How To Use Utilities

An Open Source Asset for use with TIBCO® Data Virtualization

|  |  |
| --- | --- |
| **Project Name** | AS Assets Utilities |
| **Document Location** | This document is only valid on the day it was printed. The source of the document will be found in the ASAssets\_Utilities folder (https://github.com/TIBCOSoftware) |
| **Purpose** | Self-paced instructional |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 1.0 | 08/06/2010 | Mike Tinius, Calvin Goodrich, Gordon Rose | Initial revision |
| 1.1 | 08/11/2010 | Calvin Goodrich | Added documentation for string, time, and repository functions. Added initial guidelines for submitting new procedures. |
| 1.2 | 08/19/2010 | Mike Tinius | Added additional string, time and repository procedures. |
| 1.3 | 09/30/2010 | Mike Tinius | Added getNodeFromXML |
| 1.4 | 10/11/2010 | Calvin Goodrich | Added time and repository procedures. |
| 1.5 | 10/13/2010 | Gordon Rose | Added Active Directory and encoding procedures. |
| 2010.3 | 10/18/2010 | Calvin Goodrich | Finalized for 2010 Q3 official release |
| 2010.4 | 1/14/2011 | Calvin Goodrich | Added release notes section. Please detail updates to the Utilities here. |
| 2011.2 | 4/1/2011 | Calvin Goodrich | Updated for release 2011.2 |
| 2011.3 | 7/1/2011 | Calvin Goodrich | Updated for release 2011.3 |
| 2011.4 | 10/1/2011 | Calvin Goodrich | Updated for release 2011.4 |
| 2012.1 | 1/4/2012 | Calvin Goodrich | Updated for release 2012.1 |
| 2012.2 | 4/5/2012 | Calvin Goodrich | Updated for release 2012.2 |
| 2012.3 | 8/7/2012 | Calvin Goodrich | Updated for release 2012.3 |
| 2012.4 | 11/1/2012 | Calvin Goodrich | Updated for release 2012.4 |
| 2012.401 | 11/12/2012 | Mike Tinius | Updated for release 2012.401 |
| 2012.402 | 11/20/2012 | Mike Tinius | Updated for release 2012.402 |
| 2013.1 | 2/7/2013 | Calvin Goodrich | Updated for release 2013.1 |
| 2013.2 | 5/1/2013 | Calvin Goodrich | Updated for release 2013.2 |
| 2013.3 | 8/16/2013 | Calvin Goodrich | Updated for release 2013.3 |
| 2013.301 | 8/27/2013 | Mike Tinius | Updated for release 2013.301 |
| 2013.4 | 11/12/2013 | Calvin Goodrich | Updated for release 2013.4 |
| 2014.1 | 2/18/2014 | Calvin Goodrich | Updated for release 2014.1 |
| 2014.2 | 5/12/2014 | Calvin Goodrich | Updated for release 2014.2 |
| 2014.3 | 8/1/2014 | Calvin Goodrich | Updated for release 2014.3 |
| 2014.4 | 11/11/2014 | Calvin Goodrich | Updated for release 2014.4 |
| 2015.1 | 2/19/2015 | Calvin Goodrich | Updated for release 2015.1 |
| 2015.2 | 5/1/2015 | Calvin Goodrich | Updated for release 2015.2 |
| 2015.3 | 8/10/2015 | Calvin Goodrich | Updated for release 2015.3 |
| 2015.4 | 11/5/2015 | Calvin Goodrich | Updated for release 2015.4 |
| 2016.1 | 1/29/2016 | Calvin Goodrich | Updated for release 2016.1 |
| 2017.2 | 5/1/2017 | Mike Tinius | Updated for release 2017.2 |
| 2017.4 | 12/13/2017 | Mike Tinius | Transitioned to Tibco. |
| 2018.1 | 3/21/2018 | Mike Tinius | Updated for release 2018.1 |
| 2018.101 | 4/23/2018 | Mike Tinius | Updated installation notes for 2018.101 |
| 2018.4 | 11/13/2018 | Mike Tinius | Updated for release 2018.4 |
| 2019.200 | 06/17/2019 | Mike Tinius | Updated for release 2019.200 |
| 2019.300 | 07/16/2019 | Mike Tinius | Updated for release 2019.300 |
| 2019.301 | 08/01/2019 | Mike Tinius | Updated for release 2019.301 |
| 2020.100 | 01/03/2020 | Mike Tinius | Updated for release 2020.100 |
| 2020.101 | 01/23/2020 | Mike Tinius | Updated for release 2020.101 bug fix. |
| 2020.200 | 03/13/2020 | Mike Tinius | Updated for release 2020.200 |
| 2020.201 | 05/27/2020 | Mike Tinius | Updated for release 2020.201 |
| 2020.202 | 06/08/2020 | Mike Tinius | Updated for release 2020.202 |
| 2020.300 | 08/302020 | Mike Tinius | Updated for release 2020.300 |
| 2020.400 | 10/15/2020 | Mike Tinius | Updated for release 2020.400: Added XML unescape characters for all attributes. Added XMLTEXT to escape path and name variables. |
| 2020.401 | 11/4/2020 | Mike Tinius | Updated for CoE Automation requirements |
| 2020.402 | 12/12/2020 | Mike Tinius | Updated for CoE Automation and Best Practices Privilege Script requirements. |
| 2021.100 | 01/25/2021 | Mike Tinius | Updated for CoE Automation. Updated reintrospectDatasource. Replaced calls to BigintToTimestamp. |
| 2021.101 | 02/03/2021 | Mike Tinius | Added /modify folder for code modification. |
| 2021.102 | 03/31/2021 | Mike Tinius | Added archive/generateExportList. |
| 2021.200 | 05/18/2021 | Mike Tinius | Fixed archive export and import for v8.x. Other bug fixes. |
| 2021.201 | 06/16/2021 | Mike Tinius | Added getResourceLineageDependencies and getOptionsFileDuplicateTokens. Fixed all usage of getAttribute to use versionFull. |
| 2021.300 | 07/14/2021 | Mike Tinius | Fixed cacheRefreshSynch when Parallel and RefreshResourceStatistics is on. |
| 2021.400 | 09/21/2021 | Mike Tinius | validateDeployment – check for CJP. Removed check for source code differences as the change could be to the model and not the source code. Will rely on modified date differences only. Reformatted message output. Added /xml/getXMLNode, /xml/parseProperties, /request/getSqlPlan, /request/getSqlPlanChild |
| 2021.401 | 10/21/2021 | Mike Tinius | Fixed validateDeployment. Removed check for source code differences as the change could be to the model and not the source code. Will rely on modified date differences only. Reformatted message output. |
| 2021.402 | 11/30/2021 | Mike Tinius | Added the ability to parse an entire directory structure of views and return the query plan. Added the ability to export a package export list along with CoE deployment introspection, reintrospection, design, remove, relocate and rebind lists.  Dropped support for TDV 7.x. |
| 2022.100 | 03/25/2022 | Mike Tinius | Added new procedures for data source migration. And updated generateViews. |
| 2022.200 | 04/26/2022 | Mike Tinius | Added Security “cbs” section. Modified generateViews and getConstantV2. |
| 2022.300 | 08/04/2022 | Mike Tinius | Updated introspectResources and updateServerConfiguration |
| 2022.301 | 09/16/2022 | Mike Tinius | Modified output parameters for /shared/ASAssets/Utilities/request/getSqlPlan, getAllViewsSqlPlan, unescapeXML, escapeXML, generateViews, destroyDependentLineage, and modifyConstant. |
| 2022.400 | 11/10/2022 | Mike Tinius | Added /model folder to provide easier access to TDV catalog metadata. Fixed migrateDatasource and introspectResources |

Supported Versions

|  |  |
| --- | --- |
| **Name** | **Version** |
| TIBCO® Data Virtualization Server | 8.4 or later |

Third Party Software Licenses

|  |  |  |
| --- | --- | --- |
| **Product Name** | **Product Description and download site** | **License Type** |
| jTidy | http://jtidy.sourceforge.net/ | MIT Open Source |
| Apache Commons Net | http://commons.apache.org/proper/commons-net/ | Apache |
| jaxen |  | The Werken Company |
| jdom | http://www.jdom.org/ | Jason Hunter & Brett McLaughlin |
| xerces | http://www.apache.org/ | Apache |

**Table of Contents**

[1 Release Notes 19](#_Toc119006278)

[Introduction 19](#_Toc119006279)

[Regression Test Versions 19](#_Toc119006280)

[New Resources 19](#_Toc119006281)

[2022 Q400 19](#_Toc119006282)

[2022 Q200 19](#_Toc119006283)

[2022 Q100 19](#_Toc119006284)

[2021 Q402 19](#_Toc119006285)

[2021 Q401 20](#_Toc119006286)

[2021 Q201 20](#_Toc119006287)

[2021 Q200 20](#_Toc119006288)

[2021 Q102 20](#_Toc119006289)

[2021 Q101 20](#_Toc119006290)

[2021 Q100 20](#_Toc119006291)

[2020 Q402 21](#_Toc119006292)

[2022 Q300 21](#_Toc119006293)

[2020 Q400 21](#_Toc119006294)

[2020 Q300 21](#_Toc119006295)

[2020 Q202 21](#_Toc119006296)

[2020 Q201 21](#_Toc119006297)

[2020 Q200 21](#_Toc119006298)

[2020 Q101 21](#_Toc119006299)

[2020 Q100 21](#_Toc119006300)

[2019 Q301 22](#_Toc119006301)

[2019 Q200 22](#_Toc119006302)

[Updated Resources 23](#_Toc119006303)

[2022 Q400 23](#_Toc119006304)

[2022 Q301 23](#_Toc119006305)

[2022 Q300 23](#_Toc119006306)

[2022 Q200 24](#_Toc119006307)

[2022 Q100 24](#_Toc119006308)

[2021 Q400 24](#_Toc119006309)

[2021 Q300 24](#_Toc119006310)

[2021 Q201 24](#_Toc119006311)

[2021 Q200 24](#_Toc119006312)

[2021 Q102 25](#_Toc119006313)

[2021 Q100 25](#_Toc119006314)

[2020 Q402 25](#_Toc119006315)

[2020 Q401 25](#_Toc119006316)

[2020 Q400 25](#_Toc119006317)

[2020 Q300 26](#_Toc119006318)

[2020 Q202 26](#_Toc119006319)

[2020 Q201 26](#_Toc119006320)

[2020 Q200 27](#_Toc119006321)

[2020 Q101 27](#_Toc119006322)

[2020 Q100 27](#_Toc119006323)

[2019 Q301 28](#_Toc119006324)

[2019 Q200 28](#_Toc119006325)

[Removed Resources 30](#_Toc119006326)

[2020 Q200 30](#_Toc119006327)

[2019 Q200 30](#_Toc119006328)

[Deprecated Resources 30](#_Toc119006329)

[2020 Q100 30](#_Toc119006330)

[2019 Q301 31](#_Toc119006331)

[2019 Q200 31](#_Toc119006332)

[2018 Q100 31](#_Toc119006333)

[2015 Q300 32](#_Toc119006334)

[2015 Q200 32](#_Toc119006335)

[2014 Q400 32](#_Toc119006336)

[2014 Q300 32](#_Toc119006337)

[2014 Q100 32](#_Toc119006338)

[2012 Q400 32](#_Toc119006339)

[2012 Q100 33](#_Toc119006340)

[2011 Q300 33](#_Toc119006341)

[2 Introduction 34](#_Toc119006342)

[Purpose 34](#_Toc119006343)

[History 35](#_Toc119006344)

[Audience 35](#_Toc119006345)

[Installation Notes 35](#_Toc119006346)

[New Folder Structure 35](#_Toc119006347)

[Reserved Word List 37](#_Toc119006348)

[Recursive Procedure Use 37](#_Toc119006349)

[getEnvName Usage 37](#_Toc119006350)

[3 Top Level Utilities Procedures 39](#_Toc119006351)

[Introduction 39](#_Toc119006352)

[ExceptionDefinitions 39](#_Toc119006353)

[getUtilitiesVersion (Custom Function) 39](#_Toc119006354)

[reintrospectCJPs 39](#_Toc119006355)

[TypeDefinitions 40](#_Toc119006356)

[Custom Function Disclaimer 40](#_Toc119006357)

[Custom Function List 40](#_Toc119006358)

[4 How To Use ‘Active Directory’ Procedures 44](#_Toc119006359)

[Introduction 44](#_Toc119006360)

[ActiveDirectoryInt8ToDate (Custom Function) 44](#_Toc119006361)

[ActiveDirectoryTSToSQLTimeStamp (Custom Function) 45](#_Toc119006362)

[SimpleBinaryAND (Custom Function) 45](#_Toc119006363)

[5 How To Use ‘Archive’ Procedures 47](#_Toc119006364)

[Introduction 47](#_Toc119006365)

[backup\_export 47](#_Toc119006366)

[exportResources 47](#_Toc119006367)

[exportArchiveFile 50](#_Toc119006368)

[getExportResourceList 51](#_Toc119006369)

[getPkgExportList 53](#_Toc119006370)

[getPkgExportListByLabel 54](#_Toc119006371)

[importArchiveFile 59](#_Toc119006372)

[6 How To Use ‘Cache’ Procedures 64](#_Toc119006373)

[Introduction 64](#_Toc119006374)

[cacheRefreshSynch 64](#_Toc119006375)

[getCacheAgeHours 65](#_Toc119006376)

[getCacheStatusRows 66](#_Toc119006377)

[7 How To Use ‘Calculation’ Procedures 69](#_Toc119006378)

[Introduction 69](#_Toc119006379)

[calculateAge (Custom Function) 69](#_Toc119006380)

[medianFromQuery (Custom Function) 69](#_Toc119006381)

[8 How To Use ‘Conversion’ Procedures 71](#_Toc119006382)

[Introduction 71](#_Toc119006383)

[convertBit (Custom Function) 71](#_Toc119006384)

[convertBoolean (Custom Function) 71](#_Toc119006385)

[convertDoubleToInteger (Custom Function) 72](#_Toc119006386)

[convertTemperatureUnit (Custom Function) 72](#_Toc119006387)

[convertYN (Custom Function) 73](#_Toc119006388)

[9 How To Use ‘Deployment’ Procedures 74](#_Toc119006389)

[Introduction 74](#_Toc119006390)

[deployment/optionsfile/generateOptionsFile 74](#_Toc119006391)

[deployment/optionsfile/generateOptionsFileRebind 76](#_Toc119006392)

[deployment/optionsfile/getOptionsFileDuplicateTokens 79](#_Toc119006393)

[deployment/optionsfile/listOptionsFileTokens 81](#_Toc119006394)

[deployment/privileges/importResourceOwnership 84](#_Toc119006395)

[deployment/privileges/importResourcePrivileges 85](#_Toc119006396)

[deployment/privileges/templates/runPrivilegeExport\_[1\_DEV|2\_TEST|3\_PROD]\_template 86](#_Toc119006397)

[deployment/privileges/runAfterImport\_template 88](#_Toc119006398)

[Deployment Script: deployAutomation.[bat|sh] 89](#_Toc119006399)

[Deployment Script: deployProject.[bat|sh] 91](#_Toc119006400)

[Privilege Script: deployPrivs.[bat|sh] (DEPRECATED) 96](#_Toc119006401)

[deployment/privilegesBestPractices/updatePrivilegesDriverInterface 96](#_Toc119006402)

[deployment/privilegesBestPractices/validatePrivilegesDriverInterface 101](#_Toc119006403)

[10 How To Use ‘Documentation’ Procedures 106](#_Toc119006404)

[Introduction 106](#_Toc119006405)

[getDocumentationDriver 106](#_Toc119006406)

[getAllDocumentationAPI 108](#_Toc119006407)

[constants 112](#_Toc119006408)

[documentationTrigger 114](#_Toc119006409)

[helpers 115](#_Toc119006410)

[helpers/getDocConstant (Custom Function) 115](#_Toc119006411)

[helpers/getDocCounts 115](#_Toc119006412)

[helpers/parseDocSwitches (Custom Function) 116](#_Toc119006413)

[implementations 118](#_Toc119006414)

[implementations/getDocPreambleImpl1 118](#_Toc119006415)

[implementations/getDocResourceFormatImpl1 119](#_Toc119006416)

[implementations/getDocResourceFormatImpl1\_resource 120](#_Toc119006417)

[modules 123](#_Toc119006418)

[modules/getDocDataSourceLineage 123](#_Toc119006419)

[modules/getDocResourceProjection 125](#_Toc119006420)

[modules/getDocResourcesUsed 126](#_Toc119006421)

[11 How To Use ‘Encoding’ Procedures 128](#_Toc119006422)

[Introduction 128](#_Toc119006423)

[CIS\_JCE\_PROVIDERS\_VIEW 128](#_Toc119006424)

[EncodingCJP 128](#_Toc119006425)

[EncodingCJP/Base64Decode (Custom Function) 128](#_Toc119006426)

[EncodingCJP/Base64Encode (Custom Function) 129](#_Toc119006427)

[EncodingCJP/CISSecurityProviders 129](#_Toc119006428)

[EncodingCJP/DecryptFrom3DES 130](#_Toc119006429)

[EncodingCJP/DecryptFromAES 130](#_Toc119006430)

[EncodingCJP/DecryptWithCISPrivKey 131](#_Toc119006431)

[EncodingCJP/EncryptWith3DES 132](#_Toc119006432)

[EncodingCJP/EncryptWithAES 132](#_Toc119006433)

[EncodingCJP/EncryptWithCISPubKey 133](#_Toc119006434)

[EncodingCJP/MD5Hash (Custom Function) 134](#_Toc119006435)

[EncodingCJP/SHA1Hash (Custom Function) 134](#_Toc119006436)

[12 How To Use ‘Environment’ Procedures 136](#_Toc119006437)

[Introduction 136](#_Toc119006438)

[getEnvName (Custom Function) 136](#_Toc119006439)

[13 How To Use ‘File’ Procedures 137](#_Toc119006440)

[Introduction 137](#_Toc119006441)

[copyAll 137](#_Toc119006442)

[getCisHome (Custom Function) 137](#_Toc119006443)

[getFileSeparator (Custom Function) 138](#_Toc119006444)

[removeAllFilter 138](#_Toc119006445)

[FileProcessingCJP 139](#_Toc119006446)

[FileProcessingCJP/archiveFile 139](#_Toc119006447)

[FileProcessingCJP/archiveFileTimestamp 139](#_Toc119006448)

[mFileProcessingCJP/copyFile 140](#_Toc119006449)

[FileProcessingCJP/createFileASCII 140](#_Toc119006450)

[FileProcessingCJP/createFileBinary 141](#_Toc119006451)

[FileProcessingCJP/existsDir (Custom Function) 141](#_Toc119006452)

[FileProcessingCJP/existsFile (Custom Function) 142](#_Toc119006453)

[FileProcessingCJP/getFileContentsAscii (Custom Function) 142](#_Toc119006454)

[FileProcessingCJP/getFileContentsBinary (Custom Function) 142](#_Toc119006455)

[FileProcessingCJP/getFileInfo 143](#_Toc119006456)

[FileProcessingCJP/getNewFiles 144](#_Toc119006457)

[FileProcessingCJP/gunzipFile (Custom Function) 144](#_Toc119006458)

[FileProcessingCJP/makeDirs (Custom Function) 145](#_Toc119006459)

[FileProcessingCJP/removeAll (Custom Function) 145](#_Toc119006460)

[FileProcessingCJP/remove (Custom Function) 146](#_Toc119006461)

[FileProcessingCJP/unzipFile (Custom Function) 146](#_Toc119006462)

[14 How To Use ‘Generate’ Procedures 147](#_Toc119006463)

[Introduction 147](#_Toc119006464)

[generateGuid 147](#_Toc119006465)

[generateViews 147](#_Toc119006466)

[destroyDependentLineage [CONTAINER/TABLE/LINK only] 152](#_Toc119006467)

[destroyUsedLineage [CONTAINER/LINK/TABLE only] 155](#_Toc119006468)

[/helpers/createResourceProcess 157](#_Toc119006469)

[/examples/generate 158](#_Toc119006470)

[15 How To Use ‘Logging’ Procedures 160](#_Toc119006471)

[Introduction 160](#_Toc119006472)

[auditLogger (deprecated) 160](#_Toc119006473)

[auditLoggerV2 160](#_Toc119006474)

[logDebugMessage 162](#_Toc119006475)

[LogUtils 163](#_Toc119006476)

[LogUtils/GetServerMetadataLog 163](#_Toc119006477)

[16 How To Use ‘Model’ Views and Procedures 165](#_Toc119006478)

[Introduction 165](#_Toc119006479)

[/Cache/procedures/getCacheProperties 165](#_Toc119006480)

[/Cache/CACHE\_TABLE 166](#_Toc119006481)

[/Cache/CACHE\_TABLE\_PROPERTIES 166](#_Toc119006482)

[/Cache/CACHE\_TABLE\_PROPERTIES\_PIVOT 166](#_Toc119006483)

[/Cache/CACHE\_PROCEDURE 167](#_Toc119006484)

[/Cache/CACHE\_PROCEDURE\_PROPERTIES 167](#_Toc119006485)

[/Cache/CACHE\_PROCEDURE\_PROPERTIES\_PIVOT 167](#_Toc119006486)

[/Resources/procedures/getDataSourceAttributesNormalized 167](#_Toc119006487)

[/Resources/procedures/getResourceAttributesNormalized 168](#_Toc119006488)

[/Resources/procedures/getUsedResourcesDirectCursorNormalized 169](#_Toc119006489)

[/Resources/procedures/tdv\_82\_lower [folder of views] 169](#_Toc119006490)

[/Resources/COLUMN\_PROPERTIES 169](#_Toc119006491)

[/Resources/DATASOURCE\_PROPERTIES 170](#_Toc119006492)

[/Resources/PARAMETER\_PROPERTIES 170](#_Toc119006493)

[/Resources/PROCEDURE\_PROPERTIES 171](#_Toc119006494)

[/Resources/RESOURCE\_OTHER\_PROPERTIES 171](#_Toc119006495)

[/Resources/TABLE\_PROPERTIES 171](#_Toc119006496)

[/Resources/WSDL\_OPERATIONS\_PROPERTIES 172](#_Toc119006497)

[17 How To Use ‘Modify’ Procedures 173](#_Toc119006498)

[Introduction 173](#_Toc119006499)

[controllerTemplate 173](#_Toc119006500)

[18 How To Use ‘Monitor Procedures 176](#_Toc119006501)

[Introduction 176](#_Toc119006502)

[getSysCluster 176](#_Toc119006503)

[getSysRequests 177](#_Toc119006504)

[19 How To Use ‘Net’ Procedures 180](#_Toc119006505)

[Introduction 180](#_Toc119006506)

[NetUtils 180](#_Toc119006507)

[ftpFile 180](#_Toc119006508)

[20 How To Use ‘PDTool’ Procedures 181](#_Toc119006509)

[Introduction 181](#_Toc119006510)

[generatePDToolDeployableResourcePlanByDate 181](#_Toc119006511)

[generatePDToolDeployableResourcePlanByLineage 182](#_Toc119006512)

[template\_generatePDToolDeployableResourcePlan 184](#_Toc119006513)

[helpers 184](#_Toc119006514)

[helpers/getDeployableResourceListByDate 184](#_Toc119006515)

[helpers/getDeployableResourceListByLineage 185](#_Toc119006516)

[helpers/getDistinctDeployableResourceListByDate 186](#_Toc119006517)

[helpers/getDistinctDeployableResourceListByLineage 187](#_Toc119006518)

[21 How To Use ‘Repository’ Procedures 188](#_Toc119006519)

[Introduction 188](#_Toc119006520)

[CIS Types and Subtypes listing 188](#_Toc119006521)

[Listing of CIS resource types and subtypes 188](#_Toc119006522)

[The ACCESS\_TOOLS Right 190](#_Toc119006523)

[Note On Using Repository Helper Procedures With Triggers and Cache Procedures 190](#_Toc119006524)

[CIS Repository Helper Procedures 191](#_Toc119006525)

[applyReservedListToPath (Custom Function) 191](#_Toc119006526)

[applyReservedListToWord (Custom Function) 191](#_Toc119006527)

[configureReservedList 191](#_Toc119006528)

[cachedResources 191](#_Toc119006529)

[changePassword 193](#_Toc119006530)

[changeResourceOwner 193](#_Toc119006531)

[clearIntrospectableResourceIdCache 194](#_Toc119006532)

[compareCisVersions (Custom Function) 195](#_Toc119006533)

[copyResource 195](#_Toc119006534)

[copyResources 197](#_Toc119006535)

[copyResourceAnnotations 197](#_Toc119006536)

[copyResourcesPrivileges (deprecated) 198](#_Toc119006537)

[copyResourcesPrivilegesV2 198](#_Toc119006538)

[createAllFolders 200](#_Toc119006539)

[createAllFoldersPrivileges 201](#_Toc119006540)

[createConnector 202](#_Toc119006541)

[createConsumingViews 203](#_Toc119006542)

[createDataSource 205](#_Toc119006543)

[createFolder 207](#_Toc119006544)

[createOrUpdateConnector 208](#_Toc119006545)

[createResource 209](#_Toc119006546)

[createResourceCopy 210](#_Toc119006547)

[createUnionView 211](#_Toc119006548)

[deleteAllConnectors 212](#_Toc119006549)

[deleteConnector 212](#_Toc119006550)

[destroyResource 213](#_Toc119006551)

[expireProcCacheEntryByName 213](#_Toc119006552)

[exportResourceDefinitions 214](#_Toc119006553)

[exportResourcePrivileges 215](#_Toc119006554)

[findDependentResources 216](#_Toc119006555)

[findVectorInResources 218](#_Toc119006556)

[fixLeadingCharactersInFolderPath (Custom Function) 218](#_Toc119006557)

[freeUnusedMemory 219](#_Toc119006558)

[(Custom Function) 219](#_Toc119006559)

[generateOptFile 220](#_Toc119006560)

[getAllDataSourceChildren 220](#_Toc119006561)

[getAllDataSources 221](#_Toc119006562)

[getAncestorResources 222](#_Toc119006563)

[GetAnsi2NativeMapping 224](#_Toc119006564)

[getBasicResourceCursor 225](#_Toc119006565)

[getBasicResourceCursor\_ActionAttributes 227](#_Toc119006566)

[getBasicResourceCursor\_PROCEDURE 228](#_Toc119006567)

[getBasicResourceCursor\_PROCEDURE\_CURSOR 229](#_Toc119006568)

[getBasicResourceCursor\_ResourceAttributes 230](#_Toc119006569)

[getBasicResourceCursor\_SQL\_TABLE 231](#_Toc119006570)

[getBasicResourceCursor\_SQL\_TABLE\_FOREIGNKEYS 233](#_Toc119006571)

[getBasicResourceCursor\_SQL\_TABLE\_SQLINDEXES 234](#_Toc119006572)

[getBasicResourceCursor\_XSLT\_TEXT 235](#_Toc119006573)

[getChildResourcesCursor 236](#_Toc119006574)

[getCisVersion (Custom Function) 237](#_Toc119006575)

[getConnectors 238](#_Toc119006576)

[getContainer 239](#_Toc119006577)

[getDataSourceAttributeDefs 239](#_Toc119006578)

[getDataSourceAttributes 241](#_Toc119006579)

[getDataSourceAttributesV2 242](#_Toc119006580)

[getDataSourceCacheConfig 243](#_Toc119006581)

[getDatasourceListByType 243](#_Toc119006582)

[getDataSourceRootPath 245](#_Toc119006583)

[getDataSourceStatsConfig 246](#_Toc119006584)

[getDefSetDefs 247](#_Toc119006585)

[getDependentResourcesCursor 248](#_Toc119006586)

[getDependentResourcesRecurseCursor 249](#_Toc119006587)

[getDependentResourcesDirectCursor 250](#_Toc119006588)

[getDependentResourcesDirectRecurseCursor 251](#_Toc119006589)

[getImpactedResources 252](#_Toc119006590)

[getIntrospectableResourceIdsResult 253](#_Toc119006591)

[getIntrospectableResourceIdsTask 254](#_Toc119006592)

[getIntrospectedResourceIdsResult 256](#_Toc119006593)

[getIntrospectedResourceIdsTask 257](#_Toc119006594)

[getLockedResources 257](#_Toc119006595)

[getOutputColDefs 258](#_Toc119006596)

[getResourceAnnotations 259](#_Toc119006597)

[getResourceCacheConfig 260](#_Toc119006598)

[getResourceCacheConfigCursor 260](#_Toc119006599)

[getResourceCreated 262](#_Toc119006600)

[getResourceImpactedCursor 262](#_Toc119006601)

[getResourceLastModified 263](#_Toc119006602)

[getResourceLineageDatasources 264](#_Toc119006603)

[getResourceLineageRecursive 265](#_Toc119006604)

[getResourceLineageDependencies 268](#_Toc119006605)

[getResourceLineageDirectRecursive 274](#_Toc119006606)

[getResourceLineageRecursiveAncestors 277](#_Toc119006607)

[getResourceListByAnnotation 280](#_Toc119006608)

[getResourceListChildren 284](#_Toc119006609)

[getResourceListRecursive 286](#_Toc119006610)

[getResourceListUnpublished 288](#_Toc119006611)

[getResourcePrivilegeDependencies 289](#_Toc119006612)

[getResourcePrivileges 291](#_Toc119006613)

[getResourcePrivilegesByUser 294](#_Toc119006614)

[getResourcePrivilegesGroupsUsers 294](#_Toc119006615)

[getResourcesByDate 297](#_Toc119006616)

[getResourceSqlTable 299](#_Toc119006617)

[getScriptText (Custom Function) 300](#_Toc119006618)

[getTableColumnStatisticsConfiguration 301](#_Toc119006619)

[getTrigger 302](#_Toc119006620)

[getUsedResourcesCursor 305](#_Toc119006621)

[getUsedResourcesRecurseCursor 306](#_Toc119006622)

[getUsedResourcesDirectCursor 308](#_Toc119006623)

[getUsedResourcesDirectRecurseCursor 309](#_Toc119006624)

[getUserPermissionsRecursive 310](#_Toc119006625)

[impactedTargetsList 311](#_Toc119006626)

[importResourcePrivileges 312](#_Toc119006627)

[introspectResources 313](#_Toc119006628)

[introspectResourcesResultCursor 317](#_Toc119006629)

[migrateDatasource 318](#_Toc119006630)

[moveResource 323](#_Toc119006631)

[parseAnnotationLabel 323](#_Toc119006632)

[performServerAction 327](#_Toc119006633)

[publishResource 328](#_Toc119006634)

[rebindFolder 329](#_Toc119006635)

[rebindResource 330](#_Toc119006636)

[recoverFailedCacheRefresh 331](#_Toc119006637)

[refreshResourceStatistics 332](#_Toc119006638)

[reintrospectDataSource 333](#_Toc119006639)

[removeAllFolders 333](#_Toc119006640)

[removePathQuotes 334](#_Toc119006641)

[renameResource 334](#_Toc119006642)

[replaceStringInAnnotations 335](#_Toc119006643)

[replaceStringInResources 336](#_Toc119006644)

[returnFolderNameAndFolderPath 338](#_Toc119006645)

[searchAnnotations 338](#_Toc119006646)

[searchResources 340](#_Toc119006647)

[translateResourceType 343](#_Toc119006648)

[updateBasicTransformationProcedure 345](#_Toc119006649)

[updateConnector 345](#_Toc119006650)

[updateDefSetDef 346](#_Toc119006651)

[UpdateDsColumnAnnotation 347](#_Toc119006652)

[updateExternalSQLProcedure 348](#_Toc119006653)

[updateImpactedResource 349](#_Toc119006654)

[updateImpactedResources 349](#_Toc119006655)

[updateResourceAnnotations (deprecated) 350](#_Toc119006656)

[updateResourceAnnotationsV2 350](#_Toc119006657)

[updateResourceCacheConfig 351](#_Toc119006658)

[updateResourceCacheConfiguration (deprecated) 352](#_Toc119006659)

[updateResourceCacheConfigurationV2 352](#_Toc119006660)

[updateResourceCacheEnabled 357](#_Toc119006661)

[updateResourceDataSource (deprecated) 358](#_Toc119006662)

[updateResourceDataSourceV2 358](#_Toc119006663)

[updateResourceEnabled 360](#_Toc119006664)

[updateResourceOwner 360](#_Toc119006665)

[updateResourcePrivileges (deprecated) 361](#_Toc119006666)

[updateResourcePrivilegesV2 361](#_Toc119006667)

[updateResourcesSqlTable (deprecated) 364](#_Toc119006668)

[updateResourcesSqlTableV2 364](#_Toc119006669)

[updateSqlScript 366](#_Toc119006670)

[updateSqlTable 367](#_Toc119006671)

[updateSqlTableTextAndModel 368](#_Toc119006672)

[updateStreamTransformationProcedure 368](#_Toc119006673)

[updateTableColumnStatisticsConfiguration 369](#_Toc119006674)

[updateTrigger 370](#_Toc119006675)

[updateXsltTransformationProcedure 371](#_Toc119006676)

[RepoUtils 372](#_Toc119006677)

[RepoUtils/applyReservedListToPath 372](#_Toc119006678)

[RepoUtils/applyReservedListToWord 373](#_Toc119006679)

[RepoUtils/EncryptPassword (Custom Function) 373](#_Toc119006680)

[RepoUtils/ForceWriteRepoUtils 374](#_Toc119006681)

[RepoUtils/GetAnsi2NativeMapping 374](#_Toc119006682)

[RepoUtils/getReservedWordList 375](#_Toc119006683)

[RepoUtils/GetSystemProperties 376](#_Toc119006684)

[RepoUtils/GetUserGroups 376](#_Toc119006685)

[RepoUtils/isReservedWord (Custom Function) 377](#_Toc119006686)

[RepoUtils/UpdateDsColumnAnnotation 377](#_Toc119006687)

[CIS Repository Definition Procedures 378](#_Toc119006688)

[definitions/RepositoryDefinitions 378](#_Toc119006689)

[definitions/RepositoryDefinitionsRecursive 378](#_Toc119006690)

[CIS Repository Execute Procedures 378](#_Toc119006691)

[execute/executeProcedure 378](#_Toc119006692)

[execute/executeProcedureResults 379](#_Toc119006693)

[CIS Repository Server Procedures 380](#_Toc119006694)

[server/addLicense 380](#_Toc119006695)

[server/compareServerAttributes 380](#_Toc119006696)

[server/getServerAttribute (DEPRECATED) 383](#_Toc119006697)

[server/getServerAttributeList (DEPRECATED) 384](#_Toc119006698)

[server/getServerAttributeMap (DEPRECATED) 384](#_Toc119006699)

[server/getServerAttributeMapByKey (DEPRECATED) 385](#_Toc119006700)

[server/getServerAttributesAll 385](#_Toc119006701)

[server/getSpecifiedServerAttributes 387](#_Toc119006702)

[server/updateServerAttribute 390](#_Toc119006703)

[server/updateServerConfiguration 391](#_Toc119006704)

[CIS Repository User Procedures 393](#_Toc119006705)

[user/createGroup 394](#_Toc119006706)

[user/createResourcePrivilege 394](#_Toc119006707)

[user/createUser 395](#_Toc119006708)

[user/deleteGroup 396](#_Toc119006709)

[user/deleteUser 396](#_Toc119006710)

[user/getDomainGroups 397](#_Toc119006711)

[user/getDomains 398](#_Toc119006712)

[user/getDomainUsers 398](#_Toc119006713)

[user/getDomainUsersConsolidated 399](#_Toc119006714)

[user/getGroup 401](#_Toc119006715)

[user/getUser 402](#_Toc119006716)

[user/removeUserFromGroups 403](#_Toc119006717)

[user/updateUser 404](#_Toc119006718)

[22 How To Use ‘Request’ Procedures 405](#_Toc119006719)

[Introduction 405](#_Toc119006720)

[getAllViewsSqlPlan 405](#_Toc119006721)

[getSqlPlan 407](#_Toc119006722)

[getSqlPlanChild (helper) 410](#_Toc119006723)

[terminateIdleSessions 412](#_Toc119006724)

[terminateRequest 412](#_Toc119006725)

[terminateSession 413](#_Toc119006726)

[RequestUtils 413](#_Toc119006727)

[RequestUtils/DirectSqlRequest (Custom Function) 414](#_Toc119006728)

[RequestUtils/OriginalRequest (Custom Function) 414](#_Toc119006729)

[RequestUtils/ReadInEqClause (Custom Function) 414](#_Toc119006730)

[RequestUtils/TopSqlRequest (Custom Function) 415](#_Toc119006731)

[23 How To Use ‘Security Procedures 416](#_Toc119006732)

[Introduction 416](#_Toc119006733)

[cbsAddAssignments 416](#_Toc119006734)

[cbsCreatePolicies 417](#_Toc119006735)

[cbsCreatePolicyFromXML 418](#_Toc119006736)

[cbsDeletePolicies 419](#_Toc119006737)

[cbsGetAssignments 420](#_Toc119006738)

[cbsGetPolicies 421](#_Toc119006739)

[cbsGetPolicy 422](#_Toc119006740)

[24 How To Use ‘String’ Procedures 423](#_Toc119006741)

[Introduction 423](#_Toc119006742)

[encdodedValues (SQL Definition Set) 423](#_Toc119006743)

[addQuotesInList (Custom Function) 423](#_Toc119006744)

[basename (Custom Function) 423](#_Toc119006745)

[concatNotNull (Custom Function) 424](#_Toc119006746)

[dirname (Custom Function) 425](#_Toc119006747)

[emptyStr (Custom Function) 425](#_Toc119006748)

[entityConstants 426](#_Toc119006749)

[entityExtract 426](#_Toc119006750)

[entityExtractToPipe 427](#_Toc119006751)

[entityExtractToString (Custom Function) 428](#_Toc119006752)

[escapeCSV (Custom Function) 429](#_Toc119006753)

[extractBiDelimitedText (Custom Function) 430](#_Toc119006754)

[extractDelimitedText 431](#_Toc119006755)

[extractTextList 433](#_Toc119006756)

[findOpenClosePair 435](#_Toc119006757)

[findString (Custom Function) 436](#_Toc119006758)

[findStringInList (Custom Function) 437](#_Toc119006759)

[fixQuotes (Custom Function) 438](#_Toc119006760)

[getCodedString 438](#_Toc119006761)

[getConstant (Custom Function) [DEPRECATED. Use getConstantV2] 440](#_Toc119006762)

[getConstantV2 (Custom Function) 440](#_Toc119006763)

[getDelimitedOccurrence (Custom Function) 442](#_Toc119006764)

[getDelimitedSum (Custom Function) 442](#_Toc119006765)

[indent (Custom Function) 443](#_Toc119006766)

[isEmpty (Custom Function) 443](#_Toc119006767)

[joinCursorByDelimiter 444](#_Toc119006768)

[joinVectorByDelimiter 445](#_Toc119006769)

[last4ofSSN (Custom Function) 445](#_Toc119006770)

[modifyConstant (Custom Function) 446](#_Toc119006771)

[normalizeRowsToPipe 447](#_Toc119006772)

[normalizeRowsToString 447](#_Toc119006773)

[numOccurrences 448](#_Toc119006774)

[p\_DelimitedStringToCursor 448](#_Toc119006775)

[p\_FixedStringToCursor 449](#_Toc119006776)

[ParseCSVLine 450](#_Toc119006777)

[removeDoubleQuotes 450](#_Toc119006778)

[removeSingleQuotes 451](#_Toc119006779)

[RegexPatterns 451](#_Toc119006780)

[splitByDelimiter 452](#_Toc119006781)

[TextUtils 453](#_Toc119006782)

[TextUtils/Blob2Varchar (Custom Function) 453](#_Toc119006783)

[TextUtils/CCNumberFormatter (Custom Function) 453](#_Toc119006784)

[TextUtils/CSVFromCISQuery (Custom Function) 454](#_Toc119006785)

[TextUtils/CSVFromCISQueryToFile 454](#_Toc119006786)

[TextUtils/FixedFromCISQuery (Custom Function) 455](#_Toc119006787)

[TextUtils/FixedFromCISQueryToFile 456](#_Toc119006788)

[TextUtils/FormatXML (Custom Function) 457](#_Toc119006789)

[TextUtils/GenerateGUID 458](#_Toc119006790)

[TextUtils/HexToRaw (Custom Function) 458](#_Toc119006791)

[TextUtils/LocalCurrencyFormatter (Custom Function) 459](#_Toc119006792)

[TextUtils/LocalCurrencyParser (Custom Function) 459](#_Toc119006793)

[TextUtils/LocalDateFormatter (Custom Function) 460](#_Toc119006794)

[TextUtils/LocalDateParser (Custom Function) 461](#_Toc119006795)

[TextUtils/LocalNumberFormatter (Custom Function) 462](#_Toc119006796)

[TextUtils/LocalNumberParser (Custom Function) 462](#_Toc119006797)

[TextUtils/LocalTimeFormatter (Custom Function) 463](#_Toc119006798)

[TextUtils/LocalTimeParser (Custom Function) 464](#_Toc119006799)

[TextUtils/LocalTimestampFormatter (Custom Function) 465](#_Toc119006800)

[TextUtils/LocalTimestampParser (Custom Function) 465](#_Toc119006801)

[TextUtils/PhoneNumberFormatter (Custom Function) 466](#_Toc119006802)

[TextUtils/RawToHex (Custom Function) 467](#_Toc119006803)

[TextUtils/RegexCount (Custom Function) 468](#_Toc119006804)

[TextUtils/RegexFind (Custom Function) 468](#_Toc119006805)

[TextUtils/RegexGetGroups 469](#_Toc119006806)

[TextUtils/RegexPosition (Custom Function) 470](#_Toc119006807)

[TextUtils/RegexReplace (Custom Function) 471](#_Toc119006808)

[TextUtils/RegexSplit 472](#_Toc119006809)

[TextUtils/SSNumberFormatter (Custom Function) 473](#_Toc119006810)

[TextUtils/URLDecode 474](#_Toc119006811)

[TextUtils/URLEncode 474](#_Toc119006812)

[25 How To Use ‘Templates’ Procedures 476](#_Toc119006813)

[Introduction 476](#_Toc119006814)

[procedureTemplate 476](#_Toc119006815)

[26 How To Use ‘Time’ Procedures 477](#_Toc119006816)

[Introduction 477](#_Toc119006817)

[DefaultValues (deprecated) 477](#_Toc119006818)

[extractDate (Custom Function) 477](#_Toc119006819)

[extractTime (Custom Function) 478](#_Toc119006820)

[extractTimestamp (Custom Function) 479](#_Toc119006821)

[getCurrentTimestamp (Custom Function) 479](#_Toc119006822)

[getTimestampInterval (Custom Function) 480](#_Toc119006823)

[intervalDay2Seconds (Custom Function) 480](#_Toc119006824)

[period2IntervalDay (Custom Function) 481](#_Toc119006825)

[DateUtils 481](#_Toc119006826)

[BigintToTimestamp (Custom Function) 482](#_Toc119006827)

[DateUtils/DateAddDate (Custom Function) 482](#_Toc119006828)

[DateUtils/DateAddTimestamp (Custom Function) 483](#_Toc119006829)

[DateUtils/DateDiffDate (Custom Function) 483](#_Toc119006830)

[DateUtils/DateDiffTimestamp (Custom Function) 484](#_Toc119006831)

[DateUtils/GetServerTimezone (Custom Function) 484](#_Toc119006832)

[TimestampToBigint (Custom Function) 485](#_Toc119006833)

[DateUtils/TZConverter (Custom Function) 486](#_Toc119006834)

[27 How To Use ‘Upgrade’ Procedures 487](#_Toc119006835)

[Introduction 487](#_Toc119006836)

[getDatabaseTests 487](#_Toc119006837)

[getServiceTests 487](#_Toc119006838)

[updateCacheConfigTables 487](#_Toc119006839)

[helpers 488](#_Toc119006840)

[helpers/configuredCaches 488](#_Toc119006841)

[helpers/findCaches 489](#_Toc119006842)

[helpers/returnColumnOrderingString 489](#_Toc119006843)

[28 How To Use ‘Validate’ Procedures 491](#_Toc119006844)

[Introduction 491](#_Toc119006845)

[compareMetadataTargetServer 491](#_Toc119006846)

[compareResourcesTargetServer 492](#_Toc119006847)

[validateDeployment 493](#_Toc119006848)

[validatePublishedResources 494](#_Toc119006849)

[validateSharedResourcesRules 495](#_Toc119006850)

[helpers 497](#_Toc119006851)

[helpers/evaluateResource 497](#_Toc119006852)

[helpers/evaluateResourceOrderBy 499](#_Toc119006853)

[29 How To Use ‘XML’ Procedures 500](#_Toc119006854)

[Introduction 500](#_Toc119006855)

[castXMLTextNodeAsVarchar (Custom Function) 500](#_Toc119006856)

[escapeXML (Custom Function) 500](#_Toc119006857)

[extractXMLOccurrence 501](#_Toc119006858)

[getNodeFromXML 503](#_Toc119006859)

[getValueFromXML (Custom Function) 504](#_Toc119006860)

[getXMLNode 505](#_Toc119006861)

[parseAndModifyXML (deprecated) 507](#_Toc119006862)

[parseAndModifyXMLV2 507](#_Toc119006863)

[parseProperties 509](#_Toc119006864)

[pruneXML (deprecated) 510](#_Toc119006865)

[pruneXMLV2 510](#_Toc119006866)

[reverseXML (Custom Function) 512](#_Toc119006867)

[stripInvalidXMLChars (Custom Function) 512](#_Toc119006868)

[unescapeXML (Custom Function) 512](#_Toc119006869)

[XMLUtils 513](#_Toc119006870)

[XMLUtils/CSVFromXMLToFile 513](#_Toc119006871)

[XMLUtils/DeleteElement (Custom Function) 515](#_Toc119006872)

[XMLUtils/DeleteElementSpareChildren (Custom Function) 516](#_Toc119006873)

[XMLUtils/FixedFromXMLToFile 517](#_Toc119006874)

[XMLUtils/HTMLtoXML 519](#_Toc119006875)

[XMLUtils/InsertElementDemoteChildren (Custom Function) 519](#_Toc119006876)

[30 How To Submit New Procedures 521](#_Toc119006877)

[Introduction 521](#_Toc119006878)

[Documentation 521](#_Toc119006879)

[Regression Test Cases 522](#_Toc119006880)

[Source Code Control 522](#_Toc119006881)

[Peer Review 522](#_Toc119006882)

[Team Members 522](#_Toc119006883)

1. Release Notes

## Introduction

This section provides release notes for the latest release of the Utilities. Please review the Deprecated Resources section carefully for resources that have been removed or will be removed from the Utilities distribution soon.

## Regression Test Versions

The following list contains the version/patch/hotfix levels used for each regression test:

**8.4.0.003**

## New Resources

All of the following resources are related to creating a version of the original resource but without the use of vectors. The use of a vector is associated with a memory leak issue.

### 2022 Q400

/shared/ASAssets/Utilities/environment/getServerHostPort

/shared/ASAssets/Utilities/model – entire folder is new

/shared/ASAssets/Utilities/cache/getCacheStatusRows

/shared/ASAssets/Utilities/repository/getColumnDependencies

/shared/ASAssets/Utilities/repository/getColumnDependenciesDerivations

/shared/ASAssets/Utilities/repository/performServerAction

### 2022 Q200

/shared/ASAssets/Utilities/security/cbs/cbsAddAssignments

/shared/ASAssets/Utilities/security/cbs/cbsCreatePolicies

/shared/ASAssets/Utilities/security/cbs/cbsCreatePolicyFromXML

/shared/ASAssets/Utilities/security/cbs/cbsDeletePolicies

/shared/ASAssets/Utilities/security/cbs/cbsGetAssignments

/shared/ASAssets/Utilities/security/cbs/cbsGetPolicies

/shared/ASAssets/Utilities/security/cbs/cbsGetPolicy

### 2022 Q100

/shared/ASAssets/Utilities/repository/getDataSourceAttributeDefs

/shared/ASAssets/Utilities/repository/getDataSourceAttributesV2

/shared/ASAssets/Utilities/repository/migrateDatasource

/shared/ASAssets/Utilities/repository/renameResource

### 2021 Q402

/shared/ASAssets/Utilities/request/getAllViewsSqlPlan

/shared/ASAssets/Utilities/archive/getPkgExportListByLabel

/shared/ASAssets/Utilities/repository/getResourceListByAnnotation

/shared/ASAssets/Utilities/repository/parseAnnotationLabel

/shared/ASAssets/Utilities/repository/translateResourceType

### 2021 Q401

/shared/ASAssets/Utilities/request/getSqlPlan

/shared/ASAssets/Utilities/request/getSqlPlanChild

/shared/ASAssets/Utilities/xml/getXMLNode

/shared/ASAssets/Utilities/xml/parseProperties

### 2021 Q201

/shared/ASAssets/Utilities/repository/getResourceLineageDependencies

/shared/ASAssets/Utilities/repository/lowerLevelProcedures/getResourceLineageDependenciesDerivations

/services/databases/ASAssets/Utilities/deployment/getOptionsFileDuplicateTokens

/shared/ASAssets/Utilities/deployment/optionsfile/getOptionsFileDuplicateTokens

/shared/ASAssets/Utilities/deployment/optionsfile/listOptionsFileTokens

### 2021 Q200

/services/databases/ASAssets/Utilities/deployment/generateOptionsFile

/services/databases/ASAssets/Utilities/deployment/generateOptionsFileRebind

/services/databases/ASAssets/Utilities/repository/updateResourceCacheEnabled

/shared/ASAssets/Utilities/archive/lowerLevelProcedures/getArchiveExportDataXSLTNonQualified

/shared/ASAssets/Utilities/archive/lowerLevelProcedures/getArchiveExportDataXSLTQualified

/services/databases/ASAssets/Utilities/deployment/validatePrivilegesDriverInterface

/shared/ASAssets/Utilities/deployment/privilegesBestPractices/validatePrivilegesDriverInterface

### 2021 Q102

/shared/ASAssets/Utilities/archive/generateExportList

/shared/ASAssets/Utilities/archive/lowerLevelProcedures/getArchiveExportDataXSLTNonQualified

/shared/ASAssets/Utilities/archive/lowerLevelProcedures/getArchiveExportDataXSLTQualified

### 2021 Q101

/shared/ASAssets/Utilities/modify/controllerTemplate

/shared/ASAssets/Utilities/modify/example [example folder]

/shared/ASAssets/Utilities/modify/helpers [code folder]

### 2021 Q100

/shared/ASAssets/Utilities/repository/getDatasourceListByType

/shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateRebindList

/shared/ASAssets/Utilities/deployment/optionsfile/generateOptionsFileRebind

### 2020 Q402

/services/databases/ASAssets/Utilities/monitor/getSysCluster

/services/databases/ASAssets/Utilities/monitor/getSysRequests

/shared/ASAssets/Utilities/monitor/getSysCluster

/shared/ASAssets/Utilities/monitor/getSysRequests

/shared/ASAssets/Utilities/repository/user/getDomainUsersConsolidated

### 2022 Q300

/shared/ASAssets/Utilities/repository/introspectResources

/shared/ASAssets/Utilities/repository/server/updateServerConfiguration

### 2020 Q400

No new resources added in this release.

### 2020 Q300

Ability to set environment properties

/shared/ASAssets/Utilities/environment/setEnvironmentProperties

Ability to export a car file

/shared/ASAssets/Utilities/archive/exportResources

/shared/ASAssets/Utilities/archive/exportArchiveFile

/shared/ASAssets/Utilities/archive/getExportResourceList

RepoUtils.properties – added reserved words “host” and “port” required by 8.3.

### 2020 Q202

Deployment Automation Scripts: deployAutomation.sh and deployAutomation.bat

### 2020 Q201

/shared/ASAssets/Utilities/repository/user/removeUserFromGroups

Published the following resources to ASAssets.Utlities.repository

getBasicResourceCursor\_PROCEDURE\_CURSOR

getResourceListRecursive

getResourceLineageDirectRecursive

getUsedResourcesDirectCursor

### 2020 Q200

/shared/ASAssets/Utilities/repository/freeUnusedMemory

### 2020 Q101

/shared/ASAssets/Utilities/repository/findDependentResources

/shared/ASAssets/Utilities/repository/getTrigger

/shared/ASAssets/Utilities/repository/user/updateUser

### 2020 Q100

/cache/cacheRefreshSynch

/cache/getCacheAgeHours

/deployment/privilegesBestPractices/updatePrivilegesDriverInterface

/generate/generateGuid

/generate/template/localRestServices

/repository/copyResource

/repository/moveResource

/repository/updateResourceEnabled

/repository/server/compareServerAttributes

/repository/server/getServerAttributesAll

/repository/server/getSpecifiedServerAttributes

/repository/server/updateServerConfiguration

The following are new published resources:

/services/databases/ASAssets/Utilities

/deployment

/updatePrivilegesDriverInterface

/validateDeployment

/repository

/getBasicResourceCursor

/getBasicResourceCursor\_ResourceAttributes

/getServerAttributeDefs

/getServerAttributesAll

/GetServerTimezone

### 2019 Q301

/string/getConstantV2 – replaces getConstant

### 2019 Q200

/log/auditLoggerV2

/repository/copyResourcesPrivilegesV2

/repository/updateResourceAnnotationsV2

/repository/updateResourceCacheConfigurationV2

/repository/updateResourceDataSourceV2

/repository/updateResourcePrivilegesV2

/repository/updateResourcesSqlTableV2

/xml/parseAndModifyXMLV2

/xml/pruneXMLV2

The following are new capabilities:

/deployment/optionsfile/helpers/generateOptions

/deployment/optionsfile/generateOptionsFile

/deployment/privileges/helpers/exportResourcesPrivileges

/deployment/privileges/helpers/getResourcesPrivileges

/deployment/privileges/helpers/splitResourceType

/deployment/privileges/templates/runPrivilegeExport\_1\_DEV\_template

/deployment/privileges/templates/runPrivilegeExport\_2\_TEST\_template

/deployment/privileges/templates/runPrivilegeExport\_3\_PROD\_template

/deployment/privileges/importResourceOwnership

/deployment/privileges/importResourcePrivileges

/deployment/run/runAfterImport\_template

/repository/changeResourceOwner

/repository/findVectorInResources

/repository/getResourcePrivilegesGroupsUsers

/repository/user/getDomains

/string/getCodedString – migrated from KPImetrics

/string/encodedValues – migrated from KPImetrics

/string/findOpenClosePair – migrated from KPImetrics

/validate/helpers/evaluateResource

/validate/helpers/evaluateResourceOrderBy

/validate/compareMetadataTargetServer

/validate/compareResourcesTargetServer

/validate/validatePublishedResources

/validate/validateSharedResourcesRules

/xml/extractXMLOccurrence

The following is a new published datasource:

/services/databases/ASAssets/Utilities

/deployment

/importResourceOwnership

/importResourcePrivileges

/repository

/getBasicResourceCursor\_SQL\_TABLE

/getScriptText

/getServerAttribute

## Updated Resources

The following resources were updated.

### 2022 Q400

/shared/ASAssets/Utilities/generate/generateViews

/shared/ASAssets/Utilities/log/auditLoggerV2

/shared/ASAssets/Utilities/repository/getResourceLineageDependencies

/shared/ASAssets/Utilities/repository/introspectResources

/shared/ASAssets/Utilities/repository/introspectResourcesResultCursor

/shared/ASAssets/Utilities/repository/migrateDatasource

### 2022 Q301

/shared/ASAssets/Utilities/request/getSqlPlan

/shared/ASAssets/Utilities/request/getAllViewsSqlPlan

/shared/ASAssets/Utilities/xml/escapeXML

/shared/ASAssets/Utilities/xml/unescapeXML

/shared/ASAssets/Utilities/generate/generateViews

/shared/ASAssets/Utilities/generate/destroyDependentLineage

/shared/ASAssets/Utilities/string/modifyConstant

### 2022 Q300

/shared/ASAssets/Utilities/repository/introspectResources

/shared/ASAssets/Utilities/repository/server/updateServerConfiguration

### 2022 Q200

/shared/ASAssets/Utilities/generate/generateViews

/shared/ASAssets/Utilities/generate/helpers/createResourceProcess

/shared/ASAssets/Utilities/repository/getResourceListChildren

/shared/ASAssets/Utilities/string/getConstantV2

### 2022 Q100

/shared/ASAssets/Utilities/generate/generateViews

/shared/ASAssets/Utilities/generate/helpers/createResourceProcess

/shared/ASAssets/Utilities/repository/introspectResources

### 2021 Q400

/shared/ASAssets/Utilities/deployment/validate/validateDeployment

### 2021 Q300

/shared/ASAssets/Utilities/cache/cacheRefreshSynch

### 2021 Q201

/shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateOptions

/shared/ASAssets/Utilities/deployment/optionsfile/generateOptionsFile

/shared/ASAssets/Utilities/deployment/validate/helpers/validateDeploymentInit

/shared/ASAssets/Utilities/repository/server/compareServerAttributes

/shared/ASAssets/Utilities/repository/getImpactedResources

/shared/ASAssets/Utilities/repository/impactedTargetsList

/shared/ASAssets/Utilities/repository/updateImpactedResource

/shared/ASAssets/Utilities/validate/compareMetadataTargetServer

/shared/ASAssets/Utilities/validate/validatePublishedResources

/shared/ASAssets/Utilities/validate/validateSharedResourcesRules

/shared/ASAssets/Utilities\_regression\_tests/encoding/EncodingCJP/Test\_DecryptWithCISPrivKey

/shared/ASAssets/Utilities\_regression\_tests/encoding/EncodingCJP/Test\_EncryptWithCISPubKey

### 2021 Q200

/shared/ASAssets/Utilities/archive/backup\_export

/shared/ASAssets/Utilities/archive/exportArchiveFile

/shared/ASAssets/Utilities/archive/exportResources

/shared/ASAssets/Utilities/archive/importArchiveFile

/shared/ASAssets/Utilities/deployment/optionsfile/generateOptionsFile

/shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateOptions

/shared/ASAssets/Utilities/deployment/validate/helpers/validateDeploymentInit

/shared/ASAssets/Utilities/deployment/validate/helpers/validateDeploymentInit\_CreateDrop

/shared/ASAssets/Utilities/repository/server/updateServerAttribute

/shared/ASAssets/Utilities/repository/server/updateServerConfiguration

/shared/ASAssets/Utilities/upgrade/helpers/getDatabaseTests

Removed the following:

/shared/ASAssets/Utilities/archive/lowerLevelProcedures/getArchiveExportDataXSLT

### 2021 Q102

/shared/ASAssets/Utilities/archive/backup\_export

/shared/ASAssets/Utilities/archive/exportArchiveFile

/shared/ASAssets/Utilities/archive/exportResources

/shared/ASAssets/Utilities/archive/importArchiveFile

/shared/ASAssets/Utilities/string/getConstantV2

/shared/ASAssets/Utilities/string/modifyConstant

Removed the following:

/shared/ASAssets/Utilities/archive/lowerLevelProcedures/getArchiveExportDataXSLT

### 2021 Q100

/shared/ASAssets/Utilities/repository/updateResourcePrivilegesV2

/shared/ASAssets/Utilities\_Examples/examples/repository/test\_updateResourcesPrivileges

/shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateOptions

/shared/ASAssets/Utilities/deployment/optionsfile/generateOptionsFile

/shared/ASAssets/Utilities/deployment/validate/validateDeployment

/shared/ASAssets/Utilities/repository/getBasicResourceCursor

/shared/ASAssets/Utilities/repository/getResourceLastModified

/shared/ASAssets/Utilities/repository/getResourceCreated

/shared/ASAssets/Utilities/repository/getTrigger

/shared/ASAssets/Utilities/repository/getResourceListChildren

/shared/ASAssets/Utilities/repository/getAncestorResources

/shared/ASAssets/Utilities/repository/reintrospectDataSource

/shared/ASAssets/Utilities/validate/helpers/evaluateResource

### 2020 Q402

/shared/ASAssets/Utilities/repository/user/createUser

/shared/ASAssets/Utilities/repository/RepoUtils - java - added keywords in RepoUtils - [D,SOURCE,T,HOST,PORT,TRANSACTIONS,CLUSTER,STORED,SUPPORTING,AVRO,JSON,ORC,BUCKETS,RCFILE,SEQUENCEFILE,PARQUET]

/shared/ASAssets/Utilities/repository/getAncestorResources

/shared/ASAssets/Utilities/repository/getBasicResourceCursor

/shared/ASAssets/Utilities/repository/getBasicResourceCursor\_SQL\_TABLE

/shared/ASAssets/Utilities/repository/getBasicResourceCursor\_PROCEDURE\_CURSOR

/shared/ASAssets/Utilities/repository/getResourceListChildren

/shared/ASAssets/Utilities/repository/introspectResources

### 2020 Q401

/shared/ASAssets/Utilities/repository/server/updateServerAttribute

/shared/ASAssets/Utilities/repository/server/updateServerConfiguration

### 2020 Q400

Added XML unescape characters for all attributes. Added XMLTEXT to escape path and name input variables.

/shared/ASAssets/Utilities/archive/exportArchiveFile

/shared/ASAssets/Utilities/archive/getExportResourceList

/shared/ASAssets/Utilities/archive/importArchiveFile

/shared/ASAssets/Utilities/repository/copyResource

/shared/ASAssets/Utilities/repository/getAncestorResources

/shared/ASAssets/Utilities/repository/getBasicResourceCursor

/shared/ASAssets/Utilities/repository/getBasicResourceCursor\_PROCEDURE\_CURSOR

/shared/ASAssets/Utilities/repository/getBasicResourceCursor\_SQL\_TABLE

/shared/ASAssets/Utilities/repository/getResourceListChildren

/shared/ASAssets/Utilities/repository/getResourceSqlTable

/shared/ASAssets/Utilities/repository/getTableColumnStatisticsConfiguration

/shared/ASAssets/Utilities/repository/introspectResources

/shared/ASAssets/Utilities/repository/introspectResourcesResultCursor

/shared/ASAssets/Utilities/repository/moveResource

/shared/ASAssets/Utilities/repository/updateResourceEnabled

/shared/ASAssets/Utilities/repository/updateResourceOwner

/shared/ASAssets/Utilities/repository/updateTableColumnStatisticsConfiguration

/shared/ASAssets/Utilities/repository/user/createResourcePrivilege

/shared/ASAssets/Utilities/repository/execute/executeProcedure

/shared/ASAssets/Utilities/repository/execute/executeProcedureResults

### 2020 Q300

/shared/ASAssets/Utilities/request/terminateIdleSessions – 8.3 host reserved word

/shared/ASAssets/Utilities/pdtool/helpers/getDeployableResourceListByDate

/shared/ASAssets/Utilities/pdtool/helpers/getDeployableResourceListByLineage

/shared/ASAssets/Utilities/repository/definitions/RepositoryDefinitionsRecursive - lineageTreeType

/shared/ASAssets/Utilities/repository/getBasicResourceCursor – LTRIM on resourcePath

Return creation and modification attributes:

/shared/ASAssets/Utilities/repository/getResourceLineageDirectRecursive

/shared/ASAssets/Utilities/repository/getResourceLineageRecursive

/shared/ASAssets/Utilities/repository/getResourceLineageRecursiveAncestors

/shared/ASAssets/Utilities/repository/getResourcesByDate

/shared/ASAssets/Utilities/reintrospectCJPs

### 2020 Q202

/shared/ASAssets/Utilities/deployment/validate/validateDeployment

/shared/ASAssets/Utilities/deployment/validate/helpers/validateDeploymentInit

/shared/ASAssets/Utilities/deployment/validate/helpers/validateDeploymentInit\_CreateDrop

Deployment Scripts: deployProject.sh and deployProject.bat

### 2020 Q201

/shared/ASAssets/Utilities/deployment/validate/helpers/validateDeploymentInit

/shared/ASAssets/Utilities/pdtool/helpers/getDeployableResourceListByDate

/shared/ASAssets/Utilities/pdtool/helpers/getDeployableResourceListByLineage

/shared/ASAssets/Utilities/repository/copyResources

/shared/ASAssets/Utilities/repository/getAllDataSources

/shared/ASAssets/Utilities/repository/getAncestorResources

/shared/ASAssets/Utilities/repository/getBasicResourceCursor

/shared/ASAssets/Utilities/repository/getResourceListChildren

/shared/ASAssets/Utilities/repository/getResourceListRecursive

/shared/ASAssets/Utilities/repository/getResourceLineageDirectRecursive

/shared/ASAssets/Utilities/repository/getResourceLineageRecursiveAncestors

/shared/ASAssets/Utilities/repository/getResourcesByDate

/shared/ASAssets/Utilities/repository/server/getSpecifiedServerAttributes

/shared/ASAssets/Utilities/repository/rebindFolder

/shared/ASAssets/Utilities/repository/user/createGroup

/shared/ASAssets/Utilities/repository/user/deleteGroup

/shared/ASAssets/Utilities/repository/user/createUser

/shared/ASAssets/Utilities/repository/user/deleteUser

/shared/ASAssets/Utilities/repository/user/getDomainUsers

/shared/ASAssets/Utilities/repository/user/getGroup

/shared/ASAssets/Utilities/repository/user/getUser

/shared/ASAssets/Utilities/repository/user/updateUser

/shared/ASAssets/Utilities/time/getCurrentTimestamp

### 2020 Q200

/shared/ASAssets/Utilities/documentation/modules/getDocResourceProjection

/shared/ASAssets/Utilities/repository/copyResource

/shared/ASAssets/Utilities/repository/getBasicResourceCursor

/shared/ASAssets/Utilities/repository/getBasicResourceCursor\_PROCEDURE\_CURSOR

/shared/ASAssets/Utilities/repository/getBasicResourceCursor\_SQL\_TABLE

/shared/ASAssets/Utilities/repository/getDataSourceCacheConfig

/shared/ASAssets/Utilities/repository/getTrigger

/shared/ASAssets/Utilities/repository/updateResourceAnnotationsV2

/shared/ASAssets/Utilities/repository/updateResourceEnabled

/shared/ASAssets/Utilities/repository/user/updateUser

/shared/ASAssets/Utilities/string/getConstantV2

/shared/ASAssets/Utilities/string/modifyConstant

### 2020 Q101

/shared/ASAssets/Utilities/repository/server/compareServerAttributes

### 2020 Q100

/generate/helpers/createResourceProcess

/repository/getServerAttribute

/repository/getServerAttributeDefs

/repository/server/getServerAttribute

/repository/server/updateServerAttribute

/repository/lowerLevelProcedures/getServerAttributesXSLT

/repository/getDataSourceRootPath

/string/getConstantV2

/validate/compareResourcesTargetServer

Examples separated out from main code into /shared/ASAssets/Utilities\_Examples

Scripts: deployProject.bat and deployProject.sh

### 2019 Q301

1. Changes made for ASAssets BestPractices.

/string/extractTextList

/string/modifyConstant

1. Additional changes.

/repository/recoverFailedCacheRefresh

/repository/replaceStringInResources

/repository/replaceStringInAnnotations

### 2019 Q200

(1) Removed the use of a vector without impacting the procedure signature as a vector is related to memory leak issues.

(2) Replaced custom functions with explicit paths in order to alleviate issues where customers have copies of old Utilities still in their environment. Note: There should “always, only” be 1 copy of Utilities on a DV server. All other copies and versions must be deleted as custom function use is impacted with multiple copies.

(3) Added capabilities, fixed bugs or made more efficient.

getUtilitiesVersion – modified with the latest version for this release.

/calcualations/medianFromAuery – replaced vector usage.

/documentation/getDocumentationDriver – replaced custom functions with explicit calls.

/documentation/helpers/parseDocSwitches – improved efficiency.

/documentation/modules/getDocDataSourceLineage – replaced vector usage.

/documentation/implementation/getDocResourceFormatImpl1 – replaced vector usage.

/documentation/implementation/getDocResourceFormatImpl1\_resource – replaced custom functions with explicit calls.

/generate/destroyDependentLineage – replaced vector usage.

/generate/destroyUsedLineage – replaced vector usage.

/generate/helpers/createResourceProcess – replaced vector usage.

/repository/copyResourceAnnotations – replaced vector usage.

/repository/createAllFoldersPrivileges – replaced vector usage.

/repository/generateOptFile – Moved and renamed to /deployment/optionsfile/helpers/generateOptions. Modified to allow the generation of all datasource options which was the original and is the default. Allow the option generation of a single resource path or a comma-separated list of paths. This is useful when doing maintenance on the options file and you only need to generate for 1 or more specific paths. The exclusion list allows you prune out datasource options from this list. Both the resourcePathList and the excludePathList may contain partial paths. Do not include % for a wild-card. It is done automatically.

/repository/getResourceCacheConfigCursor – modified the output parameter "clearPeriod" to "clearRule" as it was not returning the data properly.

/repository/getResourceResourceLineageRecursive – Fixed a bug with exclusion list. Changed "= 1" to "> 0" in this line "if (INSTR(resourcePath, object) = 1) then".

/repository/impactedTargetsList – replaced vector usage.

/repository/introspectResources – replaced use of vector parameters for introspection and copy privileges as vector parameters are associated with memory leaks. Removed use of lowerLevelProcedures/introspectResourcesTask and consolidated into this procedure. Removed use of lowerLevelProcedures/introspectResourcesResult and consolidated into this procedure. Added the ability to define a resource name, type and subtype in the tableNames or procedureNames input list using this format: name(TYPE:SUBTYPE).

/repository/rebindFolder – replaced vector usage. Consolidated code and eliminated procedure calls to be more efficient.

/repository/rebindResource – kept the vector parameter signature. Replaced vector usage within the procedure. Consolidated code and eliminated procedure calls to be more efficient.

/repository/searchResources – Changed the implementation to use /services/databases/system/ALL\_RESOURCES instead of getResourceListRecursive.

/repository/rebind/rebindReplaceText – added the ability to update a trigger.

/repository/updateBasicTransformationProcedure – consolidated assignAnnotation code.

/repository/updateExternalSqlProcedure – consolidated assignAnnotation code.

/repository updateStreamTransformProcedure – consolidated assignAnnotation code.

/repository/updateXsltTransformProcedure – consolidated assignAnnotation code.

/repository/definitions/RepositoryDefinitions – expanded Attribute Value to LONGVARCHAR for AttributeType.attrValue.

/repository/lowerLevelProcedures/getResourceCacheConfig621XSLT – modified the output parameter "clearPeriod" to "clearRule" as it was not returning the data properly.

/repository/lowerLevelProcedures/splitResourceType – removed getDelimitedOccurrence and replace with simple INSTR and SUBSTRING to be more efficient.

/repository/server/getServerAttribute – changed custom functions to explicit paths.

/repository/user/createResourcePrivilege – replaced vector usage.

/repository/user/createUser – replaced vector usage.

/string/addQuotesInList – removed the call to indent() and just put the code in-line to be more efficient.

/string/extractBiDelimitedText – fixed bug where server is set with ignore trailing spaces=true and there is a trailing space in the delimiter.

/string/extractDelimitedText – fixed bug where server is set with ignore trailing spaces=true and there is a trailing space in the delimiter.

/string/extractTextList – removed the call to extractDelimitedText and put the code directly in-line to make it more efficient.

/string/modifyConstant – changed custom functions to explicit paths. Added ability to modify DECLARE PUBLIC statements in addition to the existing SET variable\_name = statements..

/string/splitByDelimiter – fixed bug where server is set with ignore trailing spaces=true and there is a trailing space in the delimiter.

/time/getTimestampInterval – modified FROM clause to use /services/databases/system/DUAL.

/time/extractDate – replaced vector usage.

/time/extractTime – replaced vector usage.

/time/extractTimestamp – replaced vector usage.

## Removed Resources

### 2020 Q200

/shared/ASAssets/Utilities/repository/getBasicResourceCursor\_All – use getBasicResourceCursor

Removed all lower level procedures with a 62 in the name as 6.2 is no longer supported. Rebound transformations to use non-62 XSLT.

### 2019 Q200

/repository/lowerLevelProcecures/introspectResourcesResult – consolidated functionality directly into introspectResources in order to reduce overhead.

/repository/lowerLevelProcecures/introspectResourcesTask – consolidated functionality directly into introspectResources in order to reduce overhead.

/repository/rebind/getRebindableResources – consolidated functionality into rebindFolder.

/repository/rebind/rebindCheckPathExists – consolidated functionality into rebindFolder and rebindResource.

/repository/rebind/rebindFolderImpl – consolidated functionality into rebindFolder.

/repository/rebind/rebindResourceImpl– consolidated functionality into rebindResource.

/repository/rebind/assignAnnotation – consolidated functionality into various update procedures.

## Deprecated Resources

### 2020 Q100

/repository/server/getServerAttribute

/repository/server/getServerAttributeList

/repository/server/getServerAttributeMap

/repository/server/getServerAttributeMapByKey

/repository/server/helper/getServerAttributeValueListXSLT

/repository/server/helper/getServerAttributeValueMapXSLT

/repository/server/helper/getServerAttributeValueXSLT

/repository/server/helper/getServerAttributeXML

Scripts: deployPrivs.bat and deployPrivs.sh

### 2019 Q301

/string/getConstant – created getConstantV2 with additional capabilities.

### 2019 Q200

/log/auditLogger – created auditLoggerV2 with no vector parameter to replace auditLogger.

/repository/copyResourcesPrivileges – created copyResourcesPrivilegesV2 with no vector parameter to replace copyResourcesPrivileges.

/repository/updateResourceAnnotations– created updateResourceAnnotationsV2 with no vector parameter to replace updateResourceAnnotations.

/repository/updateResourceCacheConfiguration – created updateResourceCacheConfigurationV2 with no vector parameter to replace updateResourceCacheConfiguration.

/repository/updateResourceDataSource– created updateResourceDataSourceV2 with no vector parameter to replace updateResourceDataSource.

/repository/updateResourcePrivileges – created updateResourcePrivilegesV2 with no vector parameter to replace updateResourcePrivileges.

/repository/updateResourcesSqlTable– created updateResourceSqlTableV2 with no vector parameter to replace updateResourceSqlTable.

/time/DefaultValues– no longer needed with updated versions of extractTime, extractDate and extractTimestamp which use an in-line longvarchar instead of a vector.

/xml/parseAndModifyXML– created parseAndModifyXMLV2 with no vector parameter to replace parseAndModifyXML.

/xml/pruneXML– created pruneXMLV2 with no vector parameter to replace pruneXML.

### 2018 Q100

/time/DateUtils/TZConverter – This function is a duplicate of the DV internal function TZCONVERTOR and therefore is no longer needed. Users should convert over to using the DV internal function. This function will be removed in a future release.

/repository/getBasicResourceCursor\_All – This function is a duplicate of “getBasicResourceCursor” and is outdated as a result of the improvements made to “getBasicResourceCursor”. Use “getBasicResourceCursor” instead of this procedure. This will be removed in a future release.

/repository/renameResource – This function has been implemented in DV itself (/lib/resource/RenameResource). This has been removed.

/repository/resourceExists – This function has been implemented in DV itself (/lib/resource/ResourceExists). This has been removed.

### 2015 Q300

/encoding/EncodingCJP/MD5Hash – This function has been implemented in CIS itself (HASHMD5). This will be removed in a future release.

/encoding/EncodingCJP/SHA1Hash – This function has been implemented in CIS itself (HASHSHA1). This will be removed in a future release.

### 2015 Q200

/repository/renameResource – This function has been implemented in CIS itself (/lib/resource/RenameResource). This will be removed in a future release.

/repository/resourceExists – This function has been implemented in CIS itself (/lib/resource/ResourceExists). This will be removed in a future release.

### 2014 Q400

/documentation/getDocConstants – Replaced with string/getConstants(). This has been removed.

/log/errorNotification – Replaced with log/auditLogger(). This has been removed.

/request/DUAL – This view has been implemented in CIS itself. The view can be found at /services/databases/system/DUAL. This has been removed.

### 2014 Q300

/repository/RepoUtils/EncryptPassword – Publishing the source code for this CJP exposes the CIS internals of how passwords are encrypted. This has been removed.

### 2014 Q100

/repository/RepoUtils62 – This CJP data source’s procedures have been folded into repository/RepoUtils. This data source has been removed.

### 2012 Q400

/repository/addRemoveDataSourceChildren – This function uses the deprecated introspection API (which now appears to be broken in 6.2 SP1.) Please use the new repository/introspectResourcesTask() and repository/introspectResourcesResult() utilities instead. This has been removed.

/repository/getResourceLineageParent – This function is no longer being used by the documentation procedure and has been subsequently replaced by repository/getResourceLineageRecursive. This procedure has been removed.

/repository/lowerLevelProcedures/getResourceLineageRecursive – This function is no longer being used by the documentation procedure and has been subsequently replaced by repository/getResourceLineageRecursive. This procedure has been removed.

/documentation/helpers/findDatabases – This procedure is no longer being used by the documentation procedures and has been removed.

### 2012 Q100

/string/LPAD – This function has been implemented in CIS itself. This has been removed.

/string/RPAD – This function has been implemented in CIS itself. This has been removed.

/xml/CreateXmlString2CursorXForm – The xml/reverseXML function does a MUCH better job and is MUCH easier to use. This has been removed.

### 2011 Q300

/repository/applyReservedListToPath – This has been rewritten as a CJP in repository/RepoUtils. This will be removed in a future release.

/repository/applyReservedListToWord – This has been rewritten as a CJP in repository/RepoUtils. This will be removed in a future release.

/repository/configureReservedList – This will be removed in a future release.

1. Introduction

## Purpose

The purpose of this document is to provide guidance on how to use the consolidated custom “Utilities” library.

This document provides documentation on the following functions:

1. **Active Directory** – functions for working with Active Directory.
2. **Archive** – functions for creating backup and package exports.
3. **Cache** – functions related to caching.
4. **Calculations** –general calculations.
5. **Conversions** – general conversions.
6. **Deployment** – deployment tools
7. **Documentation** – tools for documenting resources.
8. **Encoding** – encoding conversions.
9. **Environment** – environment values.
10. **Examples** – example resources that illustrate the usage of many of the utilities.
11. **File** – full Create, Read, Update, and Delete for files plus many other useful file capabilities.
12. **Generate** – guid or view generation and deletion scripts for data abstraction best practices layers.
13. **Logging** – general-purpose logging and error notification.
14. **Modify** – code modification tools and templates.
15. **Monitor** – general monitoring capabilities.
16. **Net** – tools for accessing the network.
17. **PDTool** – tools for generating PDTool deployment plans.
18. **Repository** – general-purpose repository API interaction functions.
19. **Request** – functions for accessing request source code.
20. **Security** – functions for encrypting and decrypting text.
21. **String** – general-purpose string manipulation functions.
22. **Time** – general-purpose time manipulation functions.
23. **Upgrade** – views and procedures that assist with major CIS upgrades.
24. **Validate** – validation and comparison procedures.
25. **XML** – general-purpose XML manipulation functions.

## History

Over the years, a number of development resources and utility functions have been developed by members of the professional services team, the sales engineering team, and various other technical folks within the former company Composite. In an effort to consolidate these utility procedures and prevent the “reinvention of the wheel”, a small team was formed to collect these highly useful and timesaving procedures into a single distribution and make them open source.

The utilities presented in this distribution are not full implementations or solutions to a particular problem. They are simple tools for accomplishing administrative/development tasks or tasks that are slightly outside of the designed use of Data Virtualization (DV) and not likely to ever be rolled into the DV product itself.

## Audience

This document is intended to provide guidance for the following users:

* **Developers**
* **Administrators**

## Installation Notes

### New Folder Structure

Many Advanced Services (AS) assets are being consolidated under a single folder, /shared/ASAssets. From this point forward, the Utilities will be distributed in a CAR file that expects this structure.

To facilitate the management of the Utilities and other Advanced Services (AS) assets moving forward, the following are some guidelines for fresh and existing installations:

For **ALL** installation scenarios presented below

* Log in as “admin”
* Continue to the scenario that fits your environment

For **NEW** installs of the Utilities where no **ASAssets** have been installed yet:

* Create a new published datasource ASAssets if it does not exist.
* Right click on Desktop (admin) and select “Import …”, choose the Utilities distribution CAR file [Utilities\_YYYYQnnn.car] in the resulting dialog, and click the “Import>” button.

For **EXISTING** installs of the Utilities where an **ASAssets** folder already exists:

* **IMPORTANT**: Before installing, make a copy of /shared/ASAssets/Utilities/environment/getEnvName or note the value of envName within the procedure. Re-install the copy or the original value when completed with the upgrade.
* Create a new published datasource ASAssets if it does not exist.
* Right click on Desktop (admin) and select “Import …”, choose the Utilities distribution CAR file [Utilities\_YYYYQnnn.car] in the resulting dialog, and click the “Import>” button.
* Using the “Overwrite” option will only overwrite the Utilities folder and leave everything else in “ASAssets” intact.
* Restore the envName to getEnvName.
* Execute the procedure /shared/ASAssets/reintrospectCJPs to reintrospect all the CJP data sources in the Utilities distribution.
  + If an exception occurs during execution of reintrospectCJPs, then the Data Virtualization server will have to be restarted so the CJP’s are read into DV memory.

For **EXISTING** installs of the Utilities where the former **PSAssets** folder already exists:

* Rename PSAssets to ASAssets
* Create a new published datasource ASAssets if it does not exist.
* Right click on Desktop (admin) and select “Import …”, choose the Utilities distribution CAR file [Utilities\_YYYYQnnn.car] in the resulting dialog, and click the “Import>” button.
* Using the “Overwrite” option will only overwrite the Utilities folder and leave everything else in “ASAssets” intact.
* Execute the procedure /shared/ASAssets/reintrospectCJPs to reintrospect all the CJP data sources in the Utilities distribution.
  + If an exception occurs during execution of reintrospectCJPs, then the Data Virtualization server will have to be restarted so the CJP’s are read into DV memory.

For **EXISTING** installs of the Utilities where the former **/shared/Utilities** folder already exists:

* Create a new published datasource ASAssets if it does not exist.
* Create a folder in /shared called “ASAssets”. Spelling and capitalization are important here so please use this exact spelling and case.
* Edit the following procedures by changing the PATH statement (after the initial BEGIN keyword) from “/shared/Utilities” to “/shared/ASAssets/Utilities”. This will prevent issues with impacted resources after the cut and paste operation below.
  + /shared/ASAssets/Utilities/repository/definitions/RepositoryDefinitions
  + /shared/ASAssets/Utilities/repository/definitions/RepositoryDefinitionsRecursive
* Cut the existing Utilities folder from /shared and paste it into ASAssets. Cut and paste will ensure that any resources using the Utilities will be rebound to use the new location (this does not rebind references embedded in character string values, however.) **DO NOT COPY** the Utilities folder into ASAssets as this will not rebind dependent resources.
* Right click on Desktop (admin) and select “Import …”, choose the Utilities distribution CAR file [Utilities\_YYYYQnnn.car] in the resulting dialog, and click the “Import>” button.
* Using the “Overwrite” option will only overwrite the Utilities folder and leave everything else in “ASAssets” intact.
* Execute the procedure /shared/ASAssets/reintrospectCJPs to reintrospect all the CJP data sources in the Utilities distribution.
  + If an exception occurs during execution of reintrospectCJPs, then the Data Virtualization server will have to be restarted so the CJP’s are read into DV memory.
* If any dependent procedures that were using the old Utilities version are impacted, they will require being updated as the API signatures could have changed.
  + It is also possible that some Utilities may have been deprecated. Consult the “Deprecated Resources” section in this document.

### Reserved Word List

The repository/RepoUtils/applyReservedWordTo\* procedures use a flat properties file ($CIS\_HOME/conf/customjars/RepoUtils.properties) to get the reserved word list. If the properties file is missing (usually when the Utilities are installed for the first time) it gets generated the first time one of these procedures is used. **It is never replaced** with an updated version under the assumption that a user might manually edit it to include new reserved words after an upgrade of CIS (without requiring an upgrade of the Utilities.)

### Recursive Procedure Use

The AS Assets Utilities use a number of procedures that are recursive in nature (repeatedly call themselves until a condition is reached.) With that in mind, it's recommended that the "Maximum Request Depth" setting in CIS be updated from its default setting of 30 to 100. This setting can be found in the Studio's Configuration panel in "Server" > "Configuration" > "Transactions" > "Maximum Request Depth".

### getEnvName Usage

The AS Assets Utilities procedure /shared/ASAssets/Utilities/environment/getEnvName may contain a customized value for a given environment. It is a strategy for naming the environment and allowing code to make decisions based on the value. The Tibco Open Source KPImetrics may use this procedure so take care to replace the original value when upgrading the Utilities.

1. Top Level Utilities Procedures

## Introduction

This section describes the procedures found directly under /shared/ASAssets/Utilities.

### ExceptionDefinitions

Contains commonly used custom exceptions used throughout the /shared/ASAssets/Utilities folder.

### getUtilitiesVersion (Custom Function)

Returns the current version of /shared/ASAssets/Utilities installed on the system. Can be used to enforce the minimum version of /shared/ASAssets/Utilities required for a script to function properly:

IF (getUtilitiesVersion() < 2017.4) THEN  
 RAISE System.NotSupportedException   
 VALUE '/shared/ASAssets/Utilities must be version 2017.4';  
END IF;

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | DOUBLE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | 2016.1 |

### reintrospectCJPs

This procedure walks the folder tree of the Utilities distribution and performs a reintrospection on any CJP data source that it finds. This is usually only needs to be executed once if/when a Utilities distribution is relocated within CIS. It expects that the Utilities are installed in /shared/ASAssets/Utilities. If not installed there, update the UTILITIES\_HOME constant at the beginning of the procedure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | success | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | success | 1 |

### TypeDefinitions

Contains commonly used custom data types used throughout the /shared/ASAssets/Utilities folder.

### Custom Function Disclaimer

This is disclaimer regarding the use of custom functions.

**IMPORTANT: Custom functions should not be used as a general practice. The full path to the procedure should be used instead within the custom code. In many cases these resources are not managed properly and multiple copies infiltrate the environment causing problems. When resources are exported with “dependency” box checked and those resource reference Utilities, they bring in a copy of those Utilities and the custom function is marked to use the “rogue” copy instead of the true original copy. Therefore, it is not a good idea to use custom functions in your code. Use the full path to the true original ASAssets utility: /shared/ASAssets/Utilities/<folder>/<resourceName>**

### Custom Function List

This is a list of all the custom functions should be checked.

|  |  |
| --- | --- |
| name | functionPath |
| ActiveDirectoryInt8ToDate | /shared/ASAssets/Utilities/activedirectory/ActiveDirectoryInt8ToDate |
| ActiveDirectoryTSToSQLTimeStamp | /shared/ASAssets/Utilities/activedirectory/ActiveDirectoryTSToSQLTimeStamp |
| SimpleBinaryAND | /shared/ASAssets/Utilities/activedirectory/SimpleBinaryAND |
| calculateAge | /shared/ASAssets/Utilities/calculations/calculateAge |
| medianFromQuery | /shared/ASAssets/Utilities/calculations/medianFromQuery |
| convertBit | /shared/ASAssets/Utilities/conversions/convertBit |
| convertBoolean | /shared/ASAssets/Utilities/conversions/convertBoolean |
| convertDoubleToInteger | /shared/ASAssets/Utilities/conversions/convertDoubleToInteger |
| convertTemperatureUnit | /shared/ASAssets/Utilities/conversions/convertTemperatureUnit |
| convertYN | /shared/ASAssets/Utilities/conversions/convertYN |
| parseDocSwitches | /shared/ASAssets/Utilities/documentation/helpers/parseDocSwitches |
| Base64Decode | /shared/ASAssets/Utilities/encoding/EncodingCJP/Base64Decode |
| Base64Encode | /shared/ASAssets/Utilities/encoding/EncodingCJP/Base64Encode |
| MD5Hash | /shared/ASAssets/Utilities/encoding/EncodingCJP/MD5Hash |
| SHA1Hash | /shared/ASAssets/Utilities/encoding/EncodingCJP/SHA1Hash |
| getEnvName | /shared/ASAssets/Utilities/environment/getEnvName |
| existsDir | /shared/ASAssets/Utilities/file/FileProcessingCJP/existsDir |
| existsFile | /shared/ASAssets/Utilities/file/FileProcessingCJP/existsFile |
| getFileContentsAscii | /shared/ASAssets/Utilities/file/FileProcessingCJP/getFileContentsAscii |
| getFileContentsBinary | /shared/ASAssets/Utilities/file/FileProcessingCJP/getFileContentsBinary |
| gunzipFile | /shared/ASAssets/Utilities/file/FileProcessingCJP/gunzipFile |
| makeDirs | /shared/ASAssets/Utilities/file/FileProcessingCJP/makeDirs |
| remove | /shared/ASAssets/Utilities/file/FileProcessingCJP/remove |
| removeAll | /shared/ASAssets/Utilities/file/FileProcessingCJP/removeAll |
| unzipFile | /shared/ASAssets/Utilities/file/FileProcessingCJP/unzipFile |
| getCisHome | /shared/ASAssets/Utilities/file/getCisHome |
| getFileSeparator | /shared/ASAssets/Utilities/file/getFileSeparator |
| getUtilitiesVersion | /shared/ASAssets/Utilities/getUtilitiesVersion |
| isReservedWord | /shared/ASAssets/Utilities/repository/RepoUtils/isReservedWord |
| applyReservedListToPath | /shared/ASAssets/Utilities/repository/applyReservedListToPath |
| applyReservedListToWord | /shared/ASAssets/Utilities/repository/applyReservedListToWord |
| compareCisVersions | /shared/ASAssets/Utilities/repository/compareCisVersions |
| fixLeadingCharactersInFolderPath | /shared/ASAssets/Utilities/repository/fixLeadingCharactersInFolderPath |
| getCisVersion | /shared/ASAssets/Utilities/repository/getCisVersion |
| getScriptText | /shared/ASAssets/Utilities/repository/getScriptText |
| getBasicResourceXML | /shared/ASAssets/Utilities/repository/lowerLevelProcedures/getBasicResourceXML |
| getChildResourcesXML | /shared/ASAssets/Utilities/repository/lowerLevelProcedures/getChildResourcesXML |
| getUsedResourcesXML | /shared/ASAssets/Utilities/repository/lowerLevelProcedures/getUsedResourcesXML |
| getServerAttribute | /shared/ASAssets/Utilities/repository/server/getServerAttribute |
| getServerAttributeMapByKey | /shared/ASAssets/Utilities/repository/server/getServerAttributeMapByKey |
| DirectSqlRequest | /shared/ASAssets/Utilities/request/RequestUtils/DirectSqlRequest |
| OriginalRequest | /shared/ASAssets/Utilities/request/RequestUtils/OriginalRequest |
| TopSqlRequest | /shared/ASAssets/Utilities/request/RequestUtils/TopSqlRequest |
| Blob2Varchar | /shared/ASAssets/Utilities/string/TextUtils/Blob2Varchar |
| CCNumberFormatter | /shared/ASAssets/Utilities/string/TextUtils/CCNumberFormatter |
| CSVFromCISQuery | /shared/ASAssets/Utilities/string/TextUtils/CSVFromCISQuery |
| FixedFromCISQuery | /shared/ASAssets/Utilities/string/TextUtils/FixedFromCISQuery |
| FormatXML | /shared/ASAssets/Utilities/string/TextUtils/FormatXML |
| GenerateGuid | /shared/ASAssets/Utilities/string/TextUtils/GenerateGuid |
| HexToRaw | /shared/ASAssets/Utilities/string/TextUtils/HexToRaw |
| LocalCurrencyFormatter | /shared/ASAssets/Utilities/string/TextUtils/LocalCurrencyFormatter |
| LocalCurrencyParser | /shared/ASAssets/Utilities/string/TextUtils/LocalCurrencyParser |
| LocalDateFormatter | /shared/ASAssets/Utilities/string/TextUtils/LocalDateFormatter |
| LocalDateParser | /shared/ASAssets/Utilities/string/TextUtils/LocalDateParser |
| LocalNumberFormatter | /shared/ASAssets/Utilities/string/TextUtils/LocalNumberFormatter |
| LocalNumberParser | /shared/ASAssets/Utilities/string/TextUtils/LocalNumberParser |
| LocalTimeFormatter | /shared/ASAssets/Utilities/string/TextUtils/LocalTimeFormatter |
| LocalTimeParser | /shared/ASAssets/Utilities/string/TextUtils/LocalTimeParser |
| LocalTimestampFormatter | /shared/ASAssets/Utilities/string/TextUtils/LocalTimestampFormatter |
| LocalTimestampParser | /shared/ASAssets/Utilities/string/TextUtils/LocalTimestampParser |
| PhoneNumberFormatter | /shared/ASAssets/Utilities/string/TextUtils/PhoneNumberFormatter |
| RawToHex | /shared/ASAssets/Utilities/string/TextUtils/RawToHex |
| RegexCount | /shared/ASAssets/Utilities/string/TextUtils/RegexCount |
| RegexFind | /shared/ASAssets/Utilities/string/TextUtils/RegexFind |
| RegexPosition | /shared/ASAssets/Utilities/string/TextUtils/RegexPosition |
| RegexReplace | /shared/ASAssets/Utilities/string/TextUtils/RegexReplace |
| SSNumberFormatter | /shared/ASAssets/Utilities/string/TextUtils/SSNumberFormatter |
| URLDecode | /shared/ASAssets/Utilities/string/TextUtils/URLDecode |
| URLEncode | /shared/ASAssets/Utilities/string/TextUtils/URLEncode |
| addQuotesInList | /shared/ASAssets/Utilities/string/addQuotesInList |
| basename | /shared/ASAssets/Utilities/string/basename |
| concatNotNull | /shared/ASAssets/Utilities/string/concatNotNull |
| dirname | /shared/ASAssets/Utilities/string/dirname |
| emptyStr | /shared/ASAssets/Utilities/string/emptyStr |
| entityExtractToString | /shared/ASAssets/Utilities/string/entityExtractToString |
| escapeCSV | /shared/ASAssets/Utilities/string/escapeCSV |
| extractBiDelimitedText | /shared/ASAssets/Utilities/string/extractBiDelimitedText |
| findString | /shared/ASAssets/Utilities/string/findString |
| findStringInList | /shared/ASAssets/Utilities/string/findStringInList |
| fixQuotes | /shared/ASAssets/Utilities/string/fixQuotes |
| getConstant | /shared/ASAssets/Utilities/string/getConstant |
| getConstantV2 | /shared/ASAssets/Utilities/string/getConstantV2 |
| getDelimitedOccurrence | /shared/ASAssets/Utilities/string/getDelimitedOccurrence |
| getDelimitedSum | /shared/ASAssets/Utilities/string/getDelimitedSum |
| indent | /shared/ASAssets/Utilities/string/indent |
| isEmpty | /shared/ASAssets/Utilities/string/isEmpty |
| last4ofSSN | /shared/ASAssets/Utilities/string/last4ofSSN |
| removeDoubleQuotes | /shared/ASAssets/Utilities/string/removeDoubleQuotes |
| removeSingleQuotes | /shared/ASAssets/Utilities/string/removeSingleQuotes |
| BigintToTimestamp | /shared/ASAssets/Utilities/time/DateUtils/BigintToTimestamp |
| DateAddDate | /shared/ASAssets/Utilities/time/DateUtils/DateAddDate |
| DateAddTimestamp | /shared/ASAssets/Utilities/time/DateUtils/DateAddTimestamp |
| DateDiffDate | /shared/ASAssets/Utilities/time/DateUtils/DateDiffDate |
| DateDiffTimestamp | /shared/ASAssets/Utilities/time/DateUtils/DateDiffTimestamp |
| GetServerTimezone | /shared/ASAssets/Utilities/time/DateUtils/GetServerTimezone |
| TZConverter | /shared/ASAssets/Utilities/time/DateUtils/TZConverter |
| TimestampToBigint | /shared/ASAssets/Utilities/time/DateUtils/TimestampToBigint |
| extractDate | /shared/ASAssets/Utilities/time/extractDate |
| extractTime | /shared/ASAssets/Utilities/time/extractTime |
| extractTimestamp | /shared/ASAssets/Utilities/time/extractTimestamp |
| getCurrentTimestamp | /shared/ASAssets/Utilities/time/getCurrentTimestamp |
| getTimestampInterval | /shared/ASAssets/Utilities/time/getTimestampInterval |
| intervalDay2Seconds | /shared/ASAssets/Utilities/time/intervalDay2Seconds |
| period2IntervalDay | /shared/ASAssets/Utilities/time/period2IntervalDay |
| returnColumnOrderingString | /shared/ASAssets/Utilities/upgrade/helpers/returnColumnOrderingString |
| castXMLTextNodeAsVarchar | /shared/ASAssets/Utilities/xml/castXMLTextNodeAsVarchar |
| escapeXML | /shared/ASAssets/Utilities/xml/escapeXML |
| getNodeFromXML | /shared/ASAssets/Utilities/xml/getNodeFromXML |
| getValueFromXML | /shared/ASAssets/Utilities/xml/getValueFromXML |
| reverseXML | /shared/ASAssets/Utilities/xml/reverseXML |
| stripInvalidXMLChars | /shared/ASAssets/Utilities/xml/stripInvalidXMLChars |
| unescapeXML | /shared/ASAssets/Utilities/xml/unescapeXML |

1. How To Use ‘Active Directory’ Procedures

## Introduction

This section will show how to use the Active Directory procedures.

### ActiveDirectoryInt8ToDate (Custom Function)

Active Directory represents some Date values internally as an Integer8, which translates to a BIGINT. The number represents the number of 100-nanosecond intervals since 12:00 AM January 1, 1601. In addition, the fact that the CIS function UTC\_TO\_TIMESTAMP() uses 1/1/1970 as its base date must also be accounted for in the conversion.

Please note, Active Directory stores dates using Greenwich Mean Time (GMT), or a GMT offset of 0. The function supports converting the date value according to the number of hours that the local time zone is offset from GMT.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | ADInt8 | BIGINT |
| IN | GMTOffsetInHours | INT |
| OUT | ADDate | DATE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | ADInt8 | 129296016000000000 |
| IN | GMTOffsetInHours | 6 |
| OUT | ADDate | ‘2010-09-21’ |

* 1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | ADInt8 | 129296016000000000 |
| IN | GMTOffsetInHours | 0 |
| OUT | ADDate | ‘2010-09-22’ |

### ActiveDirectoryTSToSQLTimeStamp (Custom Function)

Accept an Active Directory-formatted string as input and returns a Composite Timestamp. Supports a GMT offset in the same manner as activedirectory/ActiveDirectoryInt8ToDate (see previous function.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | TimeStampString | STRING |
| IN | GMTOffsetInHours | INT |
| OUT | TimeStampOut | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | TimeStampString | ‘20100922123423’ |
| IN | GMTOffsetInHours | 0 |
| OUT | TimeStampOut | ‘2010-09-22 12:34:23’ |

### SimpleBinaryAND (Custom Function)

Accepts a BIGINT and a power of 2 (e.g. 2, 4, 8, etc.) and ANDs the two numbers to indicate whether the bit in the BIGINT at the position of the power of 2 is 1 or 0. It is a simple function originally created for determining whether or not an Active Directory user is enabled or disabled, which is stored in a bit mask in Active Directory. The function works for numbers 2 ^ 31 and smaller.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Num | BIGINT |
| IN | PowerOfTwo | BIGINT |
| OUT | Result | BIGINT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Num | 3567 |
| IN | PowerOfTwo | 16 |
| OUT | Result | 0 |

1. How To Use ‘Archive’ Procedures

## Introduction

This section will show how to use the ‘Archive’ procedures.

### backup\_export

This procedure performs a full server backup of the local CIS instance. It cannot be used to back up remote CIS instances.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | archiveFilePath | VARCHAR(4096) |
| IN | exportOptions | LONGVARCHAR |
| OUT | success | BIT |
| OUT | responseXML | XML |
| OUT | faultXML | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | archiveFilePath | ‘C:\my\_fsb.car’ |
| IN | exportOptions | ENCRYPTION\_PASSWORD=password |
| OUT | success | 1 |
| OUT | responseXML | <xml …> |
| OUT | faultXML | NULL |

### exportResources

This procedure selects a list of resources, formats them into XML and then performs a package export of a list of resources for the TDV instance and saves to a specified path on the filesystem of the local TDV host. It is the equivalent of running the pkg\_export script in <TDV\_HOME>/bin against the local server.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | *debug* - Y=debug on. N=debug off. | CHAR(1) |
| IN | *optionType* - the option type by which the resource list will be derived.  -- 0: Export a single resource, folder, or double-piped resource list with or without children as per the input variable "includeChildren"  -- 1: Find the n most recent modified resources in the path (default of n is 1). results are not ordered  -- 2: Find the n least recent modified resources in the path (default of n is 1). results are not ordered  -- 3: Find all resources modified after the passed in resource date  -- 4: Find all resources modifed before the passed in resource date  -- 5: Find all resources created after the passed in resource date  -- 6: Find all resources created before the passed in resource date  -- 7: [Lineage + modification date] Find the lineage of the passed in resource path where the modification date is after the passed in resource date and exclude paths if applicable. | INTEGER |
| IN | *resourcePathList* - May be an explicit single resource, a container path, a data source path, a published path or a double pipe-separated list of resources to export.  Example 1: single resource: /shared/test00/CisResources/my\_view\_1  Example 2: folder resource: /shared/test00/CisResources  Example 3: data source: /shared/test00/DataSources/ds\_orders  Example 4: published web service: /services/webservices/CustomerWS  Example 5: published table: /services/databases/MYTEST/CAT1/SCH2/ViewSales  Example 6: Multiple resources [path||path]: /services/databases/MYTEST/CAT1/SCH2||/shared/test00/DataSources||/shared/test00/CisResources | LONGVARCHAR |
| IN | *resourceNum* - The number of resources to return in the list for optionType 1 or 2. The default is 1 | INTEGER |
| IN | *resourceDate* - A timestamp to compare with the various resources for optionType 3-7. | TIMESTAMP |
| IN | *excludePaths* - A comma-separated list of paths to exclude from the output. The INSTR function is used so that the paths can be partial paths. Used by option 7.  Values: e.g. /shared/lab0/Physical/Metadata/ -- Exclude all physical metadata resources | LONGVARCHAR |
| IN | *includeChildren* - true(default) or false. If the resource path is a CONTAINER or DATA\_SOURCE then includeChildren=true will bring all children of that resource.  All subfolders of a parent container are also exported. If the resource path contains multiple double-piped paths, the includeChildren will be applied as provided to each path. | VARCHAR |
| IN | *archiveFilePath* - The path to the location on the TDV host filesystem to store the backup archive. | VARCHAR(4096) |
| IN | *exportOptions* - a pipe separated list of export options  INCLUDE\_CACHING: Include caching configurations for resources.  INCLUDE\_CUSTOM\_JAVA\_JARS: Include custom Java JARs in the export. (ADMIN ONLY)  INCLUDE\_STATISTICS: Include any resources stats known about objects including the table boundaries, and column boundaries.  INCLUDE\_DEPENDENCY: Gather and include all dependent resources for the resources you choose to export.  INCLUDE\_PHYSICAL\_SOURCE\_INFO: Include sensitive connection information for included physical sources. (OWNER ONLY)  INCLUDE\_REQUIRED\_USERS: Include the information about the required users in the export file.  INCLUDE\_SECURITY: Include resource privilege settings. (OWNER ONLY)  ENCRYPTION\_PASSWORD=password: TDV 8.x and up may use an encryption password. | LONGVARCHAR |
| OUT | *success* - Indicates if the backup export operation succeeded or not. Values: 1 = success, 0 = failure | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | N |
| IN | optionType | 7 |
| IN | resourcePathList | /services/databases/MY DB/MY CATALOG/MY SCHEMA/CompositeView |
| IN | resourceNum | null |
| IN | resourceDate | 2020-05-01 00:00:00 |
| IN | excludePaths | /shared/examples/ds\_orders |
| IN | includeChildren | true |
| IN | archiveFilePath | c:\temp\my\_export.car |
| IN | exportOptions | INCLUDE\_CACHING| INCLUDE\_PHYSICAL\_SOURCE\_INFO| INCLUDE\_REQUIRED\_USERS |
| OUT | success | 1 |

### exportArchiveFile

This procedure gets invoked by “exportResources”. This procedure performs a package export of a list of resources for the TDV instance and saves to a specified path on the filesystem of the local TDV host. It is the equivalent of running the pkg\_export script in <TDV\_HOME>/bin against the local server.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | *debug* - Y=debug on. N=debug off. | CHAR(1) |
| IN | *archiveFilePath* - The path to the location on the TDV host filesystem to store the backup archive. | VARCHAR(4096) |
| IN | *inputXML* - XML formatted list of resources to export. | LONGVARCHAR |
| IN | *exportOptions* - a pipe separated list of export options  INCLUDE\_CACHING: Include caching configurations for resources.  INCLUDE\_CUSTOM\_JAVA\_JARS: Include custom Java JARs in the export. (ADMIN ONLY)  INCLUDE\_STATISTICS: Include any resources stats known about objects including the table boundaries, and column boundaries.  INCLUDE\_DEPENDENCY: Gather and include all dependent resources for the resources you choose to export.  INCLUDE\_PHYSICAL\_SOURCE\_INFO: Include sensitive connection information for included physical sources. (OWNER ONLY)  INCLUDE\_REQUIRED\_USERS: Include the information about the required users in the export file.  INCLUDE\_SECURITY: Include resource privilege settings. (OWNER ONLY)  ENCRYPTION\_PASSWORD=password: TDV 8.x and up may use an encryption password. | LONGVARCHAR |
| OUT | *success* - Indicates if the backup export operation succeeded or not. Values: 1 = success, 0 = failure | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Debug | N |
| IN | archiveFilePath | 'C:\temp\my\_export.car' |
| IN | inputXML | <archive:resource>  <resource:path>/shared/test00/Business/Logical</resource:path>  <resource:type>CONTAINER</resource:type>  <archive:includeChildren>true</archive:includeChildren>  </archive:resource> |
| IN | exportOptions | INCLUDE\_CACHING| INCLUDE\_PHYSICAL\_SOURCE\_INFO| INCLUDE\_REQUIRED\_USERS |
| OUT | success | 1 |

### getExportResourceList

This procedure gets invoked by “exportResources”. Construct the list of resources to be exported into a well-formatted XML ready for export. This procedure is used in conjunction with exportArchiveFile.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | *debug* - Y=debug on. N=debug off. | CHAR(1) |
| IN | *optionType* - the option type by which the resource list will be derived.  -- 0: Export a single resource, folder, or double-piped resource list with or without children as per the input variable "includeChildren"  -- 1: Find the n most recent modified resources in the path (default of n is 1). results are not ordered  -- 2: Find the n least recent modified resources in the path (default of n is 1). results are not ordered  -- 3: Find all resources modified after the passed in resource date  -- 4: Find all resources modifed before the passed in resource date  -- 5: Find all resources created after the passed in resource date  -- 6: Find all resources created before the passed in resource date  -- 7: [Lineage + modification date] Find the lineage of the passed in resource path where the modification date is after the passed in resource date and exclude paths if applicable. | INTEGER |
| IN | *resourcePathList* - May be an explicit single resource, a container path, a data source path, a published path or a double pipe-separated list of resources to export.  Example 1: single resource: /shared/test00/CisResources/my\_view\_1  Example 2: folder resource: /shared/test00/CisResources  Example 3: data source: /shared/test00/DataSources/ds\_orders  Example 4: published web service: /services/webservices/CustomerWS  Example 5: published table: /services/databases/MYTEST/CAT1/SCH2/ViewSales  Example 6: Multiple resources [path||path]: /services/databases/MYTEST/CAT1/SCH2||/shared/test00/DataSources||/shared/test00/CisResources | LONGVARCHAR |
| IN | *resourceNum* - The number of resources to return in the list for optionType 1 or 2. The default is 1 | INTEGER |
| IN | *resourceDate* - A timestamp to compare with the various resources for optionType 3-7. | TIMESTAMP |
| IN | *excludePaths* - A comma-separated list of paths to exclude from the output. The INSTR function is used so that the paths can be partial paths. Used by option 7.  Values: e.g. /shared/lab0/Physical/Metadata/ -- Exclude all physical metadata resources | LONGVARCHAR |
| IN | *includeChildren* - true(default) or false. If the resource path is a CONTAINER or DATA\_SOURCE then includeChildren=true will bring all children of that resource.  All subfolders of a parent container are also exported. If the resource path contains multiple double-piped paths, the includeChildren will be applied as provided to each path. | VARCHAR |
| IN | *archiveFilePath* - The path to the location on the TDV host filesystem to store the backup archive. | VARCHAR(4096) |
| IN | *exportOptions* - a pipe separated list of export options  INCLUDE\_CACHING: Include caching configurations for resources.  INCLUDE\_CUSTOM\_JAVA\_JARS: Include custom Java JARs in the export. (ADMIN ONLY)  INCLUDE\_STATISTICS: Include any resources stats known about objects including the table boundaries, and column boundaries.  INCLUDE\_DEPENDENCY: Gather and include all dependent resources for the resources you choose to export.  INCLUDE\_PHYSICAL\_SOURCE\_INFO: Include sensitive connection information for included physical sources. (OWNER ONLY)  INCLUDE\_REQUIRED\_USERS: Include the information about the required users in the export file.  INCLUDE\_SECURITY: Include resource privilege settings. (OWNER ONLY) | LONGVARCHAR |
| OUT | *resourceListXML* - An XML formatted list of resources ready for archive export. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | N |
| IN | optionType | 7 |
| IN | resourcePathList | /services/databases/MY DB/MY CATALOG/MY SCHEMA/CompositeView |
| IN | resourceNum | null |
| IN | resourceDate | 2020-05-01 00:00:00 |
| IN | excludePaths | /shared/examples/ds\_orders |
| IN | includeChildren | true |
| OUT | resourceListXML | <archive:resource>  <resource:path>/shared/test00/Business/Logical</resource:path>  <resource:type>CONTAINER</resource:type>  <archive:includeChildren>true</archive:includeChildren>  </archive:resource> |

### getPkgExportList

This procedure is used to generate a simple resource export list for the pkg\_exort option file. The output allows you to copy and paste into an option file whereby each resource is on its own line. This procedure will automatically exclude TDV system paths including the following:

/services/databases/system

/services/databases/transient

/services/webservices/system.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | *resourcePathList* - pipe-separated list of paths to get the children list from.  /services/databases|/services/webservices|/shared | LONGVARCHAR |
| IN | *excludeResourcePathList* - pipe-separated list of paths to exclude.  e.g. /shared/examples | LONGVARCHAR |
| OUT | *generatedPathList* - Each resource path is generated on its own line which can be used for pkg\_export option file. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePathList | /services/databases|/services/webservices|/shared |
| IN | excludeResourcePathList | /shared/examples|/services/databases/examples |
| OUT | generatedPathList | /shared/ASAssets  /shared/BestPracticesTestSmall  /shared/Cache\_DB  /services/databases/ASAssets  /services/databases/CoE  /services/databases/lab00  /services/webservices/CoE  /services/webservices/CustomerWS  /services/webservices/DynamicFile |

### getPkgExportListByLabel

This procedure is used to generate a simple resource export list for the pkg\_exort option file based on searching resource annotations for a particular release label string.

This will generate the introspect list, reintrospection list, design-by-example, remove, rebind and relocate lists. This is meant to auto-generate the necessary files to be used with exporting and deploying a car file using the CoE deployProject.sh|bat command-line script.

* generatedExportList - write out to the export option file /master\_files/CoE/Files/option\_files/export\_option\_file\_ENV.txt
* generatedIntrospectList - write out to /master\_files/CoE/Files/metadata/introspect\_list.txt
* generatedReintrospectList - write out to /master\_files/CoE/Files/metadata/reintrospect\_list.txt
* generatedDesignList - write out to /master\_files/CoE/Files/metadata/design\_list.txt
* generatedRemoveList - write out to /master\_files/CoE/Files/metadata/remove\_list.txt
* generatedRebindList - append to the deployment /master\_files/CoE/Files/option\_files/option\_file\_ENV.txt
* generatedRelocateList - append to the deployment /master\_files/CoE/Files/option\_files/option\_file\_ENV.txt

This procedure will automatically exclude TDV system paths including the following:

/services/databases/system

/services/databases/transient

/services/webservices/system

Rules:

1. The annotation requires a uniform strategy in which a unique label can be determined using a simple search method.

2. The "Label" is unique and is the key search term. If that search term is present then the resource is chosen for the export list.

3. The "Release Date" is simply provides additional information to developers regarding the date of the release but is not used for searching.

4. The "command Extension" provides additional insight into what other deployment metadata files should be generated based on the type of command and the context of the resource.

5. No spaces are allowed in the label or the release date.

6. Each label should be a row which is delimited by a newline also known as CHR(10)=linefeed.

7. The command=command text may be a pipe-separated list up to 4 commands.

Scenario 1: 2 chained commands are known about which is introspect|design=varchar(255):abc

Scenario 2: Scenario 2: 2 chained commands are known about which is reintrospect|relocate=DVKPI7/dbo DVKPI8/DVKPI, DVKPI7 DVKPI8

8. Commands are applicable to certain resource types and only should be used in their annotations.

introspect – only applicable with data source children like database tables and procedures

* Introspect will add new resources into TDV if they don't exist and update existing resources with changes.
* Introspect ignores a resource if there are no changes.
* The format of the command is simply introspect. example: introspect

reintrospect – only applicable with data source resources

* The format of the command is simply reintrospect. example: reintrospect

design – only applicable with data source procedure children requiring design-by-example.

* Specify a comma-separated list of parameters for the procedure.
* Enclose "type:value" within double quotes if the value contains a comma. example: "varchar(50):a, list, of, words"
* The format of the command is design= "type:value", "type:value", "type:value"
  + example: design=VARCHAR(255):abc,"varchar(100):a,list,of,words"

rebind – applicable to views that need to be rebound to a different data source catalog or schema upon import.

* The format of the command is rebind=from-tdv-path to-tdv-path
* User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"
* example: rebind=/shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI7/dbo /shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI8/DVKPI

relocate – applicable to data sources where the catalog and/or schema needs to be relocated to a different name upon import.

* The format of the command is relocate=catalog1/schema1 catalog2/schema2, catalog1 catalog2
* Always specify the relative schema path first and then the relative catalog using comma-separated lists.
  + Leading / is optional for each relative path
  + The context is the data source where the annotation exists.
* User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"
* example: relocate=DVKPI7/dbo DVKPI8/DVKPI, DVKPI7 DVKPI8

remove - this command can only be specified in the label of a container or datasource. A container may be folder, catalog or schema.

* The format of the remove command is remove=resource:type. The type is either the internal type or the type from the Studio info tab (display type).
* To remove multiple resources within the same container, use the format remove=resource:type,resource:type,resource:type.
  + The resource name is case sensitive. The type is not case sensitive.
  + User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"
* If removing a folder then the label of the parent folder must specify the remove.
* If removing a database table or procedure then the remove command must go into the direct parent folder which is typically the schema container but it could be the data source if it has not schemas.
* If removing a database schema then the remove command goes one level up into the catalog container if it exists or the datasource.

export - only applicable to containers and datasources. A container may be folder, catalog or schema.

* The export command will be ignored on any other type of resource.
* Normally, export is assumed if a matching label is found on search and no other commands are specified.
* However, the export command should be specified in the label of a folder/container/datasource if there are other commands present in order to distinguish what the user intends to accomplish.
* When the intention is to execute multiple commands such as export|remove=resource:type or export|reintrospect for data sources then the export is registered along with the other commands.
* When the intention is not to export but a label is present along with other commands such as remove=resource:type then only that command is registered and not the export.

The following is an example only:

Label Release Date command extension

V0000001 2021-11-07 command|command=optional command text

V0000002 2021-11-08 introspect

V0000003 2021-11-09 reintrospect

V0000004 2021-11-10 design=VARCHAR(255):abc,"varchar(100):a,list,of,words"

V0000005 2021-11-11 relocate=dvops/dbo dvops/db1,dvops dvops2

V0000006 2021-11-12 rebind=/shared/TEST/DS1/dvops/dbo /shared/TEST1/DS1/dvops2/db1

V0000007 2021-11-13 remove=proc1:procedure,view1:view

V0000008 2021-11-14 export

V0000009 2021-11-15

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | *debug* - Y=debug on. N=debug off. | CHAR(1) |
| IN | *convertLF* - Convert message and all "generated\*" variables linefeeds to "[LF]". N/null (default)=Do not convert. Allow CHR(10) linefeed to be returned. Y=Convert CHR(10) to [LF] for bash/windows script processing ease of use. | CHAR(1) |
| IN | *startPath* - The starting path list is a colon-separated list of paths to evaluate the child resource annotations for the release label. Use a wild-card % symbol to select all child paths for the base path. The base path is also included in the evaluation.  Example 1: search labels in myproject: /services/databases/myprojectDB%:/services/webservices/myproject%:/shared/myproject%  Example 2: search labels in repository: /services/databases%:/services/webservices%:/shared%:/policy% | LONGVARCHAR |
| IN | *searchStr* - Can be any string with no spaces representing a unique release label in the resource annotation. e.g. V0000007 | LONGVARCHAR |
| IN | *excludePathList* - colon-separated list of paths to exclude. | LONGVARCHAR |
| OUT | *status -* SUCCESS, FAIL | VARCHAR |
|  | Note: This applies to message and all "generated\*" variables:  If convertLF=Y then convert linefeed[CHR(10)] to [LF] for use by the invoking command-line script. |  |
| OUT | *message* - a status message with summary or EXCEPTION. | LONGVARCHAR |
| OUT | *generatedExportList* - Export path list. Each resource path is generated on its own line which can be used for pkg\_export option file. | LONGVARCHAR |
| OUT | *generatedIntrospectList* - Introspection list | LONGVARCHAR |
| OUT | *generatedReintrospectList* - Reintrospection list | LONGVARCHAR |
| OUT | *generatedDesignList* - Design-by-example list | LONGVARCHAR |
| OUT | *generatedRemoveList* - Remove list | LONGVARCHAR |
| OUT | *generatedRebindList* - Rebind list | LONGVARCHAR |
| OUT | *generatedRelocateList* - Relocate list | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | convertLF | N |
| IN | startPath | /shared%:/services/databases%:/services/webservices%:/policy% |
| IN | searchStr | V0000002 |
| IN | excludePathList | null |
| OUT | status | SUCCESS |
| OUT | message | SUMMARY: exportCount=4 introspectCount=5 reintrospectCount=5 designCount=2 removeCount=3 rebindCount=1 relocateCount=2 |
| OUT | generatedExportList | # Export resource list:  /shared/TEST/a\_test\_proc  /shared/TEST/a\_test\_sqlserver\_view  /shared/TEST/DSora  /shared/TEST/DSsqlsrv |
| OUT | generatedIntrospectList | # Introspect resource list:  /shared/TEST/DSora:DATA\_SOURCE  dvops2/P\_PROC:PROCEDURE  dvops2/P\_PROC2:PROCEDURE  /shared/TEST/DSsqlsrv:DATA\_SOURCE  dvops/dbo/TEST1:TABLE  dvops/dbo/TEST5:TABLE  dvops2/db1/TEST1:TABLE |
| OUT | generatedReintrospectList | # Reintrospect resource list:  /shared/TEST/DSora  /shared/TEST/DSsqlsrv |
| OUT | generatedDesignList | # Design-by-example resource list:  /shared/TEST/DSora/dvops2/P\_PROC|VARCHAR(255):2  /shared/TEST/DSora/dvops2/P\_PROC2|VARCHAR(255):2 |
| OUT | generatedRemoveList | # Remove resource list:  /shared/A\_TEST1/a\_test\_sqlserver\_view2:TABLE  /shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI7/dbo/TEST4:TABLE  /shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI7/dbo/TEST:TABLE |
| OUT | generatedRebindList | # Rebind resource list:  -rebind /shared/TEST/DSsqlsrv/dvops/dbo /shared/TEST/DSsqlsrv/dvops2/db1 |
| OUT | generatedRelocateList | # Relocate resource list:  -relocate /shared/TEST/DSsqlsrv/dvops/dbo /shared/TEST/DSsqlsrv/dvops/db1  -relocate /shared/TEST/DSsqlsrv/dvops /shared/TEST/DSsqlsrv/dvops2 |

### importArchiveFile

This procedure performs an import of an archive CAR file in the Data Virtualization (DV) server. It gets the archive CAR file from the local DV file system where the DV server is running. This procedure uses two lower level procedures:

createImportArchiveXQuery – This procedure performs a transformation on the VARBINARY car file archive data into the request XML required for the procedure /services/webservices/system/admin/archive/operations/createImportArchive.

getArchiveFile – This procedure retrieves the car from the Data Virtualization server file system.

Examples of using this procedure can be found here: /shared/ASAssets/Utilities/examples/archive/test\_importArchiveFile

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y or N | CHAR(1) |
| IN | car\_file\_os\_full\_path – full path to the archive file on the DV server. | LONGVARCHAR |
| IN | excludeResources - optional - A list resources that should not be imported. | VECTOR(ROW(  resourcePath LONGVARCHAR,  resourceType VARCHAR  )) |
| IN | relocateResources - optional - A mapping of resources from their location in the archive to where they should be imported. | VECTOR(ROW(  fromResourcePath LONGVARCHAR,  fromResourceType VARCHAR,  toResourcePath LONGVARCHAR,  toResourceType VARCHAR  )) |
| IN | rebindResources - optional - A mapping of resources references within the archive to where they should refer to. All resources containing the fromResourcePath and fromResourceType are rebound to the toResourcePath and toResourceType. | VECTOR(ROW(  fromResourcePath LONGVARCHAR,  fromResourceType VARCHAR,  toResourcePath LONGVARCHAR,  toResourceType VARCHAR  )) |
| IN | rebindUsers - optional - A mapping of users within the archive to whom they should be. (change ownership). All resources with the current ownership in fromDomain and fromUser and changed to the owner toDomain and toUser. | VECTOR(ROW(  fromDomain VARCHAR,  fromUser VARCHAR,  toDomain VARCHAR,  toUser VARCHAR  )) |
| IN | remapAttributes - optional - A list of resource attribute settings that should be applied on import. | VECTOR(ROW(  resourcePath LONGVARCHAR,  resourceType VARCHAR,  attributes VECTOR(/shared/ASAssets/Utilities/ repository/definitions/ RepositoryDefinitions.AttributeCompleteType)  attrName VARCHAR(255),-- mandatory  attrType VARCHAR(255),-- mandatory  attrValue LONGVARCHAR,-- optional (1 of attrValue, valueList, valueMap or valueArray must be supplied)  valueList VECTOR(itemType), -- optional  "type" VARCHAR(255),  "value" LONGVARCHAR  valueMap VECTOR(entryType),-- optional  "key" VECTOR(itemType),-- this vector always only contains 1 key  "type" VARCHAR(255),  "value" LONGVARCHAR  "value" VECTOR(itemType)-- this vector always only contains 1 value  "type" VARCHAR(255),  "value" LONGVARCHAR  valueArray VECTOR(LONGVARCHAR), -- optional  unset BIT-- optional - 1 is unset, 0 is set or null to ignore  )) |
| IN | importOptions - optional - A list of archive options indicating what additional features should be imported. By default, the same options used for export will be used for import.  OVERWRITE: Overwrite the existing resources if they exist.  INCLUDE\_CACHING: Include caching configurations for resources.  INCLUDE\_CUSTOM\_JAVA\_JARS: Include custom Java JARs in the export. (ADMIN ONLY)  INCLUDE\_STATISTICS: Include any resources statistics known about the table boundaries, and column boundaries.  INCLUDE\_DEPENDENCY: Gather and include all dependent resources for the resources you choose to export.  INCLUDE\_PHYSICAL\_SOURCE\_INFO: Include sensitive connection information for included physical sources. (OWNER ONLY)  INCLUDE\_REQUIRED\_USERS: Include the information about the required users in the export file.  INCLUDE\_SECURITY: Include resource privilege settings. (OWNER ONLY)  ENCRYPTION\_PASSWORD=password: TDV 8.x and up may use an encryption password.  IGNORE\_ENCRYPTION=true|false: TDV 8.x and up may ignore encryption. | VECTOR(LONGVARCHAR) |
| OUT | success | BIT |
| OUT | message | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | car\_file\_os\_full\_path | 'C:\my\_fsb.car' |
| IN | excludeResources - optional - A list resources that should not be imported. | VECTOR[  ('/shared/a/x2','TABLE'),  ('/shared/a/x1','PROCEDURE')  ] |
| IN | relocateResources - optional - A mapping of resources from their location in the archive to where they should be imported. | VECTOR[('/shared/a/x3','TABLE', '/shared/a/f1/x3','TABLE')]; |
| IN | rebindResources - optional - A mapping of resources references within the archive to where they should refer to. All resources containing the fromResourcePath and fromResourceType are rebound to the toResourcePath and toResourceType. | VECTOR[('/shared/a/x2','TABLE', '/shared/a/x3','TABLE'),]; |
| IN | rebindUsers - optional - A mapping of users within the archive to whom they should be. (change ownership). All resources with the current ownership in fromDomain and fromUser and changed to the owner toDomain and toUser. | VECTOR[('composite','admin', 'composite','user1')]; |
| IN | remapAttributes - optional - A list of resource attribute settings that should be applied on import. | VECTOR[  ('/shared/a/ds\_orders','DATA\_SOURCE',  VECTOR[('url', 'STRING', 'jdbc:postresql://localhost:9408/orders', null,null,null,null)])  ]; |
| IN | importOptions - optional - A list of archive options indicating what additional features should be imported. By default, the same options used for export will be used for import.  OVERWRITE: Overwrite the existing resources if they exist.  INCLUDE\_CACHING: Include caching configurations for resources.  INCLUDE\_CUSTOM\_JAVA\_JARS: Include custom Java JARs in the export. (ADMIN ONLY)  INCLUDE\_STATISTICS: Include any resources statistics known about the table boundaries, and column boundaries.  INCLUDE\_DEPENDENCY: Gather and include all dependent resources for the resources you choose to export.  INCLUDE\_PHYSICAL\_SOURCE\_INFO: Include sensitive connection information for included physical sources. (OWNER ONLY)  INCLUDE\_REQUIRED\_USERS: Include the information about the required users in the export file.  INCLUDE\_SECURITY: Include resource privilege settings. (OWNER ONLY) | VECTOR[('OVERWRITE'), ('INCLUDE\_PHYSICAL\_SOURCE\_INFO'), ('INCLUDE\_CACHING'), ('INCLUDE\_SECURITY')]; |
| OUT | success | 1 |
| OUT | message | Successfully imported archive file. |

1. How To Use ‘Cache’ Procedures

## Introduction

This section will show how to use the ‘Cache’ procedures.

### cacheRefreshSynch

This procedure is used in lieu of the DV cache policy capability for synchronized cache refresh control. Multiple views can be synchronized for caching in a single operation. The operations can be configured to be "S" synchronous or "P" parallel. When parallel is configured, multiple caches will be executed asynchronously. A "W" wait command can be placed after the parallel grouping to wait for all parallel caches to complete before proceeding.

IMPORTANT NOTE:

If you enable/disable the cache within CacheRefreshList, you must run this procedure manually once to change the cache setting because "updateResourceCacheEnabled" cannot be executed in background. It must be executed by a logged in user.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on, N=debug off | CHAR(1) |
| IN | cacheRefreshListPath - The base path where CacheRefreshList is located. [SELECT \* FROM <cacheRefreshListPath>]  The following columns must be exposed by the "cacheRefreshListPath". The resource may be a VIEW, PROCEDURE or table TABLE. It must have these columns in this order.  CacheType - Identifies the type cache refresh methodology to use  S=synchronous cache execution  P=parallel (asynchronous) cache execution  W=wait for parallel to finish  Enabled - Enable/Disable the resource cache  1=enable the cache before starting  0=disable the cache before starting.  GatherStats - Gather resource stats on the cache resource  1=true, gather stats  0=false, do not gather status  ResourcePath - The full path to the resource to be cached.  ResourceType - The type of resource [TABLE,PROCEDURE] | VARCHAR(4000) |
| IN | emailRecipients - One or more email aliases in a comma-separated list. If null then no emails will be sent. | VARCHAR(1000) |
| IN | cachePause - Number of milliseconds to pause in between asynchronous loops to check if cache has completed. | INTEGER |
| IN | skipAgeHours - Skip the cache based on age in hours. Typically set to null. | DECIMAL(6,2) |
| IN | startAtCacheRow - The row number to start from CacheRefreshList. | SMALLINT |

1. **Examples:**
   1. **Assumptions:** 
      1. **CURRENT\_TIMESTAMP is used at the time of invocation. For example, the format would be something like this: ‘2010-07-27 10:30:00’**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | personBirthdayTimestamp | ‘1990-01-01 00:00:00’ |
| OUT | age | 20 |

### getCacheAgeHours

Get the age of the cache resource returned as a decimal in hours.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | cacheResourcePath – The full path to the resource to be cached. | VARCHAR(4000) |
| OUT | cacheAgeHours – The age of the cache. | DECIMAL(6,2) |

1. **Examples:**
   1. **Assumptions:**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | cacheResourcePath | /shared/examples/orders |
| OUT | cacheAgeHours | 5.2 |

### getCacheStatusRows

This script is used to get the cache status rows for a particular view path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath – The cache resource path. a.k.a resourceid. | VARCHAR(4000) |
| IN | resourceType - The cache resource type: TABLE or PROCEDURE | VARCHAR |
| IN | statusFilter - null to retrieve all status rows or an explicit value such as A=sucessfully cached, I=in-progress, F=failed, C=clear mode, K=key | CHAR(1) |
| IN | serveridFilter - null/N - do not use the serverid as a filter. Y=use the serverid as a filter. This may be important to eliminate issues where different TDV servers are using the same cache database which is not recommended. | CHAR(1) |
| OUT | result cursor | OUT result PIPE(  clustered VARCHAR(40), - TDV cluster id or null if no cluster  serverid VARCHAR(255), - TDV server id: hostname.domain-port--uniqueid  resourceid VARCHAR(4000), - cached resource id. a.k.a. full path to cached table or procedure.  parameters VARCHAR(255), - procedure parameters or null if a table cache  Status VARCHAR, - status of cache entry  starttime TIMESTAMP, - cache status start time  finishtime TIMESTAMP, - cache status finish time  cleartime TIMESTAMP, - time when cache was cleared  bytes BIGINT, - the number of bytes cached  cachekey INTEGER, - cache key number  message VARCHAR(1024), - cache status message  bucket VARCHAR, - multi-table bucket - /CIS\_CACHE/dbo/V\_TEST\_FULL\_MULTI2  incrementalstatus VARCHAR, - incremental cache status, I=in progress, null=no status  incrementalmaintenancelevel VARCHAR, - incremental maintenance level is the max value of the last incremental cache. If null it triggers the initial snapshot load. If a value is present it triggers the delta load.  bucketprefixname VARCHAR, - the bucket prefix name. e.g. V\_TEST\_FULL\_MULTI  bucketcurrentname VARCHAR - the bucket current name. e.g. V\_TEST\_FULL\_MULTI2  ) |

1. **Examples:**
   1. **Assumptions:**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath – The cache resource path. a.k.a resourceid. | /shared/examples/orders |
| IN | resourceType - The cache resource type: TABLE or PROCEDURE | TABLE |
| IN | statusFilter - null to retrieve all status rows or an explicit value such as A=sucessfully cached, I=in-progress, F=failed, C=clear mode, K=key | A |
| IN | serveridFilter - null/N - do not use the serverid as a filter. Y=use the serverid as a filter. This may be important to eliminate issues where different TDV servers are using the same cache database which is not recommended. | Y |
| OUT | result cursor | |  |  | | --- | --- | | clusterid: | [NULL] | | serverid: | hostname.com-9810--1385555455 | | resourceid: | /shared/…/Views/test | | parameters: | [NULL] | | status: | A | | starttime: | 2022-10-27 07:57:14.211 | | finishtime: | 2022-10-27 07:57:14.244 | | cleartime: | [NULL] | | bytes: | 0 | | cachekey: | 33015 | | message: | [NULL] | | bucket: | /CIS\_CACHE/dbo/test | | incrementalstatus: | [NULL] | | incrementalmaintenancelevel: | [NULL] | | bucketprefixname: | test | | bucketcurrentname: | test2 | |

1. How To Use ‘Calculation’ Procedures

## Introduction

This section will show how to use the ‘Calculation’ procedures.

### calculateAge (Custom Function)

This function is used to calculate a person’s age given their birthday timestamp and the current timestamp at the time of calculation.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | personBirthdayTimestamp | TIMESTAMP |
| OUT | age | INTEGER |

1. **Examples:**
   1. **Assumptions:** 
      1. **CURRENT\_TIMESTAMP is used at the time of invocation. For example, the format would be something like this: ‘2010-07-27 10:30:00’**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | personBirthdayTimestamp | ‘1990-01-01 00:00:00’ |
| OUT | age | 20 |

### medianFromQuery (Custom Function)

This function calculates the median value from a single column query result. The query must be ordered for the median function to work properly (the query must include an ORDER BY clause or it will throw an exception.) If the number of rows in the result is odd, the function will return the middle value. If the number of rows in the result is even, the function will return the average of the two middle values. If there are no rows in the result, then the function will return NULL.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | query | /lib/util/System.Text (VARCHAR (2147483647)) |
| OUT | result | DOUBLE |

1. **Examples:**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | query | ‘SELECT FreightCharge FROM /shared/examples/ds\_orders/orders ORDER BY FreightCharge’ |
| OUT | result | 26.0 |

1. How To Use ‘Conversion’ Procedures

## Introduction

This section will show how to use the ‘Conversion’ procedures.

### convertBit (Custom Function)

Convert a string (T, F, 1, 0, Y, N, yes, no, true, or false) into a BIT response (1 or 0.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | request | VARCHAR(255) |
| OUT | response | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | request | ‘Yes’ |
| OUT | response | 1 |

### convertBoolean (Custom Function)

Convert a string (T, F, 1, 0, Y, N, yes, no, true, or false) into a Boolean response so that it makes it easier to test conditions.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | request | VARCHAR(255) |
| OUT | response | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | request | ‘Yes’ |
| OUT | response | true |

### convertDoubleToInteger (Custom Function)

Convert a double into an integer so as to remove trailing ‘.00000…’ values. It is useful for Oracle ID fields that were defined as NUMBER with no qualifying .0 decimal place. E.g. NUMBER(38.0).

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | d | DOUBLE |
| OUT | i | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | d | 1650.000000000 |
| OUT | i | 1650 |

### convertTemperatureUnit (Custom Function)

This procedure is used to convert passed in temperatures from one unit of measurement to another. Temperatures can be converted from / to degrees in Fahrenheit, Celsius and Kelvin.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputTemperature | FLOAT |
| IN | inputUnits | VARCHAR(1) |
| IN | targetUnits | VARCHAR(1) |
| OUT | convertedTemperature | FLOAT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputTemperature | 98.6 |
| IN | inputUnits | 'F' |
| IN | targetUnits | 'C' |
| OUT | convertedTemperature | 37.002959999999995 |

### convertYN (Custom Function)

Convert a string (T,F,1,0,Y,N,yes,no,true,false) into a Y or N response.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | request | VARCHAR(255) |
| OUT | response | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | request | ‘0’ |
| OUT | response | ‘N’ |

## 

1. How To Use ‘Deployment’ Procedures

## Introduction

This section will show how to use the ‘Deployment’ procedures.

### deployment/optionsfile/generateOptionsFile

This procedure is used to generate an options file on the TDV server. The procedure invokes generateOptions. If a file already exists, it adds an \_copy# to the end of the file name. You don't want to accidently overwrite the real file which contains the valid passwords. Once the copy is generated, it needs to be updated with the correct passwords. If the file path is left null or blank, the file options are simply returned in the output variable.

This procedure uses a helper procedure: /shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateOptions

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | envName – optional - environment name used with changePasswordToken ENV\_NAME if provided. | VARCHAR |
| IN | changePasswordToken - null=use default CHANGE\_PASSWORD.  General rule: Any spaces found in changePasswordToken are replaced with 2 underscores.  Format: [PRETEXT]\_[UPPER|LOWER(ENVNAME)]\_[UPPER|LOWER(PATH)]\_ [UPPER|LOWER(GRANDPARENT)]\_[UPPER|LOWER(PARENTNAME)]\_ [UPPER|LOWER(DATASOURCENAME)]\_[POSTTEXT]  PRETEXT - optional user provided text  ENVNAME - optional keyword denoting the environment name passed in as variable envName.  This can go before GRANDPARENT or after DATASOURCENAME but not in between.  Default: When UPPER(ENVNAME) or ENVNAME then the results will be upper.  When LOWER(ENVNAME) then the results will be lower.  PATH - optional keyword denoting the entire resource path.  Default: When UPPER(PATH) or PATH then the results will be upper.  When LOWER(PATH) then the results will be lower.  The use of path supersedes GRANDPARENT, PARENTNAME and DATASOURCENAME since it is inclusive of the entire path.  GRANDPARENT - optional keyword denoting the grandparent folder name.  Default: When UPPER(GRANDPARENT) or GRANDPARENT then the results will be upper.  When LOWER(GRANDPARENT) then the results will be lower.  Not applicable when PATH is used.  PARENTNAME - optional keyword denoting the parent folder name.  Default: When UPPER(PARENTNAME) or PARENTNAME then the results will be upper.  When LOWER(PARENTNAME) then the results will be lower.  Not applicable when PATH is used.  DATASOURCENAME - mandatory keyword denoting the data source name  Default: When UPPER(DATASOURCENAME) or DATASOURCENAME then the results will be upper.  When LOWER(DATASOURCENAME) then the results will be lower.  Not applicable when PATH is used.  POSTTEXT - optional user provided text  Example:  Given the path: /shared/labCommon/Physical/Metadata/postgres/ds\_inventory  Given the envName: TEST  Given the template: APP\_PSWD\_UPPER(ENVNAME)\_UPPER(PARENTNAME)\_UPPER(DATASOURCENAME)\_TOKEN or REPLACE\_PSWD\_ENVNAME\_PARENTNAME\_DATASOURCENAME\_TOKEN  Password token: APP\_PSWD\_TEST\_POSTGRES\_DS\_INVENTORY\_TOKEN  Given the template: APP\_PSWD\_LOWER(PARENTNAME)\_LOWER(DATASOURCENAME)\_TOKEN  Password token: APP\_PSWD\_postgres\_ds\_inventory\_TOKEN  Given the template: APP\_PSWD\_ENVNAME\_GRANDPARENT\_PARENTNAME\_DATASOURCENAME\_TOKEN  Password token: APP\_PSWD\_TEST\_METADATA\_POSTGRES\_DS\_INVENTORY\_TOKEN | LONGVARCHAR |
| IN | filePath - The path of the options file on the TDV server to generate. | LONGVARCHAR |
| IN | resourcePathList - null=retrieve options for all paths. comma-separated path list=retrieve options for paths in the list. | LONGVARCHAR |
| IN | excludePathList - null=no path exclusions. comma-separated path list=exclude paths in the list. | LONGVARCHAR |
| OUT | options | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | envName | ‘DEV’ |
| IN | changePasswordToken | ‘APP\_PSWD\_PARENTNAME\_DATASOURCENAME\_TOKEN’ |
| IN | filePath | ‘C:\TIBCO\master\_files\CoE\Files\option\_files\options.txt’ |
| IN | resourcePathList | ‘/shared/examples’ |
| IN | excludePathList |  |
| OUT | options | -set /shared/test\_import/"source"/Advisory DATA\_SOURCE path "C:\MyFiles\datafiles"  -set /shared/test\_import/"source"/ds\_XML DATA\_SOURCE raw:url file:///C:\MySW\TDV7.0.8\docs\examples/productCatalog.xml  -set /shared/test\_import/"source"/ds\_XMLCopy DATA\_SOURCE raw:url file:///C:\MySW\TDV7.0.8\docs\examples/productCatalog.xml  -set /shared/test\_import/"source"/ds\_inventory DATA\_SOURCE database inventory  -set /shared/test\_import/"source"/ds\_inventory DATA\_SOURCE host localhost  -set /shared/test\_import/"source"/ds\_inventory DATA\_SOURCE password CHANGE\_PASSWORD  -set /shared/test\_import/"source"/ds\_inventory DATA\_SOURCE port 9408  -set /shared/test\_import/"source"/ds\_inventory DATA\_SOURCE user tutorial  -set /shared/test\_import/"source"/ds\_orders DATA\_SOURCE database orders  -set /shared/test\_import/"source"/ds\_orders DATA\_SOURCE host localhost  -set /shared/test\_import/"source"/ds\_orders DATA\_SOURCE password CHANGE\_PASSWORD  -set /shared/test\_import/"source"/ds\_orders DATA\_SOURCE port 9408  -set /shared/test\_import/"source"/ds\_orders DATA\_SOURCE user tutorial  -set /shared/test\_import/"source"/ds\_ordersCopy DATA\_SOURCE database orders  -set /shared/test\_import/"source"/ds\_ordersCopy DATA\_SOURCE host localhost  -set /shared/test\_import/"source"/ds\_ordersCopy DATA\_SOURCE password CHANGE\_PASSWORD  -set /shared/test\_import/"source"/ds\_ordersCopy DATA\_SOURCE port 9408  -set /shared/test\_import/"source"/ds\_ordersCopy DATA\_SOURCE user tutorial |

### deployment/optionsfile/generateOptionsFileRebind

This procedure is used to generate an options file on the TDV server. The procedure invokes /helpers/generateRebindList. If a file already exists, it adds an \_copy# to the end of the file name. You don't want to accidently overwrite the real file which contains the valid passwords. Once the copy is generated, it needs to be updated with the correct passwords. If the file path is left null or blank, the file options are simply returned in the output variable.

This procedure is used to generate an options file rebind list. A rebind list specifies how views get rebound from one data source path to a different one. This procedure is meant to be executed on the source TDV server. It generates a list based on the starting path and the list type. If a mapping pairs list is provided on input it will attempt to map the source folder to a target folder. Therefore, the mapping pair list should be specifically for the target TDV server. Different target TDV servers may have different pair mappings. Mapping pairs are explained in detail below but essentially are how a view gets mapped from a source data source catalog and or schema to a data source with a different catalog and/or schema.

Note: The database object names are the same. Those do not get mapped. That being said, if resource level rebind generation was chosen, it would generate a low-level resource/object mapping. However, the automated mapping pairs will not provide that functionality of mapping 1 object name to another. The -rebind statement would have to be manually modified after it was generated for that low-level use case.

This procedure uses a helper procedure: /shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateRebindList

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | startingPath – The starting path in which to create the list. This may be a path from the published, shared or users section of TDV. | VARCHAR(4000) |
| IN | listType - This provides a way of filtering the result set. It may be a comma-separated list of 1 or more list types.  A=all data source resources including the data source. A is the equivalent of (D,C,S,R,T,P,L,TR,TP).  D=data source list  C=catalog list  S=schema list  R=resource list [all resources under a schema]  T=database table list  P=database procedure list  L=link list which comes from published resources  TR=tree list for XML type resources  TP=database tables and procedures only | VARCHAR |
|  | mappingPairs – Optional pipe-separated list of pairs for mapping datasource catalogs and schemas.  The list must contain a unique source to search for. Duplicates source is not allowed and an exception will be thrown.  Note: If a leading slash is not found for either the source or target it will automatically be added to the pair.  The following format shows 4 different layers of mappings:  Format: /DS1->/DS2|/DS1/SCHEMA1->/DS2/SCHEMA2|/DS1/CATALOG1/SCHEMA1->/DS2/CATALOG2/SCHEMA2|/DS1/CATALOG1->/DS2/CATALOG2|  Datasource to Datasource  Only the data source changes when listType=D and/or R,T,P,L,TR,TP  It only gets applied when there is a loan entry of a DATA\_SOURCE type.  /DS1->/DS2|  Datasource/Schema to Datasource/Schema  The datasource may be the same or different. The schema may be the same or different.  It only gets applied on Schema entries when listType=S and/or R,T,P,L,TR,TP  /DS1/SCHEMA1->/DS2/SCHEMA2|  Datasource/Catalog/Schema to Datasource/Catalog/Schema  The datasource may be the same or different. The Catalog may the same or different. The schema may be the same or different.  It only gets applied on Schema entries when listType=S and/or R,T,P,L,TR,TP  /DS1/CATALOG1/SCHEMA1-> /DS2/CATALOG2/SCHEMA2|  Datasource/Catalog to Datasource/Catalog  The datasource may be the same or different. The Catalog may the same or different.  It only gets applied on Catalog entries when listType=C and/or R,T,P,L,TR,TP  /DS1/CATALOG1->/DS2/CATALOG2| | LONGVARCHAR |
| IN | filePath - The path of the options file on the TDV server to generate. | LONGVARCHAR |
| OUT | message – Indicates whether the file was written or not. | LONGVARCHAR |
| OUT | options – The generated list of options. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | startingPath | /shared/TEST |
| IN | listType | D,S |
| IN | mappingPairs | /DS\_oracle\_12c/DVKPI7->/DS\_oracle\_12c/DVKPI8|  /DS\_oracle\_12c->/DS\_oracle\_12c|  /DS\_sqlserver\_2016/DVKPI7/dbo->/DS\_sqlserver\_2016/DVKPI8/dbo|  /DS\_sqlserver\_2016/DVKPI7->/DS\_sqlserver\_2016/DVKPI8| /ds\_XML->/ds\_XML1| |
| IN | filePath | C:\master\_files\CoE\Files\option\_files\options.txt |
| IN | message | SUCCESS - File was written to path=c:\temp\option-rebind.txt |
| OUT | options | -rebind /shared/TEST/DS\_oracle\_12c/DVKPI7/p1 /shared/TEST/DS\_oracle\_12c/DVKPI8/p1  -rebind /shared/TEST/DS\_oracle\_12c/DVKPI7/p1 /shared/TEST/DS\_oracle\_12c/DVKPI8/p2  -rebind /shared/TEST/DS\_sqlserver\_2016/DVKPI7/dbo/T1 /shared/TEST/DS\_sqlserver\_2016/DVKPI8/dbo/T1  -rebind /shared/TEST/DS\_sqlserver\_2016/DVKPI7/dbo/T2 /shared/TEST/DS\_sqlserver\_2016/DVKPI8/dbo/T2  -rebind /shared/TEST/ds\_XML/productCatalog.xml /shared/TEST/ds\_XML1/productCatalog.xml  -rebind /shared/TEST/ds\_XML/productCatalogDefinitions /shared/TEST/ds\_XML1/productCatalogDefinitions |

### deployment/optionsfile/getOptionsFileDuplicateTokens

This procedure is used to detect duplicate password token names for a generated options file on the TDV server. The procedure uses the same parameters to invoke /helpers/generateOptions just like the procedure that generates the option file: /shared/ASAssets/Utilities/deployment/optionsfile/generateOptionsFile

The output will provide a list of duplicate tokens and the associated datasource paths.

This procedure uses a helper procedure: /shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateOptions

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | envName – optional - environment name used with changePasswordToken ENV\_NAME if provided. | VARCHAR |
| IN | changePasswordToken - null=use default CHANGE\_PASSWORD.  General rule: Any spaces found in changePasswordToken are replaced with 2 underscores.  Format: [PRETEXT]\_[UPPER|LOWER(ENVNAME)]\_[UPPER|LOWER(PATH)]\_ [UPPER|LOWER(GRANDPARENT)]\_[UPPER|LOWER(PARENTNAME)]\_ [UPPER|LOWER(DATASOURCENAME)]\_[POSTTEXT]  PRETEXT - optional user provided text  ENVNAME - optional keyword denoting the environment name passed in as variable envName.  This can go before GRANDPARENT or after DATASOURCENAME but not in between.  Default: When UPPER(ENVNAME) or ENVNAME then the results will be upper.  When LOWER(ENVNAME) then the results will be lower.  PATH - optional keyword denoting the entire resource path.  Default: When UPPER(PATH) or PATH then the results will be upper.  When LOWER(PATH) then the results will be lower.  The use of path supersedes GRANDPARENT, PARENTNAME and DATASOURCENAME since it is inclusive of the entire path.  GRANDPARENT - optional keyword denoting the grandparent folder name.  Default: When UPPER(GRANDPARENT) or GRANDPARENT then the results will be upper.  When LOWER(GRANDPARENT) then the results will be lower.  Not applicable when PATH is used.  PARENTNAME - optional keyword denoting the parent folder name.  Default: When UPPER(PARENTNAME) or PARENTNAME then the results will be upper.  When LOWER(PARENTNAME) then the results will be lower.  Not applicable when PATH is used.  DATASOURCENAME - mandatory keyword denoting the data source name  Default: When UPPER(DATASOURCENAME) or DATASOURCENAME then the results will be upper.  When LOWER(DATASOURCENAME) then the results will be lower.  Not applicable when PATH is used.  POSTTEXT - optional user provided text  Example:  Given the path: /shared/labCommon/Physical/Metadata/postgres/ds\_inventory  Given the envName: TEST  Given the template: APP\_PSWD\_UPPER(ENVNAME)\_UPPER(PARENTNAME)\_UPPER(DATASOURCENAME)\_TOKEN or REPLACE\_PSWD\_ENVNAME\_PARENTNAME\_DATASOURCENAME\_TOKEN  Password token: APP\_PSWD\_TEST\_POSTGRES\_DS\_INVENTORY\_TOKEN  Given the template: APP\_PSWD\_LOWER(PARENTNAME)\_LOWER(DATASOURCENAME)\_TOKEN  Password token: APP\_PSWD\_postgres\_ds\_inventory\_TOKEN  Given the template: APP\_PSWD\_ENVNAME\_GRANDPARENT\_PARENTNAME\_DATASOURCENAME\_TOKEN  Password token: APP\_PSWD\_TEST\_METADATA\_POSTGRES\_DS\_INVENTORY\_TOKEN | LONGVARCHAR |
| IN | resourcePathList - null=retrieve options for all paths. comma-separated path list=retrieve options for paths in the list. | LONGVARCHAR |
| IN | excludePathList - null=no path exclusions. comma-separated path list=exclude paths in the list. | LONGVARCHAR |
| OUT | status - SUCCESS=no duplicates found. FAIL=duplicates entries were found. To resolve the issue, rename the datasource or parent folder to make unique. | VARCHAR |
| OUT | duplicateTokenList - null if no duplicates otherwise a list of duplicates in the format: count=[#] passwordToken=[token\_name] datasourcePath=[datasource\_path] | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | envName | ‘DEV’ |
| IN | changePasswordToken | ‘APP\_PSWD\_PARENTNAME\_DATASOURCENAME\_TOKEN’ |
| IN | resourcePathList | ‘/shared/examples’ |
| IN | excludePathList |  |
| OUT | status | FAIL |
| OUT | duplicateTokenList | count=[2] passwordToken=[APP\_PSWD\_F1\_DS\_\_ORDERS\_TOKEN] datasourcePath=[/shared/examples/f1/f1/"ds orders"]  count=[2] passwordToken=[APP\_PSWD\_F1\_DS\_\_ORDERS\_TOKEN] datasourcePath=[/shared/examples/f2/f1/"ds orders"] |

### deployment/optionsfile/listOptionsFileTokens

This procedure is used to list all password token names for a generated options file on the TDV server based on the input parameters resourcePathList and excludePathList.

The procedure uses the same parameters to invoke /helpers/generateOptions just like the procedure that generates the option file: /shared/ASAssets/Utilities/deployment/optionsfile/generateOptionsFile

The output will provide a list of password tokens and the associated datasource paths.

This procedure uses a helper procedure: /shared/ASAssets/Utilities/deployment/optionsfile/helpers/generateOptions

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | envName – optional - environment name used with changePasswordToken ENV\_NAME if provided. | VARCHAR |
| IN | changePasswordToken - null=use default CHANGE\_PASSWORD.  General rule: Any spaces found in changePasswordToken are replaced with 2 underscores.  Format: [PRETEXT]\_[UPPER|LOWER(ENVNAME)]\_[UPPER|LOWER(PATH)]\_ [UPPER|LOWER(GRANDPARENT)]\_[UPPER|LOWER(PARENTNAME)]\_ [UPPER|LOWER(DATASOURCENAME)]\_[POSTTEXT]  PRETEXT - optional user provided text  ENVNAME - optional keyword denoting the environment name passed in as variable envName.  This can go before GRANDPARENT or after DATASOURCENAME but not in between.  Default: When UPPER(ENVNAME) or ENVNAME then the results will be upper.  When LOWER(ENVNAME) then the results will be lower.  PATH - optional keyword denoting the entire resource path.  Default: When UPPER(PATH) or PATH then the results will be upper.  When LOWER(PATH) then the results will be lower.  The use of path supersedes GRANDPARENT, PARENTNAME and DATASOURCENAME since it is inclusive of the entire path.  GRANDPARENT - optional keyword denoting the grandparent folder name.  Default: When UPPER(GRANDPARENT) or GRANDPARENT then the results will be upper.  When LOWER(GRANDPARENT) then the results will be lower.  Not applicable when PATH is used.  PARENTNAME - optional keyword denoting the parent folder name.  Default: When UPPER(PARENTNAME) or PARENTNAME then the results will be upper.  When LOWER(PARENTNAME) then the results will be lower.  Not applicable when PATH is used.  DATASOURCENAME - mandatory keyword denoting the data source name  Default: When UPPER(DATASOURCENAME) or DATASOURCENAME then the results will be upper.  When LOWER(DATASOURCENAME) then the results will be lower.  Not applicable when PATH is used.  POSTTEXT - optional user provided text  Example:  Given the path: /shared/labCommon/Physical/Metadata/postgres/ds\_inventory  Given the envName: TEST  Given the template: APP\_PSWD\_UPPER(ENVNAME)\_UPPER(PARENTNAME)\_UPPER(DATASOURCENAME)\_TOKEN or REPLACE\_PSWD\_ENVNAME\_PARENTNAME\_DATASOURCENAME\_TOKEN  Password token: APP\_PSWD\_TEST\_POSTGRES\_DS\_INVENTORY\_TOKEN  Given the template: APP\_PSWD\_LOWER(PARENTNAME)\_LOWER(DATASOURCENAME)\_TOKEN  Password token: APP\_PSWD\_postgres\_ds\_inventory\_TOKEN  Given the template: APP\_PSWD\_ENVNAME\_GRANDPARENT\_PARENTNAME\_DATASOURCENAME\_TOKEN  Password token: APP\_PSWD\_TEST\_METADATA\_POSTGRES\_DS\_INVENTORY\_TOKEN | LONGVARCHAR |
| IN | resourcePathList - null=retrieve options for all paths. comma-separated path list=retrieve options for paths in the list. | LONGVARCHAR |
| IN | excludePathList - null=no path exclusions. comma-separated path list=exclude paths in the list. | LONGVARCHAR |
| OUT | result CURSOR (  cnt INTEGER,  tokenLength INTEGER,  passwordToken VARCHAR(4000),  datasourcePath VARCHAR(4000)  ) | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | envName | ‘DEV’ |
| IN | changePasswordToken | ‘APP\_PSWD\_PATH\_TOKEN’ |
| IN | resourcePathList | ‘/shared/examples’ |
| IN | excludePathList |  |
| OUT | result |  |
| |  |  |  |  | | --- | --- | --- | --- | | cnt | Token Length | passwordToken | datasourcePath | | 1 | 28 | APP\_PSWD\_EXAMPLES\_DS\_\_ORDERS | /shared/examples/"ds orders" | | 1 | 30 | APP\_PSWD\_EXAMPLES\_DS\_INVENTORY | /shared/examples/ds\_inventory | | 1 | 27 | APP\_PSWD\_EXAMPLES\_DS\_ORDERS | /shared/examples/ds\_orders | | 1 | 34 | APP\_PSWD\_EXAMPLES\_F1\_F1\_DS\_\_ORDERS | /shared/examples/f1/f1/"ds orders" | | 1 | 34 | APP\_PSWD\_EXAMPLES\_F2\_F1\_DS\_\_ORDERS | /shared/examples/f2/f1/"ds orders" | | | |

### deployment/privileges/importResourceOwnership

This procedure is typically invoked by the deployment script via the published datasource ASAssets to apply resource ownership from the previously saved resource\_ownership.txt file that lives on the TDV server.

This script uses helper procedures /shared/ASAssets/Utilities/deployment/privileges/helpers/splitResourceType

A deployment shell script may execute the following:

$DIRNAME/JdbcSample.[sh|bat] $DATABASE $HOST $DBPORT $USER $PASSWORD $DOMAIN "SELECT \* FROM Utilities.deployment.importResourceOwnership('N', '$RESOWNERFILE')".

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on, N=debug off. Debug writes to CONSOLE and LOG. | CHAR(1) |
| IN | resourceOwnershipFileName - The full TDV server file path to put the resource ownership into. | LONGVARCHAR |
| OUT | result - SUCCESS if no invalid paths, WARNING if invalid path list otherwise an exception is thrown. | VARCHAR |
| OUT | invalidPathList - If not null, then a comma-separated list of paths [path:type] that are invalid as they do not exist and no ownership was applied. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | resourceOwnershipFileName | ‘C:\TDVScripts\7.0\deployment\privileges\resource\_ownership.txt’ |
| OUT | result | SUCCESS |
| OUT | invalidPathList |  |

### deployment/privileges/importResourcePrivileges

This procedure is typically invoked by the deployment script via the published datasource ASAssets to apply privileges from the previously saved privileges.xml file that lives on the TDV server.

A deployment shell script may execute the following:

$DIRNAME/JdbcSample.[sh|bat] $DATABASE $HOST $DBPORT $USER $PASSWORD $DOMAIN "SELECT \* FROM Utilities.deployment.importResourcePrivileges('N', 1, 0, 0, '$PRIVSFILE', 'SET\_EXACTLY')"

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on, N=debug off. Debug writes to CONSOLE and LOG. | CHAR(1) |
| IN | recurseChildResources - A bit flag indicating whether the privileges of the resources in the resourcePrivsList should be recursively applied to any child resources (assumes the resource is a container.)  Values: 1=recurse children or 0=do not recurse | BIT |
| IN | recurseDependencies - A bit flag indicating whether the privileges of the resources in the resourcePrivsList resourcePrivsList should be applied to any resources that they use.  Values: 1=recurse dependencies or 0=do not recurse | BIT |
| IN | recurseDependents - A bit flag indicating whether the privileges of the resources in the resourcePrivsList resourcePrivsList should be applied to any resources that are used by them.  Values: 1=recurse dependents or 0=do not recurse | BIT |
| IN | privilegeFileName - The TDV server file path to the privileges.xml file. | LONGVARCHAR |
| IN | mode - OVERWRITE\_APPEND (default), SET\_EXACTLY | VARCHAR |
| OUT | result - SUCCESS if no invalid paths, WARNING if invalid path list otherwise an exception is thrown. | VARCHAR |
| OUT | invalidPathList - If not null, then a comma-separated list of paths [path:type] that are invalid as they do not exist and no ownership was applied. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | resourceOwnershipFileName | ‘C:\TDVScripts\7.0\deployment\privileges\privileges.xml’ |
| OUT | result | SUCCESS |
| OUT | invalidPathList |  |

### deployment/privileges/templates/runPrivilegeExport\_[1\_DEV|2\_TEST|3\_PROD]\_template

These three procedures are templates for determining what folders should have privileges exported and subsequently imported and applied. This procedure should be copied to a different location so that it does not get overwritten when the Utilities are upgraded.

runPrivilegeExport\_1\_DEV\_template – This template should be used for DEV only because it has null:null ownership:domain settings.

runPrivilegeExport\_2\_TEST\_template – This template should be used for higher environments such as TEST because it has admin:composite ownership:domain settings.

runPrivilegeExport\_3\_PROD\_template – This template should be used for higher environments such as PROD because it has admin:composite ownership:domain settings.

These scripts use helper procedures:

/shared/ASAssets/Utilities/deployment/privileges/helpers/exportResourcesPrivileges

/shared/ASAssets/Utilities/deployment/privileges/helpers/getResourcesPrivileges

/shared/ASAssets/Utilities/deployment/privileges/helpers/splitResourceType

The concept behind this is simple.

* This procedure is used to take the "AS-IS" snapshot of privileges for the given folder paths listed below at the time of execution.
* It is very important that the folder privileges for each folder listed be set exactly as they should be set before exporting the privilege snapshot to the privileges.xml file.
* Subsequently, the privilege can be re-applied from the snapshot [privileges.xml] using importResourcePrivileges.
* For deployment, it is recommended to invoke the published importResourcePrivileges after importing the car file.

DataSource=ASAssets

Procedure: Utilities.deployment.importResourcePrivileges

Instructions:

1. Modify /shared/ASAssets/Utilities/environment/getEnvName() and provide an environment name such as DEV, TEST, PROD etc based on the TDV server environment.
2. Copy this template to a different location outside of the /shared/ASAssets/Utilities folder so that it does not get overwritten if the Utilities are updated.
3. Modify the variable "scriptEnv" below to match environment name that comes from getEnvName().
4. Modify the paths and ownership. For DEV environments the ownership:domain should always be null:null because you don't want to change ownership of your DEV resources.
5. Set privileges on those folder paths exactly as they should be set for this point in time.
6. Execute the procedure to export the privilege.xml file and the resource\_ownership.txt file to the TDV server path that you provide.

Input:

Debug CHAR(1), - Y=debug on, N=debug off

privilegeFileName LONGVARCHAR - The full TDV server path location to privileges.xml file. e.g. /home/files/deployment/privileges/privileges.xml

resourceOwnershipFileName LONGVARCHAR - The full TDV server path location to the resource\_ownership.txt file. e.g. /home/files/deployment/privileges/resource\_ownership.txt

Output:

invalidPathList LONGVARCHAR - A comma-separated list of paths from this procedure that are not valid [do not exist].

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on, N=debug off. Debug writes to CONSOLE and LOG. | CHAR(1) |
| IN | privilegeFileName - The full TDV server path location to privileges.xml file. Example: /home/files/deployment/privileges/privileges.xml | LONGVARCHAR |
| IN | resourceOwnershipFileName - The full TDV server path location to the resource\_ownership.txt file. Example: /home/files/deployment/privileges/resource\_ownership.txt | LONGVARCHAR |
| OUT | invalidPathList - A comm-separated list of paths from this procedure that are not valid [do not exist]. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | privilegeFileName | ‘C:\TDVScripts\7.0\deployment\privileges\resource\_ownership.txt’ |
| IN | resourceOwnershipFileName | SUCCESS |
| OUT | invalidPathList |  |

### deployment/privileges/runAfterImport\_template

This procedure is a template for running internal DV procedures after import of a CAR file during a DV migration. This procedure should be copied to a different location so that it does not get overwritten when the Utilities are upgraded.

Instructions:

1. Modify /shared/ASAssets/Utilities/environment/getEnvName() and provide an environment name such as DEV, TEST, PROD etc based on the TDV server environment.
2. Copy this template to a different location outside of the /shared/ASAssets/Utilities folder so that it does not get overwritten if the Utilities are updated.
3. Modify the copied procedure
   1. Modify the variable "scriptEnv" to match environment name that comes from getEnvName().
   2. Add procedure calls to add additional logic that you want to "run after import".
4. Publish the copied procedure "runAfterImport" to /services/databases/ASAssets/Utilities/deployment
5. The script deployProject.[bat|sh] would invoke runAfterImport. See the following for script details: /shared/ASAssets/Utilities/deployment/\_README
6. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on, N=debug off. Debug writes to CONSOLE and LOG. | CHAR(1) |
| OUT | result - SUCCESS otherwise an exception is thrown. | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| OUT | result | SUCCESS |

### Deployment Script: deployAutomation.[bat|sh]

The purpose of deployAutomation.[bat|sh] is to automate the deployment of .car files to the target environment where this script is installed. This script works on the concept of an inbox, where the inbox represents a folder structure. The folder is a direct representation of the TDV folder structure and how privileges are to be applied in TDV.

The script will find all .car files in the sub-folder structure from oldest to newest and deploy them in sequence one-at-a-time. If collision-detection is configured in deployProjects.sh then .car file resources will be checked for timestamp collisions.

The script will output a one-line log entry in ./logs/Deploy\_Automation.log with success or failure of a .car file deployment.

**Requirements**:

* This script requires TDV 7.x or 8.x to be installed on the same server as these scripts.
* Windows only: This script requires powershell to be enabled for execution.

**Integration Options**:

* Option 1 – Execute stand-alone.
* Option 2 – Integrate with a scheduler
  + UNIX: Integrate with /etc/crontab.
    - \*/15 \* \* \* \* /opt/tibco/tdv/share/config/deployment/scripts/deployAutomation.sh -p deploy\_user\_password
  + Windows: Integrate with Windows Task Scheduler
* Option 3 – Integrate with a 3rd party Deployment tool.
  + The 3rd party tool should be able to invoke batch scripts on the TDV server.
  + If the 3rd party tool can only invoke local scripts then the TDV binaries must be installed on the 3rd party tool server to satisfy pkg\_import.sh requirements.

**Usage**:

deployAutomation.[bat|sh] [-f car-file-path] [-p DEPLOY\_USER\_PASSWORD] [-e external-id]

optional: -f full path to a .car file to deploy.

if -f is NOT provided then this script will search all sub-folders for .car files.

optional: -p deployment user password

If -p is NOT provided then this script will use local variable DEPLOY\_USER\_PASSWORD.

Optional: -e external id

Comes from a 3rd party deployment tool and is no more than 50 characters.

**Assumptions for apply privileges**:

Organization Name: ABCBank Filter in the privilege spreadsheet.

Project Name: Finance Identifies the project in the privilege spreadsheet.

SubProject Name: Accounting Identifies the sub-project in the privilege spreadsheet.

**TDV Folder Structure**:

/services/databases/Finance

/Accounting

/GL

/Taxes

/services/webservices/Finance

/Accounting

/GL

/Taxes

/shared/Finance

/Application

/Views

/Accounting

/GL

/Taxes

/Business

/Business

/Accounting

/GL

/Taxes

/Logical

/Accounting

/GL

/Taxes

/Physical

/Formatting

/Metadata

**Inbox Folder Structure**:

Base folder is a share accessible from UNIX and Windows desktop:

/tibco/tdv/share

Standard deployment scripts structure is added to the share:

/tibco/tdv/share/config/deployment/carfiles

The customized path representing TDV project folders and privileges is added:

/tibco/tdv/share/config/deployment/carfiles/ABCBank/Finance/Accounting

Org Project SubProject

/ABCBank/Finance

/Accounting

/Taxes

/GL

/Sources

### Deployment Script: deployProject.[bat|sh]

This batch script is for windows or UNIX and provides a way to deploy a package .car file from one environment to another. If the -c option is used, the batch file will automatically detect if .car file conversion is required when migrating a package .car file from 8.x to 7.x.

This script also replaces deployPrivs.[bat|sh] and allows the execution of privileges only using the -privsOnly option.

DISCLAIMER:

Migrating resources from 8.x to 7.x is not generally supported. However, it does provide a way to move basic functionality coded in 8.x to 7.x. It does not support the ability to move new features that exist in 8.x but do not exist in 7.x. Exceptions may be thrown in this circumstance.

**Usage**: deployProject.[bat|sh] -i <import\_CAR\_file> -o <options\_file> -h <hostname> -p <wsport> -u <username> -d <domain> -up <user\_password> -ep <encryptionPassword>

[-v] [-c] [-e] [-print] [-printOnly] [-printWarning] [-inp1 value] [-inp2 value] [-inp3 value]

[-privsOnly] [-pe privilege\_environment] [-pd privilege\_datasource]

[-po privilege\_organization] [-pp privilege\_project] [-ps privilege\_sub\_project]

[-sn privilege\_sheet\_name] [-rp privilege\_resource\_path] [-rt privilege\_resource\_type]

[-gn privilege\_group\_name] [-gt privilege\_group\_type] [-gd privilege\_group\_domain]

[-be privilege\_bypass\_errors]

**Parameter Definitions**:

-i [mandatory] import archive (CAR) file path (full or relative).

-h [mandatory] host name or ip address of the DV server deploying to.

-p [mandatory] web service port of the target DV server deploying to. e.g. 9400

-u [mandatory] username with admin privileges.

-d [mandatory] domain of the username.

-up [mandatory] user password.

-o [optional] options file path (full or relative).

-ep [optional] encryption password for the archive .CAR file for TDV 8.x.

-v [optional] verbose mode. Verbose is turned on for secondary script calls. Otherwise the default is verbose is off.

-c [optional] execute package .car file version check and conversion.

Use -c in environments where you are migrating from DV 8.x into DV 7.x.

If not provided, version checking and .car file conversion will not be done which would be optimal to use when all environments are of the same major DV version such as all DV 7.x or all DV 8.x

-e [optional] Encrypt the communication between client and TDV server.

-print [optional] print info and contents of the package .car file and import the car file. If -print is not used, the car will still be imported.

-printOnly [optional] only print info and contents of the package .car file and do not import or execute any other option. This option overrides -print.

-printWarning [optional] print the warnings for updatePrivilegesDriverInterface, importResourcePrivileges, importResourceOwnership and runAfterImport otherwise do not print them.

-privsOnly [optional] execute the configured privilege strategy only. Do no execute the full deployment.

Execute either privilege strategy 1 or 2 based on configuration. If strategy 2 is configured, then resource ownership may also be executed if configured.

**The following parameters may be passed into Strategy 1 for Privileges: updatePrivilegesDriverInterface**

These parameters act as filters against the spreadsheet or database table. The most common parameters are -pd, -pe, -po, -pp and -ps

-pe [mandatory] privilege environment name. [DEV, UAT, PROD]

-pd [optional] privilege datasource type. [EXCEL, DB\_LLE\_ORA, DB\_LLE\_SS, DB\_PROD\_ORA, DB\_PROD\_SS]

-po [optional] privilege organization name.

-pp [optional] privilege project name.

-ps [optional] privilege sub-project name.

-sn [optional] privilege excel sheet name. [Privileges\_shared, Privileges\_databases, Privileges\_webservices]

-rp [optional] privilege resource path - The resource path in which to get/update privileges. It may contain a wildcard "%".

-rt [optional] privilege resource type - The resource type in which to get/update privileges. It is always upper case.

This will only be used when no "Resource\_Path" or a single "Resource\_Path" is provided. It is not used when a list of "Resource\_Path" entries are provided.

E.g. DATA\_SOURCE - a published datasource or physical metadata datasource.

CONTAINER - a folder path, a catalog or schema path.

COLUMN - a column from a table

LINK - a published table or procedure. If it resides in the path /services and points to a TABLE or PROCEDURE then it is a LINK.

TABLE - a view in the /shared path.

PROCEDURE a procedure in the /shared path.

-gn [optional] privilege group name - The user/group name in which to get/update privileges.

-gt [optional] privilege group type - Valid values are USER or GROUP

-gd [optional] privilege group domain - The domain name in which to get/update privileges.

-be [optional] privilege bypass errors - Bypass errors. Throw exception when paths not found. N/Null (default) Do not bypass errors. Y=bypass resource not found errors but report them.

**The following parameters may be passed into Strategy 2 for Privileges: importResourcePrivileges**

-recurseChildResources [1 or 0] - A bit [default=1] flag indicating whether the privileges of the resources in the XML file should be recursively applied to any child resources (assumes the resource is a container).

-recurseDependencies [1 or 0] - A bit [default=0] flag indicating whether the privileges of the resources in the XML file should be applied to any resources that they use.

-recurseDependents [1 or 0] - A bit [default=0] flag indicating whether the privileges of the resources in the XML file should be applied to any resources that are used by them.

**The following parameter may be passed into validateDeployment and/or runAfterImport**:

-inp1 [optional] Use this to represent a unique id for validating the deployment contents with an external log.

Format: -inp1 value

-inp1 signals the variable input.

value is the actual value with double quotes when spaces are present.

-inp2 [optional] Use this to represent any value

Format: -inp2 value

-inp2 signals the variable input.

value is the actual value with double quotes when spaces are present.

-inp3 [optional] Use this to represent any value

Format: -inp3 value

-inp3 signals the variable input.

value is the actual value with double quotes when spaces are present.

**Dependencies**:

convertPkgFileV11\_to\_V10.[bat|sh]

JdbcSample.[bat|sh]

JdbcSample.class

**Instructions**:

1. Place the scripts together on the TDV server because they need access to DV\_HOME/bin directory backup\_export.[bat|sh], pkg\_import.[bat|sh] and init\_server\_keystore\_files.[bat|sh]
2. Modify the environment variables within the top of the script for the specific environment in which they will be used. The environment variables exist before the text: “DO NOT MODIFY BELOW THIS POINT:”
   1. Configure DV\_HOME
   2. Configure FULL\_BACKUP\_PATH
   3. If performing a conversion from 8.x to 7.x .car file, configure SERVER\_ATTRIBUTE\_DATABASE
   4. If using Strategy 1 privileges, configure STRATEGY1\_RESOURCE\_PRIVILEGE\_DATABASE
   5. If using Strategy 2 privileges, configure STRATEGY2\_RESOURCE\_PRIVILEGE\_DATABASE, STRATEGY2\_RESOURCE\_PRIVILEGE\_FILE, STRATEGY2\_RESOURCE\_OWNERSHIP\_DATABASE, STRATEGY2\_RESOURCE\_OWNERSHIP\_FILE
   6. If executing post-deployment procedures, then configure RUN\_AFTER\_IMPORT\_DATABASE and RUN\_AFTER\_IMPORT\_URL
      1. Publish the copied procedure "runAfterImport" to /services/databases/ADMIN
   7. If validating the deployment .car file contents against previous deployments then configure VALIDATE\_DEPLOYMENT\_DATABASE, VALIDATE\_DEPLOYMENT\_DIR, VALIDATE\_DV\_TABLE\_PATH, VALIDATE\_DV\_PROCEDURE\_PATH
      1. Create the database table DV\_DEPLOYMENT\_VALIDATION and sequence DV\_DEPLOY\_SEQ
      2. Create the procedure to get the sequence.
3. Configure JdbcSample.[bat|sh]
   1. Modify the DV\_HOME
      1. Windows: SET DV\_HOME=C:\MySW\TDV8.2
      2. UNIX: DV\_HOME="/MySW/TDV8.2"
   2. Modify \jre to \jdk for TDV 8.2 and higher
      1. Windows: set JAVA\_HOME=%DV\_HOME%\jdk
      2. UNIX: JAVA\_HOME="$DV\_HOME/jdk"
4. Review the following for script details: /shared/ASAssets/Utilities/deployment/\_README

### Privilege Script: deployPrivs.[bat|sh] (DEPRECATED)

The functionality in this script has been subsumed by deployProject.[bat|sh] -privsOnly.

### deployment/privilegesBestPractices/updatePrivilegesDriverInterface

REQUIREMENT:

This procedure requires Data Abstraction Best Practices to be installed /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/updatePrivilegesDriver

This version of the update privileges procedure allows for more fine-grained control over debug output and the ability to skip resource paths that do not exist and report the errors on them.

This procedure is used to update privileges by reading the privileges from an excel spreadsheet - Resource\_Privileges\_LOAD\_DB.xlsx.

sheets: Privileges\_shared - Update privileges for /shared

Privileges\_databases - Update privileges for /services/databases

Privileges\_webservices - Update privileges for /services/webservices

Multiple parameter iterations – This procedure supports multiple parameter iterations. For single values, DO NOT use commas. For multiple values, DO NOT add extra commas to the end of the line. The following parameters may contain multiple comma-separated parameters that are all coordinated with each other according to their ordinal position. Multiple parameters mean multiple iterations for the "updatePrivilegesDriver" procedure for a single invocation of this procedure. If there are no commas present then the parameters is only used once. If there are commas present, then the same number of commas must be present in other parameters in order to represent the same ordinal position unless there are no parameters for those parameters to be passed in. The value may be empty between commas indicating a null value. Commas may not be present within the data values unless they are escaped with "\_002C". The escape will be converted back into a comma upon processing. All values are trimmed [left and right] when processed. There should not be any spaces at the end of any parameter.

EXAMPLE 1 - Here is an example of multiple projects correlated with sub-projects:

Project= p1,p2,p3

SubProject= s1,,s3

The above will result in 3 iterations of the procedure.

Iteration 1: Project=p1, SubProject=s1

Iteration 2: Project=p2, SubProject=null

Iteration 3: Project=p3, SubProject=s3

EXAMPLE 2 - Here is an example of single values

Project= p1

SubProject= s1

The above will result in 1 iteration of the procedure.

Iteration 1: Project=p1, SubProject=s1

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Datasource – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:  EXCEL - Excel spreadsheet which gets uploaded to each CIS server.  DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality.  DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | Batch\_Privileges – REQUIRED\_FILTER: 1=Batch all privileges for the same path, type and recursion setting. 0=Process each spreadsheet line separately (no batching).  When the resource path in the spreadsheet changes, a batch is executed.  When the resource type in the spreadsheet changes, a batch is executed.  When the recursion identifier in the spreadsheet changes, a batch is executed.  A group of rows with like recursion may only be batched together otherwise the meaning of the privilege setting is not the same. | INTEGER |
| IN | Environment\_Name – REQUIRED FILTER: A single value of the Composite server environment in which to get/update privileges for: Example [DEV, TEST, UAT, PROD] | VARCHAR(255) |
|  | MULTIPLE ITERATION PARAMETERS BEGIN |  |
| IN | Organization – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2  This provides a filter for only applying privileges for the given organization. If left blank, all rows for the organization will be updated.  Multiple comma-separated organizations may be passed resulting in multiple executions of the updatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | VARCHAR(255) |
| IN | Project – RECOMMENDED FILTER: The name of the project that is hosted on CIS for this "Env Type".  This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated.  Multiple comma-separated projects may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | SubProject – RECOMMENDED FILTER: The name of the sub-project that is hosted on CIS for this "Env Type".  This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the sub-project will be updated.  Multiple comma-separated sub-projects may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END | LONGVARCHAR |
| IN | Sheet\_Name – OPTIONAL FILTER: The name of the Excel spreadsheet in which to get/update privileges from: [shared, databases, webservices]  Multiple comma-separated sub-projects may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Resource\_Path – OPTIONAL FILTER: The resource path in which to get/update privileges. It may contain a wildcard "%".  Multiple comma-separated resource-paths may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Resource\_Type – OPTIONAL FILTER: The resource type in which to get/update privileges. It is always upper case.  Multiple comma-separated resource-types may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END.   * DATA\_SOURCE - a published datasource or physical metadata datasource. * CONTAINER - a folder path, a catalog or schema path. * COLUMN - a column from a table * LINK - a published table or procedure. If it resides in the path /services and points to a TABLE or PROCEDURE then it is a LINK. * TABLE - a view in the /shared path. * PROCEDURE a procedure in the /shared path. | LONGVARCHAR |
| IN | Name\_ – OPTIONAL FILTER: The user/group name in which to get/update privileges.  Multiple comma-separated user/group names may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Name\_Type – OPTIONAL FILTER: Valid values are USER or GROUP  Multiple comma-separated name-types may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Domain\_Name – OPTIONAL FILTER: The domain name in which to get/update privileges.  Multiple comma-separated domain-names may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
|  | MULTIPLE ITERATION PARAMETERS END |  |
| IN | preview – N or null(default)=Do not preview. Execute the privileges. Y=preview what will get set but don't actually execute the privileges. | CHAR(1) |
| IN | forceDomainAsComposite – OPTIONAL PARAM: Y=true, N=false.   * Provides a way to force a conversion of the group from an LDAP domain name like "ldap" to a the "composite" domain. * This is useful when setting up the privileges for testing with composite groups that mirror the LDAP groups. This way the spreadsheet does not have to be modified. * For example, the LDAP group Composite\_Servicing\_Dev is automatically converted to the composite domain group composite\_servicing\_dev. | CHAR(1) |
| IN | inDebug1 – N/Null=No debug, Y (default)=Debug for this procedure - basic messages | CHAR(1) |
| IN | inDebug1Console – N/Null (default)=No debug, Y=Debug console. Determines whether to print the studio console window or not. | CHAR(1) |
| IN | inDebug1CISLog – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the CIS log or not. | CHAR(1) |
| IN | inDebug1ReadOnly – N/Null (default)=No debug, Y=Debug for READ only messages | CHAR(1) |
| IN | inDebug1RevokeAll – N/Null (default)=No debug, Y=Debug for Revoke All messages. | CHAR(1) |
| IN | inDebug2 – N/Null (default)=No debug, Y=Debug for 2nd and 3rd tier procedures (deep debug) | CHAR(1) |
| IN | inBypassErrors – Bypass errors. Throw exception when paths not found.   * N/Null (default) Do not bypass errors. * Y=bypass resource not found errors but report them. | CHAR(1) |
| OUT | result – SUCCESS if no invalid paths, WARNING if 0 privileges applied. | VARCHAR(255) |
| OUT | logOutputResult – Contains an iteration of the following with pipe delimiters:   * executeNum - The iteration number of this execution. * rowStatus - SUCCESS if no invalid paths, WARNING if 0 privileges applied or a resource path was bypassed because it was NOT FOUND. * rowsProcessed - Number of rows processed from the spreadsheet * sqlStatement - The invocation SQL Statement with all the parameters * logOutput - The output log that gets printed to the command line console is also output for any rows matching 'BYPASS \*\*ERROR\*\*' | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Typical parameter settings**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Datasource | EXCEL |
| IN | Batch\_Privileges | 1 |
| IN | Environment\_Name | PROD |
| IN | Organization | Org1 |
| IN | Project | Project1 |
| IN | SubProject | Subproject1 |
| IN | Sheet\_Name | [NULL] |
| IN | Resource\_Path | [NULL] |
| IN | Resource\_Type | [NULL] |
| IN | Name\_ | [NULL] |
| IN | Name\_Type | [NULL] |
| IN | Domain\_Name | [NULL] |
| IN | preview | [NULL] |
| IN | forceDomainAsComposite | N |
| IN | inDebug1 | Y |
| IN | inDebug1Console | N |
| IN | inDebug1CISLog | N |
| IN | inDebug1ReadOnly | N |
| IN | inDebug1RevokeAll | Y |
| IN | inDebug2 | N |
| IN | inBypassErrors | N |
| OUT | result |  |
| OUT | logOutputResult |  |

### deployment/privilegesBestPractices/validatePrivilegesDriverInterface

REQUIREMENT:

This procedure requires Data Abstraction Best Practices to be installed /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/validatePrivilegesDriver

This version of the validate privileges procedure allows for more fine-grained control over debug output and the ability to skip resource paths that do not exist and report the errors on them.

This procedure is used to validate privileges by reading the privileges from an excel spreadsheet - Resource\_Privileges\_LOAD\_DB.xlsx.

sheets: Privileges\_shared - Validate privileges for /shared

Privileges\_databases - Validate privileges for /services/databases

Privileges\_webservices - Validate privileges for /services/webservices

Multiple parameter iterations – This procedure supports multiple parameter iterations. For single values, DO NOT use commas. For multiple values, DO NOT add extra commas to the end of the line. The following parameters may contain multiple comma-separated parameters that are all coordinated with each other according to their ordinal position. Multiple parameters mean multiple iterations for the "validatePrivilegesDriver" procedure for a single invocation of this procedure. If there are no commas present then the parameters is only used once. If there are commas present, then the same number of commas must be present in other parameters in order to represent the same ordinal position unless there are no parameters for those parameters to be passed in. The value may be empty between commas indicating a null value. Commas may not be present within the data values unless they are escaped with "\_002C". The escape will be converted back into a comma upon processing. All values are trimmed [left and right] when processed. There should not be any spaces at the end of any parameter.

EXAMPLE 1 - Here is an example of multiple projects correlated with sub-projects:

Project= p1,p2,p3

SubProject= s1,,s3

The above will result in 3 iterations of the procedure.

Iteration 1: Project=p1, SubProject=s1

Iteration 2: Project=p2, SubProject=null

Iteration 3: Project=p3, SubProject=s3

EXAMPLE 2 - Here is an example of single values

Project= p1

SubProject= s1

The above will result in 1 iteration of the procedure.

Iteration 1: Project=p1, SubProject=s1

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Datasource – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:  EXCEL - Excel spreadsheet which gets uploaded to each CIS server.  DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality.  DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | Environment\_Name – REQUIRED FILTER: A single value of the Composite server environment in which to get/validate privileges for: Example [DEV, TEST, UAT, PROD] | VARCHAR(255) |
|  | **MULTIPLE ITERATION PARAMETERS BEGIN** |  |
| IN | Organization – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2  This provides a filter for only validating privileges for the given organization. If left blank, all rows for the organization will be updated.  Multiple comma-separated organizations may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | VARCHAR(255) |
| IN | Project – RECOMMENDED FILTER: The name of the project that is hosted on CIS for this "Env Type".  This provides a filter for only validating privileges for the given project. If left blank, all projects will be updated.  Multiple comma-separated projects may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | SubProject – RECOMMENDED FILTER: The name of the sub-project that is hosted on CIS for this "Env Type".  This provides a filter for only validating privileges for the given sub-project. If left blank, all rows for the sub-project will be updated.  Multiple comma-separated sub-projects may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END | LONGVARCHAR |
| IN | Sheet\_Name – OPTIONAL FILTER: The name of the Excel spreadsheet in which to get/update privileges from: [shared, databases, webservices]  Multiple comma-separated sub-projects may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Resource\_Path – OPTIONAL FILTER: The resource path in which to get/validateprivileges. It may contain a wildcard "%".  Multiple comma-separated resource-paths may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Resource\_Type – OPTIONAL FILTER: The resource type in which to get/validateprivileges. It is always upper case.  Multiple comma-separated resource-types may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END.   * DATA\_SOURCE - a published datasource or physical metadata datasource. * CONTAINER - a folder path, a catalog or schema path. * COLUMN - a column from a table * LINK - a published table or procedure. If it resides in the path /services and points to a TABLE or PROCEDURE then it is a LINK. * TABLE - a view in the /shared path. * PROCEDURE a procedure in the /shared path. | LONGVARCHAR |
| IN | Name\_ – OPTIONAL FILTER: The user/group name in which to get/validateprivileges.  Multiple comma-separated user/group names may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Name\_Type – OPTIONAL FILTER: Valid values are USER or GROUP  Multiple comma-separated name-types may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
| IN | Domain\_Name – OPTIONAL FILTER: The domain name in which to get/validateprivileges.  Multiple comma-separated domain-names may be passed resulting in multiple executions of the validatePrivilegesDriver. DO NOT ADD EXTRA COMMAS AT THE END. | LONGVARCHAR |
|  | **MULTIPLE ITERATION PARAMETERS END** |  |
| IN | forceDomainAsComposite – OPTIONAL PARAM: Y=true, N=false.   * Provides a way to force a conversion of the group from an LDAP domain name like "ldap" to a the "composite" domain. * This is useful when setting up the privileges for testing with composite groups that mirror the LDAP groups. This way the spreadsheet does not have to be modified. * For example, the LDAP group Composite\_Servicing\_Dev is automatically converted to the composite domain group composite\_servicing\_dev. | CHAR(1) |
| IN | inDebug1 – N/Null=No debug, Y (default)=Debug for this procedure - basic messages | CHAR(1) |
| IN | inDebug1Console – N/Null (default)=No debug, Y=Debug console. Determines whether to print the studio console window or not. | CHAR(1) |
| IN | inDebug1CISLog – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the CIS log or not. | CHAR(1) |
| IN | inDebug1ReadOnly – N/Null (default)=No debug, Y=Debug for READ only messages | CHAR(1) |
| IN | inDebug2 – N/Null (default)=No debug, Y=Debug for 2nd and 3rd tier procedures (deep debug) | CHAR(1) |
| IN | inBypassErrors – Bypass errors. Throw exception when paths not found.   * N/Null (default) Do not bypass errors. * Y=bypass resource not found errors but report them. | CHAR(1) |
| OUT | result – SUCCESS if no invalid paths, WARNING if 0 privileges applied. | VARCHAR(255) |
| OUT | logOutputResult – Contains an iteration of the following with pipe delimiters:   * executeNum - The iteration number of this execution. * rowStatus - SUCCESS if no invalid paths, WARNING if 0 privileges applied or a resource path was bypassed because it was NOT FOUND. * rowsProcessed - Number of rows processed from the spreadsheet * sqlStatement - The invocation SQL Statement with all the parameters * logOutput - The output log that gets printed to the command line console is also output for any rows matching 'BYPASS \*\*ERROR\*\*' | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Typical parameter settings**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Datasource | EXCEL |
| IN | Environment\_Name | PROD |
| IN | Organization | Org1 |
| IN | Project | Project1 |
| IN | SubProject | Subproject1 |
| IN | Sheet\_Name | [NULL] |
| IN | Resource\_Path | [NULL] |
| IN | Resource\_Type | [NULL] |
| IN | Name\_ | [NULL] |
| IN | Name\_Type | [NULL] |
| IN | Domain\_Name | [NULL] |
| IN | forceDomainAsComposite | [NULL] |
| IN | inDebug1 | Y |
| IN | inDebug1Console | N |
| IN | inDebug1CISLog | N |
| IN | inDebug1ReadOnly | N |
| IN | inDebug2 | N |
| IN | inBypassErrors | N |
| OUT | result |  |
| OUT | logOutputResult |  |

1. How To Use ‘Documentation’ Procedures

## Introduction

This section describes the routines using the “documentation” procedures.

### getDocumentationDriver

This is the documentation driver procedure. It is used to generate documentation for composite resources and save to a file. The “/shared/ASAssets/Utilities/documentation/constants” procedure sets the defaults for a number of parameters. Because there is no output for this procedure it could be used as a trigger procedure if a customer wanted to generate documentation on a scheduled basis.

The best practice for the developer is to copy the constants() procedure to the project directory and modify the constants there. Pass in the location of that procedure to this driver. This allows you to customize the default values once.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **in\_resourcePath** – The starting CIS path for which to introspect resources and generate documentation. | pathType |
| IN | **in\_resourceType** – The starting CIS resource type for the resource path. | VARCHAR |
| IN | **in\_filePath** – The full file system path to generate the documentation file to. | pathType |
| IN | **in\_docPreambleImpl** – The CIS path to the procedure that represents the preamble of the documentation. If left null, the default “geDocPreambleImpl1” is used. | pathType |
| IN | **in\_docResourceFormatImpl** – The CIS path to the procedure that performs the documentation formatting. If left null, the default “getDocResourceFormatImpl1” is used. | pathType |
| IN | **in\_constantPath** – This is the path to the constants file | pathType |
| IN | **in\_switches** – Provides guidance on what documentation to print. The format is print\_switch=[option1|{default\_option2}|option3]  This is a space separate list with no spaces before or after the equal sign.  Print\_containers=[{none}|all] – print the resource container (folder)  print\_annotations=[none|{all}|nonblank|blank] – print all annotations whether they are blank or not  print\_resource\_projections=[none|{all}] – print the resource projections  print\_resources\_used=[none|{all}] – print the immediate child resources used by the parent resource  print\_datasource\_accessed=[none|{all}] – print the data source accessed list  print\_datasource\_lineage=[none|{all}] – print the data source lineage  print\_time=[{no}|yes] – print the time it takes to retrieve the full documentation for each resource and the final time  save\_file=[{no}|yes] – save the results to a file  save\_file\_intermediate=[{no}|yes] – save the file intermediately after each resource is completed  Example:   1. switches: when left blank or null then the defaults are taken   result: all documentation modules are printed   1. switches: print\_annotations=nonblank print\_resource\_projections=none print\_resources\_used=none print\_datasource\_lineage=none   result: only non-blank annotations are printed and nothing else | LONGVARCHAR |
| IN | **in\_excludeKeywordsInPathList** – Exclude keywords in path, case insensitive. Comma separated list. These are whole words and not wild cards. A word is defined by what exists between folder separators “/”. i.e. ‘analysis,archive’ | LONGVARCHAR |
| IN | **in\_excludePathsList** – Exclude actual paths. Double quotes are not required. Comma separated list. The exclude path list simply has to be present in any part of the resource path. This means that it can be a partial path. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | in\_resourcePath | /shared/examples |
| IN | in\_resourceType | CONTAINER |
| IN | in\_filePath | /temp/cis\_resource\_docs.txt |
| IN | in\_docPreambleImpl | /shared/ASAssets/Utilities/documentation/implementations/geDocPreambleImpl1 |
| IN | in\_docResourceFormatImpl | /shared/ASAssets/Utilities/documentation/implementations/getDocResourceFormatImpl1 |
| IN | in\_constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| IN | in\_switches | print\_containers=no print\_time=yes save\_file=yes save\_file\_intermediate=yes |
| IN | in\_excludeKeywordsInPathList | analysis,archive,test,validation |
| IN | in\_excludePathsList | /shared/ASAssets/Utilities,/shared/BestPractices,/lib |

### getAllDocumentationAPI

This procedure serves as an "API" procedure. It can be invoked by other application procedures to return the documentation. If the file path is left null, then it does not attempt to write to a file.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath** – The starting CIS path for which to introspect resources and generate documentation. | pathType |
| IN | **resourceType** – The starting CIS resource type for the resource path. | VARCHAR |
| IN | **docPreambleImpl** – The CIS path to the procedure that represents the preamble of the documentation. If left null, the default "geDocPreambleImpl1" is used. | pathType |
| IN | **docResourceFormatImpl** – The CIS path to the procedure that performs the documentation formatting. If left null, the default "getDocResourceFormatImpl1" is used. | pathType |
| IN | **constantPath** – This is the path to the constants file | pathType |
| IN | **switches** – Provides guidance on what documentation to print. The format is print\_switch=[option1|{default\_option2}|option3]  This is a space separate list with no spaces before or after the equal sign.  print\_containers=[{none}|all] - print the resource container (folder)  print\_annotations=[none|{all}|nonblank|blank] - print all annotations whether they are blank or not  print\_resource\_projections=[none|{all}] - print the resource projections  print\_resources\_used=[none|{all}] - print the immediate child resources used by the parent resource  print\_datasource\_accessed=[none|{all}] - print the data source accessed list  print\_datasource\_lineage=[none|{all}] - print the data source lineage  print\_time=[{no}|yes] - print the time it takes to retrieve the full documentation for each resource and the final time  save\_file=[{no}|yes] - save the results to a file  save\_file\_intermediate=[{no}|yes] - save the file intermediately after each resource is completed  Example:   1. switches: when left blank or null then the defaults are taken   result: all documentation modules are printed  2) switches: print\_annotations=nonblank print\_resource\_projections=none print\_resources\_used=none print\_datasource\_lineage=none  result: only non-blank annotations are printed and nothing else | LONGVARCHAR |
| IN | **excludeKeywordsInPathList** – Exclude keywords in path, case insensitive. Comma separated list. These are whole words and not wild cards. A word is defined by what exists between folder separators "/". i.e. 'analysis,archive' | LONGVARCHAR |
| IN | **excludePathsList** – Exclude actual paths. Double quotes are not required. Comma separated list. The exclude path list simply has to be present in any part of the resource path. This means that it can be a partial path. | LONGVARCHAR |
| IN | **filePath** – The full file system path to generate the documentation file to. | pathType |
| OUT | **formattedText** – The formatted text is returned for all documentation | /lib/util/System.Text |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples |
| IN | resourceType | CONTAINER |
| IN | docPreambleImpl | /shared/ASAssets/Utilities/documentation/implementations/geDocPreambleImpl1 |
| IN | docResourceFormatImpl | /shared/ASAssets/Utilities/documentation/implementations/getDocResourceFormatImpl1 |
| IN | constantPath | /shared/myproject/documentation/constants |
| IN | switches | print\_containers=no print\_time=yes save\_file=yes save\_file\_intermediate=yes |
| IN | excludeKeywordsInPathList | analysis,archive,test,validation |
| IN | excludePathsList | /shared/ASAssets/Utilities,/shared/BestPractices,/lib |
| IN | filePath | /temp/cis\_resource\_docs.txt |
| OUT | formattedText | See below: |

Composite Software DocumentationGenerated on 2012-11-11 08:10:34.614

================================================================================Resource Name: CompositeViewResource Path: /shared/examples/CompositeViewResource Type: TABLE SubType: SQL\_TABLEDescription:------------NoneResource Column Projection:--------------------------- Column Name Column Type Native Base Type Native Type ------------------------ -------------------- -------------------- -------------------- OrderID INTEGER N/A N/A ProductID INTEGER N/A N/A Discount DOUBLE N/A N/A OrderDate DATE N/A N/A CompanyName VARCHAR(50) N/A N/A CustomerContactFirstName VARCHAR(30) N/A N/A CustomerContactLastName VARCHAR(50) N/A N/A CustomerContactPhone VARCHAR(30) N/A N/A ProductName VARCHAR(32768) N/A N/A TransactionID INTEGER N/A N/A DateRequired DATE N/A N/A DatePromised DATE N/A N/A ShipDate DATE N/A N/A SupplierID INTEGER N/A N/A SupplierName VARCHAR(50) N/A N/A SupplierContactName VARCHAR(50) N/A N/A SupplierPhoneNumber VARCHAR(30) N/A N/A Resources Used:--------------- Resource Name Resource Type Subtype Resource Path -------------- -------------- ---------- ----------------------------- ViewOrder TABLE SQL\_TABLE /shared/examples/ViewOrder ViewSales TABLE SQL\_TABLE /shared/examples/ViewSales ViewSupplier TABLE SQL\_TABLE /shared/examples/ViewSupplier Data Source Accessed List:--------------------------- Datasource Name Enabled Type Subtype Datasource Path ----------------------------- ------- ----------- ------------------------ ------------------------------------------- ds\_orders 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_orders ds\_XML 1 DATA\_SOURCE XML\_FILE\_DATA\_SOURCE /shared/examples/ds\_XML ds\_inventory 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_inventory Data Source Lineage:--------------------seqnum id pid depth resource path 1 - 20587 0 /shared/examples/CompositeView 2 - 20658 20587 1 /shared/examples/ViewOrder 3 - 20741 20658 2 [CS] /shared/examples/ds\_orders/customers [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 4 - 20679 20658 2 [CS] /shared/examples/ds\_orders/orderdetails [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 5 - 20711 20658 2 [CS] /shared/examples/ds\_orders/orders [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 6 - 20670 20711 3 /shared/examples/ds\_orders 7 - 20729 20670 4 [CS] /shared/examples/ds\_orders/cache\_status [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 8 - 20671 20670 4 [CS] /shared/examples/ds\_orders/cache\_tracking [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 9 - 20689 20711 3 [CS] /shared/examples/ds\_orders/orders\_cache [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 10 - 20774 20587 1 /shared/examples/ViewSales 11 - 20679 20774 2 [CS] /shared/examples/ds\_orders/orderdetails [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 12 - 20786 20774 2 /shared/examples/productCatalog\_Transformation 13 - 20757 20786 3 [CS] /shared/examples/ds\_XML/productCatalog.xml [TREE.XML\_FILE\_TREE] [DS] /shared/examples/ds\_XML 14 - 20763 20587 1 /shared/examples/ViewSupplier 15 - 20606 20763 2 [CS] /shared/examples/ds\_inventory/inventorytransactions [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_inventory 16 - 20619 20763 2 [CS] /shared/examples/ds\_inventory/purchaseorders [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_inventory 17 - 20644 20763 2 [CS] /shared/examples/ds\_inventory/suppliers [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_inventoryResource Documentation Generation Time=0 00:00:00.266--------------------------------------------------------------------------------...

================================================================================---------------------Documentation Summary---------------------Starting Root Path: /shared/examples---------------------------------------------------------Print Switches Input: save\_file=yes save\_file\_intermediate=yes print\_time=yesprint\_containers=0 Key:[{none}=0|all=1]print\_annotations=1 Key:[none=0|{all}=1|nonblank=2|blank=3]print\_resource\_projections=1 Key:[none=0|{all}=1]print\_resources\_used=1 Key:[none=0|{all}=1]print\_datasource\_accessed=1 Key:[none=0|{all}=1]print\_datasource\_lineage=1 Key:[none=0|{all}=1]print\_time=1 Key:[{no}=0|yes=1]save\_file=1 Key:[{no}=0|yes=1]save\_file\_intermediate=1 Key:[{no}=0|yes=1]---------------------------------------------------------Total Number of Resources: 23---------------------------------------------------------Number of Published (LINK): 0Number of Folders (CONTAINER): 0Number of Views (TABLE): 15Number of Procedures (PROCEDURE): 3Number of Data Sources (DATA\_SOURCE): 3Number of XML (TREE): 1Number of Triggers (TRIGGER): 0Number of Connectors (CONNECTOR): 0Number of Def. Sets (DEFINITION\_SET): 1Number of Other resource type: 0---------------------------------------------------------Documentation Generation Time=0 00:00:01.7--------------------------------------------------------------------------------

### constants

These are default constants used by the documentation procedures.

The best practice for this procedure is to copy it and paste it into the project folder and configure the constant values as project specific rather than configuring /shared/ASAssets/Utilities/documentation/constants. If you were to upgrade the Utilities, you would lose the changes.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| CONSTANT | **docPreambleImpl** – Default location to the documentation preamble implementation procedure. | pathType |
| CONSTANT | **docResourceFormatImpl** – Default location to the resource formatting implementation procedure. | pathType |
| CONSTANT | **debug** – debug flag | CHAR(1) |
| CONSTANT | **debugTime** – first level of time display | CHAR(1) |
| CONSTANT | **debugTime2** – second level of time display for the resource vector loop | CHAR(1) |
| CONSTANT | **switches –** default switches | LONGVARCHAR |
| CONSTANT | **filePath** – the location for the output file | pathType |
| CONSTANT | **resourcePath** – default resource path to introspect | pathType |
| CONSTANT | **resourceType** – default resource type | VARCHAR |
| CONSTANT | **eol** – the end of line character | VARCHAR |
| CONSTANT | **indent2** –indent 2 spaces | VARCHAR |
| CONSTANT | **indent4** – indent 4 spaces | VARCHAR |
| CONSTANT | **padChar** – the padding characters for formatted printing | VARCHAR |
| CONSTANT | **beginSeparator** – the beginning separator for a resource grouping. | VARCHAR |
| CONSTANT | **endSeparator** – the ending separator for a resource grouping. | VARCHAR |
| CONSTANT | **minorSeparator** – the minor separator which may be used within a grouping. | VARCHAR |
| CONSTANT | **excludeKeywordsInPathList –** exclude keywords in path, case insensitive. Comma separated list. These are whole words and not wild cards. | LONGVARCHAR |
| CONSTANT | **excludePathsList –** exclude actual paths. Double quotes are not required. Comma separated list. | LONGVARCHAR |
| CONSTANT | **excludeDSPathsList** – exclude paths when finding matches for datasources. This is a comma separated list of paths to exclude from processing. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | | **Parameter Name** | **Parameter Value** |
| --- | --- | --- | --- |
| CONSTANT | docPreambleImpl | | /shared/ASAssets/Utilities/documentation/implementations/geDocPreambleImpl1 |
| CONSTANT | docResourceFormatImpl | | /shared/ASAssets/Utilities/documentation/implementations/getDocResourceFormatImpl1 |
| CONSTANT | debug | | Y or N |
| CONSTANT | debugTime | | Y or N |
| CONSTANT | debugTime2 | | Y or N |
| CONSTANT | switches | |  |
| CONSTANT | filePath | | /temp/cis\_resource\_docs.txt |
| CONSTANT | resourcePath | | /shared |
| CONSTANT | resourceType | | CONTAINER |
| CONSTANT | eol | | CHR(13) |
| CONSTANT | indent2 | | ‘ ‘ |
| CONSTANT | indent4 | | ‘ ‘ |
| CONSTANT | padChar | | ‘ ‘ |
| CONSTANT | beginSeparator | | 80 x ‘=’ |
| CONSTANT | endSeparator | | 80 x ‘-‘ |
| CONSTANT | minorSeparator | | ‘’ |
| CONSTANT | excludeKeywordsInPathList | | analysis,archive,test,validation |
| CONSTANT | excludePathsList | | /shared/ASAssets/Utilities,/shared/BestPractices,/lib |
| CONSTANT | excludeDSPathsList | | /shared/Common/COMPOSITE\_CACHE |

### documentationTrigger

The documentation trigger provides a template for a developer to copy and configure to automatically wake up and generate the documentation for a project.

The best practice for this procedure is to copy it and paste it into the project folder and configure the parameters as project specific rather than configuring /shared/ASAssets/Utilities/documentation/documentationTrigger. If you were to upgrade the Utilities, you would lose the changes.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Procedure Path** – The path to the documentation driver procedure. | pathType |
| IN | **Parameter Values** – refer to the procedure “getDocumentationDriver” for details on what to pass in. | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Procedure Path | /shared/ASAssets/Utilities/documentation/getDocumentationDriver |
| IN | Parameter Values | NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL, NULL  Refer to the procedure “getDocumentationDriver” for details on what to pass in. |

## helpers

This section describes the auxiliary procedures for documentation.

### helpers/getDocConstant (Custom Function)

This procedure gets a constant value from a dynamic constant path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **constantPath** – The path to the constants file ending with procedure parenthesis (). | pathType |
| IN | **constantName** – The name of the constant. | VARCHAR |
| OUT | **outValue**  – The value of the constant | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| IN | constantName | switches |
| OUT | outValue | print\_time=yes save\_file=yes save\_file\_intermediate=yes |

### helpers/getDocCounts

This procedure is used to increment the counts for the various resource type counters.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourceType** – The type of resource. | VARCHAR |
| INOUT | **numResources** – Total count for all resources. | ITNEGER |
| INOUT | **numContainers** – Count of containers (folders). | ITNEGER |
| INOUT | **numConnectors** – Count of connectors. | ITNEGER |
| INOUT | **numDefinitionSets** – Count of definition sets. | ITNEGER |
| INOUT | **numTriggers** – Count of triggers. | ITNEGER |
| INOUT | **numViews** – Count of views or database tables. | ITNEGER |
| INOUT | **numProcs** – Count of procedures. | ITNEGER |
| INOUT | **numTree** – Count of XML / Tree resources. | ITNEGER |
| INOUT | **numDatasources** – Count of data sources. | ITNEGER |
| INOUT | **numPublished** – Count of published (link) resources. | ITNEGER |
| INOUT | **numOtherType** – Count of other resources not covered above (catch-all). | ITNEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourceType | TABLE |
| INOUT | numResources | 1 |
| INOUT | numContainers | 1 |
| INOUT | numConnectors | 1 |
| INOUT | numDefinitionSets | 1 |
| INOUT | numTriggers | 1 |
| INOUT | numViews | 1 |
| INOUT | numProcs | 1 |
| INOUT | numTree | 1 |
| INOUT | numDatasources | 1 |
| INOUT | numPublished | 1 |
| INOUT | numOtherType | 1 |

### helpers/parseDocSwitches (Custom Function)

This procedure parses the switches that are passed in to determine what behavior should be taken.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **switches** – A space separated list of switches with no embedded spaces. Definitions:  print\_containers= [{none}|all] - print the resource container (folder)  print\_annotations= [none|{all}|nonblank|blank] - print all annotations whether they are blank or not  print\_resource\_projections= [none|{all}] - print the resource projections  print\_resources\_used= [none|{all}] - print the immediate child resources used by the parent resource  print\_datasource\_accessed= [none|{all}] - print the data source accessed list  print\_datasource\_lineage= [none|{all}] - print the data source lineage  print\_time= [{no}|yes] - print the time it takes to retrieve the full documentation for each resource and the final time  save\_file= [{no}|yes] - save the results to a file  save\_file\_intermediate= [{no}|yes] - save the file intermediately after each resource is completed | LONGVARCHAR |
| IN | **command** – One of the following:  print\_containers | print\_annotations | print\_resource\_projections | print\_resources\_used | print\_datasource\_accessed | print\_datasource\_lineage | print\_time | save\_file | save\_file\_intermediate | VARCHAR |
| OUT | **commandOptionValue** – The result command value [0,1,2,3] – none|no=0, all|yes=1, nonblank=2, blank=3 | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | | **Parameter Name** | **Parameter Value** |
| --- | --- | --- | --- |
| IN | switches | | print\_time=yes save\_file=yes save\_file\_intermediate=yes |
| IN | command | | print\_time |
| OUT | commandOptionValue | | 1 |

Default values

case

when command = 'print\_containers' then set commandOptionValue = 0; -- none

when command = 'print\_annotations' then set commandOptionValue = 1; -- all

when command = 'print\_resource\_projections' then set commandOptionValue = 1; -- all

when command = 'print\_resources\_used' then set commandOptionValue = 1; -- all

when command = 'print\_datasource\_accessed' then set commandOptionValue = 1; -- all

when command = 'print\_datasource\_lineage' then set commandOptionValue = 1; -- all

when command = 'print\_time' then set commandOptionValue = 0; -- no

when command = 'save\_file' then set commandOptionValue = 0; -- no

when command = 'save\_file\_initialize' then set commandOptionValue = 0; -- no

else set commandOptionValue = 0;

end case;

## implementations

This folder contains the different preamble and formatting implementations.

### implementations/getDocPreambleImpl1

This procedure provides a default implementation for retrieving the preamble to the documentation. The preamble is the text that occurs at the beginning of the documentation prior to the repeatable formatted resource text.

This procedure is invoked by the getAllDocumentationAPI in a loop. The output of this procedure provides the formatting for the preamble of the documentation. A user may wish to customize the preamble text for their specific project. The idea behind this procedure is that it provides a template for an implementation. A user of the documentation utilities may choose to copy and create a new implementation and then customize it for their project. Any new implementation “must” follow the input and output interface definitions as shown below.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath** – (optional) Full resource path which includes the path and the resource name . | pathType |
| IN | **resourceType** – The starting CIS resource type for the resource path. | VARCHAR |
| IN | **constantPath** – This is the path to the constants file. | pathType |
| OUT | **formattedText** – formatted text is out output complete with a separator at the beginning of the resource. | PIPE (formattedText /lib/util/System.Tex) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples |
| IN | resourceType | CONTAINER |
| IN | constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| OUT | formattedText | See below: |

Composite Software Documentation

Generated on 2012-07-29 00:25:56.510

### implementations/getDocResourceFormatImpl1

This procedure provides an implementation to retrieve and format the documentation for all resources located in the passed in starting folder.

This procedure is invoked by the getAllDocumentationAPI. This procedure recursively loops through all of the resources found within the given starting folder (CONTAINER). The output of this procedure provides the formatting for all CIS resource. The idea behind this procedure is that it provides a template for an implementation. A user of the documentation utilities may choose to copy and create a new implementation and then customize it for their project. Any new implementation “must” follow the input and output interface definitions as shown below..

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **startingResourcePath** – The starting folder path from which to start formatting the documentation. It is always of type CONTAINER. | pathType |
| IN | **startingResourceType** – The starting CIS resource type for the resource path. | VARCHAR |
| IN | **constantPath** – This is the path to the constants file. | pathType |
| IN | **switches** – Provides guidance on what to print for the documentation and save files. | LONGVARCHAR |
| IN | **excludeKeywordsInPathList** – Provides guidance on what to print documentation for | LONGVARCHAR |
| IN | **excludePathsList** – Exclude actual paths. Double quotes are not required. Comma separated list. | LONGVARCHAR |
| IN | **filePath** – The full file system path to generate the documentation file to. | pathType |
| OUT | **formattedText** – formatted text is out output complete with a separator at the beginning of the resource. | PIPE (formattedText /lib/util/System.Text) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | startingResourcePath | /shared/examples |
| IN | startingResourceType | CONTAINER |
| IN | constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| IN | switches | print\_time=yes save\_file=yes save\_file\_intermediate=yes |
| IN | excludeKeywordsInPathList | analysis,archive,test,validation |
| IN | excludePathsList | /shared/ASAssets/Utilities,/shared/BestPractices,/lib |
| IN | filePath | /temp/cis\_resource\_docs.txt |
| OUT | formattedText | See below: |

### implementations/getDocResourceFormatImpl1\_resource

This procedure provides an implementation to retrieve and format the documentation for a single CIS resource.

This procedure is invoked by the getDocResourceFormatImpl1 which controls the loop. The output of this procedure provides the formatting for a single CIS resource. The idea behind this procedure is that it provides a template for an implementation. A user of the documentation utilities may choose to copy and create a new implementation and then customize it for their project. Any new implementation “must” follow the input and output interface definitions as shown below.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath** – The the path to the resource. | pathType |
| IN | **resourceType** – The type of resource. | VARCHAR |
| IN | **constantPath** – This is the path to the constants file. | pathType |
| IN | **switches** – Provides guidance on what to print for documentation and save files. | LONGVARCHAR |
| OUT | **formattedText** – formatted text is out output complete with a separator at the beginning of the resource. | PIPE (formattedText /lib/util/System.Text) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples/CompositeView |
| IN | resourceType | TABLE |
| IN | constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| IN | switches | print\_time=yes save\_file=yes save\_file\_intermediate=yes |
| OUT | formattedText | See below: |

================================================================================Resource Name: CompositeViewResource Path: /shared/examples/CompositeViewResource Type: TABLE Subtype: SQL\_TABLEDescription:------------NoneResource Column Projection:--------------------------- Column Name Column Type Native Base Type Native Type ------------------------ -------------------- -------------------- -------------------- OrderID INTEGER N/A N/A ProductID INTEGER N/A N/A Discount DOUBLE N/A N/A OrderDate DATE N/A N/A CompanyName VARCHAR(50) N/A N/A CustomerContactFirstName VARCHAR(30) N/A N/A CustomerContactLastName VARCHAR(50) N/A N/A CustomerContactPhone VARCHAR(30) N/A N/A ProductName VARCHAR(32768) N/A N/A TransactionID INTEGER N/A N/A DateRequired DATE N/A N/A DatePromised DATE N/A N/A ShipDate DATE N/A N/A SupplierID INTEGER N/A N/A SupplierName VARCHAR(50) N/A N/A SupplierContactName VARCHAR(50) N/A N/A SupplierPhoneNumber VARCHAR(30) N/A N/A Resources Used:--------------- Resource Name Resource Type Subtype Resource Path -------------- -------------- ---------- ----------------------------- ViewOrder TABLE SQL\_TABLE /shared/examples/ViewOrder ViewSales TABLE SQL\_TABLE /shared/examples/ViewSales ViewSupplier TABLE SQL\_TABLE /shared/examples/ViewSupplier Data Source Accessed List:--------------------------- Datasource Name Enabled Type Subtype Datasource Path ---------------- ------- ----------- ------------------------ ------------------------------ ds\_orders 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_orders ds\_XML 1 DATA\_SOURCE XML\_FILE\_DATA\_SOURCE /shared/examples/ds\_XML ds\_inventory 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_inventory

Data Source Lineage:--------------------seqnum id pid depth resource path 1 - 20587 0 /shared/examples/CompositeView 2 - 20658 20587 1 /shared/examples/ViewOrder 3 - 20741 20658 2 [CS] /shared/examples/ds\_orders/customers [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 4 - 20679 20658 2 [CS] /shared/examples/ds\_orders/orderdetails [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 5 - 20711 20658 2 [CS] /shared/examples/ds\_orders/orders [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 6 - 20670 20711 3 /shared/examples/ds\_orders 7 - 20729 20670 4 [CS] /shared/examples/ds\_orders/cache\_status [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 8 - 20671 20670 4 [CS] /shared/examples/ds\_orders/cache\_tracking [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 9 - 20689 20711 3 [CS] /shared/examples/ds\_orders/orders\_cache [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 10 - 20774 20587 1 /shared/examples/ViewSales 11 - 20679 20774 2 [CS] /shared/examples/ds\_orders/orderdetails [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 12 - 20786 20774 2 /shared/examples/productCatalog\_Transformation 13 - 20757 20786 3 [CS] /shared/examples/ds\_XML/productCatalog.xml [TREE.XML\_FILE\_TREE] [DS] /shared/examples/ds\_XML 14 - 20763 20587 1 /shared/examples/ViewSupplier 15 - 20606 20763 2 [CS] /shared/examples/ds\_inventory/inventorytransactions [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_inventory 16 - 20619 20763 2 [CS] /shared/examples/ds\_inventory/purchaseorders [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_inventory 17 - 20644 20763 2 [CS] /shared/examples/ds\_inventory/suppliers [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_inventory--------------------------------------------------------------------------------

## modules

This section describes the modules used in creating documentation.

### modules/getDocDataSourceLineage

This procedure returns all the DATA\_SOURCE type resources found under the starting path. It returns the formatted text for two sections: "Data Sources Accessed List" and "Data Sources Lineage". The data sources accessed is a distinct list of data sources along with their type, path and whether they are enabled or not.

An example output is shown below:

Data Source Accessed List:--------------------------- Datasource Name Enabled Type Subtype Datasource Path ---------------- ------- ----------- ------------------------ ------------------------------ ds\_orders 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_orders ds\_XML 1 DATA\_SOURCE XML\_FILE\_DATA\_SOURCE /shared/examples/ds\_XML ds\_inventory 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_inventory

The data source lineage provides a top to bottom lineage starting with the "resources used" list. The format of the lineage shows indenting and a depth counter when the depth of the resource changes. Additionally, when a child source is found an indicator of [CS] is placed in front of the resource. The type of that resource is placed at the end of the child resource path in the format of [TYPE.SUBTYPE]. The parent data source path is placed under neath the child with a [DS] indicator in front of it.

An example output is shown below:

Data Source Lineage:--------------------seqnum id pid depth resource path 1 - 20587 0 /shared/examples/CompositeView 2 - 20658 20587 1 /shared/examples/ViewOrder 3 - 20741 20658 2 [CS] /shared/examples/ds\_orders/customers [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 4 - 20679 20658 2 [CS] /shared/examples/ds\_orders/orderdetails [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 5 - 20711 20658 2 [CS] /shared/examples/ds\_orders/orders [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 6 - 20670 20711 3 /shared/examples/ds\_orders 7 - 20729 20670 4 [CS] /shared/examples/ds\_orders/cache\_status [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 8 - 20671 20670 4 [CS] /shared/examples/ds\_orders/cache\_tracking [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders 9 - 20689 20711 3 [CS] /shared/examples/ds\_orders/orders\_cache [TABLE.DATABASE\_TABLE] [DS] /shared/examples/ds\_orders**Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath –** Full resource path which includes the path and the resource name . | pathType |
| IN | **resourceType –** The type of resource. | VARCHAR |
| IN | **constantPath** – This is the path to the constants file. | pathType |
| IN | **commandOptionValueDsAccessed** – The result command ('print\_datasource\_accessed') value [0,1] - none=0, all=1 | INTEGER |
| IN | **commandOptionValueDsLineage** – The result command ('print\_datasource\_lineage') value [0,1] - none=0, all=1 | INTEGER |
| OUT | **formattedText –** formatted text is out output complete with a separator at the beginning of the resource. | /lib/util/System.Text |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples/CompositeView |
| IN | resourceType | TABLE |
| IN | constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| IN | commandOptionValueDsAccessed | 1 |
| IN | commandOptionValueDsLineage | 1 |
| OUT | formattedText | See below: |

Data Source Accessed List:

---------------------------

Datasource Name Enabled Type Subtype Datasource Path

---------------- ------- ----------- ------------------------ ------------------------------

ds\_orders 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_orders

ds\_XML 1 DATA\_SOURCE XML\_FILE\_DATA\_SOURCE /shared/examples/ds\_XML

ds\_inventory 1 DATA\_SOURCE RELATIONAL\_DATA\_SOURCE /shared/examples/ds\_inventory

Data Source Lineage:

--------------------

seqnum id pid depth resource path

1 - 20587 0 /shared/examples/CompositeView

2 - 20658 20587 1 /shared/examples/ViewOrder

3 - 20741 20658 2 [CS] /shared/examples/ds\_orders/customers [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_orders

4 - 20679 20658 2 [CS] /shared/examples/ds\_orders/orderdetails [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_orders

5 - 20711 20658 2 [CS] /shared/examples/ds\_orders/orders [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_orders

6 - 20670 20711 3 /shared/examples/ds\_orders

7 - 20729 20670 4 [CS] /shared/examples/ds\_orders/cache\_status [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_orders

8 - 20671 20670 4 [CS] /shared/examples/ds\_orders/cache\_tracking [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_orders

9 - 20689 20711 3 [CS] /shared/examples/ds\_orders/orders\_cache [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_orders

10 - 20774 20587 1 /shared/examples/ViewSales

11 - 20679 20774 2 [CS] /shared/examples/ds\_orders/orderdetails [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_orders

12 - 20786 20774 2 /shared/examples/productCatalog\_Transformation

13 - 20757 20786 3 [CS] /shared/examples/ds\_XML/productCatalog.xml [TREE.XML\_FILE\_TREE]

[DS] /shared/examples/ds\_XML

14 - 20763 20587 1 /shared/examples/ViewSupplier

15 - 20606 20763 2 [CS] /shared/examples/ds\_inventory/inventorytransactions [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_inventory

16 - 20619 20763 2 [CS] /shared/examples/ds\_inventory/purchaseorders [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_inventory

17 - 20644 20763 2 [CS] /shared/examples/ds\_inventory/suppliers [TABLE.DATABASE\_TABLE]

[DS] /shared/examples/ds\_inventory

### modules/getDocResourceProjection

This procedure returns the column projection for TABLES and PROCEDURES.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath –** Full resource path which includes the path and the resource name . | pathType |
| IN | **resourceType –** Type of CIS resource to be created | VARCHAR |
| IN | **constantPath** – This is the path to the constants file. | pathType |
| OUT | **formattedText –** formatted text is out output complete with a separator at the beginning of the resource. | /lib/util/System.Text |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples/CompositeView |
| IN | resourceType | TABLE |
| IN | constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| OUT | formattedText | See below: |

Resource Column Projection:

---------------------------

Column Name Column Type Native Base Type Native Type

------------------------ -------------------- -------------------- -----------------

OrderID INTEGER N/A N/A

ProductID INTEGER N/A N/A

Discount DOUBLE N/A N/A

OrderDate DATE N/A N/A

CompanyName VARCHAR(50) N/A N/A

CustomerContactFirstName VARCHAR(30) N/A N/A

CustomerContactLastName VARCHAR(50) N/A N/A

CustomerContactPhone VARCHAR(30) N/A N/A

ProductName VARCHAR(32768) N/A N/A

TransactionID INTEGER N/A N/A

DateRequired DATE N/A N/A

DatePromised DATE N/A N/A

ShipDate DATE N/A N/A

SupplierID INTEGER N/A N/A

SupplierName VARCHAR(50) N/A N/A

SupplierContactName VARCHAR(50) N/A N/A

SupplierPhoneNumber VARCHAR(30) N/A N/A

### modules/getDocResourcesUsed

This procedure returns the list of resources (level 1) used by this resources. The immediate resource list is the list of resources that are directly invoked by the current resource being formatted. In the example below the resource being formatted is /shared/examples/CompositeView. The CompositeView has three resources that it uses for immediate invocation which include ViewOrder, ViewSales and ViewSupplier.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath –** Full resource path which includes the path and the resource name . | pathType |
| IN | **resourceType –** Type of CIS resource to be created | VARCHAR |
| IN | **constantPath** – This is the path to the constants file. | pathType |
| OUT | **formattedText –** formatted text is out output complete with a separator at the beginning of the resource. | /lib/util/System.Text |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples/CompositeView |
| IN | resourceType | TABLE |
| IN | constantPath | /shared/ASAssets/Utilities/documentation/constants() |
| OUT | formattedText | See below: |

Resources Used:

---------------

Resource Name Resource Type Subtype Resource Path

-------------- -------------- ---------- -----------------------------

ViewOrder TABLE SQL\_TABLE /shared/examples/ViewOrder

ViewSales TABLE SQL\_TABLE /shared/examples/ViewSales

ViewSupplier TABLE SQL\_TABLE /shared/examples/ViewSupplier

1. How To Use ‘Encoding’ Procedures

## Introduction

This section describes the routines for encoding, decoding, and encrypting text.

### CIS\_JCE\_PROVIDERS\_VIEW

A wrapper view for encoding/EncodingCJP/CISSecurityProviders().

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | results | CURSOR (  Provider LONGVARCHAR  Algorithm LONGVARCHAR  "Service Description" LONGVARCHAR  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | results | ('SUN version 1.6', 'CaseExactJKS', 'SUN: KeyStore.CaseExactJKS -> sun.security.provider.JavaKeyStore$CaseExactJKS'),  … |

## EncodingCJP

This section will show how to use the ‘Encoding’ CJP procedures.

### EncodingCJP/Base64Decode (Custom Function)

Accepts a Base64 encoded string as input and returns the Base64 decoded value of the string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | base64EncodedString | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | base64EncodedString | ‘Zm9v’ |
| OUT | result | ‘foo’ |

### EncodingCJP/Base64Encode (Custom Function)

Accepts a string as input and returns the Base64 encoded value of the string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputString | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString | ‘foo’ |
| OUT | result | ‘Zm9v’ |

### EncodingCJP/CISSecurityProviders

Lists all JCE providers, services and algorithms configured in the CIS JVM. A simple wrapper view, CIS\_JCE\_PROVIDERS\_VIEW, on top of this procedure (see above) allows for the lookup of algorithms. This procedure can be used, for example, to track down the root cause of failures in a client certificate and/or mutual authentication schemes between CIS and secure data providers (REST and SOAP web-services, some DBMS with advanced security mechanisms) or clients (SOAP and REST service consumes, app servers, ESBs, etc.) when these failures are caused by unsupported security algorithms.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | results | CURSOR (  Provider LONGVARCHAR  Algorithm LONGVARCHAR  "Service Description" LONGVARCHAR  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | results | ('SUN version 1.6', 'CaseExactJKS', 'SUN: KeyStore.CaseExactJKS -> sun.security.provider.JavaKeyStore$CaseExactJKS'),  … |

### EncodingCJP/DecryptFrom3DES

Decrypts a symmetrical Triple DES encrypted string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | "encrypted hex string" | LONGVARCHAR |
| IN | "digest seed" | LONGVARCHAR |
| OUT | "plain text" | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | "encrypted hex string" | '975d580dc4134fefbf290d8841f66dac' |
| IN | "digest seed" | 'the text used as the digest seed' |
| OUT | "plain text" | 'this is a test' |

### EncodingCJP/DecryptFromAES

Decrypts an AES-encrypted string.

This procedure provides an implementation of symmetrical AES decryption. This can be used to decode sensitive data in CIS views when such a security requirement exists.

With OOTB installation of CIS, the procedure will be capable of using a 128-bit encryption key. Stronger encryption (192-bit and 256-bit) are also supported, but you must replace the policy files local\_policy.jar and US\_export\_policy.jar files under jre/lib/security with the unlimited strength policies downloadable from the Oracle web site, and restart CIS.

Unlike the 3DES implementation, this AES decryption procedure allows the caller to specify both the encryption key and the initialization vector (IV) to feed into the algorithm.

However, for simplicity, if the caller does not supply the IV, it will be auto-generated from the encryption key. Note that this approach weakens the encryption, as the IV remains constant for all messages encrypted with the same key. Hence this practice is not recommended (or rather recommended not to follow.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | "encrypted hex string" | LONGVARCHAR |
| IN | "key" | LONGVARCHAR |
| IN | "IV" | LONGVARCHAR |
| OUT | "plain text" | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | "encrypted hex string" | 'Tvc7ThIwAyZ3EJ381gnIXQ==' |
| IN | "key" | '1234567812345678' |
| IN | "IV" | NULL |
| OUT | "plain text" | 'this is a test' |

### EncodingCJP/DecryptWithCISPrivKey

Decrypts a string encrypted using CIS's built in SSL certificate.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | "encrypted hex string" | LONGVARCHAR |
| IN | "keystore password" | LONGVARCHAR |
| OUT | "plain text" | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | "encrypted hex string" | '1232F6C78E9CE0D3EEB9CE05463942458C7689F19CF45664B9BA8FFADDDBFBE075880BAE9CEBA93FF560368D96896E28E100FC79604CC75175E4286B3B3D4BFDA6EA5A32E0568ECFB4C745D8FC8A58CFCBF3BBE1C00BC55A97E23C75717052BFE51131E38D8504FB35E8393C277E7BEF9E5E36CDEF5D19F6769CA673F2AD65EE' |
| IN | "keystore password" | 'changeit' |
| OUT | "plain text" | 'this is a test' |

### EncodingCJP/EncryptWith3DES

Encrypts a string using symmetrical Triple DES encryption.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | "plain text" | LONGVARCHAR |
| IN | "digest seed" | LONGVARCHAR |
| OUT | "encrypted raw bytes" | LONGVARBINARY |
| OUT | "encrypted hex string" | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | "plain text" | 'this is a test' |
| IN | "digest seed" | 'any text should work here' |
| OUT | "encrypted raw bytes" | b1bca4647b3b859f6f4736c595596fc0 |
| OUT | "encrypted hex string" | 'B1BCA4647B3B859F6F4736C595596FC0' |

### EncodingCJP/EncryptWithAES

Encrypts a string using AES encryption (in CBC mode.)

This procedure provides an implementation of symmetrical AES encryption. This can be used to encode sensitive data in CIS views when such a security requirement exists.

With OOTB installation of CIS, the procedure will be capable of using a 128-bit encryption key. Stronger encryption (192-bit and 256-bit) are also supported, but you must replace the policy files local\_policy.jar and US\_export\_policy.jar files under jre/lib/security with the unlimited strength policies downloadable from the Oracle web site, and restart CIS.

Unlike the 3DES implementation, this AES encryption procedure allows the caller to specify both the encryption key and the initialization vector (IV) to feed into the algorithm.

However, for simplicity, if the caller does not supply the IV, it will be auto-generated from the encryption key. Note that this approach weakens the encryption, as the IV remains constant for all messages encrypted with the same key. Hence this practice is not recommended (or rather recommended not to follow.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | "plain text" | LONGVARCHAR |
| IN | "key" | LONGVARCHAR |
| IN | "IV" | LONGVARCHAR |
| OUT | "encrypted raw bytes" | LONGVARBINARY |
| OUT | "encrypted string (base-64)" | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | "plain text" | 'this is a test' |
| IN | "key" | '1234567812345678' |
| IN | "IV" | NULL |
| OUT | "encrypted raw bytes" | 4ef73b4e1230032677109dfcd609c85d |
| OUT | "encrypted string (base-64)" | 'Tvc7ThIwAyZ3EJ381gnIXQ==' |

### EncodingCJP/EncryptWithCISPubKey

Encrypts a string using CIS's built in SSL certificate.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | "encrypted hex string" | LONGVARCHAR |
| IN | "keystore password" | LONGVARCHAR |
| OUT | "plain text" | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | "plain text" | 'this is a test' |
| IN | "keystore password" | 'changeit' |
| OUT | "encrypted raw bytes" | 1232f6c78e9ce0d3eeb9ce05463942458c7689f1...128 |
| OUT | "encrypted hex string" | '1232F6C78E9CE0D3EEB9CE05463942458C7689F19CF45664B9BA8FFADDDBFBE075880BAE9CEBA93FF560368D96896E28E100FC79604CC75175E4286B3B3D4BFDA6EA5A32E0568ECFB4C745D8FC8A58CFCBF3BBE1C00BC55A97E23C75717052BFE51131E38D8504FB35E8393C277E7BEF9E5E36CDEF5D19F6769CA673F2AD65EE' |

### EncodingCJP/MD5Hash (Custom Function)

Accepts a string as input and returns the MD5 hash value of the string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputString | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString | ‘foo’ |
| OUT | result | ‘acbd18db4cc2f85cedef654fccc4a4d8’ |

### EncodingCJP/SHA1Hash (Custom Function)

Accepts a string as input and returns the SHA1 hash value of the string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputString | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString | ‘foo’ |
| OUT | result | ‘0beec7b5ea3f0fdbc95d0dd47f3c5bc275da8a33’ |

1. How To Use ‘Environment’ Procedures

## Introduction

This section will show how to use the ‘Environment’ procedures.

NOTE: The environment procedures provide a way to get data virtualization environment settings.

### getEnvName (Custom Function)

This procedure returns the environment name for this specific server. It is useful when sending emails to be able to identify the environment type in the subject line giving the recipient a quick way of determining which environment is having issues. It may also be used by procedural logic to perform different logic based on the environment type.

It is intended to be modified for each environment. For a cluster it will have the same value by default because the DV code is duplicated on each server. The value can be whatever the DV development team wants up to 255 characters.

Some examples are shown below:

DEV=TDV Development environment

TEST=TDV Test environment

PROD=TDV Production environment

DR=TDV Disaster Recovery environment

\*\* WARNING \*\*: Be aware that when you update the Utilities, you must modify the environment setting again as it will be overwritten.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | envName | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** | |
| --- | --- | --- | --- |
| OUT | envName | DEV |

1. How To Use ‘File’ Procedures

## Introduction

This section will show how to use the ‘File’ procedures.

NOTE: When accessing a CIS instance remotely, the ‘File’ procedures act on the file system of the CIS instance’s host and not the file system of the host running the Composite Studio client.

### copyAll

This procedure is used to copy all of the files from one source location to target destination.

In this example, the end point folder name in the sourcePath will be used as the starting point and concatenated onto the end point of the target folder. Therefore, the result will be all folders under /Temp/vcs1/Utilities will be recursively copied to /Temp/vcs2 resulting in /Temp/vcs2/Utilities....

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| IN | newFilePath | VARCHAR(2147483647) |
| INOUT | mkdirCount (null initially) | INTEGER |
| INOUT | copyCount (null initially) | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourcePath | ‘/Temp/vcs1/Utilities’ |
| IN | targetpath | ‘/Temp/vcs2’ |
| INOUT | mkdirCount | 350 |
| INOUT | copyCount | 451 |

### getCisHome (Custom Function)

This function returns the folder on the CIS host where CIS is installed.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | VARCHAR(4096) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | ‘C:\Program Files\Composite Software\CIS 6.0.0’ |

### getFileSeparator (Custom Function)

This function returns the character used to separate folders/files in a path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | CHAR(1) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | ‘\’ (when run on a Windows host.) |

### removeAllFilter

This procedure is used to recursively remove all of the designated filter directories starting at a given source Path. For example, to remove the .svn directory from all levels of a source path, set the directoryFilter='.svn'.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourcePath | VARCHAR(2147483647) |
| IN | directoryFilter | VARCHAR |
| INOUT | removeCount (null initially) | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourcePath | ‘/Temp/vcs2/Utilities’ |
| IN | directoryFilter | ‘.svn’ |
| INOUT | removeCount | 39 |

## FileProcessingCJP

This section will show how to use the ‘File’ CJP procedures.

### FileProcessingCJP/archiveFile

Archive a file by moving it from a source directory to a target directory.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| IN | archivalDirectoryPath | VARCHAR(2147483647) |
| OUT | None: Throws exception upon failure |  |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/file1.txt’ |
| OUT | archivalDirectoryPath | ‘/files/archive’ |

### FileProcessingCJP/archiveFileTimestamp

Archive a file by moving it from a source directory to a target directory. Also renames the file to include a timestamp.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| IN | archivalDirectoryPath | VARCHAR(2147483647) |
| OUT | None: Throws exception upon failure |  |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/file1.txt’ |
| OUT | archivalDirectoryPath | ‘/files/archive’ |

### mFileProcessingCJP/copyFile

Copy a file from a source directory to a target directory.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| IN | newFilePath | VARCHAR(2147483647) |
| OUT | None: Throws exception upon failure |  |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/file1.txt’ |
| OUT | newFilePath | ‘/files/copydir’ |

### FileProcessingCJP/createFileASCII

Create an ASCII text file in a target directory. Provides an option to append which is useful for logging type files.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| IN | Append  0=do not append file, 1=append file | SMALLINT |
| IN | fileContent | LONGVARCHAR |
| OUT | None: Throws exception upon failure |  |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/newfile.txt’ |
| IN | append | 0 |
| IN | fileContent | ‘This is new file text.’ |

### FileProcessingCJP/createFileBinary

Create a binary file in a target directory.

1. **Parameters:**

| **Direction** | | **Parameter Name** | **Parameter Type** |
| --- | --- | --- | --- |
| IN | filePath | | VARCHAR(2147483647) |
| IN | Append  0=do not append file, 1=append file | | SMALLINT |
| IN | fileContent | | LONGVARBINARY |
| OUT | None: Throws exception upon failure | |  |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/newfile.dat’ |
| IN | append | 0 |
| IN | fileContent | --binary content is sent-- |

### FileProcessingCJP/existsDir (Custom Function)

Check to see if the requested director exists.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dirPath | VARCHAR(2147483647) |
| OUT | success | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dirPath | ‘/files/incoming’ |
| OUT | success (1=true, 0=false) | 1 |

### FileProcessingCJP/existsFile (Custom Function)

Check to see if the requested file exists.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| OUT | success | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/newfile.txt’ |
| OUT | success (1=true, 0=false) | 1 |

### FileProcessingCJP/getFileContentsAscii (Custom Function)

Get the contents of an ASCII file from the requested file path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| OUT | fileContent | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | 0 |
| OUT | fileContent | 20 |

### FileProcessingCJP/getFileContentsBinary (Custom Function)

Get the contents of a binary file from the requested file path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| OUT | fileContent | LONGVARBINARY |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | 0 |
| OUT | fileContent | 20 |

### FileProcessingCJP/getFileInfo

Get the file metadata.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | directoryPath | VARCHAR(2147483647) |
| IN | includeDirs – include directories names in the response. Y=include, N=do not include | VARCHAR(255) |
| OUT | FileInfo | CURSOR |
|  | filePath | VARCHAR(2147483647) |
|  | fileName | VARCHAR(2147483647) |
|  | fileTimestamp | TIMESTAMP |
|  | fileSize | BIGINT |
|  | isFile | SMALLINT |
|  | isDir | SMALLINT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | | | | **Parameter Value** | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| IN | | directoryPath | | | ‘/files/incoming’ | | | |
| IN | | includeDirs – include directories names in the response. Y=include, N=do not include | | | ‘Y’ | | | |
| OUT | | FileInfo | | | CURSOR | | | |
| filePath | | | filename | fileTimestamp | | fileSize | isFile | isDir |
| \files\incoming | | | incoming | 2010-07-16 13:00:28.65 | | 0 | 0 | 1 |
| \files\incoming\file1.txt | | | file1.txt | 2010-07-16 13:28:17.915 | | 512 | 1 | 0 |
| \files\incoming\file2.txt | | | file2.txt | 2010-07-16 13:29:01.001 | | 1024 | 1 | 0 |
| \files\incoming\file3.txt | | | file3.txt | 2010-07-16 13:30:43.873 | | 58 | 1 | 0 |

### FileProcessingCJP/getNewFiles

Get the new files that appear in the requested directory.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | directoryPath | VARCHAR(2147483647) |
| OUT | newFilenames | CURSOR |
|  | filePath | VARCHAR(2147483647) |
|  | fileName | VARCHAR(2147483647) |
|  | fileTimestamp | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | | | **Parameter Value** |
| --- | --- | --- | --- | --- |
| IN | directoryPath | | | ‘/files/incoming’ |
| OUT | FileInfo | | | CURSOR |
| filePath | | filename | fileTimestamp | |
| \files\incoming\file1.txt | | file1.txt | 2010-07-16 13:28:17.915 | |
| \files\incoming\file2.txt | | file2.txt | 2010-07-16 13:29:01.001 | |
| \files\incoming\file3.txt | | file3.txt | 2010-07-16 13:30:43.873 | |

### FileProcessingCJP/gunzipFile (Custom Function)

Gunzip a gzip file for the requested file path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| OUT | success (1=true, 0=false) | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/files.gzip’ |
| OUT | success (1=true, 0=false) | 1 |

### FileProcessingCJP/makeDirs (Custom Function)

Make the all the directories for the requested directory path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dirPath | VARCHAR(2147483647) |
| OUT | success (1=true, 0=false) | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dirPath | ‘/files/incoming/newdir’ |
| OUT | success (1=true, 0=false) | 1 |

### FileProcessingCJP/removeAll (Custom Function)

Remove all files and folders in the specific file path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dirPath | VARCHAR(2147483647) |
| IN | removeDirs  Y=remove all files and directories.  N=remove only files and leave the directory structure in place. | VARCHAR |
| OUT | success (1=true, 0=false) | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dirPath | ‘/files/incoming/newdir’ |
| IN | removeDirs | ‘Y’ |
| OUT | success (1=true, 0=false) | 1 |

### FileProcessingCJP/remove (Custom Function)

Remove a specific file or directory from the file system. For directories, it only removes a single directory at the end of a path and not the entire path of directories.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dirPath | VARCHAR(2147483647) |
| OUT | success (1=true, 0=false) | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dirPath | ‘/files/incoming/newdir’ |
| OUT | success (1=true, 0=false) | 1 |

### FileProcessingCJP/unzipFile (Custom Function)

Unzip a zip file for the requested file path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | filePath | VARCHAR(2147483647) |
| OUT | success (1=true, 0=false) | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | filePath | ‘/files/incoming/files.zip’ |
| OUT | success (1=true, 0=false) | 1 |

1. How To Use ‘Generate’ Procedures

## Introduction

This section will show how to use the ‘Generate’ procedures.

NOTE: The generation procedures provide a framework to generate views from a relational data source into the Data Abstraction Best Practices layers. In addition to generation there are also procedures that can be used to destroy resources from the Data Abstraction Best Practices layers.

### generateGuid

This procedure is used to generate a unique guid. This procedure is only supported for 8.x and higher. This procedure is not supported for DV 7.x and lower.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | generateUuidPath - The full DV path to the "localRestServices" datasource "generateUUID" path. Since this is a custom path it is relevant for each customer implementation. | VARCHAR(4000) |
| OUT | Result - A unique guid string | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | generateUuidPath | ‘/shared/Common/Datasources/localRestServices/generateUUID’ |
| OUT | result | 3709e0b2-36ba-42b6-a659-58c0ce3d8bda |

### generateViews

This script is used to provide a framework for introspection of a relational data source and generating views to the various Data Abstraction Best Practices layers. It provides a simple view generation into the four main layers described by the Data Abstraction Best Practices. This procedure copies privileges from the parent folder so it is important that proper privileges be assigned ahead of time on the project level folders and recursively pushed down to the layer folders. This is a simplified version of the open source Data Abstraction Best Practices in that it does not allow a formatting layer logical names to be derived from a spreadsheet.

This script will also copy both table and column annotations if they exist at the data source level. This script will copy SQL indexes and foreign keys from the data source to the layers. The published layer will inherit indexes and foreign keys directly from the view/table it was published from. Published views will have their own annotation but they will inherit column annotations from the view they were published from.

Published

/services/databases/Published\_Database/[Catalog]/Schema - mandatory

Application

/Application/Views/folder - optional layer generation

Business

/Business/Logical/folder - optional layer generation

Physical

/Physical/Formatting/folder - optional layer generation

/Physical/Metadata/[catalog]/schema - mandatory

The top [published] and bottom [metadata] layers are mandatory. The middle layers are optional and the input can be left null to indicate that no generation is required in those layers. Below is a representation of the Data Abstraction Layers and how the input variables map to those layers.

/services/databases

/TEST <-- exists ----^

/CAT1 <-- create ----^

/SCH1 <-- create ----^ = Published\_DB\_Layer\_Path

/V\_T1 <-- create ----^

/shared |

/TEST |

/Application |

/Views/SCH1 <-- create ^ = Application\_Layer\_Path

/T1 <-- create |

/Business |

/Logical/SCH1 <-- create ^ = Business\_Layer\_Path

/T1 <-- create |

/Physical |

/Formatting/SCH1 <-- create ^ = Formating\_Layer\_Path

/T1 <-- create |

/Metadata/Oracle/DS1 | = datasourcePath

/SCH1 <-- exists | = Source\_Phyisical\_Path [Schema Path] and just the schemaName

/T1 <-- exists - | [Table exists]

/T2

1. **Parameters:**
   1. Input Environment: set Environment to change behavior:
      1. utilities.generate.generateViews.**copyPrivilegeMode**
         1. [optional] Y (default)=copy privileges from the parent folder to the generated resource
         2. N=do not copy privileges from the parent to the generated resource
      2. Use the following code before invoking /shared/ASAssets/Utilities/generate/generateViews
      3. CALL /lib/util/SetEnvironment('utilities.generate.generateViews.copyPrivilegeMode',CAST(copyPrivilegeMode AS VARCHAR));
   2. utilities.generate.generateViews.**pauseMilliseconds**
      1. [optional] the number of milliseconds to pause before each view creation. e.g. 500 milliseconds-1/2 second or 1000 milliseonds-1 second
      2. Use the following code before invoking /shared/ASAssets/Utilities/generate/generateViews
      3. CALL /lib/util/SetEnvironment('utilities.generate.generateViews.pauseMilliseconds',CAST(pauseMilliseconds AS VARCHAR));
   3. utilities.generate.generateViews.**requestedActions**
      1. [optional] ADD or UPDATE or ADD,UPDATE
         1. ADD - [default behavior] Generate views based on newly added views. Since this is the default, it is not necessary to set this parameter.
         2. UPDATE - Generate views based on a view being updated.
         3. ADD,UPDATE - Generate views based on a view being added or updated.
      2. Use the following code before invoking /shared/ASAssets/Utilities/generate/generateViews
      3. CALL /lib/util/SetEnvironment('utilities.generate.generateViews.requestedActions',requestedActions);
   4. utilities.generate.generateViews.**sourceTableRule**
      1. [optional] REPLACE\_SLASH or RETURN\_SLASH - this would be used for source tables that contain a / such as SAP Hana tables. TDV cannot generate view that contain a slash "/".
         1. REPLACE\_SLASH=char - [default behavior] REPLACE\_SLASH=\_ - Use a single character to replace the slash "/" with such as hypon "-" or underscore "\_".
         2. RETURN\_SLASH=before|after. RETURN\_SLASH=after. Return the source table name value before the slash "/" or after the slash. e.g. package/myview. RETURN\_SLASH=after ==> myview RETURN\_SLASH=before ==> package
      2. Use the following code before invoking /shared/ASAssets/Utilities/generate/generateViews
      3. CALL /lib/util/SetEnvironment('utilities.generate.generateViews.sourceTableRule, sourceTableRule);

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Source\_Phyisical\_Path | VARCHAR(32768) |
| IN | datasourcePath | VARCHAR(4096) |
| IN | catalogName | VARCHAR |
| IN | schemaName | VARCHAR |
| IN | schemaTablePatterns | VARCHAR(4096) |
| IN | tableNames | LONGVARCHAR |
| IN | schemaProcedurePatterns | VARCHAR(4096) |
| IN | procedureNames | LONGVARCHAR |
| IN | separator | VARCHAR |
| IN | Formating\_Layer\_Path | VARCHAR(32768) |
| IN | Business\_Layer\_Path | VARCHAR(32768) |
| IN | Application\_Layer\_Path | VARCHAR(32768) |
| IN | Published\_DB\_Layer\_Path | VARCHAR(32768) |
| IN | prefix | VARCHAR(255) |
| IN | overwrite | SMALLINT |
| IN | copyAnnotation | SMALLINT |
| IN | copySqlIndexes | SMALLINT |
| IN | copyForeignKeys | SMALLINT |
| IN | debug | CHAR(1) |
| OUT | errStatus | VARCHAR |
| OUT | errMessage | LONGVARCHAR |
| OUT | introspectionResult | LONGVARCHAR |
| OUT | numResourcesIntrospectAdd | INTEGER |
| OUT | numTablesIntrospectAdd | INTEGER |
| OUT | tablesIntrospectAdd | LONGVARCHAR |
| OUT | numResourcesIntrospectSkip | INTEGER |
| OUT | numTablesIntrospectSkip | INTEGER |
| OUT | tablesIntrospectSkip | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** | |
| --- | --- | --- | --- |
| IN | Source\_Phyisical\_Path | /shared/ASAssets/Utilities/generate/examples/ExampleProject /Physical/Metadata/postgres/ds\_inventory/tutorial | |
| IN | datasourcePath | /shared/ASAssets/Utilities/generate/examples/ExampleProject /Physical/Metadata/postgres/ds\_inventory | |
| IN | catalogName | null | |
| IN | schemaName | tutorial | |
| IN | schemaTablePatterns | null | |
| IN | tableNames | categories,products | |
| IN | schemaProcedurePatterns | null | |
| IN | procedureNames | null | |
| IN | separator | , | |
| IN | Formating\_Layer\_Path | /shared/ASAssets/Utilities/generate/examples/ExampleProject /Physical/Formatting/tutorial | |
| IN | Business\_Layer\_Path | /shared/ASAssets/Utilities/generate/examples/ExampleProject /Business/Logical/tutorial | |
| IN | Application\_Layer\_Path | /shared/ASAssets/Utilities/generate/examples/ExampleProject /Application/Views/tutorial | |
| IN | Published\_DB\_Layer\_Path | /services/databases/ExampleProject | |
| IN | prefix | v\_ | |
| IN | overwrite | 1 | |
| IN | copyAnnotation | 1 | |
| IN | copySqlIndexes | 1 | |
| IN | copyForeignKeys | 1 | |
| IN | debug | Y | |
| OUT | errStatus | SUCCESS | |
| OUT | errMessage | null | |
| OUT | introspectionResult | <output from introspection> | |
| OUT | numResourcesIntrospectAdd | 2 |
| OUT | numTablesIntrospectAdd | 2 |
| OUT | tablesIntrospectAdd | inventorytransactions,purchaseorders |
| OUT | numResourcesIntrospectSkip | 5 |
| OUT | numTablesIntrospectSkip | 4 |
| OUT | tablesIntrospectSkip | categories,employees,products,suppliers |

### destroyDependentLineage [CONTAINER/TABLE/LINK only]

This procedure recursively destroys all of the "dependent" resources for a given starting folder (container), table or link resource. The general use case is to destroy a metadata, data source, schema folder or table which in turn destroys their dependent lineage from bottom to top with respect to the data abstraction layers. The starting path may be at any layer within the Data Abstraction Best Practices layers. In essence, the table/view lineage traverses the layers as shown in the graphic below. The term "dependent" refers to any table/view that selects another view. In reality give the layered model the deleting happens to the table/view that was selected and table/views that are considered above in the layers all the way to the published resource. It is important to note that this procedure operates from the bottom up in terms of the lineage. In other words, given a folder of resources, it will destroy the dependent lineage for each table resource found in that folder and the resources that select that table.

Caution: If multiple resources (tables) are selecting a single resource then \*\*ALL\*\* resources in that lineage will be destroyed no matter where those resources exist in the data virtualization server. Think of this deletion as a tree. It will start at the root level, move up the trunk and fan out to all branches that are connected via the lineage and destroy all leaves of the tree that are connected.

A cursor of metadata is returned that shows which resources were destroyed. For example, if the invoker of this procedure wants to destroy the Physical/Metadata datasource schema and all views that the resource is dependent upon throughout the layers, then this procedure will accomplish that goal.

Values: ex1 (view): /shared/TEST/Physical/Metadata/Oracle/DS1/T1

In this scenario, only the data source table "T1" and its lineage is removed.

The database table T1 will be destroyed if the input destroyInitialResource=1.

If a folder is empty the folder will be removed if the input destroyEmptyParentFolder=1.

/services/databases

/TEST

/CAT1

/SCH1

/V\_T1 <-- destroy ^

/shared |

/TEST |

/Application |

/Published/V\_T1 <-- destroy ^

/Views/V\_T1 <-- destroy ^

/Business |

/Business/V\_T1 <-- destroy ^

/Logical/V\_T1 <-- destroy ^

/Physical |

/Formatting/T1 <-- destroy ^

/Metadata/Oracle/DS1 |

/T1 <-- start here | - destroy table and lineage

/T2

Values: ex2 (folder): /shared/TEST/Physical/Metadata/Oracle/DS1

In this scenario, the data source schema "DS1" and its table lineage is removed.

The database tables will be destroyed if the input destroyInitialResource=1.

If a folder is empty the folder will be removed if the input destroyEmptyParentFolder=1.

/services/databases

/TEST

/CAT1

/SCH1

/V\_T1 <-- destroy ^

/V\_T2 <-- destroy ^

/shared |

/TEST |

/Application |

/Published/V\_T1 <-- destroy ^

/V\_T2 <-- destroy ^

/Views/V\_T1 <-- destroy ^

/V\_T2 <-- destroy ^

/Business |

/Business/V\_T1 <-- destroy ^

/V\_T2 <-- destroy ^

/Logical/V\_T1 <-- destroy ^

/V\_T2 <-- destroy ^

/Physical |

/Formatting/T1 <-- destroy ^

/T2 <-- destroy ^

/Metadata/Oracle/DS1 <-- start here | - destroy starting folder of resources

/T1 <-- destroy |

/T2 <-- destroy |

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | destroyInitialResource | SMALLINT |
| IN | destroyEmptyParentFolder | SMALLINT |
| IN | fullResourcePath | LONGVARCHAR |
| IN | inDebug | CHAR(1) |
| OUT | result PIPE (  actionType  id  resourceName  resourcePath  resourceType  subtype  ownerDomain  ownerName  impactLevel  impactMessage  enabled  ) | VARCHAR(255),  INTEGER,  VARCHAR(255),  VARCHAR(4096),  VARCHAR(40),  VARCHAR(40),  VARCHAR(255),  VARCHAR(255),  VARCHAR(255),  VARCHAR(32768),  BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | destroyInitialResource | 1 |
| IN | destroyEmptyParentFolder | 1 |
| IN | fullResourcePath | /shared/ASAssets/Utilities/generate/examples/ExampleProject /Physical/Metadata/postgres/ds\_inventory/tutorial |
| IN | inDebug | Y |
| OUT | result | Cursor of resources removed |

### destroyUsedLineage [CONTAINER/LINK/TABLE only]

This procedure recursively destroys the "used" resource for a given starting published resource, shared resource or folder. The general use case is to destroy a published folder or view which in turn destroys their used lineage from top to bottom with respect to the data abstraction layers. The resource "may" be a published link resource or folder container. If it is a published link then the "used" lineage is destroyed except any related foreign key and cache resources. If it is a published folder then all of the link/table resources are destroyed along with their "used" resource lineage. Related foreign key and cache are not destroyed.

The resource "may" be a /shared table/view resource or folder container. The same rules apply as above. However, an additional rule pertains to the dependent resources of a table resource. If a shared table is removed and it has a dependent resource, the dependent resource is left orphaned and will show up as red/impacted in Studio.

The valid resource types are CONTAINER (a.k.a FOLDER), LINK (published) or TABLE (shared). Procedures and non-TABLE/LINK resources are not supported and will be bypassed.

A cursor of metadata is returned that shows which resources were destroyed. For example, if the invoker of this procedure wants to destroy the published schema table and all views that the resource is using throughout the layers, then this procedure will accomplish that goal.

A specific view (link) resource may be deleted with its lineage or an entire schema of resources and their lineage may be deleted. The following is an example of the fullResourcePath:

Values: ex1 (view): /services/databases/TEST/CAT1/SCH1/V\_T1

In this scenario, only the published view "V\_T1" and its lineage is removed.

The database table T1 will be destroyed if the input destroyDatasourceResource=1.

If a folder is empty the folder will be removed if the input destroyEmptyParentFolder=1.

/services/databases

/TEST

/CAT1

/SCH1

/V\_T1 <-- start here v

/shared |

/TEST |

/Application |

/Published/V\_T1 <-- destroy v

/Views/V\_T1 <-- destroy v

/Business |

/Business/V\_T1 <-- destroy v

/Logical/V\_T1 <-- destroy v

/Physical |

/Formatting/T1 <-- destroy v

/Metadata/Oracle/DS1 |

/T1 <-- destroy |

/T2

Values: ex2 (folder): /services/databases/TEST/CAT1/SCH1

In this scenario, the entire schema "SCH1" and all the views and their lineage is removed.

This includes all layers and the metadata schema as long as the metadata schema is empty and the input destroyEmptyParentFolder=1.

/services/databases

/TEST

/CAT1

/SCH1 <-- start here | - destroy the schema, all views and lineage

/V\_T1 <-- destroy v - destroy V\_T1 and its lineage only

/V\_T2 <-- destroy v - destroy V\_T2 and its lineage only

/shared |

/TEST |

/Application |

/Published/SCH1/V\_T1 <-- destroy v - destroy V\_T1 and SCH1 if no objects remain

/V\_T2 <-- destroy v - destroy V\_T2

/Views/SCH1/V\_T1 <-- destroy v - destroy V\_T1 and SCH1 if no objects remain

/V\_T2 <-- destroy v - destroy V\_T2

/Business |

/Business/SCH1/V\_T1 <-- destroy v - destroy V\_T1 and SCH1 if no objects remain

/V\_T2 <-- destroy v - destroy V\_T2

/Logical/SCH1/V\_T1 <-- destroy v - destroy V\_T1 and SCH1 if no objects remain

/V\_T2 <-- destroy v - destroy V\_T2

/Physical |

/Formatting/SCH1/T1 <-- destroy v - destroy T1 and SCH1 if no objects remain

/T2 <-- destroy v - destroy T2

/Metadata/Oracle/DS1 |

/SCH1 <-- destroy | - destroy SCH1 if no objects remain

/T1 <-- destroy | - destroy T1

/T2 <-- destroy | - destroy T2

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | destroyDatasourceResource | SMALLINT |
| IN | destroyEmptyParentFolder | SMALLINT |
| IN | fullResourcePath | LONGVARCHAR |
| IN | inDebug | CHAR(1) |
| OUT | result PIPE (  actionType  seqNum  id  parentID  resourceDepth  treeType  resourceName  resourcePath  resourceType  subtype  enabled  ) | VARCHAR(255),  INTEGER,  INTEGER,  INTEGER,  INTEGER,  VARCHAR(255),  VARCHAR(255),  VARCHAR(4096),  VARCHAR(255),  VARCHAR(255),  BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | destroyDatasourceResource | 1 |
| IN | destroyEmptyParentFolder | /1 |
| IN | fullResourcePath | /services/databases/ExampleProject/TestCatalog/tutorial |
| IN | inDebug | Y |
| OUT | result | Cursor of resources removed |

### /helpers/createResourceProcess

The is a helper procedure which is used to manage the creation of the folder and views within the Data Abstraction Best Practices layers. It is invoked by generateViews().

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Res\_Source\_Path | VARCHAR(32768) |
| IN | Res\_Target\_Path | VARCHAR(4096) |
| IN | tableNames | VARCHAR |
| IN | separator | VARCHAR |
| IN | prefix | VARCHAR(255) |
| IN | overwrite | SMALLINT |
| IN | copyAnnotation | SMALLINT |
| IN | copySqlIndexes | SMALLINT |
| IN | copyForeignKeys | SMALLINT |
| IN | inDebug | CHAR(1) |
| OUT | numCreated | INTEGER |
| OUT | numUpdated | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Res\_Source\_Path | /shared/ASAssets/Utilities/generate/examples/ExampleProject /Physical/Metadata/postgres/ds\_inventory/tutorial |
| IN | Res\_Target\_Path | /services/databases/ExampleProject/TestCatalog/tutorial |
| IN | tableNames | products |
| IN | separator | , |
| IN | prefix | V\_ |
| IN | overwrite | 1 |
| IN | copyAnnotation | 1 |
| IN | copySqlIndexes | 0 |
| IN | copyForeignKeys | 0 |
| IN | inDebug | Y |
| OUT | numCreated | 1 |
| OUT | numUpdated | 0 |

### /examples/generate

The examples provide background information on various view generation and deletion strategies. A summary of information is provided below. Detailed instructions can be found in the “\_readme” script in the /generate/examples folder.

Summary

*1. pEx0\_create\_published\_ds\_group\_and\_user* - Create a default group [group1], user [user1], published data source [/services/databases/ExampleProject,] and assign privileges to [group1].

*2. pEx1\_create\_single\_table\_no\_layers* - Generate Views for a single table with no layers [products]

*3. pEx1\_destroy\_single\_table\_prefix* - Destroy Views for a single table [products]

*4. pEx2\_create\_entire\_schema\_no\_layers* - Generate Views for all tables with no layers [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*5. pEx2\_destroy\_entire\_schema* - Destroy Views for all tables [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*6. pEx3\_create\_entire\_schema\_all\_layers* - Generate Views for all tables through all layers [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*7. pEx4\_create\_entire\_schema\_all\_layers\_with\_fk* - Generate Views for all tables through all layers with foreign keys and annotations [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*8. pEx4\_destroy\_entire\_schema* - Destroy Views for all tables in all layers [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*9. pEx5\_create\_multiple\_tables\_all\_layers\_prefix* - Generate Views for multiple tables through all layers with a prefix [categories,products]

*10. pEx4\_destroy\_entire\_schema* - Destroy Views for all tables in all layers [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*11. pEx6\_create\_entire\_schema\_all\_layers\_no\_idx\_fk* - Generate Views for all table through all layers with no annotations, indexe or foreign keys [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*12. pEx4\_destroy\_entire\_schema* - Destroy Views for all tables in all layers [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

*13. pEx7\_create\_multiple\_tables\_all\_layers\_no\_prefix* - Generate Views for multiple tables through all layers with no prefix [categories,products]

*14. pEx4\_destroy\_entire\_schema* - Destroy Views for all tables in all layers [categories, employees, inventorytransactions, products, purchaseorders, suppliers]

1. How To Use ‘Logging’ Procedures

## Introduction

This section will show how to use the ‘Logging’ procedures.

### auditLogger (deprecated)

Use auditLoggerV2.

### auditLoggerV2

Provides the developer with a way in which to programmatically send error messages to a File, Email, Database and/or Studio console.

The "attributeList" parameter governs how the procedure will deal with messages. An example of declaring a variable to contain the attribute XML entries:

SET attributeList =

-- General options

'<attribute><name>**debug**</name><value>'||NVL(debug,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**loggingType**</name><value>'||NVL(loggingType,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**notificationType**</name><value>'||NVL(notificationType,'null')||'</value></attribute>'||CHR(10)||

-- Database logging options

'<attribute><name>**auditTablePath**</name><value>'||NVL(auditTablePath,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**sequenceNum**</name><value>'||NVL(CAST(sequenceNum AS VARCHAR),'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**organizationName**</name><value>'||NVL(organizationName,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**applicationName**</name><value>'||NVL(applicationName,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**origUserName**</name><value>'||NVL(origUserName,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**resourceName**</name><value>'||NVL(resourceName,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**moduleName**</name><value>'||NVL(moduleName,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**cachekey**</name><value>'||NVL(CAST(cachekey AS VARCHAR),'null')||'</value></attribute>'||CHR(10)||

-- Send email options

'<attribute><name>**emailFrom**</name><value>'||NVL(emailFrom,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**emailReplyTo**</name><value>'||NVL(emailReplyTo,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**emailTo**</name><value>'||NVL(emailTo,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**emailCC**</name><value>'||NVL(emailCC,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**emailBCC**</name><value>'||NVL(emailBCC,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**emailSubject**</name><value>'||NVL(emailSubject,'null')||'</value></attribute>'||CHR(10)||

'<attribute><name>**emailContentType**</name><value>'||NVL(emailContentType,'null')||'</value></attribute>'||CHR(10)||

'';

There are a number of attributes that can be specified:

| **Attribute** | **Description** |
| --- | --- |
| debug | Y=debug or N=do not debug. Debugging within the auditLogger procedure only. |
| loggingType | The type of logging to perform. One or more of this list (comma or space separated):   * LOG - Write to the <CIS\_HOME>/logs/cs\_server.log * EMAIL - Send an email (Email settings must be configured on the CIS instance in the Configuration panel.) * DB - Insert the message into the AUDIT\_LOG table. (created and introspected by developer during initialization) * PRINT - Print to the Studio console tab |
| notificationType | The type of notification that is being logged. One and only one of:   * ERROR - Output error message with severity level ERROR. * AUDIT - Output audit message with severity level INFO. * INFO - Output info message with severity level INFO. * DEBUG - Output debug message with severity level INFO. |
| auditTablePath | The CIS path to the AUDIT\_LOG table. e.g. /shared/Cache\_DB/Cache\_Repo/CACHE1/AUDIT\_LOG |
| sequenceNum | A sequence number (cast as a VARCHAR) used to correlate multiple messages across different log messages. |
| organizationName | The name of the organization which can be used to filter messages. e.g. Mortgage, Operations, CustomerSatisfaction. |
| applicationName | The application name that is using Composite within the organization which can be used as an additional filter. e.g. HomeLoans, Bankruptcy, etc. |
| origUserName | The original user name from the application: format=username@domain. e.g. user1@ldap or user2@composite |
| resourceName | The name of the resource being acted upon such as VIEW\_NAME\_INCR. |
| moduleName | The name of the module or procedure that is invoking the auditLoger which provides context for the code such as RefreshCache. |
| cachekey | The cachekey being used to refresh the cache or 0 if not applicable. This is especially helpful when invoked from incremental cache refresh scripts. |
| emailFrom | The address the message is from. NULL causes use of the server's configured "from" address. Only NULL is supported in this release. |
| emailReplyTo | The address to place in the replyTo field of the message. |
| emailTo | A comma separated list of e-mail addresses |
| emailCC | A comma separated list of e-mail addresses |
| emailBCC | A comma separated list of e-mail addresses |
| emailSubject | The message subject |
| emailContentType | This can be 'TEXT\_PLAIN' or 'TEXT\_HTML' |

**Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | messageText | LONGVARCHAR |
| IN | attributeList | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | messageText | 'This is a test message' |
| IN | attributeList | XML list as described above |

### logDebugMessage

Provides the developer with a built in mechanism to log a message to the CIS log file whereby a single parameter can turn debugging on or off globally. Additionally the log message will contain the CIS module (procedure) where the message was registered. This allows a developer to locate the problem area more quickly.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | moduleName  The name of the invoking procedure | /shared/ASAssets/Utilities/TypeDefinitions.moduleNameType |
| IN | debug  Defines the debugging options for this procedure. Values: Y or T = debugging turned on, N or F = debugging turned off | CHAR(1) |
| IN | messageText  The message to be to be sent or stored | /shared/ASAssets/Utilities/TypeDefinitions.messageType |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | moduleName | Any procedure name |
| IN | debug | Y/N or T/F |
| IN | messageText | Message text |

## LogUtils

This section will show how to use the ‘Log’ CJP procedures.

### LogUtils/GetServerMetadataLog

Parses the local server's metadata change log files into rows and columns. Reads all metadata log files in the server's logs folder, oldest to newest. The result set is therefore naturally sorted in ascending order by change time.

NOTE: By default, CIS will only keep around 100Mb of change metadata logs (older log files will be deleted.) However, the file $CIS\_HOME/conf/server/log4j.properties can be updated to store more data (requires CIS restart.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | CURSOR (  change\_time TIMESTAMP,  cid INTEGER,  domain VARCHAR,  user VARCHAR,  userid INTEGER,  hostname VARCHAR,  operation VARCHAR,  resource\_id INTEGER,  resource\_path VARCHAR,  resource\_type VARCHAR,  message VARCHAR ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | <result too large to display> |

1. How To Use ‘Model’ Views and Procedures

## Introduction

This section will show how to use the Model views and procedures. The Model views are based on TDV 8.3 and above where /services/databases/system/model tables were introduced which provide better access to the TDV catalog. Not all data is resident in these tables so the views in the Utilities model folder provides a consistent schema across different resources types to acquire the information.

Add the following to the XML capabilities file for postgresql\_9\_1

Location: TDV\_HOME/conf/adapters/system/postgresql\_9\_1/postgresql\_9\_1\_values.xml

<?xml version="1.1" encoding="UTF-8"?>

<common:attributes xmlns:common="http://www.compositesw.com/services/system/util/common">

ADD -->

<common:attribute xmlns:common="http://www.compositesw.com/services/system/util/common">

<common:name>/custom/TO\_TIMESTAMP(~number)</common:name>

<common:type>STRING</common:type>

<common:value>TO\_TIMESTAMP($1)</common:value>

<common:configID>TO\_TIMESTAMP(~number)</common:configID>

</common:attribute>

### /Cache/procedures/getCacheProperties

This procedure is used to return the cacheConfigured, cacheEnabled, cacheType and cache properties list for a cached view or procedure. It uses some additional views that ultimately use the TDV model catalog tables

/model/ALL\_TABLES

/model/ALL\_PROCEDURES

/model/ALL\_RESOURCE\_PROPERTIES

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | cacheResourcePath - the full TDV path to the cached resource | VARCHAR(4000) |
| IN | cacheResourceType - TABLE or PROCEDURE | VARCHAR |
| OUT | cacheConfigured - true or false | VARCHAR |
| OUT | cacheEnbled - true or false | VARCHAR |
| OUT | cacheType - SINGLE, MULTITABLE, INCREMENTAL | VARCHAR |
| OUT | cachePropertiesList - Double pipe separated list of properties. name=value||name=value|| etc. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | cacheResourcePath | /shared/project/views/test |
| IN | cacheResourceType | TABLE |
| OUT | cacheConfigured | Y |
| OUT | cacheEnbled | Y |
| OUT | cacheType | MULTITABLE |
| OUT | cachePropertiesList | cacheBucketCatalog=CIS\_CACHE|| cacheBucketMode=AUTO\_GEN|| cacheBucketPrefix=test|| cacheBucketSchema=dbo|| cacheClearRule=MANUAL|| cacheDropCreateIdx=true|| cacheLastTracking=/shared/project/views/DS/cache/dbo/cache\_tracking|| cacheNumBuckets=3|| cachePullBasedIncremental=false|| cacheRefreshMode=MANUAL|| cacheStorageSource=/shared/project/views/DS|| cacheStorageTargets={result=/shared/project/views/DS/CIS\_CACHE/dbo/test0, result1=/shared/project/views/DS/cache/dbo/test1, result2=/shared/project/views/DS/cache/dbo/test2}|| cacheStorageType=DATA\_SOURCE\_OTPS|| |

### /Cache/CACHE\_TABLE

This view is will return a row for each table cache and uses the following TDV catalog model tables.

/model/ALL\_TABLES

/model/ALL\_RESOURCE\_PROPERTIES

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

### /Cache/CACHE\_TABLE\_PROPERTIES

This view is will returns table cache properties and uses the following TDV catalog model tables. A row is returned for each property.

/model/ALL\_TABLES

/model/ALL\_RESOURCE\_PROPERTIES

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

### /Cache/CACHE\_TABLE\_PROPERTIES\_PIVOT

This view will return a row for each table cache properties and uses the following TDV catalog model tables. A row is returned for each property. The difference between this view and CACHE\_TABLE\_PROPERTIES is that this view pivots the properties into columns thus displaying them in one big row. Over time, it is possible for TDV to introduce cache properties that are not displayed in this pivot.

/model/ALL\_TABLES

/model/ALL\_RESOURCE\_PROPERTIES

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

### /Cache/CACHE\_PROCEDURE

This view is will return a row for each procedure cache and uses the following TDV catalog model tables.

/model/ALL\_PROCEDURES

/model/ALL\_RESOURCE\_PROPERTIES

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

### /Cache/CACHE\_PROCEDURE\_PROPERTIES

This view is will returns procedure cache properties and uses the following TDV catalog model tables. A row is returned for each property.

/model/ALL\_PROCEDURES

/model/ALL\_RESOURCE\_PROPERTIES

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

### /Cache/CACHE\_PROCEDURE\_PROPERTIES\_PIVOT

This view will return a row for each procedure cache properties and uses the following TDV catalog model tables. A row is returned for each property. The difference between this view and CACHE\_PROCEDURE\_PROPERTIES is that this view pivots the properties into columns thus displaying them in one big row. Over time, it is possible for TDV to introduce cache properties that are not displayed in this pivot.

/model/ALL\_PROCEDURES

/model/ALL\_RESOURCE\_PROPERTIES

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

### /Resources/procedures/getDataSourceAttributesNormalized

This procedure retrieves the data source attributes into a single row output. This procedure only supports the pMODEL\_ALL… procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | datasourcePath - Full resource path to the datasource | VARCHAR(4000) |
| OUT | Result PIPE | PIPE (  DATASOURCE\_PATH VARCHAR(4000),  DATASOURCE\_TYPE VARCHAR(40),  DATASOURCE\_ADAPTER\_TYPE VARCHAR(50),  DATASOURCE\_HOSTNAME VARCHAR(255),  DATASOURCE\_PORT INTEGER,  DATASOURCE\_DATABASE\_NAME VARCHAR(100),  DATASOURCE\_LOGIN VARCHAR(255),  DATASOURCE\_PASSWORD VARCHAR(255),  DATASOURCE\_URL VARCHAR(500),  DATASOURCE\_PATTERN\_STR VARCHAR(500),  DATASOURCE\_ROOT VARCHAR(500),  DATASOURCE\_CACHE\_STATUS VARCHAR(4000),  DATASOURCE\_CACHE\_TRACKING VARCHAR(4000),  CREATION\_DATE TIMESTAMP,  CREATION\_DATE\_INT BIGINT,  CREATOR\_USER\_NAME VARCHAR(50),  CREATOR\_USER\_DOMAIN VARCHAR(50),  CREATOR\_USER\_ID INTEGER,  LAST\_MODIFIED\_DATE TIMESTAMP,  LAST\_MODIFIED\_DATE\_INT BIGINT,  LAST\_MODIFIED\_USER\_NAME VARCHAR(50),  LAST\_MODIFIED\_USER\_DOMAIN VARCHAR(50),  LAST\_MODIFIED\_USER\_ID INTEGER  ) |

### /Resources/procedures/getResourceAttributesNormalized

This procedure retrieves the resource attributes into a single row output. If the resource is a published LINK it will retrieve the RESOURCE\_LINK\_TYPE which is the resource the LINK is pointing to [TABLE,PROCEDURE]. This procedure only supports the pMODEL\_ALL… procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | VARCHAR(4000) |
| IN | resourceType | VARCHAR |
| OUT | Result PIPE | PIPE (  RESOURCE\_PATH VARCHAR(4000),  PARENT\_PATH VARCHAR(4000),  RESOURCE\_NAME VARCHAR,  RESOURCE\_TYPE VARCHAR(40),  CREATOR\_DATE TIMESTAMP,  CREATOR\_DATE\_INT BIGINT,  CREATOR\_USER\_NAME VARCHAR(50),  CREATOR\_USER\_DOMAIN VARCHAR(50),  CREATOR\_USER\_ID INTEGER,  LAST\_MODIFIED\_DATE TIMESTAMP,  LAST\_MODIFIED\_DATE\_INT BIGINT,  LAST\_MODIFIED\_USER\_NAME VARCHAR(50),  LAST\_MODIFIED\_USER\_DOMAIN VARCHAR(50),  LAST\_MODIFIED\_USER\_ID INTEGER  ) |

### /Resources/procedures/getUsedResourcesDirectCursorNormalized

If the resource is a published LINK it will retrieve the RESOURCE\_LINK\_TYPE which is the resource the LINK is pointing to [TABLE,PROCEDURE]. This procedure only supports the pMODEL\_ALL… procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | VARCHAR(4000) |
| IN | resourceType | VARCHAR |
| OUT | Result PIPE | PIPE (  RESOURCE\_PATH VARCHAR(4000),  RESOURCE\_TYPE VARCHAR(40),  RESOURCE\_LINK\_TYPE VARCHAR(40),  ) |

### /Resources/procedures/tdv\_82\_lower [folder of views]

WARNING: As soon as TDV 8.2 is retired, these views will also be retired.

### /Resources/COLUMN\_PROPERTIES

This view is will return a row for each column and uses the following TDV catalog model tables.

/model/ALL\_COLUMNS

/model/ALL\_TABLES

/model/ALL\_RESOURCE\_PROPERTIES

/system/ALL\_USERS

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

/model/Resources/procedures/**pMODEL\_ALL\_COLUMNS** – a procedure that is used to transition between TDV 8.2 and TDV 8.3 and above (model tables) as well as inject nodehost and nodeport into the output. Invokes /model/Resources/COLUMN\_PROPERTIES dynamically. Used by KPImetrics.

### /Resources/DATASOURCE\_PROPERTIES

This view is will return a row for each datasource and uses the following TDV catalog model tables.

/model/ALL\_DATASOURCES

/model/ALL\_RESOURCE\_PROPERTIES

/system/ALL\_USERS

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

/model/Resources/procedures/**pMODEL\_ALL\_DATASOURCES** – a procedure that is used to transition between TDV 8.2 and TDV 8.3 and above (model tables) as well as inject nodehost and nodeport into the output. Invokes /model/Resources/DATASOURCE\_PROPERTIES dynamically. Used by KPImetrics.

### /Resources/PARAMETER\_PROPERTIES

This view is will return a row for each parameter and uses the following TDV catalog model tables.

/model/ALL\_PARAMETERS

/model/ALL\_RESOURCE\_PROPERTIES

/system/ALL\_USERS

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

/model/Resources/procedures/**pMODEL\_ALL\_PARAMETERS** – a procedure that is used to transition between TDV 8.2 and TDV 8.3 and above (model tables) as well as inject nodehost and nodeport into the output. Invokes /model/Resources/PARAMETER\_PROPERTIES dynamically. Used by KPImetrics.

### /Resources/PROCEDURE\_PROPERTIES

This view is will return a row for each procedure and uses the following TDV catalog model tables and system tables.

/model/ALL\_PROCEDURES

/model/ALL\_RESOURCE\_PROPERTIES

/system/ALL\_USERS

/system/ALL\_RESOURCES [does not push down]

This view does “**not**” push down to postgres due to requirements for columns that don’t exist in the model tables.

/model/Resources/procedures/**pMODEL\_ALL\_PROCEDURES** – a procedure that is used to transition between TDV 8.2 and TDV 8.3 and above (model tables) as well as inject nodehost and nodeport into the output. Invokes /model/Resources/PROCEDURE\_PROPERTIES dynamically. Used by KPImetrics.

### /Resources/RESOURCE\_OTHER\_PROPERTIES

This view is will return a row for each “resource other” and uses the following TDV catalog model tables and system tables.

/system/ALL\_RESOURCES [does not push down]

/system/ALL\_USERS

/model/ALL\_RESOURCE\_PROPERTIES

This view does “**not**” push down to postgres due to requirements for columns that don’t exist in the model tables.

/model/Resources/procedures/**pMODEL\_ALL\_RESOURCES\_OTHER** – a procedure that is used to transition between TDV 8.2 and TDV 8.3 and above (model tables) as well as inject nodehost and nodeport into the output. Invokes /model/Resources/RESOURCE\_OTHER\_PROPERTIES dynamically. Used by KPImetrics.

### /Resources/TABLE\_PROPERTIES

This view is will return a row for each table and uses the following TDV catalog model tables.

/model/ALL\_TABLES

/model/ALL\_RESOURCE\_PROPERTIES

/system/ALL\_USERS

This view pushes down to postgres as long as the XML capabilities file is properly modified as instructed earlier.

/model/Resources/procedures/**pMODEL\_ALL\_TABLES** – a procedure that is used to transition between TDV 8.2 and TDV 8.3 and above (model tables) as well as inject nodehost and nodeport into the output. Invokes /model/Resources/TABLE\_PROPERTIES dynamically. Used by KPImetrics.

### /Resources/WSDL\_OPERATIONS\_PROPERTIES

This view is will return a row for each published “WSDL operation” and uses the following TDV catalog model tables and system tables.

/system/ALL\_WSDL\_OPERATIONS [does not push down]

/system/ALL\_USERS

/shared/ASAssets/Utilities/model/Resources/procedures/getUsedResourcesDirectCursorNormalized(A.RESOURCE\_PATH, A.RESOURCE\_TYPE) [does not push down]

/model/ALL\_RESOURCE\_PROPERTIES

This view does “**not**” push down to postgres due to requirements for columns that don’t exist in the model tables.

/model/Resources/procedures/**pMODEL\_ALL\_WSDL\_OPERATIONS** – a procedure that is used to transition between TDV 8.2 and TDV 8.3 and above (model tables) as well as inject nodehost and nodeport into the output. Invokes /model/Resources/WSDL\_OPERATION\_PROPERTIES dynamically. Used by KPImetrics.

1. How To Use ‘Modify’ Procedures

## Introduction

This section will show how to use the ‘Modify procedures.

### controllerTemplate

This procedure serves as a template to be copied and modified for a specific use case that involves procedure code modification. This is a highly evolved search and replace capability that can add or replace code in a TDV procedure. It does not perform delete but you can use the replace capability to search for text and replace it with empty string. It must be an exact search.

The codeInsertVector is used to setup the code modification. Code modification can be done based on searching and replacing text or standard procedure code blocks.

\*\*\*\* CAUTION \*\*\*\*

Take a backup of the code before executing this procedure. There is no going back without a backup.

The following search and replace capabilities are provided:

when location=REPLACE, REPLACE\_BEFORE or REPLACE\_AFTER then codeSection can be any free-form code text. Use CHR(9) for tab. Use CHR(10) for linefeed.

* REPLACE: Replace all occurrences of the existing code found exactly as the lineToAdd with the code in codeSection.
* REPLACE\_BEFORE: Replace the existing code found before the lineToAdd with the code in codeSection.
* REPLACE\_AFTER: Replace the existing code found after the lineToAdd with the code in codeSection.

when location=BEFORE or AFTER, then the code section may be a standard block.

valid blocks= |PROCEDURE\_PARAM|PROCEDURE\_BEGIN|PROCEDURE\_END|BLOCK\_DECLARE|PROCEDURE\_EXCEPTION|PROCEDURE\_ELSE|BLOCK\_BEGIN|BLOCK\_EXCEPTION|BLOCK\_ELSE|BLOCK\_END|

* PROCEDURE\_PARAM - add text BEFORE or AFTER the procedure parameters (PARAM) block.
* PROCEDURE\_BEGIN - replace text BEFORE or AFTER the procedure BEGIN block.
* PROCEDURE\_END - replace text BEFORE or AFTER the procedure END block.
* BLOCK\_DECLARE - replace text BEFORE or AFTER the procedure DECLARE block.
* PROCEDURE\_EXCEPTION - replace text BEFORE or AFTER the procedure EXCEPTION block which occurs at the bottom of the procedure before the END bock.
* PROCEDURE\_ELSE - replace text BEFORE or AFTER the procedure ELSE block which occurs at the bottom of the procedure after the procedure EXCEPTION block and before the END bock.
* BLOCK\_BEGIN - replace text BEFORE or AFTER the procedure BEGIN block
* BLOCK\_EXCEPTION - replace text BEFORE or AFTER the procedure BEGIN EXCEPTION block
* BLOCK\_ELSE - replace text BEFORE or AFTER the procedure BEGIN ELSE block
* BLOCK\_END - replace text BEFORE or AFTER the procedure BEGIN END block

Helper Scripts Invoked by controllerTemplate:

* /shared/ASAssets/Utilities/modify/helpers/controllerScript
* /shared/ASAssets/Utilities/modify/helpers/parseSqlScriptText
* /shared/ASAssets/Utilities/modify/helpers/getSqlScriptList
* /shared/ASAssets/Utilities/modify/helpers/formatSqlScript
* /shared/ASAssets/Utilities/modify/helpers/typeDefinitions

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug1 – Y=debug on. N=debug off. 1st level debug for controllerScript only. | CHAR(1) |
| IN | debug2 – Y=debug on. N=debug off. 2nd level debug passed into getSqlScript and parseSqlScript. | CHAR(1) |
| IN | debug3 – Y=debug on. N=debug off. 3rd level debug for controllerScript do display final SQL Script. | CHAR(1) |
| IN | performUpdate – Y=debug on. N=debug off. | CHAR(1) |
| IN | startingPath - path to container/folder or a specific procedure path | VARCHAR(4000) |
| OUT | status - SUCCESS or FAIL | VARCHAR |
| OUT | numResourcesUpdated - the number of resources that were updated. | INTEGER |
| OUT | errorMessage - null if no error otherwise contains an error message when there is an error. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug1 | Y |
| IN | debug2 | Y |
| IN | performUpdate | Y |
| IN | debug1 | Y |
| IN | startingPath | /shared/ASAssets/Utilities/modify/example/test |
| OUT | status | SUCCESS |
| OUT | numResourcesUpdated | 3 |
| OUT | errorMessage | null |

1. How To Use ‘Monitor Procedures

## Introduction

This section will show how to use the ‘Monitor’ procedures.

### getSysCluster

This procedure returns the results of the current host, port and system.SYS\_CLUSTER. It returns no rows if there is no cluster defined.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | CURSOR (  CURRENT\_HOST VARCHAR(255),  CURRENT\_PORT INTEGER,  SERVER\_ID VARCHAR(255),  SERVER\_NAME VARCHAR(255),  SERVER\_HOST VARCHAR(255),  SERVER\_PORT INTEGER,  STATUS VARCHAR(24),  TIMEKEEPER CHAR(1),  CONNECT\_TIME TIMESTAMP,  LAST\_CONTACT TIMESTAMP,  IDLE\_DURATION BIGINT,  NUM\_CONNECTS INTEGER,  BYTES\_TO BIGINT,  BYTES\_FROM BIGINT,  MESSAGE VARCHAR(65535)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | |  |  | | --- | --- | | CURRENT\_HOST: | HOST-X1E2 | | CURRENT\_PORT: | 9830 | | SERVER\_ID: | HOST-X1E2.na.tibco.com-9830-1600398585 | | SERVER\_NAME: | HOST-X1E2.na.tibco.com-9830-1600398585 | | SERVER\_HOST: | HOST-X1E2 | | SERVER\_PORT: | 9830 | | STATUS: | OPERATIONAL [5607] | | TIMEKEEPER: | Y | | CONNECT\_TIME: | [NULL] | | LAST\_CONTACT: | [NULL] | | IDLE\_DURATION: | 0 | | NUM\_CONNECTS: | 0 | | BYTES\_TO: | 0 | | BYTES\_FROM: | 0 | | MESSAGE: |  | |

### getSysRequests

This procedure returns the results of the current host, port and system.SYS\_REQUESTS.

Status of the query

SUCCESS, FAILED, RUNNING, TERMINATED, TOP\_MEMORY, TOP\_TIME

Request type

SQL, SQL Script, Java Procedure (built-in)

Memory and disk columns are measured in bytes.

1 k = 1024

1 MB = 1048576

1 GB = 1073741824

Duration columns are measured in milliseconds.

1 sec = 1000 ms

1 min = 60000 ms

1 hour = 3600000 ms

1 day = 86400000 ms

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** | |
| --- | --- | --- | --- |
| OUT | result | CURSOR (  CURRENT\_HOST VARCHAR(255),  CURRENT\_PORT INTEGER,  DOMAIN\_ID INTEGER,  DOMAIN\_NAME VARCHAR(255),  REQUEST\_ID BIGINT,  PARENT\_ID BIGINT,  SESSION\_ID BIGINT,  TRANSACTION\_ID BIGINT,  OWNER\_ID INTEGER,  OWNER VARCHAR(255),  REQUEST\_TYPE VARCHAR(255),  STATUS VARCHAR(20),  DESCRIPTION VARCHAR(65535),  START\_TIME TIMESTAMP,  END\_TIME TIMESTAMP,  TOTAL\_DURATION BIGINT,  SERVER\_DURATION BIGINT,  ROWS\_AFFECTED BIGINT,  MAX\_MEMORY BIGINT,  MAX\_DISK BIGINT,  CURRENT\_MEMORY BIGINT,  CURRENT\_DISK BIGINT,  MESSAGE VARCHAR(65535),  MAX\_USED\_MEMORY BIGINT,  CURRENT\_USED\_MEMORY BIGINT,  PROCEDURE\_ID INTEGER  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | | **Parameter Value** |
| --- | --- | --- | --- |
| OUT | | result | |  |  | | --- | --- | | CURRENT\_HOST: | HOST-X1E2 | | CURRENT\_PORT: | 9830 | | DOMAIN\_ID: | 11 | | DOMAIN\_NAME: | composite | | REQUEST\_ID: | 7204309 | | PARENT\_ID: | 7204308 | | SESSION\_ID: | 2602023 | | TRANSACTION\_ID: | 2506001 | | OWNER\_ID: | -1973 | | OWNER: | admin | | REQUEST\_TYPE: | SQL Script | | STATUS: | SUCCESS | | DESCRIPTION: | CALL /shared/ASAssets/BestPractices\_v81/DynamicFileFramework/constants() | | START\_TIME: | 2020-12-03 23:10:00.045 | | END\_TIME: | 2020-12-03 23:10:00.045 | | TOTAL\_DURATION: | 0 | | SERVER\_DURATION: | 0 | | ROWS\_AFFECTED: | -1 | | MAX\_MEMORY: | 2000000 | | MAX\_DISK: | 0 | | CURRENT\_MEMORY: | 0 | | CURRENT\_DISK: | 0 | | MESSAGE: | [NULL] | | MAX\_USED\_MEMORY: | 524288 | | CURRENT\_USED\_MEMORY: | 0 | | PROCEDURE\_ID: | 21572 | |

1. How To Use ‘Net’ Procedures

## Introduction

This section will show how to use the ‘Net’ procedures.

## NetUtils

This section will show how to use the ‘Net’ CJP procedures.

### ftpFile

Connects to an FTP server and retrieves a file and places it into a folder on the CIS host filesystem.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fileName | LONGVARCHAR |
| IN | hostIp | LONGVARCHAR |
| IN | userId | LONGVARCHAR |
| IN | userPass | LONGVARCHAR |
| IN | ftpDirName | LONGVARCHAR |
| IN | dirName | LONGVARCHAR |
| OUT | success | BOOLEAN |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fileName | 'test.xml' |
| IN | hostIp | 'ftp.company.com' |
| IN | userId | 'anonymous' |
| IN | userPass | 'cgoodric@company.com' |
| IN | ftpDirName | 'pub' |
| IN | dirName | 'C:\Users\cgoodric\Desktop' |
| OUT | success | true |

1. How To Use ‘PDTool’ Procedures

## Introduction

This section describes the routines using the “PDTool” procedures.

### generatePDToolDeployableResourcePlanByDate

This procedure forms the basis by which to generate PDTool incremental deployment plans based on a resource date. It provides the developers the control to create a deployment plan for only those resources that have changed based resources >= the passed in resource date. This procedure is only valid for CIS 6.2.2 and higher due to the fact that this release introduced a "lastModifiedDate" and "creationDate" in the metadata. Prior to this release their no date values to compare with.

This procedure will automatically insert "createFolder" actions to insure that the necessary folder path will exist when the resource is deployed from VCS. If the folder does not exist on the target server, an error would be thrown.

Any resource paths found in the exclude paths list "excludePathsList" will be excluded from the result. Any resources that are not inclusive of the include base paths "includeBasePath" will be excluded from the result.

PDTool (Promotion and Deployment Tool) is used to execute generated plan files.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | planFilePath | VARCHAR(1024) |
| IN | bufferSize | INTEGER |
| IN | PDToolPlanTemplateVCS | LONGVARCHAR |
| IN | PDToolPlanTemplateFolders | LONGVARCHAR |
| IN | resourcePathList | LONGVARCHAR |
| IN | resourceTimestamp | TIMESTAMP |
| IN | includeBasePath | LONGVARCHAR |
| IN | excludePathsList | LONGVARCHAR |
| IN | debug | CHAR(1) |
| OUT | success | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | planFilePath | 'D:/PDTool62/resources/plans/examples.dp' |
| IN | bufferSize | 100 |
| IN | PDToolPlanTemplateVCS | 'PASS TRUE ExecuteAction vcsCheckout2 $SERVERID $VCONN "$RESOURCE\_PATH" "$RESOURCE\_TYPE" HEAD "$MODULE\_HOME/VCSModule.xml" "$MODULE\_HOME/servers.xml"' |
| IN | PDToolPlanTemplateFolders | 'PASS TRUE ExecuteAction createFolder $SERVERID "$RESOURCE\_PATH" "$MODULE\_HOME/servers.xml" true' |
| IN | resourcePathList | '/shared/examples, /services/databases/examples' |
| IN | resourceTimestamp | '2013-08-16 00:00:00' |
| IN | includeBasePath | NULL |
| IN | excludePathsList | NULL |
| IN | Debug | 0 |
| OUT | Success | 1 (plan file is written to the path specified by planFilePath.) |

### generatePDToolDeployableResourcePlanByLineage

This procedure forms the basis by which to generate PDTool incremental deployment plans based on published resources. It provides the developers the control to create a deployment plan for only those resources that have changed based on a starting published resource. A published resource is any resource found in "/services/databases" or "/services/webservices".

This procedure will automatically insert "createFolder" actions to insure that the necessary folder path will exist when the resource is deployed from VCS. If the folder does not exist on the target server, an error would be thrown.

Any resource paths found in the exclude paths list "excludePathsList" will be excluded from the result. Any resources that are not inclusive of the include base paths "includeBasePath" will be excluded from the result.

PDTool (Promotion and Deployment Tool) is used to execute generated plan files.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | planFilePath | VARCHAR(1024) |
| IN | bufferSize | INTEGER |
| IN | PDToolPlanTemplateVCS | LONGVARCHAR |
| IN | PDToolPlanTemplateFolders | LONGVARCHAR |
| IN | resourcePathList | LONGVARCHAR |
| IN | includeDependentTriggers | INTEGER |
| IN | includeBasePath | LONGVARCHAR |
| IN | excludePathsList | LONGVARCHAR |
| IN | inIgnoreResourceDoesNotExist | INTEGER |
| IN | debug | CHAR(1) |
| OUT | success | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | planFilePath | 'C:\PDTool62\resources\plans' |
| IN | bufferSize | 100 |
| IN | PDToolPlanTemplateVCS | 'PASS TRUE ExecuteAction vcsCheckout2 $SERVERID $VCONN "$RESOURCE\_PATH" "$RESOURCE\_TYPE" HEAD "$MODULE\_HOME/VCSModule.xml" "$MODULE\_HOME/servers.xml"' |
| IN | PDToolPlanTemplateFolders | 'PASS TRUE ExecuteAction createFolder $SERVERID "$RESOURCE\_PATH" "$MODULE\_HOME/servers.xml" true' |
| IN | resourcePathList | '/shared/examples, /services/databases/examples' |
| IN | includeDependentTriggers | 1 |
| IN | includeBasePath | NULL |
| IN | excludePathsList | NULL |
| IN | inIgnoreResourceDoesNotExist | 1 |
| IN | debug | 0 |
| OUT | success | 1 (plan file is written to the path specified by planFilePath.) |

### template\_generatePDToolDeployableResourcePlan

This procedure provides a template for invoking generatePDToolDeployableResourcePlanByLineage or generatePDToolDeployableResourcePlanByDate. This procedure should be copied to a working folder within the project and modified by the user for developers to use with ease.

Note that when using "generatePDToolDeployableResourcePlanByDate" to create a deployment plan based on a resource date can only be used with CIS 6.2.2 and higher. CIS 6.2.2 and higher introduced a "lastModifiedDate" and "creationDate" in the metadata. Prior to this release there are no date values to compare with.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | planOption | INTEGER |
| OUT | success | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | planOption | 1 |
| OUT | Success | 1 |

## helpers

This section describes the auxiliary procedures for the PDTool utilities.

### helpers/getDeployableResourceListByDate

This procedure provides the logic for retrieving a complete list of paths based on the creation date or last modified date. If either the creation date or the last modified date is not null and is greater than the passed in resource timestamp then it appears in the result. Any resource paths found in the exclude paths list "excludePathsList" will be excluded from the result. Any resources that are not inclusive of the include base paths "includeBasePath" will be excluded from the result.

PDTool (Promotion and Deployment Tool) is used to execute generated plan files.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePathList | LONGVARCHAR |
| IN | resourceTimestamp | TIMESTAMP |
| IN | includeBasePath | LONGVARCHAR |
| IN | excludePathsList | LONGVARCHAR |
| IN | debug | INTEGER |
| OUT | resourceTreeList | CURSOR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePathList | '/shared/examples' |
| IN | resourceTimestamp | '2013-08-16 00:00:00' |
| IN | includeBasePath | NULL |
| IN | excludePathsList | NULL |
| IN | debug | 0 |
| OUT | resourceTreeList | <row set> |

### helpers/getDeployableResourceListByLineage

This procedure provides the logic for retrieving a complete list of paths based on the lineage of all the resources specified by the input parameter "resourcePathList". Any resource paths found in the exclude paths list "excludePathsList" will be excluded from the result. Any resources that are not inclusive of the include base paths "includeBasePath" will be excluded from the result.

PDTool (Promotion and Deployment Tool) is used to execute generated plan files.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePathList | LONGVARCHAR |
| IN | includeDependentTriggers | INTEGER |
| IN | includeBasePath | LONGVARCHAR |
| IN | excludePathsList | LONGVARCHAR |
| IN | debug | INTEGER |
| IN | inIgnoreResourceDoesNotExist | INTEGER |
| OUT | resourceTreeList | CURSOR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePathList | '/shared/examples' |
| IN | includeDependentTriggers | 1 |
| IN | includeBasePath | NULL |
| IN | excludePathsList | NULL |
| IN | debug | 0 |
| IN | inIgnoreResourceDoesNotExist | 1 |
| OUT | resourceTreeList | <row set> |

### helpers/getDistinctDeployableResourceListByDate

This procedure provides the logic for retrieving a distinct list of paths. A resource may be shared with other resources but it only needs to be deployed once. This retrieves deployable resources based on the creation date or last modified date. If either the creation date or the last modified date is not null and is greater than the passed in resource timestamp then it appears in the result. Any resource paths found in the exclude paths list "excludePathsList" will be excluded from the result. Any resources that are not inclusive of the include base paths "includeBasePath" will be excluded from the result.

PDTool (Promotion and Deployment Tool) is used to execute generated plan files.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePathList | LONGVARCHAR |
| IN | resourceTimestamp | TIMESTAMP |
| IN | includeBasePath | LONGVARCHAR |
| IN | excludePathsList | LONGVARCHAR |
| IN | debug | INTEGER |
| OUT | resourceTreeList | CURSOR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePathList | '/shared/examples' |
| IN | resourceTimestamp | '2013-08-16 00:00:00' |
| IN | includeBasePath | NULL |
| IN | excludePathsList | NULL |
| IN | debug | 0 |
| OUT | resourceTreeList | <row set> |

### helpers/getDistinctDeployableResourceListByLineage

This procedure provides the logic for retrieving a distinct list of paths. A resource may be shared with other resources but it only needs to be deployed once. This retrieves deployable resources based on lineage of the resource paths. If include triggers is set then any triggers associated associated with a resource are also returned. Any resource paths found in the exclude paths list "excludePathsList" will be excluded from the result. Any resources that are not inclusive of the include base paths "includeBasePath" will be excluded from the result.

PDTool (Promotion and Deployment Tool) is used to execute generated plan files.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePathList | LONGVARCHAR |
| IN | includeDependentTriggers | INTEGER |
| IN | includeBasePath | LONGVARCHAR |
| IN | excludePathsList | LONGVARCHAR |
| IN | inIgnoreResourceDoesNotExist | INTEGER |
| IN | debug | INTEGER |
| OUT | resourceTreeList | CURSOR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePathList | '/shared/examples' |
| IN | includeDependentTriggers | 1 |
| IN | includeBasePath | NULL |
| IN | excludePathsList | NULL |
| IN | inIgnoreResourceDoesNotExist | 1 |
| IN | debug | 0 |
| OUT | resourceTreeList | <row set> |

1. How To Use ‘Repository’ Procedures

## Introduction

This section will show how to use the ‘CIS Repository’ API procedures. These numerous helper procedures are built around the CIS repository API to provide a higher-level of abstraction from the raw API’s. This helps to increase developer productivity.

## CIS Types and Subtypes listing

This section provides a list of CIS repository resource types and subtypes. Most of these can also be found as constants in the /lib/resource/ResourceDefs SQL definition set. These constants’ names all start with either “RESOURCE\_TYPE\_” or “RESOURCE\_SUBTYPE\_”.

### Listing of CIS resource types and subtypes

The following resource types/subtypes are supported by several repository helper procedures. The format below is RESOURCE TYPE / SUB TYPE – Description.

(Basic CIS folder)

* CONTAINER / FOLDER\_CONTAINER - A Composite folder. Cannot be created anywhere under /services except in another FOLDER under /services/webservices.

(Database)

* CONTAINER / CATALOG\_CONTAINER - A Composite catalog folder under a data source. Can only be created within a data source.
* CONTAINER / SCHEMA\_CONTAINER - A Composite schema container. Can only be created within a CATALOG\_CONTAINER or data source.

(Web Services)

* CONTAINER / SERVICE\_CONTAINER - A web service container for the service. Can only be created within a Composite Web Services data source that is under /services/webservices.
* CONTAINER / OPERATIONS\_CONTAINER - A web service container for the operations
* CONTAINER / PORT\_CONTAINER - A Composite web service container for port. Can only be created within a SERVICE under /services/webservices.

(Misc Containers)

* CONTAINER / CONNECTOR\_CONTAINER - A Composite container for connectors.
* CONTAINER / DIRECTORY\_CONTAINER - A folder within an LDAP data source.

(Connectors)

* CONNECTOR / JMS - A Composite JMS Connector. Created with no connection information
* CONNECTOR / HTTP - A Composite HTTP Connector. Created with no connection information

(Data Sources)

* DATA\_SOURCE / RELATIONAL\_DATA\_SOURCE - A relational database source.
* DATA\_SOURCE / FILE\_DATA\_SOURCE - A comma separate file data source.
* DATA\_SOURCE / XML\_FILE\_DATA\_SOURCE - An XML file data source.
* DATA\_SOURCE / COMPOSITE\_WEB\_SERVICE - A published web service.
* DATA\_SOURCE / WSDL\_DATA\_SOURCE - A Composite web service data source or published legacy web service.
* DATA\_SOURCE / XML\_HTTP\_DATA\_SOURCE - A simple XML over HTTP data source.
* DATA\_SOURCE / REST\_DATA\_SOURCE - A REST data source.
* DATA\_SOURCE / NONE - A custom java procedure data source.

(Definition Sets)

* DEFINITION\_SET / SQL\_DEFINITION\_SET - A Composite SQL Definition set.
* DEFINITION\_SET / XML\_SCHEMA\_DEFINITION\_SET - A Composite XML Schema Definition set.
* DEFINITION\_SET / WSDL\_DEFINITION\_SET - A Composite WSDL Definition set.
* DEFINITION\_SET / ABSTRACT\_WSDL\_DEFINITION\_SET - A Composite Abstract WSDL Definition set such as the ones imported from Designer.
* DEFINITION\_SET / SCDL\_DEFINITION\_SET - A Composite SCA composite Definition set imported from Designer.

(Published Resources)

* LINK / NONE – A resource published in /services.

(Model Resources)

* MODEL / NONE – A Discovery model.

(Policies)

* POLICY / CACHE\_POLICY - A Composite cache refresh policy. Created disabled.
* POLICY / NONE - A custom web services security policy.

(CIS procedures)

* PROCEDURE / SQL\_SCRIPT\_PROCEDURE - A Composite SQL Procedure. Created with a simple default script body that is runnable.

(Custom java procedures)

* PROCEDURE / JAVA\_PROCEDURE - A Composite java data source procedure. Created from a java data source (jar file).

(Database procedures)

* PROCEDURE / EXTERNAL\_SQL\_PROCEDURE - A Composite Packaged Query. Created with no SQL text, so it is not runnable.
* PROCEDURE / DATABASE\_PROCEDURE - A database stored procedure.
* PROCEDURE / NATIVE\_FUNCTION - A database function.

(XML procedures)

* PROCEDURE / BASIC\_TRANSFORM\_PROCEDURE - A Composite Basic XSLT Transformation procedure. Created with no target procedure and no output columns, so it is not runnable.
* PROCEDURE / TRANSFORM\_PROCEDURE - A Composite Transformation Editor (Any to Any) procedure. Created with no target model or output definition so is not runnable.
* PROCEDURE / XSLT\_ PROCEDURE - A Composite XSLT procedure. Created with no target procedure, so it is not runnable.
* PROCEDURE / XSLT\_TRANSFORM\_PROCEDURE - A Composite XSLT Transformation procedure. Created with no target procedure and no output columns, so it is not runnable.
* PROCEDURE / STREAM\_TRANSFORM\_PROCEDURE - A Composite XSLT Streaming Transformation procedure. Created with no target procedure and no output columns, so it is not runnable.
* PROCEDURE / XQUERY\_ PROCEDURE - A Composite XQUERY procedure. Created with no target procedure, so it is not runnable.
* PROCEDURE / XQUERY\_TRANSFORM\_PROCEDURE - A Composite XQUERY Transformation Procedure. Created with no target schema and no model, so it is not runnable.

(Misc procedures)

* PROCEDURE / OPERATION\_PROCEDURE - A Composite web service or HTTP procedure operation.

(Relationship Resources)

* RELATIONSHIP / NONE – A Discovery relationship.

(Tables or Views)

* TABLE / SQL\_TABLE - A Composite View. Created with no SQL text or model, so it is not runnable.
* TABLE / DATABASE\_TABLE - A data source table or view.
* TABLE / DELIMITED\_FILE\_TABLE - A delimited file data source "table".
* TABLE / EXCEL\_NON\_ODBC\_POI - An Excel (non-ODBC) worksheet "table".
* TABLE / SYSTEM\_TABLE - A Composite system table view. Cannot be created or modified.

(XML Structures)

* TREE / XML\_FILE\_TREE - The XML tree structure associated with a file-XML data source.

(Triggers)

* TRIGGER / NONE - A Composite trigger. Created disabled.

## The ACCESS\_TOOLS Right

Because the Repository Helper procedures make use of the published Admin web services API (see /services/webservices/system/admin), users that need to make use of the Repository Helper procedures will need to have the ACCESS\_TOOLS right. A security exception will be thrown otherwise.

## Note On Using Repository Helper Procedures With Triggers and Cache Procedures

CIS does not by default support **updating** the repository with a task session (i.e. a triggered procedure or pre/post cache procedure.) Any task session that attempts to update the repository (create a resource, update a resource’s attributes, move a resource, etc.) will cause an exception to be thrown. Triggered processes should be able to use Repository Helper procedures to **read** from the repository without any problems (assuming the task session's owner has the ACCESS\_TOOLS right as mentioned above.)

In 6.2, the ability to update the repository from a task session was supported as a configuration setting that needs to be explicitly enabled. See Server -> Configuration -> Security -> Enable Task Session API Access in the Studio Configuration panel.

## CIS Repository Helper Procedures

This section describes each ‘CIS Repository’ API helper procedure. It provides the name of the procedure and a description of what it does. It provides a chart of the parameters that define the direction of the parameter, the parameter name and the type. Finally, a chart with sample parameter is data is provided to give the user an idea of what is expected.

### applyReservedListToPath (Custom Function)

This procedure has been deprecated. See RepoUtils/applyReservedListToPath below.

### applyReservedListToWord (Custom Function)

This procedure has been deprecated. See RepoUtils/applyReservedListToWord below.

### configureReservedList

This procedure has been deprecated. It has been functionally replaced with the properties file found on the CIS host filesystem at $CIS\_HOME/conf/customjars/RepoUtils.properties.

### cachedResources

This procedure is used to manipulate cached resources within a starting folder. This procedure can retrieve, enable, or disable cached resources within a designated folder. It operates recursively.

Input:

**operation** - R=retrieve, E=enable caches, D=disable caches. The operation acts upon all resources found in the path where caching is configured and the includePathList\_ and excludePathList\_ filters are applied.

**startingPath** - The path to recursively start searching

**includePathList** - A comma separated list of paths or partial paths to include as filters (only execute on these paths). A partial path only has to be present anywhere within the path, not just the beginning of the path.

e.g. startingPath=/shared/project, includePathList=/F2

Searched paths would include:

- /shared/project/F1/F2

- /shared/project/F2/F2

- /shared/project/F3/F2

but not:

- /shared/project/F4/F1

**excludePathList** - A comma separated list of paths or partial paths to exclude from the list (do not execute on these paths). This works in a similar manner to includePathList.

**debug** - Y=debug is on, N=do not debug

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | operation | VARCHAR(255) |
| IN | startingPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | includePathList | LONGVARCHAR |
| IN | excludePathList | LONGVARCHAR |
| IN | debug | CHAR(1) |
| OUT | result | CURSOR (  operation VARCHAR(255)  prevStatus VARCHAR(255)  currStatus VARCHAR(255)  resourceType ResourceType  resourcePath ResourcePath ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | operation | ‘R’ |
| IN | startingPath | ‘/shared/examples’ |
| IN | includePathList | NULL |
| IN | excludePathList | NULL |
| IN | debug | ‘N’ |
| OUT | result | (  ‘R’,  ‘ENABLED’,  ‘ENABLED’,  ‘TABLE’,  ‘/shared/examples/ds\_orders/orders’ ) |

### changePassword

If published, this procedure allows a “composite” domain user logged in from an external client to change his/her password programmatically. If a non-composite domain user attempts to use this procedure or if the newPassword and confirmNewPassword arguments don’t match, an IllegalArgumentException will be thrown.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | oldPassword | VARCHAR(255) |
| IN | newPassword | VARCHAR(255) |
| IN | confirmNewPassword | VARCHAR(255) |
| OUT | result | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | oldPassword | ‘0ld p4ssw0rd’ |
| IN | newPassword | ‘n3w p4ssw0rd’ |
| IN | confirmNewPassword | ‘n3w p4ssw0rd’ |
| OUT | result | ‘Password successfully updated.’ |

### changeResourceOwner

This procedure is used to change the resource ownership for a given resource path. An exception “ex” is thrown if the internal faultXML response is NOT NULL. The value of faultXML is thrown with the exception “ex”.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | recurseChildren - 1=recurse all children, 0=do not recurse. only set resource path. | BIT |
| IN | resourcePath - The resource path. This may be a CONTAINER or DATA\_SOURCE path where ownership is pushed recursively to the children. | VARCHAR(4096) |
| IN | resourceType - The resource path type: CONTAINER, DATA\_SOURCE, TABLE, PROCEDURE, LINK, etc. | VARCHAR(255) |
| IN | newOwner - The DV owner name. | VARCHAR(255) |
| IN | newOwnerDomain | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | recurseChildren | 1 |
| IN | resourcePath | ‘/shared/ASAssets/Utilities’ |
| IN | resourceType | ‘CONTAINER’ |
| IN | newOwner | ‘admin’ |
| IN | newOwnerDomain | ‘composite’ |

### clearIntrospectableResourceIdCache

This procedure clears the cache of introspectable resources from a data source. This will force a re-scan of available resources when getIntrospectableResourceIdsTask() is called. It will throw an exception if the faultXML is not null.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dataSourcePath - The path to the data source. | /lib/resource/ResourceDefs.ResourcePath |
| OUT | success - An integer indicating whether the call was successful or not. Values: 1 = success, 0 = failure | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dataSourcePath | ‘/shared/ASAssets/Utilities/examples/repository/source/ds\_orders’ |
| OUT | success | 1 |

### compareCisVersions (Custom Function)

This method compares two CIS version (baseline and current). It converts the version string to an integer and performs a comparison. The following is returned based on the comparison:

-1 - if the current version is less than the baseline version.

0 - if the current version is equal to the baseline version.

1 - if the current version is greater than the baseline version.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | baseCisVersion | VARCHAR |
| IN | currentCisVersion | VARCHAR |
| OUT | status | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | baseCisVersion | '6.1.0.01.09' |
| IN | currentCisVersion | '6.1.0.01.14' |
| OUT | status | 1 |

### copyResource

This is a procedure is used to copy a single resource from the source to a target.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceResource - DV source resource path. Resource must exist. | VARCHAR(4000) |
| IN | sourceType - the type of resource. | VARCHAR |
| IN | targetResource - DV target resource path. Gets created if it does not exist. | VARCHAR(4000) |
| IN | copyMode - The "copyMode" element is used to control behavior in the case where a resource exists with the same name and type in the container specified by "newPath". The options are:  "ALTER\_NAME\_IF\_EXISTS" - If a resource of the same name and type of the source resource already exists in the target container, then avoid conflicts by automatically generating a new name. Names are generated by appending a number to the end of the provided name.  "FAIL\_IF\_EXISTS" - Fails if a resource of the same name and type already exists in the target container. The resource will not be copied if this occurs.  "OVERWRITE\_MERGE\_IF\_EXISTS" - If a resource of the same name and type of the source resource already exists in the target container, then overwrite the resource in the target container. If the source resource is a container, then merge the contents of the source container with the corresponding resource in the target. All resources in the source container will overwrite those resources with the same name in the target, but child resources in the target with different names will not be overwritten and remain unaltered.  "OVERWRITE\_REPLACE\_IF\_EXISTS" - If a resource of the same name and type of the source resource already exists in the target container, then overwrite the resource in the target container. If the source resource is a container, then replace the container within the target container with the source container. This is equivalent to deleting the container in the target before copying the source. | VARCHAR |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceResource | /shared/examples/CompositeView |
| IN | sourceType | TABLE |
| IN | targetResource | /shared/examples/CompositeView2 |
| IN | copyMode | OVERWRITE\_REPLACE\_IF\_EXISTS |
| OUT | success | 1 |
| OUT | faultResponse | null |

### copyResources

This procedure is used to copy all of the CIS resources from a source folder to a target folder. If the target folder does not exist, then it is created. An exception is thrown if the source does not exist.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceFolderPath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | targetFolderPath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceFolderPath | ‘/shared/ASAssets/Utilities/repository/examples/source’ |
| IN | targetFolderPath | ‘/shared/ASAssets/Utilities/repository/examples/target’ |
| OUT | success | 1 |
| OUT | faultResponse | null |

### copyResourceAnnotations

This procedure is used to copy all of the annotations of one resource to another. If both resources are of type "TABLE", the column annotations are copied as well (where the column names are the same, ignoring case.) This procedure is NOT recursive when resources of type "CONTAINER" are specified as input. There are a couple of Utilities that can walk a resource or dependency tree and can be used in conjunction with this procedure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inSourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | inSourceType | /lib/resource/ResourceDefs.ResourceType |
| IN | inDestPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | inDestType | /lib/resource/ResourceDefs.ResourceType |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inSourcePath | '/shared/examples/ds\_orders/orders' |
| IN | inSourceType | 'TABLE' |
| IN | inDestPath | '/shared/examples/CompositeView' |
| IN | inDestType | 'TABLE' |

### copyResourcesPrivileges (deprecated)

Use copyResourcePrivilegesV2.

### copyResourcesPrivilegesV2

This is a procedure is used to copy resource privileges from one resource to another.

This procedure enables changes to resource privileges for users and groups, by copying privileges from other resources. Changes can be made to one or more resources with different source resource for one or many users and groups. Resource privileges can be set for a specified set of users and groups without modifying any existing privileges for other users and groups, or the procedure can set resource privileges restrictively to only privileges of source resource explicitly.

Only a user with GRANT privilege on a resource can modify the privileges for that resource. The owner of a resource always has GRANT privilege, as do users with the MODIFY\_ALL\_RESOURCES right.

When "mode" is "OVERWRITE\_APPEND", or is not supplied, privileges are applied on a per-user or per-group basis, so that updating privileges for one user or group does not alter privileges from any other user or group. The privileges applied for a user or group replace the previous value for that user or group.

When "mode" is "SET\_EXACTLY", all privileges on the resource are made to look exactly like the privileges of source resource.

When "updateRecursively" is "false", the privileges are applied only to the specified resources. When it is "true", the privileges are recursively applied into any CONTAINER or DATA\_SOURCE resource specified. When recursively applying privileges, the privilege change is ignored for any resource the user lacks owner privileges for.

Privileges that are not applicable for a given resource type are automatically stripped down to the set that is legal for each resource. TABLE resources support NONE, READ, WRITE, SELECT, INSERT, UPDATE, and DELETE. PROCEDURE resources support NONE, READ, WRITE, and EXECUTE. All other resource types only support NONE, READ, and WRITE.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | updateRecursively  0 (false) or null - only update the given resource and not the children.  1 (true) - update children recursively | BIT |
| IN | copyPrivilegeMode  null (default) - do not set any privileges at all  0 - set mode to "OVERWRITE\_APPEND" - merges and does not update privileges for users or groups not mentioned.  1 - set the mode to "SET\_EXACTLY" - makes privileges look exactly like those provided in the call. | INTEGER |
| IN | inCopyPrivilegeEntries | LONGVARCHAR  XML formatted string containing the copy privilege entry. An example is shown below of how to set the input.  There can be mulitiple "copyPrivilegeEntry" XML entries indicating multiple sources each with a pairing of multiple destinations.  There is a single <resource:srcResource> entry. There may be multiple <resource:dstResource> entries.  See documentation above for examples. |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | updateRecursively | 1 |
| IN | copyPrivilegeMode | 0 |
| IN | inCopyPrivilegeEntries | <resource:copyPrivilegeEntry>  <resource:srcResource>  <resource:path>/shared/ASAssets/Utilities/examples/repository/source</resource:path>  <resource:type>CONTAINER</resource:type>  </resource:srcResource>  <resource:dstResource>  <resource:path>/shared/ASAssets/Utilities/examples/repository/source/folder1</resource:path>  <resource:type>CONTAINER</resource:type>  </resource:dstResource>  <resource:dstResource>  <resource:path>/shared/ASAssets/Utilities/examples/repository/source/folder3</resource:path>  <resource:type>CONTAINER</resource:type>  </resource:dstResource>  </resource:copyPrivilegeEntry>  <resource:copyPrivilegeEntry>  <resource:srcResource>  <resource:path>/shared/ASAssets/Utilities/examples/repository/source</resource:path>  <resource:type>CONTAINER</resource:type>  </resource:srcResource>  <resource:dstResource>  <resource:path>/shared/ASAssets/Utilities/examples/repository/source/ds\_ordersCopy</resource:path>  <resource:type>DATA\_SOURCE</resource:type>  </resource:dstResource>  </resource:copyPrivilegeEntry> |
| OUT | success | 1 |
| OUT | faultResponse | null |

### createAllFolders

This procedure is used to create all the CIS folders designated by the incoming folder path variable.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceFolderPath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceFolderPath | ‘/shared/ASAssets/Utilities/repository/examples/target/newfolder’ |
| OUT | success | 1 |
| OUT | faultResponse | null |

### createAllFoldersPrivileges

This procedure is used to create all the CIS folders designated by the incoming folder path variable and copy privileges from the parent folder. The parent folder is determined to be the last folder encountered in the path that exists. All other folders beyond the existing folder ar considered children and must be created.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceFolderPath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | updatePrivilegesRecursively  0 (false) or null - only update privileges for the given resource and not the children.  1 (true) - update privileges for children recursively | BIT |
| IN | copyPrivilegeMode  null (default) - do not set any privileges at all  0 - set mode to "OVERWRITE\_APPEND" - merges and does not update privileges for users or groups not mentioned.  1 - set the mode to "SET\_EXACTLY" - makes privileges look exactly like those provided in the call. | BIT |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceFolderPath | ‘/shared/ASAssets/Utilities/repository/examples/target/newfolder’ |
| IN | updatePrivilegesRecursively | 0 |
| IN | copyPrivilegeMode | 0 |
| OUT | success | 1 |
| OUT | faultResponse | null |

### createConnector

This procedure creates a JMS connector

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | name | VARCHAR(100) |
| IN | groupName | VARCHAR(100) |
| IN | jmsClientID | VARCHAR(1024) |
| IN | annotation | VARCHAR(1024) |
| IN | jndiContextFactory | VARCHAR(1024) |
| IN | jndiProperties | LONGVARCHAR |
| IN | jndiProviderUrl | VARCHAR(1024) |
| IN | jndiUser | VARCHAR(50) |
| IN | jndiPassword | VARCHAR(50) |
| IN | queueConnectionFactory | VARCHAR(1024) |
| IN | minPool | INTEGER |
| IN | maxPool | INTEGER |
| IN | poolTimeout | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | name | 'myMQ' |
| IN | groupName | '<Group Name>' |
| IN | jmsClientID | '<JMS Client ID>' |
| IN | annotation | 'This is a JMS message queue' |
| IN | jndiContextFactory | '<JNDI context factory>' |
| IN | jndiProperties | '<JNDI Properties XML>' |
| IN | jndiProviderUrl | '<JNDI Provider URL>' |
| IN | jndiUser | 'myMQuser' |
| IN | jndiPassword | 'myMQpassword' |
| IN | queueConnectionFactory | '<Queue Connection Factory>' |
| IN | minPool | 1 |
| IN | maxPool | 10 |
| IN | poolTimeout | 300 |

### createConsumingViews

This procedure creates views that consume any views found in the input starting container. Creating views that consume procedures is not supported at this time (though very probably will in a future Utilities release.)

This is used for quickly creating 1-to-1 views up the best practices layers.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceContainer - The absolute path to a container to examine for views to build consuming views from.  Values: Any container path | / ResourceDefs.ResourcePath |
| IN | destinationContainer - The absolute path to the container to put the consuming views into.  Values: Any container path | ResourceDefs.ResourcePath |
| IN | recurse - A flag indicating whether the script should recurse into any child containers.  Values: 1 = yes, 0 = no | BIT |
| IN | overwrite - A flag indicating whether the script should replace any existing views it encounters.  Values: 1 = yes, 0 = no | BIT |
| OUT | success - Indicates whether the operation was a success or not.  Values: 1 = yes, 0 = no | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceContainer | ‘/shared/ASAssets/Utilities/examples/repository/source/ds\_inventory/tutorial’ |
| IN | destinationContainer | ‘/shared/ASAssets/Utilities/examples/repository/source/ds\_inventory/tutorial’ |
| IN | recurse | 1 |
| IN | overwrite | 1 |
| OUT | success | 1 |

### createDataSource

This procedure creates an empty data source. See chart below for valid data source types and the folders where they can be created. As of 5.2.0.02.27 some data source types had issues when attempting to create data sources for them. They are also noted in the chart below.

After creation, repository/updateResourceDataSource must be used to add connectivity information. Then repository/addRemoveDataSourceChildren must be used to introspect resources.

| **Data Source Type** | **Valid Location** | **Notes** |
| --- | --- | --- |
| 'COMPOSITE\_DATABASE' | /services/Databases |  |
| 'COMPOSITE\_SERVICE' | /services/Web Services |  |
| 'Custom Java Procedure' | /shared/... |  |
| 'DB2 v7 (Type 2)' | /shared/... |  |
| 'DB2 v7 (Type 4)' | /shared/... |  |
| 'DB2 v8 (Type 2)' | /shared/... |  |
| 'DB2 v8 (Type 4)' | /shared/... |  |
| 'DB2 v9 (Type 2)' | /shared/... |  |
| 'DB2 v9 (Type 4)' | /shared/... |  |
| 'DB2 z/OS v8 (Type 4)' | /shared/... |  |
| 'DataDirect Mainframe' | /shared/... | Requires DataDirect Mainframe connector software from DataDirect |
| 'Essbase' | /shared/... | Requires appropriate CAC software and license |
| 'File-Cache' | /shared/... |  |
| 'File-Delimited' | /shared/... |  |
| 'File-XML' | /shared/... |  |
| 'Greenplum 3.3' | /shared/... |  |
| 'Informix 9.x' | /shared/... | This one causes a problem when used: java.lang.String cannot be cast to java.lang.Integer |
| 'LDAP' | /shared/... |  |
| 'Microsoft Access' | /shared/... |  |
| 'Microsoft Excel' | /shared/... |  |
| 'Microsoft Excel (non-ODBC)' | /shared/... |  |
| 'Microsoft SQL Server 2000' | /shared/... |  |
| 'Microsoft SQL Server 2005' | /shared/... |  |
| 'Microsoft SQL Server 2008' | /shared/... |  |
| 'MySQL 4.0' | /shared/... |  |
| 'MySQL 5.0' | /shared/... |  |
| 'NeoView 2.3' | /shared/... |  |
| 'NeoView 2.4' | /shared/... |  |
| 'Netezza 3.0' | /shared/... | This one causes a problem when used: java.lang.String cannot be cast to java.lang.Integer |
| 'Netezza 4.5' | /shared/... | This one causes a problem when used: java.lang.String cannot be cast to java.lang.Integer |
| 'Netezza 5.0' | /shared/... | This one causes a problem when used: java.lang.String cannot be cast to java.lang.Integer |
| 'Oracle 10g (OCI Driver)' | /shared/... |  |
| 'Oracle 10g (Thin Driver)' | /shared/... |  |
| 'Oracle 11g (OCI Driver)' | /shared/... |  |
| 'Oracle 11g (Thin Driver)' | /shared/... |  |
| 'Oracle 8i (OCI Driver)' | /shared/... |  |
| 'Oracle 8i (Thin Driver)' | /shared/... |  |
| 'Oracle 9i (OCI Driver)' | /shared/... |  |
| 'Oracle 9i (Thin Driver)' | /shared/... |  |
| 'Oracle E-Business Suite on 10g' | /shared/... | Requires appropriate CAC software and license |
| 'Oracle E-Business Suite on 8i' | /shared/... | Requires appropriate CAC software and license |
| 'Oracle E-Business Suite on 9i' | /shared/... | Requires appropriate CAC software and license |
| 'Salesforce.com' | /shared/... | Requires appropriate CAC software and license |
| 'SAP' | /shared/... | Requires appropriate CAC software and license |
| 'SAP BW' | /shared/... | Requires appropriate CAC software and license |
| 'Siebel' | /shared/... | Requires appropriate CAC software and license |
| 'Sybase' | /shared/... |  |
| 'Sybase IQ' | /shared/... |  |
| 'System' | /shared/... |  |
| 'Teradata' | /shared/... |  |
| 'Teradata 12' | /shared/... |  |
| 'Teradata 13' | /shared/... |  |
| 'XML/HTTP' | /shared/... |  |

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dataSourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | dataSourceName | /lib/resource/ResourceDefs.ResourceName |
| IN | dataSourceType | /lib/resource/ResourceDefs.ResourceType |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dataSourcePath | ‘/shared’ |
| IN | dataSourceName | ‘DS\_Oracle11g’ |
| IN | dataSourceType | 'Oracle 11g (Thin Driver)’ |
| OUT | success | 1 |
| OUT | createResponse | XML |
| OUT | faultResponse | null |

### createFolder

This procedure creates a single folder at the end of the folder path. All other intermediate folders must exist. For example if /shared/intermediate/folder1 is to be created, then /shared and /shared/intermediate must exist for this procedure to work properly. Use the repository/createAllFolders procedure to create all of the entire folder path structure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceFolderPath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceFolderPath | ‘/shared/ASAssets/Utilities/repository/examples/source’ |
| OUT | success | 1 |
| OUT | faultResponse | null |

### createOrUpdateConnector

This procedure creates or updates a JMS connector

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | forceNullUpdate | SMALLINT |
| IN | name | VARCHAR(100) |
| IN | groupName | VARCHAR(100) |
| IN | jmsClientID | VARCHAR(1024) |
| IN | annotation | VARCHAR(1024) |
| IN | jndiContextFactory | VARCHAR(1024) |
| IN | jndiProperties | LONGVARCHAR |
| IN | jndiProviderUrl | VARCHAR(1024) |
| IN | jndiUser | VARCHAR(50) |
| IN | jndiPassword | VARCHAR(50) |
| IN | queueConnectionFactory | VARCHAR(1024) |
| IN | minPool | INTEGER |
| IN | maxPool | INTEGER |
| IN | poolTimeout | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | forceNullUpdate | 0 |
| IN | name | 'myMQ' |
| IN | groupName | '<Group Name>' |
| IN | jmsClientID | '<JMS Client ID>' |
| IN | annotation | 'This is a JMS message queue' |
| IN | jndiContextFactory | '<JNDI context factory>' |
| IN | jndiProperties | '<JNDI Properties XML>' |
| IN | jndiProviderUrl | '<JNDI Provider URL>' |
| IN | jndiUser | 'myMQuser' |
| IN | jndiPassword | 'myMQpassword' |
| IN | queueConnectionFactory | '<Queue Connection Factory>' |
| IN | minPool | 1 |
| IN | maxPool | 10 |
| IN | poolTimeout | 300 |

### createResource

Create a resource in a default state. The actual content of the resource is not provided here. This procedure only handles the initial creation. There are other procedures to update the resource with its content such as the following:

* repository/updateDefSetDef
* repository/updateResourceCacheConfig
* repository/updateResourceCacheConfiguration
* repository/updateResourceCacheEnabled
* repository/updateResourceDataSource
* repository/updateResourcesSqlTable
* repository/updateSqlScript
* repository/updateSqlTable
* repository/updateTrigger

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceName | VARCHAR(255) |
| IN | resourceType | VARCHAR(255) |
| IN | resourceSubType | VARCHAR(255) |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/ASAssets/Utilities/repository/examples/target’ |
| IN | resourceName | ‘PRODUCT\_VIEW’ |
| IN | resourceType | ‘TABLE’ |
| IN | resourceSubType | ‘SQL\_TABLE' |
| OUT | success | 1 |
| OUT | createResponse | Create Response XML: <resource> |
| OUT | faultResponse | null |

### createResourceCopy

This is a procedure is used to create a copy of a resource by adding \_Copy\_1 and etc. It looks for existing copies and increments the number. The resource can be any resource including folders.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | mode  C=Copy resource and leave original, R=Rename resource to the "copied" name. If left null, the default is copy and leave original in place | CHAR(1) |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR |
| OUT | success | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | mode | C |
| IN | resourcePath | ‘/shared/ASAssets/Utilities/repository/examples/target/newfolder’ |
| IN | resourceType | ‘CONTAINER’ |
| OUT | success | 1 |

### createUnionView

This procedure creates a union view composed of the columns of two other views. The two views do not necessarily need to have the same columns (though logically, there should at least be one column in common.) The procedure will substitute NULLs for columns that don't exist in a particular view (cast to the data type of the other view's column.) If a column is common across both views but either the case of the column name or the data types differ, a flag is used to indicate which view's column definition to give preference to. The "AllIndicator" input specifies whether the view should be created as UNION (0) or UNION ALL (1).

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | UnionViewPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | View1Path | /lib/resource/ResourceDefs.ResourcePath |
| IN | View2Path | /lib/resource/ResourceDefs.ResourcePath |
| IN | NameTypeConflictPreference | INTEGER |
| IN | AllIndicator | BIT |
| OUT | success | BIT |
| OUT | responseXML | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: A view has been wrapped around /shared/examples/productCatalog\_Transformation called "productCatalog\_wrapper"**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | UnionViewPath | '/shared/examples/products\_union' |
| IN | View1Path | '/shared/examples/ds\_inventory/products' |
| IN | View2Path | '/shared/examples/productCatalog\_wrapper' |
| IN | NameTypeConflictPreference | 2 |
| IN | AllIndicator | 1 |
| OUT | success | 1 |
| OUT | responseXML | (response XML) |
| OUT | faultResponse | NULL |

### deleteAllConnectors

This procedure deletes all the configured JMS connectors.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |

### deleteConnector

This procedure deletes a configured JMS connector.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | CONN\_NAME | VARCHAR(100) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | CONN\_NAME | 'myMQ' |

### destroyResource

This procedure is used to destroy/delete a resource.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceName | VARCHAR(255) |
| IN | resourceType | VARCHAR(255) |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/ASAssets/Utilities/repository/examples/target’ |
| IN | resourceName | ‘PRODUCT\_VIEW’ |
| IN | resourceType | ‘TABLE’ |
| OUT | success | 1 |
| OUT | createResponse | Create Response XML: <resource> |
| OUT | faultResponse | null |

### expireProcCacheEntryByName

The expireProcCacheEntryByName procedure can be used for any procedural cache to clear a single entry per call for a given set of parameters. This allows the selective expiration of part of a procedural cache’s contents without completely expiring the entire cache. After expiring the results associated with a set of parameters, the Composite server will re-execute the procedure for the given set of parameters and will cache the newly calculated result.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | cachedResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | cacheStatusPath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | params | VARCHAR(255) |
| OUT | isSuccessful | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | cachedResourcePath | ‘/shared/examples/LookupProduct’ |
| IN | cacheStatusPath | '/shared/examples/ds\_orders/cache\_status' |
| IN | params | ''1'' (single quoted 1) |
| OUT | isSuccessful | 1 |

### exportResourceDefinitions

This procedure generates a complete export of the definitions for all resources defined in CIS under shared and its subdirectories.

A complete export of the definitions for all view and procedure resources contained under the folder /shared are written under the starting folder indicated by the input parameter outputDirectory in a folder structure on the CIS host that matches the structure of the namespace tree.

Resource definitions are exported to files with a name matching the exported resource. The file extension of the output file is set by the input parameter outputFileExtension.

Directories and files are written using the credentials of the account that the CIS server is running under. Attempts to export resource definitions to directories that the CIS app account does not have sufficient rights to read and write to will fail.

Please note that the underlying CIS system tables used by this procedure require the ACCESS\_TOOLS right to query, so this procedure will not work successfully if the invoking user does not have ACCESS\_TOOLS. Only resources that the user has the READ privilege on will be exported.

Also note that this view will not expose the java code for CJP procedures. The author must provide Java source code.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | outputDirectory | LONGVARCHAR |
| IN | outputFileExtension | VARCHAR(10) |
| OUT | result | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | outputDirectory | 'C:\Tmp' |
| IN | outputFileExtension | '.sql' |
| OUT | result | 'SUCCESS' |

### exportResourcePrivileges

This procedure generates a complete export of the privileges for all the specified resources. An optional filter can be specified so that only those privileges that were explicitly set (and not inherited through a global right or group membership) are exported.

The generated export is written to the CIS host filesystem in XML format. This file can be modified (to apply new privileges to existing resources) or left alone (to re-apply existing privilege settings) and used by the resources/importResourcePrivileges() procedure.

The **in\_path\_list** parameter specifies the resources paths and types of the resource privileges to export. It is composed of a comma separated list of colon separated resource path and type pairs. For example:

/shared/examples:CONTAINER,  
 /services/databases/examples:DATA\_SOURCE

The **Constants** section of the resource /lib/resources/ResourceDefs lists the possible values of resource types.

The **in\_filter** input parameter can be used to specify whether all privileges should be exported (use either NULL or an empty string) or just those that were explicitly set (use the string 'ALL\_EXPLICIT'.)

If column level security is being used on any views whose privileges are being exported, set the **in\_includeColumns** parameter to 1 to preserve column level privilege settings. Otherwise set the parameter to 0 to conserve space.

The procedure does not output any parameters. If the procedure exits without throwing an error, then the export operation was a success.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | in\_path\_list | LONGVARCHAR |
| IN | in\_filter | VARCHAR(255) |
| IN | in\_includeColumns | BIT |
| IN | in\_filename | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | in\_path\_list | '/shared/examples:CONTAINER, /services/databases/examples:DATA\_SOURCE' |
| IN | in\_filter | 'ALL\_EXPLICIT' |
| IN | in\_includeColumns | 0 |
| IN | in\_filename | 'C:\cis\_examples\_privileges.xml' |

### findDependentResources

This procedure is used to find any dependent resource for the resources found within the boundaries of the starting path. A dependent resource is one in which invokes the source resource designated by "srcResourcePath".

Example 1: Find all dependents of ASAssets folder resources the startingResourcePath=/shared/ASAssets/

By adding a forward slash at the end of the path, it insures that only /shared/ASAssets resources are searched and not other folders that start like /shared/ASAssets.

Example 2: Find all dependents of ASAssets folder resources the startingResourcePath=/shared/ASAssets/Utilities/

Example 3: Find all dependents of ASAssets folder resources the startingResourcePath=/shared/Common/Utilities/

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | displayImpactedOnly - 1=display impacted only. 0=display all | BIT |
| IN | startingResourcePath - Any DV folder/container path. | VARCHAR(4000) |
| IN | resourceTypeList - A comma-separated list of resource types to filter on. Null to select all. e.g. TABLE, PROCEDURE, LINK, TREE, DATA\_SOURCE, DEFINITION\_SET, TRIGGER | VARCHAR |
| OUT | result | PIPE (  srcResourceName VARCHAR,  srcResourcePath VARCHAR(4000),  srcResourceType VARCHAR(40),  depResourceName VARCHAR(255),  depResourcePath VARCHAR(4000),  depResourceType VARCHAR(40),  depSubtype VARCHAR(40),  depId INTEGER,  depOwnerDomain VARCHAR(255),  depOwnerName VARCHAR(255),  depImpactLevel VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | N |
| IN | displayImpactedOnly - 1=display impacted only. 0=display all | 0 |
| IN | startingResourcePath - Any DV folder/container path. | ‘/shared/ASAssets/Utilities’ |
| IN | resourceTypeList - A comma-separated list of resource types to filter on. Null to select all. e.g. TABLE, PROCEDURE, LINK, TREE, DATA\_SOURCE, DEFINITION\_SET, TRIGGER | TABLE,PROCEDURE,DEFINITION\_SET |
| OUT | result | A list of utilities which still contain dependent resources. |

### findVectorInResources

This procedure is used to locate any procedures that contain a VECTOR in the IN or OUT parameters or in the DECLARE statements.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | startingParentPath | VARCHAR(4096) |
| OUT | result | PIPE (  resourcePath VARCHAR(4096), -- The resource path where the resource was found  resourceType VARCHAR, -- The resource type  areaFound VARCHAR, -- IN[#] OUT[#] DECLARE[#]. The # sign indicates how many instances were found in that area.  definition LONGVARCHAR -- The procedure definition  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | startingParentPath | ‘/shared/ASAssets/Utilities’ |
| OUT | result | A list of utilities which still contain vectors. |

### fixLeadingCharactersInFolderPath (Custom Function)

This procedure is used to fix the leading characters in a folder path. Any path that contains a leading underscore '\_' or a number '0123456789' must have double quotes inserted around that portion of the folder. This procedure would be called in conjunction with other procedures. For example, when generating a view based off of another view, the SELECT statement's FROM clause would require that the path to the underlying view be fixed with double quotes if it finds.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | fixedResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/ASAssets/Utilities/repository/examples/1folder/\_folder’ |
| OUT | fixedResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/"1folder"/"\_folder"’ |

### freeUnusedMemory

This procedure is used to free unused memory.

1. **Parameters: NONE**

### (Custom Function)

This procedure is used to fix the leading characters in a folder path. Any path that contains a leading underscore '\_' or a number '0123456789' must have double quotes inserted around that portion of the folder. This procedure would be called in conjunction with other procedures. For example, when generating a view based off of another view, the SELECT statement's FROM clause would require that the path to the underlying view be fixed with double quotes if it finds.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | fixedResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/ASAssets/Utilities/repository/examples/1folder/\_folder’ |
| OUT | fixedResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/"1folder"/"\_folder"’ |

### generateOptFile

This procedure loops through all of the data source connections inside the DV instance and generates the -set lines needed to create an "options file" to deploy those data sources to another DV server. You can export the lines to a file and then modify the parameters for each line to be suitable to the target deployment server. For example you can run this script on "development" and then modify the host, port, username, password, etc to match "Test" and then use those -set lines in your options file to perform the import into "Test"..

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | resourcePathList - null=retrieve options for all paths. comma-separated path list=retrieve options for paths in the list. | LONGVARCHAR |
| IN | | excludePathList - null=no path exclusions. comma-separated path list=exclude paths in the list. | LONGVARCHAR |
| OUT | | resultPipe - A cursor containing each -set line for each data source configured. | CURSOR (line LONGVARCHAR) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePathList | ‘/shared/ASAssets/KPImetrics’ |
| IN | excludePathList | null |
| OUT | resultPipe | -set /shared/ASAssets/KPImetrics/Physical/Metadata/KPI\_oracle\_11g DATA\_SOURCE database XE  -set /shared/ASAssets/KPImetrics/Physical/Metadata/KPI\_oracle\_11g DATA\_SOURCE host localhost  -set /shared/ASAssets/KPImetrics/Physical/Metadata/KPI\_oracle\_11g DATA\_SOURCE password CHANGE\_PASSWORD  -set /shared/ASAssets/KPImetrics/Physical/Metadata/KPI\_oracle\_11g DATA\_SOURCE port 1521  -set /shared/ASAssets/KPImetrics/Physical/Metadata/KPI\_oracle\_11g DATA\_SOURCE user CIS\_KPI |

### getAllDataSourceChildren

This procedure returns all the children for a given DATA\_SOURCE type resources found under the starting path.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | | includeColumns | BIT |
| OUT | | childResCursor | CURSOR (resourceName VARCHAR(255),  resourcePath VARCHAR(1024),  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  isNullable VARCHAR(255),  columnName VARCHAR(255),  columnType VARCHAR(255),  nativeBaseType VARCHAR(255),  nativeType VARCHAR(255) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples’ |
| IN | includeColumns | 1 |
| OUT | childResCursor | (result too large to display) |

### getAllDataSources

This procedure returns all the DATA\_SOURCE type resources found under the starting path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | resourceTreeList | CURSOR (  name VARCHAR,  resPath TypeDefinitions.pathType,  resType VARCHAR,  subType VARCHAR) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples' |
| OUT | resourceTreeList | See chart below |

* 1. Chart showing example output for resourceTreeList:

|  |  |  |  |
| --- | --- | --- | --- |
| name | resPath | resType | subType |
| ds\_inventory | /shared/examples/ds\_inventory | DATA\_SOURCE | RELATIONAL\_DATA\_SOURCE |
| ds\_orders | /shared/examples/ds\_orders | DATA\_SOURCE | RELATIONAL\_DATA\_SOURCE |
| ds\_XML | /shared/examples/ds\_XML | DATA\_SOURCE | XML\_FILE\_DATA\_SOURCE |

### getAncestorResources

Get all of the ancestors of the specified resource up to and including the root resource.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath** - CIS source path to an actual resource | pathType |
| IN | **resourceType** - Type of CIS resource to be created. It is null on the first invocation. | VARCHAR |
| OUT | result | OUT result PIPE (  resourceName VARCHAR(255),  resourcePath VARCHAR(4096),  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  changeid INTEGER,  version VARCHAR(255),  introspectState VARCHAR(255),  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  annotation LONGVARCHAR,  explicitlyDesigned BIT,  isFunction BIT,  tableType VARCHAR(255),  sqlText LONGVARCHAR,  scriptText LONGVARCHAR,  proprietaryModel LONGVARCHAR,  childCount INTEGER,  dataSourceType VARCHAR(255),  creationDate VARCHAR(255),  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples/ds\_orders/customers |
| IN | resourceType | TABLE |
| OUT | result | See charts below |

* 1. Chart 1: Columns (1-4)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | resourceType | subtype |  |  |
| [NULL] | / | CONTAINER | NONE |  |  |
| Shared | /shared | CONTAINER | FOLDER\_CONTAINER |  |  |
| Examples | /shared/examples | CONTAINER | FOLDER\_CONTAINER |  |  |
| ds\_orders | /shared/examples/ds\_orders | DATA\_SOURCE | RELATIONAL\_DATA\_SOURCE |  |  |

* 1. Chart 2: Columns (5-11)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| enabled | id | changeid | ownerDomain | ownerName | impactLevel | childCount |
| 1 | 1 | 202 | composite | system | NONE | 6 |
| 1 | 10104 | 202 | composite | admin | NONE | 9 |
| 1 | 20586 | 205 | composite | admin | NONE | 11 |
| 1 | 20670 | 205 | composite | admin | NONE | 7 |

* 1. Chart 3: Columns (11-15)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dsID | dsResName | dsResPath | dsResType | dsResSubType |
| [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |
| 20605 | ds\_inventory | /shared/examples/ds\_inventory | DATA\_SOURCE | RELATIONAL\_DATA\_SOURCE |

* 1. Chart 4: Columns (16-18)

|  |  |  |
| --- | --- | --- |
| dsEnabled | dsChildCount | datasourceType |
| [NULL] | [NULL] | [NULL] |
| 1 | 4 | PostgresSQL 9.1 |

### GetAnsi2NativeMapping

Given a Composite (ANSI) data type and a path to a data source, this procedure returns the data source's data type equivalent. This is a wrapper script that automatically detects which version of CIS is running and calls the appropriate CJP.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | datasourcePath | VARCHAR(2147483647) |
| IN | cisType | VARCHAR(2147483647) |
| OUT | result | CURSOR(  cisType VARCHAR(2147483647),  cisNormalizedType VARCHAR(2147483647),  cisBaseType VARCHAR(2147483647),  cisScale INTEGER,  cisPrecision INTEGER,  dataTypeId INTEGER,  dataTypeName VARCHAR(2147483647),  nativeType VARCHAR(2147483647),  nativeBaseType VARCHAR(2147483647),  nativeScale INTEGER,  nativePrecision INTEGER ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | datasourcePath | '/shared/examples/ds\_orders' |
| IN | cisType | 'LONGVARCHAR' |
| OUT | result | (  'LONGVARCHAR',  'LONGVARCHAR',  'LONGVARCHAR',  -1,  -1,  -983,  'LONGVARCHAR',  'varchar(2147483647)',  'varchar',  '2147483647',  '-1' ) |

### getBasicResourceCursor

This procedure retrieves the basic resource metadata for a given resource. A cursor of metadata is returned. The results being returned will contain values or be null based on the type of resource as described below:

* If a resource is impacted (showing red in Studio), then the impactLevel and impactMessage will contain the level and the reason for being impacted otherwise impactLevel=NONE.
* If the resource is a data source or a container (folder, catalog, schema) then childCount will contain the number of children.
* If the resource is a data source then the dataSourceType will contain what type it is such as “File-XML”, “Oracle 11g (Thin Driver)” or “PostgreSQL 9.1”
* If the resource is a procedure then the scriptText will contain the actual procedure definition.
* If the resource is a view then the tableType=VIEW and the sqlText contains the view definition.
* If the resource is a database table then the tableType=TABLE and sqlText is null.

This procedure invokes the native TDV API directly.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | result | CURSOR (  name VARCHAR(255),  "path" VARCHAR(32768),  "type" VARCHAR(32768),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  changeid INTEGER,  version VARCHAR(255),  introspectState VARCHAR(255),  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  annotation LONGVARCHAR,  explicitlyDesigned BIT,  isFunction BIT,  tableType VARCHAR(255),  sqlText LONGVARCHAR,  scriptText LONGVARCHAR,  proprietaryModel LONGVARCHAR,  childCount INTEGER,  dataSourceType VARCHAR(255),  creationDate VARCHAR(255),  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/repository/examples/source/proc1 |
| IN | resourceType | PROCEDURE |
| OUT | result | |  |  | | --- | --- | | name: | ‘proc1’ | | path: | ‘/shared/ASAssets/Utilities/repository/examples/source/proc1’ | | type: | ‘PROCEDURE’ | | subtype: | ‘SQL\_SCRIPT\_PROCEDURE’ | | enabled: | 1 | | id | 548377 | | changeid | 16318 | | version | null | | introspectState | null | | ownerDomain | composite | | ownerName | admin | | impactLevel | NONE | | impactMessage | null | | annotation | Proc1 description | | explicitlyDesigned | 0 | | tableType | null | | scriptText | <too large to fit in this window> | | childCount | null | | dataSourceType | null | |

### getBasicResourceCursor\_ActionAttributes

This procedure retrieves the resource metadata for a given resource that is contains Action Attributes such as a TRIGGER. A cursor of metadata is returned. This procedure invokes 2 lower level API procedures:

* repository/lowerLevelProcedures/getBasicResourceXML - this performs the actual invocation to the CIS repository API and returns XML
* repository/lowerLevelProcedures/getBasicResourceXSLT\_ActionAttributes - this procedure takes the XML response and turns it into a cursor which is more usable by other CIS procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (   actionType VARCHAR(255),   name VARCHAR(255),   "type" VARCHAR(255),   “value” VARCHAR(255) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/trigger1’ |
| IN | resourceType | TRIGGER |
| OUT | resourceCursor | |  |  |  |  | | --- | --- | --- | --- | | actionType | name | type | value | | PROCEDURE | PARAMETERS | STRING | 'string2' | | PROCEDURE | PATH | STRING | ‘/shared/ASAssets/Utilities/repository/examples/ source/proc1' | |

### getBasicResourceCursor\_PROCEDURE

This procedure retrieves the resource metadata for a given resource that is a PROCEDURE (SQL Script, DB store procedure, transformation, CJP, etc.) A cursor of metadata is returned. This procedure invokes 2 lower level API procedures:

* repository/lowerLevelProcedures/getBasicResourceXML - this performs the actual invocation to the CIS repository API and returns XML
* repository/lowerLevelProcedures/getBasicResourceXSLT\_PROCEDURE - this procedure takes the XML response and turns it into a cursor which is more usable by other CIS procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (  resourceName VARCHAR(255),   resourcePath VARCHAR(1024),   resourceType VARCHAR(255),   subtype VARCHAR(255),   enabled BIT,   annotation LONGVARCHAR,   parameterName VARCHAR(255),   parameterType VARCHAR(255),   parameterDirection VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | '/shared/examples/LookupProduct' |
| IN | resourceType | 'PROCEDURE' |
| OUT | resourceCursor | (See table below for resourceCursor example output) |

* 1. resourceCursor example output:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | Resource Type | subtype | enabled | annotation | parameterName | parameterType | parameterDirection |
| LookupProduct | $1/ LookupProduct | PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 1 |  | [NULL] | [NULL] | [NULL] |
| LookupProduct | $1/ LookupProduct | PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 1 |  | ProductID | INTEGER | IN |
| LookupProduct | $1/ LookupProduct | PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 1 |  | result | CURSOR(… | OUT |

Note: Path $1 = /shared/examples

### getBasicResourceCursor\_PROCEDURE\_CURSOR

This procedure retrieves the cursor output metadata for a given PROCEDURE (SQL Script, DB store procedure, transformation, CJP, etc.) The user must pass a number indicating which cursor input/ouput parameter to return. A cursor of metadata is then returned. This procedure invokes the getResource() native API directly.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40)) |
| IN | cursorNum | INTEGER |
| OUT | resourceCursor | CURSOR ( resourceId BIGINT, --The resource id.  resourceName VARCHAR(255), --The name of the resource.  resourcePath VARCHAR(4096), --The full resource path.  resourceType VARCHAR(255), --The resource type: PROCEDURE  subtype VARCHAR(255), --The resource sub type. e.g. SQL\_SCRIPT\_PROCEDURE, XSLT\_TRANSFORM\_PROCEDURE  changed BIGINT, --The change id.  ownerName VARCHAR(255), --The owner name  ownerDomain VARCHAR(255), --The owner domain.  impactLevel VARCHAR(255), --The impact level - NONE for no impact.  enabled BIT, --Indicates whether the resource is enabled (1) or not (0)  annotation LONGVARCHAR,--Provides the annotation for the resource  explicitlyDesigned BIT, --Determines whether the model was explicitly designed or no  tableType VARCHAR(255), --Provides the tableType which is null for procedures. This column is here for consistency with getBasicResourceCursor\_SQL\_TABLE.  sqlText LONGVARCHAR, --The scriptText for a cursor  proprietaryModel LONGVARCHAR, --The proprietary model  columnNum INTEGER,--The ordinal value of the column (column number or column order)  columnName VARCHAR(255), --The name of the column.  columnType VARCHAR(4096), --The actual column type. e.g. CURSOR, XML, VARCHAR(255), DECIMAL(32,2), BIGINT, NUMERIC(2147483647,0), and etc.  columnBaseType. VARCHAR(255),--The base column type. e.g. VECTOR, CURSOR, XML, BIGINT, INTEGER, DECIMAL, VARCHAR, NUMERIC, and etc.  dataType VARCHAR(255),--1 of [sqlType, xmlType, cursorType, cursorColumn]  paramDirection VARCHAR(255),--The direction of the parameter. e.g. 1 of [IN,OUT,INOUT]  nativeBaseType VARCHAR(32768),--The native base type. Currently defaulted to null. Future expansion.  nativeType VARCHAR(32768) --The native column type. Currently defaulted to null. Future expansion. ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | '/shared/examples/LookupProduct' |
| IN | resourceType | 'PROCEDURE' |
| IN | cursorNum | 1 |
| OUT | resourceCursor | (Result too large to display here) |

### getBasicResourceCursor\_ResourceAttributes

This procedure retrieves the resource metadata for a given resource that is a PROCEDURE (SQL Script, DB store procedure, transformation, CJP, etc.) A cursor of metadata is returned. This procedure invokes 2 lower level API procedures:

* repository/lowerLevelProcedures/getBasicResourceXML - this performs the actual invocation to the CIS repository API and returns XML
* repository/lowerLevelProcedures/getBasicResourceXSLT\_PROCEDURE - this procedure takes the XML response and turns it into a cursor which is more usable by other CIS procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (  name VARCHAR(255),   type VARCHAR(1024),   value LONGVARCHAR,   valueList LONGVARCHAR,   valueMap LONGVARCHAR  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | '/shared/examples/ds\_orders' |
| IN | resourceType | 'DATA\_SOURCE' |
| OUT | resourceCursor | ('autoAddChildren', 'BOOLEAN', 'true', NULL, NULL), ('login', 'STRING', 'tutorial', NULL, NULL), … |

### getBasicResourceCursor\_SQL\_TABLE

This procedure retrieves the resource metadata for a given resource that is a SQL TABLE/VIEW. A cursor of metadata is returned. This procedure invokes the getResource() native API directly.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (  resourceId BIGINT,  resourceName VARCHAR(255),  resourcePath VARCHAR(1024),  resourceType VARCHAR(255),  subtype VARCHAR(255),  changeId BIGINT,  ownerName VARCHAR(255),  ownerDomain VARCHAR(255),  impactLevel VARCHAR(255),  enabled BIT,  annotation LONGVARCHAR,  explicitlyDesigned BIT,  tableType VARCHAR(255),  sqlText LONGVARCHAR,  proprietaryModel LONGVARCHAR,  columnNum INTEGER,  columnName VARCHAR(255),  columnType VARCHAR(255),  nativeBaseType VARCHAR(32768),  nativeType VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/repository/examples/source/PRODUCT\_VIEW |
| IN | resourceType | TABLE |
| OUT | resourceCursor | See table below for resourceCursor example output |

* 1. resourceCursor example output:
     1. Note: Path $1 = /shared/ASAssets/Utilities/repository/examples/source

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | Resource Type | subtype | enabled | Table Type | SQL Text | columnName | columnType |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | Not shown | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | NULL | ProductID | INTEGER |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | VIEW | ProductName | VARCHAR(50) |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | VIEW | ProductDescription | VARCHAR(255) |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | VIEW | CategoryID | INTEGER |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | VIEW | SerialNumber | VARCHAR(50) |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | VIEW | UnitPrice | DECIMAL(12,2) |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | VIEW | ReorderLevel | INTEGER |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | VIEW | VIEW | LeadTime | VARCHAR(30) |

Note: Path $1 = /shared/ASAssets/Utilities/repository/examples/source

### getBasicResourceCursor\_SQL\_TABLE\_FOREIGNKEYS

This procedure retrieves the foreign key resource metadata for a given resource that is a SQL TABLE/VIEW. A cursor of metadata is returned. This procedure invokes 2 lower level API procedures:

* repository/lowerLevelProcedures/getBasicResourceXML - this performs the actual invocation to the CIS repository API and returns XML
* repository/lowerLevelProcedures/getBasicResourceXSLT\_SQL\_FOREIGNKEYS - this procedure takes the XML response turns it into a cursor which is more usable by other CIS procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | VARCHAR(4096) |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (  name VARCHAR(255),  primaryKeyName VARCHAR(255),  primaryKeyTable VARCHAR(4000),  foreignKeyColumnName BIT,  primaryKeyColumnNameVARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/examples/repository/source/ds\_inventory/tutorial/products’ |
| IN | resourceType | ‘TABLE’ |
| OUT | resourceCursor | Name: fk\_category\_key  primaryKeyName: categories\_pkey  primaryKeyTable: /shared/ASAssets/Utilities/examples/repository/source/ds\_inventory/tutorial /categories  foreignKeyColumnName: categoryid  primaryKeyColumnName: categoryid |

### getBasicResourceCursor\_SQL\_TABLE\_SQLINDEXES

This procedure retrieves the resource metadata for a given resource that is a SQL TABLE/VIEW. A cursor of metadata is returned. This procedure invokes 2 lower level API procedures:

* repository/lowerLevelProcedures/getBasicResourceXML - this performs the actual invocation to the CIS repository API and returns XML
* repository/lowerLevelProcedures/getBasicResourceXSLT\_SQL\_TABLE - this procedure takes the XML response turns it into a cursor which is more usable by other CIS procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (resourceName VARCHAR(255),  sqlIndexName VARCHAR(255),  sqlIndexType VARCHAR(255),  sqlIndexUnique BIT,  sqlIndexColName VARCHAR(255),  sqlIndexColOrder VARCHAR(255) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/PRODUCT\_VIEW’ |
| IN | resourceType | ‘TABLE’ |
| OUT | resourceCursor | See table below for resourceCursor example output |

* 1. resourceCursor example output:
     1. Note: Path $1 = /shared/ASAssets/Utilities/repository/examples/source

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| sqlIndexName | sqlIndexType | sqlIndexUnique | sqlIndexColName | sqlIndexColOrder |
| productPK | PRIMARY\_KEY | 1 | ProductID | ASCENDING |

Note: Path $1 = /shared/ASAssets/Utilities/repository/examples/source

### getBasicResourceCursor\_XSLT\_TEXT

This procedure retrieves the resource metadata for a given resource that is an XSLT procedure. A cursor of metadata is returned. This procedure invokes 2 lower level API procedures:

* repository/lowerLevelProcedures/getBasicResourceXML - this performs the actual invocation to the CIS repository API and returns XML
* repository/lowerLevelProcedures/getBasicResourceXSLT\_XSLT\_TEXT - this procedure takes the XML response and turns it into a cursor which is more usable by other CIS procedures.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (name VARCHAR(255),  "path" VARCHAR(32768),  "type" VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  explicitlyDesigned BIT,  transformSourcePath VARCHAR(32768),  transformSourceType VARCHAR(255),  xsltText VARCHAR(32768) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/lowerLevelProcedures/resourceExistsXSLT’ |
| IN | resourceType | ‘PROCEDURE’ |
| OUT | resourceCursor | See table below for resourceCursor example output |

* 1. resourceCursor example output:

|  |  |
| --- | --- |
| name: | resourceExistsXSLT |
| path: | /shared/ASAssets/Utilities/repository/lowerLevelProcedures/resourceExistsXSLT |
| type: | PROCEDURE |
| subtype: | XSLT\_TRANSFORM\_PROCEDURE |
| enabled: | 1 |
| explicitlyDesigned: | 0 |
| transformSourcePath: | /services/webservices/system/admin/resource/operations/resourceExists |
| transformSourceType: | PROCEDURE |
| xsltText: | <xslt:stylesheet version="1.0" xmlns:csw-xform="http://www.compositesw.com/2003/xform" xmlns:ns1="http://www.compositesw.com/services/system/admin/resource" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xslt="http://www.w3.org/1999/XSL/Transform"> <xslt:template match="/"> <xslt:variable name="\_resourceExists"/> <xslt:element name="results"> <xslt:for-each select="ns1:resourceExistsResponse"> <xslt:variable name="\_resourceExists" select="ns1:exists"/> <xslt:element name="result"> <xslt:element name="resourceExists"> <xslt:value-of select="$\_resourceExists"/> </xslt:element> </xslt:element> </xslt:for-each> </xslt:element> </xslt:template> </xslt:stylesheet> |

### getChildResourcesCursor

This procedure retrieves the child metadata for a given resource. A cursor of metadata is returned. This procedure invokes 2 lower level API procedures:

* repository/lowerLevelProcedures/getChildResourcesXML - this performs the actual invocation to the CIS repository API and returns XML
* repository/lowerLevelProcedures/getChildResourcesXSLT - this procedure takes the XML response and turns it into a cursor which is more usable by other CIS procedures..

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | resourceCursor | CURSOR (resourceName VARCHAR(255),  resourcePath VARCHAR(1024),  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  isNullable VARCHAR(255),  columnName VARCHAR(255),  columnType VARCHAR(255),  nativeBaseType VARCHAR(255),  nativeType VARCHAR(255) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/definitions’ |
| IN | resourceType | ‘CONTAINER’ |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | resource Type | subtype | enabled | isNullable | columnName | columnType | ntive BaseType | naive Type |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | ProductID | INTEGER | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | ProductName | VARCHAR(50) | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | ProductDescription | VARCHAR(255) | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | CategoryID | INTEGER | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | SerialNumber | VARCHAR(50) | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | UnitPrice | DECIMAL(12,2) | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | ReorderLevel | INTEGER | [NULL] | [NULL] |
| PRODUCT\_VIEW | $1/PRODUCT\_VIEW | TABLE | SQL\_TABLE | 1 | [NULL] | LeadTime | VARCHAR(30) | [NULL] | [NULL] |
| foder1 | $1/PRODUCT\_VIEW | CONTAINER | FOLDER\_CONTAINER | 1 | [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |
| folder2 | $1/PRODUCT\_VIEW | CONTAINER | FOLDER\_CONTAINER | 1 | [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |
| poc1 | $1/PRODUCT\_VIEW | PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 1 | [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |

Note: Path $1 = /shared/ASAssets/Utilities/repository/examples/target

### getCisVersion (Custom Function)

This function returns version (including patch/hotfix) that the current instance of CIS is running.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | VARCHAR(15) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | ‘6.0.0.01.05’ |

### getConnectors

This procedure retrieves metadata for all the configured JMS connectors.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | name | VARCHAR(100) |
| OUT | resourceCursor | CURSOR (  name VARCHAR(1024),   annotation VARCHAR(1024),   connectorType VARCHAR(1024),   groupName VARCHAR(1024),   jmsClientID VARCHAR(1024),   jndiContextFactory VARCHAR(1024),   jndiProperties LONGVARCHAR,   jndiProviderUrl VARCHAR(1024),   jndiUser VARCHAR(50),   jndiPassword VARCHAR(50),   maxPool INTEGER,   minPool INTEGER,   poolTimeout INTEGER,   queueConnectionFactory VARCHAR(1024),   useJNDI VARCHAR(50)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | name | 'myMQ' |
| OUT | resourceCursor | (Result too large to display here.) |

### getContainer

This procedure retrieves information about the container of the specified resource.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inResourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | inResourceType | /lib/resource/ResourceDefs.ResourceType |
| OUT | parentPath | /lib/resource/ResourceDefs.ResourcePath |
| OUT | parentType | /lib/resource/ResourceDefs.ResourceType |
| OUT | parentSubtype | /lib/resource/ResourceDefs.ResourceType |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inResourcePath | '/shared/examples/CompositeView' |
| IN | inResourceType | 'TABLE' |
| OUT | parentPath | '/shared/examples' |
| OUT | parentType | 'CONTAINER' |
| OUT | parentSubtype | 'FOLDER\_CONTAINER' |

### getDataSourceAttributeDefs

The purpose of this procedure is to get the data source attribute definitions. If the model/SYS\_DATASOURCE\_ATTRIBUTE\_DEFS exists this procedure will use that or it will call the API:

Option 1 if exist: /services/databases/system/model/SYS\_DATASOURCE\_ATTRIBUTE\_DEFS

Option 2: /services/webservices/system/admin/resource/operations/getDataSourceAttributeDefs

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y-debug on. N=debug off. | CHAR(1) |
| IN | adapterName – The adapter name which is shown on the "Basic" tab of a data source as "Adatper Name". e.g. Microsoft SQL Server 2019 | VARCHAR |
| IN | propertyName – The adapter property name. If null, then return all properties for the adapter name. otherwise provide a name such as login, urlPatternStr etc. | VARCHAR |
| OUT | result  -- SYS\_DATASOURCE\_ATTRIBUTE\_DEFS -- getDataSourceAttributeDefs  ADAPTER\_NAME VARCHAR(255), -- adapterName  ADAPTER\_TYPE VARCHAR(255), -- null  ADAPTER\_TYPE\_CATEGORY VARCHAR(255), -- null  DEFINITION\_NAME VARCHAR(255), -- common:name  DISPLAY\_NAME VARCHAR(255), -- common:displayName  DEFINITION\_TYPE VARCHAR(255), -- common:type  REQUIRED BIT, -- common:required  DEFAULT\_VALUE VARCHAR(255), -- common:defaultValue  ALLOWED\_VALUES VARCHAR(255), -- common:allowedValues/common:item  EDITOR\_HINT VARCHAR(255), -- common:editorHint  IS\_ADVANCED BIT, -- if present common:parentName=advancedTab  DISPLAY\_PARENT\_NAME VARCHAR(255), -- common:parentName  DEPENDENCY\_EXPRESSION VARCHAR(255), -- common:dependencyExpression  UPDATE\_RULE VARCHAR(255), -- common:updateRule  ANNOTATION VARCHAR(255), -- common:annotation  DEFINITION\_PARENT\_NAME VARCHAR(255), -- null  GROUP\_NAME VARCHAR(255), -- null  IS\_VISIBLE BIT -- common:visible | |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | adapterName | Microsoft SQL Server 2019 |
| IN | propertyName | urlPort |
| OUT | result   |  |  | | --- | --- | | ADAPTER\_NAME: | Microsoft SQL Server 2019 | | ADAPTER\_TYPE: | SqlServer | | ADAPTER\_TYPE\_CATEGORY: | RELATIONAL | | DEFINITION\_NAME: | urlPort | | DISPLAY\_NAME: | Port | | DEFINITION\_TYPE: | INTEGER | | REQUIRED: | 0 | | DEFAULT\_VALUE: | 1433 | | ALLOWED\_VALUES: | [NULL] | | EDITOR\_HINT: | TEXT | | IS\_ADVANCED: | 0 | | DISPLAY\_PARENT\_NAME: | [NULL] | | DEPENDENCY\_EXPRESSION: | [NULL] | | UPDATE\_RULE: | READ\_WRITE | | ANNOTATION: | [NULL] | | DEFINITION\_PARENT\_NAME: | [NULL] | | GROUP\_NAME: | [NULL] | | IS\_VISIBLE: | [NULL] | | |

### getDataSourceAttributes

This procedure returns the attributes of a data source in a cursor.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dataSourcePath - Path to the data source.  Values: Any valid data source path | /ResourceDefs.ResourcePath |
| OUT | result | CURSOR (  attrName VARCHAR(32768),  attrValue VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dataSourcePath | '/shared/ASAssets/Utilities/repository/RepoUtils' |
| OUT | resourceCursor | |  |  | | --- | --- | | attrName | attrValue | | autoAddChildren | true | | cacheStatusAndTrackingDS | null | | classpath | null | | creationDate | 1376679119330 | | creatorUserDomain | composite | | creatorUserName | admin | | lastModifiedDate | 1521062814036 | | lastModifiedUserDomain | composite | | lastModifiedUserName | admin | | reintrospection\_recommended | true | | selectionState | 1 | | type | CustomProc | | url | file:///C:/MySW/TDV7.0.8/conf/customjars/RepoUtils.jar | | userDeletedSet | null | |

### getDataSourceAttributesV2

This procedure returns the attributes of a data source in a cursor using filters for exclude attribute names and whether the attribute is updatable or not. For TDV 8.3 and above it will use the /system/model tables instead of the API's to retrieve the attribute metadata..

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y-debug on. N=debug off. | CHAR(1) |
| IN | dataSourcePath – the full TDV path to the data source. | VARCHAR(4096) |
| IN | excludeAttributeList – comma-separated list of attributes to ignore with wild-card "%" optional: [password,creator%,creation%,lastModified%] | LONGVARCHAR |
| IN | isUpdatable – Y=UPDATE\_RULE=READ\_WRITE or null for any type of attribute | CHAR(1) |
| OUT | result | CURSOR (  attrName VARCHAR(32768),  attrValue VARCHAR(32768),  updateRule VARCHAR,  attrType VARCHAR,  adapterName VARCHAR ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Debug | Y |
| IN | dataSourcePath | /shared/A\_TEST1/DS\_sqlserver\_2016\_V1 |
| IN | excludeAttributeList | creator%,creation%,lastModified% |
| IN | isUpdatable | Y |
| OUT | result | Results too large to display |

### getDataSourceCacheConfig

Returns the cache status and cache tracking table paths of a data source. Returns NULL for both output values if the data source is not configured for caching.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDataSourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| OUT | StatusTablePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| OUT | TrackingTablePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inDataSourcePath | ‘/shared/examples/ds\_orders’ |
| OUT | StatusTablePath | ‘/shared/examples/ds\_orders/cache\_status’ |
| OUT | TrackingTablePath | ‘/shared/examples/ds\_orders/cache\_tracking’ |

### getDatasourceListByType

This procedure returns a list of data source resources based on the "listType" filter that is passed in.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - N=debug off. Y=debug on. | CHAR(1) |
| IN | listType - This provides a way of filtering the result set. It may be a comma-separated list of 1 or more list types.  A=all data source resources including the data source. A is the equivalent of (D,C,S,R,T,P,L,TR,TP).  D=data source list  C=catalog list  S=schema list  R=resource list [all resources under a schema]  T=database table list  P=database procedure list  L=link list which comes from published resources  TR=tree list for XML type resources  TP=database tables and procedures only | VARCHAR |
| IN | startingPath - The starting path in which to create the list. This may be a path from the published, shared or users section of TDV | VARCHAR(4000) |
| OUT | result | OUT result PIPE (  name VARCHAR(255),  resPath VARCHAR(4096),  resType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  changeid INTEGER,  version VARCHAR(255),  introspectState VARCHAR(255),  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  annotation LONGVARCHAR,  explicitlyDesigned BIT,  isFunction BIT,  tableType VARCHAR(255),  sqlText LONGVARCHAR,  scriptText LONGVARCHAR,  proprietaryModel LONGVARCHAR,  childCount INTEGER,  dataSourceType VARCHAR(255),  creationDate VARCHAR(255),  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | listType | D,S |
| IN | startingPath | /shared/myproject |

### getDataSourceRootPath

This procedure returns the file root/url path for a given file datasource path. The root/url path is the actual file system path when no file system security is being used. When file system security is in place, it is the root name mapping.

Output: fileRootPath - The "root" path to the file. If file system security is in place then this will be the root name mapping value. Example values:

No file system security: C:\files\myfile.txt

With file system security: MY\_FILES\_ROOT\_NAME

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dsPath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | debug | CHAR(1) |
| OUT | fileRootPath | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dsPath | ‘/shared/myproject/ds\_files’ |
| IN | debug | ‘Y’ |
| OUT | fileRootPath | ‘C:\files\myfile.txt’ |

### getDataSourceStatsConfig

Retrieve the statistics configuration for a given data source.

Usage Note: The calling user must have:   
 - The ACCESS\_TOOLS right   
 - Read permission on the data source set  
 - Read permission on any of the data source’s parent folders

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | dsPath | VARCHAR(4096) |
| OUT | curs | CURSOR (  configured BIT,  useEnabled BIT,  tableGatherDefault VARCHAR,  numThreads INTEGER,  maxTime INTEGER,  refreshMode VARCHAR,  scheduleMode VARCHAR,  startTime TIMESTAMP,  fromTimeInADay BIGINT,  endTimeInADay BIGINT,  recurringDay INTEGER,  interval INTEGER,  period VARCHAR,  count INTEGER,  isCluster BIT, ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dsPath | '/shared/examples/ds\_orders' |
| OUT | result | (0, NULL, …) |

### getDefSetDefs

Dumps the contents of an SQL definition set to a cursor.

Usage Note: The calling user must have:   
 - The ACCESS\_TOOLS right   
 - Read permission on the definition set  
 - Read permission on any of the definition set’s parent folders

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | defSetPath | VARCHAR(4096) |
| OUT | result | CURSOR (  defName VARCHAR(32768),  defType VARCHAR(32768),  dataType VARCHAR(32768),  defValue VARCHAR(32768) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | defSetPath | ‘/lib/utils/System’ |
| OUT | result | (‘CannotExecuteSelectException’, ‘EXCEPTION\_DEFINITION’, NULL, NULL),  …  (‘CANCELED’, ‘CONSTANT\_DEFINITION’, ‘VARCHAR(255)’, ‘CANCELED’),  …  (‘CONTENT’, ‘TYPE\_DEFINITION’, ‘VARCHAR(65536)’, NULL),  … |

### getDependentResourcesCursor

This procedure retrieves the immediately dependent metadata for a given resource. A cursor of metadata is returned. This procedure invokes repository/lowerLevelProcedures/getDependentResourcesXSLT.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | fullResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1) |
| OUT | | resourceCursor | CURSOR (  resourceName VARCHAR(255),  resourcePath VARCHAR(4096),  resourceType VARCHAR(40),  subtype VARCHAR(40),  id INTEGER,  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  enabled BIT  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/examples/ds\_orders/orders’ |
| IN | resourceType | ‘TABLE’ |
| OUT | resourceCursor | (‘ViewOrder’,  ‘/shared/examples/ViewOrder’,  ‘TABLE’,  ‘SQL\_TABLE’,  1003  ‘composite’,  ‘admin’,  ‘NONE’,  NULL,  1) |

### getDependentResourcesRecurseCursor

This procedure is similar to repository/getDependentResourcesCursor() (and in fact uses it to generate its results), however it recursively returns all the resource’s dependents instead of just the immediate dependents.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | fullResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1) |
| OUT | | resourceCursor | CURSOR (  resourceName VARCHAR(255),  resourcePath VARCHAR(4096),  resourceType VARCHAR(40),  subtype VARCHAR(40),  id INTEGER,  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  enabled BIT  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/examples/ds\_orders/orders’ |
| IN | resourceType | ‘TABLE’ |
| OUT | resourceCursor | (‘ViewOrder’,  ‘/shared/examples/ViewOrder’,  ‘TABLE’,  ‘SQL\_TABLE’,  1003  ‘composite’,  ‘admin’,  ‘NONE’,  NULL,  1),  … |

### getDependentResourcesDirectCursor

This procedure retrieves the "direct" dependent resource metadata for a given resource. If a dependent resource is a foreign key reference to another view it is not returned. If a dependent resource is a data source reference, it is not returned. A cursor of metadata is returned. This procedure invokes repository/lowerLevelProcedures/getDependentResourcesXSLT.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | fullResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1) |
| OUT | | resourceCursor | CURSOR (  resourceName VARCHAR(255),  resourcePath VARCHAR(4096),  resourceType VARCHAR(40),  subtype VARCHAR(40),  id INTEGER,  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  enabled BIT  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/examples/ds\_orders/orders’ |
| IN | resourceType | ‘TABLE’ |
| OUT | resourceCursor | (‘ViewOrder’,  ‘/shared/examples/ViewOrder’,  ‘TABLE’,  ‘SQL\_TABLE’,  1003  ‘composite’,  ‘admin’,  ‘NONE’,  NULL,  1) |

### getDependentResourcesDirectRecurseCursor

This procedure recursively retrieves the "direct" dependent resource metadata for a given resource. If a dependent resource is a foreign key reference to another view it is not returned. If a dependent resource is a data source reference, it is not returned. A cursor of metadata is returned.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | fullResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1) |
| IN | | includeInitialPath | SMALLINT |
| IN | | inLineageResourceIdList | LONGVARCHAR |
| OUT | | resourceCursor | CURSOR (  resourceName VARCHAR(255),  resourcePath VARCHAR(4096),  resourceType VARCHAR(40),  subtype VARCHAR(40),  id INTEGER,  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  enabled BIT  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/examples/ds\_orders/orders’ |
| IN | resourceType | ‘TABLE’ |
| IN | includeInitialPath | 1 |
| IN | inLineageResourceIdList | null |
| OUT | resourceCursor | (‘ViewOrder’,  ‘/shared/examples/ViewOrder’,  ‘TABLE’,  ‘SQL\_TABLE’,  1003  ‘composite’,  ‘admin’,  ‘NONE’,  NULL,  1),  … |

### getImpactedResources

This procedure takes a folder path as input, walks the tree of resources in the input folder, and reports on any impacted resources. If no resources are marked as impacted, an empty result set will be returned.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | startingFolder | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| OUT | result | CURSOR (  ResourcePath VARCHAR(4096),  ResourceType VARCHAR(40),  ImpactLevel VARCHAR(32768),  ImpactMessage VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/examples’ |
| OUT | resourceCursor | (  ‘/shared/examples/NewView’,  ‘TABLE’,  ‘UNKNOWN’,  ‘View is newly created and has not been saved.’ ), … |

### getIntrospectableResourceIdsResult

This procedure gathers the results from a call to repository/getIntrospectableResourceIdsTask(). If the introspection task is still running, the procedure can be called in such a way as to block execution until the task completes before returning results.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | taskId | LONGVARCHAR |
| IN | | block | BIT |
| IN | | pageSize | INTEGER |
| IN | | pageStart | INTEGER |
| OUT | | result | CURSOR (  totalResults INTEGER,  completed BIT,  lastUpdate TIMESTAMP,  "path" VARCHAR(4096),  "type" VARCHAR(40),  subtype VARCHAR(40)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | taskId | '1000' |
| IN | block | 1 |
| IN | pageSize | NULL |
| IN | pageStart | NULL |
| OUT | result | (  8, 1, '2012-11-01 10:45:00', 'cache\_status', 'TABLE', 'DATABASE\_TABLE'  ),  … |

### getIntrospectableResourceIdsTask

This procedure begins an asynchronous thread that scans a data source for introspectable objects. This asynchronous thread will survive server restarts. This is used to populate the introspection cache and may take some time to run. Use repository/getIntrospectableResourceIdsResult() to retrieve the results.

**Input:**

dsPath - The path to the data source.   
 Values: Any valid data source path.

dsContainerPath - The relative path to the data source container to begin introspection.   
 Values: Any relative path (i.e. 'mySchema' or 'myCatalog/mySchema'). May be NULL to indicate the entire data source should be scanned.

dsContainerType - The type of the data source container to begin introspection.   
 Values: This will nearly always be 'CONTAINER' (see /lib/resource/ResourceDefs for other types.)

dsContainerSubType - The subtype of the data source container to begin introspection.   
 Values: Any container sub-type (see /lib/resource/ResourceDefs.)

recurse - indicates whether introspection should be recursive or not.   
 Values: 1 or 0

**Output:**

taskId - The introspection task ID. Use this with repository/getIntrospectableResourceIdsResult().  
 Values: A task ID

totalResults - Total size of the result set (if known.)  
 Values: A positive integer or NULL.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | dsPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | | dsContainerPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | | dsContainerType | /lib/resource/ResourceDefs.ResourceType |
| IN | | dsContainerSubType | /lib/resource/ResourceDefs.ResourceType |
| IN | | recurse | BIT |
| OUT | | taskId | VARCHAR(32768) |
| OUT | | totalResults | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dsPath | '/shared/examples/ds\_orders' |
| IN | dsContainerPath | NULL |
| IN | dsContainerType | NULL |
| IN | dsContainerSubType | NULL |
| IN | recurse | 0 |
| OUT | taskId | '1000' |
| OUT | totalResults | NULL |

### getIntrospectedResourceIdsResult

This procedure gathers the results from a call to repository/getIntrospectedResourceIdsTask(). If the introspection task is still running, the procedure can be called in such a way as to block execution until the task completes before returning results.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | taskId | LONGVARCHAR |
| IN | | block | BIT |
| IN | | pageSize | INTEGER |
| IN | | pageStart | INTEGER |
| OUT | | result | CURSOR (  totalResults INTEGER,  completed BIT,  lastUpdate TIMESTAMP,  "path" VARCHAR(4096),  "type" VARCHAR(40)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | taskId | '1000' |
| IN | block | 1 |
| IN | pageSize | NULL |
| IN | pageStart | NULL |
| OUT | result | (  8, 1, '2012-11-01 10:45:00', 'cache\_status', 'TABLE'  ),  … |

### getIntrospectedResourceIdsTask

This procedure begins an asynchronous thread that scans a data source for introspected objects. This asynchronous thread will survive server restarts. Use repository/getIntrospectedResourceIdsResult() to retrieve the results.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | dsPath | /lib/resource/ResourceDefs.ResourcePath |
| OUT | | taskId | VARCHAR(32768) |
| OUT | | totalResults | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dsPath | '/shared/examples/ds\_orders' |
| OUT | taskId | '1000' |
| OUT | totalResults | NULL |

### getLockedResources

This procedure is used to generate a string containing the output columns returned by a view or procedure for easy definition of a custom row type. The procedure returns the full list of output colums for a table or view and the full list of out and inout columns for a procedure. The procedure can also strip out column definitions of any output cursors and include them in the list of returned columns.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | detail | VARCHAR(6) |
| IN | | includeOnlyUnlockableResources | VARCHAR(5) |
| OUT | | result | CURSOR (  name VARCHAR(32768),  "path" VARCHAR(32768),  "type" VARCHAR(32768),  subtype VARCHAR(32768),  id VARCHAR(32768),  changeId INTEGER,  version VARCHAR(32768),  introspectState VARCHAR(32768),  ownerDomain VARCHAR(32768),  ownerName VARCHAR(32768),  impactLevel VARCHAR(32768),  impactMessage VARCHAR(32768),  enabled BIT,  annotation VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | detail | ‘FULL’ |
| IN | includeOnlyUnlockableResources | ‘FALSE’ |
| OUT | result | (  ‘CompositeView’,  ‘/shared/examples/CompositeView’,  ‘TABLE’,  ‘SQL\_TABLE’,  21874,  14124,  NULL,  NULL,  ‘composite’,  ‘admin’,  ‘NONE’,  NULL,  1,  NULL ) |

### getOutputColDefs

This procedure is used to generate a string containing the output columns returned by a view or procedure for easy definition of a custom row type. The procedure returns the full list of output colums for a table or view and the full list of out and inout columns for a procedure. The procedure can also strip out column definitions of any output cursors and include them in the list of returned columns.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | fullResourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | | resourceType | /lib/resource/ResourceDefs.ResourceType |
| IN | | convertCursorsToCols | BIT |
| OUT | | rowDef | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResroucePath | '/shared/examples/ds\_orders/customers' |
| IN | resourceType | 'TABLE' |
| IN | convertCursorsToCols | 0 |
| OUT | rowDef | ' CustomerID INTEGER, CompanyName VARCHAR(50), ContactFirstName VARCHAR(30), ContactLastName VARCHAR(50), BillingAddress VARCHAR(255), City VARCHAR(50), StateOrProvince VARCHAR(20), PostalCode VARCHAR(20), CountryRegion VARCHAR(50), ContactTitle VARCHAR(50), PhoneNumber VARCHAR(30), FaxNumber VARCHAR(30)' |

### getResourceAnnotations

This procedure returns the annotations of a resource and its columns (if any.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40)) |
| OUT | result | CURSOR TypeDefinitions.ColumnAnnotationRow |

1. **Examples:**
   1. **Assumptions: none.**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples' |
| IN | resourceType | 'CONTAINER' |
| OUT | result | |  |  | | --- | --- | | columnName: | NULL | | columnType: | NULL | | annotation: | 'This folder contains pre-created resources.' | |

### getResourceCacheConfig

This procedure returns the metadata for a resource that has caching configured.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | cacheConfigured | VARCHAR(255) |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source’ |
| IN | resourceType | ‘PROCEDURE’ |
| OUT | cacheConfigured | ‘true’ or ‘false’ |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### getResourceCacheConfigCursor

Expands on repository/getResourceCacheConfig by providing more detailed information about a resource’s caching configuration.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | inType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1) |
| OUT | result | CURSOR (  configured BIT,   enabled BIT,   storageMode VARCHAR(32768),   storageDataSourcePath VARCHAR(32768),   storageTargetName VARCHAR(32768),   storagePath VARCHAR(32768),   storageType VARCHAR(32768),   refreshMode VARCHAR(32768),   scheduleMode VARCHAR(32768),   startTime TIMESTAMP,   fromTimeInADay BIGINT,   endTimeInADay BIGINT,   recurringDay INTEGER,   "interval" INTEGER,   period VARCHAR(32768),   "count" INTEGER,   isCluster BIT,   expirationPeriod BIGINT,   clearRule VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inResourcePath | ‘/shared/examples/ds\_orders/orders’ |
| IN | inType | ‘TABLE’ |
| OUT | result | |  |  | | --- | --- | | configured: | 1 | | enabled: | 1 | | storageMode: | DATA\_SOURCE | | storageDataSourcePath: | /shared/examples/ds\_orders | | storageTargetName: | result | | storagePath: | /shared/examples/ds\_orders/orders\_cache | | storageType: | TABLE | | refreshMode: | MANUAL | | scheduleMode: | [NULL] | | startTime: | [NULL] | | fromTimeInADay: | [NULL] | | endTimeInADay: | [NULL] | | recurringDay: | [NULL] | | interval: | [NULL] | | period: | [NULL] | | count: | [NULL] | | isCluster: | [NULL] | | expirationPeriod: | 0 | | clearRule: | NONE | |

### getResourceCreated

This procedure returns the creation date of a resource. If no creation date is recorded, then NULL is returned.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inResourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | inResourceType | /lib/resource/ResourceDefs.ResourceType |
| OUT | created | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inResourcePath | '/shared/examples/CompositeView' |
| IN | inResourceType | 'TABLE' |
| OUT | created | NULL |

### getResourceImpactedCursor

Returns the impacted information for a resource (if any.) A resource may be impacted because of a syntax error of its code, a missing dependent resource, caching misconfiguration, etc.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | inType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1) |
| OUT | result | CURSOR (  impactLevel VARCHAR(32768),   impactMessage VARCHAR(32768) ) |

1. **Examples:**
   1. **Assumptions: A new view created in /shared/examples/NewView that has not been edited or saved.**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inResourcePath | ‘/shared/examples/NewView’ |
| IN | inType | ‘TABLE’ |
| OUT | result | |  |  | | --- | --- | | impactLevel: | UNKNOWN | | impactMessage: | View is newly created and has not been saved. | |

### getResourceLastModified

This procedure returns the last modified date of a resource. If no last modified date is recorded then the creation date is returned. If no last modified or creation date is recorded, then NULL is returned.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inResourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | inResourceType | /lib/resource/ResourceDefs.ResourceType |
| OUT | lastModified | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inResourcePath | '/shared/examples/CompositeView' |
| IN | inResourceType | 'TABLE' |
| OUT | lastModified | 2014-03-28 15:44:48.258 |

### getResourceLineageDatasources

Return the list of Data Sources used by a given resource path.

1. Start with the resource path and recursively walk its lineage until no more resources are found.
2. For each resource found, determine if that resources has as associated resource of type=DATA\_SOURCE.
3. Find the distinct list of data sources for this resource.
4. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **resourcePath** – CIS source resource path to being assessing the parent data lineage | pathType |
| IN | **resourceType** – Type of CIS resource to be created. | VARCHAR |
| IN | **excludePathsList** – comma separated list of resource paths or partials paths to exclude | LONGVARCHAR |
| IN | **datasourceAncestor** – flag to get the data source ancestors. 1=yes, 0=no | INTEGER |
| IN | **inIgnoreResourceDoesNotExist –** flag to ignore missing resources. 1=yes, 0=no | INTEGER |
| OUT | **datasourceResource** | CURSOR (  id INTEGER,  resourceName VARCHAR(255),  resourcePath pathType,  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  childCount INTEGER,  datasourceType VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples/ViewSales’ |
| IN | resourceType | TABLE |
| IN | excludePathsList | null |
| IN | datasourceAncestor | 1 |
| IN | inIgnoreResourceDoesNotExist | 1 |
| OUT | datasourceResource | See charts below |

* 1. Chart 1: Columns (1-5)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | resourceName | resourcePath | resourceType | subtype |
| 20670 | ds\_orders | /shared/examples/ds\_orders | DATA\_SOURCE | RELATIONAL\_DATA\_SOURCE |
| 20756 | ds\_XML | /shared/examples/ds\_XML | DATA\_SOURCE | XML\_FILE\_DATA\_SOURCE |

* 1. Chart 2: Columns (6-8)

|  |  |  |
| --- | --- | --- |
| enabled | childCount | datasourceType |
| 1 | 1 | PostgreSQL 9.1 |
| 1 | 7 | FileXML |
|  |  |  |

### getResourceLineageRecursive

This procedure recursively walks the dependent tree to discover resource lineage.

This procedure uses the resource ID to show the lineage by returning the correlation of the resourceID and the parentID. The parentID refers back to the resourceID. The lineage is discovered as the procedure recursively walks the tree. Additionally, the for each child falling within a set of conditions, the ancestor of that child is returned to find out if it has any DATA\_SOURCE types in its upstream lineage. That information is returned along with the resource record. This is a flattened way of returning the information all in one record rather than making a separate call to get this information outside of this procedure. However, the user of this procedure may determine that they do not want to calculate ancestors so there is an option to turn that off.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **inSeqNum** – sequence number starting with 1. | INTEGER |
| IN | **inParentID** – the id of the parent. | INTEGER |
| IN | **inResourceDepth** – the depth of the parent (number of levels which recursion has occurred). | INTEGER |
| IN | **inLineageResourceIdList** - a list of space separated resource ids built up as the resources are traversed. | LONGVARCHAR |
| IN | **resourcePath** - CIS source path to an actual resource | pathType |
| IN | **resourceType** - Type of CIS resource to be created. It is null on the first invocation. | VARCHAR |
| IN | **excludePathsList** - comma separate list of resource paths or partials paths to exclude | LONGVARCHAR |
| IN | **datasourceAncestor** - flag to get the data source ancestor | INTEGER |
| IN | **inIgnoreResourceDoesNotExist –** flag to ignore missing resources | INTEGER |
| OUT | resourceTreeList | CURSOR lineageTreeType (  seqNum INTEGER, -- generated sequence number  resourceID INTEGER, -- resource id from CIS  parentID INTEGER, -- how this row relates to resourceID  resDepth INTEGER, -- depth as related to the start  treeType VARCHAR(255), -- Parent, Child  resName VARCHAR(255), -- resource name  resPath pathType, -- resource path  resType VARCHAR(255), -- resource type  subType VARCHAR(255), -- resource sub type  enabled BIT, -- enabled or not (1 or 0)  dsID INTEGER, -- datasource ancestor id  dsResName VARCHAR(255), -- datasource ancestor name  dsResPath pathType, -- datasource ancestor path  dsResType VARCHAR(255), -- datasource ancestor type  dsResSubType VARCHAR(255), -- datasource ancestor sub type  dsEnabled BIT, -- datasource ancestor enabled (1 or 0)  dsChildCount INTEGER -- datasource ancestor number of children  datasourceType VARCHAR(255),  creationDate VARCHAR(255),  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ); |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inSeqNum | null |
| IN | inParentID | null |
| IN | inResourceDepth | null |
| IN | resourcePath | /shared/examples/LookupProduct |
| IN | resourceType | PROCEDURE |
| IN | excludePathsList | null |
| IN | datasourceAncestor | 1 |
| IN | inIgnoreResourceDoesNotExist | 1 |
| OUT | resourceTreeList | See charts below |

* 1. Chart 1: Columns (1-6)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| seqnum | resourceID | parentID | resDepth | treeType | resName |  |  |
| 1 | 20760 | [NULL] | 0 | Parent | LookupProduct |  |  |
| 2 | 20633 | 20760 | 1 | Child | products |  |  |

* 1. Chart 2: Columns (7-10)

|  |  |  |  |
| --- | --- | --- | --- |
| resPath | resType | subType | enabled |
| /shared/examples/LookupProduct | PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 1 |
| /shared/examples/ds\_inventory/products | TABLE | DATABASE\_TABLE | 1 |

* 1. Chart 3: Columns (11-15)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dsID | dsResName | dsResPath | dsResType | dsResSubType |
| [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |
| 20605 | ds\_inventory | /shared/examples/ds\_inventory | DATA\_SOURCE | RELATIONAL\_DATA\_SOURCE |

* 1. Chart 4: Columns (16-18)

|  |  |  |
| --- | --- | --- |
| dsEnabled | dsChildCount | datasourceType |
| [NULL] | [NULL] | [NULL] |
| 1 | 4 | PostgreSQL 9.1 |
|  |  |  |

### getResourceLineageDependencies

This procedure is used to produce the dependency metadata for a given input resource path.

It returns the datasource id, path and dependency resource path along with the lineage for that path. For each new datasource found, it will return a row with the aforementioned results. What makes this procedure different than getResourceLineageRecursive() and getResourceLineageDirectRecursive() is that it has the “From Path” and “To Path” for the lineage on the same result line whereas those procedures return the results on separate lines.

The input resource path may be from /services/databases/, /services/webservices or /shared/ paths. This procedure uses /lib/resource/GetColumnDependencies when a resource path is of type LINK or TABLE. When a resource path is PROCEDURE or a PROCEDURE is found during processing of GetColumnDependencies(), the algorithm will utilize getResourceLineageDirectRecursive() to complete the lineage thus providing a full picture of lineage for resources containing TABLE and PROCEDURE types within the lineage. GetColumnDependencies() produces a derivation column list which gets processed and distilled down to unique resource paths with no columns. The derivation list is processed by getResourceLineageDependenciesDerivations.

What makes this procedure different from /lib/resource/GetColumnDependencies is that it does not return column dependencies. It is only concerned with returning resource dependencies. Therefore, a row is produced only when the dependency resource path changes.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **debug** – Y=debug on. N=debug off. | CHAR(1) |
| IN | **inResourcePath** - CIS source path to an actual resource | VARCHAR(4000) |
| IN | **inResourceType** - Type of CIS resource to be created. It is null on the first invocation. | VARCHAR |
| IN | **inResourceLinkType** - Only necessary when the resourcePath is in "/services/databases/" or "/services/webservices/".  The resource type for the resource being invoked by the published resource. [TABLE|PROCEDURE] | VARCHAR |
| IN | **inReturnAllLineage -** (default) null or Y=return all lineage path rows for the input resource path.  N=do not return all lineage. Only return a list of physical data source tables. | CHAR(1) |
| IN | **inSeqNum** – Initially is null. Sequence number starting with 1. | INTEGER |
| IN | **inResDepth** – Initially is null. The depth of the parent (number of levels which recursion has occurred). | INTEGER |
| IN | **inTrackFromToInsertedPairs** - Initially is null. Will be populated on recursive calls. | LONGVARCHAR |
| OUT | result | PIPE (  seqNum INTEGER, -- A unique sequence number starting at 1  resDepth INTEGER, -- The depth of the resource in the each branch of the lineage tree. Also known as lineageOrder.  fromResourceId BIGINT, -- The from (source) resource id  fromResourceName VARCHAR, -- The from (source) resource name  fromResourcePath VARCHAR(4000), -- The from (source) resource path  fromResourceType VARCHAR, -- The from (source) resource type  fromResourceSubtype VARCHAR, -- The from (source) resource subtype  toResourceId BIGINT, -- The to (target) resource id  toResourceName VARCHAR, -- The to (target) resource name  toResourcePath VARCHAR(4000), -- The to (target) resource path  toResourceType VARCHAR, -- The to (target) resource type  toResourceSubtype VARCHAR, -- The to (target) resource subtype  dependencyDatasourcePath VARCHAR(4000), -- The dependency data source path  dependencyDatasourceId BIGINT, -- The dependency data source id  dependencyDatasourceType VARCHAR, -- The dependency data source type  dependencyDatasourceSubtype VARCHAR, -- The dependency data source subtype  dependencyResourcePath VARCHAR(4000), -- The dependency resource path  dependencyResourceType VARCHAR, -- The dependency resource type. Example: [TABLE|PROCEDURE|TREE]  derivations LONGVARCHAR, -- The complete lineage derivations for the given dependency resource path.  -- Delimiters: The line delimiter="!|!" The From/To path delimiter=" --> " The resource type/path delimiter=":"  -- Example:   * LINK:/services/databases/lab00/orders/CompositeView --> TABLE:/shared/lab00/Application/Views/CompositeView!|! * TABLE:/shared/lab00/Application/Views/CompositeView --> TABLE:/shared/lab00/Business/Business/ViewSupplier!|! * TABLE:/shared/lab00/Business/Business/ViewSupplier --> TABLE:/shared/labCommon/Physical/Formatting/postgres2/tutorial/suppliers!|! * TABLE:/shared/labCommon/Physical/Formatting/postgres2/tutorial/suppliers --> TABLE:/shared/labCommon/Physical/Metadata/postgres2/ds\_inventory/tutorial/suppliers!|!   -- The from (source) create and modified columns  fromCreationDate VARCHAR(255),  fromCreationDateBigint BIGINT,  fromCreatorUserId VARCHAR(255),  fromCreatorUserName VARCHAR(255),  fromCreatorUserDomain VARCHAR(255),  fromLastModifiedDate VARCHAR(255),  fromLastModifiedDateBigint BIGINT,  fromLastModifiedUserId VARCHAR(255),  fromLastModifiedUserName VARCHAR(255),  fromLastModifiedUserDomain VARCHAR(255),  -- The to (target) create and modified columns  toCreationDate VARCHAR(255),  toCreationDateBigint BIGINT,  toCreatorUserId VARCHAR(255),  toCreatorUserName VARCHAR(255),  toCreatorUserDomain VARCHAR(255),  toLastModifiedDate VARCHAR(255),  toLastModifiedDateBigint BIGINT,  toLastModifiedUserId VARCHAR(255),  toLastModifiedUserName VARCHAR(255),  toLastModifiedUserDomain VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | N |
| IN | inResourcePath | /services/databases/MY\_DB/MY\_CATALOG/MY\_SCHEMA/LookupProduct |
| IN | inResourceType | LINK |
| IN | inResourceLineType | PROCEDURE |
| IN | inSeqNum | null |
| IN | inResDepth | null |
| IN | inTrackFromToInsertedPairs | null |
| OUT | resourceTreeList | See charts below |

* 1. Result Row #1

|  |  |
| --- | --- |
| seqNum: | 1 |
| resDepth: | 1 |
| fromResourceId: | 500602 |
| fromResourceName: | LookupProduct |
| fromResourcePath: | /services/databases/MY\_DB/MY\_CATALOG/MY\_SCHEMA/LookupProduct |
| fromResourceType: | LINK |
| fromResourceSubtype: | NONE |
| toResourceId: | 21084 |
| toResourceName: | LookupProduct |
| toResourcePath: | /shared/examples/LookupProduct |
| toResourceType: | PROCEDURE |
| toResourceSubtype: | SQL\_SCRIPT\_PROCEDURE |
| dependencyDatasourcePath: | /shared/examples/ds\_inventory |
| dependencyDatasourceId: | 21105 |
| dependencyDatasourceType: | PostgreSQL 9.1 |
| dependencyDatasourceSubtype: | RELATIONAL\_DATA\_SOURCE |
| dependencyResourcePath: | /shared/examples/ds\_inventory/tutorial/products |
| dependencyResourceType: | TABLE |
| derivations: | LINK:/services/databases/MY\_DB/MY\_CATALOG/MY\_SCHEMA/LookupProduct --> PROCEDURE:/shared/examples/LookupProduct!|! PROCEDURE:/shared/examples/LookupProduct --> TABLE:/shared/examples/ds\_inventory/tutorial/products!|! |
| fromCreationDate: | 2015-04-17 12:11:38 |
| fromCreationDateBigint: | 1429287098248 |
| fromCreatorUserId: | [NULL] |
| fromCreatorUserName: | admin |
| fromCreatorUserDomain: | composite |
| fromLastModifiedDate: | 2015-06-06 10:09:05 |
| fromLastModifiedDateBigint: | 1433599745224 |
| fromLastModifiedUserId: | [NULL] |
| fromLastModifiedUserName: | admin |
| fromLastModifiedUserDomain: | composite |
| toCreationDate: | 1969-12-31 19:00:00 |
| toCreationDateBigint: | 0 |
| toCreatorUserId: | [NULL] |
| toCreatorUserName: | [NULL] |
| toCreatorUserDomain: | [NULL] |
| toLastModifiedDate: | 2014-05-27 16:11:05 |
| toLastModifiedDateBigint: | 1401221465456 |
| toLastModifiedUserId: | [NULL] |
| toLastModifiedUserName: | admin |
| toLastModifiedUserDomain: | composite |

* 1. Result Row #2

|  |  |
| --- | --- |
| seqNum: | 2 |
| resDepth: | 2 |
| fromResourceId: | 21084 |
| fromResourceName: | LookupProduct |
| fromResourcePath: | /shared/examples/LookupProduct |
| fromResourceType: | PROCEDURE |
| fromResourceSubtype: | SQL\_SCRIPT\_PROCEDURE |
| toResourceId: | 21157 |
| toResourceName: | products |
| toResourcePath: | /shared/examples/ds\_inventory/tutorial/products |
| toResourceType: | TABLE |
| toResourceSubtype: | DATABASE\_TABLE |
| dependencyDatasourcePath: | /shared/examples/ds\_inventory |
| dependencyDatasourceId: | 21105 |
| dependencyDatasourceType: | PostgreSQL 9.1 |
| dependencyDatasourceSubtype: | RELATIONAL\_DATA\_SOURCE |
| dependencyResourcePath: | /shared/examples/ds\_inventory/tutorial/products |
| dependencyResourceType: | TABLE |
| derivations: | LINK:/services/databases/MY\_DB/MY\_CATALOG/MY\_SCHEMA/LookupProduct --> PROCEDURE:/shared/examples/LookupProduct!|! PROCEDURE:/shared/examples/LookupProduct --> TABLE:/shared/examples/ds\_inventory/tutorial/products!|! |
| fromCreationDate: | 1969-12-31 19:00:00 |
| fromCreationDateBigint: | 0 |
| fromCreatorUserId: | [NULL] |
| fromCreatorUserName: | [NULL] |
| fromCreatorUserDomain: | [NULL] |
| fromLastModifiedDate: | 2014-05-27 16:11:05 |
| fromLastModifiedDateBigint: | 1401221465456 |
| fromLastModifiedUserId: | [NULL] |
| fromLastModifiedUserName: | admin |
| fromLastModifiedUserDomain: | composite |
| toCreationDate: | 1969-12-31 19:00:00 |
| toCreationDateBigint: | 0 |
| toCreatorUserId: | [NULL] |
| toCreatorUserName: | [NULL] |
| toCreatorUserDomain: | [NULL] |
| toLastModifiedDate: | [NULL] |
| toLastModifiedDateBigint: | [NULL] |
| toLastModifiedUserId: | [NULL] |
| toLastModifiedUserName: | [NULL] |
| toLastModifiedUserDomain: | [NULL] |

### getResourceLineageDirectRecursive

This procedure recursively walks the "direct" descendent tree to discover resource lineage. It only returns "direct" descendants and not foreign key descendants and not cache related descendants.

This procedure uses the resource ID to show the lineage by returning the correlation of the resourceID and the parentID. The parentID refers back to the resourceID. The lineage is discovered as the procedure recursively walks the tree. Additionally, the for each child falling within a set of conditions, the ancestor of that child is returned to find out if it has any DATA\_SOURCE types in its upstream lineage. That information is returned along with the resource record. This is a flattened way of returning the information all in one record rather than making a separate call to get this information outside of this procedure. However, the user of this procedure may determine that they do not want to calculate ancestors so there is an option to turn that off.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **inSeqNum** - sequence number starting with 1. | INTEGER |
| IN | **inParentID** - the id of the parent. | INTEGER |
| IN | **inResourceDepth** - the depth of the parent (number of levels which recursion has occurred). | INTEGER |
| IN | **inLineageResourceIdList** - null to start with. A list of space separated resource ids built up as the resources are traversed. | LONGVARCHAR |
| IN | **resourcePath** – CIS source path to an actual resource | pathType |
| IN | **resourceType** – Type of CIS resource to be created. It is null on the first invocation. | VARCHAR |
| IN | **excludePathsList** – comma separate list of resource paths or partials paths to exclude | LONGVARCHAR |
| IN | **datasourceAncestor** – flag to get the data source ancestor | INTEGER |
| IN | **inIgnoreResourceDoesNotExist –** flag to ignore missing resources | INTEGER |
| OUT | resourceTreeList | CURSOR lineageTreeType (  seqNum INTEGER, -- generated sequence number  resourceID INTEGER, -- resource id from CIS  parentID INTEGER, -- how this row relates to resourceID  resDepth INTEGER, -- depth as related to the start  treeType VARCHAR(255), -- Parent, Child  resName VARCHAR(255), -- resource name  resPath pathType, -- resource path  resType VARCHAR(255), -- resource type  ubtype VARCHAR(255), -- resource sub type  enabled BIT, -- enabled or not (1 or 0)  dsID INTEGER, -- datasource ancestor id  dsResName VARCHAR(255), -- datasource ancestor name  dsResPath pathType, -- datasource ancestor path  dsResType VARCHAR(255), -- datasource ancestor type  dsResSubType VARCHAR(255), -- datasource ancestor sub type  dsEnabled BIT, -- datasource ancestor enabled (1 or 0)  dsChildCount INTEGER -- datasource ancestor number of children  datasourceType VARCHAR(255),  creationDate VARCHAR(255),  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ); |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inSeqNum | null |
| IN | inParentID | null |
| IN | inResourceDepth | null |
| IN | resourcePath | /shared/examples/LookupProduct |
| IN | resourceType | PROCEDURE |
| IN | excludePathsList | null |
| IN | datasourceAncestor | 1 |
| IN | inIgnoreResourceDoesNotExist | 1 |
| OUT | resourceTreeList | See charts below |

### getResourceLineageRecursiveAncestors

This procedure was built as a building block with the intention to mimic the Composite Manager capability to show "Dependency Privileges".

This procedure provides the first step, which is to recursively walk the resource lineage depencecy tree to discover child resources for a given input resource. Additionally, the Ancestor folder structure is also retrieved. The reason for the ancestor folders is that privileges need to be assigned to each folder of the ancestry tree. This procedure provides the listing of those folders. The "driver" procedure "getResourcePrivilegeDependencies" sits on top of this procedure and will be responsible for retrieving privileges. Any procedure that invokes this procedure will most likely want to do a "SELECT DISTINCT" on the columns so that repeating resource paths are trimmed out.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **inSeqNum** - sequence number starting with 1. | INTEGER |
| IN | **inParentID** - the id of the parent. | INTEGER |
| IN | **inResourceDepth** - the depth of the parent (number of levels which recursion has occurred). | INTEGER |
| IN | **inLineageResourceIdList** - a list of space separated resource ids built up as the resources are traversed. | INTEGER |
| IN | **resourcePath** - CIS source path to an actual resource | pathType |
| IN | **resourceType** - Type of CIS resource to be created. It is null on the first invocation. | VARCHAR |
| IN | **excludePathsList** - comma separate list of resource paths or partials paths to exclude | LONGVARCHAR |
| IN | **datasourceAncestor** - flag to get the data source ancestor | INTEGER |
| IN | **inIgnoreResourceDoesNotExist –** flag to ignore missing resources | INTEGER |
| OUT | resourceTreeList | CURSOR lineageTreeType (  seqNum INTEGER, -- generated sequence number  resourceID INTEGER, -- resource id from CIS  parentID INTEGER, -- how this row relates to resourceID  resDepth INTEGER, -- depth as related to the start  treeType VARCHAR(255), -- Parent, Ancestor  resName VARCHAR(255), -- resource name  resPath pathType, -- resource path  resType VARCHAR(255), -- resource type  subType VARCHAR(255), -- resource sub type  enabled BIT, -- enabled or not (1 or 0)  dsID INTEGER, -- datasource ancestor id  dsResName VARCHAR(255), -- datasource ancestor name  dsResPath pathType, -- datasource ancestor path  dsResType VARCHAR(255), -- datasource ancestor type  dsResSubType VARCHAR(255), -- datasource ancestor sub type  dsEnabled BIT, -- datasource ancestor enabled (1 or 0)  dsChildCount INTEGER -- datasource ancestor number of children  datasourceType VARCHAR(255),  creationDate VARCHAR(255),  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ); |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inSeqNum | null |
| IN | inParentID | null |
| IN | inResourceDepth | null |
| IN | inLineageResourceIdList | null |
| IN | resourcePath | /shared/examples/ViewSales |
| IN | resourceType | TABLE |
| IN | excludePathsList | null |
| IN | datasourceAncestor | 1 |
| IN | inIgnoreResourceDoesNotExist | 1 |
| OUT | resourceTreeList | See charts below |

* 1. Chart 1: Columns (1-6)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| seqnum | resourceID | parentID | resDepth | treeType | resName |  |  |
| 1 | 22063 | [NULL] | 0 | Parent | ViewSales |  |  |
| 1 | 1 | [NULL] | 0 | Ancestor | [NULL] |  |  |
| … |  |  |  |  |  |  |  |

* 1. Chart 2: Columns (7-10)

|  |  |  |  |
| --- | --- | --- | --- |
| resPath | resType | subType | enabled |
| /shared/examples/ViewSales | TABLE | SQL\_TABLE | 1 |
| / | CONTAINER | NONE | 1 |
| … |  |  |  |

* 1. Chart 3: Columns (11-15)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dsID | dsResName | dsResPath | dsResType | dsResSubType |
| [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |
| [NULL] | [NULL] | [NULL] | [NULL] | [NULL] |
| … |  |  |  |  |

* 1. Chart 4: Columns (16-18)

|  |  |  |
| --- | --- | --- |
| dsEnabled | dsChildCount | datasourceType |
| [NULL] | [NULL] | [NULL] |
| [NULL] | [NULL] | [NULL] |
| … |  |  |

### getResourceListByAnnotation

This procedure is used to generate a resource list based on searching resource annotations for a particular release label string. It is returned as a cursor output. The Utilities procedure "/shared/ASAssets/Utilities/archive/getPkgExportListByLabel" uses this procedure. This procedure invokes the procedure "/shared/ASAssets/Utilities/repository/parseAnnotationLabel".

This procedure will automatically exclude TDV system paths including the following:

/services/databases/system

/services/databases/transient

/services/webservices/system

Rules:

1. The annotation requires a uniform strategy in which a unique label can be determined using a simple search method.

2. The "Label" is unique and is the key search term. If that search term is present then the resource is chosen for the export list.

3. The "Release Date" is simply provides additional information to developers regarding the date of the release but is not used for searching.

4. The "command Extension" provides additional insight into what other deployment metadata files should be generated based on the type of command and the context of the resource.

5. No spaces are allowed in the label or the release date.

6. Each label should be a row which is delimited by a newline also known as CHR(10)=linefeed.

7. The command=command text may be a pipe-separated list up to 4 commands.

Scenario 1: 2 chained commands are known about which is introspect|design=varchar(255):abc

Scenario 2: Scenario 2: 2 chained commands are known about which is reintrospect|relocate=DVKPI7/dbo DVKPI8/DVKPI, DVKPI7 DVKPI8

8. Commands are applicable to certain resource types and only should be used in their annotations.

introspect – only applicable with data source children like database tables and procedures

* The format of the command is simply introspect. example: introspect

reintrospect – only applicable with data source resources

* The format of the command is simply reintrospect. example: reintrospect

design – only applicable with data source procedure children requiring design-by-example.

* The format of the command is desgin=type:value. example: design=VARCHAR(255):abc

rebind – applicable to views that need to be rebound to a different data source catalog or schema upon import.

* The format of the command is rebind=from-tdv-path to-tdv-path
* User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"
* example: rebind=/shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI7/dbo /shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI8/DVKPI

relocate – applicable to data sources where the catalog and/or schema needs to be relocated to a different name upon import.

* The format of the command is relocate=catalog1/schema1 catalog2/schema2, catalog1 catalog2
* Always specify the relative schema path first and then the relative catalog using comma-separated lists. The context is the data source where the annotation exists.
* User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"
* example: relocate=DVKPI7/dbo DVKPI8/DVKPI, DVKPI7 DVKPI8

export - only applicable to containers and datasources. A container may be folder, catalog or schema.

* The export command will be ignored on any other type of resource.
* Normally, export is assumed if a matching label is found on search and no other commands are specified.
* However, the export command should be specified in the label of a folder/container/datasource if there are other commands present in order to distinguish what the user intends to accomplish.
* When the intention is to execute multiple commands such as export|remove=resource:type or export|reintrospect for data sources then the export is registered along with the other commands.
* When the intention is not to export but a label is present along with other commands such as remove=resource:type then only that command is registered and not the export.

remove - this command can only be specified in the label of a container or datasource. A container may be folder, catalog or schema.

* The format of the remove command is remove=resource:type. The type is either the internal type or the type from the Studio info tab (display type).
* To remove multiple resources within the same container, use the format remove=resource:type,resource:type,resource:type.
* The resource name is case sensitive. The type is not case sensitive.
* User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"
* If removing a folder then the label of the parent folder must specify the remove.
* If removing a database table or procedure then the remove command must go into the direct parent folder which is typically the schema container but it could be the data source if it has not schemas.
* If removing a database schema then the remove command goes one level up into the catalog container if it exists or the datasource.

The following is an example only:

Label Release Date command extension

V0000001 2021-11-07 command|command=optional command text

V0000002 2021-11-08 introspect

V0000003 2021-11-09 reintrospect

V0000004 2021-11-10 design=VARCHAR(255):abc

V0000005 2021-11-11 relocate=dvops/dbo dvops/db1,dvops dvops2

V0000006 2021-11-12 rebind=/shared/TEST/DS1/dvops/dbo /shared/TEST1/DS1/dvops2/db1

V0000007 2021-11-13 remove=proc1:procedure,view1:view

V0000008 2021-11-14 export

V0000009 2021-11-15

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | *debug* - Y=debug on. N=debug off. | | CHAR(1) |
| IN | *startPath* - The starting path list is a colon-separated list of paths to evaluate the child resource annotations for the release label. Use a wild-card % symbol to select all child paths for the base path. The base path is also included in the evaluation. e.g. /services/databases/myprojectDB%: /services/webservices/myproject%:/shared/myproject% | | LONGVARCHAR |
| IN | *searchStr* - Can be any string with no spaces representing a unique release label. e.g. V0000007 | | LONGVARCHAR |
| IN | *excludePathList* - colon-separated list of paths to exclude. | | LONGVARCHAR |
| OUT | result | **id**  BIGINT, -- the resource id  **name** VARCHAR, -- the resource name  **resPath** VARCHAR(4000), -- the resource path  **resType** VARCHAR, -- the resource type  **subtype** VARCHAR, -- the resource subtype  **owner** VARCHAR, -- The resource owner  **ownerId** BIGINT, -- The resource owner id  **len** INTEGER, -- The annotation length  **annotation** LONGVARCHAR, -- The resource annotation  **datasourceChild** SMALLINT, -- 1=is a datasource child. 0=is not a datasource child.  **datasourcePath** VARCHAR(4000), -- the path to the datasource associated with this resource.  **datasourceCatalog** VARCHAR, -- the catalog name of the datasource  **datasourceSchema** VARCHAR, -- the schema name of the datasource  **releaseLabel** VARCHAR, -- the release label  **releaseDate** VARCHAR, -- the release date  **command1** VARCHAR, -- Command. 1 of |introspect|reintrospect|design|relocate|rebind|  **command1Text** LONGVARCHAR, -- The text following the command if present and qualifies the command with additional instructions.  **command2** VARCHAR, -- Command. 1 of |introspect|reintrospect|design|relocate|rebind|  **command2Text** LONGVARCHAR, -- The text following the command if present and qualifies the command with additional instructions.  **command3** VARCHAR, -- Command. 1 of |introspect|reintrospect|design|relocate|rebind|  **command3Text** LONGVARCHAR, -- The text following the command if present and qualifies the command with additional instructions.  **command4** VARCHAR, -- Command. 1 of |introspect|reintrospect|design|relocate|rebind|  **command4Text** LONGVARCHAR -- The text following the command if present and qualifies the command with additional instructions. | |
|  |  |  | |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | | **Parameter Value** | |
| --- | --- | --- | --- | --- |
| IN | | debug | | Y | |
| IN | | startPath | | /shared/TEST% | |
| IN | | searchStr | | V0000002 | |
| IN | | caseSensitive | | 0 | |
| IN | | excludePathList | | null | |
| OUT | | result | | Too big to display | |

### getResourceListChildren

Return a list of resources, resource path, type and subtype for a given folder path. The input to this is a starting folder and not an actual resource. Return all immediate children of the starting folder including sub-folders (containers) and non-folder resources.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType |
| OUT | result | OUT result PIPE (  name VARCHAR(255),  resPath VARCHAR(4096),  resType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  changeid INTEGER,  version VARCHAR(255),  introspectState VARCHAR(255),  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  annotation LONGVARCHAR,  explicitlyDesigned BIT,  isFunction BIT,  tableType VARCHAR(255),  sqlText LONGVARCHAR,  scriptText LONGVARCHAR,  proprietaryModel LONGVARCHAR,  childCount INTEGER,  dataSourceType VARCHAR(255),  creationDate VARCHAR(255),  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ) |
|  |  |  |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/ASAssets/Utilities’ |
| IN | resourceType | 'CONTAINER' |
| OUT | resourceTreeList | See chart below |

* 1. Chart showing example output for resourceTreeList:

|  |  |  |  |
| --- | --- | --- | --- |
| name | resPath | resType | subType |
| calculations | /shared/ASAssets/Utilities/calculations | CONTAINER | FOLDER\_CONTAINER |
| conversions | /shared/ASAssets/Utilities/conversions | CONTAINER | FOLDER\_CONTAINER |
| file | /shared/ASAssets/Utilities/file | CONTAINER | FOLDER\_CONTAINER |
| log | /shared/ASAssets/Utilities/log | CONTAINER | FOLDER\_CONTAINER |
| repository | /shared/ASAssets/Utilities/repository | CONTAINER | FOLDER\_CONTAINER |
| Etc… |  |  |  |

### getResourceListRecursive

Return a list of resources, resource path, type and subtype for a given folder path. The input to this is a starting folder and not an actual resource. Recursively walk down the tree to identify the list of resources. Return all resources including folders (containers) and non-folder resources.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType |
| OUT | result | OUT result PIPE (  name VARCHAR(255),  resPath VARCHAR(4096),  resType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  changeid INTEGER,  version VARCHAR(255),  introspectState VARCHAR(255),  ownerDomain VARCHAR(255),  ownerName VARCHAR(255),  impactLevel VARCHAR(255),  impactMessage VARCHAR(32768),  annotation LONGVARCHAR,  explicitlyDesigned BIT,  isFunction BIT,  tableType VARCHAR(255),  sqlText LONGVARCHAR,  scriptText LONGVARCHAR,  proprietaryModel LONGVARCHAR,  childCount INTEGER,  dataSourceType VARCHAR(255),  creationDate VARCHAR(255),  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserId VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate VARCHAR(255),  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserId VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/ASAssets/Utilities’ |
| IN | resourceType | 'CONTAINER' |
| OUT | resourceTreeList | See chart below |

* 1. Chart showing example output for resourceTreeList:

|  |  |  |  |
| --- | --- | --- | --- |
| name | resPath | resType | subType |
| TypeDefinitions | /shared/ASAssets/Utilities/TypeDefinitions | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| calculations | /shared/ASAssets/Utilities/calculations | CONTAINER | FOLDER\_CONTAINER |
| calculateAge | /shared/ASAssets/Utilities/calculations/calculateAge | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| conversions | /shared/ASAssets/Utilities/conversions | CONTAINER | FOLDER\_CONTAINER |
| convertBoolean | /shared/ASAssets/Utilities/conversions/convertBoolean | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| convertDoubleToInteger | /shared/ASAssets/Utilities/conversions/convertDoubleToInteger | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| convertYN | /shared/ASAssets/Utilities/conversions/convertYN | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| file | /shared/ASAssets/Utilities/file | CONTAINER | FOLDER\_CONTAINER |
| FileProcessingCJP | /shared/ASAssets/Utilities/file/FileProcessingCJP | DATA\_SOURCE | NONE |
| log | /shared/ASAssets/Utilities/log | CONTAINER | FOLDER\_CONTAINER |
| errorNotification | /shared/ASAssets/Utilities/log/errorNotification | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| logDebugMessage | /shared/ASAssets/Utilities/log/logDebugMessage | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| repository | /shared/ASAssets/Utilities/repository | CONTAINER | FOLDER\_CONTAINER |
| \_debug | /shared/ASAssets/Utilities/repository/\_debug | PROCEDURE | SQL\_SCRIPT\_PROCEDURE |
| Etc… |  |  |  |

### getResourceListUnpublished

This procedure looks at the resources in a starting folder and reports on any that cannot be traced to a dependency that is either a published resource or trigger. In other words, it reports on any orphaned resources that cannot be accessed by an external user over one of the access protocols.

If **excludeTypes** is NULL, then the procedure will automatically exclude resources of type CONTAINER and TRIGGER in its search.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | startingFolder | /lib/resource/ResourceDefs.ResourcePath |
| IN | excludeTypes | VARCHAR(1024) |
| IN | excludePathsList | LONGVARCHAR |
| OUT | result | CURSOR (  name /lib/resource/ResourceDefs.ResourceName,  resPath /lib/resource/ResourceDefs.ResourcePath,  resType /lib/resource/ResourceDefs.ResourceType,  subtype /lib/resource/ResourceDefs.ResourceType,  creationDate TIMESTAMP,  creationDateBigint BIGINT,  creatorUserDomain VARCHAR(255),  creatorUserName VARCHAR(255),  lastModifiedDate TIMESTAMP,  lastModifiedDateBigint BIGINT,  lastModifiedUserDomain VARCHAR(255),  lastModifiedUserName VARCHAR(255)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | startingFolder | ‘/shared/examples’ |
| IN | excludeTypes | NULL |
| IN | excludePathsList | NULL |
| OUT | resourceTreeList | [output too large to display] |

### getResourcePrivilegeDependencies

This procedure was built with the intention to mimic the Composite Manager capability to show "Dependency Privileges". This procedure returns a list of user/group resource privileges for a specified resource path and its dependencies given various inclusion and exclusion filters. This procedure excludes the system paths shown here: /, /shared, /services, /services/databases, /services/webservices. This procedure returns a privilegeStatus= [PASS,FAIL] which indicates whether the privileges for a given dependent view meet the combined privilege criteria [PASS] or they do not [FAIL].

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(4096)) |
| IN | excludePathsList | LONGVARCHAR |
| IN | inIgnoreResourceDoesNotExist | BIT |
| IN | nameTypeFilter | VARCHAR |
| IN | domainFilter | VARCHAR |
| IN | includeNameEqualFilter | LONGVARCHAR |
| IN | includeNameLikeFilter | LONGVARCHAR |
| IN | excludeNameNotEqualFilter | LONGVARCHAR |
| IN | excludeNameNotLikeFilter | LONGVARCHAR |
| IN | includePrivsEqualFilter | VARCHAR |
| IN | includePrivsLikeFilter | VARCHAR |
| IN | excludePrivsNotEqualFilter | VARCHAR |
| IN | excludePrivsNotLikeFilter | VARCHAR |
| IN | includeColumnPrivs | BIT |
| IN | debug | CHAR(1) |
| OUT | result | CURSOR (  resourceID INTEGER,  treeType VARCHAR(255),  resName VARCHAR(255),  path VARCHAR(4096),  type VARCHAR(40),  subtype VARCHAR(40),  enabled BIT,  name VARCHAR(255),  nameType VARCHAR(255),  domain VARCHAR(255),  privilegeStatus VARCHAR(255),  privilegeRunTimeAnalysis VARCHAR(255),  privilegeDesignTimeAnalysis VARCHAR(255),  privs VARCHAR(255),  combinedPrivs VARCHAR(255),  inheritedPrivs VARCHAR(255),  p\_N BIT,  p\_R BIT,  p\_W BIT,  p\_E BIT,  p\_S BIT,  p\_U BIT,  p\_I BIT,  p\_D BIT,  p\_G BIT,  c\_N BIT,  c\_R BIT,  c\_W BIT,  c\_E BIT,  c\_S BIT,  c\_U BIT,  c\_I BIT,  c\_D BIT,  c\_G BIT,  i\_N BIT,  i\_R BIT,  i\_W BIT,  i\_E BIT,  i\_S BIT,  i\_U BIT,  i\_I BIT,  i\_D BIT,  i\_G BIT  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples' |
| IN | resourceType | 'CONTAINER' |
| IN | excludePathsList | '/shared/examples/not\_this\_folder, /shared/examples/not\_that\_folder' |
| IN | inIgnoreResourceDoesNotExist | 0 |
| IN | nameTypeFilter | 'GROUP' |
| IN | domainFilter | 'composite' |
| IN | includeNameEqualFilter | 'all' |
| IN | includeNameLikeFilter | NULL |
| IN | excludeNameNotEqualFilter | NULL |
| IN | excludeNameNotLikeFilter | NULL |
| IN | includePrivsEqualFilter | NULL |
| IN | includePrivsLikeFilter | NULL |
| IN | excludePrivsNotEqualFilter | NULL |
| IN | excludePrivsNotLikeFilter | NULL |
| IN | includeColumnPrivs | 0 |
| IN | debug | 'N' |
| OUT | result | <Too large to display> |

### getResourcePrivileges

This procedure returns a list of user resource privileges for a specified resource path given various inclusion and exclusion filters.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(4096)) |
| IN | nameTypeFilter | VARCHAR |
| IN | domainFilter | VARCHAR |
| IN | includeNameEqualFilter | LONGVARCHAR |
| IN | includeNameLikeFilter | LONGVARCHAR |
| IN | excludeNameNotEqualFilter | LONGVARCHAR |
| IN | excludeNameNotLikeFilter | LONGVARCHAR |
| IN | includePrivsEqualFilter | VARCHAR |
| IN | includePrivsLikeFilter | VARCHAR |
| IN | excludePrivsNotEqualFilter | VARCHAR |
| IN | excludePrivsNotLikeFilter | VARCHAR |
| IN | includeColumnPrivs | BIT |
| IN | debug | CHAR(1) |
| OUT | result | CURSOR (  name VARCHAR(255),  path VARCHAR(4096),  type VARCHAR(40),  nameType VARCHAR(255),  domain VARCHAR(255),  privs VARCHAR(255),  combinedPrivs VARCHAR(255),  inheritedPrivs VARCHAR(255),  p\_N BIT,  p\_R BIT,  p\_W BIT,  p\_E BIT,  p\_S BIT,  p\_U BIT,  p\_I BIT,  p\_D BIT,  p\_G BIT,  c\_N BIT,  c\_R BIT,  c\_W BIT,  c\_E BIT,  c\_S BIT,  c\_U BIT,  c\_I BIT,  c\_D BIT,  c\_G BIT,  i\_N BIT,  i\_R BIT,  i\_W BIT,  i\_E BIT,  i\_S BIT,  i\_U BIT,  i\_I BIT,  i\_D BIT,  i\_G BIT  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples' |
| IN | resourceType | 'CONTAINER' |
| IN | nameTypeFilter | 'GROUP' |
| IN | domainFilter | 'composite' |
| IN | includeNameEqualFilter | 'all' |
| IN | includeNameLikeFilter | NULL |
| IN | excludeNameNotEqualFilter | NULL |
| IN | excludeNameNotLikeFilter | NULL |
| IN | includePrivsEqualFilter | NULL |
| IN | includePrivsLikeFilter | NULL |
| IN | excludePrivsNotEqualFilter | NULL |
| IN | excludePrivsNotLikeFilter | NULL |
| IN | includeColumnPrivs | 0 |
| IN | debug | 'N' |
| OUT | result | <Too large to display> |

### getResourcePrivilegesByUser

Return a list of privileges for a resource. Shows explicit privileges, inherited privileges, and combined privileges for each user that has any kind of privileges on the resource.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(4096)) |
| OUT | result | CURSOR (  “path” VARCHAR(32768),  “type” VARCHAR(32768),  “name” VARCHAR(32768),  “domain” VARCHAR(32768),  privs VARCHAR(32768),  combinedPrivs VARCHAR(32768),  inheritedPrivs VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples’ |
| IN | resourceType | ‘CONTAINER’ |
| OUT | result | See chart below |

* 1. Chart showing example output for result:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Path | Type | Name | Domain | Privs | CombinedPrivs | InheritedPrivs |
| /shared/examples | CONTAINER | admin | composite | READ WRITE EXE… | READ WRITE EXE… | READ WRITE EXE… |
| /shared/examples | CONTAINER | nobody | composite | NONE | NONE | NONE |
| Etc. |  |  |  |  |  |  |

### getResourcePrivilegesGroupsUsers

This procedure returns a list of user or group resource privileges for a specified resource path given various inclusion and exclusion filters. The various includeActualPrivs... and excludeAcutalPrivs are applied against the "privs" field which are the actual/explicitly set privileges. To bring back only privileges that are explicitly set then set the input variable "excludeActualPrivsNotEqualFilter" to NONE. The various includePrivs... and excludePrivs... filters are applied against the combinedPrivs field.

The returned privileges per user or group are the privileges specifically given to that user or group. In each "privilegeEntry", the "combinedPrivs" element contains the effective privileges for that user or group based on their membership in all other groups. In each "privilegeEntry", the "inheritedPrivs" element only contains the privileges that were inherited due to group membership. Logically OR'ing the "privs" and "inheritedPrivs" is the same as the "combinedPrivs".

A user with GRANT privilege or with READ\_ALL\_RESOURCES right will receive all privilege information for all users for a that resource. Other users will only receive their own privilege information.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | VARCHAR(4096) |
| IN | processFoldersRecursive | CHAR(1) |
| IN | userOutputType | CHAR(1) |
| IN | Indebug | CHAR(1) |
| IN | nameTypeFilter | VARCHAR |
| IN | domainFilter | VARCHAR |
|  | ---- Name Filter --- |  |
| IN | includeNameEqualFilter | LONGVARCHAR |
| IN | includeNameLikeFilter | LONGVARCHAR |
| IN | excludeNameNotEqualFilter | LONGVARCHAR |
| IN | excludeNameNotLikeFilter | LONGVARCHAR |
|  | --- Actual Privileges Filters --- |  |
| IN | includeActualPrivsEqualFilter | LONGVARCHAR |
| IN | includeActualPrivsLikeFilter | LONGVARCHAR |
| IN | excludeActualPrivsNotEqualFilter | LONGVARCHAR |
| IN | excludeActualPrivsNotLikeFilter | LONGVARCHAR |
|  | --- Combined Privileges Filters --- |  |
| IN | includePrivsEqualFilter | VARCHAR |
| IN | includePrivsLikeFilter | VARCHAR |
| IN | excludePrivsNotEqualFilter | VARCHAR |
| IN | excludePrivsNotLikeFilter | VARCHAR |
|  | --- Column Privileges Inclusion --- |  |
| IN | includeColumnPrivs | BIT |
| IN | debug | CHAR(1) |
| OUT | result | CURSOR (  path VARCHAR(4096),  type VARCHAR(40),  name VARCHAR(255),  nameType VARCHAR(255),  domain VARCHAR(255),  userList LONGVARCHAR,  privs VARCHAR(255),  combinedPrivs VARCHAR(255),  inheritedPrivs VARCHAR(255),  p\_N BIT,  p\_R BIT,  p\_W BIT,  p\_E BIT,  p\_S BIT,  p\_U BIT,  p\_I BIT,  p\_D BIT,  p\_G BIT,  c\_N BIT,  c\_R BIT,  c\_W BIT,  c\_E BIT,  c\_S BIT,  c\_U BIT,  c\_I BIT,  c\_D BIT,  c\_G BIT,  i\_N BIT,  i\_R BIT,  i\_W BIT,  i\_E BIT,  i\_S BIT,  i\_U BIT,  i\_I BIT,  i\_D BIT,  i\_G BIT  ) |

1. **Examples:**
   1. **Assumptions: Display privileges under /shared/examples where the actual privileges are not equal to ‘NONE’.**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples' |
| IN | resourceType | 'CONTAINER' |
| IN | nameTypeFilter | 'GROUP' |
| IN | domainFilter | 'composite' |
| IN | includeNameEqualFilter | 'all' |
| IN | includeNameLikeFilter | NULL |
| IN | excludeNameNotEqualFilter | NULL |
| IN | excludeNameNotLikeFilter | NULL |
| IN | includeActualPrivsEqualFilter | NULL |
| IN | includeActualPrivsLikeFilter | NULL |
| IN | excludeActualPrivsNotEqualFilter | NONE |
| IN | excludeActualPrivsNotLikeFilter | NULL |
| IN | includePrivsEqualFilter | NULL |
| IN | includePrivsLikeFilter | NULL |
| IN | excludePrivsNotEqualFilter | NULL |
| IN | excludePrivsNotLikeFilter | NULL |
| IN | includeColumnPrivs | 0 |
| IN | debug | 'N' |
| OUT | result | <Too large to display> |

### getResourcesByDate

This procedure is used to return a list of resources from a given starting folder based on the options described in in the "optionType" input parameter. Depending on the option type selected, it may use the creation date attribute or the modification date attribute when determining the result set. The starting resource is included in the output list if applicable for the option and parameters passed in.

Because the results are not ordered, the user of this method may wish to execute the procedure with an ORDER BY.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | *debug* - Y=debug on. N=debug off. | CHAR(1) |
| IN | *optionType*  -- 1: Find the n most recent modified resources in the path (default of n is 1 if left null or 0). Results are not ordered.  -- 2: Find the n least recent modified resources in the path (default of n is 1 if left null or 0). Results are not ordered.  -- 3: Find all resources modified after the passed in resource timestamp  -- 4: Find all resources modified before the passed in resource timestamp  -- 5: Find all resources created after the passed in resource timestamp  -- 6: Find all resources created before the passed in resource timestamp  -- 7: [Lineage + modification date] Find the lineage of the passed in resource path where the modification date is after the passed in resource date and exclude paths if applicable. | INTEGER |
| IN | *resourcePath* - DV resource path to begin deriving a list of resources | VARCHAR(4096) |
| IN | *resourceType* - DV resource type. e.g. CONTAINER, LINK, TABLE, PROCEDURE, DATA\_SOURCE | VARCHAR |
| IN | *resourceNum* - The number of resources to return in the list for optionType 1 or 2. The default is 1 | INTEGER |
| IN | *resourceDate* - A timestamp to compare with the various resources for optionType 3-7. | TIMESTAMP |
| OUT | result | CURSOR |

1. **Examples:**
   1. **Assumptions: none.**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Debug | N |
| IN | optionType | 1 |
| IN | resourcePath | '/shared/examples' |
| IN | resourceType | 'CONTAINER' |
| IN | resourceNum | 1 |
| IN | resourceDate | '2013-08-16 00:00:00' |
| OUT | result | <row set> |

### getResourceSqlTable

This procedure is used to retrieve table or view metadata. This metadata can be used with resources/updateResourcesSqlTable().

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath - Full resource path which includes the path and the resource name | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | scripttext - SQL Table text to be updated | LONGVARCHAR |
| OUT | columnList - a vector array of sql columns and definitions | childResourceType ROW (  resourceName VARCHAR,  resourcePath TypeDefinitions.pathType,  resourceType VARCHAR,  columnName VARCHAR,  columnType VARCHAR ); |
| OUT | sqlIndexList - a vector array of sql indexes | sqlIndexType ROW (  sqlIndexName VARCHAR(255),  sqlIndexType VARCHAR(255),  sqlIndexUnique BIT,  sqlIndexColName VARCHAR(255),  sqlIndexColOrder VARCHAR(255)  ); |
| OUT | foreignKeyList - a vector array of foreign keys | foreignKeyType ROW (  fkName VARCHAR(255),  fkPrimaryKeyName VARCHAR(255),  fkPrimaryKeyTable TypeDefinitions.pathType,  fkForeignKeyColumnName VARCHAR(255),  fkPrimaryKeyColumnName VARCHAR(255)  ); |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/target/ PRODUCT\_VIEW2’ |
| OUT | scripttext | ‘SELECT \* FROM /shared/examples/ds\_inventory/products products’ |
| OUT | columnList | [('products','$products','TABLE','ProductID','INTEGER'),  ('products','$products','TABLE','ProductName','VARCHAR(50)'),  ('products','$products','TABLE','ProductDescription','VARCHAR(255)'),  ('products','$products','TABLE','CategoryID','INTEGER'),  ('products','$products','TABLE','SerialNumber','VARCHAR(50)'),  ('products','$products','TABLE','UnitPrice','DECIMAL(12,2)'),  ('products','$products','TABLE','ReorderLevel','INTEGER'),  ('products','$products','TABLE','LeadTime','VARCHAR(30)')] |
| OUT | sqlIndexList | [('productsPK','PRIMARY\_KEY',1,'ProductID','ASCENDING')] |
| OUT | foreignKeyList | [('categoriesFK','categoriesPK','$categories','CategoryID','CategoryID')] |

$products = /shared/ASAssets/Utilities/repository/examples/source/ds\_inventory/products

$categories = /shared/ASAssets/Utilities/repository/examples/source/ds\_inventory/categories

### getScriptText (Custom Function)

This procedure returns the script text for a procedure. This allows a program to get the text modify the text and then use another procedure to update the procedure.

The following resource types and sub-types are supported:

resourceType = 'PROCEDURE'

subtype = 'SQL\_SCRIPT\_PROCEDURE' -- Get Regular Procedure

subtype = 'EXTERNAL\_SQL\_PROCEDURE' -- Get Packaged Query Procedure

subtype = 'XSLT\_TRANSFORM\_PROCEDURE' -- Get XSLT Transformation text

subtype = 'XQUERY\_TRANSFORM\_PROCEDURE' -- Get XQuery Transformation text

resourceType = 'TABLE'

subtype = 'SQL\_TABLE' -- Get Regular View

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | subType | VARCHAR(255) |
| OUT | scriptText | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/proc2’ |
| IN | resourceType | ‘PROCEDURE’ |
| OUT | subType | ‘SQL\_SCRIPT\_PROCEDURE’ |
| OUT | scriptText | ‘PROCEDURE proc2() BEGIN END’ |

### getTableColumnStatisticsConfiguration

This procedure returns the statistics configuration and manual override information for the table specified in resourcePath, as well as its columns.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | cardinalityMin | INTEGER |
| OUT | cardinalityMax | INTEGER |
| OUT | cardinalityExpected | INTEGER |
| OUT | gatherEnabled | VARCHAR(20) |
| OUT | maxTime | INTEGER |
| OUT | columnSettings | CURSOR ( name VARCHAR(1000),  flags VARCHAR(1000),  columnMin DOUBLE,  columnMax DOUBLE,  columnDistinct INTEGER,  enableColumnOverride BIT ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples/ds\_orders/orders’ |
| OUT | cardinalityMin | NULL |
| OUT | cardinalityMax | NULL |
| OUT | cardinalityExpected | NULL |
| OUT | gatherEnabled | 'DEFAULT' |
| OUT | maxTime | -1 |
| OUT | columnSettings | See table below: |

* 1. Chart 1: Columns (1-6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| name | flags | columnMin | columnMax | columnDistinct | enableColumnOverride |
| 'OrderID' | 'NONE' | NULL | NULL | NULL | 1 |
| 'CustomerID' | 'NONE' | NULL | NULL | NULL | 1 |
| 'EmployeeID' | 'NONE' | NULL | NULL | NULL | 1 |
| … |  |  |  |  |  |

### getTrigger

This is a procedure is used to get trigger details. It returns general trigger information, condition information and action information.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off. | CHAR(1) |
| IN | triggerPath - DV source trigger path. Resource must exist. | VARCHAR(4000) |
| OUT | result | CURSOR ( ------------------------  -- General Variables  ------------------------  name VARCHAR, -- The trigger name  "path" VARCHAR(4000), -- The trigger path  "type" VARCHAR, -- The type is always TRIGGER  subtype VARCHAR, -- The subtype is always NONE  id BIGINT, -- The resource id  changeId BIGINT, -- The change id  ownerDomain VARCHAR, -- The owner domain  ownerName VARCHAR, -- The owner user name  impactLevel VARCHAR, -- NONE=no impact. anything else indicates that the trigger is impacted and will show red in Studio  enabled BIT, -- 1=enabled 0=disabled  annotation LONGVARCHAR, -- The trigger annotation  creationDateLong BIGINT, -- The creation date in long format (stored this way)  creationDate TIMESTAMP, -- The creation date in timestamp format  creatorUserDomain VARCHAR, -- The creator domain  creatorUserName VARCHAR, -- The creator user name  lastModifiedDateLong BIGINT, -- The last modified date in long format (stored this way)  lastModifiedDate TIMESTAMP, -- The last modified date in timestamp format  lastModifiedUserDomain VARCHAR,-- The last modified user domain  lastModifiedUserName VARCHAR,-- The last modified user name  maxEventsQueued INTEGER, -- Maximum events that can be queued up for the trigger.  ------------------------  -- Condition Variables  ------------------------  conditionType VARCHAR, -- Condition Type: [TIMER|JMS|SYSTEM\_EVENT|USER\_DEFINED]  **conditionType=SYSTEM\_EVENT or USER\_DEFINED**  eventName VARCHAR, -- The name of the SYSTEM\_EVENT user defined event or USER\_DEFINED event.  **conditionType=TIMER**  timerMode VARCHAR, -- mode: [INTERVAL|CALENDAR]  timerStartTime VARCHAR,  timerPeriod VARCHAR, -- One of [MINUTE|HOUR|DAY|WEEK|MONTH|YEAR]  timerPeriodQty INTEGER,  -- If timerMode=INTERVAL then timerPeriod=MINUTE and timerPeriodQty=#of min.  -- If timerMode=CALENDER then timerPeriod=as is value and timerPeriodQty=as is value  timerFromTimeInADay INTEGER, -- NULL or The time of day to start integer value of the formatted version below.  timerFromTimeInADayFormatted VARCHAR, -- NULL or The time of day to start the trigger. Format = HH24:MM:SS  timerEndTimeInADay INTEGER, -- NULL or The time of day to end integer value of the formatted version below.  timerEndTimeInADayFormatted VARCHAR, -- NULL or The time of day to end the trigger. Format = HH24:MM:SS  timerRecurringDay INTEGER, -- The integer value of the recurring day list.  timerRecurringDayList VARCHAR,-- A list of one or more items: [SUN, MON, TUE, WED, THU, FRI, SAT]. May be space or comma delimited.  timerIsCluster BIT,  **conditionType=JMS**  jmsConnector VARCHAR(4000), -- The JMS Connector string.  jmsSelector VARCHAR(4000), -- The JMS Selector string.  jmsDestination VARCHAR(4000), -- The JMS Destination string.  ------------------------  -- Action Variables  ------------------------  actionType VARCHAR,-- Action Type: [PROCEDURE|STATISTICS|REINTROSPECT|EMAIL]  **ActionType=PROCEDURE or STATISTICS or REINTROSPECT or EMAIL Variables**  actionPath VARCHAR(4000), -- The resource path  -- when actionType=PROCEDURE: actionPath=procedure path and actionPathType=PROCEDURE  -- when actionType=STATISTICS: actionPath=data source path and actionPathType=DATA\_SOURCE  -- when actionType=REINTROSPECT: actionPath=data source path and actionPathType=DATA\_SOURCE  -- when actionType=EMAIL: actionPath=resource path and actionPathType=the resource path type  actionPathType VARCHAR, -- Action Path type: [TABLE|PROCEDURE...]  **ActionType=PROCEDURE Variables**  actionExhaustCursors BIT, -- 1=true, 0=false to exhaust output cursors  actionParameters LONGVARCHAR,-- The trigger parameters for a procedure call.  actionNumParameters INTEGER, -- The number of parameters being passed in to the procedure  **ActionType=EMAIL or REINTROSPECT**  emailTo VARCHAR(4000), -- The send to email list  emailCC VARCHAR(4000), -- The cc email list  emailBCC VARCHAR(4000), -- The bcc email list  emailReplyTo VARCHAR(4000), -- The replay to email list  emailSubject LONGVARCHAR, -- The email subject  emailContent LONGVARCHAR, -- The email content  emailInclueSummary BIT, -- 1=true, 0=false to include summary  emailSkipIfNoResults BIT, -- 1=true, 0=false to skip sending email if no results  **ActionType=REINTROSPECT**  reintrospectNoCommit BIT -- 1=true, 0=false Do not persist/commit detected changes  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off. | ‘N’ |
| IN | triggerPath - DV source trigger path. Resource must exist. | ‘/shared/examples/trigger1’ |
| OUT | result | Too large to display here |

### getUsedResourcesCursor

This procedure retrieves a cursor of metadata describing what resources are used by the input resource path. The full resource path and resource type must be provided.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | usedResCursor | CURSOR (  resourceName VARCHAR(255),  resourcePath TypeDefinitions.pathType,  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  tableType VARCHAR(255),  explicitlyDesigned BIT,  sqlText VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples/CompositeView’ |
| IN | resourceType | ‘TABLE’ |
| OUT | usedResCursor | <See table below> |

* 1. Chart 1: Columns (1-6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | resourceType | subtype | Enabled | Id |
| VIewOrder | /shared/examples/ViewOrder | TABLE | SQL\_TABLE | 1 | 20658 |
| ViewSales | /shared/examples/ViewSales | TABLE | SQL\_TABLE | 1 | 20774 |
| ViewSupplier | /shared/examples/ViewSupplier | TABLE | SQL\_TABLE | 1 | 20763 |

* 1. Chart 2: Columns (7-9)

|  |  |  |
| --- | --- | --- |
| tableType | explicitlyDesigned | sqlText |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |

### getUsedResourcesRecurseCursor

This procedure recursively retrieves a cursor of metadata describing what resources are "used" by the resource path provided. For each child dependency resource found for the parent, retrieve its "used" dependencies until the entire lineage has been discovered. The full resource path and resource type must be provided. Use the resource type "LINK" for any published database or web service resources.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType |
| IN | inParentID | INTEGER |
| IN | inLineageResourceIdList | LONGVARCHAR |
| IN | inIgnoreResourceDoesNotExist | BIT |
| OUT | result | CURSOR (  resourceName VARCHAR(255),  resourcePath TypeDefinitions.pathType,  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  tableType VARCHAR(255),  explicitlyDesigned BIT,  sqlText VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples/CompositeView’ |
| IN | resourceType | ‘TABLE’ |
| IN | inParentID | NULL |
| IN | inLineageResourceIdList | NULL |
| IN | inIgnoreResourceDoesNotExist | 0 |
| OUT | result | <See table below> |

* 1. Chart 1: Columns (1-6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | resourceType | subtype | Enabled | Id |
| VIewOrder | /shared/examples/ViewOrder | TABLE | SQL\_TABLE | 1 | 20658 |
| ViewSales | /shared/examples/ViewSales | TABLE | SQL\_TABLE | 1 | 20774 |
| ViewSupplier | /shared/examples/ViewSupplier | TABLE | SQL\_TABLE | 1 | 20763 |
| … |  |  |  |  |  |

* 1. Chart 2: Columns (7-9)

|  |  |  |
| --- | --- | --- |
| tableType | explicitlyDesigned | sqlText |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |
| … |  |  |

### getUsedResourcesDirectCursor

This procedure retrieves a cursor of metadata describing what resources are "directly" "used" by the resource path provided. It only returns "direct" descendants and not foreign key descendants or cache table or data source references. The full resource path and resource type must be provided.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | usedResCursor | CURSOR (  resourceName VARCHAR(255),  resourcePath TypeDefinitions.pathType,  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  tableType VARCHAR(255),  explicitlyDesigned BIT,  sqlText VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples/CompositeView’ |
| IN | resourceType | ‘TABLE’ |
| OUT | usedResCursor | <See table below> |

* 1. Chart 1: Columns (1-6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | resourceType | subtype | Enabled | Id |
| VIewOrder | /shared/examples/ViewOrder | TABLE | SQL\_TABLE | 1 | 20658 |
| ViewSales | /shared/examples/ViewSales | TABLE | SQL\_TABLE | 1 | 20774 |
| ViewSupplier | /shared/examples/ViewSupplier | TABLE | SQL\_TABLE | 1 | 20763 |

* 1. Chart 2: Columns (7-9)

|  |  |  |
| --- | --- | --- |
| tableType | explicitlyDesigned | sqlText |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |

### getUsedResourcesDirectRecurseCursor

This procedure recursively retrieves a cursor of metadata describing what resources are "directly" "used" by the resource path provided. It only returns "direct" descendants and not foreign key descendants or cache table or data source references.

For each child dependency resource found for the parent, retrieve its "used" dependencies until the entire lineage has been discovered. The full resource path and resource type must be provided. Use the resource type "LINK" for any published database or web service resources.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType |
| IN | inParentID | INTEGER |
| IN | inLineageResourceIdList | LONGVARCHAR |
| IN | inIgnoreResourceDoesNotExist | BIT |
| OUT | result | CURSOR (  resourceName VARCHAR(255),  resourcePath TypeDefinitions.pathType,  resourceType VARCHAR(255),  subtype VARCHAR(255),  enabled BIT,  id INTEGER,  tableType VARCHAR(255),  explicitlyDesigned BIT,  sqlText VARCHAR(32768)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/examples/CompositeView’ |
| IN | resourceType | ‘TABLE’ |
| IN | inParentID | NULL |
| IN | inLineageResourceIdList | NULL |
| IN | inIgnoreResourceDoesNotExist | 0 |
| OUT | result | <See table below> |

* 1. Chart 1: Columns (1-6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| resourceName | resourcePath | resourceType | subtype | Enabled | Id |
| VIewOrder | /shared/examples/ViewOrder | TABLE | SQL\_TABLE | 1 | 20658 |
| ViewSales | /shared/examples/ViewSales | TABLE | SQL\_TABLE | 1 | 20774 |
| ViewSupplier | /shared/examples/ViewSupplier | TABLE | SQL\_TABLE | 1 | 20763 |
| … |  |  |  |  |  |

* 1. Chart 2: Columns (7-9)

|  |  |  |
| --- | --- | --- |
| tableType | explicitlyDesigned | sqlText |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |
| UNKNOWN | 0 | SELECT … |
| … |  |  |

### getUserPermissionsRecursive

This procedure retrieves a cursor of metadata containing the privileges a user has for a given starting CONTAINER. The procedure cursively inspects and reports on the privileges for all the child resources.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | userName | /lib/users/UserDefs.UserName (VARCHAR(255)) |
| IN | domainName | /lib/users/UserDefs.DomainName (VARCHAR(255)) |
| IN | beginFolder | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| OUT | result | CURSOR (  resPath /lib/resource/ResourceDefs.ResourcePath,  privRead CHAR(1),  privWrite CHAR(1),  privExecute CHAR(1),  privSelect CHAR(1),  privDelete CHAR(1),  privInsert CHAR(1),  privDelete CHAR(1),  privGrant CHAR(1)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | userName | ‘admin’ |
| IN | domainName | ‘composite’ |
| IN | beginFolder | ‘/shared/examples’ |
| OUT | result | See chart below |

* 1. Chart showing example output for result:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| resPath | privRead | privWrite | privExecute | privSelect | privUpdate | privInsert | privDelete | privGrant |
| /shared/examples/CompositeView | Y | Y | Y | Y | Y | Y | Y | Y |
| /shared/examples/LookupProduct | Y | Y | Y | Y | Y | Y | Y | Y |
| Etc. |  |  |  |  |  |  |  |  |

### impactedTargetsList

This procedure crawls through a starting folder and locates all of the resources that are impacted and produces an output cursor with the resource location, impact level and script text if possible.

**inExcludePathsKeywords** is a comma separated list of keywords used to exclude paths containing these any of the keywords (case insensitive.) Examples: Analysis, Archive, save, validation

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | inStartingFolders | LONGVARCHAR |
| IN | inExcludePathsKeywords | LONGVARCHAR |
| OUT | result | CURSOR (  resourcePath /lib/resource/ResourceDefs.ResourcePath,  resourceType /lib/resource/ResourceDefs.ResourceType,  subType /lib/resource/ResourceDefs.ResourceType,  impactLevel VARCHAR(1024),  impactMessage VARCHAR(32768),  scriptText LONGVARCHAR  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | inStartingFolders | ‘/shared/examples’ |
| IN | inExcludePathsKeywords | NULL |
| OUT | result | (  ‘/shared/examples/NewView’,  ‘TABLE’,  ‘SQL\_TABLE’,  ‘UNKNOWN’,  ‘View is newly created and has not been saved.’,  ‘SELECT \* FROM’ ) |

### importResourcePrivileges

This procedure imports the privileges specified in an XML file on the CIS host filesystem. See the resources/importResourcePrivileges() procedure.

The input parameter **updateRecursively** indicates whether to recursively apply the specified privileges to the target resources' children (if the target resource is a CONTAINER, DATA\_SOURCE, or TABLE.)

The input parameter **updateDependenciesRecursively** indicates whether to recursively apply the target resources' privileges to the targets' dependencies (resources that are used by the target.)

The input parameter **updateDependentsRecursively** indicates whether to recursively apply the target resources' privileges to the targets' dependents (resources that use the target.)

The **mode** input parameter indicates whether the privileges settings should be applied without modifying any unreferenced privileges ('OVERWRITE\_APPEND') or should the privileges be applied exactly as presented in the XML file ('SET\_EXACTLY'). If this parameter is NULL then 'OVERWRITE\_APPEND' will be used.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | updateRecursively | BIT |
| IN | updateDependenciesRecursively | BIT |
| IN | updateDependentsRecursively | BIT |
| IN | filename | LONGVARCHAR |
| IN | mode | VARCHAR |
| OUT | updateResourcePrivilegesResponse | XML |
| OUT | fault | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | updateRecursively | 1 |
| IN | updateDependenciesRecursively | 0 |
| IN | updateDependentsRecursively | 0 |
| IN | filename | 'C:\cis\_examples\_privileges.xml' |
| IN | mode | 'SET\_EXACTLY' |
| OUT | updateResourcePrivilegesResponse | <xml> |
| OUT | fault | NULL |

### introspectResources

This script is used to provide a consistent and generic interface for introspecting database tables. It creates a necessary transaction around introspectResourcesTask and introspectResourcesResult. Since the output variable "introspectionResult" is a string result, the invoker of this procedure may also invoke "/shared/ASAssets/Utilities/repository/introspectResourcesResultCursor" to extract the results into a cursor format. The errStatus may return ‘SUCCESS’ or ‘FAILED’. The following actions are applicable with introspectResources:

* SKIP - no change to the resource so it is skipped.
* ADD - a resource container [catalog, schema], table, or procedure is added to TDV from the data source. Privileges "may" be added. See Input, catalogName for details.
* UPDATE - if the structure, indexes, foreign keys change in the data source, it is updated in TDV.
* REMOVE - if the resource is removed from the data source then it is removed from TDV.

Privileges:

It will copy privileges with SET\_EXACTLY from the datasource path down to specified catalog, schema and tables or procedures where applicable.

Copy Privilege Rules:

1. Only the "added" [action=ADD] resources are applied privileges by default unless catalogName contains "=noprivs".
2. Resources that are "updated" [action=UPDATE], "skipped" [action=SKIP], or "removed" [action=REMOVE] will not have privileges applied to them.
3. Resources that were not in the tableNames or procedureNames list will not have privileges applied to them.

Generically, this procedure has the ability to define a resource name, type and subtype in the tableNames or procedureNames input list using this format: name(TYPE:SUBTYPE). For example when introspecting an XML file that also contains a definition set, the user may request in the tableNames parameter the following: productCatalog.xml,productCatalogDefinitions(DEFINITION\_SET:XML\_SCHEMA\_DEFINITION\_SET)

In the above example, productCatalog.xml is not explicitly defined because it uses the default settings for an XML datasource which are TYPE=TRE and SUBTYPE=XML\_FILE\_TREE.

The following section provides detailed on the input and output variables with a description of each field:

Input:

IN dsPath /lib/resource/ResourceDefs.ResourcePath, -- mandatory. Full path to datasource

IN catalogName VARCHAR, -- optional (null) -

Relational Database - database catalog Name if applicable.

CSV - leave this null as it is not applicable

Excel Non-ODBC - leave this null as it is not applicable

IN schemaNames LONGVARCHAR, -- optional (null) - NULL=no specified schema list OR one or more comma separated schemas to introspect. Note: If a list of schema names are provided and a list of table names are provided the list of table names will be applied in total to each schema name. It is not currently in scope to correlate a list of tables to a list of schemas.

Relational Database - schema name or list of schema names.

CSV - leave this null as it is not applicable

Excel Non-ODBC - the excel file name or list of file names.

IN schemaTablePatterns VARCHAR(4096), -- **\*\*\* NOT SUPPORTED for introspection. \*\*\*** Only used during re-introspection.

comma separated list of patterns such as D%, E%, F%. Introspect all tables starting with D, E and F or a bracketed correlated and comma-separated list of table patterns associated with each schema:

schemaNames= SCHEMA1,SCHEMA2 - This is the list of comma-separated schemas

schemaTablePatterns=[M%] [P%] - The 1st set of bracketed table patterns goes with the 1st schema. The 2nd bracketed set of table patterns goes with the 2nd schema

IN tableNames LONGVARCHAR, -- optional. NULL=no specified table list OR one or more comma separated table namess to introspect or a bracketed correlated and comma-separated list of table names associated with each schema:

schemaNames= SCHEMA1,SCHEMA2 - This is the list of comma-separated schemas

tableNames= [T1,T2] [T3,T4] - The 1st set of bracketed tables goes with the 1st schema. The 2nd bracketed set of tables goes with the 2nd schema.

Relational Database - When NULL, all tables for the schema are introspected, otherwise the list of tables provided are introspected.

CSV - When NULL, all CSV files are introspected, otherwise the list of files provided are introspected.

Excel Non-ODBC - When NULL, all sheets are introspected, otherwise the list of sheets provided are introspected.

Table names with spaces may be use double quotes to enclose the name but it is not necessary as spaces are preserved.

IN schemaProcedurePatterns VARCHAR(4096), -- **\*\*\* NOT SUPPORTED for introspection. \*\*\*** Only used during re-introspection.

comma separated list of patterns such as D%, E%, F%. Introspect all new procedure patterns starting with D, E and F or a bracketed correlated and comma-separated list of procedure patterns associated with each schema:

schemaNames= SCHEMA1,SCHEMA2 - This is the list of comma-separated schemas

schemaProcedurePatterns=[N%] [O%] - The 1st set of bracketed procedure patterns goes with the 1st schema. The 2nd bracketed set of procedure patterns goes with the 2nd schema

IN procedureNames LONGVARCHAR, -- optional. NULL=no specified procedure list OR 1 or more comma separated procedure names to introspect or a bracketed correlated and comma-separated list of procedure names associated with each schema:

schemaNames= SCHEMA1,SCHEMA2 - This is the list of comma-separated schemas

procedureNames= [P1][P2,P3,P4] - The 1st set of bracketed procedures goes with the 1st schema. The 2nd bracketed set of procedures goes with the 2nd schema

Relational Database - When NULL, all procedures are introspected, otherwise the list of procedures provided are introspected.

CSV - leave this null as it is not applicable

Excel Non-ODBC - leave this null as it is not applicable

Procedure names with spaces may be use double quotes to enclose the name but it is not necessary as spaces are preserved.

IN separator VARCHAR, -- Default=, The separator used to separate lists for the input parameters: tableNames, schemaTablePatterns, schemaProcedurePatterns and procedureNames. The separator value should not exist within the tableNames or procedureNames variable.

IN inDebug CHAR(1), -- Y=debug on, N=debug off.

Output:

OUT errStatus VARCHAR, -- SUCCESS or FAILED

OUT errMessage LONGVARCHAR, -- Error message if errStatus=FAILED, otherwise null

OUT introspectionResult LONGVARCHAR, -- A line (CHR(10)) delimited string of results. Use introspectResourcesResultCursor() to return a cursor.

OUT dataSourceType VARCHAR, -- The type of data source that was introspected.

OUT dataSourceSubtype VARCHAR -- The subtype of data source that was introspected.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | dsPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | | catalogName | VARCHAR |
| IN | | schemaNames | LONGVARCHAR |
| IN | | schemaTablePatterns | VARCHAR(4096) |
| IN | | tableNames | LONGVARCHAR |
| IN | | schemaProcedurePatterns | VARCHAR(4096) |
| IN | | procedureNames | LONGVARCHAR |
| IN | | separator | VARCHAR |
| IN | | inDebug | CHAR(1) |
| OUT | | errStatus | VARCHAR |
| OUT | | errMessage | LONGVARCHAR |
| OUT | | introspectionResult | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | dsPath | '/shared/examples/ds\_inventory' |
| IN | catalogName | NULL |
| IN | schemaName | 'tutorial' |
| IN | schemaTablePatterns | NULL |
| IN | tableNames | 'categories,employees,products' |
| IN | schemaProcedurePatterns | NULL |
| IN | procedureNames | NULL |
| IN | separator | ',' |
| OUT | errStatus | SUCCESS |
| OUT | errMessage | NULL |
| OUT | introspectionResult | (result too large to display) |

### introspectResourcesResultCursor

This script is used to extract the introspection result text into a cursor of results. The input to this procedure is the output variable [introspectionResult LONGVARCHAR] from the procedure introspectResources.

1. **Parameters:**

| **Direction** | **Parameter Name** | | **Parameter Type** |
| --- | --- | --- | --- |
| IN | | introspectionResult | LONGVARCHAR |
| OUT | | result | CURSOR (  totalResults INTEGER, completed BIT, status VARCHAR(32768), introspectorVersion INTEGER, startTime TIMESTAMP, endTime TIMESTAMP, addedCount INTEGER, removedCount INTEGER, updatedCount INTEGER, skippedCount INTEGER, totalCompletedCount INTEGER, toBeAddedCount INTEGER, toBeRemovedCount INTEGER, toBeUpdatedCount INTEGER, totalToBeCompletedCount INTEGER, warningCount INTEGER, errorCount INTEGER, "path" VARCHAR(4096), "type" VARCHAR(40), subtype VARCHAR(40), "action" VARCHAR(32768), durationMs INTEGER, entryStatus VARCHAR(32768), code VARCHAR(32768), name VARCHAR(32768), message LONGVARCHAR, detail VARCHAR(32768), severity VARCHAR(32768)  ) |

### migrateDatasource

The purpose of this procedure is to migrate a data source from an old adapter type to a new adapter type through automation. The primary target type is for relational adapters. Data sources such as Files, Rest, Web Services are not applicable for this operation.

\*\*\*\* IMPORTANT \*\*\*\*

1. The adapter jar files must be present in the proper TDV\_HOME/conf/adapters/[custom|system]/lower\_case\_adapter\_name directory prior to running this procedure.
2. Any XML configurations must be edited ahead of time as well.
3. TDV must be restarted in order to pick up new jar files from the file system.
4. Repeat this for each node in the cluster.

\*\*\*\* WARNING \*\*\*\*

1. Migrating a cache data source that contains depenendent cached views will invalidate the cache.
2. All caches must be refreshed after migrating the data source.
3. Take special care to backup incremental cache data prior to executing this procedure.

Migration Steps:

1. Verify source and target data source existence.

2. Override delete target data source

2a. If targetOverride=Y and target exists then delete the target data source.

3. If target data source does not exist:

3a. Create target data source based on the adapter type.

3b. Copy source attributes to the target data source.

3c. Update new attributes from the input parameter "targetAttributesOverride"

4. Test the connection. If failed test connection, raise an exception and stop.

5. Get a list of all source resources [catalogs, schemas, tables, procedures]

6. Introspect [add] resources to the target data source.

7. Copy privileges from source to target if resources added and requested via input parameter.

7a. If targetCopyPrivileges=N then do not copy privileges.

8. Verify the source resources against the target resources.

8a. If missing resources, raise an exception and stop.

9. Remove the soure data source

10. Rename target data source back to the source data source

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – [optional] general debug for migrateDatasource. Y=debug on. N=debug off. | CHAR(1) |
| IN | debug2 – [optional] debug level 2. deep debug when getting data source attribute definitions - getDataSourceAttributeDefs | CHAR(1) |
| IN | debug3 – [optional] debug level 3. deep debug when introspecting the data source | CHAR(1) |
| IN | sourceDatasourcePath – [required] The full path to the source data source to migrate from. | VARCHAR(4000) |
| IN | targetDatasourcePath – [required] The full path to the target data source to migrate to [if not exist, create it]. | VARCHAR(4000) |
| IN | tdvServerBackupFolder - [required] The full file system path on the TDV server to store data source backup CAR files. | VARCHAR(4000) |
| IN | encryptionPassword - CAR file export encryption password or "tibco1" by default if null. | VARCHAR |
| IN | targetOverride – [optional] – delete the target data source if it exists.   * N/null = (default) Do not override target. Do not delete it. * Y = override the target data source by deleting it if it exists. | CHAR(1) |
| IN | targetCopyPrivileges – [optional] - copy privileges from the source data source.   * N/null = (default) Do not copy privileges. * Y = copy privileges recursively from the source data source to the target data source. | CHAR(1) |
| IN | targetAdapterName - [optional] The adapter name which is shown on the "Basic" tab of a data source as "Adapter Name".   * Option 1: null if data source already exists. * Option 2: provide the adapter name if the data source does not exist and needs to be created. Example Adapter names:   + Microsoft SQL Server 2019   + Oracle 19c (Thin Driver) | VARCHAR |
| IN | targetAttributesOverride - [required] - The name/value attribute pairs used to configure the data source that override the attributes in the source data source.  The list may contain existing attributes that will override the source attributes. The list may contain new attributes that exist for the new adapter type but not the old adapter type. A "password" is a required field for the migration to work as password is "not" transferred from the source to the target.  Format: [delimiter=||]name=value||name=value||name=value||...name=value||  The "delimiter" is the name=value pair separator. It is always enclosed in square brackets as [delimiter=<value>]  It can be whatever you like as long as it does not occurr in the value. Double-pipe is suggested.  It always occurs first in the string and is mandatory. If not present an exception will be thrown.  The "name" of the attributes are listed here for TDV 8.3 and above:  /services/databases/system/model/SYS\_DATASOURCE\_ATTRIBUTE\_DEFS  Example: ADAPTER\_NAME: Oracle 11g (Thin Driver)  DEFINITION\_NAME: urlPatternStr  The "value" follows 'name=' and must be a valid entry properly formatted for the attribute type being updated.  Example:  [delimiter=||]password=Password1|| urlPatternStr=jdbc:sqlserver://<HOST>:<PORT>; databaseName=<DATABASE\_NAME>; selectMethod=<SELECT\_MODE>|| | LONGVARCHAR |
| IN | excludeAttributes - [optional] - A comma-separated list of attribute names to exclude when copying from the source data source to the target.  Format: name1,name2,name3,name4...nameN  The wild-card symbol "%" may be used at the end of name field to include all atrributes like the attribute name.  The "name" of the attributes are listed here for TDV 8.3 and above: /services/databases/system/model/SYS\_DATASOURCE\_ATTRIBUTE\_DEFS  Example: ADAPTER\_NAME: Oracle 11g (Thin Driver)  DEFINITION\_NAME: urlPatternStr  Example: OAuthConfig  Default Exclude List - the default list is a mandatory exclude list and is concatenated with excludeAttributes:  creator%,creation%,lastModified%,type, anyCharWildcard,introspectionReport,isEnabled,patternSeparator, resourceModifiedDateOnSourceCISSite, searchCaseSensitively, searchEscape,selectionState, singleCharWildcard, statsConfigured,statsNumThreads  Option 1: if null then only the default exclude list is applied  Option 2: if not null this excludeAttributes + the default exclude list are applied | LONGVARCHAR |
| IN | allResourcePath - [optional] - the full TDV resource path to the ALL\_RESOURCES table. If null the default path is /services/databases/system/ALL\_RESOURCES  If not null then provide the full path to a cached table with the same or similar structure.  Depending on the size of ALL\_RESOURCE, performance would be better with a cached view.  Required columns include: RESOURCE\_ID, RESOURCE\_NAME, PARENT\_PATH, RESOURCE\_TYPE, RESOURCE\_SUBTYPE | VARCHAR(4000) |
| IN | preserveTargetDsOnError - [optional] - If there is an exception, preserve the target datasource and do not delete it.  Y=preserve target datasource on error  N=delete target datasource on error  Preserving the datasource may be useful for debugging what went wrong. For example, what if the wrong password was passed in and the datasource could not connect. It would be useful to be able to open the datasource and debug what went wrong in order to fix the process. | CHAR(1) |
| OUT | result  RESOURCE\_MESSAGE VARCHAR, -- the child resource message - SUCCESS, FAIL  REBIND\_ACTION VARCHAR, -- the result of the rebind action: depnum||' RESOURCES REBOUND'  CACHE\_CONFIG\_ACTION VARCHAR, -- the result of the rebind action: cachenum||' CACHE CONFIG'  FROM\_RESOURCE\_PATH VARCHAR(4000), -- the from resource path  TO\_RESOURCE\_PATH VARCHAR(4000), -- the from resource path  RESOURCE\_TYPE VARCHAR, -- the resource type  RESOURCE\_SUBTYPE VARCHAR, -- the resource subtype  DEPENDENT\_REBIND\_LIST LONGVARCHAR, -- the list of dependent resources for rebind actions  DEPENDENT\_CACHE\_LIST LONGVARCHAR -- the list of dependent resources for cache config actions | |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | debug2 | N |
| IN | debug3 | N |
| IN | sourceDatasourcePath | /shared/A\_TEST1/DS\_sqlserver\_2016\_V1 |
| IN | targetDatasourcePath | /shared/A\_TEST1/DS\_sqlserver\_2016\_V2 |
| IN | tdvServerBackupFolder | C:\tmp\carfile\_backup |
| IN | encryptionPassword | tibco1 |
| IN | targetOverride | N |
| IN | targetCopyPrivileges | Y |
| IN | targetAdapterName | Microsoft SQL Server 2019 |
| IN | targetAttributesOverride | [delimiter=||]password=Password1|| |
| IN | excludeAttributes |  |
| IN | allResourcePath |  |
| IN | preserveTargetDsOnError | N |
| OUT | Result | Too large to display |

### moveResource

This is a procedure is used to move a single resource from the source to a target.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off. | CHAR(1) |
| IN | sourceResource - DV source resource path. Resource must exist. | VARCHAR(4000) |
| IN | sourceType - the type of resource. | VARCHAR |
| IN | targetResource - DV target resource path. Gets created if it does not exist. | VARCHAR(4000) |
| IN | overwrite - If a resource exists in the target container with the same name and type of the target resource and overwrite is "1", then the resource within the target container will be overwritten. If overwrite is "0", then a DuplicateName fault will be generated and resource will not be moved. | BIT |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceResource | /shared/examples/CompositeView |
| IN | sourceType | TABLE |
| IN | targetResource | /shared/examples/CompositeView2 |
| IN | overwrite | 1 |
| OUT | success | 1 |
| OUT | faultResponse | null |

### parseAnnotationLabel

This procedure is used to parse a resource annotation for a particular release label string found in the resource annotation.

Rules:

1. The annotation requires a uniform strategy in which a unique label can be determined using a simple search method.

2. The "Label" is unique and is the key search term. If that search term is present then the resource is chosen for the output list.

3. The "Release Date" is simply provides additional information to developers regarding the date of the release but is not used for searching.

4. The "command Extension" provides additional insight into what other deployment metadata files should be generated based on the type of command and the context of the resource. The CoE deployment script deployProject.sh|bat would utilize these extensions.

5. No spaces are allowed in the label or the release date.

6. Each label should be a row which is delimited by a newline also known as CHR(10)=linefeed.

7. The command=command text may be a pipe-separated list up to 4 commands.

Scenario 1: 2 chained commands are known about which is introspect|design=varchar(255):abc

Scenario 2: 2 chained commands are known about which is reintrospect|relocate=DVKPI7/dbo DVKPI8/DVKPI, DVKPI7 DVKPI8

8. Commands are applicable to certain resource types and only should be used in their annotations.

* introspect – only applicable with data source children like database tables and procedures. Introspect will add new resources into TDV if they don't exist and update existing resources with changes.

Introspect ignores a resource if there are no changes.

The format of the command is simply introspect.

example: introspect

* reintrospect – only applicable with data source resources.

The format of the command is simply reintrospect.

example: reintrospect

* design – only applicable with data source procedure children requiring design-by-example.

Specify a comma-separated list of parameters for the procedure.

Enclose "type:value" within double quotes if the value contains a comma. example: "varchar(50):a, list, of, words"

The format of the command is design= "type:value", "type:value", "type:value"

example: design=VARCHAR(255):abc,"varchar(100):a,list,of,words"

* rebind – applicable to views that need to be rebound to a different data source catalog or schema upon import.

The format of the command is rebind=from-tdv-path to-tdv-path

User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"

example: rebind=/shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI7/dbo /shared/A\_TEST1/DS\_sqlserver\_2016/DVKPI8/DVKPI

* relocate – applicable to data sources where the catalog and/or schema needs to be relocated to a different name upon import.

The format of the command is relocate=catalog1/schema1 catalog2/schema2, catalog1 catalog2

Always specify the relative schema path first and then the relative catalog using comma-separated lists.

Leading / is optional for each relative path

The context is the data source where the annotation exists.

User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"

example: relocate=DVKPI7/dbo DVKPI8/DVKPI, DVKPI7 DVKPI8

* remove - this command can only be specified in the label of a container or datasource. A container may be folder, catalog or schema.

The format of the remove command is remove=resource:type. The type is either the internal type or the type from the Studio info tab (display type).

To remove multiple resources within the same container use the format remove=resource:type,resource:type,resource:type.

The resource name is case sensitive. The type is not case sensitive.

User double quotes around resource names with spaces, commas or forward slahes. e.g. "my view", "my,vew" or "sap\_hana/table1"

If removing a folder then the label of the parent folder must specify the remove.

If removing a database table or procedure then the remove command must go into the direct parent folder which is typically the schema container but it could be the data source if it has not schemas.

If removing a database schema then the remove command goes one level up into the catalog container if it exists or the datasource.

* export - only applicable to containers and datasources. A container may be folder, catalog or schema.

The export command will be ignored on any other type of resource.

Normally, export is assumed if a matching label is found on search and no other commands are specified.

However the export command should be specified in the label of a folder/container/datasource if there are other commands present in order to distinguish what the user intends to accomplish.

When the intention is to execute multiple commands such as export|remove=resource:type or export|reintrospect for data sources then the export is registered along with the other commands.

When the intention is not to export but a label is present along with other commands such as remove=resource:type then only that command is registered and not the export.

The following is an example only:

Label Release Date command extension

V0000001 2021-11-07 command|command=optional command text

V0000002 2021-11-08 introspect

V0000003 2021-11-09 reintrospect

V0000004 2021-11-10 design=VARCHAR(255):abc,"varchar(100):a,list,of,words"

V0000005 2021-11-11 relocate=dvops/dbo dvops/db1,dvops dvops2

V0000006 2021-11-12 rebind=/shared/TEST/DS1/dvops/dbo /shared/TEST1/DS1/dvops2/db1

V0000007 2021-11-13 remove=proc1:procedure,view1:view

V0000008 2021-11-14 export

V0000009 2021-11-15

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | searchStr - The label string to search for within the annotation. e.g. V0000001 | LONGVARCHAR |
| IN | annotation - Resource annotation with linefeed to deliniate label lines. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | searchStr | V0000008 |
| IN | annotation | V0000001 2021-11-07 command|command=optional command text  V0000002 2021-11-08 introspect  V0000003 2021-11-09 reintrospect  V0000004 2021-11-10 design=VARCHAR(255):abc,"varchar(100):a,list,of,words"  V0000005 2021-11-11 relocate=dvops/dbo dvops/db1,dvops dvops2  V0000006 2021-11-12 rebind=/shared/TEST/DS1/dvops/dbo /shared/TEST1/DS1/dvops2/db1  V0000007 2021-11-13 remove=proc1:procedure,view1:view  V0000008 2021-11-14 export  V0000009 2021-11-15 |

### performServerAction

This procedure is used to perform a server action.

Perform a server action. Use "getServerActions" to get a complete list of available actions and their attribute definitions. Get a list of all available server actions and their definitions.

* ClearDataSourceConnectionPools - dataSourcePaths: Resource paths to the data sources for which the connection pools will cleared.
* ClearQueryPlanCache - Clear and resets the query plan cache.
  + Faults:
  + Security: The caller must have ACCESS\_TOOLS, MODIFY\_ALL\_STATUS rights.
* ClearRepositoryCache - Clear and resets the repository cache.
* TerminateSessions - sessionIds: The session ID's of the sessions to be terminated.
* TerminateTransactions - transactionIds: The transaction ID's of the transactions to be terminated.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | serverAction - this is the server action to perform. | VARCHAR |
| IN | serverAttributes - XML value for the server attributes | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | serverAction | ClearQueryPlanCache |
| IN | serverAttributes | null |

### publishResource

Publish a resource as a database object or web service operation.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath - The full path to the resource to be published | ResourceDefs.ResourcePath |
| IN | resourceType - Type of CIS resource to be published | ResourceDefs.ResourceType |
| IN | publishFolder - Folder to publish the resource to | ResourceDefs.ResourcePath |
| IN | publishName - The name of the published resource | ResourceDefs.ResourceName |
| IN | annotation - The annotation to be placed in the "annotation" field of the published resource | LONGVARCHAR |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | /shared/examples/CompositeView |
| IN | resourceType | TABLE |
| IN | publishFolder | /services/databases/examples |
| IN | publishName | CompositeView |
| IN | annotation | Any text |
| OUT | success | 1 |
| OUT | createResponse | null |
| OUT | faultResponse | null |

### rebindFolder

This procedure provides the capability to rebind all the resources in a folder. For example, if a View points to a data source table, you may want to rebind to a different data source that has the same structure. This may be useful when redeploying from Dev to Test to Production or simply rebinding to a different development instance of the database. Rules:

1) If a resource in the folder has both the source and the target sources present, it will use rebindResource to do an explicit rebind.

2) If a resource in the folder does not have the source present, it will rebind using explicit text modification techniques instead of rebindResource. The following text modification techniques are supported for the given resource type:

resourceType = 'TABLE'

subtype = 'SQL\_TABLE' -- Regular View not a database table

resourceType = 'PROCEDURE'

subtype = 'SQL\_SCRIPT\_PROCEDURE' -- Custom Procedure or Parameterized query

subtype = 'EXTERNAL\_SQL\_PROCEDURE' -- Packaged Query Procedure

subtype = 'BASIC\_TRANSFORM\_PROCEDURE' -- XSLT Basic Transformation definition

subtype = 'XSLT\_TRANSFORM\_PROCEDURE' -- XSLT Transformation text

subtype = 'STREAM\_TRANSFORM\_PROCEDURE' -- XSLT Stream Transformation text

3) If a resource in the folder does not have the target present, that is an error and an exception is raised.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | startingResourceFolder | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | rebindFromFolder | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | rebindToFolder | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| OUT | success | BIT |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | startingResourceFolder | ‘/shared/examples/rebind’ |
| IN | rebindFromFolder | ‘/shared/examples/ds\_orders’ |
| IN | rebindToFolder | ‘/shared/examples/ds\_orders\_Copy1’ |
| OUT | success | 1 or 0 |
| OUT | faultResponse | XML not shown here |

### rebindResource

This procedure provides the capability to rebind the resources inside of the requested resource. For example, if a View points to a data source table, you may want to rebind to a different data source that has the same structure. This may be useful when redeploying from Dev to Test to Production or simply rebinding to a different development instance of the database.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| IN | rebindVector | VECTOR (rebindRow ROW (  oldPath TypeDefinitions.pathType,  oldType VARCHAR(255),  newPath TypeDefinitions.pathType,  newType VARCHAR(255)  )) |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/PRODUCT\_VIEW’ |
| IN | resourceType | ‘TABLE’ |
| IN | rebindVector | {('/shared/examples/ds\_inventory/products',  'TABLE',  '/shared/ASAssets/Utilities/repository/examples/source/ds\_inventory/products',  'TABLE' )} |
| OUT | success | 1 or 0 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### recoverFailedCacheRefresh

Occasionally a cache refresh request will exit without CIS noticing it. This script clears the cache status that says a refresh is "in progress" so that CIS will be able to kick off a new refresh.

Because the data will presumably be in an inconsistent state if an incremental cache refresh is cancelled, this script will change the "in progress" state to "failed" so that any future incremental refresh requests will be forced to do a full refresh.

DO NOT run this on resources whose cache refreshes really are in progress.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inResourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | inResourceType | /lib/resource/ResourceDefs.ResourceType |
| OUT | resultCode | INTEGER |
| OUT | resultMessage | VARCHAR(65536) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inResourcePath | ‘/shared/examples/ds\_orders/orders’ |
| IN | inResourceType | ‘TABLE’ |
| OUT | resultCode | 0 |
| OUT | resultMessage | ‘Cache status update successful.’ |

### refreshResourceStatistics

Refreshes the statistics on a resource. For enabling statistics gathering, also see repository/updateResourceStatisticsConfig. Cardinality statistics must be configured on the Data Source. You can execute against a data source or a table/view in the data source.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/ds\_inventory/products’ |
| IN | resourceType | ‘TABLE’ |
| OUT | success | 1 or 0 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### reintrospectDataSource

This procedure starts either a blocking or non-blocking data source introspection.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | fullResourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | isBlocking | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | fullResourcePath | '/shared/examples/ds\_orders' |
| IN | isBlocking | 1 |

### removeAllFolders

This procedure removes one or more folders and their respective contents. The parameter fullResourcePathList should be a comma separated list of full folder paths to remove.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | fullResourcePathList | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | fullResourcePathList | '/shared/myfolder1,/shared/myfolder2' |

### removePathQuotes

This procedure removes all quote characters from a resource path string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | LONGVARCHAR |
| OUT | resultPath | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/"XML"/"path with spaces"/test' |
| OUT | resultPath | '/shared/XML/path with spaces/test' |

### renameResource

This is a procedure is used to rename a single resource from the source to a target.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceResource - source resource path. Resource must exist. | VARCHAR(4000) |
| IN | sourceType - the type of resource. e.g. CONTAINER, TABLE, PROCEDURE, DATA\_SOURCE, TRIGGER | VARCHAR |
| IN | targetName - target resource name. | VARCHAR |
| OUT | success - true(1) or false(0) | BIT |
| OUT | faultResponse - null if successful otherwise contains a fault response. Values: Fault response XML value | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceResource | /shared/examples/ViewOrder |
| IN | sourceType | TABLE |
| IN | targetName | ViewOrder1 |
| OUT | success | 1 |
| OUT | faultResponse | null |

### replaceStringInAnnotations

This procedure is used to replace a target string in the annotations of one or more resources contained under the resource identified by the value “startPath”. The "searchStr" is case sensitive and used like a wild card on the annotation. If the "searchStr" is null then this procedure will only search for null or empty annotations and update the annotation with the text in "replaceStr". Conversely, if "replaceStr" is null and this procedure will update found resources based on "searchStr" text and set the annotation to null effectively removing the annotation. The “startPath” is a source folder path to assess and fix and has the following rules:

Values: folder exact match: /shared/tmp/1folder/\_folder/XML – use this scenario to only search for what is in this exact folder and no sub-folders.

Values: wildcard begin: %/1folder/\_folder/XML

Values: wildcard end: /shared/tmp/1folder/\_folder/XML% - use this scenario to iterate through sub-folders.

Values: wildcard surrounding: %/1folder/\_folder/XML%

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | startPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | startResourceType [deprecated] | /lib/resource/ResourceDefs.ResourceType |
| IN | searchStr – the string to search for what is to be removed/replaced in the target resource. The string search is is case sensitive but is searched like a wildcard so the text can occur anywhere in the annotation. The string may contain multi-line text. If this string is null then specifically search for null or empty annotations. | LONGVARCHAR |
| IN | replaceStr – the new string in which to replace with. The string may contain multi-line text. If this string is null then set the target annotation to null. | LONGVARCHAR |
| OUT | success | BIT |
| OUT | updateResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: The folder /shared/ASAssets/Utilities/examples has an annotation of “This folder contains pre-created resources.”**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | startPath | ‘/shared/examples’ |
| IN | startResourceType [deprecated] | NULL |
| IN | searchStr | ‘folder’ |
| IN | replaceStr | ‘container’ |
| OUT | success | 1 |
| OUT | updateResponse | [XML response] |
| OUT | faultResponse | NULL |

### replaceStringInResources

This procedure is used to replace a target string in the body of one or more resources contained under the resource identified by the value “startPath”. This procedure only looks at views and procedures as they may have a script body whereas other resources do not. All occurrences of the string are replaced. A related procedure, “/Utilities/repository/replaceStringInAnnotations”, is used to replace strings in the annotations of a resource.

Values: folder exact match: /shared/tmp/1folder/\_folder/XML – use this scenario to only search for what is in this exact folder and no sub-folders.

Values: wildcard begin: %/1folder/\_folder/XML

Values: wildcard end: /shared/tmp/1folder/\_folder/XML% - use this scenario to iterate through sub-folders.

Values: wildcard surrounding: %/1folder/\_folder/XML%

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | startPath | /lib/resource/ResourceDefs.ResourcePath |
| IN | searchStr – the string to search for what is to be removed/replaced in the target resource. The string may contain multi-line text. | LONGVARCHAR |
| IN | replaceStr – the string to replace with. It may contain multi-line text. It may not be null but it can be an empty string if the use case is to remove the “searchStr” text. | LONGVARCHAR |
| IN | caseSensitive – 1/null [default]=Search for “searchStr” with case sensitivity (actual text as is). 0=Perform search with no case sensitivity. | BIT |
| OUT | numResourcesUpdated | INTEGER |
| OUT | success | BIT |
| OUT | updateResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: The resource /shared/ASAssets/Utilities/examples/xml/getNodeFromXML\_SAVE has an annotation of “CSW Version.”**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | startPath | ‘/shared/examples’ |
| IN | startResourceType | ‘CONTAINER’ |
| IN | searchStr | ‘CSW Version’ |
| IN | replaceStr | ‘DV Version’ |
| OUT | success | 1 |
| OUT | updateResponse | [XML response] |
| OUT | faultResponse | NULL |

### returnFolderNameAndFolderPath

Return the root folder name and the remaining folder path. Used for traversing folder structures either top down or bottom up. This procedure is also used when creating a folder structure from beginning to end.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | mode | CHAR(1) |
| OUT | folderName | VARCHAR(255) |
| OUT | folderPath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository’ |
| IN | mode | ‘B’ |
| OUT | folderName | 'shared' |
| OUT | folderPath | '/Utilities/repository' |

### searchAnnotations

This procedure is used to search a target string in the annotations of one or more resources contained under the resource identified by the value startPath. The search string may be multi-line text. The input parameter "caseSensitive" allows the user to determine if they want to search for annotations using case sensitivity. Allow the starting path to contain a colon-separated list of paths. Use a wild-card % symbol to select all child paths for the base path. The base path is also included in the evaluation. e.g. /services/databases/myprojectDB%:/services/webservices/myproject%:/shared/myproject%

Values: folder exact match: /shared/tmp/1folder/\_folder/XML – use this scenario to only search for what is in this exact folder and no sub-folders.

Values: wildcard begin: %/1folder/\_folder/XML

Values: wildcard end: /shared/tmp/1folder/\_folder/XML% - use this scenario to iterate through sub-folders.

Values: wildcard surrounding: %/1folder/\_folder/XML%

Example 1: search labels in myproject: /services/databases/myprojectDB%:/services/webservices/myproject%:/shared/myproject%

Example 2: search labels in repository: /services/databases%:/services/webservices%:/shared%:/policy%

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | startPath | LONGVARCHAR |
| IN | searchStr – the string to search for in the target resource. The string may contain multi-line text. If the string is null then specifically search for null or empty annotations. May be a colon-separated list of paths that may contain a wildcard %. | LONGVARCHAR |
| IN | caseSensitive – 1/null [default]=Search for “searchStr” with case sensitivity (actual text as is). 0=Perform search with no case sensitivity. | BIT |
| OUT | result PIPE (  resourceId - The resource identifier  resourcePath - The full path to the resource  resourceType - The type of resource  subtype - The sub-type of the resource  owner - The resource owner  ownerId - The resource owner id  len - The length of the annotation  annotation - The resource text that where the keywordList was found  ) | BIGINT  VARCHAR(4000)  VARCHAR(255)  VARCHAR(255),  VARCHAR(255)  BIGINT  INTEGER  LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: The folder /shared/ASAssets/Utilities/examples/string.**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | startPath | ‘/shared/ASAssets/Utilities/examples/string’ |
| IN | searchStr | ‘All rights reserved’ |
| IN | caseSensitive | 1 |
| OUT | result | Cursor of 3 rows |

### searchResources

This procedure searches for keywords within the script text of a resource. It returns a cursor of resource paths, types and subtypes for resources that contain the keywords. It also returns the text position of the occurrence of any keyword matches and which keyword was matched. It will not search a resource that is impacted. The impactLevel may be 'NONE' or 'SYNTAX\_ERROR'. It will skip starting resource paths that do not exist.

The following resource types and sub-types are supported:

resourceType = 'PROCEDURE'

subtype = 'SQL\_SCRIPT\_PROCEDURE' -- Get Regular Procedure

subtype = 'EXTERNAL\_SQL\_PROCEDURE' -- Get Packaged Query Procedure

subtype = 'XSLT\_TRANSFORM\_PROCEDURE' -- Get XSLT Transformation text

subtype = 'XQUERY\_TRANSFORM\_PROCEDURE' -- Get XQuery Transformation text

resourceType = 'TABLE'

subtype = 'SQL\_TABLE' -- Get Regular View

This procedure uses RegexPosition which has the following rules:

Finds an occurrence of a regular expression match in a VARCHAR and returns the position of the match (similar to the SQL POSITION function, positions start at 1 with 0 indicating a match was not found.) The value of the occurrence input value determines which occurrence to return a numbered occurrence starting at 1 from left to right. (Use negative values to number occurrences from right to left.) If a NULL value is passed in as the value of any of the inputs, a NULL is returned. Zero may not be used as a value for an occurrence.

The regular expression language used is what is supported by the JDK used by CIS (currently 1.5 in CIS 4.0.1) See the javadoc for java.util.regex.Pattern for details on what is supported. The following characters require an escape to be used in front of the character: '\', '(', ')', '[', '{', '?', '\*', '+','''. The escape character is a backslas “\”. For example: \\, \(, \), \[, \{, \?, \\*, \+, \’

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **startingFolders** - comma separated list of starting folder to begin searching. | LONGVARCHAR |
| IN | **keywordList** - comma separated list of keyword strings to search for. If the string contains a comma then it must be enclosed in double quotes like "a,b",c,"d,". The keyword list may also contain regular expressions. For example to search for the data type Integer or integer but not INTEGER use the regular expression [Ii]nteger in the keyword list. The following characters in the text require an escape using a backslash “\” character: '\', '(', ')', '[', '{', '?', '\*', '+',''' | LONGVARCHAR |
| IN | **keywordOccurrence** - comma separated list of keyword occurrences which exactly match the number of keywords. If this entry is left null then it is assumed that the occurrence for each keyword is 1. The value of the occurrence input value determines which occurrence to return (numbered starting at 1 from left to right. Use negative values to number occurrences from right to left.) | LONGVARCHAR |
| OUT | result | OUT result PIPE (  **startOrder** INTEGER, -- The starting path execution order  **startPath** VARCHAR(1024), -- The starting path extracted from the comma separated startingFolderss  **resourcePath** VARCHAR(1024), -- The full path to the resource  **resourceType** VARCHAR(255), -- The type of resource  **subtype** VARCHAR(255), -- The sub-type of the resource  **impactLevel** VARCHAR(255), -- The impact level of the resource NONE and SYNTAX\_ERROR are permitted.  **pos** INTEGER, -- The position of the first occurrence of any of the keywords  **keyword** LONGVARCHAR, -- The keyword that was found from the keyword list  **keywordNum** INTEGER, -- The keyword position number within the keyword list  **occurrence** INTEGER – The occurrence of the keyword  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | **startingFolders** | /shared/examples, /shared/ASAssets/Utilities/repository/examples |
| IN | **keywordList** | “ProductID,”, [Ii]nsert, ProductName |
| IN | **keywordOccurrence** | 1,2,-1 |
| OUT | **result** | See Chart 1 below |

* 1. Chart 1: columns 1-3

|  |  |  |
| --- | --- | --- |
| **startOrder** | **startPath** | **resourcePath** |
| 1 | /shared/examples | /shared/examples/CompositeView |
| 1 | /shared/examples | /shared/examples/LookupProduct |
| 1 | /shared/examples | /shared/examples/ViewOrder |
| 1 | /shared/examples | /shared/examples/ViewSupplier |
| 1 | /shared/examples | /shared/examples/getInventoryTransactions |
| 1 | /shared/examples | /shared/examples/productCatalog\_Transformation |
| 2 | /shared/ASAssets/Utilities/repository/examples | /shared/ASAssets/Utilities/repository/examples/returnGetChildResourcesResponseXML |
| 2 | /shared/ASAssets/Utilities/repository/examples | /shared/ASAssets/Utilities/repository/examples/test\_multiple\_cursors |
| 2 | /shared/ASAssets/Utilities/repository/examples | /shared/ASAssets/Utilities/repository/examples/test\_searchResources |
| 2 | /shared/ASAssets/Utilities/repository/examples | shared/Utilities/repository/examples/test\_updateResourcesSqlTable |

* 1. Chart 1: columns 4-9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **resourceType** | **subtype** | **pos** | **keyword** | **keywordNum** | **occurrence** |
| TABLE | SQL\_TABLE | 41 | ProductName | 3 | -1 |
| PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 277 | ProductID, | 1 | 1 |
| TABLE | SQL\_TABLE | 53 | ProductID, | 1 | 1 |
| TABLE | SQL\_TABLE | 35 | ProductID, | 1 | 1 |
| PROCEDURE | XQUERY\_TRANSFORM\_PROCEDURE | 2068 | ProductName | 3 | -1 |
| PROCEDURE | XSLT\_TRANSFORM\_PROCEDURE | 3072 | ProductName | 3 | -1 |
| PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 10578 | ProductName | 3 | -1 |
| PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 850 | [Ii]nsert | 2 | 2 |
| PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 919 | ProductID, | 1 | 1 |
| PROCEDURE | SQL\_SCRIPT\_PROCEDURE | 2300 | ProductName | 3 | -1 |

### translateResourceType

This procedure is used to translate TDV display type on the info screen to internal resources types.

**Display Type Internal Type Location**

Hierarchical TREE /shared

Trigger TRIGGER /shared

Model MODEL /shared

Published Resource LINK /services

XML Schema Definition Set DEFINITION\_SET /shared

SQL Definition Set DEFINITION\_SET /shared

Web Service Definitions DEFINITION\_SET /services

Data Source DATA\_SOURCE /shared or /services

Datasource DATA\_SOURCE /shared or /services

Composite Database DATA\_SOURCE /services

Composite Web Service DATA\_SOURCE /services

Legacy Web Service DATA\_SOURCE /services

Folder CONTAINER /shared or /services

Catalog CONTAINER /shared or /services

Schema CONTAINER /shared or /services

Web Service Service CONTAINER /services

Web Service Port CONTAINER /services

Web Service Operations CONTAINER /services

Policy POLICY /policy

Cache Policy POLICY /policy

Metrics POLICY /policy

**All are tables:** TABLE: /shared

View

Table

Delimited File

Composite SQL View

Database Table

Excel Table

System Table

SAP RT Table

SAP RFC Table

SAP AQ Query Table

SAP Infoset Query Table

Siebel Table

**All are procedures**: PROCEDURE: /shared

Script

Procedure

Packaged Query

Java Procedure

Web Service Operation /shared or /services

Database Stored Procedure

Composite SQL Script Procedure

Parameterized Query

XQuery Procedure

XSLT Procedure

Transform Procedure

Transformation

Basic Transform Procedure

Basic Transformation

Stream Transform Procedure

Stream Transformation

XQuery Transform Procedure

XQuery Transformation

XSLT Transform Procedure

XSLT Transformation

XQuery Transformation

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resTypeIn | VARCHAR |
| OUT | resTypeOut | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resTypeIn | View |
| OUT | resTypeOut | VARCHAR |

### updateBasicTransformationProcedure

This procedure is used to update a basic transformation procedure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | transformSourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | transformSourceType | VARCHAR |
| IN | annotation | LONGVARCHAR |
| IN | attributeVector | RepositoryDefinitions.AttributeCompleteVectorType |
| OUT | Success | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples/basicXSLT' |
| IN | transformSourcePath | '/shared/examples/ds\_XML/productCatalog.xml' |
| IN | transformSourceType | 'TREE' |
| IN | annotation | 'Product catalog transformation' |
| IN | attributeVector | NULL |
| OUT | success | 1 |

### updateConnector

This procedure updates a JMS connector

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | name | VARCHAR(100) |
| IN | groupName | VARCHAR(100) |
| IN | jmsClientID | VARCHAR(1024) |
| IN | annotation | VARCHAR(1024) |
| IN | jndiContextFactory | VARCHAR(1024) |
| IN | jndiProperties | LONGVARCHAR |
| IN | jndiProviderUrl | VARCHAR(1024) |
| IN | jndiUser | VARCHAR(50) |
| IN | jndiPassword | VARCHAR(50) |
| IN | queueConnectionFactory | VARCHAR(1024) |
| IN | minPool | INTEGER |
| IN | maxPool | INTEGER |
| IN | poolTimeout | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | name | 'myMQ' |
| IN | groupName | '<Group Name>' |
| IN | jmsClientID | '<JMS Client ID>' |
| IN | annotation | 'This is a JMS message queue' |
| IN | jndiContextFactory | '<JNDI context factory>' |
| IN | jndiProperties | '<JNDI Properties XML>' |
| IN | jndiProviderUrl | '<JNDI Provider URL>' |
| IN | jndiUser | 'myMQuser' |
| IN | jndiPassword | 'myMQpassword' |
| IN | queueConnectionFactory | '<Queue Connection Factory>' |
| IN | minPool | 1 |
| IN | maxPool | 10 |
| IN | poolTimeout | 300 |

### updateDefSetDef

Programmatically updates an entry in a definition set. Inserting a definition that already exists will perform an update instead. Updating a definition that does not exist will do nothing.

NOTE: Updating data types is apparently not supported by the Admin API, so you must first delete then insert to update data types.

Usage Note: The calling user must have:   
 - The ACCESS\_TOOLS right   
 - Read and write permission on the definition set  
 - Read permission on any of the definition set’s parent folders

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | defSetPath | VARCHAR(4096) |
| IN | updateType  values: ‘INSERT’, ‘UPDATE’, or ‘DELETE’ | VARCHAR(255) |
| IN | defName | VARCHAR(255) |
| IN | defType  values: ‘EXCEPTION\_DEFINTION’, ‘TYPE\_DEFINITION’, or ‘CONSTANT\_DEFINITION’ | VARCHAR(255) |
| IN | dataType (NULL for exception definitions) | VARCHAR(255) |
| IN | defValue (NULL for exception or type definitions) | VARCHAR(255) |
| OUT | None: Throws exception upon failure |  |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | defSetPath | ‘/shared/ASAssets/Utilities/repository/examples/testDefSet’ |
| IN | updateType | ‘INSERT’ |
| IN | defName | ‘MyNewConstant’ |
| IN | defType | ‘CONSTANT\_DEFINITION’ |
| IN | dataType | ‘VARCHAR(255)’ |
| IN | defValue | ‘Hello World!’ |

### UpdateDsColumnAnnotation

This procedure is used to update annotations for data source table columns since there is no Admin API this particular operation. For other types of table/view columns, please use the updateSqlTable() admin API.) This is a wrapper script that automatically detects which version of CIS is running and calls the appropriate CJP.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | column\_path | LONGVARCHAR |
| IN | annotation | LONGVARCHAR |
| OUT | result | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Dependency on** configureReservedList

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | column\_path | '/shared/examples/ds\_orders/orders/OrderID' |
| IN | Annotation | 'OrderID column annotation' |
| OUT | Result | 'Column annotation updated.' |

### updateExternalSQLProcedure

This procedure is used to update a packaged query procedure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | externalSqlText | LONGVARCHAR |
| IN | externalDatasourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | parameterVector | RepositoryDefinitions.AttributeCompleteVectorType |
| IN | annotation | LONGVARCHAR |
| IN | attributeVector | RepositoryDefinitions.AttributeCompleteVectorType |
| OUT | success | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples/packagedCustomerQuery' |
| IN | externalSqlText | 'select \* from customers' |
| IN | externalDatasourcePath | '/shared/examples/ds\_orders' |
| IN | parameterVector | NULL |
| IN | annotation | 'Customer information' |
| IN | attributeVector | NULL |
| OUT | success | 1 |

### updateImpactedResource

This procedure is used to update a single impacted resource. The main objective is to work around issues in CIS that are fixable by simply opening, modifying and saving a resource. Typical issues that this can repair are views or procedures that are impacted after import into CIS. Typical error messages might be "session may not be null" or "session is closed". This procedure will not attempt to fix views or procedures that contain an impact level of "SYNTAX\_ERROR". Moved from Best Practices to Utilities.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Debug – Y or N | CHAR(1) |
| IN | resourcePath | LONGVARCHAR |
| IN | resourceType – TABLE or PROCEDURE | VARCHAR |
| OUT | success – 0 or 1 | BIT |
| OUT | message – exception message if success=0 otherwise null | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N’ |
| IN | resourcePath | ‘/shared/project/myview’ |
| IN | resourceType | TABLE |
| OUT | success | 1 |
| OUT | message | null |

### updateImpactedResources

This procedure crawls through a starting folder and attempts to update a VIEW or PROCEDURE resource that is impacted. There are some errors that can be resolved that are not syntax related errors. There are some Composite versions that will mark a resource as impacted. Simply reading those resources and saving them back out fixes the impacted issue. This procedure attempts to automate that process. An example of an error is "session is null". This type of error can be fixed simply by reading and writing the resource.

**inExcludePathsKeywords** is a comma separated list of keywords used to exclude paths containing these any of the keywords (case insensitive.) Examples: Analysis, Archive, save, validation

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | inStartingFolders | LONGVARCHAR |
| IN | inExcludePathsKeywords | LONGVARCHAR |
| OUT | success | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N’ |
| IN | inStartingFolders | ‘/shared/myImportedFolder’ |
| IN | inExcludePathsKeywords | NULL |
| OUT | success | 1 |

### updateResourceAnnotations (deprecated)

Use updateResourceAnnotationsV2.

### updateResourceAnnotationsV2

Updates a resource's annotations, including columns (if any.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType |
| IN | annotation | LONGVARCHAR |
| IN | inColumnAnnotations | LONGVARCHAR |
| OUT | success | BIT |
| OUT | updateResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples/ds\_orders/orders' |
| IN | resourceType | 'TABLE' |
| IN | annotation | 'this is a table annotation' |
| IN | inColumnAnnotations | <resource:column>  <resource:name>productid</resource:name>  <resource:annotation>Product identifier</resource:annotation>  </resource:column>  <resource:column>  <resource:name>productname</resource:name>  <resource:annotation>Product name</resource:annotation>  </resource:column>  <resource:column>  <resource:name>productdescription</resource:name>  <resource:annotation></resource:annotation>  </resource:column>  <resource:column>  <resource:name>categoryid</resource:name>  <resource:annotation>The category identifier</resource:annotation>  </resource:column>  <resource:column>  <resource:name>serialnumber</resource:name>  <resource:annotation>The serial number</resource:annotation>  </resource:column> |
| OUT | success | 1 |
| OUT | updateResponse | (Update response XML) |
| OUT | faultResponse | NULL |

### updateResourceCacheConfig

This procedure updates a resource's cache configuration setting.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| IN | cacheConfigured | VARCHAR(255) |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/proc1’ |
| IN | resourceType | ‘PROCEDURE’ |
| IN | cacheConfigured | 'true' |
| OUT | success | 1 or 0 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateResourceCacheConfiguration (deprecated)

Use updateResourceCacheConfigurationV2.

### updateResourceCacheConfigurationV2

Sets a resource's cache configuration. Other than "inResourcePath" and "inResourceType", any value can be set to NULL.

NOTE: Only supports configuring resources with a single cursor or a set of scalar outputs. Use /services/webservices/system/admin/resource/operations/updateResourceCacheConfig directly, otherwise.

NOTE 2: Does NOT create any database tables necessary for the storage of cache data. This must still be done using the Studio GUI or by hand-edited DDL on the caching data source itself.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inResourcePath - Full resource path which includes the path and the resource name. | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | inResourceType - Type of DV resource to be created. | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1) |
| IN | inConfigured - Indicates whether the resource has a cache configured or not.  Values: 1=cache configured, 0=cache not configured | BIT |
| IN | inEnabled - Indicates enabled state of cache  Values: 1=cache enabled, 0=cache disabled | BIT |
| IN | inStorageMode - Storage type used for the cache.  Values: May be either "AUTOMATIC", "DATA\_SOURCE", "DATA\_SOURCE\_OTPS" | VARCHAR(255) |
| IN | inStorageDataSourcePath - If the mode is "DATA\_SOURCE", this identifies the path to the data source being used to store cache data.  Values: Path to a cache configured data source. | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN | inStorageTargets - The list of database tables used as storage targets.  Values: An XML formatted string containing the list of target resources. It must conform to this XML format. | LONGVARCHAR |
| IN | inRefreshMode - How the cache should be refreshed.  Values: May be "MANUAL" or "SCHEDULED". | VARCHAR(255) |
| IN | inScheduleMode - Scheduling mode. Values: If 'inRefreshMode" is "SCHEDULED", this should always be "INTERVAL". | VARCHAR(255) |
| IN | inStartTime - Start timestamp for the schedule  Values: If 'inRefreshMode" is "SCHEDULED", A timestamp. | TIMESTAMP |
| IN | inPeriod - The units of measure for the interval between cache refreshes  Values: SECOND, MINUTE, HOUR, DAY, WEEK, MONTH, or YEAR | INTEGER |
| IN | inCount - The number of inPeriod units in the interval between cache refreshes  Values: Any positive integer | INTEGER |
| IN | inExpirationPeriod - The amount of time in milliseconds that the cache will be cleared after it is refreshed.  Values: If less than zero, the period will be set to zero. If zero then the cache will never expire. If set to NULL, the enable setting will be left unaltered. | BIGINT |
| IN | inClearRule - Indicates when old cache data should be cleared.  Values: One of "NONE", "ON\_LOAD", or "ON\_FAILURE". Normally old cache data is cleared on expiration and when a cache refresh successfully completes. In the latter case the old cache data is replaced by the new cached data.  If "NONE", then the normal behavior will be used. If "ON\_LOAD", in addition to the normal behavior the old cache data will be cleared when a refresh is started. If "ON\_FAILURE", in addition to the normal behavior the old cache data will be cleared when a refresh fails. If set to NULL, the setting will be left unaltered. | VARCHAR(255) |
| IN | inIncremental - Indicates whether the cache refresh is full or incremental  Values: 1 (incremental) or 0 (full). Defaults to 0 if NULL. | BIT |
| IN | inStorageBucketMode - Indicates storage mode for multi-table storage. Ignored if storage mode is not 'DATA\_SOURCE\_OTPS'  Values: May be "AUTO\_GEN" or "MANUAL" for multi-table storage. NULL for single-table storage. | VARCHAR(255) |
| IN | inStorageBucketCatalog - Indicates the catalog to use for auto-generated multi-table storage. Ignored if bucket mode isn't "AUTO\_GEN".  Values: A valid catalog name in the cache target database. | VARCHAR(255) |
| IN | inStorageBucketSchema - Indicates the catalog to use for auto-generated multi-table storage. Ignored if bucket mode isn't "AUTO\_GEN".  Values: A valid schema name in the cache target database. | VARCHAR(255) |
| IN | inStorageBucketPrefix - The string to use as the base table name (CIS appends numeric suffixes starting with 0 to each storage target table name.) Ignored if bucket mode isn't "AUTO\_GEN".  Values: Any legal table name. | VARCHAR(255) |
| IN | inStorageBucketNumBuckets - The number of storage target tables to use (CIS then round robin's across each target table in turn. Ignored if bucket mode isn't "AUTO\_GEN".  Values: Any positive integer | INTEGER |
| IN | inStorageDropCreateIdx - Indicates whether or not to drop indexes before the refresh and recreate them afterwards.  Values: 1 (true) or 0 (false) | BIT |
| IN | inFirstRefreshCallback - With full cache refreshes, this is the procedure to run before the refresh. With incremental refreshes, this is the procedure that performs a full refresh.  Values: The path to a procedure | VARCHAR(32768) |
| IN | inSecondRefreshCallback - With full cache refreshes, this is the procedure to run after the refresh. With incremental refreshes, this is the procedure that performs an incremental refresh.  Values: The path to a procedure | VARCHAR(32768) |
| OUT | success | BIT |
| OUT | updateResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: A database table “customers\_cache” has been created in the database pointed to by /shared/examples/ds\_orders with the correct signature for caching the “customers” table in the same database.**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inResourcePath | ‘/shared/examples/ds\_orders/customers’ |
| IN | inResourceType | ‘TABLE’ |
| IN | inConfigured | 1 |
| IN | inEnabled | 1 |
| IN | inStorageMode | ‘DATA\_SOURCE’ |
| IN | inStorageDataSourcePath | ‘/shared/examples/ds\_orders’ |
| IN | inStorageTargets | <resource:storageTargets>  <resource:entry>  <resource:targetName>result</resource:targetName>  <resource:path>/shared/examples/ds\_orders/customers\_cache </resource:path>  <resource:type>TABLE</resource:type>  </resource:entry>  </resource:storageTargets> |
| IN | inRefreshMode | ‘MANUAL’ |
| IN | inScheduleMode | NULL |
| IN | inStartTime | NULL |
| IN | inPeriod | NULL |
| IN | inCount |  |
| IN | inExpirationPeriod | 0 |
| IN | inClearRule | ‘NONE’ |
| IN | inIncremental | 1 |
| IN | inStorageBucketMode | NULL |
| IN | inStorageBucketCatalog | NULL |
| IN | inStorageBucketSchema | NULL |
| IN | inStorageBucketPrefix | NULL |
| IN | inStorageBucketNumBuckets | 0 |
| IN | inStorageDropCreateIdx | 0 |
| IN | inFirstRefreshCallback | '/shared/customersCacheInit' |
| IN | inSecondRefreshCallback | '/shared/customersCacheUpdate' |
| OUT | success | 1 or 0 |
| OUT | updateResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateResourceCacheEnabled

Sets or unsets the “enabled” flag of the input resource’s cache configuration.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | resourceType | VARCHAR(255) |
| IN | cacheEnabled | VARCHAR(255) |
| OUT | success | BIT |
| OUT | updateResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/examples/ds\_orders/orders’ |
| IN | resourceType | ‘TABLE’ |
| IN | cacheEnabled | ‘false’ |
| OUT | success | 1 or 0 |
| OUT | updateResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateResourceDataSource (deprecated)

Use updateResourceDataSourceV2.

### updateResourceDataSourceV2

This procedure is used to update a resource that is a Data Source type. Use this to rebind resources such as an XML Schema and a source input directory for an XML file.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath - full DV source folder path to the resource being updated.  Values: e.g. /shared/exmaples/ds\_XML | VARCHAR(4096) |
| IN | resourceType - Type of DV resource to be created  Values: see "TYPES / SUBTYPES" listing below. e.g. DATA\_SOURCE | VARCHAR(255) |
| IN | resourceSubType - Subtype of the CIS resource to be created  Values: see "TYPES / SUBTYPES" listing below. e.g. XML\_FILE\_DATA\_SOURCE  \* DATA\_SOURCE / RELATIONAL\_DATA\_SOURCE - A relational database source.  \* DATA\_SOURCE / FILE\_DATA\_SOURCE - A comma separate file data source.  \* DATA\_SOURCE / XML\_FILE\_DATA\_SOURCE - An XML file data source.  \* DATA\_SOURCE / WSDL\_DATA\_SOURCE - A Composite web service data source.  \* DATA\_SOURCE / XML\_HTTP\_DATA\_SOURCE - An HTTP XML data source.  \* DATA\_SOURCE / NONE - A custom java procedure data source. | VARCHAR(255) |
| IN | dataSourceType - Type of Data Source  Values: File-XML (See repository/createDataSource)  Use /shared/ASAssets/Utilities/repository/getBasicResourceCursor to get the dataSourceType for a given datasource. | VARCHAR(255) |
| IN | inDataSourceAttrXML - XML list of attributes  set dataSourceAttrXML = dataSourceAttrXML ||  ' <common:attribute xmlns:common="http://www.compositesw.com/services/system/util/common">'|| CHR(10)||  ' <common:name>'||CAST(XMLTEXT(attrName) AS LONGVARCHAR)||'</common:name>'||CHR(10)||  ' <common:type>'||CAST(XMLTEXT(attrType) AS LONGVARCHAR)||'</common:type>'||CHR(10)||  ' <common:value>'||CAST(XMLTEXT(attrValue) AS LONGVARCHAR)||'</common:value>'||CHR(10)||  ' </common:attribute>'||CHR(10)||  ''; | LONGVARCHAR |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/Advisory’ |
| IN | resourceType | ‘DATA\_SOURCE’ |
| IN | resourceSubType | ‘XML\_FILE\_DATA\_SOURCE’ |
| IN | dataSourceType | ‘File-XML’ |
| IN | inDataSourceAttrXML | set dataSourceAttrXML = dataSourceAttrXML ||  ' <common:attribute xmlns:common="http://www.compositesw.com/services/system/util/common">'|| CHR(10)||  ' <common:name>noNamespaceSchemaLocation</common:name>'||CHR(10)||  ' <common:type>STRING</common:type>'||CHR(10)||  ' <common:value>file:/CompositeSoftware/Advisory.xsd</common:value>'||CHR(10)||  ' </common:attribute>'||CHR(10)||  ' <common:attribute xmlns:common="http://www.compositesw.com/services/system/util/common">'|| CHR(10)||  ' <common:name>root</common:name>'||CHR(10)||  ' <common:type>STRING</common:type>'||CHR(10)||  ' <common:value>file:/CompositeSoftware</common:value>'|| CHR(10)||  ' </common:attribute>'||CHR(10)||  ''; |
| OUT | success | 1 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | NULL |

### updateResourceEnabled

This procedure is used to enable or disable a resource such as a datasource or a trigger.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on, N=debug off | CHAR(1) |
| IN | resourcePath - The resource path | VARCHAR(4000) |
| IN | resourceType - The resource type: DATA\_SOURCE, TRIGGER | VARCHAR(255) |
| IN | enable - 1=enable resource, 2=disable resource | BIT |
| OUT | status - 'SUCCESS' or an exception is thrown upon failure | VARCHAR(10) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | fullResourcePath | /shared/examples/ds\_XML |
| IN | resourceType | DATA\_SOURCE |
| IN | enable | 1 |
| OUT | status | SUCCESS |

### updateResourceOwner

This script is used to change resource ownership of a resource. This script can be used to change ownership of a single resource or a folder with multiple resource. By default, the ownership is pushed recursively to all the resources in a folder.

The library function used in this script requires ACCESS TOOLS, READ ALL RESOURCES, READ ALL USERS role, therefore ideally, this script should be executed by an Admin user. The script has no output parameters and may throw an exception.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | resourcePath | LONGVARCHAR |
| IN | resourceType | VARCHAR(255) |
| IN | newOwnerName | VARCHAR(255) |
| IN | newOwnerDomain | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘Y’ |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source’ |
| IN | resourceType | ‘CONTAINER’ |
| IN | newOwnerName | ‘user1’ |
| IN | newOwnerDomain | ‘composite’ |

### updateResourcePrivileges (deprecated)

Use updateResourcePrivilegesV2.

### updateResourcePrivilegesV2

This procedure is used to update the privileges of one or more resources. It is typically used after a deployment to completely reset the privileges of all the resources that were deployed.

It can be used to set privileges exactly or to modify existing privileges. It can also be used to recursively set privileges on child resources of containers or data sources, dependencies (resources that are used by the target resource), and/or dependents (resources that use the target resource.)

A public XML structure has been defined for the resource privileges list input:

SET privilegeEntriesXML =

-- Repeat <resource:privilegeEntry> for each unique resource path/type

' <resource:privilegeEntry>'||CHR(10)||

' <resource:path>' || resourcePath || '</resource:path>'||CHR(10)||

' <resource:type>' || resourceType || '</resource:type>'||CHR(10)||

' <resource:privileges>'||CHR(10)||

-- Repeat <resource:privilege> for each privilege where the user name/domain/type is different

' <resource:privilege>'||CHR(10)||

' <resource:domain>' || domainName || '</resource:domain>'||CHR(10)||

' <resource:name>' || name || '</resource:name>'||CHR(10)||

' <resource:nameType>' || nameType || '</resource:nameType>'||CHR(10)||

' <resource:privs>' || COALESCE (privs, '') || '</resource:privs>'||CHR(10)||

' </resource:privilege>'||CHR(10)||

' </resource:privileges>'||CHR(10)||

' </resource:privilegeEntry>'||CHR(10)||

'';

For example, the following XML could be used to set the privileges on a view and revoke them on one of the view's columns:

SET privilegeEntriesXML =

-- Repeat <resource:privilegeEntry> for each unique resource path/type

' <resource:privilegeEntry>'||CHR(10)||

' <resource:path>/users/composite/admin/virt\_col\_test</resource:path>'||CHR(10)||

' <resource:type>TABLE</resource:type>'||CHR(10)||

' <resource:privileges>'||CHR(10)||

' <resource:privilege>'||CHR(10)||

' <resource:domain>composite</resource:domain>'||CHR(10)||

' <resource:name>cgoodric</resource:name>'||CHR(10)||

' <resource:nameType>USER</resource:nameType>'||CHR(10)||

' <resource:privs>READ SELECT</resource:privs>'||CHR(10)||

' </resource:privilege>'||CHR(10)||

' </resource:privileges>'||CHR(10)||

' </resource:privilegeEntry>'||CHR(10)||

' <resource:privilegeEntry>'||CHR(10)||

' <resource:path>/users/composite/admin/virt\_col\_test/customerid</resource:path>'||CHR(10)||

' <resource:type>COLUMN</resource:type>'||CHR(10)||

' <resource:privileges>'||CHR(10)||

' <resource:privilege>'||CHR(10)||

' <resource:domain>composite</resource:domain>'||CHR(10)||

' <resource:name>cgoodric</resource:name>'||CHR(10)||

' <resource:nameType>USER</resource:nameType>'||CHR(10)||

' <resource:privs>NONE</resource:privs>'||CHR(10)||

' </resource:privilege>'||CHR(10)||

' </resource:privileges>'||CHR(10)||

' </resource:privilegeEntry>'||CHR(10)||

'';

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | setExactly - 0=OVERWRITE\_APPEND, 1=SET\_EXACTLY | BIT |
| IN | recurseChildren - 1=recurse children or 0=do not recurse | BIT |
| IN | recurseDependencies - 1=recurse dependencies or 0=do not recurse | BIT |
| IN | recurseDependents - 1=recurse dependents or 0=do not recurse | BIT |
| IN | inPrivilegeEntriesXML - XML formatted privileges | LONGVARCHAR |
| OUT | success - true(1) or false(0) | BIT |
| OUT | faultResponse - null if succesful otherwise an exception is thrown if not null. | LONGVARCHAR |
| OUT | invalidPathList | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | setExactly | 1 |
| IN | recurseChildren | 1 |
| IN | recurseDependencies | 0 |
| IN | recurseDependencies | 0 |
| IN | inPrivilegeEntriesXML | See example in the description above. |
| OUT | success | 1 |
| OUT | faultResponse | XML not shown here |
| OUT | invalidPathList | NULL |

### updateResourcesSqlTable (deprecated)

Use updateResourceSqlTableV2.

### updateResourcesSqlTableV2

This procedure is used to update the content of a SQL Table View including the indexes for that VIEW. In order to achieve this, an XML array of the columns and their types along with an XML array of the SQL indexes and XML array of foreign keys are passed into this procedure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath - Full resource path which includes the path and the resource name | VARCHAR(4096) |
| IN | scripttext - SQL Table text to be updated | LONGVARCHAR |
| IN | inColumnList - XML formatted column list | LONGVARCHAR |
| IN | inSqlIndexList - XML formatted sql index list | LONGVARCHAR |
| IN | inForeignKeyList - XML formatted sql foreign key list | LONGVARCHAR |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/target/ PRODUCT\_VIEW2’ |
| IN | scripttext | ‘SELECT \* FROM /shared/examples/ds\_inventory/products products’ |
| IN | inColumnList | SET columnList =  ' <resource:columns>'||CHR(10)||  ' <resource:column>'||CHR(10)||  ' <resource:name>'||CAST(XMLTEXT(TRIM(NVL(columnName,' '))) AS LONGVARCHAR)||'</resource:name>'||CHR(10)||  ' <resource:dataType>'||CHR(10)||  ' <common:sqlType xmlns:common="http://www.compositesw.com/services/system/util/common">'||CHR(10)||  ' <common:definition>'||CAST(XMLTEXT(TRIM(NVL(columnType,' '))) AS LONGVARCHAR)||'</common:definition>'||CHR(10)||  ' </common:sqlType>'||CHR(10)||  ' </resource:dataType>'||CHR(10)||  ' <resource:annotation>'||CAST(XMLTEXT(TRIM(NVL(annotation,' '))) AS LONGVARCHAR)||'</resource:annotation>'||CHR(10)||  ' </resource:column>'||CHR(10)||  ' </resource:columns>'||CHR(10)||  ''; |
| IN | inSqlIndexList | SET sqlIndexList =  ' <resource:sqlIndexes>'||CHR(10)||  ' <resource:index>'||CHR(10)||  ' <resource:name>'||CAST(XMLTEXT(sqlIndexName) AS LONGVARCHAR)||'</resource:name>'||CHR(10)||  ' <resource:type>'||CAST(XMLTEXT(sqlIndexType) AS LONGVARCHAR)||'</resource:type>'||CHR(10)||  ' <resource:unique>'||CAST(XMLTEXT(sqlIndexUnique) AS LONGVARCHAR)||'</resource:unique>'||CHR(10)||  ' <resource:columns>'||CHR(10)||  ' <column>'||CHR(10)||  ' <resource:name>'||CAST(XMLTEXT(sqlIndexColName) AS LONGVARCHAR)||'</resource:name>'||CHR(10)||  ' <resource:order>'||CAST(XMLTEXT(sqlIndexColOrder) AS LONGVARCHAR)||'</resource:order>'||CHR(10)||  ' </column>'||CHR(10)||  -- If it is a multi-column key then add more <column> nodes here:  ' <column>'||CHR(10)||  ' <resource:name>'||CAST(XMLTEXT(sqlIndexColName) AS LONGVARCHAR)||'</resource:name>'||CHR(10)||  ' <resource:order>'||CAST(XMLTEXT(sqlIndexColOrder) AS LONGVARCHAR)||'</resource:order>'||CHR(10)||  ' </column>'||CHR(10)||  ' </resource:columns>'||CHR(10)||  ' </resource:index>'||CHR(10)||  ' </resource:sqlIndexes>'||CHR(10)||  ''; |
| IN | inForeignKeyList | SET foreignKeyList =  ' <resource:sqlForeignKeys>'||CHR(10)||  ' <resource:foreignKey>'||CHR(10)||  ' <resource:name>'||CAST(XMLTEXT(fkName) AS LONGVARCHAR)||'</resource:name>'||CHR(10)||  ' <resource:primaryKeyName>'||CAST(XMLTEXT(fkPrimaryKeyName) AS LONGVARCHAR)||'</resource:primaryKeyName>'||CHR(10)||  ' <resource:primaryKeyTable>'||CAST(XMLTEXT(fkPrimaryKeyTable) AS LONGVARCHAR)||'</resource:primaryKeyTable>'||CHR(10)||  ' <resource:columns>'||CHR(10)||  ' <column>'||CHR(10)||  ' <resource:foreignKeyColumnName>'||CAST(XMLTEXT(fkForeignKeyColumnName) AS LONGVARCHAR)||'</resource:foreignKeyColumnName>'||CHR(10)||  ' <resource:primaryKeyColumnName>'||CAST(XMLTEXT(fkPrimaryKeyColumnName) AS LONGVARCHAR)||'</resource:primaryKeyColumnName>'||CHR(10)||  ' </column>'||CHR(10)||  -- If multiple foreign key columns then add more <column> nodes here:  ' <column>'||CHR(10)||  ' <resource:foreignKeyColumnName>'||CAST(XMLTEXT(fkForeignKeyColumnName) AS LONGVARCHAR)||'</resource:foreignKeyColumnName>'||CHR(10)||  ' <resource:primaryKeyColumnName>'||CAST(XMLTEXT(fkPrimaryKeyColumnName) AS LONGVARCHAR)||'</resource:primaryKeyColumnName>'||CHR(10)||  ' </column>'||CHR(10)||  ' </resource:columns>'||CHR(10)||  ' </resource:foreignKey>'||CHR(10)||  ' </resource:sqlForeignKeys>'||CHR(10)||  ''; |
| OUT | success | 1 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateSqlScript

This procedure is used to update the content of a SQL Procedure script.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | scripttext | LONGVARCHAR |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/proc2’ |
| IN | scripttext | ‘PROCEDURE proc2()  BEGIN  DECLARE var varchar;  END’ |
| OUT | success | 1 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateSqlTable

Updates the definition of a SQL Table resource.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | inScripttext | LONGVARCHAR |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/target/PRODUCT\_VIEW’ |
| IN | inScripttext | ‘SELECT \* FROM /shared/examples/ds\_inventory/products products’ |
| OUT | success | 1 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateSqlTableTextAndModel

Updates the SQL text and proprietary model of a SQL Table resource. This is generally used to copy the SQL and model of an existing view to another view without impacting any of the other attributes (privileges, caching, etc.) of the target view. The binary code that describes a proprietary model cannot be generated without Studio, so the only source for model binaries is repository/getBasicResourceCursor\_SQL\_TABLE()

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | inScripttext | LONGVARCHAR |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/target/PRODUCT\_VIEW’ |
| IN | inScripttext | ‘SELECT \* FROM /shared/examples/ds\_inventory/products products’ |
| OUT | success | 1 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateStreamTransformationProcedure

This procedure is used to update a streaming transformation procedure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | transformSourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | transformSourceType | VARCHAR |
| IN | streamModel | RepositoryDefinitions.XsltModelVectorType |
| IN | annotation | LONGVARCHAR |
| IN | isExplicitDesign | BIT |
| IN | parameterVector | RepositoryDefinitions.AttributeCompleteVectorType |
| IN | attributeVector | RepositoryDefinitions.AttributeCompleteVectorType |
| OUT | Success | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples/streamingXSLT' |
| IN | transformSourcePath | '/shared/examples/ds\_XML/productCatalog.xml' |
| IN | transformSourceType | 'TREE' |
| IN | streamModel | NULL |
| IN | annotation | 'Product catalog transformation' |
| IN | isExplicitDesign | 0 |
| IN | parameterVector | NULL |
| IN | attributeVector | NULL |
| OUT | success | 1 |

### updateTableColumnStatisticsConfiguration

This procedure updates the statistics configuration and manual override information for the table specified in resourcePath, as it's columns.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath |
| IN | cardinalityMin | INTEGER |
| IN | cardinalityMax | INTEGER |
| IN | cardinalityExpected | INTEGER |
| IN | gatherEnabled | VARCHAR(20) |
| IN | maxTime | INTEGER |
| IN | columnSettings | VECTOR(  ROW(  name VARCHAR(1000),  flags VARCHAR(1000),  columnMin DOUBLE,  columnMax DOUBLE,  columnDistinct DOUBLE  ) ) |
| OUT | responseXML | XML |
| OUT | faultXML | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples/ds\_orders/orders' |
| IN | cardinalityMin | 36 |
| IN | cardinalityMax | 36 |
| IN | cardinalityExpected | 36 |
| IN | gatherEnabled | 'DEFAULT' |
| IN | maxTime | -1 |
| IN | columnSettings | VECTOR[…] |
| OUT | responseXML | <XML> |
| OUT | faultXML | NULL­ |

### updateTrigger

Update the definition of a Trigger resource that invokes a SQL Procedure containing a parameter.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | procedurePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | procedureParam | VARCHAR(255) |
| OUT | success | BIT |
| OUT | createResponse | XML |
| OUT | faultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/trigger1’ |
| IN | procedurePath | ‘/shared/ASAssets/Utilities/repository/examples/source/proc1’ |
| IN | procedureParam | 'string' |
| OUT | success | 1 |
| OUT | createResponse | XML not shown here |
| OUT | faultResponse | XML not shown here |

### updateXsltTransformationProcedure

This procedure is used to update a custom XSLT transformation procedure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | transformSourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | transformSourceType | VARCHAR |
| IN | xsltText | VARCHAR |
| IN | xsltModel | RepositoryDefinitions.XsltModelVectorType |
| IN | annotation | LONGVARCHAR |
| IN | isExplicitDesign | BIT |
| IN | parameterVector | RepositoryDefinitions.AttributeCompleteVectorType |
| IN | attributeVector | RepositoryDefinitions.AttributeCompleteVectorType |
| OUT | Success | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | '/shared/examples/customXSLT' |
| IN | transformSourcePath | '/shared/examples/ds\_XML/productCatalog.xml' |
| IN | transformSourceType | 'TREE' |
| IN | xsltText | NULL or XSLT text |
| IN | xsltModel | NULL |
| IN | annotation | 'Product catalog transformation' |
| IN | isExplicitDesign | 0 |
| IN | parameterVector | NULL |
| IN | attributeVector | NULL |
| OUT | success | 1 |

## RepoUtils

This section describes the custom java procedure ‘RepoUtils’ which contains several repository lookup and manipulation utilities.

### RepoUtils/applyReservedListToPath

This CJP is used to fix the leading characters and CIS reserved words used in a folder path. Any path that contains a reserved word, leading underscore '\_' or a number '0123456789', or a special character must have double quotes inserted around that portion of the folder. This procedure would be called in conjunction with other procedures. For example, when generating a view based off of another view, the SELECT statement's FROM clause would require that the path to the underlying view be fixed with double quotes if it meets any of the quoting criteria above. This CJP uses $CIS\_HOME/conf/customjars/RepoUtils.properties to store the quoting rules and reserved words as regular expressions. The properties file can be updated at any time and this CJP will pick up the change without requiring a CIS restart.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath – path to a resource | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | debug – Y/N or T/F | CHAR(1) |
| OUT | fixedResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |

1. **Examples:**
   1. **Assumptions: Dependency on** configureReservedList

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | resourcePath | ‘/shared/tmp/1folder/\_folder/XML/local’ |
| IN | debug | ‘N’ |
| OUT | fixedResourcePath | ‘/shared/tmp/"1folder"/"\_folder"/"XML"/"local"’ |

### RepoUtils/applyReservedListToWord

This CJP is used to fix the leading characters and CIS reserved words used in a word. It is assumed that a word is really a portion of a path representing the value in between two slashes '/'. Any word that contains a reserved word, leading underscore '\_' or a number '0123456789', or a special character must have double quotes inserted around it. This procedure would be called in conjunction with other procedures. For example, when generating a view based off of another view, the SELECT statement's FROM clause would require that the path to the underlying view be fixed with double quotes if it meets any of the quoting criteria above. This CJP uses $CIS\_HOME/conf/customjars/RepoUtils.properties to store the quoting rules and reserved words as regular expressions. The properties file can be updated at any time and this CJP will pick up the change without requiring a CIS restart.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | resourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN | debug | CHAR(1) |
| OUT | fixedResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |

1. **Examples:**
   1. **Assumptions: Dependency on** configureReservedList

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | word | ‘XML’ |
| IN | debug | ‘N’ |
| OUT | result | ‘”XML”’ |

### RepoUtils/EncryptPassword (Custom Function)

This CJP encrypts an input string using Composite’s password encryption API. This uses the TEAV encryption algorithm.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inString | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: Dependency on** configureReservedList

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inString | ‘P4ssw0rd’ |
| OUT | result | ‘5AC41E7EC9AC80CE’ |

### RepoUtils/ForceWriteRepoUtils

Forces a new RepoUtils.properties file to be written out to the CIS customjars folder.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | BOOLEAN |

1. **Examples:**
   1. **Assumptions:**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | true |

### RepoUtils/GetAnsi2NativeMapping

Given a Composite (ANSI) data type and a path to a data source, this procedure returns the data source's data type equivalent.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | datasourcePath | VARCHAR(2147483647) |
| IN | cisType | VARCHAR(2147483647) |
| OUT | result | CURSOR(  cisType VARCHAR(2147483647),  cisNormalizedType VARCHAR(2147483647),  cisBaseType VARCHAR(2147483647),  cisScale INTEGER,  cisPrecision INTEGER,  dataTypeId INTEGER,  dataTypeName VARCHAR(2147483647),  nativeType VARCHAR(2147483647),  nativeBaseType VARCHAR(2147483647),  nativeScale INTEGER,  nativePrecision INTEGER ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | datasourcePath | '/shared/examples/ds\_orders' |
| IN | cisType | 'LONGVARCHAR' |
| OUT | result | (  'LONGVARCHAR',  'LONGVARCHAR',  'LONGVARCHAR',  -1,  -1,  -983,  'LONGVARCHAR',  'varchar(2147483647)',  'varchar',  '2147483647',  '-1' ) |

### RepoUtils/getReservedWordList

This CJP retrieves the reserved word list stored in $CIS\_HOME/conf/customjars/RepoUtils.properties The properties file can be updated at any time and this CJP will pick up the change without requiring a CIS restart.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | CURSOR (  ReservedWord LONGVARCHAR ) |

1. **Examples:**
   1. **Assumptions: Dependency on** configureReservedList

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | {('abs'), ('absolute'),('acos'), ('action'), …} |

### RepoUtils/GetSystemProperties

Returns all the properties defined in the JVM running the current instance of CIS.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | CURSOR(  Property VARCHAR(2147483647),  Value VARCHAR(2147483647) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | (‘path.separator’, ‘;’),  (‘user.home’, ‘C:\Users\bob’),  … |

### RepoUtils/GetUserGroups

Returns all the groups that a user belongs to. This functionality isn’t available in the administrative web services API, so it can only be done from within Java itself.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | DomainName | VARCHAR(2147483647) |
| IN | UserName | VARCHAR(2147483647) |
| OUT | result | CURSOR(  DomainName VARCHAR(2147483647),  GroupName VARCHAR(2147483647) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | DomainName | ‘composite’ |
| IN | UserName | ‘admin’ |
| OUT | result | (‘composite’, ‘all’),  (‘composite’, ‘admin’) |

### RepoUtils/isReservedWord (Custom Function)

This CJP is used to detect whether a string is a reserved word (or would otherwise need quoting when referenced as part of a resource path.) It is assumed that a word is really a portion of a path representing the value in between two slashes '/'. Any word that contains a reserved word, leading underscore '\_' or a number '0123456789', or a special character must have double quotes inserted around it. This procedure would be called in conjunction with other procedures. This CJP uses $CIS\_HOME/conf/customjars/RepoUtils.properties to store the quoting rules and reserved words as regular expressions. The properties file can be updated at any time and this CJP will pick up the change without requiring a CIS restart.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inWord | System.Text (VARCHAR(2147483647)) |
| OUT | fixedResourcePath | BOOLEAN |

1. **Examples:**
   1. **Assumptions: Dependency on** configureReservedList

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | word | ‘XML’ |
| OUT | result | TRUE |

### RepoUtils/UpdateDsColumnAnnotation

This procedure is used to update annotations for data source table columns since there is no Admin API this particular operation. For other types of table/view columns, please use the updateSqlTable() admin API.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | column\_path | LONGVARCHAR |
| IN | annotation | LONGVARCHAR |
| OUT | result | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Dependency on** configureReservedList

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | column\_path | '/shared/examples/ds\_orders/orders/OrderID' |
| IN | Annotation | 'OrderID column annotation' |
| OUT | Result | 'Column annotation updated.' |

## CIS Repository Definition Procedures

This section describes each ‘CIS Repository’ definition procedures. The repository definitions provide a global set of type definitions across all of the API’s. The definitions are broken down into two files: one for recursive type definitions and one for non-recursive type definitions.

### definitions/RepositoryDefinitions

Provides global type definitions for the procedures in the /shared/ASAssets/Utilities/repository directory.

1. **Parameters: none**
2. **Examples: none**

### definitions/RepositoryDefinitionsRecursive

Provides global type definitions for the recursive procedures in the /shared/ASAssets/Utilities/repository directory.

If an \*\*\*error\*\*\* is encountered in the 'RepositoryDefinitionsRecursive' file and you get a java memory error when trying to save this file, it is because CIS cannot save the file due to the recursive nature of the procedure 'getResourceTreeList'.

Go to 'getResourceTreeList' and read the comments in the procedure body regarding the temporary commenting of the recursive section until you can properly save the 'RepositoryDefinitionsRecursive' file. Once you save this file, you can then uncomment and save the 'getResourceTreeList' procedure.

1. **Parameters: none**
2. **Examples: none**

## CIS Repository Execute Procedures

This section describes procedures that are used for executing code inside the CIS engine.

### execute/executeProcedure

This procedure calls a procedure in either a blocking or non-blocking manner. It does not return a result set, but is used merely for functional or performance testing. Procedure inputs should be specified in the resourceParams field. Each parameter should be specified using the format <type>=<value> and separated by the '|' character. For example, procedure that takes a single INTEGER input might have a resourceParams value of 'INTEGER=1'. However, a procedure that takes a VARCHAR and an INTEGER as input parameters might have a resourceParams value of 'VARCHAR='abc'|INTEGER=1'.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | blocking | BIT |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceParams | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | blocking | 1 |
| IN | resourcePath | '/shared/examples/LookupProduct' |
| IN | resourceParams | 'INTEGER=1' |

### execute/executeProcedureResults

This procedure calls a procedure in a blocking manner and returns a result set. Procedure inputs should be specified in the resourceParams field. Each parameter should be specified using the format <type>=<value> and separated by the '|' character. For example, procedure that takes a single INTEGER input might have a resourceParams value of 'INTEGER=1'. However, a procedure that takes a VARCHAR and an INTEGER as input parameters might have a resourceParams value of 'VARCHAR='abc'|INTEGER=1'. Output will be returned in either of the two output parameters, depending on whether the procedure being called returns scalar values or a cursor. The unused output value will be NULL.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceParams | LONGVARCHAR |
| OUT | outputScalarResultResponse | XML |
| OUT | outputCursorResultResponse | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | resourcePath | '/shared/examples/LookupProduct' |
| IN | resourceParams | 'INTEGER=1' |
| OUT | outputScalarResultResponse | NULL |
| OUT | outputCursorResultResponse | XML not shown here |

## CIS Repository Server Procedures

This section describes procedures that are used for working with server settings. Calling user will need READ\_ALL\_CONFIG and/or MODIFY\_ALL\_CONFIG rights to make use of these.

### server/addLicense

This procedure adds a license key to the local CIS instance.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | licenseText | VARCHAR(4096) |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | licenseText | XML not shown here |

### server/compareServerAttributes

This procedure compares the server attributes between 2 different DV servers. Server values may be single scalar, a list, a map, or an array.

Requirements:

Users calling this procedure must have the ACCESS\_TOOLS right and, depending on whether the attribute is public or not, may need the READ\_ALL\_CONFIG right as well.

The target server must also publish the following resources which is typically provided out-of-the-box as /services/databases/ASAssets/Utilities/repository

/shared/ASAssets/Utilities/repository/server/getServerAttribute /shared/ASAssets/Utilities/repository/server/getServerAttributesAll /shared/ASAssets/Utilities/repository/getServerAttributeDefs

The server invoking this procedure must provide a Composite Datasource which introspects the above published resources. The path to the schema of that datasource is provided as a parameter "assetsSchemaPath".

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | startingAttribute | VARCHAR(4096) |
| IN | displayComparisonType | VARCHAR |
| IN | sourceEnvName | VARCHAR |
| IN | targetEnvName | VARCHAR |
| IN | assetsSchemaPath | VARCHAR(4096) |
| OUT | result | PIPE (  attrName VARCHAR(32768), - the attribute name  attrType VARCHAR(32768), - the value type. if not LIST, MAP, or ARRAY, then the scalar value's type  displayName VARCHAR(32768), - the attribute's name as displayed in the Studio Configuration panel  status VARCHAR,- The status of the row where the source server is compared with the target server.   * SAME - the row values are the same. * DIFFERENT - the row values are different. * NOT\_EXIST\_SOURCE - the attribute name does not exist on the source server * NOT\_EXIST\_TARGET - the attribute name does not exist on the target server   Source attributes  sourceEnvName VARCHAR,- the environment name  sourceUpdateRule VARCHAR(32768), - indicates whether the field is updatable or not  sourceMinValue VARCHAR(32768), - the minimum allowed value (NULL if there is none)  sourceMaxValue VARCHAR(32768), - the maximum allowed value (NULL if there is none)  sourceUnitName VARCHAR(32768), - the attribute's value's unit name  sourceDefaultValue VARCHAR(32768), - indicates the default value (NULL if there is none.)  sourceAttrValue VARCHAR(32768), - the scalar value (NULL if the value type is LIST, MAP, or ARRAY)  sourceListEntryType VARCHAR(32768), - if the attribute value is a list, this is the list entry's type  sourceListEntryValue VARCHAR(32768), - if the attribute value is a list, this is the list entry's value  sourcemMapKeyTypeVARCHAR(32768), - if the attribute value is a map, this is the map entry's key type  sourceMapKeyValue VARCHAR(32768), - if the attribute value is a map, this is the map entry's value type  sourceMapValueType VARCHAR(32768), - if the attribute value is a map, this is the map entry's value type  sourceMapValueValue VARCHAR(32768), - if the attribute value is a map, this is the map entry's value value  sourceArrayItem VARCHAR(32768), - if the attribute value is an array, this is the array entry's value  sourceVisible BIT, - indicates whether the configuration setting is visible in the Studio Configuration panel  sourceLocallyDefined BIT,- indicates whether this attribute is locally defined or not  sourcePattern VARCHAR(32768), - a regular expression pattern that is used to test for valid values  sourceAnnotation VARCHAR(32768),- a description of the attribute  Target Attributes  targetEnvName VARCHAR,- the environment name  targetUpdateRule VARCHAR(32768), - indicates whether the field is updatable or not  targetMinValue VARCHAR(32768), - the minimum allowed value (NULL if there is none)  targetMaxValue VARCHAR(32768), - the maximum allowed value (NULL if there is none)  targetUnitNameVARCHAR(32768), -- the attribute's value's unit name  targetDefaultValue VARCHAR(32768), - indicates the default value (NULL if there is none.)  targetAttrValue VARCHAR(32768), - the scalar value (NULL if the value type is LIST, MAP, or ARRAY)  targetListEntryType VARCHAR(32768), - if the attribute value is a list, this is the list entry's type  targetListEntryValue VARCHAR(32768), - if the attribute value is a list, this is the list entry's value  targetmMapKeyType VARCHAR(32768), - if the attribute value is a map, this is the map entry's key type  targetMapKeyValue VARCHAR(32768), - if the attribute value is a map, this is the map entry's value type  targetMapValueType VARCHAR(32768), - if the attribute value is a map, this is the map entry's value type  targetMapValueValue VARCHAR(32768), - if the attribute value is a map, this is the map entry's value value  targetArrayItem VARCHAR(32768), - if the attribute value is an array, this is the array entry's value  targetVisible BIT, - indicates whether the configuration setting is visible in the Studio Configuration panel  targetLocallyDefined BIT, - indicates whether this attribute is locally defined or not  targetPattern VARCHAR(32768), - a regular expression pattern that is used to test for valid values  targetAnnotation VARCHAR(32768) - a description of the attribute |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | startingAttribute | /server/config |
| IN | displayComparisonType | A |
| IN | sourceEnvName | DEV |
| IN | targetEnvName | UAT |
| IN | assetsSchemaPath | /shared/Common/ComparisonTools/DV\_Server\_82 /Utilities/repository/getServerAttributesAll |
| OUT | Too large to display here |  |

### server/getServerAttribute (DEPRECATED)

This procedure retrieves the attribute value at the specified attribute path. This procedure only returns values for simple attribute types like STRING, INTEGER, etc. See server/getServerAttributeList and server/getServerAttributeMap for retrieving more complex attribute types.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | attributePath | LONGVARCHAR |
| OUT | keyValue | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | absolutePath | '/server/webservices/timezoneBaseOnGMT' |
| OUT | keyValue | 'true' |

### server/getServerAttributeList (DEPRECATED)

This procedure retrieves the attribute list value at the specified attribute path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | attributePath | LONGVARCHAR |
| OUT | result | CURSOR (  keyName LONGVARCHAR ) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | absolutePath | '/studio/data/examplelist' |
| OUT | result | (no rows) |

### server/getServerAttributeMap (DEPRECATED)

This procedure retrieves the attribute map value at the specified attribute path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | attributePath | LONGVARCHAR |
| OUT | result | CURSOR (  keyName LONGVARCHAR,  keyValue LONGVARCHAR ) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | absolutePath | '/studio/data/examplemap' |
| OUT | result | (no rows) |

### server/getServerAttributeMapByKey (DEPRECATED)

This procedure retrieves the attribute value at the specified attribute map path and map key.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | attributePath | LONGVARCHAR |
| IN | keyName | LONGVARCHAR |
| OUT | keyValue | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | absolutePath | '/studio/data/examplemap' |
| IN | keyName | 'this\_key\_name\_does\_not\_exist' |
| OUT | result | NULL |

### server/getServerAttributesAll

This procedure returns the server attribute values at the input server attribute location.

Example:

DEV, / - return all of the attributes for the DEV environment.

DEV, /server - return all of the server attributes for the DEV environment.

Server values may be single scalar or may be a list, a map, or an array.

Users calling this procedure must have the ACCESS\_TOOLS right and, depending on whether the attribute is public or not, may need the READ\_ALL\_CONFIG right as well.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | environmentName - A unique environment name qualifier that will be returned with the data. | VARCHAR |
| IN | startingAttribute - The starting attribute location. Values: Any path to an attribute. root=/ | VARCHAR(4096) |
| IN | updateRuleFilter - [default] null - return all, otherwise 1 of [READ\_WRITE, READ\_ONLY, WRITE\_ON\_IMPORT] | VARCHAR |
| IN | visible - [default] null - return all, 1 - return attributes that are visible to studio. 0 - return attributes that are not visible to studio. | BIT |
| OUT | result | PIPE (  environmentVARCHAR, - the environment name  categoryNameVARCHAR, - Category: cms, diagnostic, discovery, monitor, server, sources, studio  attrName VARCHAR(32768), - the attribute name  attrType VARCHAR(32768), - the value type. if not LIST, MAP, or ARRAY, then the scalar value's type  displayNameVARCHAR(32768), - the attribute's name as displayed in the Studio Configuration panel  updateRuleVARCHAR(32768), - indicates whether the field is updatable or not  minValue VARCHAR(32768), - the minimum allowed value (NULL if there is none)  maxValue VARCHAR(32768), - the maximum allowed value (NULL if there is none)  unitName VARCHAR(32768), - the attribute's value's unit name  defaultValueVARCHAR(32768), - indicates the default value (NULL if there is none.)  attrValue VARCHAR(32768), - the scalar value (NULL if the value type is LIST, MAP, or ARRAY)  listEntryType VARCHAR(32768), - if the attribute value is a list, this is the list entry's type  listEntryValue VARCHAR(32768), - if the attribute value is a list, this is the list entry's value  mapKeyType VARCHAR(32768), - if the attribute value is a map, this is the map entry's key type  mapKeyValue VARCHAR(32768), - if the attribute value is a map, this is the map entry's value type  mapValueType VARCHAR(32768), - if the attribute value is a map, this is the map entry's value type  mapValueValue VARCHAR(32768), - if the attribute value is a map, this is the map entry's value value  arrayItem VARCHAR(32768), - if the attribute value is an array, this is the array entry's value  visible BIT, - indicates whether the configuration setting is visible in the Studio Configuration panel  locallyDefinedBIT, - indicates whether this attribute is locally defined or not  pattern VARCHAR(32768), - a regular expression pattern that is used to test for valid values  annotation VARCHAR(32768) - a description of the attribute |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | environmentName | DEV |
| IN | startingAttribute | /server/config |
| IN | updateRuleFilter | READ\_WRITE |
| IN | visible | 1 |
| OUT | Too large to display here |  |

### server/getSpecifiedServerAttributes

This procedure returns the specified server attribute values as a PIPE. Optionally, the invoker may have the attributes written out to a CSV file which can be modified and used as an EXCEL datasource for updating server attributes.

For updating server attributes see /shared/ASAssets/Utilities/repository/server/updateServerConfiguration

Only attributes that are visible=1 and updateRule=READ\_WRITE are retrieved. If the attrType=MAP, the attrValue contains the following format [key=value]: mapKeyValue=mapValueValue

The output will require parsing on the '=' sign to extract the key and value pair.

Example: /server/config/security/authentication/WindowsDomainMapping

MYDOMAIN=domain@mycompany.com

The following optional capabilities are available and are shown in their order of precedence:

If none of the optional inputs are provided, this procedure simply returns the default "commonly used" attribute name/paths.

1. Option 1 - Retrieve all changed attributes. The attribute value is different than the default value.
2. Option 2 - Retrieve attributes for the given attribute path or partial path.
3. Option 3 - Override attribute name list. The invoker can specify their own list of attributes to retrieve.
4. Option 4 - Additional attributes with default list. The invoker can specify an additional attribute name list in addition to the default name/paths found in this procedure.

Server values may be single scalar or may be a list, a map, or an array. Users calling this procedure must have the ACCESS\_TOOLS right and, depending on whether the attribute is public or not, may need the READ\_ALL\_CONFIG right as well.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | environmentName - The input environment name that is being configured. Example: DEV, UAT, PROD | VARCHAR |
| IN | csvFilePath - [optional-0] If null then no file is output otherwise provide the full file system path to a CSV file for comma-separate attributes. Regardless of this option, the results are still returned by this procedure as a PIPE output. | VARCHAR(4000) |
| Order of precedence is shown in [optional-#] for the following mutually exclusive input parameters: | | |
| IN | changedAttrValues - [optional-1] Display all attributes that are different from their default values. 1=look for changed attribute values. 0/null=ignore.  Mutually exclusive from attributeName, overrideAttributeNameList, and additionalAttributeNameList.  If set to 1, then ignore attributeName, attributeNameList and defaultAttributeNameList. | BIT |
| IN | attributeName - [optional-2] The attribute name/path or partial name/path to return the server configuration attributes.  Example: / - return all server attributes.  /server - returns all attributes that start with /server.  /studio/data/cursorFetchLimit - return a specific attribute.  Mutually exclusive from changedAttrValues, overrideAttributeNameList, and additionalAttributeNameList. | VARCHAR |
| IN | overrideAttributeNameList - [optional-3] If null then use the default attribute list (defaultAttributeNameList) within this procedure is used.  Otherwise the user provides a comma-separate list of attribute name paths to retrieve.  Mutually exclusive from changedAttrValues, attributeName, and additionalAttributeNameList. |  |
| IN | additionalAttributeNameList - [optional-4] Add these attribute names onto the defaultAttributeNameList.  The user provides a comma-separate list of attribute name paths to retrieve in addition to the default attribute list.  Mutually exclusive from changedAttrValues, attributeName, and overrideAttributeNameList. |  |
| OUT | result | PIPE (  environment VARCHAR, - the environment that is being configured. Example: DEV, UAT, PROD  displayName VARCHAR(32768), - the attribute's name as displayed in the Studio Configuration panel  attrValue VARCHAR(32768), - the scalar value (NULL if the value type is LIST, MAP, or STRING\_ARRAY)  unitName VARCHAR(32768), - the attribute's value's unit name  defaultValue VARCHAR(32768), - indicates the default value (NULL if there is none.)  attrName VARCHAR(32768), - the attribute name path.  attrType VARCHAR(32768), - the value type. if not LIST, MAP, or STRING\_ARRAY, then the scalar value's type  locallyDefined BIT, - indicates whether this attribute is locally defined or not  updateRule VARCHAR, - indicates whether the attribute is READ\_WRITE or READ\_ONLY  annotation VARCHAR(32768) - a description of the attribute |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | environmentName | DEV |
| IN | csvFilePath | /tmp/outputfile.csv |
| Order of precedence is shown in [optional-#] for the following mutually exclusive input parameters: | | |
| IN | changedAttrValues |  |
| IN | attributeName |  |
| IN | overrideAttributeNameList |  |
| IN | additionalAttributeNameList |  |
| OUT | Too large to display here |  |

### server/updateServerAttribute

This procedure updates an attribute value at the specified attribute path. Attributes types can be 'STRING', 'BOOLEAN', 'LIST', or 'MAP'. Examples of attribute list and map values in XML appear below:

Example of LIST attribute value:

<common:item>  
 <common:type>STRING</common:type>  
 <common:value>a1</common:value>  
 </common:item>  
 <common:item>  
 <common:type>STRING</common:type>  
 <common:value>b2</common:value>  
 </common:item>

Example of MAP attribute value:

<common:entry>  
 <common:key>  
 <common:type>STRING</common:type>  
 <common:value>Environment</common:value>  
 </common:key>  
 <common:value>  
 <common:type>STRING</common:type>  
 <common:value>DEV1</common:value>  
 </common:value>  
 </common:entry>

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | ATTR | VARCHAR(1024) |
| IN | ATTR\_TYPE | VARCHAR(1024) |
| IN | NEWVAL | VARCHAR(1024) |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | ATTR | '/server/sql/language/caseSensitive' |
| IN | ATTR\_TYPE | 'BOOLEAN' |
| IN | NEWVAL | 'true' |

### server/updateServerConfiguration

This procedure is used to update the TDV server configuration parameters. The invoker of this procedure must provide a SQL Statement with 3 columns to select configuration attributes from a spreadsheet, table or view.

example: select environment, attrName, attrValue from /shared/Common/ServerConfig/ServerConfiguration/ServerConfiguration.xlsx/ServerConfiguration

The input parameter "environment" acts as a filter and specifies which TDV environment parameters should be read. Any attribute value in the spreadsheet, table or view which contains the word 'null' will be converted to an actual null value.

The invoker must have the following studio rights: ACCESS\_TOOLS, READ\_ALL\_CONFIG, MODIFY\_ALL\_CONFIG

To produce the (csv) spreadsheet with existing values in the server consider executing one of the following procedures:

/shared/ASAssets/Utilities/repository/server/getServerAttributesForSpreadsheet

/shared/ASAssets/Utilities/repository/server/getSpecifiedServerAttributes

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off. | CHAR(1) |
| IN | displayPasswords - Y=display on. N=display off. | CHAR(1) |
| IN | environmentName - The environment name used as a filter against the spreadsheet or table. | VARCHAR(1024) |
| IN | serverConfigurationSql - The SQL statement to the ServerConfiguration spreadsheet, table or view. Procedures are not supported within the FROM clause.  The SQL Statement must have only 3 columns [environment, attrName, attrValue]. Where clause and order by are not needed.  The spreadsheet, table or view may have more columns but the SQL statement must only specify the three shown below:   1. environment - the environment designation for the parameters listed. Allows more than one environment to be stored within source. 2. attrName - the full path to the attribute. 3. attrValue - the configuration value to be updated. If the value of 'null' is found, it is converted to an actual null value.   Example: select environment, attrName, attrValue from /shared/Common/ServerConfig/ServerConfiguration/ ServerConfiguration.xlsx/ServerConfiguration  Note: It is not necessary to place double-quotes around paths. This will be taken care of within this procedure. | VARCHAR(1024) |
| IN | dynamicAttributeList - A delimited list of attribute=value pairs that are used to dynamically modify attributes such as passwords.  Format: attributeName or Path=[variable]=value:SEP:attributeName or Path=[variable]=value  Attribute name must be unique amongst all attributes otherwise use the full attribute path  Typical examples: smtpAuthUserPassword, keystorePasswordOnServerRestart, truststorePasswordOnServerRestart, proxyPassword, userAuthenticationPassword, userPrivacyPassword  The [variable] is the variable that is used in the CoeServerConfiguration.xlsx spreadsheet such as [password]  The value is what is to replace the [variable] upon execution.  Example 1: smtpAuthUserPassword=[password]=my@pas$w0rd  Example 2: /server/config/email/smtpAuthUserPassword=[password]=my@pas$w0rd  Example 3: /server/config/email/smtpAuthUserPassword=[password]=my@pas$w0rd :SEP:keystorePasswordOnServerRestart=[password]=my@pas$w0rd :SEP:truststorePasswordOnServerRestart=[password]=my@pas$w0rd | LONGVARCHAR |
| OUT | numRowsTotal - The total number of rows processed. | INTEGER |
| OUT | numRowsUpdated - The number of rows updated. | INTEGER |
| OUT | numRowsSkipped - The number of rows skipped and not updated. | INTEGER |
| OUT | numRowsError - The number of rows with exceptions. | INTEGER |
| OUT | message - Null if no messages or the list of messages. Format: EXCEPTION=[exception message]CHR(10) | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_CONFIG right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | N |
| IN | environmentName | DEV |
| IN | environmentName | select environment, attrName, attrValue from /shared/Common/ServerConfig/ServerConfiguration /ServerConfiguration.xlsx/ServerConfiguration |
| OUT | numRowsTotal | 1 |
| OUT | numRowsUpdated | 1 |
| OUT | numRowsSkipped | 0 |
| OUT | numRowsError | 0 |
| OUT | message | null |

## CIS Repository User Procedures

This section describes procedures that are used for managing users and groups inside the CIS engine. Calling user will need MODIFY\_ALL\_USERS right to make use of most of these.

### user/createGroup

This procedure creates a group in the specified domain. Group privileges are a space separated list of the global rights a group should have: ACCESS\_TOOLS, UNLOCK\_RESOURCE, READ\_ALL\_RESOURCES, MODIFY\_ALL\_RESOURCES, etc. See /services/webservices/system/admin/user/operations/UserSchema for the full list.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | groupName | VARCHAR(255) |
| IN | groupDomain | VARCHAR(255) |
| IN | groupPrivileges | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | groupName | 'myDevGroup' |
| IN | groupDomain | 'composite' |
| IN | groupPrivileges | 'ACCESS\_TOOLS READ\_ALL\_STATUS' |

### user/createResourcePrivilege

This procedure adds or updates privileges on a resource for the specified user or group. Calling user must have the ability to reassign ownership and privileges on the resource. Valid values:

recurse: '1' or '0'

nameType: 'GROUP' or 'USER'

privileges: space separated list of one or more of READ, WRITE, SELECT, INSERT, UPDATE, DELETE, and GRANT.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | owner | VARCHAR(255) |
| IN | ownerDomain | VARCHAR(255) |
| IN | resourcePath | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096)) |
| IN | resourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40)) |
| IN | recurse | CHAR(1) |
| IN | name | VARCHAR(255) |
| IN | domainName | VARCHAR(255) |
| IN | nameType | VARCHAR(255) |
| IN | privileges | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: Calling user has appropriate privileges and/or rights**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | owner | 'admin' |
| IN | ownerDomain | 'composite' |
| IN | resourcePath | '/shared/examples/CompositeView' |
| IN | resourceType | 'TABLE' |
| IN | recurse | '0' |
| IN | name | 'all' |
| IN | domainName | 'composite' |
| IN | nameType | 'GROUP' |
| IN | privileges | 'READ SELECT' |

### user/createUser

This procedure creates a user in the specified domain.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | username | VARCHAR(255) |
| IN | password | VARCHAR(255) |
| IN | forcePassword | CHAR(1) |
| IN | domainName | VARCHAR(255) |
| IN | groupNameAndDomainList | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | username | 'bob' |
| IN | password | 'password' |
| IN | forcePassword | '0' |
| IN | domainName | 'composite' |
| IN | groupNameAndDomainList | 'bobsgroup|composite dummygroup|composite' |

### user/deleteGroup

This procedure deletes a group in the specified domain.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | groupName | VARCHAR(255) |
| IN | groupDomain | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | groupName | 'myDevGroup' |
| IN | groupDomain | 'composite' |

### user/deleteUser

This procedure deletes a group in the specified domain.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1), either 'Y' or 'N' |
| IN | userName | VARCHAR(255) |
| IN | domainName | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: Calling user has MODIFY\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | userName | 'bob' |
| IN | domainName | 'composite' |

### user/getDomainGroups

This procedure retrieves all the groups in the specified domain.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDomainName | VARCHAR(255) |
| OUT | result | CURSOR (  name VARCHAR(32768),   domainName VARCHAR(32768),   id INTEGER,   explicitRights VARCHAR(32768),   effectiveRights VARCHAR(32768),   inheritedRights VARCHAR(32768),   annotation VARCHAR(32768) ) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inDomainName | ‘composite’ |
| OUT | result | (  ‘admin’,  ‘composite’,  1,  ‘ACCESS\_TOOLS, …’,  ‘ACCESS\_TOOLS, …’,  ‘NONE’,  ‘Administrator group’ ), … |

### user/getDomains

This procedure retrieves a list of all the domains.

* repository/lowerLevelProcedures/getDomainsXSLT - this performs the actual invocation to the CIS repository API and returns a transformed XML list of domains in tabular format.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDomainName | VARCHAR(255) |
| OUT | result | OUT results PIPE (  name VARCHAR,  domainType VARCHAR,  guid VARCHAR,  annotation VARCHAR(32768),  ) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | (  ‘dynamic’,  ‘DYNAMIC’,  NULL,  ‘Dynamic authentication domain’),  (  ‘composite’,  ‘COMPOSITE’,  NULL,  ‘Composite authentication domain’ ), … |

### user/getDomainUsers

This procedure retrieves all the users in the specified domain.

* repository/lowerLevelProcedures/getDomainsXSLT - this performs the actual invocation to the CIS repository API and returns a transformed XML list of domains in tabular format.
* repository/lowerLevelProcedures/getDomainUsersXSLT – this performs the actual invocation to the CIS repository API and returns a transformed XML list of domain users in tabular format.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDomainName | VARCHAR(255) |
| OUT | result | OUT results PIPE (  name VARCHAR,  domainName VARCHAR,  id INTEGER,  explicitRights VARCHAR(1024),  effectiveRights VARCHAR(1024),  inheritedRights VARCHAR(1024),  annotation VARCHAR(1024),  isLocked VARCHAR,  groupName VARCHAR,  groupDomain VARCHAR  ) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inDomainName | ‘composite’ |
| OUT | result | (  ‘admin’,  ‘Administrator user’,  ‘composite’,  -1973,  ‘ACCESS\_TOOLS, …’,  ‘ACCESS\_TOOLS, …’,  ‘NONE’,  ‘N’,  ‘all|composite’ ), … |

### user/getDomainUsersConsolidated

Return a unique list of users and their groups. Each row represents a unique user and contains a list of consolidated groups.

The group list is formatted as follows:

* Each group and domain pair are pipe-separated: group|domain
* The group list may be empty showing no group|domain pairs.
* Each pair is assumed to be in all|composite and therefore this combination is not returned.
* May contain a list of pairs separated by spaces.
* Example: Group1|Domain1 Group2|Domain1 Group3|Domain1

The group list format is specifically formatted to be used with the "GroupList" column on the "User\_List" tab in the Resource\_Privileges\_LOAD\_DB.xlsx spreadsheet. This is used by Best Practices module procedure shown below:

/shared/ASAssets/BestPractices\_v81/PrivilegeScripts/updateUsersDriver

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDomainNameList - [optional] The domain in which to collect the users and groups list. If null/blank the default is all domains. | VARCHAR(255) |
| IN | inIncludeUserList - [optional] A comma-separated list of user names to include as a filter to get the groups for those users. If left null/blank all users are retrieved. | LONGVARCHAR |
| IN | inIgnoreUserList - [optional] A comma separated list of user names to ignore and not return groups for. e.g. admin, anonymous, dre, dre\_manager, monitor, nobody, system | LONGVARCHAR |
| OUT | result | OUT results PIPE (  name VARCHAR,  annotation VARCHAR(1024),  domainName VARCHAR,  groupList LONGVARCHAR,  id INTEGER,  explicitRights VARCHAR(1024),  effectiveRights VARCHAR(1024),  inheritedRights VARCHAR(1024),  isLocked VARCHAR  ) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inDomainName | ‘composite’ |
|  |  |  |
|  |  |  |
|  |  |  |
| OUT | result | (  ‘admin’,  ‘composite’,  -1973,  ‘ACCESS\_TOOLS, …’,  ‘ACCESS\_TOOLS, …’,  ‘NONE’,  ‘Administrator user’,  ‘N’,  ‘all’,  ‘composite’, ), … |

### user/getGroup

This procedure determines the existence of a group and returns its global rights.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | groupName | VARCHAR(255) |
| IN | groupDomain | VARCHAR(255) |
| OUT | groupExists | BIT |
| OUT | groupExplicitRights | VARCHAR(255) |
| OUT | groupEffectiveRights | VARCHAR(255) |
| OUT | groupInheritedRights | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | groupName | 'admin' |
| IN | groupDomain | 'composite' |
| OUT | groupExists | 1 |
| OUT | groupExplicitRights | 'ACCESS\_TOOLS DM MODIFY\_ALL\_CONFIG MODIFY\_ALL\_RESOURCES MODIFY\_ALL\_STATUS MODIFY\_ALL\_USERS READ\_ALL\_CONFIG READ\_ALL\_RESOURCES READ\_ALL\_STATUS READ\_ALL\_USERS UNLOCK\_RESOURCE' |
| OUT | groupEffectiveRights | 'ACCESS\_TOOLS MODIFY\_ALL\_CONFIG MODIFY\_ALL\_RESOURCES MODIFY\_ALL\_STATUS MODIFY\_ALL\_USERS READ\_ALL\_CONFIG READ\_ALL\_RESOURCES READ\_ALL\_STATUS READ\_ALL\_USERS UNLOCK\_RESOURCE' |
| OUT | groupInheritedRights | 'NONE' |

### user/getUser

This procedure determines the existence of a user and returns its global rights.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | userName | VARCHAR(255) |
| IN | userDomain | VARCHAR(255) |
| OUT | userExists | BIT |
| OUT | userExplicitRights | VARCHAR(255) |
| OUT | userEffectiveRights | VARCHAR(255) |
| OUT | userInheritedRights | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | 'N' |
| IN | userName | 'admin' |
| IN | userDomain | 'composite' |
| OUT | userExists | 1 |
| OUT | userExplicitRights | 'ACCESS\_TOOLS DM MODIFY\_ALL\_CONFIG MODIFY\_ALL\_RESOURCES MODIFY\_ALL\_STATUS MODIFY\_ALL\_USERS READ\_ALL\_CONFIG READ\_ALL\_RESOURCES READ\_ALL\_STATUS READ\_ALL\_USERS UNLOCK\_RESOURCE' |
| OUT | userEffectiveRights | 'ACCESS\_TOOLS DM MODIFY\_ALL\_CONFIG MODIFY\_ALL\_RESOURCES MODIFY\_ALL\_STATUS MODIFY\_ALL\_USERS READ\_ALL\_CONFIG READ\_ALL\_RESOURCES READ\_ALL\_STATUS READ\_ALL\_USERS UNLOCK\_RESOURCE' |
| OUT | userInheritedRights | 'ACCESS\_TOOLS DM MODIFY\_ALL\_CONFIG MODIFY\_ALL\_RESOURCES MODIFY\_ALL\_STATUS MODIFY\_ALL\_USERS READ\_ALL\_CONFIG READ\_ALL\_RESOURCES READ\_ALL\_STATUS READ\_ALL\_USERS UNLOCK\_RESOURCE' |

### user/removeUserFromGroups

The purpose of this procedure is to remove an existing user from the groups identified in groupNameAndDomainList.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | username | VARCHAR(255) |
| IN | domainName | VARCHAR(255) |
| IN | groupNameAndDomainList | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | username | 'user1' |
| IN | domainName | 'composite' |
| IN | groupNameAndDomainList | group1|composite group2|composite |

### user/updateUser

The purpose of this procedure is to update the user studio rights.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | userName | VARCHAR(255) |
| IN | userDomain | VARCHAR(255) |
| IN | explicitRights | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: Calling user has READ\_ALL\_USERS right**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | userName | 'admin' |
| IN | userDomain | 'composite' |
| IN | explicitRights | 'ACCESS\_TOOLS DM MODIFY\_ALL\_CONFIG MODIFY\_ALL\_RESOURCES MODIFY\_ALL\_STATUS MODIFY\_ALL\_USERS READ\_ALL\_CONFIG READ\_ALL\_RESOURCES READ\_ALL\_STATUS READ\_ALL\_USERS UNLOCK\_RESOURCE' |

1. How To Use ‘Request’ Procedures

## Introduction

This section describes the routines for examining the SQL of calling requests.

### getAllViewsSqlPlan

Get the SQL Plans for all views specified by the starting folder path and optionally recurse folders and optionally filter by resource name.

The SQL Query Plan and output a formatted text query plan that looks similar to how TDV Studio outputs it. Additionally, determine if the query is pushed down to the database or not. Push down is true if only a SELECT/FETCH is returned otherwise it is false.

This procedure queries the system catalog ALL\_REOURCES to get a baseline set of resources which is then used to invoke the Utilities procedure: /shared/ASAssets/Utilities/request/getSqlPlan

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off | CHAR(1) |
| IN | startingPath - [mandatory] Locate all views in the starting "folder" path execept for datasource tables. May be published or shared paths. | VARCHAR(4000) |
| IN | recursePath - [optional] Execute the gathering of views in the starting path recursively to subfolders. | CHAR(1) |
| IN | resourceName - [optional] Use the resource name as a filter to narrow the scope to a single view. Use % for a wild card to select multiple like pattern views. | VARCHAR |
| OUT | result – cursor output | RESOURCE\_ID INTEGER, - The resource ID.  RESOURCE\_NAME VARCHAR, - The resource name.  RESOURCE\_PATH VARCHAR(4000), - The resource path.  RESOURCE\_TYPE VARCHAR, - The resource type.  RESOURCE\_SUBTYPE VARCHAR, - The resource subtype.  SQL\_QUERY VARCHAR(4000), - The SQL query used to get the SQL plan.  IS\_PUSH\_DOWN BOOLEAN, - true=The query pushes down with only a SELECT/FETCH node. false=The query does not push down. It has more than SELECT/FETCH.  FORMATTED\_SQL\_PLAN LONGVARCHAR - The formatted SQL Query plan like what is seen in Studio  PUSHED\_DOWN\_SQL LONGVARCHAR - If IS\_PUSH\_DOWN=true then this contains the pushed down SQL text otherwise it is null |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | startingPath | /shared/examples |
| IN | recursePath | ‘N’ |
| IN | resourceName |  |
| OUT | formattedSqlPlan | See below: |
| 4 rows are display here   |  |  | | --- | --- | | RESOURCE\_ID: | 18402 | | RESOURCE\_NAME: | ViewSupplier | | RESOURCE\_PATH: | /shared/examples/ViewSupplier | | RESOURCE\_TYPE: | TABLE | | RESOURCE\_SUBTYPE: | SQL\_TABLE | | SQL\_QUERY | SELECT COUNT(1) cnt FROM /shared/examples/ViewSupplier | | IS\_PUSH\_DOWN: | true | | FORMATTED\_SQL\_PLAN: | Too much info to display here |   PUSHED\_DOWN\_SQL:   |  |  | | --- | --- | | RESOURCE\_ID: | 18281 | | RESOURCE\_NAME: | ViewSales | | RESOURCE\_PATH: | /shared/examples/ViewSales | | RESOURCE\_TYPE: | TABLE | | RESOURCE\_SUBTYPE: | SQL\_TABLE | | SQL\_QUERY | SELECT COUNT(1) cnt FROM /shared/examples/ViewSales | | | IS\_PUSH\_DOWN: | false | | FORMATTED\_SQL\_PLAN: | Too much info to display here |   PUSHED\_DOWN\_SQL:   |  |  | | --- | --- | | RESOURCE\_ID: | 18393 | | RESOURCE\_NAME: | ViewOrder | | RESOURCE\_PATH: | /shared/examples/ViewOrder | | RESOURCE\_TYPE: | TABLE | | RESOURCE\_SUBTYPE: | SQL\_TABLE | | SQL\_QUERY | SELECT COUNT(1) cnt FROM /shared/examples/ViewOrder | | | IS\_PUSH\_DOWN: | true | | FORMATTED\_SQL\_PLAN: | Too much info to display here |   PUSHED\_DOWN\_SQL:   |  |  |  | | --- | --- | --- | | RESOURCE\_ID: | 18174 | | | RESOURCE\_NAME: | CompositeView | | | RESOURCE\_PATH: | /shared/examples/CompositeView | | | RESOURCE\_TYPE: | TABLE | | | RESOURCE\_SUBTYPE: | SQL\_TABLE | | | SQL\_QUERY | SELECT COUNT(1) cnt FROM /shared/examples/CompositeView | | IS\_PUSH\_DOWN: | false | | | FORMATTED\_SQL\_PLAN: | Too much info to display here | | | PUSHED\_DOWN\_SQL: |  | | | | |

### getSqlPlan

Get the SQL Query Plan and output a formatted text query plan that looks similar to how TDV Studio outputs it. Additionally, determine if the query is pushed down to the database or not. Push down is true if only a SELECT/FETCH is returned otherwise it is false.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off | CHAR(1) |
| IN | rootNodeTitle - The name to give to the root node. | VARCHAR |
| IN | sqlTextToParse - The full SQL query to parse or alternatively the TDV resource path to parse in /shared, /services, or /users | LONGVARCHAR |
| OUT | isPushedDown - true=The query pushes down with only a SELECT/FETCH node. false=The query does not push down. It has more than SELECT/FETCH. | BOOLEAN |
| OUT | formattedSqlPlan - The formatted SQL Query plan like what is seen in Studio | LONGVARCHAR |
| OUT | pushedDownSQL - If isPushedDown=true then this contains the pushed down SQL text otherwise it is null | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | rootNodeTitle | ViewOrder |
| IN | sqlTextToParse | select \* from /shared/examples/ViewOrder |
| OUT | isPushedDown | false |
| OUT | formattedSqlPlan | See below: |
| -------------------------------------------------------  [ 1] ViewOrder  [ 2] + SELECT  [ 3] + FETCH  -------------------------------------------------------  [1]  Path: ViewOrder  Level: 1  Server Version: TIBCO Data Virtualization 7.0.8 build 17 (2018-09-07 12:15:17) [Patch 00, Hotfix 00]  7.0.8 build 04 (2019-10-16 19.50.48 PDT) [Patch 00, Hotfix 04]  Submitted SQL: select \* from /shared/examples/ViewOrder  Resolved SQL: SELECT ViewOrder.OrderID, ViewOrder.ProductID, ViewOrder.Discount, ViewOrder.OrderDate, ViewOrder.CompanyName, ViewOrder.CustomerContactFirstName, ViewOrder.CustomerContactLastName, ViewOrder.CustomerContactPhone  FROM (SELECT orderdetails.orderid OrderID, orderdetails.productid ProductID, orderdetails.discount Discount, orders.orderdate OrderDate, customers.companyname CompanyName, customers.contactfirstname CustomerContactFirstName, customers.contactlastname CustomerContactLastName, customers.phonenumber CustomerContactPhone  FROM (SELECT orders\_cache.freightcharge freightcharge, orders\_cache.shippostalcode shippostalcode, orders\_cache.purchaseordernumber purchaseordernumber, orders\_cache.shipcountry shipcountry, orders\_cache.orderid orderid, orders\_cache.shipaddress shipaddress, orders\_cache.shipstateorprovince shipstateorprovince, orders\_cache.orderdate orderdate, orders\_cache.shippingmethodid shippingmethodid, orders\_cache.employeeid employeeid, orders\_cache.shipcity shipcity, orders\_cache.customerid customerid, orders\_cache.shipphonenumber shipphonenumber, orders\_cache.shipname shipname, orders\_cache.shipdate shipdate  FROM /shared/examples/ds\_orders/tutorial/orders\_cache  WHERE orders\_cache.cachekey = [CurrentCacheKey]) orders INNER JOIN  /shared/examples/ds\_orders/tutorial/orderdetails orderdetails ON orders.orderid = orderdetails.orderid INNER JOIN  /shared/examples/ds\_orders/tutorial/customers customers ON customers.customerid = orders.customerid) ViewOrder  -------------------------------------------------------  [2]  Path: ViewOrder/SELECT  Level: 2  Name: SELECT  Estimated Rows Returned: Unknown  Projection: orderdetails.orderid OrderID, orderdetails.productid ProductID, orderdetails.discount Discount, orders.orderdate OrderDate, customers.companyname CompanyName, customers.contactfirstname CustomerContactFirstName, customers.contactlastname CustomerContactLastName, customers.phonenumber CustomerContactPhone  -------------------------------------------------------  [3]  Path: ViewOrder/SELECT/FETCH  Level: 3  Name: FETCH  Estimated Rows Returned: Unknown  Data source path: /shared/examples/ds\_orders  Data source type: PostgreSQL  Data source driver name: PostgreSQL 9.1  SQL: SELECT "orderdetails"."orderid" AS "OrderID","orderdetails"."productid" AS "ProductID","orderdetails"."discount" AS "Discount","orders"."orderdate" AS "OrderDate","customers"."companyname" AS "CompanyName","customers"."contactfirstname" AS "CustomerContactFirstName","customers"."contactlastname" AS "CustomerContactLastName","customers"."phonenumber" AS "CustomerContactPhone" FROM ((SELECT "tutorial"."orders\_cache"."orderdate" AS "orderdate","tutorial"."orders\_cache"."customerid" AS "customerid","tutorial"."orders\_cache"."orderid" AS "orderid" FROM "tutorial"."orders\_cache" WHERE "tutorial"."orders\_cache"."cachekey" = NULL) "orders" INNER JOIN "tutorial"."orderdetails" "orderdetails" ON "orders"."orderid" = "orderdetails"."orderid") INNER JOIN "tutorial"."customers" "customers" ON "orders"."customerid" = "customers"."customerid"  ------------------------------------------------------- | | |
| OUT | pushedDownSQL | SELECT "orderdetails"."orderid" AS "OrderID","orderdetails"."productid" AS "ProductID","orderdetails"."discount" AS "Discount","orders"."orderdate" AS "OrderDate","customers"."companyname" AS "CompanyName","customers"."contactfirstname" AS "CustomerContactFirstName","customers"."contactlastname" AS "CustomerContactLastName","customers"."phonenumber" AS "CustomerContactPhone" FROM ((SELECT "tutorial"."orders\_cache"."orderdate" AS "orderdate","tutorial"."orders\_cache"."customerid" AS "customerid","tutorial"."orders\_cache"."orderid" AS "orderid" FROM "tutorial"."orders\_cache" WHERE "tutorial"."orders\_cache"."cachekey" = NULL) "orders" INNER JOIN "tutorial"."orderdetails" "orderdetails" ON "orders"."orderid" = "orderdetails"."orderid") INNER JOIN "tutorial"."customers" "customers" ON "orders"."customerid" = "customers"."customerid" |

### getSqlPlanChild (helper)

Invoked by getSqlPlan(). Not intended to invoke directly.

Parse the <execute:child> node recursively.

1. Extract the child name

2. Extract the child type

3. Extract the child properties

4. Extract the children and recursively invoke this procedure again until all XML levels have been processed.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y=debug on. N=debug off | CHAR(1) |
| IN | childXML - The child node from the XML query plan. | LONGVARCHAR |
| IN | levelnum - The level number starting with 0 for the root node. Each recursive call increments the level by 1 to match the hierarchy. | INTEGER |
| OUT | result – cursor output | PIPE (  levelnum INTEGER,  childname VARCHAR(255),  childtype VARCHAR(255),  childproperties LONGVARCHAR,  childrenxml LONGVARCHAR  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | childXML | <execute:child> <execute:name>ViewOrder</execute:name> <execute:type>ROOT</execute:type> <execute:properties> <execute:property> <execute:name>Server Version</execute:name> <execute:value>TIBCO Data Virtualization ...</execute:value> </execute:property> <execute:property> <execute:name>Submitted SQL</execute:name> <execute:value>select OrderID from /shared/examples/ViewOrder</execute:value> </execute:property> <execute:property> <execute:name>Resolved SQL</execute:name> <execute:value>SELECT ViewOrder.OrderID[LF]FROM (SELECT orderdetails.orderid OrderID, orderdetails.productid ProductID, orderdetails.discount Discount, orders.orderdate OrderDate, customers.companyname CompanyName, customers.contactfirstname CustomerContactFirstName, customers.contactlastname CustomerContactLastName, customers.phonenumber CustomerContactPhone[LF]FROM (SELECT orders\_cache.freightcharge freightcharge, orders\_cache.shippostalcode shippostalcode, orders\_cache.purchaseordernumber purchaseordernumber, orders\_cache.shipcountry shipcountry, orders\_cache.orderid orderid, orders\_cache.shipaddress shipaddress, orders\_cache.shipstateorprovince shipstateorprovince, orders\_cache.orderdate orderdate, orders\_cache.shippingmethodid shippingmethodid, orders\_cache.employeeid employeeid, orders\_cache.shipcity shipcity, orders\_cache.customerid customerid, orders\_cache.shipphonenumber shipphonenumber, orders\_cache.shipname shipname, orders\_cache.shipdate shipdate[LF]FROM /shared/examples/ds\_orders/tutorial/orders\_cache[LF]WHERE orders\_cache.cachekey = [CurrentCacheKey]) orders INNER JOIN[LF] /shared/examples/ds\_orders/tutorial/orderdetails orderdetails ON orders.orderid = orderdetails.orderid INNER JOIN[LF] /shared/examples/ds\_orders/tutorial/customers customers ON customers.customerid = orders.customerid) ViewOrder</execute:value> </execute:property> </execute:properties> <execute:children> <execute:child> <execute:name>SELECT</execute:name> <execute:type>SELECT</execute:type> <execute:properties> <execute:property> <execute:name>Name</execute:name> <execute:value>SELECT</execute:value> </execute:property> <execute:property> <execute:name>Estimated Rows Returned</execute:name> <execute:value>Unknown</execute:value> </execute:property> <execute:property> <execute:name>Projection</execute:name> <execute:value>orderdetails.orderid OrderID</execute:value> </execute:property> </execute:properties> <execute:children> <execute:child> <execute:name>FETCH</execute:name> <execute:type>SCAN\_PHYSICAL</execute:type> <execute:properties> <execute:property> <execute:name>Name</execute:name> <execute:value>FETCH</execute:value> </execute:property> <execute:property> <execute:name>Estimated Rows Returned</execute:name> <execute:value>Unknown</execute:value> </execute:property> <execute:property> <execute:name>Data source path</execute:name> <execute:value>/shared/examples/ds\_orders</execute:value> </execute:property> <execute:property> <execute:name>Data source type</execute:name> <execute:value>PostgreSQL</execute:value> </execute:property> <execute:property> <execute:name>Data source driver name </execute:name> <execute:value>PostgreSQL 9.1</execute:value> </execute:property> <execute:property> <execute:name>SQL</execute:name> <execute:value>SELECT "orderdetails"."orderid" AS "OrderID" FROM ((SELECT "tutorial"."orders\_cache"."customerid" AS "customerid","tutorial"."orders\_cache"."orderid" AS "orderid" FROM "tutorial"."orders\_cache" WHERE "tutorial"."orders\_cache"."cachekey" = NULL) "orders" INNER JOIN "tutorial"."orderdetails" "orderdetails" ON "orders"."orderid" = "orderdetails"."orderid") INNER JOIN "tutorial"."customers" "customers" ON "orders"."customerid" = "customers"."customerid"</execute:value> </execute:property> </execute:properties> </execute:child> </execute:children> </execute:child> </execute:children> </execute:child> |
| IN | levelnum | 1 |
| OUT | result | Too much information to display here. |

### terminateIdleSessions

This procedure terminates any ODBC/JDBC/ADO.NET/HTTP/HTTPS sessions that have been idle for longer than the specified input (in minutes). The caller must have ACCESS\_TOOLS and MODIFY\_ALL\_STATUS rights.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | idleMinutes | INTEGER |
| OUT | result | CURSOR (  session\_id BIGINT,   success BIT,   faultXML XML ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | idleMinutes | 5 |
| OUT | result | (12345, 1, NULL), … |

### terminateRequest

This procedure terminates a request given a request ID. The calling user must have the ACCESS\_TOOLS and MODIFY\_ALL\_STATUS global rights.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | requestID | VARCHAR |
| OUT | success | BIT |
| OUT | responseXML | XML |
| OUT | faultXML | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | requestID | '12345' |
| OUT | success | 1 |
| OUT | responseXML | <XML> |
| OUT | faultXML | [NULL] |

### terminateSession

This procedure terminates a session given a session ID. The calling user must have the ACCESS\_TOOLS and MODIFY\_ALL\_STATUS global rights.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sessionID | VARCHAR |
| OUT | success | BIT |
| OUT | responseXML | XML |
| OUT | faultXML | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sessionID | '12345' |
| OUT | success | 1 |
| OUT | responseXML | <XML> |
| OUT | faultXML | [NULL] |

## RequestUtils

This section will show how to use the ‘Request’ CJP procedures.

### RequestUtils/DirectSqlRequest (Custom Function)

Walks the stack of requests that resulted in the call to this CJP and returns the SQL of this CJP's immediate parent request (or lowest level request that generated an SQL statement.) Note that this CJP does not return code for procedure requests.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | requestSrc | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | requestSrc | ‘SELECT \* FROM /shared/examples/procedure\_that\_directly\_calls\_DirectSqlRequest()’ |

### RequestUtils/OriginalRequest (Custom Function)

Walks the stack of requests that resulted in the call to this CJP and returns the original client request (SQL statement or procedure call.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | requestSrc | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | requestSrc | ‘SELECT \* FROM myCat.mySchema.published\_procedure\_that\_eventually\_calls\_OriginalRequest()’ |

### RequestUtils/ReadInEqClause (Custom Function)

Accepts SQL text (as returned by one of the RequestUtils) and a column name. The SQL text is parsed and scanned for any expressions like ‘column name’ = ‘val’ or ‘column name’ IN (val1, val2, ..valX). All the values found are returned in a cursor.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sql | VARCHAR(2147483647) |
| IN | columnName | VARCHAR(2147483647) |
| OUT | result | CURSOR(  value VARCHAR(2147483647)  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sql | ‘SELECT \* FROM published\_view WHERE mycol IN (1, 2, 3)’ |
| IN | columnName | ‘mycol’ |
| OUT | result | (‘1’),  (‘2’),  (‘3’) |

### RequestUtils/TopSqlRequest (Custom Function)

Walks the stack of requests that resulted in the call to this CJP and returns the SQL of the original client request (or highest level request that generated an SQL statement, e.g. if the original request is a procedure, then TopSqlRequest will return the first SQL request in the chain.) Note that this CJP does not return code for procedure requests.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | requestSrc | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | requestSrc | ‘SELECT \* FROM /shared/examples/procedure\_that\_eventualy\_calls\_TopSqlRequest()’ |

1. How To Use ‘Security Procedures

## Introduction

This section will show how to use the ‘Security’ manipulation procedures for column-based security. A future enhancement will include “rbs” row-based security APIs.

### cbsAddAssignments

This procedure is used add assignments to a CBS policy. Policy Path defaults to /policy/cbs/<policyName>.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off | CHAR(1) |
| IN | policyName – [mandatory] The name of the policy | VARCHAR |
| IN | resourcePath – [mandatory] The TDV resource path | VARCHAR(4000) |
| IN | columnName – [mandatory] The TDV resource column name for the assignment | VARCHAR |
| IN | paramList – [optional] The parameter list | VARCHAR |
| OUT | status – SUCCESS OR FAIL | VARCHAR |
| OUT | message | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | policyName | Hash\_Title3 |
| IN | resourcePath | /shared/test/view1 |
| IN | columnName | Title |
| IN | paramList | null |
| OUT | status | SUCCESS |
| OUT | message |  |

### cbsCreatePolicies

This procedure is used to create a CBS policy. Policy Path defaults to /policy/cbs/<policyName>.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off | CHAR(1) |
| IN | policyName – [mandatory] The name of the policy | VARCHAR |
| IN | policyType – [mandatory] The policy type. [string,integer,decimal,decfloat,double,date,time,datetime] | VARCHAR |
| IN | maskingRule – [mandatory] XML list of masking rules. Example:  <security:maskingRule>  <security:domainName>composite</security:domainName>  <security:userGroupName>group1</security:userGroupName>  <security:isGroup>true</security:isGroup>  <security:isDefaultRule>false</security:isDefaultRule>  <security:selectableString/>  <security:ruleType>PASS\_THROUGH</security:ruleType>  </security:maskingRule>  <security:maskingRule>  <security:domainName>Default</security:domainName>  <security:userGroupName>Default</security:userGroupName>  <security:isGroup>false</security:isGroup>  <security:isDefaultRule>true</security:isDefaultRule>  <security:selectableString>HASHSHA(Title)</security:selectableString>  <security:ruleType>EXPRESSION</security:ruleType>  </security:maskingRule> | LONGVARCHAR |
| IN | isEnabled – [mandatory] 1=enabled, 0=disabled | BIT |
| IN | annotation – [optional] The policy annotation | LONGVARCHAR |
| OUT | status – SUCCESS OR FAIL | VARCHAR |
| OUT | message – if 'SUCCESS' this contains the XML response. if 'FAIL' this contains the error message. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | Y |
| IN | policyName |  |
| IN | policyType |  |
| IN | maskingRule |  |
| IN | isEnabled |  |
| OUT | annotation |  |
| OUT | status |  |
| OUT | message |  |

### cbsCreatePolicyFromXML

This procedure is use to create a column-based security "cbs" policy and its assignment from an input XML. The XML may be passed via the "cbsPolicyXML" parameter or passed from a file in the operation system via the "cbsFilePathXML" parameter.

Note: option 1 and option 2 are mutually exclusive. Option 1 overrides option 2 if both are not null.

Use the procedure "getAllColumnBasedSecurity" to generate the XML needed for input into this procedure.

Policy Path defaults to /policy/cbs/<policyName>.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off | CHAR(1) |
| IN | override – Y=override / delete the existing policy and re-create it. N=throw an exception if the current policy exists. | CHAR(1) |
|  | -- Note: option 1 and option 2 are mutually exclusive. Option 1 overrides option 2 if both are not null |  |
| IN | cbsPolicyXML – option 1: the full XML for a policy in the format of <ns1:cbsGetPoliciesRespons>... </ns1:cbsGetPoliciesRespons> | LONGVARCHAR |
| IN | cbsFilePathXML – option 2: the full path to an XML file containing a single cbsGetPoliciesResponse. | VARCHAR(4000) |
| OUT | status – SUCCESS OR FAIL | VARCHAR |
| OUT | message – message providing more details on status. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug |  |
| IN | override |  |
| IN | -- Note: option 1 and option 2 are mutually exclusive. Option 1 overrides option 2 if both are not null |  |
| IN | cbsPolicyXML |  |
| IN | cbsFilePathXML |  |
| OUT | status |  |
| OUT | message |  |

### cbsDeletePolicies

This procedure is used to delete a CBS policy. Upon successful deletion, the policy assignments to other resources will automatically be deleted. Policy Path defaults to /policy/cbs/<policyName>.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off | CHAR(1) |
| IN | policyName – [optional] \*\*CAUTION\*\* The name of a policy to be deleted. If it's empty, All policies will be deleted. | VARCHAR |
| IN | deleteAllPolicies – [optional] Y=deleta all policies. N/null=do not delete all policies. | CHAR(1) |
| OUT | status – SUCCESS OR FAIL | VARCHAR |
| OUT | message | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug |  |
| IN | policyName |  |
| IN | deleteAllPolicies |  |
| OUT | status |  |
| OUT | message |  |

### cbsGetAssignments

This procedure is used to get an XML response of the policy assignments. If all input parameters are left null/blank, then all policy assignments are returned. Policy Path defaults to /policy/cbs/<policyName>.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off | CHAR(1) |
| IN | policyName – [optional] If a specific policy name is provided then return all policy assignments for that policy. | VARCHAR |
| IN | resourcePath – [optional] If a specific resource path is provided then return those assignments only. | VARCHAR(4000) |
| IN | resourceType – [optional] If a specific resource type such as TABLE is provided then return all assignemnts of that type. | VARCHAR |
| IN | columnName – [optional] If a specific column name is provided then return all assignments associated with that column name. | VARCHAR |
| OUT | status – SUCCESS OR FAIL | VARCHAR |
| OUT | message | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug |  |
| IN | policyName |  |
| IN | resourcePath |  |
| IN | resourceType |  |
| IN | columnName |  |
| OUT | status |  |
| OUT | message |  |

### cbsGetPolicies

This procedure is used get a list of all CBS policies. Optionally, write the XML for each policy to the specified parent path folder using the policy\_name.xml as the file name. Policy Path defaults to /policy/cbs/<policyName>.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off | CHAR(1) |
| IN | policyNameLike – [optional] If null get all policies. If not null the find policy like this name provided. | VARCHAR |
| IN | parentFolderPath – [optional] The parent folder path in the operating system where to save each XML file based on the ccbs resource name. | VARCHAR(4000) |
| OUT | status – SUCCESS OR FAIL | VARCHAR |
| OUT | message | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug |  |
| IN | policyNameLike |  |
| IN | parentFolderPath |  |
| OUT | status – SUCCESS OR FAIL |  |
| OUT | message |  |

### cbsGetPolicy

This procedure is used to get an XML response for a specific policy based on input. Policy Path defaults to /policy/cbs/<policyName>.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug – Y=debug on. N=debug off | CHAR(1) |
| IN | policyName – [mandatory] The name of the policy | VARCHAR |
| OUT | status – SUCCESS OR FAIL | VARCHAR |
| OUT | message | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug |  |
| IN | policyName |  |
| OUT | status |  |
| OUT | message |  |

1. How To Use ‘String’ Procedures

## Introduction

This section will show how to use the ‘String’ manipulation procedures.

### encdodedValues (SQL Definition Set)

This SQL Definition set contains constants that are used by getCodedString.

### addQuotesInList (Custom Function)

Wraps the values in a CSV string in quotes. The function decomposes the list into its constituent values using the input delimiter as a separator and then reconstructs the CSV using the input delimiter and spacing requirements as specified by the input parameters.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inString | LONGVARCHAR |
| IN | quoteType | CHAR(1) (either ' or ") |
| IN | delimiter | VARCHAR |
| IN | trimList | INTEGER (1 or 0) |
| IN | numSpacesAfterDelim | INTEGER |
| OUT | modifiedString | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inString | 'a, b, c' |
| IN | quoteType | '"' |
| IN | delimiter | ',' |
| IN | trimList | 1 |
| IN | numSpacesAfterDelim | 1 |
| OUT | modifiedString | '"a", "b", "c"' |

### basename (Custom Function)

Returns the resource name from an absolute resource path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputString | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| OUT | outputString | /lib/resource/ResourceDefs.ResourceName (VARCHAR(255) as of CIS 5.1) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString | ‘/shared/examples/CompositeView’ |
| OUT | outputString | ‘CompositeView’ |

### concatNotNull (Custom Function)

Concatenate the two strings together according to the mode parameter:

| **mode** | **Action** |
| --- | --- |
| 0 / NULL | Return null string when either inputString1 (part1) or inputString2 (part2) or both is null. |
| 1 | Replace null with blank ('') in inputString1 (part1) or inputString2 (part2) so that a string is returned no matter what. |
| 2 | Return empty string when inputString2 is null. This is a prefix example. |
| 3 | Return empty string when inputString1 is null. This is a suffix example. |

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputString1 | LONGVARCHAR |
| IN | inputString2 | LONGVARCHAR |
| IN | mode | INTEGER |
| OUT | outputString | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString1 | 'a' |
| IN | inputString2 | NULL |
| IN | mode | 1 |
| OUT | outputString | 'a' |

### dirname (Custom Function)

Returns the resource’s parent folder from an absolute resource path.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputString | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| OUT | outputString | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString | ‘/shared/examples/CompositeView’ |
| OUT | outputString | ‘/shared/examples’ |

### emptyStr (Custom Function)

Return a blank if input is null or blank – used extensively when writing to the log – null strings result in null output which is not useful. If the string is not blank, then return the original string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inputString | LONGVARCHAR |
| OUT | outputString | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString | null |
| OUT | outputString | ‘’ |

### entityConstants

This procedure is used by "entityExtract" to perform keyword/entity extraction on text. In this procedure are 4 vectors defined as constants that feed into string/entityExtract(). Review these vectors and make modifications as required to adjust what gets stripped out of the text and what simply gets ignored and not returned as a keyword entity.

### entityExtract

Take an incoming string and return a vector of keyword entities except common words like “a, an, the, of” etc.

1. **Parameters:**
   1. **Assumptions:** Depends on 4 vectors defined as constants define in the procedure string/entityConstants(). Review that procedure and make modifications as required.
      1. **symbols1Vector –** Single value symbols and punctuation such as '!,@,#,$,%' and etc. that get replaced with a non-null, empty character ''.
      2. **symbols2Vector –** Multi-character value symbols such as ' - ' that get replaced with a single space, ' '.
      3. **symbols3Vector –** Hidden character symbols such as tabs that get replaced with a single space, ' '.
      4. **nonEntityVector –** non-Entity based words that are not extracted from the incoming text. Example include: ‘a, and, the, of, this, that’ and many more.

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inText  any text. | LONGVARCHAR |
| IN | inDelimiter  space, pipe or some single character delimiter | CHAR(1) |
| IN | inNumWords  the number of words to produce. If 0 or null then produce all words. | INTEGER |
| IN | random  true(1)=random-produce a list randomly from the incoming text  false(0)=sequential-produce a list sequentially from the incoming text | BIT |
| IN | removeSymbols  true(1)=remove symbols and punctuation prior to extraction  false(0)=do not remove symbols and punctuation prior to extraction | BIT |
| OUT | entityVector | VECTOR(VACHAR) |

1. **Examples:**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inText | ‘A long string of text’ |
| IN | inDelimiter | ‘ ‘ |
| IN | inNumWords | 0 |
| IN | random | 0 |
| IN | removeSymbols | 1 |
| OUT | entityVector | {long,string,text} |

### entityExtractToPipe

Invoke string/entityExtract() and return the keywords in a pipe cursor.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inText  any text. | LONGVARCHAR |
| IN | inDelimiter  space, pipe or some single character delimiter | CHAR(1) |
| IN | inNumWords  the number of words to produce. If 0 or null then produce all words. | INTEGER |
| IN | random  true(1)=random-produce a list randomly from the incoming text  false(0)=sequential-produce a list sequentially from the incoming text | BIT |
| IN | removeSymbols  true(1)=remove symbols and punctuation prior to extraction  false(0)=do not remove symbols and punctuation prior to extraction | BIT |
| OUT | keywordsPipe | Pipe (keyword LONGVARCHAR) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inText | ‘A long string of text’ |
| IN | inDelimiter | ‘ ‘ |
| IN | inNumWords | 0 |
| IN | random | 0 |
| IN | removeSymbols | 1 |
| OUT | keywordsPipe | long  string  text |

### entityExtractToString (Custom Function)

Invoke string/entityExtract() and return the keywords in a single comma separate string variable.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inText  any text | LONGVARCHAR |
| IN | inDelimiter  space, pipe or some single character delimiter | CHAR(1) |
| IN | inNumWords  the number of words to produce. If 0 or null then produce all words. | INTEGER |
| IN | random  true(1)=random-produce a list randomly from the incoming text  false(0)=sequential-produce a list sequentially from the incoming text | BIT |
| IN | removeSymbols  true(1)=remove symbols and punctuation prior to extraction  false(0)=do not remove symbols and punctuation prior to extraction | BIT |
| OUT | outWords  comma separated list of words | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inText | ‘A long string of text’ |
| IN | inDelimiter | ‘ ‘ |
| IN | inNumWords | 0 |
| IN | random | 0 |
| IN | removeSymbols | 1 |
| OUT | outWords | long, string, text |

### escapeCSV (Custom Function)

Looks for separators or qualifiers in a string and escapes them if present.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inStr | LONGVARCHAR |
| IN | separator | VARCHAR(1) |
| IN | qualifier | VARCHAR(1) |
| OUT | outputString | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inStr | ‘john, paul, ringo, ”bob”, and george’ |
| IN | separator | ‘,’ |
| IN | qualifier | ‘”’ |
| OUT | outStr | ‘”john, paul, ringo, ””bob””, and george”’ |

### extractBiDelimitedText (Custom Function)

Extract bi-delimited text refers to the ability to locate text based on a search term where the search term encloses the sought after text in a beginning and ending delimiter thus bi-delimited. Where this can be useful is when you have to throw an exception and you want to embed both the custom error code and error message in the exception: raise ex VALUE 'Custom exception: ERROR\_CODE(C1) ERROR\_MESSAGE(Error is foo.)'. This allows you to have a generic error processing routine that can extract specific error codes and values. Of course this is not limited to that. You can be very creative in the usage of this routine.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inText | LONGVARCHAR |
| IN | searchTerm  a word with no spaces that represents the keyword search term to locate | VARCHAR |
| IN | openingDelim  The beginning delimiter which directly follows the searchTerm. Allowed: Single: '[', '(', '{', '<' Doubles: '[[', '((', '{{', '<<' | VARCHAR |
| IN | closingDelim  The ending delimiter which directly follows the content. Allowed: Single: ']', ')', '}', '>' Doubles: ']]', '))', '}}', '>>' | VARCHAR |
| IN | inOccurrence  The value of the occurrence input value determines which occurrence to return (numbered starting at 1 from left to right. Use negative values to number occurrences from right to left.) If a NULL value or zero is passed in for occurrence, a default of 1 is used. | INTEGER |
| IN | trimText  0=do not trim result, 1=do trim result text (default=0) | INTEGER |
| IN | caseSensitive  0=no search term case sensitivity, 1=case sensitive search term. | INTEGER |
| OUT | result | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inText | ‘john, paul, ringo, ”bob”, and george’ |
| IN | searchTerm | ‘,’ |
| IN | openingDelim | ‘”’ |
| IN | closingDelim |  |
| IN | inOccurrence |  |
| IN | trimText |  |
| IN | caseSensitive |  |
| OUT | result |  |

### extractDelimitedText

This is a generic procedure for extracting values from a list using a single or multi-character delimiter. The results are returned as a cursor of objects. This does not use the expensive REGEX function or VECTORS for parsing the delimited text. It does not parse character by character. It uses a more efficient INSTR and SUBSTRING to extract the results.

Example Usage:

1. Extract the last object in a DV path:

inputString=/a1/b2/c3/d4

delimiter=/

occurrence=1

reverseInputText=1

reverseOutput=1

trimText=1

includeDelimiter=0

result: d4

2. Extract a comma-separated list

inputString=a1,b2,c3,d4

delimiter=,

occurrence=0

reverseInputText=0

reverseOutput=0

trimText=1

includeDelimiter=0

result: a1

b2

c3

d4

3. Extract the 4th object [database] in a DV path:

object: 1 2 3 4 5 6

inputString=/services/databases/db/catalog/schema

delimiter=/

occurrence=4

reverseInputText=0

reverseOutput=0

trimText=1

includeDelimiter=0

result: db

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **inputString** - A text string containing a list with delimeters to be extracted into a cursor of output. | LONGVARCHAR |
| IN | **delimiter** - 1 or more characters such as single characters: space , | / or double characters such as || or any repeatable combination. | VARCHAR |
| IN | **occurrence** - 0=all occurrences, n>0=the occurrence of delimited text or null if the occurrence requested is not applicable. | SMALLINT |
| IN | **reverseInputText** - 0=do not reverse input text, 1=reverse input text before parsing | SMALLINT |
| IN | **reverseOutput** - 0=do not reverse output text, 1=reverse output text before returning. | SMALLINT |
| IN | **trimText** - 0=do not trim, 1=do trim result | SMALLINT |
| IN | **includeDelimiter** - 0=exclude delimiter from output, 1=include delimiter in output | SMALLINT |
| OUT | **result** – cursor of objects. A cursor of extracted text. If no delimiter was found, then the original input text is returned. If the delimiter appears at the beginning of the text before any other characters, an empty row is output and counts as 1 occurrence. For example: /a/b/c/d with delimiter=/ and occurrence=2 would result in 'a' being returned. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inputString | ‘john || paul || ringo || ”bob” || george’ |
| IN | delimiter | ‘,’ |
| IN | occurrence | 0 |
| IN | reverseInputText | 0 |
| IN | reverseOutput | 0 |
| IN | trimText | 1 |
| IN | includeDelimiter | 0 |
| OUT | result | ‘john’  ‘paul’  ‘ringo’  ‘”bob”’  ‘george’ |

### extractTextList

The extractTextList is used to extract a separated list of values containing embedded separators within double quotes, single quotes. The result is returned as a cursor based on the boundaries of the the qualifiers: double quotes, single quotes or paired parenthesis. The separator value is preserved within the qualifier if the flag for that qualifier is set to 1 (true).

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | textList | LONGVARCHAR |
| IN | separator  The separator value (typically a comma) that will be used to define the boundary of text expressions. Becuase this uses regex, any special separators need to be escaped with a \ character. It is not necessary to escape a comma separator in regex. The following are potential use cases and are shown within brackets [] to better show the use of spaces: [\ ] - this is used to split text on a space such as finding all the words in a sentence. [\\] - this is a backslash separator escaped with a backslash [\^] - this is a caret separator escaped with a backslash. | VARCHAR |
| IN | preserveDoubleQuotes  1 or 0/null (default). Indicates whether to preserve the context of commas within the boundaries of a double quoted qualifier string. | BIT |
| IN | preserveSingleQuotes  1 or 0/null (default). Indicates whether to preserve the context of commas within the boundaries of a single quoted qualifier string. | BIT |
| IN | preserveParenthesis  1 or 0/null (default). Indicates whether to preserve the context of commas within the boundaries of left and right parenthesis qualifier pairs. | BIT |
| IN | preserveQualifier  1/null (default) or 0. In this case, the default is to preserve the qualifer value on output. The qualifiers may be double quotes, single quotes or left and right parenthesis. If set to 0 (do not preserve), the qualifiers are only remmoved if they exist as pairs in the first and last characters and the length of the text being returned is at least 2 characters. Otherwise if the above conditions are not met, any attempt to remove embedded qualifiers will not be completed. The assumption is that qualifiers exist at the boundaries of the comma separator such as "orders,customers", orders which would yield:  "orders,customers"  orders  This example would not remove the qualifier: text "more text" text,text  text "more text" text  text | BIT |
| IN | trimResults  1 or 0/null (default) - if set to 1, then trim the results of any white space otherwise do not. | BIT |
| OUT | result | PIPE(textExpression LONGVARCHAR) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | textList | "orders,customers",orders,customers |
| IN | Separator | ‘,’ |
| IN | preserveDoubleQuotes | 1 |
| IN | preserveSingleQuotes | 1 |
| IN | preserveParenthesiss | 1 |
| IN | preserveQualifier | null |
| IN | trimResults | null |
| OUT | result | "orders,customers"  orders  customers |

### findOpenClosePair

Search through the sql script to find the corresponding left open and right close pairs. This script was originally developed to parse through SQL text to find functions. Only a single character value is supported for these delimiters. Valid pairs include: (), [], {}, <>, /\

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sqlScript | LONGVARCHAR |
| IN | begPos | INTEGER |
| IN | openingDelim | VARCHAR |
| IN | closingDelim | VARCHAR |
| IN | trimText | INTEGER |
| OUT | openingDelimPos | INTEGER |
| OUT | closingDelimPos | INTEGER |
| OUT | extractedScript | LONGVARCHAR |
| OUT | extractedScriptBefore | LONGVARCHAR |
| OUT | extractedScriptAfter | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sqlScript | ‘P1(P2( a ))’ |
| IN | begPos | 1 |
| IN | openingDelim | ( |
| IN | closingDelim | ) |
| IN | trimText | 1 |
| OUT | openingDelimPos | 3 |
| OUT | closingDelimPos | 17 |
| OUT | extractedScript | ‘P2( a )’ |
| OUT | extractedScriptBefore | ‘P1’ |
| OUT | extractedScriptAfter | ‘’ |

### findString (Custom Function)

Given two input strings, this function returns an integer value representing the starting position of the first string within the second string. The third parameter indicates which direction to begin searching. When 'F', begin searching forward from the beginning of the second string. When 'R', begin search in reverse from the end of the second string.

* This function is case-sensitive.
* All string types, all numeric types, and all data types are accepted as input arguments.
* The output is always an integer provided none of the input strings is NULL. Otherwise, NULL is returned.
* If any of the arguments is NULL, the function returns NULL.
* If the first argument is a blank string, the function returns 1 (one).
* If the first argument is not found within the second argument, the function returns 0 (zero).
* A blank non-null string (‘’) for the search string always produces 1 for an output.

Examples:

direction='F'

POSITION(‘it’ IN ‘case-sensitivity’),

Output: 10 ^

POSITION(‘’ IN ‘mistake’)

Output: 1

POSITION(‘no’ IN ’yes’)

Output: 0

direction='R'

POSITION(‘it’ IN ‘case-sensitivity’)

Output: 14 ^

POSITION(‘no’ IN ’yes’)

Output: 0

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | searchString | LONGVARCHAR |
| IN | stringToSearch | LONGVARCHAR |
| IN | direction | CHAR(1) |
| OUT | pos | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | searchString | ‘case-sensitivity’ |
| IN | stringToSearch | ‘it’ |
| IN | direction | ‘R’ |
| OUT | pos | 14 |

### findStringInList (Custom Function)

Given two input strings, this function returns an integer value representing the field position of the first string within the second delimited string. The third parameter indicates the delimiter string to use. The value of pos is 1-based (0 indicates the searchString parameter was not found.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | searchString | LONGVARCHAR |
| IN | stringToSearch | LONGVARCHAR |
| IN | delimiter | VARCHAR |
| OUT | pos | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | searchString | ‘csv’ |
| IN | stringToSearch | ‘my,csv,value’ |
| IN | delimiter | ‘,’ |
| OUT | pos | 2 |

### fixQuotes (Custom Function)

Turns single quotes into two single quotes so that when you are constructing a dynamic SQL statement this procedure insures that values with quotes in them are fixed to be double quoted and the SQL statement will execute correctly.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inMessage | LONGVARCHAR |
| OUT | outMessage | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inMessage | this is a 'quoted' string |
| OUT | outMessage | this is a ''quoted'' string |

### getCodedString

Perform encoding or decoding of special characters within double quoted, single quoted or parenthesis pair strings.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inCodingType - The type of coding to perform.  **E=encode special characters**,  e.g. encode from /a/b/"1\_002e3\_0020c" /"\_0020d\_0020"/"1\_002c2\_0021" /"\_0020\_0021\_0020\_0025\_0020 \_0028\_0020\_0029\_0020\_002a\_0020 \_002b\_0020\_002d\_0020\_002f\_0020"  **D=decode special characters**  e.g. decode from /a/b/"1.3 c"/" d "/"1,2!"/" ! % ( ) \* + - / " | CHAR(1) |
| IN | inEncodingActions - A directive on which action to take. A space or comma separated list of directives below:  P=encode/decode between Parenthesis,  D=encode/decode between double quotes,  S=encode/decode between single quotes,  N=encode/decode anywhere in the string,  null when codingType=D (this parameter is ignored) | VARCHAR(255) |
| IN | codingList - A comma separate list of encoded values to encode or decode. e.g. \_0020,\_002c,\_002e or HEXADECIMAL in conjunction with inEncodingActions=S | LONGVARCHAR |
| IN | inString - The string to encode or decode | LONGVARCHAR |
| IN | inModuleNameInvoking - The module from which this script is being invoked. | VARCHAR(255) |
| IN | inDebug - Debug: Y or N | CHAR(1) |
| IN | inModuleQualifer - The module qualifier is a name or type of the "inString" being encoded/decoded to be used for debugging purposes. | VARCHAR |
| OUT | outString - The result string | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inCodingType | E |
| IN | inEncodingActions | D |
| IN | codingList | \_0020,\_002E |
| IN | inString | “a.b c” |
| IN | inModuleNameInvoking | test |
| IN | inDebug | N |
| IN | inModuleQualifer | "a\_002Eb\_0020c" d e f |
| OUT | outString | '/shared' |

### getConstant (Custom Function) [DEPRECATED. Use getConstantV2]

This procedure gets a constant value from a dynamic constant path. All constants must be of type VARCHAR. If the value is actually an integer THEN the application must take care of casting to the proper value.

The constant path should be a procedure that outputs the named constant as a scalar value.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | constantPath | TypeDefinitions.pathType |
| IN | constantName | VARCHAR(255) |
| OUT | outValue | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | constantPath | ‘/shared/ASAssets/Utilities/documentation/constants’ |
| IN | constantName | ‘resourcePath’ |
| OUT | outValue | ‘/shared’ |

### getConstantV2 (Custom Function)

This procedure subsumes the capabilities of getConstant and should be used going forward. This procedure gets a constant value or the actual constant string from a dynamic constant path.

Option 1: Return the value of the constant which was the original intent of this procedure.

Option 2: Return the definition string of the constant.

constantsPath - The path to the constants file ending with procedure parenthesis (). i.e. /shared/PSAsSETs/Utilities/documentation/constants()

constantsName - The name of the constant.

optionReturn - option 1 [default and same as getConstant()] or option 2.

Option 1: Return the value of the constant which was the original intent of this procedure and the default behavior.

Assumption: somvar=/shared/myproject

Example: set optionName = somevar || '/somefolder'

Return: /shared/myproject/somefolder

Option 2: Return the definition string of the constant.

Example: set optionName = somevar || '/somefolder'

Return: somevar || '/somefolder'

All results are returned as LONGVARCHAR. If the value is actually an integer THEN the application must take care of casting to the proper value.

**Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | constantPath | TypeDefinitions.pathType |
| IN | constantName | VARCHAR(255) |
| IN | optionReturn | SMALLINT |
| OUT | outValue | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | constantPath | '/shared/ASAssets/Utilities/documentation/constants' |
| IN | constantName | 'resourcePath' |
| IN | optionReturn | 1 |
| OUT | outValue | '/shared' |

### getDelimitedOccurrence (Custom Function)

Given a delimited string, this procedure will return the value at a specified field number. Searches may start at the beginning (mode value of 'F') or from the end (mode value of 'R') of the string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inText | LONGVARCHAR |
| IN | mode | CHAR(1) |
| IN | delimiter | VARCHAR |
| IN | inOccurrence | INTEGER |
| IN | inTrimText | INTEGER |
| OUT | result | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inText | 'my,csv,value' |
| IN | mode | 'F' |
| IN | delimiter | ‘,’ |
| IN | inOccurrence | 2 |
| IN | inTrimText | 0 |
| OUT | result | 'csv' |

### getDelimitedSum (Custom Function)

Given a delimited string, this procedure will return the sum of the delimited fields starting at a specified field position.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inText | LONGVARCHAR |
| IN | delimiter | VARCHAR |
| IN | inSstartingOccurrence | INTEGER |
| OUT | result | DECIMAL(32,2) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inText | '1, 2, 3, 4, 5, 6, 7, 8, 9' |
| IN | delimiter | ‘,’ |
| IN | inSstartingOccurrence | 2 |
| OUT | result | 44 |

### indent (Custom Function)

This procedure indents text.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **depthNum** – The depth of the incoming resource. Tells this procedure how many times to indent. If null or zero, the original value is returned and not indented. | INTEGER |
| IN | **indent** – The amount of spaces to indent. | VARCHAR |
| IN | **inValue** – The value to indent. | LONGVARCHAR |
| OUT | **outValue** – debug flag | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | depthNum | 2 [the depth is 2 – 2nd level] |
| IN | indent | ‘ ‘ [4 spaces] |
|  | inValue | ‘Text’ [the word Text] |
| OUT | outValue | ‘ Text’ [8 spaces precede the work Text] |

### isEmpty (Custom Function)

For a given input string, return 1 if empty or 0 if not.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | request | LONGVARCHAR |
| OUT | response | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | request | ‘’ |
| OUT | response | 1 |

### joinCursorByDelimiter

This procedure joins the result set of a CURSOR containing a single column of LONGVARCHARs into a single string with values separated by a delimiter string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inCursor - Contains the list of strings to join together. The cursor must be open when it's passed in.  Values: Any CURSOR with a single column of VARCHARs. If NULL or empty, this procedure will return a NULL. | CURSOR (strValue LONGVARCHAR) |
| IN | delimiter - The string value to use to separate string values in the result LONGVARCHAR.  Values: Any text value. If NULL, this procedure will use an empty string. | VARCHAR |
| IN | nullValue - The string to use when a row is NULL (concatenating a NULL will cause the result LONGVARCHAR to be NULL.)  Values: Any text value. If NULL, this procedure will use '[NULL]' to represent NULL values. | VARCHAR |
| OUT | result - Contains the joined string  Values: Any text value | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inCursor | ‘s1,s2,s3,s4’ |
| IN | delimiter | ‘,’ |
| IN | nullValue | ‘’ |
| OUT | result |  |

### joinVectorByDelimiter

This procedure joins the contents of a VECTOR of LONGVARCHARs into a single string with values separated by a delimiter string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inVector - Contains the list of strings to join together.  Values: Any VECTOR of VARCHARs. If NULL or empty, this procedure will return a NULL. | CURSOR (strValue LONGVARCHAR) |
| IN | delimiter - The string value to use to separate string values in the result LONGVARCHAR.  Values: Any text value. If NULL, this procedure will use an empty string. | VARCHAR |
| IN | nullValue - The string to use when a row is NULL (concatenating a NULL will cause the result LONGVARCHAR to be NULL.)  Values: Any text value. If NULL, this procedure will use '[NULL]' to represent NULL values. | VARCHAR |
| OUT | result - Contains the joined string  Values: Any text value | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inVector | [(s1,s2,s3,s4)] |
| IN | delimiter | ‘,’ |
| IN | nullValue | ‘’ |
| OUT | result |  |

### last4ofSSN (Custom Function)

Return ‘x0000’ format for a 9 digit string assumed to be an SSN. Concatenates an ‘x’ in front of the last 4 characters.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inSSN | VARCHAR(255) |
| OUT | outSSN | VARCHAR(255) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inSSN | ‘123456789’ |
| OUT | outSSN | ‘x6789’ |

### modifyConstant (Custom Function)

This script is used to modify a Constants procedure and change values at the time of deployment in order to enable automation. The Constants file is any SQL Script procedure that contains a format:

SET varname = value;

This script will search for "SET varname" in the script text in order to modify the value. The value that is passed in is modified between the = and ; which means that any single quotes must be provided if the original value has them. For example a Constants procedure may have many variables set but we are only interested in setting one of the variables which will be uniquely defined within the context of the procedure. Example:

SET EnvironmentType = 'DEV';

In the above example the objective is to search for the namePair=EnvironmentType and replace the value with valuePair='UAT'. Notice that the value contains all necessary surrounding quotes. The table below shows how to escape values such as single and double quotes if necessary.

The following values that are passed in may be escaped for the values passed in by the variable valuePair:

| **Description** | **Value** | **Escaped Value** |
| --- | --- | --- |
| quote | " | &quot; |
| apostrophe | ' | &apos; |
| ampersand | & | &amp; |
| less than | < | &lt; |
| greater than | > | &gt; |

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | constantsPath | VARCHAR(4096) |
| IN | namePair | VARCHAR(255) |
| IN | valuePair | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | constantsPath | '/shared/ASAssets/Utilities/documentation/constants' |
| IN | namePair | 'resourcePath' |
| IN | valuePair | '''/shared/examples''' |

### normalizeRowsToPipe

De-dupe an incoming cursor of strings. For example, maybe there is a list of keywords and you only want the unique values. Output the keywords as a pipe cursor. It can be used in conjunction with string/entityExtractToPipe().

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | curs | CURSOR(str VARCHAR) |
| OUT | pipeStr | PIPE (str VARCHAR) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | curs | {‘Words’,  ‘Extracted’,  ‘Words’} |
| OUT | pipeStr | {‘Words’,  ‘Extracted’} |

### normalizeRowsToString

De-dupe an incoming cursor of strings. For example, maybe there is a list of keywords and you only want the unique values. Output the values in a single comma-delimited string. It can be used in conjunction with string/entityExtractToPipe() to produce the keywords cursor.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | curs | CURSOR(str VARCHAR) |
| OUT | result  Comma separate list of words | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | curs | {‘Words’  ‘Extracted’,  ‘Words’} |
| OUT | result | ‘Words, Extracted’ |

### numOccurrences

Given two input strings, this function returns an integer value representing the number of occurrences of "searchString" within "stringToSearch".

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | searchString | LONGVARCHAR |
| IN | stringToSearch | LONGVARCHAR |
| OUT | num | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | searchString | ‘This is a string’ |
| IN | stringToSearch | ‘is‘ |
| OUT | num | 1 |

### p\_DelimitedStringToCursor

Converts a delimited VARCHAR into a cursor of VARCHARs. The delimiter is currently limited to a single character. The delimiter character can optionally be included in each token. If any input parameter is NULL, the result is NULL.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inStr | LONGVARCHAR |
| IN | inDelimiter | VARCHAR |
| IN | inIncludeDelimiter | BIT |
| OUT | outTokens | PIPE (  strToken LONGVARCHAR ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inStr | ‘This is a string’ |
| IN | inDelimiter | ‘ ‘ |
| IN | inIncludeDelimiter | 0 |
| OUT | outTokens | {‘this’,’is’,’a’,’string’} |

### p\_FixedStringToCursor

Converts a VARCHAR into a cursor of fixed length VARCHARs. If any input parameter is NULL, the result is NULL.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inStr | VARCHAR |
| IN | inLength | INTEGER |
| IN | inNormalizeFinalToken | BIT |
| OUT | outTokens | PIPE (strToken VARCHAR) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inStr | ‘string1string2string3string4’ |
| IN | inDelimiter | 7 |
| IN | inIncludeDelimiter | 0 |
| OUT | outTokens | {‘string1’,’string2’,’string3’,’string4’} |

### ParseCSVLine

Converts a line of CSV text to a cursor containing the CSV values.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | csvLine | System.Text (VARCHAR(2147483647)) |
| IN | separator | VARCHAR(1) |
| IN | qualifier | VARCHAR(1) |
| OUT | result | PIPE (CSValue VARCHAR(32768)) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | csvLine | ‘string1, ”string2”, “string “”quoted”” 3”, string4’ |
| IN | separator | ‘,’ |
| IN | qualifier | ‘”’ |
| OUT | result | {string1, string2, string “quoted” 3, string4} |

### removeDoubleQuotes

Remove the double quotes from a string. Can be useful in circumstances when building a CIS resource path and some of the text contains double quotes. Doubled quotes will be replaced with a single double quote.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inString | LONGVARCHAR |
| OUT | outString | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inString | ‘This string “contains” examples of ””double”” quoted text.’ |
| OUT | outString | ‘This string contains examples of “double” quoted text.’ |

### removeSingleQuotes

Remove the single quotes from a string. Can be useful in circumstances when building dynamic SQL and some of the text contains single quotes.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inString | LONGVARCHAR |
| OUT | outString | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inString | “This string ‘contains’ examples of ’’double’’ quoted text.” |
| OUT | outString | “This string contains examples of ‘double’ quoted text.” |

### RegexPatterns

This procedure contains useful and commonly used regular expression patterns that can be used for validating string values (using string/TextUtils/RegexFind() or string/TextUtils/RegexPosition()) or (for those patterns with matching groups defined) for parsing string values (using string/TextUtils/RegexGetGroups()) and pulling out parsed components. See string/examples/ParseUSPhoneNumber() for example usage.

1. **Parameters: none**
2. **Examples: none**
3. **Available Patterns:**

| **Pattern** | **Purpose** | **Matching Groups** |
| --- | --- | --- |
| NUMBER\_INTEGER | Validates that a string is a valid integer number | N/A |
| NUMBER\_DECIMAL | Validates that a string is a valid decimal number | N/A |
| NUMBER\_SCIENTIFIC | Validates that a string is a valid number expressed using scientific notation. | N/A |
| SQL\_DATA\_TYPE | Validates that a string is a valid CIS SQL data type | N/A |
| SQL\_DATA\_TYPE\_NAMES | Validates that a string is a valid CIS SQL data type name | N/A |
| SQL\_DATA\_TYPE\_NAMES\_W\_SCALE | Validates that a string is a valid CIS SQL data type name that allows the definition of scale | N/A |
| SQL\_DATA\_TYPE\_NAMES\_W\_SCALE\_AND\_PRECISION | Validates that a string is a valid CIS SQL data type name that allows the definition of scale and precision. | N/A |
| US\_PHONE\_NUMBER | Validates or parses a U.S./Canada phone number | 1 – Country code (NULL if missing)  2 – Area code  3 – Central office code  4 – Subscriber number |
| XML\_DATE\_TIME | Validates or parses an XML dateTime string | 1 – Date component  2 – Time component  3 – Time zone (NULL if missing) |

### splitByDelimiter

Split a string by a defined delimiter and return the results in a vector of varchar(255) strings. See also string/TextUtils/RegexSplit.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inString | LONGVARCHAR |
| IN | inDelimiter  any single delimiter character. Values: Common separators include: ' ' or '|' or ',' | CHAR(1) |
| IN | debug - Y/N or T/F | CHAR(1) |
| OUT | outString | VECTOR(VARCHAR(255)) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inString | ‘This is a delimited string’ |
| IN | inDelimiter | ‘ ‘ |
| IN | debug | N |
| OUT | outString | {‘This’,’is’,’a’,’delimited’,’string’} |

## TextUtils

This section describes the custom java procedure ‘TextUtils’ which contains several text manipulation utilities.

### TextUtils/Blob2Varchar (Custom Function)

Converts a BLOB data type to a VARCHAR. CIS does not natively support this conversion (it *does* natively support CLOB to VARCHAR, however.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | BlobVal | BLOB |
| OUT | result | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | BlobVal | <blob value> |
| OUT | result | <blob value as VARCHAR> |

### TextUtils/CCNumberFormatter (Custom Function)

Provides basic check and standard formatting of a Credit Card number. Validates the length of the supplied numeric field, tries matching it to one of the Visa, MC, AmEx or Discover card patterns, and performs Luhn's validation on the number.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inCCNumber | VARCHAR(2147483647) |
| OUT | outCCNumber | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inCCNumber | ‘5412345678901232’ |
| OUT | outCCNumber | ‘5412 3456 7890 1232’ |

### TextUtils/CSVFromCISQuery (Custom Function)

Converts a result set from a CIS query into a CSV string. The inputs “separator\_character” and “qualifier\_character” should be either a single character or NULL. The input “create\_column\_headers” indicates whether to include column names as the first row. It should be either “true” or “false”.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | query\_string | VARCHAR(2147483647) |
| IN | separator\_character | VARCHAR(2147483647) |
| IN | qualifier\_character | VARCHAR(2147483647) |
| IN | create\_column\_headers | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | query\_string | ‘SELECT \* FROM /shared/examples/ds\_orders/shippingmethods’ |
| IN | separator\_character | ‘,’ |
| IN | qualifier\_character | ‘”’ |
| IN | create\_column\_headers | ‘true’ |
| OUT | result | ‘ShippingMethodID, ShippingMethod  1, UPS Ground  …’ |

### TextUtils/CSVFromCISQueryToFile

Similar to TextUtils/CSVFromCISQuery, this CJP converts a result set from a CIS query into a CSV string, however the result is then dumped to a file on the CIS host filesystem. The inputs “separator\_character” and “qualifier\_character” should be either a single character or NULL. The input “create\_column\_headers” indicates whether to include column names as the first row. It should be either “true” or “false”. The “total\_columns” field indicates the expected number of columns in the result and is used as a validation check. The “append” input field indicates whether to append to a file if it already exists (0 = “do not append”, 1 = “append”.) An integer is returned indicating success (0) or failure (1).

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | query\_string | VARCHAR(2147483647) |
| IN | separator\_character | VARCHAR(2147483647) |
| IN | qualifier\_character | VARCHAR(2147483647) |
| IN | create\_column\_headers | VARCHAR(2147483647) |
| IN | total\_columns | INTEGER |
| IN | file\_path | VARCHAR(2147483647) |
| IN | append | SMALLINT |
| OUT | result | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | query\_string | ‘SELECT \* FROM /shared/examples/ds\_orders/shippingmethods’ |
| IN | separator\_character | ‘,’ |
| IN | qualifier\_character | ‘”’ |
| IN | create\_column\_headers | ‘true’ |
| IN | total\_columns | 2 |
| IN | file\_path | ‘C:\shippingmethods.csv’ |
| IN | append | 0 |
| OUT | result | 0 |

### TextUtils/FixedFromCISQuery (Custom Function)

Converts a result set from a CIS query into a fixed-width formatted string. The input “format\_string” indicates the format of each fixed width row. The format is a pipe separated list of integers indicating the width of each column (col1\_Size|col2\_Size|...|coln\_Size). The input “create\_column\_headers” indicates whether to include column names as the first row. It should be either “true” or “false”. The “total\_columns” field indicates the expected number of columns in the result and is used as a validation check.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | query\_string | VARCHAR(2147483647) |
| IN | format\_string | VARCHAR(2147483647) |
| IN | create\_column\_headers | VARCHAR(2147483647) |
| IN | total\_columns | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | query\_string | ‘SELECT \* FROM /shared/examples/ds\_orders/shippingmethods’ |
| IN | format\_string | ‘2|20’ |
| IN | create\_column\_headers | ‘true’ |
| IN | total\_columns | 2 |
| OUT | result | ‘ShShippingMethod  1 UPS Ground  …’ |

### TextUtils/FixedFromCISQueryToFile

Similar to TextUtils/FixedFromCISQuery, this CJP converts a result set from a CIS query into a fixed-width formatted string, however the result is then dumped to a file on the CIS host filesystem. The input “format\_string” indicates the format of each fixed width row. The format is a pipe separated list of integers indicating the width of each column (col1\_Size|col2\_Size|...|coln\_Size). The input “create\_column\_headers” indicates whether to include column names as the first row. It should be either “true” or “false”. The “total\_columns” field indicates the expected number of columns in the result and is used as a validation check. The “append” input field indicates whether to append to a file if it already exists (0 = “do not append”, 1 = “append”.) An integer is returned indicating success (0) or failure (1).

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | query\_string | VARCHAR(2147483647) |
| IN | format\_string | VARCHAR(2147483647) |
| IN | create\_column\_headers | VARCHAR(2147483647) |
| IN | total\_columns | INTEGER |
| IN | file\_path | VARCHAR(2147483647) |
| IN | append | SMALLINT |
| OUT | result | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | query\_string | ‘SELECT \* FROM /shared/examples/ds\_orders/shippingmethods’ |
| IN | format\_string | ‘2|20’ |
| IN | create\_column\_headers | ‘true’ |
| IN | total\_columns | 2 |
| IN | file\_path | ‘C:\shippingmethods.txt’ |
| IN | append | 0 |
| OUT | result | 0 |

### TextUtils/FormatXML (Custom Function)

Takes an XML string (that has no spaces or newlines) as input and formats the XML, effectively "pretty printing" it.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | UnformattedXML | VARCHAR(2147483647) |
| OUT | FormatedXML | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | UnformattedXML | '<a><b>b text</b><c>c text</c></a>' |
| OUT | FormattedXML | '<a>  <b>b text</b>  <c>c text</c> </a>' |

### TextUtils/GenerateGUID

Generates a random GUID value.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | '42c135c5-945c-40e1-a5a6-67f385e0cea5' |

### TextUtils/HexToRaw (Custom Function)

This procedure converts a hexadecimal string into a binary array value.

NOTE - Calls to this procedure can be pushed to Oracle data sources by adding some custom code to the Oracle capabilities file. Simply edit the $CIS\_HOME/conf/adapters/system/oracle\_<ver>\_<type>\_driver/oracle\_<ver>\_<type>\_driver\_values.xml by adding the following lines just before the closing "</common:attributes>" line. CIS may need to be restarted after making this change.

<ns726:attribute xmlns:ns726="http://www.compositesw.com/services/system/util/common">  
 <ns726:name>/custom/HexToRaw(@null)</ns726:name>  
 <ns726:type>STRING</ns726:type>  
 <ns726:value>HexToRaw($1)</ns726:value>  
 <ns726:configID>HexToRaw(~string)</ns726:configID>  
 </ns726:attribute>  
 <ns725:attribute xmlns:ns725="http://www.compositesw.com/services/system/util/common">  
 <ns725:name>/custom/RawToHex(@null)</ns725:name>  
 <ns725:type>STRING</ns725:type>  
 <ns725:value>RawToHex($1)</ns725:value>  
 <ns725:configID>RawToHex(~binary)</ns725:configID>  
 </ns725:attribute>

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | hexVal | LONGVARCHAR |
| OUT | rawVal | LONGVARBINARY |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | hexVal | '1f2e3d4c' |
| OUT | rawVal | 1F2E3D4C |

### TextUtils/LocalCurrencyFormatter (Custom Function)

Converts a decimal into a localized formatted currency string. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | DOUBLE |
| IN | inFractionLength | INTEGER |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | 12345.6789 |
| IN | inFractionLength | 2 |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | ‘$12,345.68’ |

### TextUtils/LocalCurrencyParser (Custom Function)

Convert a localized formatted currency string to a decimal. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | DOUBLE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | ‘$12,345.68’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | 12345.68 |

### TextUtils/LocalDateFormatter (Custom Function)

Convert a date into a localized formatted date string. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | DATE |
| IN | inStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | 2010-08-10 |
| IN | inStyle | ‘FULL’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | ‘Tuesday, August 10, 2010’ |

### TextUtils/LocalDateParser (Custom Function)

Convert a localized formatted date string to a date. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | VARCHAR |
| IN | inStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | DATE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | ‘Tuesday, August 10, 2010’ |
| IN | inStyle | ‘FULL’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | 2010-08-10 |

### TextUtils/LocalNumberFormatter (Custom Function)

Convert a decimal into a localized formatted numeric string. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | DOUBLE |
| IN | inFractionLength | INTEGER |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | 12345.6789 |
| IN | inFractionLength | 2 |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | ‘12,345.68’ |

### TextUtils/LocalNumberParser (Custom Function)

Convert a localized formatted numeric string to a decimal. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | DOUBLE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | ‘12,345.68’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | 12345.68 |

### TextUtils/LocalTimeFormatter (Custom Function)

Convert a date into a localized formatted time string. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | TIME |
| IN | inStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | 12:34:56 |
| IN | inStyle | ‘FULL’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | ‘12:34:56 PM PST’ |

### TextUtils/LocalTimeParser (Custom Function)

Convert a localized formatted time string to a time. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | VARCHAR |
| IN | inStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | TIME |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | ‘12:34:56 PM PST’ |
| IN | inStyle | ‘FULL’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | 12:34:56 |

### TextUtils/LocalTimestampFormatter (Custom Function)

Convert a date into a localized formatted time string. Country code is optional (pass in a NULL.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | TIMESTAMP |
| IN | inDateStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | inTimeStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | 2010-08-10 12:34:56 |
| IN | inDateStyle | ‘FULL’ |
| IN | inTimeStyle | ‘FULL’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | ‘Tuesday, August 10, 2010 12:34:56 PM PDT’ |

### TextUtils/LocalTimestampParser (Custom Function)

Convert a localized formatted timestamp string to a timestamp. Country code is optional (pass in a NULL.) See also time/extractTimestamp.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inValue | VARCHAR |
| IN | inDateStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | inTimeStyle  ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’ | VARCHAR |
| IN | ISO639LangCode  (see http://www.loc.gov/standards/iso639-2/php/English\_list.php) | VARCHAR |
| IN | ISO3166CountryCode  (see http://www.iso.org/iso/country\_codes/iso\_3166\_code\_lists/ english\_country\_names\_and\_code\_elements.htm) | VARCHAR |
| OUT | outValue | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inValue | ‘Tuesday, August 10, 2010 12:34:56 PM PDT’ |
| IN | inDateStyle | ‘FULL’ |
| IN | inTimeStyle | ‘FULL’ |
| IN | ISO639LangCode | ‘EN’ |
| IN | ISO3166CountryCode | ‘US’ |
| OUT | outValue | 2010-08-10 12:34:56 |

### TextUtils/PhoneNumberFormatter (Custom Function)

Provides standard formatting of phone numbers. If format string is not specified as an input (null or blank), a number will be formatted using the '%.3s-%.3s-%.4s' format (i.e. 999-999-9999.) See <http://download.oracle.com/javase/6/docs/api/java/util/Formatter.html> for details on formatting syntax.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inPhoneNumber | VARCHAR(2147483647) |
| IN | inOutputFormat | VARCHAR(2147483647) |
| OUT | outPhoneNumber | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inPhoneNumber | ‘(732) 236-5438’ |
| IN | inOutputFormat | NULL |
| OUT | outPhoneNumber | ‘732-236-5438’ |

### TextUtils/RawToHex (Custom Function)

This procedure converts a binary array value into a hexadecimal string.

NOTE - Calls to this procedure can be pushed to Oracle data sources by adding some custom code to the Oracle capabilities file. Simply edit the $CIS\_HOME/conf/adapters/system/oracle\_<ver>\_<type>\_driver/oracle\_<ver>\_<type>\_driver\_values.xml by adding the following lines just before the closing "</common:attributes>" line. CIS may need to be restarted after making this change.

<ns726:attribute xmlns:ns726="http://www.compositesw.com/services/system/util/common">  
 <ns726:name>/custom/HexToRaw(@null)</ns726:name>  
 <ns726:type>STRING</ns726:type>  
 <ns726:value>HexToRaw($1)</ns726:value>  
 <ns726:configID>HexToRaw(~string)</ns726:configID>  
 </ns726:attribute>  
 <ns725:attribute xmlns:ns725="http://www.compositesw.com/services/system/util/common">  
 <ns725:name>/custom/RawToHex(@null)</ns725:name>  
 <ns725:type>STRING</ns725:type>  
 <ns725:value>RawToHex($1)</ns725:value>  
 <ns725:configID>RawToHex(~binary)</ns725:configID>  
 </ns725:attribute>

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | rawVal | LONGVARBINARY |
| OUT | hexVal | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | rawVal | 1F2E3D4C |
| OUT | hexVal | '1f2e3d4c' |

### TextUtils/RegexCount (Custom Function)

Count all occurrences of a regular expression match in a VARCHAR and returns the count of the match (similar to the SQL POSITION function, positions start at 1 with 0 indicating a match was not found.) If a NULL value is passed in as the value of any of the inputs, a NULL is returned.

The regular expression language used is what is supported by the JDK used by CIS (currently 1.5 in CIS 4.0.1) See the javadoc for java.util.regex.Pattern for details on what is supported.

See the performance note for RegexFind above for details on how RegEx patterns are cached.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Input Text | LONGVARCHAR |
| IN | Regular Expression | LONGVARCHAR |
| OUT | result | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Input Text | ‘abaabaaabaaaa’ |
| IN | Regular Expression | ‘ba+’ (matches at least one ‘a’) |
| OUT | result | 3 |

### TextUtils/RegexFind (Custom Function)

Finds an occurrence of a regular expression match in a VARCHAR and returns the match. The value of the occurrence input value determines which occurrence to return (numbered starting at 1 from left to right. Use negative values to number occurrences from right to left.) If no match is found, then a NULL is returned. If a NULL value is passed in as the value of any of the inputs, a NULL is returned. Zero may not be used as a value for an occurrence.

The regular expression language used is what is supported by the JDK used by CIS (currently 1.6 in CIS 5.1.0) See the javadoc for java.util.regex.Pattern for details on what is supported.

Performance note: Instead of compiling a new regular expression pattern every time one of the RegEx CJP’s is called, the CJP looks up the pattern in a pattern cache to see if it hasn’t already been compiled. This greatly enhances performance when the same pattern is used repeatedly (i.e. as a function call on a result set column.) The cache is capped at 256 patterns (with the least recently used pattern being replaced when a new pattern is compiled) so that CIS’s memory management system is not impacted by having a lot of compiled patterns taking up memory.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Input Text | VARCHAR |
| IN | Regular Expression | VARCHAR |
| IN | Occurrence | INTEGER |
| OUT | result | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Input Text | ‘abaabaaabaaaa’ |
| IN | Regular Expression | ‘a+’ (matches at least one ‘a’) |
| IN | Occurrence | 3 |
| OUT | result | ‘aaa’ |

### TextUtils/RegexGetGroups

Similar to RegexFind, RegexGetGroups finds an occurrence of a regular expression match in a VARCHAR and returns the matched groups (parenthesized groupings) as rows in a cursor. Group 0 is traditionally the entire matched expression and will always be returned if the regular expression matches. The value of the occurrence input value determines which occurrence to return (numbered starting at 1 from left to right. Use negative values to number occurrences from right to left.) If no match is found, then an empty result set is returned. If a NULL value is passed in as the value of any of the inputs, an empty result set is returned. Zero may not be used as a value for an occurrence.

The regular expression language used is what is supported by the JDK used by CIS (currently 1.6 in CIS 5.1.0) See the javadoc for java.util.regex.Pattern for details on what is supported.

See the performance note for RegexFind above for details on how RegEx patterns are cached.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Input Text | VARCHAR |
| IN | Regular Expression | VARCHAR |
| IN | Occurrence | INTEGER |
| OUT | result | CURSOR(  groupNumber INTEGER,  matchedGroup VARCHAR ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Input Text | ‘(650) 227-8200’ |
| IN | Regular Expression | ‘\(?(\d{3})\)?[\s\-]\*(\d{3})[\s\-]\* (\d{4})’  Parses a U.S. phone number in several formats. |
| IN | Occurrence | 1 |
| OUT | result | (0, ‘(650) 227-8200’), (1, ‘650’), (2, ‘227’), (3, ‘8200’) |

### TextUtils/RegexPosition (Custom Function)

Finds an occurrence of a regular expression match in a VARCHAR and returns the position of the match (similar to the SQL POSITION function, positions start at 1 with 0 indicating a match was not found.) The value of the occurrence input value determines which occurrence to return (numbered starting at 1 from left to right. Use negative values to number occurrences from right to left.) If a NULL value is passed in as the value of any of the inputs, a NULL is returned. Zero may not be used as a value for an occurrence.

The regular expression language used is what is supported by the JDK used by CIS (currently 1.6 in CIS 5.1.0) See the javadoc for java.util.regex.Pattern for details on what is supported.

See the performance note for RegexFind above for details on how RegEx patterns are cached.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Input Text | VARCHAR |
| IN | Regular Expression | VARCHAR |
| IN | Occurrence | INTEGER |
| OUT | result | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Input Text | ‘abaabaaabaaaa’ |
| IN | Regular Expression | ‘a+’ (matches at least one ‘a’) |
| IN | Occurrence | 3 |
| OUT | result | 6 |

### TextUtils/RegexReplace (Custom Function)

Finds an occurrence of a regular expression match in a VARCHAR and replaces the match with the replacement text input value. The value of the occurrence input value determines which occurrence to replace (numbered starting at 1 from left to right. Use negative values to number occurrences from right to left.) Zero may be used as a value for an occurrence and indicates that ALL matches should be replaced. If no match is found, then the original input text is returned. If a NULL value is passed in as the value of any of the inputs, the original input text is returned.

The regular expression language used is what is supported by the JDK used by CIS (currently 1.6 in CIS 5.1.0) See the javadoc for java.util.regex.Pattern for details on what is supported. Also see the javadoc for java.util.regex.Matcher (specifically for the appendReplacement() method) for detail on how to include grouped (as distinguished from "matched") text in the replacement text.

See the performance note for RegexFind above for details on how RegEx patterns are cached.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Input Text | VARCHAR |
| IN | Regular Expression | VARCHAR |
| IN | Replacement Text | VARCHAR |
| IN | Occurrence | INTEGER |
| OUT | result | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Input Text | ‘abaabaaabaaaa’ |
| IN | Regular Expression | ‘a+’ (matches at least one ‘a’) |
| IN | Replacement Text | ‘i$0’ (puts an ‘i’ in front of the matched text.) |
| IN | Occurrence | 0 (all occurrences) |
| OUT | result | ‘iabiaabiaaabiaaaa’ |

### TextUtils/RegexSplit

This function uses Java's String.split() method to split a string using a regular expression and a limit.

Paraphrased from String's javadoc:

The cursor returned by this function contains each substring of this string that is terminated by another substring that matches the given expression or is terminated by the end of the string. The substrings in the cursor are in the order in which they occur in this string. If the expression does not match any part of the input then the resulting cursor has just one row, namely this string.

The limit parameter controls the number of times the pattern is applied and therefore affects the length of the resulting cursor. If the limit n is greater than zero then the pattern will be applied at most n - 1 times, the cursor's cardinality will be no greater than n, and the cursor's last row will contain all input beyond the last matched delimiter. If n is non-positive then the pattern will be applied as many times as possible and the cursor can have any number of rows. If n is zero then the pattern will be applied as many times as possible, the cursor can have any number of rows, and trailing empty strings will be discarded.

The string 'boo:and:foo', for example, yields the following results with these parameters:

|  |  |  |
| --- | --- | --- |
| **Regex** | **Limit** | **Result** |
| : | 2 | 'boo', 'and:foo' |
| : | 5 | 'boo', 'and', 'foo' |
| : | -2 | 'boo', 'and', 'foo' |
| o | 5 | 'b', '', ':and:f', '', '' |
| o | -2 | 'b', '', ':and:f', '', '' |
| o | 0 | 'b', '', ':and:f' |

The regular expression language used is what is supported by the JDK used by CIS (currently 1.6 in CIS 5.1.0) See the javadoc for java.util.regex.Pattern for details on what is supported.

See also string/splitByDelimiter().

See the performance note for RegexFind above for details on how RegEx patterns are cached.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Input Text | VARCHAR |
| IN | Regular Expression | VARCHAR |
| IN | Limit | INTEGER |
| OUT | result | CURSOR(  splitElement VARCHAR ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | Input Text | ‘abaacaaabaaaa’ |
| IN | Regular Expression | ‘[bc]’ (delimit using the characters ‘b’ or ‘c’) |
| IN | Limit | 0 |
| OUT | result | (‘a’), (‘aa’), (‘aaa’) , (‘aaaa’) |

### TextUtils/SSNumberFormatter (Custom Function)

Provides standard formatting of a Social Security number.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inSSNumber | VARCHAR(2147483647) |
| OUT | outSSNumber | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inSSNumber | ‘111223456’ |
| OUT | outSSNumber | ‘111-22-3456’ |

### TextUtils/URLDecode

Decodes a URL encoded string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | InputText - The text to decode | VARCHAR(2147483647) |
| IN | CharacterEncoding - The name of a valid character encoding. 'ISO-8859-1' if NULL. | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | InputText |  |
| IN | CharacterEncoding |  |
| OUT | result |  |

### TextUtils/URLEncode

URL encodes a string.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | InputText - The text to encode | VARCHAR(2147483647) |
| IN | CharacterEncoding - The name of a valid character encoding. 'ISO-8859-1' if NULL. | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | InputText |  |
| IN | CharacterEncoding |  |
| OUT | result |  |

1. How To Use ‘Templates’ Procedures

## Introduction

This section will show how to use the Templates procedures.

### procedureTemplate

This entire procedure is a template that shows the best practice for documenting a CIS Procedure. This is a procedure description that describes the purpose of this procedure.

1. **Parameters: none**
2. **Examples: none**
3. How To Use ‘Time’ Procedures

## Introduction

This section will show how to use the ‘Time’ manipulation procedures.

### DefaultValues (deprecated)

This procedure is now deprecated and contains a vector of valid date, time and timestamp formats. It is NO LONGER used by time/extractDate(), time/extractTime(), and time/extractTimestamp().

### extractDate (Custom Function)

This procedure takes an input string that is expected to contain a date value. It compares this string against a (non-exhaustive) set of date formats, and extracts the date value.

Exceptions: InvalidDateException

If isMandatory=0 and no format could be found then return null.

If isMandatory=1 and no format could be found then an exception is thrown.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDateStr | VARCHAR(255) |
| IN | isMandatory | SMALLINT |
| IN | debug  Y/N or T/F | CHAR(1) |
| OUT | extractDate | DATE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inDateStr | Aug 7, 2010 |
| IN | isMandatory  0 means not mandatory,  1 means mandatory. | 0 |
| IN | debug | Y |
| OUT | extractDate  If isMandatory=0 and no format could be found then return null.  If isMandatory=1 and no format could be found then an exception is thrown. | 2010-07-27 |

### extractTime (Custom Function)

This procedure takes an input string that is expected to contain a time value. It compares this string against a (non-exhaustive) set of time formats, and extracts the time value.

Exceptions: InvalidTimeException

If isMandatory=0 and no format could be found then return null.

If isMandatory=1 and no format could be found then an exception is thrown.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inTimeStr | VARCHAR(255) |
| IN | isMandatory | SMALLINT |
| IN | debug  Y/N or T/F | CHAR(1) |
| OUT | extractTime | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inTimeStr | ‘12:00:00’ |
| IN | isMandatory  0 means not mandatory,  1 means mandatory. | 0 |
| IN | debug | Y |
| OUT | extractTimeIf isMandatory=0 and no format could be found then return null.  If isMandatory=1 and no format could be found then an exception is thrown. | 12:00:00 |

### extractTimestamp (Custom Function)

Extract a timestamp from a string using a list of default timestamp formats. Throws an exception (InvalidTimestampException) if no valid timestamp could be extracted. See also string/TextUtils/LocalTimestampParser().

Exceptions: InvalidTimestampException

If isMandatory=0 and no format could be found then return null.

If isMandatory=1 and no format could be found then an exception is thrown.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inTimestampStr | VARCHAR(255) |
| IN | isMandatory | SMALLINT |
| IN | debug  Y/N or T/F | CHAR(1) |
| OUT | extractTimestamp | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inTimestampStr | ‘Aug 7, 2010 12:00:00.101’ |
| IN | isMandatory  0 means not mandatory,  1 means mandatory. | 0 |
| IN | debug | Y |
| OUT | extractTimestamp  If isMandatory=0 and no format could be found then return null.  If isMandatory=1 and no format could be found then an exception is thrown. | 2010-08-07 12:00:00.101 |

### getCurrentTimestamp (Custom Function)

Get the current timestamp. Added this procedure getCurrentTimestamp to workaround a bug introduced by 7.0.3 server when setting CURRENT\_TIMESTAMP within the context of a procedure. Modified to get the current timestamp from the server instead of using CURRENT\_TIMESTAMP. CURRENT\_TIMESTAMP has some issues with time zone.

CALL /services/webservices/system/admin/server/operations/getServerAttributes(

'<server:getServerAttributes xmlns:server="http://www.compositesw.com/services/system/admin/server">

<server:paths>

<server:path>/server/config/info/currentTime</server:path>

</server:paths>

</server:getServerAttributes>', resultResponse, faultResponse);

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | currentTS | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | currentTS | 2018-02-01 12:00:00 |

### getTimestampInterval (Custom Function)

Add a value of “hours” to a given timestamp value to calculate time.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inInterval  measured in hours | INTEGER |
| IN | inTimestamp | TIMESTAMP |
| OUT | endDatetime | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inInterval | 25 |
| IN | inTimestamp | 2010-07-10 00:00:00 |
| OUT | endDatetime | 2010-07-11 01:00:00 |

### intervalDay2Seconds (Custom Function)

Converts an INTERVAL DAY data type to an equivalent number of seconds. If either input is NULL a NULL will be returned.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inIntervalDay – An INTERVAL DAY value. | INTERVAL DAY TO SECOND |
| OUT | result – The number of seconds (including any fractional component) in the input interval. | DOUBLE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inIntervalDay | 0 00:05:00 |
| OUT | result | 300.0 |

### period2IntervalDay (Custom Function)

Converts a specified period to an INTERVAL DAY data type. If either input is NULL a NULL will be returned.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | amount - The number of units (specified by periodName) | INTEGER |
| IN | periodName - The unit of measure for the "amount" input: second, minute, hour, day, week, month (31 days), year | VARCHAR(20) |
| OUT | result - An interval of the specified period | INTERVAL DAY(9) TO SECOND |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | amount | 1 |
| IN | periodName | ‘week’ |
| OUT | result | 7 00:00:00 |

## DateUtils

This section describes the use of the custom java procedure (DateUtils) which are used for various date manipulations.

### BigintToTimestamp (Custom Function)

This procedure converts a long integer (in milliseconds since The Epoch, otherwise known as midnight on January 1, 1970 GMT) to a TIMESTAMP. Negative long integer values are allowed.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inBigint | BIGINT |
| OUT | result | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inBigint | 0 |
| OUT | Result | '1969-12-31 16:00:00' (if server is in PST timezone.) |

### DateUtils/DateAddDate (Custom Function)

Returns a new Date value based on adding a datePart to the specified Date. It is leap year aware. Valid values for datePart are ‘second’, ‘minute’, ‘hour’, ‘day’, ‘week’, ‘month’ and ‘year’ (these not case sensitive.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | datePart | VARCHAR |
| IN | dateLength | INTEGER |
| IN | startDate | DATE |
| OUT | endDate | DATE |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | datePart | ‘day’ |
| IN | dateLength | 29 |
| IN | startDate | ‘2008-02-01’ |
| OUT | endDate | ‘2008-03-01’ |

### DateUtils/DateAddTimestamp (Custom Function)

Returns a new Timestamp value based on adding a datePart to the specified Timestamp. It is leap year aware. Valid values for datePart are ‘second’, ‘minute’, ‘hour’, ‘day’, ‘week’, ‘month’ and ‘year’ (these not case sensitive.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | datePart | VARCHAR |
| IN | dateLength | INTEGER |
| IN | startDate | TIMESTAMP |
| OUT | endDate | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | datePart | ‘second’ |
| IN | dateLength | 2505600 |
| IN | startDate | ‘2008-02-01 00:00:00’ |
| OUT | endDate | ‘2008-03-01 00:00:00’ |

### DateUtils/DateDiffDate (Custom Function)

Returns the difference between two dates in the specified unit of measure. It is leap year aware. Valid values for datePart are ‘second’, ‘minute’, ‘hour’, ‘day’, ‘week’, ‘month’ and ‘year’ (these not case sensitive.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | datePart | VARCHAR |
| IN | startDate | DATE |
| IN | endDate | DATE |
| OUT | dateLength | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | datePart | ‘day’ |
| IN | startDate | ‘2008-02-01’ |
| IN | endDate | ‘2008-03-01’ |
| OUT | dateLength | 29 |

### DateUtils/DateDiffTimestamp (Custom Function)

Returns the difference between two timestamps in the specified unit of measure. It is leap year aware. Valid values for datePart are ‘second’, ‘minute’, ‘hour’, ‘day’, ‘week’, ‘month’ and ‘year’ (these not case sensitive.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | datePart | VARCHAR |
| IN | startTimestamp | TIMESTAMP |
| IN | endTimestamp | TIMESTAMP |
| OUT | dateLength | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | datePart | ‘second’ |
| IN | startTimestamp | ‘2008-02-01 00:00:00’ |
| IN | endTimestamp | ‘2008-03-01 00:00:00’ |
| OUT | dateLength | 2505600 |

### DateUtils/GetServerTimezone (Custom Function)

Returns the time zone this instance of CIS is running in. Various display types are available:

|  |  |
| --- | --- |
| **Display Type** | **Description** |
| ‘ID’ | The time zone ID (according to Java, i.e. "America/Los\_Angeles") |
| ‘SHORT\_NAME’ | The name of the time zone in short format (i.e. "PDT") |
| ‘LONG\_NAME’ | The name of the time zone in long format (i.e. "Pacific Daylight Time") |
| ‘OFFSET’ | The number of milliseconds to add to GMT time to get the current time zone's time (i.e. "-28800000") |
| 'XML' | The offset in hours and minutes from GMT (i.e. "-08:00") that is expected for an XML dateTime or time field. |

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | displayType | VARCHAR |
| OUT | outValue | VARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | displayType | ‘SHORT\_NAME’ |
| OUT | extractTimestamp | ‘PDT’ |

### TimestampToBigint (Custom Function)

This procedure converts a TIMESTAMP to a long integer (in milliseconds since The Epoch, otherwise known as midnight on January 1, 1970 GMT.) Timestamp values from before The Epoch are allowed.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inTimestamp | TIMESTAMP |
| OUT | result | BIGINT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inBigint | '1969-12-31 16:00:00' (if server is in PST timezone.) |
| OUT | Result | 0 |

### DateUtils/TZConverter (Custom Function)

Converts a timestamp from one time zone to another. Valid time zone input values can be found in the “Info” tab of this CJP.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | sourceTimestamp | TIMESTAMP |
| IN | fromTimeZone | VARCHAR |
| IN | toTimeZone | VARCHAR |
| OUT | targetTimestamp | TIMESTAMP |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | sourceTimestamp | ‘2010-01-01 00:00:00’ |
| IN | fromTimeZone | ‘UTC’ |
| IN | toTimeZone | ‘PST8PDT’ |
| OUT | targetTimeZone | ‘2009-12-31 16:00:00’ |

1. How To Use ‘Upgrade’ Procedures

## Introduction

This section describes the routines using the “upgrade” procedures. These procedures and views are designed to assist with major upgrades of CIS.

### getDatabaseTests

This view returns default regression testing entries that can be used with JMeter regression suite for upgrade testing. It returns queries for all published views for all virtual databases, except the 'system' or 'examples' databases.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | N/A | CURSOR (  "Skip Execution" CHAR(2),  "Test Name" VARCHAR(21),  "Test Query" LONGVARCHAR,  "Test Plan" VARCHAR(260),  "Service Name" VARCHAR(255) ) |

### getServiceTests

This view returns default regression testing entries that can be used with JMeter regression suite for upgrade testing. It returns queries for all published operations for all virtual web services, except the 'admin' or 'util' services.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | N/A | CURSOR (  "Skip Execution" CHAR(2),  "Test Name" VARCHAR(21),  "Operation Path" LONGVARCHAR,  "Test Plan" VARCHAR(260),  "Service Name" VARCHAR(255) ) |

### updateCacheConfigTables

This procedure will update the cache config tables - cache\_status and cache\_tracking - to allow a new server to use existing cache data as part of a CIS server version upgrade. This can be done as part of a migration from CIS 6.2 to CIS 7.0, and allow cache data to be preserved after the upgrade.

**NOTE:** BEFORE RUNNING THIS PROCEDURE, ENSURE THAT CACHING IS DISABLED ON THE OLD SERVER AND NEW SERVER.

**NOTE:** THIS PROCEDURE IS TO BE RUN ON THE TARGET SITE FOR UPGRADE - I.E. IF YOU ARE UPGRADING FROM CIS 6.2 TO 7.0, EXECUTE THE PROCEDURE ON THE CIS 7.0 SERVER, PROVIDING THE CIS 6.2 SERVER'S SERVER\_ID AS INPUT.

**NOTE:** AFTER RUNNING THIS PROCEDURE, RE-ENABLE CACHING ON THE TARGET SITE FOR UPGRADE ONLY.

**NOTE:** ALSO RECOMMEND SETTING CACHE STATUS SYNC INTERVAL TO 60 SECONDS ON SOURCE AND TARGET SERVER BEFORE RUNNING.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **performDelete** – Indicates whether to delete existing entries for this server. | BIT |
| IN | **performInsert** – Indicates whether to insert new entries for this server. | BIT |
| IN | **previousServerID** – Server ID of the previous CIS server that is being upgraded | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | performDelete | 1 |
| IN | performInsert | 1 |
| IN | previousServerID | 'cgoodric-vm-win7x64-9400-945983814' |

## helpers

This section describes the auxiliary procedures for upgrade.

### helpers/configuredCaches

This view returns all distinct caching target data sources

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | N/A | CURSOR (  DATASOURCE\_PATH VARCHAR(4096),  STATUS\_PATH VARCHAR(4096),  TRACKING\_PATH VARCHAR(4096) ) |

### helpers/findCaches

This procedure finds cached resources configured in CIS managed cache\_status tables.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| OUT | result | CURSOR (  RESOURCE\_PATH VARCHAR(4096),  DATASOURCE\_PATH VARCHAR(4096),  STATUS\_PATH VARCHAR(4096),  TRACKING\_PATH VARCHAR(4096) ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| OUT | result | (  '/shared/examples/ds\_orders/tutorial/orders',  '/shared/examples/ds\_orders',  '/shared/examples/ds\_orders/tutorial/cache\_status',  '/shared/examples/ds\_orders/tutorial/cache\_tracking' ), … |

### helpers/returnColumnOrderingString

This procedure generates an ORDER BY string with the numCols numbered columns. It is useful for generating an ORDER BY clause for use with automated regression testing.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **numCols** – The number of columns to include in the generated ORDER BY string. | INTEGER |
| OUT | **orderByString** – The ORDER BY string result | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | | **Parameter Name** | **Parameter Value** |
| --- | --- | --- | --- |
| IN | numCols | | 10 |
| OUT | orderByString | | 'ORDER BY 1, 2, 3, 4, 5, 6, 7, 8, 9, 10' |

1. How To Use ‘Validate’ Procedures

## Introduction

This section describes the routines using the “validate” procedures. These procedures and views are designed to assist with validating DV resources.

### compareMetadataTargetServer

This procedure is used to evaluate the lineage of a published resource and compare the metadata tables against a target server environment metadata. Resources from source environment are assumed to be in the same location on the target server. If the resource does not exist, it is reported that way.

The objective is alert the development team of differences in the metadata on the target server before a deployment is performed. The idea is to know what is going to happen before it happens so that the developer can prepare ahead of time.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDebug - N=no debug. Y=debug on. | CHAR(1) |
| IN | startingResourcePath - The resource path to interrogate. May be published resource, folder or datasource. | LONGVARCHAR |
| IN | targetDVSchemaPath - DV Path to a "Composite Datasource" schema where the following resources are published:  getBasicResourceCursor\_SQL\_TABLE and getServerAttribute  Example: /shared/Common/ComparisonTools/DV\_Server/ Utilities/repository | LONGVARCHAR |
| IN | sendEmail - null/N=do not send an email. Y=send email for each resource with a violation/warning F or W. | CHAR(1) |
| IN | emailList - A comma-separated list of valid emails | LONGVARCHAR |
| OUT | result | PIPE (startingLineagePath VARCHAR(4000),  message LONGVARCHAR) |

### compareResourcesTargetServer

This procedure is used to compare resources on the current "source" DV server with the same resource path on a "target" DV server. Pre-Requisites:

A DV "Composite Datasource" must be created somewhere in the "source" DV server that introspects the script "/shared/ASAssets/Utilities/repository/getScriptText" on the target DV server.

A DV dynamic path to a "Composite Datasource" with getScriptText must be provided as input.

Do not applyReservedListToPath as that will be done within this procedure.

Example: /shared/Common/ComparisonTools/TDV\_Server/Utilities/repository/getScriptText

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDebug - N=no debug. Y=debug on. | CHAR(1) |
| IN | startingResource - starting container or resource path | LONGVARCHAR |
| IN | targetDVSchemaPath - DV Path to a "Composite Datasource" schema where the following resources are published:  getScriptText  Example: /shared/Common/ComparisonTools/ DV\_Server/Utilities/repository | LONGVARCHAR |
| IN | statusFilterList - If null, then display all STATUS, otherwise display the status that are listed separated by spaces or commas.  Status List:  EQUAL = The two resources are exactly the same (equivalent).  DIFFERENT = The two resources are different.  TARGET\_NOT\_EXIST = The target resource does not exist for the same path as the source resource.  BOTH\_UNREADABLE = The getScriptText procedure could not read the resource type requested on both DV server.  SOURCE\_UNREADABLE = The getScriptText procedure could not read the resource type requested on the source DV server.  TARGET\_UNREADABLE = The getScriptText procedure could not read the resource type requested on the target DV server.  UNKNOWN = The status could not be determined | VARCHAR |
| OUT | result | PIPE (  resPath VARCHAR(4096),  resType VARCHAR,  subtype VARCHAR,  status VARCHAR, -- EQUAL, DIFFERENT, TARGET\_NOT\_EXIST, BOTH\_UNREADABLE, SOURCE\_UNREADABLE, TARGET\_UNREADABLE, UNKNOWN  sourceNumChars INTEGER,  targetNumChars INTEGER,  sourceScript LONGVARCHAR,  targetScript LONGVARCHAR,  message LONGVARCHAR  ) |

### validateDeployment

The purpose of this procedure is to validate the .car file metadata.xml resources against previous deployments. The objective is to both record the resources being deployed and validate that a resource archive date is not going to overwrite a previous deployment where the archive date is greater than the one being deployed. Under that circumstance, the older archive could overwrite newer resource logic.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **debug** – N=no debug. Y=debug on. | CHAR(1) |
| IN | **externalID** – An external id to correlate to an external system. | VARCHAR(50) |
| IN | **archiveCreationDateStrIn** – Format in archive file: Tue Sep 10 14:26:28 EDT 2019 | VARCHAR |
| IN | **deploymentDateStr** – Format: YYYY-MM-DD HH24:MI:SS.FFF | VARCHAR |
| IN | **nodeHost** - The hostname of the server deploying to. | VARCHAR |
| IN | **nodePort** - The port of the server deploying to. | INTEGER |
| IN | **resourceMetadataFilePath** - The file path to the package archive metadata file [metadata.xml]. Unzipped from the .car file. | VARCHAR(4000) |
| OUT | **Result** - SUCCESS upon completion or EXCEPTION | LONGVARCHAR |

### validatePublishedResources

This procedure is used to evaluate a group of DV source code resources for a given starting path and validate any published resources that are impacted. Send an email with the list of impacted resources.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **inDebug** – N=no debug. Y=debug on. | CHAR(1) |
| IN | **startingContainer** – The folder path to interrogate. Must be a published resource path only. | LONGVARCHAR |
| IN | **sendEmail** – null/N=do not send an email. Y=send email for each resource with a violation/warning F or W. | CHAR(1) |
| IN | **emailList** – A comma-separated list of valid emails | LONGVARCHAR |
| OUT | **result** | PIPE (  groupNum INTEGER, -- A numerical sequential value indicating what group number the set of resources belong to.  violation CHAR(1), -- F=Fail, W=Warning, P=pass.  ruleViolations VARCHAR(255), -- A list of rule violations that have occurred for a given resource.  ruleWarnings VARCHAR(255), -- A list of rule warnings that have occurred for a given resource.  ruleMessage VARCHAR(500), -- A message to be relayed to the user.  parentPath VARCHAR(4000), -- The parent path of the resource being evaluated.  resourcePath VARCHAR(4000), -- The actual path of the resource being evaluated.  resourceType VARCHAR(255), -- The resource type of the resource being evaluated.  resourceName VARCHAR(255), -- The resource name of the resource being evaluated.  ownerName VARCHAR(255), -- The owner name of the resource being modified.  ownerDomain VARCHAR(255), -- The owner domain of the resource being modified.  modUser VARCHAR(255), -- The modified user of the resource being modified.  modDomain VARCHAR(255), -- The modified user domain of the resource being modified.  modDate TIMESTAMP, -- The modified timestamp of the resource being modified.  impactLevel VARCHAR(255), -- The impact level (type) of the resource being modified.  impactMessage VARCHAR(32767) -- The impact message of the resource beaing modified.  ) |

### validateSharedResourcesRules

This procedure is used to evaluate a group of DV source code resources for a given starting path and provide an output of code that fails the source code check rules.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **inDebug** – N=no debug. Y=debug on. | CHAR(1) |
| IN | **startingContainer** – The folder path to interrogate. Must be a published resource path only. | LONGVARCHAR |
| IN | **displayViolations** - Display violation types: A[all],P[pass],W[warning],F[fail] or a combination like W,F. A comma separated list. Null returns all. | VARCHAR |
| IN | **includeList** - Comma separated list of evaluation rules to include. | VARCHAR |
| IN | **excludeList** - Comma separated list of evaluation rules to exclude. ExcludeList overrides includeList. | VARCHAR |
| IN | **sendEmail** – null/N=do not send an email. Y=send email for each resource with a violation/warning F or W. | CHAR(1) |
| IN | **emailList** – A comma-separated list of valid emails | LONGVARCHAR |
| IN | **pathNotAllowed** - comma separated list of resource paths or partials paths to exclude when traversing a published resources lineage using getResourceLineageDirectRecursive().  Example: '/shared/EnterpriseDatabase/Physical/Formatting, /lib,/services/webservices/system' | LONGVARCHAR |
| IN | **excludePathsList** - Comma separated list of resource paths or partials paths to exclude while processing. | LONGVARCHAR |
| IN | **maxNumberWebServiceOperations** - maximum number of web service operations allowed per web service | INTEGER |
| IN | **waiverList** - A formatted list of resources that have been granted a waiver from processing. These resources were previously in violation.  The format is as follows: '|' || w.resourceType || ':' || w.ResourcePath || '|'  e.g. |/shared/path1/view1:TABLE| /shared/path2/proc2:PROCEDURE| | LONGVARCHAR |
| OUT | **result** | PIPE (  groupNum INTEGER, -- A numerical sequential value indicating what group number the set of resources belong to.  violation CHAR(1), -- F=Fail, W=Warning, P=pass.  ruleViolations VARCHAR(255), -- A list of rule violations that have occurred for a given resource.  ruleWarnings VARCHAR(255), -- A list of rule warnings that have occurred for a given resource.  ruleMessage VARCHAR(500), -- A message to be relayed to the user.  parentPath VARCHAR(4000), -- The parent path of the resource being evaluated.  resourcePath VARCHAR(4000), -- The actual path of the resource being evaluated.  resourceType VARCHAR(255), -- The resource type of the resource being evaluated.  resourceName VARCHAR(255), -- The resource name of the resource being evaluated.  ownerName VARCHAR(255), -- The owner name of the resource being modified.  ownerDomain VARCHAR(255), -- The owner domain of the resource being modified.  modUser VARCHAR(255), -- The modified user of the resource being modified.  modDomain VARCHAR(255), -- The modified user domain of the resource being modified.  modDate TIMESTAMP, -- The modified timestamp of the resource being modified.  impactLevel VARCHAR(255), -- The impact level (type) of the resource being modified.  impactMessage VARCHAR(32767) -- The impact message of the resource beaing modified.  ) |

## helpers

This section describes the auxiliary procedures for validation.

### helpers/evaluateResource

This procedure is used to evaluate DV source code and provide an output of code that fails the source code check rules. Evaluate rules list:

1=impacted resource

2=select distinct

3=select \*

4=order by

5=published path not allowed

6=shared path(s) not allowed (specified)

7=users path not allowed

8=limit number of web service operations

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inDebug - N=no debug. Y=debug on. | CHAR(1) |
| IN | resourcePath - The actual resource path | LONGVARCHAR |
| IN | resourceType - The resource type [LINK if published, TABLE or PROCEDURE if in /shared area]. | VARCHAR |
| IN | displayViolations - Display violation types: A[all],P[pass],W[warning],F[fail] or a combination like W,F. A comma separated list. Null returns all. | VARCHAR |
| IN | includeList - Comma separated list of evaluation rules to include. | VARCHAR |
| IN | excludeList - Comma separated list of evaluation rules to exclude. ExcludeList overrides includeList. | VARCHAR |
| IN | inGroupNum - The group number associated with the list of resource generated by the incoming resourcePath. | INTEGER |
| IN | pathNowAllowed - Comma separated list of paths that are not allowed to be directly accessed from views above the formatting layer. | LONGVARCHAR |
| OUT | result | PIPE (  groupNum INTEGER, -- A numerical sequential value indicating what group number the set of resources belong to.  violation CHAR(1), -- F=Fail, W=Warning, P=pass.  ruleViolations VARCHAR(255), -- A list of rule violations that have occurred for a given resource.  ruleWarnings VARCHAR(255), -- A list of rule warnings that have occurred for a given resource.  ruleMessage VARCHAR(500), -- A message to be relayed to the user.  parentPath VARCHAR(4000), -- The parent path of the resource being evaluated.  resourcePath VARCHAR(4000), -- The actual path of the resource being evaluated.  resourceType VARCHAR(255), -- The resource type of the resource being evaluated.  resourceName VARCHAR(255), -- The resource name of the resource being evaluated.  ownerName VARCHAR(255), -- The owner name of the resource being modified.  ownerDomain VARCHAR(255), -- The owner domain of the resource being modified.  modUser VARCHAR(255), -- The modified user of the resource being modified.  modDomain VARCHAR(255), -- The modified user domain of the resource being modified.  modDate TIMESTAMP, -- The modified timestamp of the resource being modified.  impactLevel VARCHAR(255), -- The impact level (type) of the resource being modified.  impactMessage VARCHAR(32767) -- The impact message of the resource beaing modified.  ) |

### helpers/evaluateResourceOrderBy

This procedure is used to determine whether "order by" is being used within the SQL script except when used within the context of analytical functions which is valid.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | inSqlScript | LONGVARCHAR |
| OUT | containsOrderBy | BIT |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | inSqlScript | Select \* from orders order by 1 |
| OUT | containsOrderBy | 1 |

1. How To Use ‘XML’ Procedures

## Introduction

This section will show how to use the ‘XML’ manipulation procedures.

### castXMLTextNodeAsVarchar (Custom Function)

This script converts XML text node to varchar. Does appropriate conversion from XML schema dateTime or time to ANSI format (and even adjusts timezone to server's timezone if timezone provided.)

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | xmlValue | XML |
| OUT | result | /lib/util/System.Text  (VARCHAR(2147483647)) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inStr | XML(‘this is a test’) |
| OUT | outStr | ’this is a test’ |

* 1. **Assumptions: CIS instance in PDT timezone.**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inStr | XML(‘2010-12-15T12:50:00-08:00’) |
| OUT | outStr | ’2010-12-15 12:50:00’ |

### escapeXML (Custom Function)

Change xml tags to their corresponding escape sequences.

The Escape Characters

When writing parsed data to XML:

Character in HDF5 Output Character in XML

quote (") &quot;

apostrophe (') &apos;

ampersand (&) &amp;

less than (<) &lt;

greater than (>) &gt;

slash (\) No escape required

space ( ) No escape required

linefeed CHR(10) &#xa;

carriage return CHR(13) &#xd;

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inStr | LONGVARCHAR |
| OUT | outStr | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inStr | ‘<tag>somevalue</tag>’ |
| OUT | outStr | ’&lt;tag&gt;somevalue&lt;/tag&gt;’ |

### extractXMLOccurrence

Extract XML occurrence refers to the ability to locate XML text based on an opening and closing XML tag. This is useful when needing to extract a particular occurrence of a repeating XML string without having to use expensive XML parsing utilities. If the XML contains escaped XML, the content will automatically be unescaped before proceeding with extraction. Example:

Escaped XML:

&lt;xmlRow&gt;&lt;name&gt;FirstName&lt;/name&gt;&lt;type&gt;STRING&lt;/type&gt;&lt;value&gt;Bob&lt;/value&gt;&lt;/xmlRow&gt;

Converted to actual XML:

<xmlRow><name>FirstName</name><type>STRING</type><value>Bob</value></xmlRow>

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inContentXML | LONGVARCHAR |
| IN | inTagName - This is the opening and closing XML tag name. It is only the tag name with no angle brackets. The opening <tagname> and closing </tagname> brackets will applied within this procedure. The tag anme must be a unique repeating element within the XML. e.g. xmlRow | VARCHAR |
| IN | inOccurrence - The occurrence of the opening and closing delimiter. Must be greater than 0. | INTEGER |
| IN | inIncludeDelimiter - 0 (default) do not include, 1=do include delimiter | INTEGER |
| IN | inTrimText - 0 (default) do not trim, 1=do trim | INTEGER |
| IN | inCaseSensitive - 0 (default) case insensitivie search, 1=case sensitive search | INTEGER |
| OUT | status - FOUND=occurrence found. NOT\_FOUND=occurrence not found | VARCHAR |
| OUT | result  -- if FOUND and result is NULL then a null tag was found. <tagname/>  -- if FOUND and result is empty '' then an empty tag was found <tagname></tagname>  -- if FOUND and result is not emtpy then a value was found <tagname>value</tagname>  -- if NOT\_FOUND then the result is NULL | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inContentXML | <xmlRow>  <name>FirstName</name>  <type>STRING</type>  <value>Bob</value>  </xmlRow>  <xmlRow/>  <xmlRow>  <name>LastName</name>  <type>STRING</type>  <value>Smith</value>  </xmlRow>  <xmlRow>  <name>Age</name>  <type>INTEGER</type>  <value/>  </xmlRow> |
| IN | inTagName | xmlRow |
| IN | inOccurrence | 4 |
| IN | inIncludeDelimiter | 1 |
| IN | inTrimText | 1 |
| IN | inCaseSensitive | 0 |
| OUT | status | FOUND |
| OUT | result | <xmlRow> <name>Age</name> <type>INTEGER</type> <value/> </xmlRow> |

### getNodeFromXML

Given a qualified XPath, extract the node tree from the incoming XML.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug - Y or T = debugging turned on, N or F = debugging turned off | CHAR(1) |
| IN | namespaces - any string with the valid namespaces for the incoming XML | LONGVARCHAR |
| IN | inXpath - xpath statement used to extract the value at that location in the XML | LONGVARCHAR |
| IN | iteration - 0=get all iterations, 1=get 1st occurrence only | INTEGER |
| IN | inXml - any valid XML document | XML |
| OUT | outValue - text value from the XML document as directed by the XPath. Empty String returned if node not found. | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | namespaces | ‘xmlns:sam=”http://www.compositesw.com/samples/mynamespace/v1.0”  xmlns:xsd=”http://www.w3.org/2001/XMLSchema”  xmlns:xlink=”http://www.w3.org/1999/xlink”’ |
| IN | inXpath | ‘/Book/sam:Name/@sam:isbn’ |
| IN | iteration | 1 |
| IN | inXml | ‘<Book xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0">  <sam:Name sam:isbn="12-3456-123">Test</sam:Name>  <Chapter>Test Data</Chapter>  </Book>’ |
| OUT | outValue | 12-3456-123 |

### getValueFromXML (Custom Function)

Given an XPath, extract the value from any XML document.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | debug | CHAR(1) |
| IN | namespaces | LONGVARCHAR |
| IN | inXpath | LONGVARCHAR |
| IN | inXml | XML |
| OUT | outValue | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | namespaces | ‘xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"  xmlns:xsd="http://www.w3.org/2001/XMLSchema"  xmlns:xlink="http://www.w3.org/1999/xlink"’ |
| IN | inXpath | ‘/Book/sam:Name/@sam:isbn’ |
| IN | inXml | ‘<Book xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0">  <sam:Name sam:isbn="12-3456-123">Test</sam:Name>  <Chapter>Test Data</Chapter>  </Book>’ |
| OUT | outValue | ‘12-3456-123’ |

### getXMLNode

Search through the XML text to find the "first" occurrence of the corresponding opening <node\_name> and closing </node\_name> and extract it from the XML and return it. This is a text search. It does not use XSLT. It does not use xpath statements. It does preserve namespaces. Subsequent calls to this script may find additional siblings of a node name given that the closing position is returned and the user can invoke this script with a beginning position. If the opening and closing node names contain a namespace qualifier then it must be included on input.

openingName=<qualifer:node\_name>

closingName=</qualifer:node\_name>

If the node name in the XML content contains a namespace then "DO NOT" include it in the openingName. The xmlns: namespaces will automatically be handled in the script.

openingName=<qualifier:node\_name>

closingName=</qualifer:node\_name>

e.g. <q:a xmlns:qualifer="http://xyz">

openingName=<q:a>

closingName=</q:a>

Example XML

<q:a1 xmlns:qualifer="http://abc">

<q:a xmlns:qualifer="http://xyz">

<b>

<a>a</a>

<a>a</a>

</b>

</q:a>

<q:a xmlns:qualifer="http://xyz">

<c>c</c>

</q:a>

</q:a1>

INVOKE: getXMLNode('N', 1, 'xml-content', '<q:a>', '</q:a>', 0)

DEBUG:

getXMLNode : INPUT: begPos=1 openingName=[<q:a>] openNameLen=5 closingName=[</q:a>] closeNameLen=6

getXMLNode : INPUT: sqlScript=[<q:a1 xmlns:qualifer="http://abc"> <q:a xmlns:qualifer="http://xyz"> <b> <a>a</a> <a>a</a> </b> </q:a> <q:a xmlns:qualifer="http://xyz"> <c>c</c> </q:a> </q:a1>]

getXMLNode : sqlScriptMod=[<q:a1> <q:a> <b> <a>a</a> <a>a</a> </b> </q:a> <q:a> <c>c</c> </q:a> </q:a1>]

getXMLNode : Extract the script text:

getXMLNode : pairCount=0 openList=00000012o,00000076o, closeList=00000065c,00000098c,

getXMLNode : Match the node pair count:

getXMLNode : i=1 pairCount=1 slen=00000012o

getXMLNode : i=2 pairCount=0 slen=00000065c

getXMLNode : EXTRACT: openingPos=12 closingPos=65 + closeNameLen=6

getXMLNode : FINAL extracted results

getXMLNode : status=0 openingPos=40 closingPos=127

getXMLNode : extractedXml=[<q:a xmlns:qualifer="http://xyz"> <b> <a>a</a> <a>a</a> </b> </q:a>]

getXMLNode : extractedXmlBefore=[<q:a1 xmlns:qualifer="http://abc"> ]

getXMLNode : extractedXmlAfter=[ <q:a xmlns:qualifer="http://xyz"> <c>c</c> </q:a> </q:a1>]

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **debug** - Y or T = debugging turned on, N or F = debugging turned off | CHAR(1) |
| IN | **trimText** - Trim the extractedXml. 0 do not trim, 1=do trim. | INTEGER |
| IN | **beginPos** - The position to start searching for the first open/close node name. | INTEGER |
| IN | **openingName** - The opening node name: <qualifer:node\_name> Do not include any namespaces such as <qualifer:node\_name xmlns:qualifier="http://abc"> | VARCHAR |
| IN | **closingName** - The closing node name: </qualifer:node\_name> The opening and closing node names cannot be the same. | VARCHAR |
| IN | **xmlText** - The xml text to parse | LONGVARCHAR |
| OUT | **status** - 0=success (open/close pair found). -1=fail (no open/close pair match found) | INTEGER |
| OUT | **openingPos** - The position of the beginning of the left open node name for the pair. | INTEGER |
| OUT | **closingPos** - The position after the right close node name for the pair. position+node\_name\_len | INTEGER |
| OUT | **extractedXml** - This is the first extracted "xml" between the opening and closing <qualifer:node\_name></qualifer:node\_name>. | LONGVARCHAR |
| OUT | **extractedXmlBefore** - This is the first extracted "xml" before the opening pair "<xml\_before> <qualifer:node\_name> </qualifer:node\_name> <xml\_after>". | LONGVARCHAR |
| OUT | **extractedXmlAfter** - This is the first extracted "xml" after the closing pair "<xml\_before> <qualifer:node\_name> </qualifer:node\_name> <xml\_after>". | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | trimText | 0 |
| IN | beginPos | 1 |
| IN | openingName | <q:a> |
| IN | closingName | </q:a> |
| IN | xmlText | <q:a1 xmlns:q="http://xyz"><q:a xmlns:q="http://abc"><b><q:a xmlns:q="http://abc">a</q:a><q:a>a</q:a></b></q:a><q:a xmlns:q="http://abc"><c>c</c></q:a></q:a1> |
| OUT | Status | 0 |
| OUT | openingPos | 28 |
| OUT | closingPos | 112 |
| OUT | extractedXml | <q:a xmlns:q="http://abc"><b><q:a xmlns:q="http://abc">a</q:a><q:a>a</q:a></b></q:a> |
| OUT | extractedXmlBefore | <q:a1 xmlns:q="http://xyz"> |
| OUT | extractedXmlAfter | <q:a xmlns:q="http://abc"><c>c</c></q:a></q:a1> |

### parseAndModifyXML (deprecated)

Use parseAndModifyXMLV2.

### parseAndModifyXMLV2

Parse and XML document given an XML list of XPath statements and modify the values and reconstruct the XML.

IMPORTANT: When updating an element with a list of attributes, you must supply all of the original attributes in order to retain what you had. This routine overwrites the original element so be careful.

Use cases supported:

* Use Case 1: Modify the attribute for an element
* Use Case 2: Modify the value of the the element with attributes
* Use Case 3: Modify the element with no attributes
* Use Case 4: Modify the element with no attributes and set it to null
* Use Case 5: Retain the original attributes and modify the element with attributes and set it to null

Limitations: This method does not modify an iteration of the same node.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Debug - Y/N or T/F | CHAR(1) |
| IN | namespaces | LONGVARCHAR |
| IN | ElemAttrXML  contains a list of qualified elements, qualified attributes and the value to change in this format:  SET ElemAttrXML =  -- Repeat the section <qualElemAttr> for as many as are required.  '<qualElemAttr><qualElem></qualElem> <qualAttr></qualAttr> <value></value> </qualElemAttr>'||CHR(10)||  '<qualElemAttr><qualElem></qualElem> <qualAttr></qualAttr> <value></value> </qualElemAttr>'||CHR(10); | LONGVARCHAR |
| IN | inXml | XML |
| OUT | outXml | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | namespaces | 'xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"' |
| IN | ElemAttrXML | <qualElemAttr><qualElem>sam:Name</qualElem><qualAttr>sam:isbn</qualAttr><value>00-0000-000</value></qualElemAttr>  <qualElemAttr><qualElem>sam:Name</qualElem><qualAttr></qualAttr><value>The Book</value></qualElemAttr>  <qualElemAttr><qualElem>Chapter</qualElem><qualAttr></qualAttr><value>Chapter 1</value></qualElemAttr>  <qualElemAttr><qualElem>Paragraph</qualElem><qualAttr></qualAttr><value></value></qualElemAttr>  <qualElemAttr><qualElem>sam:Author</qualElem><qualAttr>sam:firstname</qualAttr><value>Joe</value></qualElemAttr>  <qualElemAttr><qualElem>sam:Author</qualElem><qualAttr>sam:lastname</qualAttr><value>Author</value></qualElemAttr>  <qualElemAttr><qualElem>sam:Author</qualElem><qualAttr></qualAttr><value></value></qualElemAttr> |
| IN | inXml | ‘<Book xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0">  <sam:Name sam:isbn="12-3456-123"/>  <Chapter>Test Data</Chapter>  <Paragraph>Paragraph 1</Paragraph>  </Book> ' |
| OUT | outXml | ‘<Book xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0">  <sam:Name sam:isbn="00-0000-000">Joe Author</sam:Name>  <Chapter>Chapter 1</Chapter>  <Paragraph xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:nil="true"/>  </Book>’ |

### parseProperties

Parse the SQL plan properties XML from the SQL Plan.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **debug** - Y or T = debugging turned on, N or F = debugging turned off | CHAR(1) |
| IN | **propertiesEntry** - An XML string containing a property node with name/value pair entries in it. | LONGVARCHAR |
| OUT | **result** – pipe output cursor with an iteration of property name and value pairs. | PIPE (  propname VARCHAR(255),  propvalue LONGVARCHAR  ) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | propertiesEntry | <execute:properties>  <execute:property>  <execute:name>Name</execute:name>  <execute:value>SELECT</execute:value>  </execute:property>  <execute:property>  <execute:name>Estimated Rows Returned</execute:name>  <execute:value>Unknown (at least 0 rows)</execute:value>  </execute:property>  <execute:property>  <execute:name>Projection</execute:name>  <execute:value>orderdetails.orderid OrderID  </execute:property>  <execute:property>  <execute:name>No Parallelization Reason</execute:name>  <execute:value>Unsupported operating system</execute:value>  </execute:property>  </execute:properties> |
| OUT | result | |  |  | | --- | --- | | propname | propvalue | | Name | SELECT | | Estimated Rows Returned | Unknown (at least 0 rows) | | No Parallelization Reason | Unsupported operating system | |

### pruneXML (deprecated)

Use pruneXMLV2.

### pruneXMLV2

Prune out empty XML nodes except those in the retainXPathList.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | Debug - Y/N or T/F | CHAR(1) |
| IN | namespaces - any string with the valid namespaces for the incoming XML | LONGVARCHAR |
| IN | retainXPathList - contains a double || separated list of XPath statements for the incoming XML that insure those nodes are retained in the XML if they are empty. The invoking of this procedure must define and populate the delimited list. The invoker will determine the XPath statement by consulting the XML Schema for required elements and the deterimine from their application which nodes have the possibility of being empty. Those are the candidates for this list.  Values: e.g. /rootnode/XYZ/A/@foo||/rootnode/XYZ/C | LONGVARCHAR |
| IN | inXml – any valid XML document | XML |
| OUT | outXml – any valid XML document | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | debug | ‘N’ |
| IN | namespaces | ‘xmlns:tns="http://mynamespace" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xlink="http://www.w3.org/1999/xlink"’ |
| IN | retainXPathVector | /rootnode/XYZ/A/@foo||/rootnode/XYZ/C |
| IN | inXml | ‘<rootnode xmlns:tns="http://mynamespace">  <XYZ abc="def" ghi="">  <A foo="">aa</A>  <B>bbb</B>  <C/>  </XYZ>  </rootnode>’ |
| OUT | outXml | ‘<rootnode xmlns:tns="http://mynamespace">  <XYZ abc="def">  <A foo="">aa</A>  <B>bbb</B>  <C/>  </XYZ>  </rootnode>’ |

### reverseXML (Custom Function)

This provides a way to reverse the direction of an XML document so that CIS thinks it is calling a data source.  This is the only way that you can change an XSLT data source into a transformation.   CIS does not allow you to simply build a pure XML transformation.  It always thinks you are transforming an XML file.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inXml | XML |
| OUT | outXml | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inXml | <tag>somevalue</tag> |
| OUT | outXml | <tag>somevalue</tag> |

### stripInvalidXMLChars (Custom Function)

Strip the invalid characters from an XML document. This procedure is still under construction

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inXml | XML |
| OUT | outXml | XML |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inXml |  |
| OUT | outXml |  |

### unescapeXML (Custom Function)

Change xml entities to their corresponding individual characters.

The Escape Characters

When writing parsed data to XML:

Character in HDF5 Output Character in XML

quote (") &quot;

apostrophe (') &apos;

ampersand (&) &amp;

less than (<) &lt;

greater than (>) &gt;

slash (\) No escape required

space ( ) No escape required

linefeed CHR(10) &#xa;

carriage return CHR(13) &#xd;

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inStr | LONGVARCHAR |
| OUT | outStr | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inStr | ‘&quot;A &amp; B &lt; C &amp; D&quot;’ |
| OUT | outStr | ’“A & B < C & D“’ |

## XMLUtils

This section describes the use of the custom java procedure (DateUtils) which are used for various date manipulations.

### XMLUtils/CSVFromXMLToFile

Similar to TextUtils/CSVFromCISQueryToFile, this CJP converts a result set in XML format into a CSV string and dumps the result to a file on the CIS host filesystem. The inputs “separator\_character” and “qualifier\_character” should be either a single character or NULL. The input “create\_column\_headers” indicates whether to include column names as the first row. It should be either “true” or “false”. The “total\_columns” field indicates the expected number of columns in the result and is used as a validation check. The “append” input field indicates whether to append to a file if it already exists (0 = “do not append”, 1 = “append”.) An integer is returned indicating success (0) or failure (1).

Note: For best results, the XML string should be formatted with repeated rows containing all expected columns in each row. Deviation from this pattern may result in unexpected behavior. The specific XML node names and number of columns do not matter as long as it follows the example pattern shown below:

<?xml version="1.0"?>

<p1:Customer xmlns:p1="http://www.compositesw.com/ps/FileProcessor">

<row>

<customerID>1</customerID>

<companyName>Composite Software</companyName>

<contactFirstName>John</contactFirstName>

<contactLastName>Doe</contactLastName>

<billingAddress>1234 First Avenue NE</billingAddress>

<city>Reston</city>

<stateOrProvince>VA</stateOrProvince>

<postalCode>22190</postalCode>

<countryRegion>USA</countryRegion>

<contactTitle>Mr</contactTitle>

<phoneNumber>(703)111-2222</phoneNumber>

<faxNumber>(703)111-3333</faxNumber>

</row>

<row>

<customerID>2</customerID>

<companyName>Company 2</companyName>

<contactFirstName>Jane</contactFirstName>

<contactLastName>Doe</contactLastName>

<billingAddress>5678 Second Street NW</billingAddress>

<city>Washington</city>

<stateOrProvince>DC</stateOrProvince>

<postalCode>10002</postalCode>

<countryRegion>US</countryRegion>

<contactTitle>Mrs</contactTitle>

<phoneNumber>202-111-2222</phoneNumber>

<faxNumber>202-111-3333</faxNumber>

</row>

</p1:Customer>

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | xml\_string | VARCHAR(2147483647) |
| IN | separator\_character | VARCHAR(2147483647) |
| IN | qualifier\_character | VARCHAR(2147483647) |
| IN | create\_column\_headers | VARCHAR(2147483647) |
| IN | total\_columns | INTEGER |
| IN | file\_path | VARCHAR(2147483647) |
| IN | append | SMALLINT |
| OUT | result | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | query\_string | See example XML above |
| IN | separator\_character | ‘,’ |
| IN | qualifier\_character | ‘”’ |
| IN | create\_column\_headers | ‘true’ |
| IN | total\_columns | 2 |
| IN | file\_path | ‘C:\customer.csv’ |
| IN | append | 0 |
| OUT | result | 0 |

### XMLUtils/DeleteElement (Custom Function)

Removes an element (including it’s children) from an XML structure.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inXML | VARCHAR(2147483647) |
| IN | inElementName | VARCHAR(2147483647) |
| IN | inElementNamespace | VARCHAR(2147483647) |
| IN | occurrence | INTEGER |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inXML | ‘<xmldoc xmlns=”uri:mynamespace”>  <parent>  <child>  <grandchild/>  </child>  </parent>  </xmldoc> |
| IN | inElementName | ‘child’ |
| IN | inElementNamespace | ‘uri:mynamespace’ |
| IN | occurrence | 1 |
| OUT | result | ‘<xmldoc xmlns=”uri:mynamespace”>  <parent/>  </xmldoc>’ |

### XMLUtils/DeleteElementSpareChildren (Custom Function)

Removes an element from an XML structure. The element’s children become children of the element’s parent.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inXML | VARCHAR(2147483647) |
| IN | inElementName | VARCHAR(2147483647) |
| IN | inElementNamespace | VARCHAR(2147483647) |
| IN | occurrence | INTEGER |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inXML | ‘<xmldoc xmlns=”uri:mynamespace”>  <parent>  <child>  <grandchild/>  </child>  </parent>  </xmldoc>’ |
| IN | inElementName | ‘child’ |
| IN | inElementNamespace | ‘uri:mynamespace’ |
| IN | occurrence | 1 |
| OUT | result | ‘<xmldoc xmlns=”uri:mynamespace”>  <parent>  <grandchild/>  </parent>  </xmldoc>’ |

### XMLUtils/FixedFromXMLToFile

Similar to TextUtils/FixedFromCISQueryToFile, this CJP converts a result set from a result set in XML format into a fixed-width formatted string then the result is dumped to a file on the CIS host filesystem. The input “format\_string” indicates the format of each fixed width row. The format is a pipe separated list of integers indicating the width of each column (col1\_Size|col2\_Size|...|coln\_Size). The input “create\_column\_headers” indicates whether to include column names as the first row. It should be either “true” or “false”. The “total\_columns” field indicates the expected number of columns in the result and is used as a validation check. The “append” input field indicates whether to append to a file if it already exists (0 = “do not append”, 1 = “append”.) An integer is returned indicating success (0) or failure (1).

Note: For best results, the XML string should be formatted with repeated rows containing all expected columns in each row. Deviation from this pattern may result in unexpected behavior. The specific XML node names and number of columns do not matter as long as it follows the example pattern shown below:

<?xml version="1.0"?>

<p1:Customer xmlns:p1="http://www.compositesw.com/ps/FileProcessor">

<row>

<customerID>1</customerID>

<companyName>Composite Software</companyName>

<contactFirstName>John</contactFirstName>

<contactLastName>Doe</contactLastName>

<billingAddress>1234 First Avenue NE</billingAddress>

<city>Reston</city>

<stateOrProvince>VA</stateOrProvince>

<postalCode>22190</postalCode>

<countryRegion>USA</countryRegion>

<contactTitle>Mr</contactTitle>

<phoneNumber>(703)111-2222</phoneNumber>

<faxNumber>(703)111-3333</faxNumber>

</row>

<row>

<customerID>2</customerID>

<companyName>Company 2</companyName>

<contactFirstName>Jane</contactFirstName>

<contactLastName>Doe</contactLastName>

<billingAddress>5678 Second Street NW</billingAddress>

<city>Washington</city>

<stateOrProvince>DC</stateOrProvince>

<postalCode>10002</postalCode>

<countryRegion>US</countryRegion>

<contactTitle>Mrs</contactTitle>

<phoneNumber>202-111-2222</phoneNumber>

<faxNumber>202-111-3333</faxNumber>

</row>

</p1:Customer>

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | xml\_string | VARCHAR(2147483647) |
| IN | format\_string | VARCHAR(2147483647) |
| IN | create\_column\_headers | VARCHAR(2147483647) |
| IN | total\_columns | INTEGER |
| IN | file\_path | VARCHAR(2147483647) |
| IN | append | SMALLINT |
| OUT | result | INTEGER |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | query\_string | See XML example above |
| IN | format\_string | ‘5|40|20|40|40|40|5|10|2|10|20|20’ |
| IN | create\_column\_headers | ‘true’ |
| IN | total\_columns | 2 |
| IN | file\_path | ‘C:\customer.txt’ |
| IN | append | 0 |
| OUT | result | 0 |

### XMLUtils/HTMLtoXML

This procedure converts HTML into XHTML (XML). HTML has a looser tagging syntax than XML so this procedure uses the JTidy library (http://jtidy.sourceforge.net) to clean up the HTML to conform to the XHTML standard.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | inHTML | LONGVARCHAR |
| OUT | outXML | LONGVARCHAR |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inHTML | ‘<html><body><table><tr><td></table></body></html>’ |
| OUT | outXML | ‘<html> <head> <META http-equiv="Content-Type" content="text/html; charset=UTF-8"> <meta name="generator" content="HTML Tidy for Java (vers. 2009-12-01), see jtidy.sourceforge.net"> <title></title> </head> <body> <table> <tr> <td></td> </tr> </table> </body> </html>’ |

### XMLUtils/InsertElementDemoteChildren (Custom Function)

Inserts an element from into an XML structure. The parent element’s children become children of the new element.

1. **Parameters:**

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | XML | VARCHAR(2147483647) |
| IN | parentElementName | VARCHAR(2147483647) |
| IN | parentElementNamespace | VARCHAR(2147483647) |
| IN | occurrence | INTEGER |
| IN | ElementName | VARCHAR(2147483647) |
| IN | ElementNamespace | VARCHAR(2147483647) |
| OUT | result | VARCHAR(2147483647) |

1. **Examples:**
   1. **Assumptions: none**

| **Direction** | **Parameter Name** | **Parameter Value** |
| --- | --- | --- |
| IN | inXML | ‘<xmldoc xmlns=”uri:mynamespace”>  <parent>  <grandchild/>  </parent>  </xmldoc>’ |
| IN | parentElementName | ‘parent’ |
| IN | parentElementNamespace | ‘uri:mynamespace’ |
| IN | occurrence | 1 |
| IN | ElementName | ‘child’ |
| IN | ElementNamespace | ‘uri:mynamespace’ |
| OUT | result | ‘<xmldoc xmlns=”uri:mynamespace”>  <parent>  <child>  <grandchild/>  </child>  </parent>  </xmldoc> |

1. How To Submit New Procedures

## Introduction

This section will provide guidelines for submitting new procedures to the consolidated CIS development utilities library.

## Documentation

More than anything else, the procedures in this library need to be documented. SQL Scripts need to have documentation similar to the following in both a header comment and also in the “Annotations” field in the “Info” tab. With CJP’s, the code is not visible to the CIS developer, but the source code should also have comments with the same information and also have documentation in each CJP’s “Annotation” field. (Having documentation in the “Annotation” field keeps things consistent regardless of whether the procedure is an SQL Script or a CJP.)

Recommended documentation format:

/\*

Description:

Description of the procedure

Usage note: Describe any requirements (such as "calling user must have ACCESS\_TOOLS right") or other things to be aware of.

Inputs:

myInput1 - Describe the input(s), whether or not it is optional (along with what to pass to indicate that the input should be ignored, usually NULL), and what will be used as a default.

Outputs:

myOutput1 - Describe the output(s)

Exceptions:

myException1 - Describe any explicitly raised exception(s) (no need to detail every possible raised exception, just the ones explicitly raised in the code.)

Modified Date: Modified By: CSW Version: Reason:

mm/dd/yyyy (Author’s Name) (Lowest supported CIS version)(Reason for change)

Example:

Modified Date: Modified By: CSW Version: Reason:

05/01/2013 Mike Tinius 6.2.3 Created new

08/01/2013 Mike Tinius 6.2.4 Fixed format issue

\*/

The procedure found in templates/procedureTemplate() has comments and annotations already set up for this. Please copy and use this as a basis for all development utilities scripts. (A similar CJP template needs to be set up as well.)

As much as possible, we should provide examples of how utilities are used so that other developers can get a better understanding of how each utility works. Please put such examples in the /shared/ASAssets/Utilities/examples folder.

## Regression Test Cases

Along with the new procedure and documentation, a set of regression test cases (sample inputs and expected outputs) will help us automate the process of QA testing new releases and allow us to more easily see if new changes will impact previously known good test cases. Please submit as many of these as possible.

## Source Code Control

The master image of the Utilities distribution is currently on <http://github.com/TibcoSoftware/ASAssets_Utilities>. All official distributions of the Utilities are created from this instance.

CJP source code will be checked into the JavaSource. CIS resources will be exported using the VCS system and checked into the DVSource folder.

## Peer Review

Peer review provides a mechanism to reduce duplication of effort and also a way to enforce consistency across all the utilities.

Before being included in the development utilities library, new submissions will be peer reviewed by one or more of the development utilities team. Subsequent changes can be checked in by the original developer but will again be peer reviewed. This peer review will also be conducted whenever a member of the development utilities team submits new or updated procedures.

## Team Members

The following are members of the development utilities team:

* Michael Tinius ([mtinius@tibco.com)](mailto:mtinius@tibco.com))