during process deployment. 10. Performer and Reviewer The performer defines a team (group of users) that can work on an activity. The reviewer can also define a team that has to review the activity in a workflow with dual control 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 to be used 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 to be used 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.

Related Information

For more information about common aspects ofSAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

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.



Environment	Func...
IT Cost Management	SXI
Integrate Data Sources	DSIDS
Data Input	DSDAI
Review General Ledger Data	QEGLD
Update Plan and Forecast Data	QEPFD
Update Tower Drivers	QETOD
Update Service Drivers	QESED
Update Business Unit Drivers	QEBUD
Processing	DSPRO
Union GL, Plan and Forecast Data	JOGPD
Derive Pools and Flow Types	DEPOC
Allocate Pools to Towers	ALPTO

Data Input

Data Input is coming partially from external systems and partially it is updated during the execution of the process activities:

1. Review General Ledger Data:

General Ledger data are actuals coming from an external system, which represents the single point of truth for that data.
As for all actuals there is no versioning concept applied and no management overriding shall be allowed for this data.
Therefore the general ledger data can only be reviewed, but not changed.

2. Update Plan and Forecast Data:

Plan and Forecast data needs to be updated by execution users during the course of the process execution.
Multiple versions of plan and forecast data can exist, which are divided by a version field in the data.
Therefore the plan and forecast data can be displayed, but also changed.

3. Update Tower Drivers:

Tower Drivers are updated by the IT Team

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.

The following table explains the key features available:

Key Feature	Use
Documentation Editor	The editor and its simple formatting options can be used to provide descriptions and documentation about the function configuration and why things where modeled this way.

Related Information

For more details about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

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 Feature	Use
Header	In the allocation header, you define the principal behavior of the allocation. You can choose between the following allocation types: • 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. • 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. • 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. Other specific options include iterative allocations which are explained below. • Iterative Indicator: You use the *Iterative* indicator to specify whether the allocation process is executed iteratively. If you select 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.

Key Feature	Use
	You define 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 Indicator: You use the *Periodic* indicator to specify whether the allocation process is executed based on a defined period or time interval. If you select periodic processing, the system runs the allocation process using the period/time interval defined in the allocation settings in the *Advanced* tab. In the advanced allocation settings, you can specify the fiscal year and period intervals. You can also choose whether periodic processing is cumulative. 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: ● 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. ● Last Row: The system adds the value difference to the last receiver (corresponding to the specified sender). ● Biggest Value Row: The difference in value is added to the receiver (corresponding to the specified sender) with the highest allocated amount. ● 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.
Sender, Receiver and Reference	On the *Sender* and *Receiver* tab, you define the input for the allocation function. 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. This means that receiver data from the current month can be used, but driver values from the last month, for example.

Key Feature	Use
Rules	Each allocation rule defines one segment of an allocation. For each allocation rule, the sender and receiver rule need to be specified. On the *Sender* tab the following options are available: • Sender Rule: ○ Posted Amounts: The sender input is used or the result of an allocation rule at a lower level. • 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. • Sender Value Fields: Defines the value fields that have to be allocated from the sender to the receiver. • Mapping Method: When senders are allocated to receivers in direct allocation, receivers are matched based on the mapping method chosen. ○ 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 only to any value in the characteristics of the receiver. In other words, characteristic values from the receiver are ignored when the characteristics are empty in the sender. • Subview: You can apply further selections, formulas and groupings here if needed. On the *Receiver* tab, the following options are available: • Receiver Rule: ○ 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:

Key Feature	Use
	The receiver input is used and no distribution base is needed because the sender is evenly distributed. ● 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. ● Distribution Base: Is the basis for the allocation. The specific treatment is dependent on the receiver rule (see above). ● Driver Result: If a field is entered 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. ● Subview: You can apply further selections, formulas and groupings if needed.

Key Feature	Use
	Further options comprise 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.

Key Feature	Use
Advanced	If iterative allocation is defined in the header, you need to make additional settings on the Advanced tab. The following settings are relevant for iterations: ● Cycle Maximum Value: The number of iterations is limited to the value entered (for example, 100). ● Iteration Counter: If you register a field in this optional setting, the result shows which records have been allocated in which iteration cycle. ● Early Exit Check: If you have registered a check in this optional setting, it will be 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. The following settings are relevant for periodic processing: ● Periodic Counter: The *Periodic Counter* field in the output provides information about the period for which a particular allocation record is created in periodic processing. Prerequisite: The field has been defined as an action field in the signature of the allocation. ● Fiscal Year: The *Fiscal Year* field contains financial year information in the sender and/or receiver data. Prerequisite: The field has been defined as a selection field in the signature of the allocation. ● Fiscal Year Value: Fiscal Year Value is the financial year for which the allocation is executed. The value specified here is used to restrict sender and receiver data for a given financial year. ● Period: The *Period* field contains term/timeframe information in the sender and/or receiver data. 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. ● Last Period Value:

Key Feature	Use
	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 different 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 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: First the senders from a given period are allocated 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 period before. If the system still finds no receivers in that period for some senders, the system 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. 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. You can specify two types of offset mappings: 1. Offset:

Key Feature	Use
	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.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

See also:

- Example: Indirect Allocation with Offset Records [page 69]
- Example: Direct Allocation with Unassigned Items [page 70]

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
−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

Product	Channel	Customer	Amount	Financial Period
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
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
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

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.

> **i 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 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.

The following table explains the key features available:

Key Feature	Use
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.
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 32].

See also:

- Example: Simple Condition [page 74]
- Example: Ensure Distinct Result [page 74]

1.3.5.1 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

Result

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

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

1.3.5.2 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

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

If Branch = B1 and Customer Type = LOYAL then Addittional Interest = 0.0500

If Customer Type = LOYAL and Deposit Amount is greater than 50000 then Additional Interest is 0.0250

In the first rule, loyal customers from B1 will already be derived. In the second rule, they will no longer be included in the derivation.

Additional Interest will be added respectively to the derived values and deposit with interest will be computed.

1.3.6 Join

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

The following table explains the key features available:

Key Feature	Use
Header	In the header, you define the principal behavior of the join. You can choose between different join types: ● Implicit Fields The fields of all inputs are automatically kept as much as possible to avoid destroying information. In case of a join rule, the field content is taken from the first input that contains the field. ● Explicit Fields Only the fields of inputs that are explicitly defined in the rules are kept. 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. You can control the behavior of joins with the Auto Filling option. The following settings are possible: ● No: The behavior is as described above. ● 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.

Key Feature	Use
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.

i 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.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

See also:

- Example: Full Outer Join – Autofilling Set to No [page 78]
- Example: If Null/Initial Then First to Last [page 80]
- Example: If Null Then First to Last [page 81]
- Example: Inner Join [page 82]
- Example: Left Outer Join [page 83]
- Example: Cross Join Implicit [page 84]
- Example: Union All [page 85]
- Example: Lookup Auto Predicates [page 86]

1.3.6.1 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 for RULE2 and RULE3.

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
P0005	M1012	10
P0005	M1011	30
	?	?
	?	?

Product Code	Product	Price	Unit
	?	?	?
	?	?	?
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 join of Rule 1 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`).

1.3.6.2 Full Outer Join Special Scenarios for Auto Filling field

1.3.6.2.1 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
PROD10	DE_CUST10	0	180

1.3.6.2.2 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	?

Product	Customer	Amount	Price
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

1.3.6.3 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

1.3.6.4 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 and P0003/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

1.3.6.5 Example: Cross Join Implicit

Input Tables

Material Table

Material Code	Material	Cost per Order	Unit
M1011	Leather	10	USD
M1012	Thread	5	USD

Product Table

Product Code	Product
P0001	Shoe
P0002	Watch

Order Table

Branch	Order
BR001	10
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.

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

Material Code	Material	Cost per Order	Unit	Product Code	Product	Branch	Order
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 first table multiplied by the number of rows in the second 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*.

1.3.6.6 Example: Union All

Input Tables

JO - Material Table

Material Code	Material	Cost per Order	Currency
M1011	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
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

1.3.6.7 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
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.

> i 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	
P0004	Shorts	20	EUR		0	
P0005				M1012	10	Thread
P0006				M1011	30	Leather

1.3.7 Funds Transfer Pricing

The Funds Transfer Pricing function provides a variety of different rule types to calculate liquidity and funding component rates as well as funds and liquidity costs.

The following table explains the key features available:

Key Feature	Use
Rules	Funds transfer pricing calculations usually consist of several financial product-specific steps. Each rule represents one of these steps in the configuration. For example, for commercial loans the flow generation based on financial conditions, rate modeling that applies interest calculations based on variable interest conditions, then matched maturity calculations for various FTP / LTP components, like base rate plus liquidity premium, and lastly the calculation of funding costs. Therefore hierarchical rules are supported by assigning higher levels. The hierarchy of levels is resolved starting with the lowest level, feeding as an input to the higher levels and ending with the highest level. Most of the rule types have line types underneath that provide further options. The following rule types and line types are available: 1. Characteristic Formula: Application of Formulas and SQL Functions to Characteristics 2. Conversion 1. Currency Conversion: 2. Unit Conversion 3. Duration: 1. Macaulay Duration 2. Fisher-Weil Duration 3. Modified Duration 4. Flow Generation 1. Single Flow 2. Periodic Fixed Amount Flow 3. Periodic Fixed Even Flow 4. Periodic Fixed Rate Flow 5. Periodic Fixed Value Flow 6. Single Residual Flow 5. Flow Merge 1. Include Events to Flows 6. Series Generation 7. Key Figure Formula 8. Market Interest Rate 1. Market Interest Rate

Key Feature	Use
	2. Effective Capital 3. Capital Growth 4. Effective Interest 5. Net Present Value 9. Matched Maturity 10. Rate Modeling 1. Date to Term 2. Lookup Rate by Interpolation 3. Periodic Fixed Interest 4. Periodic Variable Interest 11. Running Total 12. Strip Funding 13. Weighted Average Rate
Sub View	For each rule, you can define further selections, aggregations and sorting orders.

Example

See SAP Note 2614017 - Sample Content for Funds Transfer Pricing

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.8 Valuation

The Valuation function provides a variety of different rule types to calculate valuations like discounting and value aggregations.

The following table explains the key features available:

Key Feature	Use
Rules	Valuation calculations usually consist of several product-specific or service-specific steps. Each rule represents one of these steps in the configuration. For example, for commercial insurance contracts discounting is carried out at cash flow level and value aggregation calculates the statistical median. Therefore, you can use hierarchical rules by assigning higher levels. The hierarchy of levels is resolved starting with the lowest level, which is used as input to the higher levels and ends with the highest level. Most of the rule types have line types underneath that provide further options. The following rule types and line types are available: 1. Duration 1. Macaulay Duration 2. Fisher-Weil Duration 3. Modified Duration 2. Discounting 3. Interpolation 1. Extrapolation None 2. Extrapolation Linear 3. Extrapolation Constant 4. Running Total 5. Value Aggregation 1. Number of Rows 2. Minimum Value 3. Statistical Median 4. Maximum Value 5. Arithmetical Mean 6. Standard Deviation Square Root of Variance 7. Standard Deviation Square Root of Proportional Variance 8. Standard Deviation of Sample Variance 9. Variance Value 10. Population Variance Value 11. Sample Variance Value 6. Line Item Valuations 1. Balance

Key Feature	Use
	2. Formula 3. Lag 4. Lead 5. Running Balance 6. Register 7. Scaled Weighted Average
Rule Lines	• Balance Granularity Fields: Tells the rule and line type the granularity level on which the line items exist in your data. The granularity is the event ID, but it could also be a business transaction ID or a combination, such as document ID and line item ID. The effect is that line item valuation calculates sequentially at this level of granularity to ensure correct results. • Selection: This defines what needs to be calculated and when. You define what needs to be calculated by choosing a specific line type, for example “Balance” or “Scaled Weighted Average”. In the selection, you therefore need to specify when this has to be calculated. • Factor: This optional factor allows you to manipulate the calculation, if required. For example, if both inflows and outflows are positive in your data, your financial position or inventory grows infinitely. You can use this factor to avoid this, for example, by defining “-1” as a factor in case of outflows. You can also apply a formula here, for example, CASE WHEN DEBIT_CREDIT_INDICATOR = ‘X’ THEN -1 ELSE 1 END. • Value: This mandatory input field defines the value field in your data records that carries the delta inflow or outflow value and influences the position or inventory you want to valuate. • Quantity: This field is only available for the “Scaled Weighted Average” line type: The quantity is multiplied by the entered value to get to a correct balance total. For example, a quantity of 5 pieces with a value of EUR 10 results in 5*10 = EUR 50, or a financial position of 100 shares with an acquisition cost of 10 USD each results in 100*10 = 1000 USD. • Result: In this mandatory field, you define the field in which the system enters the result in your data records.

Key Feature	Use
Sub View	For each rule, you can define further selections, aggregations and sorting orders.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.9 Flow Modeling

The Flow Modeling function provides a variety of different rule types to calculate the best-estimate cash flow (BECF).

The following table explains the key features available:

Key Feature	Use
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 are available: 1. Characteristic Formula: Applies formulas and SQL functions to characteristics. 2. Key Figure Formula: Applies formulas and SQL functions to key figures. 3. 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. 4. Series Generation: Generates series data by providing several parameters like step size, series type, period from and period to. . 5. 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). 6. 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. 7. 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 the original pattern structure. 8. 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 which are of balance type (cumula-

Key Feature	Use
	tive factor/amount values) or movement type (delta factors/amounts). 9. 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) 10. Incremental Value Calculation: Distributes factor 2 (due factor) of one period to incurred periods by using factor 1 (incurred factor pattern) as a distribution key. The distribution key is adjusted by factors allocated in previous periods. 11. Redistribution: Calculates estimate values prior to the Reference Date for Redistribution (RDR) and redistributes them to the future periods after the Reference Date for Redistribution (RDR). 12. Scale Factor: Ratio of two corresponding values with similar field/data types (division). 13. Scaling: Applies a scale factor to the actuarial model stream (multiplication). 14. Acknowledge Actuals (Acknowledgement of Cedent Data): Enriches the actuals by determining the missing earlier life cycle date information by applying matching logic, and also determines the regime for every cashflow item. 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: 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.

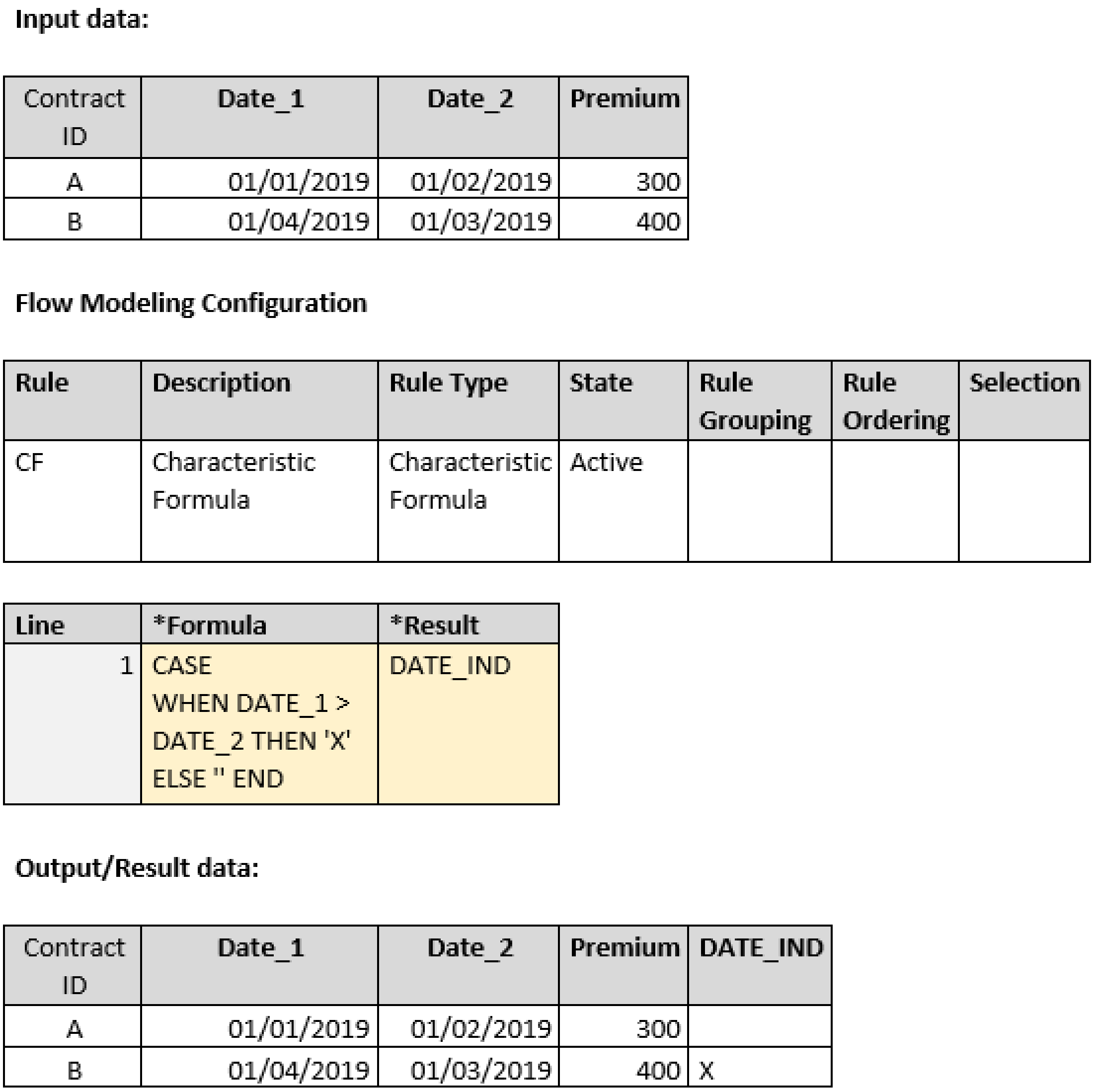
Key Feature	Use
	16. Modulation Out: ○ Selects amounts in the basis cash flow where the exposure date > coverage end date. ○ If Modulation Out for contract T is updated and contract T+1 exists, Modulation In for contract T+1 must be (re-)calculated. 17. Modulation In: ○ New contract: Modulation In for 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. 18. 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 ise filled by this rule type). 19. Cashflow Regime: 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 applythe formula (Amount * Factor / (1 - 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.

Key Feature	Use
	20. Clear Actual Dates Clears the actual date information in the cashflows arising from actuals. 21. 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 lag and Primary Insurer to Reinsurer lag. 22. Factor for Additional Incurred 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 32].

1.3.9.1 Example: Characteristic Formula



Input data:

Contract ID	Date_1	Date_2	Premium
A	01/01/2019	01/02/2019	300
B	01/04/2019	01/03/2019	400

Flow Modeling Configuration

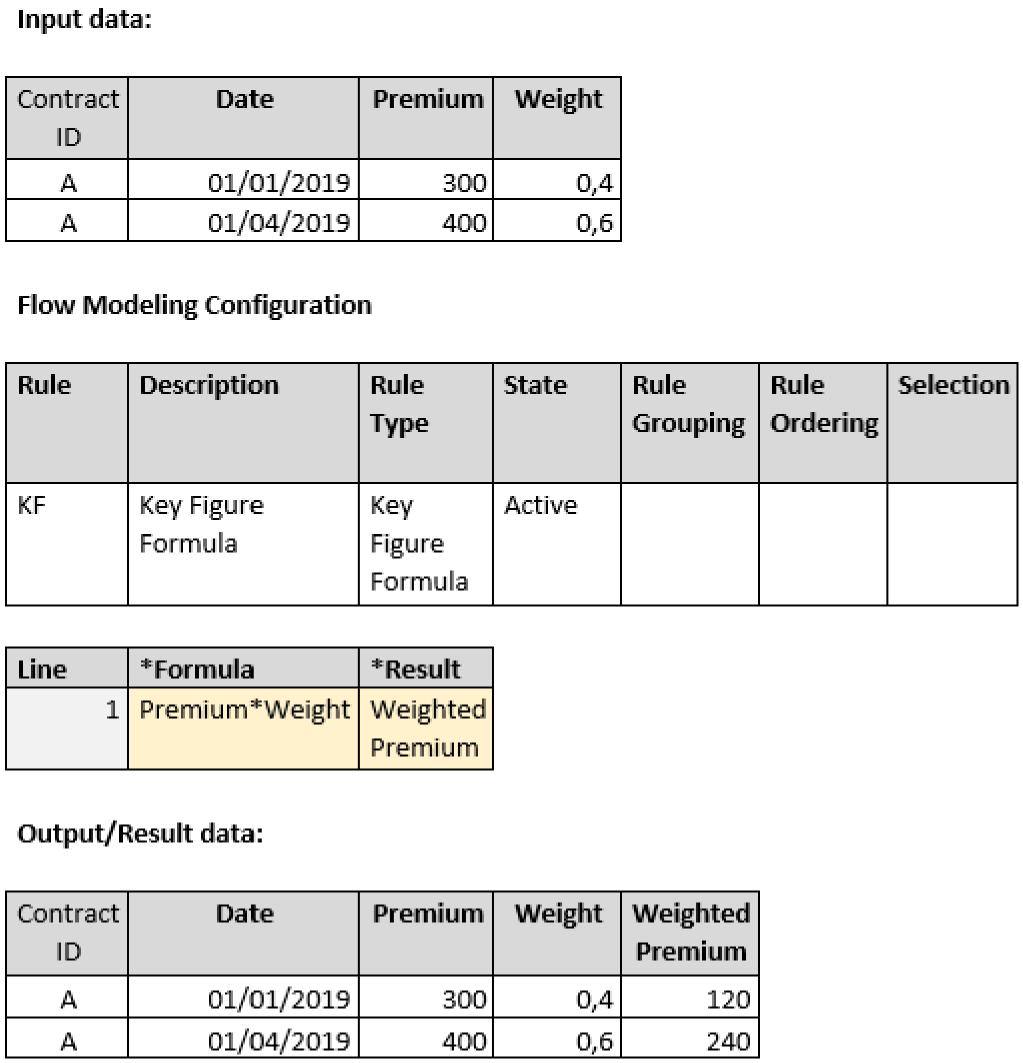
Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
CF	Characteristic Formula	Characteristic Formula	Active			

Line	*Formula	*Result
1	CASE WHEN DATE_1 > DATE_2 THEN 'X' ELSE '' END	DATE_IND

Output/Result data:

Contract ID	Date_1	Date_2	Premium	DATE_IND
A	01/01/2019	01/02/2019	300	
B	01/04/2019	01/03/2019	400	X

1.3.9.2 Example: Key Figure Formula



Input data:

Contract ID	Date	Premium	Weight
A	01/01/2019	300	0,4
A	01/04/2019	400	0,6

Flow Modeling Configuration

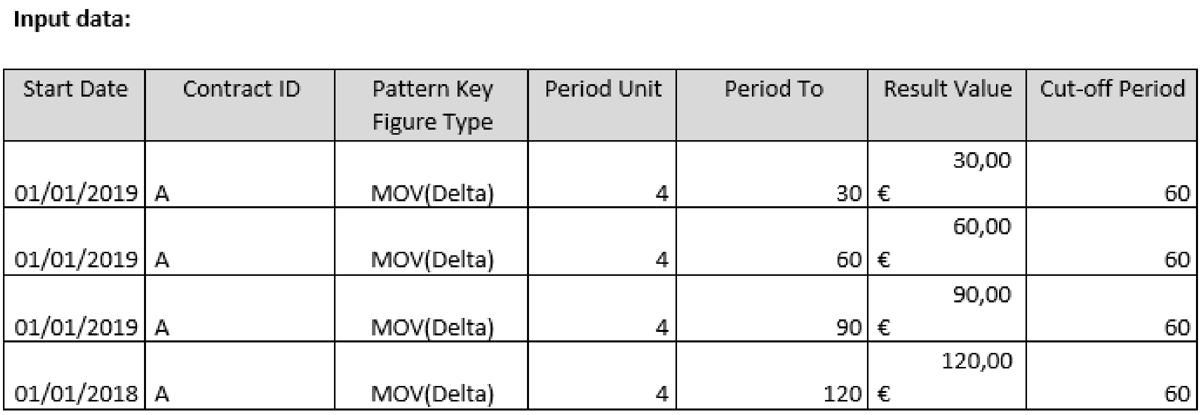
Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
KF	Key Figure Formula	Key Figure Formula	Active			

Line	*Formula	*Result
1	Premium*Weight	Weighted Premium

Output/Result data:

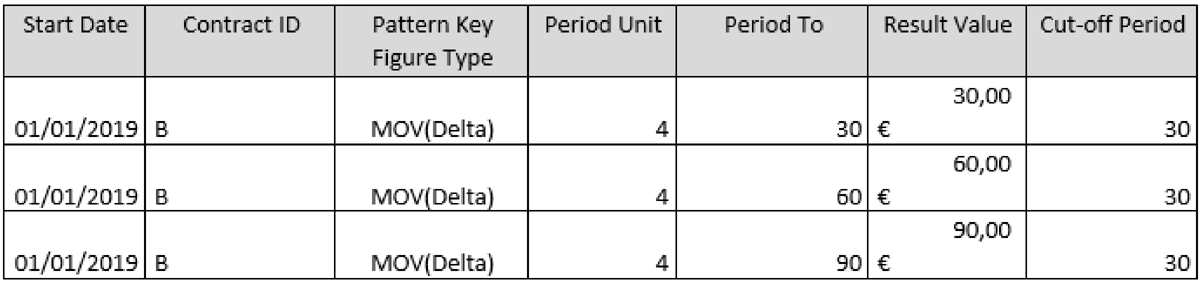
Contract ID	Date	Premium	Weight	Weighted Premium
A	01/01/2019	300	0,4	120
A	01/04/2019	400	0,6	240

1.3.9.3 Example: Flow Cut-Off

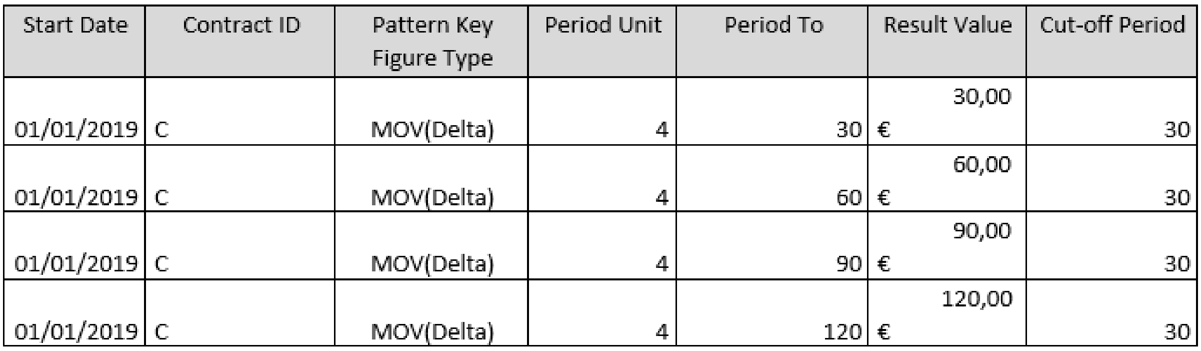


Input data:

Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	Result Value	Cut-off Period
01/01/2019	A	MOV(Delta)	4	30	30,00 €	60
01/01/2019	A	MOV(Delta)	4	60	60,00 €	60
01/01/2019	A	MOV(Delta)	4	90	90,00 €	60
01/01/2018	A	MOV(Delta)	4	120	120,00 €	60

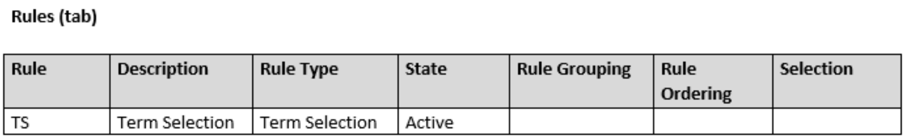


Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	Result Value	Cut-off Period
01/01/2019	B	MOV(Delta)	4	30	30,00 €	30
01/01/2019	B	MOV(Delta)	4	60	60,00 €	30
01/01/2019	B	MOV(Delta)	4	90	90,00 €	30



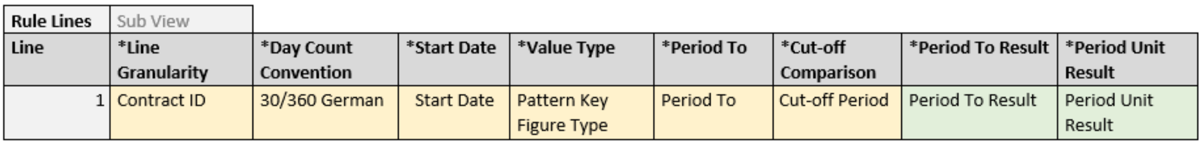
Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	Result Value	Cut-off Period
01/01/2019	C	MOV(Delta)	4	30	30,00 €	30
01/01/2019	C	MOV(Delta)	4	60	60,00 €	30
01/01/2019	C	MOV(Delta)	4	90	90,00 €	30
01/01/2019	C	MOV(Delta)	4	120	120,00 €	30

Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TS	Term Selection	Term Selection	Active			



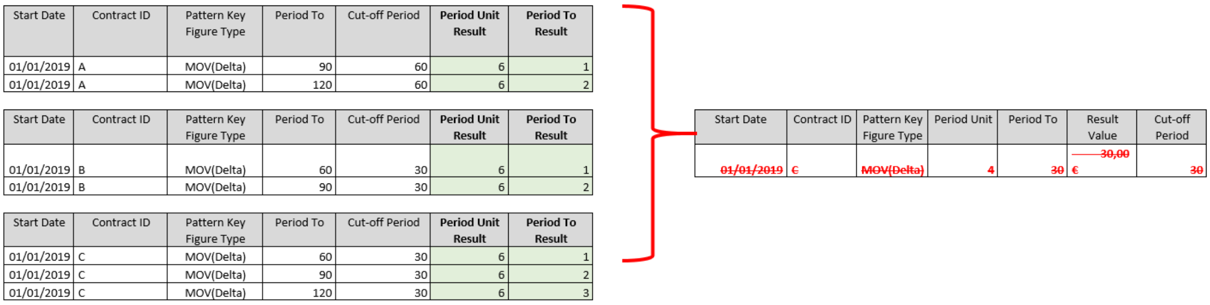
Rule Lines	Sub View							
Line	*Line Granularity	*Day Count Convention	*Start Date	*Value Type	*Period To	*Cut-off Comparison	*Period To Result	*Period Unit Result
1	Contract ID	30/360 German	Start Date	Pattern Key Figure Type	Period To	Cut-off Period	Period To Result	Period Unit Result

i Note

- Yellow marked fields are input fields whereas green marked fields are the output fields.
- Cut-off Comparison = Key field to identify which cash flow items need to be deleted (Rule in case of delta items: Cut all periods <= Cut-off period. In case of balance itsm the cut-off period is perceived as new T-0 period).
- Day Count Convention = Day count convention used to determine the result periods.
- Value Type = Input field containing either the value “MOV” or “BAL” to determine of which type the cashflow item is made of. The user has to make sure that the field contains these two values only.

Output/Result Data

The following tables show the result after applying the rule type where all cash flow items prior (or equal) to the cut-off period got deleted.



Start Date	Contract ID	Pattern Key Figure Type	Period To	Cut-off Period	Period Unit Result	Period To Result
01/01/2019	A	MOV(Delta)	90	60	6	1
01/01/2019	A	MOV(Delta)	120	60	6	2

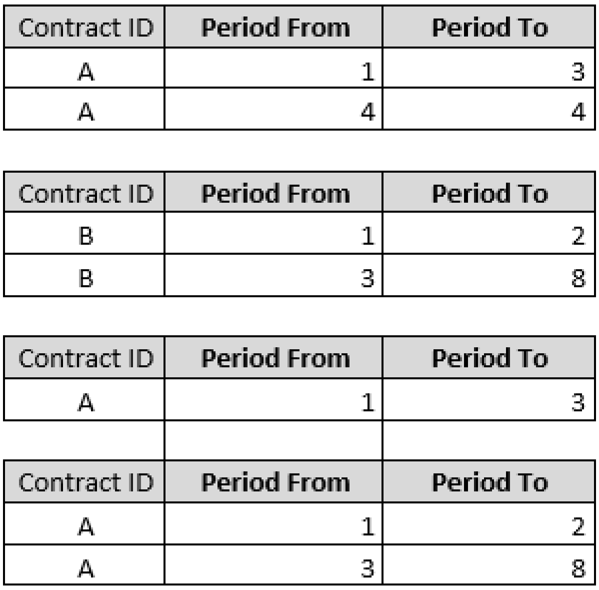
Start Date	Contract ID	Pattern Key Figure Type	Period To	Cut-off Period	Period Unit Result	Period To Result
01/01/2019	B	MOV(Delta)	60	30	6	1
01/01/2019	B	MOV(Delta)	90	30	6	2

Start Date	Contract ID	Pattern Key Figure Type	Period To	Cut-off Period	Period Unit Result	Period To Result
01/01/2019	C	MOV(Delta)	60	30	6	1
01/01/2019	C	MOV(Delta)	90	30	6	2
01/01/2019	C	MOV(Delta)	120	30	6	3

Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	Result Value	Cut-off Period
~~01/01/2019~~	~~C~~	~~MOV(Delta)~~	4	~~30~~	~~30,00~~ €	~~30~~

1.3.9.4 Example: Series Generation

Input data:



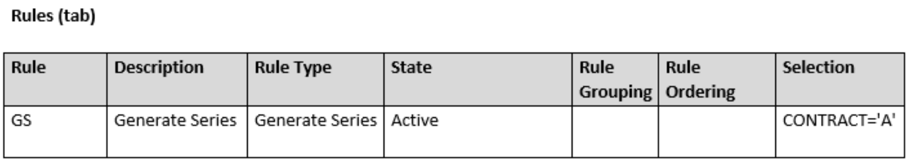
Contract ID	Period From	Period To
A	1	3
A	4	4

Contract ID	Period From	Period To
B	1	2
B	3	8

Contract ID	Period From	Period To
A	1	3

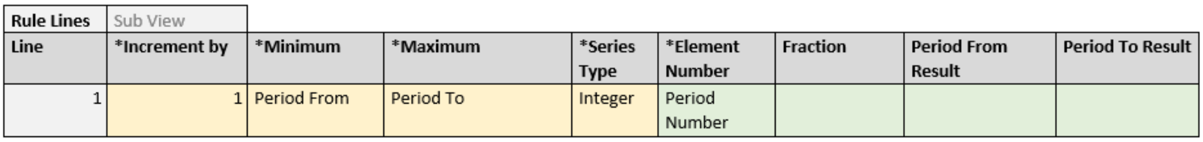
Contract ID	Period From	Period To
A	1	2
A	3	8

Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
GS	Generate Series	Generate Series	Active			CONTRACT='A'

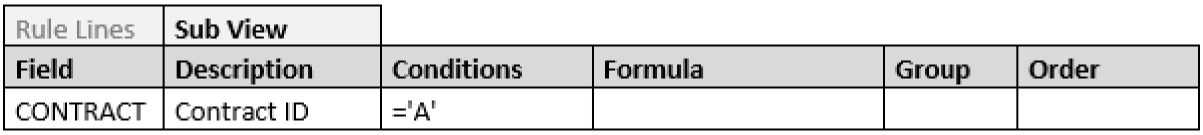


Rule Lines | Sub View

Line	*Increment by	*Minimum	*Maximum	*Series Type	*Element Number	Fraction	Period From Result	Period To Result
1	1	Period From	Period To	Integer	Period Number			

i Note

- Yellow marked fields are input fields whereas green marked fields are the output fields.
- Increment by = Step size
- Series Type = determines the type of series that should be created (e.g. integer, date)



Rule Lines | Sub View

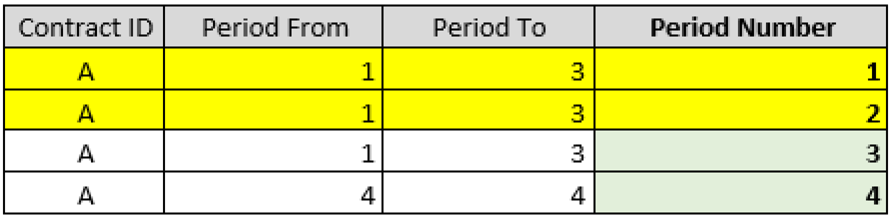
Field	Description	Conditions	Formula	Group	Order
CONTRACT	Contract ID	='A'			

i Note

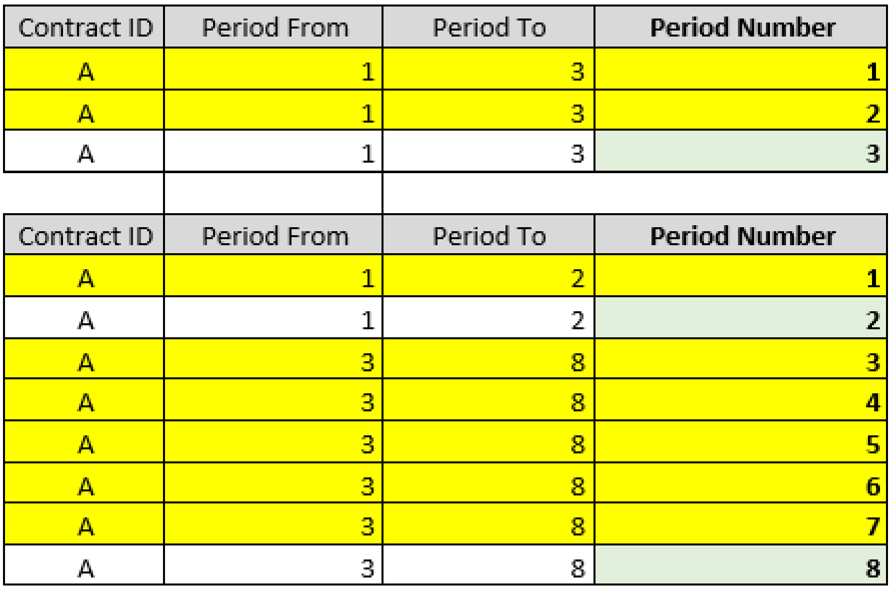
Condition = Filter options that select only cash flow items containing filter option values. In this example the rule is computing only for cashflow items which have a value of 'A' in column Contract ID.

Output/Result Data

The result tables show the generated series elements in yellow. The rule type only considers patterns with Contract ID = “A” because of the selection entered in column *Conditions* within the tab *Sub View*.



Contract ID	Period From	Period To	Period Number
A	1	3	1
A	1	3	2
A	1	3	3
A	4	4	4

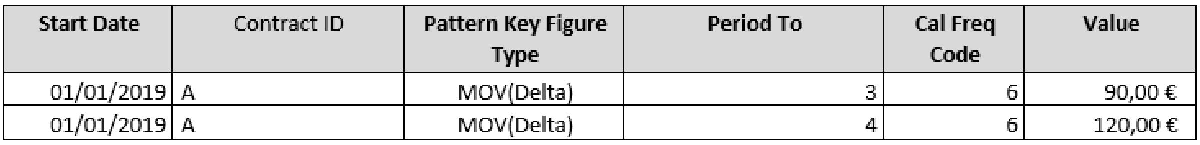


Contract ID	Period From	Period To	Period Number
A	1	3	1
A	1	3	2
A	1	3	3

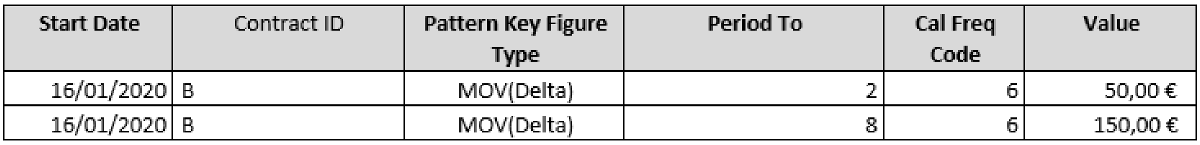
Contract ID	Period From	Period To	Period Number
A	1	2	1
A	1	2	2
A	3	8	3
A	3	8	4
A	3	8	5
A	3	8	6
A	3	8	7
A	3	8	8

1.3.9.5 Example: Term Conversion

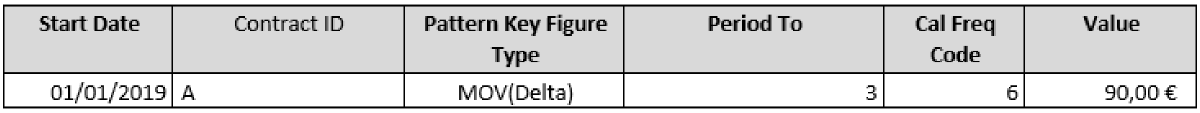
Input data:



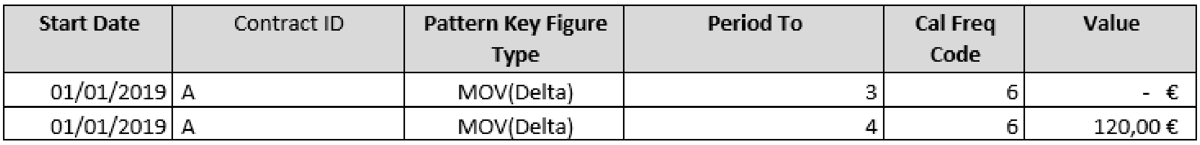
Start Date	Contract ID	Pattern Key Figure Type	Period To	Cal Freq Code	Value
01/01/2019	A	MOV(Delta)	3	6	90,00 €
01/01/2019	A	MOV(Delta)	4	6	120,00 €



Start Date	Contract ID	Pattern Key Figure Type	Period To	Cal Freq Code	Value
16/01/2020	B	MOV(Delta)	2	6	50,00 €
16/01/2020	B	MOV(Delta)	8	6	150,00 €

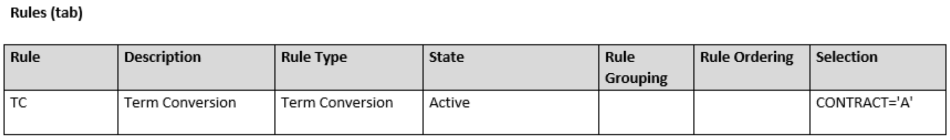


Start Date	Contract ID	Pattern Key Figure Type	Period To	Cal Freq Code	Value
01/01/2019	A	MOV(Delta)	3	6	90,00 €



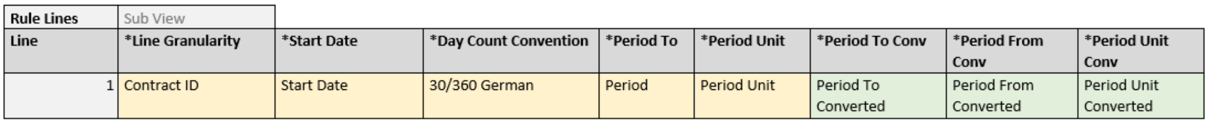
Start Date	Contract ID	Pattern Key Figure Type	Period To	Cal Freq Code	Value
01/01/2019	A	MOV(Delta)	3	6	- €
01/01/2019	A	MOV(Delta)	4	6	120,00 €

Flow Modeling Configuration



Rules (tab)

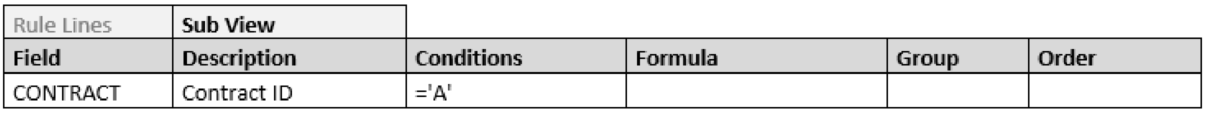
Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TC	Term Conversion	Term Conversion	Active			CONTRACT='A'



Rule Lines	Sub View							
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	30/360 German	Period	Period Unit	Period To Converted	Period From Converted	Period Unit Converted

i Note

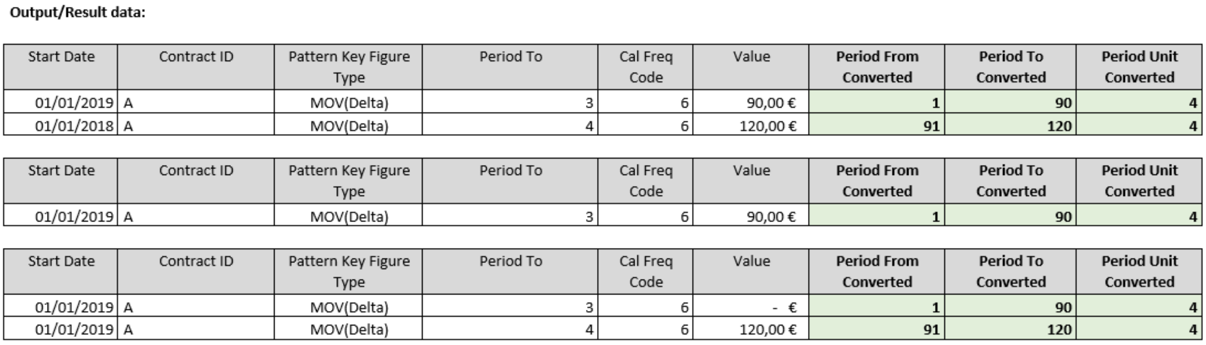
- Yellow marked fields are input fields whereas green marked fields are the output fields.
- Line Granularity = Determines the size on one partition of data.
- Day Count Convention = Day count convention used to determine the result periods.
- Period Unit = Period type of input data (e.g. monthly, quarterly)



Rule Lines	Sub View				
Field	Description	Conditions	Formula	Group	Order
CONTRACT	Contract ID	='A'			

i Note

Condition = Filter options that select only cash flow items containing filter option values. In this example the rule is computing only for cashflow items which have an “A” entry in column *Contract ID*.



Output/Result data:

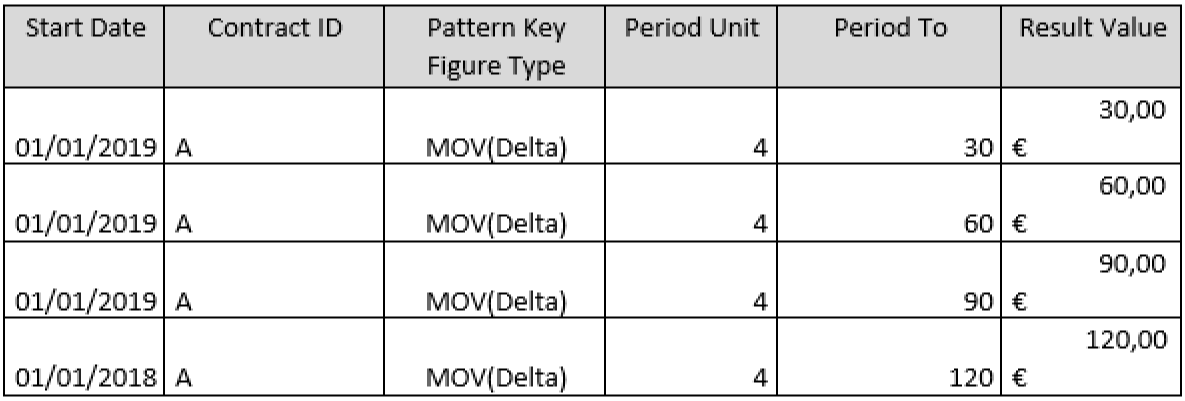
Start Date	Contract ID	Pattern Key Figure Type	Period To	Cal Freq Code	Value	Period From Converted	Period To Converted	Period Unit Converted
01/01/2019	A	MOV(Delta)	3	6	90,00 €	1	90	4
01/01/2018	A	MOV(Delta)	4	6	120,00 €	91	120	4

Start Date	Contract ID	Pattern Key Figure Type	Period To	Cal Freq Code	Value	Period From Converted	Period To Converted	Period Unit Converted
01/01/2019	A	MOV(Delta)	3	6	90,00 €	1	90	4

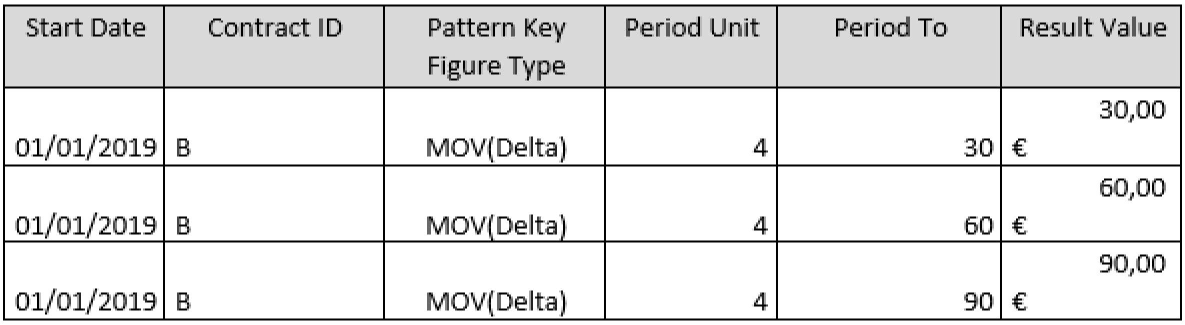
Start Date	Contract ID	Pattern Key Figure Type	Period To	Cal Freq Code	Value	Period From Converted	Period To Converted	Period Unit Converted
01/01/2019	A	MOV(Delta)	3	6	- €	1	90	4
01/01/2019	A	MOV(Delta)	4	6	120,00 €	91	120	4

1.3.9.6 Example: Term Selection

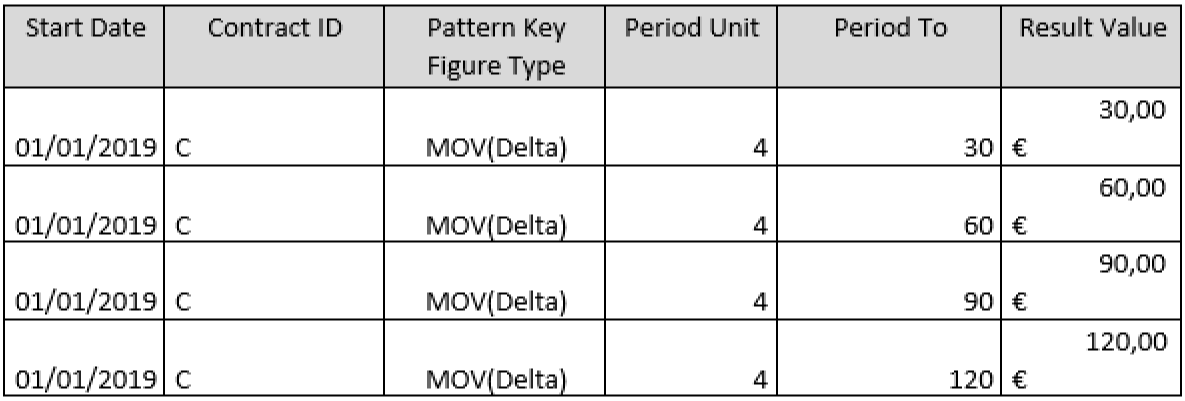
Input data:



Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	Result Value
01/01/2019	A	MOV(Delta)	4	30	30,00 €
01/01/2019	A	MOV(Delta)	4	60	60,00 €
01/01/2019	A	MOV(Delta)	4	90	90,00 €
01/01/2018	A	MOV(Delta)	4	120	120,00 €

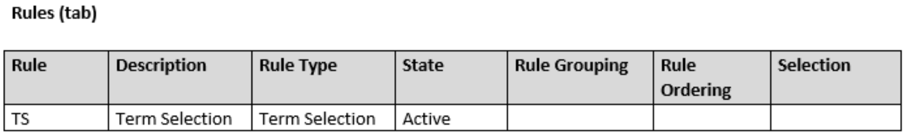


Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	Result Value
01/01/2019	B	MOV(Delta)	4	30	30,00 €
01/01/2019	B	MOV(Delta)	4	60	60,00 €
01/01/2019	B	MOV(Delta)	4	90	90,00 €



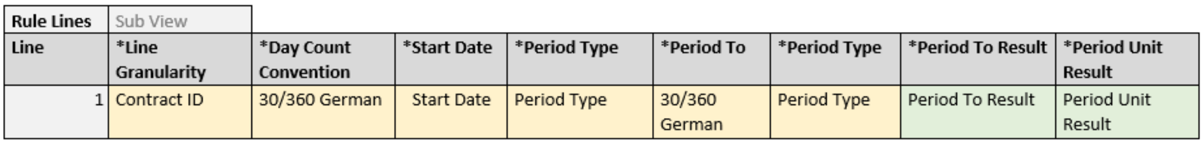
Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	Result Value
01/01/2019	C	MOV(Delta)	4	30	30,00 €
01/01/2019	C	MOV(Delta)	4	60	60,00 €
01/01/2019	C	MOV(Delta)	4	90	90,00 €
01/01/2019	C	MOV(Delta)	4	120	120,00 €

Flow Modeling Configuration



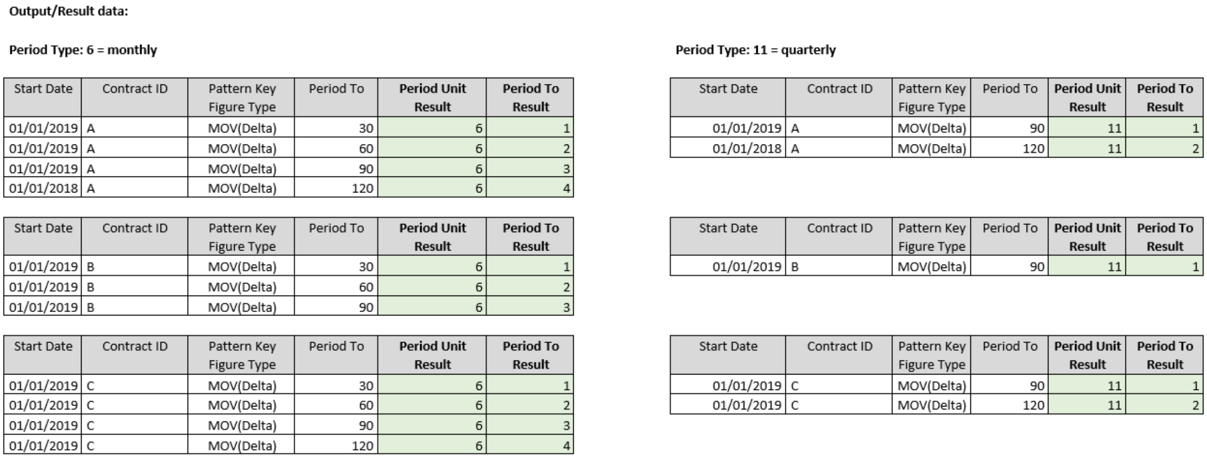
Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TS	Term Selection	Term Selection	Active			



Rule Lines	Sub View							
Line	*Line Granularity	*Day Count Convention	*Start Date	*Period Type	*Period To	*Period Type	*Period To Result	*Period Unit Result
1	Contract ID	30/360 German	Start Date	Period Type	30/360 German	Period Type	Period To Result	Period Unit Result

> i Note
>
> - Yellow marked fields are input fields whereas green marked fields are the output fields.
> - Line Granularity = Determines the size on one partition of data.
> - Period Type = Period type that should be used to determine the final pattern structure (e.g. monthly, quarterly)
> - Day Count Convention = Day count convention used to determine the result periods.



Output/Result data:

Period Type: 6 = monthly

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
01/01/2019	A	MOV(Delta)	30	6	1
01/01/2019	A	MOV(Delta)	60	6	2
01/01/2019	A	MOV(Delta)	90	6	3
01/01/2018	A	MOV(Delta)	120	6	4

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
01/01/2019	B	MOV(Delta)	30	6	1
01/01/2019	B	MOV(Delta)	60	6	2
01/01/2019	B	MOV(Delta)	90	6	3

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
01/01/2019	C	MOV(Delta)	30	6	1
01/01/2019	C	MOV(Delta)	60	6	2
01/01/2019	C	MOV(Delta)	90	6	3
01/01/2019	C	MOV(Delta)	120	6	4

Period Type: 11 = quarterly

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
01/01/2019	A	MOV(Delta)	90	11	1
01/01/2018	A	MOV(Delta)	120	11	2

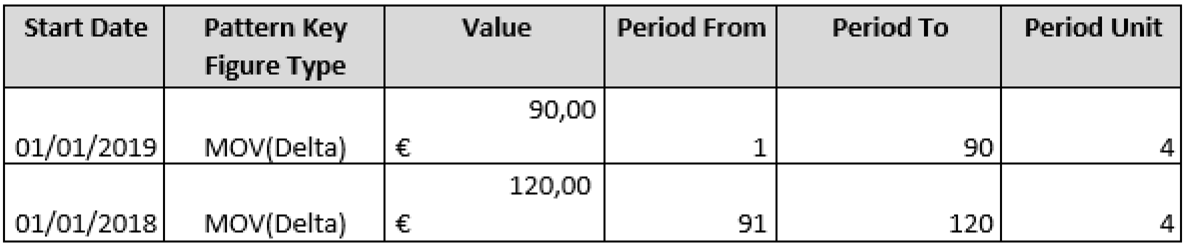
Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
01/01/2019	B	MOV(Delta)	90	11	1

Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Unit Result	Period To Result
01/01/2019	C	MOV(Delta)	90	11	1
01/01/2019	C	MOV(Delta)	120	11	2

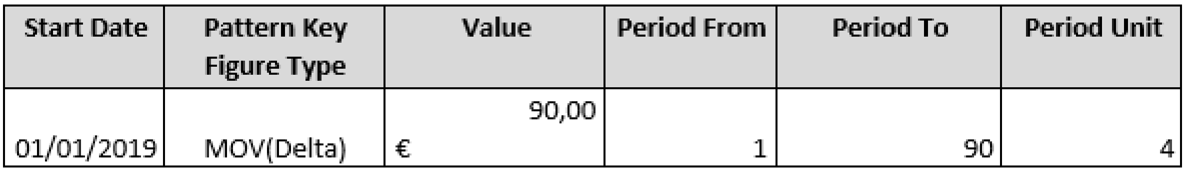
1.3.9.7 Example: Term Target

Input Data

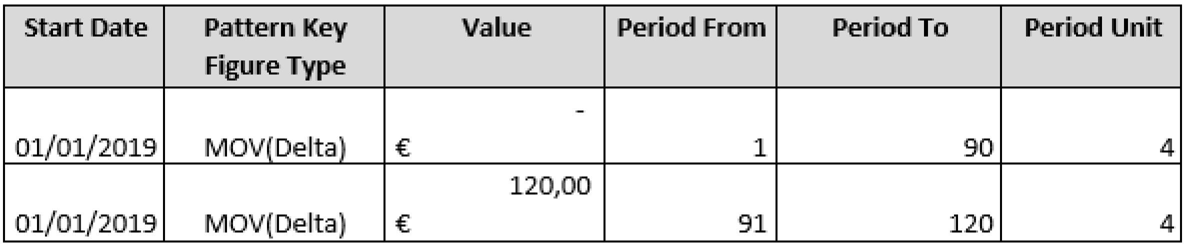
The following tables show 3 different patterns where all items are movements (delta value).



Start Date	Pattern Key Figure Type	Value	Period From	Period To	Period Unit
01/01/2019	MOV(Delta)	€ 90,00	1	90	4
01/01/2018	MOV(Delta)	€ 120,00	91	120	4

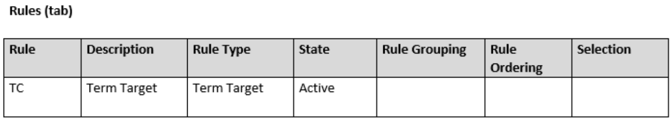


Start Date	Pattern Key Figure Type	Value	Period From	Period To	Period Unit
01/01/2019	MOV(Delta)	€ 90,00	1	90	4



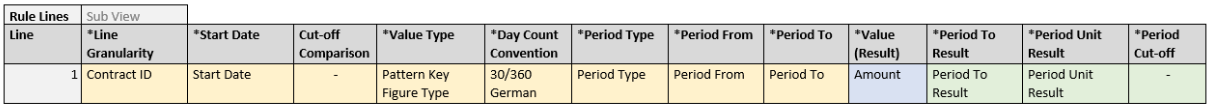
Start Date	Pattern Key Figure Type	Value	Period From	Period To	Period Unit
01/01/2019	MOV(Delta)	€ -	1	90	4
01/01/2019	MOV(Delta)	€ 120,00	91	120	4

Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TC	Term Target	Term Target	Active			



Rule Lines | Sub View

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 Key Figure Type	30/360 German	Period Type	Period From	Period To	Amount	Period To Result	Period Unit Result	-

> **Note**
>
> - Yellow marked fields are input fields whereas green marked fields are the output fields.
> - Line Granularity = determines the size on one partition of data.
> - Value Type = Input field containing either the value “MOV” or “BAL” to determine of which type the cashflow item is made of. The user has to make sure that the field contains these two values only.
> - Period Type = Period type that should be used to determine the final pattern structure (e.g. monthly, quarterly)
> - Day Count Convention = Day count convention used to determine the result periods.
> - Cut-off comparison = Cut-off date (preparational step: all items prior or equal to that date will be deleted later in case that rule type flow cut-off is applied, too)

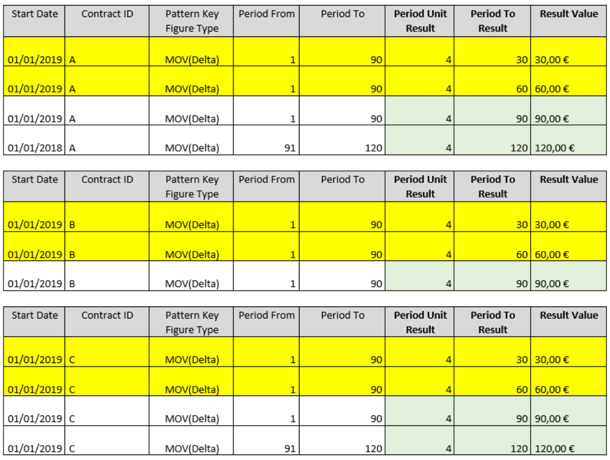
Output /Result Data

The output contains only entries matching the period type given in the rule type configurations, e.g. monthly or quarterly.

With respect to the pattern with periodicity of months the yellow marked items represent the additional pattern items created by the rule type containing the missing period values (30 and 60) and their corresponding interpolated amounts.

The pattern based on a periodicity of quarters shows also the period of day 120 instead of only having period 90 for quarter 1. This entry represents quarter 2, respectively, the end of the pattern.

Period Type: 6 = monthly

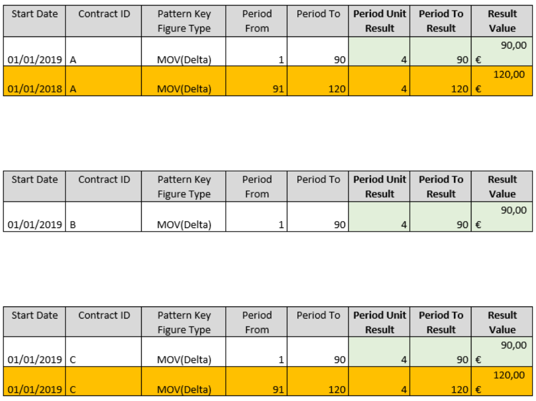


Start Date	Contract ID	Pattern Key Figure Type	Period From	Period To	**Period Unit Result**	**Period To Result**	**Result Value**
01/01/2019	A	MOV(Delta)	1	90	4	30	30,00 €
01/01/2019	A	MOV(Delta)	1	90	4	60	60,00 €
01/01/2019	A	MOV(Delta)	1	90	4	90	90,00 €
01/01/2018	A	MOV(Delta)	91	120	4	120	120,00 €

Start Date	Contract ID	Pattern Key Figure Type	Period From	Period To	**Period Unit Result**	**Period To Result**	**Result Value**
01/01/2019	B	MOV(Delta)	1	90	4	30	30,00 €
01/01/2019	B	MOV(Delta)	1	90	4	60	60,00 €
01/01/2019	B	MOV(Delta)	1	90	4	90	90,00 €

Start Date	Contract ID	Pattern Key Figure Type	Period From	Period To	**Period Unit Result**	**Period To Result**	**Result Value**
01/01/2019	C	MOV(Delta)	1	90	4	30	30,00 €
01/01/2019	C	MOV(Delta)	1	90	4	60	60,00 €
01/01/2019	C	MOV(Delta)	1	90	4	90	90,00 €
01/01/2019	C	MOV(Delta)	91	120	4	120	120,00 €

Period Type: 11 = quarterly



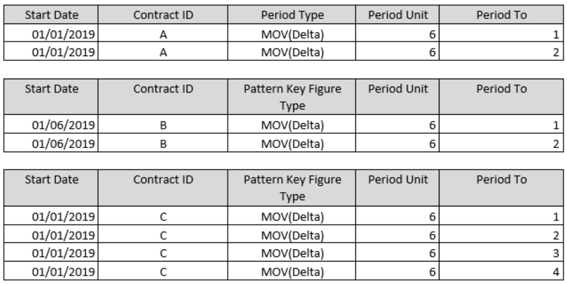
Start Date	Contract ID	Pattern Key Figure Type	Period From	Period To	**Period Unit Result**	**Period To Result**	**Result Value**
01/01/2019	A	MOV(Delta)	1	90	4	90	90,00 €
01/01/2018	A	MOV(Delta)	91	120	4	120	120,00 €

Start Date	Contract ID	Pattern Key Figure Type	Period From	Period To	**Period Unit Result**	**Period To Result**	**Result Value**
01/01/2019	B	MOV(Delta)	1	90	4	90	90,00 €

Start Date	Contract ID	Pattern Key Figure Type	Period From	Period To	**Period Unit Result**	**Period To Result**	**Result Value**
01/01/2019	C	MOV(Delta)	1	90	4	90	90,00 €
01/01/2019	C	MOV(Delta)	91	120	4	120	120,00 €

1.3.9.8 Example: Term To Date

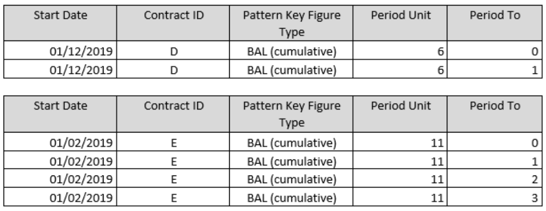
Input data:



Start Date	Contract ID	Period Type	Period Unit	Period To
01/01/2019	A	MOV(Delta)	6	1
01/01/2019	A	MOV(Delta)	6	2

Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To
01/06/2019	B	MOV(Delta)	6	1
01/06/2019	B	MOV(Delta)	6	2

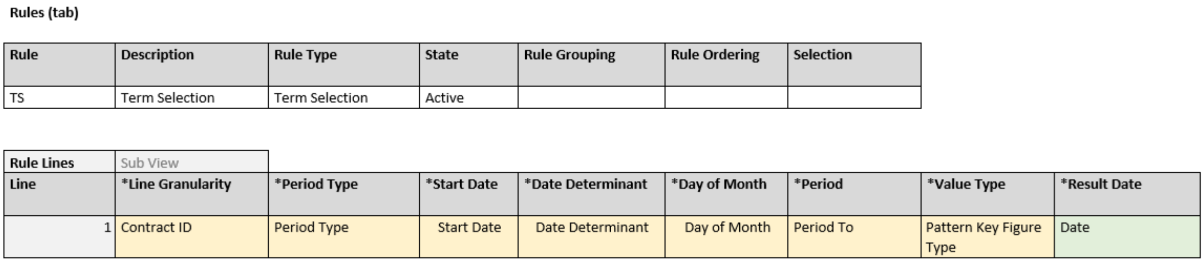
Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To
01/01/2019	C	MOV(Delta)	6	1
01/01/2019	C	MOV(Delta)	6	2
01/01/2019	C	MOV(Delta)	6	3
01/01/2019	C	MOV(Delta)	6	4



Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To
01/12/2019	D	BAL (cumulative)	6	0
01/12/2019	D	BAL (cumulative)	6	1

Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To
01/02/2019	E	BAL (cumulative)	11	0
01/02/2019	E	BAL (cumulative)	11	1
01/02/2019	E	BAL (cumulative)	11	2
01/02/2019	E	BAL (cumulative)	11	3

Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
TS	Term Selection	Term Selection	Active			

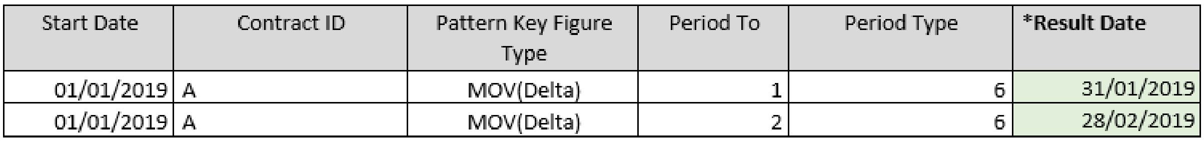
Rule Lines	Sub View							
Line	*Line Granularity	*Period Type	*Start Date	*Date Determinant	*Day of Month	*Period	*Value Type	*Result Date
1	Contract ID	Period Type	Start Date	Date Determinant	Day of Month	Period To	Pattern Key Figure Type	Date

i Note

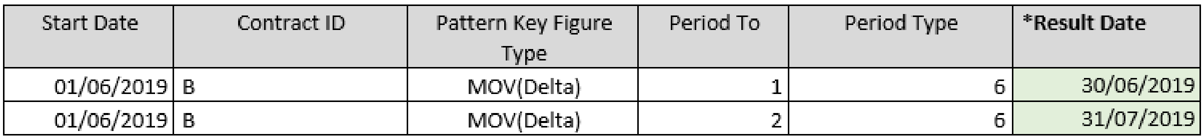
- Yellow marked fields are input fields whereas green marked fields are the output fields.
  Line Granularity = Determines the size on one partition of data.
  Value Type = Input field containing either the value “MOV” or “BAL” to determine of which type the cashflow item is made of. The user has to make sure that the field contains these two values only. This is important because different logic is applied for movements / balance types.
  Date Determinant = How the date should be determined (Codes: 1 = Start of Period, 2 = Mid of Period, 3 = End of Period, 4 = Actual Day of Period, 5 = Day of Period). In case the user enters code 5 the field *Day of Month* becomes mandatory.
  Day of Month = The user can determine the day of the result date according to the requirements (e.g. 0, 1, …., 31)

Output/Result Data

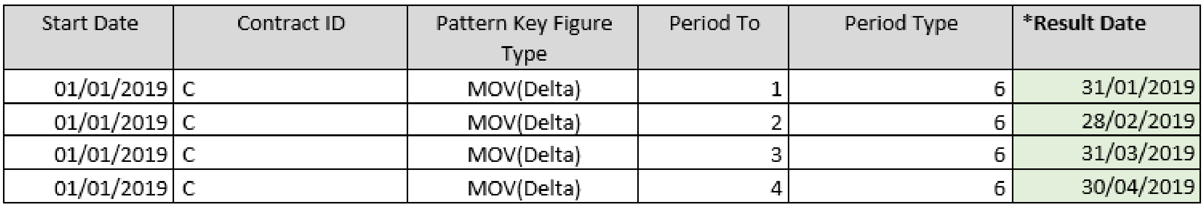
In the following examples the date determinant is set to “3” (End of Period). Period type is set to to “6” which means “monthly”.



Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
01/01/2019	A	MOV(Delta)	1	6	31/01/2019
01/01/2019	A	MOV(Delta)	2	6	28/02/2019

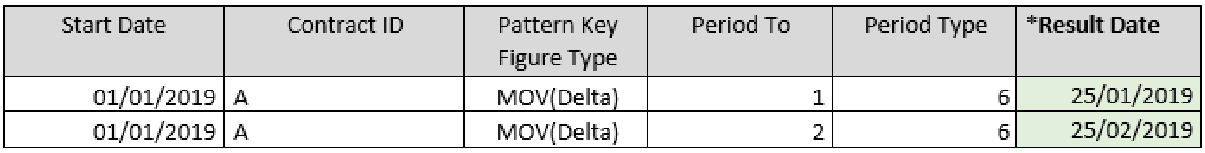


Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
01/06/2019	B	MOV(Delta)	1	6	30/06/2019
01/06/2019	B	MOV(Delta)	2	6	31/07/2019

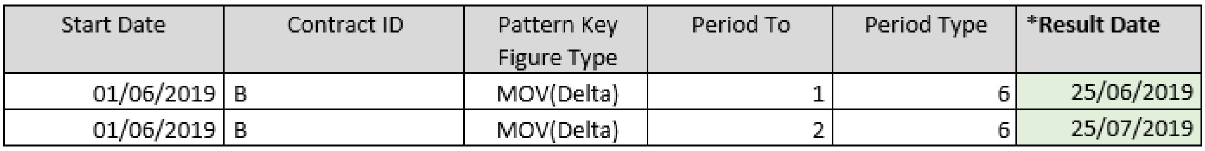


Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
01/01/2019	C	MOV(Delta)	1	6	31/01/2019
01/01/2019	C	MOV(Delta)	2	6	28/02/2019
01/01/2019	C	MOV(Delta)	3	6	31/03/2019
01/01/2019	C	MOV(Delta)	4	6	30/04/2019

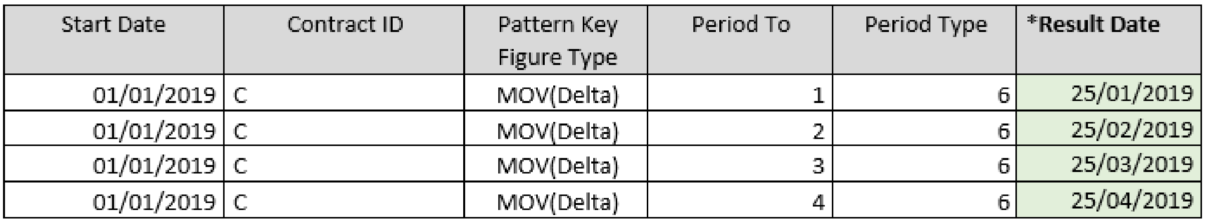
In the following examples the date determinant is set to “5” (Day of Period) where the value of *Day of Month* is set to “15”.



Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
01/01/2019	A	MOV(Delta)	1	6	25/01/2019
01/01/2019	A	MOV(Delta)	2	6	25/02/2019

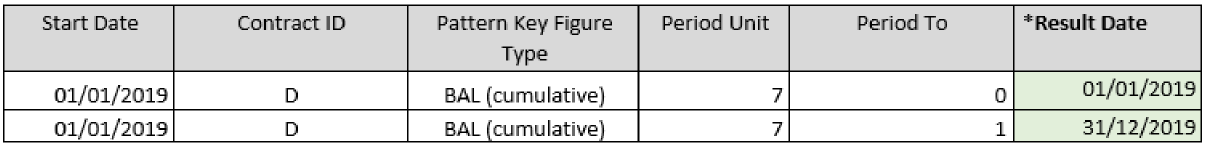


Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
01/06/2019	B	MOV(Delta)	1	6	25/06/2019
01/06/2019	B	MOV(Delta)	2	6	25/07/2019

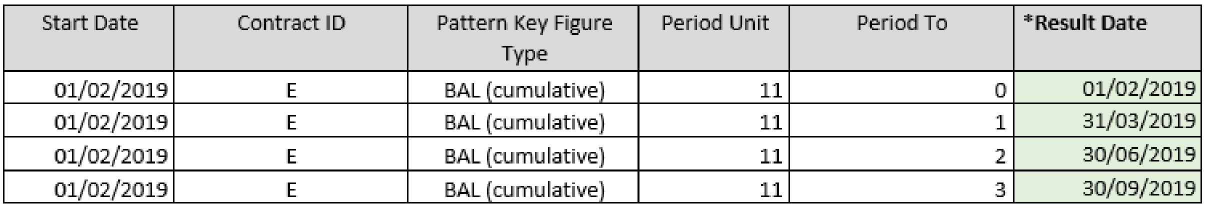


Start Date	Contract ID	Pattern Key Figure Type	Period To	Period Type	*Result Date
01/01/2019	C	MOV(Delta)	1	6	25/01/2019
01/01/2019	C	MOV(Delta)	2	6	25/02/2019
01/01/2019	C	MOV(Delta)	3	6	25/03/2019
01/01/2019	C	MOV(Delta)	4	6	25/04/2019

The two patterns below show the date determination based on different period types (quarterly, yearly).



Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	*Result Date
01/01/2019	D	BAL (cumulative)	7	0	01/01/2019
01/01/2019	D	BAL (cumulative)	7	1	31/12/2019

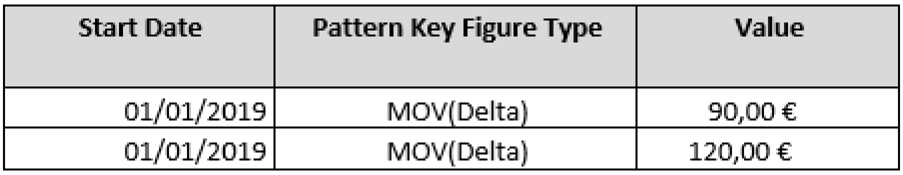


Start Date	Contract ID	Pattern Key Figure Type	Period Unit	Period To	*Result Date
01/02/2019	E	BAL (cumulative)	11	0	01/02/2019
01/02/2019	E	BAL (cumulative)	11	1	31/03/2019
01/02/2019	E	BAL (cumulative)	11	2	30/06/2019
01/02/2019	E	BAL (cumulative)	11	3	30/09/2019

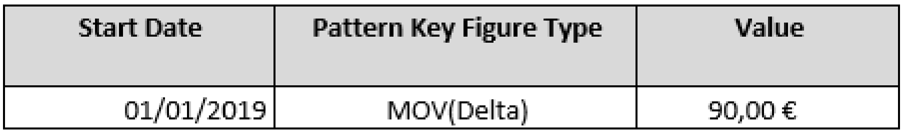
1.3.9.9 Example: Value Conversion

Input Data

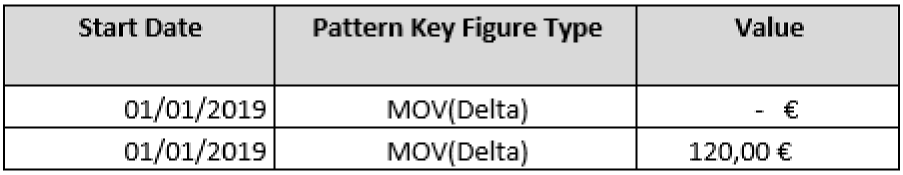
Three patterns coming from the source systems containing movement (delta) values.



Start Date	Pattern Key Figure Type	Value
01/01/2019	MOV(Delta)	90,00 €
01/01/2019	MOV(Delta)	120,00 €

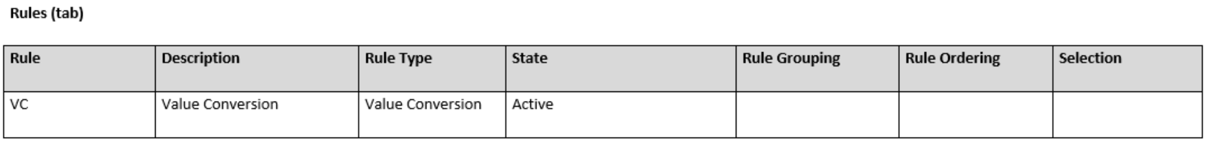


Start Date	Pattern Key Figure Type	Value
01/01/2019	MOV(Delta)	90,00 €



Start Date	Pattern Key Figure Type	Value
01/01/2019	MOV(Delta)	- €
01/01/2019	MOV(Delta)	120,00 €

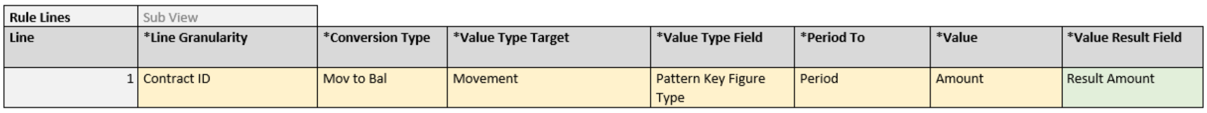
Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
VC	Value Conversion	Value Conversion	Active			

With this configuration only the cashflow items of key figure type “Movement” get converted into balance values. Pattern items which are of balance type don't get converted.

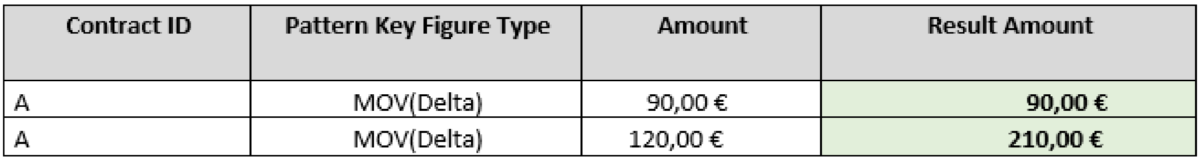


Rule Lines	Sub View						
Line	*Line Granularity	*Conversion Type	*Value Type Target	*Value Type Field	*Period To	*Value	*Value Result Field
1	Contract ID	Mov to Bal	Movement	Pattern Key Figure Type	Period	Amount	Result Amount

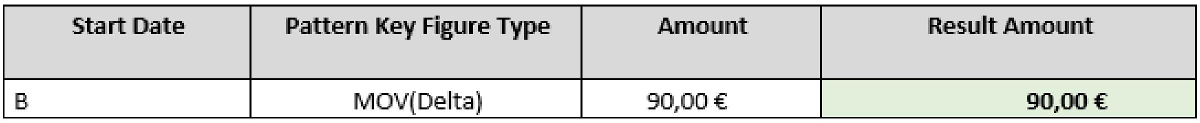
> i Note
>
> - Yellow marked fields are input fields whereas green marked fields are the output fields.
> - Line Granularity = Determines the size on one partition of data.
> - Conversion Type = Is the calculation method used. “Mov to Bal” computes the running total whereas “Bal to Mov” calculates delta values.
> - Value Type Target = This configuration field determines which key figure types (movement or balance) will be converted.
> - Value Type Field = Input field containing either the value “MOV” or “BAL” to determine of which type the cashflow item is made of. The user has to make sure that the field contains these two values only.

Output/Result Data

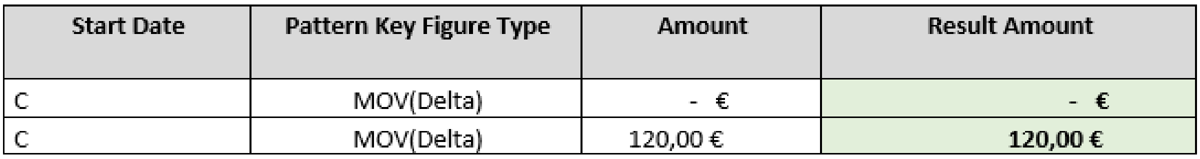
In this example the rule is computing the running total of all cash flow items having the value of “MOV” in column *Pattern Key Figure Type*.



Contract ID	Pattern Key Figure Type	Amount	Result Amount
A	MOV(Delta)	90,00 €	**90,00 €**
A	MOV(Delta)	120,00 €	**210,00 €**

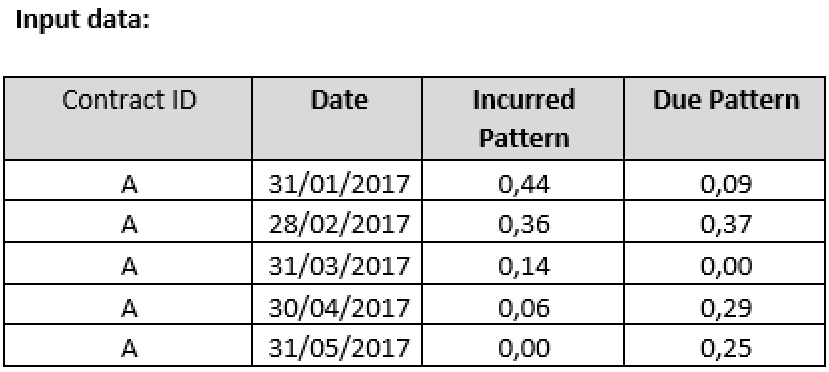


Start Date	Pattern Key Figure Type	Amount	Result Amount
B	MOV(Delta)	90,00 €	**90,00 €**



Start Date	Pattern Key Figure Type	Amount	Result Amount
C	MOV(Delta)	- €	**- €**
C	MOV(Delta)	120,00 €	**120,00 €**

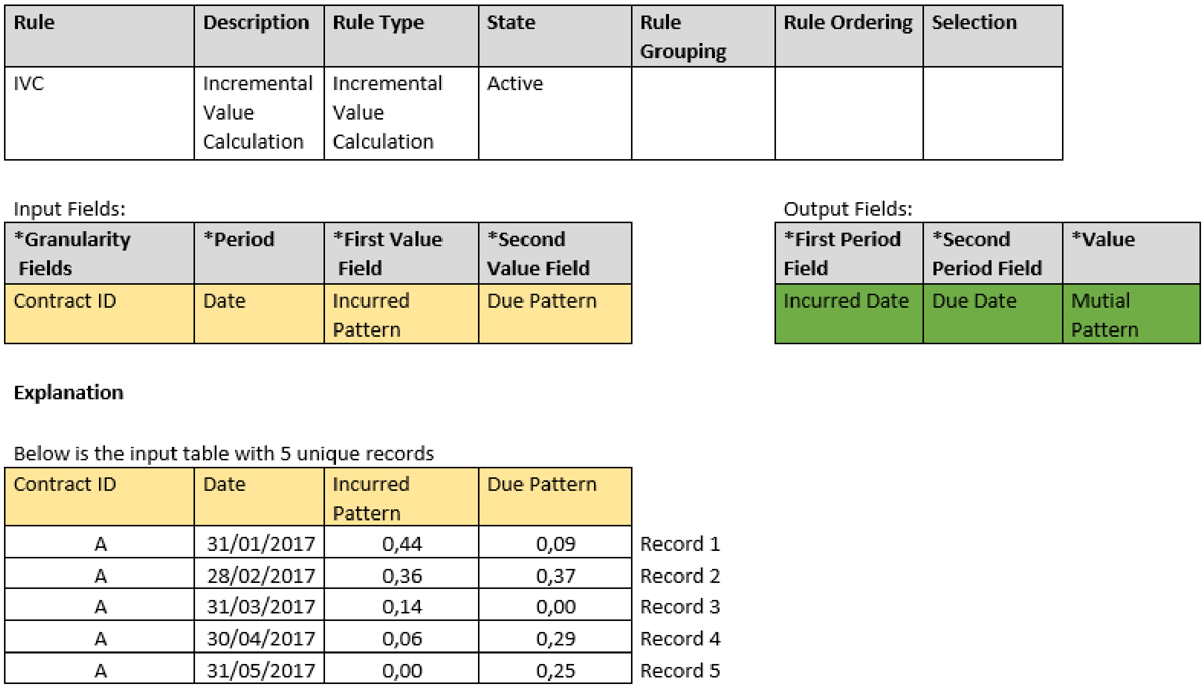
1.3.9.10 Example: Incremental Value Calculation



Input data:

Contract ID	Date	Incurred Pattern	Due Pattern
A	31/01/2017	0,44	0,09
A	28/02/2017	0,36	0,37
A	31/03/2017	0,14	0,00
A	30/04/2017	0,06	0,29
A	31/05/2017	0,00	0,25

Flow Modeling Configuration



Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
IVC	Incremental Value Calculation	Incremental Value Calculation	Active			

Input Fields:

*Granularity Fields	*Period	*First Value Field	*Second Value Field
Contract ID	Date	Incurred Pattern	Due Pattern

Output Fields:

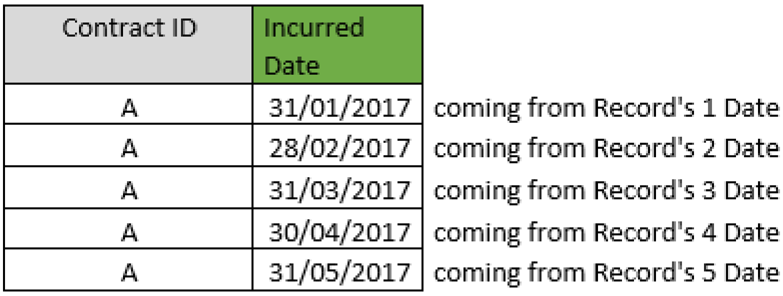
*First Period Field	*Second Period Field	*Value
Incurred Date	Due Date	Mutial Pattern

Explanation

Below is the input table with 5 unique records

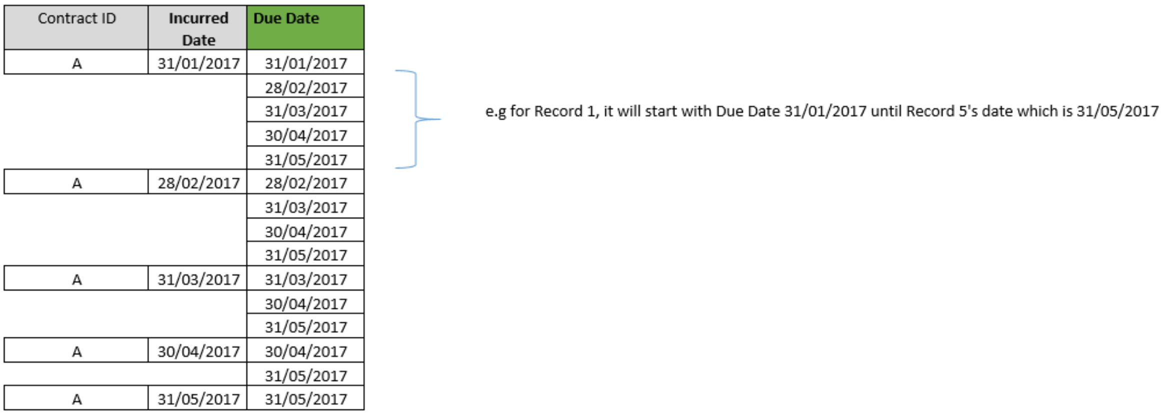
Contract ID	Date	Incurred Pattern	Due Pattern	
A	31/01/2017	0,44	0,09	Record 1
A	28/02/2017	0,36	0,37	Record 2
A	31/03/2017	0,14	0,00	Record 3
A	30/04/2017	0,06	0,29	Record 4
A	31/05/2017	0,00	0,25	Record 5

1. Explaining output field: First period field - in this example under column *Incurred Date*
   The first period field will start with the record's date from the input data, which will give 5 unique sets of data records.



Contract ID	Incurred Date	
A	31/01/2017	coming from Record's 1 Date
A	28/02/2017	coming from Record's 2 Date
A	31/03/2017	coming from Record's 3 Date
A	30/04/2017	coming from Record's 4 Date
A	31/05/2017	coming from Record's 5 Date

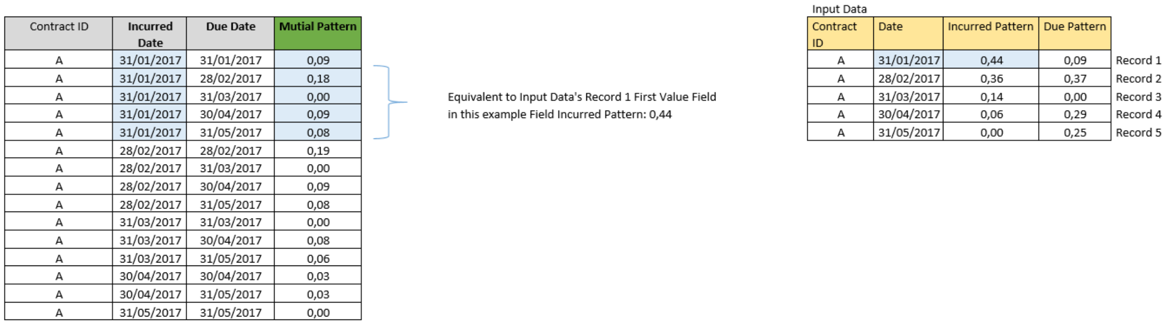
2. Explaining output field: Secondary period field - in this example under column *Due Date*
   The secondary period field will produce records based on dates coming from Records 1 to 5 starting from the date registered.



Contract ID	Incurred Date	Due Date
A	31/01/2017	31/01/2017
		28/02/2017
		31/03/2017
		30/04/2017
		31/05/2017
A	28/02/2017	28/02/2017
		31/03/2017
		30/04/2017
		31/05/2017
A	31/03/2017	31/03/2017
		30/04/2017
		31/05/2017
A	30/04/2017	30/04/2017
		31/05/2017
A	31/05/2017	31/05/2017

e.g for Record 1, it will start with Due Date 31/01/2017 until Record 5's date which is 31/05/2017

3. Explaining output field: Value - in this example under column *Mutual Pattern*
   Based on the primary period field (*Incurred Date*)



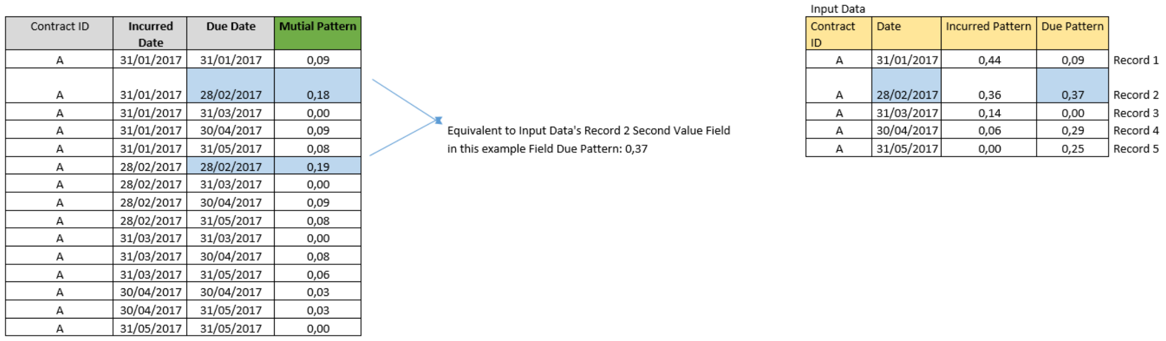
Contract ID	Incurred Date	Due Date	Mutial Pattern
A	31/01/2017	31/01/2017	0,09
A	31/01/2017	28/02/2017	0,18
A	31/01/2017	31/03/2017	0,00
A	31/01/2017	30/04/2017	0,09
A	31/01/2017	31/05/2017	0,08
A	28/02/2017	28/02/2017	0,19
A	28/02/2017	31/03/2017	0,00
A	28/02/2017	30/04/2017	0,09
A	28/02/2017	31/05/2017	0,08
A	31/03/2017	31/03/2017	0,00
A	31/03/2017	30/04/2017	0,08
A	31/03/2017	31/05/2017	0,06
A	30/04/2017	30/04/2017	0,03
A	30/04/2017	31/05/2017	0,03
A	31/05/2017	31/05/2017	0,00

Equivalent to Input Data's Record 1 First Value Field
in this example Field Incurred Pattern: 0,44

Input Data

Contract ID	Date	Incurred Pattern	Due Pattern	
A	31/01/2017	0,44	0,09	Record 1
A	28/02/2017	0,36	0,37	Record 2
A	31/03/2017	0,14	0,00	Record 3
A	30/04/2017	0,06	0,29	Record 4
A	31/05/2017	0,00	0,25	Record 5

4. Explaining output field: Value - in this example under column *Mutual Pattern*
   Based on the secondary period field (*Due Date*)



Contract ID	Incurred Date	Due Date	Mutial Pattern
A	31/01/2017	31/01/2017	0,09
A	31/01/2017	28/02/2017	0,18
A	31/01/2017	31/03/2017	0,00
A	31/01/2017	30/04/2017	0,09
A	31/01/2017	31/05/2017	0,08
A	28/02/2017	28/02/2017	0,19
A	28/02/2017	31/03/2017	0,00
A	28/02/2017	30/04/2017	0,09
A	28/02/2017	31/05/2017	0,08
A	31/03/2017	31/03/2017	0,00
A	31/03/2017	30/04/2017	0,08
A	31/03/2017	31/05/2017	0,06
A	30/04/2017	30/04/2017	0,03
A	30/04/2017	31/05/2017	0,03
A	31/05/2017	31/05/2017	0,00

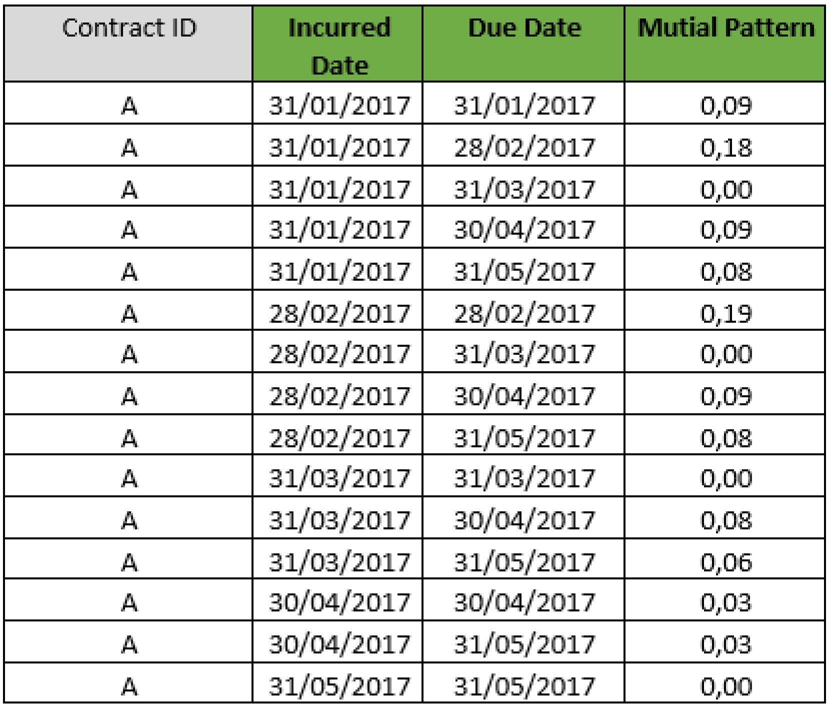
Equivalent to Input Data's Record 2 Second Value Field
in this example Field Due Pattern: 0,37

Input Data

Contract ID	Date	Incurred Pattern	Due Pattern	
A	31/01/2017	0,44	0,09	Record 1
A	28/02/2017	0,36	0,37	Record 2
A	31/03/2017	0,14	0,00	Record 3
A	30/04/2017	0,06	0,29	Record 4
A	31/05/2017	0,00	0,25	Record 5

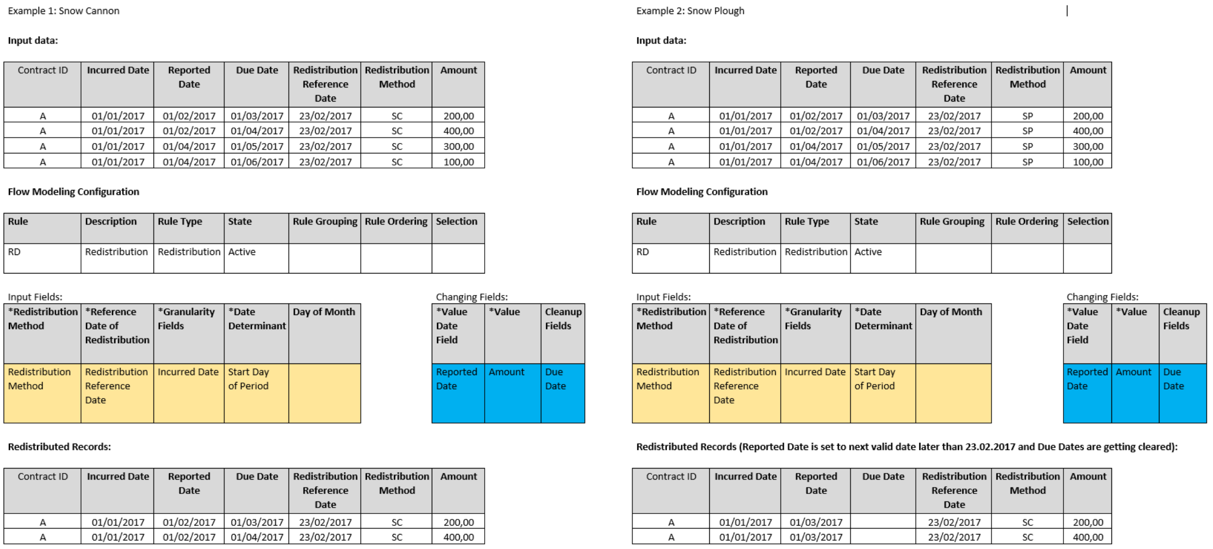
Output/Result Data

Interim result explained above will be performed by the system in all datasets based on the dates from 31/01/2017 until 31/05/2017.



Contract ID	Incurred Date	Due Date	Mutial Pattern
A	31/01/2017	31/01/2017	0,09
A	31/01/2017	28/02/2017	0,18
A	31/01/2017	31/03/2017	0,00
A	31/01/2017	30/04/2017	0,09
A	31/01/2017	31/05/2017	0,08
A	28/02/2017	28/02/2017	0,19
A	28/02/2017	31/03/2017	0,00
A	28/02/2017	30/04/2017	0,09
A	28/02/2017	31/05/2017	0,08
A	31/03/2017	31/03/2017	0,00
A	31/03/2017	30/04/2017	0,08
A	31/03/2017	31/05/2017	0,06
A	30/04/2017	30/04/2017	0,03
A	30/04/2017	31/05/2017	0,03
A	31/05/2017	31/05/2017	0,00

1.3.9.11 Example: Redistribution



Example 1: Snow Cannon

Input data:

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	01/01/2017	01/02/2017	01/03/2017	23/02/2017	SC	200,00
A	01/01/2017	01/02/2017	01/04/2017	23/02/2017	SC	400,00
A	01/01/2017	01/04/2017	01/05/2017	23/02/2017	SC	300,00
A	01/01/2017	01/04/2017	01/06/2017	23/02/2017	SC	100,00

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
RD	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

Redistributed Records:

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	01/01/2017	01/02/2017	01/03/2017	23/02/2017	SC	200,00
A	01/01/2017	01/02/2017	01/04/2017	23/02/2017	SC	400,00

Example 2: Snow Plough

Input data:

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	01/01/2017	01/02/2017	01/03/2017	23/02/2017	SP	200,00
A	01/01/2017	01/02/2017	01/04/2017	23/02/2017	SP	400,00
A	01/01/2017	01/04/2017	01/05/2017	23/02/2017	SP	300,00
A	01/01/2017	01/04/2017	01/06/2017	23/02/2017	SP	100,00

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
RD	Redistribution	Redistribution	Active			

Input Fields:

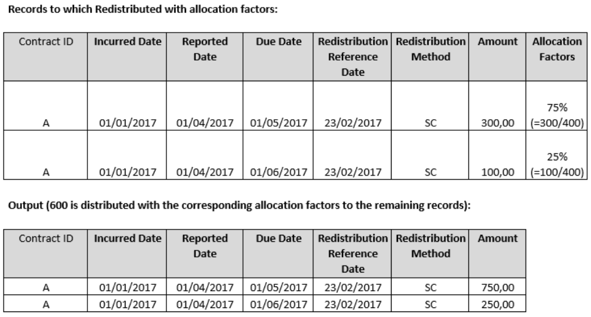
*Redistribution Method	*Reference Date of Redistribution	*Granularity Fields	*Date Determinant	Day of Month
Redistribution Method	Redistribution Reference Date	Incurred Date	Start Day of Period	

Changing Fields:

*Value Date Field	*Value	Cleanup Fields
Reported Date	Amount	Due Date

Redistributed Records (Reported Date is set to next valid date later than 23.02.2017 and Due Dates are getting cleared):

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	01/01/2017	01/03/2017		23/02/2017	SC	200,00
A	01/01/2017	01/03/2017		23/02/2017	SC	400,00

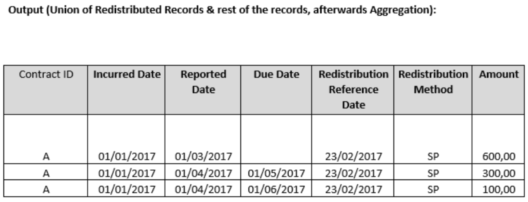


Records to which Redistributed with allocation factors:

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount	Allocation Factors
A	01/01/2017	01/04/2017	01/05/2017	23/02/2017	SC	300,00	75% (=300/400)
A	01/01/2017	01/04/2017	01/06/2017	23/02/2017	SC	100,00	25% (=100/400)

Output (600 is distributed with the corresponding allocation factors to the remaining records):

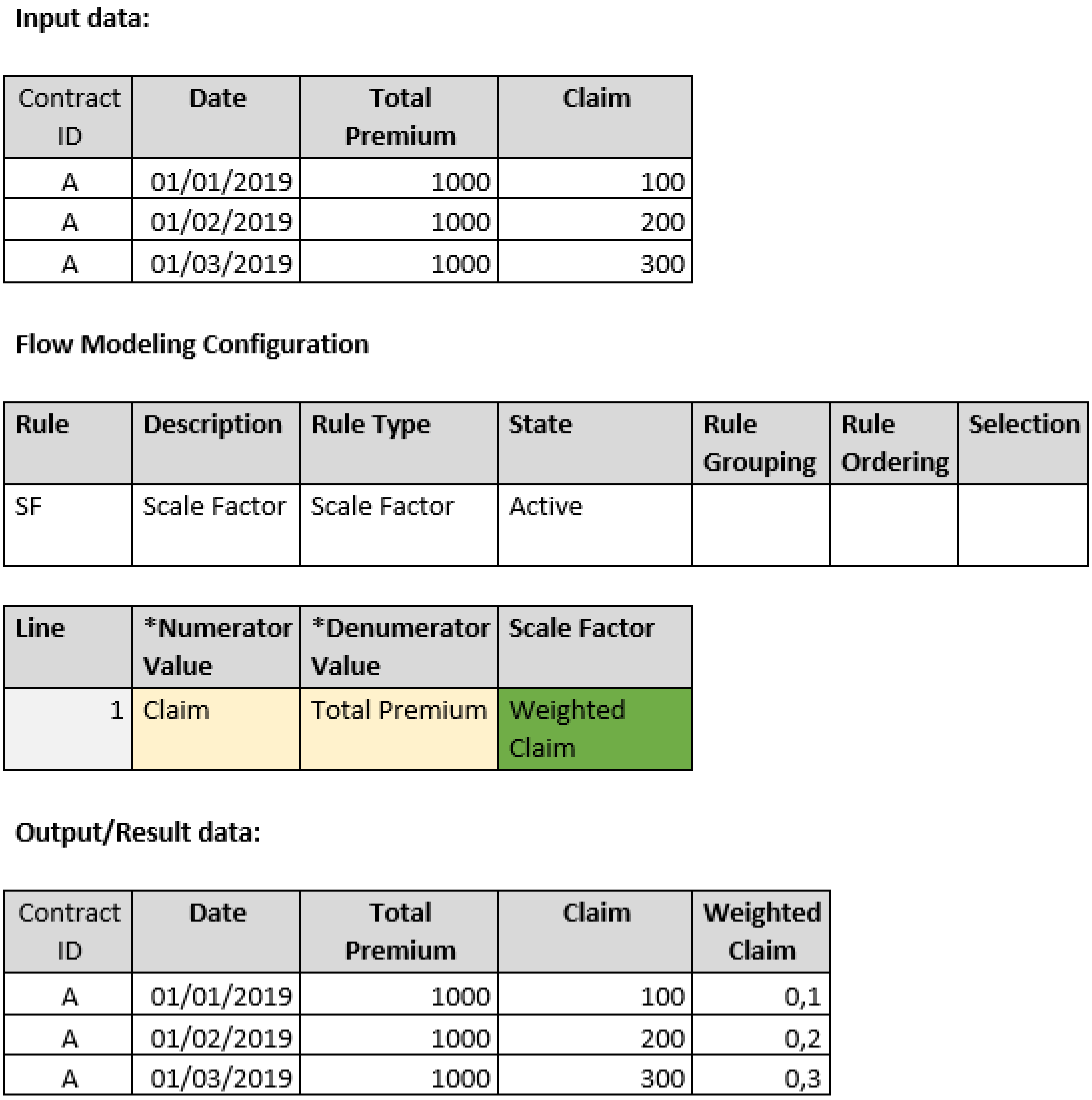
Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	01/01/2017	01/04/2017	01/05/2017	23/02/2017	SC	750,00
A	01/01/2017	01/04/2017	01/06/2017	23/02/2017	SC	250,00



Output (Union of Redistributed Records & rest of the records, afterwards Aggregation):

Contract ID	Incurred Date	Reported Date	Due Date	Redistribution Reference Date	Redistribution Method	Amount
A	01/01/2017	01/03/2017		23/02/2017	SP	600,00
A	01/01/2017	01/04/2017	01/05/2017	23/02/2017	SP	300,00
A	01/01/2017	01/04/2017	01/06/2017	23/02/2017	SP	100,00

1.3.9.12 Example: Scale Factor



Input data:

Contract ID	Date	Total Premium	Claim
A	01/01/2019	1000	100
A	01/02/2019	1000	200
A	01/03/2019	1000	300

Flow Modeling Configuration

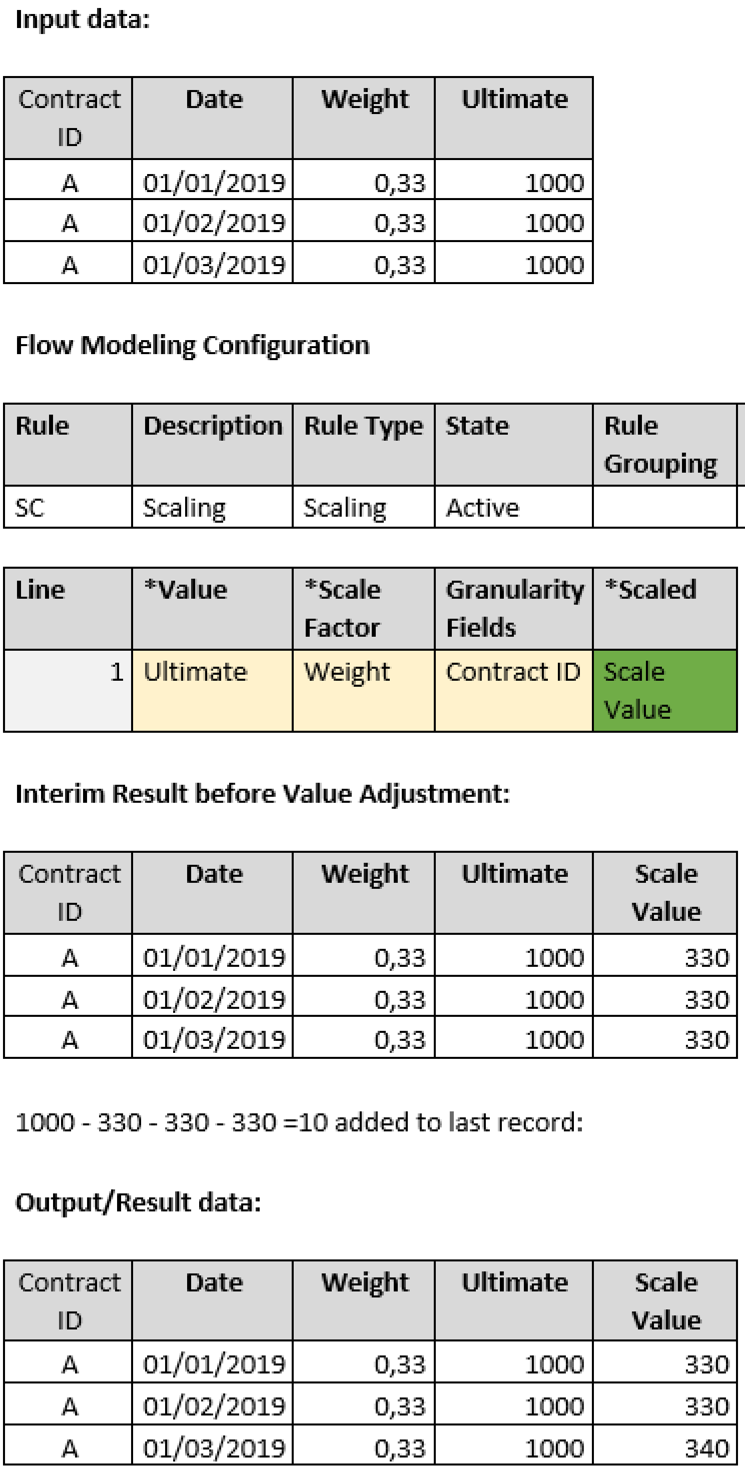
Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
SF	Scale Factor	Scale Factor	Active			

Line	*Numerator Value	*Denumerator Value	Scale Factor
1	Claim	Total Premium	Weighted Claim

Output/Result data:

Contract ID	Date	Total Premium	Claim	Weighted Claim
A	01/01/2019	1000	100	0,1
A	01/02/2019	1000	200	0,2
A	01/03/2019	1000	300	0,3

1.3.9.13 Example: Scaling



Input data:

Contract ID	Date	Weight	Ultimate
A	01/01/2019	0,33	1000
A	01/02/2019	0,33	1000
A	01/03/2019	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

Interim Result before Value Adjustment:

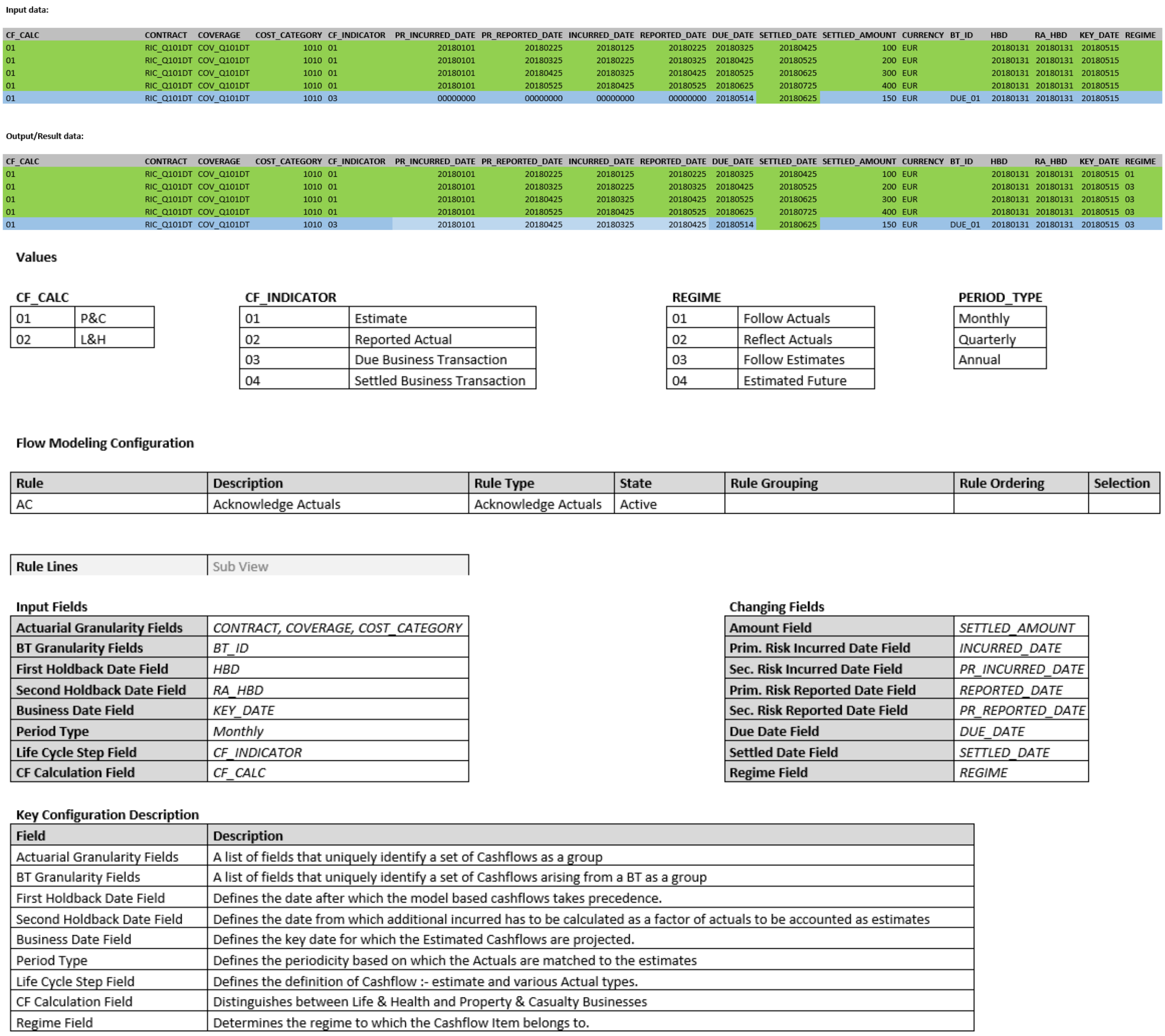
Contract ID	Date	Weight	Ultimate	Scale Value
A	01/01/2019	0,33	1000	330
A	01/02/2019	0,33	1000	330
A	01/03/2019	0,33	1000	330

1000 - 330 - 330 - 330 =10 added to last record:

Output/Result data:

Contract ID	Date	Weight	Ultimate	Scale Value
A	01/01/2019	0,33	1000	330
A	01/02/2019	0,33	1000	330
A	01/03/2019	0,33	1000	340

1.3.9.14 Example: Acknowledge Actuals



Input data:

CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	PR_INCURRED_DATE	PR_REPORTED_DATE	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	SETTLED_AMOUNT	CURRENCY	BT_ID	HBD	RA_HBD	KEY_DATE	REGIME
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180225	20180125	20180225	20180325	20180425	100	EUR		20180131	20180131	20180515	
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180325	20180225	20180325	20180425	20180525	200	EUR		20180131	20180131	20180515	
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180525	20180625	300	EUR		20180131	20180131	20180515	
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180525	20180425	20180525	20180625	20180725	400	EUR		20180131	20180131	20180515	
01	RIC_Q101DT	COV_Q101DT	1010	03	00000000	00000000	00000000	00000000	20180514	20180625	150	EUR	DUE_01	20180131	20180131	20180515	

Output/Result data:

CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	PR_INCURRED_DATE	PR_REPORTED_DATE	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	SETTLED_AMOUNT	CURRENCY	BT_ID	HBD	RA_HBD	KEY_DATE	REGIME
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180225	20180125	20180225	20180325	20180425	100	EUR		20180131	20180131	20180515	01
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180325	20180225	20180325	20180425	20180525	200	EUR		20180131	20180131	20180515	03
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180525	20180625	300	EUR		20180131	20180131	20180515	03
01	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180525	20180425	20180525	20180625	20180725	400	EUR		20180131	20180131	20180515	03
01	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180425	20180325	20180425	20180514	20180625	150	EUR	DUE_01	20180131	20180131	20180515	03

Values

CF_CALC

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 Estimates
04	Estimated Future

PERIOD_TYPE

Monthly
Quarterly
Annual

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
AC	Acknowledge Actuals	Acknowledge Actuals	Active			

Rule Lines	Sub View

Input Fields

Actuarial Granularity Fields	*CONTRACT, COVERAGE, COST_CATEGORY*
BT Granularity Fields	*BT_ID*
First Holdback Date Field	*HBD*
Second Holdback Date Field	*RA_HBD*
Business Date Field	*KEY_DATE*
Period Type	*Monthly*
Life Cycle Step Field	*CF_INDICATOR*
CF Calculation Field	*CF_CALC*

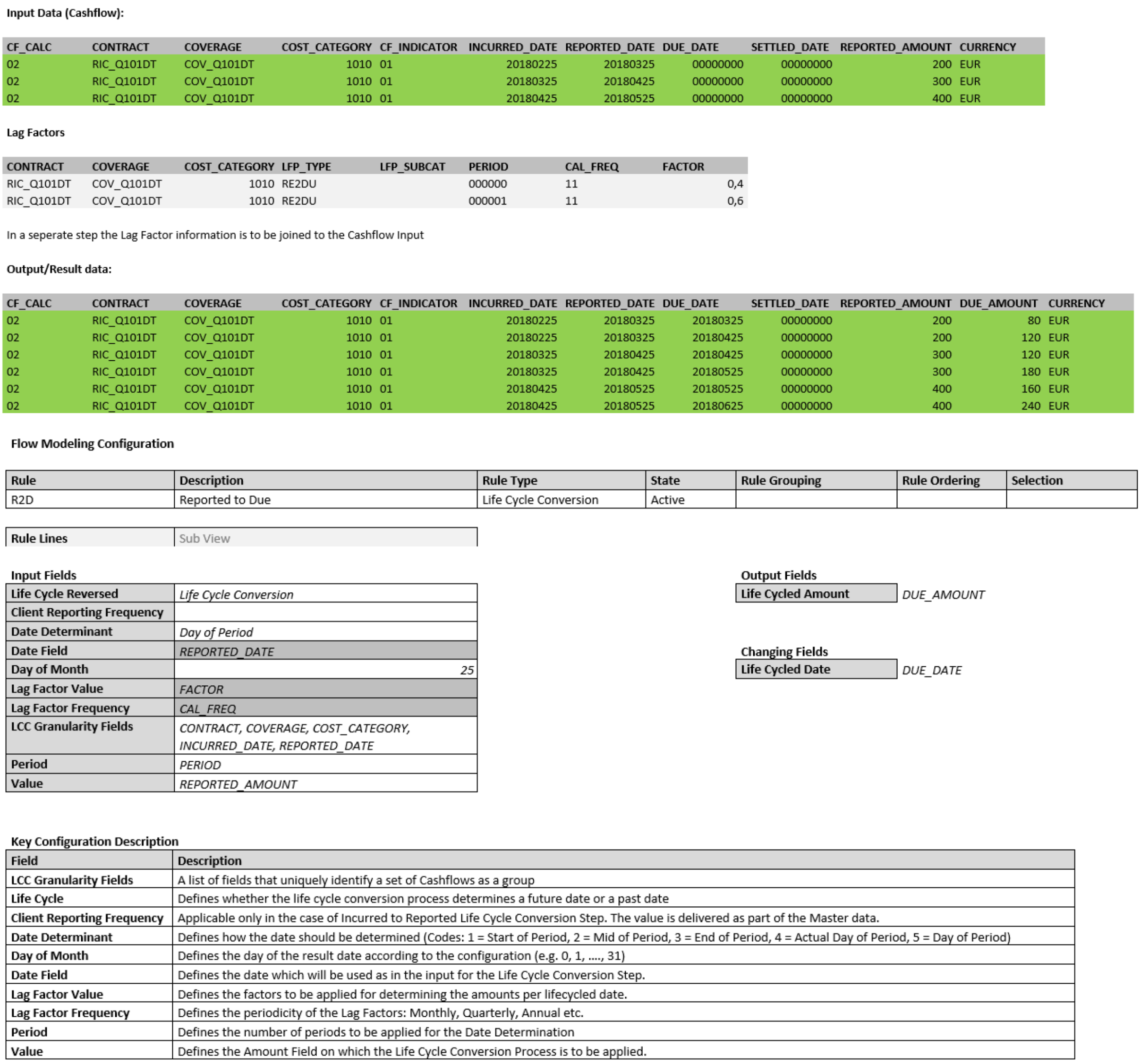
Changing Fields

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	Description
Actuarial Granularity Fields	A list of fields that uniquely identify a set of Cashflows as a group
BT Granularity Fields	A list of fields that uniquely identify a set of Cashflows arising from a BT as a group
First Holdback Date Field	Defines the date after which the model based cashflows takes precedence.
Second Holdback Date Field	Defines the date from which additional incurred has to be calculated as a factor of actuals to be accounted as estimates
Business Date Field	Defines the key date for which the Estimated Cashflows are projected.
Period Type	Defines the periodicity based on which the Actuals are matched to the estimates
Life Cycle Step Field	Defines the definition of Cashflow :- estimate and various Actual types.
CF Calculation Field	Distinguishes between Life & Health and Property & Casualty Businesses
Regime Field	Determines the regime to which the Cashflow Item belongs to.

1.3.9.15 Example: Life Cycle Conversion



Input Data (Cashflow):

CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	REPORTED_AMOUNT	CURRENCY
02	RIC_Q101DT	COV_Q101DT	1010	01	20180225	20180325	00000000	00000000	200	EUR
02	RIC_Q101DT	COV_Q101DT	1010	01	20180325	20180425	00000000	00000000	300	EUR
02	RIC_Q101DT	COV_Q101DT	1010	01	20180425	20180525	00000000	00000000	400	EUR

Lag Factors

CONTRACT	COVERAGE	COST_CATEGORY	LFP_TYPE	LFP_SUBCAT	PERIOD	CAL_FREQ	FACTOR
RIC_Q101DT	COV_Q101DT	1010	RE2DU		000000	11	0,4
RIC_Q101DT	COV_Q101DT	1010	RE2DU		000001	11	0,6

In a seperate step the Lag Factor information is to be joined to the Cashflow Input

Output/Result data:

CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	REPORTED_AMOUNT	DUE_AMOUNT	CURRENCY
02	RIC_Q101DT	COV_Q101DT	1010	01	20180225	20180325	20180325	00000000	200	80	EUR
02	RIC_Q101DT	COV_Q101DT	1010	01	20180225	20180325	20180425	00000000	200	120	EUR
02	RIC_Q101DT	COV_Q101DT	1010	01	20180325	20180425	20180425	00000000	300	120	EUR
02	RIC_Q101DT	COV_Q101DT	1010	01	20180325	20180425	20180525	00000000	300	180	EUR
02	RIC_Q101DT	COV_Q101DT	1010	01	20180425	20180525	20180525	00000000	400	160	EUR
02	RIC_Q101DT	COV_Q101DT	1010	01	20180425	20180525	20180625	00000000	400	240	EUR

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
R2D	Reported to Due	Life Cycle Conversion	Active			

Rule Lines	Sub View

Input Fields

Life Cycle Reversed	*Life Cycle Conversion*
Client Reporting Frequency	
Date Determinant	*Day of Period*
Date Field	*REPORTED_DATE*
Day of Month	*25*
Lag Factor Value	*FACTOR*
Lag Factor Frequency	*CAL_FREQ*
LCC Granularity Fields	*CONTRACT, COVERAGE, COST_CATEGORY, INCURRED_DATE, REPORTED_DATE*
Period	*PERIOD*
Value	*REPORTED_AMOUNT*

Output Fields

Life Cycled Amount	*DUE_AMOUNT*

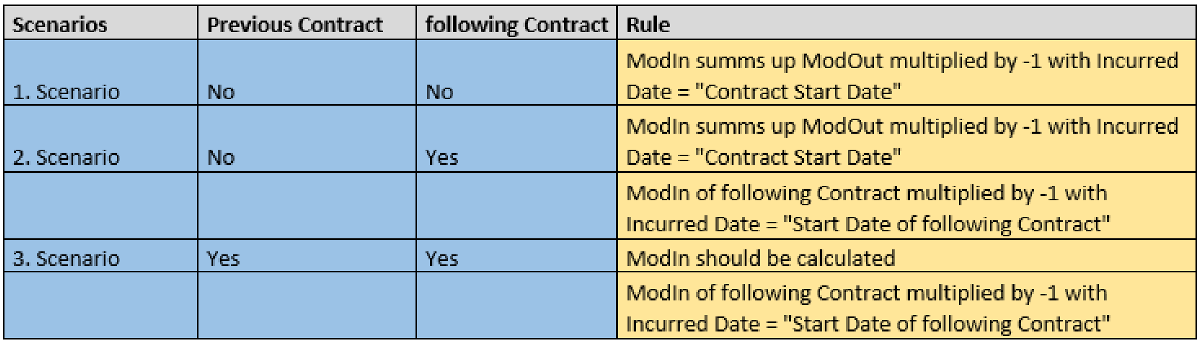
Changing Fields

Life Cycled Date	*DUE_DATE*

Key Configuration Description

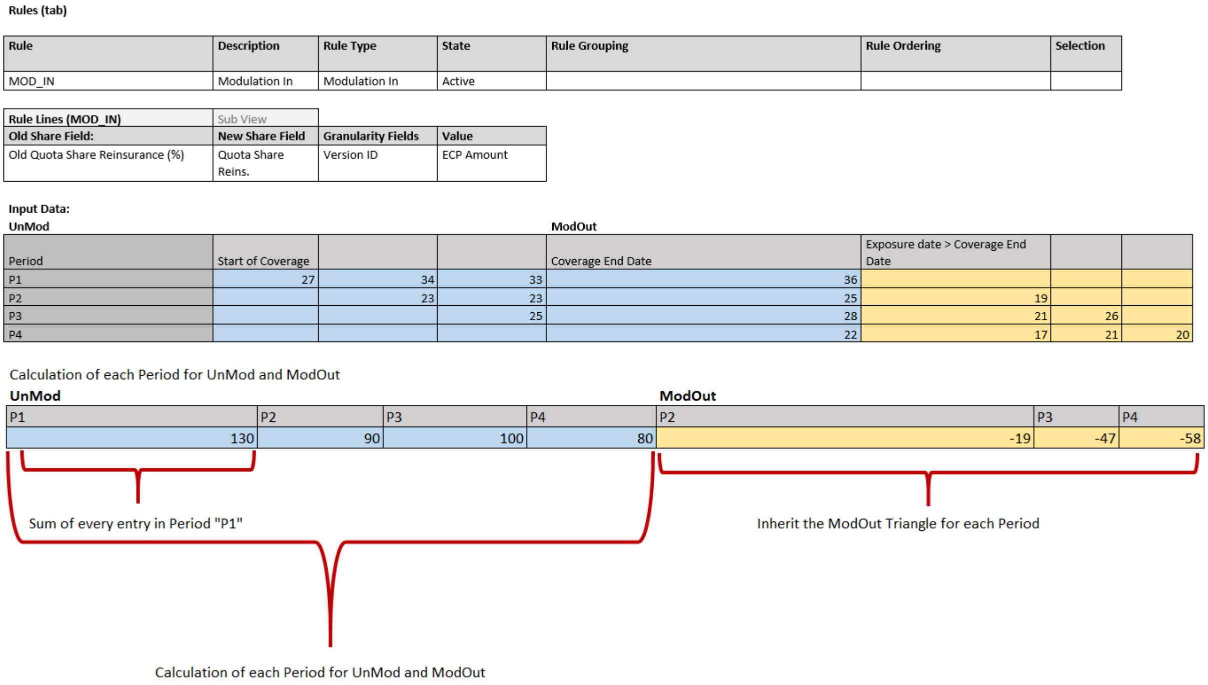
Field	Description
LCC Granularity Fields	A list of fields that uniquely identify a set of Cashflows as a group
Life Cycle	Defines whether the life cycle conversion process determines a future date or a past date
Client Reporting Frequency	Applicable only in the case of Incurred to Reported Life Cycle Conversion Step. The value is delivered as part of the Master data.
Date Determinant	Defines how the date should be determined (Codes: 1 = Start of Period, 2 = Mid of Period, 3 = End of Period, 4 = Actual Day of Period, 5 = Day of Period)
Day of Month	Defines the day of the result date according to the configuration (e.g. 0, 1, ..., 31)
Date Field	Defines the date which will be used as in the input for the Life Cycle Conversion Step.
Lag Factor Value	Defines the factors to be applied for determining the amounts per lifecycled date.
Lag Factor Frequency	Defines the periodicity of the Lag Factors: Monthly, Quarterly, Annual etc.
Period	Defines the number of periods to be applied for the Date Determination
Value	Defines the Amount Field on which the Life Cycle Conversion Process is to be applied.

1.3.9.16 Example: Modulation In



Scenarios	Previous Contract	following Contract	Rule
1. Scenario	No	No	ModIn summs up ModOut multiplied by -1 with Incurred Date = "Contract Start Date"
2. Scenario	No	Yes	ModIn summs up ModOut multiplied by -1 with Incurred Date = "Contract Start Date"
			ModIn of following Contract multiplied by -1 with Incurred Date = "Start Date of following Contract"
3. Scenario	Yes	Yes	ModIn should be calculated
			ModIn of following Contract multiplied by -1 with Incurred Date = "Start Date of following Contract"

Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
MOD_IN	Modulation In	Modulation In	Active			

Rule Lines (MOD_IN)	Sub View		
Old Share Field:	**New Share Field**	**Granularity Fields**	**Value**
Old Quota Share Reinsurance (%)	Quota Share Reins.	Version ID	ECP Amount

Input Data:

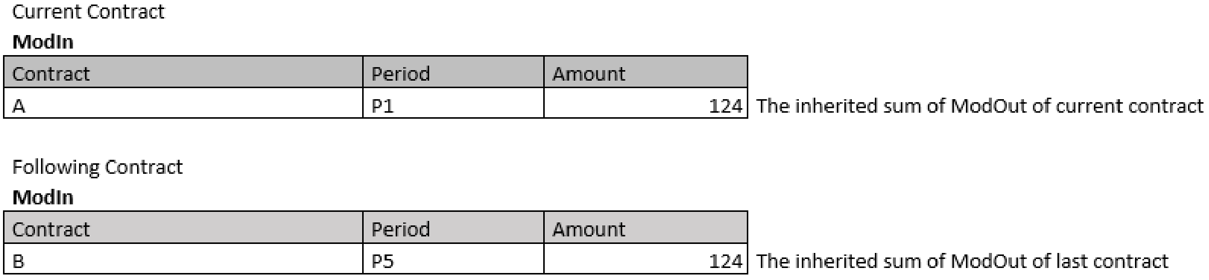
UnMod				ModOut			
Period	Start of Coverage			Coverage End Date	Exposure date > Coverage End Date		
P1	27	34	33	36			
P2		23	23	25	19		
P3			25	28	21	26	
P4				22	17	21	20

Calculation of each Period for UnMod and ModOut

UnMod				ModOut		
P1	P2	P3	P4	P2	P3	P4
130	90	100	80	-19	-47	-58

Sum of every entry in Period "P1"
Inherit the ModOut Triangle for each Period
Calculation of each Period for UnMod and ModOut

Output/Result Data



Current Contract

ModIn

Contract	Period	Amount
A	P1	124

The inherited sum of ModOut of current contract

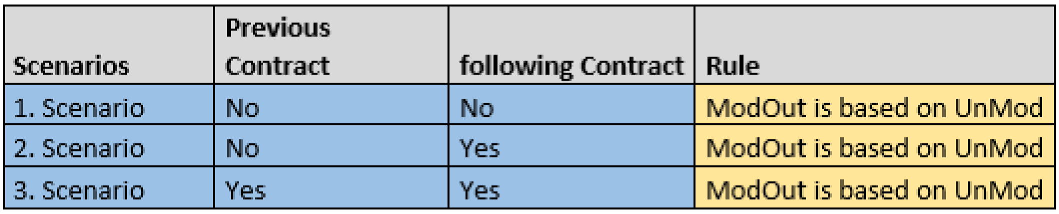
Following Contract

ModIn

Contract	Period	Amount
B	P5	124

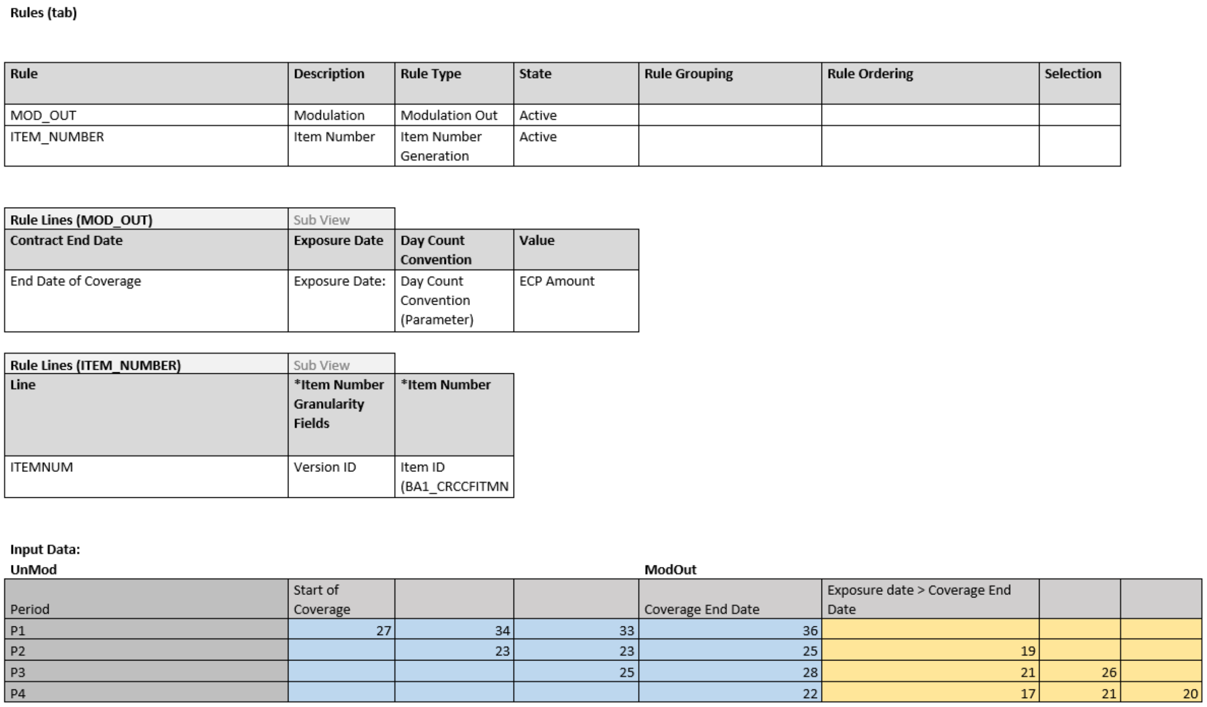
The inherited sum of ModOut of last contract

1.3.9.17 Example: Modulation Out



Scenarios	Previous Contract	following Contract	Rule
1. Scenario	No	No	ModOut is based on UnMod
2. Scenario	No	Yes	ModOut is based on UnMod
3. Scenario	Yes	Yes	ModOut is based on UnMod

Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
MOD_OUT	Modulation	Modulation Out	Active			
ITEM_NUMBER	Item Number	Item Number Generation	Active			

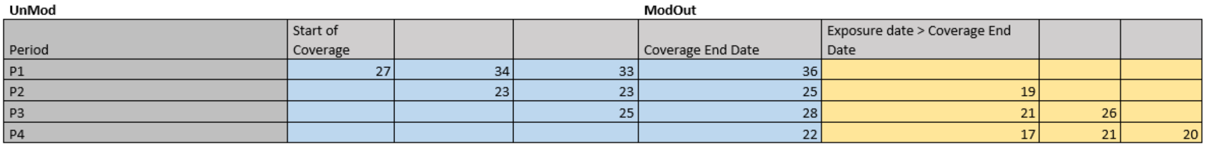
Rule Lines (MOD_OUT)	Sub View		
Contract End Date	**Exposure Date**	**Day Count Convention**	**Value**
End Date of Coverage	Exposure Date:	Day Count Convention (Parameter)	ECP Amount

Rule Lines (ITEM_NUMBER)	Sub View	
Line	***Item Number Granularity Fields**	***Item Number**
ITEMNUM	Version ID	Item ID (BA1_CRCCFITMN

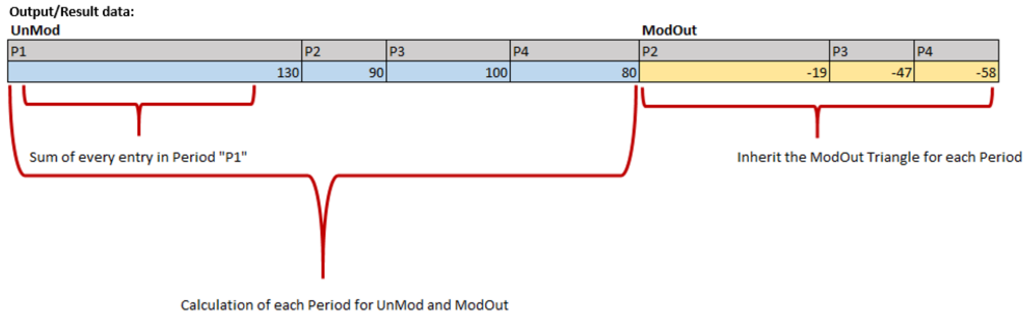
Input Data:

UnMod				ModOut			
Period	Start of Coverage			Coverage End Date	Exposure date > Coverage End Date		
P1	27	34	33	36			
P2		23	23	25	19		
P3			25	28	21	26	
P4				22	17	21	20

Input Data



UnMod				ModOut			
Period	Start of Coverage			Coverage End Date	Exposure date > Coverage End Date		
P1	27	34	33	36			
P2		23	23	25	19		
P3			25	28	21	26	
P4				22	17	21	20

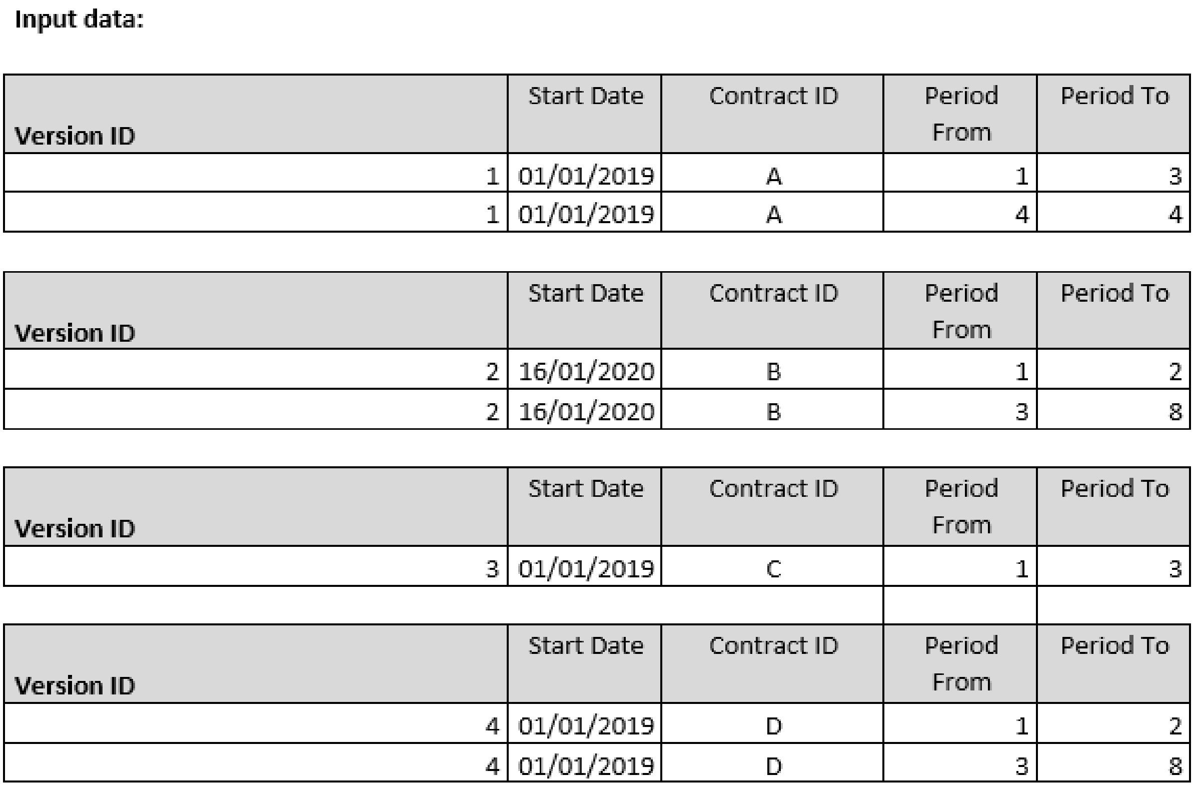


Output/Result data:

UnMod				ModOut		
P1	P2	P3	P4	P2	P3	P4
130	90	100	80	-19	-47	-58

Sum of every entry in Period "P1"
Inherit the ModOut Triangle for each Period
Calculation of each Period for UnMod and ModOut

1.3.9.18 Example: Item Number Generation



Input data:

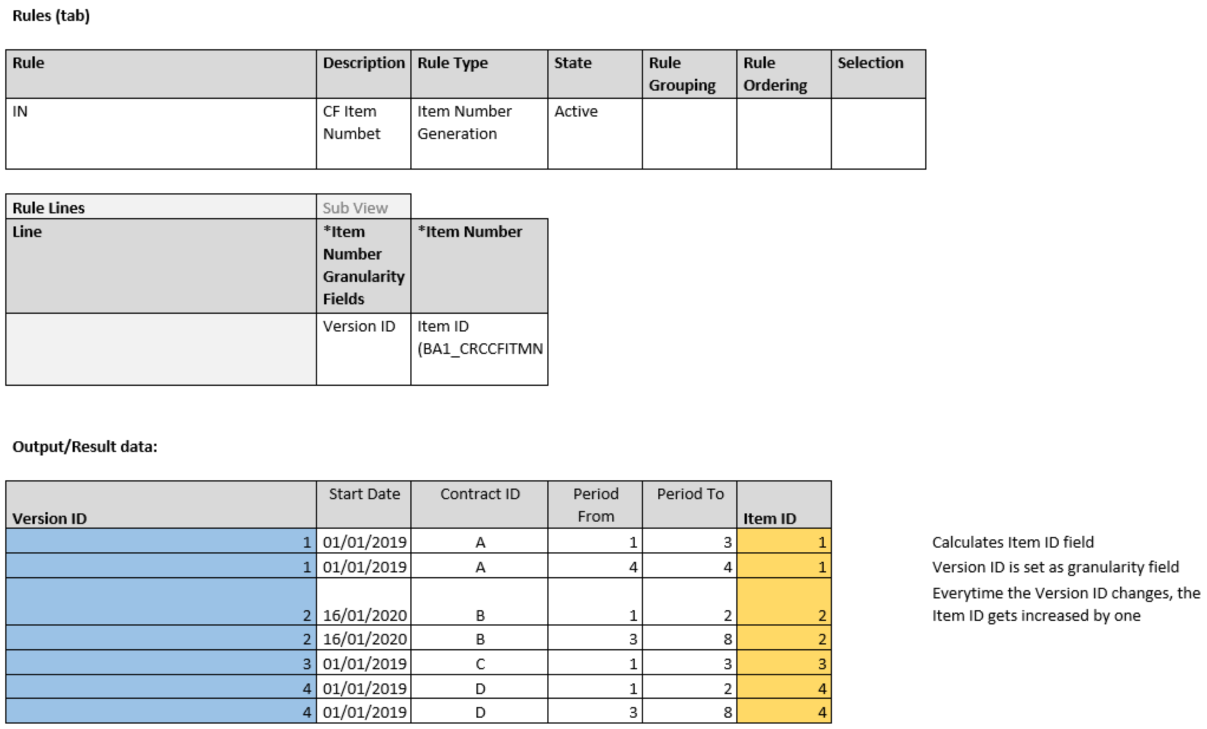
Version ID	Start Date	Contract ID	Period From	Period To
1	01/01/2019	A	1	3
1	01/01/2019	A	4	4

Version ID	Start Date	Contract ID	Period From	Period To
2	16/01/2020	B	1	2
2	16/01/2020	B	3	8

Version ID	Start Date	Contract ID	Period From	Period To
3	01/01/2019	C	1	3

Version ID	Start Date	Contract ID	Period From	Period To
4	01/01/2019	D	1	2
4	01/01/2019	D	3	8

Flow Modeling Configuration



Rules (tab)

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
IN	CF Item Numbet	Item Number Generation	Active			

Rule Lines	Sub View	
Line	*Item Number Granularity Fields	*Item Number
	Version ID	Item ID (BA1_CRCCFITMN

Output/Result data:

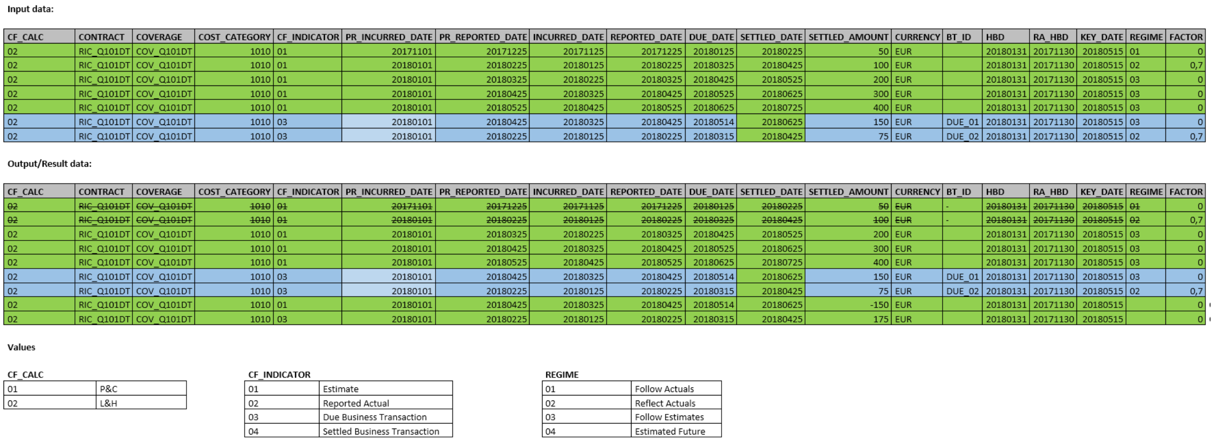
Version ID	Start Date	Contract ID	Period From	Period To	Item ID
1	01/01/2019	A	1	3	1
1	01/01/2019	A	4	4	1
2	16/01/2020	B	1	2	2
2	16/01/2020	B	3	8	2
3	01/01/2019	C	1	3	3
4	01/01/2019	D	1	2	4
4	01/01/2019	D	3	8	4

Calculates Item ID field

Version ID is set as granularity field

Everytime the Version ID changes, the Item ID gets increased by one

1.3.9.19 Example: Cashflow Regime



Input data:

CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	PR_INCURRED_DATE	PR_REPORTED_DATE	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	SETTLED_AMOUNT	CURRENCY	BT_ID	HBD	RA_HBD	KEY_DATE	REGIME	FACTOR
02	RIC_Q101DT	COV_Q101DT	1010	01	20171101	20171225	20171125	20171225	20180125	20180225	50	EUR		20180131	20171130	20180515	01	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180225	20180125	20180225	20180325	20180425	100	EUR		20180131	20171130	20180515	02	0,7
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180325	20180225	20180325	20180425	20180525	200	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180525	20180625	300	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180525	20180425	20180525	20180625	20180725	400	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180425	20180325	20180425	20180514	20180625	150	EUR	DUE_01	20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180225	20180125	20180225	20180315	20180425	75	EUR	DUE_02	20180131	20171130	20180515	02	0,7

Output/Result data:

CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	PR_INCURRED_DATE	PR_REPORTED_DATE	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	SETTLED_AMOUNT	CURRENCY	BT_ID	HBD	RA_HBD	KEY_DATE	REGIME	FACTOR
~~02~~	~~RIC_Q101DT~~	~~COV_Q101DT~~	~~1010~~	~~01~~	~~20171101~~	~~20171225~~	~~20171125~~	~~20171225~~	~~20180125~~	~~20180225~~	~~50~~	~~EUR~~	-	~~20180131~~	~~20171130~~	~~20180515~~	~~01~~	0
~~02~~	~~RIC_Q101DT~~	~~COV_Q101DT~~	~~1010~~	~~01~~	~~20180101~~	~~20180225~~	~~20180125~~	~~20180225~~	~~20180325~~	~~20180425~~	~~100~~	~~EUR~~	-	~~20180131~~	~~20171130~~	~~20180515~~	~~02~~	0,7
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180325	20180225	20180325	20180425	20180525	200	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180525	20180625	300	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180525	20180425	20180525	20180625	20180725	400	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180425	20180325	20180425	20180514	20180625	150	EUR	DUE_01	20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180225	20180125	20180225	20180315	20180425	75	EUR	DUE_02	20180131	20171130	20180515	02	0,7
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180514	20180625	-150	EUR		20180131	20171130	20180515		0
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180225	20180125	20180225	20180315	20180425	175	EUR		20180131	20171130	20180515		0

Values

CF_CALC

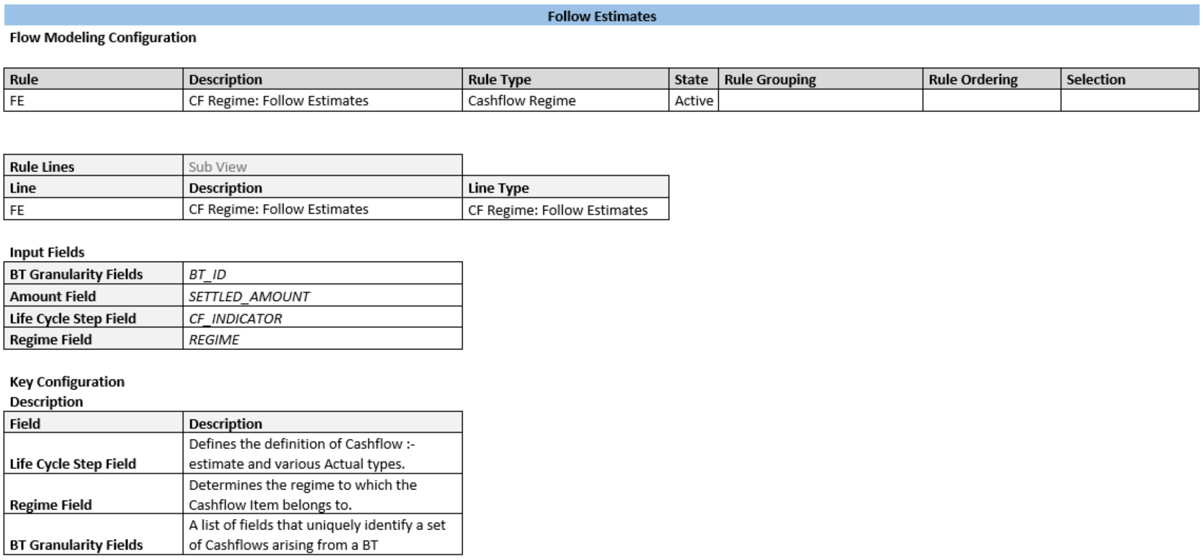
01	P&C
02	L&H

CF_INDICATOR

01	Estimate
02	Reported Actual
03	Due Business Transaction
04	Settled Business Transaction

REGIME

01	Follow Actuals
02	Reflect Actuals
03	Follow Estimates
04	Estimated Future



Follow Estimates

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
FE	CF Regime: Follow Estimates	Cashflow Regime	Active			

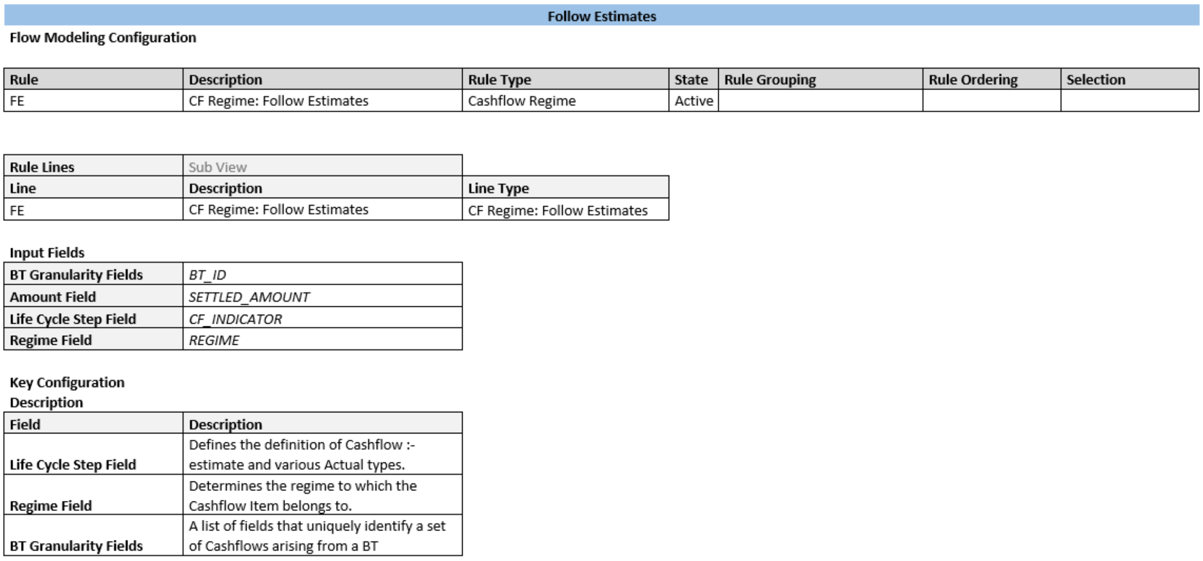
Rule Lines	Sub View	
Line	**Description**	**Line Type**
FE	CF Regime: Follow Estimates	CF Regime: Follow Estimates

Input Fields

BT Granularity Fields	*BT_ID*
Amount Field	*SETTLED_AMOUNT*
Life Cycle Step Field	*CF_INDICATOR*
Regime Field	*REGIME*

Key Configuration Description

Field	Description
Life Cycle Step Field	Defines the definition of Cashflow :- estimate and various Actual types.
Regime Field	Determines the regime to which the Cashflow Item belongs to.
BT Granularity Fields	A list of fields that uniquely identify a set of Cashflows arising from a BT



Follow Estimates

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
FE	CF Regime: Follow Estimates	Cashflow Regime	Active			

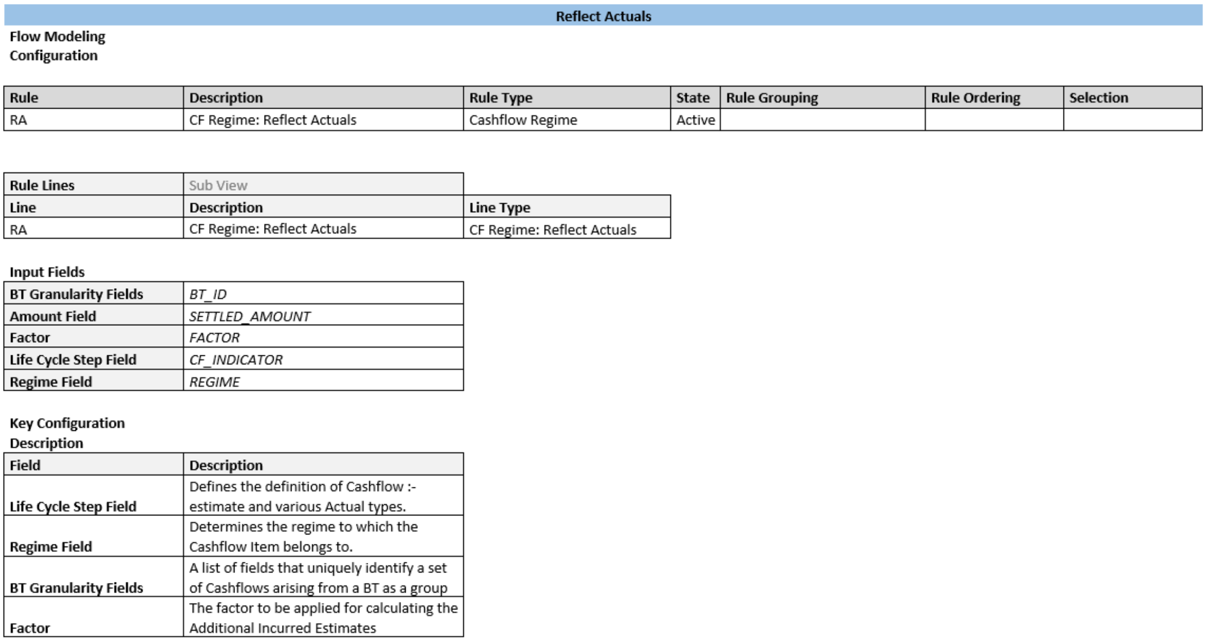
Rule Lines	Sub View	
Line	**Description**	**Line Type**
FE	CF Regime: Follow Estimates	CF Regime: Follow Estimates

Input Fields

BT Granularity Fields	*BT_ID*
Amount Field	*SETTLED_AMOUNT*
Life Cycle Step Field	*CF_INDICATOR*
Regime Field	*REGIME*

Key Configuration Description

Field	Description
Life Cycle Step Field	Defines the definition of Cashflow :- estimate and various Actual types.
Regime Field	Determines the regime to which the Cashflow Item belongs to.
BT Granularity Fields	A list of fields that uniquely identify a set of Cashflows arising from a BT



Reflect Actuals

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
RA	CF Regime: Reflect Actuals	Cashflow Regime	Active			

Rule Lines	Sub View	
Line	**Description**	**Line Type**
RA	CF Regime: Reflect Actuals	CF Regime: Reflect Actuals

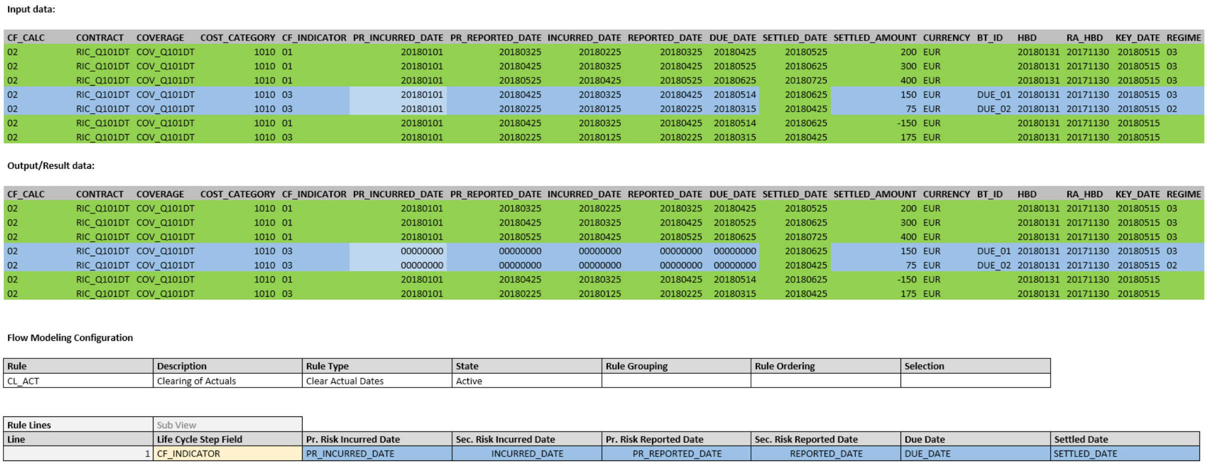
Input Fields

BT Granularity Fields	*BT_ID*
Amount Field	*SETTLED_AMOUNT*
Factor	*FACTOR*
Life Cycle Step Field	*CF_INDICATOR*
Regime Field	*REGIME*

Key Configuration Description

Field	Description
Life Cycle Step Field	Defines the definition of Cashflow :- estimate and various Actual types.
Regime Field	Determines the regime to which the Cashflow Item belongs to.
BT Granularity Fields	A list of fields that uniquely identify a set of Cashflows arising from a BT as a group
Factor	The factor to be applied for calculating the Additional Incurred Estimates

1.3.9.20 Example: Clear Actual Dates



Input data:

CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	PR_INCURRED_DATE	PR_REPORTED_DATE	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	SETTLED_AMOUNT	CURRENCY	BT_ID	HBD	RA_HBD	KEY_DATE	REGIME
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180325	20180225	20180325	20180425	20180525	200	EUR		20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180525	20180625	300	EUR		20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180525	20180425	20180525	20180625	20180725	400	EUR		20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180425	20180325	20180425	20180514	20180625	150	EUR	DUE_01	20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180225	20180125	20180225	20180315	20180425	75	EUR	DUE_02	20180131	20171130	20180515	02
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180514	20180625	-150	EUR		20180131	20171130	20180515	
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180225	20180125	20180225	20180315	20180425	175	EUR		20180131	20171130	20180515	

Output/Result data:

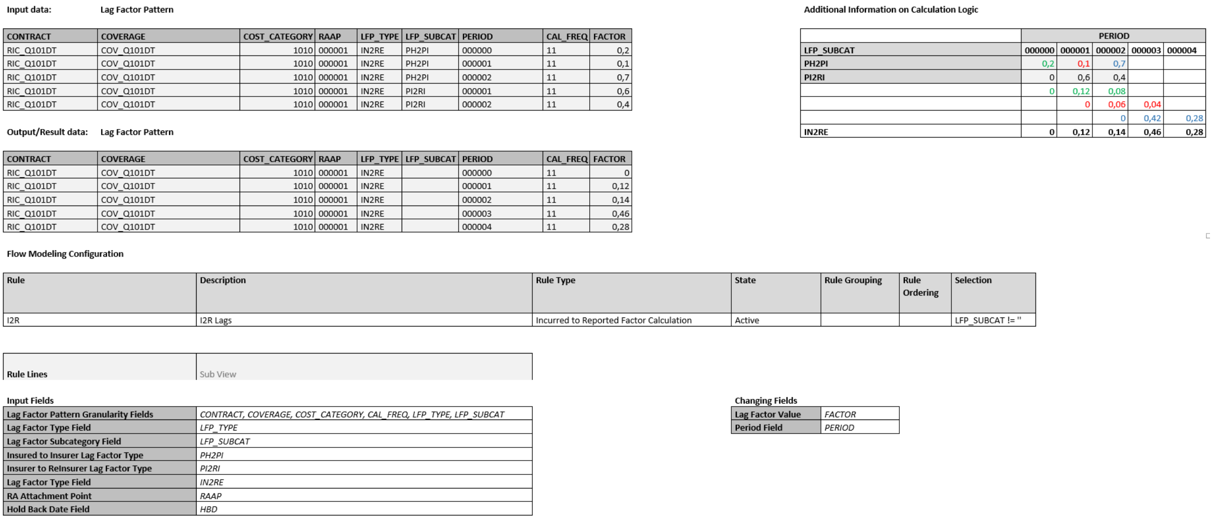
CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	PR_INCURRED_DATE	PR_REPORTED_DATE	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	SETTLED_AMOUNT	CURRENCY	BT_ID	HBD	RA_HBD	KEY_DATE	REGIME
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180325	20180225	20180325	20180425	20180525	200	EUR		20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180525	20180625	300	EUR		20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180525	20180425	20180525	20180625	20180725	400	EUR		20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	03	00000000	00000000	00000000	00000000	00000000	20180625	150	EUR	DUE_01	20180131	20171130	20180515	03
02	RIC_Q101DT	COV_Q101DT	1010	03	00000000	00000000	00000000	00000000	00000000	20180425	75	EUR	DUE_02	20180131	20171130	20180515	02
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180514	20180625	-150	EUR		20180131	20171130	20180515	
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180225	20180125	20180225	20180315	20180425	175	EUR		20180131	20171130	20180515	

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
CL_ACT	Clearing of Actuals	Clear Actual Dates	Active			

Rule Lines	Sub View						
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**
1	CF_INDICATOR	PR_INCURRED_DATE	INCURRED_DATE	PR_REPORTED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE

1.3.9.21 Example: Incurred to Reported Factor Calculation



Input data: **Lag Factor Pattern**

CONTRACT	COVERAGE	COST_CATEGORY	RAAP	LFP_TYPE	LFP_SUBCAT	PERIOD	CAL_FREQ	FACTOR
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000000	11	0,2
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000001	11	0,1
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000002	11	0,7
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PI2RI	000001	11	0,6
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PI2RI	000002	11	0,4

Output/Result data: **Lag Factor Pattern**

CONTRACT	COVERAGE	COST_CATEGORY	RAAP	LFP_TYPE	LFP_SUBCAT	PERIOD	CAL_FREQ	FACTOR
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE		000000	11	0
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE		000001	11	0,12
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE		000002	11	0,14
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE		000003	11	0,46
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE		000004	11	0,28

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		
	0	0,12	0,08		
		0	0,06	0,04	
			0	0,42	0,28
IN2RE	**0**	**0,12**	**0,14**	**0,46**	**0,28**

Flow Modeling Configuration

Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
I2R	I2R Lags	Incurred to Reported Factor Calculation	Active			LFP_SUBCAT != ''

Rule Lines	Sub View

Input Fields

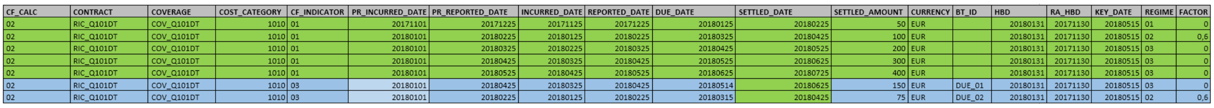
Lag Factor Pattern Granularity Fields	*CONTRACT, COVERAGE, COST_CATEGORY, CAL_FREQ, LFP_TYPE, LFP_SUBCAT*
Lag Factor Type Field	*LFP_TYPE*
Lag Factor Subcategory Field	*LFP_SUBCAT*
Insured to Insurer Lag Factor Type	*PH2PI*
Insurer to ReInsurer Lag Factor Type	*PI2RI*
Lag Factor Type Field	*IN2RE*
RA Attachment Point	*RAAP*
Hold Back Date Field	*HBD*

Changing Fields

Lag Factor Value	*FACTOR*
Period Field	*PERIOD*

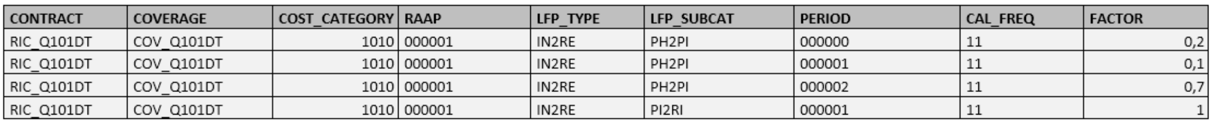
1.3.9.22 Example: Factor for Additional Incurred

Input Data 1: Cashflow



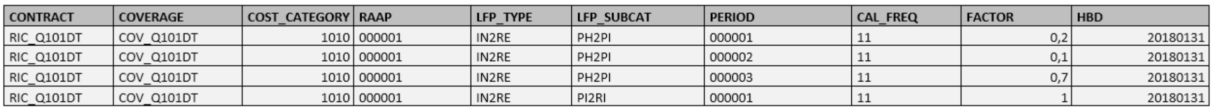
CF_CALC	CONTRACT	COVERAGE	COST_CATEGORY	CF_INDICATOR	PR_INCURRED_DATE	PR_REPORTED_DATE	INCURRED_DATE	REPORTED_DATE	DUE_DATE	SETTLED_DATE	SETTLED_AMOUNT	CURRENCY	BT_ID	HBD	RA_HBD	KEY_DATE	REGIME	FACTOR
02	RIC_Q101DT	COV_Q101DT	1010	01	20171101	20171225	20171125	20171225	20180125	20180225	50	EUR		20180131	20171130	20180515	01	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180225	20180125	20180225	20180325	20180425	100	EUR		20180131	20171130	20180515	02	0,6
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180325	20180225	20180325	20180425	20180525	200	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180425	20180325	20180425	20180525	20180625	300	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	01	20180101	20180525	20180425	20180525	20180625	20180725	400	EUR		20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180425	20180325	20180425	20180514	20180625	150	EUR	DUE_01	20180131	20171130	20180515	03	0
02	RIC_Q101DT	COV_Q101DT	1010	03	20180101	20180225	20180125	20180225	20180315	20180425	75	EUR	DUE_02	20180131	20171130	20180515	02	0,6

Input data 2: Lag Factor Pattern



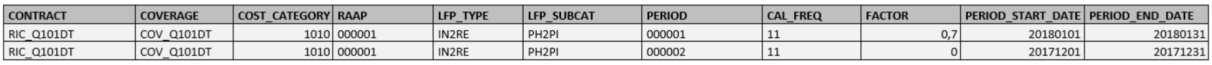
CONTRACT	COVERAGE	COST_CATEGORY	RAAP	LFP_TYPE	LFP_SUBCAT	PERIOD	CAL_FREQ	FACTOR
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000000	11	0,2
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000001	11	0,1
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000002	11	0,7
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PI2RI	000001	11	1

In a separate configuration step the Hold Back Date is to be looked up for the Lag Factor Granularity.



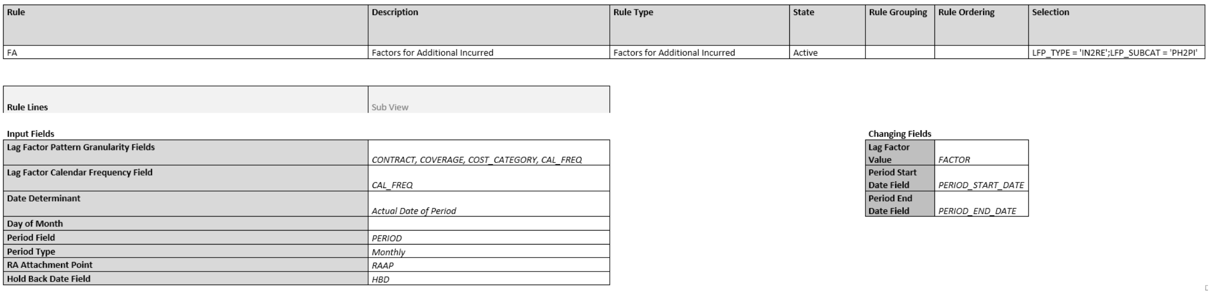
CONTRACT	COVERAGE	COST_CATEGORY	RAAP	LFP_TYPE	LFP_SUBCAT	PERIOD	CAL_FREQ	FACTOR	HBD
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000001	11	0,2	20180131
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000002	11	0,1	20180131
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000003	11	0,7	20180131
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PI2RI	000001	11	1	20180131

Output/Result data: Lag Factor Pattern



CONTRACT	COVERAGE	COST_CATEGORY	RAAP	LFP_TYPE	LFP_SUBCAT	PERIOD	CAL_FREQ	FACTOR	PERIOD_START_DATE	PERIOD_END_DATE
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000001	11	0,7	20180101	20180131
RIC_Q101DT	COV_Q101DT	1010	000001	IN2RE	PH2PI	000002	11	0	20171201	20171231

Flow Modeling Configuration



Rule	Description	Rule Type	State	Rule Grouping	Rule Ordering	Selection
FA	Factors for Additional Incurred	Factors for Additional Incurred	Active			LFP_TYPE = 'IN2RE';LFP_SUBCAT = 'PH2PI'

Rule Lines	Sub View

Input Fields

Lag Factor Pattern Granularity Fields	*CONTRACT, COVERAGE, COST_CATEGORY, CAL_FREQ*
Lag Factor Calendar Frequency Field	*CAL_FREQ*
Date Determinant	*Actual Date of Period*
Day of Month	
Period Field	*PERIOD*
Period Type	*Monthly*
RA Attachment Point	*RAAP*
Hold Back Date Field	*HBD*

Changing Fields

Lag Factor Value	*FACTOR*
Period Start Date Field	*PERIOD_START_DATE*
Period End Date Field	*PERIOD_END_DATE*

1.3.10 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.

The following table explains the key features available:

Key Features	Use
Header	In the header, you define the principal behavior of the File Adapter function. 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.

Key Features	Use
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 me 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 32].

1.3.11 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.

The following table explains the key features available:

Key Features	Use
Header	In the header, you define the principal behavior of the Remote Function Adapter function. Function type: • Finance General Ledger: Each input record is mapped to one posting and posted to FI-GL. The RFC destination entered on environment level is used. • Finance Accounts Payable/Receivable: Each input record is mapped to two postings and posted to accounts payables and accounts receivables. The RFC destination entered on environment level is used. • Finance Extended: This is a combination of the function types Finance General Ledger and Finance Acounts Payable/Receivable • 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. The function name has to be entered. An RFC destination can be entered if the function is remote. • 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. The authoring schema and the stored procedure name have to be entered as well. • 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.

Key Features	Use
Rules	The Rules tab adapts automatically to the header settings. A mapping list is displayed for the function types "Finance General Ledger" and "Finance Accounts Payable/Receivable". This list needs to be maintained for all mandatory fields. The script for the function type "R Script" can be entered directly in an editor. The expected interface is displayed for all other function types.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.11.1 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) |
> |---|
> | 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 configurer 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

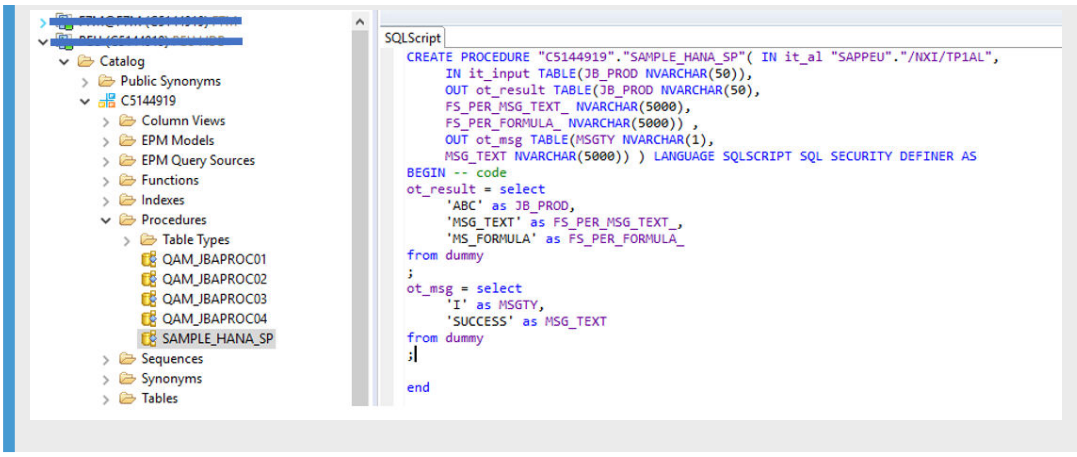
```
CREATE PROCEDURE "<SCHEMA>"."<HANAPROCEDURE>"
(IN it_al "<DEFAULT SCHEMA>"."/NXI/TP1AL",
IN it_input TABLE(JB_PROD NVARCHAR(50)),
OUT ot_result TABLE(JB_PROD NVARCHAR(50), FS_PER_MSG_TEXT_
NVARCHAR(5000), FS_PER_FORMULA_ NVARCHAR(5000)),
OUT ot_msg TABLE(MSGTY NVARCHAR(1), MSG_TEXT NVARCHAR(5000))
) 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;
```

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).



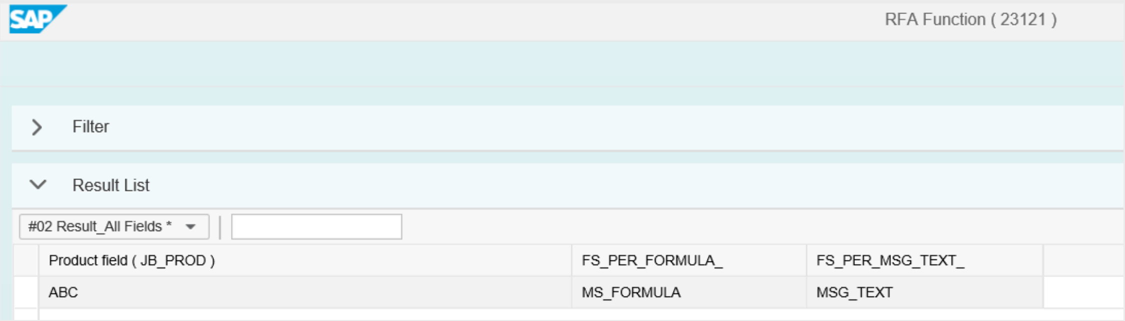
```
CREATE PROCEDURE "C5144919"."SAMPLE_HANA_SP"( IN it_al "SAPPEU"."/NXI/TP1AL",
     IN it_input TABLE(JB_PROD NVARCHAR(50)),
     OUT ot_result TABLE(JB_PROD NVARCHAR(50),
     FS_PER_MSG_TEXT_ NVARCHAR(5000),
     FS_PER_FORMULA_ NVARCHAR(5000)) ,
     OUT ot_msg TABLE(MSGTY NVARCHAR(1),
     MSG_TEXT 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*.

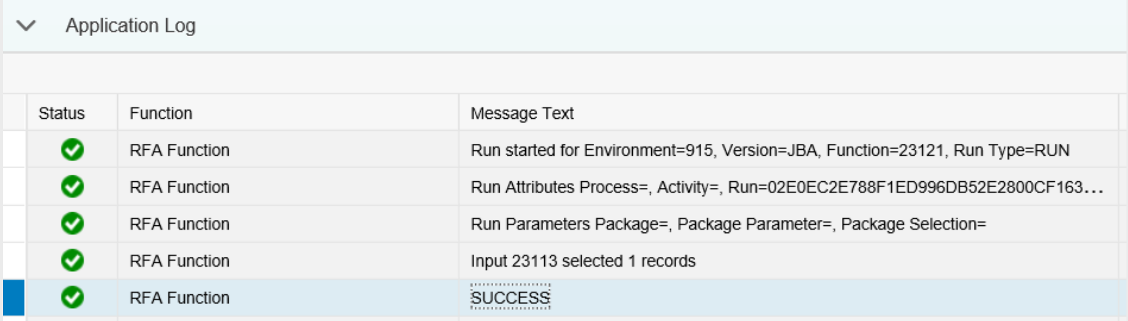
Example

Here is the result of the remote function adapter function after calling a SAP HANA stored procedure to process the input function:



Product field (JB_PROD)	FS_PER_FORMULA_	FS_PER_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

1.3.12 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.

The following table explains the key features available:

Key Feature	Use
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.
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 32].

1.3.12.1 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
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

Channel	Account	Customer	Product	Quantity	Amount
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
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

Channel	Account	Customer	Product	Quantity	Amount
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 (Quantity * MTCA2.Amount [Channel=90AH5] / 2), where Quantity = Quantity from Data Table 1 and MTCA2./ZQA/ZMT/[Channel=90AH5] = 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:
   Quantity * MTCA2.Amount [Channel=90AH5] / 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:
   Quantity * MTCA2.Amount [Channel=90AH5] / 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:
   Quantity * MTCA2.Amount [Channel=90AH5] / 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

1.3.12.2 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
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

Channel	Account	Customer	Product	Quantity	Amount
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	
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	

Chanel	Account	Customer	Product	Quantity	Amount	Premium
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 selection condition: Product = PROD01

Rule 2 selection condition: Product = PROD02

Rule 1 formula: Quantity[1]* Amount [1]

Rule 2 formula: Quantity[-1]* Amount [-1]

> ℹ 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

1.3.13 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.

The following table explains the key features available:

Key Feature	Use
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 32].

1.3.13.1 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

1.3.13.2 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

1.3.14 Model BW

Model BW is a data model function that allows you to define and access BW InfoProviders.

The following table explains the key features available:

Key Feature	Use
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.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.15 Model Table

Model Table is a data model function that allows you to define and access local and remote database tables.

The following table explains the key features available:

Key Feature	Use
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.

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 poulates 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.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.16 Model RDL

Model RDL is a data model function that allows you to access SAP HANA-optimized Result Data Layers of the Finance and Risk Data Platform.

The following table explains the key features available:

Key Feature	Use
Header	In the header, you define the results data area and the result type.
Fields	The fields of the RDL are listed on the Fields tab. You can also exclude fields from read access.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.17 Model View

Model View is a data model function that allows read access to local and remote database tables and views.

The following table explains the key features available:

Key Feature	Use
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.

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.

Related Information

For more information about common aspects of SAP Profitability and Performance Management functions, see Concepts for Key Users [page 32].

1.3.18 Query

Query is a reporting function that allows the output and input of data.

The following table explains the key features available:

Key Feature	Use
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 . Technically, this uses embedded BW 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. Optionally, a key date (constant or parameter) can be specified, which is then propagated to fields with time-dependent master data. • 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. Optionally, a key date (constant or parameter) can be specified, which is then propagated to fields with time-dependent master data. Technically, this uses NetWeaver CDS technology.
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.

Key Feature	Use
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 choosen 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.

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 thisis 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 32].

1.3.19 Model Writer

Model Writer is a processing function that allows you to write in model tables, model BWs and model RDLs. Therefore, the model writer covers all the technical complexity of the different access modes.

The following table explains the key features available:

Key Feature	Use
Header	You first need to define the output function. For this you can use all the model tables, model BWs and model RDLs 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 HANA-based data transfer process, and users who are editing data cannot continue to work during this time. Model Writer Type: • Insert: Datais 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 32].

1.3.20 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.

The following table explains the key features available:

Key Feature	Use
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 Outputtab; 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.

Key Feature	Use
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 32].

1.3.20.1 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
	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	60

Customer	Product	Quantity	Amount
		33	
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

1.3.20.2 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

Input Number	Divisible
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	Dibisible 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 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

Input Number	Divisible by 3
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.

1.3.21 Machine Learning

The Machine Learning function provides a rule type to train and use a time-series forecast model based on input data.

A predictive time-series forecast 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 input fields are available:

- *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, the values of which, are used for the forecast.
- *Excluded Fields*: List of fields not being relevant for the forecast, or the impact of which 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 if 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 output fields are available:

- *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 was automatically assigned by the forecast algorithm.

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 33].

For more information about common aspects of SAP Profitability and Performance Management functions, see Modeling Environment [page 13].

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.

www.sap.com/contactsap

THE BEST RUN SAP