

Oracle® Hyperion Calculation Manager デザイナー・ガイド



リリース 11.2.x

F26627-05

2022 年 10 月

The Oracle logo, consisting of a solid red square with the word "ORACLE" in white, uppercase, sans-serif font centered within it.

ORACLE®

Oracle Hyperion Calculation Manager デザイン・ガイド、リリース 11.2.x

F26627-05

Copyright © 2008, 2022, Oracle and/or its affiliates.

著者: EPM Information Development Team

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

目次

ドキュメントのアクセシビリティについて

ドキュメントのフィードバック

1 Calculation Manager の概要

Calculation Manager について	1-1
Calculation Manager を使用するサービス	1-2
Calculation Manager での操作に権限が及ぼす影響	1-2
Calculation Manager の起動	1-3
Calculation Manager でのオブジェクトの表示	1-3
ビューの列のカスタマイズ	1-4
オブジェクトのフィルタ処理	1-4
問合せを使用したオブジェクトのフィルタ処理	1-5
ルール、コンポーネントまたはテンプレートのグラフィカルな表示	1-5

2 ビジネス・ルールの設計

ビジネス・ルールについて	2-2
ビジネス・ルールの作成	2-3
パラメータとして使用する実行変数の選択	2-6
ビジネス・ルールとしてのパラメータの使用	2-7
ビジネス・ルールを開く	2-8
ビジネス・ルールの編集	2-9
スクリプト・モードでのビジネス・ルールの編集	2-10
ビジネス・ルールの保存	2-12
ビジネス・ルールの保存	2-13
異なる名前でのビジネス・ルールの保存	2-13
ビジネス・ルールの実行	2-14
ビジネス・ルールのコピー	2-14
ビジネス・ルール内の検索	2-15

ビジネス・ルール・スクリプト内のテキスト文字列の検索	2-15
グラフィカル・ビジネス・ルールでのテキストの検索および置換	2-15
ビジネス・ルールの印刷	2-15
ビジネス・ルールの削除	2-16
ビジネス・ルール・コンポーネントでの共通のディメンションの定義	2-17
ビジネス・ルールの分析およびデバッグ	2-17
ビジネス・ルールの分析	2-18
追跡対象のディメンションのメンバーの非表示	2-19
統計情報の取得	2-19
ビジネス・ルールのスクリプトの分析	2-19
ビジネス・ルールのスクリプトの比較	2-20
変更されたビジネス・ルールと保存されたビジネス・ルールの比較	2-21
ビジネス・ルールのデバッグ	2-22
ビジネス・ルール内のコンポーネントの無効化	2-23
Workforce Planning、Capital Asset Planning および Human Capital Management の事前定義済ビジネス・ルールのロード	2-24
ビジネス・ルールへのショートカットの作成	2-24
ビジネス・ルールまたはビジネス・ルールセットのリフレッシュ	2-25
ビジネス・ルールまたはルールセットの使用状況の表示	2-26
オブジェクトの所有者の変更	2-26

3 ビジネス・ルールセットの設計

ビジネス・ルールセットについて	3-1
ビジネス・ルールセットの作成	3-2
ビジネス・ルールセットを開く	3-4
ビジネス・ルールセット内のビジネス・ルールを開く	3-4
ビジネス・ルールのビジネス・ルールセットへの追加	3-5
ビジネス・ルールセットからのビジネス・ルールの除去	3-5
ビジネス・ルールセットの別のアプリケーションへのコピー	3-5
ビジネス・ルールセットの保存	3-6
ビジネス・ルールセットの削除	3-7

4 システム・テンプレートの操作

システム・テンプレートについて	4-1
システム・テンプレートの表示	4-2
システム・テンプレートの使用	4-2
「データのクリア」テンプレートの使用	4-3
「データのコピー」テンプレートの使用	4-4

「金額-単価-レート」テンプレートの使用	4-5
「割当て-レベル間」テンプレートの使用	4-6
「割当て」テンプレートの使用	4-8
「集約」テンプレートの使用	4-10
「SET コマンド」テンプレートの使用	4-12
「通貨換算」テンプレートの使用	4-13
「通貨換算」テンプレートの例	4-15
「データのエクスポート-インポート」テンプレートの使用	4-19
テンプレート・フローの表示	4-20
システム・テンプレートのカスタム・テンプレートとしての保存	4-21
ビジネス・ルールからのシステム・テンプレートの削除	4-22

5 カスタム・テンプレートの操作

カスタム・テンプレートについて	5-1
カスタム・テンプレートの作成	5-2
グラフィカル・カスタム・テンプレートの作成	5-2
UpperPOV 設計時プロンプトを使用するグラフィカル・テンプレートの作成例	5-4
スクリプト・カスタム・テンプレートの作成	5-5
カスタム・テンプレート用の設計時プロンプトの作成	5-6
属性 DTP	5-6
ブール DTP	5-7
ディメンション間 DTP	5-8
条件 DTP	5-9
ディメンション DTP	5-10
ディメンション(複数) DTP	5-11
メンバー DTP	5-12
メンバー(複数)DTP	5-14
メンバー範囲 DTP	5-15
数値 DTP	5-16
パスワード DTP	5-17
制限リスト DTP	5-18
区切り文字 DTP	5-19
文字列 DTP	5-19
設計時プロンプトのステップの作成	5-20
設計時プロンプトの依存関係の定義	5-21
ディメンション間、ディメンション、ディメンション(複数)、メンバーおよびメンバー範囲の設計時プロンプトに対するディメンション制限の定義	5-22
設計時プロンプト内のテキストの検索および置換	5-23

カスタム・テンプレートを開く	5-24
カスタム・テンプレートのリフレッシュ	5-24
カスタム・テンプレートの使用状況の表示	5-24
カスタム・テンプレートのコピーおよび貼付け	5-25
カスタム・テンプレートの削除	5-25
グラフィカル・カスタム・テンプレート内のテキストの検索および置換	5-26

6 コンポーネントを使用したビジネス・ルールおよびテンプレートの設計

コンポーネントについて	6-2
式コンポーネント	6-3
式コンポーネントについて	6-3
式コンポーネントの作成	6-3
式コンポーネントの設計	6-4
条件ビルダーを使用した条件文の作成	6-9
式の文のロギング情報の入力(Financial Management ユーザーのみ)	6-11
式の文のコメントの入力	6-12
式コンポーネントを開く	6-13
式コンポーネントの編集	6-13
式コンポーネントの削除	6-14
式コンポーネントのコピーおよび貼付け	6-14
スクリプト・コンポーネント	6-15
スクリプト・コンポーネントの作成	6-15
スクリプト・コンポーネントの設計	6-16
スクリプト・コンポーネントを開く	6-20
スクリプト・コンポーネントの編集	6-20
スクリプト・コンポーネントの削除	6-21
スクリプト・コンポーネントのコピーおよび貼付け	6-22
条件コンポーネント	6-22
条件コンポーネントについて	6-23
条件コンポーネントの作成	6-23
条件コンポーネントを開く	6-24
条件コンポーネントの編集	6-25
条件コンポーネントの削除	6-25
条件コンポーネントのコピーおよび貼付け	6-26
メンバー・ブロック・コンポーネント	6-27
メンバー・ブロック・コンポーネントについて	6-27
メンバー・ブロック・コンポーネントの作成	6-27
メンバー・ブロック・コンポーネントを開く	6-28

メンバー・ブロック・コンポーネントの編集	6-28
メンバー・ブロック・コンポーネントの削除	6-29
メンバー・ブロック・コンポーネントのコピーおよび貼付け	6-29
メンバー範囲コンポーネント	6-30
メンバー範囲コンポーネントについて	6-30
メンバー範囲コンポーネントの作成	6-31
メンバー範囲コンポーネントを開く	6-33
メンバー範囲コンポーネントの編集	6-34
メンバー範囲コンポーネントの削除	6-34
メンバー範囲コンポーネントのコピーおよび貼付け	6-35
データ範囲コンポーネント(Financial Management ユーザーのみ)	6-35
データ範囲コンポーネントについて(Financial Management ユーザーのみ)	6-36
データ範囲コンポーネントの作成(Financial Management ユーザーのみ)	6-36
データ範囲コンポーネントを開く(Financial Management ユーザーのみ)	6-39
データ範囲コンポーネントの編集(Financial Management ユーザーのみ)	6-39
データ範囲コンポーネントの削除(Financial Management ユーザーのみ)	6-39
データ範囲コンポーネントのコピーおよび貼付け(Financial Management ユーザーのみ)	6-40
セル・テキスト範囲コンポーネント(Financial Management ユーザーのみ)	6-40
セル・テキスト範囲コンポーネントについて(Financial Management ユーザーのみ)	6-41
セル・テキスト範囲コンポーネントの作成(Financial Management ユーザーのみ)	6-41
セル・テキスト範囲コンポーネントを開く(Financial Management ユーザーのみ)	6-43
セル・テキスト範囲コンポーネントの編集(Financial Management ユーザーのみ)	6-44
セル・テキスト範囲コンポーネントの削除(Financial Management ユーザーのみ)	6-44
セル・テキスト範囲コンポーネントのコピーおよび貼付け(Financial Management ユーザーのみ)	6-44
固定ループ・コンポーネント	6-45
固定ループ・コンポーネントについて	6-45
固定ループ・コンポーネントの作成	6-45
固定ループ・コンポーネントを開く	6-47
固定ループ・コンポーネントの編集	6-47
固定ループ・コンポーネントの削除	6-48
固定ループ・コンポーネントのコピーおよび貼付け	6-49
メタデータ・ループ・コンポーネント	6-50
メタデータ・ループ・コンポーネントについて	6-50
メタデータ・ループ・コンポーネントの作成	6-50
メタデータ・ループ・コンポーネントを開く	6-51
メタデータ・ループ・コンポーネントの削除	6-52
メタデータ・ループ・コンポーネントのコピーおよび貼付け	6-52

DTP 割当てコンポーネント	6-53
DTP 割当てコンポーネントについて	6-53
DTP 割当てコンポーネントの作成	6-53
DTP 割当てコンポーネントを開く	6-55
DTP 割当てコンポーネントの編集	6-55
DTP 割当てコンポーネントの削除	6-55
DTP 割当てコンポーネントのコピーおよび貼付け	6-55
DTP 割当てコンポーネントの設計時プロンプト関数の使用	6-56
設計時プロンプト関数について	6-57
@Compare	6-57
@Compliment	6-57
@Concat	6-58
@DenseMember	6-59
@DimMember	6-60
@DimName	6-60
@DimType	6-61
@EndsWith	6-62
@Intersect	6-62
@Member	6-63
@Notin	6-64
@Plandim	6-65
@Quote	6-66
@RemoveQuote	6-66
@ReplaceAll	6-67
@ReplaceFirst	6-68
@SmartListFromIndex	6-68
@SmartListFromValue	6-70
@SparseMember	6-71
@StartsWith	6-72
@SubString	6-72
@ToLowerCase	6-73
@ToUpperCase	6-74
@Trim	6-74
@Union	6-75
スクリプトおよび式コンポーネントの共有	6-76
スクリプトおよび式コンポーネントの共有について	6-76
式およびスクリプト・コンポーネントの共有から非共有への変更	6-76
式およびスクリプト・コンポーネントの非共有から共有への変更	6-77
コンポーネントのコピー	6-78
ビジネス・ルール・コンポーネントの子のコピーおよび貼付け	6-78

ビジネス・ルールの式またはスクリプト・コンポーネントへの参照のコピーおよび貼付け	6-79
コンポーネントの保存	6-80
コンポーネントの保存	6-80
異なる名前での式およびスクリプト・コンポーネントの保存	6-80
式およびスクリプト・コンポーネントのリフレッシュ	6-81
式およびスクリプト・コンポーネントの使用の表示	6-82
フロー・チャート内でのコンポーネントの操作	6-82
フロー・チャート内でのコンポーネントの操作について	6-83
フロー・チャート内でのコンポーネントの縮小および展開	6-83
コンポーネントのフロー・チャートからの除去	6-83
フロー・チャート内でのコンポーネントのコピーおよび貼付け	6-84
フロー・チャート内でのコンポーネントへの参照のコピーおよび貼付け	6-84
フロー・チャート内でのコンポーネント・グループのコピーおよび貼付け	6-85

7 集約ストレージ・コンポーネントを使用したビジネス・ルールの設計

集約ストレージ・コンポーネントを使用したビジネス・ルールの設計について	7-1
視点コンポーネントの操作	7-2
視点コンポーネントの作成	7-2
視点コンポーネントの編集	7-7
割当てコンポーネントの操作	7-8
割当てコンポーネントの作成	7-8
割当てコンポーネントの編集	7-12
視点または割当てコンポーネントを開く	7-13
視点または割当てコンポーネントの削除	7-13
視点または割当てコンポーネントのコピーおよび貼付け	7-14
視点または割当てコンポーネントの保存	7-14
集約ストレージ式コンポーネントの操作	7-14
集約ストレージ式コンポーネントの作成	7-15
集約ストレージ式コンポーネントを開く	7-17
集約ストレージ式コンポーネントの編集	7-18
集約ストレージ式コンポーネントの削除	7-18
集約ストレージ式コンポーネントのコピーおよび貼付け	7-18
別のアプリケーションまたはデータベースへの集約ストレージ式コンポーネントのコピー	7-19
集約ストレージ式コンポーネントの使用状況の表示	7-20

8 メンバーの選択、変数、関数、スマート・リストおよび Planning 数式を使用したコンポーネントの設計

メンバー選択、変数、関数およびスマート・リストについて	8-2
コンポーネントへのメンバーおよび関数の追加	8-2
コンポーネントへのメンバーおよび関数の追加について	8-3
1つ以上のディメンションから1つのディメンションへのメンバーまたは関数の追加	8-3
1つ以上のディメンションから1つのディメンションへの複数のメンバーまたは関数の追加	8-12
コンポーネントからのメンバーおよび関数の除去	8-20
メンバー・セレクタ内でのメンバーの検索	8-20
メンバー・セレクタでの名前、別名またはプロパティによるメンバーの検索	8-21
変数の操作	8-22
変数について	8-22
変数の作成	8-23
数値変数の変数値の入力	8-25
文字列変数の変数値の入力	8-26
配列変数の値の入力	8-27
メンバー範囲変数の値の入力	8-28
ディメンション間変数の変数値の入力	8-29
ディメンション変数の変数値の入力	8-30
メンバー変数またはメンバー(複数)変数の変数値の入力	8-31
パーセント変数の変数値の入力	8-32
整数変数の値の入力	8-32
数値としての文字列変数の変数値の入力	8-33
数値としての日付変数の変数値の入力	8-33
データ範囲変数の値の入力(Financial Management ユーザーのみ)	8-34
ブール変数の値の入力(Financial Management ユーザーのみ)	8-34
ログ情報変数の値の入力(Financial Management ユーザーのみ)	8-34
実行時プロンプト変数の入力	8-35
変数の選択	8-36
変数の編集	8-37
変数の削除	8-37
変数のリフレッシュ	8-38
変数のコピー	8-38
変数デザイナーでのテキストの検索と置換	8-39
変数の使用状況の表示	8-41
Financial Management アプリケーションへの事前定義済システム変数のロード (Financial Management ユーザーのみ)	8-41
関数の操作	8-42

Calculation Manager でサポートされている Essbase 関数	8-44
カスタム関数の操作	8-45
カスタム関数について	8-46
年パラメータがあるカスタム関数の使用	8-46
日付パラメータがある@CalcMgrExcel カスタム関数	8-46
ビットに関する関数	8-47
@CalcMgrBitAnd	8-47
@CalcMgrBitOR	8-47
@CalcMgrBitExOR	8-47
@CalcMgrBitExBoolOR	8-48
@CalcMgrBitCompliment	8-48
@CalcMgrBitShiftLeft	8-48
@CalcMgrBitShiftRight	8-48
@CalcMgrBitUnsignedShiftRight	8-49
カウンタの関数	8-49
@CalcMgrCounterAddNumber	8-49
@CalcMgrCounterAddText	8-50
@CalcMgrCounterClear	8-50
@CalcMgrCounterClearAll	8-50
@CalcMgrCounterClearKey	8-50
@CalcMgrCounterDecrement	8-51
@CalcMgrCounterDecrementKey	8-51
@CalcMgrCounterGetKeyNumber	8-51
@CalcMgrCounterGetKeyText	8-51
@CalcMgrCounterGetNumber	8-52
@CalcMgrCounterGetText	8-52
@CalcMgrCounterIncrement	8-52
@CalcMgrCounterIncrementKey	8-52
@CalcMgrCounterUpdate	8-53
@CalcMgrCounterUpdateNumber	8-53
@CalcMgrCounterUpdateNumberText	8-53
@CalcMgrCounterUpdateText	8-53
日付/時刻の関数	8-54
@CalcMgrAddDate	8-55
@CalcMgrAddDatePart	8-56
@CalcMgrAddDays	8-56
@CalcMgrAddMonths	8-56
@CalcMgrAddWeeks	8-56
@CalcMgrAddYears	8-57
@CalcMgrDateDiff	8-57
@CalcMgrDateToExcel	8-57

@CalcMgrDatesToExcel	8-57
@CalcMgrDateTimeToExcel	8-58
@CalcMgrDateTimesToExcel	8-58
@CalcMgrDateToString	8-58
@CalcMgrDaysBetween	8-58
@CalcMgrDaysDiff	8-59
@CalcMgrDiffDate	8-59
@CalcMgrExcelADD	8-60
@CalcMgrExcelDATE	8-60
@CalcMgrExcelDATEDIF	8-60
@CalcMgrExcelDAYOFYEAR	8-61
@CalcMgrExcelDAYS360	8-61
@CalcMgrExcelDAYSINMONTH	8-62
@CalcMgrExcelEOMONTH	8-62
@CalcMgrExcelHOUR	8-62
@CalcMgrExcelMINUTE	8-63
@CalcMgrExcelMONTH	8-63
@CalcMgrExcelNETWORKDAYS	8-63
@CalcMgrExcelSECOND	8-64
@CalcMgrExcelToDate	8-64
@CalcMgrExcelToDateTime	8-65
@CalcMgrExcelWEEKNUM	8-65
@CalcMgrExcelWEEKDAY	8-65
@CalcMgrExcelWORKDAY	8-66
@CalcMgrExcelYEAR	8-66
@CalcMgrExcelYEARFRAC	8-67
@CalcMgrGetCurrentDate	8-67
@CalcMgrGetCurrentDateTZ	8-67
@CalcMgrGetCurrentDateTime	8-68
@CalcMgrGetCurrentDateTimeTZ	8-68
@CalcMgrGetCustomDate	8-68
@CalcMgrGetCustomDateTime	8-68
@CalcMgrGetDatePart	8-69
@CalcMgrGetDateTimePart	8-69
@CalcMgrGetDay	8-69
@CalcMgrGetDayOfYear	8-69
@CalcMgrGetFormattedDate	8-70
@CalcMgrGetMaxDaysInMonth	8-70
@CalcMgrGetMonth	8-70
@CalcMgrGetStringFormattedDateTime	8-70
@CalcMgrGetWeekOfMonth	8-71

@CalcMgrGetWeekOfYear	8-71
@CalcMgrGetYear	8-71
@CalcMgrIsLeapYear	8-71
@CalcMgrMonthsBetween	8-72
@CalcMgrMonthsDiff	8-72
@CalcMgrRollDate	8-72
@CalcMgrRollDay	8-73
@CalcMgrRollMonth	8-73
@CalcMgrRollYear	8-73
@CalcMgrWeeksBetween	8-73
@CalcMgrWeeksDiff	8-74
@CalcMgrYearsBetween	8-74
@CalcMgrYearsDiff	8-74
財務関数	8-75
@CalcMgrExcelACCRINT	8-76
@CalcMgrExcelACCRINTM	8-76
@CalcMgrExcelAMORDEGRC	8-77
@CalcMgrExcelAMORLINC	8-77
@CalcMgrExcelCOUPDAYBS	8-77
@CalcMgrExcelCOUPDAYS	8-78
@CalcMgrExcelCOUPDAYSNC	8-78
@CalcMgrExcelCOUPNCD	8-79
@CalcMgrExcelCOUPNUM	8-79
@CalcMgrExcelCOUPPCD	8-79
@CalcMgrExcelCUMIPMT	8-80
@CalcMgrExcelCUMPRINC	8-80
@CalcMgrExcelDB	8-80
@CalcMgrExcelDDB	8-81
@CalcMgrExcelDISC	8-81
@CalcMgrExcelDOLLARDE	8-81
@CalcMgrExcelDOLLARFR	8-81
@CalcMgrExcelDURATION	8-82
@CalcMgrExcelEFFECT	8-82
@CalcMgrExcelFV	8-82
@CalcMgrExcelFVSCHEDULE	8-83
@CalcMgrExcelIMDURATION	8-83
@CalcMgrExcelINTRATE	8-83
@CalcMgrExcelIPMT	8-84
@CalcMgrExcelIRR	8-84
@CalcMgrExcelISPMT	8-84
@CalcMgrExcelMIRR	8-85

@CalcMgrExcelNPER	8-85
@CalcMgrExcelNPV	8-85
@CalcMgrExcelPPMT	8-85
@CalcMgrExcelPRICE	8-86
@CalcMgrExcelPRICEDISC	8-86
@CalcMgrExcelPRICEMAT	8-86
@CalcMgrExcelPV	8-87
@CalcMgrExcelRATE	8-87
@CalcMgrExcelRECEIVED	8-87
@CalcMgrExcelSLN	8-88
@CalcMgrExcelSYD	8-88
@CalcMgrExcelTBILLEQ	8-88
@CalcMgrExcelTBILLPRICE	8-89
@CalcMgrExcelTBILLYIELD	8-89
@CalcMgrExcelXIRR	8-89
@CalcMgrExcelXNPV	8-90
@CalcMgrExcelYIELD	8-90
@CalcMgrExcelYIELDDISC	8-90
@CalcMgrExcelYIELDMAT	8-91
ログ関数	8-91
@CalcMgrLog	8-91
@CalcMgrLogText	8-93
数学関数	8-93
@CalcMgrExcelCEILING	8-94
@CalcMgrExcelCOMBIN	8-94
@CalcMgrExcelEVEN	8-94
@CalcMgrExcelFACT	8-94
@CalcMgrExcelFLOOR	8-95
@CalcMgrExcelGCD	8-95
@CalcMgrExcelLCM	8-95
@CalcMgrExcelMROUND	8-95
@CalcMgrExcelMULTINOMIAL	8-96
@CalcMgrExcelODD	8-96
@CalcMgrExcelPOWER	8-96
@CalcMgrExcelPRODUCT	8-96
@CalcMgrExcelROUNDDOWN	8-97
@CalcMgrExcelROUNDUP	8-97
@CalcMgrExcelSQRT	8-97
@CalcMgrExcelSQRTPI	8-97
@CalcMgrExcelSUMPRODUCT	8-97
@CalcMgrExcelSUMSQ	8-98

MaxL 関数	8-98
@CalcMgrExecuteEncryptMaxLFile	8-98
@CalcMgrExecuteMaxLScript	8-98
MaxL の暗号化の使用	8-99
@CalcMgrExecuteEncryptMaxLFile を使用した MaxL スクリプト・ファイルの実行	8-99
@CalcMgrExecuteMaxLScript を使用した MaxL スクリプトの実行	8-100
MDX 関数	8-104
@CalcMgrMDXDataCopy	8-104
@CalcMgrMDXDataCopyRTP	8-106
@CalcMgrMDXDataCopyServer	8-108
@CalcMgrMDXExport	8-108
@CalcMgrMDXExportRTP	8-109
統計関数	8-110
@CalcMgrExcelAVEDEV	8-110
@CalcMgrExcelDEVSQ	8-111
@CalcMgrExcelLARGE	8-111
@CalcMgrExcelMEDIAN	8-111
@CalcMgrExcelSMALL	8-111
@CalcMgrExcelSTDEV	8-112
@CalcMgrExcelVAR	8-112
@CalcMgrExcelVARP	8-112
文字列関数	8-112
@CalcMgrCompare	8-113
@CalcMgrConcat	8-114
@CalcMgrDecimalFormat	8-114
@CalcMgrDoubleFromString	8-114
@CalcMgrDoubleToString	8-114
@CalcMgrDQuote	8-115
@CalcMgrEndsWith	8-115
@CalcMgrFindFirst	8-115
@CalcMgrFindLast	8-115
@CalcMgrFormatDouble	8-115
@CalcMgrGetListCount	8-116
@CalcMgrGetListItem	8-116
@CalcMgrIndexOf	8-116
@CalcMgrIntegerToString	8-117
@CalcMgrLastIndexOf	8-117
@CalcMgrLowercase	8-117
@CalcMgrMatches	8-117
@CalcMgrMessageFormat	8-117
@CalcMgrPadText	8-118

@CalcMgrUppercase	8-118
@CalcMgrRemoveQuotes	8-118
@CalcMgrRemovedQuotes	8-118
@CalcMgrRemoveSQuotes	8-119
@CalcMgrReplaceAll	8-119
@CalcMgrReplaceFirst	8-119
@CalcMgrSortAndReturn	8-119
@CalcMgrSortList	8-120
@CalcMgrSortValues	8-120
@CalcMgrSplit	8-120
@CalcMgrSQuote	8-120
@CalcMgrStartsWith	8-121
@CalcMgrStringsToString	8-121
@CalcMgrSubstring	8-121
@CalcMgrTextLength	8-121
@CalcMgrTrim	8-122
コンポーネントへの関数の挿入	8-122
スマート・リストの操作	8-124
スマート・リストの挿入	8-124
Planning 数式の操作	8-124
スマートリスト	8-125
ディメンション	8-125
Planning ユーザー変数	8-126
期間	8-126
Period(periodName)	8-126
NumberOfPeriodsinYear および NumberOfYears	8-127
シナリオ	8-128
相互参照	8-132
CrossRef(accountName)	8-132
CrossRef(accountName, prefix)	8-133
CrossRef(accountName, prefix, true)	8-133
Workforce キューブ年次累計	8-134
CYTD(memberName)	8-134
CYTD(memberName, calTpIndexName, fiscalTpIndexName)	8-134
文字列の ID の取得	8-135

9 検証およびデプロイ

システム・ビューからのビジネス・ルール、ビジネス・ルールセット、式およびスクリプト・コンポーネントの検証	9-1
ルール・デザイナーからのビジネス・ルールの検証	9-2

ビジネス・ルールおよびビジネス・ルールセットのデプロイ	9-3
ビジネス・ルールおよびビジネス・ルールセットのデプロイについて	9-4
ビジネス・ルールおよびビジネス・ルールセットをデプロイ可能にする、およびデプロイ可能にしない	9-5
デプロイメント・ビューからのビジネス・ルールおよびビジネス・ルールセットのデプロイ	9-5
ルール・デザイナーまたはルールセット・デザイナーからのビジネス・ルールまたはビジネス・ルールセットのデプロイ	9-7
ショートカットを含むビジネス・ルールのデプロイ	9-7
Planning に表示されるデプロイ済のビジネス・ルールの指定	9-8

10 ビジネス・ルールの起動

ビジネス・ルールの起動について	10-1
システム・ビューおよびルール・デザイナーからの Essbase ビジネス・ルールの起動	10-2
コマンド・ライン・ランチャを使用した Essbase ビジネス・ルールの起動	10-2
コマンド・ライン・ランチャの例および引数	10-2
Windows で bat ファイルを使用したコマンド・ライン・ランチャの実行	10-4
ルール・デザイナーからの Planning ビジネス・ルールの起動およびログの表示	10-4

11 ビジネス・ルール、ビジネス・ルールセット、テンプレート、式およびスクリプト・コンポーネントのエクスポートおよびインポート

エクスポートおよびインポートについて	11-1
ビジネス・ルール、ビジネス・ルールセット、テンプレート、式およびスクリプト・コンポーネントのエクスポート	11-2
アプリケーションのエクスポート	11-3
ファイルへのログ・メッセージのエクスポート	11-3
ルール、ルールセット、テンプレート、式およびスクリプトのインポート	11-4
グラフィカル・ビジネス・ルールとしての Essbase 計算スクリプトのインポート	11-5

12 Planning アプリケーションのための Essbase サーバー、アプリケーションおよびデータベースの管理

データベース・プロパティの操作	12-2
データベース・プロパティの表示および編集	12-2
一般データベース・プロパティ	12-3
ディメンション・プロパティ	12-4
統計プロパティ	12-5
集約ストレージ・アプリケーションの統計	12-5
ブロック・ストレージ・アプリケーションの統計	12-6

トランザクション・プロパティ	12-8
変更プロパティ	12-9
通貨プロパティ	12-9
キャッシュのプロパティ	12-10
MaxL スクリプトの作成および起動	12-11
データベース・オブジェクトのロックの削除	12-11
アプリケーションの開始と停止	12-12
データベースの開始と停止	12-12
アプリケーション・ログ・ファイルの削除	12-13
データベースの再構築	12-14
アウトラインの確認	12-15
データベース設定を最適化するためのシミュレーションの作成	12-16
データベースからのデータのクリア	12-18
集約ストレージ・アプリケーションからのデータのクリア	12-18
ブロック・ストレージ・アプリケーションからのデータのブロックのクリア	12-19
ブロック・ストレージ・アプリケーションのロケーション別名の操作	12-20
ブロック・ストレージ・アプリケーションのロケーション別名の操作	12-20
ロケーション別名のリストの表示	12-20
ロケーション別名のエクスポート	12-20
ロケーション別名の作成	12-21
ロケーション別名の削除	12-21
集約ストレージ・データベースでの問合せトラッキングの使用	12-21
集約ストレージ・データベース・アウトラインの圧縮	12-22
レベル 0 データのインポートとエクスポート	12-23
ASO キューブからのレベル 0 データのインポート	12-23
ASO キューブからのレベル 0 データのエクスポート	12-24
BSO キューブからのレベル 0 データのインポート	12-24
BSO キューブからのレベル 0 データのエクスポート	12-25
増分データ・スライスのマージ	12-25
データの集約	12-26
集約プロセスの実行	12-27
増分データ・スライスのマージおよびゼロ値のセルの削除	12-27
問合せトラッキングの有効化	12-27
問合せを作成するアクションの実行	12-28
問合せトラッキングを使用した集約の実行	12-28
要求の管理	12-29
Planning ドリル・スルー定義の追加	12-31

A 管理者機能の使用

Calculation Manager への移行

A-1

ドキュメントのアクセシビリティについて

オラクル社のアクセシビリティについての詳細情報は、Oracle Accessibility Program の Web サイト <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc> を参照してください。

Oracle サポートへのアクセス

サポートをご契約のお客様には、My Oracle Support を通して電子支援サービスを提供しています。詳細は、<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info>、聴覚に障害があるお客様は <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> を参照してください。

ドキュメントのフィードバック

このドキュメントに対するフィードバックを送るには、Oracle Help Center トピックのページの下部にあるフィードバック・ボタンをクリックします。epmdoc_ww@oracle.com に電子メールを送信することもできます。

1

Calculation Manager の概要

次も参照:

- [Calculation Manager について](#)
Oracle Hyperion Calculation Manager を使用すると、ビジネス上の問題を解決する計算を作成、検証、デプロイおよび起動できます。
- [Calculation Manager を使用するサービス](#)
いくつかの Enterprise Performance Management Cloud サービスで Oracle Hyperion Calculation Manager が使用されます。
- [Calculation Manager での操作に権限が及ぼす影響](#)
Oracle Hyperion Shared Services で割り当てられる役割によって、操作できるルール、ルールセット、コンポーネントおよびテンプレートと、それらに対して Oracle Hyperion Calculation Manager で何を行えるかが決まります。
- [Calculation Manager の起動](#)
Oracle Hyperion Calculation Manager を起動して、ルール、ルールセット、コンポーネントおよびテンプレートを作成します。
- [Calculation Manager でのオブジェクトの表示](#)
ビューを使用すると、Oracle Hyperion Calculation Manager オブジェクトを様々なコンテキストで表示できます。
- [ビューの列のカスタマイズ](#)
各ビューに表示される列および列が表示される順序をカスタマイズできます。
- [オブジェクトのフィルタ処理](#)
システム・ビューまたはフィルタ・ビューで Oracle Hyperion Calculation Manager のオブジェクトをフィルタ処理します。
- [問合せを使用したオブジェクトのフィルタ処理](#)
フィルタ・ビューでオブジェクトをフィルタ処理した後、問合せを使用して表示するオブジェクトを絞り込むことができます。
- [ルール、コンポーネントまたはテンプレートのグラフィカルな表示](#)
ルール、コンポーネントおよびテンプレートは、ルール・デザイナーおよびテンプレート・デザイナー内のフロー・チャートでグラフィカルに表示されます。

Calculation Manager について

Oracle Hyperion Calculation Manager を使用すると、ビジネス上の問題を解決する計算を作成、検証、デプロイおよび起動できます。

計算には次のようなオブジェクトを作成できます。

- **ルール** - コンポーネント、テンプレートおよび他のルールを含むオブジェクト
- **ルールセット** - 同時または順次計算される 2 つ以上のビジネス・ルールが含まれるオブジェクト

- **コンポーネント** - 式、スクリプト、条件、メンバーとデータ範囲、固定ループ、および設計時プロンプトを含むオブジェクト。(コンポーネントはデプロイ可能ではありません。)
- **テンプレート** - 計算または計算のセットを実行するためにビジネス・ルールで使用できるオブジェクト

Calculation Manager を使用するサービス

いくつかの Enterprise Performance Management Cloud サービスで Oracle Hyperion Calculation Manager が使用されます。

- Enterprise Planning and Budgeting Cloud
- Planning and Budgeting Cloud
- Financial Consolidation and Close Cloud

Calculation Manager での操作に権限が及ぼす影響

Oracle Hyperion Shared Services で割り当てられる役割によって、操作できるルール、ルールセット、コンポーネントおよびテンプレートと、それらに対して Oracle Hyperion Calculation Manager で何を行えるかが決まります。

表 1-1 Calculation Manager での役割とその影響

アプリケーションの状態	役割	Calculation Manager で行うことができる操作
未デプロイ	<ul style="list-style-type: none"> • Calculation Manager グローバル管理者 • Oracle Hyperion Financial Management Calculation Manager 管理者 • Oracle Hyperion Planning Calculation Manager 管理者 • Oracle Essbase 管理者 <p>Essbase の管理者役割は、Essbase から継承されます。Essbase Calculation Manager の管理者役割はありません。</p>	<p>ルールと他のオブジェクトの編集、削除、デプロイおよび起動</p> <p>アプリケーション、ルールおよび他のオブジェクトをデプロイしたら、次を行うことができます:</p> <ul style="list-style-type: none"> • Financial Management での Financial Management ビジネス・ルールセットの起動 • Planning または Calculation Manager での Planning ビジネス・ルールの起動 • Oracle Essbase Administration Services または Calculation Manager での Essbase ビジネス・ルールの起動 • General Ledger での Oracle Fusion General Ledger ビジネス・ルールの起動
デプロイ済	<p>デプロイされたアプリケーションのグローバル役割:</p> <ul style="list-style-type: none"> • Financial Management 管理者 • Planning 管理者 • Essbase 管理者 	<p>そのアプリケーションに属するビジネス・ルールの作成、編集および削除。また、アプリケーションをデプロイすることもできます。</p>
デプロイ済	<p>アプリケーション固有の役割:</p> <ul style="list-style-type: none"> • Financial Management 計算デザイナー • Planning 対話型ユーザー • Essbase 計算書込み権限 	<p>自分のルールの作成、編集、削除、および他のユーザーのルールの表示。</p> <p>ノート: 他のユーザーのルールは、そのユーザーからルールの所有権を割り当てられている場合を除き、編集できません。</p>

表 1-1 (続き) Calculation Manager での役割とその影響

アプリケーションの状態	役割	Calculation Manager で行うことができる操作
デプロイ済	アプリケーション固有の役割: <ul style="list-style-type: none"> Financial Management 参照者 Planning 表示ユーザー 	ルールの表示



ノート:

役割とセキュリティの詳細は、*Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイド*を参照してください

Calculation Manager の起動

Oracle Hyperion Calculation Manager を起動して、ルール、ルールセット、コンポーネントおよびテンプレートを作成します。

Calculation Manager を起動するには:

1. Oracle Hyperion Enterprise Performance Management Workspace を開きます。
2. 「ナビゲート」、「管理」、「Calculation Manager」の順に選択します。

Calculation Manager でのオブジェクトの表示

ビューを使用すると、Oracle Hyperion Calculation Manager オブジェクトを様々なコンテキストで表示できます。

ビューでは、**ビュー・メニュー**のオプションを使用して表示される列およびその表示順をカスタマイズできます。

Calculation Manager を起動すると、**システム・ビュー**が自動的に表示されます。ビューを切り替えるには、「**ビューの選択**」の隣にあるドロップダウンからビューを選択します。

Calculation Manager には、次のビューが含まれています。

- **システム・ビュー** - Calculation Manager を起動したときに表示されるデフォルト・ビュー。アクセスできるすべてのアプリケーションおよびオブジェクトがリストされます。アクセス権は Oracle Hyperion Shared Services で割り当てられている役割によって決定されます。アクセス権はアプリケーション・ベースで割り当てられます。
- **カスタム・ビュー** - フォルダを作成し、そのフォルダにオブジェクトを追加して必要なビューを作成します。

カスタム・ビューでフォルダを作成するには:

1. 右側のペインでアプリケーションを右クリックし、「**新規**」、「**フォルダ**」の順に選択します。
2. 「**新規フォルダ**」でフォルダ名を入力し、「**OK**」をクリックします。

3. 「既存のオブジェクト」 ペインからフォルダにオブジェクトをドラッグ・アンド・ドロップして、オブジェクトをフォルダに追加します。
- **デプロイメント・ビュー** - アプリケーション・タイプおよびアプリケーション別に、デプロイ可能なルールおよびルールセットとそのデプロイおよび検証ステータスがリストされます。デプロイ可能とするルールおよびルールセットを選択し、オブジェクトをアプリケーションにデプロイできます。
アプリケーションでの1つ以上のルールおよびルールセットのデプロイは部分デプロイメントと呼ばれ、アプリケーションでのすべてのルールおよびルールセットのデプロイは完全デプロイメントと呼ばれます。
- **フィルタ・ビュー** - 表示されるオブジェクトをフィルタします。
「フィルタ」ダイアログ・ボックスでフィルタ・オプションを定義し、問合せを使用してさらに詳しくフィルタできます。

ビューの列のカスタマイズ

各ビューに表示される列および列が表示される順序をカスタマイズできます。

ビューの列をカスタマイズするには、次のいずれかを行います。

- 「ビュー」、「列」の順に選択し、表示する列を選択します。「ビュー」、「列の順序変更」の順に選択して列を並べ替え、列順序を選択します。
- 「ビュー」、「列」、「列の管理」の順に選択します。「列の管理」ダイアログ・ボックスで、表示する列とその表示順を選択します。

ノート:

Oracle Financials Cloud では、列に新しい"ロック解除"機能があります。この列はデフォルトで表示されません。ロック解除により、管理者は別の管理者がロックしているオブジェクトのロックを解除できます。

オブジェクトのフィルタ処理

システム・ビューまたはフィルタ・ビューで Oracle Hyperion Calculation Manager のオブジェクトをフィルタ処理します。

アプリケーション・タイプ、アプリケーション、計算タイプ、プラン・タイプ、データベース、オブジェクト・タイプ(ビジネス・ルール、ビジネス・ルールセット、式およびスクリプト・コンポーネント、およびテンプレート)、およびデプロイまたは検証ステータスでオブジェクトをフィルタできます。

オブジェクトをフィルタするには:

1. システム・ビューまたはフィルタ・ビューで  をクリックします。
2. 「フィルタ」ダイアログ・ボックスで、要求された情報を入力し、「OK」をクリックします。

問合せを使用したオブジェクトのフィルタ処理

フィルタ・ビューでオブジェクトをフィルタ処理した後、問合せを使用して表示するオブジェクトを絞り込むことができます。

問合せを使用してオブジェクトをフィルタ処理するには:

1. **フィルタ・ビュー**で  をクリックし、「**フィルタ**」ダイアログ・ボックスに情報を入力します。
2.  をクリックして列の上にテキスト・ボックスを表示します。
3. 列の上のテキスト・ボックスに問合せを入力し、さらにフィルタ処理します。

ワイルド・カードまたは部分的なテキスト文字列を使用できません。プラン・タイプを検索するには、最初の語の始まりの文字を入力します。たとえば、プラン・タイプの名前が"Plan1"の場合、問合せに"la"と入力すると、"La"で始まるオブジェクトのみが表示されます。この例で希望する結果を得るには、"Pl"と入力し、"Pl"で始まるプラン・タイプをすべて表示します。

ノート:

問合せを使用して変数デザイナーでオブジェクトをフィルタ処理することもできます。

ルール、コンポーネントまたはテンプレートのグラフィカルな表示

ルール、コンポーネントおよびテンプレートは、ルール・デザイナーおよびテンプレート・デザイナー内のフロー・チャートでグラフィカルに表示されます。

ルールまたはテンプレートを開くと、そのコンポーネント(たとえば、式、スクリプト、条件、範囲、ループ)をフロー・チャートで選択し、詳細を表示できます。フロー・チャートのサイズを増減させることにより、コンポーネントの詳細を表示したり非表示にすることもできます。

フロー・チャートでコンポーネントを選択すると、フロー・チャートの下のタブに、コンポーネントのプロパティ、使用状況およびその他の情報が表示されます。コンポーネント間で移動すると、フロー・チャートの下のタブが変わります。

たとえば、式コンポーネントおよびスクリプト・コンポーネントを含むビジネス・ルールを開き、フロー・チャートで式コンポーネントを選択すると、式のプロパティ(名前、説明、アプリケーションおよびアプリケーション・タイプ)がフロー・チャートの下のタブに表示されます。スクリプト・コンポーネントをフロー・チャートで選択すると、スクリプト・コンポーネントのテキスト、プロパティおよび使用状況がフロー・チャートの下のタブに表示されます。

2

ビジネス・ルール設計

次も参照:

- [ビジネス・ルールについて](#)
Oracle Hyperion Calculation Manager では、高度なマルチディメンショナル・ビジネス・ルールの作成、検証、デプロイおよび管理ができます。
- [ビジネス・ルールの作成](#)
ビジネス・ルールは、コンポーネントにグループ化された計算で構成される Oracle Hyperion Calculation Manager オブジェクトです。
- [ビジネス・ルールを開く](#)
Oracle Hyperion Calculation Manager を開くとデフォルトで表示されるシステム・ビューからビジネス・ルールを開くことができます。
- [ビジネス・ルールの編集](#)
ビジネス・ルールの構造は、そのコンポーネント(式、スクリプト、条件、範囲およびループを含む)を追加、除去、または変更することによって編集できます。
- [スクリプト・モードでのビジネス・ルールの編集](#)
ビジネス・ルールのスクリプトを編集できます。
- [ビジネス・ルールの保存](#)
ビジネス・ルールの作成対象のアプリケーションとアプリケーション・タイプに対してビジネス・ルールを保存します。
- [ビジネス・ルールの実行](#)
ビジネス・ルールは実行する前に開く必要があります。
- [ビジネス・ルールのコピー](#)
ビジネス・ルールを別のアプリケーションおよびプラン・タイプにコピーできます。
- [ビジネス・ルール内の検索](#)
ビジネス・ルール・スクリプト内のテキスト文字列を検索します。グラフィカル・ビジネス・ルールのテキストを検索および置換します。
- [ビジネス・ルールの印刷](#)
ビジネス・ルールのプロパティ、フロー・チャートおよびコンポーネントの詳細を印刷できます。
- [ビジネス・ルールの削除](#)
ビジネス・ルールは、他のルールまたはルールセットによって使用されていない場合のみ削除できます。
- [ビジネス・ルール・コンポーネントでの共通のディメンションの定義](#)
ビジネス・ルールを開いて、各ディメンションに共通のメンバー、変数および関数を選択することで、共通ディメンションを定義します。
- [ビジネス・ルールの分析およびデバッグ](#)
ビジネス・ルールを実行し、統計情報を取得して、ビジネス・ルールを分析します。ビジネス・ルールを実行しスクリプトを調べて、ビジネス・ルールをデバッグします。
- [Workforce Planning、Capital Asset Planning および Human Capital Management の事前定義済ビジネス・ルールのロード](#)

- [ビジネス・ルールへのショートカットの作成](#)
- [ビジネス・ルールまたはビジネス・ルールセットのリフレッシュ](#)
システム・ビュー、カスタム・ビューおよびデプロイメント・ビューでは、アプリケーション・リストのレベルをリフレッシュできます。
- [ビジネス・ルールまたはルールセットの使用状況の表示](#)
ビジネス・ルールまたはビジネス・ルールセットを使用しているルール、テンプレートおよびルールセットを表示します。
- [オブジェクトの所有者の変更](#)

ビジネス・ルールについて

Oracle Hyperion Calculation Manager では、高度なマルチディメンショナル・ビジネス・ルールの作成、検証、デプロイおよび管理ができます。

通常、ビジネス・ルールは次の目的で作成します。

- エンティティ間でのコストの割当て
- 収益のモデル化
- 費用のモデル化
- 貸借対照表の準備
- キャッシュ・フローの計算
- 為替換算調整勘定の計算
- グループおよび小数株持ち分の計算
- 繰延税金の計算

ビジネス・ルールを作成する前に、データベース・アウトラインおよび使用するアプリケーションについて熟知しておく必要があります。この情報を把握しておく、ビジネス・ルールをより効率的に作成できます。また、データに関して、次も理解しておく必要があります：

- データの保管と集約方法
- データがデータベースにロードされるレベル
- 計算の順序
- 計算を実行する主な仮定

ビジネス・ルールは、式、スクリプト、ループ、データおよびメンバーの範囲、テンプレートおよび実行時プロンプト変数を含む変数などのコンポーネントを使用して作成できます。(コンポーネントを使用したビジネス・ルールおよびテンプレートの設計を参照してください。)

ビジネス・ルールを作成する際は、作業しているコンポーネント、テンプレートおよび変数を開いたままにできます。Calculation Manager では、ルールを作成する際にタブ間を簡単に移動できるように、タブ付きインターフェイスでこれらのオブジェクトが表示されます。最大 10 個のタブを開くことができます。ただし、最適なパフォーマンスを得るためには、10 個を超えるオブジェクトを同時に開かないことをお勧めします。

Oracle Hyperion Planning でビジネス・ルールを起動するには、管理者がルールに対する起動権限を付与する必要があります。

 ノート:

また、2つ以上の関連するルール(またはルールセット)を同時にまたは連続して起動できるビジネス・ルールを作成できます。[ビジネス・ルールセットの設計](#)を参照してください。

 ノート:

Oracle Hyperion Financial Management でビジネス・ルールを起動するために起動権限は必要ありません。

ビジネス・ルールの作成

ビジネス・ルールは、コンポーネントにグループ化された計算で構成される Oracle Hyperion Calculation Manager オブジェクトです。

ルールには1つ以上のコンポーネント、テンプレートまたはルールが含まれます。

アクセスできるアプリケーションのビジネス・ルールを作成できます。ルールを作成できるかどうかは、割り当てられている役割によって決定されます。(Oracle Enterprise Performance Management System ユーザー・セキュリティ管理ガイドを参照してください)。

ルールは、フロー・チャートにグラフィカルに表され、フロー・チャートにコンポーネントをドラッグ・アンド・ドロップしてルールを設計できます。

ビジネス・ルールを作成するには:

1. 次のいずれかを行います:

- システム・ビュー、カスタム・ビュー、デプロイメント・ビューまたはフィルタ・ビューで  をクリックし、「新規オブジェクト」ダイアログ・ボックスに情報を入力します。必ず「オブジェクト・タイプ」に「ルール」を選択します。
- システム・ビューで「ルール」を右クリックして「新規」を選択し、「新規ルール」ダイアログ・ボックスに情報を入力します。

 ノート:

∖r、∖n、∖t、∖f、∖b、<、>、(、)、"、∖\、{、}、[、]、*、?はルール名に使用できません。

2. ルール・デザイナの「新規オブジェクト」および「既存のオブジェクト」でオブジェクトを選択し、フロー・チャートの「開始」と「終了」の間にドロップします。

次の点に注意してください。

- 既存の式またはスクリプト・コンポーネントをフロー・チャートにドラッグすると、デフォルトで、式またはスクリプトは共有オブジェクトになります。共有しない場合は、式またはスクリプト・コンポーネントの「プロパティ」で「共有」チェック・ボ

ックスをクリアします。スクリプトおよび式コンポーネントの共有を参照してください。

- スクリプト・コンポーネントをグラフィカル形式で表示するには、フロー・チャートでスクリプト・コンポーネントを右クリックし、「**グラフィカルに変換**」を選択します。

スクリプトが有効な場合にのみ、スクリプト・コンポーネントがグラフィカル形式に変換されます。グラフィカル形式への変換を元に戻すには、スクリプト・コンポーネントを右クリックし、「**元に戻す**」を選択します。

- ルールとは関係なく式やスクリプトなどのオブジェクトを作成して、後でルールに追加できます。
- ビジネス・ルールをグラフィカルでない形式(スクリプト形式)で操作するには、「**デザイナー**」の隣のドロップダウンをクリックし、「**スクリプトの編集**」を選択します。(スクリプト・モードでのビジネス・ルールの編集を参照してください。)

3. ルールのプロパティを入力します。

ルールにコンポーネントを追加したり、フロー・チャートでコンポーネント間を移動したりすると、プロパティが変わります。特定のコンポーネントのプロパティを入力するには、フロー・チャートでそのコンポーネントを選択します。

フロー・チャートで「**開始**」または「**終了**」を選択すると、次のプロパティが表示されます。

- 一般** - 名前、説明およびコメント
- 場所** - アプリケーションおよびプラン・タイプ
- オプション** - 現在のアプリケーションに固有のオプション

表 2-1 Planning オプション

プロパティ	説明
動的メンバーの作成	<p>有効な実行時プロンプトでメンバー・タイプの変数にデフォルトの動的な親を指定すると、メンバーが作成されます。ビジネス・ルールで動的メンバーを作成し、デフォルトの動的な親を選択した場合、Oracle Hyperion Planning でルールを起動する前にその親の子メンバーが自動的に作成されます。</p> <p>ノート: 「動的メンバーの作成」を選択した場合、ビジネス・ルールが起動に失敗すると、新しく作成されたメンバーは削除されます。</p> <p>ノート: このプロパティは、Financial Consolidation and Close アプリケーションでは使用できません。</p>
動的メンバーの削除	<p>有効な実行時プロンプトでメンバー・タイプの変数にデフォルトの動的な親を指定すると、メンバーが削除されます。ビジネス・ルールで動的メンバーを削除し、デフォルトの動的な親を選択した場合、Planning でルールを起動した後にその親の子メンバーが自動的に削除されます。</p> <p>ノート: このプロパティは、Financial Consolidation and Close アプリケーションでは使用できません。</p>
通知の有効化	<p>ルールがエラーあり、またはエラーなしで起動された場合、ログオンしているユーザーに電子メール通知を送信するビジネス・ルールを有効にします。</p>

表 2-2 Financial Management オプション

プロパティ	説明
ロギングの使用可能	<p>ルールが起動されると、ログ・テキストをログ・ファイルに含めます。ルール、ルールセットおよびコンポーネントのロギングを使用可能にできます。ビジネス・ルールの式の文に含めるログ・テキストを指定します。</p> <p>ルールセットに対してはロギングが使用不可になっているが、そのルールに含まれるルールおよびコンポーネントに対しては使用可能になっている場合、ルールおよびコンポーネントのログ設定よりもルールセットのログ設定の方が優先されるため、ログ・ファイルは作成されません。</p>
タイマーの使用可能	<p>ルールが起動されると、ログ・ファイルのルールを処理するためにかかった時間を記録します。</p> <p>タイマーは、ルール、ルールセットおよびコンポーネントに対して使用可能にできます。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。</p> <p>たとえば、タイマーが使用可能になっているルールセットがあり、そのルールセットにタイマーが使用可能になっている3つのルールが含まれている場合、ルールセット、およびルールセットの各ルールの処理にかかる時間が記録されます。</p>
使用不可	<p>スクリプトが生成される間、一時的にビジネス・ルールを使用不可にします。</p>
関数	<p>「パラメータ」タブでパラメータとして指定する実行変数が含まれる関数としてルールを使用します。</p> <p>ルールを関数として使用すると、関数のように値が戻されます。「関数」が選択されたビジネス・ルールは、関数セレクトで関数として表示されます。</p>

表 2-3 General Ledger オプション

プロパティ	説明
開始日	この日付よりも後でビジネス・ルールを起動できます
終了日	この日付よりも前にビジネス・ルールを起動できます

4. ルール・デザイナの次のタブで情報を入力または確認します。
 - **グローバル範囲** - ビジネス・ルール・コンポーネントで共通のディメンションを定義します。
[ビジネス・ルール・コンポーネントでの共通のディメンションの定義](#)を参照してください。
 - **変数** - ビジネス・ルールで使用する変数に関する情報を確認および定義します。
 「変数」タブは、ビジネス・ルールに実行時プロンプト値が含まれている場合にのみ表示されます。
 非 Groovy ルールの場合、メンバーまたはメンバー(複数)タイプの変数のみが、集約ストレージ・オプション(ASO)タイプの Planning キューブでサポートされているタイプの変数です
 - **スクリプト** - ルールに対して生成されるスクリプトが表示されます。
 このタブでは変更できません。スクリプトに変更を加えるには、「デザイナ」の隣のドロップダウンで「スクリプトの編集」を選択します。

ビジネス・ルールのスクリプトを表示および編集する機能は、Oracle Hyperion Financial Management、Planning および Oracle Essbase ブロック・ストレージのみで使用できます。

- **「使用状況」** - ルールを使用するルールおよびルールセットが表示されます。
このタブの情報は編集できません。デフォルトでは、ルールを作成する時点では、ルールはどのルールにもルールセットにも使用されていません。
- **パラメータ** - パラメータとして使用する実行変数を選択します
[パラメータとして使用する実行変数の選択](#)を参照してください。
- **エラーおよび警告** -  をクリックしてビジネス・ルールのスクリプト診断を実行します。これを行うと、Calculation Manager では、ビジネス・ルール・スクリプトを分析し、次のいずれかを表示します：
 - 検証が行われない場合は、検証エラーが表示されます。
検証エラーを修正するには、エラーを右クリックし、「**スクリプトに表示**」または「**デザイナーに表示**」を選択します。「**デザイナーに表示**」を選択した場合、Calculation Manager でエラーのあるコンポーネントが表示され、必要な変更を加えてルールを保存し、スクリプト診断を再実行できます。コンポーネントを編集するにはデザイナー・ビューを表示している必要があります。
 - ルールが検証できた場合は、サマリー、警告およびブロック。
警告またはブロック・アイテムをクリックし、「**スクリプトに表示**」または「**デザイナーに表示**」を選択します。コンポーネントはデザイナー・ビューでのみ編集できます。
 - * **「サマリー」** - 計算におけるデータ・セルの数、データのパス・スルーの数、不正に使用されたディメンションの数などの統計。
 - * **「警告」** - すべての疎ディメンションがセル参照に指定されるかどうか、セルは疎メンバーを参照するかどうか、割当てでは異なるデータ・ブロックの疎ディメンション・メンバーを参照するかどうかなどの情報。
 - * **ブロック** - 各 Fix 文について Fix 文の影響を受ける可能性のある潜在的および見積りのブロック数などの情報

5.  をクリックしてルールを保存します。

パラメータとして使用する実行変数の選択

ノート:

実行変数を選択してパラメータとして使用できるのは、Oracle Hyperion Financial Management のみです。

パラメータとして実行変数を使用するには:

1. ビジネス・ルールを開く

2. ルール・デザイナの「パラメータ」タブで、実行変数を選択してから、「パラメータ」を選択します。

ビジネス・ルールを初めて開いたとき、「パラメータ」タブは空です。ビジネス・ルールにコンポーネントを追加し、それらのコンポーネントが変数を使用する場合、このタブに変数が表示されます。

実行変数がルールセット・スコープを持つ場合(同じルールセット内で1つのビジネス・ルールから別のビジネス・ルールに渡される場合)、「パラメータ」がデフォルトで選択されます。

変数がルール・スコープを持つ場合(作成されるビジネス・ルールでのみ使用できて、別のビジネス・ルールに渡されない場合)、「パラメータ」は選択されません。

3. 「渡す方法」で、「参照」または「値」を選択します。

ノート:

ビジネス・ルールに、「パラメータ」が選択される実行変数が少なくとも1つ含まれる場合、このビジネス・ルールを別のビジネス・ルールまたは同じビジネス・ルールに含めると、その実行変数に対する値を「パラメータ」タブで指定できます。

ビジネス・ルールとしてのパラメータの使用

ノート:

パラメータをビジネス・ルールとして使用できるのは、Oracle Hyperion Financial Managementのみです。

ルール・デザイナの「パラメータ」タブを使用して、ビジネス・ルールで使用するパラメータを指定します。Oracle Hyperion Calculation Managerによって、ルール内で実行変数が使用されているかどうかを確認されます。(実行変数はルールセットまたはルールのスコープを持つ場合があります。)ビジネス・ルールで使用されている実行変数はすべて、「パラメータ」タブのグリッドにリストされます。

「パラメータ」列の下のグリッドで、ルールセット・スコープを持つ実行変数のチェック・ボックスがデフォルトで選択されています。これらの実行変数はパラメータとして渡されます。ルール・スコープを持つ実行変数のチェック・ボックスはデフォルトでクリアされています。これらの実行変数は、その変数が使用されているビジネス・ルールのみにも適用されるためです。これらの実行変数はパラメータとして渡されません。グリッドにリストされた任意の変数のデフォルト設定を変更できます。

「渡す方法」列では、デフォルトの設定は「参照」ですが、「値」を選択することができます。

 **ノート:**

「**パラメータ**」タブに表示される変数は自動的に生成されます。変数のスコープを追加、削除または変更するなどして、これらのいずれかの変数を変数デザイナーで変更した場合、ルールの「**パラメータ**」タブを再度選択すると、ビジネス・ルールに影響する変更のみが表示されます。

パラメータをビジネス・ルールとして使用するには:

1. ルール・デザイナーでビジネス・ルールを作成するか開きます。
2. 「**プロパティ**」で、「**関数**」を選択してください。こうすることで、パラメータとして指定された実行変数を含む関数としてルールが扱われます。

パラメータ・パッシングは、「**関数**」オプションが選択されているかどうかにかかわらず、どのビジネス・ルールでも使用できます。「**関数**」が選択されているかどうかにかかわらず、どのビジネス・ルールでも使用できます。オプションが選択されていない場合、パラメータ・パッシング機能を備えた通常のビジネス・ルールです。オプションが選択されている場合、ビジネス・ルールは関数として扱われます。

3. ビジネス・ルールを保存します。
4. 2番目のビジネス・ルールを開きます。
5. 「**既存のオブジェクト**」から、1番目のビジネス・ルールのコピーを2番目のビジネス・ルールのフロー・チャートにドラッグします。
6. 「**パラメータ**」タブで、ビジネス・ルールとして使用するパラメータの値を入力します。
7.  をクリックします。

ビジネス・ルールを開く

Oracle Hyperion Calculation Manager を開くとデフォルトで表示されるシステム・ビューからビジネス・ルールを開くことができます。

カスタム・ビュー、フィルタ・ビューまたはデプロイメント・ビューからルールを開くこともできます。

ビジネス・ルールを開くには、次のいずれかを行います。

- ルールを右クリックし、「**開く**」を選択します。
- ルールをダブルクリックします。

ノート:

Oracle Financials Cloud で、別の管理者によってルールがロックされている場合、次のメッセージを受け取ります: 「objectname は読取り専用モードで開かれています。オブジェクトは現在次のものによって編集されています: FINUSER2」。これが発生した場合は、次のステップを完了します:

1. 「OK」をクリックします。
2. ルールまたはルールセットを右クリックし、**ロック解除**を選択します。次のメッセージが表示されます:

選択したアイテムをロック解除しますか?それを編集しているユーザーによる変更は失われます。

3. 「OK」をクリックしてルールを開くか、「**取消**」をクリックしてルールを開かずに閉じます。

ルールがロック解除された場合、ルールを編集していた管理者は、メッセージを受け取りません。ルールが元の管理者によって編集されて保存され、その後ルールをロック解除した管理者が変更を行って保存する場合、最後の保存が保持され、元の管理者が行った編集は上書きされます。

ビジネス・ルールの編集

ビジネス・ルールの構造は、そのコンポーネント(式、スクリプト、条件、範囲およびループを含む)を追加、除去、または変更することによって編集できます。

ビジネス・ルールのコンポーネントおよびビジネス・ルール自体のプロパティも編集できます。

次のビジネス・ルールのプロパティを編集できます。

- 名前およびキャプション
- 説明およびコメント
- ディメンション、メンバー、および変数
- Oracle Hyperion Financial Management アプリケーションの場合は、ビジネス・ルールにログ・ファイルおよびタイマーがあるかどうか、スクリプトの生成時にビジネス・ルールが使用不可であるかどうか、およびビジネス・ルールがパラメータとして機能するかどうか

ビジネス・ルールを編集するには:

1. ルールを開きます。
2. ルール・デザイナーで新規コンポーネントを追加したり、既存のコンポーネントをルールのフロー・チャートからコピーしたり、削除します。
 - コンポーネントを追加するには、「**新規オブジェクト**」または「**既存のオブジェクト**」からオブジェクトをドラッグし、フロー・チャートにドロップします。

既存の式またはスクリプト・コンポーネントをフロー・チャートに追加すると、デフォルトで、式またはスクリプトは共有オブジェクトになります。共有しない場合は、式またはスクリプト・コンポーネントの「**プロパティ**」で「**共有**」チェック・ボックスをクリアします。[スクリプトおよび式コンポーネントの共有](#)を参照してください。

- コンポーネントをコピーするには、コンポーネントを右クリックし、「コピー」を選択してフロー・チャートに貼り付けます。
 - コンポーネントを削除するには、コンポーネントを右クリックし、「削除」を選択します。
3. 「プロパティ」で、ルールのプロパティを編集します。
- ルールにコンポーネントを追加したり、フロー・チャートでコンポーネント間を移動したりすると、プロパティが変わります。特定のコンポーネントのプロパティを入力するには、フロー・チャートでそのコンポーネントを選択します。[コンポーネントを使用したビジネス・ルールおよびテンプレートの設計](#)を参照してください。
4.  をクリックして変更を保存します。

 **ヒント:**

ビジネス・ルールのコンポーネントを編集する際、コンポーネント・アイコンのサイズとフロー・チャートに表示される詳細情報の量を増減できます。

これを行うには、フロー・チャート内で  を使用してズーム・インおよびズーム・アウトします。

フロー・チャートを小さいサイズで表示した場合、コンポーネントのキャプションは表示されませんが、アイコン上にマウス・ポインタを置くと、キャプションが表示されます。フロー・チャート内のコンポーネントのサイズにかかわらず、コンポーネントを選択してそのプロパティを表示できます。

スクリプト・モードでのビジネス・ルールの編集

ビジネス・ルールのスクリプトを編集できます。

デフォルトでは、ビジネス・ルールは、ルールのフローをグラフィカルに設計するルール・デザイナーを使用してグラフィカル・モードで作成します。ビジネス・ルールを作成し、保存した後、グラフィカル・モードまたはスクリプト・モードで編集できます。スクリプト・モードでの編集を選択した場合、後でグラフィカル・モードでの編集に戻れます。

ビジネス・ルールをスクリプト・モードで編集するには:

1. ルールを開きます。
2. ルール・デザイナーで、「デザイナー」の隣のドロップダウンで「スクリプトの編集」を選択します。

グラフィカル・モードに戻るには、「スクリプトの編集」の隣のドロップダウンで「デザイナー」を選択します。

ビジネス・ルールに共有のコンポーネント(式およびスクリプト・コンポーネントなど)またはテンプレートが含まれる場合、グラフィカル・モードからスクリプト・モードの編集に切り替えると、ビジネス・ルールのスクリプトには、スクリプト・エディタ内の共有コンポーネントおよびテンプレートへの参照のみが含まれます。共有コンポーネントおよびテンプレートの完全に生成されたスクリプトは、下部ペインの「スクリプト」タブに表示されます。

次の点に注意してください。

- ASO ルールで「スクリプトの編集」を選択した場合、スクリプトは自動的に Groovy に変換されます。その後ルールを保存すると、スクリプトは Groovy で保存され、グラフィカルに戻すことはできません。グラフィカル・フォーマットを保持するには、「名前を付けて保存」を選択し、Groovy スクリプトを含むルールを別名で保存します。
- BSO ルールで「スクリプトの編集」を選択した場合、グラフィカル・ルールは計算スクリプトの構文に基づくスクリプト・モードに変換されます。スクリプトを Groovy に変換するには、「スクリプト・タイプ」として「Groovy スクリプト」を選択します。
- スクリプト・モードで編集する場合は次のキーワードを使用しないでください。
 - /*STARTCOMPONENT*/
 - /*STARTCOMPONENT:SCRIPT*/
 - /*STARTCOMPONENT:FORMULA*/
 - /*ENDCOMPONENT*/
 - /*STARTRULE*/
 - /*ENDRULE*/
 - /*STARTTEMPLATE*/
 - /*ENDTEMPLATE*/

3. スクリプトに必要な編集を行います。

表 2-4 スクリプト・モードでの編集時に使用可能なオプション

アイコン	説明	関連項目
	スクリプトの行番号を表示または非表示にします。行番号はデフォルトで表示されています。	
	関数とそのパラメータの挿入 関数には、自動推奨機能があります。関数の最初の数文字を入力して、[Ctrl]を押しながら[Space]を押すと、推奨内容が表示されます。推奨内容のいずれかを選択し、[Enter]を押して、関数を完成させます。	関数の操作
	ディメンション間メンバーを挿入します	コンポーネントへのメンバーおよび関数の追加
	ディメンションから選択されたメンバーの範囲を挿入します	コンポーネントへのメンバーおよび関数の追加
	変数を挿入します	変数の操作
	スマート・リストを挿入します	スマート・リストの操作
	テンプレートを編集します	システム・テンプレートの操作

表 2-4 (続き) スクリプト・モードでの編集時に使用可能なオプション

アイコン	説明	関連項目
	コメント	
	コメント解除	
	コメントの表示/非表示	
	コードのフォーマット コードをフォーマットすると、スクリプトの行がグループ化されてインデントされ、空白行が削除されます。	
	構文の確認	
	スクリプトを折り返し、ページのスクロールが必要なスクリプトの長い行が同じページで複数の行に表示されるようにします	
	コード補完オン/コード補完オフ	
	コードを完成させるための推奨内容を有効にするか抑制します たとえば、コード補完を起動するには、FIXと入力した後に、[Shift]キーを押しながら[Space]を押します。そうすると、Oracle Hyperion Calculation Manager によって 0 ENDFIX が入力されます。	
	スクリプトのテキスト文字列を検索および置換します	ビジネス・ルール・スクリプト内のテキスト文字列の検索
	テキスト文字列を検索する際、「 大文字/小文字の一致 」または「 完全に一致する単語 」を選択します	ビジネス・ルール・スクリプト内のテキスト文字列の検索

4.  をクリックします。

チュートリアル・ビデオ:



[スクリプトの確認](#)

ビジネス・ルールの保存

ビジネス・ルールの作成対象のアプリケーションとアプリケーション・タイプに対してビジネス・ルールを保存します。

次も参照:

- [ビジネス・ルールの保存](#)
- [異なる名前でのビジネス・ルールの保存](#)

ビジネス・ルールの保存

ビジネス・ルールを保存すると、作成対象のアプリケーションおよびアプリケーション・タイプに保存されます。保存後には、デプロイ、検証および起動できます。これを Oracle Hyperion Calculation Manager でデプロイおよび検証できます。起動は、Oracle Hyperion Planning (Planning ビジネス・ルールの場合)、Oracle General Ledger (Oracle General Ledger ビジネス・ルールの場合)、および Calculation Manager または Oracle Essbase Administration Services (Oracle Essbase ビジネス・ルールの場合)から実行できます。

ノート:

Oracle Hyperion Financial Management から起動できるのはビジネス・ルールセットのみです。ビジネス・ルールは起動できません。

ビジネス・ルールを作成または編集した後に保存するには、 をクリックします。

ノート:

ビジネス・ルールを保存した後**システム・ビュー**で確認するには、アプリケーション・リストのリフレッシュが必要な場合があります。これを行うには、**システム・ビュー**で任意のノードまたはオブジェクトを右クリックし、「リフレッシュ」を選択します。

異なる名前でのビジネス・ルールの保存

「名前を付けて保存」を使用すると、異なる名前でのビジネス・ルールを保存できます。「名前を付けて保存」を使用すると、同じルールセット・タイプ内のあるルールセットから別のルールセットにコピーすることもできます。「名前を付けて保存」では、元のビジネス・ルールと区別するために異なる名前のコピーが作成されます。

異なる名前でのビジネス・ルールを保存するには:

1. ルールをダブルクリックします。
2. ルール・デザイナーで、「アクション」,、「名前を付けて保存」の順に選択します。
3. 「名前を付けて保存」で、新しいルール名を入力し、「アプリケーション」を選択します。
4. 次のいずれかのタスクを行います。
 - ルールが Oracle Hyperion Financial Management ルールの場合、「**計算タイプ**」を選択します。
 - ルールが Oracle Hyperion Planning ルールの場合、「**プラン・タイプ**」を選択します。
 - ルールが Oracle Essbase または Oracle General Ledger ルールの場合、「**データベース**」を選択します。

ノート:

異なる名前で保存したルールのアプリケーション・タイプは変更できません。ルールを異なる名前で保存するときに変更できるのは、ルールが属するアプリケーションと計算、プラン・タイプまたはデータベースです。

5. 「OK」をクリックします。
新規ルールは、**システム・ビュー**のアプリケーション・リストに追加されます。

ビジネス・ルールの実行

ビジネス・ルールは実行する前に開く必要があります。

ビジネス・ルールを実行するには:

1. ルールをダブルクリックするか、ルールを右クリックして「**開く**」を選択します。
2. ルールが開いたら  (起動ボタン)をクリックします。

ビジネス・ルールのコピー

ビジネス・ルールを別のアプリケーションおよびプラン・タイプにコピーできます。

ビジネス・ルールを別のアプリケーションにコピーすると、別の名前で新規ビジネス・ルールが作成されます。ビジネス・ルールで使用される変数も、新しいアプリケーションにコピーされます。

ビジネス・ルールを別のアプリケーションおよびプラン・タイプにコピーするには:

1. **システム・ビュー**でルールを右クリックし、「**コピー先**」を選択します。
2. 「**コピー先**」に、ビジネス・ルールの新しい名前を入力するか、デフォルト名を受け入れて、アプリケーション、アプリケーション・タイプおよびプラン・タイプを選択します。

複数のアプリケーションおよびプラン・タイプにビジネス・ルールをコピーすることはできません。

3. 共有コンポーネントを含むビジネス・ルールを別のアプリケーションにコピーし、共有コンポーネントをそのアプリケーションにコピーする場合は、「**共有コンポーネントのコピー**」を選択します。

これにより、コピーされたアプリケーションに共有コンポーネントが作成され、コピーされたビジネス・ルールは、アプリケーションおよびプラン・タイプから共有コンポーネントを使用するようになります。「**共有コンポーネントのコピー**」を選択しない場合、共有コンポーネントはルールにコピーされますが、コピーされたルールは、元のルールのアプリケーションおよびプラン・タイプから共有コンポーネントを使用します。

変数を含むビジネス・ルールを別のアプリケーションにコピーすると、コピー先のアプリケーションに変数が作成されます。コピー先のアプリケーションにすでに変数が存在する場合、変数はコピー時に、ルール・レベルの変数として作成されます。

- 共有コンポーネントをコピーする場合は、「**既存のオブジェクトの置換**」を選択し、オブジェクトを同じ名前の上書きします。

ビジネス・ルール内の検索

ビジネス・ルール・スクリプト内のテキスト文字列を検索します。グラフィカル・ビジネス・ルールのテキストを検索および置換します。

次も参照:

- ビジネス・ルール・スクリプト内のテキスト文字列の検索
- グラフィカル・ビジネス・ルールでのテキストの検索および置換

ビジネス・ルール・スクリプト内のテキスト文字列の検索

ビジネス・ルール・スクリプト内のテキスト文字列を検索するには:

- スクリプトを含むビジネス・ルールを開きます。
- ルール・デザイナーで「**スクリプト**」タブを選択します。
- 検索するテキストを入力し、をクリックします。

グラフィカル・ビジネス・ルールでのテキストの検索および置換

文字列を検索する場合、Oracle Hyperion Calculation Manager は、フロー・チャートで選択されているコンポーネントの直後のコンポーネントから検索を開始してフロー・チャートの最後まで検索を続け、フロー・チャートの最初から再開してフロー・チャートで選択されているコンポーネントに到達するまで検索します。

一致するテキスト文字列が1つ見つかった後に、別の出現を検索する場合は、再度検索を開始して次の出現を探す必要があります。

グラフィカル・ビジネス・ルールでテキストを検索および置換するには:

- ビジネス・ルールを開きます。
- ルール・デザイナーで、ビジネス・ルールの「**開始**」または「**終了**」アイコン、またはビジネス・ルールのフロー・チャート内のコンポーネントを右クリックし、次のいずれかを行います
 - 「**検索**」を選択して、テキスト文字列のインスタンスを検索します。
 - 「**すべて置換**」を選択して、テキスト文字列のインスタンスを検索し、置換します。

ビジネス・ルールの印刷

ビジネス・ルールのプロパティ、フロー・チャートおよびコンポーネントの詳細を印刷できます。

たとえば、配賦費用の式コンポーネントが含まれるビジネス・ルールを印刷すると、式の構文、式を構成する関数および変数、ルールのフロー・チャートのステップのサマリー(図ではない)、およびルールのプロパティが出力されます。

スクリプト形式のみのビジネス・ルールを印刷する場合:

- PDF ファイルの「ルールの詳細」セクションには、ビジネス・ルール・スクリプトが含まれます。
- PDF ファイルの「フロー・チャート」セクションには、「開始」および「終了」ノードのみが含まれます。
- PDF ファイルのフローのサマリー・セクションは空です。

 **ノート:**

ビジネス・ルールセットおよびコンポーネントはいずれも印刷できません。コンポーネントは、印刷するビジネス・ルールで使用されている場合に印刷できます。

ビジネス・ルールを印刷するには:

1. **システム・ビュー**でビジネス・ルールを右クリックし、「印刷」を選択します。
2. 「印刷プレビュー」ダイアログ・ボックスで情報を入力し、「PDF の生成」をクリックします。

ビジネス・ルールの PDF ファイルが Adobe Acrobat で開きます。

ビジネス・ルールの削除

ビジネス・ルールは、他のルールまたはルールセットによって使用されていない場合のみ削除できます。

ルールが使用されている場合は、削除する前に、ルールを使用しているルールおよびルールセットからビジネス・ルールを除去するか、ルールを使用しているルールおよびルールセット用にコピーを作成する必要があります。

ビジネス・ルールを削除するには:

1. **システム・ビュー**でルールを右クリックし、「使用状況の表示」を選択して他のルールまたはルールセットによって使用されていないことを確認します。
2. ルールが別のルールおよびルールセットに使用されていない場合、ルールを右クリックし、「削除」を選択します。
3. 「OK」をクリックしてルールの削除を確定します。

ビジネス・ルール・コンポーネントでの共通のディメンションの定義

ビジネス・ルールを開いて、各ディメンションに共通のメンバー、変数および関数を選択することで、共通ディメンションを定義します。

ノート:

ビジネス・ルール・コンポーネントにディメンションを定義するのは Oracle Hyperion Planning、Oracle General Ledger および Oracle Essbase のみです。

ビジネス・ルール・コンポーネントで共通のディメンションを定義するには:

1. ビジネス・ルールを開く
2. ルール・デザイナーで、フロー・チャートの「開始」または「終了」をクリックします。
3. 「グローバル範囲」タブでディメンションを選択し、 をクリックして各ディメンションに共通のメンバー、変数および関数を選択します。

ディメンションに対して選択した値は、ルールの起動時に計算される値になります。

変数を選択した場合、「変数に動的にリンク」を選択してルールに変数を動的にリンクし、変数に変更が行われた場合に変更がルールで更新されるようにします。

ノート:

「グリッド値を除外」を選択して、「Fix/EndFix」スクリプトではなく「Exclude/EndExclude」スクリプトをビジネス・ルールに作成します。

ビジネス・ルールの分析およびデバッグ

ビジネス・ルールを実行し、統計情報を取得して、ビジネス・ルールを分析します。ビジネス・ルールを実行しスクリプトを調べて、ビジネス・ルールをデバッグします。

次も参照:

- [ビジネス・ルールの分析](#)
- [ビジネス・ルールのスクリプトの分析](#)
- [ビジネス・ルールのスクリプトの比較](#)
- [変更されたビジネス・ルールと保存されたビジネス・ルールの比較](#)
- [ビジネス・ルールのデバッグ](#)
- [ビジネス・ルール内のコンポーネントの無効化](#)

ビジネス・ルールの分析

ビジネス・ルールを分析する場合、ビジネス・ルールを実行し、統計情報(コンポーネントの実行にかかった時間、実行された回数、コンポーネント実行前後のメンバー交差の値など)を取得します。分析にかかる時間はシステムのメモリーに依存します。

ノート:

ビジネス・ルールを分析およびデバッグできるのは、Oracle Hyperion Planning または Oracle Essbase を使用している場合のみです。

ノート:

分析機能は、メンバー範囲、メンバー・ブロック、固定ループ、または条件コンポーネントでは使用できません。

ビジネス・ルールを分析するには:

1. ビジネス・ルールを開く
2. ルール・デザイナーで、「アクション」、「分析」の順に選択します。
3. 「分析条件を入力してください」ダイアログ・ボックスで次のようにします。
 - 分析に表示する数値エントリを入力するか、「すべて表示」を選択してエントリを表示します。

ここで入力する数は、計算および表示されるメンバー交差の数です。たとえば、アプリケーションに次のディメンションがあるとした場合:

- 年 = 12 メンバー
- メジャー = 50,525 メンバー
- 製品 = 450 メンバー
- 市場 = 2000 メンバー
- シナリオ = 4 メンバー

この例で計算されるメンバー交差の数は次のとおりです: 2,182,680,000 (12 x 50,525 x 450 x 2,000 x 4、つまり年 x メジャー x 製品 x 市場 x シナリオ)

- ディメンションごとに、分析するメンバーを選択します。
メンバーの選択に、次のオプションを使用できます。
 -  をクリックして「メンバー・セレクタ」ダイアログ・ボックスにアクセスします。
 - ディメンションを選択し、 をクリックします。
 - メンバーを入力します。

ディメンションごとに選択できるメンバーは1つのみです。

4. 「OK」をクリックして分析を開始します。
5. 分析が完了したら、ビジネス・ルールのフロー・チャートでコンポーネントをクリックし、「分析情報」タブを選択します。

ヒント:

分析情報をエクスポートするには、「分析情報」タブ内のグリッドで右クリックし、「エクスポート」を選択します。

ノート:

Essbase の最適化が原因で、分析の正しい開始値と終了値が表示されない場合があります。少なくとも1つの交差に対して分析の開始値と終了値が表示されない場合、件数と経過時間が正しくない可能性があります。

追跡対象のディメンションのメンバーの非表示

追跡されているディメンションのメンバーを非表示にするには、「分析情報」タブで「**追跡中のディメンションの非表示**」を選択します。このオプションを選択しない場合、そのディメンションで処理されるすべてのメンバーが表示されます。

「分析情報」タブに表示される前の値と後の値で、追跡メンバー値を変更したメンバーが表示されます。前の値と後の値は、追跡しているメンバーの値です。たとえば、製品 200-30 を追跡している場合は、製品 200 の行が表示されます。製品 200 の行に表示されている値は、200-30 の値です。

統計情報の取得

ビジネス・ルールの分析時に統計情報を取得するには、次の点に注意します。

- Oracle Essbase に対する内部呼出しをメンバー・ブロックで囲っておく必要があります。
- Fix 文で1つのディメンションのみを空にしておき、そのディメンションの追跡メンバーを使用しないと、そのディメンションまたはその子孫の1つがメンバー・ブロックに使用されます。
- Fix 文で1つのディメンションのみを空にしておき、そのディメンションの動的計算メンバーである追跡メンバーを指定すると、動的計算メンバーやラベルのみのメンバーではないディメンションまたはその子孫の1つがメンバー・ブロックになります。
- 「分析情報」タブには追跡メンバーの値と、メンバー・ブロックで使用されるメンバーが表示されます。

ビジネス・ルールのスクリプトの分析

グラフィカル・ビジネス・ルールまたはスクリプト・ビジネス・ルールを実行し、そのスクリプトを分析できます。

ビジネス・ルールのスクリプトを分析するには:

1. **システム・ビュー**でビジネス・ルールを右クリックし、「**開く**」を選択します。
2. ルールを表示した状態で、「**アクション**」、「**スクリプトの分析**」の順に選択します。
これによってビジネス・ルールが実行されます。
3. ビジネス・ルールに実行時プロンプト変数が含まれている場合、「**RTP 値の入力**」ダイアログ・ボックスに実行時プロンプト値を入力し、値をビジネス・ルールに適用するかどうかを指定して「**OK**」をクリックします。
4. 「**スクリプト分析**」タブでスクリプトを分析します。

スクリプトの行に分析情報がある場合、 が行の隣に表示されます。行を選択し、その分析情報を画面の右側の「**分析情報**」領域で確認します。行に分析情報がない場合、「分析情報は使用できません。」と表示されます。実行に最も時間がかかる行は赤でハイライト表示されます。

「**プロパティ**」領域に、名前、アプリケーション、プラン・タイプ、スクリプトの分析にかかった時間(ミリ秒)などのビジネス・ルールに関する情報が表示されます。

5. スクリプト情報の確認を終えたら、 をクリックして「**スクリプト分析**」タブを閉じます。

ビジネス・ルールのスクリプトの比較

ビジネス・ルールのスクリプトを比較して、それぞれがどのように違うか、変更した箇所、またはグラフィカル・ビジネス・ルールのスクリプトを確認できます。

ノート:

2つを超えるビジネス・ルールを一度に比較することはできません。また、1つのビジネス・ルールの異なるバージョンを比較することはできません。たとえば、あるビジネス・ルールの前のバージョンと直前に保存したバージョンを比較できません。

2つのビジネス・ルールのスクリプトを比較するには:

1. **システム・ビュー**で2つのビジネス・ルールを選択します。
2. 選択したビジネス・ルールを右クリックし、「**スクリプトの比較**」を選択します。
「**スクリプトの比較**」ウィンドウは、次のようになります。
 - どちらかのスクリプトに追加が行われた場合、その行の背景が薄いグレーで表示されます。
 - どちらかのスクリプトで削除が行われた場合、その行の背景が濃いグレーで表示されます。
 - いずれかのスクリプトが変更された場合は、その行の背景が黄色で表示されません。
 - 最初の列にはスクリプトの行番号が表示されます。
 - 2番目の列には、次の3種類の文字の1つが表示されます:

- *は、左列に表示されるスクリプトの行が、右列に表示されるスクリプトの行と異なることを示します
 - +は、左列に表示されているスクリプトと比較すると、右列に表示されているスクリプトに新しい行が追加されていることを示します。
 - -は、左列に表示されているスクリプトと比較すると、右列に表示されているスクリプトから行が削除されていることを示します。
 - 3番目と4番目の列には2つのスクリプトが表示されます。
 - ウィンドウの下部は2つのペインに分かれており、上部ペインで選択したスクリプトの完全な行が表示されます。1つ目のペインには左列のスクリプト、2つ目のペインには右列のスクリプトが表示されます。
3. スクリプトの比較が終了したら、「OK」をクリックします。

変更されたビジネス・ルールと保存されたビジネス・ルールの比較

保存されたバージョンのビジネス・ルールと変更されたバージョンのビジネス・ルールのスクリプトを表示して、2つのバージョンがどのように違うかの確認や変更した箇所の確認を行ったり、保存されたグラフィカル・ビジネス・ルールと変更されたグラフィカル・ビジネス・ルールのスクリプトを左右に並べて確認できます。

保存されたビジネス・ルールと変更されたビジネス・ルールを比較するには:

1. グラフィカル・ビジネス・ルールまたは計算スクリプト・ビジネス・ルールを開きます。
2. 「編集」、「保存済バージョンとの比較」の順に選択します。

「スクリプトの比較」ウィンドウは、次のようになります。

- ビジネス・ルールまたは計算スクリプトに追加が行われた場合、その行の背景が薄いグレーで表示されます。
- ビジネス・ルールまたは計算スクリプトから削除が行われた場合、その行の背景が濃いグレーで表示されます。
- ビジネス・ルールまたは計算スクリプトが変更された場合は、その行の背景が黄色で表示されます。
- 最初の列には、ビジネス・ルールまたは計算スクリプトの行番号が表示されます。
- 2番目の列には、次の3種類の文字の1つが表示されます:
 - *は、左列に表示されているビジネス・ルールまたは計算スクリプトの行が、右列に表示されているビジネス・ルールまたは計算スクリプトの行と異なることを示します。
 - +は、左列に表示されているビジネス・ルールまたは計算スクリプトと比較すると、右列に表示されているビジネス・ルールまたは計算スクリプトに新しい行が追加されたことを示します。
 - -は、左列に表示されているビジネス・ルールまたは計算スクリプトと比較すると、右列に表示されているビジネス・ルールまたは計算スクリプトから行が削除されたことを示します。
- 3番目と4番目の列には、保存されたビジネス・ルールまたは計算スクリプトと変更されたビジネス・ルールまたは計算スクリプトが表示されます。

- ウィンドウの下部は2つのペインに分かれており、上部ペインで選択したビジネス・ルールまたは計算スクリプトの完全な行が表示されます。1つ目のペインには、左列の保存されたバージョンのビジネス・ルールまたは計算スクリプトが表示され、2つ目のペインには、右列の変更されたバージョンのビジネス・ルールまたは計算スクリプトが表示されます。スクリプト内の行を選択すると、下部ペインに表示されます。保存されたバージョンが上部、変更されたバージョンが下部に表示されます。

3. 「OK」をクリックします。

ビジネス・ルールのデバッグ

ビジネス・ルールをデバッグするには、ビジネス・ルールを実行し、そのスクリプトを1行ずつ調べ、スクリプトがどのように実行されるかを確認します。

次のアプリケーションのビジネス・ルールをデバッグできます。

- 書込み権限がある場合は、Oracle Essbase ブロック・ストレージ・アプリケーション
- 起動権限がある場合は、Oracle Hyperion Planning アプリケーション

ノート:

デバッグ中は、ビジネス・ルールを編集できません。

ビジネス・ルールをデバッグするには:

1. ビジネス・ルールを開く
2. ルール・デザイナーで、「アクション」、「デバッグ」の順に選択します。

スクリプト・デバッガに、ビジネス・ルールのスクリプトが表示されます。スクリプトの各文が、別の行に表示されます。

3. ビジネス・ルール・スクリプトの文をデバッグします。

文をデバッグする際、次のことが可能です。

- 文のメンバー交差の値を検査するための、スクリプトの実行を停止するブレーク・ポイントを挿入および削除します。ブレーク・ポイントで実行が停止すると、文のすべてのメンバー交差の値が表示されます。

ブレーク・ポイントを追加するには、ブレーク・ポイントの右側を右クリックし、「**ブレーク・ポイントの追加**」を選択します。割当て文や条件文にブレーク・ポイントを追加できます。ブレーク・ポイントがグレー表示されているスクリプトの行にのみブレーク・ポイントを追加できます。ブレーク・ポイントを削除するには、ブレーク・ポイントを右クリックして、「**ブレーク・ポイントの削除**」を選択します。

- ブレーク・ポイントに条件を追加して、条件を満たす場合にのみ文の実行を停止させます。条件に使用できるのは、ブレーク・ポイントのある文で使用されているメンバーのみです。

ブレーク・ポイントに条件を追加するには、ブレーク・ポイントの右側を右クリックし、「**条件の追加**」を選択します。「**条件の追加**」ダイアログ・ボックスで、 をクリックし、「**条件ビルダー**」に条件を入力します。

条件を編集するには、条件を右クリックして、「**条件の編集**」を選択します。

- ブレーク・ポイントのある文をデバッグします。

ブレーク・ポイントのある文をデバッグするには、文を右クリックし、「**デバッグの開始**」を選択します。デバッグ中の文は、ハイライト表示されます。文のメンバー、ブレーク・ポイント、および実行前後のメンバー交差の値が、次のタブに表示されます。

- **メンバー** - デバッグ・ブレーク・ポイントの現在のメンバー交差が表示されます。次のメンバー交差を表示するには、「**デバッグの再開**」をクリックします。
- **ブレーク・ポイント** - ブレーク・ポイントを含むスクリプトの式が表示されます。「**ブレーク・ポイントでの値**」タブに、式メンバーが、デバッグの実行前後の値とともに表示されます。

ヒント:

デバッグを再開するには、をクリックします。すべてのメンバー交差をデバッグするまで、をクリックし続ける必要があります。すべてのメンバー交差をデバッグすると、「スクリプトのデバッグが完了しました」というメッセージが表示されます。

ビジネス・ルール内のコンポーネントの無効化

ビジネス・ルール内のコンポーネントを無効にできます。たとえば、コンポーネントを検証から除外し、どのコンポーネントが検証エラーの原因になっているかを分析できます。

ビジネス・ルール内のコンポーネントを無効にするには:

1. ビジネス・ルール・フロー・チャートでコンポーネントを選択します。
2. 「**プロパティ**」で、「**使用不可**」を選択します。

コンポーネントを無効にし、そのコンポーネントが属するビジネス・ルールを保存すると、ビジネス・ルール・フロー・チャートでそのコンポーネントのアイコンが無効になります。無効化されたコンポーネントに移動または作成されるコンポーネントでは、無効化されたコンポーネントの「使用不可」のステータスが継承され、ビジネス・ルールの一部として計算されません。

ノート:

条件コンポーネントを使用不可にすることはできません。共有されているコンポーネントを無効化すると、そのコンポーネントを無効化したビジネス・ルールでのみ使用不可になります。

Workforce Planning、Capital Asset Planning および Human Capital Management の事前定義済ビジネス・ルールのロード

Oracle Hyperion Planning アプリケーション管理を使用して作成された Oracle Hyperion Workforce Planning、Oracle Hyperion Capital Asset Planning または Oracle Fusion Human Capital Management のアプリケーションを使用している場合、Planning で作成および初期化される際に、事前定義済ビジネス・ルールがアプリケーションにロードされます。これらの事前定義済ビジネス・ルールを使用すると、アプリケーションで Web フォームを使用する場合に事前定義済の計算を実行できます。

これらの事前定義済ビジネス・ルールを変更または削除したら、これらを Oracle Hyperion Calculation Manager にリロードする必要があります。次の手順は、事前定義済ビジネス・ルールを、Wrkforce、Capex または Human Capital Planning プラン・タイプ(いずれかまたはすべて)が使用可能な Planning アプリケーションにロードする場合に使用します。

事前定義済ビジネス・ルールをロードするには:

1. 「システム・ビュー」で、Planning アプリケーション・タイプを展開し、事前定義済のビジネス・ルールをロードするアプリケーションを右クリックし、「事前定義済ルールのロード」を選択します。

ビジネス・ルールがプラン・タイプにロードされます。

2. 「OK」をクリックし、事前定義済ルールが正常にロードされたことを示すメッセージを閉じます。

事前定義済ルールが、ルールをロードしたアプリケーションおよびプラン・タイプの「ルール」フォルダに表示されます。

ビジネス・ルールへのショートカットの作成

ショートカットを利用すると、アプリケーションごとにビジネス・ルールのコピーを作成しなくても複数のアプリケーション間でビジネス・ルールを共有できます。ルールへのショートカットを作成する場合、ショートカットではルールへの相互参照が提供されます。

ショートカットを使用して、ビジネス・ルールが作成されたアプリケーション以外のアプリケーションにそのルールを配置することもできます。デフォルトでは、ビジネス・ルールをデプロイするとそのルールが作成されたアプリケーションにデプロイされます。

ビジネス・ルールのショートカットを作成するには:

1. 「システム・ビュー」でルールを右クリックし、「コピー先」を選択します。
2. 「ショートカットの作成」で、ルールへのショートカットを作成する対象のアプリケーションを展開します。
3. ショートカットを作成する計算タイプ、プラン・タイプおよびデータベースの横にあるオプション・ボタンを選択します。

同じアプリケーション・タイプに属する複数のアプリケーションの複数の計算タイプ、プラン・タイプおよびデータベースを選択できます。

ビジネス・ルールの名前は、アプリケーション内で一意の名前にする必要があります。そのため、ビジネス・ルールが作成されたアプリケーションは、「ショートカットの作成」ダイアログ・ボックスに表示されません。ビジネス・ルールのショートカットは、別のアプリケーション内でのみ作成できます。また、ショートカットを作成するビジネス・ルールと同名のビジネス・ルールがアプリケーションに含まれている場合、そのアプリケーションは「ショートカットの作成」ダイアログ・ボックスに表示されません。

4. 「OK」をクリックします。

ルールへのショートカットは、選択された計算タイプ、プラン・タイプまたはデータベースにコピーされます。ショートカットは、ルール・アイコンの左下に上向き矢印として表示されます。

ビジネス・ルールまたはビジネス・ルールセットのリフレッシュ

システム・ビュー、カスタム・ビューおよびデプロイメント・ビューでは、アプリケーション・リストのレベルをリフレッシュできます。

アプリケーション・タイプ(Oracle Hyperion Financial Management、Oracle Hyperion Planning または Oracle Essbase)、アプリケーション、計算タイプ、プラン・タイプ、またはデータベース、複数のルールセットやルール、あるいは1つのルールセットやルールをリフレッシュできます。

デフォルトで、アプリケーション・リストのすべてのノードをリフレッシュする場合は、それに属するすべてのルール、ルールセット、コンポーネントおよびテンプレートがリフレッシュされます。ただし、アプリケーション内でルールセットまたはルールをリフレッシュしても、アプリケーション・リストまたは他のアプリケーションに属するルールセットやルールの上位レベルはリフレッシュされません。

たとえば、Planning アプリケーションとプラン・タイプ内のルールをリフレッシュすると、そのアプリケーションとプラン・タイプ内のすべてのルールはリフレッシュされますが、他のプラン・タイプや他の Planning アプリケーション内のルールはリフレッシュされません。

ビジネス・ルールまたはルールセットをリフレッシュするには、そのルールまたはルールセットを右クリックして、「リフレッシュ」を選択します

ヒント:

リフレッシュするビジネス・ルールを含む Planning アプリケーション・タイプ、アプリケーション、またはプラン・タイプを右クリックして、「リフレッシュ」を選択することもできます。

ビジネス・ルールまたはルールセットの使用状況の表示

ビジネス・ルールまたはビジネス・ルールセットを使用しているルール、テンプレートおよびルールセットを表示します。

ルールまたはルールセットの使用状況の表示は、ルールまたはルールセットを削除する場合やこれを使用しているオブジェクトを確認する場合に役立ちます。

ビジネス・ルールまたはルールセットの使用状況を表示するには:

1. ルールまたはルールセットを右クリックし、「**使用状況の表示**」を選択します。
2. 「**使用状況**」ダイアログ・ボックスで情報を確認し、「**OK**」をクリックします。

オブジェクトの所有者の変更

オブジェクト(すなわち、ルール、ルールセット、テンプレート、式またはスクリプト・コンポーネント)が属するアプリケーションがデプロイ済であれば、システム・ビューでオブジェクトの所有者を変更できます。デフォルトでは、オブジェクトを作成したユーザーが所有権を変更しないかぎり、そのユーザーがオブジェクトの所有者です。ユーザーが編集できるのは自分が所有しているオブジェクトのみですが、管理者は例外でどのオブジェクトでも編集できます。

オブジェクトの所有者を変更するには:

1. 「**システム・ビュー**」で、アプリケーション・タイプとアプリケーションを展開します。
2. 次のいずれかのタスクを行います:
 - **Oracle Hyperion Financial Management ユーザーのみ:** ルールセットの所有権を変更するには、計算タイプおよび「**ルールセット**」を展開します。
 - **Oracle Hyperion Planning、Oracle Essbase および Oracle General Ledger ユーザーのみ:** ルールセットの所有権を変更するには、「**ルールセット**」を展開します。

ノート:

Financial Management アプリケーションの場合、アプリケーション内の計算タイプごとに「**ルールセット**」ノードがあります。
Planning、Essbase および Oracle General Ledger アプリケーションの場合、各アプリケーションに対して「**ルールセット**」ノードが1つのみあります。

- ルールの所有権を変更するには、計算タイプ、プラン・タイプまたはデータベースを展開し、「**ルール**」を展開します。
- 式コンポーネント、スクリプト・コンポーネントまたはカスタム・テンプレートの所有権を変更するには、計算タイプ、プラン・タイプまたはデータベース

を展開して、所有権を変更するオブジェクトに応じて、「式」、「スクリプト」または「テンプレート」を展開します。

3. オブジェクトを右クリックして、「所有者の変更」を選択します。
4. 「所有者の変更」で、オブジェクトの所有権を移す先の所有者を選択します。
5. 「OK」をクリックします。

所有権を割り当てられたユーザーはオブジェクトを編集できます。

3

ビジネス・ルールセットの設計

次も参照:

- [ビジネス・ルールセットについて](#)
同時または順次起動できる複数のビジネス・ルール(またはビジネス・ルールセット)を組み合わせることでビジネス・ルールセットを作成します。
- [ビジネス・ルールセットの作成](#)
ビジネス・ルールセットは、システム・ビュー、フィルタ・ビュー、カスタム・ビュー、デプロイメント・ビュー、ルールセット・デザイナーおよびテンプレート・デザイナーから作成できます。
- [ビジネス・ルールセットを開く](#)
ビジネス・ルールセットは、システム・ビュー、フィルタ・ビュー、デプロイメント・ビューおよびカスタム・ビューから開くことができます。
- [ビジネス・ルールセット内のビジネス・ルールを開く](#)
ビジネス・ルールはビジネス・ルールセットから開くことができます。
- [ビジネス・ルールのビジネス・ルールセットへの追加](#)
ビジネス・ルールをビジネス・ルールセットに追加すると、ルールセットのルールを順次または同時に起動できます。
- [ビジネス・ルールセットからのビジネス・ルールの除去](#)
ビジネス・ルールをビジネス・ルールセットから除去しても、そのルールは消去されません。ルールセットとは関係なくデータベース内に存在します。
- [ビジネス・ルールセットの別のアプリケーションへのコピー](#)
ビジネス・ルールセットを別のアプリケーションにコピーする場合、同じアプリケーション・タイプにコピーすることを確認してください。
- [ビジネス・ルールセットの保存](#)
ビジネス・ルールセットを保存すると、作成対象のアプリケーションおよびアプリケーション・タイプに保存されます。
- [ビジネス・ルールセットの削除](#)
ビジネス・ルールセットを削除できるのは、他のビジネス・ルールセットによって使用されていない場合のみです。

ビジネス・ルールセットについて

同時または順次起動できる複数のビジネス・ルール(またはビジネス・ルールセット)を組み合わせることでビジネス・ルールセットを作成します。

ルールセットを作成できるかどうかは、割り当てられている役割によって決定されます。*Oracle Fusion Cloud EPM Oracle Planning and Budgeting Cloud* の使用を参照してください。

ルールセットを作成および保存した後、*Oracle Hyperion Calculation Manager* で検証およびデプロイします。その後、アプリケーションから起動できます。

ヒント:

ビジネス・ルールセットを作成する際は、作業しているビジネス・ルール、コンポーネント、テンプレートおよび変数を開いたままにできます。**Calculation Manager** では、ビジネス・ルールセットを作成する際にタブ間を簡単に移動できるように、タブ付きインターフェースでこれらのオブジェクトが表示されます。**Calculation Manager** 内では最大 **10** 個のタブを開けますが、最適なパフォーマンスを得るためには、**10** 個を超えるオブジェクトを同時に開かないことをお勧めします。

ノート:

ルールセットは、**Oracle Essbase** 集約ストレージまたはブロック・ストレージ・アプリケーションではサポートされません。ルールセットは、**Oracle General Ledger** で使用される **Essbase** 集約ストレージ・アプリケーションでサポートされています。

ビジネス・ルールセットの作成

ビジネス・ルールセットは、システム・ビュー、フィルタ・ビュー、カスタム・ビュー、デプロイメント・ビュー、ルールセット・デザイナーおよびテンプレート・デザイナーから作成できます。

ビジネス・ルールセットを作成するには:

1. 「**ルールセット**」を右クリックし、「**新規**」を選択します。
2. 「**新規ルールセット**」ダイアログ・ボックスで名前を入力し、アプリケーション・タイプとアプリケーションを選択して「**OK**」をクリックします。

システム・ビューからルールセットを追加する場合、アプリケーション・タイプおよびアプリケーションは自動的に移入されます。

Oracle Hyperion Financial Management アプリケーション・タイプおよびアプリケーションを選択した場合は、「**計算タイプ**」を選択します。

3. 「**既存のオブジェクト**」から既存のルールおよびルールセットをドラッグし、フロー・チャートにドロップします。

Oracle Hyperion Planning および **Oracle Essbase** ルールセットには、異なるアプリケーションで作成してデプロイされたルールおよびルールセットを含められません。**Financial Management** ルールおよびルールセットは、同じ計算タイプに属するか、汎用計算タイプに属している必要があります。

Financial Management アプリケーション内のルールはルールセット内で連続して起動されるため、ルールの順序は重要です。

4. 次のいずれかのタスクを実行します:
 - ルールセットのルールの順序を変更するには、上下の矢印をクリックするか、ルールを右クリックして「**上へ移動**」または「**下へ移動**」を選択します。

- ルールセットからルールを削除するには、ルールを右クリックし、「**削除**」を選択します。
 - 編集するビジネス・ルールを開くには、ルールを右クリックし、「**開く**」を選択します。
5. 「**プロパティ**」で、ルールセットのプロパティを入力します。

ルールセット・デザイナーで、作成しているルールセット内のルールを選択すると、ルールセットのプロパティではなくそのプロパティが「**プロパティ**」に表示されます。

ルールセット内のルールまたはルールセットを同時に起動する場合は、「**並列実行の使用可能**」を選択します。デフォルトでは、1つのルールセットに属するルールおよびルールセットは順次起動します。そのルールセット内の各ルールおよびルールセットがエラーなく実行されないと、次のルールもルールセットも起動されません。

ネストされた複数のルールセットがルールセットに含まれ、これらのネストされたルールセットの「**並列実行の使用可能**」設定が親のルールセットと異なる場合は、ネストされたルールセットの設定が適用されます。たとえば、ルールセット 1(並列処理のフラグが付いている)があり、それにルール 1、ルール 2 およびルールセット 2(順次処理のフラグが付いている)が含まれる場合、ルールセット 1 に並列処理のフラグが付いていても、ルールセット 2 内のルールおよびルールセットは順次処理されます。

Financial Management ユーザーのみ:

- ルールセットのルールにログ・テキストがある場合は、ルールセットの起動時にこのログ・テキストがログ・ファイルに含められるように、「**ロギングの使用可能**」を選択します。ルール、ルールセットおよびコンポーネントのロギングを使用可能にできます。ただし、ビジネス・ルールの式の文とともに含めるログ・テキストがあれば指定します。

ノート:

ルールセットに対してはロギングが使用不可になっているが、そのルールに含まれるルールおよびコンポーネントに対しては使用可能になっている場合、ルールおよびコンポーネントのログ設定よりもルールセットのログ設定の方が優先されるため、ログ・ファイルは作成されません。

- ルールセットの起動時にログ・ファイルにルールセットの処理に要した時間が記録されるように、「**タイマーの使用可能**」を選択します。タイマーは、ルール、ルールセットおよびコンポーネントに対して使用可能にできます。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能なルールセットがあり、このルールセットにタイマーを使用可能な3つのルールが含まれている場合、ルールセットおよびこのルールセットの各ルールの処理に要した時間が記録されます。
- ルールセットを Financial Management で起動できるように、「**起動の使用可能**」を選択します。

ノート:

ビジネス・ルールは Financial Management では起動できません。

Oracle General Ledger ユーザーのみ: 「開始日」 および 「終了日」 機能を使用して、ビジネス・ルールセットを起動する日付範囲を指定します。この日付範囲外でビジネス・ルールセットを起動できません。

- ビジネス・ルールセットを起動する際の開始日を選択できるカレンダーにアクセスするには、「開始日」を選択します。
- ビジネス・ルールセットを起動する際の終了日を選択できるカレンダーにアクセスするには、「終了日」を選択します。

6. 「使用状況」 タブで、このルールを使用するルールセットがあれば、それを確認します。

デフォルトでは、作成した時点ではルールセットは他のルールセットによって使用されていないため、ビジネス・ルールセットを作成した当初はこのタブは空です。

ルールセットのルールを選択する場合は、ビジネス・ルールセットの名前を確認できます。

7. 「変数」 タブで、ルールセットで使用する変数に関する情報を確認および入力します。

ルールの起動時に各変数の最初のインスタンスのみが表示されるように、「変数のマージ」を選択して、このルールセット内のルールで使用される同じ変数のすべてのインスタンスをマージします。変数をマージしない場合、各変数のすべてのインスタンスが表示されます。

「変数のマージ」を選択する場合、ユーザーが実行時プロンプトで入力する最初の値が、検証および起動時に、後続のすべての実行時プロンプトで使用されます。

 **ノート:**

変数をマージできるのは、Planning、Essbase または Oracle General Ledger の変数です。

8.  をクリックします。

ビジネス・ルールセットを開く

ビジネス・ルールセットは、システム・ビュー、フィルタ・ビュー、デプロイメント・ビューおよびカスタム・ビューから開くことができます。

ビジネス・ルールセットを開くには、次のいずれかを行います。

- ルールセットを右クリックし、「開く」を選択します。
- ルールセットをダブルクリックします。

ビジネス・ルールセット内のビジネス・ルールを開く

ビジネス・ルールはビジネス・ルールセットから開くことができます。

- システム・ビューで「ルールセット」を展開し、目的のルールセットを展開して目的のルールをダブルクリックします。

- ビジネス・ルールセットを編集用に開いた状態で、ルールセット内の目的のルールをダブルクリックします。

ビジネス・ルール・ビジネス・ルールセットへの追加

ビジネス・ルールをビジネス・ルールセットに追加すると、ルールセットのルールを順次または同時に起動できます。

ビジネス・ルールは、同じアプリケーション・タイプに属するビジネス・ルールセットに追加できます。ルールセット内のルールは **Oracle Hyperion Financial Management** で順次に起動されるため、ルールセット内でのルールの順序は重要です。他のアプリケーション・タイプに属するルールセットでは、ルールセットのルールは順次または同時に起動できます。

ビジネス・ルールをビジネス・ルールセットに追加するには:

1. ルールセットを開きます。
2. 「既存のオブジェクト」から既存のルールをルールセットにドラッグします。
3.  をクリックします。

ビジネス・ルールセットからのビジネス・ルールの除去

ビジネス・ルールをビジネス・ルールセットから除去しても、そのルールは消去されません。ルールセットとは関係なくデータベース内に存在します。

ビジネス・ルールをビジネス・ルールセットから除去するには:

1. ルールセットを開きます。
2. 目的のルールを右クリックし、「**削除**」を選択します。

ビジネス・ルールセットの別のアプリケーションへのコピー

ビジネス・ルールセットを別のアプリケーションにコピーする場合、同じアプリケーション・タイプにコピーすることを確認してください。

アプリケーション・タイプに加えて、計算タイプ(Oracle Hyperion Financial Management ルールセットの場合)、プラン・タイプ(Oracle Hyperion Planning ルールセットの場合)またはデータベース(Oracle General Ledger で使用される Oracle Essbase ルールセットの場合)が同じアプリケーションに、ビジネス・ルールセットを必ずコピーしてください。

 ノート:

ビジネス・ルールセットは、Oracle General Ledger で使用される集約ストレージ・アプリケーションを除き、Essbase 集約ストレージまたはブロック・ストレージ・アプリケーションではサポートされません。

ビジネス・ルールセットを別のアプリケーションにコピーするには:

1. **システム・ビュー**でビジネス・ルールセットを右クリックし、「**コピー先**」を選択します。
2. 「**コピー先**」に、ビジネス・ルールセットの新しい名前を入力するか、デフォルト名を受け入れて、アプリケーション、計算タイプ、プラン・タイプまたはデータベースを選択します。

複数のアプリケーションおよび計算またはプラン・タイプにビジネス・ルールセットをコピーできません。

3. 共有コンポーネントをコピーするかどうか、および既存のコンポーネントを同じ名前前で置換するかどうかを選択します。

これらのオプションは、新しいアプリケーションにコピーするビジネス・ルールセットに共有コンポーネントがある場合のみ有効です。

4. 「**OK**」をクリックします。

選択したアプリケーションおよび計算またはプラン・タイプに新規ビジネス・ルールセットが追加されます。これをシステム・ビューで確認する際、アプリケーション・リストのリフレッシュが必要な場合があります。アプリケーション・リストをリフレッシュするには、ツールバーの「**リフレッシュ**」アイコンをクリックします。「ルールセット」ノードまたはアプリケーション・リストのその上のレベルをリフレッシュして新規ルールセットを確認することもできます。[ビジネス・ルールまたはビジネス・ルールセットのリフレッシュ](#)を参照してください。

ビジネス・ルールセットの保存

ビジネス・ルールセットを保存すると、作成対象のアプリケーションおよびアプリケーション・タイプに保存されます。

ルールセットの保存後は、それをデプロイ、検証および起動できます

Oracle Hyperion Calculation Manager でビジネス・ルールセットのデプロイと検証を行うことができます。Oracle Hyperion Financial Management (Financial Management ルールセットの場合)、Oracle Hyperion Planning (Planning ルールセットの場合)、または Oracle General Ledger (Oracle General Ledger で使用される Oracle Essbase ルールセットの場合のみ)から起動できます。

ビジネス・ルールセットを作成または編集した後に保存するには、 をクリックします。

 ノート:

ビジネス・ルールセットを保存した後**システム・ビュー**で確認するには、アプリケーション・リストのリフレッシュが必要な場合があります。これを行うには、システム・ビューで任意のノードまたはオブジェクトを右クリックし、「**リフレッシュ**」を選択します。

ビジネス・ルールセットの削除

ビジネス・ルールセットを削除できるのは、他のビジネス・ルールセットによって使用されていない場合のみです。

ルールセットが使用されている場合は、それを使用しているビジネス・ルールセットから除去するか、使用側のビジネス・ルールセット用のコピーを作成しないと、そのビジネス・ルールセットは削除できません。

ビジネス・ルールセットを削除するには:

1. **システム・ビュー**でルールセットを右クリックし、「**使用状況の表示**」を選択して他のルールセットによって使用されていないことを確認します。
2. ルールセットが別のルールセットに使用されていない場合、ルールセットを右クリックし、「**削除**」を選択します。
3. 「**OK**」をクリックしてルールセットの削除を確認します。

4

システム・テンプレートの操作

次も参照:

- [システム・テンプレートについて](#)
システム・テンプレートは、計算を実行する事前定義済テンプレートです。
- [システム・テンプレートの表示](#)
システム・テンプレートが表示される場所は、グラフィカル・テンプレートとルール・テンプレートで異なります。
- [システム・テンプレートの使用](#)
システム・テンプレートを使用するには、テンプレートを選択して、フロー・チャートまたはスクリプトまでドラッグします。
- [テンプレート・フローの表示](#)
システム・テンプレートで生成したスクリプトを確認する際、テンプレート・フローを確認すると有用な場合があります。
- [システム・テンプレートのカスタム・テンプレートとしての保存](#)
システム・テンプレートのコンテンツをカスタマイズできます。
- [ビジネス・ルールからのシステム・テンプレートの削除](#)
ビジネス・ルールからシステム・テンプレートを削除できます。

システム・テンプレートについて

システム・テンプレートは、計算を実行する事前定義済テンプレートです。

システム・テンプレートによって実行される計算には、データのクリア、データのコピー、金額、数量およびレートの計算、データの配布、値の割当て、データの集計、パフォーマンスを最大化するスクリプト・コマンドの入力および勘定値のレポート通貨への変換が含まれます。

次のものにシステム・テンプレートを含めることができます。

- グラフィカルまたはスクリプト・ビジネス・ルール
- グラフィカルまたはスクリプト・カスタム・テンプレート

ビジネス・ルールまたはカスタム・テンプレートのコンポーネントとして、システム・テンプレートには、パラメータを入力する一連のステップが含まれます。これらのパラメータをテンプレート・ロジックと組み合わせて、ビジネス・ルールまたはテンプレート内で計算スクリプトを生成します。これによって、テンプレートの使用が容易になり、計算スクリプトの構文エラーの可能性が低減します。

Oracle Hyperion Calculation Manager では、すべてのシステム・テンプレートをウィザードとして使用できます。ウィザードのシステム・テンプレートを使用すると、テンプレートの使用が容易になり、計算スクリプトの構文エラーの可能性が低減します。

システム・テンプレートのプロンプトは、ウィザードでの選択に基づいてフィルタ処理されます。たとえば、集約システム・テンプレートは、密ディメンションを選択して集約するステップを表示しますが、集約に使用可能な密ディメンションがない場合、完全または部分密ディメンションのステップは表示されません。

 ノート:

システム・テンプレートは、Oracle Hyperion Financial Management、Oracle Hyperion Planning ブロック・ストレージ・アプリケーションおよび Oracle Essbase ブロック・ストレージ・アプリケーションで使用できます。Essbase ASO アプリケーションではシステム・テンプレートはサポートされません。

システム・テンプレートの表示

システム・テンプレートが表示される場所は、グラフィカル・テンプレートとルール・テンプレートで異なります。

グラフィカル・ルールまたはテンプレートを作成したり、開いた場合、システム・テンプレートは次のいずれかで表示されます。

- **新規オブジェクト** - システム・テンプレートは「**新規オブジェクト**」では個別のオブジェクトとして表示されます。
- **既存のオブジェクト** - システム・テンプレートは「**既存のオブジェクト**」では「**システム・テンプレート**」ヘッダーの下に表示されます。

スクリプト・ルールまたはテンプレートを作成したり、開いた場合、システム・テンプレートは「**既存のオブジェクト**」にのみ表示されます。

システム・テンプレートのウィザードにアクセスするには:

- グラフィカル・ルールまたはテンプレートの場合、目的のテンプレートを選択し、フロー・チャートの「**開始**」と「**終了**」の間にドラッグします。
- スクリプト・ルールまたはテンプレートの場合、テンプレートを選択し、スクリプトにドラッグします。

システム・テンプレートの使用

システム・テンプレートを使用するには、テンプレートを選択して、フロー・チャートまたはスクリプトまでドラッグします。

次も参照:

- **「データのクリア」テンプレートの使用**
「データのクリア」テンプレートを使用してキューブ内のメンバーからデータをクリアします。
- **「データのコピー」テンプレートの使用**
「データのコピー」テンプレートを使用して、データベース内のある場所から別の場所にデータをコピーします。
- **「金額-単価-レート」テンプレートの使用**
「金額-単価-レート」テンプレートを使用して、金額、数量およびレートに入力した3つのメンバーのうちの1つを計算します。
- **「割当て - レベル間」テンプレートの使用**
「レベル間割当て」テンプレートを使用して、データベース・アウトラインであるレベルから別のレベルに割り当てます。

- 「**割当て**」テンプレートの使用
「割当て」テンプレートを使用して、ソースから宛先に均等にまたは指定したドライバに基づいて値を割り当てます。
- 「**集約**」テンプレートの使用
「集約」テンプレートを使用して、指定したメンバーのデータ値を集約します。
- 「**SET コマンド**」テンプレートの使用
「SET コマンド」テンプレートを使用して、計算スクリプトの実行を最適化するスクリプト・コマンドを入力します。
- 「**通貨換算**」テンプレートの使用
「通貨換算」テンプレートでは、システム・タイプの勘定科目または選択した勘定科目を使用して、勘定科目の値をレポート通貨に換算します。
- 「**データのエクスポート-インポート**」テンプレートの使用

「データのクリア」テンプレートの使用

「データのクリア」テンプレートを使用してキューブ内のメンバーからデータをクリアします。

データをクリアするには、クリアする値を持つメンバーを指定します。1つのメンバーまたはメンバーのブロックに対応するデータをクリアできます。

たとえば、「データのクリア」テンプレートを使用して、データを"実績"から"予測"にコピーして変更する前に予測データをクリアできます。

「データのクリア」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「**新規オブジェクト**」で「**データのクリア**」を選択し、フロー・チャートの「**開始**」と「**終了**」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「**システム・テンプレート**」を展開して「**データのクリア**」を選択し、スクリプトの目的の領域にドロップします。
3. **データのクリア・ウィザード**で要求された情報を入力します。

表 4-1 データのクリア・ウィザードのステップ

ステップ	説明
データ選択	<p>ディメンションごとに 1 つ以上のメンバーを選択し、クリアするデータを定義します。</p> <p>次のオプションを使用できます:</p> <ul style="list-style-type: none"> 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーと関数を選択します。 ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。 <p>「CLEARBLOCK オプションを選択します」で、データのクリア方法を定義するオプションを選択します。</p> <ul style="list-style-type: none"> すべて - すべてのブロックのデータをクリアします 上位 - 集計データ・ブロックのみをクリアします。 動的 - 「動的計算および保管」メンバーの組合せから導出した値を含むブロックをクリアします 非入力 - 導出した値を含むデータ・ブロックをクリアします。これは、完全に計算操作によって作成されるブロックに対してのみ動作し、任意の値がロードされるブロックに対しては動作しません。 N/A - データ・ブロックをクリアし、メンバーを #Missing に設定します
設定	<p>HSP_Rates が "疎" に設定された複数通貨アプリケーションまたは単一通貨アプリケーションがある場合、「設定」に「データ選択」で選択したメンバーが表示されます。</p> <p>ドロップダウンをクリックし、単一メンバーが選択されているディメンション(密ディメンションを推奨)を選択します。</p> <p>ノート: スクリプトでは選択したディメンションのメンバーが使用されるため、単一メンバーが選択されるディメンションを選択できない場合は、このテンプレートを使用できません。</p>

「データのコピー」テンプレートの使用

「データのコピー」テンプレートを使用して、データベース内のある場所から別の場所にデータをコピーします。

たとえば、「データのコピー」テンプレートを使用して、2014 年の勘定科目およびエンティティの値をコピーすることによって 2015 年の予算を作成できます。この場合は、2014 年の実績を 2015 年の予測にコピーします。「データのコピー」テンプレートを使用して、予算の"最悪"のシナリオから"最善"のシナリオに予算データをコピーし、変更を加えることもできます。

「データのコピー」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「新規オブジェクト」で「データのコピー」を選択し、フロー・チャートの「開始」と「終了」の間にドロップします。

- スクリプト・ルールまたはテンプレートの場合、「システム・テンプレート」を展開して「データのコピー」を選択し、スクリプトの目的の領域にドロップします。

3. データのコピー・ウィザードで要求された情報を入力します。

表 4-2 データのコピー・ウィザードのステップ

ステップ	説明
情報 視点	<p>テンプレートの関数を説明します。</p> <p>ディメンションごとに 1 つ以上のメンバーを選択し、コピーするデータを定義します。コピー元およびコピー先によって変わるディメンションは空のままにします。これらのメンバーを次のステップで指定します 次のオプションを使用できます:</p> <ul style="list-style-type: none"> • 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 • 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーと関数を選択します。 • ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。
コピー元	<p>データのコピー元のメンバーを選択します。</p> <p>ノート: ディメンションごとに 1 つのメンバーを選択します。ディメンションごとに 1 つのメンバーを選択しない場合、警告メッセージが表示されます。</p>
コピー先 オプション	<p>データのコピー先のメンバーを選択します。</p> <p>オプション: 宛先のデータ・セルの範囲に適用する増減のパーセンテージを入力します。</p> <p>たとえば、ターゲット・データを 10 パーセント増加させる場合は 10 と入力し、ターゲット・データを 10 パーセント減少させる場合は -10 と入力します。</p> <p>変数を入力するには、 をクリックします。たとえば、実行時プロンプト変数を定義し、実行時にこの増減の値を入力できます。</p>

「金額-単価-レート」テンプレートの使用

「金額-単価-レート」テンプレートを使用して、金額、数量およびレートに入力した 3 つのメンバーのうちの 1 つを計算します。

たとえば、製品の収益を計算する必要があり、個数と価格がわかっている場合、「**金額の計算**」を選択し、金額に製品の収益、数量に個数、レートに価格を入力します。個数に価格を掛けて製品の収益が計算されます。

「金額-単価-レート」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「新規オブジェクト」で「金額-単価-レート」を選択し、フロー・チャートの「開始」と「終了」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「システム・テンプレート」を展開して「金額-単価-レート」を選択し、スクリプトの目的の領域にドロップします。
3. **金額-単価-レート・ウィザード**で要求された情報を入力します。

表 4-3 金額-単価-レート・ウィザードのステップ

ステップ	説明
情報	テンプレートの関数を説明します。
計算方法の選択	<p>ドロップダウン・リストから次のいずれかのオプションを選択し、使用する計算方法を定義します。</p> <ul style="list-style-type: none"> • 金額の計算 - Amounts = Units * Rates • 数量の計算 - Units = Amounts / Rates • レートの計算 - Rates = Amounts / Units • すべて計算 - #Missing のメンバーが変わったり、不明の場合に選択します。この計算方法では、#Missing 値を検索し、適切な計算方法を適用して不明の値を解決します。
視点	<p>割当てプロセスで変更されないメンバーを定義します。メンバーの選択に、次のオプションを使用できます。</p> <ul style="list-style-type: none"> • 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 • 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーを選択します。 • ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。
金額の計算/数量の計算/レートの計算/すべて計算	<p>このステップは、計算方法に選択した内容によって異なります。</p> <ul style="list-style-type: none"> • 「金額の計算」を選択した場合、メンバーを入力するか、 をクリックして金額のメンバーを選択し、「次」をクリックして数量のメンバーを入力し、「次」をクリックしてレートのメンバーを入力します。「数量」および「レート」のステップでは、視点と異なるメンバーを入力できます。 • 「数量の計算」を選択した場合、メンバーを入力するか、 をクリックして数量のメンバーを選択し、「次」をクリックして金額のメンバーを入力し、「次」をクリックしてレートのメンバーを入力します。「金額」および「レート」のステップでは、視点と異なるメンバーを入力できます。 • 「レートの計算」を選択した場合、メンバーを入力するか、 をクリックしてレートのメンバーを選択し、「次」をクリックして金額のメンバーを入力し、「次」をクリックして数量のメンバーを入力します。「金額」および「数量」のステップでは、視点と異なるメンバーを入力できます • 「すべて計算」を選択した場合、金額、数量およびレートのメンバーを選択します。

「割当て - レベル間」テンプレートの使用

「レベル間割当て」テンプレートを使用して、データベース・アウトラインであるレベルから別のレベルに割り当てます。

たとえば、「割当て - レベル間」テンプレートを使用して、製品をあるレベルから別のレベルに割り当てます。次のような製品アウトラインがあるとします。

```
T_TP
  P_000
    T_TP1
      P_100
        P_110
```

```

etc
  T_TP2
    P_200
    P_210
etc
  T_TP3
P_291
      P_292

```

この例では、「割当て - レベル間」テンプレートでレベル/世代 3 を始まりに、レベル/世代 4 を終わりに使用し、親メンバー(T_TP1、T_TP2、T_TP3)からその子(レベル 0)メンバーに割り当てます。

「割当て - レベル間」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「新規オブジェクト」で「割当て - レベル間」を選択し、フロー・チャートの「開始」と「終了」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「システム・テンプレート」を展開して「割当て - レベル間」を選択し、スクリプトの目的の領域にドロップします。
3. 割当て - レベル間・ウィザードで要求された情報を入力します。

表 4-4 割当て - レベル間・ウィザードのステップ

ステップ	説明
視点	<p>リストされたディメンションごとに、割当て時に変更しないメンバーを 1 つ以上選択します。</p> <p>次のオプションを使用できます:</p> <ul style="list-style-type: none"> • 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 • 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーと関数を選択します。 • ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。
ソース	<p>リストされたディメンションごとに、割り当てるデータを持つメンバーを選択します。メンバーを選択したら、ソースの割当てパーセンテージを入力します。</p> <p>ノート: レベル間割当てに使用するディメンションは空にします。(次のステップでこのディメンションを選択します。)</p>
割当て範囲	<p>要求された情報を入力し、割当て範囲を定義します。</p> <p>ルールは、開始レベルとして指定されたレベルから終了レベルとして指定されたレベルまで、選択されたメンバーの下のすべてのデータを割り当てます。</p>

表 4-4 (続き) 割当て - レベル間・ウィザードのステップ

ステップ	説明
ターゲット	<ul style="list-style-type: none"> ディメンションが「ソース」ステップで定義されており、割当てに使用されていない場合、各ディメンションのターゲット・メンバーを定義します。「ソース」で選択したメンバーは、デフォルトでここに入力されます。次のいずれかのオプションを使用して、ターゲット・メンバーを定義します。 <ul style="list-style-type: none"> 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーと関数を選択します。 ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。 最適化のため、1つのメンバーのみを含むディメンションを選択してください。 上で単一メンバーを選択したディメンションを選択してください
オフセット	<p>オフセットを定義するか、オフセット計算を生成しない場合はこのステップの表を空のままにします。</p> <p>オフセットを定義する場合、次のようにします。</p> <ul style="list-style-type: none"> オフセット・ディメンションを定義します オフセット・メンバーに割り当てる値の合計を記述するには、オフセットを定義するメンバーを選択します。 割当てディメンションのオフセット・メンバーを定義します メンバー名を入力するか、 をクリックしてメンバーを選択します。
ドライバ	<p>割当て範囲の各メンバーに適用するパーセンテージの計算に使用するドライバ(基準)を定義します。</p> <ul style="list-style-type: none"> 視点と異なるディメンションのメンバーのみを選択してください。次のいずれかのオプションを使用して、メンバーを選択します。 <ul style="list-style-type: none"> 「事前に定義されている選択項目の使用」の隣の  をクリックし、ディメンションに値を移入する変数を選択します。ディメンションごとに1つのメンバーのみを選択します。 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーと関数を選択します。 ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。 割当てディメンションに対するドライバの集約を更新するかどうかを指定します。
その他オプション	<p>ウィザードの質問に答えます。</p>

「割当て」テンプレートの使用

「割当て」テンプレートを使用して、ソースから宛先に均等にまたは指定したドライバに基づいて値を割り当てます。

たとえば、「割当て」テンプレートを使用して、総売上を基準に一般管理費用を製品ディメンションのレベル0メンバーに割り当てます。

「割当て」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「新規オブジェクト」で「割当て」を選択し、フロー・チャートの「開始」と「終了」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「システム・テンプレート」を展開して「割当て」を選択し、スクリプトの目的の領域にドロップします。
3. 割当てウィザードで要求された情報を入力します。

表 4-5 割当てウィザードのステップ

ステップ	説明
情報 視点	<p>テンプレートの関数を説明します。</p> <p>割当てで変更されないメンバーを定義します。少なくとも 1 つのディメンションの 1 つのメンバーを選択する必要があります。 メンバーの選択に、次のオプションを使用できます。</p> <ul style="list-style-type: none"> • 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 • 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーを選択します。 • ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。
ソース	<p>ディメンションごとにデータの割当て元のメンバーを選択し、割り当てるソースのパーセンテージを入力します。 例, ソース値の 25% を割り当てるには、25 と入力します。</p>
割当て範囲	<p>データの割当て先のディメンションおよびこのディメンションの親ディメンションを選択します。 データは、データベース・アウトラインの指定した親メンバーの下のレベル 0 メンバーに割り当てられます。(レベル 0 メンバーはその下にメンバーのない、アウトラインで最下位レベルのメンバーです。)</p>
宛先 - ターゲット 宛先 - オフセット	<p>ソースに定義された各ディメンションのターゲット・メンバーを指定します。</p> <p>オプション。</p> <ul style="list-style-type: none"> • 各ディメンションのメンバーを入力し、オフセットを指定してオフセット計算を生成します。 オフセット値は、ソースの同じまたは反対の値として計算されます。 たとえば、ソースの \$1000 を割り当てる場合、数値を 2 倍にしないようオフセットを使用できます。ルールを実行すると、\$1000 はレベル 0 に割り当てられ、オフセットによって -1000 が指定したメンバーに作成されます。 • 割当てプロセスが開始する前にオフセット・データをクリアするかどうかを指定します。 • オフセットの値を割当てデータの反対の符号にするかどうかを指定します。
除外	<p>オプション。 割当て範囲から除外するメンバーを選択します。 これらのメンバーは割当てデータを受け取らず、そのドライバ値はドライバ・パーセンテージ基準から除外されます。</p>
ドライバ	<p>"ドライバを使用して割り当てる"か"均等に割り当てる"かを定義します。 視点と異なる各ディメンションについて、1 つのディメンション当たり 1 つのメンバーを選択します。空のままのディメンションは、視点およびルールで定義された上位メンバー範囲から継承されます</p>

表 4-5 (続き) 割当てウィザードのステップ

ステップ	説明
分散方法	「ドライバ」ステップで「均等に割当て」を選択した場合、データを分配するオプションを選択します。
その他オプション	<p>ウィザードの質問に答えます。 丸め処理方法を選択した場合、選択した丸め処理方法に応じて追加のステップがあります。次の丸め処理方法があります。</p> <ul style="list-style-type: none"> • 丸め処理メンバーの定義 - このオプションを選択した場合、次のステップはメンバーでの丸め処理で、割当てに使用する小数点以下の桁数を入力し、割当ての丸め誤差を含めるメンバーを選択します。 • 最大値の使用 - このオプションを選択した場合、次のステップは「最大値に丸める」で、割当てに使用する小数点以下の桁数を入力します。 • 最小値の使用 - このオプションを選択した場合、次のステップは「最小値に丸める」で、割当てに使用する小数点以下の桁数を入力します。 • 丸め処理なし - このオプションを選択した場合、追加のステップはありません。

「集約」テンプレートの使用

「集約」テンプレートを使用して、指定したメンバーのデータ値を集約します。

「集約」テンプレートの使用により、次のことができます:

- 視点にメンバーを入力し、集約を制限します
- 欠落した値を集約するかどうかを指定します
- 値を現地通貨に集約するかを指定します(ハイブリッド集約では未サポート)

値を集約する際、次の計算コマンドがサポートされます:

- **SET UPTOLOCAL ON:** データを現地通貨で集約します(このコマンドはハイブリッド集約ではサポートされていません。)
- **SET CACHE HIGH:** 計算機キャッシュの値を設定します(このコマンドはハイブリッド集約ではサポートされていません。)
- **SET AGGMISSG ON:** データベースの欠落値を集約します
- **SET FRMLBOTTOMUP ON:** 疎ディメンションの計算を最適化します

たとえば、「集約」テンプレートを使用し、親メンバーを使用して密または疎ディメンションを集約できます。子孫、子または祖先(すべて)などの集約のレベルを選択します。

「集約」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「**新規オブジェクト**」で「**集約**」を選択し、フロー・チャートの「**開始**」と「**終了**」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「**システム・テンプレート**」を展開して「**集約**」を選択し、スクリプトの目的の領域にドロップします。

3. 集約ウィザードで要求された情報を入力します。

表 4-6 集約ウィザードのステップ

ステップ	説明
情報	テンプレートの関数を説明します。
視点	<p>集約を制限するディメンションのメンバーを選択します。メンバーの選択に、次のオプションを使用できます。</p> <ul style="list-style-type: none"> 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーを選択します。 ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。
完全な密の集約	<p>完全に集約する密ディメンションを最大 2 個選択し、密ディメンションがレベル 0 以外のメンバーを格納したかどうかを指定します。次の点に注意してください。</p> <ul style="list-style-type: none"> 密ディメンションがレベル 0 以外のメンバーを格納した場合、スクリプトに Calc Dim(密ディメンション) が生成されます。 密ディメンションがレベル 0 以外のメンバーを格納していない場合、密ディメンションに対してスクリプトは生成されません。 密ディメンションで部分集約を行う場合、(親メンバーおよび関数 - Descendants、Children または Ancestors)、このステップでディメンションを選択しないでください。
完全な疎の集約	<p>完全に集約する疎ディメンションを最大 3 個選択し、疎ディメンションに計算の必要なメンバー式があるかどうかを指定します。疎ディメンションにメンバー式がある場合、疎ディメンションは計算されます。疎ディメンションにメンバー式がない場合、疎ディメンションは集約されます。</p>
部分的なディメンション集約 - 密	<p>部分集約する密ディメンションを最大 2 個選択します。密ディメンションごとにメンバーと集約レベルを選択します。次の点に注意してください。</p> <ul style="list-style-type: none"> 「部分的なディメンション集約 - 密」ステップは、「完全な密の集約」ステップで使用しなかった密ディメンションがある場合に表示されます。「完全な密の集約」タブで 2 つの密ディメンションを使用し、アプリケーションにこれ以外の密ディメンションがない場合、「部分的なディメンション集約 - 密」ステップは表示されません。 疎ディメンションで部分集約を行う場合、(親メンバーおよび関数 - Descendants、Children または Ancestors)、このステップでディメンションを選択しないでください。
部分的なディメンション集約 - 疎	<p>部分集約する疎ディメンションを最大 3 個選択します。疎ディメンションごとにメンバーと集約レベルを選択します。次の点に注意してください。</p> <p>ノート:</p> <ul style="list-style-type: none"> 部分集約に使用可能な疎ディメンションはグレーで表示されます。 「部分的なディメンション集約 - 疎」ステップは、「完全な疎の集約」ステップで使用しなかった疎ディメンションがある場合に表示されます。「完全な疎の集約」タブで 3 つの疎ディメンションを使用し、アプリケーションにこれ以外の疎ディメンションがない場合、「部分的なディメンション集約 - 疎」ステップは表示されません。 密ディメンションで部分集約を行う場合、(親メンバーおよび関数 - Descendants、Children または Ancestors)、このステップでディメンションを選択しないでください。
設定	このステップで設定を指定します。

「SET コマンド」テンプレートの使用

「SET コマンド」テンプレートを使用して、計算スクリプトの実行を最適化するスクリプト・コマンドを入力します。

データ・ボリューム、データ処理、メモリー使用、スレッド・スクリプト・コマンドおよびロギング・スクリプト・コマンドを含められます。

「SET コマンド」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「**新規オブジェクト**」で「**SET コマンド**」を選択し、フロー・チャートの「**開始**」と「**終了**」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「**システム・テンプレート**」を展開して「**SET コマンド**」を選択し、スクリプトの目的の領域にドロップします。
3. **SET コマンド・ウィザード**で要求された情報を入力します。

ウィザードには次のステップが含まれます。

- **データ・ボリューム**
- **データ処理**
- **メモリーの使用**
- **スレッド**
- **ロギング**

各ステップには質問があり、質問の隣のドロップダウン・リストでオプションを選択して回答できます。このテンプレートで **1** つ以上のオプションを選択できます。

ノート:

SET コマンド・ウィザードの質問にすべて答える必要はありません。状況に当てはまる質問にのみ答えることができます。

ノート:

アプリケーションが Essbase Hybrid 内の場合、**データ・ボリューム**、**データ処理**および**スレッド**・タブのみがウィザードに表示されます。

「通貨換算」テンプレートの使用

「通貨換算」テンプレートでは、システム・タイプの勘定科目または選択した勘定科目を使用して、勘定科目の値をレポート通貨に換算します。

選択した複数通貨を使用して作成された Oracle Hyperion Planning アプリケーションで「通貨換算」テンプレートを使用します。

「通貨換算」テンプレートの使用例については、「[通貨換算](#)」テンプレートの例を参照してください。

ノート:

Planning アプリケーションが複数通貨で作成されている場合、「通貨換算」テンプレートは、「システム・テンプレート」のリストに表示されません。

「通貨換算」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「**新規オブジェクト**」で「**通貨換算**」を選択し、フロー・チャートの「**開始**」と「**終了**」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「**システム・テンプレート**」を展開して「**通貨換算**」を選択し、スクリプトの目的の領域にドロップします。
3. **通貨換算ウィザード**で要求された情報を入力します。

表 4-7 通貨換算ウィザードのステップ

ステップ	説明
情報 - 通貨	テンプレートの関数を説明し、テンプレートを使用するために必要な情報を示します。

表 4-7 (続き) 通貨換算ウィザードのステップ

ステップ	説明
為替レート・オプション	<p>通貨に関するウィザードの質問に答えます。</p> <ul style="list-style-type: none"> ユーザーの通貨はどのディメンションにありますか?: 通貨ディメンションに対応するディメンションを選択します。 レポート通貨を選択 変換先のレポート通貨を選択します。 ユーザーの通貨メンバーを含む親メンバーを選択 "入力通貨"などのアプリケーションで使用されるすべての通貨の親である親メンバーを選択します。 このレポート通貨にどの為替レート(通貨)が使用されるか? 上で選択したレポート通貨に対応する通貨を選択します。たとえば、レポート通貨に「EUR レポート」を選択した場合、「EUR」を選択します(EUR は、データを EUR レポートに変換するために使用する通貨です)。 通貨換算の勘定科目タイプを使用しますか?: 通貨換算の勘定科目タイプを使用するには、「はい」を選択します。「はい」を選択した場合、費用および収益勘定科目には変換に平均為替レートが使用され、負債および資本勘定科目には期末為替レートが使用されます。 「いいえ」を選択した場合、ウィザードの後半で平均為替レートおよび期末為替レートに使用する勘定科目を指定するよう求められます 平均為替レートを計算しますか? 「平均を計算しない」を選択した場合、平均に関連するステップは表示されず、平均為替レートに基づいて通貨を計算するためのスクリプトは生成されません。 終了為替レートを計算しますか? 期末為替レートを計算するかどうかを選択します。 「期末を計算しない」を選択した場合、期末に関連するステップは表示されず、期末為替レートに基づいて通貨を計算するためのスクリプトは生成されません。
POV	<p>レポート通貨への換算で使用するメンバーを選択 メンバーの選択に、次のオプションを使用できます。</p> <ul style="list-style-type: none"> 「事前に定義されている選択項目の使用」の隣の  をクリックし、すべてのディメンションに値を移入する変数を選択します。 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーを選択します。 ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。
勘定科目 - 平均	<p>レポート通貨に変換するために平均為替レートを使用する勘定科目を入力します。 ノート: このステップは、「為替レート・オプション」ステップの「通貨換算の勘定科目タイプを使用しますか?」で「いいえ」を選択した場合のみ表示されます。</p>
勘定科目 - 期末	<p>レポート通貨に変換するために期末為替レートを使用する勘定科目を入力します。 ノート: このステップは、「為替レート・オプション」ステップの「通貨換算の勘定科目タイプを使用しますか?」で「いいえ」を選択した場合のみ表示されます。</p>
FX 平均	<p>平均為替レートを含むメンバーを選択します。 ノート: 視点のメンバーとは異なるメンバーを選択する必要があります。</p>
FX 終了	<p>期末為替レートを含むメンバーを選択します。 ノート: 視点のメンバーとは異なるメンバーを選択する必要があります。</p>

「通貨換算」テンプレートの例

1. 複数通貨を使用せずに作成された Oracle Hyperion Planning アプリケーションで、既存のディメンションに加えて、"通貨"という名前のカスタム・ディメンションを作成し、次のメンバーを追加します。
 - 入力通貨
 - USD
 - CAD
 - EUR
 - GBP
 - INR
 - レポート通貨
 - EUR レポート
 - US レポート
 - CAN レポート
2. "勘定科目"ディメンションで、FX_Rates の次のメンバーを追加します:
 - FX_Average
 - FX_Ending
3. "エンティティ"ディメンションで、エンティティ「会社仮定」を追加します。
4. Oracle Smart View for Office または Planning フォームを使用して、次のセルに為替レートを入力します:
 - **勘定科目** - FX_Average および FX_Ending
 - **期間** - 月ごとのレートを入力します
 - **年** - FY12
 - **シナリオ** - 現在
 - **バージョン** - BU Version_1
 - **エンティティ** - 会社仮定
 - **製品** - 000
 - **通貨** - 各通貨の為替レートを入力します
5. 「変数」で、次のようにします。
 - a. 「Reporting_Currency」という名前の通貨ディメンションのメンバー変数を使用してアプリケーション・レベルの実行時プロンプトを作成し、「EUR レポート」のデフォルト値を入力します。
 - b. 「Input Currencies」という名前の通貨ディメンションのメンバー変数を使用してアプリケーション・レベルの実行時プロンプトを作成し、「EUR」のデフォルト値を入力します。

- c. この Planning アプリケーションにビジネス・ルールを作成し、「通貨換算」システム・テンプレートをビジネス・ルールのフロー・チャートにドラッグ・アンド・ドロップします。

「通貨換算」ウィザードが表示されます。

6. 通貨換算ウィザードで、次のステップを実行します。

a. **ステップ 1 – 為替レート・オプション**

- ユーザーの通貨はどのディメンションにありますか?: ドロップダウンをクリックし、前述の手順で作成した通貨ディメンションを選択します。
- レポート通貨を選択:  をクリックし、「**変数**」を選択して Reporting_Currency 変数を選択します。

この例では、レポート通貨ごとの Planning でルールを起動できるように、レポート通貨の変数を使用しています。

- ユーザーの通貨メンバーを含む親メンバーを選択:  をクリックし、Input Currencies 親メンバーを選択します。

通貨メンバーの親を選択する必要がある理由は、今後通貨を追加した場合に、Planning でこのビジネス・ルールを再デプロイし、再起動するだけで済むようにするためです。テンプレートのスクリプトによって、新しい通貨がスクリプトに追加され、レポート通貨が正しく計算されます。

- このレポート通貨にどの為替レート(通貨)が使用されるか:  をクリックし、Input Currencies 変数を選択します。

この例では、レポート通貨ごとの Planning でルールを起動できるように、レポート通貨の変数を使用しています。

- 通貨換算の勘定科目タイプを使用しますか?: ドロップダウンを使用して、「**はい**」を選択します。

「いいえ」を選択すると、後半のステップで平均/期末を使用する勘定科目を指定するよう求められます。

- 平均為替レートを計算しますか?
ドロップダウンを使用して、「**平均の計算**」を選択します。
- 終了為替レートを計算しますか?
ドロップダウンを使用して、「**期末の計算**」を選択します。

b. **ステップ 2 – POV**

各ディメンションについて次の情報を入力します。

- **期間** - すべての期間に書き込まれるように、期間は空のままにします
- **年** - "FY12"
- **シナリオ** - "現在"
- **バージョン** — "BU Version_1"
- **エンティティ** - @Relative("South",0)
- **製品** - @Relative("Tennis",0),@Relative("Golf",0)

c. **ステップ 3 – FX 平均**

- 平均為替レートを含むメンバーを選択:
 - 勘定科目 - "FX_Average"
 - 期間
 - 年
 - シナリオ
 - バージョン
 - エンティティ - "会社仮定"
 - 製品 - "000"
- 各期間の為替レートを使用するように、期間ディメンションは空のままにします。
- 年、シナリオおよびバージョンは、これらのディメンションの POV 入力からのメンバーを使用するように、空のままにします。

d. ステップ 4 – FX 終了

- 平均為替レートを含むメンバーを選択:
 - 勘定科目 - "FX_Average"
 - 期間
 - 年
 - シナリオ
 - バージョン
 - エンティティ - "会社仮定"
 - 製品 - "000"
- 各期間の為替レートを使用するように、期間ディメンションは空のままにします。
- 年、シナリオおよびバージョンは、これらのディメンションの POV 入力からのメンバーを使用するように、空のままにします。

7. ビジネス・ルールを Planning に保存、検証およびデプロイします。

この例でのアプリケーションには、次のデータが含まれています:

表 4-8 データの例 パート 1

N/A	N/A	N/A	FY12	現在	BU Version_1
N/A	N/A	Tennessee	Tennessee	Florida	Florida
N/A	N/A	1 月	1 月	1 月	1 月
N/A	N/A	G400	G100	G400	G100
総売上	USD	10750	13450	9500	9699
総売上	CAD	10450	10000	14450	13000
総売上	EUR	40000	41450	42450	65000
総売上	GBP	13250	16750	172000	16300
総売上	INR	750000	750000	750000	750000
総売上	US レポート	#Missing	#Missing	#Missing	#Missing
総売上	CAN レポート	#Missing	#Missing	#Missing	#Missing

表 4-8 (続き) データの例 パート 1

N/A	N/A	N/A	FY12	現在	BU Version_1
総売上	EUR レポート	#Missing	#Missing	#Missing	#Missing
給与	USD	5000	5000	5000	5000
給与	CAD	4500	4500	4500	4500
給与	EUR	5500	5500	5500	5705
給与	GBP	1200	1200	1200	1200
給与	INR	100000	100,000	100,000	100,000
給与	US レポート	#Missing	#Missing	#Missing	#Missing
給与	CAN レポート	#Missing	#Missing	#Missing	#Missing
給与	EUR レポート	#Missing	#Missing	#Missing	#Missing

この表では、レポート通貨に値がありません。

8. Planning で、ルールを起動します。レポート通貨変数に **EUR レポート** と入力し、入力通貨に **EUR** と入力します。「**起動**」をクリックします。
9. Planning の各勘定科目に対して、**EUR レポート** に合計がユーロで示されていることを確認します。

表 4-9 データの例 パート 2

N/A	N/A	N/A	FY12	現在	BU Version_1
N/A	N/A	Tennessee	Tennessee	Florida	Florida
N/A	N/A	1 月	1 月	1 月	1 月
N/A	N/A	G400	G100	G400	G100
総売上	USD	10750	13450	9500	9699
総売上	CAD	10450	10000	14450	13000
総売上	EUR	40000	41450	42450	65000
総売上	GBP	13250	16750	172000	16300
総売上	INR	750000	750000	750000	750000
総売上	US レポート	#Missing	#Missing	#Missing	#Missing
総売上	CAN レポート	#Missing	#Missing	#Missing	#Missing
総売上	EUR レポート	52161391	52168683	52170600	52190796.69
給与	USD	5000	5000	5000	5000
給与	CAD	4500	4500	4500	4500
給与	EUR	5500	5500	5500	5705
給与	GBP	1200	1200	1200	1200
給与	INR	100000	100000	100000	100000
給与	US レポート	#Missing	#Missing	#Missing	#Missing
給与	CAN レポート	#Missing	#Missing	#Missing	#Missing
給与	EUR レポート	6963318	6963318	6963318	6965471.887

10. Planning で、各レポート通貨について必要な回数ビジネス・ルールを起動できます。変数およびアプリケーションで使用可能な各レポート通貨について、毎回変数入力を変更し、希望のレポート通貨とこれに対応する為替レート名を入力できます。

この例では、US レポートと USD で 1 回、CAN レポートと CAD で 1 回の合計 2 回ルールを実行できます。

「データのエクспорт-インポート」テンプレートの使用

データのエクспорт-インポート・テンプレートを使用して、データベース・アウトライン内のある場所に対してデータをエクспортまたはインポートします。バイナリ・ファイルをエクспортすると、「データのエクспорт」テンプレートを使用してこのファイルをインポートすることもできます。

「データのエクспорт-インポート」テンプレートを使用するには:

1. ビジネス・ルールまたはテンプレートを作成するか、開きます。
2. 次のいずれかを行います:
 - グラフィカル・ルールまたはテンプレートの場合、「**新規オブジェクト**」で「**データのエクспорт-インポート**」を選択し、フロー・チャートの「**開始**」と「**終了**」の間にドロップします。
 - スクリプト・ルールまたはテンプレートの場合、「**システム・テンプレート**」を展開して「**データのエクспорт-インポート**」を選択し、スクリプトの目的の領域にドロップします。
3. **データのエクспорт-インポート・ウィザード**で要求された情報を入力します。

表 4-10 データのエクспорт-インポート・ウィザードのステップ

ステップ	説明
モード	データをエクспортするかインポートするかを指定します。
データ範囲のエク スポーツ	<ul style="list-style-type: none"> • エクспортするデータの範囲を指定してください。 データ範囲の指定に、次のオプションを使用できます: <ul style="list-style-type: none"> - 「事前に定義されている選択項目の使用」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。 - 「メンバー・セレクタ」の隣の  をクリックし、ディメンションごとにメンバーを選択します。 - ディメンションを選択し、 をクリックして変数、メンバーまたは関数を選択します。 • データのエクспорт先ファイルの種類を選択します: <ul style="list-style-type: none"> - フラット・ファイル - データベース表 - バイナリ・ファイル <p>ノート: ファイルをエクспортした後にインポートする場合、ファイルはバイナリ・ファイルである必要があります。</p>

表 4-10 (続き) データのエクスポート-インポート・ウィザードのステップ

ステップ	説明
出力 - ファイル	<ul style="list-style-type: none"> • フラット・ファイルをエクスポートする場合: <ol style="list-style-type: none"> 1. 引用符で囲んだファイル名とパスを入力します。 2. 列区切り記号を選択します。 3. オプション: 引用符で囲んだテキスト文字列を入力して欠落したデータ値を示すか、「アクション」アイコンを使用して選択します。 • データベース表をエクスポートする場合: <ol style="list-style-type: none"> 1. 引用符で囲んだデータ・ソース名を入力します。 2. 引用符で囲んだデータ・ソース表名を入力します。 3. 引用符で囲んだデータ・ソース・ログオン名およびパスワードを入力します。 • バイナリ・ファイルをエクスポートする場合は、引用符で囲んだファイル名およびパスを入力します。
処理中	このステップで設定を指定します。

テンプレート・フローの表示

システム・テンプレートで生成したスクリプトを確認する際、テンプレート・フローを確認すると有用な場合があります。

テンプレート・フローには、次のものが表示されます。

- テンプレートで生成されるスクリプトのフロー
- テンプレートのステップでの選択によって設計時プロンプトが置き換えられるテキスト

テンプレートのフローを表示するには:

1. **システム・ビュー**でシステム・テンプレートを含むビジネス・ルールまたはカスタム・テンプレートを右クリックし、「開く」を選択します。
2. テンプレートにすべての情報をまだ入力していない場合は入力します。
3. ルール・デザイナのフロー・チャートで、システム・テンプレートを右クリックし、「**テンプレート・フローの表示**」を選択します。

テンプレート・フローがテンプレート・デザイナに読取り専用で表示されます。フロー・チャート内の各コンポーネントをクリックして、そのコンポーネントに関連付けられているスクリプト、およびそのステップで関連する DTP に入力された値を表示できます。

フロー内のコンポーネントがグレーで表示されている場合、そのコンポーネントがテンプレートでのデータの入力時に選択した内容に基づいたスクリプト生成の一部ではないことを示しています。

テンプレート・フローを確認することは、条件コンポーネント、および条件が **true** の場合と **false** の場合の両方のスクリプトがある場合特に有用です。テンプレートでスクリプトの生成に使用されるパスは太字で表示され、その他のパスはグレーで表示されます。

システム・テンプレートのカスタム・テンプレートとしての保存

システム・テンプレートのコンテンツをカスタマイズできます。

システム・テンプレートは編集できませんが、システム・テンプレートで「名前を付けて保存」を使用して、編集可能なカスタム・テンプレートを作成できます。元のシステム・テンプレートはそのまま変更されません。

ノート:

Oracle Essbase システム・テンプレートで「名前を付けて保存」を実行し、カスタム・テンプレートとして保存すると、新規カスタム・テンプレートにアプリケーションという設計時プロンプトが表示されます。ただし、Essbase の新規カスタム・テンプレートを作成する場合は、テンプレートにアプリケーション設計時プロンプトは表示されません。カスタム・テンプレートとして保存するシステム・テンプレートではアプリケーション設計時プロンプトを無視してください。

システム・テンプレートをカスタム・テンプレートとして保存するには:

1. **システム・ビュー**でシステム・テンプレートを含むビジネス・ルールを右クリックし、「開く」を選択します。
2. 「**新規オブジェクト**」または「**既存のオブジェクト**」、またはルール・デザイナのフロー・チャートまたはスクリプトでシステム・テンプレートを右クリックし、「開く」を選択します。
3. システム・テンプレートが**テンプレート・デザイナ**で開く際、読取り専用で開きます。「OK」をクリックし、「アクション」、「名前を付けて保存」の順に選択します。
4. 「名前を付けて保存」ダイアログ・ボックスでテンプレートの新しい名前を入力し、アプリケーション・タイプ、アプリケーションおよびキューブを選択して「OK」をクリックします。

選択したアプリケーションとキューブの「**テンプレート**」ノードに新しいテンプレートが表示されます。これを開いてカスタマイズできます。

ノート:

新規テンプレートを確認するために、**システム・ビュー**でアプリケーション・リストのリフレッシュが必要な場合があります。「**テンプレート**」ノードを右クリックし、「リフレッシュ」を選択します。

ビジネス・ルールからのシステム・テンプレートの削除

ビジネス・ルールからシステム・テンプレートを削除できます。

システム・テンプレートを削除するには:

1. **システム・ビュー**で、システム・テンプレートを削除するビジネス・ルールを右クリックし、「**開く**」を選択します。
2. グラフィカル・ルールの場合、ルール・デザイナのフロー・チャートでシステム・テンプレートを右クリックし、「**削除**」を選択します。
3. スクリプト・ルールの場合、システム・テンプレートを含む行全体をハイライト表示し、右クリックして「**削除**」を選択します。
4.  をクリックします。

5

カスタム・テンプレートの操作

次も参照:

- [カスタム・テンプレートについて](#)
カスタム・テンプレートはビジネス・ルールおよび他のテンプレートで使用するために管理者が設計します。
- [カスタム・テンプレートの作成](#)
グラフィカル・カスタム・テンプレートとスクリプト・カスタム・テンプレートを作成できます。
- [カスタム・テンプレート用の設計時プロンプトの作成](#)
テンプレートを使用してビジネス・ルールを設計するときに適切な情報の入力を求めるプロンプトが表示されるように、グラフィカル・カスタム・テンプレート用の設計時プロンプトを入力できます。
- [カスタム・テンプレートを開く](#)
システム・ビュー、デプロイメント・ビューおよびカスタム・ビューからカスタム・テンプレートを開くことができます。
- [カスタム・テンプレートのリフレッシュ](#)
カスタム・テンプレートを作成した後、「テンプレート」ノードに新規テンプレートを表示するためにシステム・ビューでアプリケーション・リストをリフレッシュする必要があります。場合があります。
- [カスタム・テンプレートの使用状況の表示](#)
カスタム・テンプレートを使用しているビジネス・ルールのリストを表示できます。
- [カスタム・テンプレートのコピーおよび貼付け](#)
カスタム・テンプレートをコピーして、別のビジネス・ルールに貼り付けられます。
- [カスタム・テンプレートの削除](#)
システム・ビュー、カスタム・ビューおよびデプロイメント・ビューからカスタム・テンプレートを削除できます。
- [グラフィカル・カスタム・テンプレート内のテキストの検索および置換](#)
カスタム・テンプレートのテキスト文字列を検索できます。テキスト文字列のすべてのインスタンスを置換することもできます。

カスタム・テンプレートについて

カスタム・テンプレートはビジネス・ルールおよび他のテンプレートで使用するために管理者が設計します。

ノート:

カスタム・テンプレートは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

カスタム・テンプレートには、次のものからアクセスできます。

- 任意のビュー内のプラン・タイプの「**テンプレート**」ノード
- ルール・デザイナーの「**既存のオブジェクト**」

カスタム・テンプレートには次の2種類があります。

- グラフィカル・カスタム・テンプレートには、式、スクリプト、条件、メンバー・ブロック、メンバー範囲、固定ループ、メタデータ・ループおよび DTP 割当てコンポーネントを含めることができます。
- スクリプト・カスタム・テンプレートにはスクリプトを含めることはできますが、コンポーネントを含めることはできません。

グラフィカル・カスタム・テンプレートとスクリプト・カスタム・テンプレートの両方とも、既存のルール、式およびスクリプト(共有コンポーネント)、他のカスタム・テンプレートおよびシステム・テンプレートを含めることもできます。

グラフィカル・テンプレートもスクリプト・テンプレートもルールまたは他のカスタム・テンプレートに含めることができます。

グラフィカル・テンプレートとスクリプト・テンプレートの違いは次のとおりです。

- グラフィカル・コンポーネントには、DTP 割当てコンポーネント、メタデータ・ループ・コンポーネントおよびコンポーネント内で DTP 条件を使用する機能があります。
- スクリプト・テンプレートは、Planning でデプロイおよび起動できます。

カスタム・テンプレートの作成

グラフィカル・カスタム・テンプレートとスクリプト・カスタム・テンプレートを作成できます。

次も参照:

- [グラフィカル・カスタム・テンプレートの作成](#)
- [スクリプト・カスタム・テンプレートの作成](#)

グラフィカル・カスタム・テンプレートの作成

グラフィカル・カスタム・テンプレートを作成するには:

1. システム・ビューで「**テンプレート**」を右クリックし、「**新規**」を選択します。
2. 「**新規カスタム定義テンプレート**」で、要求された情報を入力し、「**OK**」をクリックします。

「**新規カスタム定義テンプレート**」ダイアログ・ボックスには、システム・ビューで作業中の Oracle Hyperion Planning アプリケーション・タイプ、アプリケーションおよびプラン・タイプが自動的に移入されます。

3. **テンプレート・デザイナー**で、新規および既存のオブジェクトをテンプレートのフロー・チャートに追加します。

オブジェクトを追加するには、「**新規オブジェクト**」または「**既存のオブジェクト**」からオブジェクトをドラッグし、フロー・チャートの「**開始**」と「**終了**」の間にドロップします。

表 5-1 グラフィカル・カスタム・テンプレートに挿入する新規オブジェクト

オブジェクト	関連項目
式	式コンポーネント
スクリプト	スクリプト・コンポーネント
条件	条件コンポーネント
メンバー・ブロック	メンバー・ブロック・コンポーネント
メンバー範囲	メンバー範囲コンポーネント
固定ループ	固定ループ・コンポーネント
メタデータ・ループ	メタデータ・ループ・コンポーネント
DTP 割当て	DTP 割当てコンポーネント
データ範囲	データ範囲コンポーネント (Financial Management ユーザーのみ).
Oracle Hyperion Financial Management ユーザ ーのみ	

表 5-2 グラフィカル・カスタム・テンプレートに挿入する既存のオブジェクト

オブジェクト	関連項目
ルール	ビジネス・ルールの設計
スクリプト	スクリプトおよび式コンポーネントの共有
式	スクリプトおよび式コンポーネントの共有
テンプレート	カスタム・テンプレートの操作

4. 「プロパティ」で、テンプレートのプロパティを入力します。

テンプレートにコンポーネントを追加したり、フロー・チャートでコンポーネント間を移動したりすると、プロパティが変わります。特定のコンポーネントのプロパティを入力するには、フロー・チャートでそのコンポーネントを選択します。

5. 「設計時プロンプト」タブを使用して、次のタイプの設計時プロンプトを入力します。

- テンプレートの使用時に情報を入力するよう求める設計時プロンプト。
- DTP 割当てコンポーネント、有効化条件および検証条件で使用できるプロンプト不可
可能な設計時プロンプト。

カスタム・テンプレート用の設計時プロンプトの作成を参照してください。

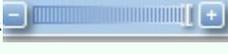
カスタム・テンプレートを開くと、必要に応じてテンプレートに追加できる次のシステム
設計時プロンプトがデフォルトで含まれます。

- **アプリケーション・タイプ** - アプリケーションがクラシック(汎用)か、Oracle Project
Financial Planning (PFP)アプリケーションかの判別に使用します。
- **アプリケーション** - アプリケーションが単一通貨か、複数通貨か、簡易通貨かの判別
に使用します。
- **UpperPOV** - グローバル範囲にメンバーがあるか、またはテンプレートがメンバー範
囲に挿入されたかの判別に使用します。UpperPOV 設計時プロンプトを使用するグ
ラフィカル・テンプレートの作成例を参照してください。

Oracle Essbase ユーザーのみ: カスタム・テンプレートを開くと、デフォルトでパスワ
ード設計時プロンプトが含まれています。Essbase の機能には、タイプ・パスワードのパ
ラメータを要求するものがあります。

6.  をクリックして、テンプレートを保存します。

 **ヒント:**

テンプレートのコンポーネントを編集する際、コンポーネント・アイコンのサイズとフロー・チャートに表示される詳細情報の量を増減できます。これを行うには、フロー・チャート内で  を使用してズーム・インおよびズーム・アウトします。

UpperPOV 設計時プロンプトを使用するグラフィカル・テンプレートの作成例

次のステップでは、UpperPOV 設計時プロンプトを使用するグラフィカル・テンプレートの作成方法を示します。この例のステップでは、サンプル Oracle Hyperion Planning アプリケーションを使用します。

1. システム・ビューで「**テンプレート**」を右クリックし、「**新規**」を選択します。
2. 「**新規カスタム定義テンプレート**」で、要求された情報を入力し、「**OK**」をクリックします。
3. 「**設計時プロンプト**」タブで  をクリックし、「**末尾に行を挿入**」を選択します。
4. 「**プロンプトの作成**」ダイアログ・ボックスで、次を行います。
 - a. 「**プロパティ**」タブで、次の情報を入力します:
 - **名前** - "MR1"を入力します。
 - **タイプ** - 「メンバー範囲」を選択します
 - 「**プロンプトしますか?**」を選択します
 - **DTP テキスト** - メンバー範囲を入力します
 - b. 「**依存関係**」タブで UpperPOV のドロップダウンをクリックし、「**排他**」を選択します。
 - c. 「**OK**」をクリックします。
5. **テンプレート・デザイナー**のフロー・チャートに**メンバー範囲**コンポーネントをドラッグします。
6. 「**メンバー範囲**」タブで  をクリックします。
7. 「**DTP セレクタ**」ダイアログ・ボックスで、ステップ 4 で作成した"メンバー範囲"設計時プロンプトを選択し、「**OK**」をクリックして「**変数に動的にリンク**」が選択されていることを確認します。
8. テンプレートにステップを作成し、"MR1"という名前の新規 DTP をステップに追加します。
9. テンプレートを保存します。

10. ルールにメンバー範囲をドラッグし、"エンティティ"および"製品"のメンバーを入力します。
11. 新規グラフィカル・テンプレートをルールのメンバー範囲へドラッグします。

メンバー範囲タイプの DTP のステップが表示されていること、および"エンティティ"および"製品"ディメンションが表示されていないことに注意してください。これは、UpperPOV を除外してメンバー範囲タイプの DTP を作成し、UpperPOV(ルールへドラッグしたメンバー範囲)で"エンティティ"および"製品"ディメンションが使用されているためです。

メンバー範囲タイプの DTP の依存性を包含に変更した場合、ルールには"エンティティ"ディメンションのみが表示され、テンプレートのステップには"製品"ディメンションのみが表示されます。

UpperPOV システム DTP をグラフィカル・テンプレートの DTP 条件で使用し、UpperPOV にメンバーがあるか(空ではない)、メンバーがないか(空である)、およびテンプレートがドラッグされたメンバー範囲コンポーネントがあるか(使用可能)ないか(使用不可)を判別します

スクリプト・カスタム・テンプレートの作成

スクリプトを使用するカスタム・テンプレートを作成すると、次のことが可能です。

- スクリプト・テンプレートを Oracle Hyperion Planning にデプロイ
- Planning でのテンプレートの起動。この場合、ステップが表示され、実行時にデータを入力できます
- 実行時間数および設計時プロンプトの使用

スクリプトを使用したカスタム・テンプレートを作成するには:

1. システム・ビューで「**テンプレート**」を右クリックし、「**新規**」を選択します。
2. 「**新規カスタム定義テンプレート**」ダイアログ・ボックスで、要求された情報を入力し、「**OK**」をクリックします。

ダイアログ・ボックスには、システム・ビューで作業中の Planning アプリケーション・タイプ、アプリケーションおよびキューブが自動的に移入されます。
3. テンプレートが開いたら、「**デザイナー**」の隣のドロップダウンをクリックし、「**スクリプトの編集**」を選択します。

これによってテンプレートがグラフィカルからスクリプトに変換されます。
4. 「**スクリプト・タイプ**」ドロップダウンで、「**計算スクリプト**」を選択します。

「**スクリプト・タイプ**」ドロップダウンが表示されない場合、テンプレート・デザイナーの右にある>>をクリックします。
5. 「**プロパティ**」で、テンプレートのプロパティを入力します。
6. 「**テンプレート・デザイナー**」で、スクリプトを入力します。
7. 「**設計時プロンプト**」タブを使用して、Planning でテンプレートを起動すると情報を入力するよう求める設計時プロンプトを入力します。[カスタム・テンプレート用の設計時プロンプトの作成](#)を参照してください
8.  をクリックして、テンプレートを保存します。
9. テンプレートを Planning にデプロイします。
10. Planning を開き、テンプレートを起動します。

カスタム・テンプレート用の設計時プロンプトの作成

テンプレートを使用してビジネス・ルールを設計するときに適切な情報の入力を求めるプロンプトが表示されるように、グラフィカル・カスタム・テンプレート用の設計時プロンプトを入力できます。

ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

次のタイプの設計時プロンプトを作成できます:

- 属性 DTP
- ブール DTP
- ディメンション間 DTP
- 条件 DTP
- ディメンション DTP
- ディメンション(複数) DTP
- メンバー DTP
- メンバー(複数)DTP
- メンバー範囲 DTP
- 数値 DTP
- パスワード DTP
- 制限リスト DTP
- 区切り文字 DTP
- 文字列 DTP

属性 DTP

カスタム・テンプレートが属するアプリケーションの属性を定義します。

たとえば、製品のサイズを入力する設計時プロンプトを作成するとします。この場合、"サイズ"ディメンションを選択できます。また、必要に応じてデフォルト値("大"など)も入力できます。「プロンプトしますか?」を選択した場合、テンプレートが使用されると、属性メンバー(この例では、ディメンション・サイズ)を入力するプロンプトがユーザーに表示されます。

属性設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**属性**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが **Oracle Hyperion Planning** で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - 「**DTP テキスト**」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「**プロンプトしますか?**」を選択した場合、「**DTP テキスト**」は必須です。
 - **オプション**。「**デフォルト値**」で、DTP のデフォルト値を割り当てます。
デフォルト値を割り当てるには、 をクリックして、「**メンバー**」を選択し、次に、「**メンバー・セレクタ**」でメンバーを選択します。

ブール DTP

型が"True"または"False"の DTP。

ブール設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**ブール**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが **Oracle Hyperion Planning** で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。

- **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
- 「**DTP テキスト**」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「**プロンプトしますか?**」を選択した場合、「**DTP テキスト**」は必須です。
- **オプション**。「**デフォルト値**」で「**TRUE**」または「**FALSE**」を選択します。

ディメンション間 DTP

テンプレートが属するアプリケーションからディメンション間構文(->)のメンバーの範囲を定義します。

ディメンション間 DTP では、各ディメンションから 1 つのメンバーを選択できます。たとえば、ディメンション間の設計時プロンプトを式のソースで使用できます。

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**ディメンション間**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが **Oracle Hyperion Planning** で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - 「**DTP テキスト**」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「**プロンプトしますか?**」を選択した場合、「**DTP テキスト**」は必須です。
 - **オプション**。「**デフォルト値**」で、DTP のデフォルト値を割り当てます。
デフォルト値を割り当てるには、次のいずれかを行います:
 -  をクリックし、各ディメンションの隣の  をクリックして「**DTP**」、「**メンバー**」または「**関数**」を選択します。
 -  をクリックして、DTP をデフォルト値として選択します。
5. **オプション**: 「**依存関係**」ボタン()をクリックしてオプションを選択します:

- **排他** - 設計時プロンプト(例: DTP2)を別の設計時プロンプト(例: DTP1)を含まないものとしてします。DTP2 のプロンプトが表示されるとき、DTP1 で使用されていないディメンションのみが表示されます。
 - **包含** - 設計時プロンプト(例: DTP4)を別の設計時プロンプト(例: DTP3)を含むものとしてします。DTP4 のプロンプトが表示されるとき、DTP3 で使用されているディメンションのみが表示されます。
 - **なし** - 設計時プロンプトに対して依存関係は定義されません。
6. **オプション: 「ディメンション限界の選択」** ボタン()をクリックして、次を実行します:
- 「**密度**」でオプションを選択します。
 - **疎の表示** - DTP に疎ディメンションのみを表示します
 - **密の表示** - DTP に密ディメンションのみを表示します
 - **両方** - DTP に密と疎の両方のディメンションを表示します
 - 「**ディメンション・タイプ**」でディメンション・タイプを選択し、選択したディメンション・タイプのみを表示するように DTP を制限します。

たとえば、*勘定科目*、*エンティティ*および*年のメンバー*のみをユーザーに入力させたい場合、「**ディメンション・タイプ**」でこれらのディメンションを選択します。入力のためのプロンプトが表示されるときに表示されるディメンションは、タイプが*勘定科目*、*エンティティ*および*年のもの*のみです。

条件 DTP

条件を定義します。

条件設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**条件**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが Oracle Hyperion Planning で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。

- 「DTP テキスト」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「プロンプトしますか?」を選択した場合、「DTP テキスト」は必須です。

ディメンション DTP

テンプレートが属するアプリケーションのディメンションを定義します。

ディメンション設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「設計時プロンプト」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「タイプ」に「ディメンション」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択して、DTP をステップに追加します。この DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが Oracle Hyperion Planning で起動されたときに、ユーザーにプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必要になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - 「DTP テキスト」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「プロンプトしますか?」を選択した場合、「DTP テキスト」は必須です。
 - **オプション**。「デフォルト値」でドロップダウンをクリックし、テンプレートが属するアプリケーションのディメンションを選択します。
5. **オプション: 「依存関係」** ボタン()をクリックしてオプションを選択します。
 - **排他** - 設計時プロンプト(例: DTP2)を別の設計時プロンプト(例: DTP1)を含まないものとします。DTP2 のプロンプトが表示される時、DTP1 で使用されていないディメンションのみが表示されます。
 - **包含** - 設計時プロンプト(例: DTP4)を別の設計時プロンプト(例: DTP3)を含むものとします。DTP4 のプロンプトが表示される時、DTP3 で使用されているディメンションのみが表示されます。
 - **なし** - 設計時プロンプトに対して依存関係は定義されません。

ディメンション DTP には、メンバー範囲、ディメンション、ディメンション(複数)またはディメンション間 DTP への依存関係がある場合があります。

6. オプション: 「**ディメンション限界の選択**」 ボタン()をクリックして、次を実行します:

- 「**密度**」でオプションを選択します。
 - **疎の表示** - DTP に疎ディメンションのみを表示します
 - **密の表示** - DTP に密ディメンションのみを表示します
 - **両方** - DTP に密と疎の両方のディメンションを表示します
- 「**ディメンション・タイプ**」でディメンション・タイプを選択し、選択したディメンション・タイプのみを表示するように DTP を制限します。

たとえば、*勘定科目*、*エンティティ*および*年のメンバー*のみをユーザーに入力させたい場合、「**ディメンション・タイプ**」でこれらのディメンションを選択します。入力のためのプロンプトが表示されるときに表示されるディメンションは、タイプが*勘定科目*、*エンティティ*および*年のもの*のみです。

ディメンション(複数) DTP

テンプレートが属するアプリケーションのディメンション(複数)を定義します。

ディメンション(複数)設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**ディメンション**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが **Oracle Hyperion Planning** で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必要になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - 「**DTP テキスト**」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「**プロンプトしますか?**」を選択した場合、「**DTP テキスト**」は必須です。
 - **オプション**。「**デフォルト値**」で、DTP のデフォルト値を割り当てます。

デフォルト値を割り当てるには、 をクリックし、「**DTP**」または「**ディメンション**」を選択し、DTP またはディメンションを選択します。

5. オプション: 「**依存関係**」 ボタン()をクリックしてオプションを選択します:

- **排他** - 設計時プロンプト(例: DTP2)を別の設計時プロンプト(例: DTP1)を含まないものとします。DTP2のプロンプトが表示される時、DTP1で使用されていないディメンションのみが表示されます。
- **包含** - 設計時プロンプト(例: DTP4)を別の設計時プロンプト(例: DTP3)を含むものとします。DTP4のプロンプトが表示される時、DTP3で使用されているディメンションのみが表示されます。
- **なし** - 設計時プロンプトに対して依存関係は定義されません。

ディメンション(複数)DTPには、メンバー範囲、ディメンション、ディメンション(複数)またはディメンション間 DTP への依存関係がある場合があります。

6. **オプション: 「ディメンション限界の選択」** ボタン()をクリックして、次を実行します:

- 「**密度**」でオプションを選択します。
 - **疎の表示** - DTP に疎ディメンションのみを表示します
 - **密の表示** - DTP に密ディメンションのみを表示します
 - **両方** - DTP に密と疎の両方のディメンションを表示します
- 「**ディメンション・タイプ**」でディメンション・タイプを選択し、選択したディメンション・タイプのみを表示するように DPT を制限します。

たとえば、*勘定科目*、*エンティティ*および*年*のメンバーのみをユーザーに入力させたい場合、「**ディメンション・タイプ**」でこれらのディメンションを選択します。入力のためのプロンプトが表示されるときに表示されるディメンションは、タイプが*勘定科目*、*エンティティ*および*年*のもののみです。

メンバー DTP

アプリケーションのディメンションのメンバーを定義します。

メンバー設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**メンバー**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが **Oracle Hyperion Planning** で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。

- **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - **「DTP テキスト」** で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「プロンプトしますか?」 を選択した場合、**「DTP テキスト」** は必須です。
 - **オプション。「デフォルト値」** で、DTP のデフォルト値を割り当てます。
デフォルト値を割り当てるには、 をクリックして、**「メンバー」** を選択し、次に、**「メンバー・セレクタ」** でメンバーを選択します。**「ディメンション」** ドロップダウンで選択されている属性ディメンションが表示されます。このディメンションを展開すると、使用可能なメンバーが表示されます。
 - **スクリプト・テンプレートのみです。「制限」** で、DTP、メンバー、またはユーザーが選択できるメンバーのセットを返す関数を入力します。
たとえば、プロンプトが表示されたときに親"Revenues"の下のレベル 0 の勘定科目のみを表示するように DTP を制限するには、**「ディメンション」** に"Account"を選択し、**「制限」** で"ILvL0Descendants(Revenues)"を入力します。
ノート: 「制限」に関数を使用する場合、Essbase 関数のかわりに Oracle Essbase 関数をお勧めします。場合によっては、Essbase 関数が必要なメンバーを戻さないことがあります。たとえば、Essbase 関数の評価に動的メンバーが含まれる場合です。メンバー設計時プロンプトの「制限」については、Essbase 関数"@Relative("Mbr Name", 0)"のかわりに"ILvL0Descendants("Mbr Name")"を使用します。
5. **オプション: 「依存関係」** ボタン()をクリックしてオプションを選択します:
- **排他** - 設計時プロンプト(例: DTP2)を別の設計時プロンプト(例: DTP1)を含まないものとし、DTP2 のプロンプトが表示される時、DTP1 で使用されていないディメンションのみが表示されます。
 - **包含** - 設計時プロンプト(例: DTP4)を別の設計時プロンプト(例: DTP3)を含むものとし、DTP4 のプロンプトが表示される時、DTP3 で使用されているディメンションのみが表示されます。
 - **なし** - 設計時プロンプトに対して依存関係は定義されません。
6. **オプション: 「ディメンション限界の選択」** ボタン()をクリックして、次を実行します:
- **「密度」** でオプションを選択します。
 - **疎の表示** - DTP に疎ディメンションのみを表示します
 - **密の表示** - DTP に密ディメンションのみを表示します
 - **両方** - DTP に密と疎の両方のディメンションを表示します
 - **「ディメンション・タイプ」** でディメンション・タイプを選択し、選択したディメンション・タイプのみを表示するように DTP を制限します。
たとえば、*勘定科目*、*エンティティ*および*年*のメンバーのみをユーザーに入力させたい場合、**「ディメンション・タイプ」** でこれらのディメンションを選択します。入力のためのプロンプトが表示される時に表示されるディメンションは、タイプが*勘定科目*、*エンティティ*および*年*のもののみです。

メンバー(複数)DTP

アプリケーションで選択したディメンションの複数のメンバーを定義します。

メンバー(複数)設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「設計時プロンプト」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:

- **名前**を入力し、「**タイプ**」に「**メンバー(複数)**」を選択します。
- 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが **Oracle Hyperion Planning** で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必要になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
- 「**DTP テキスト**」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「**プロンプトしますか?**」を選択した場合、「**DTP テキスト**」は必須です。
- **オプション**。「**デフォルト値**」で、DTP のデフォルト値を割り当てます。

デフォルト値を割り当てるには、 をクリックして、「**メンバー**」を選択し、次に、「**メンバー・セレクタ**」でメンバーを選択します。「**ディメンション**」ドロップダウンで選択されている属性ディメンションが表示されます。このディメンションを展開すると、使用可能なメンバーが表示されます。

- **スクリプト・テンプレートのみです**。「**制限**」で、DTP、メンバー、またはユーザーが選択できるメンバーのセットを返す関数を入力します。

たとえば、プロンプトが表示されたときに親"Revenues"の下のレベル 0 の勘定科目のみを表示するように DTP を制限するには、「**ディメンション**」に "Account"を選択し、「**制限**」で"ILvl0Descendants(Revenues)"を入力します。

ノート: 「制限」に関数を使用する場合、Essbase 関数のかわりに Oracle Essbase 関数をお勧めします。場合によっては、Essbase 関数が必要なメンバーを戻さないことがあります。たとえば、Essbase 関数の評価に動的メンバーが含まれる場合です。メンバー(複数)設計時プロンプトの「制限」については、Essbase 関数"@Relative("Mbr Name", 0)"のかわりに"ILvl0Descendants("Mbr Name")"を使用します。

5. **オプション: 「依存関係」** ボタン() をクリックしてオプションを選択します:

- **排他** - 設計時プロンプト(例: DTP2)を別の設計時プロンプト(例: DTP1)を含まないものとします。DTP2 のプロンプトが表示される時、DTP1 で使用されていないディメンションのみが表示されます。
 - **包含** - 設計時プロンプト(例: DTP4)を別の設計時プロンプト(例: DTP3)を含むものとします。DTP4 のプロンプトが表示される時、DTP3 で使用されているディメンションのみが表示されます。
 - **なし** - 設計時プロンプトに対して依存関係は定義されません。
6. **オプション: 「ディメンション限界の選択」** ボタン()をクリックして、次を実行します:

- 「**密度**」でオプションを選択します。
 - **疎の表示** - DTP に疎ディメンションのみを表示します
 - **密の表示** - DTP に密ディメンションのみを表示します
 - **両方** - DTP に密と疎の両方のディメンションを表示します
- 「**ディメンション・タイプ**」でディメンション・タイプを選択し、選択したディメンション・タイプのみを表示するように DTP を制限します。

たとえば、*勘定科目*、*エンティティ*および*年*のメンバーのみをユーザーに入力させたい場合、「**ディメンション・タイプ**」でこれらのディメンションを選択します。入力のためのプロンプトが表示される時に表示されるディメンションは、タイプが*勘定科目*、*エンティティ*および*年*のもののみです。

メンバー範囲 DTP

アプリケーションで選択したディメンションのメンバーの範囲を定義します。

メンバー範囲 DTP には、各ディメンションからのメンバーを 1 つまたは複数含めることができます。たとえば、メンバー範囲タイプの DTP を使用すると、ルールへのメンバーの入力をユーザーに求める視点ステップを作成できます。

メンバー範囲設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**メンバー範囲**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが Oracle Hyperion Planning で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。

- **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - **「DTP テキスト」** で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「プロンプトしますか?」 を選択した場合、**「DTP テキスト」** は必須です。
 - **オプション。「デフォルト値」** で、DTP のデフォルト値を割り当てます。
デフォルト値を割り当てるには、次のいずれかを行います:
 -  をクリックし、各ディメンションの隣の  をクリックして **「DTP」**、**「メンバー」** または **「関数」** を選択します。
 -  をクリックして、DTP をデフォルト値として使用します。
5. **オプション: 「依存関係」** ボタン() をクリックしてオプションを選択します:
- **排他** - 設計時プロンプト(例: DTP2)を別の設計時プロンプト(例: DTP1)を含まないものとします。DTP2 のプロンプトが表示される時、DTP1 で使用されていないディメンションのみが表示されます。
 - **包含** - 設計時プロンプト(例: DTP4)を別の設計時プロンプト(例: DTP3)を含むものとします。DTP4 のプロンプトが表示される時、DTP3 で使用されているディメンションのみが表示されます。
 - **なし** - 設計時プロンプトに対して依存関係は定義されません。
- メンバー範囲 DTP は、次の DTP タイプに依存関係がある場合があります: メンバー範囲、ディメンション間、ディメンションおよびディメンション(複数)。
6. **オプション: 「ディメンション限界の選択」** ボタン() をクリックして、次を実行します:
- **「密度」** でオプションを選択します。
 - **疎の表示** - DTP に疎ディメンションのみを表示します
 - **密の表示** - DTP に密ディメンションのみを表示します
 - **両方** - DTP に密と疎の両方のディメンションを表示します
 - **「ディメンション・タイプ」** でディメンション・タイプを選択し、選択したディメンション・タイプのみを表示するように DPT を制限します。
たとえば、*勘定科目*、*エンティティ*および*年*のメンバーのみをユーザーに入力させたい場合、**「ディメンション・タイプ」** でこれらのディメンションを選択します。入力のためのプロンプトが表示されるときに表示されるディメンションは、タイプが*勘定科目*、*エンティティ*および*年*のもののみです。

数値 DTP

数を定義します。たとえば、式で数値 DTP を使用できます。

数値設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。

2. 「設計時プロンプト」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**数値**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが Oracle Hyperion Planning で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - 「**DTP テキスト**」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「**プロンプトしますか?**」を選択した場合、「**DTP テキスト**」は必須です。
 - **オプション**。「**デフォルト値**」で  をクリックして「**DTP**」を選択し、DTP 値を選択します。

パスワード DTP

設計時プロンプトで使用するパスワードを定義します。

パスワード設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「設計時プロンプト」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**パスワード**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが Oracle Hyperion Planning で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。

- 「DTP テキスト」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「プロンプトしますか?」を選択した場合、「DTP テキスト」は必須です。
- オプション。「デフォルト値」でパスワードの値を入力します。

制限リスト DTP

制限リストを定義します。

たとえば、割当てシステム・テンプレートで、丸め処理方法を選択し、制限リスト設計時プロンプトを使用して、テンプレートで使用する丸め処理のタイプを選択できます。

テンプレートのスクリプトで、丸め処理の各タイプにスクリプトを作成でき、そのスクリプトは、ユーザーがその丸め処理オプションを選択した場合のみ使用されます。

制限リスト設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「設計時プロンプト」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「タイプ」に「制限リスト」を選択します。
 - 「制限リスト」ボタン()をクリックし、**ルール・ビルダー値**および**代替値**を入力します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが Oracle Hyperion Planning で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必須になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - 「DTP テキスト」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「プロンプトしますか?」を選択した場合、「DTP テキスト」は必須です。
5. **オプション: 「デフォルト値」**で、設計時プロンプトに表示されるデフォルト値を入力します。

デフォルト値を定義するには、「デフォルト値」ドロップダウンをクリックし、「制限リスト」タブで入力した**ルール・ビルダー値**を選択します。

区切り文字 DTP

区切り文字は、テンプレート・ステップ内の指示またはスペーサです。

区切り文字設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**区切り文字**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・テンプレートが **Oracle Hyperion Planning** で起動されたときに、ユーザーに输入のためのプロンプトを表示するテンプレート・ステップに表示されます。
 - **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの输入が必須になります。
 - **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
 - 「**DTP テキスト**」で、inputのためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「**プロンプトしますか?**」を選択した場合、「**DTP テキスト**」は必須です。
 - **オプション**。「**デフォルト値**」で、DTP のデフォルト値を入力します。

文字列 DTP

テキスト文字列を定義します。

文字列設計時プロンプトを作成するには:

1. **テンプレート・デザイナー**で、既存のテンプレートを開くか、新規テンプレートを作成します。
2. 「**設計時プロンプト**」タブで  をクリックします。
3. テンプレート用に定義された設計時プロンプトがすでにある場合、新規設計時プロンプトを入力する場所を選択します。
4. 次の情報を入力します:
 - **名前**を入力し、「**タイプ**」に「**文字列**」を選択します。
 - 使用するオプションを選択します:
 - **プロンプトしますか?** - このオプションを選択すると、DTP をステップで使用できます。DTP は、テンプレートがルールで使用されたとき、またはスクリプト・

テンプレートが Oracle Hyperion Planning で起動されたときに、ユーザーに入力のためのプロンプトを表示するテンプレート・ステップに表示されます。

- **必須?** - このオプションを選択すると、設計時プロンプトに関するデータの入力が必要になります。
- **読取り専用** - このオプションを選択すると、ユーザーはプロンプトの読取りのみを行えます。
- 「DTP テキスト」で、入力のためのプロンプトが表示されたときにユーザーに示されるテキストを入力します。
「プロンプトしますか?」を選択した場合、「DTP テキスト」は必須です。
- **オプション**。「デフォルト値」で  をクリックして「DTP」、「メンバー」または「関数」を選択し、デフォルト値として DTP、メンバーまたは関数を入力します。

設計時プロンプトのステップの作成

テンプレート・ウィザード・デザイナーを使用して、テンプレートのステップを作成します。ウィザードを使用すると、各ステップで表示する設計時プロンプトを決定したり、ステップを表示するかどうか(有効化条件)、およびエラーまたは警告メッセージを表示するかどうか(検証条件)を決定したりできます。

設計時プロンプトのステップを作成するには:

1. 「設計時プロンプト」タブで  をクリックします。
2. テンプレート・ウィザード・デザイナーで  をクリックしてウィザードのステップを作成します。
3. 「ステップの追加」ダイアログ・ボックスで、ステップに関する情報を入力し、「OK」をクリックします。
追加したステップは、テンプレート・ウィザード・デザイナーの「ステップ」ドロップダウン・リストに表示されます。
4. ステップに表示する設計時プロンプトを「使用可能な DTP」から「選択した DTP」に移動します。
5. 「条件の有効化」タブで、ステップを表示するか非表示にするかに関する情報を入力します。
 - 「条件接頭辞」で、ドロップダウンから接頭辞を選択します。
 - 「DTP」で  をクリックし、DTP または関数を選択します。
 - 「演算子」で、ドロップダウンから演算子を選択します。
 - 「値」で、条件の値を入力するか、選択します。

前述のステップを繰り返し、条件のすべての文を定義します。行を追加するには、最後の行のプラスのアイコン(+)をクリックします。

条件の最初の行は IF 文です。その他のそれぞれの行は AND 文です。たとえば、

各行のデフォルトは AND 文ですが、「AND」をクリックすると「OR」に変更できます。

6. 「**検証条件**」タブで、ユーザーが適切でないデータを入力したとき(エラー)、または適切でない可能性のあるデータを入力したとき(警告)にエラーまたは警告メッセージを表示するかどうかに関する情報を入力します。
 - 「**検証条件**」で  をクリックし、検証条件を定義します。
 - 「**メッセージ・レベル**」で「**エラー**」または「**警告**」を選択します。
エラー・メッセージでは次のステップに進みません。警告メッセージでは、警告メッセージの「**OK**」をクリックすると、次のステップに進めます。
 - 「**検証メッセージ**」で、ステップでユーザーに表示されるメッセージを入力します。
プラス・アイコン(+)をクリックして、追加のエラーまたは警告をステップに追加します。
7. 「**OK**」、 の順にクリックします。

設計時プロンプトの依存関係の定義

ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

ディメンション間、ディメンション(複数)、ディメンション、メンバー、メンバー(複数)およびメンバー範囲の設計時プロンプト・タイプに包含的および排他的な依存関係を定義できます。別のプロンプト(たとえば DTP2 など)を含むものとしてプロンプト(たとえば DTP1 など)を指定した場合、ユーザーに DTP1 の入力を求めるプロンプトが表示されるときに、DTP2 からのディメンションのみが表示されます。別のプロンプト(たとえば DTP4 など)を含まないものとしてプロンプト(たとえば DTP3 など)を指定した場合、ユーザーに DTP3 の入力を求めるプロンプトが表示されるときに、DTP4 で使用されていないディメンションのみが表示されます。

ノート:

設計時プロンプトは、設計時プロンプト・グリッドでそれより前に出現する設計時プロンプトのみ包含または除外できるため、設計時プロンプトの順序は非常に重要です。

依存関係を定義するには:

1. **システム・ビュー**でカスタム・テンプレートを作成するか、開きます。
2. 「**設計時プロンプト**」タブで  をクリックし、次のいずれかのタイプを選択します。
 - **ディメンション間**
 - **ディメンション**
 - **ディメンション**

- メンバー
 - メンバー(複数)
 - メンバー範囲
 - UDA
3. 「依存関係」タブで設計時プロンプトを選択し、依存関係オプションを選択します。
 - 排他 - 設計時プロンプト(例: DTP2)を別の設計時プロンプト(例: DTP1)を含まないものとします。DTP2のプロンプトが表示される時、DTP1で使用されていないディメンションのみが表示されます。
 - 包含 - 設計時プロンプト(例: DTP4)を別の設計時プロンプト(例: DTP3)を含むものとします。DTP4のプロンプトが表示される時、DTP3で使用されているディメンションのみが表示されます。
 - なし - 設計時プロンプトに対して依存関係は定義されません。
 4. 「OK」、の順にクリックします。

ディメンション間、ディメンション、ディメンション(複数)、メンバーおよびメンバー範囲の設計時プロンプトに対するディメンション制限の定義

ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

カスタム・テンプレートに対してディメンション間、ディメンション、ディメンション(複数)、メンバー、またはメンバー範囲の設計時プロンプトを作成する場合は、プロンプトを密ディメンションおよび疎ディメンションに対して表示するか、密ディメンションのみまたは疎ディメンションのみに対して表示するかを指定する必要があります。

ディメンション間、ディメンション、ディメンション(複数)、メンバー、またはメンバー範囲の設計時プロンプトを表示する対象のディメンションのタイプを選択するには:

1. システム・ビューでカスタム定義テンプレートを作成するか、開きます。
2. 「設計時プロンプト」タブで  をクリックし、次のいずれかのタイプを選択します。
 - ディメンション間
 - ディメンション
 - ディメンション
 - メンバー
 - メンバー範囲

3. 「制限の定義」タブを選択します。
 - 「密度」でオプションを選択します。
 - 疎の表示 - DTP に疎ディメンションのみを表示します
 - 密の表示 - DTP に密ディメンションのみを表示します
 - 両方 - DTP に密と疎の両方のディメンションを表示します
 - 「ディメンション・タイプ」でディメンション・タイプを選択し、選択したディメンション・タイプのみを表示するように設計時プロンプトを制限します。

たとえば、勘定科目、エンティティおよび年のメンバーのみをユーザーに入力させたい場合、「ディメンション・タイプ」でこれらのディメンションを選択します。入力のためのプロンプトが表示されるときに表示されるディメンションは、タイプが勘定科目、エンティティおよび年のもののみです。
4. 「OK」、の順にクリックします。

設計時プロンプト内のテキストの検索および置換

ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

カスタム・テンプレートの設計時プロンプトで、テキスト文字列を検索して置換できます。

注意:

設計時プロンプトを置換する順序および場所は、非常に重要です。「設計時プロンプト」タブと「テンプレート・デザイナー」タブの両方で、設計時プロンプト名を検索および置換する場合、まず「設計時プロンプト」タブで検索し、テンプレートを保存してから、「テンプレート・デザイナー」タブで検索および置換する必要があります。この順序で検索を実行しない場合、設計時プロンプト名は「テンプレート・デザイナー」タブで置換されません。

設計時プロンプト内のテキストを検索して置換するには:

1. システム・ビューでカスタム・テンプレートを開きます。
2. 「設計時プロンプト」タブで  をクリックし、検索して置換する情報を入力して「置換」または「すべて置換」をクリックします。

カスタム・テンプレートを開く

システム・ビュー、デプロイメント・ビューおよびカスタム・ビューからカスタム・テンプレートを開くことができます。

ノート:

カスタム定義テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

ルール・デザイナのビジネス・ルールのフロー・チャートからカスタム・テンプレートを開くこともできます。

カスタム・テンプレートを開くには、テンプレートをダブルクリックするか、テンプレートを右クリックして「開く」を選択します。

カスタム・テンプレートのリフレッシュ

カスタム・テンプレートを作成した後、「テンプレート」ノードに新規テンプレートを表示するためにシステム・ビューでアプリケーション・リストをリフレッシュする必要があります。

カスタム定義テンプレートが属するアプリケーション・タイプ、アプリケーション、または計算タイプやプラン・タイプをリフレッシュすると、デフォルトで「テンプレート」ノードがリフレッシュされます。ただし、「テンプレート」ノードをリフレッシュしても、アプリケーション・リストのそれより上位のレベル(つまり、計算タイプやプラン・タイプ、アプリケーション、またはアプリケーション・タイプ)はリフレッシュされません。

カスタム・テンプレートのリストをリフレッシュするには、「テンプレート」ノードを右クリックして「リフレッシュ」を選択します。

ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

カスタム・テンプレートの使用状況の表示

カスタム・テンプレートを使用しているビジネス・ルールのリストを表示できます。

カスタム・テンプレートの使用状況を表示することは、カスタム・テンプレートを削除するときに、どのオブジェクトがそのテンプレートを使用しているか(使用しているオブジェクトがある場合)を確認する必要がある場合に役立ちます。テンプレートを削

除するには、そのテンプレートを使用しているすべてのオブジェクトからテンプレートを除去する必要があります。

カスタム・テンプレートの使用状況を表示するには、**システム・ビュー**で「**テンプレート**」ノードを展開し、テンプレートを右クリックして「**使用状況の表示**」を選択します。

ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

カスタム・テンプレートのコピーおよび貼付け

カスタム・テンプレートをコピーして、別のビジネス・ルールに貼り付けられます。

コピー先のルールは、同じアプリケーション・タイプおよび計算またはプラン・タイプに属する必要があります。たとえば、カスタム・テンプレートを Oracle Hyperion Planning アプリケーション内のプラン 1 プラン・タイプのルールから Planning アプリケーション内のプラン 1 プラン・タイプの別のルールにコピーすることはできますが、資本資産プラン・タイプのルールにコピーできません。

カスタム・テンプレートをコピーして貼り付けるには、次のいずれかを行います。

- 「**テンプレート**」ノードでテンプレートを右クリックして「**コピー先**」を選択し、「**コピー先**」ダイアログ・ボックスで要求された情報を入力して「**OK**」をクリックします。
- ビジネス・ルールのフロー・チャートで、コピーするテンプレートを右クリックし、「**コピー**」を選択します。テンプレートを貼り付けるビジネス・ルールを開き、ビジネス・ルールのフロー・チャートを右クリックして「**貼付け**」を選択します。

ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

カスタム・テンプレートの削除

システム・ビュー、カスタム・ビューおよびデプロイメント・ビューからカスタム・テンプレートを削除できます。

カスタム・テンプレートを削除する前に、どのビジネス・ルールまたはカスタム・フォルダでも使用されていないことを確認します。テンプレートの使用状況を表示するには、テンプレートを右クリックして「**使用状況の表示**」を選択します。

カスタム・テンプレートを削除するには、「**テンプレート**」ノードを展開し、テンプレートを右クリックして「**削除**」を選択します。

 ノート:

スクリプト・テンプレートが Oracle Hyperion Planning にデプロイされている場合、「削除」を選択すると、次のメッセージが表示されます。

選択したアイテムを削除しますか? 選択されたオブジェクトの一部は Planning にデプロイされました。これらを Planning サーバーから削除しますか。

ボックスを選択すると、スクリプト・テンプレートは Planning から Oracle Hyperion Calculation Manager から削除されます。

 ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

グラフィカル・カスタム・テンプレート内のテキストの検索および置換

カスタム・テンプレートのテキスト文字列を検索できます。テキスト文字列のすべてのインスタンスを置換することもできます。

 ノート:

カスタム・テンプレートと設計時プロンプトは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・アプリケーションでサポートされます。

文字列を検索する場合、Oracle Hyperion Calculation Manager は、テンプレートのフロー・チャートで選択されているコンポーネントの直後のコンポーネントから検索を開始してフロー・チャートの最後まで検索を続け、その後フロー・チャートの最初から検索を開始してフロー・チャートで選択されているコンポーネントに到達するまで検索します。一致するテキスト文字列が 1 つ見つかった後に、別の出現を検索する場合は、再度検索を開始して次の出現を探する必要があります。

カスタム・テンプレート内のテキストを検索したり、検索して置換するには:

1. システム・ビューでカスタム・テンプレートを開きます。
2. 次のいずれかを行います:
 - 「テンプレート・デザイナー」タブで「編集」、「検索」の順に選択します。「検索」ダイアログ・ボックスで検索するテキストを入力し、「検索」をクリックします。

- 「**テンプレート・デザイナー**」タブで「**編集**」、「**すべて置換**」の順に選択します。「**置換**」ダイアログ・ボックスで検索して置換する情報を入力し、「**すべて置換**」を選択します。

6

コンポーネントを使用したビジネス・ルールおよびテンプレートの設計

次も参照:

- **コンポーネントについて**
ビジネス・ルールおよびテンプレートには、いくつかのタイプのコンポーネントを含めることができます。
- **式コンポーネント**
式コンポーネントは、式の計算文で構成されています。
- **スクリプト・コンポーネント**
ビジネス・ルールおよびテンプレートでスクリプト・コンポーネントを使用できます。
- **条件コンポーネント**
条件コンポーネントは、TRUE または FALSE となる条件文で構成されます。
- **メンバー・ブロック・コンポーネント**
メンバー・ブロック・コンポーネントは、スクリプト内の 1 つ以上の文を囲むようにメンバーを定義します。
- **メンバー範囲コンポーネント**
メンバー範囲コンポーネントは、Planning ディメンションからのメンバーの範囲で構成される一種のループです。
- **データ範囲コンポーネント(Financial Management ユーザーのみ)**
- **セル・テキスト範囲コンポーネント(Financial Management ユーザーのみ)**
- **固定ループ・コンポーネント**
固定ループ・コンポーネントは、メタデータ・メンバーのリストを固定の回数循環するオブジェクトです。
- **メタデータ・ループ・コンポーネント**
メタデータ・ループ・コンポーネントでは、関数(メタデータ)または開始と終了の値(固定)を使用して複数のメンバーに値を割り当てることができます。
- **DTP 割当てコンポーネント**
DTP 割当てコンポーネントを使用して、DTP、メンバー、関数または入力テキストをカスタム・テンプレートの設計時プロンプトに割り当てることができます。
- **スクリプトおよび式コンポーネントの共有**
共有された式またはスクリプト・コンポーネントは、式レベルおよびスクリプト・レベルに存在します。
- **コンポーネントのコピー**
ビジネス・ルール・コンポーネントの子をコピーして貼り付けるか、ビジネス・ルールの式またはスクリプト・コンポーネントに参照をコピーして貼り付けます。
- **コンポーネントの保存**
式およびスクリプト・コンポーネントをコンポーネント・デザイナーで設計してから保存します。

- **式およびスクリプト・コンポーネントのリフレッシュ**
式またはスクリプト・コンポーネントを作成した後、「式」ノードまたは「スクリプト」ノードで確認するために、システム・ビューでアプリケーション・リストのリフレッシュが必要な場合があります。
- **式およびスクリプト・コンポーネントの使用の表示**
スクリプトまたは式コンポーネントの使用を表示できます。
- **フロー・チャート内でのコンポーネントの操作**
フロー・チャートでコンポーネントに対してアクションを実行できます。

コンポーネントについて

ビジネス・ルールおよびテンプレートには、いくつかのタイプのコンポーネントを含めることができます。

- **式コンポーネント** - メンバー、関数および条件文を使用して記述または設計可能な計算文
- **スクリプト・コンポーネント** - Visual Basic (Oracle Hyperion Financial Management の場合)または Oracle Essbase (Oracle Hyperion Planning および Essbase の場合)の計算スクリプト文のみ。
- **条件コンポーネント** - true または false である条件文(つまり、If...Then 文)。
- **メンバー・ブロック・コンポーネント** - 指定された 1 つのメンバーが含まれます (Planning および Essbase ブロック・ストレージ・アプリケーション・ユーザーのみ)。
- **メンバー範囲コンポーネント、つまりメタデータ・ループ** - メタデータ・メンバーのリスト(勘定科目のリストなど)が含まれます。
- **固定ループ・コンポーネント** - メタデータのループ(たとえば、勘定科目などのメンバーのリスト全体でのループ)が含まれます。

Financial Management の固定ループには、データ・ループ(たとえば、Financial Management データ全体での固定回数のループ)も含まれます。
- **メタデータ・ループ・コンポーネント** - (カスタム・テンプレート・ユーザーのみ)親ディメンションの子に適用可能な 1 つ以上のパラメータが含まれます。
- **DTP 割当てコンポーネント** - (カスタム・テンプレートのユーザーのみ)定義した設計時プロンプトおよび条件が含まれます。
- **データ範囲コンポーネント(またはデータ・ループ)** - (Financial Management ユーザーのみ)データ・レコードのリストが含まれます(勘定科目の値のリストなど)。

式およびスクリプト・コンポーネントは、これらのコンポーネントを使用するルールおよびテンプレートとは別個に作成できます。これらは別個のオブジェクトであるため、システム・ビュー内で開いたり、保存、編集、削除、およびエクスポートできます。

その他のコンポーネント・タイプは、式およびスクリプト・コンポーネントとは異なり、ルールおよびテンプレート内で作成する必要があります。ルールおよびテンプレートに属するそれらのコンポーネントを、別個に開いたり、保存、削除、またはエクスポートしたりできません。

 ノート:

コンポーネントを作成する際は、作業しているビジネス・ルール、コンポーネント、テンプレートおよび変数を開いたままにできます。Oracle Hyperion Calculation Manager では、コンポーネントを作成する際にタブ間を簡単に移動できるように、タブ付きインターフェイスでこれらのオブジェクトが表示されます。Calculation Manager 内では最大 10 個のタブを開けますが、最適なパフォーマンスを得るためには、10 個を超えるオブジェクトを同時に開かないことをお勧めします。

式コンポーネント

式コンポーネントは、式の計算文で構成されています。

次も参照:

- [式コンポーネントについて](#)
- [式コンポーネントの作成](#)
- [式コンポーネントの設計](#)
- [式コンポーネントを開く](#)
- [式コンポーネントの編集](#)
- [式コンポーネントの削除](#)
- [式コンポーネントのコピーおよび貼付け](#)

式コンポーネントについて

式コンポーネントは、式の計算文で構成されています。式の計算文を作成するには、メンバー、関数および条件を入力または選択します。式を作成する際、コンポーネント・デザイナーのグリッド内の行にその各計算文がリストされます。

 ノート:

式コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

式コンポーネントの作成

式コンポーネントは、システム・ビューから作成できます。ビジネス・ルールおよびテンプレートで式コンポーネントを使用できます。

 ノート:

式コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

式コンポーネントを作成するには:

1. システム・ビューで「新規オブジェクト」アイコンをクリックします。
2. 「アプリケーション・タイプ」に入力します。
3. アプリケーションを選択します。

アプリケーションは、Financial Management、Planning アプリケーション管理を使用して作成されたアプリケーション、または Essbase ブロック・ストレージ・アプリケーションである必要があります。

4. 次のいずれかのタスクを行います:
 - a. Financial Management を選択した場合は、「計算タイプ」を選択します。
 - b. Planning を選択した場合は、「プラン・タイプ」を選択します。
 - c. Essbase を選択した場合は、「データベース」を選択します。

 ノート:

システム・ビューから、式を右クリックし、「新規」を選択して新しい式を作成すると、「新規式」ダイアログにアプリケーション・タイプ名、アプリケーション名、および使用している計算タイプ、プラン・タイプまたはデータベース名が移入されます。

5. 「オブジェクト・タイプ」として「式」を選択します。
6. 式の名前を入力し、「OK」をクリックします。

コンポーネント・デザイナーに式が表示されます。式コンポーネントを設計するには、[式コンポーネントの設計](#)を参照してください。

式コンポーネントの設計

式コンポーネントは、システム・ビューまたは他の任意のビュー内から作成できます。ビジネス・ルールやテンプレートを設計するときに、ルール・デザイナーまたはテンプレート・デザイナー内で式コンポーネントを作成することもできます。ビジネス・ルールおよびテンプレートで式コンポーネントを使用できます。

 ノート:

式コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカラム・テンプレートで使用できます。

式コンポーネントを作成するには:

1. システム・ビューで「式」を右クリックし、「新規」を選択します。

「新規式」ダイアログには、アプリケーション・タイプ、アプリケーション、および計算タイプ、プラン・タイプまたはデータベースが移入されます。

 ノート:

新しい式コンポーネントをビジネス・ルールまたはテンプレートのフロー・チャートにドラッグして、ビジネス・ルールまたはテンプレート内で式コンポーネントを作成することもできます

2. 名前、アプリケーション・タイプ、および式コンポーネントが属するアプリケーションを入力するか、デフォルトを受け入れます。次に、次のいずれかを行います:
 - **Financial Management ユーザーのみ:** Financial Management 式コンポーネントを作成する場合には、計算タイプを入力します。
 - **Planning ユーザーのみ:** Planning 式コンポーネントを作成する場合には、プラン・タイプを入力します。
 - **Essbase ユーザーのみ:** Essbase 式コンポーネントを作成する場合には、データベースを入力します。
3. 「OK」をクリックします。
4. オプション: コンポーネント・デザイナーで、次のいずれかのタスクを実行できます:
 - 「グリッドの追加」をクリックし、「グリッドの挿入(前)」、「グリッドの挿入(後)」または「末尾にグリッドを挿入」を選択して、グリッドを追加します。デフォルトでは、「式」タブに1つのグリッドが表示されます。
 - 「グリッドの削除」をクリックして、選択したグリッドを削除します。
 - 検索アイコンをクリックして、使用中の式グリッドのテキストを検索します。次のタスクを実行します。
 - スクリプト内のテキストを検索して置換する場合は、「検索/置換」アイコンをクリックします。
5. 「式」に、式のキャプションを入力します。
6. テンプレート用に式コンポーネントを作成しているユーザーのみ: 式コンポーネント内で設計時プロンプトを使用する場合は、「式」タブで「設計プロンプトの使用」を選択します。次に条件グリッドで、次のタスクを行って設計時プロンプトの条件を定義します。
 - a. 「DTP」で設計時プロンプトを選択します。
 - b. 「演算子」で、演算子「=」または「<>」を選択します。

- c. 「値」で、ドロップダウン・リストから値を選択します。
- d. これらのステップを繰り返し、条件に必要な数の文を作成します。

 ヒント:

グリッドから行を追加または削除するには、プラス(+)アイコンまたはマイナス(-)アイコンをクリックします。「AND」の横のフィールドをクリックし、ドロップダウン・リストから「OR」を選択して、「AND」を「OR」に変更します。

- 7. **Planning および Essbase ユーザーのみ:** デフォルトでは、グリッドで入力した最初のメンバーを使用して式コンポーネントの計算処理が開始されます。別のメンバーを使用して式コンポーネントの計算処理を開始するには、「**メンバー・ブロック**」にメンバーまたは関数名を入力するか、**省略記号**アイコンをクリックしてメンバー・セレクトからメンバーを選択します。[コンポーネントへのメンバーおよび関数の追加について](#)を参照してください。
- 8. 「**コメント**」に、作成する条件文および式の文のコメントを入力します。
- 9. **オプション:** 式コンポーネントの条件文(つまり IF 文)を作成するには、「**条件の追加**」をクリックします。[条件ビルダーを使用した条件文の作成](#)を参照してください。
- 10. **オプション:** 表示された IF 行に、条件文のテキストを入力するか、行の右側の列で「**条件の追加/編集**」アイコンをクリックして条件ビルダーにアクセスします。条件ビルダーを使用すると、条件文をグラフィカルに設計できます。

 ノート:

下矢印から IF、ELSE IF、および ELSE を選択できますが、デフォルトでは最初の文は IF 文にする必要があります。

- 11. **オプション:** 条件ビルダーを使用して、IF 条件文を設計します。[条件ビルダーを使用した条件文の作成](#)を参照してください。
- 12. 「**式**」行で、式の文を作成するには、等号の**左側**にある行内をクリックします。「**アクション**」アイコンをクリックして、次を選択します:

 ノート:

ステップ 9 からステップ 11 で条件文(つまり IF 文)を作成した場合、このステップで作成する式の文は条件の THEN 文です。

- 変数([変数の操作](#)を参照してください。)
- メンバー([コンポーネントへのメンバーおよび関数の追加](#)を参照してください。)
- 関数([関数の操作](#)を参照してください。)

- **Planning ユーザーのみ:** スマート・リスト([スマート・リストの操作](#)を参照してください。)
13. 「式」行で、式の文を完成させるには、等号の右側にある行内をクリックします。「アクション」アイコンをクリックして、次を選択します:
- 変数([変数の操作](#)を参照してください。)
 - メンバー([コンポーネントへのメンバーおよび関数の追加](#)を参照してください。)
 - 関数([関数の操作](#)を参照してください。)
 - **Planning ユーザーのみ:** スマート・リスト([スマート・リストの操作](#)を参照してください。)
14. **Financial Management ユーザーのみ:** 文ごとに、「**ロギング情報**」アイコンをクリックしてログのオプション条件文を入力します。ログ情報は、条件に合致する場合のみログ・ファイルに記録されます。ログ情報を保存すると、ログ・テキストが存在することを示す横線の付いた「コメント」アイコンが表示されます。[式の文のロギング情報の入力 \(Financial Management ユーザーのみ\)](#)を参照してください。

 **ノート:**

ログ・テキストをログ・ファイルに含めるには、「ロギングの使用可能」を選択する必要があります。

15. 「コメント」アイコンをクリックし、計算文行に関するコメントを入力します。「OK」をクリックします。
16. **オプション:** 別の IF 文、または ELSE IF 文や ELSE 文を作成する場合は、次のステップを実行します:
- a. 「**条件の追加**」をクリックします。
 - b. **下矢印**をクリックし、次のいずれかのオプションを選択します:
 - **IF:** IF 条件文を作成するには、これを選択します。IF 文が TRUE の場合、アクションが実行されます。IF 文が FALSE の場合、他のアクションが実行されます。
 - **ELSE IF:** ELSE IF 条件文を作成するには、これを選択します。関連付けられた IF 文が存在し、IF 文が FALSE の場合、この文のアクションが実行されます。
 - **ELSE:** ELSE 条件文を作成するには、これを選択します。関連付けられた ELSE IF 文が存在し、ELSE IF 文が FALSE の場合、この文のアクションが実行されません。

 **ノート:**

IF ブロックまたは ELSE IF ブロックから条件文を除去する場合で、次のブロックに ELSE IF 文が含まれている場合は、次のブロックが IF ブロックに変更されます。ただし、次のブロックに ELSE 文が含まれている場合は、このブロック内の条件文が除去されます。

- c. 式コンポーネント用の追加の文を設計するには、ステップ 9 からステップ 15 を繰り返します。式グリッドは、式の文のコレクションを含む 1 つ以上の式ブロックで構成

される場合があります。さらに、ブロックのコメントおよび条件を入力することもできますが、これはオプションです。

 ヒント:

必要に応じて、「+」アイコンをクリックして式の行をさらに追加します。

17. 「プロパティ」で、次のステップを実行します:

- a. **ビジネス・ルール用に式コンポーネントを作成しているユーザーのみ:** 式コンポーネントを「新規オブジェクト」からビジネス・ルールにドラッグ・アンド・ドロップする場合、「共有」は選択されません。この式を共有にするには、式の「プロパティ」タブを選択し、「共有」を選択し、式に名前を付けます。

システム・ビューから式コンポーネントを作成すると、デフォルトで「共有」が選択され、変更できません。かわりに、ビジネス・ルールの式のコピーを作成する場合は、ビジネス・ルールに共有式をドラッグし、「プロパティ」タブで「共有」チェック・ボックスをクリアします。これにより、ビジネス・ルール内に共有式のコピーが作成されます。共有式のコピーは、共有式が変更された場合に変更されません。ビジネス・ルールを保存すると、式は名前を持たなくなります。

- b. **オプション:** 新規名前を 50 文字以下で入力することにより、式の名前を編集します。(デフォルトの名前は「新規式」ダイアログで指定されます。)
- c. **オプション:** 式の説明を 255 文字以下で入力します。
- d. **オプション:** 式のキャプションを入力します。キャプションは、**デザイナー**および**テンプレート・デザイナー**のフロー・チャートで式の下に表示されます。

 ノート:

コンポーネントにキャプションがなく、コンポーネントが共有されていない場合、コンポーネントの最初の部分はフロー・チャートに表示されます。たとえば、共有されておらずキャプションのない "Salaries" = 40 という式がある場合は、フロー・チャートで式コンポーネントに対して "Salaries" = 40 と表示されます。コンポーネントが共有され、キャプションを持たない場合は、コンポーネントの名前がフロー・チャートに表示されます。

- e. 式のコメントを入力します。たとえば、式の用途をユーザーに伝えられます。

18. **Financial Management ユーザーのみ:** 「**ロギングの使用可能**」を選択して、式にログ・テキストがある場合は、ルールが起動されたときにログ・ファイルにそのテキストを追加するようにします。ルール、ルールセットおよびコンポーネントのロギングを使用可能にできます。ただし、追加するログ・テキストがある場合は、ビジネス・ルールの式の文でそのテキストを指定します。

 ノート:

ルールセットに対してはロギングが使用不可になっているが、そのルールに含まれるルールおよびコンポーネントに対しては使用可能になっている場合、ルールおよびコンポーネントのログ設定よりもルールセットのログ設定の方が優先されるため、ログ・ファイルは作成されません。

19. **Financial Management ユーザー: 「タイマーの使用可能」** を選択して、ルールが起動されたときに、式の処理に要する時間がログ・ファイルに記録されるようにします。タイマーは、ルール、ルールセットおよびコンポーネントに対して使用可能にできます。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能になっているルールがあり、そのルールにタイマーが使用可能になっている 3 つの式が含まれている場合、ルールおよびルール内の各式の処理に要する時間が記録されます。
20. **ビジネス・ルール用の式コンポーネントを作成している Financial Management ユーザーのみ: 「使用不可」** を選択し、スクリプトの作成中に、その式コンポーネントを使用するすべてのビジネス・ルール内で式コンポーネントを一時的に使用不可にします。
21. **ビジネス・ルール用に式コンポーネントを作成しているユーザーのみ: 「用途」** で、式コンポーネントを使用するルールを表示できます。

 ノート:

「使用方法」 タブの情報はどれも編集できません。

22.  をクリックします。

条件ビルダーを使用した条件文の作成

条件ビルダーを使用すると、式および条件コンポーネントの条件文を作成できます。条件文は、If...Then 文とも呼ばれます。条件文の最初の部分(If 部)が TRUE の場合、文の 2 番目の部分(Then 部)も TRUE になります。条件文の最初の部分が FALSE の場合、2 番目の部分は TRUE の場合とそうでない場合があります。

 ノート:

式および条件コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

条件文を作成するには:

1. 式を右クリックし、「開く」を選択します。
2. **コンポーネント・デザイナー**で、「条件の追加」アイコンをクリックします。

条件を作成するために使用する様々なフィールドおよびドロップダウン・リストを持つ 2 つの行がコンポーネント・デザイナーに追加されます。

3. 最初の行で、条件の最初の文を IF、ELSE または ELSE IF のいずれで開始するかを選択し、条件文のテキストを入力します。または、行の右側にある「**条件の追加/編集**」アイコンをクリックして、条件ビルダーにアクセスします。条件ビルダーを使用すると、条件文をグラフィカルに設計できます。

下矢印から IF、ELSE IF および ELSE を選択できますが、デフォルトでは最初の文は IF 文にする必要があります。
4. 条件ビルダーで、「**メタデータの条件**」または「**データの条件**」を選択します。
5. 次のいずれかのタスクを行います:
 - 「**関数**」(メタデータの条件の場合)または「**式**」(データの条件の場合)に値を入力します。
 - 「**関数**」または「**式**」行内をクリックし、「**アクション**」アイコンを表示します。「**アクション**」アイコンをクリックし、式または関数を作成するオプションを選択します:
 - 変数([変数の操作](#)を参照してください。)
 - メンバー([コンポーネントへのメンバーおよび関数の追加](#)を参照してください。)
 - 関数([関数の操作](#)を参照してください。)
6. 次の演算子のいずれかを入力します:
 - = (等しい)
 - < (より大きい)
 - > (より小さい)
 - <> (より大きいか、より小さい)
 - >= (以下)
 - <= (以上)
7. 次のいずれかのタスクを行います:
 - 「**値**」に値を入力します。
 - 「**値**」行内をクリックし、「**アクション**」アイコンを表示します。「**アクション**」をクリックし、式または関数の値を作成するオプションを選択します:
 - 変数([変数の操作](#)を参照してください。)
 - メンバー([コンポーネントへのメンバーおよび関数の追加](#)を参照してください。)
 - 関数([関数の操作](#)を参照してください。)
8. 条件のコメントを入力します。
9. 「条件」グリッドに条件を追加するには、「**+**」アイコンをクリックします。「**+**」アイコンを使用し、選択した行のコピーを作成して「条件」グリッドに追加することもできます。「条件」グリッドの選択した行を、メタデータまたはデータの条件と置換するには、「**-**」アイコンを使用します。

 **ヒント:**

グリッド内で条件文を並べ替えるには、上矢印および下矢印を使用します。「グループ」アイコンと「グループ解除」アイコンを使用して、条件文のパーツをグループ化(丸カッコを追加)およびグループ解除(丸カッコを除去)します。「AND」の横のフィールドをクリックし、ドロップダウンを使用してオプションを選択することで、「AND」を「OR」、And Not または Or Not に変更します。

10. 「OK」をクリックして「条件ビルダー」ダイアログを終了します。

IF 文に条件が挿入されます。

11. **オプション:** 作成する条件文ごとに、ステップ 2 からステップ 10 を繰り返します。

12. **オプション:** 追加の条件文ごとに、最初の列をダブルクリックしてドロップダウン・リストを表示し、各条件文を開始するための演算子を選択できます:

- IF: IF が付いた最初の条件文だけを開始できます。(これは、作成した最初の条件文にだけ表示されるデフォルトです。)
- NOT IF: NOT IF が付いた最初の条件文だけ開始できます。
- AND: 2 つ以上の条件文が複合したものを作成する場合に、AND が付いた任意の条件文(最初の文を除く)を開始できます。
- OR: 2 つ以上の条件文が複合したものを作成する場合に、OR が付いた任意の条件文(最初の文を除く)を開始できます。
- AND NOT: その後に式コンポーネントが続く条件文を含める場合に、AND NOT が付いた任意の条件文(最初の文を除く)を開始できます。
- OR NOT: その後に式コンポーネントが続く条件文を除外する場合に、OR NOT が付いた任意の条件文(最初の文を除く)を開始できます。

 **ヒント:**

条件文に丸カッコを追加および除去するには、「グループ」アイコンと「グループ解除」アイコンを使用します。複数の条件文をグループ化またはグループ解除するには、[Ctrl]キーを押しながらクリックするか、[Shift]キーを押しながらクリックして、グループ化またはグループ解除する条件文を選択します。

13. 「OK」をクリックします。

条件文が条件行に挿入されます。

式の文のロギング情報の入力(Financial Management ユーザーのみ)

Oracle Hyperion Financial Management の式の文のロギング情報を入力できます。

ロギング情報を入力するには:

1. **システム・ビュー**で、Financial Management アプリケーション・タイプ、アプリケーション、計算タイプ、「式」の順に展開します。ロギング情報を追加する式を右クリックし、「開く」を選択します。

2. 「コンポーネント・デザイナー」で、文ごとに「ロギング情報」アイコンをクリックして、ログのオプションの条件文を入力します。ログ情報は、条件に合致する場合のみログ・ファイルに書き込まれます。ロギング情報を保存すると、ログ・テキストが存在することを示す横線の付いた「コメント」アイコンが表示されます。

 ノート:

「ロギングの使用可能」を選択して、ログ・テキストをログ・ファイルに含めます。

3. 変数を使用する場合は、「変数の使用」を選択します。次に、「変数」から変数を選択します。
4. ログ・テキストを標準フォーマットで表示する場合は、「標準表示フォーマットの使用」を選択します。
5. 「ログ・テキスト」に、ログの条件文を入力するか、「条件の追加/編集」アイコンをクリックし、条件ビルダーを使用して条件文を作成します。[条件ビルダーを使用した条件文の作成](#)を参照してください。
6. 「ログ式」に式を入力し、「OK」をクリックします。

式の文のコメントの入力

 ノート:

式の文は、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

式の文のコメントを入力するには:

1. 「式」を展開した状態で、次のいずれかのタスクを実行します。
 - コメントを追加する式がビジネス・ルールに含まれている場合、「ルール」を展開し、式コンポーネントが含まれているルールを右クリックして「開く」を選択します。
 - コメントを追加する式がテンプレートに含まれている場合、「テンプレート」ノードを展開し、式コンポーネントが含まれているルールを右クリックして「開く」を選択します。
 - 式を単独で開く場合は、「式」ノードを展開し、式を右クリックして「開く」を選択します。
2. 次のいずれかのタスクを行います:
 - ビジネス・ルール内の式コンポーネントにコメントを追加する場合は、**ルール・デザイナー**のフロー・チャートで、式コンポーネントおよび「式」タブを選択します。「コメント」テキスト・ボックスにコメントを入力します。
 - テンプレート内の式にコメントを追加する場合は、**テンプレート・デザイナー**のフロー・チャートで、式コンポーネントおよび「プロパティ」タブを選択します。「コメント」テキスト・ボックスにコメントを入力します。

- 式コンポーネントのコメントを単独で追加する場合は、**コンポーネント・デザイナー**で、「**コメント**」テキスト・ボックスにコメントを入力します。

3.  をクリックします。

式コンポーネントを開く

式コンポーネントは、システム・ビューから、またはルール・デザイナーやテンプレート・デザイナーのフロー・チャート内から開くことができます。

ノート:

式コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

式コンポーネントを開くには、**システム・ビュー**で「**式**」を展開し、式を右クリックして「**開く**」を選択します。

ノート:

式コンポーネントがビジネス・ルールで使用されている場合、式コンポーネントを右クリックして「開く」を選択するか、式コンポーネントをダブルクリックして、ルール・デザイナーでルールのフロー・チャート内から開くこともできます。

式コンポーネントの編集

式コンポーネントの計算結果をログ・ファイルに含めるかどうか(Oracle Hyperion Financial Management アプリケーションの場合のみ)、および式コンポーネントのコメント、キャプション、名前および説明で構成される文を編集できます。

ノート:

式コンポーネントは、Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

式コンポーネントを編集するには:

- システム・ビュー**で式を右クリックし、「**開く**」を選択します。
- コンポーネント・デザイナー**で式のプロパティを編集し、 をクリックします。

[式コンポーネントの設計](#)および[条件ビルダーを使用した条件文の作成](#)を参照してください。

式コンポーネントの削除

式コンポーネントは、ルールまたはテンプレートで使用されていない場合にのみ削除できます。ルールまたはテンプレートで式コンポーネントが使用されているかどうかを確認するには、使用状況を表示します。[\(式およびスクリプト・コンポーネントの使用の表示を参照してください。\)](#)式コンポーネントがルールまたはテンプレートで使用されているが、そのルールまたはテンプレートでの使用が必要でない場合は、ルールまたはテンプレートから除去した後、式コンポーネントを削除できます。ルールまたはテンプレートを削除すると、それに含まれる式コンポーネントも削除されます。

ノート:

式コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

式コンポーネントを削除するには:

1. **システム・ビュー**で式を右クリックし、「**使用状況の表示**」を選択します。
ある場合は、ルールまたはテンプレートから式を除去する必要があります。
2. 式コンポーネントを使用しているルールまたはテンプレートがないことを確認します。
[コンポーネントのフロー・チャートからの除去](#)を参照してください。
3. 式を再度右クリックし、「**削除**」を選択します。

式コンポーネントのコピーおよび貼付け

ルールまたはテンプレートから式コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。式コンポーネント内の条件グリッドのコンテンツをコピーして、同じ式コンポーネントや異なる式コンポーネントに貼り付けることもできます。式コンポーネントをコピーして、別の式コンポーネントや別のコンポーネント・タイプに貼り付けることはできません。

ノート:

式コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

式コンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートで、式コンポーネントを右クリックし、「**コピー**」を選択します。

コピーするコンポーネントが共有されている場合、式コンポーネントを右クリックして「参照のコピー」を選択し、コンポーネント自体をコピーするかわりに参照を共有コンポーネントにコピーできます。(ビジネス・ルールの式またはスクリプト・コンポーネントへの参照のコピーおよび貼付けを参照してください。)

コピーするコンポーネントに他のコンポーネント(子)が含まれている場合は、[Ctrl]を押しながらそれらをすべて選択して右クリックし、「グループのコピー」を選択して、グループのすべてのコンポーネントをコピーできます。(フロー・チャート内でのコンポーネント・グループのコピーおよび貼付けを参照してください。)

3. 次のいずれかのタスクを行います:

- 式コンポーネントを同じビジネス・ルールまたはテンプレートのフロー・チャートに貼り付けるには、フロー・チャートの式コンポーネントを貼り付ける場所で右クリックし、「貼付け」を選択します。
- 式コンポーネントを異なるビジネス・ルールまたはテンプレートのフロー・チャートに貼り付けるには、コンポーネントを貼り付けるビジネス・ルールまたはテンプレートを開き、フロー・チャートの式コンポーネントを貼り付ける場所で右クリックして「貼付け」を選択します。

4.  をクリックします。

スクリプト・コンポーネント

ビジネス・ルールおよびテンプレートでスクリプト・コンポーネントを使用できます。

次も参照:

- [スクリプト・コンポーネントの作成](#)
- [スクリプト・コンポーネントの設計](#)
- [スクリプト・コンポーネントを開く](#)
- [スクリプト・コンポーネントの編集](#)
- [スクリプト・コンポーネントの削除](#)
- [スクリプト・コンポーネントのコピーおよび貼付け](#)

スクリプト・コンポーネントの作成

スクリプト・コンポーネントは、システム・ビューから作成できます。ビジネス・ルールおよびテンプレートでスクリプト・コンポーネントを使用できます。

ノート:

スクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

スクリプト・コンポーネントを作成するには:

1. スクリプトの名前およびアプリケーション・タイプを入力します。

2. **アプリケーション名**を選択します。

アプリケーション名は、Financial Management、Planning アプリケーション管理を使用して作成されたアプリケーション、または Essbase ブロック・ストレージ・アプリケーションである必要があります。

3. 次のいずれかのタスクを行います:

- a. Financial Management を選択した場合は、「**計算タイプ**」を選択します。
- b. Planning を選択した場合は、「**プラン・タイプ**」を選択します。
- c. Essbase を選択した場合は、「**データベース**」を選択します。

 **ノート:**

システム・ビューで、スクリプトを右クリックし、「新規」を選択する場合、「新規スクリプト」ダイアログ・ボックスには、アプリケーション・タイプ、アプリケーション、および使用している計算タイプ(Financial Management スクリプトの場合)、プラン・タイプ(Planning スクリプトの場合)またはデータベース(Essbase スクリプトの場合)が移入されます。

スクリプト・コンポーネントの設計

ビジネス・ルールおよびテンプレートでスクリプト・コンポーネントを使用できます。スクリプト・コンポーネントは、システム・ビューまたは他の任意のビューから作成します。ビジネス・ルールやテンプレートを設計するときに、ルール・デザイナまたはテンプレート・デザイナ内でスクリプト・コンポーネントを作成することもできます。

 **ノート:**

スクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

 **ノート:**

ビジネス・ルールまたはテンプレート内からスクリプト・コンポーネントを設計することもできます。

スクリプト・コンポーネントを設計するには:

1. システム・ビューで「**スクリプト**」を右クリックし、「**新規**」を選択します。
2. スクリプトの名前、アプリケーション・タイプおよびアプリケーションを入力します。その後、次のタスクのいずれかを行います:

- **Financial Management ユーザーのみ:** Financial Management スクリプトを作成する場合には、計算タイプを入力します。
 - **Planning ユーザーのみ:** Planning スクリプトを作成する場合には、プラン・タイプを入力します。
 - **Essbase ユーザーのみ:** Essbase スクリプトを作成する場合には、データベースを入力します。
3. 「OK」をクリックします。
 4. スクリプト・コンポーネントを設計するアイコンを使用します。

Financial Management スクリプトは Visual Basic に存在する必要があります。Planning および Essbase のスクリプトは Essbase に存在する必要があります。

ノート:

ルール、テンプレートおよび他のコンポーネントとは異なり、スクリプトを開くときは、グラフィカル・フォーマットで(フロー・チャート内に)表示できません。スクリプト・フォーマットでのみ表示できます。

スクリプトを使用して、次のタスクを実行できます:

- スクリプトの行番号を非表示にする(または表示する)には、「**行番号の非表示/行番号の表示**」アイコンをクリックします。行番号はデフォルトで表示されています。
- スクリプトに関数を挿入するには、「**関数とそのパラメータの挿入**」アイコンをクリックします。[関数の操作](#)を参照してください。
- スクリプトにメンバーを挿入するには、「**ディメンションから選択したメンバーの挿入**」アイコンをクリックします。[コンポーネントへのメンバーおよび関数の追加](#)を参照してください。
- スクリプトに変数を挿入するには、「**変数の挿入**」アイコンをクリックします。[変数の操作](#)を参照してください。

ノート:

Visual Basic スクリプトは、自己完結している必要があります。スクリプトで、ルールセットに定義された置換変数や実行変数を参照できません。スクリプトで変数を使用する必要がある場合、スクリプト・コンポーネント内で宣言する必要があります。

- **Planning ユーザーのみ:** スクリプトにスマートリストを挿入するには、「**スマートリストの挿入**」をクリックします。[スマート・リストの操作](#)を参照してください。
- スクリプトへのコメント行の追加および削除を行うには、「**コメント**」および「**コメント解除**」アイコンをクリックします。
- スクリプトでシステム生成のコメントを非表示にするには、「**コメントの非表示**」アイコンをクリックします。
- スクリプトの構文にエラーがあるかどうかを確認するには、「**構文の確認**」アイコンをクリックします。

 ノート:

構文エラーがある場合、エラー・メッセージは表示されず、エラーのあるコードのテキストが赤くなります。構文エラーがない場合は、「構文エラーは見つかりませんでした。」というメッセージが表示されます。

- 「折返し」アイコンをクリックして、ページでスクロールされるスクリプトの長い行が同じページで複数の行に表示されるようにします。
- コード補完の提案を抑制するには、「**コード補完を OFF に設定**」アイコンをクリックします。(このアイコンはトグルとして機能します。コード補完をオンにするには、アイコンを再度クリックします。コード補完はデフォルトでオンに設定されています。)
- 「置換」アイコンをクリックして、スクリプト内のテキスト文字列を検索して置換します。[ビジネス・ルール・スクリプト内のテキスト文字列の検索](#)を参照してください。
- スクリプト内のテキスト文字列を検索するには、「**検索**」アイコンをクリックします。[ビジネス・ルール・スクリプト内のテキスト文字列の検索](#)を参照してください。
- 「**検索**」に検索テキストを入力して、「**前**」または「**次**」をクリックして、テキストの前(スクリプト内を上を検索)または次(スクリプト内を下を検索)のインスタンスを検索します。
- 「**行に移動**」に行番号を入力して、矢印をクリックしスクリプト内の行番号をハイライト表示します。
- 行の左側にあるプラスまたはマイナス記号をクリックして、スクリプト内の行を展開または縮小します。たとえば、FIX 文では、表示される文のすべての行がデフォルトで表示されますが、左側にあるマイナス記号をクリックすると FIX 文の行を非表示にできます。
- コピーするスクリプトの行を選択し、「**クリップボードにコピー**」アイコンをクリックします。
- 切り取るスクリプトの行を選択し、「**クリップボードに切り取り**」アイコンをクリックします。
- 貼り付けるスクリプトの場所にカーソルを置き、「**クリップボードから貼付け**」アイコンをクリックします。
- 関数には、自動推奨機能があります。関数の最初の数文字を入力して、**[Ctrl]** を押しながら**[Space]**を押すと、推奨内容が表示されます。推奨内容のいずれかを選択してクリックするか、**[Enter]**を押して、関数を完成させます。

5. 「**プロパティ**」で、次のステップを実行します:

- a. スクリプト・コンポーネントを作成する場合、「**共有**」は選択されません。スクリプト・コンポーネントを共有にするには、「**プロパティ**」タブ、「**共有**」の順に選択し、名前を入力します。

システム・ビューからスクリプト・コンポーネントを作成すると、デフォルトで「**共有**」が選択され、変更できません。かわりに、ビジネス・ルールのスクリプトのコピーを作成する場合は、ビジネス・ルールに共有スクリプトをドラッグし、「**プロパティ**」タブで「**共有**」チェック・ボックスをクリアします。これにより、ビジネス・ルール内に共有スクリプトのコピーが作成されます。

共有スクリプトのコピーは、共有スクリプトが変更された場合に変更されません。ビジネス・ルールを保存すると、スクリプトは名前を持たなくなります。

- b. **オプション:** 新規名前を 50 文字以下で入力することにより、スクリプトの名前を編集します。(デフォルトの名前は「新規スクリプト」ダイアログで指定されます。)

ノート:

共有スクリプトにのみ名前を付けることができます。スクリプトが共有されていない場合は、「名前」フィールドに入力することはできません。

- c. スクリプトの説明を 255 文字以下で入力します。
 - d. スクリプトのキャプションを入力します。キャプションは、**ルール・デザイナー**および**テンプレート・デザイナー**のフロー・チャートでスクリプトの下に表示されます。
 - e. スクリプトのコメントを入力します。たとえば、スクリプトの用途をユーザーに伝えられます。
6. **Financial Management ユーザーのみ:** 「**タイマーの使用可能**」を選択して、ルールが起動されたときに、スクリプトの処理に要する時間がログ・ファイルに記録されるようにします。タイマーは、ルール、ルールセットおよびコンポーネントに対して使用可能にできます。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能になっているルールがあり、そのルールにタイマーが使用可能になっている 3 つのスクリプトが含まれている場合、ルールおよびルール内の各スクリプトの処理に要する時間が記録されます。
 7. **Financial Management ユーザーのみ:** 「**埋込み**」を選択し、スクリプト・コンポーネントのコンテンツを、Visual Basic スクリプトを生成するときそのコンテンツを使用するルール内に埋め込みます。
 8. **Financial Management ユーザーのみ:** 「**パラメータ**」に、スクリプト・パラメータを入力します。
 9. **Financial Management、Planning および Essbase ブロック・ストレージ・ユーザーのみ:** 「**スクリプト**」に、スクリプト・コンポーネントのキャプションを入力します。
 10. **テンプレート用にスクリプト・コンポーネントを作成しているユーザーのみ:** スクリプト・コンポーネントで設計時プロンプトを使用する場合は、「**設計プロンプトの使用**」を選択します。次に条件グリッドで、次のタスクを行って設計時プロンプトの条件を定義します。
 - a. 「**DTP**」で設計時プロンプトを選択します。
 - b. 「**演算子**」で、演算子「**=**」または「**<>**」を選択します。
 - c. 「**値**」では、ドロップダウン・リストを使用して値を選択します。このフィールドに入力することはできません。
 - d. これらのステップを繰り返し、条件に必要な数の文を作成します。

ヒント:

グリッドから行を追加または削除するには、プラス(+)アイコンまたはマイナス(-)アイコンをクリックします。

11. 独立スクリプト・コンポーネント(つまり、ビジネス・ルールまたはテンプレート内にはないスクリプト・コンポーネント)またはビジネス・ルールのスクリプト・コンポーネントを作成するユーザーのみ: 「使用方法」では、スクリプト・コンポーネントを使用するルールを表示できます。

 **ノート:**

このタブで情報は編集できません。

12.  をクリックします。

スクリプト・コンポーネントを開く

スクリプト・コンポーネントは、システム・ビューから、またはルール・デザイナーやテンプレート・デザイナーのフロー・チャート内から開くことができます。

 **ノート:**

スクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

スクリプト・コンポーネントを開くには:

1. システム・ビューで「スクリプト」を展開します。
2. スクリプトを右クリックし、「開く」を選択します。

 **ノート:**

共有スクリプト・コンポーネントがビジネス・ルールで使用されている場合、スクリプト・コンポーネントを右クリックして「開く」を選択するか、スクリプト・コンポーネントをダブルクリックして、ルール・デザイナーでルールのフロー・チャート内から開くこともできます。

スクリプト・コンポーネントの編集

使用する関数、変数およびメンバーを編集して、スクリプト・コンポーネント、スクリプト・コンポーネント内の式の文の計算結果をログ・ファイルに含めるかどうか (Oracle Hyperion Financial Management アプリケーションの場合のみ)、およびスクリプト・コンポーネントのコメント、キャプション、名前および説明を記述できます。

 **ノート:**

スクリプト・コンポーネントは、Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

スクリプト・コンポーネントを編集するには:

1. **システム・ビュー**でスクリプトを右クリックし、「**開く**」を選択します。

 **ノート:**

ルール、テンプレートおよび他のコンポーネントとは異なり、スクリプトを開くときは、グラフィカル・フォーマットで(フロー・チャート内に)表示できません。スクリプト・フォーマットでのみ表示できます。

2. スクリプトの目的のプロパティを編集し、 をクリックします。

[スクリプト・コンポーネントの設計](#)を参照してください。

スクリプト・コンポーネントの削除

スクリプト・コンポーネントは、ルールまたはテンプレートで使用されていない場合にのみ削除できます。ルールまたはテンプレートでスクリプト・コンポーネントが使用されているかどうかを確認するには、使用状況を表示します。(式およびスクリプト・コンポーネントの使用の表示を参照してください。)スクリプト・コンポーネントがルールまたはテンプレートで使用されているが、そのルールまたはテンプレートでの使用が必要でない場合は、ルールまたはテンプレートから除去した後、式コンポーネントを削除できます。ルールまたはテンプレートを削除すると、それに含まれるスクリプト・コンポーネントも削除されます。

 **ノート:**

スクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

スクリプト・コンポーネントを削除するには:

1. **システム・ビュー**でスクリプトを右クリックし、「**使用状況の表示**」を選択します。
2. スクリプト・コンポーネントを使用しているルールまたはテンプレートがないことを確認します
ある場合は、ルールまたはテンプレートからスクリプト・コンポーネントを除去する必要があります。[コンポーネントのフロー・チャートからの除去](#)を参照してください。
3. スクリプトを再度右クリックし、「**削除**」を選択します。

スクリプト・コンポーネントのコピーおよび貼付け

ルールまたはテンプレートからスクリプト・コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。スクリプト・コンポーネント内のスクリプトをコピーして、同じスクリプト・コンポーネントや異なるスクリプト・コンポーネントに貼り付けることもできます。スクリプト・コンポーネントをコピーして、別のスクリプト・コンポーネントや別のコンポーネント・タイプに貼り付けられません。

ノート:

スクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

スクリプト・コンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**でスクリプト・コンポーネントを含むルールまたはテンプレートを選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートで、スクリプト・コンポーネントを右クリックし、「**コピー**」を選択します。

ノート:

コピーするコンポーネントが共有されている場合、スクリプト・コンポーネントを右クリックして「**参照のコピー**」を選択し、コンポーネント自体をコピーするかわりに共有コンポーネントの参照をコピーできます。(ビジネス・ルールの式またはスクリプト・コンポーネントへの参照のコピーおよび貼付けを参照してください。)

3. 次のいずれかのタスクを行います:
 - スクリプト・コンポーネントを**同じ**ビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートのコンポーネントを貼り付ける場所で右クリックし、「**貼付け**」を選択します。
 - スクリプト・コンポーネントを**異なる**ビジネス・ルールまたはテンプレートに貼り付けるには、スクリプト・コンポーネントを貼り付けるビジネス・ルールまたはテンプレートを開き、フロー・チャートのコンポーネントを貼り付ける場所で右クリックして「**貼付け**」を選択します。
4.  をクリックします。

条件コンポーネント

条件コンポーネントは、TRUE または FALSE となる条件文で構成されます。

次も参照:

- [条件コンポーネントについて](#)
- [条件コンポーネントの作成](#)
- [条件コンポーネントを開く](#)
- [条件コンポーネントの編集](#)
- [条件コンポーネントの削除](#)
- [条件コンポーネントのコピーおよび貼付け](#)

条件コンポーネントについて

条件コンポーネントは、TRUE または FALSE となる条件文(つまり、IF...THEN 文)で構成されます。条件が TRUE の場合、指定したアクションが実行されます。条件が FALSE の場合、指定した他のアクションが実行されます。条件は、メタデータの条件の場合とデータの条件の場合があります。条件コンポーネントは共有できません。

 **ノート:**

条件コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

条件コンポーネントの作成

条件コンポーネントは、ビジネス・ルールまたはテンプレート内から作成できます。スクリプトおよび式コンポーネントとは異なり、条件コンポーネントを別個のオブジェクトとして作成できません。作成先のビジネス・ルールまたはテンプレートにリンクされます。これらは共有できません。

 **ノート:**

条件コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

条件コンポーネントを作成するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「開く」を選択します。
2. フロー・チャート内の条件を作成する場所を決定したら、「**新規オブジェクト・パレット**」から**条件**コンポーネントをドラッグし、フロー・チャート内のその場所にドロップします。
フロー・チャートに、条件オブジェクトが疑問符の付いたひし形として表示されます。

3. **オプション: 「条件」** で、条件コンポーネントを識別するキャプションを入力します。このキャプションは、その条件を使用するルールまたはテンプレートのフロー・チャート内で、コンポーネントの上に表示されます。
4. **テンプレート用に条件コンポーネントを作成しているユーザーのみ:** 条件コンポーネントで設計時プロンプトを使用する場合は、「**設計プロンプトの使用**」を選択します。次に条件グリッドで、次のタスクを行って設計時プロンプトの条件を定義します。
 - a. 「**DTP**」で設計時プロンプトを選択します。
 - b. 「**演算子**」で、演算子「**=**」または「**<>**」を選択します。
 - c. 「**値**」では、ドロップダウン・リストを使用して値を選択します。このフィールドに入力することはできません。
 - d. これらのステップを繰り返し、条件に必要な数の文を作成します。

ノート:

設計時プロンプトを使用して条件を定義する場合、「条件」ボックスに条件を入力したり、条件ビルダーを使用して条件を作成したりできません。

5. **Financial Management ユーザーのみ: 「タイマーの使用可能」** を選択して、ビジネス・ルールが起動されたときに、条件コンポーネントの処理に要する時間がログ・ファイルに記録されるようにします。タイマーは、ルール、ルールセットおよびコンポーネントに対して使用可能にできます。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能になっているルールがあり、そのルールにタイマーが使用可能になっている 3 つの条件コンポーネントが含まれている場合、ルールおよびルール内の各条件コンポーネントの処理に要する時間が記録されます。
6. **Planning および Essbase ユーザーのみ:** メンバー・ブロックを作成する場合、**省略記号**アイコンをクリックし、メンバーまたは関数を選択します。デフォルトではグリッドに入力した最初のメンバーから条件コンポーネント計算の処理が開始されます。別のメンバーから処理が開始されるようにするには、「**メンバー・ブロック**」にメンバー名または関数名を入力するか、**省略記号**アイコンをクリックして、メンバー・セレクトタからメンバーを選択します。[コンポーネントへのメンバーおよび関数の追加について](#)を参照してください。
7. 次のいずれかのタスクを行います:
 - 「条件」テキスト・ボックスに条件文を入力します。
 - 「**条件の追加**」をクリックし、条件ビルダーを使用して条件文を作成します。[条件ビルダーを使用した条件文の作成](#)を参照してください。
8. 条件コンポーネントのコメントを入力し、 をクリックします。

条件コンポーネントを開く

条件コンポーネントは、属するビジネス・ツールまたはテンプレートのフロー・チャート内から開きます。式およびスクリプト・コンポーネントとは異なり、システム・ビューからは開けません。

 **ノート:**

条件コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

条件コンポーネントを開くには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. ルールまたはテンプレートが開いている場合、フロー・チャートの条件コンポーネントを選択して、条件プロパティを確認します。

条件コンポーネントの編集

条件コンポーネントの作成に使用する関数、変数およびメンバー、条件コンポーネントの処理に要する時間をログ・ファイルに記録するかどうか(Oracle Hyperion Financial Management アプリケーションの場合のみ)、および条件コンポーネントのコメントおよびキャプションを編集できます。

 **ノート:**

条件コンポーネントは、Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

条件コンポーネントを編集するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「開く」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、プロパティを編集する条件を「**条件**」で選択します。次のいずれかのプロパティを編集できます。(条件コンポーネントの作成を参照してください)
 - キャプション
 - 条件文
 - コメント
 - **テンプレート・ユーザーのみ:** 設計時プロンプト
 - **Financial Management ユーザーのみ:** ログおよびタイマーのオプション
3.  をクリックします。

条件コンポーネントの削除

条件コンポーネントは、属するルールまたはテンプレートから除去することにより削除できます。条件コンポーネントは1つのビジネス・ルールまたはテンプレートでのみ使用できるため、ビジネス・ルールまたはテンプレートから除去するだけで削除できます。

 ノート:

条件コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

条件コンポーネントを削除するには:

1. **システム・デザイナー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャート内の削除する条件コンポーネントを選択します。
3. 条件を右クリックし、「**削除**」を選択します。

条件コンポーネントは共有できないため、条件コンポーネントをビジネス・ルールまたはテンプレートから削除するとデータベースから削除されます。

4. コンポーネントの削除を確定し、 をクリックします。

条件コンポーネントのコピーおよび貼付け

ルールまたはテンプレートから条件コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。条件コンポーネントをコピーして、別のスクリプト・コンポーネントや別のコンポーネント・タイプに貼り付けられません。

 ノート:

条件コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

ルール・デザイナーまたはテンプレート・デザイナーから条件コンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートで、コピーする条件コンポーネントを右クリックし、「**コピー**」を選択します。
3. 次のいずれかのタスクを行います:
 - 条件コンポーネントを**同じ**ビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートの条件コンポーネントを貼り付ける場所で右クリックし、「**貼付け**」を選択します。
 - 条件コンポーネントを**異なる**ビジネス・ルールまたはテンプレートに貼り付けるには、コンポーネントを貼り付けるビジネス・ルールまたはテンプレートを開き、フロー・チャートの条件コンポーネントを貼り付ける場所で右クリックして「**貼付け**」を選択します。

4.  をクリックします。

メンバー・ブロック・コンポーネント

メンバー・ブロック・コンポーネントは、スクリプト内の 1 つ以上の文を囲むようにメンバーを定義します。

次も参照:

- [メンバー・ブロック・コンポーネントについて](#)
- [メンバー・ブロック・コンポーネントの作成](#)
- [メンバー・ブロック・コンポーネントを開く](#)
- [メンバー・ブロック・コンポーネントの編集](#)
- [メンバー・ブロック・コンポーネントの削除](#)
- [メンバー・ブロック・コンポーネントのコピーおよび貼付け](#)

メンバー・ブロック・コンポーネントについて

メンバー・ブロック・コンポーネントは、スクリプト内の 1 つ以上の文を囲むようにメンバーを定義します。たとえば、If 条件は、メンバー・ブロックで囲む必要があります。メンバー・ブロックは、ビジネス・ルールとカスタム・テンプレートの両方で使用できます。

メンバー・ブロックは、式のターゲットの Oracle Essbase 代替変数またはディメンション間メンバーを使用している場合に必要です。メンバー・ブロックは、メンバーが式のターゲットの連続する文で繰り返される場合にも使用できます。

メンバー・ブロック・コンポーネントは、メンバー・ブロックが作成されたビジネス・ルールまたはテンプレートとは別個のデータベース・オブジェクトとして存在しません。メンバー・ブロック・コンポーネントは、それが属するテンプレートまたはビジネス・ルールの一部としてのみ存在します。メンバー・ブロックは、ルールおよびテンプレート間で共有できません。

ノート:

メンバー・ブロック・コンポーネントは、Oracle Hyperion Planning および Essbase ブロック・ストレージのビジネス・ルールおよびカスタム・テンプレートで使用できます。

メンバー・ブロック・コンポーネントの作成

メンバー・ブロック・コンポーネントは、ルールまたはテンプレートを設計するときに、ルール・デザイナーまたはテンプレート・デザイナー内で作成します。システム・ビュー内でメンバー・ブロック・コンポーネントを作成することはできません。

メンバー・ブロック・コンポーネントを作成するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。

2. フロー・チャート内のメンバー・ブロックを作成する場所を決定したら、「**新規オブジェクト**」からメンバー・ブロック・オブジェクトをドラッグし、フロー・チャートにドロップします。
フロー・チャートに、メンバー・ブロック・オブジェクトが接続線の付いた4つのブロックとして表示されます。
3. **オプション:** テンプレート用にメンバー・ブロック・コンポーネントを作成する場合は、その設計時プロンプトも作成できます。[カスタム・テンプレート用の設計時プロンプトの作成](#)を参照してください。
4. 「**メンバー・ブロック**」タブで、「**メンバー・ブロック**」の横の「**アクション**」アイコンをクリックし、次のいずれかを行います:
 - 「**変数**」を選択して、変数を検索または作成してから、「**OK**」をクリックします。[変数の操作](#)を参照してください。
 - 「**メンバー**」を選択して、メンバーを検索してから、「**OK**」をクリックします。[コンポーネントへのメンバーおよび関数の追加について](#)を参照してください。
5. 「**エラーおよび警告**」タブで、スクリプト診断を実行するボタンをクリックします。
6. 「**プロパティ**」で「**使用不可**」を選択して検証および計算からメンバー・ブロックを除外します。次に、オプションの説明およびコメントを入力します。
7.  をクリックします。

メンバー・ブロック・コンポーネントを開く

メンバー・ブロック・コンポーネントは、そのメンバー・ブロックを使用するビジネス・ルールまたはテンプレートのフロー・チャート内から開きます。システム・ビューから開くことはできません。

ノート:

メンバー・ブロック・コンポーネントは、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

メンバー・ブロック・コンポーネントを開くには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. ルールまたはテンプレートを開いたら、フロー・チャートのメンバー・ブロック・コンポーネントを選択してメンバー・ブロックのプロパティを確認します。

メンバー・ブロック・コンポーネントの編集

メンバー・ブロック内のメンバー、メンバー・ブロック・コンポーネントの処理に要した時間をログ・ファイルに記録するかどうか(Oracle Hyperion Financial Management アプリケーション・ユーザーのみ)、およびメンバー・ブロック・コンポーネントの設計時プロンプト(カスタム・テンプレート・ユーザーのみ)を変更できます。

 **ノート:**

メンバー・ブロック・コンポーネントは、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

メンバー・ブロック・コンポーネントを編集するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャート内のメンバー・ブロックを選択して、**メンバー・ブロック**のプロパティを編集します。
メンバー・ブロックに対して選択したメンバーまたは変数、メンバー・ブロックに関する説明とコメント、メンバー・ブロックが無効になっているためビジネス・ルールまたはテンプレートの検証および計算に含まれないかどうかを変更できます。
3.  をクリックします。

メンバー・ブロック・コンポーネントの削除

メンバー・ブロック・コンポーネントを削除するには、それが属するビジネス・ルールまたはテンプレートから除去します。メンバー・ブロック・コンポーネントは、1つのビジネス・ルールまたはテンプレートでのみ使用でき、他のビジネス・ルールまたはテンプレート間で共有することはできません。

 **ノート:**

メンバー・ブロック・コンポーネントは、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

メンバー・ブロック・コンポーネントを削除するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャート内の削除するメンバー・ブロック・コンポーネントを選択します。
3. メンバー・ブロックを右クリックし、「**削除**」を選択します。
メンバー・ブロックは、そのメンバー・ブロックを作成したビジネス・ルールまたはテンプレートにのみ存在するため、メンバー・ブロックを削除すると、データベースから削除されます。

メンバー・ブロック・コンポーネントのコピーおよび貼付け

ルールまたはテンプレートからメンバー・ブロック・コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。メンバー・ブロック・コンポーネントをコピーして、別のメンバー・ブロック・コンポーネントや別のコンポーネント・タイプに貼り付けられません。

 ノート:

メンバー・ブロック・コンポーネントは、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

ルール・デザイナーまたはテンプレート・デザイナーからメンバー・ブロック・コンポーネントをコピーするには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートで、コピーするメンバー・ブロック・コンポーネントを右クリックし、「コピー」を選択します。
3. 次のいずれかのタスクを行います:
 - メンバー・ブロック・コンポーネントを同じビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートの貼り付ける場所を右クリックし、「貼り付け」を選択します。
 - メンバー・ブロック・コンポーネントを異なるビジネス・ルールまたはテンプレートに貼り付けるには、メンバー・ブロック・コンポーネントを貼り付けるビジネス・ルールまたはテンプレートを開き、フロー・チャートの貼り付ける場所を右クリックして「貼り付け」を選択します。
4.  をクリックします。

メンバー範囲コンポーネント

メンバー範囲コンポーネントは、Planning ディメンションからのメンバーの範囲で構成される一種のループです。

次も参照:

- [メンバー範囲コンポーネントについて](#)
- [メンバー範囲コンポーネントの作成](#)
- [メンバー範囲コンポーネントを開く](#)
- [メンバー範囲コンポーネントの編集](#)
- [メンバー範囲コンポーネントの削除](#)
- [メンバー範囲コンポーネントのコピーおよび貼付け](#)

メンバー範囲コンポーネントについて

メンバー範囲コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning または Oracle Essbase ディメンションからのメンバーの範囲で構成される一種のループです。メンバー範囲コンポーネントは共有できないため、ビジネス・ルールまたはテンプレートに新規メンバー範囲コンポーネントを追加するたびに新規メンバー範囲コンポーネントを作成する必要があります。

 **ノート:**

メンバー範囲コンポーネントは、Financial Management、Planning および Essbase ブロック・ストレージのビジネス・ルールおよびカスタム・テンプレートで使用できます。

メンバー範囲コンポーネントの作成

メンバー範囲コンポーネントは、ルールまたはテンプレートを設計するときに、ルール・デザイナーまたはテンプレート・デザイナーで作成します。ビジネス・ルールおよびテンプレートから独立して存在するスクリプトおよび式コンポーネントとは異なり、メンバー範囲コンポーネントは、システム・ビューから作成できません。メンバー範囲コンポーネントは、それが属するビジネス・ルールおよびテンプレートにリンクされており、共有できません。

メンバー範囲コンポーネントを作成するには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. 「新規オブジェクト」からメンバー範囲オブジェクトをドラッグし、フロー・チャートにドロップします。

メンバー範囲オブジェクトが接続線の付いた 2 つの円として表示されます。

3. **カスタム・テンプレート・ユーザーのみ:** テンプレート用にメンバー範囲コンポーネントを作成する場合、「設計時プロンプト」タブで、その設計時プロンプトを作成します。[カスタム・テンプレート用の設計時プロンプトの作成](#)を参照してください。
4. 「メンバー範囲」タブで、次のいずれかを行います。
 - 変数を使用してメンバー範囲を定義するには、「**変数セレクト**」を選択します。次に、変数を入力するか選択します。[変数の操作](#)を参照してください。

 **ノート:**

このオプションを選択し、メンバーを変更した場合、次のメッセージが表示されます: このグリッドを編集すると、動的変数のリンクが削除されます。続行しますか? 変数へのリンクを削除する場合は、「はい」をクリックします。「はい」をクリックすると、変数へのリンクが削除され、変数のメンバーとともにメンバー範囲が残ります。変数へのリンクを失いたくない場合は、「いいえ」をクリックします。

このオプションを選択した場合は、「**変数に動的にリンク**」を選択します。これを選択すると、変数に変更されたとき、ルールまたはテンプレートの変数にそれらの変更が適用されて更新されます。

- **Oracle Hyperion Planning および Oracle Essbase ユーザーのみ:**
 - a. メンバー範囲を定義する各ディメンションの「値」列にメンバーを入力するか、各行をクリックして「**アクション**」アイコンを表示します。

 **ヒント:**

1 行ごとに各ディメンションのメンバーを選択するよりは、「メンバー・セレクタ」アイコンをクリックし、グリッド内のすべてのディメンションのメンバーを選択できます。メンバー・セレクタで「OK」をクリックすると、選択したメンバーのグリッドに、選択したメンバーがディメンションごとに表示されます。

- b. 「アクション」アイコンをクリックし、次のいずれかのオプションを選択してメンバー範囲を定義します:
 - 変数(変数の操作を参照してください。)
 - メンバー(コンポーネントへのメンバーおよび関数の追加を参照してください。)
 - 関数(関数の操作を参照してください。)
 - c. 「コメント」アイコンをクリックして、メンバー範囲のコメントを入力します。
 - d. メンバー範囲を定義するディメンションごとにこれらのステップを繰り返します。
 - e. 選択するメンバーを計算から除外するには、「グリッド値を除外」を選択します。
 - f. 「グリッドをリセット」をクリックして、グリッドに入力したメンバー、変数および関数をクリアします。
 - g. 「並列処理を可能にする」を選択して、メンバー範囲コマンドを順次にではなく同時に処理します。(デフォルトでは、シリアル処理が使用されますが、並列処理を使用すると計算を最適化できる場合があります。)次に、テキスト・ボックスで並列処理に使用可能なスレッド数を入力するか、「変数セレクタ」をクリックして、ビジネス・ルール内のメンバー範囲を作成している場合は数値または整数タイプ変数を、テンプレートのメンバー範囲を作成している場合は数値の設計時プロンプトを選択します。
- デフォルトでは、並列処理のスレッド数は 2 です。32 ビットのプラットフォームの場合、1 から 4 の整数を選択します。64 ビットのプラットフォームの場合、1 から 8 の整数を選択します。

- **Oracle Hyperion Financial Management ユーザーのみ:**

- a. 「ディメンション」列の下にある最初の行をクリックし、ディメンションを選択します。
- b. 「値」列にメンバーを入力するか、行内をクリックして「アクション」アイコンを表示します。「アクション」アイコンをクリックし、次のいずれかのオプションを選択してメンバー範囲を定義します:
 - 変数(変数の操作を参照してください。)
 - メンバー(コンポーネントへのメンバーおよび関数の追加を参照してください。)
 - 関数(関数の操作を参照してください。)

- c. 「変数」列に起動変数を入力するか、行内をクリックして「アクション」アイコンを表示します。「アクション」をクリックして起動変数を選択します。変数の操作を参照してください。
 - d. 「コメント」アイコンをクリックして、メンバー範囲のコメントを入力します。
 - e. メンバー範囲を定義するディメンションごとにこれらのステップを繰り返します。
5. **ビジネス・ルール・ユーザーのみ: 「エラーおよび警告」** タブで、スクリプト診断を実行するボタンをクリックします。
 6. **オプション: 「プロパティ」** で、「**使用不可**」を選択してメンバー範囲(およびそれに含まれるコンポーネント)を検証から除外します。ビジネス・ルールまたはテンプレート内のコンポーネントによりビジネス・ルールまたはテンプレートが検証されない場合、トラブルシューティングのためにこれを使用できます。

メンバー範囲に説明、コメントおよびキャプションを入力することもできます。キャプションは、そのコンポーネントが属するルールまたはテンプレートのフロー・チャート内で、コンポーネントの下に表示されます。
 7. **Financial Management ユーザーのみ: 「プロパティ」** で、「**タイマーの使用可能**」を選択して、ビジネス・ルールの起動時にログ・ファイルのメンバー範囲コンポーネントの処理に要した時間を記録します。タイマーは、ルール、ルールセットおよびコンポーネントに対して使用可能にできます。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能になっているルールがあり、そのルールにタイマーが使用可能になっている 3 つのメンバー範囲コンポーネントが含まれている場合、ルールとルール内の各メンバー範囲コンポーネントの処理に要する時間が記録されます。
 8.  をクリックします。

メンバー範囲コンポーネントを開く

メンバー範囲コンポーネントは、そのコンポーネントが使用されているビジネス・ルールまたはテンプレートのフロー・チャートから開きます。式およびスクリプト・コンポーネントとは異なり、システム・ビューからは開けません。

ノート:

メンバー範囲コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

メンバー範囲コンポーネントを開くには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. ルールまたはテンプレートが開いている場合、フロー・チャートのメンバー範囲コンポーネントを選択して、メンバー範囲プロパティを確認します。

メンバー範囲コンポーネントの編集

メンバー範囲コンポーネント、メンバー範囲コンポーネントの処理に要した時間をログ・ファイルに記録するかどうか(Oracle Hyperion Financial Management アプリケーションのみ)、およびメンバー範囲コンポーネントのコメントとキャプションを定義するために使用するディメンションおよびメンバー、または変数を編集できます。

ノート:

メンバー範囲コンポーネントは、Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

メンバー範囲コンポーネントを編集するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、プロパティを編集するメンバー範囲を「**メンバー範囲**」で選択します。メンバー範囲の次のプロパティを編集できます。(メンバー範囲コンポーネントの作成を参照してください。)
 - キャプション、説明およびコメント
 - メンバー範囲を定義するために選択した変数
 - メンバー範囲に含まれるディメンション
 - 各ディメンションのメンバー範囲を定義するメンバー
 - **Financial Management ユーザーのみ**: コンポーネントの処理に要した時間をログ・ファイルに含めるかどうか
3.  をクリックします。

メンバー範囲コンポーネントの削除

メンバー範囲コンポーネントを削除するには、それが属するルールまたはテンプレートから除去します。メンバー範囲コンポーネントは、1つのビジネス・ルールまたはテンプレートでのみ使用できるため、ビジネス・ルールまたはテンプレートから除去するだけで削除できます。

ノート:

メンバー範囲コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

メンバー範囲コンポーネントを削除するには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーで、フロー・チャート内の削除するメンバー範囲コンポーネントを選択します。
3. メンバー範囲を右クリックし、「削除」を選択します。
メンバー範囲を削除すると、データベースから削除されます。

メンバー範囲コンポーネントのコピーおよび貼付け

ルールまたはテンプレートからメンバー範囲コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。メンバー範囲コンポーネントをコピーして、別のメンバー範囲コンポーネントや別のコンポーネント・タイプに貼り付けられません。

ノート:

メンバー範囲コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

ルール・デザイナーまたはテンプレート・デザイナーからメンバー範囲コンポーネントをコピーするには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートでコピーするメンバー範囲コンポーネントを右クリックし、「コピー」を選択します。
3. 次のいずれかのタスクを行います:
 - メンバー範囲コンポーネントを同じビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートのコンポーネントを貼り付ける場所で右クリックし、「貼付け」を選択します。
 - メンバー範囲コンポーネントを異なるビジネス・ルールまたはテンプレートに貼り付けるには、コンポーネントを貼り付けるビジネス・ルールまたはテンプレートを開き、フロー・チャートのコンポーネントを貼り付ける場所で右クリックして「貼付け」を選択します。
4.  をクリックします。

データ範囲コンポーネント(Financial Management ユーザーのみ)

次も参照:

- [データ範囲コンポーネントについて\(Financial Management ユーザーのみ\)](#)
- [データ範囲コンポーネントの作成\(Financial Management ユーザーのみ\)](#)
- [データ範囲コンポーネントを開く\(Financial Management ユーザーのみ\)](#)

- データ範囲コンポーネントの編集(Financial Management ユーザーのみ)
- データ範囲コンポーネントの削除(Financial Management ユーザーのみ)
- データ範囲コンポーネントのコピーおよび貼付け(Financial Management ユーザーのみ)

データ範囲コンポーネントについて(Financial Management ユーザーのみ)

データ範囲コンポーネントは、Oracle Hyperion Financial Management アプリケーションのデータ値の範囲で構成されるループの一種です。データ範囲コンポーネントは共有できないため、ビジネス・ルールまたはテンプレートに新規データ範囲を追加するたびに新規データ範囲を作成する必要があります。

データ範囲コンポーネントの作成(Financial Management ユーザーのみ)

Oracle Hyperion Financial Management ビジネス・ルールまたはテンプレートを設計するとき、ルール・デザイナーまたはテンプレート・デザイナー内からデータ範囲コンポーネントを作成します。スクリプトまたは式コンポーネントとは異なり、データ範囲コンポーネントはそれが属するビジネス・ルールまたはテンプレートとしてのみ存在します。データ範囲コンポーネントは共有できません。

データ範囲コンポーネントを作成するには:

1. **システム・ビュー**で、**Financial Management** アプリケーション・タイプ、アプリケーション、計算タイプの順に展開します。データ範囲をルールとテンプレートのどちらに作成するかに応じて、次のいずれかのタスクを実行します:
 - ルールにデータ範囲を作成するには、「**ルール**」を展開し、ルールを右クリックして「**開く**」を選択します。
 - テンプレートにデータ範囲を作成するには、「**テンプレート**」を展開し、テンプレートを右クリックして「**開く**」を選択します。
2. フロー・チャート内のデータ範囲を作成する場所を決定したら、「**新規オブジェクト**」からデータ範囲オブジェクトをドラッグし、フロー・チャートにドロップします。
フロー・チャートに、データ範囲オブジェクトが接続線の付いた 2 つの丸として表示されます。
3. **カスタム・テンプレート・ユーザーのみ**: テンプレート用にデータ範囲コンポーネントを作成する場合、「**設計時プロンプト**」タブで、その設計時プロンプトを作成します。[カスタム・テンプレート用の設計時プロンプトの作成](#)を参照してください。
4. 「**データ範囲**」タブで、次のいずれかのタスクを行います:
 - **ビジネス・ルール**に対してデータ範囲を作成し、変数を使用してそのデータ範囲を定義する場合([変数の操作](#)を参照)、次のタスクを実行します。
 - a. 「**変数セレクタ**」をクリックし、変数を入力または作成してデータ範囲を定義します。

 **ノート:**

このオプションを選択すると、グリッド内のディメンションに対してデータ範囲を定義できません。

- b. 変数をデータ範囲コンポーネントに動的にリンクするかどうかを選択します。動的にリンクする場合、Oracle Hyperion Calculation Manager の別の場所に変更があるたびに、データ範囲コンポーネントで変数が更新されます。
- c. 「値変数」に、POV を含む置換文字列変数を選択します。(このボタンを使用して変数を選択した場合、グリッドの「値」列でメンバーを入力することと同様になります。)
- d. オプション: 「ディメンションのソート」で、ソートに使用するディメンションを選択します。
- e. オプション: 前のステップでソートに使用するディメンションを選択した場合、「ソート方法」で、ディメンションをソートする方法を選択します:
 - 「昇順」では、ディメンション・メンバーを A から Z の順にソートします。(これはデフォルト設定です。)
 - 「降順」では、ディメンション・メンバーを Z から A の順にソートします。
 - 「タイマーの使用可能」を選択して、ビジネス・ルールまたはテンプレートの計算時にデータ範囲の処理に要した時間がログ・ファイルに含まれるようにします。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能になっているルールがあり、そのルールにタイマーが使用可能になっている 3 つの固定ループ・コンポーネントが含まれている場合、ルールおよびルール内の各固定ループ・コンポーネントの処理に要した時間が記録されます。
 - 「グリッドをリセット」をクリックして、デフォルト設定でグリッドを表示します。
- **カスタム・テンプレート**に対してデータ範囲を作成し、設計時プロンプトを使用してそのデータ範囲を定義する場合、次のようにします:
 - a. 「変数セレクタ」をクリックし、変数を入力または作成します。
 - b. 変数をデータ範囲コンポーネントに動的にリンクするかどうかを選択します。動的にリンクする場合、Calculation Manager の別の場所に変更があるたびに、データ範囲コンポーネントで変数が更新されます。
 - c. 「値変数」に、「データ範囲」タイプの置換変数を選択します。(データ範囲変数には、完全なデータ範囲定義が含まれます。)
 - d. オプション: 「ディメンションのソート」で、ソートに使用するディメンションを選択します。
 - e. オプション: 前のステップでソートに使用するディメンションを選択した場合、「ソート方法」で、ディメンションをソートする方法を選択します:
 - 「昇順」では、ディメンション・メンバーを A から Z の順にソートします。(これはデフォルト設定です。)
 - 「降順」では、ディメンション・メンバーを Z から A の順にソートします。
 - 「タイマーの使用可能」を選択して、ビジネス・ルールまたはテンプレートの計算時にデータ範囲の処理に要した時間がログ・ファイルに含まれるように

します。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能になっているルールがあり、そのルールにタイマーが使用可能になっている 3 つの固定ループ・コンポーネントが含まれている場合、ルールおよびルール内の各固定ループ・コンポーネントの処理に要した時間が記録されます。

- 「**グリッドをリセット**」をクリックして、デフォルト設定でグリッドを表示します。
 - **ビジネス・ルールまたはカスタム・テンプレート**に対してデータ範囲を作成し、変数、メンバーおよび関数を使用してそのデータ範囲を定義する場合は、次のことを実行します:
 - a. データ範囲を定義する対象の最初のディメンションに対して、「**値**」列にメンバーを入力するか、行をクリックして「**アクション**」アイコンを表示します。
 - b. 「**アクション**」アイコンをクリックし、次のオプションのいずれかを選択してデータ範囲を定義します:
 - 変数([変数の操作](#)を参照してください。)
 - メンバー([コンポーネントへのメンバーおよび関数の追加](#)を参照してください。)
 - 関数([関数の操作](#)を参照してください。)
 - c. 「**変数**」列に起動変数を入力するか、行内をクリックして「**アクション**」アイコンを表示します。「**アクション**」アイコンをクリックして、「**変数**」を選択して起動変数を選択します。[変数の操作](#)を参照してください。
-  **ノート:**

シナリオ、年、期間、エンティティ、または値ディメンションに対して起動変数を含められません。
- d. 「**コメント**」アイコンをクリックして、データ範囲のコメントを入力します。
 - e. データ範囲を定義するディメンションごとにこれらのステップを繰り返します。
5. **オプション: 「プロパティ」** タブで、次のいずれかのタスクを実行します。
 - データ範囲コンポーネントをビジネス・ルールまたはテンプレートの検証および計算から除外するには、「**使用不可**」を選択します。
 - データ範囲コンポーネントを識別するキャプションを入力します。キャプションは、そのコンポーネントが属するルールまたはテンプレートのフロー・チャート内で、コンポーネントの下に表示されます。
 - データ範囲コンポーネントの説明とコメントを入力します。
 6.  をクリックします。

データ範囲コンポーネントを開く (Financial Management ユーザーのみ)

データ範囲コンポーネントは、そのコンポーネントが使用されているビジネス・ルールまたはテンプレートのフロー・チャートから開きます。式およびスクリプト・コンポーネントとは異なり、システム・ビューからは開けません。

データ範囲コンポーネントを開くには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「開く」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートで、データ範囲コンポーネントを選択してデータ範囲プロパティを確認します。

データ範囲コンポーネントの編集(Financial Management ユーザーのみ)

データ範囲コンポーネントの定義に使用したディメンションとメンバー、変数または設計時プロンプトを編集したり、データ範囲コンポーネントの処理に要した時間をログ・ファイルに記録するかどうか、およびデータ範囲コンポーネントのコメントおよびキャプションを編集できます。

データ範囲コンポーネントを編集するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「開く」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャート内のデータ範囲を選択して「データ範囲」でそのプロパティを編集します。

[データ範囲コンポーネントの作成\(Financial Management ユーザーのみ\)](#)を参照してください。

3.  をクリックします。

データ範囲コンポーネントの削除(Financial Management ユーザーのみ)

データ範囲コンポーネントを削除するには、それが属するルールまたはテンプレートから除去します。データ範囲コンポーネントは、1つのビジネス・ルールまたはテンプレートでのみ使用できるため、ビジネス・ルールまたはテンプレートから除去するだけで削除できます。

データ範囲コンポーネントを削除するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「開く」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートで、データ範囲コンポーネントを右クリックし、「削除」を選択します。

ビジネス・ルールまたはテンプレートから、データ範囲コンポーネントを削除すると、データベースから削除されます。

3. コンポーネントの削除を確認します。

4.  をクリックします。

データ範囲コンポーネントのコピーおよび貼付け(Financial Management ユーザーのみ)

ルールまたはテンプレートからデータ範囲コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。データ範囲コンポーネントをコピーして、別のデータ範囲コンポーネントや別のコンポーネント・タイプに貼り付けられません。

ルール・デザイナまたはテンプレート・デザイナからデータ範囲コンポーネントをコピーして貼り付けるには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナまたはテンプレート・デザイナで、フロー・チャート内のコピーするデータ範囲コンポーネントを右クリックし、「コピー」を選択します。
3. 次のいずれかのタスクを行います:
 - データ範囲コンポーネントを同じビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートのコンポーネントを貼り付ける場所で右クリックし、「貼付け」を選択します。
 - データ範囲コンポーネントを異なるビジネス・ルールまたはテンプレートに貼り付けるには、データ範囲コンポーネントを貼り付けるビジネス・ルールまたはテンプレートを開き、フロー・チャートのデータ範囲コンポーネントを貼り付ける場所で右クリックして「貼付け」を選択します。

4.  をクリックします。

セル・テキスト範囲コンポーネント(Financial Management ユーザーのみ)

次も参照:

- [セル・テキスト範囲コンポーネントについて\(Financial Management ユーザーのみ\)](#)
- [セル・テキスト範囲コンポーネントの作成\(Financial Management ユーザーのみ\)](#)
- [セル・テキスト範囲コンポーネントを開く\(Financial Management ユーザーのみ\)](#)
- [セル・テキスト範囲コンポーネントの編集\(Financial Management ユーザーのみ\)](#)
- [セル・テキスト範囲コンポーネントの削除\(Financial Management ユーザーのみ\)](#)
- [セル・テキスト範囲コンポーネントのコピーおよび貼付け\(Financial Management ユーザーのみ\)](#)

セル・テキスト範囲コンポーネントについて(Financial Management ユーザーのみ)

セル・テキスト範囲コンポーネントは、Oracle Hyperion Financial Management アプリケーションのセル・テキストの範囲で構成されるループの一種です。セル・テキスト範囲コンポーネントは共有できないため、ビジネス・ルールまたはテンプレートに新規セル・テキスト範囲を追加するたびに新規セル・テキスト範囲を作成する必要があります。

セル・テキスト範囲コンポーネントの作成(Financial Management ユーザーのみ)

Oracle Hyperion Financial Management ビジネス・ルールまたはテンプレートを設計するときに、ルール・デザイナまたはテンプレート・デザイナ内からセル・テキスト範囲コンポーネントを作成します。スクリプトまたは式コンポーネントとは異なり、セル・テキスト範囲コンポーネントはそれが属するビジネス・ルールまたはテンプレートとしてのみ存在します。セル・テキスト範囲コンポーネントは共有できません。

セル・テキスト範囲コンポーネントを作成するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. フロー・チャート内のセル・テキスト範囲を作成する場所を決定したら、「**新規オブジェクト**」からセル・テキスト範囲オブジェクトをドラッグし、フロー・チャートにドロップします。
フロー・チャートに、セル・テキスト範囲オブジェクトが接続線の付いた 2 つの丸として表示されます。
3. **カスタム・テンプレート・ユーザーのみ**: テンプレート用にセル・テキスト範囲コンポーネントを作成する場合、「**設計時プロンプト**」タブで、その設計時プロンプトを作成します。[カスタム・テンプレート用の設計時プロンプトの作成](#)を参照してください。
4. 「**セル・テキスト範囲**」タブで、次のいずれかのタスクを実行します:
 - **ビジネス・ルール**に対してセル・テキスト範囲を作成し、変数を使用してそのセル・テキスト範囲を定義する場合([変数の操作](#)を参照)、次のタスクを実行します。
 - a. 「**変数セレクト**」をクリックし、変数を入力または作成してセル・テキスト範囲を定義します。

ノート:

このオプションを選択すると、グリッド内のディメンションに対してセル・テキスト範囲を定義できません。

- b. 変数をセル・テキスト範囲コンポーネントに動的にリンクするかどうかを選択します。動的にリンクする場合、Oracle Hyperion Calculation Manager の別の場所に変更があるたびに、セル・テキスト範囲コンポーネントで変数が更新されます。
- c. 「**値変数**」に、POV を含む置換文字列変数を選択します。(このボタンを使用して変数を選択した場合、グリッドの「**値**」列でメンバーを入力することと同様になります。)

- d. **オプション: 「ディメンションのソート」** で、ソートに使用するディメンションを選択します。
- e. **オプション:** 前のステップでソートに使用するディメンションを選択した場合、「ソート方法」で、ディメンションをソートする方法を選択します:
 - 「昇順」では、ディメンション・メンバーを A から Z の順にソートします。(これはデフォルト設定です。)
 - 「降順」では、ディメンション・メンバーを Z から A の順にソートします。
- **カスタム・テンプレート** に対してセル・テキスト範囲を作成し、設計時プロンプトを使用してそのセル・テキスト範囲を定義する場合、次のようにします:
 - a. 「**変数セレクタ**」をクリックし、変数を入力または作成します。
 - b. 変数をセル・テキスト範囲コンポーネントに動的にリンクするかどうかを選択します。動的にリンクする場合、**Calculation Manager** の別の場所で変更があるたびに、セル・テキスト範囲コンポーネントで変数が更新されません。
 - c. 「**値変数**」に、「セル・テキスト範囲」タイプの置換変数を選択します。(セル・テキスト範囲変数には、完全なセル・テキスト範囲定義が含まれます。)
 - d. **オプション: 「ディメンションのソート」** で、ソートに使用するディメンションを選択します。
 - e. **オプション:** 前のステップでソートに使用するディメンションを選択した場合、「ソート方法」で、ディメンションをソートする方法を選択します:
 - 「昇順」では、ディメンション・メンバーを A から Z の順にソートします。(これはデフォルト設定です。)
 - 「降順」では、ディメンション・メンバーを Z から A の順にソートします。
- **ビジネス・ルールまたはカスタム・テンプレート** に対してセル・テキスト範囲を作成する際に、変数、メンバーおよび関数を使用してセル・テキスト範囲を定義する場合は、次のことを実行します:
 - a. セル・テキスト範囲を定義する対象の最初のディメンションに対して、「**値**」列にメンバーを入力するか、行をクリックして「**アクション**」アイコンを表示します。
 - b. 「**アクション**」アイコンをクリックし、次のオプションのいずれかを選択してセル・テキスト範囲を定義します:
 - 変数([変数の操作](#)を参照してください。)
 - メンバー([コンポーネントへのメンバーおよび関数の追加](#)を参照してください。)
 - 関数([関数の操作](#)を参照してください。)
 - c. 「**変数**」列に起動変数を入力するか、行内をクリックして「**アクション**」アイコンを表示します。「**アクション**」アイコンをクリックして、「**変数**」を選択して起動変数を選択します。[変数の操作](#)を参照してください。

 ノート:

シナリオ、年、期間、エンティティ、または値ディメンションに対して起動変数を含められません。

- d. 「コメント」アイコンをクリックして、セル・テキスト範囲のコメントを入力します。
 - e. セル・テキスト範囲を定義するグリッド内のディメンションごとにこれらのステップを繰り返します。
 - f. <cell text label>で、「値」列に変数値を入力するか、「アクション」アイコンをクリックして、セル・テキスト・ラベル値として使用する変数を選択します。次に、「変数」に値を入力するか、「アクション」アイコンをクリックして、範囲を定義する変数を選択します。
 - g. <POV>、<index>および<cell text>で、「変数」列に変数を入力するか、「アクション」アイコンをクリックして変数を選択します。
5. 「タイマーの使用可能」を選択して、ビジネス・ルールまたはテンプレートの計算時にセル・テキスト範囲の処理に要した時間がログ・ファイルに含まれるようにします。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されません。
 6. 「グリッドをリセット」をクリックして、デフォルト設定でグリッドを表示します。
 7. オプション: 「プロパティ」タブで、次のいずれかのタスクを実行します。
 - セル・テキスト範囲コンポーネントをビジネス・ルールまたはテンプレートの検証および計算から除外するには、「使用不可」を選択します。
 - セル・テキスト範囲コンポーネントを識別するキャプションを入力します。
キャプションは、そのコンポーネントが属するルールまたはテンプレートのフロー・チャート内で、コンポーネントの下に表示されます。
 - セル・テキスト範囲コンポーネントの説明とコメントを入力します。
 8.  をクリックします。

セル・テキスト範囲コンポーネントを開く (Financial Management ユーザーのみ)

セル・テキスト範囲コンポーネントは、そのコンポーネントが使用されているビジネス・ルールのフロー・チャートまたはテンプレートから開けます。式およびスクリプト・コンポーネントとは異なり、システム・ビューからは開けません。

セル・テキスト範囲コンポーネントを開くには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートで、セル・テキスト範囲コンポーネントを選択してセル・テキスト範囲プロパティを確認します。

セル・テキスト範囲コンポーネントの編集(Financial Management ユーザーのみ)

セル・テキスト範囲コンポーネントの定義に使用したディメンションとメンバー、変数または設計時プロンプトを編集したり、セル・テキスト範囲コンポーネントの処理に要した時間をログ・ファイルに記録するかどうか、およびセル・テキスト範囲コンポーネントのコメントおよびキャプションを編集できます。

セル・テキスト範囲コンポーネントを編集するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートで、セル・テキスト範囲を選択して「**セル・テキスト範囲**」でそのプロパティを編集します。

セル・テキスト範囲コンポーネントの作成(Financial Management ユーザーのみ)を参照してください

3.  をクリックします。

セル・テキスト範囲コンポーネントの削除(Financial Management ユーザーのみ)

セル・テキスト範囲コンポーネントは、1つのビジネス・ルールまたはテンプレートでのみ使用できるため、所属先のビジネス・ルールまたはテンプレートから除去するだけで削除できます。

セル・テキスト範囲コンポーネントを削除するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートで、セル・テキスト範囲コンポーネントを右クリックし、「**削除**」を選択します。

ビジネス・ルールまたはテンプレートから、セル・テキスト範囲コンポーネントを削除すると、データベースから削除されます。

3. コンポーネントの削除を確定し、 をクリックします。

セル・テキスト範囲コンポーネントのコピーおよび貼付け(Financial Management ユーザーのみ)

ルールまたはテンプレートからセル・テキスト範囲コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。セル・テキスト範囲コンポーネントをコピーして、別のセル・テキスト範囲コンポーネントや別のコンポーネント・タイプに貼り付けられません。

ルール・デザイナーまたはテンプレート・デザイナーからセル・テキスト範囲コンポーネントをコピーして貼り付けるには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートで、コピーするセル・テキスト範囲コンポーネントを右クリックし、「コピー」を選択します。
3. 次のいずれかのタスクを行います:
 - セル・テキスト範囲コンポーネントを同じビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートのコンポーネントを貼り付ける場所で右クリックし、「貼付け」を選択します。
 - セル・テキスト範囲コンポーネントを異なるビジネス・ルールまたはテンプレートに貼り付けるには、セル・テキスト範囲コンポーネントを貼り付けるビジネス・ルールまたはテンプレートを開き、フロー・チャートのデータ範囲コンポーネントを貼り付ける場所で右クリックして「貼付け」を選択します。
4.  をクリックします。

固定ループ・コンポーネント

固定ループ・コンポーネントは、メタデータ・メンバーのリストを固定の回数循環するオブジェクトです。

次も参照:

- [固定ループ・コンポーネントについて](#)
- [固定ループ・コンポーネントの作成](#)
- [固定ループ・コンポーネントを開く](#)
- [固定ループ・コンポーネントの編集](#)
- [固定ループ・コンポーネントの削除](#)
- [固定ループ・コンポーネントのコピーおよび貼付け](#)

固定ループ・コンポーネントについて

固定ループ・コンポーネントは、メタデータ・メンバーまたはデータ・レコード(Oracle Hyperion Financial Management アプリケーションのみ)のリストを固定回数サイクル・スルーするオブジェクトです。たとえば、勘定科目リストを 10 回ループする固定ループを作成できます。

ノート:

固定ループ・コンポーネントは、Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

固定ループ・コンポーネントの作成

固定ループ・コンポーネントをビジネス・ルールまたはテンプレート内で作成するには、そのアイコンをドラッグしてルール・デザイナーまたはテンプレート・デザイナーのフロー・チャ

ートにドロップします。固定ループ・コンポーネントは、そのために作成したビジネス・ルールまたはテンプレート内のみ存在します。固定ループ・コンポーネントはビジネス・ルールまたはテンプレート間で共有できません。

 **ノート:**

固定ループ・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

固定ループ・コンポーネントを作成するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. ビジネス・ルールまたはテンプレートのフロー・チャート内で固定ループを作成する場所を決定したら、「**新規オブジェクト**」から、「**固定ループ**」オブジェクトをドラッグし、フロー・チャートにドロップします。
固定ループは、線でつながった2つの丸として表示されます。
3. **カスタム・テンプレート・ユーザーのみ:** テンプレート用に固定ループ・コンポーネントを作成する場合、「**設計時プロンプト**」タブで、その設計時プロンプトを作成します。[カスタム・テンプレート用の設計時プロンプトの作成](#)を参照してください。
4. 「**固定ループ**」タブで、次のタスクを実行します:
 - a. 「**値**」で、ループがメタデータまたはデータでサイクル・スルーする回数を入力します。または、「**アクション**」アイコンをクリックし、「**変数**」(ビジネス・ルールを操作している場合)、「**DTP**」(カスタム・テンプレートを操作している場合)または「**関数**」(ビジネス・ルールまたはカスタム・テンプレートを操作している場合)を選択して、ループを定義する変数、設計時プロンプトまたは関数を選択します。[変数の操作](#)および[関数の操作](#)を参照してください。
 - b. 「**ブレイク変数**」で、変数を入力するか「**変数**」アイコン(ビジネス・ルールを操作する場合)または「**DTP**」アイコン(カスタム・テンプレートを操作する場合)をクリックして、それぞれ数値変数または設計時プロンプトを選択して固定ループを終了します。ブレイク変数の値は1つにする必要があります。
 - c. 「**エラーおよび警告**」タブで、スクリプト診断を実行するボタンをクリックします。
5. **Financial Management のビジネス・ルールとカスタム・テンプレートのユーザーのみ:** 「**固定ループ**」タブで、次のタスクを実行します:
 - a. 「**値**」で、ループがメタデータまたはデータでサイクル・スルーする回数を入力します。または、「**アクション**」アイコンをクリックし、「**変数**」(ビジネス・ルールを操作している場合)、「**DTP**」(カスタム・テンプレートを操作している場合)または「**関数**」(ビジネス・ルールまたはカスタム・テンプレートを操作している場合)を選択して、ループを定義する変数、設計時プロンプトまたは関数を選択します。[変数の操作](#)および[関数の操作](#)を参照してください。
 - b. 「**変数**」で、変数を入力するか「**変数**」アイコン(ビジネス・ルールを操作する場合)または「**DTP**」アイコン(カスタム・テンプレートを操作する場合)をクリックして、それぞれ数値変数または設計時プロンプトを選択します。

6. **オプション: 「プロパティ」** タブで、次のいずれかのタスクを実行します。
 - 固定ループ・コンポーネントをビジネス・ルールまたはテンプレートの検証および計算から除外するには、「**使用不可**」を選択します。
 - 固定ループ・コンポーネントを識別するキャプションを入力します。キャプションは、そのコンポーネントが属するルールまたはテンプレートのフロー・チャート内で、コンポーネントの下に表示されます。
 - 固定ループ・コンポーネントの説明とコメントを入力します。
7. **Financial Management ユーザーのみ: 「タイマーの使用可能」** を選択して、ビジネス・ルールが起動されたときに、固定ループの処理に要する時間をログ・ファイルに含めるようにします。タイマーは、ルール、ルールセットおよびコンポーネントに対して使用可能にできます。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。たとえば、タイマーが使用可能になっているルールがあり、そのルールにタイマーが使用可能になっている 3 つの固定ループ・コンポーネントが含まれている場合、ルールおよびルール内の各固定ループ・コンポーネントの処理に要した時間が記録されます。
8.  をクリックします。

固定ループ・コンポーネントを開く

固定ループ・コンポーネントは、属するビジネス・ツールまたはテンプレートのフロー・チャート内から開きます。式およびスクリプト・コンポーネントとは異なり、システム・ビューからは開けません。

ノート:

固定ループ・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

固定ループ・コンポーネントを開くには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. ルールまたはテンプレートが開いている場合、フロー・チャートの固定ループ・コンポーネントを選択して、固定ループ・プロパティを確認します。

固定ループ・コンポーネントの編集

固定ループ・コンポーネントに割り当てた値およびそれに対して選択した変数は編集できます。固定ループをビジネス・ルールに対して作成する場合は、キャプションも編集できます。固定ループを Oracle Hyperion Financial Management のビジネス・ルールに対して作成する場合は、固定ループの処理に要した時間をログ・ファイルに記録するかどうかを編集できます。固定ループを Oracle Hyperion Planning または Oracle Essbase のビジネス・ルールに対して作成する場合は、ブレイク変数を編集できます。ブレイク変数は、固定ループを終了する時点を指定するものです。

 ノート:

固定ループ・コンポーネントは、Financial Management、Planning および Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

固定ループ・コンポーネントを編集するには:

1. システム・ビューで固定ループ・コンポーネントを含むルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートで、固定ループを選択して「固定ループ」タブでそのプロパティを編集します。次のいずれかのプロパティを編集できます。(固定ループ・コンポーネントの作成を参照してください。)

- Financial Management、Planning および Essbase のビジネス・ルールおよびテンプレートの固定ループ・コンポーネントの場合: 固定ループに指定した値およびそれに対して選択した変数
- Financial Management、Planning および Essbase のビジネス・ルールおよびテンプレートの固定ループ・コンポーネントの場合: 固定ループが検証と計算で無効にされるかどうか、および固定ループに関するキャプション、説明およびコメント

固定ループが検証と計算で無効にされるかどうか、および固定ループに関するキャプション、説明およびコメント

- Planning および Essbase のビジネス・ルールの固定ループ・コンポーネントの場合: ブレーク変数
- Financial Management ビジネス・ルールの固定ループ・コンポーネントの場合: 固定ループの処理に要した時間をログ・ファイルに記録するかどうか

ルール・デザイナーまたはテンプレート・デザイナーのフロー・チャートで、固定ループを選択して「固定ループ」タブでそのプロパティを編集します。固定ループに指定する値とそれに対して選択する変数、およびブレーク変数を編集できます。(固定ループ・コンポーネントの作成を参照してください。)

3.  をクリックします。

固定ループ・コンポーネントの削除

固定ループ・コンポーネントを削除するには、それが属するルールまたはテンプレートから除去します。固定ループ・コンポーネントは、1つのビジネス・ルールまたはテンプレートでのみ使用できるため、ビジネス・ルールまたはテンプレートから除去するだけで削除できます。

 ノート:

固定ループ・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

固定ループ・コンポーネントを削除するには:

1. **システム・ビュー**で削除する固定ループ・コンポーネントを含むルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャート内の削除する固定ループ・コンポーネントを選択します。
3. 固定ループ・コンポーネントを右クリックし、「**削除**」を選択します。
4. コンポーネントの削除を確定し、 をクリックします。

固定ループ・コンポーネントのコピーおよび貼付け

ルールまたはテンプレートから固定ループ・コンポーネントをコピーして、同じルールまたはテンプレートや異なるルールまたはテンプレートに貼り付けられます。固定ループ・コンポーネントをコピーして、別の固定ループ・コンポーネントや別のコンポーネント・タイプに貼り付けられません。

 ノート:

固定ループ・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとカスタム・テンプレートで使用できます。

ルール・デザイナーまたはテンプレート・デザイナーから固定ループ・コンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**でコピーする固定ループ・コンポーネントを含むルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャートでコピーする固定ループ・コンポーネントを右クリックし、「**コピー**」を選択します。
3. 次のいずれかのタスクを行います:
 - 固定ループ・コンポーネントを**同じ**ビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートのコンポーネントを貼り付ける場所で右クリックし、「**貼付け**」を選択します。
 - 固定ループ・コンポーネントを**異なる**ビジネス・ルールに貼り付けるには、コンポーネントを貼り付けるビジネス・ルールを開き、フロー・チャートの固定ループ・コンポーネントを貼り付ける場所で右クリックして「**貼付け**」を選択します。
4.  をクリックします。

メタデータ・ループ・コンポーネント

メタデータ・ループ・コンポーネントでは、関数(メタデータ)または開始と終了の値(固定)を使用して複数のメンバーに値を割り当てることができます。

次も参照:

- [メタデータ・ループ・コンポーネントについて](#)
- [メタデータ・ループ・コンポーネントの作成](#)
- [メタデータ・ループ・コンポーネントを開く](#)
- [メタデータ・ループ・コンポーネントの削除](#)
- [メタデータ・ループ・コンポーネントのコピーおよび貼付け](#)

メタデータ・ループ・コンポーネントについて

ノート:

メタデータ・ループ・コンポーネントは、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージとカスタム・テンプレートで使用できます。

メタデータ・ループ・コンポーネントでは、関数(メタデータ)または開始と終了の値(固定)を使用して複数のメンバーに値を割り当てることができます。

たとえば、#missing を親である"総利益"下のすべての"レベル 0"勘定科目に割り当てるとします。この例では、メタデータ・ループで次のように選択します。

- ディメンション - "勘定科目"
- 関数 - @RELATIVE("GP", 0)
- 変数 - "メンバー"タイプの設計時プロンプト

メタデータ・ループ・コンポーネントの作成

メタデータ・ループ・コンポーネントを作成するには、アイコンをドラッグして、カスタム・テンプレートのフロー・チャートにドロップします。メタデータ・ループ・コンポーネントは、そのコンポーネントを作成するテンプレート内にのみ存在します。メタデータ・ループ・コンポーネントはテンプレート間で共有できません。

メタデータ・ループ・コンポーネントを作成するには:

1. **システム・ビュー**でカスタム・テンプレートを右クリックし、「**開く**」を選択します。
2. **テンプレート・デザイナー**の「**新規オブジェクト**」で、「**メタデータ・ループ**」オブジェクトを選択し、フロー・チャートにドロップします。
3. 「**メタデータ・ループ**」タブで、次のいずれかのループ・タイプを選択します。

- **メタデータ**

「メタデータ」を選択した場合、次の情報を入力します。

- **インデックス DTP** - メタデータ・ループのインデックスとして機能する既存の設計時プロンプトを選択します。メタデータ・ループの開始前に、設計時プロンプトの値はゼロにリセットされ、ループごとに **1** ずつ増分されます。
- **ディメンション** - メタデータ・ループで使用する親を含むディメンションを選択します。
- **関数** - ディメンション・メンバーに変数を適用する方法を指定する関数を選択します。
- **変数** - 変数に使用するメンバー設計時プロンプトを作成します。この設計時プロンプトは式で使用され、Oracle Hyperion Calculation Manager によって設計時プロンプトが指定した関数で作成された各メンバーに置き換えられます。

- **固定**

「固定」を選択した場合、次の情報を入力します。

- **インデックス DTP** - メタデータ・ループのインデックスとして機能する既存のステップの設計時プロンプトを選択します。メタデータ・ループの開始前に、設計時プロンプトの値はゼロにリセットされ、ループごとに **1** ずつ増分されます。
- **インデックスの開始** - ループが開始する値。  をクリックして、開始インデックスとして使用する設計時プロンプトを選択します。
- **インデックスの終了** - ループが終了する値。  をクリックして、終了インデックスとして使用する設計時プロンプトを選択します。

4. メタデータ・ループのプロパティを入力します。

- 検証および計算のために、テンプレートからメタデータ・ループ・コンポーネントを除外するには、「**使用不可**」を選択します。

ビジネス・ルールの検証が行われず、検証エラーの原因を特定する際には、コンポーネントを使用不可にすることが必要な場合があります。コンポーネントを使用不可にすると、フロー・チャートではグレーで表示されます。

- メタデータ・ループの**キャプション**を入力します。

フロー・チャート内のメタデータ・ループ・コンポーネントのアイコンの下にキャプションが表示されます。

- **説明**および**コメント**を入力します。

5.  をクリックします。

メタデータ・ループ・コンポーネントを開く

メタデータ・ループ・コンポーネントは、そのコンポーネントが含まれるカスタム・テンプレートのフロー・チャートから開きます。式コンポーネントやスクリプト・コンポーネントとは異なり、メタデータ・ループ・コンポーネントは**システム・ビュー**から直接開けません。

メタデータ・ループ・コンポーネントを開くには:

1. **システム・ビュー**でカスタム・テンプレートを右クリックし、「**開く**」を選択します。
2. **テンプレート・デザイナー**のフロー・チャートでメタデータ・ループ・コンポーネントを選択し、そのプロパティを表示します。

メタデータ・ループ・コンポーネントの削除

メタデータ・ループ・コンポーネントは、そのコンポーネントが含まれるカスタム・テンプレートから削除します。

メタデータ・ループ・コンポーネントを削除するには:

1. **システム・ビュー**でメタデータ・ループを含むカスタム・テンプレートを右クリックし、「開く」を選択します。
2. **テンプレート・デザイナー**のフロー・チャートで、メタデータ・ループ・コンポーネントを右クリックし、「削除」を選択します。

ノート:

このステップでは、メタデータ・ループのみがテンプレートから削除されます。メタデータ・ループとともに使用するために作成した設計時プロンプトも削除する必要があります。設計時プロンプトを削除するには、「設計時プロンプト」タブで設計時プロンプトを含む行を右クリックして「**行の削除**」を選択するか、設計時プロンプトを含む行を選択して  をクリックします。

メタデータ・ループ・コンポーネントのコピーおよび貼付け

カスタム・テンプレートからメタデータ・ループ・コンポーネントをコピーして、同じテンプレートまたは異なるカスタム・テンプレートに貼り付けられます。

メタデータ・ループ・コンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**でメタデータ・ループ・コンポーネントを含むカスタム・テンプレートを右クリックし、「開く」を選択します。
2. **テンプレート・デザイナー**のフロー・チャートで、コピーするメタデータ・ループ・コンポーネントを右クリックし、「コピー」を選択します。

メタデータ・ループ内の式またはスクリプト・コンポーネントをドラッグした場合、「**グループのコピー**」を選択してメタデータ・ループとメタデータ・ループ内のコンポーネントの両方をコピーします

3. 次のいずれかのタスクを行います:
 - メタデータ・ループ・コンポーネントを同じテンプレートに貼り付けるには、フロー・チャート内の目的の場所で右クリックし、「貼付け」を選択します。
 - メタデータ・ループ・コンポーネントを異なるテンプレートに貼り付けるには、目的のテンプレートを開き、フロー・チャート内の目的の場所で右クリックして「貼付け」を選択します。
4.  をクリックします。

 ノート:

メタデータ・ループで使用されている設計時プロンプトをコピーするには、「**設計時プロンプト**」で、コピーする設計時プロンプトを含む行を右クリックし、「**コピー**」を選択します。

すべての設計時プロンプトをコピーするには、「**グリッドのコピー**」を選択し、設計時プロンプトを貼り付けるテンプレートを開きます。「**設計時プロンプト**」タブでグリッドをクリックして右クリックし、「**貼付け**」を選択します。

DTP 割当てコンポーネント

DTP 割当てコンポーネントを使用して、DTP、メンバー、関数または入力テキストをカスタム・テンプレートの設計時プロンプトに割り当てることができます。

次も参照:

- [DTP 割当てコンポーネントについて](#)
- [DTP 割当てコンポーネントの作成](#)
- [DTP 割当てコンポーネントを開く](#)
- [DTP 割当てコンポーネントの編集](#)
- [DTP 割当てコンポーネントの削除](#)
- [DTP 割当てコンポーネントのコピーおよび貼付け](#)
- [DTP 割当てコンポーネントの設計時プロンプト関数の使用](#)

DTP 割当てコンポーネントについて

カスタム・テンプレートで DTP 割当てコンポーネントを使用して、DTP、メンバー、関数または入力テキストを設計時プロンプトに割り当てることができます。DTP 割当てコンポーネントを使用すると、テンプレートのフロー・チャートの複雑さが軽減され、テンプレートのロジックを開発および保守しやすくなります。

 ノート:

DTP 割当ては、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・カスタム・テンプレートで使用できます。

DTP 割当てコンポーネントの作成

テンプレート内で DTP 割当てコンポーネントを作成するには、アイコンをドラッグしてテンプレート・デザイナー・フロー・チャートにドロップします。

DTP 割当てコンポーネントを作成する場合、次の点に注意してください。

- DTP 割当てコンポーネントは、そのコンポーネントを作成するテンプレート内のみ存在します。
- DTP コンポーネントは、メンバー範囲、メンバー・ブロックまたは DTP 条件を使用しない条件コンポーネント内を除き、フロー・チャートの任意の場所に配置できます。

DTP 割当てコンポーネントを作成するには:

1. **システム・ビュー**でカスタム・テンプレートを開きます。
2. **テンプレート・デザイナー**の「**新規オブジェクト**」で、「**DTP 割当て**」オブジェクトを選択し、フロー・チャートにドロップします。
3. **オプション**: コンポーネントの設計時プロンプトを入力します。[カスタム・テンプレート用の設計時プロンプトの作成](#)を参照してください。
4. **Oracle Hyperion Financial Management ユーザーのみ**: 「**タイマーの使用可能**」を選択して、テンプレートを含むビジネス・ルールが起動されたときに、DTP 割当ての処理に要する時間をログ・ファイルに含めるようにします。処理時間は、タイマーが使用可能になっているオブジェクトごとにログ・ファイルに追加されます。

たとえば、タイマーが使用可能なルールがあり、タイマーが使用可能なカスタム・テンプレートがそのルールに含まれ、タイマーが使用可能な 3 つの DTP 割当てコンポーネントがそのテンプレートに含まれているとします。ルール、カスタム・テンプレート、およびテンプレートに含まれる各 DTP 割当ての処理にかかる時間が記録されます。

5. **Financial Management ユーザーのみ**: テンプレートを含んでいるビジネス・ルールにログ・テキストがある場合、ルールの起動時にログ・ファイルにそのテキストが記録されるようにするには、「**ロギングの使用可能**」を選択します。ルール、ルールセットおよびコンポーネントのロギングを使用可能にできます。ただし、テンプレートの式の文で、記録するログ・テキストを指定します。
6. 「**式**」タブで、テンプレートの設計時プロンプトの条件を定義します。
 - 等号の左側のドロップダウンから、値を定義する設計時プロンプトを選択します。
ドロップダウンに設計時プロンプトがない場合は、作業中のテンプレートにプロンプトが作成されていません。
 - 等号の右側のテキスト・ボックスに値を入力し、をクリックして変数、メンバーまたは関数を選択します。
 -  をクリックして、追加の設計時プロンプトに値を割り当てます。
7. DTP 割当てコンポーネントのプロパティを入力します。
 - 検証および計算のために、テンプレートからメタデータ・ループ・コンポーネントを除外するには、「**使用不可**」を選択します。

ビジネス・ルールの検証が行われず、検証エラーの原因を特定する際には、コンポーネントを使用不可にすることが必要な場合があります。コンポーネントを使用不可にすると、フロー・チャートではグレーで表示されます。
 - DTP 割当てコンポーネントの**キャプション**を入力します。

フロー・チャート内の DTP 割当てコンポーネントのアイコンの下にキャプションが表示されます

- **説明**および**コメント**を入力します。
8.  をクリックします。

DTP 割当てコンポーネントを開く

DTP 割当てコンポーネントは、属するテンプレートのフロー・チャート内から開きます。式コンポーネントやスクリプト・コンポーネントとは異なり、DTP 割当てコンポーネントは**システム・ビュー**から直接開けません。

DTP 割当てコンポーネントを開くには:

1. **システム・ビュー**でカスタム・テンプレートを右クリックし、「**開く**」を選択します。
2. **テンプレート・デザイナー**のフロー・チャートで DTP 割当てコンポーネントを選択し、そのプロパティを表示します。

DTP 割当てコンポーネントの編集

DTP 割当てコンポーネントの説明、キャプションおよびコメントを編集できます。DTP 割当てコンポーネントを使用するテンプレートの設計時プロンプトに割り当てる値も編集できます。

DTP 割当てコンポーネントを編集するには:

1. **システム・ビュー**で編集する DTP 割当てコンポーネントを含むテンプレートを右クリックし、「**開く**」を選択します。
2. **テンプレート・デザイナー**のフロー・チャートで DTP 割当てコンポーネントを選択します。
3. 必要に応じて変更を加え、 をクリックします。

DTP 割当てコンポーネントの削除

DTP 割当てコンポーネントが属するテンプレート内から DTP 割当てコンポーネントを削除します。

DTP 割当てコンポーネントを削除するには:

1. **システム・ビュー**で DTP 割当てコンポーネントを含むカスタム・テンプレートを右クリックし、「**開く**」を選択します。
2. **テンプレート・デザイナー**のフロー・チャートで、DTP 割当てコンポーネントを右クリックし、「**削除**」を選択します。

DTP 割当てコンポーネントのコピーおよび貼付け

テンプレートから DTP 割当てコンポーネントをコピーして、同じテンプレートまたは異なるテンプレートに貼り付けられます。

DTP 割当てコンポーネントをコピーして貼り付けるには

1. **システム・ビュー**で DTP 割当てコンポーネントを含むカスタム・テンプレートを右クリックし、「**開く**」を選択します。

2. **テンプレート・デザイナー**のフロー・チャートで、コピーする DTP 割当てコンポーネントを右クリックし、「**コピー**」を選択します。
3. 次のいずれかのタスクを行います:
 - DTP 割当てコンポーネントを同じテンプレートに貼り付けるには、フロー・チャート内の目的の場所で右クリックし、「**貼付け**」を選択します。
 - DTP 割当てコンポーネントを異なるテンプレートに貼り付けるには、目的のテンプレートを開き、フロー・チャート内の目的の場所で右クリックして「**貼付け**」を選択します。

DTP 割当てを新しいテンプレートに貼り付ける場合は、DTP 割当てコンポーネントで使用される設計時プロンプトを新しいテンプレートに作成またはコピーする必要があります。
4.  をクリックします。

DTP 割当てコンポーネントの設計時プロンプト関数の使用

次も参照:

- [設計時プロンプト関数について](#)
- [@Compare](#)
- [@Compliment](#)
- [@Concat](#)
- [@DenseMember](#)
- [@DimMember](#)
- [@DimName](#)
- [@DimType](#)
- [@EndsWith](#)
- [@Intersect](#)
- [@Member](#)
- [@Notin](#)
- [@Plandim](#)
- [@Quote](#)
- [@RemoveQuote](#)
- [@ReplaceAll](#)
- [@ReplaceFirst](#)
- [@SmartListFromIndex](#)
- [@SmartListFromValue](#)
- [@SparseMember](#)
- [@StartsWith](#)
- [@SubString](#)

- @ToLowerCase
- @ToUpperCase
- @Trim
- @Union

設計時プロンプト関数について

カスタム・テンプレートの DTP 割当てコンポーネントで設計時プロンプト関数を使用できます。これらの関数では、メンバーと別のメンバーの比較、2 つの設計時プロンプト間の共通または共通でないメンバーのリストの取得、文字列での引用符の追加または削除、文字列内の文字の大文字または小文字への変換などのタスクを実行できます。

@Compare

@COMPARE は、メンバーと別のメンバー、ディメンションと別のディメンション、パスワードと別のパスワード、または文字列と制限リストを比較し、"true"または"false"を返します。(制限リストの場合、@COMPARE は、代替値ではなくルール・ビルダー値を使用して比較します。)

DTP タイプ:

@COMPARE は、ブール・タイプの DTP に割り当てることができます。

構文:

```
@COMPARE (String,compareString,IgnoreCase)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト
<i>compareString</i>	設計時プロンプトまたはテキスト
<i>IgnoreCase</i>	オプション。 "true"の場合、パラメータ 1 および 2 の大文字/小文字は無視されます。"false"の場合、パラメータ 1 および 2 の大文字/小文字は一致する必要があります。IgnoreCase が空の場合、デフォルトの"true"に設定されます。

例:

```
[DTP1]=@COMPARE ([MBR1], [MBR2], true)
```

ここで:

- [DTP1]は、ブール・タイプのプロンプト不可能な DTP です
- [MBR1]=Jan
- [MBR2]=Jan

この例では、[DTP1]は"true"を返します。

@Compliment

@COMPLIMENT は、DTP2 がない DTP1 内のメンバーと、DTP1 がない DTP2 内のメンバーを返します。つまり、@COMPLIMENT は DTP1 と DTP2 に共通でないものを返します。

DTP タイプ:

@COMPLIMENT は、メンバー範囲またはディメンション間タイプの DTP に割り当てることができます。

構文:

```
@COMPLIMENT (Argument1,Argument2)
```

パラメータ:

パラメータ	説明
Argument1	設計時プロンプト、またはメンバー範囲またはディメンション間を返す関数
Argument2	設計時プロンプト、またはメンバー範囲またはディメンション間を返す関数

例 1:

```
[DTP_MR]=@COMPLIMENT ([MR1], [MR2])
```

ここで:

- [DTP_MR] は、チェック済プロンプトでないメンバー範囲タイプの DTP です。
- [MR1]="Salaries"、"FY14"、"Local"、"Tennessee"、"USA"
- [MR2]="Jan"、"Feb"、"Mar"、"FY15"、"Actual"、"Budget"、"Working"、"Florida"、"California"、"000"、"G100"

この例では、DTP_MR は、"Salaries"、"Jan"、"Feb"、"Mar"、"Actual"、"Budget"、"Working"、"Local"、"000"、"G100"、"USA"を返します

例 2:

```
[DTP_CD]=@COMPLIMENT ([CD1], [CD2])
```

ここで:

- [DTP_CD] は、チェック済プロンプトでないディメンション間タイプの DTP です。
- [CD1]="Salaries"->"Jan"->"Actual"->"Local"
- [CD2]="50100"->"Feb"->"Budget"->"Working"->"California"->"000"

この例では、[DTP_CD] は"Working"->"Local"->"California"->"000"を返します

@Concat

@CONCAT は、最初のパラメータの末尾に 2 番目のパラメータを連結します。

DTP タイプ:

@CONCAT は、メンバー、ディメンション、文字列、制限リスト、パスワードまたは属性タイプの DTP に割り当てることができます。

構文:

```
@CONCAT(Text, Concat String, Seperator)
```

パラメータ:

パラメータ	説明
<i>Text</i>	設計時プロンプトまたはテキスト
<i>Concat String</i>	設計時プロンプトまたはテキスト ノート: <i>concatString</i> は <i>String</i> の末尾に追加されます
<i>Seperator</i>	オプション: 区切り文字を使用する場合、区切り文字は <i>Text</i> に値がある場合にのみ <i>Text</i> と <i>Concat String</i> の間に追加されます。

例:

```
[DTP_CTmbr]=@CONCAT([MBR1],[MBR2])
```

ここで:

- [DTP_CTmbr] は、チェック済プロンプトでないメンバー・タイプの DTP です。
- [MBR1]="California"
- [MBR2]="Washington"

この例では、[DTP_CTmbr] は "CaliforniaWashington" を返します。

@DenseMember

@DENSEMEMBER は、ディメンション間内の最初の密ディメンション・メンバーを戻します。

DTP タイプ:

@DENSEMEMBER は、メンバー、メンバー(複数)、ディメンション、ディメンション(複数)、メンバー範囲、ディメンション間または文字列タイプの DTP に割り当てることができます。

構文:

```
@DENSEMEMBER(Members)
```

パラメータ:

パラメータ	説明
<i>Members</i>	設計時プロンプト、メンバー、またはディメンション間またはメンバー範囲を返す関数

例:

```
[DenseMbr]=@DENSEMEMBER([CD1])
```

ここで:

- [DenseMbr] は、チェック済プロンプトでないディメンション間タイプの DTP です。

- [CD1] は、プロンプト可能なディメンション間タイプの DTP です。
- [CD1]="Salaries"->"Jan"->"Working"->"Tennessee"->"G401"

この例では、"Salaries"がディメンション間リストで最初の密メンバーであるため、[Densembr]は"Salaries"を返します。

@DimMember

@DIMMEMBER は、指定されたディメンションに対してメンバーが有効な場合、メンバーを返します。

DTP タイプ:

@DIMMEMBER は、メンバー、メンバー(複数)、ディメンション間、メンバー範囲、ディメンションまたはディメンション(複数)タイプの DTP に割り当てることができます。

構文:

```
@DIMMEMBER (Dimension, Member)
```

パラメータ:

パラメータ	説明
<i>Dimension</i>	設計時プロンプトまたはディメンション
<i>Member</i>	設計時プロンプト、メンバーまたはアプリケーション内のディメンションのリストが表示され、そこから選択することができる関数ドロップダウン

例:

```
[DIM_mbr1]=@DIMMEMBER ([DIM], [Member])
```

ここで:

- [DIM_mbr1] は、チェック済プロンプトでないメンバー・タイプの DTP です。
- [DIM] は、チェック済プロンプトであるディメンション・タイプの DTP です。
- [Member] は、チェック済プロンプトであるメンバー・タイプの DTP です。
- [Dim]="Entity"
- [Member]="Washington"

この例では、Washington は Entity ディメンションのメンバーであるため、[DIM_mbr1] は"Washington"を返します。

@DimName

@DIMNAME は、ディメンションの名前がデータベースに対して有効な場合、ディメンションの名前を返します。

DTP タイプ:

@DIMNAME は、ディメンション・タイプの DTP に割り当てることができます。

構文:

@DIMNAME (*Dimension*)

パラメータ:

パラメータ	説明
<i>Dimension</i>	設計時プロンプト、ディメンションまたは入力テキスト。 ディメンションをクリックして使用可能なディメンションのリストを表示します。

例:

[DIMN_Product]=@DIMNAME(["Product"])

ここで:

- [DIMN_Product]は、チェック済プロンプトでないディメンション・タイプの DTP です。
- "Product"は関数に入力されます。

"Product"がこのプラン・タイプに対して有効なディメンション名の場合、[DIMN_Product]は "Product"を返します。

"Product"がこのプラン・タイプに対して有効なディメンションでない場合、[DIMN_Product] は"空"で返されます。

@DimType

@DIMTYPE は、指定されたタイプに一致するディメンションを戻します。

DTP タイプ:

@DIMTYPE は、ディメンション、ディメンション(*複数*)または文字列タイプの DTP に割り当てることができます。

構文:

@DIMTYPE (*Dimension Type*)

パラメータ:

パラメータ	説明
<i>Dimension Type</i>	アプリケーションで使用されるディメンション・タイプ

例:

[Dims]=@DIMTYPE (Account)

ここで:

- [Dims]は、チェック済プロンプトでないディメンション・タイプの DTP です。
- [Account]は、ドロップダウンから選択されます。

この例では、[Dims]は"Account"を返します。Account デイメンションがアプリケーションでメジャーに指定されていた場合、"メジャー"が返されます。

@EndsWith

@ENDSWITH は、文字列が指定された接尾辞で終わるかどうかをテストします。

DTP タイプ:

@ENDSWITH は、ブール・タイプの DTP に割り当てることができます。

構文:

```
@ENDSWITH(String,Suffix)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト
<i>Suffix</i>	設計時プロンプトまたはテキスト

例:

```
[DTP_end]=@ENDSWITH([Mbr1],[String1])
```

ここで:

- [Mbr1]は、チェック済プロンプトであるメンバー・タイプの DTP です。
- String1 は、チェック済プロンプトである文字列タイプの DTP です。

次の値が入力だとします。

- [Mbr1]: "Joe Smith"
- [String1]: "Smith"

この例では、[DTP_end]は"true"を返します。

@Intersect

@INTERSECT は、共通のデイメンションのメンバーを戻します。

DTP タイプ:

@INTERSECT は、メンバー範囲またはデイメンション間タイプの DTP に割り当てることができます。

構文:

```
@INTERSECT(Argument1,Argument2)
```

パラメータ:

パラメータ	説明
<i>Argument1</i>	設計時プロンプト、関数またはテキスト
<i>Argument2</i>	設計時プロンプト、関数またはテキスト

例:

```
[IntersectMR]=@INTERSECT([MR1],[MR2])
```

ここで:

- [MR1]は、チェック済プロンプトであるメンバー範囲タイプの DTP です。
- [MR2]は、チェック済プロンプトであるメンバー範囲タイプの DTP です。

MR1 の入力は次のとおりとします。

- 勘定科目: "Salaries"
- バージョン: "Target"
- エンティティ: "Tennessee"、"Florida"

MR2 の入力は次のとおりです。

- 勘定科目: "50590"、"50350"
- 年: "FY10"
- シナリオ: "Budget"
- バージョン: "First Pass"

この例では、[IntersectMR]は"50590"、"50350"、"Salaries"、"First Pass"、"Target"を返します。"エンティティ"のメンバーは MR1 にのみ入力され、"シナリオ"のメンバーは MR2 にのみ入力されているため、これらのメンバーは[IntersectMR]には含まれません。

@Member

@MEMBER は、有効なメンバーであるかぎり、メンバーを戻します。

DTP タイプ:

@MEMBER は、メンバー・タイプの DTP に割り当てることができます。

構文:

```
@MEMBER (Member)
```

パラメータ:

パラメータ	説明
<i>Member</i>	設計時プロンプト、メンバー、または別の設計時プロンプト関数

例:

```
[Mbr_mr1]=@UNION([MR1],@MEMBER([Member]))
```

ここで:

- [Mbr_mr1]は、チェック済プロンプトでないメンバー範囲タイプの DTP です。
- [@UNION]は、入力されたメンバーが有効なメンバーである場合、DTP、MR1 および DTP メンバーの範囲を作成します。
- [MR1]は、チェック済プロンプトであるメンバー範囲タイプの DTP です。
- [Member]は、チェック済プロンプトであるメンバー・タイプの DTP です。

次の値が入力だとします。

- [MR1]: 勘定科目"Salaries"、シナリオ"Actual"、エンティティ "Tennessee"
- [Member]: "000"。これは製品ディメンションのメンバーです

この例では、[Mbr_mr1]は"Salaries"、"Actual"、"Tennessee"、"000"を返します

@Notin

@NOTIN は、一方の式には含まれるが他方の式には含まれないメンバーを返します。つまり、@NOTIN は式間に共通でないものを返します。

DTP タイプ:

@NOTIN は、メンバー範囲またはディメンション間タイプの DTP に割り当てることができます。

構文:

```
@NOTIN (Argument1,Argument2)
```

パラメータ:

パラメータ	説明
Argument1	設計時プロンプト、関数またはメンバー
Argument2	設計時プロンプト、関数またはメンバー

例:

```
[NOTIN1]=@NOTIN([MR1],[MR2])
```

ここで:

- [NOTIN1]は、チェック済プロンプトでないメンバー範囲タイプの DTP です。
- [MR1]は、チェック済プロンプトであるメンバー範囲タイプの DTP です。
- [MR2]は、チェック済プロンプトであるメンバー範囲タイプの DTP です。

MR1 の入力は次のとおりとします。

- 期間: "Jan"、"Feb"、"Mar"

- シナリオ: "Actual"
- 通貨: "Local"
- 製品: "000"

MR2 の入力は次のとおりとします。

- 期間: "Feb"
- バージョン: "Working"
- 通貨: "Local"
- 製品: "000"

この例では、MR2 ではなく、MR1 にあるメンバーは **Actual** のみであるため、[@NOTIN1]は "Actual"を返します。

@Plandim

@PLANDIM は、アプリケーションのデータベース内にディメンション名が存在する場合、そのディメンション名を返します。

DTP タイプ:

@PLANDIM は、メンバーまたはディメンション・タイプの DTP に割り当てることができます。
@PLANDIM は、第 2 引数に 1 つの値のみが使用されている場合に機能します

構文:

```
@PLANDIM(Database Name,Dimension)
```

パラメータ:

パラメータ	説明
<i>Database Name</i>	設計時プロンプトまたはテキスト
<i>Dimension</i>	設計時プロンプト、ディメンションまたはテキスト

例:

```
[PDIM_mbr]=@PLANDIM([PlTy],[Dim])
```

ここで:

- [PDIM_mbr] は、チェック済プロンプトでないメンバー・タイプの DTP です。
- [PlTy] は、チェック済プロンプトである文字列タイプの DTP です。
- [Dim] は、チェック済プロンプトであるディメンション・タイプの DTP です。

次の値が入力だとします。

- [PlTy]: "Plan1"
- Dim: "Product"

この例では、**Product** は"Plan1"内の有効なディメンションであるため、[PDIM_mbr]は"Product"を返します。

@Quote

@QUOTE は、文字列の前後に二重引用符を追加します。

DTP タイプ:

@QUOTE は、メンバー、文字列またはパスワード・タイプの DTP に割り当てることができます。

構文:

```
@QUOTE (String)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト

例:

```
[Quote_St]=@QUOTE([String1])
```

ここで:

- [Quote_St] は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1] は、チェック済プロンプトである文字列タイプの DTP です。

[String1] の入力が "Yellow" とします。

この例では、[Quote_St] は "Yellow" を返します。

@RemoveQuote

@REMOVEQUOTE は、文字列から二重引用符を除去します。

DTP タイプ:

@REMOVEQUOTE は、メンバー、文字列またはパスワード・タイプの DTP に割り当てることができます。

構文:

```
@REMOVEQUOTE (String)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト

例:

```
[RemoveQuote_St]=@REMOVEQUOTE([String1])
```

ここで:

- [RemoveQuote_St]は文字列タイプの DTP で、チェック済プロンプトではありません。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。

[String1]の入力が"Yellow"とします。

この例では、[RemoveQuote_St]は"Yellow"を返します。

@ReplaceAll

@REPLACEALL は、式を含む文字列の一部を置換文字列に置き換えます。

DTP タイプ:

@REPLACEALL は、文字列、パスワード、メンバー、メンバー(複数)、ディメンションまたはディメンション(複数)タイプの DTP に割り当てることができます。

構文:

```
@REPLACEALL(String,Regular Expression,replaceString)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト
<i>Regular Expression</i>	設計時プロンプトまたはテキスト
<i>replaceString</i>	設計時プロンプトまたはテキスト

例:

```
[ReplaceAll_Str]=@REPLACEALL([String1],[Rep_String],[new_String])
```

ここで:

- [ReplaceAll_Str]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。
- [Rep_String]は、チェック済プロンプトである文字列タイプの DTP です。
- [new_String]は、チェック済プロンプトである文字列タイプの DTP です。

パラメータが次のとおりであるとします。

- 文字列: "This is a test of a testing string"
- 正規表現: "Test"
- 置換文字列: "XYZ"

この例では、[ReplaceAll_Str]は、"This is a XYZ of a XYZing string for one XYZ and two XYZ"を返します。"test"が"XYZ"に置き換えられました。

@ReplaceFirst

@REPLACEFIRST は、正規表現の最初の出現を置換文字列で置き換えます。

DTP タイプ:

@REPLACEFIRST は、文字列、パスワード、メンバー、メンバー(複数)、ディメンションまたはディメンション(複数)タイプの DTP に割り当てることができます。

構文:

```
@REPLACEFIRST(String,Regular Expression,replaceString)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト
<i>Regular Expression</i>	設計時プロンプトまたはテキスト
<i>replaceString</i>	設計時プロンプトまたはテキスト

例:

```
[ReplaceFirst_Str]=@REPLACEFIRST([String1],[Rep_String],[new_String])
```

ここで:

- [ReplaceFirst_Str]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。
- [Rep_String]は、チェック済プロンプトである文字列タイプの DTP です。
- [new_String]は、チェック済プロンプトである文字列タイプの DTP です。

パラメータが次のとおりであるとします。

- 文字列: "This is a testing of the new test of a test today"
- 正規表現: "Test"
- 置換文字列: "XYZ"

この例では、[ReplaceFirst_Str]は、"This is a XYZing of the new test of a test today"を返します。test のすべての出現ではなく、最初の出現のみ置換されます。

@SmartListFromIndex

@SMARTLISTFROMINDEX は、スマート・リスト内のメンバーの順序に基づいてスマート・リスト・メンバーを返します。

DTP タイプ:

@SMARTLISTFROMINDEX は、文字列、パスワード、メンバー、メンバー(複数)タイプの DTP に割り当てることができます。

構文:

```
@SMARTLISTFROMINDEX (SmartList, Index)
```

パラメータ:

パラメータ	説明
<i>SmartList</i>	設計時プロンプトまたはディメンション。ディメンションには、アプリケーション内のスマート・リスト・ディメンションが示されます。
<i>Index</i>	設計時プロンプトまたはテキスト

例:

```
[sl_ind_Str]=@SmartListFromIndex([String1],[index_ST])
```

ここで:

- [sl_ind_Str]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。
- [index_ST]は、チェック済プロンプトである数値タイプの DTP です。

パラメータが次のとおりであるとします。

- スマート・リスト: "CreditRatings"
- インデックス: "2"

この例では、スマート・リスト"CreditRatings"には次のデータが含まれます。

名前	ラベル
AA	ID_ENUMNETLABEL_AA
AAA	ID_ENUMNETLABEL_AAA
AAMinus	ID_ENUMNETLABEL_AA22
A	ID_ENUMNETLABEL_A
AMinus	ID_ENUMNETLABEL_AA22
APlus	ID_ENUMNETLABEL_AA11
APlus	ID_ENUMNETLABEL_A11
B	ID_ENUMNETLABEL_B
BPlus	ID_ENUMNETLABEL_B11
BMinus	ID_ENUMNETLABEL_B22
BB	ID_ENUMNETLABEL_BB
BBPlus	ID_ENUMNETLABEL_BB11
BBMinus	ID_ENUMNETLABEL_BB22
BBB	ID_ENUMNETLABEL_BBB
AAAMinus	ID_ENUMNETLABEL_AAA-
BBBPlus	ID_ENUMNETLABEL_BBB+

名前	ラベル
BBBMinus	ID_ENUMNETLABEL_BBB-
CCCPlus	ID_ENUMNETLABEL_CCC+
CCC	ID_ENUMNETLABEL_CCC

2 番目のインデックス位置にあるスマート・リスト・メンバーの名前は **AAMinus** であるため、[sl_ind_Str]は"AAMinus"を返します。

@SmartListFromValue

@SMARTLISTFROMVALUE は、スマート・リスト内のメンバーの ID に基づいてスマート・リスト・メンバーを返します。

DTP タイプ:

@SMARTLISTFROMVALUE は、1 つのメンバーが入力された場合のみ文字列、パスワード、メンバー、メンバー(複数)タイプの DTP に割り当てることができます。

構文:

```
@SMARTLISTFROMVALUE (SmartList, Value)
```

パラメータ:

パラメータ	説明
Smartlist	設計時プロンプトまたはディメンション。ディメンションには、アプリケーション内のスマート・リスト・ディメンションが示されます。
Value	設計時プロンプトまたは数値

例:

```
[sl_val_Str]=@SmartListFromValue ([String1], [value_ST])
```

ここで:

- [sl_val_Str]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。
- [value_ST]は、チェック済プロンプトである数値タイプの DTP です。

パラメータが次のとおりであるとします。

- スマート・リスト: "CreditRatings"
- 値: "5"

この例では、スマート・リスト"CreditRatings"には次のデータが含まれます。

名前	ラベル
AA	ID_ENUMNETLABEL_AA
AAA	ID_ENUMNETLABEL_AAA
AAMinus	ID_ENUMNETLABEL_AA22

名前	ラベル
A	ID_ENUMNETLABEL_A
AMinus	ID_ENUMNETLABEL_A22
AAPlus	ID_ENUMNETLABEL_AA11
APlus	ID_ENUMNETLABEL_A11
B	ID_ENUMNETLABEL_B
BPlus	ID_ENUMNETLABEL_B11
BMinus	ID_ENUMNETLABEL_B22
BB	ID_ENUMNETLABEL_BB
BBPlus	ID_ENUMNETLABEL_BB11
BBMinus	ID_ENUMNETLABEL_BB22
BBB	ID_ENUMNETLABEL_BBB
AAAMinus	ID_ENUMNETLABEL_AAA-
BBBPlus	ID_ENUMNETLABEL_BBB+
BBBMinus	ID_ENUMNETLABEL_BBB-
CCCPlus	ID_ENUMNETLABEL_CCC+
CCC	ID_ENUMNETLABEL_CCC

ID が 5 のスマート・リスト・メンバーの名前は AMinus であるため、[s1_val_str]は"AMinus"を返します。

@SparseMember

@SPARSEMEMBER は、ディメンション間内の最初の疎ディメンション・メンバーを戻します。

DTP タイプ:

@SPARSEMEMBER は、メンバー、メンバー(複数)、ディメンション、ディメンション(複数)、メンバー範囲、ディメンション間または文字列タイプの DTP に割り当てることができます。

構文:

@SPARSEMEMBER (Members)

パラメータ:

パラメータ	説明
Members	設計時プロンプト、メンバーまたはメンバーのリストを返す関数

例:

```
[SparseMbr]=@SPARSEMEMBER ([CD1])
```

ここで:

- [SparseMbr]は、チェック済プロンプトでないディメンション間タイプの DTP です。
- [CD1]は、プロンプト可能なディメンション間タイプの DTP です。
- [CD1]="Gross Sales"->"Jan"->"FY14"->"Working"->"California"

この例では、FY14 がディメンション間リストで最初の疎メンバーであるため、[Sparsembr] は "FY14" を返します。

@StartsWith

@STARTSWITH は、文字列が指定された接頭辞で始まるかどうかをテストします。

DTP タイプ:

@STARTSWITH は、ブール・タイプの DTP に割り当てることができます。

構文:

```
@STARTSWITH (String, Prefix)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト
<i>Prefix</i>	設計時プロンプトまたはテキスト

例:

```
[DTP_Start]=@STARTSWITH([Mbr1],[String1])
```

ここで:

- [Mbr1] は、チェック済プロンプトであるメンバー・タイプの DTP です。
- [String1] は、チェック済プロンプトである文字列タイプの DTP です。

次の値が入力だとします。

- [Mbr1]: "Joe Smith"
- [String1]: "Smith"

この例では、[DTP_Start] は "false" を返します。

@SubString

@SUBSTRING は、先頭インデックスの位置にある文字から開始して最終インデックスの直前の文字まで延長した部分文字列を戻します。先頭インデックスはゼロから始まり、返されるテキストに終了インデックスは含まれません。最終インデックスが文字列内の文字数を超える場合、何も戻されません。

DTP タイプ:

@SUBSTRING は、メンバー、メンバー(複数)、ディメンション、ディメンション(複数)、文字列またはパスワード・タイプの DTP に割り当てることができます。

構文:

```
@SUBSTRING (String, Beginning Index, Ending Index)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト
<i>Beginning Index</i>	数値またはテキスト設計時プロンプト
<i>Ending Index</i>	数値またはテキスト設計時プロンプト

例:

```
[SUBSTRING_Str]=@SUBSTRING([String1],[start_ind],[end_ind])
```

ここで:

- [SUBSTRING_Str]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。
- [start_ind]は、チェック済プロンプトである数値タイプの DTP です。
- [end_ind]は、チェック済プロンプトである数値タイプの DTP です。

パラメータが次のとおりであるとします。

- 文字列: "This is a test of testing today."
- 先頭インデックス: "2"
- 最終インデックス: "16"

この例では、文字列の 2 番目の文字は this 内の h で、15 番目の文字は単語 of の先頭にある o であるため、[SUBSTRING_Str]は"this is a test o"を返します。

@ToLowerCase

@TOLOWERCASE は、文字列内の文字を小文字に変更します。

DTP タイプ:

@TOLOWERCASE は、メンバー、メンバー(複数)、ディメンション、ディメンション(複数)、文字列またはパスワード・タイプの DTP に割り当てることができます。

構文:

```
@TOLOWERCASE (String)
```

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト

例:

```
[TOLOWERCASE_Str]=@TOLOWERCASE([String1])
```

ここで:

- [TOLOWERCASE_Str]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。

パラメータが次のとおりであるとして。

文字列: "Testing for Today"

この例では、[TOLOWERCASE_Str]は"Testing for Today"を返します

@ToUpperCase

@TOUPPERCASE は、文字列内の文字を大文字に変更します。

DTP タイプ:

@TOUPPERCASE は、メンバー、メンバー(複数)、ディメンション、ディメンション(複数)、文字列またはパスワード・タイプの DTP に割り当てることができます。

構文:

@TOUPPERCASE (String)

パラメータ:

パラメータ	説明
String	設計時プロンプトまたはテキスト

例:

[TOUPPERCASE_Str]=@TOUPPERCASE([String1])

ここで:

- [TOUPPERCASE_Str]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。

パラメータが次のとおりであるとして。

文字列: "Testing to convert to upper case."

この例では、[TOUPPERCASE_Str]は"TESTING TO CONVERT TO UPPER CASE."を返します

@Trim

@TRIM は、先頭または末尾のホワイト・スペースを削除します。

DTP タイプ:

@TRIM は、メンバー、メンバー(複数)、文字列またはパスワード・タイプの DTP に割り当てることができます。

構文:

@TRIM (String)

パラメータ:

パラメータ	説明
<i>String</i>	設計時プロンプトまたはテキスト

例:

```
[TRIM_ST]=@TRIM([String1])
```

ここで:

- [TRIM_ST]は、チェック済プロンプトでない文字列タイプの DTP です。
- [String1]は、チェック済プロンプトである文字列タイプの DTP です。

パラメータが次のとおりであるとします。

文字列: "This is a test of leading and trailing spaces"

パラメータは、テキストの前後にホワイト・スペースが含まれています。次のように表示されます。

```
String1 =          this is a test of leading and trailing spaces
```

@Union

@UNION は、複数の設計時プロンプトで使用されるメンバーの組合せを戻します。

DTP タイプ:

@UNION は、メンバー範囲またはディメンション間タイプの DTP に割り当てることができます。

構文:

```
@UNION (Argument1,Argument2)
```

パラメータ:

パラメータ	説明
<i>Argument1</i>	設計時プロンプト、関数またはメンバー
<i>Argument2</i>	設計時プロンプト、関数またはメンバー

例:

```
[UnionMR]=@UNION([MR1],[MR2])
```

ここで:

- [UnionMR]は、チェック済プロンプトでないメンバー範囲タイプの DTP です。
- [MR1]は、チェック済プロンプトであるメンバー範囲タイプの DTP です。
- [MR2]は、チェック済プロンプトであるメンバー範囲タイプの DTP です。

MR1 および MR2 の入力は次のとおりとします。

- [MR1]: "50350"、@Relative("Q1", 0)、"California"、"Tennessee"
- [MR2]: "Apr"、"Washington"、"Tennessee"

この例では、UnionMR は次のものを返します。

"50350"、@Relative ("Q1", 0)、"Apr"、"California"、"Tennessee"、"Washington"

スクリプトおよび式コンポーネントの共有

共有された式またはスクリプト・コンポーネントは、式レベルおよびスクリプト・レベルに存在します。

次も参照:

- [スクリプトおよび式コンポーネントの共有について](#)
式およびスクリプト・コンポーネントは、同じアプリケーション・タイプに属している、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとテンプレートで共有できます。
- [式およびスクリプト・コンポーネントの共有から非共有への変更](#)
- [式およびスクリプト・コンポーネントの非共有から共有への変更](#)

スクリプトおよび式コンポーネントの共有について

式およびスクリプト・コンポーネントは、同じアプリケーション・タイプに属している、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとテンプレートで共有できます。

ノート:

メンバー範囲、条件、固定ループ、メンバー・ブロックまたは DTP 割当てコンポーネントは共有できません。

共有された式またはスクリプト・コンポーネントは、式レベルおよびスクリプト・レベルに存在します。共有された式およびスクリプト・コンポーネントは、式またはスクリプトを変更した場合に、共有コンポーネントが使用されているすべてのルールおよびテンプレートにその変更が反映されるようにするために使用されます。コンポーネントを共有すると、元のコンポーネントへの相互参照が作成されます。元のコンポーネントのコピーではなく相互参照を作成することにより、データベースの使用量が減少し、処理時間が短縮される場合があります。

式およびスクリプト・コンポーネントの共有から非共有への変更

共有された式またはスクリプト・コンポーネントを非共有に変更するには、その前に、複数のビジネス・ルールまたはテンプレートで使用されていないことを確認する必要があります。使用状況の表示機能を使用して、式またはスクリプト・コンポーネント

がどのビジネス・ルールおよびテンプレートで使用されているかを確認できます。(式およびスクリプト・コンポーネントの使用の表示を参照してください。)次に、ルールおよびテンプレート内からコンポーネントの「共有」チェック・ボックスをクリアして、使用されている各ビジネス・ルールおよびテンプレートに対して共有コンポーネントのコピーを作成できます。

ノート:

式およびスクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールとテンプレートで使用できます。

式またはスクリプト・コンポーネントを共有から非共有に変更するには:

1. システム・ビューでルールまたはテンプレートを右クリックし、「開く」を選択します。
2. ルール・デザイナーまたはテンプレート・デザイナーで、次のいずれかのタスクを実行します:
 - フロー・チャートに追加する式またはスクリプト・コンポーネントを共有しない場合は、そのコンポーネントをフロー・チャートにドラッグした後、「プロパティ」で「共有」をクリアします。
 - フロー・チャートにすでにある式またはスクリプト・コンポーネントを共有しない場合は、フロー・チャート上でそのコンポーネントを選択し、「プロパティ」で「共有」をクリアします。これによってルールまたはテンプレート内のコンポーネントのコピーが作成されます。
3.  をクリックします。

式およびスクリプト・コンポーネントの非共有から共有への変更

式またはスクリプト・コンポーネントを非共有から共有に変更するには、同じアプリケーション・タイプ内(Oracle Hyperion Financial Management、Oracle Hyperion Planning または Oracle Essbase)に、名前が同じを持つ共有または非共有の式またはスクリプト・コンポーネントがないことを確認する必要があります。共有オブジェクトの名前はアプリケーション間で一意にする必要があるため、すでに使用されている名前を持つ共有オブジェクトは作成できません。

式またはスクリプト・コンポーネントを共有するには、既存のコンポーネントをルールまたはテンプレートのフロー・チャートにドラッグ・アンド・ドロップするときに、その「プロパティ」タブで「共有」チェック・ボックスを選択します。(「共有」チェック・ボックスはデフォルトで選択されています。)

ノート:

式およびスクリプト・コンポーネントは、Financial Management、Planning および Essbase ブロック・ストレージのビジネス・ルールおよびテンプレートで使用できます。

式またはスクリプト・コンポーネントを非共有から共有に変更するには:

1. **システム・ビュー**でルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、次のいずれかのタスクを実行します:
 - 式またはスクリプト・コンポーネントを共有するには:
 - 「**新規オブジェクト**」で、式またはスクリプト・コンポーネントをドラッグして、フロー・チャート内の必要な場所にドロップします。
 - そのコンポーネントの「**プロパティ**」タブで、「**共有**」を選択し、コンポーネントに名前を付けます。
 - すでにフロー・チャートにある式またはスクリプト・コンポーネントを共有するには:
 - a. フロー・チャート内でコンポーネントを選択します。
 - b. そのコンポーネントの「**プロパティ**」タブで、「**共有**」を選択し、コンポーネントに名前を付けます。すべての共有コンポーネントに名前が必要です。
3. 「**保存**」を選択します。

コンポーネントのコピー

ビジネス・ルール・コンポーネントの子をコピーして貼り付けるか、ビジネス・ルールの式またはスクリプト・コンポーネントに参照をコピーして貼り付けます。

次も参照:

- [ビジネス・ルール・コンポーネントの子のコピーおよび貼付け](#)
- [ビジネス・ルールの式またはスクリプト・コンポーネントへの参照のコピーおよび貼付け](#)

ビジネス・ルール・コンポーネントの子のコピーおよび貼付け

コンポーネントがあるビジネス・ルールを操作する場合、コンポーネントの子(すなわち、フロー・チャートの1つのコンポーネントの下にグループ化されたコンポーネント)をコピーできます。コンポーネントの子をコピーして、同じビジネス・ルールまたは異なるビジネス・ルールに貼り付けることができます。

ビジネス・ルール・コンポーネントの子をコピーして貼り付けるには:

1. ビジネス・ルールを開きます。
2. ビジネス・ルールのフロー・チャートで、子をコピーするメンバー範囲、固定ループ、または条件コンポーネントを右クリックし、「**グループのコピー**」を選択します。
3. 次のいずれかのタスクを行います:
 - コンポーネントの子を**同じ**ビジネス・ルールに貼り付けるには、コンポーネントの子をフロー・チャートで表示する位置の左にあるコンポーネントを右クリ

ックし、「貼付け」を選択します。(選択したコンポーネントの右にコンポーネントの子が表示されます。)

- コンポーネントの子を異なるビジネス・ルールに貼り付けるには、コンポーネントの子を貼り付けるビジネス・ルールを開いて、その子をフロー・チャートで表示する位置の左にあるコンポーネントを右クリックし、「貼付け」を選択します。

4.  をクリックします。

ノート:

コンポーネントの子をコピーして貼り付けると、共有されているコンポーネントもコピーされます。

ビジネス・ルールの式またはスクリプト・コンポーネントへの参照のコピーおよび貼付け

ビジネス・ルールの共有スクリプトまたは共有式コンポーネントの参照をコピーして貼り付ける場合、コンポーネント自体はコピーされず、コンポーネントへのショートカットがコピーされて貼り付けられます。コンポーネントのコピーは、参照のコピー元のビジネス・ルールに1つのみ存在します。参照は、共有コンポーネントが含まれるアプリケーションおよびプラン・タイプへのポイントとして機能します。共有されていないコンポーネントに対する参照のコピーを選択すると、貼り付けられるコンポーネントは、コンポーネントのコピーとなり、元のコンポーネントへのショートカットは作成されません。この場合、参照のコピーはコピーと同様に動作します。

共有式または共有スクリプト・コンポーネントへの参照をコピーして、同じビジネス・ルールまたは異なるビジネス・ルールに貼り付けることができます。

ノート:

Oracle Hyperion Financial Management ビジネス・ルールおよびコンポーネントの場合に、コンポーネントへの参照をコピーする際には、同じ計算タイプまたは汎用計算タイプのビジネス・ルールに貼り付ける必要があります。

共有式または共有スクリプト・コンポーネントへの参照をコピーして貼り付けるには:

1. 共有式または共有スクリプト・コンポーネントが含まれるルールを右クリックし、「開く」を選択します。
2. ルール・デザイナのフロー・チャートで共有式または共有スクリプト・コンポーネントを右クリックし、「参照のコピー」を選択します。
3. 次のいずれかのタスクを行います:
 - コンポーネントの参照を同じビジネス・ルールに貼り付けるには、フロー・チャートでそのコンポーネントの参照を表示する位置の左にあるコンポーネントを右クリックします。(選択したコンポーネントの右にコンポーネントの参照が表示されます。)次に、「貼付け」を選択します。

- コンポーネントの参照を異なるビジネス・ルールに貼り付けるには、コンポーネントの参照を貼り付けるビジネス・ルールを開き、フロー・チャートでそのコンポーネントの参照を表示する位置の左にあるコンポーネントを右クリックします。次に、「貼付け」をクリックします。

ノート:

アプリケーションが同じアプリケーション・タイプに属していれば、同じアプリケーションに属するビジネス・ルールにでも、異なるアプリケーションに属するビジネス・ルールにでも、参照をコピーして貼り付けることができます。

4.  をクリックします。

コンポーネントの保存

式およびスクリプト・コンポーネントをコンポーネント・デザイナーで設計してから保存します。

次も参照:

- [コンポーネントの保存](#)
- [異なる名前での式およびスクリプト・コンポーネントの保存](#)

コンポーネントの保存

式およびスクリプト・コンポーネントをコンポーネント・デザイナーで設計してから保存します。条件コンポーネント、範囲コンポーネントおよび固定ループ・コンポーネントが属するビジネス・ルールまたはテンプレートを、ルール・デザイナーまたはテンプレート・デザイナーで保存すると、これらのコンポーネントも保存されます。

Oracle Hyperion Financial Management、Oracle Hyperion Planning または Oracle Essbase のブロック・ストレージ・コンポーネントを操作している場合、そのコンポーネントをカスタム・テンプレートとして保存できます。式またはスクリプト・コンポーネントを扱う場合には、異なる名前でも保存できます。

コンポーネント・デザイナー、テンプレート・デザイナーまたはルール・デザイナーでコンポーネントの設計を終了した後には、 をクリックします。

異なる名前での式およびスクリプト・コンポーネントの保存

「名前を付けて保存」を使用して、スクリプトおよび式コンポーネントに別の名前を付けて保存できます。「名前を付けて保存」では、式またはスクリプト・コンポーネントのコピーが作成されます。共有コンポーネントの場合、コンポーネントのコピーを作成して、コンポーネントを共有させないことも可能です。[スクリプトおよび式コンポーネントの共有](#)を参照してください。

 ノート:

Oracle Hyperion Financial Management、Oracle Hyperion Planning または Oracle Essbase のブロック・ストレージ・ビジネス・ルールに作成した式およびスクリプト・コンポーネントは、別の名前を付けて保存できます。

スクリプトまたは式コンポーネントを異なる名前では保存するには:

1. **システム・ビュー**で式またはスクリプトを右クリックし、「開く」を選択します。
2. コンポーネント・デザイナーで「ファイル」、「名前を付けて保存」の順に選択します。
3. 「名前を付けて保存」に式またはスクリプトの新しい名前を入力し、アプリケーションを選択します。次に、次のいずれかを行います:
 - Financial Management の式またはスクリプトであれば、「計算タイプ」を選択します
 - Planning の式またはスクリプトであれば、「プラン・タイプ」を選択します
 - Essbase の式またはスクリプトであれば、「データベース」を選択します。

 ノート:

異なる名前では保存した式またはスクリプトのアプリケーション・タイプは変更できません。新規の式またはスクリプトのアプリケーション・タイプは元の式またはスクリプトと同じにする必要があります。また、式またはスクリプトの対象が **Financial Management** の場合は、計算タイプを変更できません。新規の式またはスクリプトの計算タイプは、元の式またはスクリプトと同じにする必要があります。

4. 「OK」をクリックします。

保存後、式またはスクリプト・コンポーネントを表示するには、システム・ビューでアプリケーション・リストのリフレッシュが必要な場合があります。[式およびスクリプト・コンポーネントのリフレッシュ](#)を参照してください。

式およびスクリプト・コンポーネントのリフレッシュ

式またはスクリプト・コンポーネントを作成した後、「式」ノードまたは「スクリプト」ノードで確認するために、システム・ビューでアプリケーション・リストのリフレッシュが必要な場合があります。

アプリケーション・タイプ、アプリケーション、あるいは式またはスクリプト・コンポーネントが属する計算またはプラン・タイプをリフレッシュする場合、デフォルトでは式およびスクリプト・コンポーネントをリフレッシュします。ただし、式またはスクリプト・コンポーネントをリフレッシュしても、アプリケーション・リストのより上位のレベル(すなわち、計算またはプラン・タイプ、アプリケーション、あるいはアプリケーション・タイプ)はリフレッシュされません。

 **ノート:**

式およびスクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールで作成できます。

式またはスクリプト・コンポーネントをリフレッシュするには、**システム・ビュー**で「**スクリプト**」または「**式**」を右クリックし、「**リフレッシュ**」を選択します。

 **ノート:**

また、データベース・アウトラインの上位レベルをリフレッシュして、その中のオブジェクトをリフレッシュすることもできます。たとえば、アプリケーションのスクリプトと式をリフレッシュするには、アプリケーション名を右クリックして「**リフレッシュ**」を選択します。

式およびスクリプト・コンポーネントの使用の表示

スクリプトまたは式コンポーネントの使用を表示できます。

 **ノート:**

式およびスクリプト・コンポーネントは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase のブロック・ストレージ・ビジネス・ルールで作成できます。

スクリプトまたは式コンポーネントの使用を表示するには:

1. **システム・ビュー**でスクリプトまたは式を右クリックし、「**使用状況の表示**」を選択します。
2. 「**使用状況**」ウィンドウで情報を確認し、「**OK**」をクリックします。

フロー・チャート内でのコンポーネントの操作

フロー・チャートでコンポーネントに対してアクションを実行できます。

次も参照:

- [フロー・チャート内でのコンポーネントの操作について](#)
- [フロー・チャート内でのコンポーネントの縮小および展開](#)
- [コンポーネントのフロー・チャートからの除去](#)

- フロー・チャート内でのコンポーネントのコピーおよび貼付け
- フロー・チャート内でのコンポーネントへの参照のコピーおよび貼付け
- フロー・チャート内でのコンポーネント・グループのコピーおよび貼付け

フロー・チャート内でのコンポーネントの操作について

フロー・チャート内で、コンポーネントを展開および縮小して詳細を表示または非表示にする、コンポーネントを除去する、コンポーネントをテンプレートとして保存する、コンポーネントをコピーして貼り付ける、コンポーネント・グループおよびコンポーネント参照をコピーして貼り付けるなどのアクションを行えます。テンプレート・デザイナーおよびルール・デザイナーのフロー・チャートでの作業中に、コンポーネントに対して行った変更の多くを元に戻すことができます。

フロー・チャート内でのコンポーネントの縮小および展開

ビジネス・ルールまたはテンプレートに多数の複合コンポーネントがある場合、フロー・チャートでその一部を縮小または展開できます。フロー・チャートのコンポーネントの縮小および展開により、操作するコンポーネントの表示スペースを拡大し、使用しないコンポーネントの表示スペースを縮小できます。

コンポーネントをフロー・チャートで縮小するには:

1. **システム・ビュー**でビジネス・ルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートで、コンポーネントに対していずれかのアクションを実行します:
 - コンポーネントを展開するには、コンポーネントを右クリックし、「**展開**」を選択します。
 - コンポーネントを縮小するには、コンポーネントを右クリックし、「**縮小**」を選択します。
3.  をクリックします。

コンポーネントのフロー・チャートからの除去

条件、メンバー範囲、データ範囲または固定ループ・コンポーネントをビジネス・ルールまたはテンプレートのフロー・チャートから除去すると、そのコンポーネントは削除されます。これらのコンポーネントは、共有できないため、1つのビジネス・ルールまたはテンプレートにのみ存在します。

式またはスクリプト・コンポーネントをフロー・チャートから除去しても、そのコンポーネントが完全に削除される場合とされない場合があります。式またはスクリプト・コンポーネントが共有されていない場合は、コンポーネントを除去すると完全に削除されます。式またはスクリプト・コンポーネントが共有されている場合は、除去元のビジネス・ルールまたはテンプレートからのみ削除されます。

コンポーネントをフロー・チャートから除去するには:

1. **システム・ビュー**でビジネス・ルールまたはテンプレートを右クリックし、「**開く**」を選択します。

2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートで、削除するコンポーネントを右クリックし、「**削除**」を選択します。
3.  をクリックします。

フロー・チャート内でのコンポーネントのコピーおよび貼付け

ビジネス・ルールまたはテンプレートのフロー・チャート内のコンポーネントをコピーすると、同じビジネス・ルールまたはテンプレートのフロー・チャート内の別の場所に貼り付けられるか、同じアプリケーション・タイプに属する場合は、異なるビジネス・ルールまたはテンプレートのフロー・チャートに貼り付けられます。

コンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**でビジネス・ルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**のフロー・チャートでコピーするコンポーネントを右クリックして、「**コピー**」を選択します。

ヒント:

「編集」、「コピー」を順に選択して使用もできます。

3. 次のいずれかのタスクを行います:
 - コンポーネントを**同じ**ビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートのコンポーネントを貼り付ける場所で右クリックし、「**貼付け**」を選択します。
 - コンポーネントを**異なる**ビジネス・ルールまたはテンプレートに貼り付けるには、ビジネス・ルールまたはテンプレートを開き、フロー・チャートのコンポーネントを貼り付ける場所で右クリックして「**貼付け**」を選択します。
4.  をクリックします。

フロー・チャート内でのコンポーネントへの参照のコピーおよび貼付け

コンポーネント自体とは異なり、コンポーネントへの参照をコピーすると、そのコンポーネントへのポインタのみがコピーされます。コンポーネントへの参照をコピーすると、コンポーネント自体は作成した元の場所にのみ存在します。

コンポーネントへの参照をコピーするとき、同じビジネス・ルールまたはテンプレートに貼り付けることも、同じアプリケーション・タイプに属する別のビジネス・ルールまたはテンプレートに貼り付けることもできます。

フロー・チャート内でコンポーネントへの参照をコピーして貼り付けるには:

1. **システム・ビュー**でビジネス・ルールまたはテンプレートを右クリックし、「**開く**」を選択します。

2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャート内のコピーするコンポーネントの参照を右クリックして、「**参照のコピー**」を選択します。

 **ヒント:**

「編集」、「参照のコピー」を順に選択して使用もできます。

 **ノート:**

「参照のコピー」は、共有コンポーネント(式およびスクリプト)または別のルールまたはテンプレートのフローで使用されるルールおよびテンプレートに対してのみ使用可能です。

3. 次のいずれかのタスクを行います:
 - コンポーネントの参照を *同じ* ビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートの参照を貼り付ける場所で右クリックし、「**貼付け**」を選択します。
 - コンポーネントの参照を *異なる* ビジネス・ルールまたはテンプレートに貼り付けるには、ビジネス・ルールまたはテンプレートを開き、フロー・チャートの参照を貼り付ける場所で右クリックして「**貼付け**」を選択します。
4.  をクリックします。

フロー・チャート内でのコンポーネント・グループのコピーおよび貼付け

コンポーネントに他のコンポーネントが含まれている(すなわち、コンポーネント内にコンポーネントがある)場合は、コンポーネントのグループをコピーして、同じフロー・チャート内の別の場所に貼り付ける、または別のフロー・チャートに貼り付けることができます。

フロー・チャート内でコンポーネント・グループをコピーして貼り付けるには:

1. **システム・ビュー**でビジネス・ルールまたはテンプレートを右クリックし、「**開く**」を選択します。
2. **ルール・デザイナー**または**テンプレート・デザイナー**で、フロー・チャート内のコピーするコンポーネント・グループを右クリックして、「**グループのコピー**」を選択します。

 **ヒント:**

「編集」、「グループのコピー」も使用できます。

3. 次のいずれかのタスクを行います:
 - コンポーネント・グループを *同じ* ビジネス・ルールまたはテンプレートに貼り付けるには、フロー・チャートのグループを貼り付ける場所で右クリックし、「**貼付け**」を選択します。

- コンポーネント・グループを異なるビジネス・ルールまたはテンプレートに貼り付けるには、ビジネス・ルールまたはテンプレートを開き、フロー・チャートのグループを貼り付ける場所で右クリックして「**貼付け**」を選択します。

4.  をクリックします。

集約ストレージ・コンポーネントを使用したビジネス・ルール設計

次も参照:

- **集約ストレージ・コンポーネントを使用したビジネス・ルール設計について**
集約ストレージ・コンポーネントを使用してビジネス・ルールを設計します。
- **視点コンポーネントの操作**
視点コンポーネントを作成および編集できます。
- **割当てコンポーネントの操作**
割当てコンポーネントを作成および編集できます。
- **視点または割当てコンポーネントを開く**
視点または割当てコンポーネントは、それらが属するビジネス・ルールのフロー・チャート内から開きます。システム・ビューで開くことはできません。
- **視点または割当てコンポーネントの削除**
視点または割当てコンポーネントを削除するには、それが属するビジネス・ルールから除去します。
- **視点または割当てコンポーネントのコピーおよび貼付け**
ビジネス・ルールの視点コンポーネントまたは割当てコンポーネントをコピーしてから、そのコンポーネントを同じビジネス・ルールまたは別のビジネス・ルールに貼り付けることができます。
- **視点または割当てコンポーネントの保存**
ルール・デザイナーでビジネス・ルールを保存する際、そのルールに属する視点または割当てコンポーネントが保存されます。
- **集約ストレージ式コンポーネントの操作**
集約ストレージ式コンポーネントの作成、オープン、編集、削除、コピー、貼付け、および使用状況の表示を行います。

集約ストレージ・コンポーネントを使用したビジネス・ルール設計について

集約ストレージ・コンポーネントを使用してビジネス・ルールを設計します。

Oracle Essbase 集約ストレージ・アプリケーション(Oracle General Ledger で使用するアプリケーションを含む)のビジネス・ルールは、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Essbase ブロック・ストレージ・アプリケーション内のビジネス・ルールとは異なるコンポーネントから構成されています。

Essbase 集約ストレージ・アプリケーションでのビジネス・ルール設計には、次のコンポーネントを使用します:

 ノート:

これらのコンポーネントは、Essbase 集約ストレージ・アプリケーションおよび Oracle General Ledger アプリケーションでのみ使用できます。

- 視点コンポーネントには、メタデータ・メンバーのリストが含まれます(勘定科目のリストなど)。

 ノート:

視点コンポーネントを別の視点コンポーネント内部にネストできます。

- 割当てコンポーネントには、データベース・アウトラインの1つのレベルにあるメンバーからアウトラインの他のメンバーにデータを配布するための計算が含まれます。
- 式コンポーネントには、メンバー、関数および変数を使用して設計する計算文が含まれます。

コンポーネントを作成する際は、作業しているビジネス・ルール、コンポーネント、テンプレートおよび変数を開いたままにできます。Oracle Hyperion Calculation Manager では、コンポーネントを作成する際にタブ間を簡単に移動できるように、タブ付きインターフェイスでこれらのオブジェクトが表示されます。Calculation Manager 内では最大 10 個のタブを開けますが、最適なパフォーマンスを得るためには、10 個を超えるオブジェクトを同時に開かないことをお勧めします。

視点コンポーネントの操作

視点コンポーネントを作成および編集できます。

次も参照:

- [視点コンポーネントの作成](#)
- [視点コンポーネントの編集](#)

視点コンポーネントの作成

 ノート:

これらのコンポーネントは、Oracle Essbase 集約ストレージ・アプリケーションおよび Oracle General Ledger アプリケーションでのみ使用できます。

ビジネス・ルール内から視点コンポーネントを作成します。

視点コンポーネントを作成するには:

1. **システム・ビュー**でルールを右クリックし、「開く」を選択します。

ビジネス・ルールがルール・デザイナーに表示されます。

2. ビジネス・ルールのフロー・チャート内の視点コンポーネントを作成する場所を決定したら、**新規オブジェクト・パレット**から「**視点**」コンポーネントをドラッグし、フロー・チャート内のその場所にドロップします。

視点オブジェクトが内部に矢印の付いた2つの丸として表示されます。

3. 「**視点**」で、視点コンポーネントを識別するキャプションを入力します。このキャプションは、その視点を使用するルールのフロー・チャート内で、コンポーネントの上に表示されます。
4. **オプション:** 視点のグローバル範囲を定義するには、次のいずれかのタスクを行います:

ノート:

視点コンポーネントを作成する対象のビジネス・ルールにグローバル範囲が定義されている場合、「視点」タブにはビジネス・ルールのメンバー選択がデフォルトで表示されます。グローバル範囲がビジネス・ルールに定義されているかどうかを確認するには、フロー・チャートで「開始」または「終了」タブを選択します。次に、「グローバル範囲」タブをクリックして、メンバー、関数または変数が定義されているかどうかを確認します。

- 「**変数セレクト**」をクリックして、視点を定義する変数を選択または作成します。変数を選択する場合、「**変数に動的にリンク**」を選択すると、変更が行われる際に変数が動的に確実に更新されます。
- 「**メンバー・セレクト**」をクリックして、視点を定義するメンバーを選択します。
- 「**値**」列でディメンションの行をクリックして、視点を定義するメンバーの名前を入力します。
- ディメンションの行をクリックし、「**アクション**」アイコンをクリックし、次のいずれかのオプションを選択してメンバーを入力します。

- [メンバー\(コンポーネントへのメンバーおよび関数の追加を参照\)](#)
- [変数\(変数の操作を参照\)](#)

非 Groovy ルールの場合、メンバーまたはメンバー(複数)タイプの変数のみが、**集約ストレージ・オプション(ASO)**タイプの **Oracle Hyperion Planning** キューブでサポートされているタイプの変数です

- [関数\(関数の操作を参照\)](#)

入力する関数はレベル 0 のメンバーのみを戻し、関数名の前に@記号が含まれる必要があります。List パラメータを持つすべての関数では、単一のメンバー、または 1 つのメンバーかメンバー・リストを戻す関数を入力する必要があります。

入力できる関数は次のとおりです:

- * 入力メンバーを指定すると、@ANCESTOR(Member Name, Index|Layer) は指定されたインデックスまたはレイヤーの祖先を戻します。(ルールの POV でのみ @ANCESTOR を使用します。)
- * @ATTRIBUTE(Attribute Member Name) は、指定された属性メンバーに関連付けられているすべての基本メンバーのリストを生成します。

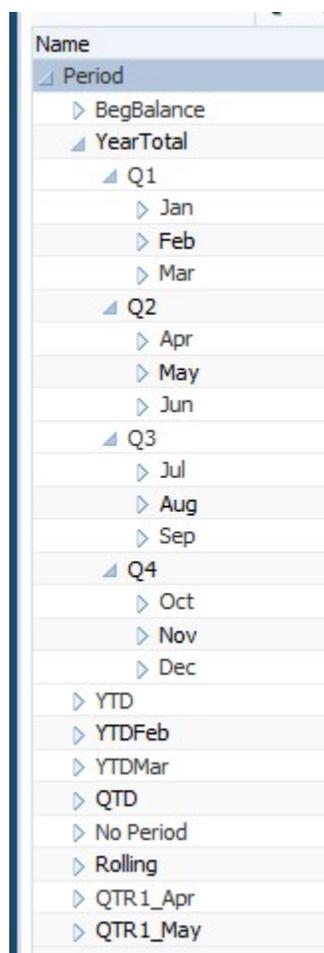
- * @COUSIN(*Member Name*,*Cousin Member Name*) は、別の祖先のメンバーと同じ位置の子メンバーを戻します(ルールの **POV** でのみ@COUSINを使用します。)
- * @DISTINCT(*List1*) は、重複するタプルをセットから削除します(ルールの **POV** でのみ@DISTINCTを使用します。)
- * @EXCEPT(*List1*,*List2*) は、2つのセット間の違いを含むサブセットを戻し、オプションで重複を保持します(ルールの **POV** でのみ@EXCEPTを使用します。)
- * @FilterDynamic(*Dimension Name*,*Member Name*) は、メンバーのリストからすべての動的メンバーを削除します(ルールの **POV** でのみ@FilterDynamicを使用します。)
- * @FilterShared(*Dimension Name*,*Member Name*) は、メンバーのリストからすべての共有メンバーを削除します(ルールの **POV** でのみ@FilterSharedを使用します。)
- * @FilterSharedAndDynamic(*Dimension Name*,*Member Name*) は、メンバーのリストから、動的で共有されているすべてのメンバーを削除します(ルールの **POV** でのみ@FilterSharedAndDynamicを使用します。)
- * @FIRSTCHILD(*Member Name*) は、入力メンバーの最初の子を戻します(ルールの **POV** でのみ@FIRSTCHILDを使用します。)
- * @FIRSTSIBLING(*Member Name*) は、入力メンバーの親の最初の子を戻します。グローバル **POV** でのみ@FIRSTSIBLINGを使用します。
- * @GEN(*Member Name*) は、指定されたメンバーの入力世代数によって指定されるメンバーを戻します(ルールの **POV** でのみ@GENを使用します。)
- * @GENMBRS(*Member Name*,*Generation*) は、入力世代数によって指定された世代を戻します。(ルールの **POV** でのみ@GENMBRSを使用します。)
- * @ILSIBLINGS は、指定されたメンバーおよびその左側の兄弟を戻します。(ルールの **POV** でのみ@ILSIBLINGSを使用します。)
- * @INTERSECT(*List1*,*List2*) は、2つの入力セットの共通部分を戻します(オプションで重複の保持が可能)(ルールの **POV** でのみ@INTERSECTを使用します。)
- * @IRSIBLINGS は、指定されたメンバーおよびその右側の兄弟を戻します。(ルールの **POV** でのみ@IRSIBLINGSを使用します。)
- * @LAGGEN(*Member Name*,*Index*) は、データベース・アウトラインに存在するメンバーの順序を使用して、同じ世代の指定されたメンバーより *n* ステップ後ろのメンバーを戻します。グローバル **POV** でのみ@LAGGENを使用します。
- * @LAGLEV(*Member Name*,*Index*) は、データベース・アウトラインに存在するメンバーの順序を使用して、同じレベルの指定されたメンバーより *n* ステップ後ろのメンバーを戻します。グローバル **POV** でのみ@LAGLEVを使用します。

- * @LASTCHILD (*Member Name*) は、入力メンバーの最後の子を戻します(ルールの **POV** でのみ@LASTCHILD を使用します。)
- * @LASTSIBLING (*Member Name*) は、入力メンバーの親の最後の子を戻します。グローバル **POV** でのみ@LASTSIBLING を使用します。
- * @LAYERGEN (*Member Name, Index*) は、指定されたメンバーのレイヤーに基づく世代を戻します。

@LAYERGEN は、たとえば、@ANCESTOR (*member, layer, index*) など、パラメータがレイヤーを検索する関数内でのみ使用できます。@LAYERGEN を単独で使用することはできません。ルールの **POV** でのみ@LAYERGEN を使用します。

次の「期間」ディメンションで、

@Level0Descendants (@ANCESTOR (Jul, @LAYERGEN (Period, 3))) は「7月」、「8月」、「9月」を戻します



- * @LAYERLEVEL (*Member Name, Index*) は、指定されたメンバーのレイヤーに基づくレベルを戻します。

@LAYERLEVEL は、たとえば、@ANCESTOR (*member, layer, index*) など、パラメータがレイヤーを検索する関数内でのみ使用できます。@LAYERLEVEL を単独で使用することはできません。ルールの **POV** でのみ@LAYERLEVEL を使用します。

次の「期間」ディメンションで、
@ANCESTOR(Oct,@LAYERLEVEL(Period,0))は「10月」、「11月」、「12月」を戻します

Name
Period
> BegBalance
▲ YearTotal
▲ Q1
> Jan
> Feb
> Mar
▲ Q2
> Apr
> May
> Jun
▲ Q3
> Jul
> Aug
> Sep
▲ Q4
> Oct
> Nov
> Dec
> YTD
> YTDFeb
> YTDMar
> QTD
> No Period
> Rolling
> QTR1_Apr
> QTR1_May

- * @LEADGEN(*Member Name*,*Index*)は、データベース・アウトラインに存在するメンバーの順序を使用して、同じ世代の指定されたメンバーより *n* ステップ先のメンバーを戻します。グローバル POVでのみ@LEADGENを使用します。
- * @LEADLEV(*Member Name*,*Index*)は、データベース・アウトラインに存在するメンバーの順序を使用して、同じレベルの指定されたメンバーより *n* ステップ先のメンバーを戻します。グローバル POVでのみ@LEADLEVを使用します。
- * @Level0Descendant(*Member Name*)は、展開して、指定されたメンバーのすべてのレベル 0の子孫を含めます
- * @LIST(*Argument*)は、メンバーのリストを戻します(ルールの POVでのみ@LISTを使用します。)
- * @LSIBLINGSは、指定されたメンバーの左側の兄弟を戻します。(ルールの POVでのみ@LSIBLINGSを使用します。)

- * @NEXTLEVMBR (*Member Name*) は、データベース・アウトラインに存在するメンバーの順序を使用して、同じレベルの次のメンバーを戻します。グローバル POV でのみ@NEXTLEVMBR を使用します。
 - * @NEXTMBR (*Member Name*) は、データベース・アウトラインに存在するメンバーの順序に従って、同じ世代またはレベルの次のメンバーを戻します(ルールの POV でのみ@NEXTMBR を使用します。)
 - * @PARENT (*Member Name*) は、メンバーの親を戻します。(ルールの POV でのみ@PARENT を使用します。)
 - * @PREVLEVMBR (*Member Name*) は、データベース・アウトラインに存在するメンバーの順序を使用して、同じレベルの前のメンバーを戻します。グローバル POV でのみ@PREVLEVMBR を使用します。
 - * @PREVMBR (*Member Name*) は、データベース・アウトラインに存在するメンバーの順序を使用して、同じ世代またはレベルの前のメンバーを戻します。(ルールの POV でのみ@PREVMBR を使用します。)
 - * @RSIBLINGS は、指定されたメンバーの右側の兄弟を戻します。(ルールの POV でのみ@RSIBLINGS を使用します。)
 - * @Siblings (*Member Name*) は、展開して、指定されたメンバーのすべての兄弟を含めます
 - * @UDA (*Dimension Name, User-Defined Attribute String*) は、Essbase サーバーでユーザー定義属性(UDA)として定義した共通属性に基づいてメンバーを選択します
5. 選択したディメンションのメンバーに関するコメントを入力するには、「**コメント**」をクリックします。
 6. グリッドに作成したすべてのエントリをクリアするには、「**グリッドをリセット**」をクリックします。
 7.  をクリックします。

視点コンポーネントの編集

視点コンポーネントのグローバル範囲を構成するメンバー、変数および関数を編集できます。また、フロー・チャートのコンポーネントおよび視点のディメンションごとに選択した値に入力されたコメントの上に表示されるキャプションを編集できます。

視点コンポーネントを編集するには:

1. **システム・ビュー**で視点を含むビジネス・ルールを右クリックし、「**開く**」を選択します。
2. ルール・デザイナーで、編集する視点コンポーネントをフロー・チャートで選択してそのプロパティを表示します。次の視点コンポーネントのプロパティを編集できます。(視点コンポーネントの作成を参照してください)
 - ビジネス・ルールのフロー・チャートで視点コンポーネントの上に表示されるキャプション
 - 視点を定義するメンバー、変数および関数
 - 変数に変更された場合、視点コンポーネントに使用されている変数を動的に更新するかどうか

- 視点のグローバル範囲を定義するディメンションおよびメンバーにコメントを入力するかどうか
- 属するビジネス・ルールが検証または起動された場合、視点コンポーネントのメンバーの値を計算するかどうか

3.  をクリックします。

割当てコンポーネントの操作

割当てコンポーネントを作成および編集できます。

次も参照:

- [割当てコンポーネントの作成](#)
- [割当てコンポーネントの編集](#)

割当てコンポーネントの作成

ノート:

これらのコンポーネントは、Oracle Essbase 集約ストレージ・アプリケーションおよび Oracle General Ledger アプリケーションでのみ使用できます。

割当てコンポーネントを使用すると、メンバーからメンバーのレベル 0 の子孫にデータを配布できます。割当てコンポーネントはビジネス・ルール内から作成します。これはそのビジネス・ルールのみが存在し、ビジネス・ルール間で共有することはできません。

割当てコンポーネントを作成するには:

1. システム・ビューでルールを右クリックし、「開く」を選択します。
ビジネス・ルールがルール・デザイナに表示されます。
2. 割当てコンポーネントを作成するビジネス・ルールのフロー・チャート内の場所を決定し、「新規オブジェクト」パレットから「割当て」コンポーネントをドラッグし、フロー・チャート内の該当する場所にドロップします。

ノート:

視点コンポーネントを別の視点コンポーネント内にドロップすると、最初(上位)の視点のメンバー、変数および関数が 2 番目の視点に継承されません。

3. 「割当」ウィザードの「視点」で、割当て時に変更しないリスト内のディメンションごとに次のいずれかのタスクを行います:
 - 「事前に定義されている選択項目の使用」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。

- 「**メンバー・セレクタ**」アイコンをクリックし、リスト内のディメンションごとにメンバーと変数を選択します。

 **ノート:**

メンバー・セレクタでは、ウィザードの現在のステップにリストされるディメンションを「ディメンション」から選択できます。そのため、ウィザードの現在のステップにリストされるディメンションのメンバーと関数を選択できます。

選択したすべてのメンバーが有効なレベル0のメンバーであることを確認してください。

- リスト内のディメンションを選択し、「**アクション**」をクリックしてメンバーまたは変数を選択します。[メンバーの選択](#)、[変数](#)、[関数](#)、[スマート・リスト](#)および [Planning 数式を使用したコンポーネントの設計](#)を参照してください。
4. 「割当て」ウィザードの「**ソース**」で、リスト内のディメンションごとに、次のタスクのいずれかを行って、割り当てるデータを所有するメンバーを選択します。

 **ノート:**

リストされるディメンションごとにメンバーを選択する必要があります。
ソース・メンバーにはレベル0以外のメンバーを指定できます。

- 「**事前に定義されている選択項目の使用**」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。

 **ノート:**

リストされる各ディメンションについて、事前定義済の選択に値が入力されない場合、空になっているすべてのディメンションの値を入力する必要があります。

- 「**メンバー・セレクタ**」アイコンをクリックし、リスト内のディメンションごとにメンバーを選択します。
- リスト内のディメンションを選択し、「**アクション**」をクリックしてメンバーまたは変数を選択します。[メンバーの選択](#)、[変数](#)、[関数](#)、[スマート・リスト](#)および [Planning 数式を使用したコンポーネントの設計](#)を参照してください。

 **ノート:**

割当てコンポーネントのこのステップに関数を使用できません。

- **オプション**で、特定の値を割り当てるには、前述の選択内容のかわりに割り当てる量を入力します。

5. 割り当てるソース金額がゼロの場合、ドロップダウンで次のいずれかのオプションを選択します。
 - 次のプール・レコードを選択します。
 - 割当ての処理を停止します。
6. 「**割当て範囲**」で、次を行います:
 - 割当てに使用するディメンションの親メンバーを入力します。
親メンバーを入力するには、次のいずれかのタスクを行います:
 - 「**事前に定義されている選択項目の使用**」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。
 - 「**メンバー・セレクタ**」アイコンをクリックして、データの割当て先のディメンションの親メンバーを選択します。
 - 親メンバーを入力するか、リスト内のディメンションを選択し、「**アクション**」アイコンをクリックしてデータの割当て先の(主ディメンションの)親メンバーを選択します。[メンバーの選択、変数、関数、スマート・リストおよび Planning 数式を使用したコンポーネントの設計](#)を参照してください。

データは、データベース・アウトラインの親メンバーの下のレベル 0 メンバー(つまり、下にメンバーのない、アウトライン内で最下位のメンバー)に割り当てられます。
 - 割当てプロセスの前に領域をクリアするかどうかを選択します。
「はい」を選択して領域をクリアする場合、割当て実行前にクリアするメンバーを「**領域のクリア**」フィールドに入力します(ステップ 8 を参照)。
7. 残りのディメンションの「**ターゲット**」で、データの割当て先のレベル 0 のメンバーを選択します。次のいずれかのタスクを行います:
 - 「**事前に定義されている選択項目の使用**」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。
 - 「**メンバー・セレクタ**」アイコンをクリックし、リスト内のディメンションごとにメンバーを選択します。
 - リスト内のディメンションを選択し、「**アクション**」アイコンをクリックしてメンバーまたは変数を選択します。[メンバーの選択、変数、関数、スマート・リストおよび Planning 数式を使用したコンポーネントの設計](#)を参照してください。
8. 「**領域のクリア**」に、割当てプロセスの前に論理クリアでクリアするレベル 0 のメンバーを入力します。次のいずれかのタスクを行います:
 - 「**事前に定義されている選択項目の使用**」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。
 - 「**メンバー・セレクタ**」アイコンをクリックし、リスト内のディメンションごとにメンバーを選択します。
 - リスト内のディメンションを選択し、「**アクション**」アイコンをクリックしてメンバーまたは変数を選択します。[メンバーの選択、変数、関数、スマート・リストおよび Planning 数式を使用したコンポーネントの設計](#)を参照してください。

 ノート:

「領域のクリア」が表示されない場合は、割当て前にデータをクリアするために「割当て範囲」で「はい」を選択していません。

9. 「オフセット」で、次のいずれかのタスクを行います:

- 「事前に定義されている選択項目の使用」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。
- 「メンバー・セレクタ」アイコンをクリックし、リスト内のディメンションごとにメンバーを選択します。
- リスト内のディメンションを選択し、「アクション」アイコンをクリックしてメンバーまたは変数を選択します。[メンバーの選択](#)、[変数](#)、[関数](#)、[スマート・リスト](#)および [Planning 数式を使用したコンポーネントの設計](#)を参照してください。

 ノート:

オフセットのメンバー指定する必要があります。空のままにすることはできません。

10. オプション: 「除外」で、割当てから除外するメンバーを選択します。次のいずれかのタスクを行います:

- 「事前に定義されている選択項目の使用」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。
- 「メンバー・セレクタ」アイコンをクリックし、リスト内のディメンションごとにメンバーを選択します。
- リスト内のディメンションを選択し、「アクション」アイコンをクリックしてメンバーまたは変数を選択します。[メンバーの選択](#)、[変数](#)、[関数](#)、[スマート・リスト](#)および [Planning 数式を使用したコンポーネントの設計](#)を参照してください。

11. 「基準」で、次のタスクを行います:

- a. 割当て方法を選択してデータを割り当てる方法を指定します。
 - 「均等に割当て」を選択して、データ値を割当て範囲に均等に割り当てます。次に、[均等方法の基準オプション](#)で、基準が負、ゼロ、値がない、またはすべてのメンバーが除外されている場合に実行する処理を指定します。
 - 「ドライバを使用して割当て」を選択して、割当て範囲の各メンバーに適用されるパーセンテージを計算します。[基準オプション](#)に、基準が負またはゼロに等しい場合に実行する処理を指定します。
- b. 指定していないディメンション・メンバーは事前に定義した POV から継承されますが、次のいずれかのタスクを実行すると、POV の選択肢を上書きできます:
 - 「事前に定義されている選択項目の使用」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。
 - 「メンバー・セレクタ」アイコンをクリックし、リスト内のディメンションごとにメンバーを選択します。

- リスト内のディメンションを選択し、「アクション」アイコンをクリックしてメンバーまたは変数を選択します。メンバーの選択、変数、関数、スマート・リストおよび Planning 数式を使用したコンポーネントの設計を参照してください。
 - c. 「次」をクリックします。
12. 「丸め処理」で、次のステップを実行します:
- a. この割当てに使用する小数点以下の桁数を入力するか、「アクション」アイコンをクリックして、この値を示すメンバーまたは変数を選択します。
 - b. 丸め差異を適用する場所を選択します。
 - 丸め差異を適用するメンバーを指定するには、場所の定義を選択します。
 - データ値を最大値に丸めるには、「最大値の使用」を選択します
 - データ値を最小値に丸めるには、最小値の使用を選択します。
 - 割り当てたデータ値をそのまま使用する場合は、丸め誤差の破棄を選択します。
13. 前のステップで場所の定義を選択した場合、「丸め処理メンバー」で、次のいずれかのタスクを行います:
- 「事前に定義されている選択項目の使用」から事前定義済の選択肢を選択し、値とともにディメンションのリストを移入します。
 - 「メンバー・セレクタ」アイコンをクリックし、リスト内のディメンションごとにメンバーを選択します。
 - リスト内のディメンションを選択し、「アクション」アイコンをクリックしてメンバーまたは変数を選択します。メンバーの選択、変数、関数、スマート・リストおよび Planning 数式を使用したコンポーネントの設計を参照してください。

 ノート:

このステップで選択するメンバーは割当て範囲に含まれている必要があります。

14. 「完了」をクリックします。

割当てコンポーネントの編集

ビジネス・ルールを開いて、そのルールに属する割当てコンポーネントを編集できます。ルール・デザイナーにビジネス・ルールが表示されたら、ビジネス・ルールのフロー・チャートでコンポーネントを選択して、割当てコンポーネントのプロパティを表示できます。

割当てコンポーネントを編集するには:

1. システム・ビューで割当てコンポーネントを含むビジネス・ルールを右クリックし、「開く」を選択します。
2. ルール・デザイナーで、フロー・チャート内の編集する割当てコンポーネントを選択して、そのプロパティを表示します。次の割当てコンポーネントのプロパティを編集できます。(割当てコンポーネントの作成を参照してください。)

- 割り当てるデータを所有するメンバー
- データを割り当てるレベル0のメンバー
- 割り当てるデータおよびデータ量
- 割り当てたデータの合計数をオフセット・メンバーに書き込むかどうか
- ドライバを使用して、データを等分に割り当てるか、様々な量で割り当てるか
- 割り当てられたデータを丸めるかどうか。実行する場合の丸め処理の方法

3.  をクリックします。

視点または割当てコンポーネントを開く

視点または割当てコンポーネントは、それらが属するビジネス・ルール・フロー・チャート内から開きます。システム・ビューで開くことはできません。

視点または割当てコンポーネントを開くには:

1. **システム・ビュー**でコンポーネントを含むルールを右クリックし、「**開く**」を選択します。ビジネス・ルールがルール・デザイナーに表示されます。
2. ルールを開いたら、ルール・フロー・チャートで、視点コンポーネントをクリックする、または割当てコンポーネントをダブルクリックして、コンポーネントを開きます。

視点または割当てコンポーネントの削除

視点または割当てコンポーネントを削除するには、それが属するビジネス・ルールから除去します。

視点および割当てコンポーネントは1つのビジネス・ルールでのみ使用できるため、ビジネス・ルールに属するこれらのコンポーネントをビジネス・ルールから除去すると、コンポーネントは削除されます。

視点または割当てコンポーネントを削除するには:

1. **システム・ビュー**で視点または割当てコンポーネントを含むルールを右クリックし、「**開く**」を選択します。
2. ルール・デザイナーで、フロー・チャートで削除する視点または割当てコンポーネントを選択します。
3. 視点または割当てコンポーネントを右クリックし、「**削除**」を選択します。

割当てコンポーネントが視点コンポーネント内に含まれる場合、視点コンポーネントを削除すると、割当てコンポーネントが削除されます。

4.  をクリックします。

視点または割当てコンポーネントのコピーおよび貼付け

ビジネス・ルールの視点コンポーネントまたは割当てコンポーネントをコピーしてから、そのコンポーネントを同じビジネス・ルールまたは別のビジネス・ルールに貼り付けることができます。

ビジネス・ルールのフロー・チャートで視点または割当てコンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**で視点または割当てコンポーネントを含むルールを右クリックし、「開く」を選択します。
ビジネス・ルールがルール・デザイナーに表示されます。
2. **ルール・デザイナー**で、ビジネス・ルールのフロー・チャートにコピーする視点または割当てコンポーネントを右クリックし、コンポーネントのみをコピーする場合は「コピー」を選択し、コンポーネントおよびそれに含まれるすべてのコンポーネントをコピーする場合は「グループのコピー」を選択します。
3. 次のいずれかのタスクを行います:
 - コンポーネントを同じビジネス・ルールに貼り付けるには、フロー・チャートのコンポーネントを貼り付ける場所で右クリックし、「貼付け」を選択します。
 - コンポーネントを異なるビジネス・ルールに貼り付けるには、ビジネス・ルールを開き、フロー・チャートのコンポーネントを貼り付ける場所で右クリックして「貼付け」を選択します。
4.  をクリックします。

視点または割当てコンポーネントの保存

ルール・デザイナーでビジネス・ルールを保存する際、そのルールに属する視点または割当てコンポーネントが保存されます。

式コンポーネントとは異なり、視点および割当てコンポーネントは、それらが作成されるビジネス・ルールから独立して存在することはできません。

視点または割当てコンポーネントを保存するには、コンポーネントを設計した後、 をクリックします。

集約ストレージ式コンポーネントの操作

集約ストレージ式コンポーネントの作成、オープン、編集、削除、コピー、貼付け、および使用状況の表示を行います。

次も参照:

- [集約ストレージ式コンポーネントの作成](#)
- [集約ストレージ式コンポーネントを開く](#)
- [集約ストレージ式コンポーネントの編集](#)

- 集約ストレージ式コンポーネントの削除
- 集約ストレージ式コンポーネントのコピーおよび貼付け
- 別のアプリケーションまたはデータベースへの集約ストレージ式コンポーネントのコピー
- 集約ストレージ式コンポーネントの使用状況の表示

集約ストレージ式コンポーネントの作成

ノート:

集約ストレージ式コンポーネントは、Oracle Essbase 集約ストレージ・アプリケーションおよび Oracle General Ledger アプリケーションのみで使用できます。

集約ストレージ式コンポーネントは、式の計算文で構成されています。式の計算文を作成するには、メンバーおよび変数を入力または選択します。式を作成する際、コンポーネント・デザイナーのグリッド内の行にその各計算文がリストされます。

式コンポーネントは、システム・ビューまたはルール・デザイナーで作成できます。式コンポーネントは独立したオブジェクトとしてデータベースに存在するため、ビジネス・ルール間で共有できます。

集約ストレージ・アプリケーション用の式コンポーネントを作成するには:

1. システム・ビューで「式」を右クリックし、「新規」を選択します。
2. 式名およびアプリケーション・タイプを入力します。
3. アプリケーションを選択します。

アプリケーション名は、有効な Essbase 集約ストレージまたは Oracle General Ledger アプリケーションであることが必要です。

4. 「データベース」を選択します。

ノート:

「式」を右クリックし、「新規」を選択して新しい式を作成する場合、システム・ビュー内で操作しているアプリケーション・タイプ、アプリケーションおよびデータベースが「新規式」ダイアログに挿入されます。

5. 「OK」をクリックします。
6. コンポーネント・デザイナーの「プロパティ」で、次のステップを実行します。
 - a. オプション: 式は、作成時にデフォルトで共有されます。「共有」チェック・ボックスを選択する、またはクリアすることはできません。

 **ヒント:**

共有でない式を作成するには、ビジネス・ルールを開いて、新しい式コンポーネントをビジネス・ルールのフロー・チャートにドラッグします。「共有」チェック・ボックスは選択されていません。式を共有する場合は、「共有」チェック・ボックスを選択します。

- b. **オプション:** 新規名前を 50 文字以下で入力することにより、式の名前を編集します。(デフォルトの名前は「新規式」ダイアログで指定されます。)

 **ノート:**

Essbase 集約ストレージ・アプリケーションおよび Oracle General Ledger アプリケーションのメンバーの名前は一意である必要はありません。一意ではない名前がある場合、メンバーが使用されているオブジェクトが検証される際に、その名前がフラグ付けされます。これが発生した場合は、メンバーのフル・パスを入力する必要があります。フル・パスの構文は、[ディメンション名].[親名].[メンバー名]です。

- c. **オプション:** 式の説明を 255 文字以下で入力します。
- d. **オプション:** 式のキャプションを入力します。キャプションは、**ルール・デザイン**のフロー・チャートで式の下に表示されます。

 **ノート:**

キャプションを入力しない場合、フロー・チャートではコンポーネントの名前が表示されます。

- e. **オプション:** 式のコメントを入力します。たとえば、式コンポーネントの使用方法をユーザーに伝えられます。
7. 「式」に、式のキャプションを入力します。
8. **オプション:** 「オフセット・メンバー」から、オフセット・メンバーまたはディメンション間メンバーに書き込まれる式コンポーネントの式のすべての **合計金額**が必要な場合、**省略記号**アイコンをクリックしてメンバーを選択します。

 **ノート:**

オフセット・メンバーは、下で作成する式内に手動で定義できます。式コンポーネントで定義されたオフセットは、すべての計算済金額の合計として計算されます。

9. 式の文を作成するには、等号の左側にある 1 番目の式の文の行内をクリックします。次に、メンバーまたはディメンション間メンバーの選択を入力するか、「**アクション**」をクリックして、次を選択します:

- 変数(変数の操作を参照してください。)
- メンバー(コンポーネントへのメンバーおよび関数の追加を参照してください。)

 ノート:

ディメンション間メンバーの選択を入力する場合は、各メンバー名を右矢印で区切って入力します。たとえば、mem1->mem2->mem3 のようになります。

10. 式の文を完成するには、等号の右側にある行内をクリックします。次に、メンバーまたはディメンション間メンバーの選択を入力するか、「アクション」をクリックして、次を選択します:
 - 変数(変数の操作を参照してください。)
 - メンバー(コンポーネントへのメンバーおよび関数の追加を参照してください。)
11. オプション: 式の文行ごとに、「コメント」アイコンをクリックし、式の文に関するコメントを入力します。「OK」をクリックします。
12. 「使用状況」では、式コンポーネントを使用するルールを確認できます。

 ノート:

このタブの情報はどれも編集できません。

13.  をクリックします。

集約ストレージ式コンポーネントを開く

集約ストレージ式コンポーネントは、システム・ビューから、または式コンポーネントを使用するビジネス・ルールのルール・デザイナのフロー・チャート内から開くことができます。

集約ストレージ式コンポーネントを開くには、システム・ビューで「式」を展開し、次のいずれかのタスクを実行します。

- 開く式を右クリックし、「開く」を選択します。
- 開く式をダブルクリックします。
式コンポーネントがコンポーネント・デザイナで開かれます。

 ノート:

式コンポーネントをビジネス・ルール内で開くには、式コンポーネントを右クリックして「開く」を選択するか、そのコンポーネントをダブルクリックして、ルールのフロー・チャート内から開きます。

集約ストレージ式コンポーネントの編集

集約ストレージ式コンポーネント、および式コンポーネントのコメント、キャプション、名前および説明で構成される式の文を編集できます。

集約ストレージ式コンポーネントを編集するには:

1. **システム・ビュー**で式コンポーネントを右クリックし、「**開く**」を選択します。
2. コンポーネント・デザイナーで、式コンポーネントの次のプロパティを編集できます。[集約ストレージ式コンポーネントの作成](#)を参照してください。
 - キャプション
 - 式の文
 - 名前
 - 説明
 - コメント
3.  をクリックします。

集約ストレージ式コンポーネントの削除

集約ストレージ式コンポーネントは、どのビジネス・ルールでも使用されていない場合にのみ削除できます。ビジネス・ルールが式コンポーネントを使用しているかどうかを調べるには、式コンポーネントの使用状況を表示します。[集約ストレージ式コンポーネントの使用状況の表示](#)を参照してください。

式コンポーネントがビジネス・ルールで使用されているが、そのルールで使用する必要がなくなった場合は、ルールから除去した後に、式コンポーネントを削除します。式コンポーネントを使用しているビジネス・ルールが不要になった場合は、ビジネス・ルールを削除できます。

ビジネス・ルールで使用されていない式コンポーネントは、削除できます。

集約ストレージ式コンポーネントを削除するには:

1. **システム・ビュー**で式を右クリックし、「**使用状況の表示**」を選択します。
2. 式コンポーネントを使用しているビジネス・ルールがないことを確認します。[集約ストレージ式コンポーネントの使用状況の表示](#)を参照してください。
3. 式コンポーネントを使用しているビジネス・ルールから式コンポーネントを削除します。[コンポーネントのフロー・チャートからの除去](#)を参照してください。
4. 式を再度右クリックし、「**削除**」を選択します。
5. 式の削除を確認します。

集約ストレージ式コンポーネントのコピーおよび貼付け

ルールから集約ストレージ式コンポーネントをコピーして、同じルールや異なるルールに貼り付けることができます。式コンポーネント内のグリッドのコンテンツをコピ

ーして、同じ式コンポーネントや異なる式コンポーネントに貼り付けることもできます。式コンポーネントをコピーして、別の式コンポーネントや別のコンポーネント・タイプに貼り付けることはできません。

集約ストレージ式コンポーネントをコピーして貼り付けるには:

1. **システム・ビュー**で式コンポーネントを含むルールを右クリックし、「開く」を選択します。
2. ルール・デザイナーのフロー・チャートで、コピーする式コンポーネントを右クリックし、「コピー」を選択します。

ノート:

コピーするコンポーネントが共有されている場合、コンポーネント自体をコピーするかわりに、「編集」、「参照のコピー」を使用して参照を共有コンポーネントにコピーできます。(ビジネス・ルールの式またはスクリプト・コンポーネントへの参照のコピーおよび貼付けを参照してください。)

3. 次のいずれかのタスクを行います:
 - 式コンポーネントを同じビジネス・ルールのフロー・チャートに貼り付けるには、フロー・チャートの貼り付ける場所で右クリックし、「貼付け」を選択します。
 - 式コンポーネントを異なるビジネス・ルールのフロー・チャートに貼り付けるには、コンポーネントを貼り付けるビジネス・ルールを開き、貼り付ける場所で右クリックして「貼付け」を選択します。
4.  をクリックします。

別のアプリケーションまたはデータベースへの集約ストレージ式コンポーネントのコピー

1つのアプリケーションから別のアプリケーションおよびデータベースに、または同じアプリケーション内で1つのデータベースから別のデータベースに、集約ストレージ式コンポーネントをコピーできます。

集約ストレージ式コンポーネントをコピーするには:

1. **システム・ビュー**でコピーする式コンポーネントを右クリックし、「コピー先」を選択します。
2. 「名前を付けて保存」で、次のタスクのいずれかを行います:
 - 式コンポーネントを別のアプリケーションにコピーするには、「アプリケーション」に名前を入力します。
 - 式コンポーネントを別のアプリケーションおよびデータベースにコピーするには、「アプリケーション」および「データベース」に名前を入力します。
 - 式コンポーネントを同じアプリケーション内の別のデータベースにコピーするには、「データベース」に名前を入力します。
3. 「OK」をクリックします。

式コンポーネントがコピー先のアプリケーションおよびデータベースの「式」ノードに表示されます。

 ノート:

式コンポーネントのコピー先のアプリケーション・ノードまたはデータベース・ノードのリフレッシュが必要な場合があります。アプリケーション・ノードまたはデータベース・ノードを右クリックして「リフレッシュ」を選択します。

集約ストレージ式コンポーネントの使用状況の表示

集約ストレージ式コンポーネントを使用しているビジネス・ルール、およびビジネス・ルールに関するその他の情報を確認するには、システム・ビューで式コンポーネントの使用状況を表示します。

集約ストレージの式コンポーネントの使用状況を表示するには:

1. **システム・ビュー**で使用状況の表示対象の式を右クリックし、「**使用状況の表示**」を選択します。
2. 式コンポーネントに関する次の情報を表示できます。
 - その式コンポーネントを使用しているビジネス・ルールの名前
 - その式コンポーネントを使用しているビジネス・ルールのアプリケーション名
 - その式コンポーネントを使用しているビジネス・ルールのデータベース名
 - 式コンポーネントの所有者
 - その式コンポーネントを使用しているビジネス・ルールをデプロイするかどうか
 - その式コンポーネントを使用しているビジネス・ルールを検証するかどうか
 - その式コンポーネントを使用しているビジネス・ルールの説明

 ノート:

式コンポーネントの使用は、「用途」タブのコンポーネント・デザイナーから確認することもできます。

8

メンバーの選択、変数、関数、スマート・リストおよび Planning 数式を使用したコンポーネントの設計

次も参照:

- [メンバー選択、変数、関数およびスマート・リストについて](#)
コンポーネントを使用してビジネス・ルールを設計するのと同様に、メンバー、変数および関数を使用してコンポーネントを設計します。
- [コンポーネントへのメンバーおよび関数の追加](#)
式コンポーネント、スクリプト・コンポーネント、条件コンポーネント、メンバー範囲コンポーネントおよびデータ範囲コンポーネントにメンバーを追加します。
- [変数の操作](#)
ビジネス・ルールおよびテンプレートの設計時にコンポーネントで変数を使用します。
- [関数の操作](#)
関数は、データ値またはメンバーを戻すメンバー式を定義します。
- [Calculation Manager でサポートされている Essbase 関数](#)
次の Oracle Essbase 関数は、ブロック・ストレージ・アプリケーションで Oracle Hyperion Calculation Manager によってサポートされています。
- [カスタム関数の操作](#)
カスタム関数を使用して、データのコピーとエクスポート、テキスト文字列に対する一重引用符または二重引用符の削除と追加、2つのテキスト文字列の比較、別の形式への日付の変換などのタスクを実行します。
- [コンポーネントへの関数の挿入](#)
選択できる関数のタイプは使用しているアプリケーションのタイプおよびコンポーネントのタイプによって異なります。
- [スマート・リストの操作](#)
スマート・リストは Planning アプリケーションの Oracle Hyperion Planning データ・フォームのセルからアクセスできるカスタムのドロップダウン・リストです。
- [スマート・リストの挿入](#)
Oracle Hyperion Planning ビジネス・ルール、式コンポーネントまたはスクリプト・コンポーネントのスマート・リストを使用します。
- [Planning 数式の操作](#)
Oracle Hyperion Calculation Manager のグラフィカル・ルールまたはスクリプト・ルールで、Oracle Hyperion Planning の数式を使用します。

メンバー選択、変数、関数およびスマート・リストについて

コンポーネントを使用してビジネス・ルールを設計するのと同様に、メンバー、変数および関数を使用してコンポーネントを設計します。

式、スクリプト、条件、メンバーおよびデータ範囲、およびメンバー・ブロックの各コンポーネントでメンバー選択を使用して、メンバーのリスト(Oracle Hyperion Planning および Oracle Essbase アプリケーションの場合)またはメンバーおよびメンバー・リスト(Oracle Hyperion Financial Management アプリケーションの場合)に戻すメンバーおよび関数を選択します。コンポーネントへのメンバーおよび関数の追加を参照してください。

変数は、式、ループおよび条件の各コンポーネントに対して式を構築するために使用します。ビジネス・ルールが起動されると計算される実行変数と、より複雑な式または関数を代替する置換変数の 2 種類の変数があります。(変数の操作を参照してください。)

次の種類の変数を作成できます:

- アプリケーション・タイプに属するすべてのアプリケーションで使用するグローバル変数
- 1 つのアプリケーションのみに使用するアプリケーション変数
1 つのプラン・タイプまたはデータベースで使用する場合はプラン・タイプまたはデータベース変数
- 1 つのビジネス・ルールのみ使用するルール変数

関数は、ループ、条件および式の各コンポーネントで使用できる事前に定義された式です。コンポーネントで使用できる関数は、Financial Management、Planning および Essbase アプリケーションで異なります。(Financial Management、Planning および Essbase アプリケーションで使用できる関数の種類の説明は関数の操作を参照。Financial Management コンポーネントで使用できる関数の詳細なリストおよび説明については『Oracle Hyperion Financial Management 管理者ガイド』を参照。Planning および Essbase アプリケーションで使用できる関数の詳細なリストおよび説明については『Oracle Essbase テクニカル・リファレンス』を参照してください。)関数を使用して次のような計算を実行できます:

- 日付文字列から数字への変換
- 範囲をまたがったメンバーの平均値の計算
- ある期間における資産の減価償却の計算
- 時間ディメンションのメンバーの期間累計値の計算

Planning のデータ・フォーム・セルでオプションを入力するかわりに、スマート・リストを使用して事前定義済のオプションを選択します。

コンポーネントへのメンバーおよび関数の追加

式コンポーネント、スクリプト・コンポーネント、条件コンポーネント、メンバー範囲コンポーネントおよびデータ範囲コンポーネントにメンバーを追加します。

次も参照:

- [コンポーネントへのメンバーおよび関数の追加について](#)
- [1つ以上のディメンションから1つのディメンションへのメンバーまたは関数の追加](#)
- [1つ以上のディメンションから1つのディメンションへの複数のメンバーまたは関数の追加](#)
- [コンポーネントからのメンバーおよび関数の除去](#)
- [メンバー・セレクト内でのメンバーの検索](#)
- [メンバー・セレクトでの名前、別名またはプロパティによるメンバーの検索](#)

コンポーネントへのメンバーおよび関数の追加について

式コンポーネント、スクリプト・コンポーネント、条件コンポーネント、メンバー範囲コンポーネントおよびデータ範囲コンポーネントにメンバーを追加できます。式コンポーネント、スクリプト・コンポーネントおよび条件コンポーネントにメンバーのリストを戻す関数も追加できます。コンポーネントが属するアプリケーションのディメンションからメンバーおよび関数を選択できます。

操作するコンポーネントによって、1つのディメンションから1つ以上のメンバーまたは関数を選択するか、複数のディメンションから1つ以上のメンバーまたは関数を選択できます。

- 複数のディメンションに対して1つのメンバーを選択できる条件は次のとおりです:
 - 式コンポーネントの式グリッド内
 - 必要なパラメータが1つのメンバーである関数内
- タイプがメンバーである変数値を定義する際、複数のディメンションに対して複数のメンバーを選択できます。
- 1つのディメンションに対して複数のメンバーを選択できる条件は次のとおりです:
 - ビジネス・ルールのグローバル範囲内
 - 設計時プロンプト・タイプがデータ交差部であるテンプレート内
 - タイプがメンバーである変数値の定義内
 - タイプがメンバーである変数制限の定義内
 - ループ・コンポーネント内
 - パラメータがメンバーであるすべての関数内
- 1つのディメンションに対して1つのメンバーを選択できる条件は次のとおりです:
 - タイプがメンバーである変数値の定義内
 - パラメータがメンバーであるすべての関数内

1つ以上のディメンションから1つのディメンションへのメンバーまたは関数の追加

メンバー・セレクトによって、ディメンション内のメンバーおよび関数を選択できます。ディメンション内のメンバーを展開および縮小するときは、「+」および「-」を使用します。

「メンバー・セレクト」ダイアログ・ボックスには 3 つのタブがあり、選択したディメンションのすべてのメンバーおよび関数は「メンバー」タブおよび関数」タブに表示されます。3 つ目の「検索」タブは、メンバーまたはメンバーの説明を検索するために使用します。選択するメンバーおよび関数は右側の「選択」にリストされます。

1 つ以上のディメンションをコンポーネントに追加するには:

1. **システム・ビュー**でテンプレートまたはビジネス・ルールを右クリックし、「開く」を選択します。
2. ビジネス・ルールまたはテンプレートが開いたら、そのフロー・チャートで、メンバーを追加するコンポーネントを選択します。

 **ノート:**

固定ループ・コンポーネントにはメンバーを追加できません。

3. 「**メンバー・セレクト**」アイコンをクリックします。
4. 「**メンバー・セレクト**」で、「**ディメンション**」からディメンションを選択します。選択したディメンションのメンバーが「**メンバー**」に表示されます。
5. 次のいずれかのタスクを行います:
 - 「**メンバー**」でメンバーを選択し、**右矢印**ボタンをクリックして「**選択**」リストに移動します。この表のオプションを使用してさらに選択を定義することもできます。

表 8-1 「メンバー・セレクタ」 ボタン

ボタン	説明
特殊項目の追加	<p>(Oracle Hyperion Planning および Oracle Essbase ブロック・ストレージ・アプリケーション・ユーザーのみ) タブで選択したメンバーまたは関数に関連する追加のメンバーまたは関数を追加するには、次のいずれかを選択します:</p> <ul style="list-style-type: none"> - メンバー - 子 - 子(含む) - 子孫 - 子孫(含む) - 兄弟 - 兄弟(含む) - 親 - 親(含む) - 祖先 - 祖先(含む) - 相対 - レベル 0(ベース) - 包含 <p>(Essbase 集約ストレージ・アプリケーション・ユーザーのみ) 次のいずれかを選択して、タブで選択したメンバーまたは関数に関連するメンバーまたは関数を追加します:</p> <ul style="list-style-type: none"> - 兄弟 - レベル 0 の子孫
	<p> ノート:</p> <p>Planning の「特殊項目の追加」の選択肢には、「レベル 0(ベース)」または「含む」はありません。</p>
選択	選択すると、メンバーまたは関数が「選択」リストに移動します。
選択解除	選択すると、メンバーまたは関数が「選択」リストから除去されます。
すべて選択解除	選択すると、すべてのメンバーおよび関数が選択リストから除去されます。

- 「関数」で次のタスクを実行します。

Planning および Essbase ブロック・ストレージ・アプリケーション・ユーザーのみ:

- a. 関数を選択します。
- b. この表に従って、関数の必要な値を入力します:

表 8-2 関数および値

関数	パラメータ	説明
@ALLANCESTORS	メンバー名 — メンバー名を入力するか、「メンバー」をクリックしてメンバーを選択します。	選択項目を展開して、指定されたメンバーのすべての祖先を含めます。これには、共有メンバーの祖先も含まれます

表 8-2 (続き) 関数および値

関数	パラメータ	説明
@ANCEST	<ul style="list-style-type: none"> ディメンション名 — 「ディメンション」で選択したディメンション名 世代レベル番号 — 祖先値が戻る元の世代またはレベルの番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 メンバー名 — 任意のメンバー名またはメンバーの組合せ 	指定されたディメンション内で計算されている現在のメンバーの指定された世代またはレベルの祖先が戻されます。オプションのメンバー名を指定した場合、その祖先と、指定したメンバーが組み合わせられます。
@ANCESTORS	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ 世代レベル番号 — メンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 世代レベル名 — 選択にメンバーを含める最上位のレベル名と世代名 	選択項目を展開して、指定されたメンバーのすべての祖先を含めます
@ATTRIBUTE	属性メンバー名 — 入力したディメンションに対して、選択に含める属性メンバー名またはメンバーの組合せ	指定された属性メンバー(mbrName)に関連付けられているすべての基本メンバーのリストを生成します
@CHILDREN	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数	展開して、指定されたメンバーのすべての子を含めます
@CURRMBR	ディメンション名 — ディメンション名	指定されたディメンションの現在計算中のメンバーを戻します
@DESCENDANTS	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ 世代レベル番号 — メンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します 世代レベル名 — 選択にメンバーを含める最上位のレベル名と世代名 	展開して、指定されたメンバーのすべての子孫を含めます
@GENMBRS	<ul style="list-style-type: none"> ディメンション名 — ディメンション名 genName — dimName の世代名。正の整数は世代番号を表します 	展開して、指定されたメンバーのすべての子孫を含めます
@IALLANCESTORS	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数	選択項目を展開して、指定されたメンバーとそのすべての祖先を含めます。これには、共有メンバーとして指定されたメンバーすべての祖先も含まれます
@IANCESTORS	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数 世代レベル番号 — 選択にメンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 	選択項目を展開して、指定されたメンバーと、指定されたメンバーのすべての祖先または指定された世代やレベルまでの祖先を含めます
@ICHILDREN	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数。	展開して、指定されたメンバーとそのすぐ下のレベルのすべてのメンバーを含めます

表 8-2 (続き) 関数および値

関数	パラメータ	説明
@IDESCENDANTS	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数。 世代レベル番号 — 選択にメンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 	展開して、指定されたメンバーと、指定されたメンバーのすべての子孫または指定された世代やレベルまでの子孫を含めます
@ILSIBLINGS	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数	展開して、指定されたメンバーとそのメンバーの左側の兄弟をすべて含めます
@IRDESCENDANTS	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数 世代レベル番号 — 選択にメンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 	指定されたメンバーと、指定されたメンバーのすべての子孫またはオプションで指定された世代やレベルまでの子孫を戻します
@IRSIBLINGS	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数。	展開して、メンバーと指定されたメンバーの右側の兄弟をすべて含めます
@ISIBLINGS	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数	展開して、メンバーのすべての兄弟を含めます
@LEVELODESCENDANT	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数。	展開して、指定されたメンバーのすべてのレベル 0 の子孫を含めます
@LEVMBRS	<ul style="list-style-type: none"> ディメンション名 — ディメンション名 レベル名 — レベルの番号を定義するレベル名または整数値。整数値は 0 または正の整数にする必要があります。 	選択項目を展開して、指定されたディメンションの指定されたレベル番号またはレベル名を持つすべてのメンバーを含めます
@LIST	引数 — 親関数によって処理できるように、1 つの引数として収集して扱われる引数のリスト。引数として使用できるのは、メンバー名、メンバーの組合せ、メンバー・セット関数、範囲関数および数値式です。	リスト引数を必要とする関数によって処理されるリストを作成し、識別します
@LSIBLINGS	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数	展開して、指定されたメンバーの左側の兄弟をすべて含めます

表 8-2 (続き) 関数および値

関数	パラメータ	説明
@MATCH	<ul style="list-style-type: none"> メンバー、世代 — 選択の基準となるメンバーのデフォルト名またはユーザー定義名。システムは、指定されたメンバーおよびその子孫のメンバー名および別名を検索します。 genName — 選択の基準となる世代のデフォルト名またはユーザー定義名。システムは、その世代のすべてのメンバー名およびメンバー別名を検索します。 照合するパターン — ワイルドカード文字(*または?)を含めた検索対象の文字パターン。?は任意の 1 文字を表します。?はパターン内の任意の場所で使用できます。*は任意の数の文字を表します。使用できるのはパターンの末尾のみです。文字パターンにスペースを含めるには、二重引用符("")でパターンを囲みます。 	末尾のワイルドカード・メンバー選択を実行します
@MEMBER	文字列 — 二重引用符で囲まれた文字列、または文字列を戻す関数。	文字列として提供される名前を持つメンバーが戻されます
@MERGE	<ul style="list-style-type: none"> リスト 1 — マージするメンバーの第 1 リスト。 リスト 2 — マージするメンバーの第 2 リスト。 	別の関数によって処理される 2 つのメンバー・リストをマージします
@PARENT	<ul style="list-style-type: none"> ディメンション名 — ディメンション名 メンバー名 — メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数。 	指定されたディメンション内で計算されている現在のメンバーの親を戻します
@RANGE	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数 範囲リスト — メンバー名、メンバー名のカンマ区切りリスト、メンバー・セット関数または範囲関数。rangeList が指定されていない場合、システムは <i>Time</i> としてタグ付けされたディメンションのレベル 0 のメンバーを使用します。 	あるディメンション(mbrName)の指定されたメンバーと、別のディメンション(rangeList)の指定されたメンバー範囲が交差する、メンバー・リストを戻します
@RDESCENDANTS	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数 世代レベル番号 — メンバーを選択する下限の絶対世代番号または絶対レベル番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 	指定されたメンバーのすべての子孫または指定された世代やレベルまでの子孫を戻します

表 8-2 (続き) 関数および値

関数	パラメータ	説明
@RELATIVE	<ul style="list-style-type: none"> メンバー名 — メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数 世代レベル番号 — メンバーを選択する下限の絶対世代番号または絶対レベル番号を定義する整数値。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 	指定されたメンバーより上位または下位の指定された世代またはレベルのすべてのメンバーを選択できます
@REMOVE	<ul style="list-style-type: none"> リスト 1 — マージするメンバーの第 1 リスト。 リスト 2 — マージするメンバーの第 2 リスト。 	メッセージをデバッグ、情報、警告またはエラーとして戻します
@RSIBLINGS	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数	展開して、メンバーの右側の兄弟をすべて含めます
@SHARE	範囲リスト — メンバー、メンバーを戻す関数、またはメンバーの範囲のカンマ区切りリスト。rangeList のすべてのメンバーが同じディメンションからのものである必要があります。	rangeList の各メンバーをチェックして共有メンバーがあるかどうかを確認し、見つかった共有メンバーのリストを戻します
@SIBLING	メンバー名 — メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数	展開して、指定されたメンバーのすべての兄弟を含めます
@UDA	<ul style="list-style-type: none"> ディメンション名 — ユーザー定義属性に関連付けられているディメンション名 ユーザー定義属性 — データベース・アウトラインに表示されるユーザー定義属性の名前 	Essbase サーバーでユーザー定義属性 (UDA) として定義した共通属性に基づいてメンバーを選択します
@WITHATTR	<ul style="list-style-type: none"> ディメンション名 — 属性ディメンションの名前 演算子 — 引用符(" ")で囲んだ演算子の指定 値 — 満たす必要がある条件を演算子と組み合わせて定義する値。値として使用できるのは、属性メンバー仕様、定数または日付フォーマット関数(すなわち @TODATE)です。 	指定条件を満たす属性に関連付けられたすべてのメンバーを戻します
@XRANGE	<ul style="list-style-type: none"> メンバー名 1 — メンバー名、メンバーの組合せまたは単一のメンバーを戻す関数 メンバー名 2 — メンバー名、メンバーの組合せまたは単一のメンバーを戻す関数。mbrName1 がディメンションをまたぐメンバー(実績->1月など)である場合は、mbrName2 も同じにする必要があります、ディメンションの順序が mbrName1 で使用される順序と一致する必要があります。 	同じレベルの 2 つの指定された単一メンバー間またはディメンション間メンバー間(両端を含む)のメンバー範囲を戻します

Oracle Hyperion Financial Management、Planning および Essbase ブロック・ストレージのアプリケーションでサポートされる関数タイプのリストは、[関数の操作](#)を参照してください。

- c. **右矢印**ボタンをクリックして関数を「**選択**」リストに移動します。[表 1](#)のオプションを使用して選択項目を詳細に定義することもできます。
- 「**検索**」で、メンバーを検索するために次のタスクを行います:
 - a. 「**ディメンション**」から、メンバーを検索するディメンションを選択します。
 - b. 「**検索**」で、検索するメンバーのタイプ、メンバー名またはその説明を選択します。
 - c. 検索するメンバー名またはその説明を入力するか、ディメンションのすべてのメンバーを表示するため、デフォルトのワイルドカード(*)を受け入れます。
 - d. 「**検索**」を選択すると、フィールドに入力したメンバーが検索されます。[\(メンバー・セレクト内でのメンバーの検索を参照してください。\)](#)
 - e. 「**詳細検索**」を選択すると、詳細検索オプションにアクセスできます。[\(メンバー・セレクト内でのメンバーの検索を参照してください。\)](#)
 - f. メンバーを選択し、**右矢印**をクリックして「**選択**」リストに移動します。[表 1](#)のオプションを使用して選択項目を詳細に定義することもできます。
- **Financial Management ユーザーのみ**: 「**リスト**」では、次のタスクを行います:
 - a. 選択したディメンションのメンバー・リストを選択して、この表に従ってリストのパラメータを入力します:

表 8-3 Financial Management メンバー・リストおよびパラメータ

ディメンション	メンバー・リストおよびパラメータ
勘定科目	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」、「子」および「ベース」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - システムにはパラメータを入力しません。
エンティティ	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」、「子」および「ベース」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - 調整エンティティ および 親調整エンティティ には値を入力しません。

表 8-3 (続き) Financial Management メンバー・リストおよびパラメータ

ディメンション	メンバー・リストおよびパラメータ
ICP	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」、「子」および「ベース」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - システムには値を入力しません。
期間	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」および「子」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - First Generation、Second Generation、Third Generation、Fourth Generation、Fifth Generation および Sixth Generation には値を入力しません。
シナリオ	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」および「親」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - サポート・プロセス管理には値を入力しません。
値	<ul style="list-style-type: none"> - 「階層」および「子孫」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - 「入力」、「調整」、「合計」および「デフォルト通貨」には値を入力しません。
ビュー	<ul style="list-style-type: none"> - 「階層」および「子孫」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> - 最上位メンバー - アクティブなエンティティ - シナリオ - 年 - 期間

表 8-3 (続き) Financial Management メンバー・リストおよびパラメータ

ディメンション	メンバー・リストおよびパラメータ
年	<p>「階層」および「子孫」に対して、次のパラメータの値を入力します:</p> <ul style="list-style-type: none"> - 最上位メンバー - アクティブなエンティティ - シナリオ - 年 - 期間

- b. 右矢印をクリックしてメンバー・リストを「選択」リストに移動します。表 1 のオプションを使用して選択項目を詳細に定義することもできます。

6. 次のいずれかのタスクを行います:
- 次のいずれかのメンバー、関数またはメンバー・リストを選択する場合は、ステップ 7 に進みます。
 - タイプがメンバーである変数値
 - パラメータがメンバーである関数
 - 次のいずれかに対してメンバー、関数またはメンバー・リストを選択する場合は、各ディメンションのメンバー、関数またはメンバー・リストを選択するまで、前述の 2 つのステップを繰り返し、次のステップに進みます。
 - Planning または Essbase ビジネス・ルールのグローバル範囲
 - 設計時プロンプト・タイプがメンバー交差部であるテンプレート
 - タイプがメンバーである変数値
 - タイプがメンバーである変数制限
 - ループ・コンポーネント
 - パラメータがメンバーである関数
7. 「OK」をクリックします。

1 つ以上のディメンションから 1 つのディメンションへの複数のメンバーまたは関数の追加

メンバー・セレクタによって、ディメンション内のメンバーを選択できます。ディメンション内のメンバーを展開および縮小するときは、「+」および「-」を使用します。

「メンバー・セレクタ」ダイアログ・ボックスには 3 つのタブがあり、選択したディメンションのすべてのメンバーおよび関数は「メンバー」タブおよび関数」タブに表示されます。3 つ目の「検索」タブは、メンバーまたはメンバーの説明を検索するために使用します。選択するメンバーおよび関数は右側の「選択」にリストされます。

複数のメンバーが選択可能なコンポーネントの場合は、[Shift]キーを押しながらクリックするか、[Ctrl]キーを押しながらクリックして、連続または非連続のメンバーを選択できます。

1 つ以上のディメンションから複数のメンバーを 1 つのコンポーネントに追加するには:

1. **システム・ビュー**でテンプレートまたはビジネス・ルールを右クリックし、「開く」を選択します。
2. ビジネス・ルールまたはテンプレートが開いたら、そのフロー・チャートで、メンバーを追加するコンポーネントを選択します。

ノート:

固定ループ・コンポーネントにはメンバーを追加できません。

3. 「メンバー・セレクタ」アイコンをクリックします。
4. 「メンバー・セレクタ」で、「ディメンション」からディメンションを選択します。
選択したディメンションのメンバーが「メンバー」に表示されます。
5. 次のいずれかのタスクを実行して、メンバーを検索し選択します:
 - 「メンバー」で、[Ctrl]キーを押しながらクリックするか、または[Shift]キーを押しながらクリックしてメンバーを選択します。メンバーを「選択」リストに移動するには右矢印をクリックします。この表のオプションを使用して選択項目を詳細に定義することもできます。

表 8-4 「メンバー・セレクタ」ボタン

ボタン	説明
特殊項目の追加	<p>(Oracle Hyperion Planning および Oracle Essbase ブロック・ストレージ・アプリケーション・ユーザーのみ) 次のいずれかを選択して、タブで選択したメンバーまたは関数に関連するメンバーまたは関数をさらに追加します:</p> <ul style="list-style-type: none"> - メンバー - 子 - 子(含む) - 子孫 - 子孫(含む) - 兄弟 - 兄弟(含む) - 親 - 親(含む) - 祖先 - 祖先(含む) - 相対 - レベル 0(ベース) - 包含 <p>(Essbase 集約ストレージ・アプリケーション・ユーザーのみ) 次のいずれかを選択して、タブで選択したメンバーまたは関数に関連するメンバーまたは関数を追加します:</p> <ul style="list-style-type: none"> - 兄弟 - レベル 0 の子孫 <p>ノート: Planning の「特殊項目の追加」の選択肢には、「レベル 0(ベース)」または「含む」はありません。</p>
選択	選択すると、メンバーまたは関数が「選択」リストに移動します。
選択解除	選択すると、メンバーまたは関数が「選択」リストから除去されます。

表 8-4 (続き) 「メンバー・セレクト」 ボタン

ボタン	説明
すべて選択解除	選択すると、すべてのメンバーおよび関数が選択リストから除去されます。除

- 「関数」で、次のいずれかのタスクを実行します。

Planning および Essbase ブロック・ストレージ・アプリケーション・ユーザーのみ:

- 関数を選択するには、**[Ctrl]キーを押しながらクリック**するか、または**[Shift]キーを押しながらクリック**します。
- この表に従って、関数の必要な値を入力します:

表 8-5 関数および値

関数	入力する値	説明
@ALLANCESTORS	メンバー名	メンバー名を入力するか、「メンバー」をクリックしてメンバーを選択します。
@ANCEST	<ul style="list-style-type: none"> - ディメンション名 - 世代レベル番号 - メンバー名 	<ol style="list-style-type: none"> 「ディメンション」で選択したディメンション名を入力します。 祖先値が戻る元の世代またはレベルの番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 任意のメンバー名またはメンバーの組合せを入力します。
@ANCESTORS	<ul style="list-style-type: none"> - メンバー名 - 世代レベル番号 - 世代レベル名 	<ol style="list-style-type: none"> 任意のメンバー名またはメンバーの組合せを入力します。 メンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 選択にメンバーを含める最高のレベル名と世代名を入力します。
@ATTRIBUTE	属性メンバー名	入力したディメンションに対して、選択に含める属性メンバー名またはメンバーの組合せを入力します。
@CHILDREN	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@CURRMBR	ディメンション名	ディメンション名を入力します。
@DESCENDANTS	<ul style="list-style-type: none"> - メンバー名 - 世代レベル番号 - 世代レベル名 	<ol style="list-style-type: none"> 任意のメンバー名またはメンバーの組合せを入力します。 メンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。 選択にメンバーを含める最高のレベル名と世代名を入力します。
@GENMBRS	<ul style="list-style-type: none"> - ディメンション名 - genName 	<ol style="list-style-type: none"> ディメンション名を入力します。 dimName から世代名を入力します。正の整数は世代番号を表します。
@IALLANCESTORS	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。

表 8-5 (続き) 関数および値

関数	入力する値	説明
@IANCESTORS	<ul style="list-style-type: none"> - メンバー名 - 世代レベル番号 	<ul style="list-style-type: none"> i. メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。 ii. 選択にメンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。
@ICHILDREN	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@IDESCENDANTS	<ul style="list-style-type: none"> - メンバー名 - 世代レベル番号 	<ul style="list-style-type: none"> i. メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。 ii. 選択にメンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。
@ILSIBLINGS	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@IRDESCENDANTS	<ul style="list-style-type: none"> - メンバー名 - 世代レベル番号 	<ul style="list-style-type: none"> i. メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。 ii. 選択にメンバーを含める上限の絶対世代番号または絶対レベル番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。
@IRSIBLINGS	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@ISIBLINGS	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@LEVMBRS	<ul style="list-style-type: none"> - ディメンション名 - レベル名 	<ul style="list-style-type: none"> i. ディメンション名を入力します。 ii. レベルの番号を定義するレベル名または整数値を入力します。整数値は 0 または正の整数にする必要があります。
@LIST	引数	親関数によって処理できるように、1 つの引数として収集および処理される引数のリストを入力します。引数として使用できるのは、メンバー名、メンバーの組合せ、メンバー・セット関数、範囲関数および数値式です。
@LSIBLINGS	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@MATCH	<ul style="list-style-type: none"> - メンバー、世代 - genName - 照合するパターン 	<ul style="list-style-type: none"> i. 選択の基準となるメンバーのデフォルト名またはユーザー定義名を入力します。システムは、指定されたメンバーおよびその子孫のメンバー名および別名を検索します。 ii. 選択の基準となる世代のデフォルト名またはユーザー定義名を入力します。システムは、その世代のすべてのメンバー名およびメンバー別名を検索します。 iii. ワイルドカード(*または?)を含めた検索対象の文字パターンを入力します。?は任意の 1 文字を表します。?はパターン内の任意の場所で使用できます。*は任意の数の文字を表します。*を使用できるのはパターンの末尾のみです。文字パターンにスペースを含めるには、二重引用符("")でパターンを囲みます。

表 8-5 (続き) 関数および値

関数	入力する値	説明
@MEMBER	文字列	文字列(二重引用符で囲まれている)または文字列を戻す関数を入力します。
@MERGE	<ul style="list-style-type: none"> - リスト 1 - リスト 2 	<ul style="list-style-type: none"> i. マージするメンバーの第 1 リストを入力します。 ii. マージするメンバーの第 2 リストを入力します。
@PARENT	<ul style="list-style-type: none"> - ディメンション名 - メンバー名 	<ul style="list-style-type: none"> i. ディメンション名を入力します。 ii. メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数を入力します。
@RANGE	<ul style="list-style-type: none"> - メンバー名 - 範囲リスト 	<ul style="list-style-type: none"> i. メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数を入力します。 ii. メンバー名、メンバー名のカンマ区切りリスト、メンバー・セット関数または範囲関数を入力します。<code>rangeList</code> が指定されていない場合、システムは時間としてタグ付けされたディメンションのレベル 0 のメンバーを使用します。
@RDESCENDANTS	<ul style="list-style-type: none"> - メンバー名 - 世代レベル番号 	<ul style="list-style-type: none"> i. メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数を入力します。 ii. メンバーを選択する下限の絶対世代番号または絶対レベル番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。
@RELATIVE	<ul style="list-style-type: none"> - メンバー名 - 世代レベル番号 	<ul style="list-style-type: none"> i. メンバー名またはメンバーの組合せ、あるいは戻された親と組み合わせるメンバーまたはメンバーの組合せを戻す関数を入力します。 ii. メンバーを選択する下限の絶対世代番号または絶対レベル番号を定義する整数値を入力します。正の整数は世代番号を表します。0 または負の整数はレベル番号を表します。
@REMOVE	<ul style="list-style-type: none"> - リスト 1 - リスト 2 	<ul style="list-style-type: none"> i. マージするメンバーの第 1 リストを入力します。 ii. マージするメンバーの第 2 リストを入力します。
@RSIBLINGS	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@SHARE	範囲リスト	メンバーのカンマ区切りリスト、メンバーを戻す関数またはメンバーの範囲を入力します。 <code>rangeList</code> のすべてのメンバーが同じディメンションからのものである必要があります。
@SIBLINGS	メンバー名	メンバー名またはメンバーの組合せ、あるいはメンバーまたはメンバーの組合せを戻す関数を入力します。
@UDA	<ul style="list-style-type: none"> - ディメンション名 - ユーザー定義の属性 	<ul style="list-style-type: none"> i. ユーザー定義属性が関連付けられるディメンションの名前を入力します。 ii. データベース・アウトラインに表示されるユーザー定義属性の名前を入力します。

表 8-5 (続き) 関数および値

関数	入力する値	説明
@WITHATTR	<ul style="list-style-type: none"> - ディメンション名 - 演算子 - 値 	<ul style="list-style-type: none"> i. 属性ディメンションの名前を入力します。 ii. 演算子仕様を二重引用符(“”)で囲んで入力します。 iii. 満たす必要がある条件を演算子と組み合わせて定義する値を入力します。値として使用できるのは、属性メンバー仕様、定数または日付フォーマット関数(すなわち@TODATE)です。
@XRANGE	<ul style="list-style-type: none"> - メンバー名 1 - メンバー名 2 	<ul style="list-style-type: none"> i. メンバー名、メンバーの組合せまたは単一のメンバーを戻す関数を入力します。 ii. メンバー名、メンバーの組合せまたは単一のメンバーを戻す関数を入力します。mbrName1 がディメンションをまたぐメンバー(実績->1月など)である場合は、mbrName2 も同じにする必要があり、ディメンションの順序が mbrName1 で使用する順序と一致させる必要があります。

- c. **右矢印**をクリックして「**選択**」リストに移動します。表 1 のオプションを使用して選択項目を詳細に定義することもできます。
- 「**検索**」で、メンバーを検索するために次のタスクを行います:
 - a. 「**ディメンション**」から、メンバーを検索するディメンションを選択します。
 - b. 「**検索**」で、検索するメンバーのタイプ、メンバー名またはその説明を選択します。
 - c. 検索するメンバー名またはその説明を入力するか、ディメンションのすべてのメンバーを表示するため、デフォルトのワイルドカード(*)を受け入れます。
 - d. 「**検索**」を選択すると、フィールドに入力したメンバーが検索されます。(メンバー・セレクタ内でのメンバーの検索を参照してください。)
 - e. 「**詳細検索**」を選択すると、詳細検索オプションにアクセスできます。(メンバー・セレクタ内でのメンバーの検索を参照してください。)
 - f. メンバーを選択し、**右矢印**をクリックして「**選択**」リストに移動します。表 1 のオプションを使用して選択項目を詳細に定義することもできます。
- **Oracle Hyperion Financial Management ユーザーのみ:** 「**リスト**」で次のタスクを実行します:
 - a. 選択したディメンションのメンバー・リストを選択して、この表に従ってリストのパラメータを入力します:

表 8-6 Financial Management メンバー・リストおよびパラメータ

ディメンション	メンバー・リストおよびパラメータ
勘定科目	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」、「子」 および 「ベース」 に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - システムにはパラメータを入力しません。
エンティティ	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」、「子」 および 「ベース」 に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - 調整エンティティ および 親調整エンティティ には値を入力しません。
ICP	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」、「子」 および 「ベース」 に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - システムには値を入力しません。
期間	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」、「親」 および 「子」 に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - First Generation、Second Generation、Third Generation、Fourth Generation、Fifth Generation および Sixth Generation には値を入力しません。
シナリオ	<ul style="list-style-type: none"> - 「階層」、「子孫」、「祖先」 および 「親」 に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - サポート・プロセス管理には値を入力しません。

表 8-6 (続き) Financial Management メンバー・リストおよびパラメータ

ディメンション	メンバー・リストおよびパラメータ
値	<ul style="list-style-type: none"> - 「階層」および「子孫」に対して、次のパラメータの値を入力します: <ul style="list-style-type: none"> * 最上位メンバー * アクティブなエンティティ * シナリオ * 年 * 期間 - 「入力」、「調整」、「合計」および「デフォルト通貨」には値を入力しません。
ビュー	<p>「階層」および「子孫」に対して、次のパラメータの値を入力します:</p> <ul style="list-style-type: none"> - 最上位メンバー - アクティブなエンティティ - シナリオ - 年 - 期間
年	<p>「階層」および「子孫」に対して、次のパラメータの値を入力します:</p> <ul style="list-style-type: none"> - 最上位メンバー - アクティブなエンティティ - シナリオ - 年 - 期間

- b. 右矢印ボタンをクリックしてメンバー・リストを「選択」リストに移動します。
表 1 のオプションを使用して選択項目を詳細に定義することもできます。

6. 次のいずれかのタスクを行います:

- タイプがメンバーである変数値を定義するためにメンバー、関数またはメンバー・リストを選択する場合は、ディメンションごとにメンバー、関数またはメンバー・リストを選択するまで、前述の 2 つのステップを繰り返します。次のステップに進みます。
- これらのいずれかのメンバー、関数またはメンバー・リストを選択する場合は、次のステップに進みます。
 - ビジネス・ルールのグローバル範囲
 - 設計時プロンプト・タイプがメンバー交差部であるテンプレート
 - タイプがメンバーである変数値
 - タイプがメンバーである変数制限
 - ループ・コンポーネント
 - パラメータがメンバーである関数

7. 「OK」をクリックします。

コンポーネントからのメンバーおよび関数の除去

式、スクリプト、条件、メンバーおよびデータ範囲のコンポーネントからメンバーを除去できます。式、スクリプトおよび条件のコンポーネントから関数を除去できます。

コンポーネントからメンバーおよび関数を除去しても、データベースからは削除されません。共有コンポーネントからメンバーおよび関数を除去するには、最初にコンポーネントを非共有にする必要があります。

コンポーネントからメンバーまたは関数を除去するには:

1. **システム・ビュー**でテンプレートまたはビジネス・ルールを右クリックし、「**開く**」を選択します。
2. ビジネス・ルールまたはテンプレートが開いたら、そのフロー・チャートで、除去するメンバーまたは関数を含むコンポーネントを選択します。
3. 「**アクション**」を選択し、「**メンバー**」を選択します。
4. 「**選択**」で、ディメンションをクリックしてそのディメンションからすべてのメンバーを削除するか、メンバーをクリックしてそのメンバーのみをディメンションから削除します。
5. 「**選択**」からメンバーまたは関数を「**メンバー**」または「**関数**」に移動するには左矢印を使用します。表 1 を参照してください。
6. 複数のディメンションからメンバーを除去するには、「**次**」をクリックし、ステップ 3 からステップ 5 を繰り返します。
7. 「**OK**」、の順にクリックします。

メンバー・セレクタ内でのメンバーの検索

「メンバー・セレクタ」内でメンバーを検索できます。

メンバーを検索するには:

1. **システム・ビュー**でテンプレートまたはビジネス・ルールを右クリックし、「**開く**」を選択します。
2. ビジネス・ルールまたはテンプレートが開いたら、そのフロー・チャートで、メンバーを検索するコンポーネントを選択します。
3. 「**アクション**」をクリックして、「**メンバー**」を選択します。
4. 「**メンバー・セレクタ**」で、「**ディメンション**」から、検索するメンバーを含むディメンションを選択します。
ディメンション、その別名およびその数が、「**メンバー**」タブに表示されます。
5. ディメンションのメンバーを表示するには、次を選択します:

ノート:

デフォルトでは、ディメンションを選択するとアウトラインが縮小します。

- ディメンションの 1 つ下のレベルのメンバーを表示するには、「**展開**」を選択します。(「**縮小**」を選択するとディメンションのみを表示できます。)
 - ディメンションの下のすべてのメンバーを表示するには、「**すべて展開**」を選択します。(「**すべて縮小**」を選択するとディメンションのみを表示できます。)
6. 「**検索**」タブを選択します。
 - a. 「**検索**」で、検索するメンバーのタイプ、メンバー名またはその説明を選択します。
 - b. 検索するメンバー名またはその説明を入力するか、ディメンションのすべてのメンバーを表示するため、デフォルトのワイルドカード(*)を受け入れます。
 - c. 「**検索**」をクリックして、メンバーまたはその説明を検索します。
メンバーが検索されると「**結果**」に表示されます。アウトラインは表示されず、検出されたメンバーのみが表示されます。
 - d. **オプション:** メンバーをその名前、別名、またはそのプロパティの 1 つで検索するには、「**詳細検索**」をクリックします。[メンバー・セレクトタでの名前、別名またはプロパティによるメンバーの検索](#)を参照してください。
 - e. メンバーを選択し、**右矢印**をクリックして「**選択**」リストに移動します。[表 1](#)のオプションを使用して選択項目を詳細に定義することもできます。
 7. 「**OK**」をクリックします。

メンバー・セレクトタでの名前、別名またはプロパティによるメンバーの検索

「メンバー・セレクトタ」内の「詳細検索」を使用して、メンバーをその名前、別名、またはプロパティの 1 つで検索できます。

名前、別名、またはプロパティでメンバーを検索するには:

1. 「**メンバー・セレクトタ**」から、「**検索**」タブを選択して「**詳細検索**」をクリックします。
2. 「**メンバーの検索**」で、「**検索基準**」から、次のオプションのいずれかを選択します:
 - メンバーを名前で検索する場合、「**名前**」。ステップ 4 に進みます。
 - メンバーを別名で検索する場合、「**別名**」。ステップ 4 に進みます。
 - メンバーをプロパティの 1 つで検索する場合、「**プロパティ**」。ステップ 3 に進みます。
3. 「**プロパティ**」を選択した場合は、「**プロパティ名**」に入力するか選択します。
4. 名前、別名、またはプロパティの値を入力します。
5. 「**OK**」をクリックします。

別名、名前またはプロパティが見つかった場合は、「**結果**」に表示されます。アウトラインは表示されず、検出されたメンバーのみが表示されます。

 ノート:

別名でメンバーを検索する場合、他の言語の別名を持つメンバーを含む、検索条件に一致する別名を持つすべてのメンバーが「結果」に表示されます。アウトラインは表示されず、検出されたメンバーのみが表示されます。ただし、使用されている言語のメンバーの別名のみが「メンバー・セレクト」に表示されます。

6. **オプション:** 複数のメンバーが検索条件に一致する場合は、上および下矢印、または使用可能な場合にはスクロール・バーを使用して上下に移動し、検索条件に一致するすべてのメンバーを参照します。

変数の操作

ビジネス・ルールおよびテンプレートの設計時にコンポーネントで変数を使用します。

次も参照:

- [変数について](#)
- [変数の作成](#)
- [実行時プロンプト変数の入力](#)
- [変数の選択](#)
- [変数の編集](#)
- [変数の削除](#)
- [変数のリフレッシュ](#)
- [変数のコピー](#)
- [変数デザイナーでのテキストの検索と置換](#)
- [変数の使用状況の表示](#)
- [Financial Management アプリケーションへの事前定義済システム変数のロード \(Financial Management ユーザーのみ\)](#)

変数について

変数は、それらに対して定義する値を想定します。ビジネス・ルールおよびテンプレートの設計時にコンポーネントでこれらを使用します。

次の方法で変数を作成することができます:

-  をクリックして、変数デザイナーを起動します。
- ルール、スクリプト、式またはテンプレート内の、「変数セレクト」ダイアログ・ボックスを使用できる任意の場所から変数を作成します。
例:
 - ルールを開き、「メンバー範囲」コンポーネントにドラッグします。
 - ディメンションの横の  をクリックし、「**変数**」を選択します。

- 「**変数の選択**」ダイアログ・ボックスで、「**作成**」をクリックして変数を作成します。

変数を作成する場合、その変数は、選択したレベル(グローバル、アプリケーション、プラン・タイプまたはビジネス・ルール)で作成されます。同じ名前の変数が各レベルに作成された場合、最低レベルの変数がルールで使用されます。たとえば、vMonth という名前のグローバル変数と、Month という名前のプラン・タイプの変数を作成した場合、プラン・タイプの変数がルールで使用されます。

変数には、次の 2 つのタイプがあります:

- **実行** - ビジネス・ルールを起動すると、変数に対して定義された計算が実行されます。スクリプト・コンポーネントまたは固定ループのコンポーネントで実行変数を使用できます。

ノート:

実行変数は、Oracle General Ledger で使用されるアプリケーションも含めて、Oracle Essbase 集約ストレージ・アプリケーションでサポートされていません。

- **置換** - ビジネス・ルールの設計または起動時に、計算内で変数が置換されます。任意のコンポーネントで置換変数を使用できます。

いくつかのタイプの実行および置換変数を作成できます。作成できる変数は、アプリケーション・タイプと、実行変数と置換変数のどちらを作成するかに応じて異なります。

ノート:

非 Groovy ルールの場合、メンバーまたはメンバー(複数)タイプの変数のみが、集約ストレージ・オプション(ASO)タイプの Oracle Hyperion Planning キューブでサポートされているタイプの変数です。

ビジネス・ルールを起動する際にユーザーに情報を入力するよう求める変数を作成できます。これらの実行時プロンプト変数は、メンバー、テキスト、日付または数値などの情報の入力をユーザーに求めます。これらのプロンプトは、どのデータ型が必要かをユーザーに示します。

例:

- 月を選択します。
- 四半期ごとの予想顧客数を入力します。
- 翌月に予測される収益変化のパーセンテージを指定します。

変数を作成するアプリケーション・タイプに応じて、最大 4 つのデータベース・オブジェクトに変数を関連付けられます。変数は同時に複数のオブジェクトに存在し、各オブジェクトで同じ名前を使用できます。

変数の作成

変数を作成するには:

1. システム・ビュー、カスタム・ビュー、フィルタ・ビューまたはデプロイメント・ビューで、 をクリックします。
2. 「変数ナビゲータ」で、アプリケーション・タイプを展開します。
たとえば、**Planning** を展開します。
3. 変数を作成するレベルを選択します。
 - **グローバル** - 同じアプリケーション・タイプの任意のアプリケーションで使用できる変数を作成するには、<グローバル>を右クリックして、「新規」を選択します。
Oracle Hyperion Financial Management および Oracle Hyperion Planning ユーザー:
 - **アプリケーション** - そのアプリケーションのみで使用できる変数を作成するには、アプリケーションを右クリックして、「新規」を選択します。
 - **プランまたはデータベース** - そのプラン・タイプまたはデータベースのみで使用できる変数を作成するには、プラン・タイプまたはデータベースを右クリックして、「新規」を選択します。
Planning、Oracle General Ledger および Oracle Essbase ユーザーのみ:
 - **ビジネス・ルール** - そのビジネス・ルールのみで使用できる変数を作成するには、ビジネス・ルールを右クリックして、「新規」を選択します。
Planning、Oracle General Ledger および Essbase ユーザーのみ:
4. 作成する変数のタイプを選択します:
 - **置換** - ビジネス・ルールの設計または起動時に、変数は計算に置換できます。任意のコンポーネントで置換変数を使用できます。
置換変数を作成するには、「置換」タブで、「アクション」、「新規」の順に選択し、次の情報を入力します:
 - **名前** - 変数名
 - **説明** - 変数の説明
 - **グループ** - この変数をグループに含めるには、グループ名を入力します。グループ名は変数を保存およびリフレッシュした後で、「グループ」列の下に表示されます。
 - **タイプ** - ドロップダウンをクリックし、タイプを選択します。
 - **RTP** - 実行時プロンプト変数の場合:
 - * 「RTP」を選択して、変数が使用されるたびに表示されるテキストを入力します。
 - * 必要に応じて、「デフォルト値」を入力します。
 - * プロンプトに最後に入力された値を、次回プロンプトが表示されたときのデフォルト値として表示する場合、「最後に入力した値の使用」を選択します。
 - **実行** - ビジネス・ルールを起動すると、変数に対して定義された計算が実行されます。スクリプト・コンポーネントまたは固定ループで実行変数を使用できます

実行変数を作成するには、「実行」タブで、「アクション」、「新規」の順に選択し、次の情報を入力します:

- **名前** - 変数名
- **グループ** - この変数をグループに含めるには、グループ名を入力します。グループ名は変数を保存およびリフレッシュした後で、「グループ」列の下に表示されません。
- **値** - 変数の値
 - * 数値変数については、[数値変数の変数値の入力](#)を参照してください
 - * 文字列変数については、[文字列変数の変数値の入力](#)を参照してください
 - * メンバー範囲変数については、[メンバー変数またはメンバー\(複数\)変数の変数値の入力](#)を参照してください
 - * データ範囲変数の詳細は、[データ範囲変数の値の入力\(Financial Management ユーザーのみ\)](#)を参照してください
 - * ログ情報変数の詳細は、[ログ情報変数の値の入力\(Financial Management ユーザーのみ\)](#)を参照してください

5.  をクリックします

数値変数の変数値の入力

数値変数として、Oracle Hyperion Planning、Oracle Essbase、Oracle General Ledger または Oracle Hyperion Financial Management の置換変数を使用できます。数値変数として Financial Management の実行変数を使用することもできます。

数値変数の値を入力するには:

1. 「タイプ」で、「数値」を選択します。
2. スマート・リストを使用するには、アプリケーション、プラン・タイプまたはルール・レベルで変数を作成します。(グローバル・レベルではスマート・リストを使用できません。)「スマート・リスト」ボックスをクリックし、ドロップダウン・リストをクリックして「スマート・リスト」を選択します。

Planning ユーザーのみ

このリリースの *Oracle Planning and Budgeting Cloud Service Planning* の管理を参照してください。たとえば、値 1 から値 5、年次(1)、四半期ごと(2)、月次(3)、日次(4)および 1 時間ごと(5)を持つレポート・サイクルの整数のスマート・リストを設定できます。ユーザーが「月次」を選択する場合は数値 3 がデータベースに保管されます。これによりユーザーが数値を記憶する必要がなくなります。

スマート・リストの値としてテキスト文字列または日付を設定することもできます。

ノート:

スマート・リストの使用をサポートする Planning アプリケーションを選択する必要があります。

- 数値の実行時プロンプトを使用するには、「スマート・リスト」ボックスを空のままにして、次のステップに進みます。
- (Planning ユーザーのみ) 「制限」で、スマート・リストの制限を選択します。

 ノート:

実行時プロンプト値として Essbase 代替変数を使用し、その代替変数の値が変数制限外の場合、ルールを起動するとエラーが発生しないで正常に起動します。

- オプション: 変数のデフォルト値を入力します。
- デフォルトでは、「RTP」が選択されます。この変数の実行時プロンプトを作成しない場合は、「RTP」をクリアします。

Planning ユーザーのみ)

 ノート:

「RTP」を選択しない場合は、変数にデフォルト値を入力する必要があります。

- 「RTP」を選択した場合、ユーザーにデフォルト値として表示する実行時プロンプト・テキストを入力します。

Planning および Essbase ユーザーのみ

- 欠落したデータ値が許可されているかどうかを指定します。

Planning ユーザーのみ

-  をクリックします。

Financial Management の置換数値変数または実行数値変数に値を入力するには:

- 値を入力するか、「値」フィールド内をクリックして、「アクション」アイコンを表示します。
- 次のいずれかのタスクを行います:
 - メンバーを入力するには、「メンバー」を選択します。[コンポーネントへのメンバーおよび関数の追加](#)を参照してください。
 - 関数を入力するには、「関数」を選択します。[関数の操作](#)を参照してください。

-  をクリックします。

文字列変数の変数値の入力

文字列変数として、Oracle Hyperion Financial Management、Oracle Hyperion Planning、Oracle General Ledger または Oracle Essbase の置換変数を使用できます。文字列変数として Financial Management の実行変数を使用することもできます。文

字列変数は英数字で、255 文字以内にする必要があります。NULL 値を含められますが、値に前方&(アンパサンド)文字を含められません。

置換文字列変数の値を入力するには:

1. 「タイプ」で、「文字列」を選択します。
2. 「値」表で、変数に値を入力します。
3. デフォルトでは、「RTP」が選択されます。この変数の実行時プロンプトを作成しない場合は、「RTP」をクリアします。

Planning ユーザーのみ

ノート:

「RTP」を選択しない場合は、変数にデフォルト値を入力する必要があります。

4. 「RTP」を選択した場合、ユーザーに表示する実行時プロンプト・テキストを入力します。

Planning および Essbase ユーザーのみ

5.  をクリックします。

Financial Management の置換変数または実行文字列変数に値を入力するには:

1. 「タイプ」で、「文字列」を選択します。
2. 値を入力するか、「値」フィールド内をクリックして、「アクション」アイコンを表示します。
3. 次のいずれかのタスクを行います:
 - メンバーを入力するには、「メンバー」を選択します。[コンポーネントへのメンバーおよび関数の追加](#)を参照してください。
 - 関数を入力するには、「関数」を選択します。[関数の操作](#)を参照してください。
4.  をクリックします。

配列変数の値の入力

配列変数として、Oracle Hyperion Financial Management、Oracle Hyperion Planning、Oracle General Ledger または Oracle Essbase の実行変数を使用できます。配列には多次元の値のリストが含まれます。

通常、配列はメンバー式の一部として変数を格納するために使用されます。配列変数のサイズは、対応するディメンションのメンバー数によって決まります。たとえば、シナリオ・ディメンションに 4 つのメンバーがある場合、次のコマンドに 4 つのエントリを持つ Discount と呼ばれる配列が作成されます。一度に複数の配列を使用できます。

```
ARRAY Discount[Scenario];
```

配列変数に値を入力するには:

1. 「配列」チェック・ボックスを選択して、この実行変数を配列にします。「配列」を選択した後は、ドロップダウン・リストからディメンションを選択する必要があります。そうしないと、プラン・タイプのディメンションにデフォルト設定されます。

2. 「値」フィールドに、プラン・レベルで変数を作成する値を入力します。
3. オプション: 変数のグループを入力します。
4.  をクリックします。

メンバー範囲変数の値の入力

メンバー範囲変数として、Oracle Hyperion Financial Management、Oracle Hyperion Planning、Oracle General Ledger または Oracle Essbase の置換変数を使用できます。メンバー範囲変数には、メンバーの範囲を含める必要があります。

メンバー範囲変数の値を入力するには:

1. 「タイプ」で、「メンバー範囲」を選択します。
2. 変数グリッドで、Financial Management、Planning、Oracle General Ledger または Essbase のいずれのメンバー範囲を作成しているかに応じて、次のいずれかを行います。
 - a. Financial Management のメンバー範囲変数を作成する場合は、次の手順に従います:
 - i. メンバー選択を制限するディメンションを選択します。ディメンションを選択すると、メンバー・セレクトタにそのディメンションのメンバーのみが表示されます。ディメンションを選択しないと、メンバー・セレクトタにすべてのディメンションが表示されます。
 - ii. 「値」に、変数の値を入力するか、「アクション」アイコンを選択して、メンバー、変数、または変数値の関数を入力します。
 - 変数を入力する方法は、[変数の操作](#)を参照してください。
 - メンバーを入力する方法は、[コンポーネントへのメンバーおよび関数の追加](#)を参照してください。
 - 関数を入力する方法は、[関数の操作](#)を参照してください。
 - iii. 「変数」で、「変数」アイコンを選択して、メンバー範囲の変数を選択します。
 - iv. 「コメント」アイコンをクリックして、変数のコメントを入力します。
 - v. メンバー範囲を定義するディメンションごとにこれらのステップを繰り返します。
 - b. Planning、Oracle General Ledger または Essbase のメンバー範囲変数を作成する場合は、次の手順に従います:
 - i. メンバー範囲の制限を選択する表内の各ディメンションで、制限フィールド内をクリックし、制限を入力します。(表示されるディメンションは変数を作成しているアプリケーションに属するディメンションです)

グローバル・レベルで、「ディメンション・タイプ」オプションを選択すると、標準ディメンション・タイプのみが表示されます。「ディメンション名」オプションを選択した場合は、任意のディメンション名を入力できます。

 ノート:

(Planning および Essbase ユーザーのみ) 「制限」フィールドにテキストを入力する前に、RTP を選択する必要があります。

 ノート:

「制限」に関数を使用する場合、Essbase 関数のかわりに Planning 関数をお勧めします。場合によっては、Essbase 関数が必要なメンバーを戻さないことがあります。たとえば、Essbase 関数の評価に動的メンバーが含まれる場合です。メンバー範囲変数の「制限」については、Essbase 関数"@Relative("Mbr Name", 0)"のかわりに"ILvl0Descendants("Mbr Name")"を使用します。

- ii. メンバー範囲のデフォルト値を入力またはメンバー・セレクタを使用して選択するか、変数が実行時プロンプトの場合はデフォルト値を空にできます。表示されるディメンションごとに複数のメンバーおよび関数を選択できます。
- iii. **(Planning および Essbase ユーザーのみ)**変数を使用して起動時に情報を求めるプロンプトを表示する場合は、リストされたディメンションごとに「RTP」を選択します。

 ノート:

「RTP」を選択しない場合は、変数に値を入力する必要があります。

- iv. **(Planning および Essbase ユーザーのみ)** 「RTP」を選択したディメンションごとに、実行時プロンプト・テキストを入力します。
- v. グリッドの上の「RTP」テキスト・ボックスで、変数とそのディメンションに対して起動されるたびに表示する RTP テキストを入力します。

3.  をクリックします。

ディメンション間変数の変数値の入力

ディメンション間変数として、Oracle Hyperion Planning、Oracle General Ledger または Oracle Essbase の置換変数を使用できます。これには、ビジネス・ルールをディメンション間で起動できる複数のディメンションからのメンバーが含まれます。

ディメンション間変数の値を入力するには:

1. 「タイプ」で、「ディメンション間」を選択します。
2. **オプション:** 変数の制限を選択する表の各ディメンションについて、制限フィールドをクリックし、制限を入力します。(表示されるディメンションは変数を作成しているアプリケーションに属するディメンションです。)

 ノート:

- 「制限」フィールドにテキストを入力する前に、「RTP」を選択する必要があります。
- 「制限」に関数を使用する場合、Essbase 関数のかわりに Planning 関数をお勧めします。場合によっては、Essbase 関数が必要なメンバーを戻さないことがあります。たとえば、Essbase 関数の評価に動的メンバーが含まれる場合です。ディメンション間変数の「制限」については、Essbase 関数"@Relative("Mbr Name", 0)"のかわりに "lLvl0Descendants("Mbr Name")"を使用します。
- 関数を使用することは可能ですが、関数は有効な選択肢にする 1 つのメンバーをディメンションから戻す必要があります。

3. メンバー・セレクタに入力するかこれを使用して、変数の値を選択します。メンバーまたは関数を選択できます。

4. 変数に実行時プロンプトがある場合は、「RTP」を選択します。

Planning および Essbase ユーザーのみ

 ノート:

- 「RTP」を選択しない場合は、変数に値を入力する必要があります。
- 関数を使用することは可能ですが、関数は有効な選択肢にする 1 つのメンバーをディメンションから戻す必要があります。

5. 実行時プロンプト・テキストを入力します。

Planning および Essbase ユーザーのみ

6.  をクリックします。

ディメンション変数の変数値の入力

ディメンション変数として、Oracle Hyperion Planning、Oracle General Ledger または Oracle Essbase の置換変数を使用できます。この変数には、選択するディメンションが含まれます。

ディメンション変数の値を入力するには:

1. 「タイプ」で、「ディメンション」を選択します。
2. 変数グリッドで、ディメンションを選択します。表示されるディメンションは変数を作成するアプリケーションに属するディメンションです。
3. デフォルトでは、「RTP」が選択されます。変数に実行時プロンプトが必要ない場合は、「RTP」をクリアします。

「RTP」を選択しない場合は、変数に値を入力する必要があります。

4. (Planning および Essbase ユーザーのみ) 「RTP」を選択した場合、ユーザーに表示する実行時プロンプト・テキストを入力します。

5.  をクリックします。

メンバー変数またはメンバー(複数)変数の変数値の入力

メンバー変数およびメンバー(複数)変数として、Oracle Hyperion Planning、Oracle General Ledger および Oracle Essbase の置換変数を使用できます。これらの変数には、選択したディメンションから 1 つのメンバーまたは複数のメンバーが含まれます。

1 つまたは複数のメンバー変数の値を入力するには:

1. 「タイプ」で、「メンバー」または「メンバー(複数)」を選択します。
2. 変数グリッドで、ディメンションを選択します。表示されるディメンションは、変数を作成するアプリケーションに属するディメンションです。
3. メンバー・セレクタに入力するかこれを使用して、変数の制限を選択します。ステップ 2 で選択したディメンションからのみメンバーを選択できます。関数を選択することもできます。関数の操作を参照してください。

ノート:

「制限」に関数を使用する場合、Essbase 関数のかわりに Planning 関数をお勧めします。場合によっては、Essbase 関数が必要なメンバーを戻さないことがあります。たとえば、Essbase 関数の評価に動的メンバーが含まれる場合があります。メンバー変数またはメンバー(複数)変数の「制限」については、Essbase 関数 "@Relative("Mbr Name", 0)" のかわりに "ILvl0Descendants("Mbr Name")" を使用します。

4. メンバー・セレクタに入力するかこれを使用して、変数のデフォルト値を選択します。1 つのメンバー変数に対して 1 つのメンバーまたは関数を選択できます。また、1 つのメンバー変数に対して複数のメンバーおよび関数も選択できます。
5. デフォルトでは、「RTP」が選択されます。実行時プロンプト変数を作成しない場合は、「RTP」をクリアします。

Planning ユーザーのみ

ノート:

「RTP」を選択しない場合は、変数に値を入力する必要があります。

6. 「RTP」を選択した場合、ユーザーに表示する実行時プロンプト・テキストを入力します。
Planning および Essbase ユーザーのみ
7. Workforce Planning、Capital Asset Planning、Public Sector Budgeting および Project Financial Planning のプラン・タイプの Planning ユーザー: 「動的メンバーの作成」 および/または 「動的メンバーの削除」設定が有効なビジネス・ルールでメンバー・タイプ変数を使用している場合、「動的メンバーの親」列に、デフォルトの動的親メンバーを入力します。これらのオプションの両方を使用すると、Planning では、ルールの起動前にこの親のメンバーを動的に作成したり、ルールの起動後にこの親のメンバーを削除したりします。

 **ノート:**

Planning データベース・アウトライン内で、動的な子に対して動的メンバーの親を使用可能にする必要があります。この設定を使用可能にした後で、データベースをリフレッシュする必要があります。

8. 「ファイル」、「保存」の順に選択します。

パーセント変数の変数値の入力

パーセント変数として、Oracle Hyperion Planning、Oracle General Ledger および Oracle Essbase の置換変数を使用できます。パーセント変数として、Planning および Essbase ブロック・ストレージの実行変数を使用することもできます。この変数には、指定するパーセンテージが含まれます。

パーセント変数の値を入力するには:

1. 「タイプ」で、「パーセント」を選択します。
2. 「変数」グリッドで、「制限」をクリックして、変数の最小値および最大値を定義します。
3. 変数の数値を入力します。
4. デフォルトでは、「RTP」が選択されます。実行時プロンプトを作成しない場合は、「RTP」をクリアします。

Planning ユーザーのみ)

 **ノート:**

「RTP」を選択しない場合は、変数に値を入力する必要があります。

5. 「RTP」を選択した場合、ユーザーに表示する実行時プロンプト・テキストを入力します。

(Planning および Essbase ユーザーのみ)

6. 欠落しているデータ値を許可するかどうかを選択します。
7. 「ファイル」、「保存」を選択します。

整数変数の値の入力

整数変数として、Oracle Hyperion Planning、Oracle General Ledger および Oracle Essbase の置換変数を使用できます。

整数変数に値を入力するには:

1. 「タイプ」から、「整数」を選択します。
2. オプション: 変数の最大および最小の整数値を定義するには、「値」表で「制限」をクリックします。
3. オプション: デフォルト値をクリックし、変数に整数を入力します。

4. デフォルトでは、「RTP」が選択されます。この変数の実行時プロンプトを作成しない場合は、「RTP」をクリアします。
Planning ユーザーのみ
「RTP」を選択しない場合は、変数にデフォルト値を入力する必要があります。
5. 「RTP」を選択した場合、ユーザーに表示する実行時プロンプト・テキストを入力します。
Planning および Essbase ユーザーのみ
6. #Missing 値を許可するかどうかを選択します。
7. 「ファイル」、「保存」を選択します。

数値としての文字列変数の変数値の入力

数値としての文字列変数は、Oracle Hyperion Planning の置換変数または実行変数になり得ます。

数値としての文字列変数の値を入力するには:

1. 「タイプ」で、「数値としての文字列」を選択します。
2. オプション: 「値」表で、「制限」をクリックして、変数の最小値および最大値を定義します。最小値と最大値は YYYYMMDD の形式で数値として入力する必要があります。
3. オプション: 変数の数値を入力します。
4. デフォルトでは、「RTP」が選択されます。StringAsNumber 変数の「RTP」は必須であるため、「RTP」チェック・ボックスをクリアすることはできません。
5. #Missing 値を許可するかどうかを選択します。
6. 「最後に入力した値の使用」を選択して、最後に入力した値をユーザーが使用できるようにします。
7.  をクリックします。

数値としての日付変数の変数値の入力

数値としての日付変数は、Oracle Hyperion Planning の置換変数または実行変数になり得ます。

数値としての日付変数の値を入力するには:

1. 「タイプ」で、「数値としての日付」を選択します。
2. オプション: 「値」表で、「制限」をクリックして、変数の最小値および最大値を定義します。最小値と最大値は YYYYMMDD の形式で数値として入力する必要があります。
3. オプション: 変数の数値を入力します。dateasnumber 変数の場合は、YYYYMMDD の数値形式で日付を入力します。
4. デフォルトでは、「RTP」が選択されます。実行時プロンプトを作成しない場合は、「RTP」をクリアします。
「RTP」を選択しない場合は、変数に値を入力する必要があります。
5. 「RTP」を選択した場合、ユーザーに表示する実行時プロンプト・テキストを入力します。
6. #Missing 値を許可するかどうかを選択します。

7. 「**最後に入力した値の使用**」を選択して、最後に入力した値をユーザーが使用できるようにします。
8.  をクリックします。

データ範囲変数の値の入力(Financial Management ユーザーのみ)

データ範囲変数は、Oracle Hyperion Financial Management の置換変数です。選択するディメンションからのデータ値の範囲が含まれます。

データ範囲変数の値を入力するには:

1. 「**タイプ**」で、「**データ範囲**」を選択します。
2. 「**値**」表で、表示されるディメンションごとに、「**アクション**」アイコンを選択して変数の値を入力します。
 - 変数を入力する方法は、[変数の操作](#)を参照してください。
 - メンバーを入力する方法は、[コンポーネントへのメンバーおよび関数の追加](#)を参照してください。
 - 関数を入力する方法は、[関数の操作](#)を参照してください。
3. 「**変数**」で、「**変数**」アイコンをクリックして、データ範囲の変数を選択します。
4. 「**コメント**」アイコンをクリックして、変数のコメントを入力します。
5. データ範囲を定義するディメンションごとにこれらのステップを繰り返します。
6.  をクリックします。

ブール変数の値の入力(Financial Management ユーザーのみ)

ブール変数は、Oracle Hyperion Financial Management の実行変数です。ブール変数は TRUE または FALSE 値を持つ変数です。

ブール変数の値を入力するには:

1. 「**タイプ**」で、「**ブール**」を選択します。
2. 「**値**」に変数の値を入力します。
3.  をクリックします。

ログ情報変数の値の入力(Financial Management ユーザーのみ)

ログ情報変数は、Oracle Hyperion Financial Management の置換変数であります。

ログ情報変数の値を入力するには:

1. 「**タイプ**」で、「**ログ情報**」を選択します。
2. 「**値**」表で、ログ・テキストおよびログ式を入力するか、「**条件の追加/編集**」をクリックして条件ビルダーを使用します。[条件ビルダーを使用した条件文の作成](#)を参照してください。
3.  をクリックします。

実行時プロンプト変数の入力

ノート:

- Oracle Hyperion Calculation Manager では、Oracle Essbase ビジネス・ルールのスクリプトに RUNTIMESUBVARS セクションを追加することはできません。Calculation Manager で作成されたルール・レベルの実行時プロンプト変数が RUNTIMESUBVARS に変換されるのは、Essbase アプリケーションをデプロイする場合のみです。
- Administration Services、MaxL、または計算スクリプトを起動できる任意のコンポーネント内の実行時プロンプトを使用してビジネス・ルールを起動する場合、このビジネス・ルール内の実行時プロンプトにはデフォルト値が設定されている必要があります。
- Calculation Manager のルールの検証、デバッグ、デプロイ、分析または起動を行う場合、実行時プロンプト変数の値を入力または編集できます。ビジネス・ルールセットの検証やデプロイを行う場合にも、実行時プロンプト変数を入力または編集できます。実行時プロンプトにメンバー制限が含まれている場合、検証では有効なメンバー名のみを確認します(メンバーが制限内かどうかは検証されません)。数値および整数タイプの変数は実行時プロンプト制限に対して検証されます。
- 実行時プロンプトで Essbase ビジネス・ルールを Essbase にデプロイすると、Oracle Essbase Administration Services、MaxL または計算スクリプトを起動できるコンポーネントを使用してビジネス・ルールを起動できます。これらのビジネス・ルールをデプロイすると、これらは Essbase 計算スクリプトに変換され、実行時プロンプト変数は Essbase スクリプト内の RUNTIMESUBVARS に変換されます。
- メンバーの承認セキュリティを保持する実行時プロンプトの設計については、[Oracle Hyperion Planning 管理者ガイドの承認セキュリティの実行時プロンプトについて](#)を参照してください。

実行時プロンプト変数の値を入力するには:

1. ビジネス・ルールセットの検証、デバッグ、デプロイ、分析または起動を行うとき、またはビジネス・ルールセットの検証またはデプロイを行うとき、エラーがない場合は「**RTP 値の入力**」ダイアログ・ボックスが表示されます。

ビジネス・ルールの検証、デバッグ、デプロイまたは分析を行うとき、または、ビジネス・ルールセットの検証またはデプロイを行うとき、「RTP 値の入力」ダイアログが表示されるのは、ビジネス・ルール(またはビジネス・ルールセット)が使用している 1 つ以上の実行時プロンプト変数に、欠落した値がある場合のみです。すべての実行時プロンプト変数に値がある場合は、「RTP 値の入力」ダイアログは表示されません。

ビジネス・ルールを起動するときは、実行時プロンプト変数に値があるかどうかにかかわらず、起動するたびに「RTP 値の入力」ダイアログが表示されます。値がある場合、その値が「RTP 値の入力」ダイアログにデフォルトで表示されます。

2. リストされる実行時プロンプトごとに、値を入力するか 1 つを選択します。

3. **オプション:** ビジネス・ルールを使用している場合、「**ルールに値を適用**」チェック・ボックスを選択すると、指定した値が変数の値で動的に更新され、ルール・デザイナー内の「**変数**」タブの「**値**」列で表示できるようになります。
システム・ビューからビジネス・ルールを検証している場合、このチェック・ボックスは使用できません。
4. 「**OK**」をクリックします。
5. 検証エラーがある場合、それを修正し、実行時プロンプト変数の値を入力するタスクを繰り返します。

変数の選択

様々な場所から変数を選択できます。コンポーネント・デザイナー内からコンポーネントを作成する際、テンプレート・デザイナー内から設計時プロンプトを作成する際、あるいは Oracle Hyperion Calculation Manager 内のその他の場所から変数を選択できます。

変数を選択するには:

1. 次のいずれかのタスクを行います:
 - 変数を追加するコンポーネントが含まれるテンプレートを右クリックし、「**開く**」を選択します。
 - 変数を追加するコンポーネントが含まれるビジネス・ルールを右クリックし、「**開く**」を選択します。
2. ビジネス・ルールまたはテンプレートが開いたら、そのフロー・チャートで、変数を挿入するコンポーネントを選択します。
3. フロー・チャートの下のタブで、次のいずれかのタスクを行います:
 - メンバー範囲コンポーネントの場合は、ディメンション・フィールド内をクリックして「**アクション**」アイコンを選択し、「**変数**」を選択してメンバー範囲変数を選択します。または、「**変数セレクタ**」をクリックしてメンバー範囲変数を選択します。
 - データ範囲および固定ループの各コンポーネントの場合、「**変数**」フィールドで、「**変数**」アイコンを選択します。
 - 式コンポーネントの場合、「**アクション**」アイコンをクリックして、「**変数**」を選択します。
 - スクリプト・コンポーネントの場合、「**変数の挿入**」アイコンをクリックします。
 - 条件コンポーネントの場合、**条件ビルダー**を起動して、「**アクション**」アイコンをクリックし、「**変数**」を選択します。
4. 「**変数の選択**」で、次のいずれかのタスクを行います:
 - 新規変数を作成するには、「**作成**」をクリックして、**変数デザイナー**にアクセスします。[変数の作成](#)を参照してください。
 - 既存の変数を選択するには、「**カテゴリ**」から、使用する変数を含むレベルを選択します。次を選択できます:
 - **グローバル:** 変数はグローバル・レベルで作成され、このアプリケーション・タイプのすべてのアプリケーションで使用できます。

Oracle Hyperion Financial Management および Oracle Hyperion Planning ユーザーのみ

- アプリケーション: 変数はアプリケーション・レベルで作成され、このアプリケーションにのみ表示されます。
- プラン・タイプまたはデータベース: 変数はプラン・タイプまたはデータベース・レベルで作成され、このプラン・タイプまたはデータベースにのみ表示されます。

Planning、Oracle Fusion General Ledger および Oracle Essbase ユーザーのみ

- ルール: 変数はルール・レベルで作成され、それを作成した対象のルールにのみ表示されます。

Planning、General Ledger および Essbase ユーザーのみ)

ノート:

Planning および Essbase ユーザーのみ: メンバー範囲、メンバー・ブロック、式、固定ループおよび条件コンポーネントの場合、表示される変数はコンポーネントで使用されることが予想される変数のタイプに制限されるため、選択したスコープで使用可能なすべての変数がデフォルトで表示されません。選択したスコープで使用可能なすべての変数を参照するには、「すべての変数の表示」チェック・ボックスを選択します。

5. 「置換」または「実行」で、1 つまたは複数の変数を選択してそれをコンポーネントに挿入します。
6. 「OK」、 の順にクリックします。

変数の編集

変数デザイナーから変数の任意のプロパティを編集できます。変数に変更を加える際に、その変数がルールで使用されている場合は、ルールを開いて保存し、検証および再デプロイする必要があります。[検証およびデプロイ](#)を参照してください。

変数の削除

変数がコンポーネントまたはメンバー式で使用されない場合には、変数デザイナー内から削除できます。変数がコンポーネントで使用される場合は、変数を削除する前にコンポーネントから除去する必要があります。

変数を削除するには:

1. システム・ビューまたはフィルタ・ビューで、「変数デザイナー」アイコンを選択します。
2. 変数ナビゲータで、アプリケーション・タイプおよびアプリケーションを展開します。
3. 次のいずれかのタスクを行います:
 - 変数がグローバル変数である場合、**<グローバル>**を選択します。

Oracle Hyperion Financial Management および Oracle Hyperion Planning ユーザーのみ

- 変数がアプリケーション変数である場合は、変数が関連付けられているアプリケーションを選択します。
- 変数がプラン・タイプまたはデータベース変数である場合は、変数が関連付けられているプラン・タイプまたはデータベースを選択します。

Planning、Oracle Fusion General Ledger および Oracle Essbase ユーザーのみ

- 変数がビジネス・ルール変数の場合は、変数が関連付けられているビジネス・ルールを選択します。

Planning、General Ledger および Essbase ユーザーのみ

アプリケーション・タイプ、アプリケーション、計算タイプ、プラン・タイプまたはデータベース、およびビジネス・ルールに関連付けられている変数は、「置換」または「実行」に表示されます。

4. 「置換」または「実行」で、削除する変数を右クリックして、「削除」を選択します。
5. 「確認の削除」で、「はい」を選択して変数の削除を確認します。

変数のリフレッシュ

変数ナビゲータで変数のリストをリフレッシュして、変数を追加、削除、または変更を加えた後で、最新のリストを表示できます。

変数ナビゲータで、変数のリストをリフレッシュするには:

1. システム・ビューまたはフィルタ・ビューで、「変数デザイナー」アイコンを選択します。
2. 変数ナビゲータで、変数を作成、編集、または削除します。
3. 「置換」または「実行」タブの上で、「リフレッシュ」アイコンをクリックします。

変数のコピー

コピーおよび貼付けを使用して、同じ変数スコープや異なる変数スコープ(グローバル・レベル、アプリケーション・レベル、連結、プランまたはデータベース・レベル、ルール・レベル)に、変数をコピーできます。コピーしようとする変数と同じ名前の変数がコピー先の場所にある場合、コピーしようとする変数に新しい名前を付けるか、変数のコピーをスキップするか、変数のコンテンツを上書きすることができます。

変数をコピーして貼り付けるには:

1. システム・ビューまたはフィルタ・ビューで、「変数デザイナー」アイコンを選択します。
2. 変数ナビゲータで、次のタスクのいずれかを実行します:
 - **Oracle Hyperion Financial Management ユーザーのみ:** 連結を展開し、コピーする変数を含んでいるかどうかに応じて、アプリケーションまたは「グローバル」を選択します。
 - **Oracle Hyperion Planning ユーザーのみ:** 「Planning」を展開し、「グローバル」、またはコピーする変数に関連付けられているアプリケーション、プラン・タイプ、またはビジネス・ルールを選択します。

- **Oracle Essbase および Oracle Fusion General Ledger ユーザーのみ:** 「Essbase」を展開し、コピーする変数に関連付けられているアプリケーション、データベースまたはビジネス・ルールを選択します。

選択したオブジェクトに関連付けられた変数は、「置換」タブまたは「実行」タブ(あるいはその両方)に表示されます。

3. コピーする変数を右クリックし、「コピー」を選択します。
 4. コピーした変数を貼り付ける場所または変数を右クリックし、「貼付け」を選択します。(たとえば、Planning のグローバル変数をコピーする場合、Planning の別のグローバル変数としてコピーすることも、プラン・タイプの変数としてコピーすることもできます。)
- 変数をコピーして、同じ名前の変数が含まれる場所に貼り付けようとする、**「競合の解決」** ダイアログが表示されます。次のいずれかのタスクを行います:
 - 変数に新しい名前を付けます。(同じ場所に、同じ名前の変数を 2 つ含めることはできません。)
 - 変数のコピーをスキップするよう指定します。(コピーされた変数のコンテンツは、新しい場所に貼り付けられません。)
 - 変数を上書きするよう指定します。(コピーされた変数のコンテンツは、新しい場所に貼り付けられ、コピー先の変数のコンテンツは上書きされます。)
 - 変数をコピーして、同じ名前の変数が含まれない場所に貼り付けようとする、変数は新しい場所に貼り付けられます。

変数デザイナーでのテキストの検索と置換

変数デザイナーで、変数内のテキストを検索および置換できます。任意のスコープ(グローバル、アプリケーション、プランまたはデータベース、またはビジネス・ルール)の変数を検索できます。

変数デザイナーの検索機能で、変数の名前を入力することによって、変数を検索できます。変数のテキスト文字列を検索することもできます。(たとえば、変数で使用されるデフォルト値を検索できます。)

デフォルトでは、Oracle Hyperion Calculation Manager は、変数のデフォルト値、それに対して定義された制限および任意のプロンプト・テキストを使用して検索します。検索に変数の基本プロパティを含める場合、Calculation Manager では、変数名、グループおよび説明を使用して検索を行います。

変数またはテキスト文字列のすべてのインスタンスを置換することも、選択したインスタンスを置換することもできます。テキストを置換するときに、タイプがスマートリストまたは数値である変数が「制限」フィールドに含まれている場合、これらは置換操作から除外されます。これらのフィールドを置換すると、変数の定義が正しくない状態になる可能性があります(たとえば、指定した制限内にはない無効なスマート・リスト名またはデフォルト値を変数が持つ可能性があります)。

変数デザイナーでテキストを検索するには:

1. 任意のビューから、「**変数デザイナー**」アイコンをクリックします。
2. **変数ナビゲータ**で、検索対象のアプリケーション・タイプ、グローバル(Oracle Hyperion Planning および Oracle Hyperion Financial Management ユーザーのみ)、プラン・タイプまたはデータベース、またはビジネス・ルールを右クリックし、「**検索**」を選択します。
3. 「**任意のテキスト**」から、次を選択します:

- 「**次で始まる**」、指定する文字で始まる名前の変数のみを表示
 - 「**次で終わる**」、指定する文字で終わる名前の変数のみを表示
 - 「**次を含む**」、指定する文字を含む名前の変数のみを表示
 - **次と一致**」、指定する文字と一致する名前の変数のみを表示
4. 「**検索**」フィールドに、検索する変数のテキストを入力します。
5. 次のオプションの1つ以上を選択します:
- 検索するテキストの大文字と小文字の区別と、「**検索**」フィールドに入力するテキストの大文字と小文字の区別を一致させる必要がない場合は、「**大文字と小文字を区別しない**」を選択します。
 - 変数名、説明およびグループを検索に含めるには、「**基本プロパティを含める**」を選択します。

変数デザイナを起動したときに、「大文字と小文字を区別しない」および「基本プロパティを含める」はデフォルトで選択されています。これらのチェック・ボックスをクリアすると、これらのチェック・ボックスは変数デザイナを閉じてから再度開くまで選択が解除されたままになります。
 - 選択したノードの下位レベルの変数を検索するには、「**子スコープに変数を含める**」を選択します。アプリケーション・タイプ・レベル(たとえば、Oracle Essbase または Planning)で変数を検索する場合、このチェック・ボックスはデフォルトで選択されており、変更できません。このオプションが選択されている場合、アプリケーション、プラン・タイプ、集計タイプ、データベースおよびビジネス・ルールの変数が表示されます。変数名、説明、デフォルト値、グループおよび所有者の列に加えて、「アプリケーション」列、「プラン・タイプ」列および「ルール」列も変数のリストに表示されます。

アプリケーション・レベルで変数を検索する場合は、このチェック・ボックスを選択することもクリアすることもできます。このオプションを選択する場合、アプリケーション、プラン・タイプ、集計タイプ、データベースおよびビジネス・ルールの変数が表示されます。変数名、説明、デフォルト値、グループおよび所有者の列に加えて、「プラン・タイプ」列および「ルール」列も表示されます。

このオプションは、Planning でグローバル変数を検索する際には使用できません。グローバル変数を検索する場合、グローバル変数のみが表示されます。
6. 「**OK**」をクリックします。
- 検索対象のテキストが見つかった場合、テキストが見つかった変数が「置換」または「実行」タブにリストされます。
- テキストが見つかったら、テキストの1つ以上のインスタンスを置換できます。
- 変数デザイナでテキストを置換するには、次のいずれかのタスクを実行します。
- 選択したテキスト文字列のインスタンスを置換するには:
 1. テキスト文字列を置換する変数を選択します。
 2. 「**アクション**」、「**選択項目の置換**」の順に選択します。
 3. 「**選択項目の置換**」ダイアログ・ボックスで、「**置換後の文字列**」に、テキスト文字列を置換するテキストを入力します。
 4. 「**選択項目の置換**」をクリックします。
 - テキスト文字列のすべてのインスタンスを置換するには:

1. 「アクション」、「すべて置換」の順に選択します。
2. 「すべて置換」ダイアログ・ボックスで、「置換後の文字列」に、テキスト文字列を置換するテキストを入力します。
3. 「すべて置換」をクリックします。

 ノート:

テキスト文字列の検索時に選択したオプションはすべて、「選択項目の置換」および「すべて置換」ダイアログ・ボックスの「検索」領域でデフォルトで選択されており、変更できません。たとえば、テキスト文字列の検索時に選択した「大文字と小文字を区別しない」チェック・ボックスをクリアした場合、「大文字と小文字を区別しない」チェック・ボックスは「選択項目の置換/すべて置換」ダイアログ・ボックスで選択が解除されており、変更できません。

変数の使用状況の表示

変数を使用するビジネス・ルールを表示できます。変数の使用を表示すると、次の情報が表示されます:

- その変数を使用しているビジネス・ルールの名前
- その変数を使用しているビジネス・ルールのアプリケーション名
- その変数を使用しているビジネス・ルールの計算タイプ、プラン・タイプまたはデータベース
- その変数を使用しているビジネス・ルールの所有者
- その変数を使用しているビジネス・ルールがデプロイされているかどうか
- その変数を使用しているビジネス・ルールが検証されているかどうか
- その変数を使用しているビジネス・ルールの説明

変数の使用を表示するには:

1. システム・ビューまたはフィルタ・ビューから、「**変数デザイナー**」アイコンをクリックします。
2. 変数ナビゲータで、使用を確認する変数を含むデータベース・オブジェクトを選択します。そのオブジェクトに定義された変数は、変数デザイナーの「置換」タブおよび「実行」タブに表示されます。
3. 使用を確認する変数を右クリックし、「**使用状況の表示**」を選択します。
4. 情報を確認した後、「OK」をクリックします。

Financial Management アプリケーションへの事前定義済システム変数のロード (Financial Management ユーザーのみ)

Oracle Hyperion Financial Management システム・テンプレートを使用するには、まずシステム・テンプレートを使用するアプリケーションに事前定義済システム変数をロードする必要があります。これらのシステム変数は、システム・テンプレートを使用できるようにする実行および置換変数です。

 ノート:

これらのシステム変数を変更または削除しないでください。変更または削除すると、システム・テンプレートは指定どおりに動作しません。システム変数を誤って変更または削除した場合は、システム変数をアプリケーションにリロードできます。

事前定義済システム変数をアプリケーションにロードしたら、別のアプリケーションにコピーし、アプリケーション変数として使用できます。このようにコピーされたバージョンはシステム変数にリンクされていないため変更または削除できます。

事前定義済システム変数を **Financial Management** アプリケーションにロードするには:

1. システム・ビューまたはフィルタ・ビューから、「**変数デザイナー**」アイコンをクリックします。
2. 変数ナビゲータで、「**連結**」を展開し、アプリケーションを表示します。
3. システム・テンプレートを使用するアプリケーションを右クリックし、「**システム変数のロード**」を選択します。

システム変数がアプリケーションに正常にロードされたことを通知するメッセージが表示されます。ロード先のアプリケーションを選択すると、事前定義済システム変数がアプリケーション変数とともに「実行」および「置換」タブに表示されます。

 ノート:

システム・テンプレートを使用する各アプリケーションに対してこの手順を実行する必要があります。

関数の操作

関数は、データ値またはメンバーを戻すメンバー式を定義します。

たとえば、関数(および算術演算子や論理演算子)を使用して、指定したメンバーの兄弟、親、または子のリストを戻したり、指定した値より大きいまたは小さいデータ値のリストを戻したり、指定したメンバーからデータ値を割り当てられます。関数を選択すると、正しいパラメータを入力するようにプロンプト表示されます。

Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Essbase ブロック・ストレージのアプリケーションを使用している場合は、式、スクリプト、条件およびメンバー範囲の各コンポーネントで関数を使用します。Oracle General Ledger で使用されるアプリケーションも含め、Essbase 集約ストレージ・アプリケーションを使用している場合は、視点コンポーネントでのみ関数を使用します。

計算で使用できる関数は、Financial Management、Planning、Oracle General Ledger および Essbase アプリケーションでは異なります。

以下に Financial Management コンポーネントで使用できる関数のタイプのリストを示します。Financial Management の関数は、使用できるルールタイプによって分類

されます(**Financial Management** アプリケーションのコンポーネントで使用できる関数の、完全なリストおよび説明については、『**Oracle Hyperion Financial Management 管理者ガイド**』を参照してください。)

- 計算
- 換算
- 連結
- 割当て
- 動的計算
- トランザクション

Planning および **Essbase** ブロック・ストレージ・コンポーネントで使用できる関数のタイプのリストを次に示します。(関数の完全なリストおよび説明については、[Oracle Essbase テクニカル・リファレンス](#)を参照してください。)

- ブール
- リレーションシップ
- 計算演算子
- 制御フロー
- データの宣言
- 機能
- 算術
- メンバー・セット
- 範囲(財務)
- 割当て
- 予測
- 統計
- 日付と時間
- その他
- カスタム

ノート:

関数は、メンバー・セクタおよび関数セクタで使用できます。

Oracle General Ledger で使用されるアプリケーションも含めて、**Essbase** 集約ストレージ・アプリケーションおよびコンポーネントでは、メンバー・セット関数のみを使用します。(メンバー・セット関数の完全なリストおよび説明については、[Oracle Essbase テクニカル・リファレンス](#)を参照してください。)

Calculation Manager でサポートされている Essbase 関数

次の Oracle Essbase 関数は、ブロック・ストレージ・アプリケーションで Oracle Hyperion Calculation Manager によってサポートされています。

表 8-7 Calculation Manager でサポートされている Essbase 関数

@ABS	@ISANCEST	@MOVSUMX
@ACCUM	@ISATTRIBUTE	@NAME
@ALLANCESTORS	@ISCHILD	@NEXT
@ALIAS	@ISDESC	@NEXTS
@ALLOCATE	@ISGEN	@NEXTSIBLING
@ANCEST	@ISIANCEST	@NOTEQUAL
@ANCESTORS	@ISIBLINGS	@NPV
@ANCESTVAL	@ISICHILD	@PARENT
@ATTRIBUTE	@ISIDESC	@PARENTVAL
@ATTRIBUTEVAL	@ISIPARENT	@POWER
@ATTRIBUTESVAL	@ISISIBLING	@PREVSIBLING
@ATTRIBUTEVAL	@ISLEV	@PRIOR
@AVG	@ISMBR	@PRIORS
@AVGRANGE	@ISMBRUDA	@PTD
@BETWEEN	@ISMBRWITHATTR	@RANGE
@CALCMODE	@ISPARENT	@RANGEFIRSTVAL
@CHILDREN	@ISRANGENONEMPTY	@RANGELASTVAL
@COMPOUND	@ISSAMEGEN	@RANK
@COMPOUNDGROWTH	@ISSAMELEV	@RDESCENDANTS
@CONCATENATE	@ISSIBLING	@RELATIVE
@CORRELATION	@ISUDA	@RELXRANGE
@COUNT	@LANCESTORS	@REMAINDER
@CREATEBLOCK	@LDESCENDANTS	@REMOVE
@CURGEN	@LEV	@RETURN
@CURLEV	@LEVMBRS	@ROUND
@CURRMBR	@LIKE	@RSIBLINGS
@CURRMBRRANGE	@LIST	@SANCESTVAL
@DATEDIFF	@LN	@SHARE
@DATEPART	@LOG	@SHIFT
@DATEROLL	@LOG10	@SHIFTMINUS
@DECLINE	@LSIBLINGS	@SHIFTPLUS
@DESCENDANTS	@MATCH	@SHIFTSIBLING
@DISCOUNT	@MAX	@SIBLINGS
@ENUMVALUE	@MAXRANGE	@SLN
@EQUAL	@MAXS	@SPARENTVAL

表 8-7 (続き) Calculation Manager でサポートされている Essbase 関数

@EXP	@MAXSRANGE	@SPLINE
@EXPAND	@MBRCOMPARE	@STDEV
@FACTORIAL	@MBRPARENT	@STDEVP
@FORMATDATE	@MDALLOCATE	@STDEV RANGE
@GEN	@MDANCESTVAL	@SUBSTRING
@GENMBRS	@MDPARENTVAL	@SUM
@GROWTH	@MDSHIFT	@SUMRANGE
@IALLANCESTORS	@MEDIAN	@SYD
@IANCESTORS	@MEMBER	@TODATE
@ICHILDREN	@MEMBERAT	@TODATEEX
@IDESCENDANTS	@MERGE	@TODAY
@IALLANCESTORS	@MIN	@TREND
@ILDESCENDANTS	@MINRANGE	@TRUNCATE
@ILSIBLINGS	@MINS	@UDA
@INT	@MINSRANGE	@VAR
@INTEREST	@MOD	@VARPER
@INTERSECT	@MODE	@VARIANCE
@IRDESCENDANTS	@MOVAVG	@VARIANCEP
@IRR	@MOVMAX	@WITHATTR
@IRREX	@MOV MED	@XRANGE
@IRSIBLINGS	@MOV MIN	@XREF
@ISACCTYPE	@MOV SUM	@XWRITE

カスタム関数の操作

カスタム関数を使用して、データのコピーとエクスポート、テキスト文字列に対する一重引用符または二重引用符の削除と追加、2つのテキスト文字列の比較、別の形式への日付の変換などのタスクを実行します。

次も参照:

- [カスタム関数について](#)
- [年パラメータがあるカスタム関数の使用](#)
- [日付パラメータがある@CalcMgrExcel カスタム関数](#)
- [ビットに関する関数](#)
- [カウンタの関数](#)
- [日付/時刻の関数](#)
- [財務関数](#)
- [ログ関数](#)
- [数学関数](#)

- MaxL 関数
- MDX 関数
- 統計関数
- 文字列関数

カスタム関数について

カスタム関数には Oracle Hyperion Calculation Manager の条件、スクリプトおよび式コンポーネントの関数セレクタからアクセスします。

カスタム関数を使用すると、暗号化された MaxL スクリプトおよびファイルの起動、データのコピーとエクスポート、テキスト文字列に対する一重引用符または二重引用符の削除と追加、2 つの文字列の比較、YYYYMMDD 書式への日付の変換などのタスクを実行できます。

Oracle Essbase および Oracle Hyperion Planning のビジネス・ルール・コンポーネントでカスタム関数を使用できます。

年パラメータがあるカスタム関数の使用

一部のカスタム関数では、パラメータに年を選択肢を含むドロップダウンがある場合があります。アプリケーションに年という名前のディメンションまたはメンバーがある場合、年を選択肢を持つカスタム関数を使用することはできません。これは、ディメンション名またはメンバー名でもあるパラメータの選択肢(年だけでなく)の問題です。

この問題を回避するには、パラメータのドロップダウンから年を選択した後、ルールでそのまわりに@name()を追加し、@name(年)となるようにします。

日付パラメータがある@CalcMgrExcel カスタム関数

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。これらの関数を使用して、日付を YYYYMMDD 形式(シリアル形式)から Excel 形式に変換します:

- @CalcMgrExcelDATE は、特定の日付のシリアル値を返します。
次の例は、20181214 (YYYYMMDD 形式)を Excel の日付に変換します
`@CalcMgrExcelDATE(20181214)`
- @CalcMgrDateToExcel は、YYYYMMDD 形式の単一日付を Excel の日付に変換します
- @CalcMgrDatesToExcel は、YYYYMMDD 形式の複数の日付を Excel の日付に変換します

次の例では:

```
@CalcMgrDatesToExcel (@LIST ("Jan"->"Date_123", "Feb"->"Date_123"))
```

"Jan"->"Date_123"が 01/31/19 として表示され、"Feb"->"Date_123"が 02/31/19 として表示されます

ビットに関する関数

次も参照:

- [@CalcMgrBitAnd](#)
- [@CalcMgrBitOR](#)
- [@CalcMgrBitExOR](#)
- [@CalcMgrBitExBoolOR](#)
- [@CalcMgrBitCompliment](#)
- [@CalcMgrBitShiftLeft](#)
- [@CalcMgrBitShiftRight](#)
- [@CalcMgrBitUnsignedShiftRight](#)

@CalcMgrBitAnd

目的:

ビット単位の AND 演算を実行し、1 つ目のオペランドの各ビットを 2 つ目のオペランドの対応するビットと比較します。両方のビットが 1 の場合、その結果のビットは 1 に設定されます。そうではない場合、結果のビットは 0 に設定されます。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.AND(double,double)
```

CDF 仕様: @CalcMgrBitAnd(number1,number2)

@CalcMgrBitOR

目的:

ビット単位の OR 演算を実行し、1 つ目のオペランドの各ビットを 2 つ目のオペランドの対応するビットと比較します。いずれかのビットが 1 の場合、その結果のビットは 1 に設定されます。いずれも 1 ではない場合、結果のビットは 0 に設定されます。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.OR(double,double)
```

CDF 仕様: @CalcMgrBitOR(number1,number2)

@CalcMgrBitExOR

目的:

ビット単位の排他的 OR 演算を実行し、1 つ目のオペランドの各ビットを 2 つ目のオペランドの対応するビットと比較します。いずれかのビットが 1 の場合、その結果のビットは 1 に設定されます。いずれも 1 ではない場合、結果のビットは 0 に設定されます。

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.EXCLUSIVEOR(double,
double)

CDF 仕様: @CalcMgrBitExOR(number1,number2)

@CalcMgrBitExBoolOR

目的:

ビット単位の排他的 OR 演算を実行し、1 つ目のオペランドの各ビットを 2 つ目のオペランドの対応するビットと比較します。いずれかのビットが 1 の場合、その結果のビットは 1 に設定されます。いずれも 1 ではない場合、結果のビットは 0 に設定されます。

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.EXCLUSIVEOR(double,
double)

CDF 仕様: @CalcMgrBitExOR(number1,number2)

@CalcMgrBitCompliment

目的:

ビット単位の単項補数演算を実行し、各ビットを反転させます。

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.UNARYCOMPLIMENT(double)

CDF 仕様: @CalcMgrBitCompliment(number1)

@CalcMgrBitShiftLeft

目的:

符号付き左シフトを実行します。

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.SIGNEDSHIFTLEFT(double,
double)

CDF 仕様: @CalcMgrBitShiftLeft(number1,number2)

@CalcMgrBitShiftRight

目的:

符号付き右シフトを実行します。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.SIGNEDSHIFTRIGHT(double, double)
```

CDF 仕様: @CalcMgrBitShiftRight(number1,number2)

@CalcMgrBitUnsignedShiftRight

目的:

符号なし右シフトを実行します。

構文:

```
com.hyperion.calcmgr.common.excel.cdf.BitwiseFunctions.UNSIGNEDSHIFTRIGHT(double, double)
```

@CalcMgrBitUnsignedShiftRight(number1,number2)

カウンタの関数

次も参照:

- [@CalcMgrCounterAddNumber](#)
- [@CalcMgrCounterAddText](#)
- [@CalcMgrCounterClear](#)
- [@CalcMgrCounterClearAll](#)
- [@CalcMgrCounterClearKey](#)
- [@CalcMgrCounterDecrement](#)
- [@CalcMgrCounterDecrementKey](#)
- [@CalcMgrCounterGetKeyNumber](#)
- [@CalcMgrCounterGetKeyText](#)
- [@CalcMgrCounterGetNumber](#)
- [@CalcMgrCounterGetText](#)
- [@CalcMgrCounterIncrement](#)
- [@CalcMgrCounterIncrementKey](#)
- [@CalcMgrCounterUpdate](#)
- [@CalcMgrCounterUpdateNumber](#)
- [@CalcMgrCounterUpdateNumberText](#)
- [@CalcMgrCounterUpdateText](#)

@CalcMgrCounterAddNumber

目的:

カウンタに数値を加え、キーを返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.addNumber(double)`

CDF 仕様: `@CalcMgrCounterAddNumber(number)`

@CalcMgrCounterAddText

目的:

カウンタにテキスト文字列を加え、キーを返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.addText(String)`

CDF 仕様: `@CalcMgrCounterAddText(text)`

@CalcMgrCounterClear

目的:

キーによって指定されたカウンタをクリアします

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.CounterFunctions.clear(double)`

CDF 仕様: `@CalcMgrCounterClear(key)`

@CalcMgrCounterClearAll

目的:

すべてのキーと値をカウンタから削除します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.CounterFunctions.clearAll()`

CDF 仕様: `@CalcMgrCounterClearAll()`

@CalcMgrCounterClearKey

目的:

キーに関連付けられているカウンタから値を削除します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.clearKey(String)`

CDF 仕様: `@CalcMgrCounterClearKey(key)`

@CalcMgrCounterDecrement

目的:

キーに基づいたカウンタの値を減分します。キーが見つからない場合、値ゼロがキーに設定されます

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.CounterFunctions.decrement(double)`

CDF 仕様: `@CalcMgrCounterDecrement(key)`

@CalcMgrCounterDecrementKey

目的:

キーに基づいたカウンタの値を減分します。キーが見つからない場合、値ゼロがキーに設定されます

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.CounterFunctions.decrementKey(String)`

CDF 仕様: `@CalcMgrCounterDecrementKey(key)`

@CalcMgrCounterGetKeyNumber

目的:

キーに基づいたカウンタで検出されたテキストを返します。キーが見つからない場合、欠落値を返します。

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.CounterFunctions.getKeyNumber(String, double)`

CDF 仕様: `@CalcMgrCounterGetKeyNumber(key, missing_value)`

@CalcMgrCounterGetKeyText

目的:

キーに基づいたカウンタで検出されたテキストを返します。キーが見つからない場合、欠落値を返します。

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.CounterFunctions.getKeyText(String, String)`

CDF 仕様: `@CalcMgrCounterGetKeyText(key, missing_value)`

@CalcMgrCounterGetNumber

目的:

キーによって指定されたカウンタから数値を返します。キーが見つからない場合または値が数値でない場合、欠落値を返します。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.getNumber(double, double)`

CDF 仕様: `@CalcMgrCounterGetNumber(key, missingValue)`

@CalcMgrCounterGetText

目的:

キーに基づいたカウンタで検出されたテキストを返します。キーが見つからない場合、欠落値を返します。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.getText(double, String)`

CDF 仕様: `@CalcMgrCounterGetText(key, missing_value)`

@CalcMgrCounterIncrement

目的:

キーによって指定されたカウンタの値を増分します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.increment(double)`

CDF 仕様: `@CalcMgrCounterIncrement(key)`

@CalcMgrCounterIncrementKey

目的:

キーに基づいたカウンタの値を増分します。キーが見つからない場合、値ゼロがキーに設定されます。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.incrementKey(String)`

CDF 仕様: `@CalcMgrCounterIncrementKey(key)`

@CalcMgrCounterUpdate

目的:

指定したキーを使用してカウンタの数値を設定します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.update(double, double)`

CDF 仕様: `@CalcMgrCounterUpdate(key, number)`

@CalcMgrCounterUpdateNumber

目的:

指定したキーを使用してカウンタの数値を更新します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.updateNumber(String, double)`

CDF 仕様: `@CalcMgrCounterUpdateNumber(key, number)`

@CalcMgrCounterUpdateNumberText

目的:

指定したキーを使用してカウンタの数値を更新します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.updateNumberText(double, String)`

CDF 仕様: `@CalcMgrCounterUpdateNumberText(key, number)`

@CalcMgrCounterUpdateText

目的:

指定したキーを使用してカウンタのテキストを更新します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CounterFunctions.updateText(String, String)`

CDF 仕様: `@CalcMgrCounterUpdateText(key, text)`

日付/時刻の関数

次も参照:

- [@CalcMgrAddDate](#)
- [@CalcMgrAddDatePart](#)
- [@CalcMgrAddDays](#)
- [@CalcMgrAddMonths](#)
- [@CalcMgrAddWeeks](#)
- [@CalcMgrAddYears](#)
- [@CalcMgrDateDiff](#)
- [@CalcMgrDateToExcel](#)
- [@CalcMgrDatesToExcel](#)
- [@CalcMgrDateTimeToExcel](#)
- [@CalcMgrDateTimesToExcel](#)
- [@CalcMgrDateToString](#)
- [@CalcMgrDaysBetween](#)
- [@CalcMgrDaysDiff](#)
- [@CalcMgrDiffDate](#)
- [@CalcMgrExcelADD](#)
- [@CalcMgrExcelDATE](#)
- [@CalcMgrExcelDATEDIF](#)
- [@CalcMgrExcelDAYOFYEAR](#)
- [@CalcMgrExcelDAYS360](#)
- [@CalcMgrExcelDAYSINMONTH](#)
- [@CalcMgrExcelEOMONTH](#)
- [@CalcMgrExcelHOUR](#)
- [@CalcMgrExcelMINUTE](#)
- [@CalcMgrExcelMONTH](#)
- [@CalcMgrExcelNETWORKDAYS](#)
- [@CalcMgrExcelSECOND](#)
- [@CalcMgrExcelToDate](#)
- [@CalcMgrExcelToDateTime](#)
- [@CalcMgrExcelWEEKNUM](#)
- [@CalcMgrExcelWEEKDAY](#)
- [@CalcMgrExcelWORKDAY](#)
- [@CalcMgrExcelYEAR](#)

- @CalcMgrExcelYEARFRAC
- @CalcMgrGetCurrentDate
- @CalcMgrGetCurrentDateTZ
- @CalcMgrGetCurrentDateTime
- @CalcMgrGetCurrentDateTimeTZ
- @CalcMgrGetCustomDate
- @CalcMgrGetCustomDateTime
- @CalcMgrGetDatePart
- @CalcMgrGetDateTimePart
- @CalcMgrGetDay
- @CalcMgrGetDayOfYear
- @CalcMgrGetFormattedDate
- @CalcMgrGetMaxDaysInMonth
- @CalcMgrGetMonth
- @CalcMgrGetStringFormattedDateTime
- @CalcMgrGetWeekOfMonth
- @CalcMgrGetWeekOfYear
- @CalcMgrGetYear
- @CalcMgrIsLeapYear
- @CalcMgrMonthsBetween
- @CalcMgrMonthsDiff
- @CalcMgrRollDate
- @CalcMgrRollDay
- @CalcMgrRollMonth
- @CalcMgrRollYear
- @CalcMgrWeeksBetween
- @CalcMgrWeeksDiff
- @CalcMgrYearsBetween
- @CalcMgrYearsDiff

@CalcMgrAddDate

目的:

指定した年数、月数および日数を YYYYMMDD 形式の日付に追加します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.addDate(int,int,int,int)
```

CDF 仕様: @CalcMgrAddDate(date, years, months, days)

@CalcMgrAddDatePart

目的:

指定した年数/月数/日数/週数を YYYYMMDD 形式の日付に追加します。"date_part"は、"day"、"month"、"week"、"year"のいずれかです

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.addDatePart (int,String,int)`

CDF 仕様: `@CalcMgrAddDatePart (date,date_part, amountToAdd)`

@CalcMgrAddDays

目的:

指定した日数を YYYYMMDD 形式の日付に追加します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.addDays (int,int)`

CDF 仕様: `@CalcMgrAddDays (date, daysToAdd)`

@CalcMgrAddMonths

目的:

指定した月数を YYYYMMDD 形式の日付に追加します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.addMonths (int,int)`

CDF 仕様: `@CalcMgrAddMonths (date, monthsToAdd)`

@CalcMgrAddWeeks

目的:

指定した週数を YYYYMMDD 形式の日付に追加します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.addWeeks (int,int)`

CDF 仕様: `@CalcMgrAddWeeks (date, weeksToAdd)`

@CalcMgrAddYears

目的:

指定した年数を YYYYMMDD 形式の日付に追加します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.CalendarFunctions.addYears(int,int)`

CDF 仕様: `@CalcMgrAddYears(date, yearsToAdd)`

@CalcMgrDateDiff

目的:

標準のグレゴリオ暦に従い、指定された日付部分について、YYYYMMDD 形式の 2 つの入力日付の差異(数値)を返します

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.CalendarFunctions.dateDiff(int,int,String)`

CDF 仕様: `@CalcMgrDateDiff(fromDate,toDate,datePart)`

 **ノート:**

@CalcMgrDateDiff が返すのは正の数のみです。負の数が必要な場合に負の数を返すには、[@CalcMgrDiffDate](#) を使用します。

@CalcMgrDateToExcel

目的:

YYYYMMDD 形式の単一の日付を Excel の日付に変換します

構文:

Java クラス:

`com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DATE_TOEXCEL(double)`

CDF 仕様: `@CalcMgrDateToExcel(date)`

@CalcMgrDatesToExcel

目的:

YYYYMMDD 形式の複数の日付を Excel の日付に変換します

構文:

```
Java クラス:  
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DATES_TOEXCEL(double[])  
  
CDF 仕様: @CalcMgrDatesToExcel (dates)
```

@CalcMgrDateTimeToExcel

目的:

YYYYMMDDHHMMSS 形式の単一の日付を **Excel** の日付に変換します

構文:

```
Java クラス:  
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DATETIME_TOEXCEL(double)  
  
CDF 仕様: @CalcMgrDateTimeToExcel (date)
```

@CalcMgrDateTimesToExcel

目的:

YYYYMMDDHHMMSS 形式の複数の日付を **Excel** の日付に変換します

構文:

```
Java クラス:  
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DATETIMES_TOEXCEL(double[])  
  
CDF 仕様: @CalcMgrDateTimesToExcel (dates)
```

@CalcMgrDateToString

目的:

YYYYMMDD 形式の日付を、指定した形式を使用して文字列として返します。形式については、**Java** ドキュメントの `SimpleDateFormat` を参照してください

構文:

```
Java クラス:  
com.hyperion.calcmgr.common.cdf.CalendarFunctions.dateToString(int, String)  
  
CDF 仕様: @CalcMgrDateToString (date, format)
```

@CalcMgrDaysBetween

目的:

YYYYMMDD 形式の 2 つの日付の間の日数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.daysBetween(int,int)`
CDF 仕様: `@CalcMgrDaysBetween(fromDate,toDate)`

 **ノート:**

`@CalcMgrDaysBetween` が返すのは正の数のみです。負の数が必要な場合に負の数を返すには、[@CalcMgrDaysDiff](#) を使用します。

@CalcMgrDaysDiff

目的:

YYYYMMDD 形式の 2 つの日付の間の日数を返します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.CalendarFunctions.daysDiff(int,int)`
CDF 仕様: `@CalcMgrDaysDiff(fromDate,toDate)`

 **ノート:**

`@CalcMgrDaysDiff` を使用するとき、関数で最初の日付が 2 番目の日付より後になっている場合は、負の数が返されます。関数で最初の日付が 2 番目の日付より前になっている場合は、正の数が返されます。正の数のみを返す場合には、[@CalcMgrDaysBetween](#) を使用します。

@CalcMgrDiffDate

目的:

標準のグレゴリオ暦に従い、指定された日付部分について、YYYYMMDD 形式の 2 つの入力日付の差異(数値)を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.diffDate(int,int,String)`
CDF 仕様: `@CalcMgrDiffDate(fromDate,toDate,datePart)`

 ノート:

@CalcMgrDiffDate を使用するとき、関数で最初の日付が 2 番目の日付より後になっている場合は、負の数が返されます。関数で最初の日付が 2 番目の日付より前になっている場合は、正の数が返されます。正の数のみを返す場合には、[@CalcMgrDateDiff](#) を使用します。

@CalcMgrExcelADD

目的:

日付への加算を行います

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.ADD(double, double, String)
```

CDF 仕様: @CalcMgrExcelADD(date, amount, what)

@CalcMgrExcelDATE

目的:

特定の日付のシリアル値を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DATE(double, double, double)
```

CDF 仕様: @CalcMgrExcelDATE(year, month, day)

 ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある[@CalcMgrExcel カスタム関数](#)を参照してください

@CalcMgrExcelDATEDIF

目的:

2 つの日付の間の日数、月数、年数を計算します。

年齢を計算する必要のある式で有用です

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DATEDIF (double, double, String)

CDF 仕様: @CalcMgrExcelDATEDIF (start_date, end_date, unit)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelDAYOFYEAR

目的:

シリアル値を通日に変換します

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DAYOFYEAR (double)

CDF 仕様: @CalcMgrExcelDAYOFYEAR (Date)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelDAYS360

目的:

1 年を 360 日として 2 つの日付の間の日数を計算します

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DAYS360 (double, double, boolean)

CDF 仕様: @CalcMgrExcelDAYS360 (start_date, end_date, method)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelDAYSINMONTH

目的:

シリアル値をその月の日数に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.DAYSINMONTH(double)
```

CDF 仕様: @CalcMgrExcelDAYSINMONTH(date)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelEOMONTH

目的:

指定した月数だけ前または後の月の最終日に対応するシリアル値を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.EOMONTH(double, double)
```

CDF 仕様: @CalcMgrExcelEOMONTH(dateValue, adjustmentMonths)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelHOUR

目的:

シリアル値を時刻に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.HOUR(double)
```

CDF 仕様: @CalcMgrExcelHOUR(date)

@CalcMgrExcelMINUTE

目的:

シリアル値を分に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.MINUTE(double)
```

CDF 仕様: @CalcMgrExcelMINUTE(date)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelMONTH

目的:

シリアル値を月に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.MONTH(double)
```

CDF 仕様: @CalcMgrExcelMONTH(Date)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelNETWORKDAYS

目的:

2 つの日付の間の稼働日の日数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.NETWORKDAYS(double, double, double[])
```

CDF 仕様: @CalcMgrExcelNETWORKDAYS(startDate, endDate, holidays)

 **ノート:**

日付を Oracle Hyperion Planning の日付形式で渡している場合は、[@CalcMgrDateToExcel](#) を使用して Planning の日付を Excel の日付に変換する必要があります。

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelSECOND

目的:

シリアル値を秒に変換します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.SECOND(double)`

CDF 仕様: @CalcMgrExcelSECOND (date)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelToDate

目的:

Excel の日付を YYYYMMDD 形式に変換します。

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.EXCEL_TODATE(double)`

CDF 仕様: @CalcMgrExcelToDate (excel_date)

@CalcMgrExcelToDateTime

目的:

Excel の日付を YYYYMMDDHHMMSS 形式に変換します。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.EXCEL_TODATETIME(double)
```

CDF 仕様: @CalcMgrExcelToDateTime(excel_date)

@CalcMgrExcelWEEKNUM

目的:

特定の日付がその年の何週目に当たるかを返します。たとえば、1月1日を含む週がその年の第1週目で、週1と数えられます。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.WEEKNUM(double, double)
```

CDF 仕様: @CalcMgrExcelWEEKNUM(date, method)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelWEEKDAY

目的:

日付に対応する曜日を返します。曜日は、デフォルトでは 1(日曜日)から 7(土曜日)までの整数で示されます。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.WEEKDAY(double, double)
```

CDF 仕様: @CalcMgrExcelWEEKDAY(serial_number, return_type)

 ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelWORKDAY

目的:

指定した稼働日数だけ前または後の日付に対応するシリアル値を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.WORKDAY (double, double, double[])
```

CDF 仕様: @CalcMgrExcelWORKDAY(startDate, days, holidays)

 ノート:

日付を Oracle Hyperion Planning の日付形式で渡している場合は、[@CalcMgrDateToExcel](#) を使用して Planning の日付を Excel の日付に変換する必要があります。

 ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelYEAR

目的:

シリアル値を年に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.YEAR (double)
```

CDF 仕様: @CalcMgrExcelYEAR(date)

 ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelYEARFRAC

目的:

start_date と end_date の間の日数が 1 年間に対して占める割合を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelDateTimeFunctions.YEARFRAC(double, double, double)
```

CDF 仕様: @CalcMgrExcelYEARFRAC(startDate, endDate, basis)

 ノート:

日付を Oracle Hyperion Planning の日付形式で渡している場合は、[@CalcMgrDateToExcel](#) を使用して Planning の日付を Excel の日付に変換する必要があります。

 ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrGetCurrentDate

目的:

YYYYMMDD 形式で現在の日付を返します(例: 20140101)

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.getCurrentDate(int)
```

CDF 仕様: @CalcMgrGetCurrentDate()

@CalcMgrGetCurrentDateTZ

目的:

YYYYMMDD 形式で指定されたタイムゾーンで現在の日付を返します(例: 20140101)

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getCurrentDate(String)`
CDF 仕様: `@CalcMgrGetCurrentDateTZ(timeZone)`

@CalcMgrGetCurrentDateTime

目的:

YYYYMMDDHHMMSS 形式で現在の日時を返します。例: 20140101143001
(Year_Month_Day_Hour_Minute_Second)

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getCurrentDateTime()`
CDF 仕様: `@CalcMgrGetCurrentDateTime()`

@CalcMgrGetCurrentDateTimeTZ

目的:

YYYYMMDDHHMMSS 形式で指定されたタイムゾーンで現在の日時を返します。例:
20140101143001 (Year_Month_Day_Hour_Minute_Second)

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getCurrentDateTime(String)`
CDF 仕様: `@CalcMgrGetCurrentDateTimeTZ(timeZone)`

@CalcMgrGetCustomDate

目的:

YYYYMMDD 形式でカスタム日付を返します。
たとえば、20140101 です

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getDate(double, double, double)`
CDF 仕様: `@CalcMgrGetCustomDate(year, month, day)`

@CalcMgrGetCustomDateTime

目的:

カスタム日時を YYYYMMDDHHMMSS 形式で返します。例: 20140101143001
(Year_Month_Day_Hour_Minute_Second)

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.getDateTime(double, double, double, double, double, double, double):
```

CDF 仕様: @CalcMgrGetCustomDateTime(year, month, day, hour, min, sec)

@CalcMgrGetDatePart

目的:

Year/Month/DayOfMonth/WeekOfYear/WeekOfMonth/DayOfYear を YYYYMMDD 形式の日付の数字として返します。"date_part_ex"は"dayofmonth"、"dayofyear"、"month"、"weekofmonth"、"weekofyear"または"year"です

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.datePart(Double, String)
```

CDF 仕様: @CalcMgrGetDatePart(date, date_part_ex)

@CalcMgrGetDateTimePart

目的:

日付から Year/Month/DayOfMonth/WeekOfYear/WeekOfMonth/DayOfYear/Hour/Minute/Seconds を数値として返します。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.dateTimePart(double, String)
```

CDF 仕様: @CalcMgrGetDateTimePart(date, date_part_ex)

@CalcMgrGetDay

目的:

YYYYMMDD 形式の日付から日を返します

構文:

Java クラス: com.hyperion.calcmgr.common.cdf.CalendarFunctions.getDay(int)

CDF 仕様: @CalcMgrGetDay(date)

@CalcMgrGetDayOfYear

目的:

YYYYMMDD 形式の日付から通日(1-366)を返します。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getDayOfYear(int)`
CDF 仕様: `@CalcMgrGetDayOfYear(date)`

@CalcMgrGetFormattedDate

目的:

日付を YYYYMMDD 形式に変換します。たとえば、
`@CalcMgrGetFormattedDate(12302014, "mmdyyy")` では、**20141230** が戻されます

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getFormattedDate(int,String)`
CDF 仕様: `@CalcMgrGetFormattedDate(date, format)`

@CalcMgrGetMaxDaysInMonth

目的:

YYYYMMDD 形式の日付の月の最大日数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getActualMaximumDays(int)`
CDF 仕様: `@CalcMgrGetMaxDaysInMonth(date)`

@CalcMgrGetMonth

目的:

YYYYMMDD 形式の日付から月を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.getMonth(int)`
CDF 仕様: `@CalcMgrGetMonth(date)`

@CalcMgrGetStringFormattedDateTime

目的:

形式で定義された日付を YYYYMMddHHmmss 形式の日付に変換します。
たとえば、`@CalcMgrGetFormattedDate(12302014, "MMdyyyHHmmss")` は
201412301430 を返します。

詳細は、Java doc の"SimpleDateFormat"を参照してください。

使用可能な形式の値: mmdyyyyyHHmmss、ddmmyyyyHHmmss、yyyyddmmHHmmss

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.getStringFormattedDateTime(String, String)
```

CDF 仕様: @CalcMgrGetStringFormattedDateTime(date, format)

@CalcMgrGetWeekOfMonth

YYYYMMDD 形式の日付から月の週を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.getWeekOfMonth(int)
```

CDF 仕様: @CalcMgrGetWeekOfMonth(date)

@CalcMgrGetWeekOfYear

目的:

YYYYMMDD 形式の日付から年内通算週数を返します

構文:

Java クラス: com.hyperion.calcmgr.common.cdf.CalendarFunctions.getWeekOfYear(int)

CDF 仕様: @CalcMgrGetWeekOfYear(date)

@CalcMgrGetYear

目的:

YYYYMMDD 形式の日付から年を返します。"date_part_ex"は"dayofmonth"、"dayofyear"、"month"、"weekofmonth"、"weekofyear"または"year"の必要があります

構文:

Java クラス: com.hyperion.calcmgr.common.cdf.CalendarFunctions.getYear(int)

CDF 仕様: @CalcMgrGetYear(date)

@CalcMgrIsLeapYear

目的:

指定した日付がうるう年かどうかを判別します。日付は、YYYYMMDD または YYYY 形式である必要があります(例: 20140101 または 2014)

構文:

Java クラス: com.hyperion.calcmgr.common.cdf.CalendarFunctions.isLeapYear(int)

CDF 仕様: @CalcMgrIsLeapYear(date)

@CalcMgrMonthsBetween

目的:

YYYYMMDD 形式の 2 つの日付の間の月数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.monthsBetween(int,int)
```

CDF 仕様: @CalcMgrMonthsBetween(fromDate,toDate)

ノート:

@CalcMgrMonthsBetween が返すのは正の数のみです。負の数が必要な場合に負の数を返すには、[@CalcMgrMonthsDiff](#) を使用します。

@CalcMgrMonthsDiff

目的:

YYYYMMDD 形式の 2 つの日付の間の月数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.monthsDiff(int,int)
```

CDF 仕様: @CalcMgrMonthsDiff(fromDate,toDate)

ノート:

@CalcMonthsDiff を使用するとき、関数で最初の日付が 2 番目の日付より後になっている場合は、負の数が返されます。関数で最初の日付が 2 番目の日付より前になっている場合は、正の数が返されます。正の数のみを返す場合には、[@CalcMgrMonthsBetween](#) を使用します。

@CalcMgrRollDate

目的:

指定した日付のフィールドを他のフィールドは変更せずに 1 単位時間増減(up または down)します。

たとえば、@CalcMgrRollDate(19960131,"month",@_true)の結果の日付は 19960229 です。@CalcMgrRollDate(19960131,"day",@_true)の結果の日付は 19960101 です。

date_part に指定できる値は、day、month、week および year です。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.rollDate(int,String,boolean)`
CDF 仕様: `@CalcMgrRollDate(date,date_part,up)`

@CalcMgrRollDay

目的:

日を増減して YYYYMMDD 形式の日付にします

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.rollDay(int,boolean)`
CDF 仕様: `@CalcMgrRollDay(date,up)`

@CalcMgrRollMonth

目的:

月を増減して YYYYMMDD 形式の日付にします。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.rollMonth(int,boolean)`
CDF 仕様: `@CalcMgrRollMonth(date,up)`

@CalcMgrRollYear

目的:

年を増減して YYYYMMDD 形式の日付にします。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.rollYear(int,boolean)`
CDF 仕様: `@CalcMgrRollYear(date,up)`

@CalcMgrWeeksBetween

目的:

YYYYMMDD 形式の 2 つの日付の間の週数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.CalendarFunctions.weeksBetween(int,int)`
CDF 仕様: `@CalcMgrWeeksBetween(fromDate,toDate)`

 ノート:

@CalcMgrWeeksBetween が返すのは正の数のみです。負の数が必要な場合に負の数を返すには、@CalcMgrWeeksDiff を使用します。

@CalcMgrWeeksDiff

目的:

YYYYMMDD 形式の 2 つの日付の間の週数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.weeksDiff(int,int)
```

CDF 仕様: @CalcMgrWeeksDiff(fromDate,toDate)

 ノート:

@CalcMgrWeeksDiff を使用するとき、関数で最初の日付が 2 番目の日付より後になっている場合は、負の数が返されます。関数で最初の日付が 2 番目の日付より前になっている場合は、正の数が返されます。正の数のみを返す場合には、@CalcMgrWeeksBetween を使用します。

@CalcMgrYearsBetween

目的:

YYYYMMDD 形式の 2 つの日付の間の年数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.CalendarFunctions.yearsBetween(int,int)
```

CDF 仕様: @CalcMgrYearsBetween(fromDate,toDate)

 ノート:

@CalcMgrYearsBetween が返すのは正の数のみです。負の数が必要な場合に負の数を返すには、@CalcMgrYearsDiff を使用します。

@CalcMgrYearsDiff

目的:

YYYYMMDD 形式の 2 つの日付の間の年数を返します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.CalendarFunctions.yearsDiff(int,int)`

CDF 仕様: `@CalcMgrYearsDiff(fromDate,toDate)`

ノート:

`@CalcMgrYearsDiff` を使用するとき、関数で最初の日付が 2 番目の日付より後になっている場合は、負の数が返されます。関数で最初の日付が 2 番目の日付より前になっている場合は、正の数が返されます。正の数のみを返す場合には、[@CalcMgrYearsBetween](#) を使用します。

財務関数

次も参照:

- [@CalcMgrExcelACCRINT](#)
- [@CalcMgrExcelACCRINTM](#)
- [@CalcMgrExcelAMORDEGRC](#)
- [@CalcMgrExcelAMORLINC](#)
- [@CalcMgrExcelCOUPDAYBS](#)
- [@CalcMgrExcelCOUPDAYS](#)
- [@CalcMgrExcelCOUPDAYSNC](#)
- [@CalcMgrExcelCOUPNCD](#)
- [@CalcMgrExcelCOUPNUM](#)
- [@CalcMgrExcelCOUPPCD](#)
- [@CalcMgrExcelCUMIPMT](#)
- [@CalcMgrExcelCUMPRINC](#)
- [@CalcMgrExcelDB](#)
- [@CalcMgrExcelDDB](#)
- [@CalcMgrExcelDISC](#)
- [@CalcMgrExcelDOLLARDE](#)
- [@CalcMgrExcelDOLLARFR](#)
- [@CalcMgrExcelDURATION](#)
- [@CalcMgrExcelEFFECT](#)
- [@CalcMgrExcelFV](#)
- [@CalcMgrExcelFVSCHEDULE](#)
- [@CalcMgrExcelMDURATION](#)
- [@CalcMgrExcelINTRATE](#)
- [@CalcMgrExcelPMT](#)

- @CalcMgrExcelIRR
- @CalcMgrExcelISPMT
- @CalcMgrExcelMIRR
- @CalcMgrExcelNPER
- @CalcMgrExcelNPV
- @CalcMgrExcelPPMT
- @CalcMgrExcelPRICE
- @CalcMgrExcelPRICEDISC
- @CalcMgrExcelPRICEMAT
- @CalcMgrExcelPV
- @CalcMgrExcelRATE
- @CalcMgrExcelRECEIVED
- @CalcMgrExcelSLN
- @CalcMgrExcelSYD
- @CalcMgrExcelTBILLEQ
- @CalcMgrExcelTBILLPRICE
- @CalcMgrExcelTBILLYIELD
- @CalcMgrExcelXIRR
- @CalcMgrExcelXNPV
- @CalcMgrExcelYIELD
- @CalcMgrExcelYIELDDISC
- @CalcMgrExcelYIELDMAT

@CalcMgrExcelACCRINT

目的:

定期的に利息が支払われる証券の未収利息額を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.ACCRINT(double, double, double, double, double, double, double, double, boolean)
```

CDF 仕様: @CalcMgrExcelACCRINT(issue, firstinterest, settlement, rate, par, frequency, basis, method)

@CalcMgrExcelACCRINTM

目的:

満期日に利息が支払われる証券の未収利息額を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.ACCRINTM(issue, settlement, rate, par, basis)`

CDF 仕様:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.ACCRINTM(double, double, double, double, double, double)`

@CalcMgrExcelAMORDEGRC

目的:

減価償却係数を使用して、各会計期間の減価償却費を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.AMORDEGRC(double, double, double, double, double, double, double)`

CDF 仕様: `@CalcMgrExcelAMORDEGRC(cost, purchased, firstPeriod, salvage, period, rate, basis)`

@CalcMgrExcelAMORLINC

目的:

各会計期における減価償却費を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.AMORLINC(double, double, double, double, double, double, double)`

CDF 仕様: `@CalcMgrExcelAMORLINC(cost, date_purchased, first_period, salvage, period, rate, basis)`

@CalcMgrExcelCOUPDAYBS

目的:

利払期間の第 1 日目から受渡日までの日数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.COUPDAYBS(double, double, double, double, double)`

CDF 仕様: `@CalcMgrExcelCOUPDAYBS(settlement, maturity, frequency, basis)`

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelCOUPDAYS

目的:

受渡日を含む利払期間内の日数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.COUPDAYS(double, double, double, double)
```

CDF 仕様: @CalcMgrExcelCOUPDAYS(settlement, maturity, frequency, basis)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelCOUPDAYSNC

目的:

受渡日から次の利払日までの日数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.COUPDAYSNC(double, double, double, double)
```

CDF 仕様: @CalcMgrExcelCOUPDAYSNC(settlement, maturity, frequency, basis)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelCOUPNCD

目的:

受渡日後の次の利払日を数値で返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.COUPNCD(double, double, double, double)
```

CDF 仕様: @CalcMgrExcelCOUPNCD(settlement, maturity, frequency, basis)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelCOUPNUM

目的:

受渡日と満期日の間に利息が支払われる回数を返します。端数が出た場合は切り上げられます

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.COUPNUM(double, double, double, double)
```

CDF 仕様: @CalcMgrExcelCOUPNUM(settlement, maturity, frequency, basis)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelCOUPPCD

目的:

受渡日直前の利払日を数値で返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.COUPPCD(double, double, double, double)
```

CDF 仕様: @CalcMgrExcelCOUPPCD(settlement, maturity, frequency, basis)

 ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelCUMIPMT

目的:

start_period から end_period までの期間内で貸付金に対して支払われる利息の累計を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.CUMIPMT(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelCUMIPMT(rate, nper, pv, start_period, end_period, type)

@CalcMgrExcelCUMPRINC

目的:

開始から終了までの期間に、貸付金に対して支払われる元金の累計を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.CUMPRINC(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelCUMPRINC(rate, per, nper, pv, fv, type)

@CalcMgrExcelDB

目的:

定率法を利用して、特定の期における資産の減価償却費を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.DB(double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelDB((cost, salvage, life, period, month)

@CalcMgrExcelDDB

目的:

倍額定率法または指定した他の方法を使用して、特定の期における資産の減価償却費を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.DDB(double,double,  
double,double,double)
```

CDF 仕様: @CalcMgrExcelDDB(cost, salvage, life, period, factor)

@CalcMgrExcelDISC

目的:

証券に対する割引率を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.DISC(double,double  
,double,double,double)
```

CDF 仕様: @CalcMgrExcelDISC(settlement, maturity, pr, redemption, basis)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelDOLLARDE

目的:

整数部と分数部で表されたドル価格を、1.02 などの小数で表されたドル価格に変換します。分数で表されたドル価格は証券の価格に使用されることがあります。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.DOLLARDE(double,do  
uble)
```

CDF 仕様: @CalcMgrExcelDOLLARDE(fractional_dollar, fraction)

@CalcMgrExcelDOLLARFR

目的:

小数で表されたドル価格を分数で表されたドル価格に変換します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.DOLLARFR(double, double)`

CDF 仕様: `@CalcMgrExcelDOLLARFR(decimal_dollar, fraction)`

@CalcMgrExcelDURATION

目的:

定期利払いの場合の有価証券の年間デューションを返します **ノート:**

@CalcMgrExcelMDURATION 関数を使用する場合、**Oracle Hyperion Calculation Manager** と **Excel** で計算が一致しないことがあります。数値を一致させるには、小数部を 7 に変更し、**Open Office** を使用します。

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.DURATION(double, double, double, double, double, double)`

CDF 仕様: `@CalcMgrExcelDURATION(settlement, maturity, coupon, yld, frequency, basis)`

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelEFFECT

目的:

実効年利率を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.EFFECT(double, double)`

CDF 仕様: `@CalcMgrExcelEFFECT(nominal_rate, npery)`

@CalcMgrExcelFV

目的:

投資の将来価値を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.FV(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelFV(rate, nper, pmt, pv, type)

@CalcMgrExcelFVSCCHEDULE

目的:

一連の金利を複利計算することにより、初期投資の元金の将来価値を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.FVSCCHEDULE(double, double[])
```

CDF 仕様: @CalcMgrExcelFVSCCHEDULE(principal, schedule)

@CalcMgrExcelMDURATION

目的:

推定額面が\$100 の場合の有価証券の修正マコーレ・デュレーションを返します **ノート:**

@CalcMgrExcelMDURATION 関数を使用する場合、Oracle Hyperion Calculation Manager と Excel で計算が一致しないことがあります。数値を一致させるには、小数部を 7 に変更し、Open Office を使用します。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.MDURATION(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelMDURATION(settlement, maturity, coupon, yld, frequency, basis)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある @CalcMgrExcel カスタム関数を参照してください。

@CalcMgrExcelINTRATE

目的:

全額投資された証券の利率を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.INTRATE(double, double, double, double, double)`

CDF 仕様: `@CalcMgrExcelINTRATE(settlement, maturity, investment, redemption, basis)`

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelPMT

目的:

一定期間支払が定期的に行われるような投資/返済の定期支払額を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.PMT(double, double, double, double, double)`

CDF 仕様: `@CalcMgrExcelPMT(rate, nper, pv, fv, type)`

@CalcMgrExcelIRR

目的:

一連のキャッシュ・フローに対する内部利益率を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.IRR(double[], double)`

CDF 仕様: `@CalcMgrExcelIRR(values, guess)`

@CalcMgrExcelISPMT

目的:

投資期間内の指定された期に支払われる金利を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.ISPMT(double, double, double, double)`

CDF 仕様: `@CalcMgrExcelISPMT(rate, per, nper, pv)`

@CalcMgrExcelMIRR

目的:

正のキャッシュ・フローと負のキャッシュ・フローのレートが異なる場合の内部利益率を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.MIRR(double[], double, double)
```

CDF 仕様: @CalcMgrExcelMIRR(values, finance_rate, reinvest_rate)

@CalcMgrExcelNPER

目的:

投資に必要な期間を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.NPER(double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelNPER(rate, pmt, pv, fv, type)

@CalcMgrExcelNPV

目的:

定期的に発生する一連のキャッシュ・フローおよび割引率に基づいた投資の正味現在価値を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.NPV(double, double[])
```

CDF 仕様: @CalcMgrExcelNPV(rate, values)

@CalcMgrExcelPPMT

目的:

定額の支払いを定期的に行い、利率が一定であると仮定して、投資の指定した期に支払われる元金を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.PPMT(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelPPMT(rate, per, nper, pv, fv, type)

@CalcMgrExcelPRICE

目的:

定期的に利息が支払われる証券に対して、額面\$100 当たりの価格を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.PRICE(double, double, double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelPRICE(settlement, maturity, rate, yld, redemption, frequency, basis)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelPRICEDISC

目的:

割引証券の額面\$100 当たりの価格を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.PRICEDISC(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelPRICEDISC(settlement, maturity, discount, redemption, basis)

ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelPRICEMAT

目的:

満期日に利息が支払われる証券に対して額面\$100 当たりの価格を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.PRICEMAT(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelPRICEMAT(settlement, maturity, issue, rate, yld, basis)

 **ノート:**

@CalcMgrExcel 関数で使用する日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelPV

目的:

投資の現在価値を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.PV(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelPV(rate, nper, pmt, fv, type)

@CalcMgrExcelRATE

目的:

一定期間支払が定期的に行われるような投資/返済の期間当たりの利率を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.RATE(double, double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelRATE(nper, pmt, pv, fv, type, guess)

@CalcMgrExcelRECEIVED

目的:

全額投資された証券に対して、満期日に支払われる金額を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.RECEIVED(double, double, double, double)
```

CDF 仕様: @CalcMgrExcelRECEIVED(settlement, maturity, investment, discount, basis)

 ノート:

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelSLN

目的:

定額法を使用して、資産の 1 期当たりの減価償却費を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.SLN(double, double, double)
```

CDF 仕様: @CalcMgrExcelSLN(cost, salvage, life)

@CalcMgrExcelSYD

目的:

級数法を使用して、特定の期における減価償却費を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.SYD(double, double, double, double)
```

CDF 仕様: @CalcMgrExcelSYD(cost, salvage, life, per)

@CalcMgrExcelTBILLEQ

目的:

米国財務省短期証券の債券換算利回りを返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.TBILLEQ(double, double, double)
```

CDF 仕様: @CalcMgrExcelTBILLEQ(settlement, maturity, discount)

 ノート:

@CalcMgrExcel 関数で使用する日付パラメータは、Excel 形式である必要があります。日付パラメータがある @CalcMgrExcel カスタム関数を参照してください。

@CalcMgrExcelTBILLPRICE

目的:

米国財務省短期証券の額面\$100 当たりの価格を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.TBILLPRICE(double,  
double,double)
```

CDF 仕様: @CalcMgrExcelTBILLPRICE(settlement, maturity, discount)

 ノート:

@CalcMgrExcel 関数で使用する日付パラメータは、Excel 形式である必要があります。日付パラメータがある @CalcMgrExcel カスタム関数を参照してください。

@CalcMgrExcelTBILLYIELD

目的:

米国財務省短期証券の利回りを返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.TBILLYIELD(double,  
double,double)
```

CDF 仕様: @CalcMgrExcelTBILLYIELD(settlement, maturity, pr)

 ノート:

@CalcMgrExcel 関数で使用する日付パラメータは、Excel 形式である必要があります。日付パラメータがある @CalcMgrExcel カスタム関数を参照してください。

@CalcMgrExcelXIRR

目的:

定期的でないキャッシュ・フローに対する内部利益率を返します

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.XIRR(double[],double[],double)

CDF 仕様: @CalcMgrExcelXIRR(values, dates, guess)

@CalcMgrExcelXNPV

目的:

定期的でないキャッシュ・フローに対する正味現在価値を返します

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.XNPV(double, double[],double[])

CDF 仕様: @CalcMgrExcelXNPV(rate, values, dates)

@CalcMgrExcelYIELD

目的:

利息が定期的に支払われる証券の利回りを返します

構文:

Java クラス:
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.YIELD(double, double, double, double, double, double, double)

CDF 仕様: @CalcMgrExcelYIELD(settlement, maturity, rate, pr, redemption, frequency, basis)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、Excel 形式である必要があります。日付パラメータがある @CalcMgrExcel カスタム関数を参照してください。

@CalcMgrExcelYIELDDISC

目的:

たとえば、米国財務省短期証券など、割引き債の年利回りを返します **ノート:**

@CalcMgrExcelYIELDDISC 関数を使用する場合、Oracle Hyperion Calculation Manager と Excel で計算が一致しないことがあります。数値を一致させるには、小数部を 7 に変更し、Open Office を使用します。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.YIELDDISC(double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelYIELDDISC(settlement, maturity, pr, redemption, basis)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

@CalcMgrExcelYIELDMAT

目的:

償還日に利息を支払う有価証券の年利回りを返します **ノート:** @CalcMgrExcelYIELDMAT 関数を使用する場合、**Oracle Hyperion Calculation Manager** と **Excel** で計算が一致しないことがあります。数値を一致させるには、小数部を 7 に変更し、**Open Office** を使用します。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelFinancialFunctions.YIELDMAT(double, double, double, double, double, double)
```

CDF 仕様: @CalcMgrExcelYIELDMAT(settlement, maturity, issue, rate, pr, basis)

 **ノート:**

@CalcMgrExcel 関数で使用される日付パラメータは、**Excel** 形式である必要があります。日付パラメータがある [@CalcMgrExcel カスタム関数](#) を参照してください。

ログ関数

次も参照:

- [@CalcMgrLog](#)
- [@CalcMgrLogText](#)

@CalcMgrLog

目的:

計算スクリプトの診断をロギングします。ファイル名が空の場合、メッセージはコンソールに書き込まれます

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.log(String,String,String[]
,double[],boolean)
```

CDF 仕様: CalcMgrLog(fileName,msg,strings,values,printDate)

例 1:

```
FIX ("Sales",@Children("100"),"Florida")
    "Budget" (
        @CalcMgrLog("c:/temp/cal.log", "Log: ",
            @LIST(@NAME(@CURRMBR("Year")),
                @NAME(@CURRMBR("Measures")),
                @NAME(@CURRMBR("Product")),
                @NAME(@CURRMBR("Market")),
                @NAME(@CURRMBR("Scenario"))
            ),
            @LIST(@CURRMBR("Year"),
                @CURRMBR("Measures"),
                @CURRMBR("Product"),
                @CURRMBR("Market"),
                @CURRMBR("Scenario")
            ),
            @_True
        );
    )
```

ENDFIX

The log file entry might look like this:

この例では、ログ・ファイル・エントリは次のようになります:

```
2014-12-20 15:38:13 Log: Apr Sales 100-10 Florida Budget 210.0 210.0
210.0 210.0 210.0
```

例 2:

```
FIX ("Sales",@Children("100"),"Florida")
    "Budget" (
        @CalcMgrLog(
            "c:/temp/cal.log",
            "Log: ",
            @LIST(
                @CalcMgrStringsToString(
                    @LIST(@NAME(@CURRMBR("Year")),
                        @NAME(@CURRMBR("Measures")), @NAME(@CURRMBR("Product")),
                        @NAME(@CURRMBR("Market")), @NAME(@CURRMBR("Scenario"))),
                    ",",
                    @CalcMgrFormatDouble("%7.2f", @CURRMBR("Year")-
>@CURRMBR("Measures")->@CURRMBR("Product")->@CURRMBR("Market")-
>@CURRMBR("Scenario"))
                ),
            @LIST(0.0), @_True);
    )
```

```
)  
ENDFIX
```

この例では、ログ・ファイル・エントリは次のようになります:

```
2014-12-20 15:40:10 Log: Apr,Sales,100-10,Floridshoula,Budget 210.00 0.0
```

@CalcMgrLogText

目的:

ログ・ファイルにメッセージを書き込みます。ファイル名が空の場合、メッセージはコンソールに書き込まれます。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.logText (String,String,boolean)
```

CDF 仕様: @CalcMgrLogText (fileName,msg,printDate)

例:

```
FIX ("Sales",@Children("100"),"Florida")  
    "Budget" (  
        @CalcMgrLogText (  
            "c:/temp/cal.log",  
            "text of message to display" @_False);  
        )  
ENDFIX
```

数学関数

次も参照:

- [@CalcMgrExcelCEILING](#)
- [@CalcMgrExcelCOMBIN](#)
- [@CalcMgrExcelEVEN](#)
- [@CalcMgrExcelFACT](#)
- [@CalcMgrExcelFLOOR](#)
- [@CalcMgrExcelGCD](#)
- [@CalcMgrExcelLCM](#)
- [@CalcMgrExcelMROUND](#)
- [@CalcMgrExcelMULTINOMIAL](#)
- [@CalcMgrExcelODD](#)
- [@CalcMgrExcelPOWER](#)
- [@CalcMgrExcelPRODUCT](#)
- [@CalcMgrExcelROUNDDOWN](#)

- [@CalcMgrExcelROUNDUP](#)
- [@CalcMgrExcelSQRT](#)
- [@CalcMgrExcelSQRTPI](#)
- [@CalcMgrExcelSUMPRODUCT](#)
- [@CalcMgrExcelSUMSQ](#)

@CalcMgrExcelCEILING

目的:

最も近い整数または最も近い基準値の倍数に数値を切り上げます(ゼロから離れます)

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.CEILING(double, double)
```

CDF 仕様: @CalcMgrExcelCEILING(number, significance)

@CalcMgrExcelCOMBIN

目的:

指定した数のオブジェクトの組合せの数を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.COMBIN(double, double)
```

CDF 仕様: @CalcMgrExcelCOMBIN(number, number_chosen)

@CalcMgrExcelEVEN

目的:

数値を最も近い偶数に切り上げます

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.EVEN(double)
```

CDF 仕様: @CalcMgrExcelEVEN(number)

@CalcMgrExcelFACT

目的:

数値の階乗を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.FACT(double)`
CDF 仕様: `@CalcMgrExcelFACT(number)`

@CalcMgrExcelFLOOR

目的:
数値を切り捨てます(ゼロに近づきます)

構文:
Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.FLOOR(double, double)`
CDF 仕様: `@CalcMgrExcelFLOOR(number, significance)`

@CalcMgrExcelGCD

目的:
最大公約数を返します

構文:
Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.GCD(double[])`
CDF 仕様: `@CalcMgrExcelGCD(numbers)`

@CalcMgrExcelLCM

目的:
最小公倍数を返します

構文:
Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.LCM(double[])`
CDF 仕様: `@CalcMgrExcelLCM(numbers)`

@CalcMgrExcelMROUND

目的:
数値を丸めて指定された桁数にします

構文:
Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.MROUND(double, double)`
CDF 仕様: `@CalcMgrExcelMROUND(number, num_digits)`

@CalcMgrExcelMULTINOMIAL

目的:

数値のセットの多項係数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.MULTINOMIAL(double[])`

CDF 仕様: `@CalcMgrExcelMULTINOMIAL(numbers)`

@CalcMgrExcelODD

目的:

数値を最も近い奇数に切り上げます

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.ODD(double)`

CDF 仕様: `@CalcMgrExcelODD(number)`

@CalcMgrExcelPOWER

目的:

数値のべき乗を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.POWER(double, double)`

CDF 仕様: `@CalcMgrExcelPOWER(number, power)`

@CalcMgrExcelPRODUCT

目的:

引数を乗算します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.PRODUCT(double[])`

CDF 仕様: `@CalcMgrExcelPRODUCT(numbers)`

@CalcMgrExcelROUNDDOWN

目的:

数値を切り捨てます(ゼロに近づきます)

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.ROUNDDOWN(double, double)
```

CDF 仕様: @CalcMgrExcelROUNDDOWN(number, num_digits)

@CalcMgrExcelROUNDUP

目的:

数値を切り上げます(ゼロから離れます)

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.ROUNDUP(double, double)
```

CDF 仕様: @CalcMgrExcelROUNDUP(number, num_digits)

@CalcMgrExcelSQRT

目的:

正の平方根を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.SQRT(double)
```

CDF 仕様: @CalcMgrExcelSQRT(number)

@CalcMgrExcelSQRTPI

目的:

(数値 * pi)の平方根を返します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.SQRTPI(double)
```

CDF 仕様: @CalcMgrExcelSQRTPI(number)

@CalcMgrExcelSUMPRODUCT

目的:

配列の対応する要素間の積の和を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.SUMPRODUCT(double[], double[])`

CDF 仕様: `@CalcMgrExcelSUMPRODUCT(values1, values2)`

@CalcMgrExcelSUMSQ

目的:

引数の平方和を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelMathFunctions.SUMSQ(double[])`

CDF 仕様: `@CalcMgrExcelSUMSQ(numbers)`

MaxL 関数

 **ノート:**

FIX 文で関数をコールするかわりに、RUNJAVA コマンドを使用して **MAXL** 関数を使用することをお勧めします。RUNJAVA コマンドはスクリプト・コンポーネントのみで使用できます。

@CalcMgrExecuteEncryptMaxLFile

目的:

Oracle Essbase からアクセスできる MaxL スクリプト・ファイルを使用して MaxL スクリプトを実行します。

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.MaxLFunctions`

CDF 仕様: `@CalcMgrExecuteEnMaxLFile(privateKey, maxlFileName, arguments, asynchronous)`

@CalcMgrExecuteMaxLScript

目的:

MaxL スクリプトを計算スクリプトそのものに埋め込みます。

構文:

Java クラス: com.hyperion.calcmgr.common.cdf.MaxLFunctions

CDF 仕様: @CalcMgrExecuteMaxLEnScript(privateKey, maxlScripts, arguments, asynchronous)

MaxL の暗号化の使用

MaxL 暗号化を使用するには:

1. `essmsh` を使用して公開キーと秘密キーを生成します。コマンド `essmsh -gk` を実行すると、コンソール上で公開キーおよび秘密キーが生成されます(暗号化用の公開キー: **25159,850102129**、復号化用の秘密キー: **240764119,850102129**)。
2. 公開キー `essmsh -ep admin 25159,850102129` を使用して暗号化されたトークンを生成します。
906388712099924604712352658511 のようなトークンが生成されます。
3. このトークンを MaxL スクリプト `login $key $1 $key $2 on localhost` で使用します。
4. 秘密キーとともに `-D` オプションを使用して `essmsh` を呼び出し、スクリプト `essmsh -D` を実行します。
たとえば、`essmsh C:/Temp/maxlcmd1.xmls 240764119, 850102129 906388712099924604712352658511 0893542980829559883146306518502837293210` です。

@CalcMgrExecuteEncryptMaxLFile を使用した MaxL スクリプト・ファイルの実行

@CalcMgrExecuteEncryptMaxLFile の構文は、次のとおりです:

```
FIX ("Sales","100-10","New York","Actual")
    "Jan" (
        @CalcMgrExecuteEncryptMaxLFile ("2115028571,2505324337","C:/Temp/
exportdata.xmls",
@List("906388712099924604712352658511","0893542980829559883146306518502837293
210"), "true");
    ENDFIX
```

最初の引数は秘密キー、2 番目の引数は MaxL スクリプト・ファイル、3 番目の引数はスクリプトの引数です。最後の引数は、これを非同期に実行するかどうかを示します("true"は非同期を示し、"false"は同期を示します)。

この方法にはいくつかの問題があります。最初の問題は、同じスクリプトを実行する必要がある回数です。このスクリプトは、FIX 文のすべてのメンバーの組合せに対して実行されます。2 番目の問題は、FIX 文のメンバーの組合せに対してブロックが存在しないと Oracle Essbase が判断した場合にスクリプトが一切実行されない可能性があることです。

通常、MaxL スクリプトを実行するのは 1 回のみであり、RUNJAVA コマンドを使用します:

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLFunctions
    <LOG FILE> OPTIONAL ex., "logfile=c:/Temp/maxlrule.log" The path should
be accessible from the ESSBASE process.
    <ASYNCH FLAG> OPTIONAL ex., "true" This means calc engine does not wait
```

for the completion of the MaxL script. The default is "false"
<ALL OTHER PARAMETERS YOU PASS TO ESSMSH>

例:

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLFunctions
"logfile=c:/Temp/maxlrule.log"
"-D"
"c:\\Temp\\maxl.msh"
"28567271,1186612387"
"893848844082678214004255849650"
"4647275840619320283077900267208176084380"
"{varAppName}";
```

@CalcMgrExecuteMaxLScript を使用した MaxL スクリプトの実行

@CalcMgrExecuteMaxLScript の構文は、次のとおりです:

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLScriptFunction
<LOG FILE> OPTIONAL ex., "logfile=c:/Temp/maxlrule.log" The path
should be accessible from the ESSBASE process
<ASYNCH FLAG> OPTIONAL ex., "true" This means calc engine will
not wait for the completion of the maxl script.
"mdx=true"OPTIONAL The variables are replaced with MDX syntax.
This applies only to variables within the Crossjoin sections.
"-D" if using encryption
"<PRIVATE KEY>" if using encryption
<SCRIPT LINE>
<SCRIPT LINE>
<SCRIPT LINE>
...
<ALL OTHER PARAMETERS YOU PASS TO ESSMSH>
```

例 1: @CalcMgrExecuteMaxLScript の使用

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLScriptFunction
"-D"
"28567271,1186612387"
"spool on to 'C:\\ALC\\out.log'"
"login $key 893848844082678214004255849650 $key
4647275840619320283077900267208176084380 on localhost"
"alter database 'SampleASO'. 'Basic' clear data in region
'Crossjoin(Descendants([TotalPV],Province.Levels(0)),Crossjoin(Descenda
nts([TotalLOB],LOB.Levels(0)),
Crossjoin(Descendants([TotalActivity],Activity.Levels(0)),Crossjoin(Des
cendants([TotalBrand],Brand.Levels(0)),
Crossjoin(Descendants([BS000],Account.Levels(0)),Crossjoin({[FY14]},Cro
ssjoin({[Working]},Crossjoin({[Apr]},{[Actual]}))))))' physical"
"logout"
"spool off"
```

```
"exit"
```

例 2: 実行時プロンプト変数と MaxL 引数ホルダーの使用

スクリプトが固定されておらず、実行時プロンプトまたは MaxL の引数に基づくとは定めます。引用テキスト内で実行時プロンプトを使用した場合、または MaxL の引用テキスト内で \$1 を使用した場合、Oracle Hyperion Planning は置換しません。次の例では、"Working" に対するハードコーディングのかわりに実行時プロンプトを使用します。実行時プロンプトと MaxL 引数ホルダーを使用してスクリプトを書き直すことができます。

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLScriptFunction
  "-D"
  "28567271,1186612387"
  "spool on to 'C:\\ALC\\out.log'"
  "login $key 893848844082678214004255849650 $key
4647275840619320283077900267208176084380 on localhost"
  "alter database 'SampleASO'. 'Basic' clear data in region
'Crossjoin(Descendants([TotalPV], Province.Levels(0)), Crossjoin(Descendants([T
otalLOB], LOB.Levels(0)),
Crossjoin(Descendants([TotalActivity], Activity.Levels(0)), Crossjoin(Descendan
ts([TotalBrand], Brand.Levels(0))),
Crossjoin(Descendants([BS000], Account.Levels(0)), Crossjoin({[FY14]}, Crossjoi
n({[$1]}, Crossjoin({[Apr]}, {[Actual]}))))))' physical"
  "logout"
  "spool off"
  "exit"
  {WhatVersion}
```

この例では {WhatVersion} が実行時プロンプト変数です。この例の "exit" の後にあるものはすべてスクリプトの引数とみなされます。カスタム定義関数が、引数の数に基づいて、スクリプト内の \$1、\$2..\$n を置換します。

例 3: MaxL=True の使用

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLScriptFunction
....."mdx=true"
....."-D"
....."28567271,1186612387"
....."spool on to '/tmp/Essbase_ClearAndLoadStandardFX.log'"
....."login $key 893848844082678214004255849650 $key
4647275840619320283077900267208176084380 on localhost"
....."alter database glblaso.aso clear data in region ' CrossJoin( {$1} ,
{ ( $2 , [Actual] , $3 , [USD] )} )' physical "
....."import database 'GlblASO'. 'ASO' data from server text data_file
'../../StdFxExport.txt' using server rules_file 'LdASO' on
.....error write to './tmp/EssbaseLoadASOStdFX_errors.log' "
  "logout"
  "spool off"
  "exit"
  "LE_83901,LE_83911"
```

```
"Jan"  
"2014"
```

この例の構文によって次のスクリプトが生成されます:

```
spool on to '/tmp/Essbase_ClearAndLoadStandardFX.log';  
  login $key 893848844082678214004255849650 $key  
4647275840619320283077900267208176084380 on localhost;  
  alter database glblaso.aso clear data in region '  
CrossJoin([LE_83901], [LE_83911] , { ( [Jan] , [Actual] , [2014] ,  
[USD] )} )' physical ;  
  import database 'GlblASO'. 'ASO' data from server text data_file  
'../../StdFxExport.txt' using server rules_file 'LdASO' on  
error write to './tmp/EssbaseLoadASOStdFX_errors.log' ;  
  logout;  
  spool off;  
  exit;
```

ノート: MDX=true はオプションです。MDX=true を使用すると、Crossjoin セクションの MDX 構文で変数が置換されます。

例 4: 二重引用符とセミコロンを含む関数リストの使用

CDF で次の計算を実行しようとしています:

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLScriptFunction  
  "spool on to 'c:/tmp/calc.log'"  
  "login admin password on localhost"  
  "execute calculation '  
  SET CREATEBLOCKONEQ ON;  
  SET AGGMISSG ON;  
  FIX (@IDESCENDANTS("BS"),"Plan")  
    DATACOPY "Working" to "Target";  
  ENDFIX  
  
  FIX ("Plan","Target")  
    CALC DIM ("Account","Period","Entity","Product");  
  ENDFIX'  
  on Vision.Plan1"  
  "logout"  
  "spool off"  
  "exit"  
  ;  
  
  OK/INFO - 1012553 - Copying data from [Working] to [Target] with  
fixed members [Account(1110, 1150, 1100, 1210, 1220, 1260, 1200, 1310,  
1332, 1340, 1300,  
1410, 1460, 1 471, 1400, 1010, 1510, 1520, 1530, 1540, 1550,  
1560, 1565, 1570, 1580, 1500, 1610, 1620, 1630, 1640, 1650, ].  
  OK/INFO - 1012675 - Commit Blocks Interval for the calculation is  
[3000].  
  OK/INFO - 1012675 - Commit Blocks Interval for the calculation is  
[3000].  
  OK/INFO - 1012574 - Datacopy command copied [0] source data  
blocks to [0] target data blocks.
```

```

OK/INFO - 1012675 - Commit Blocks Interval for the calculation is
[3000].
OK/INFO - 1012714 - Regular member [Sales Calls] depends on dynamic-
calc member [Rolling]..
OK/INFO - 1012714 - Regular member [Sales Calls] depends on dynamic-
calc member [YearTotal]..
OK/INFO - 1012714 - Regular member [Period] depends on dynamic-calc
member [YearTotal]..
OK/INFO - 1012684 - Multiple bitmap mode calculator cache memory usage
has a limit of [5] bitmaps..
OK/INFO - 1012668 - Calculating [ Account(All members) Period(All
members) Entity(All members) Product(All members)]
with fixed members [Scenario(Plan); Version(Target)].
OK/INFO - 1012677 - Calculating in serial.
OK/INFO - 1012550 - Total Calc Elapsed Time : [0.002] seconds.
OK/INFO - 1013274 - Calculation executed.

MAXL> logout;
User admin is logged out
MaxL Shell completed

```

このスクリプトによって次が生成されます:

```

essmsh C:\Users\SRMENON\AppData\Local\Temp\2\calc8153268024796650753.mxl

```

```

Essbase MaxL Shell 64-bit - Release 11.1.2 (ESB11.1.2.4.000B154)
Copyright (c) 2000, 2014, Oracle and/or its affiliates.
All rights reserved.

```

```

MAXL> login admin password on localhost;
""
OK/INFO - 1051034 - Logging in user [admin@Native Directory].
OK/INFO - 1241001 - Logged in to Essbase.

```

```

MAXL> execute calculation '
2> SET CREATEBLOCKONEQ ON;
3> SET AGGMISSG ON;
4> FIX (@IDESCENDANTS("BS"),"Plan")
5> DATACOPY "Working" to "Target";
6> ENDFIX
7>
8> FIX ("Plan","Target")
9> CALC DIM ("Account","Period","Entity","Product");
10> ENDFIX'
11> on Vision.Plan1;

```

ただし、**Oracle Essbase** では文字列内の">;"が正しく解釈されません。これに対処するために、次のようにすべての">"を"&で、すべての";"を:&で置き換えます:

```

RUNJAVA com.hyperion.calcmgr.common.cdf.MaxLScriptFunction
"logfile=c:/Temp/maxl.log"
"spool on to 'c:/tmp/calc.log'"
"login admin password on localhost"

```

```

"execute calculation '
SET CREATEBLOCKONEQ ON&colon&
SET AGGMISSG ON&colon&
FIX (@IDESCENDANTS (&quot;&BS&quot;&), &quot;&Plan&quot;&)
    DATACOPY &quot;&Working&quot;& to $&quot;&Target&quot;&&colon&
ENDFIX

FIX (&quot;&Plan&quot;&, &quot;&Target&quot;&)
    CALC DIM
(&quot;&Account&quot;&, &quot;&Period&quot;&, &quot;&Entity&quot;&, &quot;&Produc
t&quot;&)&colon&
ENDFIX'
on Vision.Plan1"
"logout"
"spool off"
"exit"
;

```

MDX 関数

次も参照:

- [@CalcMgrMDXDataCopy](#)
- [@CalcMgrMDXDataCopyRTP](#)
- [@CalcMgrMDXDataCopyServer](#)
- [@CalcMgrMDXExport](#)
- [@CalcMgrMDXExportRTP](#)

@CalcMgrMDXDataCopy

目的:

MDX スクリプトを使用してデータベース間でデータをコピーします。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.MDXDataCopy.copyData (String, String, String,
String, String, String, String, String, String, String, String, String, Stri
ng)
```

```
CDF 仕様: @CalcMgrMDXDataCopy (Encryption
Key, user, password, sourceApplicationName,
sourceDatabaseName, targetApplicationName,
targetDatabaseName, columnAxisMDX, rowAxisMDX,
sourceMemberMappings, targetMemberMappings,
targetPOVCols, maxRowsPerPage, loggingFileName)
```

@CalcMgrMDXDataCopy を使用したデータベース間でのデータのコピー

@CalcMgrMDXDataCopy はコピー対象の領域を定義する MDX 式を利用します。これはメモリー内にグリッドを作成し、MDX 式を使用してソース・データベースからグリッドに入力してから、ターゲット・データベースのデータを更新します。

@CalcMgrMDXDataCopy は、calcMgrCmdLine.jar を使用して暗号化されたユーザー名とパスワードを使用します。

たとえば、ユーザー名とパスワードを暗号化するには、java -jar calcMgrCmdLine.jar -gk を使用してキーを生成する必要があります。これによって 83qosW4LbxVGp6uBwqtQWb のようなキーが生成され、これをユーザー名とパスワードの暗号化に使用できます。ユーザー名を暗号化する場合、パスワードも暗号化する必要があります。CalcMgrCmdLine.jar の使用の詳細は、[コマンド・ライン・ランチャを使用した Essbase ビジネス・ルールの起動](#)を参照してください。

 **ノート:**

列および行の軸のみを持つ MDX 式がサポートされています。

構文:

```
RUNJAVA com.hyperion.calcMgr.common.cdf.MDXDataCopy
  <Key used to encrypt user name and password>
  <Encrypted user name>
  <Encrypted password>
  <Source Application>
  <Source Database>
  <Target Application>
  <Target Database>
  <MDX expression that defines column members>
  <MDX expression that defines row members>
  <Source Member Mappings>
  <Target Member Mappings>
  <Target POV columns>
  <Rows Per Page>
  <Log File>
  <Source Server>
  <Target Server>
  <Where MDX>
  <"true" or "false">
```

- <Source Member Mappings>および<Target Member Mappings>が使用されるのは、ソースのメンバー名がターゲットと異なるときです。これらはカンマで区切る必要があります。関数は指定できません。
- <Target POV columns>が使用されるのは、ターゲットのディメンションがソースよりも多い場合です。
- <ROWS PER PAGE>は、ターゲットの更新をトリガーするために使用されます。デフォルトの-1 では、更新が行われるのは完了したグリッドが移入されるときです。30 に設定すると、30 行が入力されるとすぐに更新がトリガーされます。その後、次の 30 行に続きます。
- <Source Server>および<Target Server>が使用されるのは、ソース・サーバーとターゲット・サーバーが異なるときです。同じ場合はプレース・ホルダーとして""を使用してください。

- <Where MDX>が使用されるのは、スクリプトに引数を指定するときです。
- <"true"または"false">パラメータでは、"true"によってログ・ファイルが付加、"false"によってログ・ファイルがクリアされます。このパラメータを空にしておくと、デフォルトは"true"です。

カスタム定義関数が実行される JVM のメモリーを増やすには、Oracle Essbase の opmn.xml ファイルを編集し、ESS_JVM_OPTION1 の value プロパティの数を変更します。<variable id="ESS_JVM_OPTION1" value="-Xmx256M"/>というエントリでは、JVM のメモリーとして 256MB が設定されます

例:

```
RUNJAVA com.hyperion.calcmgr.common.cdf.MDXDataCopy
      "vta2a536uC/wyX8jM2GlHA==" /* key */

"qza9xZxUX+srS1GrlR2Qboeq77InwkNSQsvBHL02siidSEglMndoIpneZVPK0tWt" /*
user */
      "eldizBRiYqdPdXl9Bpu/
HIE+YIgvRBa59aTjTDWpWM0T+ZEPjF+zyINfE0KHX0qP" /* password */
      "Sample" /* from application */
      "Basic" /* from database */
      "Samp2" /* to application */
      "Basic" /* to database */
      "crossjoin({[Jan],[Feb],[March]},{[Budget]})" /* MDX that defines
the column members */
      "crossjoin(crossjoin({[Sales]},{[100].CHILDREN}),
{[Connecticut]})" /* MDX that defines the row members */
      "Jan, Feb" /* source member mappings */
      "January, February" /* target member mappings */
      "VP, IT" /* Target POV columns, members from dimensions that do
not exist on the source*/
      "-1" /* rows per page */
      "c:\\Temp\\cpdata3.log"; /* log file could be empty */
      ""source server
      ""destination server
      ""where MDX
```

@CalcMgrMDXDataCopyRTP

目的:

実行時プロンプトを使用してデータベース間でデータをコピーします。または、MDX スクリプトを使用してサーバー間でデータをコピーします。

RTP は次の引数で使用できます:

- 列 MDX
- 行 MDX
- Where MDX
- ソース・マッピング
- ターゲット・マッピング

- POV 列

これらの引数に対しては変数を使用するかわりに \$1、\$2、\$3 などを使用します。その後、引数内で @LIST 関数を使用してこれらの変数を渡します。最初の引数の入力が \$1、2 番目の引数が \$2 のように置換します。

構文:

Java クラ

```
ス: .hyperion.calcmgr.common.cdf.MDXDataCopy.dataCopy (String,String,String,String,String,String,String,String,String,String,String,String,String,String,String)
```

CDF 仕様:

```
@CalcMgrMDXDataCopyRTP (key,user,password,sourcApplication,sourrcDatabase,dtnApplication,dtnDatabase,columQuery,rowQuery,whereQuery,sourcMappings,targetMappings,targetPOVColumns,maxRowsPerPage,arguments,loggerName)
```

@CalcMgrMDXDataCopyRPT を使用したデータベース間でのデータのコピー

この例では、実行時プロンプト引数に対して変数を使用するかわりに、\$1、\$2、\$3 などを使用します。引数で @LIST 関数を使用して、これらの変数を渡しました。最初の引数の入力が \$1、2 番目の引数の入力が \$2 のように置換します。

例:

```
FIX ("FY14", {Scenario}, {Version}, {Department})
    "Jan" (
        @CalcMgrMDXDataCopyRTP (
            "fht1qmevLWMqKSHF2yusKS",
            "py9FTBsRCwp8afii9ntCAwSvVUXr3oZbpxFMqobx5Jwkt34Cfxvhs21z4WmG9KT",
            "KvwlKgs56CoilHQsrklUwiVi9zyoE+Llel4BrYBHiJz9PF6ZDVQ/
            rF5taip707CL",
            "Vision",
            "Plan1",
            "",
            "ASO_401",
            "ASO_401",
            "",
            "{[$1]}" ,
            "NON EMPTY (Crossjoin({Descendants ([&QRFPer1],
            [Period].Levels(0)),Descendants ([&QRFPer2],
            [Period].Levels(0)),Descendants ([&QRFPer3],
            [Period].Levels(0)),Descendants ([&QRFPer4], [Period].Levels(0))},
            Crossjoin ({&QRFYr1}, [&QRFYr2], [&QRFYr3], [&QRFYr4], Crossjoin ({[$3]},
            Crossjoin (Descendants ([P_TP], [Product].Levels(0)), {[Units], [5800],
            [9000]}))))))",
            " ([$2])",
            "",
            "",
            "",
            "",
            "-1",
            @LIST (@name ({Scenario}), @name ({Department}), @name ({Version})),
            "c:/Temp/CM_DCopyVW1.log"
        );
```

```
)
ENDFIX
```

@CalcMgrMDXDataCopyServer

目的:

MDX スクリプトを使用してデータベース間またはサーバー間でデータをコピーしますが、Where 文を使用できます。

構文:

Java クラス:

```
CDF 仕様: @CalcMgrMDXDataCopyServer(Encryption Key, user,
password, sourceApplicaionName, sourceDatabaseName,
sourceServer, targetApplicationName, targetDatabaseName, targetServer,
columnAxisMDX, rowAxisMDX, whereMDX, sourceMemberMappings,
targetMemberMappings, targetPOVcols, maxRowsPerPage, loggingFileName)
```

@CalcMgrMDXExport

目的:

MDX 式を使用してデータをエクスポートします。この関数を使用し、ブロック・ストレージ・アプリケーションの計算スクリプトを使用して集約ストレージ・データベースからデータをエクスポートします。@CalcMgrMDXExport は、calcMgrCmdLine.jar を使用して暗号化トークンを生成し、暗号化されたユーザー名とパスワードを使用します。たとえば、ユーザー **admin** を暗号化するには、`java -jar calcMgrCmdLine.jar -encrypt -key vta2a536uC/wyX8jM2G1HA== admin` を使用します

ノート:

列および行の軸のみを持つ MDX 式がサポートされています。

構文:

Java クラス:

```
com.hyperion.calcMgr.common.cdf.MDXExport.exportData(String, String, String,
String, String, String, String, String, String, String, String, String, String, String, String)
```

CDF 仕様:

```
@CalcMgrMDXExport(key, user, pwd, fileName, application, database, server, column
AxisMdx, rowAxisMdx, seperator, missing, useUniqueName, useAliasName, supressZeros, rowHeaders)
```

@CalcMgrMDXExport を使用したデータベースからのデータのエクスポート

構文:

```
RUNJAVA com.hyperion.calcMgr.common.cdf.MDXExport
<KEY used to encrypt user name and password>
```

```

<Encrypted user name>
<Encrypted password>
<Export file>
<Application>
<Database>
<Server>
  <MDX expression that defines column members>
  <MDX expression that defines row members>
  <Separator>
  <#MISSING Value>
  <Use Unique Name> "true" or "false"
  <Use Alias Name> "true" or "false"
  <Supress Zeros> "true" or "false"
  <Show Row Headers> "true" or "false"
  /* where mdx */
  /* source member mappings */
  /* target member mappings */

```

例:

```

RUNJAVA com.hyperion.calcmgr.common.cdf.MDXExport
  "fht1qmevLWMqKSHF2yusKS" /* key */
  "py9FTBsRCwp8afii9ntCAwSvVUXr3oZbxpxFMqobx5Jwkt34Cfxvhs21z4WmG9KT" /*
user */
  "KvwlKgs56CoilHQsrk1UwiVi9zyoE+Llel4BrYBHiJz9PF6ZDVQ/rF5taip707CL" /*
password */
  "c:\\Temp\\export_W1.txt" /* export file */
  "Vision" /* source application name */
  "Plan1" /* source Database name */
  "server1" /* server */
  "[{Actual}]" /* columnAxisMDX */
  "NON EMPTY (Crossjoin({Descendants([&QRFPer1],
[Period].Levels(0)),Descendants([&QRFPer2],
[Period].Levels(0)),Descendants([&QRFPer3],
[Period].Levels(0)),Descendants([&QRFPer4],[Period].Levels(0))},
  Crossjoin({[Department]}},
  Crossjoin([&QRFPer1],[&QRFPer2],[&QRFPer3],[&QRFPer4]},
  Crossjoin({[Version]}},CrossJoin([Basedata]},
  {[Units],[5800],[4110]}))))))" /* row AxisMdx */
  "," /* separator */
  "" /* #MISSING */
  "false" /* use unique name */
  "false" /* use alias name */
  "true" /* suppress zero's */
  "true" /* show row headers */
  "[{P_000}]" /* where mdx */
  "Jan, Feb, Mar" /* source member mappings */
  "January, February, March" /* target member mappings */

```

@CalcMgrMDXExportRTP

目的:

MDX スクリプトを使用し、実行時プロンプトを使用してキューブからデータをエクスポートします。

構文:

Java クラ

```
ス:.hyperion.calcmgr.common.cdf.MDXExport.dataExport (String, String, String,
String, String, String, String, String, String, String, String, String, String, Stri
ng, String, String, String[])
```

CDF 仕様:

```
@CalcMgrMDXExportRTP (key, user, pwd, fileName, application, database, server, col
umnAxisMdx, rowAxisMdx, seperator, missing, useUniqueName, useAliasName, supress
Zeros, rowHeaders, args)
```

@CalcMgrMDXExportRPT を使用したデータベースからのデータのエクスポート

次の引数では実行時プロンプトを使用できます:

- 列 MDX
- 行 MDX
- Where MDX
- ソース・マッピング
- ターゲット・マッピング
- POV 列

これらの引数に対しては変数を使用するかわりに \$1、\$2、\$3 などを使用します。引数で @LIST 関数を使用して、これらの実行時プロンプトを渡します。最初の引数の入力 が \$1、2 番目の引数が \$2 のように置換します。

統計関数

次も参照:

- [@CalcMgrExcelAVEDEV](#)
- [@CalcMgrExcelDEVSQ](#)
- [@CalcMgrExcelLARGE](#)
- [@CalcMgrExcelMEDIAN](#)
- [@CalcMgrExcelSMALL](#)
- [@CalcMgrExcelSTDEV](#)
- [@CalcMgrExcelVAR](#)
- [@CalcMgrExcelVARP](#)

@CalcMgrExcelAVEDEV

目的:

データ・ポイントの平均値に対する各データ・ポイントの絶対偏差の平均を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.AVEDEV(double[])`

CDF 仕様: `@CalcMgrExcelAVEDEV(numbers)`

@CalcMgrExcelDEVSQ

目的:

偏差の平方和を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.DEVSQ(double[])`

CDF 仕様: `@CalcMgrExcelDEVSQ(numbers)`

@CalcMgrExcelLARGE

目的:

n 番目に大きい数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.NTHLARGEST(double[], double)`

CDF 仕様: `@CalcMgrExcelLARGE(values, rank)`

@CalcMgrExcelMEDIAN

目的:

指定した数値の中央値を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.MEDIAN(double[])`

CDF 仕様: `@CalcMgrExcelMEDIAN(values)`

@CalcMgrExcelSMALL

目的:

n 番目に小さい数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.NTHSMALLEST(double[], double)`

CDF 仕様: `@CalcMgrExcelSMALL(values, rank)`

@CalcMgrExcelSTDEV

目的:

サンプルを基に標準偏差を推定します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.STDEV(double[])
```

CDF 仕様: @CalcMgrExcelSTDEV(values)

@CalcMgrExcelVAR

目的:

サンプルを基に分散を推定します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.VAR(double[])
```

CDF 仕様: @CalcMgrExcelVAR(values)

@CalcMgrExcelVARP

目的:

母集団を基に分散を推定します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.excel.cdf.ExcelStatisticalFunctions.VARP(double[])
```

CDF 仕様: @CalcMgrExcelVARP(values)

文字列関数

次も参照:

- [@CalcMgrCompare](#)
- [@CalcMgrConcat](#)
- [@CalcMgrDecimalFormat](#)
- [@CalcMgrDoubleFromString](#)
- [@CalcMgrDoubleToString](#)
- [@CalcMgrDQuote](#)
- [@CalcMgrEndsWith](#)

- @CalcMgrFindFirst
- @CalcMgrFindLast
- @CalcMgrFormatDouble
- @CalcMgrGetListCount
- @CalcMgrGetListItem
- @CalcMgrIndexOf
- @CalcMgrIntegerToString
- @CalcMgrLastIndexOf
- @CalcMgrLowercase
- @CalcMgrMatches
- @CalcMgrMessageFormat
- @CalcMgrPadText
- @CalcMgrUppercase
- @CalcMgrRemoveQuotes
- @CalcMgrRemoveDQuotes
- @CalcMgrRemoveSQuotes
- @CalcMgrReplaceAll
- @CalcMgrReplaceFirst
- @CalcMgrSortAndReturn
- @CalcMgrSortList
- @CalcMgrSortValues
- @CalcMgrSplit
- @CalcMgrSQuote
- @CalcMgrStartsWith
- @CalcMgrStringsToString
- @CalcMgrSubstring
- @CalcMgrTextLength
- @CalcMgrTrim

@CalcMgrCompare

目的:

2 つの文字列を比較します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.compare(String,String,boolean)
```

CDF 仕様: @CalcMgrCompare(text1, text2,ignoreCase)

@CalcMgrConcat

目的:

指定された文字列をこの文字列の最後に連結します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.concat(String, String)
```

CDF 仕様: @CalcMgrConcat(text1, text2)

@CalcMgrDecimalFormat

目的:

指定されたフォーマット文字列を使用してフォーマットされた文字列を返します。

形式指定については、**Java** ドキュメントで小数の形式を確認してください。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.decimalFormat(String, double)
```

CDF 仕様: @CalcMgrDecimalFormat(formatString, value)

@CalcMgrDoubleFromString

目的:

文字列から倍精度浮動小数点に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.getDoubleFromString(String)
```

CDF 仕様: @CalcMgrDoubleFromString(text)

@CalcMgrDoubleToString

目的:

倍精度浮動小数点を文字列に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.MaxLFFunctions.doubleToString(double)
```

CDF 仕様: @CalcMgrDoubleToString(doubleNumber)

@CalcMgrDQuote

目的:

二重引用符で囲まれていない場合、二重引用符をテキストに追加します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.StringFunctions.dQuote(String)`

CDF 仕様: `@CalcMgrDQuote(text)`

@CalcMgrEndsWith

目的:

この文字列が指定された接尾辞で終わるかどうかをテストします

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.StringFunctions.endsWith(String, String)`

CDF 仕様: `@CalcMgrEndsWith(text, suffix)`

@CalcMgrFindFirst

目的:

指定した正規表現と一致するこの文字列の最初の部分文字列を検出します。

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.StringFunctions.findFirst(String, String, boolean)`

CDF 仕様: `@CalcMgrFindFirst(text, regExpr, ignoreCase)`

@CalcMgrFindLast

目的:

指定した正規表現と一致するこの文字列の最後の部分文字列を検出します。

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.StringFunctions.findLast(String, String, boolean)`

CDF 仕様: `@CalcMgrFindLast(text, regExpr, ignoreCase)`

@CalcMgrFormatDouble

目的:

指定されたフォーマット文字列を使用してフォーマットされた文字列を返します。

形式指定については、**Java** ドキュメントで印刷形式を確認してください。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.formatDouble(String, double)`
)

CDF 仕様: `@CalcMgrFormatDouble(formatString, value)`

@CalcMgrGetListCount

目的:

リストのアイテム数を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.getListCount(String[])`

CDF 仕様: `@CalcMgrGetListCount(list)`

@CalcMgrGetListItem

目的:

インデックス・アイテムをリストから返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.getListItem(String[],int)`

CDF 仕様: `@CalcMgrGetListCount(list,index)`

@CalcMgrIndexOf

目的:

指定したインデックスから開始した場合の指定した部分文字列の最初の出現のこの文字列内のインデックスを返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.indexOf(String,String,int)`

CDF 仕様: `@CalcMgrIndexOf(text,searchText,begIndex)`

 **ノート:**

`@CalcMgrIndexOf` または `@CalcMgrLastIndexOf`(後述)のインデックスとして `-1` を使用した場合、文字列全体が検索されます。

@CalcMgrIntegerToString

目的:

整数を文字列に変換します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.MaxLFunctions.integerToString(int)`

CDF 仕様: `@CalcMgrIntegerToString(integerNumber)`

@CalcMgrLastIndexOf

目的:

指定したインデックスから開始して逆方向に検索した場合の指定した部分文字列の最後の出現のこの文字列内のインデックスを返します

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.StringFunctions.lastIndexOf(String, String, int)`

CDF 仕様: `@CalcMgrLastIndexOf(text, searchText, begIndex)`

@CalcMgrLowercase

目的:

テキストを小文字に変換します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.StringFunctions.toLower(String)`

CDF 仕様: `@CalcMgrLowercase(text)`

@CalcMgrMatches

目的:

この文字列の最初の部分文字列が指定した正規表現と一致する場合、`true` を返します。

正規表現については、Java doc の"`java.util.regex.Pattern`"を参照してください。

構文:

Java クラス:

`com.hyperion.calcmgr.common.cdf.StringFunctions.matches(String, String, boolean)`

CDF 仕様: `@CalcMgrMatches(text, regExpr, ignoreCase)`

@CalcMgrMessageFormat

目的:

指定されたパターンで文字列を作成し、指定された引数のフォーマットに使用します。

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.messageFormat (String, String[])`
CDF 仕様: `@CalcMgrMessageFormat (text, parameters)`

@CalcMgrPadText

目的:

テキストの前後にパディング・テキストを入力し、長さを調整します。
たとえば、`@CalcMgrPadText ("01", 5, "0", @_true)` は **01000** を返します
`@CalcMgrPadText ("01", 5, "0", @_false)` は **00001** を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.padText (String, int, String, boolean)`
CDF 仕様: `@CalcMgrPadText (text, length, padText, append)`

@CalcMgrUppercase

目的:

テキストを大文字に変換します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.toUpper (String)`
CDF 仕様: `@CalcMgrUppercase (text)`

@CalcMgrRemoveQuotes

目的:

テキスト文字列を囲む一重または二重引用符を削除します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.removeQuotes (String)`
CDF 仕様: `@CalcMgrRemoveQuotes (text)`

@CalcMgrRemoveDQuotes

目的:

テキスト文字列を囲む二重引用符を削除します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.removeDQuotes (String)`

CDF 仕様: `@CalcMgrRemoveDQuotes (text)`

@CalcMgrRemoveSQuotes

目的:

テキスト文字列を囲む一重引用符を削除します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.removeSQuotes (String)`

CDF 仕様: `@CalcMgrRemoveSQuotes (text)`

@CalcMgrReplaceAll

目的:

指定された正規表現と一致するこの文字列の各部分文字列を、指定された置換で置き換えます

正規表現については、`java.util.regex.Pattern` の **Java doc** を参照してください

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.replaceAll (String,String,String)`

CDF 仕様: `@CalcMgrReplaceAll (text, regExpr, replacement)`

@CalcMgrReplaceFirst

目的:

指定された正規表現と一致するこの文字列の最初の部分文字列を、指定された置換で置き換えます

正規表現については、`java.util.regex.Pattern` の **Java doc** を参照してください

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.replaceFirst (String,String,String)`

CDF 仕様: `@CalcMgrReplaceFirst (text, regExpr, replacement)`

@CalcMgrSortAndReturn

目的:

値に基づいてリスト内のアイテムをソートし、上位 *n* の要素を返します

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.sortAndReturn(String[],double[],int,boolean)`

CDF 仕様: `@CalcMgrSortAndReturn(list, values, topN, sortAscending)`

@CalcMgrSortList

目的:

リスト内のアイテムをソートします

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.sortList(String[],boolean,boolean)`

CDF 仕様: `@CalcMgrSortList(list, caseSensitive, sortAscending)`

@CalcMgrSortValues

目的:

値に基づいてリスト内のアイテムをソートします

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.sortValues(double[],boolean)`

CDF 仕様: `@CalcMgrSortValues(values, sortAscending)`

@CalcMgrSplit

目的:

regex に基づいてテキストを分割します

正規表現については、`java.util.regex.Pattern` の **Java doc** を参照してください

構文:

Java クラス:
`com.hyperion.calcmgr.common.cdf.StringFunctions.split(String,String)`

CDF 仕様: `@CalcMgrSplit(text, regex)`

@CalcMgrSQuote

目的:

一重引用符で囲まれていない場合、一重引用符をテキストに追加します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.StringFunctions.sQuote(String)`

CDF 仕様: @CalcMgrSQuote(text)

@CalcMgrStartsWith

目的:

この文字列が指定された接頭辞で始まるかどうかをテストします

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.startsWith(String,String)
```

CDF 仕様: @CalcMgrStartsWith(text, prefix)

@CalcMgrStringsToString

目的:

文字列配列をデリミタを使用する文字列に変換します

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.MaxLFunctions.stringsToString(String[],String)
```

CDF 仕様: @CalcMgrStringsToString(strings, seperator)

@CalcMgrSubstring

目的:

この文字列の部分文字列である新規文字列を返します。

部分文字列は、startIndex で始まり、index endIndex - 1 の文字までです。このため、部分文字列の長さは endIndex - startIndex です。

endIndex がゼロ未満の場合、endIndex が最後の文字のインデックスになります。

構文:

Java クラス:

```
com.hyperion.calcmgr.common.cdf.StringFunctions.substring(String,int,int)
```

CDF 仕様: @CalcMgrSubstring(text, startIndex, endIndex)

@CalcMgrTextLength

目的:

テキストの長さを返します

構文:

Java クラス: com.hyperion.calcmgr.common.cdf.StringFunctions.length(String)

CDF 仕様: @CalcMgrTextLength(text)

@CalcMgrTrim

目的:

テキスト文字列の先頭および末尾にあるスペースを削除します

構文:

Java クラス: `com.hyperion.calcmgr.common.cdf.StringFunctions.trim(String)`

CDF 仕様: `@CalcMgrTrim(text)`

コンポーネントへの関数の挿入

選択できる関数のタイプは使用しているアプリケーションのタイプおよびコンポーネントのタイプによって異なります。

Oracle Hyperion Financial Management、Oracle Hyperion Planning または Oracle Essbase ブロック・ストレージ・アプリケーションを使用している場合、式、スクリプト、条件およびメンバー範囲コンポーネントに関数を挿入できます。

Oracle General Ledger および Planning で使用される Essbase 集約ストレージ・アプリケーションも含めて、Essbase 集約ストレージ・アプリケーションを使用する場合は、視点コンポーネントにのみ関数を挿入できます。Essbase 集約ストレージ・アプリケーションでは、メンバー・セット関数タイプから選択する関数タイプは 1 つのみです。(このトピックの 2 番目の手順を参照してください。)

式、スクリプト、条件およびメンバー範囲のコンポーネントに関数を挿入するには:

1. 次のいずれかのタスクを行います:

- 関数を挿入するコンポーネントを含むビジネス・ルールを開きます。次にビジネス・ルールのフロー・チャートに関数を挿入するコンポーネントを選択します。
- 関数を挿入するコンポーネントを含むテンプレートを開きます。次にテンプレートのフロー・チャートに関数を挿入するコンポーネントを選択します。
- 関数を挿入する式コンポーネントを開きます。
- 関数を挿入するスクリプト・コンポーネントを開きます。

2. 次のいずれかのタスクを行います:

- 式コンポーネントに関数を挿入するには、「**式**」タブで、「**式**」行内をクリックして、「**アクション**」アイコンを選択し、「**関数**」を選択します。
- スクリプト・コンポーネントに関数を挿入するには、「**スクリプト**」タブで、「**関数とそのパラメータの挿入**」アイコンをクリックします。
- メンバー範囲コンポーネントに関数を挿入するには、「**メンバー範囲**」タブで、ディメンションの「**値**」列内をクリックして、「**アクション**」アイコンを選択し、「**関数**」を選択します。
- 条件コンポーネントに関数を挿入するには、「**条件**」タブで、「**条件ビルダー**」アイコンをクリックします。条件ビルダーで、「**式**」、「**関数**」または「**値**」から、「**アクション**」アイコンを選択して、「**関数**」を選択します。

3. 関数セレクトで、次のいずれかのタスクを行います:

 ノート:

関数セレクタを開くコンテキストによっては、使用できる関数タイプは、[関数の操作](#)で説明されている関数タイプより限られる場合があります。

- 「**カテゴリ**」から、関数タイプを選択できる場合は、1つの関数を選択するか、または「**すべての関数**」を選択してすべての関数タイプの関数を表示します。カテゴリの関数、またはすべての関数がカテゴリの選択の下のリストに表示されます。
 - 「**カテゴリ**」の関数タイプ内で選択できない場合は、4に進みます。
4. 関数のリストから関数を選択します。
 5. 関数のパラメータを入力します。(Planning および Essbase コンポーネントで使用できる関数とパラメータ(Oracle Fusion General Ledger アプリケーションで使用されるものを含む)の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。Financial Management コンポーネントで使用できる関数およびパラメータの説明については、『Oracle Hyperion Financial Management 管理者ガイド』を参照してください。)
 6. 「OK」をクリックします。

視点コンポーネントに関数を挿入するには:

1. システム・ビューで、Essbase アプリケーション・タイプ、アプリケーション、データベース、「**ルール**」と展開します。
2. 関数を挿入する視点コンポーネントを含むビジネス・ルールを開きます。次にビジネス・ルールのフロー・チャートに関数を挿入する視点コンポーネントを選択します。
3. 視点(POV)コンポーネントに関数を挿入する場合は、「**視点(POV)**」タブで、ディメンションの「**値**」列内をクリックして、「**アクション**」アイコンを選択し、「**関数**」を選択します。
4. 「**関数セレクタ**」で、次の関数のリストから関数を選択します:
 - @Attribute
 - @Level0Descendants
 - @Siblings
 - @UDA

 ノート:

Oracle General Ledger で使用されるアプリケーションも含めて、Essbase 集約ストレージ・アプリケーションを使用する場合、メンバー・セット関数のみ使用できます。

5. 関数のパラメータを入力します。(関数およびパラメータの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。)
6. 「OK」をクリックします。

スマート・リストの操作

スマート・リストは **Planning** アプリケーションの **Oracle Hyperion Planning** データ・フォームのセルからアクセスできるカスタムのドロップダウン・リストです。

データを入力するデータ・フォームのセルをクリックすると、セルに入力するかわりにドロップダウン・リストからアイテムを選択できます。スマート・リストを含むセルには入力できません。

Oracle Hyperion Calculation Manager では、変数、または式、スクリプト、条件およびメンバー範囲のコンポーネントにスマート・リストを挿入できます。

スマート・リストの挿入

Oracle Hyperion Planning ビジネス・ルール、式コンポーネントまたはスクリプト・コンポーネントのスマート・リストを使用します。

スマート・リストは、**Planning** 管理者が指定する特定のデータ・セル内の **Planning** データ・フォームで使用可能です。スマート・リストは、ユーザーが選択できるオプションを含む、カスタマイズされたドロップダウン・リストです。

スマート・リストを挿入するには:

1. 次のいずれかのタスクを行います:
 - スマート・リストを挿入するコンポーネントを含むビジネス・ルールを開きます。次に、ビジネス・ルールのフロー・チャートで、スマート・リストを挿入する式またはスクリプト・コンポーネントを選択します。
 - スマート・リストを挿入する式コンポーネントを開きます。
 - スマート・リストを挿入するスクリプト・コンポーネントを開きます。
2. 次のいずれかのタスクを行います:
 - ビジネス・ルールにスマート・リストを挿入するには、「**スクリプト**」タブで、「**スマートリストの挿入**」アイコンをクリックします。
 - 式コンポーネントにスマート・リストを挿入するには、「**式**」タブで「**式**」行内をクリックして「**アクション**」アイコンを選択し、「**スマート・リスト**」を選択します。
 - スクリプト・コンポーネントにスマート・リストを挿入するには、「**スクリプト**」タブで、「**スマートリストの挿入**」アイコンをクリックします。
3. 「**保存**」をクリックします。

Planning 数式の操作

Oracle Hyperion Calculation Manager のグラフィカル・ルールまたはスクリプト・ルールで、**Oracle Hyperion Planning** の数式を使用します。

次のタイプの数式を使用できます。

- [スマートリスト](#)
- [ディメンション](#)

- [Planning ユーザー変数](#)
- [期間](#)
- [シナリオ](#)
- [相互参照](#)
- [Workforce キューブ年次累計](#)
- [文字列の ID の取得](#)

スマートリスト

スマート・リストは、変数として"Product Channel"=[[Channel.Retail]]などの数式に含めることができます。

"Product Channel"はタイプが「スマート・リスト」の勘定科目、Channel はスマート・リスト名、Retail はスマート・リスト・エントリです。**Retail** に対するスマート・リストの ID が 2 の場合、メンバー式で Channel.Retail を 2 に置き換えます(アプリケーションはスマート・リストを数字として扱います)。**Retail** に対するスマート・リストの ID が 2 の場合、計算に 2 をおき、データベースに 2 を保管します。

Calculation Manager の構文:

```
[[SLName.entryname]]
```

例:

次の構文:

```
FIX (Mar, Actual, Working, FY15, P_000, "111")  
  "Product Channel" =[[Channel.Retail]] ;  
ENDFIX
```

によって次のスクリプトが返されます:

```
FIX (Mar, Actual, Working, FY15, P_000, "111")  
  "Product Channel" =2 ;  
ENDFIX
```

ディメンション

Dimension(dimTag) は事前定義済のディメンションの名前を返します。

dimtag は次のとおりです。

- DIM_NAME_PERIOD
- DIM_NAME_YEAR
- DIM_NAME_ACCOUNT
- DIM_NAME_ENTITY
- DIM_NAME_SCENARIO
- DIM_NAME_VERSION

- DIM_NAME_CURRENCY

Calculation Manager の構文:

```
[[Dimension("DIM_NAME_ENTITY")]]
```

例:

```
CALC DIM ([[Dimension("DIM_NAME_ENTITY")]]);
```

このアプリケーションでは Entity の名前が **Entity** であるため、前述のスク립トは次を返します。

```
CALC DIM ("Entity");
```

エンティティ・ディメンションの名前が **Cost Center** の場合は、次を返します。

```
CALC DIM ("Cost Center");
```

Planning ユーザー変数

Planning ユーザー変数はユーザー変数のメンバーを返します。

Calculation Manager の構文:

```
[[PlanningFunctions.getUserVarValue("xyz")]]
```

例:

```
FIX (Feb, Actual, Working, P_000,
[[PlanningFunctions.getUserVarValue("Entity View")]] )
"5800" = 40;
ENDFIX
```

このアプリケーションには **Entity View** という名前の Planning ユーザー変数があります。このユーザーに対して、これは **112** に設定されます。したがって、前述の例ではスク립トが次を返します。

```
FIX (Feb, Actual, Working, P_000, "112")
"5800" = 40;
ENDFIX
```

期間

次も参照:

- [Period\(periodName\)](#)
- [NumberofPeriodsinYear](#) および [NumberofYears](#)

Period(periodName)

Period(periodName) は指定した期間を返します。

期間名のオプションは、次のとおりです:

- FIRST_QTR_PERIOD
- SECOND_QTR_PERIOD
- THIRD_QTR_PERIOD
- FOURTH_QTR_PERIOD
- FIRST_PERIOD
- LAST_PERIOD

Calculation Manager の構文:

```
[[Period("FIRST_QTR_PERIOD")]]
```

例:

次の構文:

```
FIX ( Mar, Actual, Working, P_000, "6100", FY15 )
    "120" = [[Period("FIRST_QTR_PERIOD")]];
ENDFIX
```

によって次のスクリプトが返されます:

```
FIX (Mar, Actual, Working, P_000, "6100", FY15)
    "120" = "Mar";
ENDFIX
```

NumberOfPeriodsinYear および NumberOfYears

NumberOfPeriodsinYear は年の期間を返し、NumberOfYears はアプリケーションの年数を返します。

Calculation Manager の構文:

```
[[NumberOfPeriodsinYear]]
```

```
[[NumberOfYears]]
```

例:

次の構文:

```
FIX (Mar, Actual, Working, P_000, "6100", FY15)
    "120"=[[NumberOfPeriodsinYear]];
    "120"=[[NumberOfYears]];
ENDFIX
```

によって次のスクリプトが返されます:

```
FIX (Mar, Actual, Working, P_000, "6100", FY15)
    "120"=12;
```

```
"120"=9;
ENDFIX
```

シナリオ

シナリオ・プランニング範囲の時間範囲およびモジュールの「プランニングと予測の準備」構成タスク情報を式として公開し、次のことを指定できます:

- **開始年:** 指定されたシナリオの開始年を文字列形式で返します。
- **終了年:** 指定されたシナリオの終了年を文字列形式で返します。
- **開始月:** 指定されたシナリオの開始月を文字列形式で返します。
- **終了月:** 指定されたシナリオの終了月を文字列形式で返します。
- **モジュール開始年:** 指定されたモジュールおよびシナリオの開始年を文字列形式で返します。
- **モジュール終了年:** 指定されたモジュールおよびシナリオの終了年を文字列形式で返します。
- **モジュール開始期間:** 指定されたモジュールおよびシナリオの開始期間を文字列で返します。
- **モジュール終了期間:** 指定されたモジュールおよびシナリオの終了期間を文字列形式で返します。
- **モジュール・プラン開始年:** 「現在の会計年」の場合は"1"、「次の会計年」の場合は"0"の整数値を返します。

ノート:

モジュール開始年、モジュール終了年、モジュール開始期間、モジュール終了期間およびモジュール・プラン開始年は、**Planning** モジュールおよび **Oracle Strategic Workforce Planning Cloud** アプリケーションに対してのみ有効で、"PlanningFunctions"構文を必要とします。

Calculation Manager の構文:

```
[[getStartYear("ScenarioName")]]
[[getEndYear("ScenarioName")]]
[[getStartMonth("ScenarioName")]]
[[getEndMonth("ScenarioName")]]
[[PlanningFunctions.getModuleStartYear("ModuleName","ScenarioName")]]
[[PlanningFunctions.getModuleEndYear("ModuleName","ScenarioName")]]
[[PlanningFunctions.getModuleStartPeriod("ModuleName","ScenarioName")]]

[[PlanningFunctions.getModuleEndPeriod("ModuleName","ScenarioName")]]
[[PlanningFunctions.isPlanStartYearSameAsCurrentFiscalYear("ModuleName")]]
```

シナリオ名

ScenarioName は入力されたシナリオ・メンバー、または Oracle Hyperion Calculation Manager の実行時プロンプト・メンバー・タイプの変数です。メンバーは二重引用符で囲む必要があります。たとえば、[[getStartYear("Actual")]] のようになります。

ScenarioName には、次の形式で代替変数も指定できます：

- `getSubVarValue("CubeName", "SubstitutionVariableName")` は、CubeName によって定義されたキューブ・レベルで、指定された代替変数の代替変数値を返します
- `getSubVarValue("SubstitutionVariableName")` は、すべてのキューブのアプリケーション・レベルで、指定された代替変数の代替変数値を返します

CubeName および SubstitutionVariableName は二重引用符で囲む必要があり、& または {} は使用しないでください。次の例 3 および例 4 を参照してください。

モジュール名

ModuleName は、Oracle Hyperion Planning および Strategic Workforce Planning の現在のモジュールの事前定義済名である必要があります。パラメータとして ModuleName を使用するこれらの式は、ScenarioName の「プランニングと予測の準備」構成タスクの時間範囲情報を使用し、アプリケーション内で計画および予測シナリオに対してモジュールごとに時間範囲が異なる場合があります。

計画および予測の他のシナリオも、これらの式で参照できますが、ディメンション・エディタでシナリオ・プランニング範囲に対して設定された時間範囲のみを返し、アプリケーション内のすべてのモジュールに対して同じになります。シナリオ・プランニング範囲の詳細は、[シナリオについて](#) を参照してください。

「プラン開始年」は、別の「プランニングと予測の準備」構成タスクであり、モジュールの計画シナリオに固有です。この式は、「現在の会計年」に設定されている場合は"1"、「次の会計年」に設定されている場合は"0"の整数値を返すように、その設定に関連した値を返します。この式は、値としてディメンション・メンバーを返さないため、条件文のコンテキスト内でのみ使用できます。次の例 5 を参照してください。

ModuleName の有効な値は次のとおりであり、特定のビジネス・プロセスに適用できます：

表 8-8 有効な ModuleName の値

ビジネス・プロセス	有効な ModuleName の値
Planning モジュール	<ul style="list-style-type: none"> • "資本" • "財務" • "プロジェクト" • "要員"
Strategic Workforce Planning	<ul style="list-style-type: none"> • "戦略要員" • "要員"

 ノート:

ModuleName の値は二重引用符で囲む必要があり、大文字と小文字が区別されません。

例 1

次の構文({rtpScenario}はタイプが *member* の実行時プロンプト変数で、デフォルト値は *actual*):

```
FIX({rtpScenario}, [[getStartYear({rtpScenario})]]):  
[[getEndYear({rtpScenario})]],  
  [[getStartMonth({rtpScenario})]]:[[getEndMonth({rtpScenario})]]  
    FIX ( Working, P_000, "111")  
      "5800" = 5500;  
    ENDFIX  
  ENDFIX
```

によって次のスクリプトが返されます:

```
FIX ("Actual", "FY10" : "FY18", "Jan" : "Dec")  
  FIX (Working, P_000, "111")  
    "5800" = 5500;  
  ENDFIX  
ENDFIX
```

例 2

次の構文:

```
FIX({rtpScenario}, [[PlanningFunctions.getModuleStartYear("CAPITAL",  
{rtpScenario})]]):  
[[PlanningFunctions.getModuleEndYear("CAPITAL",{rtpScenario})]] ,  
"Jan" : "Dec")  
  FIX(OEP_Working, P_000, "111")  
    "5800" = 5500;  
  ENDFIX  
ENDFIX
```

によって、次のスクリプトが返されます。ここで、資本モジュールの「プランニングと予測の準備」構成タスクにより、計画シナリオの「開始年」と「終了年」がそれぞれ FY18 と FY22 に設定されています:

```
FIX("OEP_Plan", "FY18" : "FY22", "Jan" : "Dec")  
  FIX(OEP_Working, P_000, "111")  
    "5800" = 5500;  
  ENDFIX  
ENDFIX
```

例 3

次の構文では、キューブ・レベルで代替変数が使用されています。この例で、Plan1 はキューブ名、CurrentMonth は代替変数名です。

```
FIX ("OEP_Plan", [[getStartMonth(getSubVarValue("Plan1", "CurrentMonth"))]],
"FY15",
"BU Version_1", "No Currency", "No Entity", "No Grades")
"Current" = 15;
ENDFIX
```

これにより次のスクリプトが作成されます:

```
FIX ("OEP_Plan", "Jan", "FY15", "BU Version_1", "No Currency", "No Entity",
"No
Grades")
"Current" = 15;
ENDFIX
```

例 4

次の構文では、アプリケーション・レベルで代替変数が使用されています。この例で、CurrentMonth は代替変数名です。

```
FIX ("OEP_Plan", [[getStartMonth(getSubVarValue("CurrentMonth"))]], "FY15",
"BU
Version_1", "No Currency", "No Entity", "No Grades")
"Current" = 15;
ENDFIX
```

これにより次のスクリプトが作成されます:

```
FIX ("OEP_Plan", "Jan", "FY15", "BU Version_1", "No Currency", "No Entity",
"No
Grades")
"Current" = 15;
ENDFIX
```

例 5

次の構文:

```
FIX("OEP_Plan", "OEP_Working", FY20:FY24)
  "Bonus"
  (
    IF([[PlanningFunctions.isPlanStartYearSameAsCurrentFiscalYear
("Workspace")]] == 1)
      "Bonus" = "Salary" * 0.2;
    ELSEIF([[PlanningFunctions.isPlanStartYearSameAsCurrentFiscalYear
("Workforce")]] == 0)
      "Bonus" = "Salary" * 0.3;
```

```
)
ENDFIX
```

により、次のスクリプトが作成されます。ここでは、「次の会計年」に設定されている場合は isPlanStartYearSameAsCurrentFiscalYear が"0"の整数を返すため、最初の条件付きテスト(IF)は失敗して実行されず、2 番目の条件付きテスト(ELSEIF)が成功して実行されるように、要員モジュールの「プランニングと予測の準備」構成タスクにより「プラン開始年」が「次の会計年」に設定されています：

```
FIX("OEP_Plan","OEP_Working", FY20:FY24)
  "Bonus"
  (
    IF(0 == 1)
      "Bonus" = "Salary" * 0.2;
    ELSEIF(0 == 0)
      "Bonus" = "Salary" * 0.3;
  )
ENDFIX
```

相互参照

次も参照:

- [CrossRef\(accountName\)](#)
- [CrossRef\(accountName, prefix\)](#)
- [CrossRef\(accountName, prefix, true\)](#)

CrossRef(accountName)

CrossRef(accountName) は、デフォルトの接頭辞の No を(通貨、期間および年を除く)各ディメンション名に追加し、その後に指定された勘定科目を追加して相互参照を生成します。

Calculation Manager の構文:

```
[[CrossRef(accountName)]]
```

例:

アプリケーションに勘定科目、期間、HSP_View、年、シナリオ、バージョン、エンティティおよび製品のディメンションがあるとします。この例では、次の構文:

```
FIX (Aug, Actual, Working, FY15, P_000, "112")
  "120" = [[CrossRef("5800")]];
ENDFIX
```

によって次のスクリプトが返されます:

```
FIX (Aug, Actual, Working, FY15, P_000, "112")
  "120" = "BegBalance"->"No HSP_View"->"No Scenario"->"No Version"-
```

```
>"No Entity"->"No Product"->"5800";  
ENDFIX
```

CrossRef(accountName, prefix)

CrossRef(accountName, prefix)は、指定された接頭辞を(通貨、期間および年を除く)各ディメンション名に追加し、その後に指定された勘定科目を追加して相互参照を生成します。接頭辞は二重引用符で囲む必要があります。

Calculation Manager の構文:

```
[[CrossRef(accountName, "prefix")]]
```

例:

アプリケーションに勘定科目、期間、HSP_View、年、シナリオ、バージョン、エンティティおよび製品のディメンションがあるとします。この例では、次の構文:

```
FIX (Aug, Actual, Working, FY15, P_000, "112")  
  "110" = [[CrossRef("5800", "No")]];  
ENDFIX
```

によって次のスクリプトが返されます:

```
FIX (Aug, Actual, Working, FY15, P_000, "112")  
  "110" = "BegBalance"->"No HSP_View"->"No Scenario"->"No Version"->"No  
Entity"->"No Product"->"5800";  
ENDFIX
```

CrossRef(accountName, prefix, true)

CrossRef(accountName, prefix, true)は、指定された接頭辞を各ディメンション名に追加し、その後に指定された勘定科目を追加して相互参照を生成します。(これには年が含まれますが、通貨と期間は含まれません。)接頭辞は二重引用符で囲む必要があります。

Calculation Manager の構文:

```
[[CrossRef(accountName, "prefix", true)]]
```

例:

アプリケーションに勘定科目、期間、HSP_View、年、シナリオ、バージョン、エンティティおよび製品のディメンションがあるとします。この例では、次の構文:

```
FIX (Aug, Actual, Working, FY15, P_000, "112")  
  "111" = [[CrossRef("5800", "NoX", true)]];  
ENDFIX
```

によって次のスクリプトが返されます:

```
FIX (Aug, Actual, Working, FY15, P_000, "112")  
  "111" = "BegBalance"->"NoXHSP_View"->"NoXYear"->"NoXScenario"-
```

```
>"NoXVersion"->"NoXEntity"->"NoXProduct"->"5800";  
ENDFIX
```

Workforce キューブ年次累計

次も参照:

- [CYTD\(memberName\)](#)
- [CYTD\(memberName, calTpIndexName, fiscalTpIndexName\)](#)

CYTD(memberName)

ノート:

CYTD(memberName) は **Workforce** キューブにのみ使用します。

CYTD(memberName) はメンバー用に暦年の年次累計を計算する式を生成します

Calculation Manager の構文:

```
[[CYTD(memberName)]]
```

例:

```
Fix (NOV, Actual, Working, FY15, P_000, "112")  
  "5800" = [[CYTD("6100")]];  
ENDFIX
```

CYTD(memberName, calTpIndexName, fiscalTpIndexName)

ノート:

CYTD(memberName, calTpIndexName, fiscalTpIndexName) は **Workforce** キューブにのみ使用します。

CYTD(memberName, calTpIndexName, fiscalTpIndexName) は、メンバー用に暦年の年次累計を計算する式と、暦年および会計年度に基づいた期間のインデックスを生成します。メンバーの名前が変更されたときに使用します。デフォルトのメンバー名は「暦期間-インデックス」および「会計期間-インデックス」です。

Calculation Manager の構文:

```
[[CYTD(accountName, "Cal TP-Index", "Fiscal TPIndex")]]
```

例:

```
Fix (Dec, Actual, Working, FY15, P_000, "112")
    "5800" = [[CYTD("6100", "Cal TP-Index", "Fiscal TPIndex")]];
ENDFIX
```

文字列の ID の取得

Oracle Hyperion Planning では、勘定科目タイプがテキストの場合は、Oracle Hyperion Calculation Manager で式を記述してテキスト値を割り当てることができます。

Calculation Manager の構文:

```
[[PlanningFunctions.getIdForString("text")]]
```

例:

Planning には「acct1 text」という名前の勘定科目があり、タイプはテキストです。FY16 Dec から FY17 Mar に値をコピーして、テキストの勘定科目を「未予算」に変更できます。

```
FIX (Actual, Working, P_000, "210")
    DATACOPY FY16->Dec TO FY17->Mar;
    Mar("acct1 text"->FY17 = [[PlanningFunctions.getIdForString("Not
Budgeted")]]);
ENDFIX
```

9

検証およびデプロイ

次も参照:

- [システム・ビューからのビジネス・ルール、ビジネス・ルールセット、式およびスクリプト・コンポーネントの検証](#)
ビジネス・ルール、ルールセット、式およびスクリプト・コンポーネントを検証して、構文が正しいことを確認してから、アプリケーションにデプロイします。
- [ルール・デザイナーからのビジネス・ルールの検証](#)
ビジネス・ルールを作成またはデバッグするとき、ルール・デザイナーで検証できます。
- [ビジネス・ルールおよびビジネス・ルールセットのデプロイ](#)
ビジネス・ルールおよびビジネス・ルールセットをデプロイします。

システム・ビューからのビジネス・ルール、ビジネス・ルールセット、式およびスクリプト・コンポーネントの検証

ビジネス・ルール、ルールセット、式およびスクリプト・コンポーネントを検証して、構文が正しいことを確認してから、アプリケーションにデプロイします。

検証プロセスにより、次のことが確認されます:

- すべてのディメンション・メンバーがアプリケーション内のディメンションに対して有効です。
- 存在するすべての関数には、正しいパラメータ数があり、アプリケーション・タイプに対して有効です。
- ビジネス・ルールのすべての変数参照が有効です。置換変数の場合、変数は最初に正しい文字列で置換されて検証されます。実行変数の場合、検証プロセスによって、アプリケーション、アプリケーション・タイプ内のアプリケーション、計算タイプ(Oracle Hyperion Financial Management ユーザー)、プラン・タイプ(Oracle Hyperion Planning ユーザー)またはデータベース(Oracle Essbase および Oracle General Ledger ユーザー)、およびビジネス・ルール(Planning および Essbase ユーザーのみ)に対して、変数が定義されていることが確認されます。
- スクリプト世代には構文エラーはありません。

Planning ユーザーのみ: 実行時プロンプトを持つビジネス・ルールをデフォルト値で検証する場合、検証プロセスにより、実行時プロンプトのすべてのメンバーが選択したプラン・タイプおよびアプリケーションに対して有効であり、構文エラーがないことが確認されます。デフォルト値なしで実行時プロンプトを持つビジネス・ルールを検証する場合は、検証は実行されません。

 ノート:

デプロイメントの前にルールおよびルールセットを検証しない場合、デプロイメントは成功しますが、ルールおよびルールセットの起動は失敗することがあります。

ビジネス・ルール、ルールセット、式またはスクリプト・コンポーネントを検証するには:

1. 次のいずれかのタスクを行います:
 - **Financial Management ユーザー:** ルールセットを検証するには、計算タイプおよび「**ルールセット**」を展開します。
 - ルールセットを検証するには、「**ルールセット**」を展開します。
Planning および Essbase ユーザー

 ノート:

Financial Management アプリケーションの場合、アプリケーション内の計算タイプごとに「ルールセット」ノードがあります。Planning アプリケーションの場合、プラン・タイプおよびデータベースと同じレベルのアプリケーションごとに「ルールセット」ノードが1つだけあります。

- ルール、式、スクリプトまたはテンプレートを検証するには、検証するオブジェクトに応じて、計算タイプ、プラン・タイプまたはデータベースを展開し、「**ルール**」、「**式**」、「**スクリプト**」または「**テンプレート**」を展開します。
2. 次のいずれかのタスクを行います:
 - 検証するオブジェクトを右クリックして、「**検証**」を選択します。
 - 検証するオブジェクトを選択して、「**アクション**」、「**検証**」の順に選択します。
 3. **Planning および Essbase ユーザーのみ:** **検証オプション**で、Planning (Planning ルールの場合)、Essbase (Essbase ルールの場合)のどちら(あるいはその両方)に対して検証するかを選択します。「**OK**」をクリックします。
 4. 次のいずれかのタスクを行います:
 - a. オブジェクトが正常に検証されたら、「**OK**」をクリックします。
 - b. エラーがある場合は、それが表示されます。エラーを修正して、オブジェクトをもう一度検証します。

ルール・デザイナーからのビジネス・ルールの検証

ビジネス・ルールを作成またはデバッグするとき、ルール・デザイナーで検証できます。

- **Financial Management** アプリケーション管理で作成された Oracle Hyperion Financial Management ビジネス・ルールは、**Financial Management** のみに対して検証します。

- **Planning** アプリケーション管理で作成された **Oracle Hyperion Planning** ビジネス・ルールは、**Planning** のみに対して検証します。
- **Essbase** アプリケーションの **Oracle Essbase** ルールは、**Essbase** のみに対して検証します。
- **Oracle General Ledger** アプリケーションの **Oracle General Ledger** ルールは、**Essbase** のみに対して検証します。

ルール・デザイナー内からビジネス・ルールを検証するには:

1. システム・ビューで、アプリケーション・タイプ、アプリケーション、計算タイプ、プラン・タイプまたはデータベースを展開し、「**ルール**」を展開します。
2. 検証するルールを右クリックし、「**開く**」を選択します。
3. ルール・デザイナーで、「**アクション**」、「**検証**」の順に選択し、次のいずれかのタスクを行います:
 - **Financial Management** ルールを使用している場合は、「**連結**」を選択します。
 - **Planning** ルールを使用している場合は、「**Planning**」を選択します。

ノート:

Financial Management または **Planning** アプリケーション管理で作成されたアプリケーションのビジネス・ルールを使用している場合、そのルールが作成されたアプリケーションに対してのみ検証できます。

- **Essbase** アプリケーションおよびルール(**Oracle General Ledger** アプリケーションのために作成されたものを含む)を使用している場合、「**Essbase**」を選択します。
- **Planning** アプリケーション管理で作成されたアプリケーションを操作し、**Planning** ルールを操作している場合は、「**Planning**」を選択します。

ルールによって検証が正常に行われると、検証に成功したことを知らせるメッセージが表示されます。検証が行われない場合は、エラーが表示されます。

ビジネス・ルールおよびビジネス・ルールセットのデプロイ

ビジネス・ルールおよびビジネス・ルールセットをデプロイします。

次も参照:

- [ビジネス・ルールおよびビジネス・ルールセットのデプロイについて](#)
- [ビジネス・ルールおよびビジネス・ルールセットをデプロイ可能にする、およびデプロイ可能にしない](#)
- [デプロイメント・ビューからのビジネス・ルールおよびビジネス・ルールセットのデプロイ](#)
- [ルール・デザイナーまたはルールセット・デザイナーからのビジネス・ルールまたはビジネス・ルールセットのデプロイ](#)
- [ショートカットを含むビジネス・ルールのデプロイ](#)

- [Planning](#) に表示されるデプロイ済のビジネス・ルールの指定

ビジネス・ルールおよびビジネス・ルールセットのデプロイについて

ビジネス・ルールは、Oracle Hyperion Planning、Oracle General Ledger および Oracle Essbase にデプロイできます。ビジネス・ルールセットは、Oracle Hyperion Financial Management、Planning および Oracle General Ledger にデプロイできます。1 つ以上のビジネス・ルールまたはビジネス・ルールセットをデプロイできます(部分デプロイメントと呼ばれる)。アプリケーションのすべてのビジネス・ルールおよびビジネス・ルールセットをデプロイすることもできます(完全デプロイメントと呼ばれる)。

ノート:

複数のレベルで存在する変数(グローバル、アプリケーション、プラン・タイプまたはルールのうち複数のレベルに存在する変数)を含む Planning ビジネス・ルールがあり、最低レベルで変数を削除する場合、この削除が Planning でこの変数を使用するすべてのルールに適用されるように、Planning アプリケーションの完全再デプロイを実行する必要があります。部分再デプロイメントしか実行しないと、変数の削除が適用されず、Planning でそのまま使用される場合があります。

Financial Management にはビジネス・ルールセットのみをデプロイできます。アプリケーションの各計算タイプから 1 つのルールセットをデプロイできます。個々のルールはデプロイできません。

ビジネス・ルールセットを Financial Management にデプロイした後は、データ・フォームまたはデータ・グリッド内から計算または連結を実行できます。ビジネス・ルールとビジネス・ルールセットを Planning にデプロイした後は、データ・フォーム内から起動したり、「起動」メニューから個別に起動したりできます。ビジネス・ルールとビジネス・ルールセットを Essbase に配置した後は、Oracle Essbase Administration Services からまたは直接 Oracle Hyperion Calculation Manager から起動できます。ビジネス・ルールとビジネス・ルールセットを Oracle General Ledger にデプロイした後は、割当ての生成リンクを使用して Oracle General Ledger 内から起動できます。

ビジネス・ルールおよびビジネス・ルールセットの起動の詳細は、次のリソースを参照してください:

- [Oracle Hyperion Financial Management ユーザー・ガイド](#)
- 『[Oracle Hyperion Planning ユーザー・ガイド](#)』
- [Oracle Essbase Administration Services オンライン・ヘルプ](#)
- このガイドの[ビジネス・ルールの起動](#)。

ビジネス・ルールおよびビジネス・ルールセットをデプロイ可能にする、およびデプロイ可能にしない

アプリケーションでビジネス・ルールおよびルールセットのサブセットをデプロイするには、デプロイ可能にする必要があります。ルールおよびルールセットをデプロイ可能にするには、**デプロイメント・ビュー**のそれぞれの名前の横にあるチェック・ボックスを選択します。

ノート:

1つのビジネス・ルールまたはビジネス・ルールセットのみをデプロイするには、**デプロイメント・ビュー**でデプロイ可能にする必要はありません。かわりに、**システム・ビュー**でルールまたはルールセットを右クリックし、「**デプロイ**」を選択してデプロイできます。

ビジネス・ルールまたはルールセットをデプロイした後に、アプリケーションからこれらを除去するには、**デプロイメント・ビュー**でその名前の横にあるチェック・ボックスをクリアします。次に、「**デプロイ**」を、右クリックして選択すると、アプリケーションの完全デプロイメントは正しく実行されます。

ビジネス・ルールおよびビジネス・ルールセットをデプロイ可能にするには:

1. **デプロイメント・ビュー**で、デプロイするルールまたはルールセットを含むアプリケーション・タイプおよびアプリケーションを展開します。
2. 「**デプロイ予定**」を展開し、デプロイするルールおよびルールセットの横にあるチェック・ボックスを選択します。

ルールまたはルールセットをデプロイする前に、構文が正しいかどうかを検証する必要があります。検証機能を使用すると、ルールおよびルールセットを手動で検証できます。[\(システム・ビューからのビジネス・ルール、ビジネス・ルールセット、式およびスクリプト・コンポーネントの検証を参照してください\)](#)。

デプロイメント・ビューからのビジネス・ルールおよびビジネス・ルールセットのデプロイ

ビジネス・ルールおよびビジネス・ルールセットは、デプロイメント・ビューからデプロイできます。ルール・デザイナー(ビジネス・ルールの場合)またはルールセット・デザイナー(ビジネス・ルールセットの場合)から Oracle Hyperion Planning、Oracle Essbase または Oracle General Ledger にビジネス・ルールまたはビジネス・ルールセットを1つデプロイすることもできます。[ルール・デザイナーまたはルールセット・デザイナーからのビジネス・ルールまたはビジネス・ルールセットのデプロイを参照してください](#)。

Oracle Hyperion Financial Management にデプロイする場合は、アプリケーションのそれぞれの計算タイプに対して1つのビジネス・ルールセットをデプロイできます。**Financial Management** に複数のビジネス・ルールをデプロイできません。

デプロイメント・ビューからビジネス・ルールおよびビジネス・ルールセットをデプロイするには:

1. システム・ビューで「**ビュー**」、「**デプロイメント・ビュー**」の順に選択します。

 **ノート:**

システム・ビューで、ビジネス・ルールおよびビジネス・ルールセットを右クリックして「デプロイ」を選択して、デプロイすることもできます。

2. デプロイメント・ビューで、アプリケーション・タイプを展開します。
3. 次のいずれかのタスクを行います:
 - **Planning および Essbase ユーザー:** アプリケーション内のすべてのルールおよびルールセットをデプロイするには、デプロイするすべてのルールまたはルールセットを選択し、アプリケーションを右クリックして「デプロイ」を選択します。
 - **Planning および Essbase ユーザー:** ビジネス・ルールおよびビジネス・ルールセットのサブセットをデプロイする(部分デプロイメント)には、アプリケーションと「**デプロイ予定**」ノードを展開します。次のステップを実行します:
 - a. デプロイするルールセットを選択していない場合は選択します。
 - b. デプロイするルールを含むプラン・タイプを展開します。
 - c. デプロイするルールを選択していない場合は選択します。
 - d. 右クリックして「**デプロイ**」を選択します。

 **ヒント:**

複数のルールまたはルールセットをデプロイする場合は、[Ctrl]キーおよび[Shift]キーを押しながらクリックして選択し、右クリックして「デプロイ」を選択します。

 **ノート:**

他のアプリケーションへのショートカットを含むビジネス・ルールをデプロイすると、ショートカットがあるそれぞれのアプリケーションにビジネス・ルールがデプロイされます。

- **Financial Management ユーザーのみ:** ビジネス・ルールセットをデプロイするには:
 - a. デプロイするルールセットを含むアプリケーション、「**デプロイ予定**」ノード、および計算タイプを展開します。
 - b. デプロイするルールセットを選択していない場合は選択します。計算タイプ当たり 1 つのルールセットをデプロイできます。

 ノート:

デプロイメントには、必ず `Generic_Ruleset` を含めてください。

- c. アプリケーションを右クリックし、「**デプロイ**」を選択します。デプロイすると、`.xml` フォーマットのオブジェクトが `.rle` フォーマットに変換され、**Financial Management** アプリケーションにロードされます。

 ヒント:

複数のルールをデプロイする場合は、`[Ctrl]`を押しながらクリックするか、`[Shift]`を押しながらクリックして選択し、右クリックして「**デプロイ**」を選択します。

正常にデプロイメントされると、「デプロイメントに成功しました」というメッセージが表示されます。

ルール・デザイナーまたはルールセット・デザイナーからのビジネス・ルールまたはビジネス・ルールセットのデプロイ

ビジネス・ルールまたはルールセットを設計した後に、直接ルール・デザイナーまたはルールセット・デザイナーから検証およびデプロイできます。

ルール・デザイナーまたはルールセット・デザイナーからビジネス・ルールまたはビジネス・ルールセットをデプロイするには:

1. 次のいずれかのタスクを行います:
 - ビジネス・ルールをデプロイするには、ルールを含むプラン・タイプまたはデータベースを展開し、「**ルール**」を展開します。
 - ビジネス・ルールセットをデプロイするには、「**ルールセット**」を展開します。
2. デプロイするルールまたはルールセットを右クリックし、「**開く**」を選択します。
3. **ルール・デザイナー**または**ルールセット・デザイナー**から「**アクション**」、「**デプロイ**」の順に選択します。

正常にデプロイメントされると、「デプロイメントに成功しました」というメッセージが表示されます。

ショートカットを含むビジネス・ルールのデプロイ

ショートカットを含むビジネス・ルールをアプリケーションにデプロイすると、ショートカットを作成したそれぞれのアプリケーションにルールのコピーがデプロイされます。

ショートカットを含むビジネス・ルールをデプロイするには:

1. **システム・ビュー**で「**ビュー**」、「**デプロイメント・ビュー**」の順に選択します。
2. アプリケーション・タイプ、アプリケーション、「**デプロイ予定**」ノードと、プラン・タイプまたはデータベースを展開します。
3. デプロイするルールを右クリックし、「**すべてデプロイ**」を選択します。

Planning に表示されるデプロイ済のビジネス・ルールの指定

Oracle Hyperion Calculation Manager にビジネス・ルールがデプロイされると、Oracle Hyperion Planning の「ビジネス・ルール」ページで表示および実行できます。

Planning に表示するビジネス・ルールを指定できます。これを行うには、次のようにします。

1. 「**デプロイメント・ビュー**」で、Planning に表示しないルールの選択を解除します。
2. アプリケーション・ノードを右クリックし、「**デプロイ**」を選択します。

ルールが以前に Planning に表示されていた場合、Calculation Manager の「デプロイメント・ビュー」でその選択を解除すれば、アプリケーションのデプロイ後、そのルールは Planning に表示されなくなります。

ビジネス・ルールの起動

次も参照:

- [ビジネス・ルールの起動について](#)
Oracle Hyperion Planning ビジネス・ルールは、Oracle Hyperion Calculation Manager のシステム・ビューまたはルール・デザイナ内から起動できます。
- [システム・ビューおよびルール・デザイナからの Essbase ビジネス・ルールの起動](#)
- [コマンド・ライン・ランチャを使用した Essbase ビジネス・ルールの起動](#)
- [ルール・デザイナからの Planning ビジネス・ルールの起動およびログの表示](#)
Oracle Hyperion Planning ビジネス・ルールを起動して、生成されたログを表示できます。

ビジネス・ルールの起動について

Oracle Hyperion Planning ビジネス・ルールは、Oracle Hyperion Calculation Manager のシステム・ビューまたはルール・デザイナ内から起動できます。

Planning ビジネス・ルールを Planning にデプロイし、Planning から起動することもできます。Planning の Planning ビジネス・ルールの起動の詳細は、『Oracle Hyperion Planning ユーザー・ガイド』を参照してください。

ノート:

- Calculation Manager のルールの検証、デバッグ、デプロイ、分析または起動を行う場合、実行時プロンプト変数の値を入力または編集できます。ビジネス・ルールセットの検証やデプロイを行う場合にも、実行時プロンプト変数を入力または編集できます。実行時プロンプトにメンバー制限が含まれている場合、検証では有効なメンバー名のみを確認します(メンバーが制限内かどうかは検証されません)。数値および整数タイプの変数は実行時プロンプト制限に対して検証されます。
- Calculation Manager のルールの起動時に、有効な組合せやセキュリティは考慮されません。
- メンバーの承認セキュリティを保持する実行時プロンプトの設計については、[Oracle Hyperion Planning 管理者ガイドの実行時プロンプトと承認セキュリティについて](#)を参照してください。

システム・ビューおよびルール・デザイナーからの Essbase ビジネス・ルールの起動

システム・ビューからまたはルール・デザイナーから Oracle Essbase ビジネス・ルールを起動できます。

Essbase ビジネス・ルールを起動するには:

1. 「システム・ビュー」で、次のタスクのいずれかを実行します:
 - 実行するビジネス・ルールを右クリックして、「実行」を選択します。
 - 実行するルールをダブルクリックします。ルールがルール・デザイナーに表示されたら、「アクション」、「実行」の順に選択します。
2. 実行時プロンプトの入力方法の詳細は、[実行時プロンプト変数の入力](#)を参照してください。

コマンド・ライン・ランチャを使用した Essbase ビジネス・ルールの起動

コマンド・ライン・ランチャ・ユーティリティは、%EPM_ORACLE_HOME%\common\calcmgr\11.1.2.0\lib の calcmgrCmdLine.jar ファイル内にあります。ユーティリティを実行するコンピュータに Java ランタイム 1.6 以上のバイナリをインストールする必要があります。ユーティリティを実行するには、ユーティリティを実行するコンピュータに jar ファイルをコピーし、次のコマンドを使用します。

```
java -jar calcmgrCmdLine.jar <Calc Server URL> <USER> <PASSWORD>
<APPLICATION NAME> <DATABASE NAME> <RULE NAME> [,<VARIABLE
NAME>=<VARIABLE VALUE>]
```

- コマンド・ライン・ランチャは、一度に 1 つのルールのみを実行できます。
- ルール内で参照されていない変数名が渡された場合は無視され、その旨のメッセージは表示されません。
- 変数がルールで参照されているのに引数として渡されていない場合、生成されたルール・スクリプトは、検証値または変数のデフォルト値(指定されている場合)を使用します。
- 起動が成功すると、エラー・レベル 0 が返されます。起動が成功しなかった場合、エラー・レベル 1 が返されます。

コマンド・ライン・ランチャの例および引数

コマンド・ライン・ランチャの例を次に示します。

```
java -jar calcmgrCmdLine.jar http://myBox:19000 admin password Sample
Basic myRule var1=\"Jan\" var2=1000 var3=\"Budget->\"001\"->FY10"
```

- <Calc Server URL> - プロトコル(http、https など)およびホスト名とポート
- <USER> <PWD> - ユーザー名とパスワード
- <APPLICATION NAME> - ルールが作成される Oracle Essbase アプリケーション
- <DATABASE NAME> - ルールが作成される Essbase データベース(アプリケーション内)
- <RULE NAME> - ビジネス・ルールの名前
- <VARIABLE NAME> - ビジネス・ルールで置き換える実行時プロンプト変数名
- <VARIABLE VALUE> - 実行時プロンプト変数の値

暗号化されていないモードの例

```
java -jar calcmgrCmdLine.jar <CalcServer URL> <USER NAME> <PWD> <Application Name> <Database Name> <Rule Name> <variable name=value>,<variable name=value>
```

```
java -jar calcmgrCmdLine.jar http://myBox:19000 admin password Sample Basic myRule var1="Jan" var2=1000
```

暗号化モードの例

暗号化パスワードをコマンド・ライン・ランチャへの引数として渡す必要がある場合、まずパスワードの暗号化されたトークンを作成します。次のようにデフォルトのキーを使用してパスワードを暗号化できます。

```
java -jar calcmgrCmdLine.jar -encrypt password
```

復号化のデフォルト・キーを使用した暗号化モードの例

デフォルトのキーを使用してパスワードを復号化するには、次のようにして復号化します:

```
java -jar calcmgrCmdLine.jar -decrypt http://myBox:19000 admin g//F19AB2qrFN9R5m141CPgcLYmjmLyiIxx2CXz7hxbheZe6HDUvjYkQuW214Oer Sample Basic myRule var1="Jan" var2=1000
```

指定された暗号化のキーを使用した暗号化モードの例

指定したキーを使用してパスワードを暗号化するには、次のようにして暗号化します:

```
java -jar calcmgrCmdLine.jar -encrypt -key AAAAAAAAAAAAAAAAAAAAAA password
```

指定された復号化のキーを使用した暗号化モードの例

```
java -jar clacmgrcmdLine.jar -key AAAAAAAAAAAAAAAAAAAAAA http://myBox:19000 admin g//F19AB2qrFN9R5m141CPgcLYmjmLyiIxx2CXz7hxbheZe6HDUvjYkQuW214Oer Sample Basic myRule var1="Jan" var2=1000
```

Windows で bat ファイルを使用したコマンド・ライン・ランチャの実行

次の例は、Windows で bat ファイルを使用してコマンド・ライン・ランチャを実行する例です。

```
@echo off
java -jar calcmgrCmdLine.jar http://myBox:19000 admin password Sample
Basic myRule var1=\"Jan\" var2=1000 var3=\"Budget->\"001\"->FY10"
IF %ERRORLEVEL%==0 goto success
echo there is error
goto end
:success
echo success
:end
```

ルール・デザイナからの Planning ビジネス・ルールの起動およびログの表示

Oracle Hyperion Planning ビジネス・ルールを起動して、生成されたログを表示できます。

ルール・デザイナで表示または編集するためにビジネス・ルールを開くと、ルールを起動して生成されたログをルール・デザイナのログ・メッセージ・タブに表示できます。ログはカンマ区切り値(.csv)ファイルにエクスポートできます。

ルール・デザイナからビジネス・ルールを起動し、ログ・メッセージを表示するには:

1. **システム・ビュー**でルールをダブルクリックします。
2. ルール・デザイナで、「**アクション**」、「**起動**」の順に選択します。

ルールを実行すると、ルールがエラー有またはエラー無のどちらで起動したかを示す確認メッセージが表示されます。「**OK**」をクリックして確認メッセージを閉じます。

3. 「**ログ・メッセージ**」タブを選択します。

ログ・メッセージには次の情報が含まれています:

- **メッセージ番号** - ログ・ファイルに表示されるメッセージの ID
- **メッセージ・レベル** - メッセージの重要度/レベル
- **メッセージ・テキスト** - メッセージの完全なテキスト
- **メッセージ・タイムスタンプ** - メッセージがいつ生成されたかを示すタイムスタンプ
- **パス番号** 現在のパスの番号。最も大きい番号/最後の番号が、ルール内のパス数です。
- **パス時間** - 実行時間(秒)。現在のパスにかかった時間です。

- **累積時間** - 合計実行時間(秒)。これは、ルールの実行が始まって以降の合計経過時間です。
ノート: 多くのログ・メッセージは、データベースから同時に生成されるため、「パス時間」(秒)は、各パスにかかった時間のみを示し、累積時間(秒)は、ルールの開始からかかった時間を示します。他の行はすべて空白となります。
 - **パス%** - ルールのそのパスの合計時間のパーセンテージ。
 - **累積%** - ルールのすべてのパスの合計時間のパーセンテージ。すべてのパスが完了すると、累積パーセンテージは 100%になります。
4. **オプション。** 表示されたログ・メッセージをフィルタします。
- 次によりフィルタできます。
- **メッセージ番号** - 「**メッセージ番号**」列の上にあるドロップ・ダウンから値を選択します。
 - **メッセージ・レベル** - 「**メッセージ・レベル**」列の上にあるドロップ・ダウンから値を選択します。
 - **メッセージ・テキスト** - 「**メッセージ・テキスト**」列の上にあるテキスト・ボックスにテキストを入力します。
 - **パスのみ** - パス番号および各パスにかかった時間のみを表示するには、「**パス数**」の上にあるドロップ・ダウンで「**パスのみ**」を選択します。完全なログ情報の表示に戻るには、「**パス数**」列の上にある空白オプションを選択します。

 **ノート:**

ビジネス・ルールを Planning にデプロイすると、Planning 内からも起動できます。詳細は、『Oracle Hyperion Planning ユーザー・ガイド』を参照してください。

 **ノート:**

ログ・メッセージはカンマ区切り値(.csv)ファイルにエクスポートできます。ファイルへのログ・メッセージのエクスポートを参照してください。

ビジネス・ルール、ビジネス・ルールセット、 テンプレート、式およびスクリプト・コンポー ネントのエクスポートおよびインポート

次も参照:

- [エクスポートおよびインポートについて](#)
アプリケーションのオブジェクトをエクスポートおよびインポートします。
- [ビジネス・ルール、ビジネス・ルールセット、テンプレート、式およびスクリプト・コンポーネントのエクスポート](#)
アプリケーション、オブジェクト、または複数のオブジェクトをエクスポートする場合は、Oracle Hyperion Calculation Manager の他のアプリケーションにインポートできる xml ファイルにエクスポートされます。
- [アプリケーションのエクスポート](#)
アプリケーションをエクスポートする場合は、アプリケーションのコンテンツが xml ファイルに保存されます。
- [ファイルへのログ・メッセージのエクスポート](#)
Oracle Hyperion Planning ビジネス・ルールを Oracle Hyperion Calculation Manager 内から起動すると、ログ・メッセージが生成され、ルール・デザイナの「ログ・メッセージ」タブに表示されます。
- [ルール、ルールセット、テンプレート、式およびスクリプトのインポート](#)
Oracle Hyperion Calculation Manager でルール、ルールセット、テンプレート、式およびスクリプトをアプリケーションにインポートします。
- [グラフィカル・ビジネス・ルールとしての Essbase 計算スクリプトのインポート](#)

エクスポートおよびインポートについて

アプリケーションのオブジェクトをエクスポートおよびインポートします。

Oracle Hyperion Financial Management、Oracle Hyperion Planning、Oracle Essbase または Oracle General Ledger アプリケーションのすべてのオブジェクトをエクスポートできます。また、アプリケーション内の個別のビジネス・ルール、ビジネス・ルールセット、テンプレート、式コンポーネントおよびスクリプト・コンポーネントもエクスポートできます。

Oracle Hyperion Business Rules からビジネス・ルール、シーケンス、マクロおよび変数をエクスポートし、Oracle Hyperion Calculation Manager にインポートすることもできます。シーケンスはビジネス・ルールセットに変換され、マクロは Calculation Manager のテンプレートに変換されます。

アプリケーションとオブジェクトをエクスポートした後で、それらを別の Financial Management、Planning、Essbase または Operational Planning アプリケーションにインポートできます。たとえば、ビジネス・ルールおよびビジネス・ルールセットを、本番コンピュータのアプリケーションからエクスポートし、それらをテスト・コンピュータの別のアプリケーションにインポートできます。

また、Essbase 計算スクリプトを、グラフィカル・ビジネス・ルールとして Calculation Manager にインポートすることも可能です。

ビジネス・ルール、ビジネス・ルールセット、テンプレート、式およびスクリプト・コンポーネントのエクスポート

アプリケーション、オブジェクト、または複数のオブジェクトをエクスポートする場合は、Oracle Hyperion Calculation Manager の他のアプリケーションにインポートできる xml ファイルにエクスポートされます。

ノート:

オブジェクトは、任意のビュー(システム・ビュー、カスタム・ビューおよび配置ビュー)からエクスポートできます。1 つ以上のオブジェクトをエクスポートできます。

オブジェクトをエクスポートするには:

- 次のいずれかのタスクを行います:
 - Oracle Hyperion Financial Management ユーザーのみ:** Financial Management のルールセットをエクスポートするには、計算タイプおよびエクスポートするルールセットを含む「**ルールセット**」ノードを展開します。
 - ルールセットをエクスポートするには、「**ルールセット**」を展開します。
Oracle Hyperion Planning、Oracle Essbase および Oracle General Ledger アプリケーションのみ
 - ルール、式、スクリプトまたはテンプレートをエクスポートするには、計算タイプ、プラン・タイプまたはデータベースを展開し、次に、「**ルール**」、「**式**」、「**スクリプト**」または「**テンプレート**」を展開します。
- 次のいずれかのタスクを行います:
 - オブジェクトを 1 つだけエクスポートするには、右クリックして「**エクスポート**」を選択します。
 - 複数のオブジェクトをエクスポートするには、エクスポートするオブジェクトを選択し、右クリックして「**エクスポート**」を選択します。**[Shift]キーを押しながらクリック**または**[Ctrl]キーを押しながらクリック**し、異なる計算、プラン・タイプまたはデータベース、異なるオブジェクト・タイプ(ビジネス・ルール、式など)、およびアプリケーション・タイプ内の異なるアプリケーションにある隣接または隣接していない複数のオブジェクトを選択します。
「エクスポート」を選択した後に、生成された.xml ファイルを開くか、または保存するかを確認するメッセージが表示されます。
- 「**ファイルのダウンロード**」で、次のいずれかのタスクを行います:
 - 生成された.xml ファイルのコンテンツを表示するには、「**開く**」を選択します。

- 生成された.xml ファイルを表示せずに保存するには、「**保存**」を選択し、ファイル名を入力して(またはデフォルト名を受け入れて)、再び「**保存**」をクリックします。

アプリケーションのエクスポート

アプリケーションをエクスポートする場合は、アプリケーションのコンテンツが xml ファイルに保存されます。

アプリケーションをエクスポートするには:

1. アプリケーションを右クリックして、「**エクスポート**」を選択します。
2. 「**ファイルのダウンロード**」で、次のいずれかのタスクを行います:
 - 生成された xml ファイルのコンテンツを表示するには、「**開く**」を選択します。
 - コンテンツを表示せずに生成された xml ファイルを保存するには、「**保存**」を選択し、ファイル名を入力して(またはデフォルト名を受け入れて)、再び「**保存**」をクリックします。

ファイルへのログ・メッセージのエクスポート

Oracle Hyperion Planning ビジネス・ルールを Oracle Hyperion Calculation Manager 内から起動すると、ログ・メッセージが生成され、ルール・デザイナの「ログ・メッセージ」タブに表示されます。

これらのログ・メッセージはカンマ区切り値(.csv)ファイルにエクスポートできます。[ルール・デザイナからの Planning ビジネス・ルールの起動およびログの表示](#)を参照してください。

Calculation Manager 内から Planning ビジネス・ルールを起動して生成されたログ・メッセージをエクスポートするには:

1. **システム・ビュー**で起動するビジネス・ルールをダブルクリックします。
2. ルールがルール・デザイナに表示されたら、「**アクション**」、「**起動**」の順に選択します。
ルールを実行すると、ルールがエラー有またはエラー無のどちらで起動したかを示す確認メッセージが表示されます。
3. 「**OK**」をクリックして確認メッセージを閉じます。
ログ・メッセージがログ・メッセージ・タブに表示されます。
4. ルールの起動時に生成されたログ・メッセージをエクスポートするには、「**アクション**」、「**エクスポート**」の順に選択します。

RuleLogMessages.csv という名前のファイルに表のログ・メッセージがすべて格納され、エクスポート後にダウンロードできます。このファイルを保存し、カンマを区切り文字として指定して **Microsoft Excel** を使用して開きます。

ルール、ルールセット、テンプレート、式およびスクリプトのインポート

Oracle Hyperion Calculation Manager でルール、ルールセット、テンプレート、式およびスクリプトをアプリケーションにインポートします。

インポート後、インポートの結果をローカル・ファイルに保存できます。

オブジェクトをインポートするには、ファイルの種類が次のいずれかである必要があります。

- **すべてのユーザー:** .xml。オブジェクトを xml フォーマットで格納するファイル
- **すべてのユーザー:** .zip。zip ファイルには xml ファイルのみが含まれます
- **Oracle Hyperion Financial Management ユーザーのみ:** .rle。オブジェクトを Visual Basic フォーマットで格納するファイル
- **Financial Management ユーザーのみ:** .cmo。オブジェクトを ASCII テキスト・フォーマットで格納するファイル
- **Oracle Hyperion Planning、Oracle Essbase および Oracle General Ledger ユーザーのみ:** .csc。オブジェクトを Essbase 計算スクリプト・フォーマットで格納するファイル

ノート:

ある製品タイプから別の製品タイプにインポートすることはできません。

オブジェクトをインポートするには:

1. システム・ビューで「アクション」、「インポート」の順に選択するか、 をクリックします。
2. 「インポート」ダイアログ・ボックスの「ファイルのインポート詳細」で「参照」をクリックし、インポートするファイルを選択します。
3. 「場所詳細」で、アプリケーション・タイプ、アプリケーションおよびキューブを入力します。
 - .csc ファイルの場合は、場所の詳細を入力する必要があります。
 - ファイルが.xml ファイルで、場所の詳細がすでにインポート・ファイルに含まれている場合、場所の詳細を入力する必要はありません。
 - ファイルが.rle ファイルで、場所の詳細を入力した場合、キーワード 'app_name=<name of the application>' は.rle ファイルの最初の行にある必要があります。たとえば、'app_name=Statutory'。
 - .cmo ファイルの場合、アプリケーションの情報はキーワードに含まれます。

「場所詳細」に入力された情報は、インポート・ファイルに指定された場所より優先されます。場所情報がインポート・ファイルに指定されていない場合、「場所詳細」に情報を入力する必要があります。そうしないと、インポートは失敗します。

4. 「インポート・オプション」で、次のオプションのいずれかを選択します。
 - **既存のオブジェクトの上書き** - インポートするオブジェクトが、アプリケーションおよび計算、キューブ、またはデータベースのオブジェクトを置換します。
 - **既存のオブジェクトをスキップ** - インポートされるオブジェクトがまだ存在しない場合、オブジェクトはアプリケーションおよび計算、キューブまたはデータベースのオブジェクトに追加されます。存在する場合、オブジェクトはインポートされず、結果のステータスは「スキップされました」と示されます。
 - **重複のエラー出力** - インポートされるオブジェクトが、アプリケーションおよび計算、キューブまたはデータベースにすでに存在するオブジェクトと重複する場合、重複するオブジェクトの名前がログ・ファイルに書き込まれ、オブジェクトはインポートされずに、インポート・プロセスが停止します。
5. **Financial Management ユーザーのみ**: .rle ファイルをインポートしている場合、次の変換オプションが有効です:
 - **スクリプト・コンポーネント** - .rle ファイルを Calculation Manager のスクリプト・コンポーネントに変換します。
 - **グラフィカル・コンポーネント** - .rle ファイルを Calculation Manager のグラフィカル・コンポーネントに変換します。

「グラフィカル・コンポーネント」を選択すると、次を実行できます:

 - **コンポーネント・レベルの条件ブロックを使用します**を選択して、式コンポーネントの一部としてフロー・チャートに条件を含めることができます。(デフォルトでは、条件は、式コンポーネントの一部としてではなく、個別の条件コンポーネントとしてフロー・チャートに含まれます。)
 - オプションを選択して、スクリプトのコメントに基づいて文をコンポーネント・オブジェクトに分割する方法を指定します:
 - * **別のコンポーネント** - スクリプトの各コメント行に対して、個別のコンポーネントを生成します。
 - * **複数ブロックのコンポーネント** - スクリプトのコメント行ごとに1ブロックを含む1つのコンポーネントを生成します。
 - * **単一ブロックのコンポーネント** - スクリプトのすべてのコメント行に対して1ブロックを含む1つのコンポーネントを生成します。(コメント行ごとに個別のブロックはありません。)
6. 「インポート」をクリックします。
7. **オプション: 「名前を付けて保存**」をクリックして、インポートの結果をローカル・ファイルに保存します。

グラフィカル・ビジネス・ルールとしての Essbase 計算スクリプトのインポート

Oracle Essbase 計算スクリプトをグラフィカル・ビジネス・ルールとして Oracle Hyperion Calculation Manager にインポートできます。

SET RUNTIMESUBVARS 計算コマンドが含まれる Essbase 計算スクリプトをインポートすると、Essbase ランタイム代替変数(スクリプト内で先頭にアンパサンド文字が付いていますがすべて {} 表記法を使用して Calculation Manager 実行時プロンプト変数に変換されます。

デフォルトでは、Essbase 計算スクリプトの `RUNTIMESUBVARS` コマンドの変数の値を使用して **Calculation Manager** で *string* の変数が作成されます。Essbase でのこれらの計算スクリプトの詳細は、**Oracle Essbase データベース管理者ガイド**を参照してください。

Calculation Manager では、変数の値は、ルール・デザイナーでビジネス・ルールを開いたときに「変数」タブの「値」列に表示されます。(変数のデフォルト値は、変数デザイナーで開いて「デフォルト値」列を参照することで表示できます。)ただし、計算スクリプトの変数に値またはデフォルト値がない場合、**Calculation Manager** には値が表示されません。

計算スクリプトの<RTSV_HINT>タグと</RTSV_HINT>タグの間にテキストがある場合、そのテキストが変数の実行時プロンプト・テキストとして使用されます。これらのタグの間にテキストがない場合、プロンプト・テキストは変数の名前です。

グラフィカル・ビジネス・ルールとして Essbase 計算スクリプトをインポートするには:

1. 「システム・ビュー」で、データベースを右クリックし、「**Essbase 計算スクリプトのインポート**」を選択します。

このメニュー・オプションは、インポートする Essbase 計算スクリプトが存在する場合にのみ使用可能になります。

2. 各計算スクリプトに関して、すでに **Calculation Manager** に存在するかどうかと、インポートするかどうかを選択します。(すべての計算スクリプトをインポートするには、「**すべて選択**」をクリックします。)

3. 次のいずれかのオプションを選択します:

- 既存のオブジェクトを同じ名前の新しいオブジェクトに置き換えるには、「**既存のオブジェクトをオーバーライド**」を選択します。
- 新しいオブジェクトが既存のオブジェクトと同じ名前の場合に既存のオブジェクトを保持するには、「**既存のオブジェクトをスキップ**」を選択します。

4. 「**インポート**」をクリックします。

ノート:

インポートする計算スクリプトをスクリプト・コンポーネントとしてインポートすると、解析エラーが発生する場合があります。ルールを検証して解析エラーが発生した場合は、次のいずれかを行って修正します。

- 1) **Calculation Manager** で解析エラーを修正してルールを検証し、スクリプト・コンポーネントを右クリックして「グラフィカルに変換」を選択し、ルールを保存します。(グラフィカルへの変換を元に戻すには、「編集」、「変換を元に戻す」の順に選択します。)
- 2) 元の計算スクリプトを開き、解析エラーを修正して計算スクリプトを保存し、その計算スクリプトを **Calculation Manager** に再度インポートします。計算スクリプトの「すでに存在します」の下にチェックマークが表示されます。「インポート」チェック・ボックスと、既存のオブジェクトを上書きを選択します。

計算スクリプトのインポート後、インポートした計算スクリプトがスクリプト・コンポーネント形式であり、それをグラフィカル・フォーマットで表示する場合は、

ルール・フロー・チャートでそのスクリプト・コンポーネントを右クリックし、「グラフィカルに変換」を選択します。

インポートの結果をローカル・ファイルに保存することもできます。

Planning アプリケーションのための Essbase サーバー、アプリケーションおよびデータベースの管理

次も参照:

- **データベース・プロパティの操作**
一般、ディメンション、統計、トランザクション、変更の各プロパティを表示して編集します。
- **MaxL スクリプトの作成および起動**
- **データベース・オブジェクトのロックの削除**
自身の持つ権限に応じて、オブジェクトを表示してロックを解除できます。
- **アプリケーションの開始と停止**
ユーザーは、自身が読取り権限以上を持っているアプリケーションを開始できます。Oracle Essbase によって、新たに開始されたアプリケーションが Essbase サーバーのメモリーにロードされます。
- **データベースの開始と停止**
データベースを開始すると、Oracle Essbase では Essbase サーバーのメモリー上にデータベースがロードされます。
- **アプリケーション・ログ・ファイルの削除**
- **データベースの再構築**
(たとえば、メンバーを密ディメンションに追加することで)データベースを再構築する場合は、データ・ブロックの再計算が必要になることがあります。
- **アウトラインの確認**
エラーがないか、Oracle Essbase アウトラインを確認できます。
- **データベース設定を最適化するためのシミュレーションの作成**
- **データベースからのデータのクリア**
集約ストレージ・アプリケーションからデータをクリアし、ブロック・ストレージ・アプリケーションからデータのブロックをクリアします。
- **ブロック・ストレージ・アプリケーションのロケーション別名の操作**
ロケーション別名はデータ・ソースの記述子です。ロケーション別名を表示してエクスポートできます。
- **集約ストレージ・データベースでの問合せトラッキングの使用**
問合せデータを使用すると、データベース用に生成される最も適切な集約ビューのセットを選択します。
- **集約ストレージ・データベース・アウトラインの圧縮**
ファイルを圧縮して、削除済のメンバーのレコードを除去し、アウトライン・ファイルのサイズを削減します。

- **レベル 0 データのインポートとエクスポート**
ASO および BSO キューブから、レベル 0 データをインポートおよびエクスポートします。
- **増分データ・スライスのマージ**
メイン・データベース・スライスにすべての増分データ・スライスをマージしたり、メイン・データベース・スライスを変更しないまま単一のデータ・スライスにすべての増分データ・スライスをマージしたりできます。
- **データの集約**
データが含まれ、ユーザーに計算権限が付与されている集約ストレージ・データベースに対して集約を計算します。
- **集約プロセスの実行**
集約プロセスを実行すると、取得のパフォーマンスが向上します。
- **要求の管理**
セッション・ウィンドウの情報を使用してアクティブな要求を管理します。
- **Planning ドリル・スルー定義の追加**
Oracle Hyperion Calculation Manager で、Oracle Hyperion Planning プラン・タイプに対するこれらのセルのドリル・スルー定義をリスト、追加、編集および削除できます。

データベース・プロパティの操作

一般、ディメンション、統計、トランザクション、変更の各プロパティを表示して編集します。

次も参照:

- [データベース・プロパティの表示および編集](#)
- [一般データベース・プロパティ](#)
- [ディメンション・プロパティ](#)
- [統計プロパティ](#)
- [トランザクション・プロパティ](#)
- [変更プロパティ](#)
- [通貨プロパティ](#)
- [キャッシュのプロパティ](#)

データベース・プロパティの表示および編集

Oracle Hyperion Planning のブロック・ストレージ・アプリケーションと集約ストレージ・アプリケーションのデータベース・プロパティを表示および編集できます。

データベース・プロパティを表示または編集するには:

1. **システム・ビュー**で  をクリックします。
2. **エンタープライズ・ビュー**で Planning アプリケーションおよびブロック・ストレージまたは集約ストレージ・アプリケーションを展開し、データベースを選択します。

3. データベース・プロパティのタブで情報を表示または編集し、 をクリックします。

 **ノート:**

データベース・プロパティを変更する場合、アプリケーションを停止して再起動する必要があります。[アプリケーションの開始と停止](#)を参照してください

一般データベース・プロパティ

データベースの一般情報。次のような領域のプロパティがあります。

- **一般** - データベースの説明を入力し、データベース・タイプ、データベース・ステータス (ロードされているかどうか) およびデータベースの最小アクセス・レベルが表示されます。
- **計算** - *ブロック・ストレージ・アプリケーションのみ*

- **欠落した値の集約** - データベースの計算時欠落した値を集約します。

Oracle Essbase のデフォルトでは、データベースの完全計算時に欠落した値 (#Missing 値) は集約されません。データを親レベルにロードしない場合は、欠落した値を集約することにより計算のパフォーマンスが改善されることがあります。データベース・マネージャの権限を持つデータベースでは、欠落した値を集約するかどうかを選択できます。

親レベルでデータをロードしない場合、欠落した値を集約することにより計算のパフォーマンスが改善されることがあります。欠落した値を集約し、データを親レベルでロードする場合、データベースの集計結果が #Missing 値であっても、親レベルの値はそのデータベース集計結果で置き換えられます。

- **「等式にブロックを作成」** - 特定のメンバーの組合せのためにデータ・ブロックを作成します。

等式でブロックを作成する場合、データ・ブロックが存在しないメンバーの組合せに対して非定数値を割り当てたときに、**Essbase** でデータ・ブロックが作成されます。等式でブロックを作成すると、非常に大規模なデータベースが生成される可能性があります。

疎ディメンションのメンバーに定数を割り当てる場合は、**Essbase** でデータ・ブロックが作成されます。このため、疎メンバーに定数を割り当てる場合 (**West = 5** など)、「等式にブロックを作成」を選択しないでください。

疎メンバーに定数以外のものを割り当てるときにブロックが作成されるようにするには、「等式にブロックを作成」を選択する必要があります。たとえば、疎の **Scenario** ディメンションのメンバーである **Actuals** にデータが存在しない場合、次の割当てを行うためには、「等式にブロックを作成」を選択する必要があります: **2002Forecast = Actuals * 1.05;**

- **2 パス計算** - 特定のメンバーを再計算します。

この「2 パス計算」を選択すると、デフォルト計算の後に、2 パスのタグが付いているメンバーが再計算されます。2 パスのタグは、会計タグが付けられたディメンションのメンバー、および任意のディメンションの「動的計算」および「動的計算および保管」のメンバーに対してのみ作用します。

- **データの取得バッファ**
 - 「**バッファ・サイズ**」 - 取得バッファのサイズ。スプレッドシート・アドインおよびレポート・スクリプトからの取得の処理および最適化に使用されます。
 - **ソート・バッファ・サイズ** - 取得ソート・バッファのサイズ
- **ストレージ - ブロック・ストレージ・アプリケーションのみ**
 - **現在の入出力アクセス・モード** - 現在のアクセス・モード
 - **保留中の入出力アクセス・モード** - 次のオプションのいずれかがデフォルトで構成されます。
 - * **バッファ入出力** - ファイル・システムのバッファ・キャッシュを使用します。データベースの作成時に `essbase.cfg` ファイルの `DIRECTION` 設定で直接入出力を指定しなかった場合、バッファ入出力がデフォルトになります。
 - * **直接入出力** - ファイル・システムのバッファ・キャッシュを使用せず、非同期のオーバーラップ入出力を実行します。これによって、応答時間が速くなり、キャッシュ・サイズをさらに最適化できるようになります。直接入出力を選択すると、データベースが開始されるたびに `Essbase` では直接入出力の使用が試みられます。直接入出力が使用できない場合、`Essbase` ではバッファ入出力が使用されます。キャッシュ・メモリのロックや、オペレーティング・システムに用意されているノー・ウェイト(非同期)入出力を使用するには、直接入出力を選択します。
 - **データ圧縮** - 次のオプションのいずれかがデフォルトで構成されます。
 - * **ビットマップ・エンコーディング** - ビットマップを使用してデータ・セルを表します。ビットマップ、ブロック・ヘッダーおよび他の制御情報のみがディスクに格納されます。ビットマップ・エンコーディングが、データを最も効率的に圧縮する方法です。`Essbase` で保管されるのは欠落していない値のみで、反復値やゼロ値は圧縮されません。データベースでは、データ・ブロックがデータ・キャッシュに移動されると、ビットマップを使用して欠落した値が再作成され、ブロックが完全に展開されます。
 - * **RLE(ランレングス符号化)** - 連続する反復値(ゼロを含む)が圧縮され、各反復値と連続反復回数が記録されます。`RLE` が適しているのは、ブロックの平均密度が 3 パーセント以下である場合や、連続するゼロの値やゼロ以外の連続する反復値がデータベースに多数存在している場合などです。
 - * **ZLIB** - 圧縮されるデータに基づくデータ・ディクショナリが作成されます。通常、`ZLIB` 圧縮は、データの密度が非常に高い場合に高い圧縮率を示します。ただし、場合によっては、他の圧縮方式のほうが高い圧縮率を示すことがあります。`ZLIB` 圧縮の場合、節約される記憶域は、欠落セル数や等しい値を持つセルの連続回数とは、ほとんど関係ありません。
 - * **圧縮しない** - データ圧縮が行われません。

ディメンション・プロパティ

データベースのディメンション情報。次のようなものがあります。

- データベース内のディメンションの数
- (ブロック・ストレージ・データベースのみ)ディメンションのタイプ(密または疎)
- ディメンション内のメンバー

- 格納されているメンバー
- (集約ストレージ・データベースのみ)各ディメンションのレベル数

 ノート:

ディメンションのプロパティは読取り専用です。

統計プロパティ

 ノート:

統計プロパティは読取り専用です。

集約ストレージ・アプリケーションの統計

- **一般** - 一般統計情報。次のようなものがあります。
 - **データベース開始時間** - データベース・サーバーのタイムゾーンに従った開始時間
 - **「データベースの経過時間」** - 時間:分:秒の形式の経過時間
 - **「接続数」** - 接続しているユーザー数
- **集約ストレージ統計** - 集約データベースのストレージ統計。次のようなものがあります。
 - アプリケーション内のディメンションごとにレベルを格納するために使用されるレベルおよびビットの数(集約ストレージ・データベースでは、すべてのディメンション・レベルが格納されるわけではありません。)
 - **最大キー長(ビット)** - すべてのディメンションで使用されるすべてのビット数の合計。たとえば、すべてのディメンションのキーには 20 ビットが含まれ、最初の 4 ビットは年ディメンションによって使用されます。
 - **最大キー長(バイト)** - キーに使用されるセル当たりのバイトの数
 - **入力レベルのセルの数** - すべてがレベル 0 のセルであることを前提として、式なしで、ユーザーがデータを入力できるディメンション全体にわたるレベル 0 の交差のセルの数
 - **増分データ・スライス数** - 即時計算ではなく増分計算(必要な場合のみ)できるデータ交差の数
 - **増分入力セル数** - 即時計算ではなく増分計算(必要な場合のみ)できる入力セルの数
 - **集約ビューの数** - 集約セルが含まれるビューの数
 - **集約セルの数** - 下位レベルの値から積み上げられるため、要求または取得されたときに計算する必要のあるセルの数。集約セルの値は、要求ごとに計算することも、事前計算してディスクに格納することもできます。
 - **増分集約セル数** - 必要な場合にのみ更新できる集約セルの数

- **増分データの問合せのコスト(総コストに対する比率)** - 関連する集約ビューから値を取得するための平均時間
- **入力レベル・データ・サイズ(KB)** - すべてのレベル 0 のセル内のデータのサイズ(KB 単位)
- **集約データ・サイズ(KB)** - すべての集約セル内の集約データのサイズ(KB 単位)
- **実行時間** - 実行時間の統計。次のようなものがあります。
 - **キャッシュ・ヒット率** - ディスクからの取得に対してキャッシュ内で情報を見つけられた成功率
 - **現在のキャッシュ・サイズ** - 動的に生成されたキャッシュ・サイズ
 - **現在のキャッシュ・サイズ制限(KB)** - キャッシュ・サイズの制限(KB 単位)
 - **最終起動以降のページ読取り数** - アプリケーションが(自動的にまたはユーザーによって)起動されてから読み取られたインデックス・ページの数
 - **最終起動以降のページ書込み数** - アプリケーションが(自動的にまたはユーザーによって)起動されてから更新されたインデックス・ページの数
 - **ページ・サイズ(KB)** - ページのサイズ(KB 単位)
 - **データに割り当てられたディスク・スペース(KB)** - データ・ストレージに割り当てられたハード・ディスク・スペースの合計量(KB 単位)
 - **データで使用されているディスク・スペース(KB)** - データ・ストレージに使用されているディスク・スペースの合計量(KB 単位)
 - **割り当てられた一時ディスク・スペース(KB)** - データ・ストレージに割り当てられた一時ディスク・スペースの合計量
 - **使用されている一時ディスク・スペース(KB)** - データ・ストレージに使用されている一時ディスク・スペースの合計量

 **ノート:**

ディスク・スペースは Default 表領域で使用されるスペースで、一時ディスク・スペースは Temp 表領域で使用されるスペースです。どちらの場合も、一部のファイル内の一部のスペースが使用されない可能性があります。

ブロック・ストレージ・アプリケーションの統計

- **一般** - 一般統計情報。
 - **データベース開始時間** - データベース・サーバーのタイムゾーンに従った開始時間
 - **「データベースの経過時間」** - 時間:分:秒の形式の経過時間
 - **「接続数」** - 接続しているユーザー数
- **ブロック** - ブロック・ストレージ・データベースのデータ・ブロックに関する統計。

- 「**既存のブロック数**」 - 存在している(データが格納されている)ブロックの合計数
 - 「**ブロック・サイズ**」 - 展開(圧縮解除)されたデータ・ブロックのバイト単位のサイズ(セル数×8; 8KB から 100KB になるのが理想的です)。ブロック・サイズを変更するには、データベースの疎/密の構成を変更する必要があります。
 - **ブロックの潜在数** - ブロックの最大数(1 つの疎ディメンションのメンバー数に別の疎ディメンションのメンバー数を掛け合わせるによって得られます)。たとえば、Sample Basic データベースでは、19 個の Product メンバーと 25 個の Market メンバーがあります(共有メンバーやラベルのみのメンバーは数えません)。Product と Market はデータが保管される疎ディメンションであるため、潜在データ・ブロックの数は $19 \times 25 = 475$ 個になります。
 - **既存のレベル 0 ブロック** - 存在している(データが含まれる)レベル 0 ブロック(子メンバーを 1 つも持たない疎ディメンション・メンバーのブロック)の合計数。データは上位レベルにロードされる場合があるため、レベル 0 ブロックの数とデータ入力によって作成されたブロックの数が同じになるとはかぎりません。
 - 「**既存の上位レベル・ブロック**」 - レベル 0 ブロック以外の、存在している(データが含まれる)ブロックの合計数。上位レベル・ブロックには、上位レベルの疎メンバーのすべての組合せ、およびレベル 0 の疎メンバーを含む上位レベルの組合せが含まれます。
 - 「**ブロックの密度(%)**」 - 既存のデータ・ブロックのサンプルに基づく、各データ・ブロック内のデータ・ポイントの平均充填率。疎/密の構成によりブロック密度が最大になります。しかし、ブロック密度を最大にするとデータ・ブロックの数が急増することがあります。ブロック・サイズとブロックの急増を考慮したうえで、ブロック密度の最大化を試みてください。
 - **既存の最大ブロック数のパーセンテージ** - 存在するブロックの数と可能なブロックの数の比を示すパーセント。この割合は、データベースの疎密の度合いを示すメジャーになります。このパーセントが非常に小さい値(1 パーセント未満など)になることは珍しくありません。
 - **圧縮率** - ディスク上に保管されているブロックの圧縮効率のメジャー。通常、圧縮率はブロック密度を示します。
 - **平均クラスタ率** - データ(.pag)ファイルの断片化レベル。最大値は 1 で、断片化なしを示します。取得、計算、データ・ロードのパフォーマンスが低下しており、クラスタ率の値が 1 よりも著しく小さい場合は、データのエクスポートとリロードを実行してデータ・ファイルの再書込みを実行することを検討してください。ファイルの再書込みによりファイルの断片化が解消され、1 に近いクラスタ率が得られるようになります。
 - **平均断片化率** - データベース内の空き領域。たとえば、平均断片化率の値が 3.174765 であれば、データベースの 3%が空き領域で断片化されています。データを更新したり計算したりするにつれ、ブロックが元の場所に収まらなくなると、ファイルの末尾に追加されるか、大きさが十分にある別の空き領域に収められ、空き領域が発生します。数値が大きいほど空き領域が多くなり、特定のレコードの取得に時間がかかるようになります。平均断片化率に基づいて、再構築を実行する必要があるかどうかを判断できます。
- **実行時間**
 - **インデックス・キャッシュ・ヒット率** - ディスクからインデックス・ページを取得する必要なくインデックス・キャッシュでインデックス情報を見つけられた成功率。
 - **データ・キャッシュ・ヒット率** - データ・ファイル・キャッシュからブロックを取得する必要なくデータ・キャッシュでデータ・ブロックを見つけられた成功率。

ヒット率が増加するにつれて、ディスクから取得される情報が減るため、パフォーマンスが向上します。ヒット率が 1.0 である場合、データがディスクから取得されることがないため、パフォーマンスは最大になります。キャッシュに割り当てるメモリの増分量は小さくすることをお勧めします。割当て量が小さくても大きくても得られるベネフィットは同じである可能性があります。また、通常、割当て量が大きくてもヒット率に与える影響には限度があります。

- **データ・ファイル・キャッシュ・ヒット率** - ディスクからブロックを取得する必要なくデータ・キャッシュでデータ・ブロックを見つけられたデータベースの成功率。
- **インデックス・ページ読取り数** - データ・ブロックを検索するために読み取られたインデックス・ページの数
- **インデックス・ページ書込み数** - データ・ブロックを更新するために書き込まれたインデックス・ページの数

インデックス(またはインデックス・ファイル)は、ブロック・ストレージのデータ取得情報を格納し、ディスクに常駐し、インデックス・ページを含みます。インデックス・ページには、データ・ブロックへのポインタ(索引エントリと呼ばれます)が含まれます。

- **データ・ブロック読取り数** - 読み取られるデータ・ブロックの数
- **データ・ブロック書込み数** - 更新されるデータ・ブロックの数

チュートリアル・ビデオ:



BSO データベース・プロパティの管理

トランザクション・プロパティ

ノート:

トランザクション・プロパティは、ブロック・ストレージ・データベースにのみ適用されます。

「トランザクション」タブには、データベースへのアクセスに関する情報が表示されます。

- **「コミット・アクセス」** を使用すると、トランザクションが完了してコミットされるまで、トランザクションに含まれているすべてのデータ・ブロックに対する読取り/書込みロックが保持されるようになります。次の同時オプションも事前構成できます。
 - **待機(秒)** - ロックされたデータ・ブロックにアクセスできるようになるまでトランザクションが待機する秒数。デフォルトは 20 秒ですが、別の値、「**無期限**」または「**待機しない**」を事前に構成することもできます。

- **プリイメージ・アクセス** - 別の同時トランザクションが実行されている間ロックされているデータ・ブロックに対してユーザーは読取り専用アクセス権を保持します。
- **「アンコミット・アクセス」** を使用すると、トランザクションでブロック単位で読取り/書き込みロックが保持されるようになります(デフォルト設定)。次の領域で同期ポイントを事前に構成することもできます。
 - **ブロックのコミット** - Oracle Essbase がコミットを実行する前に更新されるデータ・ブロックの数
 - **行のコミット** - Essbase がコミットを実行する前にデータ・ロード時に処理されるデータ・ファイルの行数

変更プロパティ

「変更」タブには、データベースに対して実行された直近の操作(アウトラインの更新、データのロードまたは計算)に関する情報が表示されます。

- **操作** - 操作のタイプ(データのロードや計算など)
- **ユーザー** - 操作を実行したユーザーの名前
- **開始時間** - Essbase Server による、データのロックなど準備タスクを含む操作の開始時刻(操作の継続時間については、アプリケーション・ログ内の経過時間エントリを参照してください。)
- **終了時間** - Essbase Server による、操作が終了した時刻
- **ノート** - オプションのコメント

通貨プロパティ

ノート:

通貨プロパティは Oracle Essbase ブロック・ストレージ・データベースのみに適用されます。

通貨プロパティは次のとおりです:

- **通貨データベース** - プロパティを表示または編集する通貨データベース。(アプリケーションで使用可能なすべての通貨データベースがリストに含まれます。)
- **換算方法** 通貨の換算方法
- **デフォルトの通貨タイプのメンバー** - 通貨換算でデフォルトとして使用する通貨タイプ・メンバー。通貨データベースで CurType ディメンションの有効なメンバーを任意に指定できます。

通貨データベースは必ず次の 4 つのディメンションで構成されます:

- **国** - 各現地通貨(米ドル、カナダ・ドル、ドイツ・ユーロなど)のレートを格納します。
- **時間** - 一般に、メイン・データベース内の時間ディメンションと同じです。このため、通貨データベースでは、長期にわたる通貨の変動を追跡し、メイン・データベースの異なる時間スライスを正確に変換できます。

- **カテゴリ** - メイン・データベース内の勘定科目のタグが付けられたディメンションのメンバーに様々なレートを適用できます。会計ディメンション用に定義されたカテゴリから、通貨データベースの通貨カテゴリ・ディメンションメンバーが生成されます。たとえば、総利益(Gross Profit)と純利益(Net Profit)の換算には特定のカテゴリのレートを使用し、その他の勘定科目の換算には他のレートを使用する必要があります。
- **通貨パーティション** - 異なる通貨換算シナリオを利用できます。たとえば、企業では実績レートとプラン・レートを格納する場合があります。シナリオ間でデータ換算を行うには、使用するレートのタイプを選択します。

キャッシュのプロパティ

ノート:

キャッシュ・プロパティは、ブロック・ストレージ・データベースにのみ適用されます。

「キャッシュ」タブに次のようにキャッシュ・サイズとキャッシュ・ロックが表示されます:

- **キャッシュ・メモリーのロック** - データベース・マネージャ権限を持っているデータベースについて、キャッシュ・メモリーのロックを使用可能にできます
メモリーをロックすると、システムのメモリー・マネージャが Oracle Essbase のキャッシュ・メモリーをスワップしたり予約したりする必要がなくなるので、データベースのパフォーマンスが向上することがあります。メモリーのロックを変更した場合、変更内容はデータベースをいったん停止して再起動したときに有効になります。
- **インデックス・ページ設定(KB)** - インデックス・ページのサイズ(8KB で変更不可能)が表示されます
- **インデックス・ページの現在値(KB)** - データベースにデータが含まれる場合、「インデックス・ページ設定」の値が表示されます
- **キャッシュ・サイズ** - インデックス・キャッシュ、データ・ファイル・キャッシュおよびデータ・キャッシュに割り当てられるメモリーの量を設定に役立つ様々なバッファのサイズ
デフォルト値、最小値、推奨値は、Oracle Essbase データベース管理者ガイドを参照してください。
- **インデックス・キャッシュ設定(KB)** - インデックスに使用されるメモリーの量(KB 単位)
- **インデックス・キャッシュの現在値(KB)** - インデックスに使用されているメモリーの量
- **データ・ファイル・キャッシュ設定(KB)** - 圧縮されたデータ・ファイル(.pag)を保持するためのメモリーの量(KB 単位)。(データベースがバッファ入出力を使用している場合は適用できません。)
- **データ・ファイル・キャッシュの現在値(KB)** - データ・ファイルに使用されているメモリーの量

- **データ・キャッシュ設定(KB)** - 圧縮されないデータ・ブロックの保持に使用されるメモリーの量(KB 単位)
- **データ・キャッシュの現在値(KB)** - データ・キャッシュに使用されているメモリーの量

MaxL スクリプトの作成および起動

MaxL スクリプト・エディタを使用して、Oracle Hyperion Calculation Manager 内での Oracle Essbase 管理タスクを自動化します。このエディタを使用すると、データベース・アウトラインおよびメタデータを定義できます。

MaxL スクリプト・エディタは、MaxL スクリプトを入力できるテキスト編集ウィンドウとして機能します。

MaxL スクリプト・エディタでは、MaxL に関する次のタスクを実行できます:

- MaxL スクリプトを作成および起動して、Essbase の管理タスクを自動化します。
- MaxL 文を対話的に入力および起動して、1 つ以上の Essbase 操作を一度に実行します。
- スクリプトの起動結果を表示します。

MaxL スクリプトを作成するには:

1. **システム・ビュー**で、「**データベース・プロパティ**」アイコンをクリックします。
2. 「**エンタープライズ・ビュー**」で、Oracle Hyperion Planning アプリケーション・タイプと、MaxL スクリプトを作成するアプリケーションを展開します。
3. アプリケーションを右クリックし、「**MaxL スクリプト・エディタ**」を選択します。
MaxL スクリプト・エディタが表示されます。
4. MaxL スクリプト・エディタで、スクリプトのコンテンツを入力します。
5. 「**起動**」ボタンをクリックしてスクリプトを起動します。

スクリプトのコンテンツは、エディタで対話的に入力、起動およびクリアを行う個別の文である場合があります。文は、スクリプトを保存しなくても起動できます。

データベース・オブジェクトのロックの削除

自身の持つ権限に応じて、オブジェクトを表示してロックを解除できます。

管理者権限を持つユーザーは、どのオブジェクトでもロックを解除できます。管理者権限を持たないユーザーは、ロックしたオブジェクトのみロックを解除できます。

サーバーでは、データベース・オブジェクト(計算スクリプト、レポート・スクリプト、ビジネス・ルール・ファイルなど)にチェックアウト機能を設けて、同じオブジェクトを複数のユーザーが同時に編集しないようにしています。

デフォルトでは、オブジェクトがロックされるのは、変更のために開くときです。オブジェクトを閉じるとロック解除されます。

Oracle Hyperion Calculation Manager のオブジェクトは、アクションを実行する際にロックされることがあります。オブジェクトのロックを解除するには、エンタープライズ・ビューでオブジェクトのロックを解除するプラン・タイプを選択します。

データベースからロックを削除するには:

1. システム・ビューで、「データベース・プロパティ」アイコンをクリックします。
2. エンタープライズ・ビューで、Oracle Hyperion Planning アプリケーション・タイプと、ロックを削除するデータベースを含むアプリケーションを展開します。
3. データベースを右クリックし、「**ロックの削除**」を選択します。

アプリケーションの開始と停止

ユーザーは、自身が読取り権限以上を持っているアプリケーションを開始できます。Oracle Essbase によって、新たに開始されたアプリケーションが Essbase サーバーのメモリーにロードされます。

親アプリケーションの開始時にデータベースも開始するよう指定できます。この場合、ユーザーがアプリケーション内のデータベースへ接続するよりも前にアプリケーションを開始しておけば、アプリケーションおよび関連する全データベースがメモリー内にロードされた状態となるため、ユーザーの初期パフォーマンス(データベース接続時)が向上します。

アプリケーションを停止すると、Essbase によって、アプリケーションと、そのアプリケーション内のすべてのデータベースが Essbase サーバーのメモリーからアンロードされます。これにより、使用可能なメモリーが増加します。アプリケーション内のデータベースが破損しないようにするには、適切な方法でアプリケーションを停止する必要があります。

アプリケーションを開始または停止するには:

1. システム・ビューで、「データベース・プロパティ」アイコンをクリックします。
2. エンタープライズ・ビューで、Oracle Hyperion Planning アプリケーション・タイプと、起動または停止するアプリケーションを展開します
3. アプリケーションを右クリックして、次を選択します:
 - アプリケーションの起動
 - アプリケーションの停止

ノート:

データベース設定の変更を行う際は必ず、アプリケーションを停止し再起動する必要があります。

データベースの開始と停止

データベースを開始すると、Oracle Essbase では Essbase サーバーのメモリー上にデータベースがロードされます。

インデックス・キャッシュは自動的に割り当てられ、データ・ファイル・キャッシュおよびデータ・キャッシュはブロックの要求時に割り当てられます。ユーザーのアクセスよりも前にデータベースを開始すれば、データベースが先にメモリー内にロードされるため、ユーザーの初期パフォーマンス(接続時)が向上します。

開始されていないアプリケーションのデータベースを開始すると、そのアプリケーションと、そのアプリケーション内のすべてのデータベースがロードされます。1 つの

アプリケーションについて 1 つのデータベース、またはすべてのデータベースを開始できます。

データベースを停止すると、Essbase では Essbase サーバー上のメモリーからデータベースがアンロードされ、更新されたデータがディスクにコミットされます。これにより、サーバー・コンピュータ上で使用可能なメモリーの量が増加します。

1 つのアプリケーションについて 1 つのデータベース、またはすべてのデータベースを停止できます。

ノート:

ブロック・ストレージ・アプリケーション・データベースを起動および停止できません。

データベースを開始または停止するには:

1. システム・ビューで、「データベース・プロパティ」アイコンをクリックします。
2. エンタープライズ・ビューで、Oracle Hyperion Planning アプリケーション・タイプと、起動または停止するデータベースを含むアプリケーションを展開します
3. データベースを右クリックして、次を選択します:
 - データベースの開始
 - データベースの停止

アプリケーションのすべてのデータベースを開始または停止するには:

1. システム・ビューで、「データベース・プロパティ」アイコンをクリックします。
2. エンタープライズ・ビューで、Planning アプリケーション・タイプと、起動または停止するデータベースを含むアプリケーションを展開します
3. アプリケーションを右クリックして、次を選択します:
 - 全データベースの開始
 - 全データベースの停止

アプリケーション・ログ・ファイルの削除

データベース・サーバーは、サーバーおよびアプリケーションで発生したアクティビティを .log 拡張子を持つテキスト・ファイルに書き込み(記録)します。サーバー上の各 Oracle Hyperion Planning アプリケーションには、アプリケーションの名前が付けられた独自のアプリケーション・ログがあります。アプリケーション・ログ内の情報は、エラーが発生した場合所や原因を特定するのに役立ちます。Oracle Hyperion Calculation Manager でこれらのアプリケーション・レベル・ログ・ファイルを削除できます。

アプリケーション・ログの情報は、次の評価に役立ちます:

- 特定の操作を実行したユーザー
- 操作が実行された日時
- 操作が実行または試行されたときに発生したエラー
- 最適化に役立つディメンションおよびメンバーに関する情報

- アーティファクトがサーバーのインスタンス上に存在する場合、操作を実行するために使用されるアーティファクト(計算を実行するために使用される計算スクリプトなど)の名前

アプリケーション・レベルのログ・ファイルを削除するには:

1. **システム・ビュー**で、「**データベース・プロパティ**」アイコンをクリックします。
2. **エンタープライズ・ビュー**で、Planning アプリケーション・タイプと、削除するログ・ファイルを含むアプリケーションを展開します。
3. アプリケーションを右クリックし、「**ログの削除**」を選択します。
4. ログを削除するかどうかを確認します。

Calculation Manager は、ログ・ファイルが正常に削除されたかどうかを知らせるメッセージを表示します。

データベースの再構築

(たとえば、メンバーを密ディメンションに追加することで)データベースを再構築する場合は、データ・ブロックの再計算が必要になることがあります。

Oracle Essbase では、すべてのデータ・ブロックが使用済としてマークされます。再構築されたデータベースを計算すると、すべてのブロックが計算されます。

ノート:

ブロック・ストレージ・アプリケーション・データベースを再構築できます。

次の場合は、データベースを再構築する必要があります。

- 密ディメンション「動的計算/保管」メンバーを追加、削除または移動します。
- 密ディメンション「動的計算/保管」メンバーを「動的計算」メンバーに変更します。
- 密ディメンション「動的計算」メンバーを「動的計算/保管」メンバーに変更します。
- 密ディメンションの非動的メンバーのストレージ・プロパティを「動的計算」に変更します。
- 密ディメンションのストレージ・プロパティを「動的計算」メンバーから非動的値に変更します。
- 疎ディメンション「動的計算」または「動的計算/保管」の非動的メンバーのストレージ・プロパティを変更します。

データベースを再構築するには:

1. **システム・ビュー**で、「**データベース・プロパティ**」アイコンをクリックします。
2. **エンタープライズ・ビュー**で、Oracle Hyperion Planning アプリケーション・タイプ、アプリケーション、およびデータベースを再構築するプラン・タイプの順に展開します。
3. プラン・タイプを右クリックし、「**データベースの再構築**」を選択します。

4. データベースを再構築するかどうかを確認します。

Oracle Hyperion Calculation Manager は、データベースが正常に再構築されたかどうかを知らせるメッセージを表示します。

アウトラインの確認

エラーがないか、Oracle Essbase アウトラインを確認できます。

アウトラインの確認プロセスでは、アウトライン・タイプ(集約ストレージまたはブロック・ストレージ)が考慮され、各タイプのルールに従ってアウトラインが確認されます。アウトラインのエラーがなくなると、メンバー式が確認されます。

アウトラインを確認するには:

1. システム・ビューで、「データベース・プロパティ」アイコンをクリックします。
2. エンタープライズ・ビューで、Planning アプリケーション・タイプ、アプリケーション、およびアウトラインを確認するプラン・タイプの順に展開します。
3. プラン・タイプを右クリックして、「アウトラインの検証」を選択します。

アウトラインの確認時に、Essbase によって次のアイテムが確認されます。

- すべてのメンバーおよび別名が有効です。メンバーおよび別名は、他のメンバー、別名、世代またはレベルと同じ名前にはできません。
- 1つのディメンションのみを、勘定科目、時間、通貨タイプまたは国としてタグ付けできます。
- 共有メンバーが有効です。
- レベル 0 メンバーがラベルのみとしてタグ付けされていません。
- ラベルのみのメンバーに式が割り当てられていません。
- 通貨カテゴリおよび通貨名が、通貨アウトラインに対して有効です。
- 疎ディメンションの「動的計算」メンバーに、100 を超える子がありません。
- 親メンバーに 1 つの子があり、子が「動的計算」メンバーの場合、親メンバーも「動的計算」である必要があります。
- 親メンバーに 1 つの子があり、子が「動的計算」、2 パス・メンバーの場合、親メンバーも「動的計算」、2 パスである必要があります。
- ブール属性ディメンションの 2 つのメンバー名が、アウトラインに定義された 2 つのブール属性ディメンション・メンバー名と同じです。
- 日付属性ディメンションのレベル 0 メンバー名は、日付形式名設定(mm-dd-yyyy または dd-mm-yyyy)と一致する必要があります。ディメンション名がレベル 0 メンバーであるためディメンションにメンバーがない場合、ディメンション名は設定と一致する必要があります。
- 数値属性ディメンションのレベル 0 メンバー名が数値です。ディメンション名がレベル 0 メンバーであるためディメンションにメンバーがない場合、ディメンション名は数値である必要があります。
- 属性ディメンションがアウトラインの末尾にあり、すべての標準ディメンションの後にあります。
- 標準ディメンションのレベル 0 の「動的計算」メンバーに式があります。

- メンバーの式が有効です。
- ハイブリッド分析アウトラインで、ディメンションのレベル 0 メンバーのみをハイブリッド分析対応にできます。

アウトラインの確認中、Essbase では、適切な数値属性ディメンション・メンバー名に対して次の変換も実行され、それらがアウトラインに表示されます。

- メンバー名のマイナス記号を名前の前から末尾に移動します。たとえば、-1 が 1- になります。
- メンバー名の先行または末尾のゼロは削除されます。たとえば、1.0 は 1、00.1 は 0.1 になります。

データベース設定を最適化するためのシミュレーションの作成

データベースのパフォーマンスは多くのパラメータによって異なります。様々なデータベース・パラメータの最適値に到達するには、データベースに異なる組合せのパラメータ値を適用してパフォーマンス統計の変化を確認することでシミュレーションを作成できます。Oracle Hyperion Calculation Manager には、シミュレーションごとに取得するパフォーマンス統計を分析し、それらのパフォーマンス統計のレポートを作成するオプションが用意されています。

ノート:

ブロック・ストレージ・アプリケーション・データベースのシミュレーションを作成できます。

シミュレーションおよびシミュレーション・レポートを作成するには:

1. システム・ビューで、「データベース・プロパティ」アイコンをクリックします。
2. 「エンタープライズ・ビュー」で、Oracle Hyperion Planning アプリケーション・タイプと、シミュレーションを作成するデータベースを含むアプリケーションを展開します。
3. プラン・タイプを右クリックし、「DB 設定の調整」を選択します。
4. 「タスクの選択」で、「アウトライン・プロパティのシミュレーションの新規作成」を選択し、「次」をクリックします。
5. 「シナリオの初期化」で、次のタスクを実行します。
 - a. 実行する繰返しの数を入力します。(最小値は 1 で、最大値は 6 です。)
 - b. 各繰返しを開始する前に実行するルールを入力または選択します。
 - c. 各繰返し中に実行するルールを入力または選択します。
 - d. 各繰返し後に実行するルールを入力または選択します。
 - e. シミュレーション中にログ・ファイルに書き出すエラーのタイプを指定するには、次のエラー・メッセージ・レベルのいずれかを選択します。
 - シミュレーション中にサーバーが実行するルーチン・アクションを取得する場合は「情報」

- サーバーによる重大でないと思われる条件を取得する場合は「警告」
 - シミュレーションの実行中に発生したエラーを取得する場合は「エラー」
 - シミュレーション中に計算統計に関する詳細情報を取得する場合は「詳細」
 - シミュレーション中に計算統計の要約を取得する場合は「要約」
 - シミュレーション中に情報またはエラーを取得しない場合は「なし」
- f. 各シミュレーション後に、新規データベースを作成するかどうかを選択します。
- g. 「次」をクリックします。
6. 「**繰返しパラメータの設定**」で、次のパラメータの初期値および増分値を設定します。
- データ・キャッシュ設定: 非圧縮データ・ブロックを保持するメモリー内のバッファです。
 - インデックス・キャッシュ設定: インデックス・ページを保持するメモリー内のバッファです。キャッシュに割り当てられているメモリー量に同時に依存するメモリー内のインデックス・ページ数。
 - バッファ・サイズ: データベース・バッファは、抽出された行データ・セルを評価の前に保持するデータベースごとのサーバー・バッファです。各データベースには、変更できる KB 単位の取得バッファ設定があります。デフォルトのバッファ・サイズは、32-bit プラットフォームでは 10KB、64-bit プラットフォームでは 20KB です。バッファのサイズを増やす場合、サイズ制限は 100,000KB に設定されていますが、100KB を超えないことをお勧めします。
 - ソート・バッファ・サイズ: 取得中にソートするデータを保持するサーバー・バッファのサイズ(KB 単位)を指定します。データベースごとにバッファ・サイズを調整できます。デフォルトのバッファ・サイズは、32-bit プラットフォームでは 10KB に設定され、64-bit プラットフォームでは 20KB に設定されます。
7. 「完了」をクリックします。

検証エラーがない場合、「**レポートの表示**」タブに次の情報が表示されます。

- 「**使用可能な統計**」領域で、次の情報を表示できます。
 - ランタイム統計: このタブには、平均クラスタ率や繰返し当たりのインデックス・キャッシュ・ヒット率などのデータベース固有のパラメータが含まれます。
 - 実行時間統計: このタブには、各繰返しで実行した各ルールの実行時間(秒単位)が表示されます。
- 「**統計チャート**」領域で、各繰返しで各ルールの実行に要した時間(秒単位)の折れ線グラフ、棒グラフまたは縦棒グラフを表示できます。
- 「**要約**」領域の「**繰返し詳細**」タブで、データおよびインデックス・キャッシュ設定やバッファ・サイズなどの各繰返しに対して設定したパラメータの詳細を確認できます。

「**ディメンション**」タブで、ディメンションの名前とタイプ(密または疎)、各ディメンションのメンバー数、各ディメンションに格納されるメンバー数など、各繰返しで使用されるディメンションとメンバーに関する情報を表示できます。
- 「**ログ**」領域の「**チャート**」タブで、各繰返しの各ルールに対して生成されるログ・メッセージのタイプの折れ線グラフ、棒グラフまたは縦棒グラフを表示できます。

「**ログ**」タブで、各繰返しの各ルールのログ・メッセージのテキストを表示できます(存在する場合)。ルール、繰返しおよびメッセージ・レベルでログ・テキストをフィルタできます。

8. 「**レポートの表示**」タブの表示中にいずれかのデータベース・パラメータを変更するには、「**概要**」タブを選択できます
データベース・パラメータを変更し、シミュレーションを再実行します。これを行った後、新しいレポートが表示されます。
9. シミュレーション・レポートをエクスポートするには、「ファイル」、「エクスポート」の順に選択します。XML ファイルの名前を入力します。
10. シミュレーション・レポートをインポートするには、「**タスクの選択**」で、「**インポート/ロード・シミュレーション・レポート**」を選択します。

データベースからのデータのクリア

集約ストレージ・アプリケーションからデータをクリアし、ブロック・ストレージ・アプリケーションからデータのブロックをクリアします。

次も参照:

- [集約ストレージ・アプリケーションからのデータのクリア](#)
- [ブロック・ストレージ・アプリケーションからのデータのブロックのクリア](#)

集約ストレージ・アプリケーションからのデータのクリア

集約ストレージ・アプリケーションからデータをクリアするには:

1. **システム・ビュー**で  をクリックします。
2. 「**データベース・プロパティ**」タブの **エンタープライズ・ビュー**で、集約ストレージ・アプリケーションを展開します。
3. ASO キューブを右クリックし、「**クリア**」を選択してオプションを選択します:
 - **すべてのデータ** - データベースからすべてのデータをクリアします
 - **すべての集約** - データベースからすべての集約データ値をクリアします
 - **部分データ** — 集約ストレージ・データベースの指定された領域からデータをクリアし、その他の領域にあるデータは保持します

部分データをクリアするよう選択した場合、MDX 式を入力してクリアする領域を定義し、「**論理**」か「**物理**」を選択してデータのクリアのタイプを指定します

- **論理** - 指定した領域内のデータが負の相殺値とともに新しいデータ・スライスに書き込まれ、その結果、クリアするセルの値がゼロになります。
- **物理** - 指定した領域内のデータが集約ストレージ・データベースから物理的に削除されます。

メンバー・セレクトアを使用して MDX 構文を作成するには、 をクリックします。「**メンバー・セレクトア**」ダイアログ・ボックスで、1 つまたは複数のメンバーを選択するか、各ディメンションの関数を使用して、「**OK**」をクリックして MDX 式を作成します。

例:

```
Crossjoin(Crossjoin(Crossjoin(Crossjoin(Crossjoin(Crossjoin(Crossjoin({[N
I].Levels(0).Members},{except(DESCENDANTS([Q1]},{[Q1]})})),{[FY12],[FY13],
[FY14]})),{[Plan]})),{[Working]}),{[100].Levels(0).Members}),
{[P_TP].Levels(0).Members}},{[FY06]}
```

「OK」をクリックしてデータをクリアすると、MDX 式が検証されることに注意してください。MDX 式を直接変更する場合、実行前に検証されます。

ノート:

権限を持っているデータベースからのみデータをクリアできます。

ブロック・ストレージ・アプリケーションからのデータのブロックのクリア

ブロック・ストレージ・アプリケーションからデータのブロックをクリアするには:

1. システム・ビューで  をクリックします。
2. 「データベース・プロパティ」タブのエンタープライズ・ビューで、ブロック・ストレージ・アプリケーションを展開します。
3. BSO キューブを右クリックし、「クリア」を選択してオプションを選択します:
 - **すべてのデータ** - データベースからすべてのデータをクリアします。
 - **上位レベルのブロック** - 上位レベルのデータ・ブロックのみクリアします。上位レベル・ブロックのデータ値は#Missing に設定されます。上位レベル・ブロックは、少なくとも 1 つの疎メンバーが親メンバーとなっている疎メンバーの組合せに対して作成されます。
 - **非入力ブロック** - 計算から導き出した値を含むデータ・ブロック(非入力ブロック)のみをクリアします。非入力ブロックをクリアすると、非入力(計算された)セルのデータ値は#Missing に設定されます。
 - **動的ブロック** - 動的に計算および格納されるデータ・ブロック(「動的計算/保管」メンバー)のみクリアします。
 - **空のブロック** - 空のデータ・ブロック(すべての値が#Missing であるブロック)のみをクリアします。

ノート:

権限を持っているデータベースからのみデータをクリアできます。

ブロック・ストレージ・アプリケーションのロケーション別名の操作

ロケーション別名はデータ・ソースの記述子です。ロケーション別名を表示してエクスポートできます。

次も参照:

- [ブロック・ストレージ・アプリケーションのロケーション別名の操作](#)
- [ロケーション別名のリストの表示](#)
- [ロケーション別名のエクスポート](#)
- [ロケーション別名の作成](#)
- [ロケーション別名の削除](#)

ブロック・ストレージ・アプリケーションのロケーション別名の操作

ロケーション別名はデータ・ソースの記述子です。ロケーション別名は、データベースの別名を、そのデータベースの物理的な場所にマップします。ロケーション別名は、データベースまたはアプリケーション・レベルで設定され、別名、サーバー、アプリケーション、データベース、ユーザー名およびパスワードを指定します。



ノート:

ロケーション別名は、集約ストレージ・データベースには適用されません。

ロケーション別名のリストの表示

ブロック・ストレージ・アプリケーションのロケーション別名のリストを表示するには:

1. システム・ビューで  をクリックします
2. エンタープライズ・ビューでブロック・ストレージ・アプリケーションを右クリックし、「ロケーション別名」を選択します。

ロケーション別名のエクスポート

ロケーション別名をエクスポートするには:

1. システム・ビューで  をクリックします
2. エンタープライズ・ビューでブロック・ストレージ・アプリケーションを右クリックし、「ロケーション別名」を選択します。

3. 「ロケーション別名」ダイアログ・ボックスで  をクリックし、エクスポートするファイルを選択して「OK」をクリックします。

ロケーション別名の作成

ロケーション別名を作成するには:

1. システム・ビューで  をクリックします
2. エンタープライズ・ビューでブロック・ストレージ・アプリケーションを右クリックし、「ロケーション別名」を選択します。
3. 「新しいロケーション別名」をクリックします。
4. 「ロケーション別名」ダイアログ・ボックスに必須情報を入力します。

ロケーション別名の削除

ロケーション別名を削除するには:

1. システム・ビューで  をクリックします
2. エンタープライズ・ビューでブロック・ストレージ・アプリケーションを右クリックし、「ロケーション別名」を選択します。
3. 「ロケーション別名」ダイアログで、削除する場所の別名を右クリックし、「削除」を選択します。

集約ストレージ・データベースでの問合せトラッキングの使用

問合せデータを使用すると、データベース用に生成される最も適切な集約ビューのセットを選択します。

問合せトラッキングを有効化して、データベースに対して実行される各問合せのコストに関するデータを取得できます。問合せのコストは、ビューから値を取得するために要する平均取得時間の見積りです。最初のビュー(デフォルトで選択済)の場合、見積りはすべての可能な問合せの平均です。問合せトラッキングが使用されるビューの場合、見積りは、追跡される問合せの平均です。したがって、同じビューでも条件が異なると見積りも変わります。特定のビューの使用のベネフィットを評価するパーセンテージを計算するには、レベル 0 の値のみを含むビューの使用の問合せコスト値をビューの問合せコスト値で除算します。

有効化された問合せトラッキングは、次のいずれかの状況が発生するまで続行されます:

- このトピックで説明されている手順に従って、データベースの問合せトラッキングが無効化されました。
- アプリケーションが停止しています。アプリケーションが停止していた場合、アプリケーションの再起動時に問合せトラッキングは自動的に再開されません。
- 追加の集約ビューがデータベースに対して生成されます。追加のビューが生成されると問合せトラッキング・データが無効になるため、新規集約ビューの生成により問合せトラッキングがリセットされます。

問合せトラッキング(メモリー内にのみ保管)には、Oracle Hyperion Web Analysis、グリッド API、レポート・スクリプト、Java API などからの問合せが含まれます。

ノート:

問合せトラッキングは、集約ストレージ・データベースに対してのみ使用できます。

問合せトラッキングを有効化または無効化するには:

1. システム・ビューで  をクリックします
2. エンタープライズ・ビューでプラン・タイプを右クリックし、「問合せトラッキング」を選択して次のいずれかのオプションを選択します。
 - **有効化:** 問合せトラッキングを有効化します
 - **無効化:** 問合せトラッキングを無効化します

問合せトラッキングを有効化するとデータベースでは問合せ情報が記録され、問合せトラッキングを無効化すると問合せ情報の記録が停止されメモリーから問合せデータがクリアされます。
3. 問合せトラッキングを有効化または無効化することを確認するには、「OK」をクリックします。

集約ストレージ・データベース・アウトラインの圧縮

ファイルを圧縮して、削除済のメンバーのレコードを除去し、アウトライン・ファイルのサイズを削減します。

たとえば、メンバーを追加または削除するときに、集約ストレージ・アウトライン・ファイル(.ot1 ファイル)を変更すると、ファイルのサイズが増加することがあります。アウトライン・ファイルが圧縮された後、以前のようにファイルは増大し続けます。

アウトライン・ファイルを圧縮すると、データベースでアウトラインが再構築されます。アウトラインを圧縮しても、データベースでデータはクリアされません。

メンバーをアウトラインから削除すると、アウトライン・ファイル内のそのメンバーに対応するレコードが削除済としてマークされますが、レコードはアウトライン・ファイルに残ります。アウトライン・ファイルを圧縮しても、削除されたメンバーのレコードは除去されません。

ノート:

集約ストレージ・データベース・アウトラインのみを圧縮できます。アウトラインの圧縮処理は、他のユーザーまたは処理がデータベースをアクティブに使用していない場合にのみ実行できます。

集約ストレージ・データベース・アウトラインを圧縮するには:

1. システム・ビューで  をクリックします
2. エンタープライズ・ビューで、アウトラインを圧縮するデータベースを含むプラン・タイプを右クリックし、「**アウトラインの圧縮**」を選択します。
3. データベース・アウトラインを圧縮することを確認するには、「**OK**」をクリックします。
「**アウトラインの圧縮アクションのステータス**」ダイアログには、圧縮の進行状況が表示されます。アウトラインを圧縮すると、「**アウトラインの圧縮アクションのステータス**」ダイアログに成功メッセージが表示されます。
4. 「**アウトラインの圧縮アクションのステータス**」ダイアログで、アウトラインの圧縮の詳細を表示するには「**詳細の表示**」を、ダイアログを閉じるには「**OK**」をクリックします。

レベル 0 データのインポートとエクスポート

ASO および BSO キューブから、レベル 0 データをインポートおよびエクスポートします。

次も参照:

- [ASO キューブからのレベル 0 データのインポート](#)
- [ASO キューブからのレベル 0 データのエクスポート](#)
- [BSO キューブからのレベル 0 データのインポート](#)
- [BSO キューブからのレベル 0 データのエクスポート](#)

ASO キューブからのレベル 0 データのインポート

ASO キューブのレベル 0 データをインポートするには:

1. システム・ビューで  をクリックします。
2. 「**データベース・プロパティ**」タブのエンタープライズ・ビューで、集約ストレージ・アプリケーションを展開します。
3. ASO キューブを右クリックして、「**レベル 0 データのインポート**」を選択します。
4. 「**キューブのレベル 0 データのインポート**」ダイアログ・ボックスで、次の情報を入力して「**OK**」をクリックします:
 - **Zip ファイル名** - インポートするデータの zip ファイル名。
 - **重複集約メソッド** - 同じセルの複数の値を組み合わせる方法を定義します。
 - **重複値の加算** - バッファに同じセルの複数の値が含まれる場合に値を加算します。
 - **すべての値が等しいと想定** - 同じセルの複数の値が同一であることを確認します。同一の場合、重複値を無視します。同じセルの複数の値が異なる場合は、データ・ロードを停止してエラー・メッセージを生成します。
 - **最後の値を使用** - 最後にロード・バッファにロードされたセルの値を使用して、重複セルを結合します。このオプションは、セル数が 10,000 までの比較的小さなデータ・ロード向けです。このオプションを使用すると、重複値がない場合でもデータ・ロードが非常に遅くなります。

- **オプション** - 必要であれば次のオプションを選択できます:
 - **欠落値を無視** - 入力データ・ストリームの #MISSING 値を無視します。
 - **ゼロ値を無視** - 入力データ・ストリームの 0 を無視します。
 - 5. 「**レベル 0 データのインポートのステータス**」ダイアログ・ボックスで、「**詳細の表示**」をクリックしてインポートの詳細情報を表示し、「**OK**」をクリックします。
- データをインポートしたら、Oracle Hyperion Planning のフォームを使用してデータを表示したり、キューブ統計を表示して値の変化を確認したりできます。

ノート:

インポートされたデータにデータのインポート元以外のソースからのプランニング・テキスト値、スマート・リスト値、または通貨値が含まれる場合、データが破損する可能性があります。

ASO キューブからのレベル 0 データのエクスポート

ASO キューブからレベル 0 データをエクスポートするには:

1. **システム・ビュー**で  をクリックします。
2. 「**データベース・プロパティ**」タブの**エンタープライズ・ビュー**で、**集約ストレージ・アプリケーション**を展開します。
3. ASO キューブを右クリックして、「**レベル 0 データのエクスポート**」を選択します。
4. 「**キューブのレベル 0 データのエクスポート**」ダイアログ・ボックスで、エクスポートするデータの zip ファイル名を入力して「**OK**」をクリックします。

Oracle Hyperion Planning でエクスポートしたレベル 0 データを含む .zip ファイルを確認するには:

1. Planning のホーム・ページで「**アプリケーション**」をクリックし、「**概要**」をクリックします。
2. 「**アプリケーション**」タブで、「**アクション**」、「**受信ボックス/送信ボックス・エクスプローラ**」の順にクリックします。
3. 「**受信ボックス/送信ボックス・エクスプローラ**」で、.zip ファイルの横の「**アクション**」アイコンをクリックし、「**ファイルのダウンロード**」を選択します。

BSO キューブからのレベル 0 データのインポート

BSO キューブのレベル 0 データをインポートするには:

1. **システム・ビュー**で  をクリックします。
2. 「**データベース・プロパティ**」タブの**エンタープライズ・ビュー**で、**ブロック・ストレージ・アプリケーション**を展開します。

3. BSO キューブを右クリックして、「**レベル・ゼロ・データのインポート**」を選択します。
4. **キューブのレベル 0 データのインポート**・ダイアログ・ボックスで、インポートするデータの zip ファイル名を入力して「**OK**」をクリックします。

BSO キューブからのレベル 0 データのエクスポート

BSO キューブからレベル 0 データをエクスポートするには:

1. **システム・ビュー**で  をクリックします。
2. 「**データベース・プロパティ**」タブの**エンタープライズ・ビュー**で、**集約ストレージ・アプリケーション**を展開します。
3. BSO キューブを右クリックして、「**レベル 0 データのエクスポート**」を選択します。
4. 「**キューブのレベル 0 データのエクスポート**」ダイアログ・ボックスで、エクスポートするデータの zip ファイル名を入力して「**OK**」をクリックします。

Oracle Hyperion Planning でエクスポートしたレベル 0 データを含む.zip ファイルを確認するには:

1. Planning のホーム・ページで「**アプリケーション**」をクリックし、「**概要**」をクリックします。
2. 「**アプリケーション**」タブで、「**アクション**」、「**受信ボックス/送信ボックス・エクスプローラ**」の順にクリックします。
3. 「**受信ボックス/送信ボックス・エクスプローラ**」で、.zip ファイルの横の「**アクション**」アイコンをクリックし、「**ファイルのダウンロード**」を選択します。

増分データ・スライスのマージ

メイン・データベース・スライスにすべての増分データ・スライスをマージしたり、メイン・データベース・スライスを変更しないまま単一のデータ・スライスにすべての増分データ・スライスをマージしたりできます。

スライスをマージするには、データをロードする場合と同じ権限(管理者またはデータベース・マネージャ権限)を持っている必要があります。

ノート:

集約ストレージ・データベースに対してのみ増分データ・スライスをマージできます。

増分データ・スライスをマージするには:

1. **システム・ビュー**で  をクリックします
2. **エンタープライズ・ビュー**で、データをマージするデータベースを含むプラン・タイプを右クリックし、「**マージ・データ**」を選択して次のいずれかのオプションを選択します。
 - **すべて**: すべてのデータ・スライスを 1 つにマージします

- 値がゼロのセルを保持(デフォルト)
 - 値がゼロのセルを削除
 - **増分:** 増分データ・スライスを 1 つにマージし、次のいずれかを実行します
 - 値がゼロのセルを保持(デフォルト)
 - 値がゼロのセルを削除
3. 「データのマージ・アクションの確認」ダイアログで、「OK」をクリックして、データのマージを確認します。

データの集約

データが含まれ、ユーザーに計算権限が付与されている集約ストレージ・データベースに対して集約を計算します。

集約を実行するには、システム推奨のビューを使用します。ビューの選択と集約のプロセスをまとめて、サーバーによって実行される 1 つの構成不可能な操作にします。オプションで、生成されるファイルの最大ディスク・スペースを指定し、ユーザー問合せパターンに基づいてビューの選択を行い、ビュー選択にロールアップ階層を含めることができます。

ノート:

集約ストレージ・データベースのデータのみを集約できます。データを集約する場合のベスト・プラクティスの例については、[集約プロセスの実行](#)を参照してください。

集約を実行するには:

1. システム・ビューで  をクリックします
2. エンタープライズ・ビューで、アウトラインを圧縮するデータベースを含むプラン・タイプを右クリックし、「集約の実行」を選択します。
3. 「集約アクションの実行 - 推奨ビューの使用」ダイアログ・ボックスで、次のオプションのいずれかを選択します。
 - **問合せデータを基準にしますか?** - 収集されたユーザー問合せパターンに基づいて、サーバーで選択されるビューを集約します。このオプションは、問合せトラッキングがオンになっている場合のみ使用可能です。
 - **ロールアップ・オプションを含めますか?** - セカンダリ階層(デフォルトのレベルの使用)をビュー選択プロセスに含めます。
 - **成長サイズ・オプションを含めますか?** - 指定された集約データベースの増大限度を超えるまで、サーバーで選択されるビューを集約します。サーバーがそれを超えると集約を停止する必要があるサイズ(MB)を入力します。
4. 「OK」をクリックします。

既存の集約データが存在する場合、既存の集約を削除してから集約プロセスを再実行するかどうかを尋ねるメッセージが表示されます。既存データが存在する場合、集約プロセスを再実行する前に削除されます。

5. 既存の集約データを削除し、集約を再実行するには、「OK」をクリックします。

集約プロセスの実行

集約プロセスを実行すると、取得のパフォーマンスが向上します。

ASO キューブでは計算スクリプトを使用してデータを集約しません。かわりに、ASO は上位レベルのメンバーを動的に計算しようとしています。これによってバッチ処理時間は短くなりますが、取得時間は長くなる可能性があります。この状況は、問合せトラッキングを有効にして、フォームの操作やアドホック・レポートの実行などの操作のための ASO キューブに対する問合せを取得することで改善できます。これらの問合せは集約プロセスで使用され、Oracle Essbase に問合せトラッキングで取得された問合せパターンを使用して集約ビューを構築するよう指示されます。集約ビューの作成後、取得のパフォーマンスが向上したことがわかります。

集約プロセスを実行する前に、次のことを行います。

- 増分データ・スライスのマージおよびゼロ値のセルの削除
- 問合せトラッキングの有効化
- 問合せを作成するアクションの実行
- 問合せトラッキングを使用した集約の実行

増分データ・スライスのマージおよびゼロ値のセルの削除

Oracle Essbase 操作の中には、論理クリアやデータのロードなどのようにゼロ値のセルを使用して増分データ・スライスを作成するものがあります。Essbase では、増分データ・スライスでの集約はできません。そのため、増分データ・スライスをマージするためにマージ操作を実行する必要がある場合があります。

マージ操作を実行し、ゼロ値のセルを削除するには:

1. システム・ビューで  をクリックし、プロパティをロードする集約ストレージ・データベースを選択します。
2. エンタープライズ・ビューでキューブを右クリックし、「マージ・データ」、「すべて」、「値がゼロのセルを削除」の順に選択します。
3. 「OK」をクリックしてデータのマージ・アクションを確定します。

これによってすべての増分スライスがメイン・データベース・スライスにマージされ、値がゼロのセルが削除されます。(データを領域から論理クリアすると、値がゼロのセルになります)。その結果、データベースのサイズが大幅に小さくなります。

マージが必要ない場合、次のようなメッセージが表示されます: 増分データがないか、すでにマージされています。指定したマージは必要ありません。(マージ・アクションのステータス・ダイアログ・ボックスで「詳細の表示」をクリックすると、完全なメッセージが表示されます。)

問合せトラッキングの有効化

ASO データベースに対して問合せトラッキングを有効にして一連の有用な問合せを記録し、記録した問合せデータを使用してそのデータベース用に生成される最も適切な集約ビューのセットを選択できます。

Oracle Hyperion Planning で行われるキューブのリフレッシュ操作では、アウトラインの再構築操作が実行されます。再構築操作の一環として、Oracle Essbase は追跡された問合せを削除します。マージ操作でも、追跡された問合せが削除されます。

問合せトラッキングを有効にするには:

1. **エンタープライズ・ビュー**で「**Planning**」を展開し、ASO アプリケーションを展開して ASO キューブを右クリックし、「**問合せトラッキングの設定**」を選択します。
2. 「**OK**」をクリックして問合せトラッキング・アクションを確定します。
3. データベースに対して問合せトラッキングが正常に有効化されたことを示す「**情報**」ダイアログ・ボックスで「**OK**」をクリックします。

問合せトラッキングの有効化後は、前述のステップを繰り返して無効にできます。問合せトラッキングを無効にすると、「**情報**」ダイアログ・ボックスに問合せトラッキングが有効であることが示され、問合せトラッキングを無効にする場合、アプリケーションを停止するか、集約プロセスを実行するよう求められます。

問合せを作成するアクションの実行

問合せトラッキングはメモリー内にもみ格納され、Oracle Hyperion Planning で ASO キューブを使用してフォームを開いたり、アドホック・レポートを実行するなどの操作のための問合せを追跡します。

問合せトラッキングを有効にした後、フォームのロード、ビジネス・ルールの実行およびレポートの実行を続けることができます。Oracle Essbase は問合せの追跡および統計の格納を続けます。集約を行う際、これらの追跡した統計を使用できます。

問合せトラッキングを使用した集約の実行

十分な量の問合せを実行したら、問合せトラッキング情報を使用して集約プロセスを実行できます。

問合せトラッキング情報を使用して集約プロセスを実行するには:

1. **エンタープライズ・ビュー**で「**Planning**」を展開し、ASO アプリケーションを展開して ASO キューブを右クリックし、「**集約の実行**」を選択します。
2. 「**集約アクションの実行**」ダイアログ・ボックスで「**問合せデータを基準にしますか?**」を選択して「**OK**」をクリックします。

この操作は完了に時間がかかることがあります。

Oracle Hyperion Calculation Manager は、実行集約プロセスの失敗の原因となる可能性のある次の状態がないかチェックします。

- 問合せトラッキングが有効でないか、問合せトラッキング・データがない場合、警告メッセージが表示されます。問合せトラッキング・オプションを使用することはできませんが、集約を処理することはできます。問合せトラッキング情報を使用するには、問合せトラッキングを有効にするか、問合せトラッキング方法を生成する操作を実行する必要があります。
- Oracle Essbase では、複数のデータベース・スライスが存在する場合、ASO キューブに対して集約ビューは作成できません。警告メッセージが表示され、スライスをマージするまで集約を実行することはできません。

この場合、**エンタープライズ・ビュー**でキューブ・ノードを右クリックし、「**マージ・データ**」、「**すべて**」、「**値がゼロのセルを削除**」の順に選択します。マージですべての追跡された問合せ情報がクリアされます。そのため、問合せトラッキング情報を生成する操作を実行する必要があります。

- 集約がすでに存在する場合、警告メッセージが表示されます。この場合、「**OK**」をクリックして、「**情報**」ダイアログ・ボックスで次のいずれかを行います。
 - 「**OK**」をクリックし、既存の集約データを削除して集約プロセスを再実行します。
 - 「**取消し**」をクリックし、既存の集約をクリアして問合せトラッキング情報を生成する操作を実行します。

集約をクリアするには:

1. ASO キューブを右クリックし、「**クリア**」、「**すべての集約**」の順に選択します。
2. 「**データベースの集約のクリアの確認**」ダイアログ・ボックスで「**OK**」をクリックします。

要求の管理

セッション・ウィンドウの情報を使用してアクティブな要求を管理します。

セッション・ウィンドウには、サーバー、アプリケーションまたはデータベースのアクティブなユーザー・セッションおよび要求がリストされます。ユーザーは、一度に複数のセッションを開くことが可能です。たとえば、1人のユーザーが2つのデータベースでセッションを開くことができます。

管理者またはアプリケーション・マネージャ権限を持っているユーザーは、ユーザー・セッションを切断したり、セッション中に行われた特定の要求を終了したりできます。

要求を終了するには:

1. **システム・ビュー**で  をクリックします
2. **エンタープライズ・ビュー**でアプリケーションを右クリックし、「**セッション**」を選択します。

セッション・ウィンドウには、アクティブなセッションと要求のリストが表示されます。管理者権限を持っている場合は、サーバー上のすべてのユーザーに対するアクティブなユーザー・セッションがウィンドウにリストされます。アプリケーション・マネージャ権限を持っている場合は、アプリケーション・マネージャ権限を持つアプリケーションに接続しているすべてのユーザー(自分を含む)のアクティブ・セッションがリストされます。

3. ユーザーをログオフするには、「**オプション**」の「**アクション**」ドロップダウンから「**ログオフ**」を選択し、次のタスクのいずれかを実行します。
 - 「**エンティティ**」から「**選択したユーザー**」を選択し、ログオフするユーザーを選択します。次に、「**適用**」をクリックし、ユーザーをログオフします。
 - 「**エンティティ**」から、「**すべてのユーザー**」を選択し、次のタスクのいずれかを実行します。
 - 「**ソース**」から、「**選択したサーバー**」を選択して、選択したサーバー上のすべてのユーザーをログオフします。次に、「**適用**」をクリックします。
 - 「**ソース**」から、「**選択したアプリケーション**」を選択して、選択したアプリケーション上のすべてのユーザーをログオフします。次に、「**適用**」をクリックします。

- 「ソース」から、「選択したデータベース」を選択して、選択したデータベース上のすべてのユーザーをログオフします。次に、「適用」をクリックします。
- 「エンティティ」から、「ユーザーのすべてのインスタンス」を選択し、次のタスクのいずれかを実行します。
 - 「ソース」から、「選択したサーバー」を選択し、ログオフするユーザー・インスタンスを選択します。「適用」をクリックし、サーバーからユーザーのすべてのインスタンスをログオフします。
 - 「ソース」から、「選択したアプリケーション」を選択し、ログオフするユーザー・インスタンスを選択します。「適用」をクリックし、アプリケーションからユーザーのすべてのインスタンスをログオフします。
 - 「ソース」から、「選択したデータベース」を選択し、ログオフするユーザー・インスタンスを選択します。「適用」をクリックし、データベースからユーザーのすべてのインスタンスをログオフします。
- 4. 要求を終了するには、「オプション」の「アクション」ドロップダウンから「強制終了」を選択し、次のタスクのいずれかを実行します。
 - 「エンティティ」から、「選択した要求」を選択し、終了する要求をセッション・リストから選択します。次に、「適用」をクリックして、選択した要求を終了します。
 - 「エンティティ」ドロップダウンから、「すべての要求」を選択し、次のタスクのいずれかを実行します。
 - 「ソース」から、「選択したサーバー」を選択して、選択したサーバー上のすべてのユーザーからのすべての要求を終了します。次に、「適用」をクリックします。
 - 「ソース」から、「選択したアプリケーション」を選択して、選択したアプリケーション上のすべてのユーザーからのすべての要求を終了します。次に、「適用」をクリックします。
 - 「ソース」から、「選択したデータベース」を選択して、選択したデータベース上のすべてのユーザーからのすべての要求を終了します。次に、「適用」をクリックします。
 - 「エンティティ」から、「ユーザーからのすべての要求」を選択し、次のタスクのいずれかを実行します。
 - 「ソース」から、「選択したサーバー」を選択し、ユーザーの要求をセッション・リストから選択します。「適用」をクリックして、サーバー上のこのユーザーからのすべての要求を終了します。
 - 「ソース」から、「選択したアプリケーション」を選択し、ユーザーの要求をセッション・リストから選択します。「適用」をクリックして、アプリケーション上のこのユーザーからのすべての要求を終了します。
 - 「ソース」から、「選択したデータベース」を選択し、ユーザーの要求をセッション・リストから選択します。「適用」をクリックして、データベース上のこのユーザーからのすべての要求を終了します。
- 5. 列の表示と非表示を切り替えるには、「表示」、「列」の順に選択します。次のいずれかの操作を行います。
 - すべての列を表示するには、「すべて表示」を選択します。デフォルトでは、「接続ソース」列を除くすべての列が表示されます。
 - 「列の管理」を選択して、「非表示列」と「表示列」リストの間で列を移動します。

- 列を並べ替えるには、「表示」、「列の順序変更」の順に選択します。希望どおり列を並べ替えるには、「列の順序変更」ダイアログで上下の矢印キーを使用します。
- 列でセッション・リストをソートするには:
 - 列を昇順でソートするには、列ヘッダーをクリックするか、列ヘッダーの横にある上矢印をクリックします。
たとえば、「ユーザー」列をアルファベット順にソートするには、列ヘッダーをクリックします。
 - 列を降順でソートするには、[Shift]を押しながら列ヘッダーをクリックするか、列ヘッダーの横にある下矢印をクリックします。
たとえば、最も長いログイン時間がリストの先頭に表示されるように「ログイン時間」列をソートするには、[Shift]を押しながら列ヘッダーをクリックします。
- セッション・リストをリフレッシュするには、「リフレッシュ」をクリックします。たとえば、処理中と表示されているセッションを終了し「リフレッシュ」をクリックすると、セッション・リストには、セッションは処理中ではないと表示されます。
- Microsoft Excel スプレッドシートにセッション・データをエクスポートするには、「エクスポート」ボタンをクリックして、ファイルを開くか保存して、「OK」をクリックします。

Planning ドリル・スルー定義の追加

Oracle Hyperion Calculation Manager で、Oracle Hyperion Planning プラン・タイプに対するこれらのセルのドリル・スルー定義をリスト、追加、編集および削除できます。

Planning プラン・タイプおよび Oracle Hyperion Financial Data Quality Management, Enterprise Edition などのソースからロードされたデータを持つメンバーを含む Planning フォームを使用する場合、ドリル・スルーしてセル・データ・ソースの詳細を表示できます。

ノート:

Planning フォームのドリル・スルーの有効化の詳細は、*Oracle Hyperion Planning 管理者ガイド*のドリル・スルー情報のためのフォームの設計に関する項を参照してください。

Planning ドリル・スルー定義を追加するには:

- システム・ビューで  をクリックします。
- エンタープライズ・ビューでアプリケーションを右クリックし、「ドリル・スルー定義」を選択します。
- 「ドリル・スルー定義」ダイアログ・ボックスで  をクリックします。
- 「ドリル・スルー定義の作成」で次の情報を入力し、ドリル・スルー定義を作成します。
 - URL 名 - ドリル・スルー定義を識別する名前
 - XML コンテンツ - URL リンクを定義する XML

サーバーおよびポート情報のない URL を入力します。URL には、\$記号で囲まれた TDATESEG 表のパラメータ名および列名が含まれている必要があります。たとえば、LEDGER_ID=\$ATTR1\$&GL_PERIOD=\$ATTR2\$を入力します。この例では、ATTR1 の値は LEDGER_ID パラメータの値として渡され、ATTR2 は GL_PERIOD パラメータの値として渡されます。パラメータは&記号で区切られます。

ドリル URL のフォーマットでクライアントとサーバー間の request-response を指定するには、次のいずれかを入力します。

- GET - フォーム・データを URL にエンコードします
たとえば、GET@http://www.oracle.com/と入力します。メソッドが入力されない場合、GET がリクエスト・レスポンスとみなされます。
- POST - メッセージ・ボディのフォーム・データを表示します
たとえば、POST@http://www.oracle.com/と入力します

XML コンテンツを入力したら、をクリックしてファイルを「XML コンテンツ」領域にインポートし、をクリックして XML コンテンツを Microsoft Excel にエクスポートします。

- **レベル 0 フラグ** - URL を領域のレベル 0 子孫のみに適用するかどうか。
たとえば、レベル 0 フラグがドリル可能な領域 DESCENDANTS ("Market"), @CHILDREN (Qtr1) に対して有効な場合、URL は "Market"のすべての州の"Qtr1"のすべての月および残りのディメンションのすべてのレベル 0 メンバーに適用可能です。
- **領域** - 指定した URL を使用したドリル・スルーを許可するデータベースのメンバー指定の定義領域
1 つ以上のディメンションのメンバーのメンバー指定を使用してドリル可能な領域を定義します。セキュリティ・フィルタの定義に使用するものと同じ Oracle Essbase メンバー・セット計算言語を使用して、メンバー指定を定義します。たとえば、次の指定は、"New York"以外のすべての東部の州の"Qtr1"の月を示す有効なメンバー指定です: @REMOVE (@DESCENDANTS ("Eastern Region"), "New York"), @CHILDREN (Qtr1)。
領域を追加するには、「**領域の追加**」をクリックします。

5. 「保存」、「OK」の順にクリックします。

A

管理者機能の使用

Calculation Manager への移行

今回のリリースでは、Financial Management または Planning で作成されたアプリケーションを使用している Oracle Hyperion Financial Management ユーザーと Oracle Hyperion Planning ユーザー、さらに Oracle Essbase ブロック・ストレージ・アプリケーションおよび集約ストレージ・アプリケーション・ユーザーが、Oracle Hyperion Calculation Manager でビジネス・ルールを作成して管理できます。

次の表は、Calculation Manager、Oracle Hyperion Business Rules (Planning および Essbase ユーザー用)、および Financial Management (Financial Management ユーザー用)のビジネス・ルールの機能を比較しています。

表 A-1 Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
ビジネス・ルールの作成	<p>ルール・デザイナー内のフロー・チャートでビジネス・ルールをグラフィカルに作成することができます。</p> <p>ビジネス・ルールを設計するために使用するコンポーネントが、Planning と Essbase ブロック・ストレージ・アプリケーションには 5 つ、Financial Management アプリケーションには 6 つあります。</p> <ol style="list-style-type: none"> 1. 式コンポーネント: メンバー、関数、条件文(オプション)を使用して作成された計算文が含まれています。 2. スクリプト・コンポーネント: Visual Basic (Financial Management) または Essbase (Planning) の計算スクリプト文のみ含まれています。 3. 条件コンポーネントには、true または false の条件コンポーネント文が含まれます(つまり、If...Then 文)。 4. ループ・コンポーネントには、メタデータ・メンバーのリストが含まれます(勘定科目のリストなど)。 5. メンバー範囲コンポーネントには、メタデータ・メンバーの一覧が含まれます(勘定科目のリストなど)。 6. (Financial Management ユーザーのみ) データ範囲コンポーネントにはデータ・レコードのリストが含まれます(勘定科目値のリストなど)。 <p>Essbase 集約ストレージ・アプリケーションには、次の 3 つのコンポーネントがあります:</p>	<p>ビジネス・ルール・グラフィカル・デザイナーおよび Oracle Essbase Administration Services の管理コンソールのビジネス・ルール内でビジネス・ルールを作成することができます。</p> <p>グラフィカルにビジネス・ルールをデザインできる 4 つのアクションおよび 8 つの式があります。</p> <ol style="list-style-type: none"> 1. データの集約アクション 2. データのコピー・アクション 3. データのクリア・アクション 4. ブロックの作成アクション 5. 比例配分率式 6. 分布係数式 7. 均等分割式 8. 増加/減少式 9. 単価レート式 10. 結合式 11. カスタム式 12. 変数式 	<p>ルール・エディタまたはテキスト・エディタでビジネス・ルールを作成できます。Visual Basic および Financial Management の関数、そして Financial Management ビジネス・ルール内のメンバーを使用することができます。</p>

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合) Financial Management
<ol style="list-style-type: none"> 1. 視点コンポーネント: メンバーの範囲コンポーネントのように、メタデータ・メンバーのリストが含まれます。 2. 割当てコンポーネント: データベース・アウトラインの 1 つのレベルにあるメンバーからアウトラインの他のメンバーにデータを配布するための計算が含まれます。 3. 式コンポーネント: ユーザーがメンバー、関数および変数を使用して設計する計算文が含まれます。 	<p>コンポーネント内で、メンバー、変数および関数を使用できます。</p>	

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
システム・テンプレート (Calculation Manager)、およびアクションと式 (Business Rules) を使用して、ビジネス・ルールを設計できます。	<p>ノート: システム・テンプレートは、Essbase 集約ストレージ・アプリケーションでサポートされていません。</p> <p>Calculation Manager には 8 つの Planning および Essbase ブロック・ストレージ・システム・テンプレートがあります:</p> <ol style="list-style-type: none"> 「集約」テンプレートはデータを集約します。 「データのコピー」テンプレートはデータをコピーします。 「データのクリア」テンプレートはデータを削除します。 「割当て - 単純」テンプレートは、ある場所から別の場所にデータを割り当てます。 「割当て - レベル間」テンプレートは、複数レベルのデータを割り当てます。 「金額・レート・単価」テンプレートは、他の 2 つに値を提供する場合に 1 つの変数を計算します。 「データのエクスポート」テンプレートはデータをデータベースまたはファイルにエクスポートします。 「SET コマンド」テンプレートでは、計算スクリプトのパフォーマンスを最適化するスクリプト・コマンドが入力できます。 <p>Calculation Manager には、8 つの Financial Management システム・テンプレートがあります:</p> <ol style="list-style-type: none"> 財務的な丸め処理テンプレートは統計的な丸め処理ではなく、データを財務的に丸めます。財務的な 	<p>Business Rules には、次の 4 つの Calculation Manager のいくつかのシステム・テンプレートのように機能するアクションがあります。</p> <ol style="list-style-type: none"> データの集約アクションは、Calculation Manager の集約テンプレートのように動作します。 「データのコピー」アクションは、Calculation Manager の「データのコピー」テンプレートのように機能します。 「データのクリア」アクションは、「ブロックの作成」アクションとともに、Calculation Manager の「データのクリア」テンプレートのように機能します。 ブロックの作成アクションは、「データのクリア」アクションと連動して、計算マネージャの「データのクリア」テンプレートのように動作します。 <p>Business Rules には、計算マネージャのシステム・テンプレートのように機能する式もあります。</p> <ol style="list-style-type: none"> 比率配分式は、計算マネージャの「割当て - 単純」テンプレートのように機能します。 3 つの単価レート式は、計算マネージャの金額レート単価テンプレートのように機能します。 	該当なし

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)
	<p>丸め処理関数は、四捨五入法を使用してデータを指定した小数点以下の桁数に丸められます。</p>	
2.	<p>月の日数の取得テンプレートは指定された年と月の数に基づいて、月の日数を生成します。月は通常の 1 から 12 の範囲外の数字として入力できます。たとえば、年(2008)および期間番号(0)を入力すると、2007 年 12 月の日数が戻されます。年(2008)および期間番号(14)を入力すると、2009 年の 2 月の日数が戻されます。この関数は、うるう年にも対応しています。</p>	
3.	<p>リスト内テンプレートでは、指定のディメンション・メンバーが指定のメンバー・リストのメンバーかどうかをテストします。</p>	
4.	<p>平均貸借対照表テンプレートでは、MTD、QTD、HYTD および YTD の頻度の平均貸借対照表比率を計算します。データは MTD< QTD、HYTD< YTD、または累積日次残高として入力できます。</p>	
5.	<p>期首残高テンプレートでは、指定した取得メソッドに基づいて勘定科目の期首残高を計算します。期首残高は同じ値通貨またはエンティティ通貨の合計から取得できます。</p>	
6.	<p>「複雑な集計」テンプレートは、各エンティティに割り当てられた連結メソッド(持株、グローバル、比例、資本)に基づいて、連結グループ内の各エンティティに対して連結と消去を行う組込み済の連結ルールです。資本、投資、純</p>	

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
	<p>利益および標準消去の計算を処理します。監査トランザクションは、ユーザーが設定する監査フラグに基づいて生成されます。このテンプレートの連結ルールで、通常の法的要件のほとんどの計算を行います。</p>		
	<p>7. 「標準連結」テンプレートは、デフォルト連結プロセスを使用して連結グループ内の各エンティティに対して連結および消去を行う組込み済の連結ルールです。</p>		
	<p>8. エンティティ 割当てテンプレートでは、指定された割当て加重に基づいて、ソース勘定科目がグループの親エンティティからリストの各エンティティに対応する宛先勘定科目に割り当てられます。</p>		
<p>カスタム定義のテンプレート (Calculation Manager) とマクロ (Business Rules) の作成によるビジネス・ルールの設計</p>	<p>ウィザードを使用して、ビジネスに固有の計算を実行する、再利用可能なカスタム定義のテンプレートを設計できます。つまり、システム・テンプレートをコピーして別の名前で作成し、これをカスタム定義のテンプレートとして使用できます。</p>	<p>ビジネスに固有の計算を実行する、再利用可能なマクロを設計できます。</p>	<p>該当なし</p>
<p>設計時プロンプトの作成</p>	<p>ウィザードを使用して、Calculation Manager のカスタム定義のテンプレート用の設計時プロンプトを作成できます。</p>	<p>マクロの変数を使用して、情報を求めるプロンプトを表示できます。</p>	<p>該当なし</p>
<p>コンポーネントの共有</p>	<p>複数のプラン・タイプ (Planning の場合)、計算タイプ (Financial Management の場合)、データベース・タイプ (Essbase の場合)、およびアプリケーションで、スクリプトと式コンポーネントを共有できます。</p>	<p>該当なし</p>	<p>該当なし</p>

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
コンポーネントのドラッグ・アンド・ドロップ	ルール・デザイナーでは、コンポーネントをビジネス・ルールのフロー・チャートにドラッグ・アンド・ドロップできます。	グラフィカル・デザイナーでは、アクションと式をビジネス・ルールのプロセス・バーにドラッグ・アンド・ドロップできます。	該当なし
詳細の表示/非表示	フロー・チャートの表示をズーム・イン/ズーム・アウトして、表示の詳細レベルを調整できます。	該当なし	該当なし
計算スクリプト・フォーマットでのビジネス・ルールの編集	ビジネス・ルールを計算スクリプト・フォーマットで編集した後で、ルール・エディタでグラフィカルな形式に戻して編集できます。	計算スクリプト・フォーマットでビジネス・ルールを編集できますが、それをグラフィカルの編集に戻すことはできません。	ビジネス・ルールはテキスト・モードでのみ編集可能です。
ビジネス・ルールの検証	Financial Management または Planning アプリケーション管理で作成されたアプリケーションで作業する場合: Financial Management または Planning それぞれに対してのみビジネス・ルールを検証できます。 ノート: Essbase ビジネス・ルールは、Essbase に対してのみ検証できます。	ビジネス・ルールを Essbase または Planning に対して検証できます。	スキャン関数を使用して、Visual Basic スクリプト・ファイルをスキャンして、関数が正しいパラメータ番号に有効であり、ディメンション・メンバーがアプリケーションに対して有効であることを確認できます。
ビジネス・ルールのデプロイ	計算タイプごとに 1 つのビジネス・ルールセットを Financial Management にデプロイできます。つまり、1 つ以上のビジネス・ルールおよびビジネス・ルールセットを Planning および Essbase にデプロイできます。ビジネス・ルールおよびビジネス・ルールセットをデプロイし、それらを Financial Management、Planning または Administration Services (Essbase の場合)内から起動する必要があります。それらの起動場所は、それらをデプロイするアプリケーションおよび計算のタイプ、プラン・タイプ、またはデータベースによって決まります。	該当なし	該当なし

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
ビジネス・ルールの起動	<p>ビジネス・ルールセットを Calculation Manager からデプロイした後、Financial Management および Planning ビジネス・ルールは Financial Management および Planning 内から起動します。Essbase ビジネス・ルールをデプロイすると、Calculation Manager または Administration Services 内からそれらのルールを起動できます。起動場所は、ビジネス・ルールおよびビジネス・ルールセットのデプロイ場所によって決まります。ビジネス・ルールの複数の起動場所を定義するには、そのショートカットを Calculation Manager に作成します。</p>	<p>起動する場所を選択することにより、ビジネス・ルールを 1 つの場所からでも、すべての場所からでも起動できます。Administration Services 管理コンソールのルール・ノード、Business Rules のグラフィック・デザイナー、Business Rules Web ランチャ、コマンド・ライン・プロンプトまたは Planning Web から、ビジネス・ルールを起動できます。</p>	<p>ビジネス・ルールは、計算、換算、連結、配賦を実行するときに実行します。</p>
ビューでの作業	<p>Calculation Manager には、アクセスできるオブジェクトを表示する、次の 4 つのビューがあります。</p> <ol style="list-style-type: none"> 1. システム・ビュー 2. カスタム・ビュー 3. デプロイメント・ビュー 4. リスト・ビュー 	<p>Business Rules にはビューはありません。ビジネス・ルールやその他のオブジェクトは、管理コンソール内でオブジェクト・タイプ別にフラット・リストに表示されます。</p>	<p>該当なし</p>
システム・ビューの使用	<p>システム・ビューには、Financial Management、Planning および Essbase アプリケーションと、それらの計算タイプ、プラン・タイプ、データベース、ユーザーがアクセスできるオブジェクトが階層的なリストで表示されます。このビューから、ビジネス・ルールの対象アプリケーションおよび計算タイプ、プラン・タイプ、またはデータベースを確認できます(これは、計算マネージャのデフォルトのビューです)。</p>	<p>該当なし</p>	<p>該当なし</p>

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
カスタム・ビューの使用	カスタム・ビューには、ユーザーが作成したフォルダと、そこにドラッグ・アンド・ドロップしたオブジェクトのリストが表示されます。このビューにより、ユーザーの目的に合わせてオブジェクトを整理できます。	¹ 該当なし	該当なし
デプロイメント・ビューの使用	デプロイメント・ビューには、デプロイ済および未デプロイのルールとルールセット、それらのデプロイと検証ステータスのリストが、アプリケーション・タイプとアプリケーション別に表示されます。	該当なし	該当なし
リスト・ビューの使用	リスト・ビューには、「フィルタ」ダイアログで選択したオブジェクトのリストが表示されます。「フィルタ」ダイアログを使用して、アプリケーション・タイプ、属するアプリケーション、計算タイプおよびプラン・タイプ、およびアクセスできるオブジェクトでフィルタされたリストを作成できます。	該当なし	該当なし
ビジネス・ルールの移行	ビジネス・ルールおよびその他のオブジェクトを前回リリースされた Business Rules から今回リリースされる Calculation Manager に移行することができます。Financial Management および Essbase のビジネス・ルールを移行するには、Calculation Manager のインポート機能を使用します。	ビジネス・ルールは、前のリリースの Business Rules から今回のリリースの Calculation Manager、および前のリリースの Business Rules から今回のリリースの Business Rules へ移行できます。	ビジネス・ルールは、前のリリースの Financial Management から今回のリリースの Calculation Manager、および前のリリースの Financial Management から今回のリリースの Financial Management へ移行できません。

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
ビジネス・ルールのインポート	ビジネス・ルール、およびテンプレートやコンポーネントなどのその他のオブジェクトは、Business Rules から Calculation Manager へ、または Calculation Manager 内の別の Financial Management、Planning または Essbase アプリケーションからインポートできます。Essbase の計算スクリプトを Calculation Manager にインポートすることもできます。インポートした計算スクリプト・ファイルは、Calculation Manager でグラフィカルなビジネス・ルールになります。	ビジネス・ルールを xml ファイルから Business Rules にインポートできます。	「ルールのロード」オプションを使用して、有効な Visual Basic スクリプト・ファイルをアプリケーションにインポートできます。
ビジネス・ルールのエクスポート	1 つ以上のビジネス・ルールおよびその他のオブジェクトを Calculation Manager から.xml ファイルにエクスポートできます。	ビジネス・ルールとその他のオブジェクトは、Business Rules から xml ファイルにエクスポートできます。	「抽出ルール」オプションを使用して、Financial Management から外部の Visual Basic スクリプト・ファイルにビジネス・ルールをエクスポートできます。
ビジネス・ルールのショートカットの使用	複数のアプリケーション、および計算タイプまたはプラン・タイプで、ビジネス・ルールへのショートカットを作成できません。ショートカットを使用してビジネス・ルールをデプロイすると、ショートカットの作成時に対象としたアプリケーション、および計算タイプまたはプラン・タイプに、ルールのコピーがデプロイされます。	該当なし	該当なし

表 A-1 (続き) Calculation Manager、Financial Management および Business Rules におけるビジネス・ルールの機能比較

機能/アクション	Calculation Manager	Business Rules (Planning および Essbase ユーザーの場合)	Financial Management
変数の操作	<p>Calculation Manager の変数タイプの個数は、Planning は 4 個、Financial Management は 2 個、Essbase は 3 個です:</p> <ol style="list-style-type: none"> 1. グローバル: あらゆる Planning または Financial Management アプリケーションで使用できます 2. アプリケーション: 作成された Planning、Financial Management または Essbase アプリケーションでのみ使用できます 3. プラン・タイプまたはデータベース: 作成された Planning プラン・タイプまたは Essbase データベースでのみ使用できます 4. ビジネス・ルール: 作成された Planning または Essbase ビジネス・ルールでのみ使用できます 	<p>Business Rules には 2 つの変数タイプがあります。</p> <ol style="list-style-type: none"> 1. グローバル: どのビジネス・ルールでも使用できます。 2. ローカル: 作成されたビジネス・ルールでのみ使用できます。 	<p>Financial Management には 2 つの変数タイプがあります。</p> <ol style="list-style-type: none"> 1. グローバル: 計算プロセス全体に適用できます。 2. ローカル: 個々のサブルーチンにのみ適用できます。
ビジネス・ルールを作成および編集するためのアクセス権限の割当て	<p>Calculation Manager では、ビジネス・ルールおよび他のオブジェクトの作成、表示および編集能力は、Oracle Hyperion Shared Services で割り当てられた役割および Calculation Manager でのオブジェクト所有権によって決まります。デフォルトでは、ユーザーは自分が作成したビジネス・ルールとオブジェクトを所有します。管理者やオブジェクトの所有者は、オブジェクトの所有権を別のユーザーに割り当てることができます。</p>	<p>Business Rules では、ビジネス・ルールおよび他のオブジェクトを変更できるユーザーとグループを選択して、それらを編集する権限を割り当てます。</p>	<p>Financial Management では、ビジネス・ルールを作成または編集するためのアクセス権限は必要ありません。</p>
ビジネス・ルールを起動するためのアクセス権限を割当て	<p>ビジネス・ルールおよびビジネス・ルールセットを Calculation Manager からデプロイしたら、Financial Management、Planning または Administration Services (Essbase 用) で、ビジネス・ルールとビジネス・ルールセットを起動する権限を割り当てます。</p>	<p>ビジネス・ルールとシーケンスを起動できるデータベースの場所(またはすべての場所)を選択し、それらを起動できるユーザーとグループを選択することにより Business Rules でビジネス・ルールとシーケンスを起動できる権限を割り当てます。</p>	<p>計算プロセスを実行できる権限を割り当てるには、Financial Management で、計算プロセスが属するアプリケーションに対する適切な役割のセキュリティ・アクセスを(ユーザーに)割り当てます。たとえば、集計プロセスを実行するには、「集計」の役割が必要です。</p>

- ¹ **Busines Rules** でプロジェクトを作成し、ビジネス・ルール、シーケンス、マクロ、変数を独自のわかりやすい方法で編成できます。