Oracle® Data Relationship Management 套件
管理员指南

11.2.x 版
F28783-03
2022 年 7 月

Oracle Data Relationship Management 套件 管理员指南 11.2.x 版

F28783-03

第一作者：EPM Information Development Team

目录

文档可访问性

有关 Oracle 对可访问性的承诺，请访问 Oracle Accessibility Program 网站 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc。

获得 Oracle 支持

购买了支持服务的 Oracle 客户可通过 My Oracle Support 获得电子支持。有关信息，请访问 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info；如果您听力受损，请访问 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs。

文档反馈

要提供有关此文档的反馈，请单击任意 Oracle 帮助中心主题中页面底部的“反馈”按钮。还可以向 epmdoc_ww@oracle.com 发送电子邮件。

1

修订历史记录

在该版本指南中，更新了以下主题：

主题	更改
数据类型	针对浮点和整数增添了以下注释：如果未定义默认值，将导出 0。 增添了以下信息：日期、日期/时间和时间数据类型的格式不因文化而异。
函数定义	更新了 Equals 函数，以说明比较区分大小写。
创建动态脚本	增添了以下注释：计算父代名称时，任何特殊字符的使用必须遵循特殊字符转义的标准 JavaScript 规则。
迁移 Data Relationship Management 元数据	增添了注释，以指明用于外部连接的连接字符串、用户 ID 和密码不会随迁移加载和提取操作进行迁移。 增添了新的一节“迁移核心属性配置和设置”。
Data Relationship Management 对象	更新了 RequestItemObject 的 NodeNamePendingInRequest 方法的说明。
计算的名称和父代属性	增添了注释，以阐明手动覆盖名称或父代时会出现的行为。
创建工作流模型	为步骤 6 中的“重新计算任务属性”要点增添了注释，以阐明手动覆盖名称或父代时会出现的行为。
管理动态脚本	更新了 HierarchyObject Methods 表中 NodeExists(abbrev) 的说明 为 RequestItemDetailObject 增添了 2 个新属性： • CalcValue • HasCalcValue
通知	进行了多处更新，以阐明和更新通知行为。
支持的 JavaScript 数据类型	增添了注释，以阐明使用数组的情况。
为移动操作创建脚本验证	增添了一个新主题“为移动操作创建脚本验证”
系统首选项	为 FindByProperties 系统首选项的说明添加了注释。 更新了 SharedNodeDelimiter 和 SharedNodeSequenceSeparator 系统首选项的说明。
函数定义	阐明了多个函数的适用范围为本地。
故障排除和技巧	在“入门”一章中增添了新的一节“故障排除和技巧”。 增添了有关粘贴到字段中的解决方法信息。 增添了应用程序性能信息。

主题	更改
验证类	增添了有关 UniqueProp 验证使用具有索引的属性的建议。

2

关于 Data Relationship Management 套件

Oracle Data Relationship Management 套件包括：

- Oracle Data Relationship Management
- Oracle Data Relationship Management Read Only Access
- Oracle Data Relationship Steward
- Oracle Data Relationship Governance
- Oracle Data Relationship Management Analytics
- 适用于 Oracle Hyperion Enterprise Planning Suite 的 Oracle Data Relationship Management
- 适用于 Oracle Hyperion Financial Close Suite 的 Oracle Data Relationship Management

3
入门

另请参阅：

- 管理 Data Relationship Management 应用程序
- 访问 Data Relationship Management
- 故障排除和技巧

管理 Data Relationship Management 应用程序

Oracle Data Relationship Management 使用应用程序来管理数据和满足访问及更改数据的用户请求。一个 Data Relationship Management 安装可以支持一个或多个应用程序。每个应用程序都使用自己的系统元数据和安全配置来管理和访问数据。同一个应用程序可支持多个数据集和多个用户，并支持用户对公共和受限数据集拥有不同级别的访问权限。但是，一个应用程序中的所有系统元数据都由相同的用户共享和管理。对系统元数据执行的任何更改都会立即生效，且所有用户和数据都会受到影响。如果需要将不同的用户组与其他组所做的任何元数据更改隔离，建议每个组使用单独的应用程序。

应用程序在配置控制台中创建，可从 Data Relationship Management 主应用程序服务器访问此配置控制台。有关创建新的应用程序的详细信息，请参阅《*Oracle Data Relationship Management* 安装指南》中的“创建应用程序”。

新的 Data Relationship Management 应用程序包括核心元数据对象，如属性定义和类别以及一个默认的管理用户。利用该初始配置，默认用户可通过执行四个任务来构建、填充和设置应用程序：

- 创建版本和层次
- 定义用户元数据对象，如查询、比较、导入、混合器和导出
- 设置和配置系统元数据对象，包括域、属性定义、变量和节点类型
- 添加用户并配置访问产品功能、对象和数据的安全性

本指南涵盖了与 Data Relationship Management 应用程序的系统元数据和用户安全相关的管理任务。有关管理版本、层次和用户元数据对象的信息，请参阅《*Oracle Data Relationship Management* 用户指南》。

访问 Data Relationship Management

要启动 Oracle Data Relationship Management 客户端：

1. 依次选择**开始**、**程序**、**Oracle EPM System**、**Data Relationship Management** 和 **Web 客户端**。
2. 输入您的用户名和密码。

   用户名和密码区分大小写。
3. 选择应用程序，然后单击**登录**。

有关详细信息，请参阅“更改密码”。

更改密码

要更改密码：

1. 在 Oracle Data Relationship Management 主页上，选择**首选项**。
2. 单击**更改我的密码**。
3. 键入当前密码。
4. 键入新密码。

> **注：**
>
> 用户执行本地身份验证，并且 PasswordPolicyEnabled 系统首选项设置为 True 时，密码必须包含以下元素中的三种元素：

- 大写字母
- 小写字母
- 数字
- 特殊字符

> **注：**
>
> 在其他情况下，密码无限制，除非用户通过 Oracle Hyperion Shared Services 执行身份验证时由外部目录进行限制。

5. 再次键入新密码。
6. 单击**确定**。

故障排除和技巧

粘贴到输入字段

在某些情况下，无法通过先右键单击再选择**粘贴**的方式来粘贴剪贴板中的内容。为解决此问题，可以使用 Ctrl-V，或者先单击**编辑**再选择**粘贴**来粘贴剪贴板中的内容。

应用程序性能

为了保持应用程序性能，采用了标准编程做法以利用称为字符串驻留的功能，该功能可以提高字符串数据的访问速度。字符串驻留是指每个字符串的不可变副本一旦存储，便会在应用程序运行期间一直保留以供后续访问。因此，随着不断访问数据和管理内容，引擎占用的内存将以增量方式明显增长，直到应用程序重新启动为止。

4

管理用户

另请参阅：

- 用户权限
- 用户角色
- 创建用户
- 用户身份验证
- 修改用户
- 删除用户
- 查看用户的登录状态
- 系统定义的用户
- 通用用户设置

用户权限

Oracle Data Relationship Management 使用三种级别的权限控制用户对产品功能和数据的访问。某些高级别的权限还包括低级别的权限。如果为用户授予了高级别的权限，那么同时也授予了所有低级别的权限。例如，如果为某个用户授予了第 1 级别的权限，那么同时也授予了它包含的所有第 2 级别和第 3 级别的权限。

版本权限

表 4-1　版本权限

权限级别 1	权限级别 2	权限级别 3
管理版本 - 用户有权访问“版本”和“层次”菜单选项	**浏览版本** - 用户有权访问在节点访问组中获得了授权的任何版本	不适用
	创建版本 - 用户可以管理（更新/删除）属于他们的任何版本。用户有权访问“版本”菜单选项。 **注意：**创建版本的用户是所有者，除非具有“管理版本”权限的用户更改了所有者。	不适用
	管理层次 - 用户有权访问“层次”菜单选项。	**浏览层次** - 用户有权访问在节点访问组中获得了授权的任何层次。如果用户具有编辑节点访问权限或更高访问权限，那么他们有权访问“节点”菜单选项。

表 4-1 （续） 版本权限

权限级别 1	权限级别 2	权限级别 3
		创建层次 - 用户可以管理（更新/删除）属于他们的任何层次。用户有权访问“层次”菜单选项。用户可以禁用属于他们的任何层次的节点类型。 **注意：**创建层次的用户是所有者，除非具有“管理层次”权限的用户更改了所有者。

请求权限

表 4-2 请求权限

权限级别 1	权限级别 2	权限级别 3
管理请求 - 用户可以删除系统中尚未最终提交的任何请求。	**创建请求** - 用户可以查询系统中的任何请求，并且可以管理（更新/删除）他们作为所有者的任何请求。	不适用
工作流参与者 - 用户可以使用监管工作流模型参与请求。	不适用	不适用

查询权限

表 4-3 查询权限

权限级别 1	权限级别 2	权限级别 3
管理系统查询 – 用户有权访问系统查询和“查询”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	**管理用户查询** – 用户有权查看和运行用户查询和标准查询。用户无权访问标准查询的“查询”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	**运行查询** - 用户可以查看和运行任何标准查询。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。如果用户具有编辑节点访问权限或更高访问权限，那么他们有权访问“节点”菜单选项。
	管理标准查询 – 用户有权访问标准查询的“查询”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	不适用

比较权限

表 4-4　比较权限

权限级别 1	权限级别 2	权限级别 3
管理系统比较 – 用户有权访问系统比较和“比较”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	**管理用户比较** – 用户有权查看和运行用户比较和标准比较。用户无权访问标准比较的“比较”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	**运行比较** – 用户可以查看和运行任何标准比较。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。如果用户具有编辑节点访问权限或更高访问权限，那么他们有权访问“节点”菜单选项。
	管理标准比较 – 用户有权访问标准比较的“比较”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	不适用

导入权限

表 4-5　导入权限

权限级别 1	权限级别 2	权限级别 3
管理系统导入 – 用户有权访问系统导入和“导入”菜单选项。根据属性类别安全性，用户对属性选择器具有受限访问权限。	**管理用户导入** – 用户有权查看和运行用户导入和标准导入。用户无权访问标准导入的“导入”菜单选项。根据属性类别安全性，用户对属性选择器具有受限访问权限。	**运行导入** – 用户可以查看和运行任何标准导入。根据属性类别安全性，用户对属性选择器具有受限访问权限。
	管理标准导入 – 用户有权访问标准导入的“导入”菜单选项。根据属性类别安全性，用户对属性选择器具有受限访问权限。	不适用

混合器权限

表 4-6　混合器权限

权限级别 1	权限级别 2	权限级别 3
管理系统混合器 – 用户有权访问系统混合器和“混合器”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	**管理用户混合器** – 用户有权查看和运行用户混合器和标准混合器。用户无权访问标准混合器的“混合器”菜单选项。	**运行混合器** – 用户可以查看和运行任何标准混合器。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。
	管理标准混合器 – 用户有权访问标准混合器的“混合器”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	不适用

导出权限

表 4-7　导出权限

权限级别 1	权限级别 2	权限级别 3
管理系统导出 – 用户有权访问系统导出和“导出”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	**管理用户导出** – 用户有权查看和运行用户导出和集以及标准导出和集。用户无权访问标准导出和集的“导出”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限	**运行导出** – 用户可以查看和运行任何标准导出。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。
	管理标准导出 – 用户有权访问标准导出和集的“导出”菜单选项。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	不适用

脚本权限

表 4-8　脚本权限

权限级别 1	权限级别 2	权限级别 3
运行操作脚本 – 用户可以运行操作脚本。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	不适用	不适用

审核权限

表 4-9　审核用户事务权限

权限级别 1	权限级别 2	权限级别 3
审核用户事务 – 用户可以查询他们执行的任何事务。事务可以包括数据和元数据更改以及记录的操作，如登录和运行异步运算。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	不适用	不适用

表 4-10　审核数据事务权限

权限级别 1	权限级别 2	权限级别 3
审核数据事务 – 用户可以查询他们在权限或节点访问组中获得了其访问权限的数据对象的任何事务。事务可以包括用户执行的事务和其他用户进行的更改。对于节点级事务，用户可以查询某个节点及其所有后代的事务（“包含子节点”选项），前提是用户还对所有后代具有读取访问权限。根据节点访问组分配和属性类别安全性，用户对版本、层次、节点和属性选择器具有受限访问权限。	不适用	不适用

表 4-11　审核系统事务权限

权限级别 1	权限级别 2	权限级别 3
审核系统事务 – 用户可以查询他们执行的任何事务。事务可以包括数据和元数据更改以及记录的操作，如登录和运行异步运算。	不适用	不适用

应用程序权限

表 4-12　应用程序权限

权限级别 1	权限级别 2	权限级别 3
管理应用程序	**管理类别**	**浏览类别** – 用户有权访问在属性类别安全性中获得了其权限的任何属性类别。
	管理属性	**浏览属性** – 用户有权访问在属性类别安全性中获得了其权限的属性类别的所有属性。
		管理属性列表 – 用户可以管理属性定义的值和查找表列表。
	管理验证	不适用
	管理节点类型	不适用
	管理首选项	不适用

访问权限

表 4-13　访问权限

权限级别 1	权限级别 2	权限级别 3
管理访问	**管理用户** – 用户无法编辑或删除自己的用户配置文件。	不适用

表 4-13 （续） 访问权限

权限级别 1	权限级别 2	权限级别 3
	管理角色 – 用户无法编辑自己的角色分配。	不适用
	管理访问组 – 用户无法编辑自己的节点访问组分配。	不适用
	管理属性访问 – 用户无法编辑自己的属性类别分配。	不适用

用户角色

Oracle Data Relationship Management 通过角色为用户分配权限。每个用户角色都与一组权限相关联，权限提供对产品功能或数据的访问。可以向用户分配一个或多个角色，这样将授予他们所有角色的组合权限。如果向用户分配的两个角色具有冲突的访问级别，则向用户授予较高的访问级别。

Data Relationship Management 提供以下用户角色，下表标出了每个用户角色分配的权限：

表 4-14 用户角色 - 权限

权限			用户角色							
级别 1	级别 2	级别 3	访问管理员	匿名用户	应用程序管理员	数据创建者	数据管理员	交互式用户	工作流用户	监管用户
管理版本							X			
	浏览版本		X	X	X	X		X	X	X
	创建版本					X				
	管理层次						X			
		浏览层次	X	X	X	X		X	X	X
		创建层次				X				
管理请求							X			
	创建请求				X				X	
管理系统查询					X					
	管理用户查询					X	X	X		
		运行查询		X					X	
	管理标准查询						X			
管理系统比较					X					
	管理用户比较					X	X	X		

表 4-14 （续） 用户角色 - 权限

权限			用户角色							
级别 1	级别 2	级别 3	访问管理员	匿名用户	应用程序管理员	数据创建者	数据管理员	交互式用户	工作流用户	监管用户
		运行比较		X					X	
	管理标准比较						X			
管理系统导入					X					
	管理用户导入					X	X			
		运行导入								
	管理标准导入						X			
管理系统混合器					X					
	管理用户混合器					X	X			
		运行混合器								
	管理标准混合器						X			
管理系统导出					X					
	管理用户导出					X	X	X		
		运行导出		X					X	
	管理标准导出						X			
运行操作脚本					X	X	X	X		
审核用户事务			X		X	X	X	X	X	
审核数据事务					X	X	X	X		
审核系统事务			X		X					
管理应用程序					X					
	管理类别									
		浏览类别	X	X		X	X	X	X	X
	管理属性									
		浏览属性	X	X		X	X	X	X	X

表 4-14 （续） 用户角色 - 权限

权限			用户角色							
级别 1	级别 2	级别 3	访问管理员	匿名用户	应用程序管理员	数据创建者	数据管理员	交互式用户	工作流用户	监管用户
		管理属性列表					X			
	管理验证									
	管理节点类型				X					
	管理首选项									
管理访问										
	管理用户		X							
	管理角色		X							
	管理访问组		X							
	管理属性访问		X							
工作流参与者										X

Analytics 角色

可以组合使用 Oracle Data Relationship Management Analytics 角色来支持多个函数。例如，具有 Analytics 用户、监管管理员和数据管理员角色的用户将可以访问所有仪表板和管理控制台。具有访问管理员和应用程序管理员角色的用户将可以访问所有报表。

表 4-15 用于仪表板和管理控制台的 Analytics 角色

角色	仪表板和管理控制台					权限
	请求	模型	更改	增长	管理控制台	
Analytics 用户			X	X		浏览版本和层次
监管管理员	X	X				浏览版本和层次
应用程序管理员					X	不适用
数据管理员					X	不适用

表 4-16　用于报表的 Analytics 角色

角色	报表						
	用户角色分配	访问组成员	层次访问组分配	工作流访问组分配	对象访问组授权	用户登录活动	元数据对象使用
访问管理员	X	X		X	X	X	
应用程序管理员			X	X			X
数据管理员			X	X			

创建用户

创建用户时，要定义唯一名称并分配角色。如果未向用户分配“数据管理员”角色，则可以向用户分配节点访问组和属性类别以控制其对数据的访问。

> **注：**
>
> 用户 ID 上的 @@ 前缀指示纯内部用户。不能使用此前缀创建用户。其他 @@ 用户包括 @@SYSTEM 和 @@STANDARD。

要创建用户：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**用户**。
3. 输入唯一用户名以及用户的全名。

> **注：**
>
> “部门”、“电话”和“电子邮件地址”都是可选的。数据监管工作流用户必须配置电子邮件地址以接收电子邮件通知。

4. 如果为 Oracle Data Relationship Management 应用程序启用了混合身份验证，请为用户选择身份验证方法。
   - **内部** - 仅在 Data Relationship Management 内部对用户进行身份验证。
   - **CSS (外部)** - 通过 Oracle Hyperion Shared Services 在外部对用户进行身份验证。
5. **可选：**从以下选项中进行选择：
   - **密码未到期** - 忽略 PasswordDuration 系统首选项设置。
   - **登录会话未到期** - 忽略 IdleTime 系统首选项设置。

> **注：**
>
> 如果选择此选项，则允许的最大空闲时间为 24 小时。空闲时间达到 24 小时后，登录会话将到期。

- **用户从锁定度量中免除** - 对此用户忽略锁定限制。

6. 在**角色**选项卡上，从**可用**列表中选择要分配给用户的角色。使用箭头将角色移到**已选中**列表中。

> **注：**
>
> 有关角色的其他信息，请参阅“用户角色”。

7. 在**节点访问组**选项卡上，从**可用**列表中选择要分配给用户的组。使用箭头将组移到**已选中**列表中。
8. 在**属性类别**选项卡上，从**可用**列表中选择要分配给用户的类别。使用箭头将类别移动到**已选中**列表中
9. 对选定列表中的每个类别执行以下操作：
   a. 单击**操作**列中的 并设置用户对类别的访问权限（“读取”或“编辑”）。
   b. 选择**操作**列中的 以保存更改。
10. 单击 。

    随即显示“更改密码”对话框。
11. 输入用户密码。
12. 再次输入密码。
13. **可选：**选择**用户下次登录时必须更改密码**，以要求用户在下一次登录时更改密码。
14. 单击**确定**。

用户身份验证

Oracle Data Relationship Management 支持使用存储的密码信息通过应用程序进行本地身份验证的用户，或通过外部用户目录进行身份验证的用户。每个 Data Relationship Management 应用程序都配置为支持一种或两种类型的用户。

您可以在 Data Relationship Management 控制台的“身份验证设置”选项卡上设置应用程序身份验证。有关详细信息，请参阅《*Oracle Data Relationship Management* 安装指南》。

为以下系统首选项定义的值确定用户密码的特性，以及内部身份验证用户的密码何时过期。

- PasswordPolicyEnabled - 如果启用，则密码必须包含下列元素中的三种元素：
  - 大写字母
  - 小写字母

  - 数字
  - 特殊字符
- PasswordMaxLength - 确定密码的最大字符长度。
- PasswordMinLength - 确定密码的最小字符长度。
- PasswordDuration - 确定密码的有效天数。
- PasswordWarningPeriod - 指示密码过期日期前 (-) 或后 (+) 多少天警告用户更改其密码，超过这段期限将禁止他们登录。负值（例如 -3）指示在密码过期前 3 天内用户登录时警告用户。正值（例如 5）指示在密码过期后 5 天内用户登录时警告用户。五天后，如果不更改密码，用户将无法登录。

> **注：**
>
> 对 PasswordDuration 和 PasswordWarningPeriod 值所做的更改在下次更改密码后才会对用户产生影响。例如，如果 PasswordDuration 设置为 30 天，且 User1 的密码是在 26 天前更改的，那么密码将在 4 天后过期。如果现在将 PasswordDuration 值更改为 60 天，那么 User1 的密码仍将在 4 天后过期。用户更改密码之后，新密码将在 60 天后过期。

修改用户

您可以更改用户密码、锁定或解锁用户，或者更改角色、组或类别分配。

更改密码

更改用户密码：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**用户**。
3. 选择用户并单击 。
4. 单击 。
5. 为用户输入一个新的密码。
6. 再次输入密码。
7. **可选：**选择**用户下次登录时必须更改密码**，以要求用户在下一次登录时更改密码。
8. 单击**确定**。

锁定用户

可以锁定用户以防止其访问 Oracle Data Relationship Management 应用程序。锁定用户时，可以为锁定提供自定义原因。用户尝试登录应用程序时，系统会向该用户显示此原因。

要锁定用户：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**用户**。

3. 选择用户并单击  。
4. 单击  。
5. 输入锁定原因。
6. 单击**确定**。

解锁用户

解锁处于锁定状态的用户将允许他们访问应用程序。

要解锁用户：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**用户**。
3. 选择用户并单击  。
4. 单击  。
5. 单击**确定**。

更改用户角色和分配

要更改用户角色和分配：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**用户**。
3. 选择用户并单击  。
4. 在**角色**选项卡上，从**可用**列表中选择要分配给用户的角色。使用箭头将角色移到**已选中**列表中。
5. 在**节点访问组**选项卡上，从**可用**列表中选择要分配给用户的组。使用箭头将组移到**已选中**列表中。
6. 在**属性类别**选项卡上，从**可用**列表中选择要分配给用户的类别。使用箭头将类别移到**已选中**列表中。
7. 对选定列表中的每个类别执行以下操作：
   a. 单击  并设置用户对该类别的访问权限（“读取”或“编辑”）。
   b. 选择  保存更改。
8. 单击  。

删除用户

不再有效的用户可以从应用程序中删除。删除某个用户时，同时也会删除与该用户关联的所有用户级元数据对象。这些元数据对象包括查询、比较、导入、混合器、导出和集。

要删除用户：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**用户**。
3. 选择用户并单击 。
4. 单击**删除此项**以确认删除。

查看用户的登录状态

您可以查看每个用户的登录统计数据和信息：

- 用户上一次有效登录的日期和时间
- 无效登录尝试的次数
- 用户是否已锁定
- 锁定用户的日期和时间
- 锁定的原因

要查看用户的登录状态：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**用户**。
3. 选择用户并单击  。
4. 选择**登录状态**选项卡。

系统定义的用户

Oracle Data Relationship Management 应用程序包括在创建应用程序存储库时添加的四个默认用户。

- **ADMIN** - 应用程序的默认管理用户。此用户的密码最初是在创建存储库过程中配置的。
- **@PROCESS** - 一个内部用户，设置用来处理服务器组件之间的进程间通信。不能在 Web 客户端访问或配置此用户。每次启动应用程序引擎时，都会为此用户记录事务。
- **@STANDARD** - 一个内部用户，设置用来管理标准对象访问组中的用户元数据对象。不能在 Web 客户端访问或配置此用户。
- **@SYSTEM** - 一个内部用户，设置用来管理系统对象访问组中的用户元数据对象。不能在 Web 客户端访问或配置此用户。

通用用户设置

通过通用用户设置功能，可以使用 Oracle Hyperion Shared Services 将用户和组设置到 Oracle Data Relationship Management 应用程序。此配置允许在通用位置以及其他 Oracle EPM 应用程序中设置 Data Relationship Management 用户。另外，有了通用用户设置，不再需要在 Data Relationship Management 应用程序中单独设置用户。设置信息可以按需或按调度从 Shared Services 同步到 Data Relationship Management。

当执行同步时，将在 Data Relationship Management 中执行以下操作：

- 添加或更新用户
  - 用户名

  - 全名
  - 电子邮件地址
- 向用户分配角色
- 将用户分配到节点访问组
- 将用户分配到属性类别
- 删除用户角色（如果已在 Shared Services 中取消设置）

启用通用用户设置后，所有外部 Data Relationship Management 用户及其角色将在 Shared Services 中管理，不能在 Data Relationship Management 中管理。

先决条件

通用用户设置在 Oracle Data Relationship Management 中默认情况下处于禁用状态，仅应在完成以下先决条件步骤后才启用：

1. 在 Oracle Hyperion Shared Services 中添加 Data Relationship Management 用户角色 - 请参阅《*Oracle Data Relationship Management* 安装指南》中的“为 Shared Services 数据库配置 Data Relationship Management 用户角色”。
2. 将 Data Relationship Management 应用程序注册到 Shared Services - 请参阅《*Oracle Data Relationship Management* 安装指南》中的“配置 EPM 注册表设置”。
3. 启用通用用户设置 - 请参阅《*Oracle Data Relationship Management* 安装指南》中的“配置通用用户设置”。

设置用户和组

可以使用通用用户设置为 Oracle Data Relationship Management 应用程序设置可在 Oracle Hyperion Shared Services 中访问的任何用户或组。可以为 Data Relationship Management 应用程序设置组（包含组和/或用户）和单个用户。在 Data Relationship Management 中运行同步任务时，将同步在 Shared Services 中为 Data Relationship Management 应用程序设置的用户和组。可以将用户单独设置到多个注册的 Data Relationship Management 应用程序。

请参阅《Oracle EPM System 用户安全管理指南》中的"设置用户和组”。

同步 Data Relationship Management 用户和组成员身份

将用户和组更改从 Oracle Hyperion Shared Services 完整同步到 Oracle Data Relationship Management 应用程序的操作可以手动执行，也可以调度在后台运行。同步在 Data Relationship Management 应用程序中创建或更新用户，并更新配置为在外部管理的节点访问组或属性类别中的组成员身份。

同步的结果显示创建和更新了多少用户、更新了多少节点访问组，以及更新了多少属性类别。运行同步时生成的错误和警告消息列表也会显示。您可以将结果复制并粘贴到外部编辑器中进一步查看或使用。

手动同步

在 Oracle Data Relationship Management 中，当启用了通用用户设置时，具有“访问管理员”角色的用户可以手动同步在 Oracle Hyperion Shared Services 中管理的用户和组。系统会显示该作业的结果，也可以在“审核”任务的“作业”页面上查看该作业的结果。

要手动同步用户和组：

1. 在主页上，选择**管理**。
2. 从工具栏中，选择  （通用用户设置同步）。

调度的同步

在 Oracle Data Relationship Management 中，启用通用用户设置后，可以调度同步在指定的时间每隔 24 小时在后台运行一次。可以通过导航到“审核”任务的“作业”页面上的相应作业来查看调度作业的结果。

- 有关查看作业的信息，请参阅《*Oracle Data Relationship Management* 用户指南》中的“查看作业历史记录”。
- 有关调度同步的信息，请参阅《*Oracle Data Relationship Management* 安装指南》中的“配置通用用户设置”。

部分同步

在以下情形中，会针对在 Oracle Hyperion Shared Services 中管理的用户和组自动执行部分实时同步：

- 用户登录 - 在创建会话之前，会自动同步正在进行身份验证的单个用户的设置信息。
- 节点访问组成员身份 - 当保存单个节点访问组时，会自动同步该组的用户成员身份。
- 属性类别成员身份 - 当保存单个属性类别时，会自动同步该类别的用户成员身份。

5

管理节点访问组

Oracle Data Relationship Management 使用节点访问组来精细控制用户对层次节点及其属性的访问。您可以为组授予访问某个 Data Relationship Management 版本中部分层次的特定节点的权限，然后将用户分配给这样的组。节点访问组使用继承为明确分配了访问级别的层次节点的后代节点分配访问权限，这些后代节点的访问权限与其祖先类似。可以在低级别的节点上覆盖此访问级别或锁定此访问级别以防止覆盖。

通常，节点访问组代表组织的功能区域，并且用户可能需要分配到多个组。如果分配的访问级别冲突，则使用最高的安全级别。

有两种类型的节点访问组。组类型控制可分配给该组用户的数据访问的类型。每个节点访问组只能属于一种组类型。

- 交互式 - 基于分配的访问级别，用户具有直接的访问权限来浏览、搜索和修改数据
- 工作流 - 基于分配的访问级别，用户具有受限的访问权限来使用监管工作流浏览、搜索和修改数据

表 5-1　交互式组类型 - 节点访问级别

级别	说明	示例用法
读取	允许只读访问 – 不允许更改	查看和生成报表
受限插入	允许插入用户对其具有（至少）全局插入权限的节点。	插入
编辑	允许编辑属性值	编辑
插入	允许插入、移动或删除节点	编辑、插入、复制、移动、删除
停用	允许停用和重新激活节点	编辑、插入、移动、删除、停用、重新激活
添加	允许添加或删除节点	编辑、插入、复制、移动、删除、停用、重新激活、添加、删除

请记住以下信息：

- 访问级别是累积的；分配“编辑”访问级别意味着授予了“只读”和“受限插入”访问级别。分配“添加”访问级别意味着授予了所有其他访问级别。
- 节点访问组安全仅应用于层次级别。节点访问组不控制对节点（例如孤立节点）全局列表的访问。
- 枝节点和叶节点分别分配访问级别，以便可以为每类节点定义不同的访问级别。用户需要能够维护层次的整体结构但是不能编辑叶节点的任何属性时，或者用户需要能够将叶节点插入现有的整体结构但是不能重新组织结构本身时，此功能非常有用。
- 节点访问组只能由具有“访问管理员”角色的用户定义。
- 节点访问组使用本地继承将访问权限分配给相关节点。节点访问组可以定义为全局，以便根据分配给控制层次的访问级别使用全局继承。

- 可以创建全局节点访问组，并且必须为每个版本定义一个控制层次。这通过将受控节点访问组分配给层次来实现。有关详细信息，请参阅《*Oracle Data Relationship Management* 用户指南》。
- 交互式和工作流节点访问组以不同的方式处理层次中节点的可见性。如果交互式访问组对层次中任一节点具有访问权限，则该组向用户提供查看整个层次的可见性。相比之下，工作流访问组仅向用户提供有限的可见性，用户只能看到为其分配有访问权限的层次中的节点。对于这两种组类型，组成员均无法查看未为其分配访问权限的层次。

工作流组类型节点访问级别

具有监管用户角色的用户可使用工作流节点访问级别来确定其对数据的访问权限。

表 5-2　工作流组类型 - 节点访问级别

级别	说明
通知	允许通知节点的更改请求
提交	允许将节点作为更改请求的一部分进行提交
批准	允许将节点作为更改请求的一部分进行批准
扩充	允许将节点作为更改请求的一部分进行扩充
最终提交	允许将对节点的更改最终提交到 Oracle Data Relationship Management

层次访问权限的工作流节点访问级别是累积的，但也可按工作流阶段筛选。

表 5-3　按层次访问权限划分的工作流节点访问级别

层次访问权限	阶段访问权限			
访问权限	**提交**	**批准**	**扩充**	**最终提交**
通知	通知	通知	通知	通知
提交	提交	通知	通知	通知
批准	提交	批准	通知	通知
扩充	提交	批准	扩充	通知
最终提交	提交	批准	扩充	最终提交

创建节点访问组

要创建节点访问组：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**节点访问组**。
3. 输入组的名称、标签和说明。

> **注：**
>
> 节点访问组将分配给 Custom 命名空间。组的全限定名称必须唯一。输入名称后会自动填写“标签”字段。节点访问组标签是为应用程序管理以外的所有功能而显示的一个便于用户理解的描述短语。为方便起见，多个节点访问组可以具有相同的标签。

4. 为节点访问组选择**组类型**。
   - **交互式** - 要使用交互式访问级别，请参阅“交互式节点访问级别”。
   - **工作流** - 在提交、扩充、批准、最终提交和请求通知上下文中对版本、层次和节点进行面向工作流的访问。请参阅“工作流节点访问级别”。
5. **可选**：选择**全局**将组设置为全局节点访问组。

> **注：**
>
> 全局节点访问组必须在将使用该组的每个版本中定义一个控制层次。创建组后，可以将其分配给每个版本中的单个层次作为受控节点访问组。

6. 如果使用通用用户设置，请在**外部组**中，选择要在 Oracle Hyperion Shared Services 中设置到 Oracle Data Relationship Management 应用程序的用户组。当在 Shared Services 中执行同步时，此外部组中的用户将获得该节点访问组的成员身份。
7. 从**可用**列表中选择用户以分配给组。使用箭头将表移到**已选中**列表中。
8. 单击 。

编辑节点访问组

要编辑节点访问组：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**节点访问组**。
3. 选择组并单击 。
4. 从**可用**列表中选择用户以分配给组。使用箭头将表移到**已选中**列表中。
5. 单击 。

删除节点访问组

要删除节点访问组：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**节点访问组**。
3. 选择组并单击 。
4. 单击**删除此项**以确认删除。

> **注：**
>
> 删除节点访问组将从用户以及从所有层次节点删除组的分配。

分配节点访问组安全性

节点访问组安全性由具有数据管理员角色的用户应用于数据。

> **注：**
>
> 在分配节点访问组安全性之前，请确保创建了适当的节点访问组并且为这些组分配了适当的用户。

要设置节点访问组安全性：

1. 打开一个版本和层次，并选择一个节点。
2. 从**节点**中，依次选择**分配**和**节点访问**。
3. 在“属性网格”中，选择“叶访问”或“枝访问”类别。
4. 为每个节点访问组分配访问级别。

   为每个节点访问组可分配的访问级别根据其组类型（交互式或工作流）而定。
5. 单击**保存**。

> **注：**
>
> 必须将对至少一个叶和一个先行枝节点的枝和叶工作流 NAG（Node Access Group，节点访问组）访问指定为“无”以外的值，才能可视化要在 DRG 节点选择器中选择的节点。通常，在层次中的顶级节点上设置枝 WNAG（Workflow Node Access Group，工作流节点访问组）访问。

6
管理对象访问组

Oracle Data Relationship Management 中的对象访问组确定用户有权访问的元数据对象，包括导出、集、导入、混合器、比较、查询、版本变量和外部连接。

表 6-1　对象访问组类型

对象访问组类型	说明	权限
用户	每个用户对其个人元数据对象都具有一个核心对象访问组。	用户对自己的对象访问组具有运行和管理权限。
标准	名为“标准”的核心对象访问组适用于所有公共对象。	所有用户都对“标准”对象访问组中的对象具有隐式运行权限。 仅具有“管理标准 [对象]”角色权限的用户对“标准”对象访问组具有管理权限。
系统	名为“系统”的核心对象访问组适用于所有系统操作/集成对象。	仅具有“数据管理员”或“应用程序管理员”角色的用户对“系统”对象访问组具有管理权限。
自定义	自定义对象访问组	仅具有“访问管理员”角色的用户可以创建、编辑或删除自定义对象访问组。具有运行权限的用户可以执行组中的对象。

自定义对象访问组针对一部分用户元数据对象（查询、比较、导入、混合器、导出和集）提供了一组特定的用户访问权限。对象访问组定义用户和节点访问组列表，并为每个用户和节点访问组设置权限级别（运行或管理）。在创建元数据对象时会将其分配给对象访问组，随后可以将这些对象复制或移动到其他组。

- 运行 - 用户可以运行组中的对象，但是不能编辑对象或保存对对象的更改
- 管理 - 用户可以创建、编辑或删除组中的对象，以及运行这些对象

下面是对象访问组的使用准则：

- 对象访问组允许用户直接成为组中的成员，或通过其节点访问组分配成为组中的成员。二者都不是必需的。
- 用户和节点访问组可以分配给多个对象访问组。
- 对象访问组中的每个用户都分配有管理权限或者运行权限。
- 对象访问组中的用户权限分配可以覆盖用户的角色安全权限。例如，在对象访问组中具有管理权限的交互式用户角色可以创建或修改对象访问组中的对象。
- 核心对象访问组（如“用户”、“标准”和“系统”）基于用户是否存在及其角色分配进行隐式管理。
- 当保存或复制某个用户元数据对象时，用户必须将该对象分配到用户具有管理权限的对象访问组。
- 一个用户元数据对象只能分配到一个对象访问组。

- “数据管理员”角色对核心“标准”对象访问组具有隐式管理权限，并且可以显式分配到自定义对象访问组。
- “应用程序管理员”角色用户对所有标准、系统和自定义对象访问组都具有隐式管理权限。这些用户要能够迁移所有对象访问组的元数据对象。

创建对象访问组

要创建自定义对象访问组：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**对象访问组**。
3. 输入组名称。说明是可选的。
4. 在**用户**选项卡上，从**可用**列表中选择要分配给该组的用户。使用箭头将表移到**已选中**列表中。

> **注：**
>
> 默认情况下，会授予每个用户运行访问权限。要更改用户的访问权限，请单击 。然后，在**访问**中选择**管理**。

5. 在**节点访问组**选项卡上，从**可用**列表中选择要分配给该组的节点访问组。使用箭头将节点访问组移到**已选中**列表中。

> **注：**
>
> 默认情况下，会授予每个节点访问组运行访问权限。要更改组的访问权限，请单击 。然后，在**访问**中选择**管理**。

6. 单击 。

编辑对象访问组

要编辑自定义对象访问组：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**对象访问组**。
3. 选择组，然后单击 。
4. 在**用户**和**节点访问组**选项卡上，更改选择的用户和组以及访问权限。
5. 单击 。

删除对象访问组

要删除对象访问组：

1. 在主页上，选择**管理**。
2. 在**安全性**下，展开**对象访问组**。
3. 选择组，然后单击  。
4. 单击**删除此对象访问组**以确认删除。

> **注意：**
>
> 当删除对象访问组时，还将删除为其分配的所有元数据对象。此操作不能撤消

7

管理域

域用于管理来自同一 Oracle Data Relationship Management 应用程序内不同源的多组节点的引用完整性。域是相同类型的一组已注册节点，用于在相同应用程序的不同版本中对这些节点进行一致的管理。域针对以下方面提供简单方法：

- 限定节点名称以确保唯一性
- 跨版本共享识别属性
- 限制特定类型的更改，例如重命名、升级、降级和删除节点
- 分配验证以确保业务规则的一致性（不管版本为何）

域节点是某个版本中的全局节点，并具有某个域的成员身份。域节点在分配为成员后，便无法重命名和从域中删除。不管域分配如何，域节点都必须有一个唯一名称。域节点的名称可以代表节点的自然标识符，或者可以使用前缀或后缀进行限定，以便在相同版本中与不同域的节点一起使用时确保引用完整性。域节点说明和非活动状态/日期由其存在于的任何版本中的域节点共享。

创建域

要创建域：

1. 从主页中选择**管理**。
2. 从**新建**中选择**域**。
3. 输入以下信息：
   - **名称**
   - **说明**（可选）
   - **限定符**（可选）– 用于完全限定节点名称的文本。任何两个域都不能使用相同的限定符文本。选择**前缀**或**后缀**以指示限定符的位置。

     > **注：**
     >
     > 为域分配了节点后，便无法更改限定符文本。

   - **分隔符**（可选）– 一个用于分隔域限定符文本和节点名称的可选字符。

     > **注：**
     >
     > 为域分配了节点后，便无法更改分隔符。

   - **允许删除节点** – 如果希望允许用户从版本中删除节点，则选择此项。
   - **允许叶编辑** – 如果希望允许用户更改域中节点的叶系统属性值，则选择此项。

4. 从**可用验证**列表中，选择要为域成员强制执行的节点级别验证，然后将其移到**选定验证**列表。

> **注：**
>
> 域级别验证分配会覆盖在节点设置的或者从祖先节点、层次或版本级别分配继承的相同验证的分配值。

5. 单击 。

编辑域

域可以在创建后进行编辑，但有两种情况例外：

- 名称不能更改
- 在将节点分配给域后，限定符和分隔符不能更改

要编辑域：

1. 从主页中选择**管理**。
2. 选择域并单击 。
3. 对域进行更改，然后单击 。

删除域

可以删除域。删除域时也会一并删除域节点记录。

> **注：**
>
> 如果删除分配了节点的域，那么分配给该域的所有节点都会恢复为非域节点。

要删除域：

1. 从主页中选择**管理**。
2. 选择域并单击 。
3. 选择**删除此域**。

8

管理属性类别

另请参阅：

- 属性类别
- 创建属性类别
- 编辑属性类别
- 删除属性类别

属性类别

属性类别允许对 Oracle Data Relationship Management 属性进行分组，用于控制属性组的安全性权限分配。默认情况下可用的核心属性只能位于单个属性类别中。应用程序管理员创建的自定义属性可以与多个属性类别相关联。

Data Relationship Management 包含下表中介绍的核心属性类别。

表 8-1　属性类别

类别	说明
系统	与节点的基本标识特性相关的属性，例如 ID、名称和说明。 可以对该类别进行的唯一一项更改是为各个用户分配只读标志。具有读取访问权限的用户无法编辑值，但是可以查看值。无法为该类别分配属性。
共享信息	提供有关哪些节点是主节点/共享节点的信息以及相关共享节点列表，并标识主节点是否缺失。 仅在通过系统首选项启用了“共享的节点”时才显示此类别。 **注意：**此类别中的所有属性都为只读属性。
统计	提供有关节点的统计信息（如子代数量和同级数量）的属性 **注意：**此类别中的所有属性都为只读属性。
验证	为节点分配的验证 - 每个验证一个属性
叶访问	节点的节点安全组及其叶访问级别 - 每个组一个属性
枝访问	节点的节点安全组及其枝访问级别 - 每个组一个属性

> **注：**
>
> 并非所有属性类别对所有用户均可见，因为用户可能只能访问特定的类别，而且节点类型可能已经过筛选。“验证”、“叶访问权限”和“枝访问权限”仅可供分配了数据管理员角色的用户使用，并且仅在分配了验证或节点访问组安全性时才可访问。

创建属性类别

要创建属性类别：

1. 从主页中选择**管理**。
2. 从**新建**中选择**属性类别**。
3. 输入属性类别的名称和说明。
4. 如果使用通用用户设置，请从**外部组 - 编辑**和**外部组 - 读取**中，选择要在 Oracle Hyperion Shared Services 中设置到 Oracle Data Relationship Management 应用程序的用户组。当在 Shared Services 中执行同步时，这些外部组中的用户将获得该属性类别的成员身份，并具有指定级别的访问权限（编辑或读取）。
5. 在**属性**选项卡上，从**可用**列表中选择要分配给属性类别的属性，然后使用箭头将属性移到**已选中**列表中。

   > **注：**
   >
   > 您可以使用 **Ctrl+单击**或 **Shift+单击**选择多个属性。双击某个属性可选择或取消选择该属性。

6. 使用箭头重新排列选定的属性，或者单击  按字母顺序排列选定的属性。
7. 在**用户**选项卡上，从**可用**列表中选择要分配给属性类别的用户，然后使用箭头将用户移到**已选中**列表中。
8. 在选定的列表中选择某个用户对应的行，然后单击**操作**列中的  。
9. 从**访问**列中，选择“读取”或“编辑”为该用户分配对属性类别的访问级别。
10. 在**操作**列中单击  保存更改，或单击  放弃更改。
11. 单击  。

编辑属性类别

要编辑属性类别：

1. 从主页中选择**管理**。
2. 选择属性类别，然后单击  。
3. 在**属性**选项卡上，从**可用**列表中选择要分配给属性类别的属性，然后使用箭头将属性移到**已选中**列表中。

> **注：**
>
> 您可以使用 **Ctrl+单击**或 **Shift+单击**选择多个属性。双击某个属性可选择或取消选择该属性。

4. 使用箭头重新排列选定的属性，或者单击  按字母顺序排列选定的属性。
5. 在**用户**选项卡上，从**可用**列表中选择要分配给属性类别的用户，然后使用箭头将用户移到**已选中**列表中。
6. 在选定的列表中选择某个用户对应的行，然后单击**操作**列中的  。
7. 从**访问**列中，选择“读取”或“编辑”为该用户分配对属性类别的访问级别。
8. 在**操作**列中单击  保存更改，或单击  放弃更改。
9. 单击  。

删除属性类别

要删除属性类别：

1. 从主页中选择**管理**。
2. 在**元数据**下，展开**属性类别**。
3. 选择属性类别，然后单击  。
4. 选择**删除此项**以确认删除。

> **注：**
>
> 删除属性类别不会导致删除与类别关联的属性。这些属性在应用程序中仍然可用。

9

管理属性定义

属性定义用于管理 Oracle Data Relationship Management 中版本、层次和节点的属性。属性可以存储各种不同的数据类型，包括文本、数字、日期和其他数据对象的引用。属性可以存储显式值，使用继承自动将值分配给后代节点，或者根据公式或查找表进行计算。属性类别可用于将属性分组和组织成相关的集合，以简化它们的使用并控制用户的访问。

默认提供的系统定义的属性用于标准的产品功能。用户定义的属性定义可以由应用程序管理员创建，以管理满足业务或系统集成要求所需的其他属性。

Data Relationship Management 中的属性定义可以来自多种源。例如，属性可以是：

- Data Relationship Management 中系统定义的属性
- 应用程序管理员创建的用户定义的属性
- 从与其他 Oracle 产品配合使用的应用程序模板加载的属性
- 使用迁移实用程序从其他 Data Relationship Management 应用程序或环境加载的属性

命名空间

在属性定义中使用命名空间来避免以下冲突：来自不同源的属性具有类似的名称，但出于数据完整性目的需要保持独立。使用命名空间前缀命名惯例区分属性名称。

表 9-1　使用命名空间的属性定义示例

字段	示例
全限定名称	Custom.AccountType
命名空间	Custom
名称	AccountType
标签	AccountType

Data Relationship Management 中存在应用于命名空间的特殊规则，用于确保不会发生冲突：

- 系统定义的属性使用 "Core" 命名空间。
- 用户定义的属性使用 "Custom" 命名空间。
- 其他命名空间则保留，以供其他 Oracle 产品的 Data Relationship Management 应用程序模板使用。

数据类型

下表中介绍了属性数据类型。

表 9-2　属性数据类型

属性数据类型	说明
关联组	关联的节点组。指向多个节点。节点回指至 Associated Group 节点，并互指。类比：兄弟会。 **注意：**此数据类型应仅用于全局节点级别属性。 **警告：**通过导入加载的关联节点属性可能无法正确指向所有其他节点，因为根据导入节点的顺序，这些其他节点可能尚未存在于版本中。
一个关联节点	一个关联的节点。指向单个其他节点。指向的节点回指至“一个关联节点”节点。类比：婚姻。 **注意：**此数据类型应仅用于全局节点级别属性。 **警告：**通过导入加载的关联节点属性可能无法正确指向所有其他节点，因为根据导入节点的顺序，这些其他节点可能尚未存在于版本中。
多个关联节点	关联的节点列表。指向多个节点。指向的节点回指至“多个关联节点”节点，但不互指。类比：朋友。 **注意：**此数据类型应仅用于全局节点级别属性。 **警告：**通过导入加载的关联节点属性可能无法正确指向所有其他节点，因为根据导入节点的顺序，这些其他节点可能尚未存在于版本中。
布尔值	True 或 False
日期	日期值的格式不因文化而异。这样将出现可预测的响应，并且可以执行操作以重新设置结果格式（如果需要）。 **警告：**必须以英语（美国）格式输入默认值、最大值和最小值。
日期/时间	日期和时间值的格式不因文化而异。这样将出现可预测的响应，并且可以执行操作以重新设置结果格式（如果需要）。 **警告：**必须以英语（美国）格式输入默认值、最大值和最小值。
浮点型	浮点值根据与用户会话关联的区域设置进行格式设置。 **注意：**如果未定义默认值，则导出时会针对值输出 0。
格式化备注	格式化备注 - 保留文本的所有格式设置（空格、制表符、新行等等）。还允许将超链接文本包括在格式化备注中。有关为超链接设置 URL 格式的详细信息，请参阅 Hyperlink 数据类型。 **注意：**在属性值中同时使用文本和超链接时，不会隐藏非 URL 文本。
全局节点	指向版本中的某个节点；当分配值后，其仅在属性网格的值字段中显示节点名称

表 9-2 （续） 属性数据类型

属性数据类型	说明
组	使用逗号分隔的项目列表
层次	指向层次
层次组	指向层次组。 层次组属性允许根据查看层次时所在的上下文以多种方式对层次进行分组。您可以根据用途在相同的版本中以不同的方式对层次进行分组。
超链接	允许 URL 文本的超链接功能。多个 URL 输入可以使用不带空格的回车换行符（CRLF 或 0x0D0A）进行分隔。输入的 URL 显示为可导航的超链接。仅显示已解析的、分隔的 URL 或已设置格式的 URL。URL 应遵循以下格式： [url=*http_URL*]*URL_Title*[/url] 其中 *http_URL* 指定超链接文本，*URL_Title* 指定显示给用户的文本。 例如，此标记示例：[url=http://support.oracle.com]Oracle Support[/url] 在属性网格中会显示为 Oracle 支持。
整数	整数值 如果未定义默认值，则导出时会针对值输出 0。
叶节点	指向层次中的叶节点。分配值后，其将在属性网格的值字段中显示层次名称和节点名称。
枝节点	指向层次中的枝节点。分配值后，其将在属性网格的值字段中显示层次名称和节点名称。
列表组	项目的检查表。可以从列表中选择多个项目。
备注	备注字段 - 不保存格式设置，并且数据合并成一行文本。备注中也可以有超链接。有关为超链接设置 URL 格式的详细信息，请参阅 Hyperlink 数据类型。 **注意：**在属性值中同时使用文本和超链接时，不会隐藏非 URL 文本。
多节点	指向多个节点
节点	指向层次中的某个节点；分配值后，其将在属性网格的值字段中显示层次名称和节点名称。
节点属性	指向节点的属性
属性	指向属性
范围列表	定义值的范围；仅接受整数值
排序	用于排序的整数值
排序属性	指向排序属性
标准查询	指向标准查询
字符串	字符串值

表 9-2 （续） 属性数据类型

属性数据类型	说明
时间	时间值的格式不因文化而异。这样将出现可预测的响应，并且可以执行操作以重新设置结果格式（如果需要）。 **警告：**必须以英语（美国）格式输入默认值、最大值和最小值。
版本	指向版本

外部查找

外部查找属性访问外部数据源以获得其可选值列表。使用外部操作访问外部数据源。外部查找属性类型允许从 Oracle 或 SQL Server 数据库返回记录集。使用外部查找的结果从外部值列表中选择项以用作属性值，或者使用外部源中的数据计算请求项属性值。可以在 Data Relationship Management 和 Data Relationship Governance 中访问属性列表的外部查找。

创建属性

要创建属性定义：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**属性定义**。
3. 输入属性的名称。

> **注：**
>
> 属性将分配给 Custom 命名空间。输入名称后，会自动填写“全限定名称”和“标签”字段。属性的全限定名称必须唯一。属性标签是为应用程序管理以外的所有功能中的属性定义而显示的一个便于用户理解的描述短语。只要不位于相同的命名空间，多个属性可以具有相同的标签。属性“说明”是显示在属性编辑器底部的可选长描述短语。

4. 定义属性的参数：

> **注：**
>
> 并非下面所有的参数都显示。显示的参数取决于选定的数据类型。

- **数据类型** - 请参阅“属性数据类型”

  可以通过选择以下某一数据类型限制向用户显示的节点列表：关联组、一个关联节点、多个关联节点、全局节点、叶节点、枝节点、多节点或节点。选择数据类型后，将显示**限制**选项卡。

- **属性级别** - 属性定义的级别：

  - **本地节点** - 针对某个特定层次中的节点来管理属性值，并且仅能在该级别访问属性值。
  - **全局节点** - 针对某个版本中的节点来管理属性值，但属性值也可在本地节点级别进行访问。
  - **层次** - 按层次管理属性值，但属性值也可在本地节点级别进行访问。
  - **版本** - 按版本管理属性值，但属性值也可在全局或本地节点级别进行访问。

> **注：**
>
> 如果要定义全局节点继承属性，则必须为该全局属性定义控制层次。在主页的“层次”选项卡上通过将受控属性分配给层次来执行此操作。

- **属性类型**
  - 定义 - 值由用户定义并存储。
  - 查找 - 基于其他属性和查找表进行查找。
  - 派生 - 使用派生程序类进行计算。

> **注：**
>
> 使用脚本派生程序类的派生属性可用于版本、层次和节点属性。公式派生程序类只可用于全局或本地节点属性。

  - 外部查找 - 使用外部数据源的查找

> **注：**
>
> 实时从外部数据源检索值。如果返回多个值，必须为属性选择特定值。

- **默认值** - 属性的默认值
- **域** - 对于数据类型为“节点”、“枝节点”、“叶节点”、“多节点”、“一个关联节点”、“多个关联节点”或“关联组”（所有都代表存储为值的一个节点或多个节点）的所有属性，都有一个“域”下拉列表。该下拉列表包含系统中定义的所有域，您可以选择一个现有域（可选）。
- **列宽** - 属性类型为“定义”时固定宽度列的宽度。
- **最小值/长度** - 属性基于数据类型的值或长度。
- **最大值/长度** - 属性基于数据类型的值或长度。

5. 从以下选项中选择：
   - **继承** - 将属性定义为继承

> **注：**
>
> 此选项对“派生”属性类型没有影响，除非是以下特殊情况 - 使用了属性派生程序（如 AncestorProp 或 DualAncestorProp）并且属性是全局属性。在此类情况下，尽管属性实际上不继承值，但启用“继承”选项允许指定控制层次。

- **可覆盖** – 允许在属性网格中覆盖属性。

> **注：**
>
> 仅针对“派生”属性类型启用此选项。

- **列表** – 仅允许从预定义的值列表中选择属性值。

> **注：**
>
> 可以使用 EnforceListProps 系统首选项限制为列表属性存储的属性值，使其仅为列表中的值。

> **注：**
>
> 可以对定义的属性或可覆盖的派生属性使用列表值。

- **隐藏** - 在属性网格中隐藏属性。
- **索引** - 为属性创建索引，以提高搜索、属性查询和验证的性能。此选项仅适用于已定义的、数据类型为字符串的属性。

> **注：**
>
> 具有索引的属性可以提高应用程序服务器上的内存使用率，并且只应当为在搜索、查询以及执行唯一性检查验证时可能使用的属性使用索引。

6. 执行以下任意操作：
   - 要将属性分配给类别，请从**可用**列表中选择类别，然后将其移至**已选中**列表。
   - 如果选择了**已定义**属性类型以及**列表**选项，请在**列表值**选项卡上执行以下操作：
     a. 单击**添加**并在列表中输入值。
     b. 在该行的“操作”列中单击**保存**。

> **注：**
>
> 对每行使用“移动”或“删除”可重新排列或删除列表值。使用“编辑”或双击某行可编辑该行，使用“取消”可取消编辑。

- 如果选择了**查找**属性类型，请选择**查找表**选项卡并执行以下操作：
  a. 单击**添加**以在列表中输入新的键值对。
  b. 在该行的“操作”列中单击**保存**。

> **注：**
>
> 对每行使用“移动”或“删除”可重新排列或删除列表值。使用“编辑”或双击某行可编辑该行，使用“取消”可取消编辑。

- 如果选择的数据类型允许设置层次限制，请选择**限制**选项卡并执行以下操作：
  a. 从**层次组属性**中选择一个属性，然后选择一个层次组。

  在节点选择器中，用户只能看到属于所选层次组的层次中的节点。

> **注：**
>
> Oracle Data Relationship Management Analytics 仅支持默认“核心”属性类型。

  b. **可选：**选择**强制对服务器属性更新施加限制**，以在通过 Web 客户端、导入、操作脚本或 Web 服务 API 更新属性时验证该限制。
- 如果选择了**派生**属性类型，则选择**参数**选项卡，为派生的属性定义公式或脚本。

  有关公式的详细信息，请参阅“创建公式”。有关脚本的详细信息，请参阅“创建动态脚本”。
- 如果选择了**外部查找**属性类型，请选择**外部查找**选项卡并输入以下信息：
  – **外部连接** - 选择数据库或 Web 服务连接
  – **操作** - 选择要执行的外部操作
  – 对于每个参数，配置：
    * **参数源类型** - 选择“文字”或“属性”。
    * **源** - 如果为源类型选择了**文字**，则在“参数源”列中输入文字值。为此“外部查找”属性调用外部操作时，为当前参数传递文字值。如果为源类型选择了**属性**，则选择属性来为外部操作提供参数值。执行“外部查找”时，参数值来自当前节点或请求项中的选定属性。
  – 在**列/属性映射**中，选择选定查找结果中的哪个结果列将为外部查找属性提供值。单击**添加**以添加可映射到不同属性的其他列，从而当选择外部查找值时，其他属性值将自动更新。

将自动定义第一个“列/属性”映射，无法删除该映射。此映射用于当前属性。必须选择某列，其默认为操作中存储的第一个列。可以修改第一行的列值，但不能修改属性值。对于其他映射，可以选择并编辑“列名”和“结果列”。

7. 单击 。

使用层次限制

层次限制可以限制在更新节点数据类型属性值时可供查看和选择的层次和节点。层次限制是使用节点数据类型的属性定义的可选配置。层次限制功能使用层次组和层次组属性，必须先配置层次组和层次组属性，然后才能分配层次限制。

可以对以下数据类型使用层次限制：

- 关联组
- 一个关联节点
- 多个关联节点
- 全局节点
- 叶节点
- 枝节点
- 多节点
- 节点

> **注：**
>
> 在设置层次限制时，“关联组”、“一个关联节点”、“多个关联节点”节点数据类型可能需要注意其他一些问题，因为关联节点会创建交叉引用。如果定义层次限制，应注意层次组中包括的所有层次可能互相关联。例如，“员工”和“成本中心”层次中的节点之间存在交叉引用。可能需要创建单独的层次组属性和层次组供层次限制使用。

编辑属性定义

如果将某个属性定义从“定义”属性类型更改为不可编辑类型（如“派生”或“查找”），则适用以下条件：

- 切换为非存储的属性类型时的确认消息被修改为可能影响更改请求项的暂挂更新的状态。
- 不再为分配了该任务的项显示、验证或最终提交正在进行的请求的暂挂属性更新。

要编辑属性定义：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**属性定义**。
3. 根据属性定义的类型展开**核心**或**自定义**。
4. 双击属性。
5. 修改任何可编辑的参数。

> **注意：**
>
> 如果将“属性类型”从定义的值（"RWDerived" 或“定义”）更改为不允许存储的值（“派生”或“查找”），定义的属性值将被删除并且此数据将丢失。在进行此类型的更改之前，您必须确认可以接受数据可能丢失。

有关详细信息，请参阅“创建属性”。

6. 单击 。

删除属性

如果从 Oracle Data Relationship Management 中删除了某个属性定义，则以下条件适用：

- 修改属性定义的相关性检查以包括工作流元数据引用，并且用户必须确认删除。工作流元数据的属性定义相关性包括以下内容
  - 工作流任务属性
  - 工作流任务验证属性
  - 更改请求项详细信息
- 在确认时，如果删除了某个属性，则将同时删除对该属性的每个相关引用，包括到工作流任务的分配、对正在进行的请求的暂挂更新以及历史更改请求。
- 对于属性定义的交互式删除，将始终保留事务历史。

要删除属性：

1. 从主页中选择**管理**。
2. 在**元数据**下，展开**属性定义**。
3. 选择属性并单击 。
4. 选择**删除属性定义**以确认删除。

> **注意：**
>
> 删除属性定义还将导致删除为该属性存储的所有值，以及从使用该属性的所有元数据对象中删除该属性。

10

管理验证

验证允许对版本、层次、节点和属性强制实施业务规则。验证可以在实时模式或批量模式下运行，也可以同时在这两种模式下运行。实时验证在修改时运行，如果操作违反了强制实施的规则，则会阻止保存更改。批量验证可以在进行编辑之前或之后显式运行，以识别无效且需要解决的数据条件。

验证类

验证类允许强制实施不同类型的业务规则。一些验证类具有通用性，而另一些验证类则具有特定用途。可以基于一组现有的验证类创建验证。针对节点的许多业务规则都可以通过使用查询来检查其逻辑的验证类强制实施。这样，验证还可以利用创建的用于分析的查询来管理数据完整性。版本和层次的规则或节点的特例可以使用其他验证类来实现。有些验证类仅用于产品测试用途，不能在生产环境中使用。

表 10-1　验证类

验证类	级别	说明	参数
BoolNodeInHier	节点	核实指定层次中的指定布尔属性是否具有 True 值	属性、层次
ContainAllProp	全局节点	核实指定层次是否包含指定属性为 True 的所有节点	层次、属性
ContainAllWith	全局节点	核实指定层次是否包含指定属性具有指定值的所有节点	层次、属性、值
CustPropQuery	节点	使用预定义的查询和预期结果进行核实 只能使用本地属性查询。	属性查询名称、失败值
DateRangeCheck	节点	核实“开始日期”是否早于或等于“结束日期”	开始日期属性、结束日期属性
Formula	节点	使用以公式表示的业务逻辑来核实节点。公式结果为 False 将导致验证失败。	Formula
GlobalPropQuery	全局节点	使用预定义的查询和预期结果进行核实	属性查询名称、失败值
HierContainsRef	节点	第一个布尔属性为 True 时，或者又一个布尔属性为 True 且该节点为叶节点时，层次包含对该节点的引用。	层次名称、针对所有节点的布尔属性、针对叶节点的布尔属性
HierFail	层次	在达到层次级别时自动失败以进行测试	无
InvalidNameLength	节点	核实节点名是否具有指定的长度。	长度

表 10-1 （续） 验证类

验证类	级别	说明	参数
MaxChildren	版本	核实每个节点的子代数量是否超出了指定的限制	子代的最大数量
MaxHierNodes	层次	核实层次中的节点数量是否超出了指定的限制	节点的最大数量
MaxVersionNodes	版本	核实版本中的节点数量是否超出了指定的限制	节点的最大数量
MergeEquiv	合并	核实受影响节点和合并节点的指定属性是否具有相同的值	全局节点属性
MergePropSet	合并	核实是否设置（覆盖）了指定属性的受影响节点属性值以及合并节点属性值（属性值无需相同）	属性
MixedKids	节点	检查同时具有枝和叶子代的节点。	无
NoBoolBranch	节点	核实指定布尔属性在指定分支上是否至少有一次设置为 True	属性
NodeFail	全局节点	在达到版本级别时自动在节点上失败以进行测试	无
NodeFailRandom	节点	在达到指定百分比的节点时自动失败以进行测试	失败百分比
NoDefaults	节点	核实指定属性是否使用了默认值	属性
NoPropBranch	节点	核实在指定的分支上是否至少设置过一次指定的属性	属性
PropEquivBool	节点	第三个布尔属性为 True 时的属性等同性。	要计算的布尔属性、第一个属性、第二个属性
PropLength	节点	核实指定属性是否未小于最小长度并且未超过最大长度	属性、最小长度、最大长度
PropRemove	移除	如果指定的一个或多个属性（在 prop1、prop2 和 prop3 参数中）等于指定的值（在 value1、value2、value3 参数中），则禁止删除节点。	Property1、Property2、Property3、Value1、Value2、Value3
RequiredField	节点	核实对于指定属性具有指定值的所有节点，必需列表中的每个属性是否都具有值： • 如果拒绝默认记录标志为 True，则必需列表中的每个属性必须具有非默认值 • 如果拒绝默认记录标志为 False，则可以接受默认值	属性、值、拒绝默认记录、必需属性

表 10-1 （续） 验证类

验证类	级别	说明	参数
Script	节点、层次、版本、全局节点、移动、删除、合并	使用动态脚本验证数据。验证通过后返回值为 True。验证失败后返回值为 False。	Script
SingleBoolBranch	节点	核实指定布尔属性在每个分支上是否仅设置为 True 一次	属性
SinglePropBranch	节点	核实指定属性在每个分支上是否仅设置过一次	属性
StrandedParent	节点	核实是否所有枝节点都具有子代	无
StrPropEqual	节点	在指定属性等于指定值的所有节点上失败	属性、值
UniqueProp	节点	核实指定属性在层次内是否有重复的值 如果“包括默认值”为 False，则不包括具有默认值的节点。 如果“排除共享项”为 True，则检查属性值的唯一性时不考虑共享节点。	属性、包括默认值、排除共享项 建议 UniqueProp 验证使用具有索引的属性。
UniquePropBranch	节点	核实指定属性在分支内是否具有唯一值	属性
VersionFail	版本	在达到版本级别时自动失败以进行测试	无
VersionUnique2Prop	全局节点	核实指定属性在版本内没有重复的值 如果“包括默认值”为 False，则不包括具有默认值的节点。 如果“排除共享项”为 True，则检查属性值的唯一性时不考虑共享节点。	第一个属性、第二个属性、包括默认值、排除共享项
VersionUniqueProp	全局节点	核实指定属性在版本内没有重复的值 如果“包括默认值”为 False，则不包括具有默认值的节点。 如果“排除共享项”为 True，则检查属性值的唯一性时不考虑共享节点。	属性、包括默认值、排除共享项

验证级别

验证级别定义业务规则的范围。对于节点验证，级别还可以包含操作类型，只有执行该类型的操作才会运行验证。下表定义了各个验证级别并指出了：

- 验证可以在批量模式、实时模式还是这两种模式下运行。
- 分配验证的位置。
- 验证在哪一个对象上运行。

表 10-2　验证级别

验证级别	在批量模式或实时模式下运行	分配位置	验证对象
节点 - 审核节点关系和属性以确保满足条件。 用于确定某个节点级别字符串属性值的长度是否有效。	实时或批量模式	版本、层次或节点	本地节点
层次 - 审核层次中的属性以确保满足条件。可以在层次或版本级别分配和运行。 用于确保层次的节点不超过 10,000 个。	批量	版本或层次	层次
版本 - 审核版本的属性。 用于确保版本所包含的节点不超过 100,000 个。	批量	版本	版本
全局节点 - 在版本级别分配。验证版本中的每个节点，无论其属于哪个层次，包括孤立节点。仅审核定义为全局属性的属性。 用于确保版本中的所有节点都具有唯一的属性值。	批量	版本	全局节点
合并 – 在执行需要合并的操作（例如删除或停用）时运行。在版本级别分配。 用于确保某个叶节点仅与另一个叶节点合并。	实时模式	版本	全局节点
移动 – 尝试移动节点时触发的验证。在层次级别分配。 用于防止在层次内移动成本中心。	实时模式	层次	本地节点
删除 – 类似于“移动”级别。尝试从层次中删除某个节点时运行。可用于防止删除指定类型的节点。 用于防止从层次中删除成本中心节点。	实时模式	版本或层次	全局节点

创建验证

要创建验证：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**验证**。

3. 输入验证的名称。

> **注：**
>
> 验证将分配给 Custom 命名空间。验证的全限定名称必须唯一。输入名称后会自动填写“标签”字段。验证标签是为应用程序管理以外的所有功能中的验证而显示的一个便于用户理解的描述短语。只要不位于相同的命名空间，多个验证可以具有相同的标签。

4. 输入验证失败时向用户显示的消息。
5. 选择一个验证类。请参阅“验证类”。

> **注：**
>
> 根据选择的类填充有效的级别。

6. 对于可以在节点级别实时运行的类，请选择包含操作类型的级别。
7. 从以下验证选项中进行选择：
   - 实时 - 执行更改时运行
   - 批量 - 显式请求时运行
   - 继承 - 针对选定的节点及其后代运行

> **注：**
>
> 根据选择的验证类，上述某些选项可能不可用或者您需要编辑所显示参数的值。

8. 定义所选验证类的参数。

   有关每个验证类的参数，请参阅“验证类”。有关创建公式的详细信息，请参阅“创建公式”。有关创建脚本的详细信息，请参阅“创建动态脚本”。

9. 单击 。

为移动操作创建脚本验证

要为移动操作创建脚本验证：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**验证**。
3. 从**类**中选择**脚本**。

默认情况下，验证级别为“节点”，运行模式为“批量”。

4. 在**运行此验证**下，选择**实时**。

这样可以针对特定操作（例如移动）触发验证。

5. 从**级别**中选择**移动**。

> **注：**
>
> “级别”选项位于在步骤 4 中选择的“实时”选项上方。

6. 保存验证。

分配验证

创建验证后，您可以将其分配给版本、层次、域和节点。可以同时分配多个验证。

> **注：**
>
> 在域级别分配时，验证由域中的所有成员节点继承。在版本级别分配时，验证由版本中的所有层次和节点继承。在层次级别分配时，验证由层次中的所有节点继承。

有关向域分配验证的信息，请参阅“管理域”。有关向版本、层次和节点分配验证的信息，请参阅《*Oracle Data Relationship Management* 用户指南》。

编辑验证

要编辑验证：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**验证**。
3. 选择验证并单击 。
4. 更改验证。

> **注：**
>
> 保存验证后，则无法修改“类”、“级别”和“运算模式”参数。

5. 单击**保存**。

删除验证

删除验证时，同时也将删除版本、层次和节点的所有验证分配。

要删除验证：

1. 从主页中选择**管理**。
2. 在**元数据**下，展开**验证**。
3. 选择验证并单击 。
4. 选择**删除此项**以确认删除。

11

管理公式

公式允许您使用 Oracle Data Relationship Management 中的本地公式语言为派生的属性和验证定义复杂的逻辑。公式由函数和字符串文字组成，并且必须遵循特定的语法规则。

有关详细信息，请参阅以下主题：

- 创建属性
- 管理验证

使用函数

函数名不区分大小写，并且不管是否需要参数，其后面都应该紧跟有括号。

函数参数必须为预期的类型和数量。参数可以是嵌套函数或字符串文字。如果参数的类型错误，则会报错。在参数太少的情况下，会报告列表索引超出范围的错误。在参数太多的情况下，会忽略多余的参数。

特殊字符

在某些参数值包含特殊字符（例如：comma、space、tab）的函数中，使用方括号 ([])。例如，`FlipList(PropValue(Custom.NodeList),[comma])` 对函数调用 `PropValue(Custom.NodeList)` 返回的逗号分隔的列表执行 FlipList 函数。

以下函数可以在方括号 ([]) 中包含 `comma`、`space` 或 `tab` 作为分隔符参数：`ArrayCount`、`ArrayIndex`、`ArrayItem`、`FlipList`、`Intersection`、`ListContains`、`PadList`、`RangeListContains`、`IsRangeListSubset`、`MinList`、`MaxList`、`AvgList`、`SumList`、`SortList`、`ListDistinct`、`ListNodePropValues` 和 `ListNodesWith`。

除了常规的文本字符串以外，`ReplaceStr` 函数（需要旧模式和新模式的参数）还可在方括号 ([]) 中包含 `comma`、`space`、`tab`、`crlf`、`cr`、`lf`、`openparen` 或 `closeparen`。

> **注：**
>
> 包含文字逗号的参数值将导致此语法错误：“参数数量无效”。将逗号分隔的列表作为函数调用的结果传入是有效的用法，并将按预期进行处理。例如：
>
> 无效语法：`FlipList(a,b,c,[comma])`
>
> 有效语法：`FlipList(PropValue(Custom.NodeList),[comma])`，其中 `Custom.NodeList` 值 = a,b,c

文字

任何不是后面跟有括号的有效函数名的值都被视为文字。文字可以是字符串、整数、浮点数或布尔文字。在字符串文字中，将空格视为字符。因此，请勿在公式中使用多余的空格，除非需要这么做才能派生正确的结果。您可以使用“删除空格”选项在保存之前去除公式中的空格。

格式字符串参数

传递到字符串格式设置例程的格式字符串包含两种类型的对象 - 文字字符和格式指定符。文字字符会逐字复制到生成的字符串。格式指定符从指定的属性获取属性值，并对其应用格式设置。格式字符串中只能存在一个指定符。

格式指定符采用以下形式：

```
"%"["-"][width]["."prec]type
```

表 11-1　格式字符串字符

字符	说明
%	指示格式指定符的开始
["—"]	左对齐指示符（可选） 通过在值后面添加空格来左对齐结果。默认情况下，通过在值的前面添加空格来右对齐结果。
[width]	宽度指定符（可选） 设置转换的最小字段宽度。如果生成的字符串小于最小字段宽度，则会添加空格以增加字段宽度。
["." prec]	精度指定符（可选）
type	转换类型字符 转换字符可以使用大写或小写进行指定。对于所有的浮点格式，用作小数和千位分隔符的实际字符通过 DecimalSeparator 和 ThousandSeparator 全局变量或它们的 TFormatSettings 等效项获取。下表中列出了类型的有效值。

表 11-2　格式字符串类型值

类型值	说明
d	十进制 属性值必须为整数。将值转换为十进制数字字符串。如果格式字符串包含精度指定符，则其指示生成的字符串必须至少包含指定的位数；如果值少于指定的位数，则生成的字符串会在左边添加零。

表 11-2 （续） 格式字符串类型值

类型值	说明
u	无符号的十进制 类似于 d，但是不输出符号。
e	科学记数法 属性值必须为浮点值。将值转换为 "-d.ddd...E+ddd" 格式的字符串。如果数字为负数，则生成的字符串以减号开头。小数点前面始终有一位数字。生成字符串中的总位数（包括小数点前面的一位）由格式字符串中的精度指定符指定；如果没有精度指定符，则使用默认精度 15。生成的字符串中的 "E" 指数字符后面始终跟有一个加号或减号以及至少三位数字。
f	固定 属性值必须为浮点值。将值转换为 "-ddd.ddd..." 格式的字符串。如果数字为负数，则生成的字符串以减号开头。小数点后面的位数由格式字符串中的精度指定符指定；如果没有精度指定符，则使用默认的两位十进制数字。
g	常规 属性值必须为浮点值。使用固定或科学记数法格式将值转换为尽可能最短的十进制字符串。生成的字符串中的有效数字位数由格式字符串中的精度指定符指定；如果没有精度指定符，则使用默认精度 15。将删除生成的字符串尾部的零，并且仅在需要时显示小数点。如果值中小数点左边的数字位数小于或等于指定的精度，并且值大于或等于 0.00001，那么生成的字符串使用固定点格式。否则，生成的字符串使用科学记数法格式。
n	数字 属性值必须为浮点值。将值转换为 "-d,ddd,ddd.ddd..." 格式的字符串。"n" 格式与 "f" 格式对应，除非生成的字符串包含千位分隔符。
m	货币 属性值必须为浮点值。将值转换为代表货币金额的字符串。转换由 CurrencyString、CurrencyFormat、NegCurrFormat、ThousandSeparator、DecimalSeparator 和 CurrencyDecimals 全局变量或它们在 TFormatSettings 数据结构中的等效项控制。如果格式字符串包含精度指定符，则它将覆盖由 CurrencyDecimals 全局变量或其 TFormatSettings 等效项指定的值。

表 11-2 （续） 格式字符串类型值

类型值	说明
s	字符串 属性值必须为字符、字符串或 PChar 值。插入字符串或字符来代替格式指定符。如果格式字符串中包含精度指定符，则精度指定符指定生成的字符串的最大长度。如果属性值是长于此最大长度的字符串，则会截断该字符串。
x	十六进制 属性值必须为整数值。将值转换为十六进制数字字符串。如果格式字符串包含精度指定符，则其指示生成的字符串必须至少包含指定的位数；如果值少于指定的位数，则生成的字符串会在左边添加零。

日期时间格式字符串

日期时间格式字符串指定日期时间值（例如 TDateTime）在转换为字符串时的格式设置。日期时间格式字符串由代表要插入格式化字符串的值的指定符组成。一些指定符（例如 "d"）设置数字或字符串的格式。其他指定符（例如 "/"）引用来自全局变量的特定于区域设置的字符串。格式中忽略指定符的大小写，但 "am/pm" 和 "a/p" 指定符除外。

指定符	显示
c	日期后面跟有时间 **注意：**如果日期时间值精确指示午夜，则不显示时间。
d	日期显示为无前导零的数字 (1–31)
dd	日期显示为带前导零的数字 (01–31)
ddd	日期显示为缩写 (Sun-Sat)
dddd	日期显示为全名 (Sunday-Saturday)
ddddd	日期的短格式
dddddd	日期的长格式
e	当前期间/时代的年份显示为无前导零的数字（仅适用于日本、韩国和中国台湾地区区域设置）
ee	当前期间/时代的年份显示为带前导零的数字（仅适用于日本、韩国和中国台湾地区区域设置）
g	期间/时代显示为缩写（仅适用于日本和中国台湾地区区域设置）
gg	期间/时代显示为全称（仅适用于日本和中国台湾地区区域设置）
m	月份显示为无前导零的数字 (1–12) **警告：**如果 "m" 指定符紧跟在 "h" 或 "hh" 指定符之后，则显示分钟而非月份。

指定符	显示
mm	月份显示为带前导零的数字 (01–12) **警告：**如果 "mm" 指定符紧跟在 "h" 或 "hh" 指定符之后，则显示分钟而非月份。
mmm	月份显示为缩写 (Jan-Dec)
mmmm	月份显示为全名 (January-December)
yy	年份显示为两位数字 (00–99)
yyyy	年份显示为四位数字 (0000–9999)
h	无前导零的小时 (0–23)
hh	带前导零的小时 (00–23)
n	无前导零的分钟 (0–59)
nn	带前导零的分钟 (00–59)
s	无前导零的秒 (0–59)
ss	带前导零的秒 (00–59)
z	无前导零的毫秒 (0–999)
zzz	带前导零的毫秒 (000–999)
t	使用 ShortTimeFormat 全局变量指定的格式的时间
tt	使用 LongTimeFormat 全局变量指定的格式的时间
am/pm	对前面的 "h" 或 "hh" 指定符使用 12 小时制时钟，并针对中午之前的所有小时显示 "am"，针对中午之后的所有小时显示 "pm"。am/pm 指定符可以使用小写或大写，也可以混合使用大小写，其结果会相应地显示。
a/p	对前面的 "h" 或 "hh" 指定符使用 12 小时制时钟，并针对中午之前的所有小时显示 "a"，针对中午之后的所有小时显示 "p"。a/p 指定符可以使用小写或大写，也可以混合使用大小写，其结果会相应地显示。
ampm	为前面的 "h" 或 "hh" 指定符使用 12 小时制时钟
/	区域设置指定的日期分隔符
:	区域设置指定的时间分隔符
'xx'/"xx"	使用单引号或双引号括起来的字符照原样显示，不影响格式设置。

公式计算

您可以在创建或修改属性定义或验证时测试公式。使用提供的属性值计算公式以得出公式的结果。此过程在可能缺少简单语法验证的公式中可能会发现逻辑错误或实施错误。公式结果和所有公式错误或状态消息都会显示。

公式从左到右进行计算，遇到函数和字符串文字时，随即计算函数和字符串文字。通过此方法，嵌套函数在显示在嵌套函数右边的其他参数之前进行计算。函数可以显式嵌套在公式中，或通过检索其他公式属性的值隐式嵌套。在大多数情况下，应避免使用循环引用（引用属性本身的属性公式，可以为显式引用，也可以为隐式引用）。Oracle Data Relationship Management 检测并阻止不良的循环引用，除非有必要并且易于理解，否则不应使用循环引用。

公式语法检查

在保存公式之前，针对以下内容核实公式语法：

- 函数名称正确。
- 属性名称正确。
- 左右括号数目相等。
- 参数的实际数量至少为每个函数的预期参数数量

诸如 `Concat` 等函数可以包含任意数量的参数。参数计数验证可核实参数的实际数量是否等于或大于参数的预期数量。因此，参数过多不会生成错误，但是参数太少却会生成错误。

语法验证不会计算公式，因此如果输入无效常量，可能会发生错误。例如：`IntToStr(ABC,3)` 通过了语法验证，但是在 Oracle Data Relationship Management 应用程序中生成了错误。您必须在保存之前计算每个公式以避免此类型的错误。

语法中的属性名称检查

为了准确地对属性名称执行语法验证，会部分计算需要属性名称的函数，以查看是否存在属性名称不是文字，而是某个函数的结果的少数情况。

请考虑以下示例：

- 公式 `PropValue(Concat(Core.Abbrev))` 有效，但是必须计算 `Concat` 函数（不仅仅验证语法）以核实属性名称。
- 公式 `PropValue(If(NodeIsLeaf(),Core.Abbrev,Custom.Label))` 有效，但是必须计算 If 函数以核实属性名称。

如果所涉及的属性名称只是公式的一部分，则仅计算需要确定属性名称的部分。例如，在公式 `Add(PropValue(Concat(Core.,I,D)),If(NodeIsLeaf(),0,1))` 中，为验证语法而计算的公式部分只有 `Concat` 函数及其参数。

在诸如 `PropValue(PropValue(NodeType))` 等情况中，计算这些公式部分变得很重要。对于该公式，除非为 *Custom.NodeType* 属性提供值，否则语法验证将失败。

使用公式的注意事项

数据类型转换

某些函数要求数据值为某种特定的数据类型，才能正确计算。例如，执行数学计算的函数要求输入参数为整数或浮点值，而字符串处理函数要求将字符串值提供为输入。在某些情况下，数据值必须从一种数据类型转换为另一种数据类型，才能成功派生。Oracle Data Relationship Management 提供了一组函数，用于在公式内处理数据类型转换。

属性级别限制

通常，为管理粒度级别较低的数据而创建的属性定义可以引用管理粒度级别较高的数据的其他属性。

- 本地节点 - 可以引用其他本地节点、全局节点、层次或版本属性
- 全局节点 - 可以引用其他全局节点或版本属性
- 层次 - 可以引用其他层次或版本属性（仅查找）
- 版本 - 可以引用其他版本属性（仅查找）

引用其他节点的属性

派生属性或验证通常计算或检索其他节点（而非正在为其计算公式的当前节点）的属性值。Data Relationship Management 提供了多个函数，可让您访问相同版本中节点的属性值。

- NodePropValue
- ParentPropValue
- HierNodePropValue
- AncestorProp
- DualAncestorProp
- AscNodeProp
- ReplacePropValue
- ListPropValues
- ListNodePropValues

从全局节点属性引用本地节点属性

全局节点属性不需要有层次上下文就可返回值，而本地节点属性则需要指定层次。针对全局节点计算的派生属性或验证无法使用标准 PropValue 或 NodePropValue 函数引用本地节点属性值。全局节点属性可以使用 HierNodePropValue 函数引用本地节点属性值，但必须指定特定的层次以检索该层次中特定本地节点的属性值。

嵌套函数

将函数组合到一个公式中称为嵌套函数。公式中一个函数的输出用作另一个函数的输入参数。当计算嵌套函数时，Data Relationship Management 首先执行最内层的函数，然后向外执行计算。函数可以在相同公式中显式嵌套或使用一个引用属性（该属性使用不同的公式）的公式来隐式嵌套。

将属性用作其他属性的变量

Data Relationship Management 允许您使用嵌套函数、对其他属性或节点的引用以及文字值的组合，这可能导致冗长或复杂的公式。您可以使用单独的属性定义来模块化公式逻辑，并简化获得相同结果所需的公式语法。此方法能够显著提高维护这些公式的便捷性。

另外，公式可能会在相同的属性定义内或指定节点的多个属性定义之间多次计算相同的数据或执行相同的计算。当此逻辑嵌套在更大的公式中或在属性定义中实施时，这些检查和计算会多次执行，从而可能会影响要求计算属性的运算的性能。您可以通过隔离单独的属性定义内重复的公式逻辑，最大程度地减少多余的处理。

使用递归遍历层次关系

层次较低级别的节点的业务规则可能要求计算它们上面的祖先节点的属性值。允许较低级别节点引用这些属性值的一种方法是对管理必须引用的值的属性定义启用继承。但是，在许多情况下，对属性定义使用继承并不合适。

您可以配合使用特定的层次公式函数和当前属性定义的自引用，向上递归层次的一个分支以检索或计算祖先节点的属性值。

ParentPropValue - 使用此函数可在当前层次中向上递归一个祖先节点分支。例如：

```
If(Equals(Integer,PropValue(Core.Level),1),Label
Only,ParentPropValue(Essbase.DataStorage))
```

HierNodePropValue - 使用此函数可在其他层次中向上递归一个祖先节点分支。例如：

```
If(Equals(Boolean,PropValue(Custom.PlanPoint),True),Abbrev(),HierNodePropV
alue(Geography,HierNodePropValue(Geography,Abbrev(),Core.Parent),Custom.Pl
anMember))
```

创建公式

可在公式编辑器（在“参数”选项卡中提供，用于创建或编辑派生属性定义和验证）中创建公式。

要创建公式：

1. 可以通过以下方式在**参数**选项卡中输入文本公式或者插入函数和属性：
   - 要插入函数，请将游标置于公式中，然后单击**插入函数**。将显示函数的列表。展开一个函数以查看其输入参数。输入参数值，然后单击**确定**。
   - 要插入属性，请将游标置于公式中，然后单击**插入属性**。将显示属性的列表。选择属性，然后单击**确定**。
2. 从以下选项中进行选择：
   - **删除空格** - 默认情况下将选中该项。如果选择该项，计算公式以及保存属性时会删除公式中的所有空格。要将公式中要计算的空格保留为文字值，请禁用此选项。
   - 要计算公式，请选择一个选项：
     - **使用选定节点计算** - 单击 … 并选择一个节点。将在公式中使用该节点的当前属性值。单击**计算**。结果显示在公式设计器的底部。
     - **计算便笺** - 手动输入属性值。也可以从节点复制值，然后修改这些值以用于进行计算。在“从节点复制”中，单击 … 并选择一个节点以在网格中显示其属性值。使用列标题下面的筛选器行筛选属性列表。使用“操作”列中的“编辑”按钮修改属性值以使用公式进行计算。单击“计算”。计算结果显示在公式设计器的底部。
3. 要测试公式，请单击**计算**。

函数定义

下面是按字母排序的可用于派生的公式属性定义的函数列表。

Abbrev

说明

返回当前节点的名称（缩写）。

语法

```
Abbrev(): String
```

示例

```
Abbrev()
```

返回值是节点的名称。

Add

说明

将两个指定的整数值相加并返回结果。

语法

```
Add(Int1,Int2:Integer):Integer
```

示例

```
Add(1,4)
```

返回值为 5。

AddedBy

说明

返回“添加者”更改跟踪属性的值。

语法

```
AddedBy():String
```

示例

```
AddedBy()
```

返回向版本中添加当前节点的用户的名称。

AddedOn

说明

以日期/时间形式返回“添加时间”更改跟踪属性的值。

语法

```
AddedOn():Date/Time
```

示例

```
AddedOn()
```

返回向版本中添加当前节点的日期和时间。

AddFloat

说明

将两个指定的浮点值相加并返回结果。

语法

```
AddFloat(Float1,Float2:Float):Float
```

示例

```
AddFloat(2.14,3.75)
```

返回值为 5.89。

AncestorProp

说明

返回属性等于指定值的第一个祖先的属性值。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

> **注：**
> 如果当前节点满足条件，则将其返回。

语法

```
AncestorProp(Operator:String,Property:String,Value:String,FromTop:Boole
an,ReturnProp:String)
```

Operator 是比较属性和值时要使用的运算符。有效值：=、<、>、>= 和 <=。

Property 是要使用的属性的名称。

Value 是要比较的值。

FromTop 指定是否从层次的顶级节点搜索。如果为 False，则从当前节点开始执行搜索。

ReturnProp 是要返回的属性的名称。

And

说明

如果所有指定布尔表达式计算结果都为 True，则返回 True。

语法

```
And(Expression1,Expression2,...ExpressionN:Boolean):Boolean
```

示例

```
And(1,T,True)
```

返回值为 True。

ArrayCount

说明

返回指定列表（数组）中的项目数。

语法

```
ArrayCount(List:String,Delimiter:String):Integer
```

List 指定要在其中进行搜索的字符串列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

示例

```
ArrayCount(Diet Cola;Root Beer;Cola,[comma])
```

返回值为 3。

ArrayIndex

说明

返回指定的项目在列表（数组）中首次出现的位置。如果找不到项目，则返回零 (0)。

语法

```
ArrayIndex(Item:String,List:String,Delimiter:String):Integer
```

Item 指定要测试的字符串值。

List 指定要在其中进行搜索的字符串列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

示例

```
ArrayIndex(Cola,Diet Cola;Root Beer;Cola,[comma])
```

返回值为 3。

ArrayItem

说明

返回列表（数组）中指定索引位置的项目。

语法

```
ArrayItem(List:String,Delimiter:String,Index:Integer):String
```

List 指定要在其中进行搜索的字符串列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

Index 是字符串在列表中的位置。负值表示列表中的最后一项。

示例

```
ArrayItem(Diet Cola;Root Beer;Cola,;,3)
```

返回值为 Cola。

AscNodeProp

说明

返回指定属性引用的关联节点的属性值。

语法

```
AscNodeProp(LookUpProperty,ReturnProperty)
```

LookupProperty 是指向节点的属性的名称。属性的数据类型必须为 Node 或 AscNode。

ReturnProperty 是要返回的关联节点属性名称。属性必须为全局属性。

AvgList

说明

返回列表中项目的平均值，忽略空白项目。如果列表包含不是指定项目类型的项目，则返回空白字符串。

语法

```
AvgList(InputList:String,Delimiter:String,ItemType:String):String
```

InputList 指定要使用的列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

ItemType 指示列表成员所需的项目数据类型。有效值：integer、float 和 datetime。默认值为 float。

示例

```
AvgList(1;2;3,[comma],Integer)
```

返回值为 2。

BoolToStr

说明

返回转换为 True 或 False 的布尔值。如果输入不表示布尔值，则返回 False。

语法

```
BoolToStr(Expression:Boolean):String
```

示例

```
BoolToStr(1)
```

返回值为 True。

Changed

说明

以布尔值形式返回“节点已更改”更改跟踪属性的值。

语法

```
Changed()
```

ChangedBy

说明

返回上次更新版本中当前节点的用户的名称。

语法

```
ChangedBy():String
```

示例

```
ChangedBy()
```

ChangedOn

说明

返回“更改时间”更改跟踪属性的值。

语法

```
ChangedOn():Date/Time
```

示例

```
ChangedOn()
```

返回上次在版本中更新当前节点的日期和时间。

Concat

说明

将两个或多个指定的字符串串联为一个字符串并返回结果。

语法

```
Concat(Item1,Item2,... ItemN:String):String
```

示例

```
Concat(Abbrev,-,Descr())
```

如果当前节点名称为 100 并且当前节点说明为 Colas，则返回值为 100–Colas。

ConcatWithDelimiter

说明

将两个或多个字符串串联为一个带分隔符的列表并返回结果。

语法

```
ConcatWithDelimiter(Delimiter:String,SkipBlanks:Boolean,Items:String)
```

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

SkipBlanks 指示是否跳过字符串列表中的空白值。有效值：1、0、T、F、t、f。

Items 指定要串联的一系列字符串。

示例

```
ConcatWithDelimiter([comma],1,Item1,Item2,Item3,Item4)
```

返回值为 Item1; Item2; Item3; Item4。

Decode

说明

返回输入字符串以及由适当字符取代的 [openparen]、[closeparen]、[comma]、[tab]、[space]、[crlf]、[cr] 和 [lf] 的所有实例。

> **注：**
>
> 此函数用于升级使用特殊字符的属性定义名称。这些特殊字符可能会导致派生属性公式的解析问题。此函数主要用于将使用过时的派生程序类的现有属性转换为 Formula 派生程序类。

语法

```
Decode(CodedString:String):String
```

CodedString 是对其执行函数的字符串值。

DefaultProp

说明

返回属性的默认值。

语法

```
DefaultProp(Property:String)
```

Property 是要使用的属性的名称。

Descr

说明

返回当前节点的说明。

语法

```
Descr():String
```

示例

如果当前节点说明为 Colas，则返回值为 Colas。

Divide

说明

将两个指定的整数值相除并返回结果。

语法

```
Divide(Int1,Int2:Integer):Integer
```

示例

```
Divide(200,10)
```

返回值为 20。

DivideFloat

说明

将两个浮点数（浮点）相除并返回结果。

语法

```
Divide(Float1,Float2:Float):Float
```

示例

```
DivideFloat(2.535,1.5)
```

返回值为 1.69。

DualAncestorProp

说明

返回两个属性等于指定值的第一个祖先的属性值。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
DualAncestorProp(Operator1:String,Property1:String,Value1:String,Operat
or2:String,Property2:String,Value2:String,FromTop:Boolean,ReturnProp:St
ring):String
```

Operator1 是比较第一个属性和值时要使用的运算符。有效值：=、<、>、>= 和 <=。

Property1 是要检查的第一个属性的名称。

Value1 是要比较的第一个值。

Operator2 是比较第二个属性和值时要使用的运算符。有效值：=、<、>、>= 和 <=。

Property2 是要检查的第二个属性的名称。

Value2 是要比较的第二个值。

FromTop 指定是否从层次的顶级节点搜索。如果为 False，则从当前节点开始执行搜索。

ReturnProp 是要返回的祖先属性名称。

Equals

说明

如果两个指定的值相等，则返回 True。此函数区分大小写。

语法

```
Equals(ParamType:String,Param1:String,Param2:String):Boolean
```

ParamType 是要用于比较值的数据类型。有效值：string、integer、float 和 date。默认值为 integer。

Param1 是要比较的第一个值。

Param2 是要比较的第二个值。

示例

```
Equals(integer,01,1)
```

返回值为 True。

FlipList

说明

返回表示指定列表的反转列表的字符串。

语法

```
FlipList(List,Delimiter:String):String
```

List 指定要翻转的字符串列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

示例

```
FlipList(DietCola;Orange Soda;Root Beer;Lemonade,[comma])
```

返回值为 Lemonade,Root Beer,Orange Soda,Diet Cola。

FloatToStr

说明

返回转换为字符串的浮点值。如果输入值不表示浮点值，则返回零 (0)。

语法

```
FloatToStr(Float1:Float):String
```

示例

```
FloatToStr(1.001)
```

返回值为 1.001。

Format

说明

使用指定的格式字符串参数类型标识符和指定类型的参数值设置值的格式。此函数只限使用一个值参数。

语法

```
Format(Format:String,ParamType:String, ValueToFormat:String):String
```

Format 是要应用的格式。

ParamType 是要用于比较值的数据类型。有效值：string、integer、float 和 date。默认值为 integer。

ValueToFormat 是对其执行函数的值。

示例

```
Format('%8.2f',Float,123.456)
```

返回值为 123.46。

FormattedDate

说明

返回使用指定格式字符串格式化的日期属性的值。

语法

```
FormattedDate(PropertyName:String,FormatString:String): String
```

PropertyName 是要使用的属性的名称。

FormatString 指定要应用的日期格式。

GreaterThan

说明

比较两个值，如果第一个值大于第二个值，则返回 True。

语法

```
GreaterThan(Value1:Integer,Value2:Integer,ParamType:String):Boolean
```

Value1 是要比较的第一个值。

Value2 是要比较的第二个值。

ParamType 是要用于比较值的数据类型。有效值：string、integer、float 和 date。默认值为 integer。

示例

```
GreaterThan(1,2)
```

返回值 False。

GreaterThanOrEqual

说明

比较两个值，如果第一个值大于或等于第二个值，则返回 True。

语法

```
GreaterThanOrEqual(Value1:Integer,Value2:Integer,ParamType:String):Bool
ean
```

Value1 是要比较的第一个值。

Value2 是要比较的第二个值。

ParamType 是要用于比较值的数据类型。有效值：string、integer、float 和 date。默认值为 integer。

示例

```
GreaterThanOrEqual(2,2)
```

返回值为 True。

HasCharacters

说明

如果指定的输入包含字符类中的字符、特殊字符或 CharList 中列出的字符，则返回 True。

语法

```
HasCharacters(Input:String,CharList:String):Boolean
```

Input 是要测试的字符串值。

CharList 是要测试的字符列表，包括可选的特殊值。特殊字符值括在括号中并使用逗号分隔。有效值：[alpha]、[numeric]、[whitespace]、[punctuation]、[uppercase]、[lowercase]、[comma]、[space]、[tab]、[crlf]、[cr]、[lf]、[openparen] 和 [closeparen]。

HasChildWith

说明

如果指定的表达式对于当前节点的所有子代都为 True，则返回 True。

语法

```
HasChildWith(Expression:Boolean):Boolean
```

示例

```
HasChildWith(GreaterThan(ID(),200))
```

如果当前节点的任何子代的 ID 大于 200，则返回值为 True。

HasParentNode

说明

如果当前本地节点具有父节点，则返回 True。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
HasParentNode():Boolean
```

示例

```
HasParentNode()
```

如果节点是层次顶级节点或任何后代节点的子代，则返回值为 True。

HasSiblingWith

说明

如果指定的表达式对于当前节点的所有同级都为 True，则返回 True。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
HasSiblingWith(Expression:Boolean):Boolean
```

示例

```
HasSiblingWith(PropValue(Leaf))
```

如果有任何子代为叶节点，则返回值为 True。

HierNodePropValue

说明

返回指定节点的指定属性在指定层次中的值。

语法

```
HierNodePropValue(HierAbbrev:String,NodeAbbrev:String,PropAbbrev:String
):String
```

HierAbbrev 是要使用的层次的名称。

NodeAbbrev 是要使用的节点的名称。

PropAbbrev 是要使用的属性的名称。

示例

```
HierNodePropValue(Assets,1000,Description)
```

如果 Assets 层次中节点 1000 的说明为 "Banking"，则返回值为 Banking。

ID

说明

返回当前节点的 ID。

语法

```
ID():Integer
```

示例

```
ID()
```

如果当前节点 ID 为 2000，则返回值为 2000。

If

说明

如果指定的表达式计算结果为 True，则返回 TrueResult 参数的值。否则将返回 FalseResult 参数的值。

语法

```
If(Expression:Boolean, TrueResult:String,FalseResult:String):String
```

Expression 是要计算的布尔表达式。

TrueResult 是在条件为 True 时返回的字符串值。

FalseResult 是在条件为 False 时返回的字符串值。

示例

```
If(Equals(String,Descr(),),Abbrev(),Concat(Abbrev,-,Descr()))
```

如果节点名称为 Colas 并且当前节点说明为空，则返回值为 Colas。

如果节点名称为 100 并且当前节点说明为 Colas，则返回值为 100–Colas。

InheritedPropOrigin

说明

返回继承属性值的来源节点的名称。如果指定的属性为全局属性，那么也会返回原始层次。如果指定的属性不是继承属性，或者如果找不到节点或属性，则返回 False。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
InheritedPropOrigin(PropAbbrev:String,Node:String):String
```

示例

```
InheritedPropOrigin(Custom.AccountType,Abbrev())
```

PropAbbrev 是要使用的属性的名称。

Node 是要使用的节点的名称。

InRange

说明

如果指定的值在指定的值范围内，则返回 True。如果输入参数为字符串，那么 Min 和 Max 参数指定要检查的字符串长度范围。对于其他类型，Min 和 Max 指定要检查的数字或日期值范围。

> **注：**
>
> 如果 MinExclusive/MaxExclusive 为 True，那么等于 Min/Max 的值包括在范围内，否则排除在外。

语法

```
InRange(DataType:String,Input:String,Min:String,Max:String,MinExclusive
:String,MaxExclusive:String):Boolean
```

DataType 是要使用的数据类型。有效值：string、integer、float 和 datetime。

Input 是要测试的字符串值。

Min 是长度或范围检查的最小值。

Max 是长度或范围检查的最大值。

MinExclusive 指定是否从要检查的范围中排除 Min 值。

MaxExclusive 指定是否从要检查的范围中排除 Max 值。

示例

```
InRange(Integer,5,1,10,False,False)
```

返回值为 True。

InternalPrefix

说明

返回当前节点名称的非数字前缀。

语法

```
InternalPrefix()
```

Intersection

说明

返回两个指定值列表共有项的集合。结果的顺序取决于项目在指定的第一个列表中的显示顺序。

语法

```
Intersection(List1:String,List2:String,Delimiter:String):String
```

List1 指定要在其中进行搜索的字符串列表。

List2 指定要在其中进行搜索的字符串列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

示例

```
Intersection(A;B;C;D;E,C;E;F;A,[comma])
```

返回值为 A,C,E。

IntToStr

说明

返回转换为字符串数据类型的指定整数值。如果输入值不表示整数，则返回零 (0)。

语法

```
IntToStr(Int1:Integer):String
```

示例

```
IntToStr(12345)
```

返回值为 12345。

InvertedLevel

说明

返回当前节点下后代的最大深度。

语法

```
InvertedLevel()
```

IsAlpha

说明

如果指定的字符串只包含字母字符（不区分大小写），则返回 True。

语法

```
IsAlpha(String:String):Boolean
```

示例

```
IsAlpha(A23D)
```

返回值 False。

IsAlphaNumeric

说明

如果指定的字符串只包含字母或数字字符（不区分大小写），则返回 True。

语法

```
IsAlphaNumeric(String:String,AllowBlanks:Boolean):Boolean
```

String 是要测试的字符串值。

AllowBlanks 指定是否应该将空白字符串视为数字。默认值为 False。

示例

```
IsAlphaNumeric(ABC123,True)
```

返回 True。

IsBlank

说明

如果指定的输入值为空字符串（零长度），则返回 True。

语法

```
IsBlank(Input:String):Boolean
```

示例

```
IsBlank(Descr())
```

如果节点说明为空，则返回 True。

IsBottomNode

说明

如果指定的节点没有子节点，则返回 True。如果找不到节点，则返回 False。

语法

```
IsBottomNode(Node:String):Boolean
```

Node 是要使用的节点的名称。

示例

```
IsBottomNode(Abbrev)
```

如果节点没有子代，则返回 True。

IsDataType

说明

如果输入值与指定的数据类型匹配，则返回 True。

语法

```
IsDataType(DataType:String,Input:String):Boolean
```

DataType 是要使用的数据类型。有效值：boolean、string、integer、float 和 datetime。

Input 是要测试的字符串值。

示例

```
IsDataType(123,Integer)
```

返回 True。

IsDefinedPropVal

说明

如果指定节点的指定属性具有定义（覆盖）的值，则返回 True。如果找不到节点或属性，则返回 False。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
IsDefinedPropVal(PropAbbrev:String,Node:String):Boolean
```

PropAbbrev 是要使用的属性的名称。

Node 是要使用的节点的名称。

示例

```
IsDefinedPropVal(Custom.AccountType,Abbrev())
```

如果“帐户类型”属性具有定义（覆盖）的值，则返回 True。

IsNodeAbove

说明

如果第一个节点在当前层次中是第二个节点的祖先，则返回 True。如果找不到 Node1 或 Node2，则返回 False。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
IsNodeAbove(Node1:String,Node2:String):Boolean
```

Node1 是要使用的第一个节点的名称。

Node2 是要使用的第二个节点的名称。

示例

```
IsNodeAbove(Parent,Child)
```

如果节点父代是子节点的祖先，则返回 True。

IsNodeBelow

说明

如果第一个节点在当前层次中是第二个节点的后代，则返回 True。如果找不到 Node1 或 Node2，则返回 False。

语法

```
IsNodeBelow(Node1:String,Node2:String):Boolean
```

Node1 是要使用的第一个节点的名称。

Node2 是要使用的第二个节点的名称。

示例

```
IsNodeBelow(Child,Parent)
```

如果节点子代是父节点的后代，则返回 True。

IsNumeric

说明

如果指定的值只包含数字字符 (0-9)，则返回 True。

语法

```
IsNumeric(String: String,AllowBlanksAsNumeric:Boolean):Boolean
```

String 是要测试的字符串值。

AllowBlanksAsNumeric 指定是否允许将空值视为字符串。默认值为 False。

示例

```
IsNumeric(12345)
```

返回值为 True。

IsRangeListSubset

说明

如果指定的值是指定范围列表的子集，则返回 True。

语法

```
IsRangeListSubset(RangeList:Range List,SubsetRangeList:Range
List,Delimiter:String):Boolean
```

RangeList 是要搜索的整数范围列表，使用指定的分隔符分隔。

SubsetRangeList 是要搜索的整数范围列表的子集，使用指定的分隔符分隔。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]

- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

Length

说明

返回指定字符串值中的字符数。

语法

```
Length(String:String):Integer
```

示例

```
Length(Desc())
```

如果当前节点的说明为 Colas，则返回值为 5。

LessThan

说明

比较两个值，如果第一个值小于第二个值，则返回 True。

语法

```
LessThan(Value1:Integer,Value2:Integer,ParamType:String):Boolean
```

Value1 是要比较的第一个值。

Value2 是要比较的第二个值。

ParamType 是要用于比较值的数据类型。有效值：string、integer、float 和 date。默认值为 integer。

示例

```
LessThan(1,2)
```

返回值为 True。

LessThanOrEqual

说明

比较两个值，如果第一个值小于或等于第二个值，则返回 True。

语法

```
LessThanOrEqual(Value1:Integer,Value2:Integer,ParamType:String):Boolean
```

Value1 是要比较的第一个值。

Value2 是要比较的第二个值。

ParamType 是要用于比较值的数据类型。有效值：string、integer、float 和 date。默认值为 integer。

示例

```
LessThanOrEqual(3,3)
```

返回值为 True。

ListAncestors

说明

返回起始于顶级节点的当前节点祖先名称的列表，以逗号分隔。如果当前节点不是本地节点，则返回空白字符串。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
ListAncestors(SortOrder:String):String
```

SortOrder 指定节点返回列表的排序顺序。支持的排序顺序值：

- [hier] - 本地上下文的默认值。节点列表按照当前层次的标准层次排序顺序返回。
- [alpha] - 返回按照节点名称排序的节点列表。
- [nodeid] - 仅限用于向后兼容。节点列表按照返回列表中每个节点的节点 ID 的数字排序返回。

> **注：**
>
> 必须使用括号将 SortOrder 参数括起来。

示例

```
ListAncestors([alpha])
```

如果 A、B、C 和 D 是 Z 的子代，而 Z 是 Y 的子代，并且当前节点是 D，则返回值为 Z,Y。

ListChildren

说明

返回当前节点的子代列表，以逗号分隔。

语法

```
ListChildren(SortOrder:String):String
```

SortOrder 指定节点返回列表的排序顺序。支持的排序顺序值：

- [hier] - 本地上下文的默认值。节点列表按照当前层次的标准层次排序顺序返回。
- [alpha] - 返回按照节点名称排序的节点列表。
- [nodeid] - 仅限用于向后兼容。节点列表按照返回列表中每个节点的节点 ID 的数字排序返回。

> **注：**
>
> 必须使用括号将 SortOrder 参数括起来。

示例

```
ListChildren([alpha])
```

如果 A、B、C 和 D 是 Z 的子代并且当前的节点为 Z，那么返回值为 A、B、C、D。

ListContains

说明

如果指定的列表包含指定的值，则返回 True。

语法

```
ListContains(List:String,Item:String,Delimiter: String):Boolean
```

List 指定要在其中进行搜索的字符串列表。

Item 指定对其执行函数的字符串值。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

示例

```
ListContains(PropValue(NodeList),Colas,[comma])
```

返回值为 True。

ListDescendants

说明

返回当前节点的后代列表，以逗号分隔。

语法

```
ListDescendants(SortOrder:String):String
```

SortOrder 指定节点返回列表的排序顺序。支持的排序顺序值：

- [hier] - 本地上下文的默认值。节点列表按照当前层次的标准层次排序顺序返回。
- [alpha] - 返回按照节点名称排序的节点列表。
- [nodeid] - 仅限用于向后兼容。节点列表按照返回列表中每个节点的节点 ID 的数字排序返回。

> **注：**
>
> 必须使用括号将 SortOrder 参数括起来。

示例

```
ListDescendants([hier])
```

如果 A、B、C 和 D 是 Z 的子代，而 Z 是 Y 的子代，并且当前节点为 Y，则返回值为 Z, A, B, C, D。

ListDistinct

说明

返回指定列表中唯一项的列表（删除了重复项）。

语法

```
ListDistinct(InputList:String,Delimiter:String):String
```

InputList 指定要使用的列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

示例

```
ListDistinct(A;B:C;A;D,[comma])
```

返回值为 A,B,C,D。

ListNodePropValues

说明

返回指定节点列表的指定属性的属性值列表。对于找不到的任何节点，在列表中返回空白字符串。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
ListNodePropValues(NodeList:String,Delimiter:String,PropAbbrev:String):
String
```

NodeList 是逗号分隔的节点名称列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

PropAbbrev 是要使用的属性的名称。

示例

```
ListNodePropValues(100;200;300,[comma],Core.Leaf)
```

如果节点 100、200 和 300 为叶节点，则返回 True,True,True。

ListNodesWith

说明

返回指定节点列表中指定表达式计算结果为 True 的节点的列表。

语法

```
ListNodesWith(NodeList:String,Delimiter:String,Expression:String):String
```

NodeList 是逗号分隔的节点名称列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

Expression 是要计算的布尔表达式。

示例

```
ListNodesWith(100;200;300,[comma],NodeIsLeaf())
```

如果节点 100、200 和 300 为叶节点，则返回 True,True,True。

ListRelatedNodesWith

说明

返回指定表达式计算结果为 True、与当前节点相关的节点的列表。

如果关系参数为 Ancestors 或 Siblings，则此函数的适用范围为本地。

语法

```
ListRelatedNodesWith(Relation:String,Expression:String,SortOrder:String,Max:I
nteger):String
```

Relation 可以为：

- Ancestors - 可以在指定的表达式中引用本地属性
- Siblings - 可以在指定的表达式中引用本地属性
- Children - 可以在指定的表达式中引用本地和全局属性
- Descendants - 可以在指定的表达式中引用本地和全局属性

Expression 是要计算的布尔表达式。

SortOrder 指定节点返回列表的排序顺序。支持的排序顺序值：

- [hier] - 本地上下文的默认值。节点列表按照当前层次的标准层次排序顺序返回。
- [alpha] - 返回按照节点名称排序的节点列表。
- [nodeid] - 仅限用于向后兼容。节点列表按照返回列表中每个节点的节点 ID 的数字排序返回。

> **注：**
>
> 必须使用括号将 SortOrder 参数括起来。

Max 是指示要返回的节点最大数目的整数值。零或无值表示没有限制并返回所有节点。

示例

```
ListRelatedNodesWith(children,True,[alpha],1000)
```

如果这些节点为当前节点的子节点，则返回 100,200,300。

ListSiblings

说明

返回当前节点的同级的列表（以逗号分隔）。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
ListSiblings(SortOrder:String):String
```

SortOrder 指定节点返回列表的排序顺序。支持的排序顺序值：

- [hier] - 本地上下文的默认值。节点列表按照当前层次的标准层次排序顺序返回。
- [alpha] - 返回按照节点名称排序的节点列表。
- [nodeid] - 仅限用于向后兼容。节点列表按照返回列表中每个节点的节点 ID 的数字排序返回。

示例

```
ListSiblings([alpha])
```

如果 A、B、C 和 D 是 Z 的子代并且当前的节点为 B，则返回值为 A, C, D。

LowerCase

说明

返回指定字符串值转换为小写字符后的结果。

语法

```
LowerCase(String:String):String
```

示例

```
LowerCase(HOBBES)
```

返回值为 hobbes。

LTrim

说明

返回指定值删除了字符串开头的所有空格后的结果。

语法

```
LTrim(String: String): String
```

示例

```
LTrim("   101203")
```

返回值为 101203。

MaxList

说明

返回指定列表中的最大项目，忽略空白项目。如果列表包含不是指定类型的项目，则返回空白字符串。

语法

```
MaxList(InputList: String,Delimiter: String,ItemType: String)
```

InputList 指定要使用的列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

ItemType 指示列表成员所需的项目数据类型。有效值：integer、float 和 datetime。默认值为 float。

示例

```
MaxList(1;2;3,[comma],Integer)
```

返回值为 3。

MinList

说明

返回指定列表中的最小项目，忽略空白项目。如果列表包含不是指定类型的项目，则返回空白字符串。

语法

```
MinList(InputList:String,Delimiter:String,ItemType:String)
```

InputList 指定要使用的列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

ItemType 指示列表成员所需的项目数据类型。有效值：integer、float 和 datetime。默认值为 float。

示例

```
MinList(1;2;3,[comma],Integer)
```

返回值为 1。

Modulus

说明

返回两个指定整数相除所得到的模数（余数）。

语法

```
Modulus(Dividend: Integer, Divisor: Integer): Integer
```

Dividend 是被除的分数的分子。

Divisor 是被除的分数的分母。

示例

```
Modulus(5,2)
```

返回值为 1。

Multiply

说明

将两个指定的整数相乘并返回结果。

语法

```
Multiply(Int1: Integer, Int2: Integer): Integer
```

示例

```
Multiply(2,5)
```

返回值为 10。

MultiplyFloat

说明

将两个指定的浮点数（浮点）相乘并返回结果。

语法

```
Multiply(Float1: Float, Float2: Float): Float
```

示例

```
MultiplyFloat(4.76,2.3)
```

返回值为 10.948。

NextSibling

说明

根据用于当前层次的排序顺序，返回当前节点的下一同级。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
NextSibling(): String
```

示例

```
NextSibling()
```

如果 A、B、C 和 D 是 Z 的子代并且当前的节点为 B，那么返回值为 C。

NodeAccessGroups

说明

返回当前用户对于当前节点的节点访问组的列表，以逗号分隔。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
NodeAccessGroups(): String
```

示例

```
NodeAccessGroups()
```

返回值为 Accounts, Finance。

NodeExists

说明

如果指定的节点存在，则返回 True。

语法

```
NodeExists(NodeAbbrev: string): Boolean
```

NodeAbbrev 是要使用的节点的名称。

示例

```
NodeExists(2000)
```

如果节点 2000 存在，则返回值为 True。

NodeInHier

说明

如果在指定的层次中存在指定的节点，则返回 True。

语法

```
NodeInHier(NodeAbbrev, HierAbbrev: string): Boolean
```

NodeAbbrev 是要使用的节点的名称。

HierAbbrev 是要使用的层次的名称。

示例

```
NodeInHier(2000,Assets)
```

如果节点 2000 位于 Assets 层次中，则返回值为 True。

NodeIsLeaf

说明

如果当前节点是叶节点，则返回 True。

语法

```
NodeIsLeaf(): Boolean
```

示例

```
NodeIsLeaf()
```

如果当前节点是叶节点，则返回值为 True。

NodeIsValidForPropertyHiers

说明

如果指定的节点满足指定属性的层次限制，则返回 True。如果属性未存储节点值或如果未针对属性定义限制，同样返回 True。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
NodeIsValidForPropertyHiers(NodeAbbrev: String, PropAbbrev: String):
Boolean
```

NodeAbbrev 是要使用的节点的名称。

PropAbbrev 是要使用的属性的名称。

NodePropValue

说明

返回指定节点的指定属性在当前层次中的值（适用于本地节点），或在当前版本中的值（适用于全局节点）。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
NodePropValue(NodeAbbrev: String, PropAbbrev: String): String
```

NodeAbbrev 是要使用的节点的名称。

PropAbbrev 是要使用的属性的名称。

示例

```
NodePropValue(2000,Abbrev())
```

返回值为 2000。

Not

说明

返回指定布尔表达式的布尔值的相反值。

语法

```
Not(Expression: Boolean): Boolean
```

示例

```
Not(NodeIsLeaf())
```

如果节点是枝节点，则返回值为 True。

Now

说明

返回当前系统日期和/或时间。

语法

```
Now([DateTimeType: String]): DateTime
```

DataTimeType 是可选的，指定要返回哪个日期部分。有效值：Date、Time、Datetime。默认值为 Date。

示例

```
Now()
```

返回当前日期和时间；例如 3/25/2010 9:20:44 AM。

```
Now(Time)
```

仅返回当前时间；例如 9:20:44 AM。

```
Now(Date)
```

仅返回当前日期；例如 3/25/2010。

NumChildWith

说明

返回当前节点中指定表达式计算结果为 True 的子代的数量。

语法

```
NumChildWith(Expression: Boolean): Integer
```

示例

```
NumChildWith(NodeIsLeaf())
```

如果节点有两个叶子代，则返回值为 2。

NumDescendantsWith

说明

返回当前节点中指定表达式计算结果为 True 的后代的数量。

语法

```
NumDescendantsWith(Expression: Boolean): Integer
```

示例

```
NumDescendantsWith(NodeIsLeaf())
```

如果节点具有两个子代并且每个子代具有 10 个叶子代，则返回值为 20。

Or

说明

如果指定的布尔表达式中有任何一个计算结果为 True，则返回 True。

语法

```
Or(Expression1, Expression2,... ExpressionN: Boolean): Boolean
```

示例

```
Or(NodeIsLeaf(),Equals(Integer,PropValue(Level),3))
```

如果当前节点是叶节点，或者位于层次中的第 3 级，则返回值为 True。

OrigPropValue

说明

使用 HasSiblingWith 或 NumDescendantsWith 函数时，返回发起节点的指定属性的值。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
OrigPropValue(PropAbbrev: String): String
```

PropAbbrev 是要使用的属性的名称。

示例

```
HasSiblingWith(GreaterThan(OrigPropValue(ID),ID())
```

如果当前节点的 ID 为 200 并且它具有任何节点 ID 大于 200 的同级，则返回值为 True。

PadChar

说明

返回使用指定填充字符加长的指定字符串。可以在原始字符串的左侧或右侧填充。生成的字符串长度至少等于指定的位数。如果原始字符串长于指定的位数，则返回原始列表。

语法

```
PadChar(String: String, PadChar: String; PadLeft: Boolean; NewLength:
Integer): String
```

String 是对其执行函数的字符串值。

PadChar 是要用于填充字符串的字符。

PadLeft 指定是否从左侧填充字符串。有效值：1、0、T、F、t 或 f。

NewLength 是指定结果长度的整数。

示例

```
PadChar(102,0,1,6)
```

返回值为 000102。

PadList

说明

返回使用指定填充字符加长的指定列表。可以在原始列表的左侧或右侧填充。生成的列表长度至少等于指定的位数。如果原始列表长于指定的位数，则返回原始列表。

语法

```
PadList(String, DelimChar, PadChr:String, PadLeft: Boolean,
NewLength:Integer): String
```

StringList 是要应用填充的字符串列表，使用指定的分隔符分隔。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在方括号中。

PadChar 是要用于填充字符串的字符。

PadLeft 指定是否从左侧填充字符串。有效值：1、0、T、F、t 或 f。

NewLength 是指定结果长度的整数。

示例

```
PadList(1;2;3;4,;,T,3)
```

返回值为 001;002;003,004。

ParentPropValue

说明

返回当前节点父节点的指定属性的值。如果节点没有父代，或者如果当前节点不是本地节点，则返回空白字符串。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
ParentPropValue(PropAbbrev: String): String
```

PropAbbrev 是要使用的属性的名称。

示例

```
ParentPropValue(Abbrev)
```

如果父节点名称为 Colas，则返回值为 Colas。

Pos

说明

使用区分大小写的搜索返回指定子字符串的第一个字符在指定字符串中的位置 (索引)。如果在字符串值内找不到该子字符串，则返回零值。

语法

```
Pos(SubString: String, String: String): Integer
```

Substring 是一个要搜索的字符串值。

String 是对其执行函数的字符串值。

示例

```
Pos(D,ABCDEFG)
```

返回值为 4。

PreviousSibling

说明

根据用于当前层次的排序顺序，返回当前节点的上一同级。

此函数的适用范围为本地，如果在全局上下文中使用，则无法正常运行。

语法

```
PreviousSibling(): String
```

示例

```
PreviousSibling()
```

如果 A、B、C 和 D 是 Z 的子代并且当前的节点为 B，那么返回值为 A。

PropControllingHier

说明

返回指定属性在当前版本中的控制层次名称。

语法

```
PropControllingHier(PropAbbrev: String): String
```

PropAbbrev 是要使用的属性的名称。

示例

```
PropControllingHier(TimeBalance)
```

返回值为 Accounts。

PropDefaultValue

说明

返回指定属性定义的默认值。

语法

```
PropDefaultValue(PropAbbrev: String): String
```

PropAbbrev 是要使用的属性的名称。

示例

```
PropDefaultValue(Currency)
```

返回值为 USD。

PropertyCategories

说明

返回当前用户的属性类别列表，以逗号分隔。

语法

```
PropertyCategories(AccessType: String) :String
```

AccessType 是属性类别的访问级别。有效值：ReadOnly、ReadWrite 或两者。

示例

```
PropertyCategories(Both)
```

返回值为 System, All, Essbase, Enterprise, HFM, Planning。

PropMaxValue

说明

返回指定属性定义的最大值。

语法

```
PropMaxValue(PropAbbrev: String): Integer
```

PropAbbrev 是要使用的属性的名称。

示例

```
PropMaxValue(Volume)
```

返回值为 10。

PropMinValue

说明

返回指定属性定义的最小值。

语法

```
PropMinValue(PropAbbrev: String): Integer
```

PropAbbrev 是要使用的属性的名称。

示例

```
PropMinValue(Volume)
```

返回值为 1。

PropValue

说明

返回当前节点指定属性的值。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
PropValue(PropAbbrev: String): String
```

PropAbbrev 是要使用的属性的名称。

示例

```
PropValue(Volume)
```

返回值为 2。

RangeListContains

说明

如果指定的范围列表包含指定的值，则返回 True。

语法

```
RangeListContains(RangeList: String, Value: Integer, Delimiter: String):
Boolean
```

RangeList 是要搜索的整数范围列表，使用指定的分隔符分隔。例如 1-100，201-300

Value 是要在范围列表中搜索的整数值。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

示例

```
RangeListContains(PropValue(MyRangeList),1,[Comma])
```

如果 MyRangeList 属性的值为 1-10, 101-10000，则返回值为 True，因为 1 包含在指定的范围中。但是，RangeListContains(PropValue(MyRangeList),11,[Comma]) 却返回 False，因为 11 未包含在指定范围内。

> **注：**
>
> 如果将 MyRangeList 更改为 "1-5,6-10,101-1000"，则 Data Relationship Management 会将此值替换为 "1-10,101-1000"，因为它会核实 RangeList 并组合具有连续边界的范围。

ReplacementAbbrev

说明

如果当前节点处于非活动状态并且指定了合并节点，则返回当前节点的替换（合并）节点名称。

语法

```
ReplacementAbbrev(): String
```

示例

```
ReplacementAbbrev()
```

ReplacePropValue

说明

如果当前节点处于非活动状态并且指定了合并节点，则返回当前节点的替换（合并）节点的指定属性值。

如果在参数中传递了本地属性，则此函数的适用范围可以为本地。

语法

```
ReplacePropValue(PropAbbrev: String): String
```

PropAbbrev 是要使用的属性的名称。

示例

```
ReplacePropValue(Description)
```

ReplaceStr

说明

返回使用新模式替换了旧模式的实例的字符串。

语法

```
ReplaceStr(String: String,OldPattern: String,NewPattern: String,ReplaceAll:
Boolean): String
```

String 是对其执行函数的字符串值。

NewPattern 是要替代所找到的字符串的字符串值。

OldPattern 是要搜索的字符串值。

ReplaceAll 指定是否使用替代字符串替代搜索字符串的所有匹配项。有效值：1、0、T、F、t 或 f。

示例

```
ReplaceStr(A1;A2;A3,A,B,T)
```

返回值为 B1;B2;B3。

RTrim

说明

返回指定值删除了字符串末尾的所有空格后的结果。

语法

```
RTrim(String: String): String
```

String 是对其执行函数的字符串值。

示例

```
RTrim("100   "))
```

返回值为 100。

SortList

说明

按照排序顺序返回指定的列表。

语法

```
SortList(InputList: String,Delimiter: String,IgnoreCase:
Boolean,ItemType: String)
```

InputList 指定要使用的列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在方括号中。

IgnoreCase 指定排序时是否忽略大小写。默认值为 False。

ItemType 指示结果列表项的目标数据类型。有效值：string、integer、float date、time 和 datetime。默认值为 string。如果有任何项目无法转换为指定的类型，则函数返回空白字符串。

StripPadChar

说明

从指定字符串的开头删除指定的填充字符，并返回修改后的值。如果原始字符串包含的填充字符少于为 StripCount 指定的值，则返回原始字符串值。

语法

```
StripPadChar(String: String, PadChar: String, StripCount: Integer):
String
```

String 是对其执行函数的字符串值。

PadChar 是要用于填充字符串的字符。

StripCount 是指定要从字符串中删除的字符数的整数。零将删除所有填充字符。

示例

```
StripPadChar(0003333,0,6)
```

返回值为 3333。

StrToBool

说明

返回基于指定字符串的布尔值。如果字符串以 Y、T 或 1（一）开头，则不管大小写或者后续的字符是什么，都返回 True 值。如果字符串以 N、F 或 0（零）开头，则不管大小写或者后续的字符是什么，都返回 False 值。

语法

```
StrToBool(String: String): Boolean
```

String 是对其执行函数的字符串值。

示例

```
StrToBool(0)
```

返回值 False。

StrToFloat

说明

返回指定字符串的浮点值。对于空格或空白字符串，返回零 (0)。

如果指定的字符串不表示浮点数，则返回错误。

语法

```
StrToFloat(String: String): Float
```

String 是对其执行函数的字符串值。

示例

```
StrToFloat(11.101)
```

返回值为 11.101。

StrToInt

说明

返回指定字符串的整数值。对于空格或空白字符串，返回零 (0)。

如果指定的字符串不表示整数，则返回错误。

语法

```
StrToInt(String: String): Integer
```

String 是对其执行函数的字符串值。

示例

```
StrToInt(101)
```

返回值为 101。

Stuff

说明

返回指定值在用指定字符串替换了指定字符后的结果。

语法

```
Stuff(PropAbbrev: String, CharsToReplace: String, ReplacementChars:
String): String
```

PropAbbrev 是要使用的属性的名称。

CharsToReplace 是要搜索的字符串值。

ReplacementChars 是要替代所找到的字符串的字符串值。

示例

```
Stuff(Abbrev(),GEO,RIO)
```

如果 Abbrev 为 GEO101，则返回值为 RIO101。

SubString

说明

返回指定字符串中从指定索引开始、包含指定字符数的一部分。

语法

```
SubString(String: String, Index: Integer, Count: Integer): String
```

SubString 是对其执行函数的字符串值。

Index 是表示搜索子字符串的起始索引位置的整数。零表示字符串中的第一个字符位置。

Count 是表示从起始索引开始要搜索的字符数的数字。

示例

```
SubString(Colas,1,2)
```

返回值为 Co。

Subtract

说明

从第一个值减去第二个整数值并返回结果。

语法

```
Subtract(Minuend: Integer,Subtrahend: Integer): Integer
```

Minuend 是一个整数值

Subtrahend 是一个整数值。

示例

```
Subtract(10,2)
```

返回值为 8。

SubtractFloat

说明

从第一个值减去第二个浮点值并返回结果。

语法

```
SubtractFloat(Minuend,Subtrahend: Float): Float
```

Minuend 是浮点值

Subtrahend 是浮点值。

示例

```
SubtractFloat(8.09,3.76)
```

返回值为 4.33。

SumList

说明

返回列表中项目的总和，忽略空白项目。如果列表包含不是指定项目类型的项目，则返回空白字符串。

语法

```
SumList(InputList: String,Delimiter: String,ItemType: String):Integer
```

InputList 指定要使用的列表。

Delimiter 是用于分隔字符串列表中项的字符。支持的特殊字符：

- [逗号]
- [空格]
- [制表符]

> **注：**
>
> 必须使用分隔符的名称（而不是字符）并将名称括在括号中。

ItemType 指示列表成员所需的项目数据类型。有效值：integer 和 float。默认值为 float。

示例

```
SumList(1;2;3,;,Integer)
```

返回值为 6。

Trim

说明

返回指定值删除了字符串开头和末尾的所有空格后的结果。

语法

```
Trim(String: String): String
```

String 是对其执行函数的字符串值。

示例

```
Trim("   101   ")
```

返回值为 101。

UpperCase

说明

返回转换为大写形式的字符串值。

语法

```
UpperCase(String: String): String
```

String 是对其执行函数的字符串值。

示例

```
UpperCase(smaller)
```

返回值为 SMALLER。

UserName

说明

返回当前用户的用户名。

语法

```
UserName(): String
```

示例

```
UserName()
```

返回值是用户名。

XOr

说明

当且仅当指定布尔表达式之一的计算结果为 True 时，才会返回 True。

语法

```
XOr(Expression1:Boolean, Expression2: Boolean): Boolean
```

示例

```
XOr(NodeIsLeaf(),Equals(Integer,PropValue(Level),3))
```

如果节点是叶节点，或者位于层次中的第 3 级，则返回值为 True。

函数组

下表按用途对函数分组。

表 11-3　函数组

函数组	函数
聚合	• AvgList • MaxList • MinList • SumList

表 11-3 （续） 函数组

函数组	函数
更改跟踪	• AddedBy • AddedOn • Changed • ChangedBy • ChangedOn • Now
比较	• Equals • GreaterThan • GreaterThanOrEqual • InRange • IsBlank • IsRangeListSubset • LessThan • LessThanOrEqual • RangeListContains
条件	• And • If • Not • Or • XOr
数据类型	• BoolToStr • FloatToStr • IntToStr • IsDataType • IsNumeric • StrToBool • StrToFloat • StrToInt
列表	• ArrayCount • ArrayIndex • ArrayItem • Intersection • ListContains • ListDistinct • ListNodePropValues • ListNodesWith • SortList

表 11-3 （续） 函数组

函数组	函数
数学	• Add • AddFloat • Divide • DivideFloat • Modulus • Multiply • MultiplyFloat • Subtract • SubtractFloat
节点	• Abbrev • ID • InternalPrefix • NodeExists • NodeInHier • NodeIsLeaf
属性	• AncestorProp • AscNodeProp • DefaultProp • Descr • DualAncestorProp • HierNodePropValue • InheritedPropOrigin • IsDefinedPropVal • NodePropValue • OrigPropValue • ParentPropValue • PropControllingHier • PropDefaultValue • PropMaxValue • PropMinValue • PropValue • ReplacePropValue

表 11-3 （续） 函数组

函数组	函数
关系	• Children • HasChildWith • HasParentNode • HasSiblingWith • InvertedLevel • IsBottomNode • IsNodeAbove • IsNodeBelow • ListAncestors • ListChildren • ListDescendants • ListRelatedNodesWith • ListSiblings • NextSibling • NumChildWith • NumDescendantsWith • PreviousSibling • ReplacementAbbrev
字符串处理	• Concat • ConcatWithDelimiter • Decode • FlipList • Format • FormattedDate • HasCharacters • IsAlpha • IsAlphaNumeric • Length • LowerCase • LTrim • PadChar • PadList • Pos • ReplaceStr • RTrim • StripPadChar • Stuff • SubString • Trim • UpperCase
用户	• NodeAccessGroups • PropertyCategories • UserName

12

管理动态脚本

通过动态脚本，您可以使用 JavaScript 为派生的属性和验证开发业务逻辑。动态脚本使用标准脚本语言，从而提供了比公式更强大、更好的执行替代方案。通过使用多个语句、变量和行内注释，脚本可提供更好的结构和更简单的逻辑。动态脚本还支持循环和正则表达式等高级概念。

执行上下文

有多种用于执行脚本的上下文：属性上下文、验证上下文和请求项属性上下文。这两种上下文定义了不同的初始参数，并返回不同类型的结果。

使用脚本的派生属性

通过脚本派生程序类，可将动态脚本用于派生属性。使用脚本的派生属性可用于版本、层次和节点。

表 12-1　属性级别说明

属性级别	参数	对象
版本	version	VersionObject
层次	hierarchy	HierarchyObject
全局节点	node	NodeObject
本地节点	node	LocalNodeObject

有关详细信息，请参阅以下主题：

- 节点派生的属性
- 版本和层次属性

节点派生的属性

在此上下文中，传递称为节点的参数。对于全局属性，该节点是 NodeObject。对于本地属性，该节点是 LocalNodeObject。派生属性的脚本必须返回一个值，该值必须对应于正在计算或执行的属性的数据类型。如果脚本返回的值与属性数据类型不匹配，则会对其进行强制转换：例如，为布尔属性返回的 null 值将被视为 false。

> **注：**
>
> 并非所有的 Oracle Data Relationship Management 属性数据类型都有 JavaScript 表示法。请参阅“数据类型转换”。

版本和层次属性

在此上下文中，可使用引用 VersionObject 的版本参数或引用 HierObject 的层次参数。计算或执行脚本时，不需要加载脚本中定义的版本。如果版本或层次派生的属性仅访问其他版本和层次级别的属性，则不管是否加载了版本，都会计算该属性。如果版本或层次派生的属性尝试访问节点级别的信息，则必须加载版本，否则计算属性时将生成错误的值。例如，如果版本级别的属性尝试获取孤立节点列表，则未加载版本时该属性将生成错误的值；加载版本后，该属性将生成正确的值。

使用脚本的验证

通过脚本验证类，可将动态脚本用于验证。有多种不同的验证级别，其中一些使用不同的参数。下面是验证级别和参数：

表 12-2　验证级别和参数

级别	参数	说明
任意级别	validation	提供有关当前执行的验证的信息
层次	hierarchy	要验证的层次的 HierarchyObject
GlobalNode	node	要验证的全局节点的 NodeObject
节点	node	要验证的节点的 LocalNodeObject
移除	node	要验证的节点的 NodeObject
移动	node	要移动的节点的 LocalNodeObject
	move	对象包含以下有关移动的信息： OldParent - 原始父代的 LocalNodeObject NewParent - 目标父代的 LocalNodeObject IsPost/IsPre - 指示此脚本在移动之前运行还是在完成移动之后运行。通常此脚本将运行两次，移动之前一次，移动之后一次。 Values - 在移动前，简单的键值对可存储在此对象中（例如 Values["key"] = "value"）。在移动后，将提供这些值，使您能够存储有关移动前状态的信息，并将其与移动后状态进行比较。所有值均将转换为字符串、数字或日期对象。目前不支持复杂对象。
合并	node	要删除或停用的节点
	merge	对象包含以下有关合并的信息： Target - 合并目标的 NodeObject IsInactivate - 如果是停用操作，则为 True IsDelete - 如果是删除操作，则为 True
版本	version	要验证的版本的 VersionObject

使用脚本的监管请求

在监管请求中可以将动态脚本与工作流任务结合使用。脚本在当前请求项的上下文中运行，并用于计算项要使用的值，例如，要更新的节点的“名称”或“父代”。

表 12-3　监管参数

参数	说明
requestitem	要计算的请求的当前 RequestItemObject

枚举常量

某些属性是对应于指定常量的数字，用于使代码更易于理解和维护。例如，对于：

`if(nodeProp.PropOrigin == 2)` 可以改用 `if(nodeProp.PropOrigin == PropOrigin.Overridden)`

属性枚举常量

- DataType - Boolean、LeafNode、Date、Time、Float、Integer、Sort、Group、Node、LimbNode、String、Hier、Version、ListGroup、MultiNode、AscNode、AscNodes、AscGroup、Memo、FormatMemo、SortProp、Property、Query、StdQuery、GlobalNode、NodeProps、RangeList、DateTime、Hyperlink 和 HierarchyGroup
- PropLevel - Node、Hier 和 Version
- PropOrigin - Default、Inherited、Overridden、InheritedHier、InheritedVer、Derived、InheritedDomain 和 Unknown
- PropType - Invalid、System、Defined、Lookup、Derived、Stats、Validation、Verification、LimbAccessGroup、LeafAccessGroup、UserSpecific、RWDerived 和 SharedInfo

验证枚举常量

- ValidationLevel - Node、Hier、Version、GlobalNodes、Merge、Move 和 Remove
- ValidationType - None、RealTime、Batch 和 Both

请求枚举常量

- WorkflowAction - AddLeaf、AddLimb、更新、停用、插入、移动、移除、删除
- WorkflowStageType - 提交、扩充、批准、最终提交
- WorkflowStatus - 无、草稿、已提交、已计算、已验证、已退回、暂挂、已分配、已申请、已升级、已降级、已驳回、已最终提交

> **注：**
>
> WorkflowStatus 枚举用于返回请求的 RequestObject.Status 当前值。但是，某些值仅限内部使用。RequestObject.Status 的有效值包括：草稿、已提交、暂挂、已申请、已升级、已驳回或已最终提交。

支持的 JavaScript 数据类型

标准 JavaScript 数据类型仍然可用，Oracle Data Relationship Management 会在任何适用的地方使用这些数据类型。例如，使用日期对象来表示日期。函数本身即为对象，通过 *new* 调用的函数可创建一个对象，其原型指向此函数的构造函数原型，正如在与 ECMA 兼容的任何 JavaScript 环境中一样。

> **注：**
>
> Data Relationship Management 脚本不支持 JavaScript 文档对象模型 (Document Object Model, DOM) 对象。

您必须熟悉 JavaScript 语法和内置对象，包括可用的方法。下面是部分可用数据类型：

- Array - 包括 length、pop、push、concat、join、reverse、slice、shift、sort 等

  > **注：**
  >
  > 由于高速缓存机制使 JavaScript 项目装箱发生了更改，因此并非所有 Array 函数都可以按预期工作或者像在以前的版本中那样工作。例如，JavaScript 中的 indexOf 将根据内存位置而不是项目的字符串或文本值来比较对象。因此，在检查数组时应考虑其他方法。IndexOf() 使用 JavaScript 中的 "===" 比较，因此没有任何可用的 "==" 定义。可以使用 JavaScript 设计模式来实现您自己的 specialIndexOf()，以提供 "==" 样式的比较。

- Boolean - 表示 True 和 False
- Date - 包括 Date.parse()、month、day、year 等
- Error - 使用 try/catch 错误处理并访问 error.message
- Function - 支持标准的调用和应用功能
- Math - 包括 random、max、pow、round、sin、cos、floor、sqrt、log 等
- Number - JavaScript 中的所有数字都是浮点型数字
- RegExp - 您可以使用针对正则表达式的语言支持，也可以显式访问它们
- String – 包括 concat、indexOf、lastIndexOf、substr、split、splice、search、replace、toUpperCase、toLowerCase 等

还提供了全局可用的函数，如 parseInt、parseFloat、isNaN、decodeURI、encodeURI。

Print 函数

通过 Print 函数可以在创建脚本时输出调试信息。结果将显示在脚本编辑器的“警告”部分中。尽管 Print 函数在测试上下文中仅生成输出，但是引擎仍必须构造参数；因此，在保存脚本以供生产环境使用之前，请注释掉所有 print 语句。

Format 函数

Format 函数提供了比标准 JavaScript 更为丰富的字符串格式设置机制。第一个参数是包含格式指定符（括在大括号中）的字符串。通过使用两个大括号来将其转义，例如 "{{" 在输出中会变为 "{"。格式指定符从零开始并逐个递增。如果在序列中省略了某个指定符，则 Format 函数的等效参数将被忽略。例如，"{1}" 将忽略 Format 的第一个值参数而使用第二个值参数。

有一个快捷方式。您可以调用 Format，传递一个不带大括号的格式指定符，并仅传递一个参数。其结果等效于 Format("{0:<*specifier*>", <*argument*>)

格式指定符的工作原理与其他语言（如 Java 或 C#）类似。其语法为 {<*paramnum*>} 或 {<*paramnum*>;<*format*>}，其中，*paramnum* 是个正整数，从零开始并按顺序递增。格式参数取决于使用该参数所传递的对象的类型。

格式参数通常会返回与用户的区域设置相对应的值，例如，在美国，"{0:0.00}" 会返回 "1.23"，而在欧洲，它将返回 "1,23"。也可以使用转义支持来显式覆盖区域设置，对所有用户都输出相同的值。例如，"#\,###\,##0" 将在所有区域中使用逗号作为千位分隔符来格式化数字（不管使用哪种语言设置）。

数据类型转换

并非所有 Oracle Data Relationship Management 属性定义数据类型在 JavaScript 中都有对应的表示法。对于无对应表示法的任何数据类型，StringValue 和 Value 将相同，您必须确保自己了解字符串值如何解析。如果返回的属性值是这些数据类型之一，您还应负责确保返回该数据类型的相应字符串表示法。如果存储的值没有针对该属性的数据类型的有效转换，则该值为未定义值。

在列表属性返回的数组中，每个元素都包含与该数据类型相对应的类型的对象。例如，使用日期属性标记的列表将返回包含日期对象的数组。

在以下情况下，查找属性可能并不总是返回预期的数据类型：查找目标无效、在查找表中未找到键，或者查找表中的值对该数据类型无效。例如，如果键值对的值为 "TEST"，但数据类型为 Date，则结果将为未定义。

以下是 Data Relationship Management 数据类型及其在 JavaScript 中的对应表示法。

表 12-4　数据类型比较

属性定义数据类型	JavaScript 数据类型
AscGroup	NodeObject 数组
AscNode	NodeObject
AscNodes	NodeObject 数组
Boolean	Boolean
Date	Date
DateTime	DateTime
Float	Number
FormatMemo	String
GlobalNode	NodeObject
Group	String 数组
Hier	HierObject

表 12-4 （续） 数据类型比较

属性定义数据类型	JavaScript 数据类型
Hierarchy Group	String（层次组名称）
Hyperlink	String（表示 URL）
Integer	Number
LeafNode	LocalNodeObject
LimbNode	LocalNodeObject
ListGroup	String 数组
Memo	String
MultiNode	LocalNodeObject 数组
Node	LocalNodeObject
NodeProps	PropDefObject 数组
Query	String（查询名称）
Property	PropDefObject
Sort	Number
SortProp	PropDefObject
StdQuery	String（查询名称）
String	String
Time	String
Version	String（版本名称）

在调用 JavaScript 派生的其他属性（或其他节点的派生属性）时，因为该派生程序返回的值不会立即转换为其字符串表示法，所以，您可以在派生程序之间传递复杂对象，并将强制转换延迟到返回最终结果之后，可通过对该复杂对象调用 *toString()* 来进行转换（除非指出进行内置转换，例如，从数组转换）。

设置数字的格式

只能通过单个快捷方式字符（例如 "G"）或指定符的组合（例如 "##0,000.0"）来设置数字的格式。如果尝试在大于一个字符的格式指定符中使用快捷方式字符，该字符将按原样复制到输出中（视为文字字符）。

在生产环境中运行导出时请选择相应的语言设置，以确保输出格式正确。

表 12-5 单字符快捷方式数字格式

格式	说明
D	整数（可感知区域设置，对负数使用负号）
D<precision>	至少 <precision> 位的整数，缺少的位以零填充。例如，设置为 "{0:D5}" 的 123 将输出为 00123。
E	指数（科学）表示法 "1.234E+10"

表 12-5 （续） 单字符快捷方式数字格式

格式	说明
F	浮点数 "123.456"（可感知区域设置的小数分隔符和对负数使用负号）
F<precision>	浮点数，小数点后舍入为 <precision> 位有效数字
G	常规数字格式
N	常规数字格式 "123,456.789"（可感知区域设置的分组/小数分隔符和对负数使用负号）
N<precision>	常规数字，小数点后舍入为 <precision> 位数字
P	百分比（对于 0.20146 将输出 "20.14%"，可感知区域设置的分组/小数分隔符和对负数使用负号）
P<precision>	百分比，舍入为 <precision> 位有效数字（对于 0.205 "{0:P0}" 将输出 "21%"）
X	十六进制输出 "4D2"

表 12-6 数字格式指定符

格式	说明
0	零占位符，如果存在数字，将其输出，否则为零
#	数字占位符，如果存在数字，将其输出，否则不输出
.	特定于区域设置的小数分隔符
,	当放在两个占位符之间时，将输出特定于区域设置的分组分隔符（对于 123456789 "{0:#,#}" 将输出 "123,456,789"）。当一个或多个逗号紧挨着小数点（或隐式小数点）左侧放置时，对于每个逗号，将数字除以 1000（对于 123456789 "{0:#,##0,,}" 将输出 "1,235"）。
%	将数字乘以 100，并输出特定于给定位置的区域设置的百分比符号
E<sign>0	指数表示法。至少需要一个零，零的数量用于指定指数中的最小位数。<sign> 是可选的，可以是以下项： • + （始终根据需要输出符号 +/-） • - （仅对负数输出 - 符号）
\<char>	转义字符（<char> 被视为字符输出）

表 12-6 （续） 数字格式指定符

格式	说明
;	部分分隔符。如果出现，允许为正数、负数和零定义不同的格式。 • 一个部分 "{0:#,#;}" - 与没有部分相同 • 两个部分 "{0:#,#;-#,0}" - 第一部分适用于正数和零，第二部分适用于负数 • 三个部分 "{0:#,#;-#,0;zero}" - 第一部分适用于正数，第二部分适用于负数（如果为空，则第一部分也适用于负数），第三部分适用于零
任何其他字符	按原样复制到输出

设置日期的格式

可通过单个快捷方式字符（例如 "G"）或指定符的组合（例如 "HH:mm"）来设置日期的格式。如果要使用单个字符作为正式指定符而非快捷方式，请为字符串加上前缀 %。例如，"%m" 将输出未填充的分钟，而非月和日。

表 12-7 单字符快捷方式日期格式

格式	说明
t	短时间 "4:05 PM"
T	长时间 "4:05:07 PM"
d	短日期 "3/9/2013"
D	长日期 "Friday, March 09, 2013"
f	长日期和短时间 "Friday, March 09, 2013 4:05 PM"
F	长日期和长时间 "Friday, March 09, 2013 4:05:07 PM"
g	短日期和短时间 "3/9/2013 4:05 PM"
G	短日期和长时间 "3/9/2013 4:05:07 PM"（默认）
m	月和日 "March 09"
y	月和年 "March, 2013"
r	RFC 1123 "Fri, 09 Mar 2013 16:05:07 GMT"
s	可排序的日期/时间 "2013-03-09T16:05:07"
u	通用可排序的日期/时间 "2013-03-09 16:05:07Z"

表 12-8　日期格式指定符（多个字符）

格式	说明 示例为 **2013-04-05 04:07:09 PM CST**
yy	年 "13"
yyyy	年 "2013"
M	月 "4"
MM	月 "04"
MMM	月 "Apr"
MMMM	月 "April"
d	日 "5"
dd	日 "05"
ddd	日 "Sun"
dddd	日 "Sunday"
h	12 小时制 "4"
hh	12 小时制 "04"
H	24 小时制 "16"（如果是 4 AM，则为 "4"）
HH	24 小时制 "16"（如果是 4 AM，则为 "04"）
m	分 "7"
MM	分 "07"
s	秒 "9"
ss	秒 "09"
f	几分之一秒（对于较高精度可重复 1-4 次）
F	末位不为零的几分之一秒（可重复 1-4 次）
t	AM 或 PM 指示符 "P"（对于只有 24 小时制的语言设置为空）
tt	AM 或 PM 指示符 "PM"（对于只有 24 小时制的语言设置为空）
z	GMT 偏移 "-6"
zz	GMT 偏移 "-06"
zzz	GMT 偏移 "-06:00"
:	时间分隔符（特定于区域设置）
/	日期分隔符（特定于区域设置）
\<char>	转义字符（<char> 被视为字符输出），例如： "{0:HH\h}" 将输出 "16h"
任何其他字符	按原样复制到输出

Data Relationship Management 对象

下面将介绍 Oracle Data Relationship Management 对象及其方法和属性。

SysObject

会自动创建一个称为 Sys 的 SysObject。此对象在所有上下文中均可用，它提供常规函数以及有关 Data Relationship Management 应用程序的信息。此对象没有属性。

表 12-9　SysObject 方法

名称	说明
FormattedDate (value, formatString)	按照公式系统规则设置日期的格式。用于实现向后兼容，可精确匹配旧公式的属性。 • value 必须是日期对象或有效的日期时间字符串 • formatString 必须是有效的格式字符串（请参阅 "FormattedDate" 函数）
GetNextID(key)	返回指定字符串键值的下一个可用整数 ID。
GetPropDef(abbrev)	针对给定的属性名称返回 PropDefObject。此名称必须是完全限定名称。
GetRequestByID(int)	按 ID 返回工作流请求。
GetSysPrefValue(abbrev)	返回给定系统首选项的值（例如 HierNodeSeparator）
InRange(dataType, input, min, max, minExclusive, maxExclusive)	对应于公式函数 InRange。必需参数为 dataType、input 和 min。
IsNodeAbove(ancestor, child)	如果祖先在层次中位于子代之上，则返回 True。如果参数不是 LocalNodeObject，或者不在同一个层次中，则返回 False。
IsNodeBelow(descendant, parent)	如果后代在层次中位于父代之下，则返回 True。如果参数不是 LocalNodeObject，或者不在同一个层次中，则返回 False。

表 12-9 （续） SysObject 方法

名称	说明
RunFormula(node, propDef, formulaString)	运行 Data Relationship Management 公式并返回字符串结果 • node 为 NodeObject 或 LocalNodeObject。当在 NodeObject 中传递公式字符串时，该公式字符串不能引用本地属性，否则会出错。传递 LocalNodeObject 时，您可以引用所有可用的全局属性和本地属性。 • propDef – 某些公式函数需要有属性定义才能正确解析或执行。使用这些函数时，您必须提供属性定义。一般而言，属性定义特征（如级别、全局与本地以及类型）必须匹配，但无需是 formulaString 所针对的实际属性。它们可以不相关。在大部分公式中，您可以为此参数传递空值。语法为 `Sys.GetPropDef(abbrev)`。例如： `Sys.RunFormula(node, Sys.GetPropDef("Custom.MyProp1"), "Concat(Prop value ', PropValue(Custom.MyProp2),' ,is, ,valid)");` • formulaString 是旧的 Data Relationship Management 公式；空格被视为此公式的文字部分，因此必要时必须删除空格。 **注意：**此操作并不是最佳做法，应仅在需要时使用，以实现与旧行为的完全匹配。使用此方法时性能会降低。

PropDefObject

此对象没有方法。

表 12-10 PropDefObject 属性

名称	说明
Abbrev	属性定义名称（包括完全限定命名空间）
Cascade	如果属性值是继承的，则为 True
ColumnWidth	默认导出列宽
DataType	DataType 枚举值，例如 DataType.String（请参阅“枚举常量”）
Descr	说明
DefaultValue	属性定义的默认值。类型取决于属性定义的数据类型。
EditorLabel	标签

表 12-10 （续） PropDefObject 属性

名称	说明
Global	如果属性为全局节点属性，则为 True
Hidden	如果属性在属性网格中是隐藏的，则为 True
ID	ID
Level	PropLevel 枚举值，例如 PropLevel.Node（请参阅“枚举常量”）
List	如果属性允许用户从值列表中进行选择，则为 True
ListValues	由多个值组成的数组，用户可从中进行选择
LookupValues	查找属性的查找键值对。使用此数组中对象的 Key 和 Value 属性。
MaxValue	最大值
MinValue	最小值
Namespace	属性定义的命名空间
PropType	PropType 枚举值，例如 PropType.Defined（请参阅“枚举常量”）
PropClass	派生程序类（Formula 或 Script）
ReadOnly	如果属性为只读（例如 Core stats 属性），则为 True

VersionObject

表 12-11 VersionObject 属性

名称	说明
Abbrev	名称
Descr	说明
HierCount	层次数
ID	ID
NodeCount	节点数

表 12-12 VersionObject 方法

名称	说明
GetHierarchies()	获取由该版本中当前用户可见的所有层次组成的数组
GetGlobalNodes()	获取由该版本中所有全局节点 (NodeObject) 组成的数组
GetOrphans()	获取由该版本中所有孤立项 (NodeObject) 组成的数组
HierByAbbrev(abbrev)	按名称获取 HierarchyObject

表 12-12 （续） VersionObject 方法

名称	说明
HierByID(id)	按 ID 获取 HierarchyObject
NodeByAbbrev(abbrev)	按名称获取 NodeObject
NodeByID(id)	按 ID 获取 NodeObject
NodeExists(abbrev)	如果存在使用给定名称的全局节点，则返回 True
Prop(abbrev)	获取该版本的给定属性的 NodePropObject
PropValue(abbrev)	获取该版本的给定属性的值。返回值类型取决于属性定义的数据类型。

HierarchyObject

表 12-13 HierarchyObject 属性

名称	说明
Abbrev	名称
Descr	说明
HierarchyUrl	层次 URL
ID	ID
NodeCount	层次中的节点数
SharedNodesEnabled	如果已启用共享节点，则为 True
TopNode	LocalNodeObject 顶级节点
Version	VersionObject
VersionAbbrev	版本名称
VersionID	版本的 ID

表 12-14 HierarchyObject 方法

名称	说明
NodeByAbbrev(abbrev)	按名称获取 NodeObject
NodeByID(id)	按 ID 获取 NodeObject
NodeExists(abbrev)	如果存在使用给定名称的本地节点，则返回 True
Prop(abbrev)	获取该版本的给定属性的 NodePropObject
PropValue(abbrev)	获取该版本的给定属性的值。返回值类型取决于属性定义的数据类型。

共有节点属性和方法

尽管 NodeObject 和 LocalNodeObject 的原型链不同，但是某些属性和方法是这两个对象共有的。

在各种情况下，值可能会因全局或本地上下文而有所不同，但均可针对该上下文返回正确的值。例如，针对 NodeObject 调用 GetChildren() 时，生成的数组将包含 NodeObject。在针对 LocalNodeObject 进行相同的调用时，生成的数组将包含 LocalNodeObject。

表 12-15　NodeObject 和 LocalNodeObject 的共有属性

名称	说明
Abbrev	Core.Abbrev
AddedBy	Core.AddedBy
AddedOn	Core.AddedOn
Changed	Core.Changed
ChangedBy	Core.ChangedBy
ChangedOn	Core.ChangedOn
ChildNodeCount	子节点数
Descr	Core.Descr
DomainAbbrev	Core.DomainAbbrev
DomainNodeAbbrev	Core.DomainNodeAbbrev
ID	Core.ID
Inactive	Core.Inactive
IsPrimary	如果节点为共享节点的主节点，则为 True；如果节点不是共享节点或者不是主节点，则为 False
IsShared	如果节点为共享节点，则为 True
Leaf	Core.Leaf
NodeApproved	Core.NodeApproved
Version	节点的所有者 VersionObject
VersionAbbrev	节点的版本名称
VersionID	节点的版本 ID

表 12-16　NodeObject 和 LocalNodeObject 的共有方法

名称	说明
GetChildren(sorted)	获取由此节点的直接子代组成的数组（可以选择对数组进行排序）。sorted 的默认值为 False。
GetDescendants(inclusive, sorted)	获取由此节点的后代组成的数组（可以选择包含此节点和/或对数组进行排序）。inclusive 的默认值为 True。sorted 的默认值为 False。
NodeByAbbrev(abbrev)	按名称获取 NodeObject
NodeByID(id)	按 ID 获取 NodeObject
NodeExists(abbrev)	如果存在使用给定名称的全局节点，则返回 True
Prop(abbrev)	获取该版本的给定属性的 NodePropObject

表 12-16 （续） NodeObject 和 LocalNodeObject 的共有方法

名称	说明
PropValue(abbrev)	获取该版本的给定属性的值。返回值类型取决于属性定义的数据类型。

LocalNodeObject

Oracle 建议您使用各种 *xxxWith* 函数定位层次中的其他节点。例如，执行 *ChildrenWith* 要比调用 GetChildren() 并迭代结果快得多。同样，执行 *GetReferenceInHier* 要比调用 GetReferences() 并迭代结果快得多且更易于使用。

表 12-17 LocalNodeObject 属性

名称	说明
GlobalNode	当前节点的全局 NodeObject
Hier	节点所在层次的 HierarchyObject
HierAbbrev	Core.HierAbbrev
HierID	Core.HierID
Level	表示层次中节点级别的数字
MissingPrimary	如果未找到主节点，则为 True
NodeUrl	节点 URL
Parent	此节点的父节点的 LocalNodeObject。如果是层次的顶级节点，则返回 Null。
ParentNodeAbbrev	父节点的名称
Primary	此共享节点的主节点。如果主节点不在此层次中，则返回出现主节点的第一个层次中的主节点。如果需要主节点所在的层次的列表，请对返回的主节点调用 GetReferences()。如果未能找到共享节点或主节点，则返回 null。
PrimaryNotInHier	如果主节点存在但不在此层次中，则为 True

表 12-18　LocalNodeObject 方法

名称	说明
AncestorsWith(func, maxResults, searchFromTop, inclusive)	在祖先链中搜索满足给定函数的节点。这是定位祖先的最快方法。返回由 LocalNodeObject 结果组成的数组。 • func 必须为使用单个节点参数的函数，如果结果中包含节点，则返回 True；如果未通过测试，则返回 False。 • maxResults 是可选的，默认值为 1。使用 0 可表示不受限制（满足条件的所有节点）。 • searchFromTop 是可选的，默认值为 False。使用 True 可从层次的顶级开始。 • inclusive 是可选的，默认值为 False。使用 True 可在潜在的匹配中包括当前节点（它必须通过测试）。
ChildrenWith(func, maxResults)	在节点的子代列表中搜索满足给定函数的节点。这是查找子代的最快方法。返回由 LocalNodeObject 结果组成的数组。 • func 必须为使用单个节点参数的函数，如果结果中包含节点，则返回 True；如果未通过测试，则返回 False。 • maxResults 是可选的，默认值为 1。使用 0 可表示不受限制（满足条件的所有子代）。
DescendantsWith(func, maxResults, inclusive, depthFirst)	在后代链中搜索满足给定函数的节点。这是查找后代的最快方法。返回由 LocalNodeObject 结果组成的数组。 • func 必须为使用单个节点参数的函数，如果结果中包含节点，则返回 True；如果未通过测试，则返回 False。 • maxResults 是可选的，默认值为 1。使用 0 可表示不受限制（满足条件的所有节点）。 • inclusive 是可选的，默认值为 False。使用 True 可在潜在的匹配中包括当前节点（它必须通过测试）。 • depthFirst 是可选的，默认值为 True。如果为 True，则会对每个分支进行全面检查（到其顶端），然后再返回树移至下一分支。如果为 False，则会首先检查节点的所有子代，然后检查每个子代的节点，以此类推。如果已经清楚地知道节点在树中的位置，则可通过在此处选择正确的值来大大加快搜索速度。
GetAncestorEnumerator()	获取枚举祖先节点的 NodeEnumeratorObject
GetAncestors(inclusive)	获取由 LocalNodeObject 祖先组成的数组
GetChildEnumerator(sorted)	获取枚举子节点的 NodeEnumeratorObject。如果 sorted 为 True，则将对子代进行排序。

表 12-18 （续） LocalNodeObject 方法

名称	说明
GetDescendantEnumerator()	获取枚举后代节点的 NodeEnumeratorObject
GetImplicitly SharedDescendants(inclusive)	获取与此共享节点相关的主节点的子节点
GetInvertedLevel()	等同于 InvertedLevel 公式函数
GetReferences()	在出现此节点的所有层次中获取由引用此节点的 LocalNodeObject 组成的数组
GetReferenceInHier(hierAbbrev)	在给定层次中获取此节点的引用。如果层次不可访问或此节点不在该层次中，则结果将为 null。
NextSibling()	按排序顺序获取此节点的下一同级
PreviousSibling()	按排序顺序获取此节点的前一同级
SiblingsWith(func, maxResults, inclusive)	在节点的同级中搜索满足给定函数的节点。返回由 LocalNodeObject 结果组成的数组。 • func 必须为使用单个节点参数的函数，如果结果中包含节点，则返回 True；如果未通过测试，则返回 False。 • maxResults 是可选的，默认值为 1。使用 0 可表示不受限制（满足条件的所有祖先）。 • inclusive 是可选的，默认值为 False。使用 True 可在潜在的匹配中包括当前节点（它必须通过测试）。

NodePropObject

表 12-19 NodePropObject 属性

名称	说明
Abbrev	属性定义的名称
ControllingHierarchy	此版本中属性定义的控制层次的 HierarchyObject。如果属性不是全局节点属性、没有控制层次或者找不到控制层次，则返回值将为 null。
Locked	如果值处于锁定状态，则为 True
Origin	PropOrigin 枚举值，例如 PropOrigin.Overridden（请参阅“枚举常量”）
Owner	与此值关联的对象（VersionObject、HierarchyObject、NodeObject 或 LocalNodeObject）
PropType	PropType 枚举值，例如 PropType.Defined（请参阅“枚举常量”）

表 12-19 （续） NodePropObject 属性

名称	说明
StringValue	此属性的原始字符串值。如果是 Derived 或 RWDerived 属性，此值可能为属性定义默认值或覆盖的值。
Value	此属性的解释值（例如，对于 DataType.Float 和 DataType.Integer，此值将为 Number 对象）。并非所有数据类型都必须具有非字符串表示法。

表 12-20 NodePropObject 方法

名称	说明
GetPropDef()	获取节点属性的 PropDefObject

RangeListObject

RangeListObject 表示由多个值组成的 RangeList，可用于检查 RangeList 属性而无需手动解析字符串。还可构造新的 RangeListObject 以从相应数据类型的派生属性返回值。

构造函数示例

```
var x = new RangeListObject();


var y = new RangeListObject("1-10,20-25");


var z = new RangeListObject([{start:1, end:10},{start:20, end:25}]);
```

表 12-21 RangeListObject 构造函数参数

参数	可选	说明
ranges	True	用于初始化的范围值。此参数为可选。接受两种格式： • 数组 - 其中的每个元素均为一个具有指示范围的起始属性和结束属性的对象。数组中没有这些属性的所有对象都将被忽略。 • 字符串 - 以逗号分隔的字符串条目列表。每个条目均包含以短划线 (-) 或等号 (=) 符号分隔的起始值和结束值。

表 12-22 RangeListObject 属性

名称	说明
Ranges	由对象组成的数组。每个对象均具有两个属性： • start - 范围条目的开始 • end - 范围条目的结束 此属性为只读属性。要修改范围，请使用以下方法。

表 12-23 RangeListObject 方法

名称	说明
AddRange(start, end)	向范围列表添加新的范围。此方法可扩展现有范围条目或创建新的范围条目。要向列表中添加单个数字，请同时对 start 参数和 end 参数使用此数字。如有必要，可将这两个参数强制设为整数。
Contains(value)	如果值在范围列表中，则返回 True，否则将返回 False。 如有必要，可将 value 强制设为整数。
IsSupersetOf(range)	如果当前 RangeListObject 为给定 RangeListObject 的超集，则返回 True。传送其他类型的对象将出错。
RemoveRange(start, end)	从列表中删除一个范围。此删除操作可能会将现有范围条目拆分为两个，或删除整个条目。要从列表中删除单个数字，请同时对 start 参数和 end 参数使用此数字。如有必要，可将这两个参数强制设为整数。

NodeEnumeratorObject

通过 NodeEnumeratorObject 对节点列表执行操作效率更高。枚举器不会一次构建整个列表，而是根据需要一次仅获取一个节点。如果要查找的内容在列表的中间位置，则可弃用枚举器。返回由节点对象组成的数组的属性和方法必须立即构建整个数组，无论要访问的项目是否位于数组的结尾。

在开始处，枚举器得到的当前值为 null。必须调用 MoveNext() 才能使枚举器进入列表中的第一个节点。

> **注：**
>
> 当需要从所有可能的匹配中仅查找少数几个节点并且只需迭代列表一次时，最好使用 *With* 方法，例如 *AncestorsWith* 或 *SiblingsWith* 方法。如果需要循环祖先节点列表多次，或者已知道将需要大多数或所有祖先，则枚举器可能更快。

表 12-24　NodeEnumeratorObject 方法

名称	说明
GetCurrent()	当前节点（根据上下文确定是 NodeObject 还是 LocalNodeObject）。
MoveNext()	使枚举器进入下一节点。如果没有更多可枚举的节点，将返回 False。

ValidationObject

表 12-25　ValidationObject 属性

名称	说明
Abbrev	验证的名称（包括完全限定命名空间）
Descr	说明
EditorLabel	标签
Cascade	如果验证分配是继承的，则为 True
ValidationClass	验证类的名称
ValidationLevel	ValidationLevel 枚举值，例如 ValidationLevel.Node（请参阅“枚举常量”）。
ValidationType	ValidationType 枚举值，例如 ValidationType.Batch（请参阅“枚举常量”）。

验证脚本

- 验证脚本返回包含名为 "success" 的属性的 JavaScript 对象。如果该脚本返回布尔值或非布尔值对象（例如，数字或字符串），则将使用 JavaScript 转换规则将其值转换为布尔值，然后分配给 success。该脚本可以选择在属性中返回命名为 "parameters" 的、由多个值组成的 JavaScript 数组。将通过字符串替代将数组值替代到验证的失败消息中。
- 可以返回布尔值（True 或 False）。如果返回 True，则表明验证成功；否则表明验证失败。如果未返回值，则视为与返回 False 相同。
- 如果返回的是非布尔值对象（例如数字或字符串），则会将该对象转换为布尔值再返回。将应用标准 JavaScript 转换。等于零的数字、空字符串以及 null 或未定义的对象将被解释为 false。其他所有值均为 true。
- 如果返回包含命名为 "success" 的属性的复杂对象，则会将该 success 属性转换为布尔值，并用作验证的返回值。可以选择在属性中返回命名为 "parameters" 的、由多个值组成的数组。这是 JavaScript 数组对象，需要进行填充，然后用于参数化的失败消息中。将通过字符串替代将参数替代到验证的失败消息中。应返回与失败消息中的占位符相对应的正确数量的值。如果返回了额外的参数，将忽略该参数。如果未返回足够的参数，则缺失的参数将被视为空字符串。

RequestObject

RequestObject 表示监管请求，包括请求标题和项。“项”属性表示添加到请求的请求项的列表。一个重要的属性是“版本”属性，请求的目标版本包括其层次和节点，全部均可通过相关脚本对象访问。

表 12-26 RequestObject 属性

名称	说明
ID	ID
Title	请求的标题
Version	请求的目标版本
ModelName	请求的工作流模型
StageName	请求的当前阶段
StageType	WorkflowStageType 枚举值，例如 WorkflowStageType.Submit（请参阅“枚举常量”）
Status	WorkflowStatus 枚举值，例如 WorkflowStatus.Submitted（请参阅“枚举常量”）
Items	添加到请求的 RequestItemObject 的列表

RequestItemObject

RequestItemObject 表示监管请求的单个请求项，包括有关当前任务和要更新的节点的信息以及该项的详细信息（属性值）。通过“请求”属性可以访问项的全部请求对象，包括标题属性和其他项。

NodeNamePendingInRequest 方法用于确定与目标版本的其他正在进行的请求冲突的潜在节点名称，如果其他暂挂请求上的项包含同一节点名称的“添加”项，则返回 True。

表 12-27 RequestItemObject 属性

名称	说明
ItemID	项 ID
RequestID	请求 ID
Request	该项所属的请求对象
NodeName	要更新的节点的 Core.Abbrev
说明	要更新的节点的 Core.Descr
HierarchyName	要更新的节点的层次
ParentName	要更新的节点的 Core.Parent
TaskName	请求项的工作流任务名称
TaskAction	WorkflowAction 枚举值，例如 WorkflowAction.AddLimb（请参阅“枚举常量”）
TaskDomain	工作流任务的域名（如果有）

表 12-27 （续） RequestItemObject 属性

名称	说明
ItemDetails	请求项的 RequestItemDetailObject 的列表

表 12-28 RequestItemObject 方法

名称	说明
NodeNamePendingInRequest(name)	接受要测试的特定节点名称参数。如果除版本的当前请求外正在进行的请求包含具有指定名称的 AddLimb/叶项，则返回 True。

RequestItemDetailObject

RequestItemDetailObject 表示监管请求的单个请求项详细信息，对应于单个属性值。

表 12-29 RequestItemDetailObject 属性

名称	说明
CalcValue	属性的计算值
HasCalcValue	如果值为计算值，则返回 True
Modified	如果请求中修改了该值，则返回 True
PropertyName	属性的名称
Value	属性的值

执行环境

Oracle Data Relationship Management 引擎是一个多线程、多计算机的环境，脚本可在多个线程以及多台计算机上同时执行。虽然您可以创建值并将其存储在全局范围内，但不应依赖于此行为，因为在其他线程上执行脚本时不会提供该全局值。同样，全局值不会在多个 Data Relationship Management 引擎实例或计算机内进行更新。此外，由于 Data Relationship Management 支持多个活动版本，因此如果您针对某个节点计算了值并将该值存储在全局范围内，则在其他脚本访问另一个节点的该属性时可能生成错误的值。

> **注：**
>
> 基于同一原因，您不应在全局范围内存储变量，还应避免修改 Data Relationship Management 内置的对象原型，因为无法确保修改应用于所有引擎实例和线程。

设置脚本超时

为防止出现过多的引擎锁，将基于超时设置终止执行时间太长而未返回值的脚本。可以针对每个属性定义和验证设置脚本超时。

超时是基于每个执行上下文的，因此，如果要导出 100 个节点的脚本属性，且该属性的超时设置为 30 秒，则导出最多需要 50 分钟，因为每个节点只能花费 30 秒来计算其属性。但是，如果脚本属性调用了其他脚本属性，则其超时值不会增加。例如，如果 PropA 的超时值为 10 秒，PropB 的超时值为 20 秒，且 PropA 调用了 PropB，而后者将启动一个运行很长时间的计算过程，则在 10 秒之后，PropA 的计算将终止，因为已超出其原始超时值。

防止无限循环

脚本进入无限循环（也称为堆栈溢出）是一种严重错误，这会导致服务器进程意外终止。虽然 Data Relationship Management 会尝试阻止此类脚本执行，但是在编写自我引用或递归脚本时，务必要小心谨慎。在将新脚本部署到生产环境之前，要始终先在开发环境中对其进行测试。

下面是一个将进入无限循环的脚本的简单示例。因为该脚本包含了对自身的调用，但是从不终止执行，所以，执行此函数的引擎最终将由于缺乏资源而终止。而且，因为该脚本从未调用 Data Relationship Management 引擎，所以就没有机会捕获溢出并停止脚本。

```
function badFunc(a) { badFunc(a); }


badFunc("oops");
```

性能注意事项

要实现最佳性能，请避免在脚本中引用公式派生的属性，反之亦然。鉴于本机硬件（包括 64 位处理器）的即时 (just-in-time, JIT) 编译等因素，与公式相比，脚本通常最有可能进行性能调优。还可以根据实际执行特征通过 JIT 编译器对脚本进行调优，而且随着时间的推移脚本运行速度将加快。

创建动态脚本

可在脚本编辑器（位于“参数”选项卡）中为派生的属性定义和验证创建动态脚本。

监管工作流任务期间计算“名称”或“父代”时，也可使用脚本编辑器。

> **注：**
>
> 计算父代名称时，任何特殊字符的使用都必须符合有关对特殊字符进行转义的标准 JavaScript 规则。有关详细信息，请参阅《*Oracle Data Relationship Management* 用户指南》中的“命名节点”。

要创建动态脚本：

1. 在文本区域中输入脚本。

> **注：**
>
> 要插入属性，请将光标置于脚本中，然后单击**插入属性**。将显示属性的列表。选择属性，然后单击**确定**。

2. 从以下选项中进行选择：
   - **脚本超时** - 脚本超时之前的秒数。
   - 要使用选定节点计算脚本，请单击  并选择节点。将在脚本中使用该节点的当前属性值。单击**计算**。结果显示在脚本设计器的底部。
3. 要测试脚本，请单击**计算**。

13

管理节点类型

节点类型允许根据层次节点的关系和属性以不同的方式查看和管理这些节点。特定节点类型的节点共享相同的特性：

- 属性
- 验证
- 图标

一个层次可以具有不同节点类型的节点，并且相同的节点在不同的层次中可以是不同的节点类型。节点类型的使用示例包括 GL 帐户、成本中心、合并实体、产品组、预测点等等。

要按节点类型对节点进行分类：

1. 确定在层次中对节点进行分类所需的节点类型。
2. 识别与每个节点类型相关（或不相关）的属性。
3. 识别与每个节点类型相关（或不相关）的验证。
4. （可选）为每个节点类型指定一个图标。

定义节点类型

要定义节点类型：

1. 在主页上，选择**管理**。
2. 从**新建**中，选择**节点类型**。
3. 输入节点类型的名称和说明。
4. **可选**：选择要用于节点类型的图标
5. 在**属性**选项卡上，从“可用”列表中选择要与节点类型关联的属性。使用箭头将属性移动到**已选中**列表中。
6. 在**验证**选项卡上，从“可用”列表中选择要与节点类型关联的验证。使用箭头将验证移动到**已选中**列表中。
7. 单击**保存**。

编辑节点类型

要编辑节点类型：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**节点类型**。
3. 选择节点类型并单击  。
4. 执行以下任意操作：

- 编辑说明。
- 更改要用于节点类型的图标
- 添加或删除属性
- 添加或删除验证

5. 单击**保存**。

删除节点类型

要删除节点类型：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**节点类型**。
3. 选择节点类型并单击  。
4. 单击**删除此项**以确认删除。

使用节点图标

图标是与节点类型关联的图像，在 Oracle Data Relationship Management 用户界面中显示为节点的图标。可以创建新图标和修改现有图标。还可以删除不再使用的图标。图标必须以 PNG 格式提供。

要添加节点图标：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**图标**。
3. 输入图标的名称并添加说明。
4. 单击**浏览**并选择 PNG 文件。
5. 单击**上传**。
6. 单击**保存**。

要修改节点图标：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**图标**。
3. 选择一个图标并单击  。
4. 单击**浏览**以选择其他 PNG 文件。
5. 单击**上传**。
6. 单击**保存**。

要删除图标：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**图标**。
3. 选择一个图标并单击  。
4. 单击**删除此项**以确认删除。

14

使用系统首选项

系统首选项允许管理用户编辑用于控制 Oracle Data Relationship Management 的行为的设置。

系统首选项

下表介绍了 Oracle Data Relationship Management 系统首选项。

表 14-1　系统首选项

系统首选项	类型	说明
AllowAsOf	布尔值	如果设置为 True，则强制捕获核心操作并创建基线版本以允许创建截止版本。如果此首选项设置为 False，则无法创建截止版本。 默认值为 True。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
AllowNextIDGeneration	布尔值	如果设置为 True，则允许自动生成下一 ID。 默认值为 False。
AllowNextIDKeyCreation	角色	允许在 NextID 功能中创建新键的角色列表。 默认值为交互式用户、数据创建者、数据管理员。
AllowPru	布尔值	如果设置为 True，则将启用修剪选项，以允许非管理员用户删除具有子代的节点。如果为 False，则非管理员用户无法删除具有子代的节点。 默认值为 True。
AllowRelaxedMove	布尔值	如果设置为 True，移动节点时，允许新父代优先于该节点在其他层次中任何与新父代冲突的父代关系。 默认值为 False。
AllwSpac	布尔值	如果设置为 True，则允许节点名使用空格。 默认值为 True。
AnalyticsNodeCountUpdateTime	字符串	使用 24 小时格式以本地时间指定一天中的某个时间，此时将更新所有已加载、标准版本的版本和层次的节点计数。例如，2:15 PM 将输入为 "1415"。默认时间为 3:00 AM。

表 14-1 （续） 系统首选项

系统首选项	类型	说明
ApprovalGroups	字符串	以逗号分隔的审批组列表。
ApprovalGroupTrackProperties	字符串	按组跟踪的审批属性的分隔列表。
ApprovalPropertyByApprovalGroup	字符串	按审批组列出的全局布尔型审批属性。
AuthMethod	字符串	用户身份验证方法： • 内部 – 仅在 Data Relationship Management 内部对用户进行身份验证。 • CSS（外部）– 仅在外部对用户进行身份验证。需要对 Shared Services 具有访问权限。 • 混合 - 根据各个用户的设置在内部或外部对用户进行身份验证。 默认值为“内部”。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
CopyLcl	布尔值	如果设置为 True，则在复制节点时复制本地值。 默认值为 True。
DefaultCurrentVersion	版本	默认当前版本。此首选项可以使用版本的“设为默认值”选项进行设置。
DefaultPreviousVersion	版本	默认早期版本。此首选项可以使用版本的“设为默认值”选项进行设置。
DefaultPropCopyMode	字符串	默认属性复制模式。 有效值为“已覆盖”、“已选择”和“强制全部”。 默认值为“已覆盖”。
EnablePropCopyOptions	角色	允许访问属性复制选项的角色列表。 默认值为交互式用户、数据创建者、数据管理员。
EnforceListProps	布尔值	如果设置为 True，则仅允许使用预定义列表中的值来更新列表属性。 默认值为 True。
FiltrChr	字符串	导出的“输出选项”屏幕上用于 Replace 函数的字符集。

表 14-1 （续） 系统首选项

系统首选项	类型	说明
FindByProperties	属性	浏览层次时可用作搜索条件的属性列表。 显示的属性是用户有权访问的属性。另外，显示的属性可能并非适用于所有层次。 **注意：**ADMIN 用户无法添加到 Data Relationship Management 中的自定义属性类别。因此，如果 FindByProperties 系统首选项中列出的某个属性未添加到 ADMIN 已是所属成员的属性类别，则 ADMIN 将无法在层次浏览窗口中使用该属性执行查找。
FindWildCardAppend	布尔值	如果设置为 True，则会在未选择"完全匹配"时，为查找条件附加一个星号 (*)。 默认值为 False。
FindWildCardPrepend	布尔值	如果设置为 True，则会在未选择"完全匹配"时，为查找条件预先附加一个星号 (*)。 默认值为 False。
GlobalPropLocalOverride	属性	要从全局属性的本地检查中排除的属性列表。在启用 GlobalPropLocalSecurity 时会使用这些属性。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
GlobalPropLocalSecurity	布尔值	如果设置为 True，则会在全局属性上强制实施本地安全性。系统会在节点所在的所有层次中对照用户的本地安全性（节点访问级别）检查全局属性发生的更改。 默认值为 False。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
HierSep	字符串	层次和节点的分隔符。 默认值为颚化符号 (~)。
IdleTime	整数	应用程序服务器的会话超时时间（以分钟为单位）。 默认值为 60。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。

表 14-1 （续） 系统首选项

系统首选项	类型	说明
Inactivate	角色	允许停用节点的用户角色列表。 默认值为所有角色。
InactiveChanges	角色	允许更改已停用节点的角色列表。 默认值为数据管理员、应用程序管理员、访问管理员。
InvDescr	字符串	节点说明属性的无效字符列表。
InvName	字符串	节点名称的无效字符列表。 **注意：**此列表中的字符无法用作共享节点的分隔符。
JobResultsMaxSize	整数	对于使用“客户端文件”选项运行的作业，结果的最大大小（以字节为单位）保存到作业历史记录中。超过此大小的作业结果不保存到作业历史记录中。默认值为 10,000,000 字节。负值指示所有结果（无论大小是多少）均保存到作业历史记录中。 **警告：**强烈建议不要通过设置为负值来禁用 JobResultsMaxSize，因为这可能严重影响大型作业的性能。 **注意：**JobResultsMaxSize 不适用于使用“服务器文件”或“数据库表”选项运行的导出。
JobResultsRetentionAge	整数	将归档作业结果详细信息保留在历史记录中的天数。零值表示永远不从历史记录中清除作业结果。 **注意：**清除作业结果以管理数据库大小。禁用清除可能导致数据库随时间显著增大。
LeafEdit	角色	允许更改“叶”属性的角色列表。 默认值为“数据管理员”、“数据创建者”、“应用程序管理员”、“访问管理员”。
LockoutInactivity	整数	锁定用户之前的最大停用天数。 默认值为 30；零表示无最大值。
LockoutInvalidLogins	整数	锁定用户之前的最大无效登录次数。 默认值为 6；零表示无最大值。
LossLevel	字符串	要捕获的丢失级别。 有效值包括： • 已定义 • 全部 默认值为“已定义”。如果选择“全部”，则在捕获带有许多属性值的已删除节点时可能会显著影响系统性能。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。

表 14-1 （续） 系统首选项

系统首选项	类型	说明
LRUPropertyCacheSize	整数	LRU 属性高速缓存的最大大小。LRU 属性高速缓存存储可以多次访问的计算值。通常，应使用此首选项的默认值，不应更改。
MaxDescr	整数	节点说明的最大字符数。有效值为 12 到 255。 默认值为 80。
MaxLeaf	整数	叶名称的最大字符数。有效值为 3 到 20。 默认值为 255。
MaxLimb	整数	枝名称的最大字符数。有效值为 3 到 20。 默认值为 255。
NodeApprovedSecurity	角色	允许查看和更新节点的 NodeApproved 系统属性的角色列表
PasswordDuration	整数	用户密码的有效天数。有效值为 1 到 9999。 默认值为 30。
PasswordMaxLength	整数	用户密码的最大字符数。有效值为 0 到 255。零表示无最小值。 默认值为零。
PasswordMinLength	整数	用户密码的最小字符数。有效值为 0 到 9999。零表示无最小值。 默认值为 6。
PasswordPolicyEnabled	布尔值	如果设置为 True，则要求密码包含下列元素中的三种元素： • 大写字母 • 小写字母 • 数字 • 特殊字符 默认值为 True。
PasswordWarningPeriod	整数	指示密码过期日期前 (-) 或后 (+) 多少天警告用户更改其密码的正数或负数，超过这段期限将禁止他们登录。有效值为 -30 到 30。 默认值为 1。
RenameLeaf	角色	允许重命名叶节点的角色列表。 默认值为数据管理员、应用程序管理员、访问管理员。
RenameLimb	角色	允许重命名枝节点的角色列表。 默认值为所有角色。

表 14-1 （续） 系统首选项

系统首选项	类型	说明
ReqMerge	布尔值	如果设置为 True，则要求在启用 UseMerge 时合并停用或删除的节点。 默认值为 False。
SharedNodeDelimiter	字符串	指定节点名和共享节点后缀之间的分隔符。 不应在影响节点名称的任何地方使用 SharedNodeDelimiter 字符。 默认值为冒号 (:)。 **警告：**当设置 SharedNodeDelimiter 和 SharedNodeSequenceSeparator 系统首选项时，必须对它们使用不同的字符。例如，如果 SharedNodeDelimiter 为冒号，则 SharedNodeSequenceSeparator 字符不能是冒号。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
SharedNodeIdentifier	字符串	指定共享节点分隔符后面使用的标识符。 默认值为“共享”。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
SharedNodeMaintenanceEnabled	布尔值	如果设置为 True，则启用共享节点。 默认值为 False。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
SharedNodeNamingType	字符串	指定共享节点的备用名。有效值包括：“后缀”或“前缀”。 默认值为“后缀”。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
SharedNodeSequenceLength	整数	指定使用数字序列类型时唯一键的长度。 默认值为 3。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。

表 14-1 （续） 系统首选项

系统首选项	类型	说明
SharedNodeSequenceSeparator	字符串	指定要放在共享节点标识符之后的分隔符。 默认值为短划线 (-)。 **警告：**当设置 SharedNodeDelimiter 和 SharedNodeSequenceSeparator 系统首选项时，必须对它们使用不同的字符。例如，如果 SharedNodeDelimiter 为冒号，则 SharedNodeSequenceSeparator 字符不能是冒号。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
SharedNodeSequenceType	字符串	指定唯一键的类型。有效值包括“数字”或“祖先”。 默认值为“数字”。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
SortLimbsFirst	布尔值	如果设置为 True，则要求先对枝节点进行排序，然后再对叶节点进行排序。如果为 False，则可以同时对枝节点和叶节点进行排序。此首选项影响层次导出、显示和节点列表。 默认值为 True。
TopNodeParentString	字符串	在导入和导出中用于指示顶级节点的父代值。 默认值为“无”。
TransactionLevels	字符串	要捕获的事务级别列表。打开截止或者指定结果或丢失操作会强制捕获核心操作。 有效值包括： • 记录的操作 • 核心操作 • 结果操作 • 丢失操作 **注意：**不管此系统首选项是什么，始终记录管理级别的事务。 默认值为“记录的操作”、“核心操作”、“结果操作”、“丢失操作”。 **注意：**更改此首选项需要重新启动 Data Relationship Management 应用程序。
UpName	布尔值	如果设置为 True，则节点名始终使用大写字母。 默认值为 False

表 14-1 （续） 系统首选项

系统首选项	类型	说明
UseChangeApproval	布尔值	如果设置为 True，则将启用更改审批。 默认值为 False。
UseMerge	布尔值	如果设置为 True，则允许对停用和删除的节点使用合并方法。 **注意：**如果 ReqMerge 为 True，则系统要求指定合并节点。如果 ReqMerge 为 False，则合并节点是可选的，除非节点已批准属性为 True。节点已批准属性在以下情况下设置为 True，版本已最终确定，或者具有适当访问权限的用户专门将其设置为 True。 默认值为 False。
ValSec	布尔值	如果设置为 True，则将检查节点访问组安全性以确定用户是否可以对节点运行批量验证。 默认值为 False。
WarnHL	整数	“后代”、“子代”、“查询结果”等列表显示的最大节点数。最小值为 1000。如果设置为小于 1000 的值，则显示 1000 个节点。 默认值为 5000。

有关详细信息，请参阅以下主题：

- 设置事务历史日志记录级别
- 设置更改审批

设置事务历史日志记录级别

您必须具有应用程序管理员权限才能设置 Oracle Data Relationship Management 事务历史日志记录级别。设置 TransactionLevels 系统首选项以指定要在事务历史中捕获的操作类型。

全局属性的本地安全性

使用两个系统首选项（GlobalPropLocalSecurity 和 GlobalPropLocalOverride）控制全局属性的本地安全性。

要设置事务历史日志记录级别：

1. 在 Data Relationship Management Web 客户端中，选择**管理**。
2. 在**元数据**下，展开**系统首选项**，并编辑 **TransactionLevels** 首选项。
3. 在 **TransactionLevels** 中，选择事务级别类型：
   - **记录的操作**记录基本的日志信息，例如用户登录和注销。

- **核心操作**记录更改版本、层次或节点信息的操作，例如添加节点、更改属性或移动节点。
- **结果操作**记录源于核心操作的操作。例如，如果执行了“清除以下所有项”核心操作，则会清除各个节点上的属性。清除各个节点上的属性就是结果操作。
- **丢失操作**记录因为核心操作而发生的数据丢失。例如，删除节点时，会删除该节点的定义属性，这就是一个丢失操作。丢失操作由 LossLevel 系统首选项控制。

> **注：**
>
> 如果指定了“丢失操作”或者启用了 AllowAsOf 系统首选项，则会跟踪核心操作，即使未在 TransactionLevels 系统首选项中设置也是如此。

4. 设置 LossLevel 首选项：
   - **已定义** – 删除节点时仅跟踪在节点上专门设置的值。
   - **所有项** – 在 LossAction 中跟踪派生的、默认的和继承的值。
5. 停止并重新启动应用程序，或重新启动 Data Relationship Management 服务。

设置更改审批

Oracle Data Relationship Management 中的更改审批系统允许您定义审批组，并将它们关联到由一组属性或特殊操作触发的审批标志。这样，普通用户可以执行更改，审批者可以运行查询，然后根据需要设置审批标志。

Data Relationship Management 中确定更改审批行为的系统首选项包括：

- UseChangeApproval - 设置为 True 将启用更改审批。
- ApprovalGroups - 系统中使用的审批组名称的逗号分隔列表。
- ApprovalGroupTrackProperties - 如果 UseChangeApproval 为 True，则定义跟踪的属性，这些属性是此组的审批标志更改为 False 的触发器。格式为 xxx[a,b,c],yyy[d,e,f]... 其中 xxx 和 yyy 是 ApprovalGroups 首选项中定义的销售组，而 a、b、c、d、e、f 是属性名称。例如 Sales[Custom.SalesGroup,{NodeMove}],Treasury[Custom.AccountDescription,{NodeAdd}]。

  可包含在属性列表中的特殊操作有：

  – {NodeAdd} - 对添加的节点触发“需要审批”机制。
  – {NodeInactivate} - 对已停用的节点触发“需要审批”机制。
  – {NodeReactivate} - 对重新激活的节点触发“需要审批”机制。
  – {NodeInsert} - 对插入的节点触发“需要审批”机制。
  – {NodeRemove} - 对删除的节点触发“需要审批”机制。
  – {NodeMove} - 对移动的节点触发“需要审批”机制。
- ApprovalPropertyByApprovalGroup - 如果 UseChangeApproval 为 True，则定义全局布尔属性，以便在任何触发器属性发生更改或使用特殊操作时设置为 False。格式为 xxx:bbbb,yyy:cccc…其中 xxx 和 yyy 是 ApprovalGroups 首选项中定义的销售组，而 bbbb 和 cccc 是用于存储组审批标志的全局布尔属性的名称，例如 Sales:Custom.SalesApprovedFlag,Treasury:Custom.TreasuryApprovedFlag。

配置系统首选项

要配置系统首选项：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**系统首选项**。
3. 选择一个系统首选项，然后单击  。
4. 修改值，然后单击**保存**。

15

使用外部连接

应用程序管理员可以定义和配置外部文件系统、数据库和 Web 服务的通用连接。导入、导出和集可以共享文件和数据库连接，以最大程度地减少连接信息的维护工作。数据库和 Web 服务连接可以配置有外部操作以查找外部系统中的数据或者向外部系统最终提交数据更改。外部连接允许应用程序服务器直接访问这些资源、从中读取或向其中写入数据。

> **注：**
>
> 在定义外部连接之前，必须首先设置外部资源。

外部操作

可以为 Web 服务或数据库外部连接定义外部操作。外部操作将配置为查找或最终提交。查找操作从外部系统读取数据。最终提交操作向外部系统写入数据。数据库和 Web 服务连接可以支持多个操作。有关详细信息，请参阅“外部最终提交”和“外部查找”。

定义外部连接

要定义外部连接：

1. 在主页上，选择**管理**。
2. 从**新建**中选择**外部连接**。
3. 输入名称和说明。
4. 从**对象访问**中，选择**标准**、**系统**或一个自定义组。
5. 选择连接类型：**服务器文件**、**FTP**、**数据库**或 **Web 服务**。
6. 执行下列操作之一：

   - 如果选择了**服务器文件**，请输入服务器的 UNC 路径，然后单击  。

     > **注：**
     >
     > Oracle Data Relationship Management 应用程序服务器使用的 Windows 用户帐户会自动用于服务器文件连接。**Oracle DRM 服务器进程** Windows 服务使用的默认 Windows 用户帐户为**本地系统帐户**。用于服务的帐户必须能够访问 UNC 路径以进行正确的服务器文件连接。此外，UNC 路径必须具有适当的权限以便服务帐户读取和写入文件。

   - 如果选择了 **FTP**，请输入以下信息：
     - 主机服务器

- 用户 ID
- 用户密码
- 单击  。

- 如果选择了**数据库**：
  - 选择**数据访问提供程序**：Oracle、SqlServer 或 OleDb。
    - 输入数据库连接超时值
    - 输入数据库命令超时值
  - 输入**连接字符串**。
  - 输入您的用户 ID 和密码

> **注：**
>
> 要建立可写入的外部连接，管理员必须具有 SELECT、INSERT 和 DELETE 访问权限。仅有 SELECT 访问权限的用户可以建立到表和视图的只读外部连接。

  - 单击  。
  - 在**允许的对象**选项卡上，要筛选较大列表，请执行以下任意操作：
    - 选择或输入架构/所有者，如果需要可以使用通配符。
    - 输入对象名称，如果需要可以使用通配符。
    - 选择**包括视图**以包括权限至少为 SELECT 的视图。请注意，视图始终为只读。
    - 选择**包括只读表**以包括权限至少为 SELECT 但不包括 INSERT 和 DELETE 的表。
    - 单击  ，然后从**可用**列表中选择对象。使用箭头将对象移到**已选中**列表中。
    - **可选：**要使用**快速添加**部分，请输入要添加的对象的架构/所有者和名称，然后单击箭头将其移动到**已选中**列表。
  - 要添加外部操作，请单击**外部操作**选项卡，再单击**添加**，然后执行以下操作：
    - 输入操作的**名称**。该名称对于父代外部连接必须唯一。
    - 输入**说明**文本，说明操作的用途。
    - 选择**操作类型** -“查找”或“最终提交”。此选择用于筛选可以使用“外部查找”和“外部最终提交”功能进行选择的操作的列表。
    - 选择**数据库操作类型** -“语句”或“存储过程”。
      - 如果选择了**语句**，则单击**添加**，然后执行以下操作：
        - 输入调用该操作时要传递的参数：
          - **参数名称** - 参数的名称。不允许有空格。

        - **参数说明** - 参数的说明
        - **测试值** - 用于测试操作的值。将存储该值以便重复使用。
      - 在 **SQL 语句**字段中，输入要执行的单个 SQL 语句。可以在 SQL 语句中使用替代参数来传递运行时值。替代参数格式为 <%ParamKey%>，其中 <% 和 %> 表示替代参数，ParamKey 是要用于替代的参数的名称。例如，<%TopNode%>。
      - 单击  测试操作。“回滚”选项将回滚脚本对数据库进行的所有更改。默认情况下选择“回滚”。测试操作时，参数的测试值将插入语句并执行。单击**结果**选项卡查看测试结果。
    - 如果选择了**存储过程**：
      - 输入要执行的**存储过程名称**，可以包括程序包名称作为前缀。
      - 输入操作的**名称**。该名称对于父代外部连接必须唯一。
      - 输入**说明**文本，说明操作的用途。
      - 查看存储过程的参数列表。为“结果参数”选择 True 以在 Data Relationship Management 操作结果中返回参数。仅一个参数可以选作结果参数。仅为“查找”操作返回结果参数。对于“最终提交”操作，仅指示成功或失败。
      - **测试值** - 用于测试操作的值。将存储该值以便重复使用。
      - 单击  测试操作。“回滚”选项将回滚存储过程对数据库进行的所有更改。默认情况下选择“回滚”。测试操作时，参数的测试值将插入存储过程并执行。单击**结果**选项卡查看测试结果。
- 如果选择了 **Web 服务**：
  - 选择**协议**："HTTP" 或 "HTTPS"。
  - 输入**主机名**
  - 输入**端口** - 如果指定了端口 0，标准端口 80 和 443 将分别用于 HTTP 和 HTTPS
  - 选择**身份验证类型** - 如果设置为“基本”，则可以保存“用户 ID”和“密码”。
  - 输入**用户 ID** 和**密码**。
  - 要添加外部操作，请单击**添加**，然后执行以下操作：
    - 输入操作的**名称**。该名称对于父代外部连接必须唯一。
    - 输入**说明**文本，说明操作的用途。
    - 选择**操作类型** -“查找”或“最终提交”。此选择用于筛选可以使用“外部查找”和“外部最终提交”功能进行选择的操作的列表。
    - 在**请求**选项卡上，单击**添加**，然后输入调用该操作时要传递的参数：
      - **参数名称** - 参数的名称。不允许有空格。
      - **参数说明** - 参数的说明
      - **测试值** - 用于测试操作的值。将存储该值以便重复使用。
    - 在 **HTTP 操作**中，选择 "GET"、"POST"、"PUT" 或 "DELETE"。

> **注：**
>
> 仅 "POST" 和 "PUT" 允许发送“HTTP 正文”内容。

- 输入 Web 服务消息的 HTTP **URI**。
- 输入 **HTTP 标头**的原始内容。
- 输入 **HTTP 正文**的文本内容。
- **响应**选项卡 - 显示 Web 服务操作的完整传出和传入消息。传出消息中使用的参数会将其测试值插入请求中。Web 服务返回的传入消息的 HTTP 正文应该为 XML 或 JSON 格式。对于外部查找操作，传入消息需要转换为表格格式（行和列）以用于外部查找属性。要处理此转换，可以使用 XPath 表达式。“列表标识符表达式”参数标识传入消息中作为结果集中的行的元素。“结果列”标识显示为结果集中的列的行元素的属性。

  要预览“列表标识符表达式”和“结果列”配置的结果，请单击**预览**选项卡。结果显示在数据网格中。

可以在 "URI"、“HTTP 标头”和“HTTP 正文”中使用替代参数向外部操作传递运行时值。替代参数格式为 <%ParamKey%>，其中 <% 和 %> 表示替代参数，ParamKey 是要用于替代的参数的名称。例如，<%TopNode%>。

要测试配置，请单击  。“HTTP 请求”将构建并发送到端点。用户界面自动切换到**响应**选项卡并显示完整传出消息和传入响应。传出消息中使用的参数会将其测试值插入请求中。

7. 单击  验证选定项目，以核实可通过连接用户名和密码在相应的级别访问它们。
8. 单击  保存外部连接。

编辑外部连接

要编辑外部连接：

1. 在主页上，选择**管理**。
2. 在**元数据**下，展开**外部连接**。
3. 选择外部连接并单击  。
4. 根据需要进行更改。
5. 单击  保存外部连接。

删除外部连接

删除外部连接时，使用该连接的所有导入和导出配置文件都会受到影响。

要删除外部连接：

1. 从主页中选择**管理**。
2. 在**元数据**下，展开**外部连接**。

3. 选择外部连接并单击  。
4. 选择**删除此项**以确认删除。

16

配置监管工作流

监管工作流是格式化的流程，用于控制对节点、关系和属性值的更改的输入、审批、验证和最终提交。

由应用程序管理员定义用于管理业务用户提交的更改请求和数据管理者提交的修复请求的工作流任务和工作流模型。

建议您阅读《*Oracle Data Relationship Management* 用户指南》中的“监管工作流”，以了解监管工作流概念的其他信息。

管理工作流任务

工作流任务是用户针对请求上下文中的本地节点执行的一组更改。请求中的请求项由工作流任务控制。

工作流任务由操作类型、用户说明、要查看或编辑的属性以及验证组成。工作流任务的操作类型指定执行的基本操作类型，例如添加、移动或更新节点。每个操作类型都定义有关节点和父代的选择规则、属性更新的应用以及验证和最终提交请求时要执行的操作。

> **注：**
>
> 工作流请求中不支持以下操作：

- 合并节点
- 废除节点
- 重新激活节点
- 插入孤立节点
- 添加域节点（其中域不同于父代）

任务属性

可以配置工作流任务属性以控制为请求项显示的属性，属性是否可编辑，以及是否为必需值。可根据需要配置可编辑属性。不能从任务中删除某个操作类型的默认属性。

任务和属性说明

可以在请求页面上添加说明以帮助指导用户完成请求项的创建、扩充和审批。可以为工作流任务及其属性定义说明。当在“提交”、“扩充”或“最终提交”阶段查看请求项时，将为请求项的发起任务显示任务说明。将显示对分配给“批准”或“扩充”阶段的“更新”工作流任务的任务说明，而不显示发起任务说明。可以为单个请求项的属性显示任务属性说明。

超链接可以包含在任务和属性说明中。URL 可以直接插入到说明字段中，或者 URL 可以使用语法 `[url=http_URL]URL_Title[/url]`，其中 `http_URL` 指定超链接文本，而 `URL_Title` 指定

向用户显示的文本。例如，示例 [url=http://support.oracle.com]Oracle Support[/url] 在属性网格中会显示为 Oracle Support。

任务验证

任务验证是可选的节点级别验证，必须首先对请求项成功执行此验证，然后才能针对特定的工作流阶段提交或批准请求。配置为以批处理模式运行的验证可用于以任务验证方式进行选择。任务验证可以与任务属性关联，以便将验证消息与可能需要更正的特定属性链接。

计算的名称和父代属性

工作流任务中使用的“名称”和“父代”属性可以确定要进行更改的节点和层次位置。这些属性的值通常由用户手动定义或从源文件中加载。工作流任务可用的“计算名称”和“计算父代”选项可用于使用动态脚本计算这些属性的值，而不必显式定义或加载这些值。

“计算名称”选项可用于使用“添加叶节点”或“添加枝节点”操作类型的工作流任务。“计算父代”选项可用于这些任务以及“插入”和“移动”任务。脚本的计算逻辑可以访问以下数据源：

- NextID 函数
- 请求的版本属性
- 版本中的层次及其属性
- 节点及其属性
- 节点之间的层次关系
- 请求的属性
- 请求项及其属性
- 请求项任务及其操作类型

在以下阶段中计算监管请求时执行对“名称”和“父代”属性的计算：使用启用了这些选项的工作流任务添加请求项。可在请求项的发起阶段或之后的阶段（已配置为重新计算这些属性）重新计算这些值。

> **注：**
>
> 如果某个工作流模型设置为允许“重新计算任务属性”并且手动覆盖了计算的“名称”或“父代”，则在该阶段或任何后续阶段中将不会再次计算“名称”或“父代”。

外部最终提交

可以在工作流任务上选择性地配置外部最终提交，以便在最终提交监管请求时将该请求中的已批准更改立即同步到外部目标系统。例如，外部操作可以运行用于插入、更新或删除数据的 SQL 语句，它也可以调用 SOAP 或 REST Web 服务来在外部系统中创建、更新或删除数据。在 Oracle Data Relationship Governance 中使用外部最终提交时，可以在成功最终提交 Data Relationship Governance 请求后启动外部数据更新。使用为数据库和 Web 服务连接定义的外部操作访问外部数据源。

成功最终提交 Data Relationship Governance 请求后，可以按照每项的任务的配置执行每项的外部操作。

- 操作按照项和任务定义的顺序同步执行。
- 操作在请求项的本地节点上下文中执行，允许输出参数基于可能未为该任务选择的属性。
- 如果外部操作过程中发生错误，错误消息将添加到请求项（作为“外部最终提交失败”）。
- 在每个外部操作后，使用成功或失败信息更新请求活动。
- 如果为外部操作定义了“最终提交状态”属性，则如果操作完成时没有错误该属性将更新为 True，如果操作完成时有错误该属性将更新为 False。
- 如果任何外部操作未成功完成，都将通知数据管理员和“最终提交”阶段参与者。

创建工作流任务

要创建工作流任务：

1. 在主页上，选择**管理**。
2. 从**新建**中，选择**工作流任务**。
3. 输入工作流任务名称。
4. 从**操作类型**中，为任务选择操作类型：
   - **添加叶节点** – 添加具有全局和本地属性的叶节点
   - **添加枝节点** – 添加具有全局和本地属性的枝节点
   - **删除** – 更新节点的全局/本地属性并删除节点
   - **停用** – 更新节点的全局和本地属性并停用节点
   - **插入** – 向层次中插入节点并更新其全局/本地属性
   - **移动** – 将节点移到其他父代并更新其全局/本地属性
   - **移除** – 更新节点的全局/本地属性并移除节点
   - **更新** – 更新节点的全局和本地属性

     > **注：**
     >
     > 如果用户打算从文件将项上传到请求，则需要在任务中定义以下属性（以及用户要上传的文件）：
     >
     > – 对于“添加”操作：指定“名称”、“父代”和“说明”
     > – 对于“插入”操作：指定“名称”和“父代”
     > – 对于“移动”操作：指定“名称”和“父代”

   - **重新激活** - 更新节点的全局和本地属性，并重新激活停用的节点。
5. **可选**：执行以下任一任务：
   - 在**说明**字段中输入用户的文本。

     URL 可以直接插入到说明字段中，或者 URL 可以使用语法 `[url=http_URL]URL_Title[/url]`，其中 `http_URL` 指定超链接文本，而 `URL_Title` 指定向用户显示的文本。例如，示例 [url=http://support.oracle.com]Oracle Support[/url] 在属性网格中会显示为 Oracle Support。
   - 选择要对其筛选的层次组。

> **注：**
>
> 为工作流任务选择的层次组与为工作流模型配置的层次组属性一起使用，可以筛选可用于选择任务的层次。

- 为“添加枝节点”或“添加叶节点”任务选择一个**域**。

> **注：**
>
> 为工作流任务配置的域必须与使用该任务的请求项的目标版本所使用的域匹配。如果版本没有使用该任务的域，则请求项节点无法添加到版本。

> **注：**
>
> 如果域已分配，则需要“添加枝节点”和“添加叶节点”任务的说明属性。

6. 在**属性**选项卡上，从**可用**列表中选择要分配给任务的属性。使用箭头将属性移动到**已选中**列表中。使用向上箭头和向下箭头对属性进行排序。
7. 单击属性的 以更新以下选项：
   - **可编辑** - 选择此选项将允许编辑属性。
   - **必需** - 选择此选项将使属性成为必需属性。
   - **计算** - 对于“添加枝节点”或“添加叶节点”任务，选择该选项可以根据动态脚本计算名称值。如果选择，则“名称”属性的“可编辑”选项为 "False"，被禁用。选择此选项时，**计算名称**选项卡会变为可用，然后您可以输入用于计算“名称”值的脚本。

     要使用动态脚本为“添加枝节点”、“添加叶节点”、“移动”和“插入”任务计算“父代”值，请单击父节点旁边的 ，然后选择**计算**。如果选择，则“父代”属性的“可编辑”选项为 "False"，被禁用。选择此选项时，**计算父代**选项卡会变为可用，然后您可以输入用于计算“父代”值的脚本。有关编写动态脚本的信息，请参阅“管理动态脚本”。
   - **自定义标签** - **可选：**输入属性的备用标签。该标签显示在项目详细信息上的属性标签列中。
   - **属性说明** - **可选：**输入特定的属性说明。不可编辑的属性也可添加说明。说明在项详细信息中显示在属性值上面。

     URL 可以直接插入到说明字段中，或者 URL 可以使用语法 `[url=http_URL]URL_Title[/url]`，其中 `http_URL` 指定超链接文本，而 `URL_Title` 指定向用户显示的文本。例如，示例 [url=http://support.oracle.com]Oracle Support[/url] 在属性网格中会显示为 Oracle Support。

   单击 保存更改，或者单击 取消更改。
8. 在**验证**选项卡上，从**可用**列表中选择要分配给任务的验证。使用箭头将验证移动到**已选中**列表中。

9. 单击  将验证与特定的任务属性关联。如果所选验证失败，则将显示指定属性的验证消息。

   单击  保存更改，或者单击  取消更改。

10. 如果您选择了计算名称或父代，则选择**计算名称**或**计算父代**选项卡，然后执行以下操作：
    - 输入动态脚本以计算名称或父代。有关编写动态脚本的信息，请参阅“创建动态脚本”。
    - 输入以下信息：
      - **请求 ID** - 指定在计算脚本时使用的请求 ID。
      - **请求项编号** - 指定在计算脚本时使用的请求项编号。
      - **脚本超时** - 脚本超时之前的秒数。
    - **可选：**选择**隐藏**指定正在计算的名称或父代的隐藏属性。如果选择，则计算的名称或父代不在请求项详细信息中显示。
    - 单击**计算**。结果显示在脚本设计器的底部。

11. **可选：**选择**外部最终提交**选项卡，单击**添加**，然后配置以下设置：
    - **外部连接** - 选择外部连接
    - **操作** - 选择要执行的外部操作

      > **注：**
      >
      > 该操作必须已定义为连接中的“最终提交”类型操作。

    - 对于每个外部操作参数，配置：
      - **参数源类型** - 选择“文字”或“属性”
      - **源** - 如果为源类型选择了**文字**，则在“参数源”列中输入文字值。调用外部操作时，为当前参数传递文字值。如果为源类型选择了**属性**，则选择属性来为外部操作提供参数值。执行“外部最终提交”时，参数值来自当前节点或请求项中的选定属性。
    - **最终提交状态属性** - 选择“布尔型”属性以指示节点是否具有任何外部最终提交错误。将为请求的目标版本中的节点设置此属性。在外部最终提交失败时，可以使用此属性标识未成功最终提交到外部系统的版本中的更改。

12. 单击  保存工作流任务。

编辑工作流任务

创建工作流任务后，可以编辑该任务的属性和验证列表。保存任务后，将无法修改工作流任务的操作类型。

为工作流任务添加或删除属性、将属性从可编辑更改为只读或者重新排序时，现有请求的请求项属性将受影响。将不再为使用任务的请求项显示删除的任务属性。为所用的任务属性已从可编辑更改为只读的请求项定义的属性值将被丢弃。

要编辑工作流任务：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流任务**。

3. 选择一个任务，然后单击  。
4. 在**属性和验证**选项卡上，更改属性和验证选择。
5. 单击  。

复制工作流任务

可以通过复制现有任务来创建工作流任务。将会复制操作类型、属性和验证，并且在保存之前可以对其进行编辑。

要复制工作流任务：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流任务**。
3. 右键单击要复制的任务，然后选择**复制**。
4. 为任务输入新名称。
5. 对任务进行任何其他更改，然后单击  以保存工作流任务。

删除工作流任务

如果某个工作流任务不再分配到已分配给某个更改请求的任何模型，则可删除该工作流任务。如果将某个任务分配到无法删除的模型，则无法删除该任务。

要删除工作流任务：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流任务**。
3. 选择一个工作流任务，然后单击  。
4. 单击**删除工作流任务**以确认删除。

管理工作流模型

工作流模型定义指定类型的一组更改管理任务，这些任务可基于该模型包含在单个请求中。模型定义将更改最终提交到版本中之前所需执行的一组审批和扩充步骤。

工作流阶段

工作流阶段针对每个工作流模型进行定义，无法在工作流模型之间共享。

阶段类型

为工作流模型分配阶段后，阶段类型属性定义用户在该工作流阶段中的参与类型。

表 16-1　工作流阶段类型

工作流阶段类型	说明	操作类型
提交	“提交”阶段用于定义要包含在请求中的初始请求项。可将多个工作流任务与此阶段类型关联。在“提交”阶段，必须至少将一个请求项添加到请求中。 “添加叶节点”或“添加枝节点”任务可以选择性地配置有从属工作流任务。系统为原始工作流任务添加请求项，为每个从属任务添加其他请求项。 主任务不能也是从属任务。在主任务上计算添加项的名称时，主任务和从属任务视为相关组。如果在名称计算仍暂挂时删除主任务，非添加从属任务也将被删除。 **注意：**每个请求只有一个“提交”阶段。无法为此阶段定义工作流阶段标准。	• 添加枝节点 • 添加叶节点 • 更新 • 停用 • 插入 • 移动 • 移除 • 删除
扩充	“扩充”阶段用于更新在“提交”阶段所添加的请求项或者添加请求项。可以为此阶段定义工作流阶段标准。 一个“扩充”阶段只有一个与其关联的工作流任务。典型的“扩充”阶段使用的工作流任务包含对现有请求项的更新操作。但是，有些“扩充”阶段可能需要创建其他行项目，例如： • 将单个节点插入多个层次 • 更新单个节点在多个层次中的本地属性 此阶段在“提交”和“最终提交”阶段之间发生。 **注意：**工作流模型可定义任意数量的“扩充”阶段。	• 更新（现有请求项） • 插入（添加新项目） • 移动（添加新项） • 可用于“提交”阶段的所有操作类型
批准	“批准”阶段用于查看和批准“提交”阶段所添加的或者“扩充”阶段所添加或更新的所有请求项。在“批准”阶段用户无法添加或编辑请求项。可以为此阶段定义工作流阶段标准。 “批准”阶段使用单个工作流任务来查看属性，并在请求处于该阶段时对请求项运行验证。更新任务可供在只读模式下在“批准”阶段中使用。要更新中间阶段中的请求项的属性，请改用“扩充”阶段类型。 此阶段在“提交”和“最终提交”阶段之间发生。 **注意：**工作流模型可定义任意数量的“批准”阶段。	更新（现有请求项）

表 16-1 （续） 工作流阶段类型

工作流阶段类型	说明	操作类型
最终提交	“最终提交”阶段用于对请求进行最终审批，以触发最终提交来将请求中的请求项最终提交到目标版本。最终提交用户必须审批请求中的所有请求项。可以为此阶段定义工作流阶段标准，但不能在此阶段拆分请求。 “最终提交”阶段没有与其关联的工作流任务。而是显示属性的超集并运行适用于之前“提交”和“扩充”阶段的请求项的验证超集。“最终提交”阶段中的用户可以更新任何针对请求项显示的可编辑属性，以做出最终的调整。 这是最后的工作流阶段。 **注意：**每个请求只有一个“最终提交”阶段。	不适用

阶段条件

阶段条件可用于根据为请求中的项目计算的指定标准，更改特定请求的工作流路径。可以为阶段设置条件并选择满足条件时应执行的操作；例如，请求是否可进入阶段，或者某些请求项是否拆分为单独的请求。可以根据以下标准计算工作流阶段条件：

- **属性标准** - 使用属性查询运算符和文字值计算为阶段的阶段标准。
- **选定验证** - 选择一个或多个要作为阶段的阶段标准运行的验证。可以为“批准”、“扩充”或“最终提交”阶段选择此选项。
- **任务验证** - 分配给工作流任务的验证失败。选中时，分配给任务的验证也将作为阶段的阶段标准运行。可以为“批准”或“扩充”阶段选择此选项。如果没有向分配到阶段的任务分配任何验证，则此选项不可用。

如果有任何请求项满足某个工作流阶段的阶段条件，则可执行以下操作之一：

- **进入阶段** - 对于“批准”、“扩充”或“最终提交”阶段，请求被分配给阶段中的用户。请求进入阶段，工作流处理继续针对该阶段。
- **拆分请求项** - 对于“批准”或“扩充”阶段，满足阶段条件的请求项将被移到使用相同工作流模型的单独的已提交请求中。新请求进入工作流阶段并被分配给阶段中的用户。不满足阶段条件的请求项留在原始请求中，并且原始请求将跳过该阶段。如果所有请求项都满足阶段标准，则不拆分请求，并进入“拆分”阶段。

如果请求项不满足某个工作流阶段的阶段条件，则将跳过该阶段，并且该请求将移到工作流模型中的下一个阶段。

审批方法

可以选择哪些用户必须批准请求中的某个阶段：

- **任何组** - 已分配节点访问组的任何用户都可以批准请求，使其进入下一个工作流阶段。必须为节点访问组分配对该层次中当前阶段类型的访问权限或更高访问权限。如果该阶段的已分配访问组对请求中的请求项都没有适当的数据访问权限，则只要提供了必需的值并且所有请求项均通过了验证，即可跳过该阶段。

- **所有组** - 所有已分配节点访问组都必须至少有一个用户批准请求，该请求才能进入下一阶段。如果该阶段的已分配访问组对请求中的请求项都没有适当的数据访问权限，则该请求将升级给数据管理员解决。

重新审批

如果请求被退回之前的阶段并且退回时修改了请求项，则对请求所做的更改可能需要由初始审批原始请求的用户重新审批。此选项用于确定在退回模式下对每个阶段所做的更改是否需要由其他用户重新审批。选择下列选项之一：

- **当前** - 如果在此阶段中对请求进行了更改，则仅当前阶段必须进行重新审批。批准后，请求将分配给之前退回该请求的用户。
- **全部** - 如果在此阶段中对请求进行了更改，则后续阶段必须进行重新审批。

职责分离

可根据需要将工作流阶段配置为要求一个独立的审批用户，该用户尚未对请求中任何其他阶段执行提交或批准。启用“职责分离”选项时，已为其他工作流阶段执行了提交或批准的用户将无法在启用了该选项的阶段中申请请求。请注意以下例外：

- 提交者可以申请已退回到“提交”阶段的请求。
- 该阶段之前的批准者可以申请退回到“批准”或“扩充”阶段的请求。
- 数据管理员角色用户可以申请分配给他们的任何请求，不管是否之前审批。

通知

通知包括 Web 客户端警报和电子邮件通知。您可以设置是否针对某一工作流阶段向工作流用户发送警报和通知以及何时发送。通知根据该阶段的“通知”设置和触发通知的工作流事件类型进行筛选，以发送给特定用户。

> **注：**
>
> 用户不会收到他们执行的操作的相关通知。

对于每个阶段，从以下“通知”选项中进行选择：

- **无** - 对于此工作流阶段，不通知用户要执行的操作。
- **任务接受者** - 发生分配、批准、最终提交或驳回操作时，将通知属于当前分配给请求的任何工作流节点访问组的用户。

  仅当满足以下条件时，才会通知任务接受者：他们是分配给阶段的工作流访问组的成员，并且阶段采用的“通知”设置是“任务接受者”或者“任务接受者和参与者”。
- **参与者**
  - 当发生最终提交或驳回操作时，会通知提交或申请请求的用户。
  - 当发生批准或提升操作时，会通知提交请求的用户。

  仅当满足以下条件时，才会通知参与者：他们是分配给阶段的工作流访问组的成员，并且阶段采用的“通知”设置是“参与者”或者“任务接受者和参与者”。
- **任务接受者和参与者** - 通知任务接受者和参与者。

下表列出了触发通知的操作以及基于每个阶段的“通知”设置的通知收件人。

表 16-2　工作流警报

工作流操作	通知发送给			
	任务接受者	提交者	参与者	通知用户
分配	X			
批准	X	X		X
提升		X		X
升级	X			X
驳回	X		X	X
最终提交	X		X	X

> **注:**
>
> 通知用户是分配给某个阶段的工作流节点访问组的成员用户，对请求项仅具有通知访问权限。仅当“通知”设置为“任务接受者”或者“任务接受者和参与者”时才通知他们。如果“通知”选项为“无”或“参与者”，则不通知这些用户

从属工作流任务

当另一个任务正在执行时，可以使用从属工作流任务自动执行监管请求中的工作流任务。例如，添加节点时，还可以将该节点插入其他层次，以便在最终提交该请求时确保所有层次中的同步。可以使用“添加叶节点”和“添加枝节点”操作类型为主工作流任务配置从属任务。

某个请求项添加到请求时，该项的选定任务是主任务。如果主任务配置有从属任务，将为每个从属任务自动向该请求添加其他请求项。

模型筛选器

可以限制用户能查看以及为特定类型的请求选择的版本、层次和节点类型。

- **版本变量** - 限制针对特定工作流模型中请求的请求项可以选择的版本。
- **层次组属性** - 限制针对特定工作流模型中请求的请求项可以从中选择节点的层次。
- **层次组** - 指定了“层次组属性”时，必须指定此选项。
- **节点类型** - 限制在特定工作流模型的请求中可以添加为请求项的节点。

请求和申请持续时间

可以为请求的工作流模型配置请求或申请持续时间，以便控制监管工作流基于为特定类型请求估算的预期时间对请求进行自动处理。

- **请求持续时间** - 批准和最终提交请求预期需要的天数。当请求的存在时间超过请求持续时间之后，会将该请求标记为“过期”。
- **申请持续时间** - 指示监管用户在某个工作流阶段使请求处于申请状态预期需要的天数。当请求的申请时间超过申请持续时间之后，会自动对该请求取消申请，以使其可供其他分配的用户进行申请。

> **注：**
>
> 任一选项的值为零都表示对该工作流模型禁用过期和自动取消申请功能。

创建工作流模型

要创建工作流模型：

1. 在主页上，选择**管理**。
2. 从**新建**中，选择**工作流模型**。
3. 输入工作流模型的名称、标签和说明。

   名称是工作流模型的唯一名称。标签是用户友好的工作流模型标签，可以与名称相同。说明是可选项。

   URL 可以直接插入到说明字段中，或者 URL 可以使用语法 `[url=http_URL]URL_Title[/url]`，其中 `http_URL` 指定超链接文本，而 `URL_Title` 指定向用户显示的文本。例如，示例 [url=http://support.oracle.com]Oracle Support[/url] 在属性网格中会显示为 Oracle Support。
4. **可选：**输入**请求持续时间**和**申请持续时间**的天数
5. 在**工作流阶段**选项卡上，双击某个阶段（“提交”或“最终提交”），或者单击**添加阶段**。
6. 在**阶段**选项卡上，配置以下选项。有关这些选项的其他信息，请参阅“工作流阶段”。
   - **标签** - 输入阶段的标签。阶段标签可以随时编辑，即使已经存在对模型的请求之后也可以。
   - **类型** - 选择阶段类型。在针对模型的请求存在之前，可以编辑阶段类型；之后将无法更改。
   - **工作流方法** - 指定必须由哪些节点访问组来批准请求中的阶段。
   - **重新审批** - 指定是仅当前阶段中所做的更改需要审批，还是所有阶段中所做的更改都需要审批。
   - **通知** - 指定向谁发送通知和警报。
   - **职责分离** - 选择此选项将要求一个独立的审批用户，该用户尚未对请求中任何其他阶段执行提交或批准。
   - **重新计算任务属性** - 选择以与外部查找属性配合使用或者允许重新计算已计算的名称或父代值。在之后的工作流阶段中输入数据时此选项是必需的，用于计算请求项的最终“名称”或“父代”。

     > **注：**
     >
     > 如果某个工作流模型设置为允许“重新计算任务属性”并且手动覆盖了计算的“名称”或“父代”，则在该阶段或任何后续阶段中将不会再次计算“名称”或“父代”。
7. 仅对于“提交”阶段任务，在**任务**选项卡上，为阶段配置任务：
   - 使用左右箭头按钮选择要分配给阶段的任务

- 使用上下箭头按钮按所需顺序放置任务。
- 如果任务是从属任务，需要设置其依赖的主任务。对于从属任务，单击 并从“主任务”下拉列表中选择主任务。

> **注：**
>
> 仅“添加枝节点”或“添加叶节点”任务可以设置为主任务。主任务不能隐藏，也不能是从属任务。

- **隐藏** - 如果为从属任务选择此项，则该任务不显示在请求中的“添加项”对话框中。

> **注：**
>
> 在针对模型的请求存在之前，可以编辑选定的任务；之后将无法更改。

8. 在**节点访问组**选项卡上，选择要与工作流阶段关联的工作流节点访问组。

   只能为阶段分配“工作流”类型的节点访问组。

9. **可选：**要为工作流阶段添加标准，请在**条件**选项卡上选择条件的类型，再选择要执行的操作，然后单击 ：
   - **类型**
     - **属性标准** - 选择一个或多个要计算为阶段的阶段标准的属性。单击**添加**以插入条件行。选择行的一个**属性**和**运算符**，然后输入一个**值**。
     - **选定验证** - 选择一个或多个要作为阶段的阶段标准运行的验证。单击箭头可将验证移到**选定**列表。
     - **任务验证** - 选择此选项将运行作为阶段标准分配给任务的验证。
   - **操作** - 选择当满足阶段标准时要对工作流阶段执行的操作（“进入阶段”或“拆分请求项”）。有关详细信息，请参阅“阶段条件”。
10. 单击 保存工作流阶段。
11. **可选：**在**筛选器**选项卡上进行选择，以限制对于特定类型的请求用户可以查看和选择的版本、层次和节点类型。
12. **可选：**单击**添加阶段**将“扩充”或“批准”阶段添加到工作流模型，然后对添加的每个阶段执行步骤 6-8。
13. 单击 保存工作流模型。

编辑工作流模型

将会限制对为其创建了请求的工作流模型进行某些编辑，以便确保在处理工作流期间现有请求不会受到负面影响，并且在完成请求之后其内容不会被更改。对于已更改请求的模型，适用以下编辑限制：

- 不能添加、删除模型的工作流阶段，也不能对其重新排序。

- 不能更改阶段的类型。
- 不能更改模型上的工作流阶段的任务。

要编辑工作流模型：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流模型**。
3. 选择模型，然后单击 。
4. 对工作流模型进行更改，然后单击 。

复制工作流模型

可以通过复制现有模型来创建工作流模型。将会复制所有工作流阶段、模型筛选器和持续时间设置，并且在保存之前可以对其进行编辑。在需要编辑用于当前请求的现有工作流模型以便以不同方式处理将来请求的情况下，可以复制该模型，然后对新模型进行更改。然后，可以将编辑后的模型副本用于新创建的请求。

要复制工作流模型：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流模型**。
3. 选择要复制的模型，然后单击 。
4. 为模型输入新名称。
5. 对模型进行任何其他更改，然后单击 以保存工作流模型。

重命名工作流模型

要随着时间变化支持不同的工作流要求，可以复制工作流模型以对其配置应用编辑。在这些情况下，可以重命名模型副本以匹配监管用户已经熟悉的原始工作流模型的名称。

要重命名工作流模型：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流模型**。
3. 选择要重命名的模型，然后单击 。
4. 为模型输入新名称，然后单击 。

隐藏工作流模型

可以隐藏工作流模型以防止用户使用这些模型创建新的请求。在隐藏工作流模型之前创建的现有请求将继续通过模型，直到完成。在复制并修改工作流模型以替换原始模型时，可以隐藏原始模型，以便仅有一个模型实例可用于新请求。

> **注：**
>
> 使用您选择隐藏的工作流模型的请求将继续其进程流直至完成。

要隐藏工作流模型：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流模型**。
3. 选择要隐藏的模型，然后单击  。
4. 选择**隐藏**，然后单击  。

删除工作流模型

仅当没有与其关联的请求（包括正在进行的请求或历史请求）时，才能删除工作流模型。在删除请求的版本之前，将保留完成的请求，因此要求工作流模型也可用以便查看请求。

> **提示：**
>
> 根据“隐藏工作流模型”中的信息来确定这是否为更合适的选择。

要删除工作流模型：

1. 在主页上，选择**管理**。
2. 在**工作流**下，展开**工作流模型**。
3. 选择模型，然后单击  。
4. 单击**删除此工作流模型**以确认删除。

17

管理 Data Relationship Management Analytics

Oracle Data Relationship Management Analytics 提供了用于显示以下内容的仪表板：更改跟踪、增长分析、请求监控、工作流模型性能以及参与者和用户组的性能。Data Relationship Management Analytics 仪表板包括：

- **更改** - 提供一段时间内 Oracle Data Relationship Management 系统中发生的更改的聚合视图。此仪表板中的度量基于最终提交的请求以及所有交互式更改。此仪表板包括节点中的添加、更新、移动和删除等更改操作以及属性更改以便按层次、节点类型、属性类别等提供更改透视。用户可以通过更改方法、交互或工作流来了解更改趋势，以进行监管。用户可以根据上下文钻取每个更改来检查事务详细信息以及将这些详细信息导出到平面文件以便脱机进行进一步分析。
- **增长** - 通过显示孤立和共享节点数、总节点数以及自先前版本节点的增加或减少总数（对于世系版本）和过去 30 天的增加或减少总数（对于非世系版本），提供版本和层次如何随着时间而更改的分析信息。
- **请求** - 显示与打开的 Oracle Data Relationship Governance 请求相关的关键绩效指标（您可以通过这些指标标识出瓶颈以及逾期或即将到期的请求），并提供往回钻取至 Data Relationship Governance 请求以更改请求的能力。
- **模型** - 通过显示已完成（已最终提交或已驳回）请求的历史性能（包括参与者行为趋势和资源工作量）来提供 Data Relationship Governance 工作量模型设计分析，并提供往回钻取至 Data Relationship Governance 请求的能力。工作量模型分析报告每个工作量模型处理的已完成请求的性能，以基于服务级别协议、实现的自动化级别、周期时间、已最终提交资源、请求工作量、吞吐量和参与者的参与来了解模型性能。
- **报表** - 用于查看用户和组成员身份、安全性和活动。提供的信息包括用户角色分配、访问组分配报表和用户登录活动。
  - **用户角色分配报表** - 按角色提供用户列表或者按用户提供角色列表以及按许可用户类型显示计数。
  - **访问组成员报表** - 按交互式和工作流用户组提供用户列表。
  - **对象访问组授权报表** - 提供用户和用户组到特定 Data Relationship Management 对象的映射。
  - **层次访问组分配报表** - 提供用户和组对层次中节点的数据授权信息。
  - **工作流访问组分配报表** - 提供用户和组对工作流模型阶段的数据授权信息。
  - **用户登录活动报表** - 提供一段时间内用户登录活动的趋势报表。
  - **元数据对象使用报表** - 提供以下 Data Relationship Management 对象的频率分布和存在时间信息：查询、比较、导入、导出、混合器和集。

访问 Data Relationship Analytics

配置 Oracle Data Relationship Management Analytics 之前，请确保已完成以下任务：

- 设置 Analytics URL - 提供从 Oracle Data Relationship Management 指向 Data Relationship Management Analytics 的链接。请参阅《*Oracle Data Relationship Management* 安装指南》中的“配置 Analytics URL”。
- 设置 Web 场 - 提供从 Data Relationship Management Analytics 到 Data Relationship Management 的回钻。请参阅《*Oracle Data Relationship Management* 安装指南》中的“配置 Web 服务器”。
- 已设置版本世系 - 版本世系允许 Data Relationship Management Analytics 聚合世系以及多个版本中的更改。请参阅《*Oracle Data Relationship Management* 用户指南》中的“编辑版本属性”。
- 层次和版本节点计数更新时在 Data Relationship Management 中进行设置。打开、保存或关闭版本时将更新节点计数，并如系统首选项中所指定的那样。请参阅“系统首选项”中的 AnalyticsNodeCountUpdateTime。
- 将层次组属性设置为默认“核心”属性类型。Data Relationship Management Analytics 中仅支持默认“核心”属性类型。请参阅“创建属性”中的步骤 6。

在 Data Relationship Management 中，单击 "Analytics" 链接。

> **注：**
>
> 仅当将用户分配给以下任何角色时 "Analytics" 链接才可用：Analytics 用户、监管管理员、访问管理员、数据管理员、应用程序管理员。

使用首选项

创建执行计划之前，需要配置首选项。

要设置首选项：

1. 单击  。
2. **可选：**执行以下操作：
   - **批大小** - 输入批大小值。用于模型分析。默认值为 250 MB，并且除非绝对需要，否则不应更改该默认值。批大小越大，内存和数据库需求越大。
   - **初始提取日期** - 设置为所有 Oracle Data Relationship Management Analytics 任务提取数据的起始日期。
3. 单击**保存**。

使用执行计划

预定义的任务从 Oracle Data Relationship Management 提取信息并将其返回给特定的 Oracle Data Relationship Management Analytics 仪表板，可以在其中筛选和查看该信息。作业包含特定于仪表板的任务。执行计划中可以包含多个作业。

执行计划包含调度以及一个或多个作业及其任务。执行计划可以配置为每天、每周或每月运行，并且可以进行调度以“简单”（立即运行或在将来日期/时间运行）或 Cron（使用

Cron 表达式指示调度信息）形式运行。执行计划可以进行编辑、在不使用时停用以及当不再需要时删除。

表 17-1　作业任务

作业	任务
更改分析	事务事实表 事务聚合 事务属性聚合 版本世系
用户活动报表	事务事实表
增长分析	版本世系 层次计数 版本计数
模型分析	模型分析

创建执行计划

要创建执行计划：

1. 在 Oracle Data Relationship Management Analytics 仪表板中，选择**设置**。
2. 单击**创建**并输入以下信息：
   - **名称** - 输入执行计划的名称
   - **调度类型** - 从以下选项中选择：
     - **简单** - 用于指定开始和结束日期
     - **Cron** - 用于指定 Cron 表达式
   - **调度器时间范围** - 选择“立即运行”或“将来”。
3. 单击**下一步**。
4. 执行以下操作：
   - 如果为“调度类型”选择了**简单**并为“调度器时间范围”选择了**立即运行**，请执行以下操作：
     a. **可选：**选择**截断并加载**可以截断与此作业关联的任何表并根据系统中的初始提取日期重新加载。如果未选择，则运行增量式加载。
     b. 如果您确定要截断并加载，请单击**确定**。
   - 如果选择了**简单**作为“调度类型”并为“调度器时间范围”选择了**将来**，请执行以下操作：
     a. 选择运行执行计划的“频率”：每天、每周或每月。
     b. 单击  输入开始日期和时间。
     c. **可选：**单击  输入结束日期和时间。
   - 如果选择了 **Cron** 作为“调度类型”，则输入调度器运行时间的 Cron 表达式。
5. 单击**下一步**。

6. 选择要添加到执行计划的作业。使用“移动”、“移动全部”、“删除”和“全部删除”按钮将作业从“可用”列表移动到“选定的”列表。
7. 单击**下一步**。
8. 查看执行计划设置，然后单击**调度计划**。

> **注：**
>
> 对于要运行的执行计划，必须启动调度器。要启动调度器，请单击  并选择**启动**。

9. 单击**确定**来确认调度计划。

编辑执行计划

编辑执行计划时，除了计划名称外，可以编辑所有字段。

要编辑执行计划：

1. 选择要编辑的计划。
2. 单击  ，执行“创建执行计划”中的步骤 2-9 以更改计划。

> **注：**
>
> 无法更改计划名称。如果需要更改计划名称，则删除该计划并创建新计划。

停用和重新激活执行计划

停用执行计划时，将从调度器中删除所有将来调度的计划并且该计划将移至“非活动计划”选项卡。要重新激活计划，请在“非活动计划”选项卡上编辑该计划，然后调度该计划。

要停用执行计划：

1. 选择  ，然后选择要停用的计划。
2. 单击计划名称旁边的  。

要重新激活执行计划：

1. 选择  ，然后选择要重新激活的计划。
2. 单击  ，执行“创建执行计划”中的步骤 2-9 以更改计划。

> **注：**
>
> 无法更改计划名称。如果需要更改计划名称，则删除该计划并创建新计划。

删除执行计划

要删除执行计划：

1. 选择您要删除的执行计划。
2. 单击计划名称旁边的  。
3. 单击**确定**确认删除。

查看活动

在“最近的活动”节中，可以查看已经运行的执行计划的结果。可以查看执行计划的开始和结束时间、运行的持续时间、处理的记录数以及运行的状态。请注意，如果您在同一计划中调度多个作业并且多个作业包括另一个作业已经运行的任务，则该执行将在后续作业中跳过该任务并将在执行计划结果中显示为“跳过重复项”。

要查看已经运行的执行计划的结果：

1. 单击  或单击  。
2. 通过单击计划名称左侧的箭头，展开要查看的执行计划。可以展开计划中的作业来查看关联的任务。
3. **可选：**单击筛选器栏并设置筛选器选项：
   - **时间范围** - 输入显示计划活动的天数。例如，如果输入 2，则显示自过去 2 天以来的计划活动。
   - **名称** - 选择**全部**或选择要包括在结果中的执行计划名称。
   - **状态** - 选择**全部**或选择要包括在结果中的执行计划状态。计划状态为“完成”、“部分失败”、“已失败”和“正在处理”。

> **注：**
>
> “状态”筛选条件仅应用于执行计划状态，而不应用于作业或任务状态。

18

与外部工作流应用程序相集成

可以使用外部工作流应用程序处理外部源对 Oracle Data Relationship Management 的建议更改。在外部工作流程中，Web 服务 API 提供了一个外部请求界面，可用于将多个更改组成一个工作单元以便进行验证和最终提交。API 用户必须具有“工作流用户”角色才能参与外部请求。该请求界面是常规界面，不支持在 Web 客户端中使用工作流模型、工作流任务或“工作列表”页面。这些常规的外部请求记录在请求历史记录中，且只能从其中进行访问。

有关外部请求的 API 支持的详细信息，请参阅 "Oracle Data Relationship Management API Reference"。

外部请求

您可创建外部请求以便：

- 添加层次
- 添加节点
- 插入和移动节点
- 激活、停用和删除节点
- 更新属性
- 删除属性值

外部请求可以存储为草稿状态，以供审批并针对某 Oracle Data Relationship Management 版本进行验证，而不是立即最终提交对该版本的更改。处于这种待审批状态的外部请求可由多个用户在不同时间进行更新，并根据需要重新进行验证。请求获得批准后，请求中的事务将最终提交到 Data Relationship Management 版本。

> **注：**
>
> 外部请求获得批准后，将无法再修改该请求，并且在删除关联版本之前无法删除该请求。

外部请求由以下元素组成：

- 目标 Data Relationship Management 版本。
- 请求所有者 - 有效的 Data Relationship Management 用户 ID。
- 自定义工作流 ID - 工作流应用程序中请求的标识符。
- 自定义工作流标签 - 工作流应用程序中请求的简短说明。
- 自定义工作流状态 - 管理工作流应用程序中请求的状态。
- 自定义工作流信息 - 存储工作流应用程序需要的额外信息。
- 请求注释 - 请求的注释。

- 创建者 -- 创建初始请求的用户。
- 创建日期 -- 创建请求的日期。
- 更新者 -- 上次更新请求的用户。
- 更新日期 -- 上次更新请求的日期。
- 批准者 -- 批准请求的用户。
- 批准日期 -- 批准请求的日期。
- 验证标志 -- 表明请求自上次更新后是否已经过验证。
- 批准标志 -- 表明请求是否已获得批准。
- 其他批量验证 - 只能在验证或批准操作过程中应用于请求中的操作
- 操作项列表 - 影响当前请求的层次和节点

19

迁移 Data Relationship Management 元数据

Oracle Data Relationship Management 迁移实用程序允许应用程序管理员在 Data Relationship Management 应用程序之间移动元数据对象类型。

在迁移实用程序中，您可以：

- 将 Data Relationship Management 应用程序中的元数据对象类型提取到 XML 文件中，并从结果中生成 HTML 报表
- 从 XML 文件将元数据加载到 Data Relationship Management 应用程序中
- 比较两个源之间元数据的差别，使用这些差别创建 XML 文件，并从结果中生成 HTML 报表
- 查看 XML 文件中的元数据，并从该文件生成 HTML 报表

您可以提取、加载、比较和查看以下类型的元数据：

- 属性定义
- 属性类别
- 验证
- 节点类型
- 图标
- 节点访问组
- 层次组
- 查询（标准、系统和自定义）
- 比较（标准、系统和自定义）
- 域
- 版本变量（标准、系统和自定义）
- 导出（标准、系统和自定义）
- 导出集（标准、系统和自定义）
- 导入（标准、系统和自定义）
- 混合器（标准、系统和自定义）
- 系统首选项
- 外部连接（标准、系统和自定义）

  外部连接仅显示连接名称；未添加对象访问组名称前缀。

> **注：**
>
> 连接字符串、用户 ID 和密码不会随迁移加载和提取操作进行迁移。

- 对象访问组
- 工作流任务
- 工作流模型

迁移核心属性配置和设置

可以使用元数据迁移实用程序在 Data Relationship Management 实例（同一版本）之间迁移以下核心属性配置和设置：

- Core.DefaultDisplayBy [默认显示属性]
- Core.DefaultPasteProps [默认粘贴属性]
- Core.DefaultSynchBy [默认匹配方式]
- Core.EnableSharedNodes [启用共享节点]
- Core.HierarchyNodeType [层次节点类型]
- Core.IDLengthLeafProp [ID 长度叶属性]
- Core.IDLengthLimbProp [ID 长度枝属性]
- Core.PrefillLeafProp [预填充叶属性]
- Core.PrefillLimbProp [预填充枝属性]
- Core.SortOrder [排序顺序]
- Core.StandardHierSort [标准层次排序]

打开迁移实用程序

默认情况下，迁移实用程序安装在以下路径中：

MIDDLEWARE_HOME\EPMSystem11R1\products\DataRelationshipManagement\client

要打开迁移实用程序，请双击 **Data Relationship Management 迁移实用程序**。

提取元数据

您可以选择要从 Oracle Data Relationship Management 应用程序提取的元数据类型。可以将信息提取到 XML 文件，然后可以查看该文件、将其加载到其他 Data Relationship Management 应用程序、将其与另一 XML 文件进行比较，或将其与另一 Data Relationship Management 应用程序进行比较。还可以将该文件用于备份、存储和审核用途。

您可以基于创建的 XML 文件中的信息生成报表。

要从 Data Relationship Management 应用程序提取元数据：

1. 在主菜单上，单击**提取**。
2. 输入 Data Relationship Management 连接信息，然后单击**登录**。
3. 选择要提取的对象类型或对象，然后单击**下一步**。

> **注：**
>
> 单击层次树中的加号可查看对象。选中某个对象类型的复选框可选择该对象类型及其所有对象，或者选中要提取的对象的复选框。单击某个对象名称可在新窗口中显示对象类型定义。

4. **可选：**单击**查找**以搜索元数据对象类型或对象。

> **注：**
>
> 将返回包含所输入文本的所有对象类型。要导航到结果中的特定对象，请单击“跳转到”链接。

5. 查看摘要信息。

> **注：**
>
> 迁移实用程序会对具有依赖项的对象类型执行其他检查。例如，某项导出可能依赖于属性定义，或者某个属性定义可能引用了另一属性定义。如果摘要中缺失了某些依赖项，您可以选择要包含的特定依赖项。可以包含排除的所有依赖项或排除所有依赖项。

> **注：**
>
> 增加页面大小以允许您定义要在页面上查看的对象类型数量。

6. **可选：**为此提取输入元数据详细信息。

   可以输入以下信息：

   - **标题** - 最多 255 个字符
   - **目的** - 格式化备注
   - **用法** - 格式化备注
   - **应用程序版本** - 最多 20 个字符
   - **文件版本** - 最多 20 个字符

7. 单击**运行提取**。
8. 执行以下任意操作：
   - 单击**下载元数据文件**以打开或保存 XML 文件。
   - 单击**查看元数据文件**以查看 XML 文件的详细信息。
   - 单击**加载元数据文件**以将 XML 文件加载到 Data Relationship Management 应用程序。有关详细信息，请参阅“加载元数据”。
   - 单击**为元数据文件生成报表**以基于 XML 文件生成报表。有关详细信息，请参阅“生成报表”。

加载元数据

仅 Oracle Data Relationship Management XML 格式的文件可以加载到 Data Relationship Management 应用程序中。执行加载后将创建日志文件，并显示以下严重程度的数据：审核、信息、警告和错误消息。

> **注：**
>
> 在加载元数据文件之前，建议先执行现有元数据的提取，以便在您希望还原到先前的配置时使用。在加载元数据之前执行数据库备份也是一个不错的办法，特别是在将迁移文件加载到生产环境时。

要从 XML 文件将元数据加载到 Data Relationship Management 应用程序中：

1. 在主菜单上，单击**加载**。
2. 单击**浏览**，选择要加载的 XML 文件，然后单击**上传**。

   > **注：**
   >
   > 迁移文件必须采用 UTF-8 编码。

3. 查看上传文件信息并单击**下一步**。
4. 输入 Data Relationship Management 连接信息，然后单击**登录**。
5. 选择要加载的对象类型或对象，然后单击**下一步**。

   > **注：**
   >
   > 单击层次树中的加号可查看对象。选中某个对象类型的复选框可选择该对象类型及其所有对象，或者选中要加载的对象的复选框。单击某个对象名称可在新窗口中显示对象类型定义。

6. 查看摘要信息并单击**下一步**。

   > **注：**
   >
   > 页面大小允许您定义要在页面上查看的对象类型数量。

7. **可选：**选择**出错后继续加载**，可在遇到错误时继续加载。
8. 单击**运行加载**。
9. 查看加载结果。

   可以通过选择要显示的详细信息的严重程度来更改日志文件的视图：审核、信息、警告和错误。要保存日志文件，请单击**下载**。

> **注：**
>
> 可以使用列标题链接按任意列对日志项进行排序。

比较元数据

您可以比较两个元数据源。可以比较两个 Oracle Data Relationship Management 应用程序之间、两个 XML 文件之间，或一个 Data Relationship Management 应用程序和一个 XML 文件之间元数据的差别。可以生成包含两个元数据源不同之处的 XML 文件。这些结果可用于恢复数据、撤消非授权的更改，或查找错误的对象类型配置。

您可以基于创建的 XML 文件中的信息生成报表。

要比较元数据：

1. 在主菜单上，单击**差异**。
2. 从**源 #1** 下拉列表中选择源的类型：服务器连接或 XML 文件。
3. 执行下列操作之一：
   - 如果选择**服务器连接**，请输入 Data Relationship Management 连接信息并单击**登录**。
   - 如果选择 **XML 文件**，请单击**浏览**并选择要用于比较的 XML 文件，然后单击**上传**。
4. 上传文件后，查看上传文件信息，然后单击**下一步**。否则，请跳到下一步。
5. 对源 #2 重复步骤 2–4。
6. 单击**下一步**。
7. 通过以下操作来选择要包含在差异文件中的对象类型：
   - 选择筛选器
   - 单击 **>** 以从源 #1 中选择对象类型。
   - 单击 **<** 以从源 #2 中选择对象类型。
   - 单击 **X** 以取消选择对象类型。
   - 单击左侧的列标题以根据选择的筛选器选择源 #1 中的所有对象。
   - 单击右侧的列标题以根据选择的筛选器选择源 #2 中的所有对象。
   - 单击中间的列标题以根据选择的筛选器取消选择所有对象。
   - 单击比较结果顶部的页面链接以切换到其他页面。

   > **注：**
   >
   > 页面大小允许您定义要在页面上查看的对象类型数量。

8. 单击**创建差异文件**。
9. 执行以下任意操作：
   - 单击**下载元数据差异文件**以打开或保存 XML 文件。
   - 单击**查看元数据差异文件**以查看 XML 文件的详细信息。

- 单击**加载元数据差异文件**以将文件加载到 Data Relationship Management 应用程序。有关详细信息，请参阅“加载元数据”。
- 单击**为元数据文件生成报表**以基于 XML 文件生成报表。有关详细信息，请参阅“生成报表”。

查看元数据

您可以查看元数据文件并基于其中的信息生成报表。

要查看 XML 文件中的元数据：

1. 在主菜单上，单击**查看文件**。
2. 单击**浏览**，选择要查看的 XML 文件，然后单击**上传**。
3. 查看上传文件信息并单击**下一步**。
4. 单击层次树中的加号以查看元数据对象。
5. **可选：**单击**查找**以搜索文件中的项。

> **注：**
>
> 将返回包含文本的所有对象类型。要导航到结果中的特定对象，请单击“跳转到”链接。

6. **可选：**单击**报表**选项卡以基于文件生成 HTML 报表。

元数据文件限制

默认情况下，迁移实用程序中上传文件的大小限制为 4 MB。使用迁移实用程序加载或查看大型元数据文件时，如果文件大小超过了配置的限制，则可能出现以下错误。

“意外错误，尝试处理您的请求时发生意外错误：超过了最大请求长度。”

有关配置较大文件大小的信息，请参阅《*Oracle Data Relationship Management* 安装指南》中的“配置迁移实用程序”。

生成报表

您可以基于提取后生成的 XML 文件、差异报表，以及查看的元数据文件生成 HTML 报表。

要生成 HTML 报表：

1. 执行下列操作之一：
   - 提取元数据或创建差异报表后，单击**为元数据文件生成报表**。
   - 查看元数据文件后，单击**报表**。
2. 执行下列操作之一：
   - 单击**查看报表**以显示报表。
   - 单击**下载报表**以保存报表。