



***Advanced Services Assets***  
***Field Developed Utilities***

Data Virtualization Business Unit Advanced Services  
Q4 2014

## TABLE OF CONTENTS

|   |           |
|---|-----------|
| <b>RELEASE NOTES FOR VERSION 2014 Q4 .....</b>          | <b>13</b> |
| Introduction.....                                       | 13        |
| New Resources.....                                      | 13        |
| Updated Resources.....                                  | 14        |
| Deprecated Resources.....                               | 14        |
| 2014 Q4.....  | 14        |
| 2014 Q3.....  | 14        |
| 2014 Q1.....  | 15        |
| 2012 Q4.....  | 15        |
| 2012 Q1.....  | 15        |
| 2011 Q3.....  | 15        |
| Regression Test Versions.....                           | 15        |
| <b>INTRODUCTION .....</b>                               | <b>16</b> |
| Purpose.....  | 16        |
| History.....  | 16        |
| Audience .....  | 17        |
| Installation Notes .....                                | 17        |
| New Folder Structure .....                              | 17        |
| Reserved Word List .....                                | 19        |
| <b>TOP LEVEL UTILITIES PROCEDURES.....</b>              | <b>20</b> |
| Introduction.....                                       | 20        |
| ExceptionDefinitions .....                              | 20        |
| getUtilitiesVersion (Custom Function).....              | 20        |
| reintrospectCJPs .....                                  | 20        |
| TypeDefinitions .....                                   | 21        |
| <b>HOW TO USE ‘ACTIVE DIRECTORY’ PROCEDURES .....</b>   | <b>22</b> |
| Introduction.....                                       | 22        |
| ActiveDirectoryInt8ToDate (Custom Function).....        | 22        |
| ActiveDirectoryTSToSQLTimeStamp (Custom Function) ..... | 23        |
| SimpleBinaryAND (Custom Function).....                  | 23        |
| <b>HOW TO USE ‘CALCULATION’ PROCEDURES.....</b>         | <b>25</b> |
| Introduction.....                                       | 25        |
| calculateAge (Custom Function).....                     | 25        |
| medianFromQuery (Custom Function) .....                 | 25        |
| <b>HOW TO USE ‘CONVERSION’ PROCEDURES .....</b>         | <b>27</b> |
| Introduction.....                                       | 27        |

|  |           |
|--|-----------|
| convertBit (Custom Function).....                        | 27        |
| convertBoolean (Custom Function).....                    | 27        |
| convertDoubleToInteger (Custom Function).....            | 28        |
| convertTemperatureUnit (Custom Function) .....           | 28        |
| convertYN (Custom Function).....                         | 29        |
| <b>HOW TO USE 'DOCUMENTATION' PROCEDURES .....</b>       | <b>30</b> |
| Introduction.....  | 30        |
| getDocumentationDriver .....                             | 30        |
| getAllDocumentationAPI .....                             | 32        |
| constants .....  | 36        |
| documentationTrigger.....                                | 38        |
| helpers .....  | 39        |
| helpers/getDocConstant (Custom Function).....            | 39        |
| helpers/getDocCounts .....                               | 39        |
| helpers/parseDocSwitches (Custom Function) .....         | 40        |
| implementations.....                                     | 42        |
| implementations/getDocPreambleImpl1 .....                | 42        |
| implementations/getDocResourceFormatImpl1 .....          | 43        |
| implementations/getDocResourceFormatImpl1_resource ..... | 44        |
| modules .....  | 47        |
| modules/getDocDataSourceLineage .....                    | 47        |
| modules/getDocResourceProjection.....                    | 49        |
| modules/getDocResourcesUsed .....                        | 50        |
| <b>HOW TO USE 'ENCODING' PROCEDURES .....</b>            | <b>52</b> |
| Introduction.....  | 52        |
| CIS_JCE_PROVIDERS_VIEW .....                             | 52        |
| EncodingCJP .....  | 52        |
| EncodingCJP/Base64Decode (Custom Function).....          | 52        |
| EncodingCJP/Base64Encode (Custom Function) .....         | 53        |
| EncodingCJP/CISSecurityProviders .....                   | 53        |
| EncodingCJP/DecryptFrom3DES .....                        | 54        |
| EncodingCJP/DecryptWithCISPrivKey .....                  | 54        |
| EncodingCJP/EncryptWith3DES .....                        | 55        |
| EncodingCJP/EncryptWithCISPubKey .....                   | 55        |
| EncodingCJP/MD5Hash (Custom Function) .....              | 56        |
| EncodingCJP/SHA1Hash (Custom Function) .....             | 56        |
| <b>HOW TO USE 'FILE' PROCEDURES.....</b>                 | <b>58</b> |
| Introduction.....  | 58        |
| copyAll.....   | 58        |
| getCisHome (Custom Function) .....                       | 58        |
| getFileSeparator (Custom Function) .....                 | 59        |
| removeAllFilter .....                                    | 59        |
| FileProcessingCJP.....                                   | 60        |
| FileProcessingCJP/archiveFile .....                      | 60        |

|   |           |
|---|-----------|
| FileProcessingCJP/archiveFileTimestamp .....  | 60        |
| FileProcessingCJP/copyFile .....  | 61        |
| FileProcessingCJP/createFileASCII .....   | 61        |
| FileProcessingCJP/createFileBinary .....  | 62        |
| FileProcessingCJP/existsDir (Custom Function) .....                                 | 62        |
| FileProcessingCJP/existsFile (Custom Function) .....                                | 62        |
| FileProcessingCJP/getFileContentsAscii (Custom Function) .....                      | 63        |
| FileProcessingCJP/getFileContentsBinary (Custom Function) .....                     | 63        |
| FileProcessingCJP/getFileInfo .....   | 64        |
| FileProcessingCJP/getNewFiles .....   | 65        |
| FileProcessingCJP/gunzipFile (Custom Function) .....                                | 65        |
| FileProcessingCJP/makeDirs (Custom Function) .....                                  | 66        |
| FileProcessingCJP/removeAll (Custom Function) .....                                 | 66        |
| FileProcessingCJP/remove (Custom Function) .....                                    | 67        |
| FileProcessingCJP/unzipFile (Custom Function) .....                                 | 67        |
| <b>HOW TO USE 'LOGGING' PROCEDURES.....</b>   | <b>68</b> |
| Introduction .....  | 68        |
| auditLogger .....   | 68        |
| logDebugMessage .....   | 70        |
| <b>HOW TO USE 'NET' PROCEDURES .....</b>  | <b>71</b> |
| Introduction .....  | 71        |
| NetUtils .....  | 71        |
| ftpFile .....   | 71        |
| <b>HOW TO USE 'PDTOOL' PROCEDURES.....</b>  | <b>72</b> |
| Introduction .....  | 72        |
| generatePDTToolDeployableResourcePlanByDate .....                                   | 72        |
| generatePDTToolDeployableResourcePlanByLineage .....                                | 73        |
| template_generatePDTToolDeployableResourcePlan .....                                | 75        |
| helpers .....   | 75        |
| helpers/getDeployableResourceListByDate .....                                       | 75        |
| helpers/getDeployableResourceListByLineage .....                                    | 76        |
| helpers/getDistinctDeployableResourceListByDate .....                               | 77        |
| helpers/getDistinctDeployableResourceListByLineage .....                            | 78        |
| <b>HOW TO USE 'REPOSITORY' PROCEDURES .....</b>                                     | <b>80</b> |
| Introduction .....  | 80        |
| CIS Types and Subtypes listing .....  | 80        |
| Listing of CIS resource types and subtypes .....                                    | 80        |
| The ACCESS_TOOLS Right .....  | 82        |
| Note On Using Repository Helper Procedures With Triggers and Cache Procedures ..... | 82        |
| CIS Repository Helper Procedures .....  | 83        |
| addRemoveDataSourceChildren .....   | 83        |
| applyReservedListToPath (Custom Function) .....                                     | 84        |
| applyReservedListToWord (Custom Function) .....                                     | 84        |

|   |     |
|---|-----|
| configureReservedList.....                              | 84  |
| cachedResources .....                                   | 84  |
| changePassword .....                                    | 85  |
| compareCisVersions (Custom Function).....               | 86  |
| copyResources .....                                     | 87  |
| copyResourceAnnotations.....                            | 87  |
| copyResourcesPrivileges.....                            | 88  |
| createAllFolders.....                                   | 89  |
| createAllFoldersPrivileges .....                        | 90  |
| createConnector .....                                   | 91  |
| createDataSource .....                                  | 92  |
| createFolder .....                                      | 95  |
| createOrUpdateConnector.....                            | 96  |
| createResource .....                                    | 97  |
| createResourceCopy .....                                | 98  |
| createUnionView .....                                   | 99  |
| deleteAllConnectors.....                                | 100 |
| deleteConnector .....                                   | 100 |
| destroyResource .....                                   | 100 |
| expireProcCacheEntryByName .....                        | 101 |
| exportResourceDefinitions.....                          | 102 |
| fixLeadingCharactersInFolderPath (Custom Function)..... | 103 |
| getAllDataSourceChildren .....                          | 103 |
| getAllDataSources .....                                 | 104 |
| getAncestorResources .....                              | 105 |
| GetAnsi2NativeMapping .....                             | 106 |
| getBasicResourceCursor .....                            | 107 |
| getBasicResourceCursor_All .....                        | 108 |
| getBasicResourceCursor_ActionAttributes .....           | 109 |
| getBasicResourceCursor PROCEDURE .....                  | 110 |
| getBasicResourceCursor PROCEDURE_CURSOR .....           | 111 |
| getBasicResourceCursor_ResourceAttributes .....         | 112 |
| getBasicResourceCursor_SQL_TABLE .....                  | 113 |
| getBasicResourceCursor_SQL_TABLE_SQLINDEXES .....       | 115 |
| getBasicResourceCursor_XSLT_TEXT .....                  | 116 |
| getChildResourcesCursor.....                            | 117 |
| getCisVersion (Custom Function).....                    | 118 |
| getConnectors.....                                      | 119 |
| getDataSourceCacheConfig .....                          | 120 |
| getDataSourceStatsConfig .....                          | 120 |
| getDefSetDefs .....                                     | 121 |
| getDependentResourcesCursor .....                       | 122 |
| getDependentResourcesRecurseCursor .....                | 123 |
| getImpactedResources .....                              | 124 |
| getIntrospectableResourceIdsResult .....                | 125 |

|   |     |
|---|-----|
| getIntrospectableResourceIdsTask .....      | 126 |
| getIntrospectedResourceIdsResult .....      | 127 |
| getIntrospectedResourceIdsTask .....        | 128 |
| getLockedResources .....                    | 129 |
| getOutputColDefs .....                      | 130 |
| getResourceAnnotations .....                | 131 |
| getResourceByDate .....                     | 131 |
| getResourceCacheConfig .....                | 132 |
| getResourceCacheConfigCursor .....          | 133 |
| getResourceImpactedCursor .....             | 134 |
| getResourceLineageDatasources .....         | 135 |
| getResourceLineageRecursive .....           | 136 |
| getResourceLineageRecursiveAncestors .....  | 139 |
| getResourceListChildren .....               | 142 |
| getResourceListRecursive .....              | 143 |
| getResourceListUnpublished .....            | 144 |
| getResourcePrivilegeDependencies .....      | 145 |
| getResourcePrivileges .....                 | 148 |
| getResourcePrivilegesByUser .....           | 150 |
| getScriptText (Custom Function) .....       | 151 |
| getTableColumnStatisticsConfiguration ..... | 152 |
| getUsedResourcesCursor .....                | 153 |
| getUsedResourcesRecurseCursor .....         | 154 |
| getUserPermissionsRecursive .....           | 156 |
| impactedTargetsList .....                   | 157 |
| introspectResourcesResult .....             | 158 |
| introspectResourcesTask .....               | 159 |
| rebindFolder .....                          | 162 |
| rebindResource .....                        | 163 |
| recoverFailedCacheRefresh .....             | 164 |
| refreshResourceStatistics .....             | 165 |
| reintrospectDataSource .....                | 166 |
| removeAllFolders .....                      | 166 |
| removePathQuotes .....                      | 167 |
| replaceStringInAnnotations .....            | 167 |
| resourceExists (Custom Function) .....      | 168 |
| returnFolderNameAndFolderPath .....         | 168 |
| searchResources .....                       | 169 |
| updateBasicTransformationProcedure .....    | 172 |
| updateConnector .....                       | 172 |
| updateDefSetDef .....                       | 174 |
| UpdateDsColumnAnnotation .....              | 175 |
| updateExternalSQLProcedure .....            | 175 |
| updateImpactedResources .....               | 176 |
| updateResourceAnnotations .....             | 177 |

|   |            |
|---|------------|
| updateResourceCacheConfig .....                           | 177        |
| updateResourceCacheConfiguration .....                    | 178        |
| updateResourceCacheEnabled .....                          | 180        |
| updateResourceDataSource .....                            | 181        |
| updateResourcesSqlTable .....                             | 182        |
| updateSqlScript .....                                     | 184        |
| updateSqlTable .....                                      | 184        |
| updateSqlTableTextAndModel .....                          | 185        |
| updateStreamTransformationProcedure .....                 | 186        |
| updateTableColumnStatisticsConfiguration .....            | 187        |
| updateTrigger .....                                       | 188        |
| updateXsltTransformationProcedure .....                   | 189        |
| RepoUtils .....   | 189        |
| RepoUtils/applyReservedListToPath .....                   | 190        |
| RepoUtils/applyReservedListToWord .....                   | 190        |
| RepoUtils/EncryptPassword (Custom Function) .....         | 191        |
| RepoUtils/GetAnsi2NativeMapping .....                     | 191        |
| RepoUtils/getReservedWordList .....                       | 192        |
| RepoUtils/GetSystemProperties .....                       | 193        |
| RepoUtils/ GetUserGroups .....                            | 193        |
| RepoUtils/isReservedWord (Custom Function) .....          | 194        |
| RepoUtils/UpdateDsColumnAnnotation .....                  | 194        |
| CIS Repository Definition Procedures .....                | 195        |
| definitions/RepositoryDefinitions .....                   | 195        |
| definitions/RepositoryDefinitionsRecursive .....          | 195        |
| CIS Repository Execute Procedures .....                   | 196        |
| execute/executeProcedure .....                            | 196        |
| execute/executeProcedureResults .....                     | 196        |
| CIS Repository Server Procedures .....                    | 197        |
| server/addLicense .....                                   | 197        |
| server/getServerAttribute (Custom Function) .....         | 198        |
| server/getServerAttributeList .....                       | 198        |
| server/getServerAttributeMap .....                        | 199        |
| server/getServerAttributeMapByKey (Custom Function) ..... | 199        |
| server/updateServerAttribute .....                        | 200        |
| CIS Repository User Procedures .....                      | 201        |
| user/createGroup .....                                    | 201        |
| user/createResourcePrivilege .....                        | 201        |
| user/createUser .....                                     | 202        |
| user/deleteGroup .....                                    | 203        |
| user/deleteUser .....                                     | 204        |
| user/getDomainGroups .....                                | 204        |
| user/getGroup .....                                       | 205        |
| user/getUser .....  | 206        |
| <b>HOW TO USE 'REQUEST' PROCEDURES .....</b>              | <b>208</b> |

|  |            |
|--|------------|
| Introduction.....                                    | 208        |
| terminateIdleSessions .....                          | 208        |
| terminateRequest .....                               | 208        |
| terminateSession .....                               | 209        |
| RequestUtils.....                                    | 209        |
| RequestUtils/DirectSqlRequest (Custom Function)..... | 209        |
| RequestUtils/OriginalRequest (Custom Function).....  | 210        |
| RequestUtils/ReadInEqClause (Custom Function) .....  | 210        |
| RequestUtils/TopSqlRequest (Custom Function) .....   | 211        |
| <b>HOW TO USE ‘STRING’ PROCEDURES .....</b>          | <b>212</b> |
| Introduction.....                                    | 212        |
| addQuotesInList (Custom Function) .....              | 212        |
| basename (Custom Function).....                      | 212        |
| concatNotNull (Custom Function).....                 | 213        |
| dirname (Custom Function) .....                      | 214        |
| emptyStr (Custom Function) .....                     | 214        |
| entityConstants .....                                | 215        |
| entityExtract.....                                   | 215        |
| entityExtractToPipe .....                            | 216        |
| entityExtractToString (Custom Function) .....        | 217        |
| escapeCSV (Custom Function) .....                    | 218        |
| extractBiDelimitedText (Custom Function).....        | 219        |
| extractTextList.....                                 | 220        |
| findString (Custom Function).....                    | 222        |
| findStringInList (Custom Function) .....             | 223        |
| fixQuotes (Custom Function).....                     | 224        |
| getConstant (Custom Function) .....                  | 224        |
| getDelimitedOccurrence (Custom Function).....        | 225        |
| getDelimitedSum (Custom Function) .....              | 225        |
| indent (Custom Function) .....                       | 226        |
| isEmpty (Custom Function) .....                      | 227        |
| last4ofSSN (Custom Function) .....                   | 227        |
| modifyConstant (Custom Function) .....               | 227        |
| normalizeRowsToPipe.....                             | 229        |
| normalizeRowsToString .....                          | 229        |
| p_DelimitedStringToCursor .....                      | 230        |
| p_FixedStringToCursor .....                          | 230        |
| ParseCSVLine .....                                   | 231        |
| removeDoubleQuotes.....                              | 231        |
| removeSingleQuotes .....                             | 232        |
| RegexPatterns .....                                  | 232        |
| splitByDelimiter .....                               | 233        |
| TextUtils .....                                      | 234        |
| TextUtils/Blob2Varchar (Custom Function).....        | 234        |
| TextUtils/CCNumberFormatter (Custom Function) .....  | 234        |

|   |            |
|---|------------|
| TextUtils/CSVFromCISQuery (Custom Function).....          | 235        |
| TextUtils/CSVFromCISQueryToFile .....                     | 236        |
| TextUtils/FixedFromCISQuery (Custom Function) .....       | 237        |
| TextUtils/FixedFromCISQueryToFile .....                   | 237        |
| TextUtils/FormatXML (Custom Function).....                | 238        |
| TextUtils/GenerateGUID .....                              | 239        |
| TextUtils/HexToRaw (Custom Function).....                 | 239        |
| TextUtils/LocalCurrencyFormatter (Custom Function).....   | 240        |
| TextUtils/LocalCurrencyParser (Custom Function) .....     | 241        |
| TextUtils/LocalDateFormatter (Custom Function) .....      | 241        |
| TextUtils/LocalDateParser (Custom Function) .....         | 242        |
| TextUtils/LocalNumberFormatter (Custom Function).....     | 243        |
| TextUtils/LocalNumberParser (Custom Function) .....       | 244        |
| TextUtils/LocalTimeFormatter (Custom Function) .....      | 244        |
| TextUtils/LocalTimeParser (Custom Function).....          | 245        |
| TextUtils/LocalTimestampFormatter (Custom Function) ..... | 246        |
| TextUtils/LocalTimestampParser (Custom Function).....     | 247        |
| TextUtils/PhoneNumberFormatter (Custom Function).....     | 248        |
| TextUtils/RawToHex (Custom Function) .....                | 248        |
| TextUtils/RegexFind (Custom Function).....                | 249        |
| TextUtils/RegexGetGroups.....                             | 250        |
| TextUtils/RegexPosition (Custom Function) .....           | 251        |
| TextUtils/RegexReplace (Custom Function) .....            | 252        |
| TextUtils/RegexSplit.....                                 | 253        |
| TextUtils/SSNumberFormatter (Custom Function) .....       | 254        |
| <b>HOW TO USE ‘TEMPLATES’ PROCEDURES .....</b>            | <b>255</b> |
| Introduction.....   | 255        |
| procedureTemplate .....                                   | 255        |
| <b>HOW TO USE ‘TIME’ PROCEDURES .....</b>                 | <b>256</b> |
| Introduction.....   | 256        |
| ADD_MONTHS (Custom Function).....                         | 256        |
| DefaultValues.....  | 256        |
| extractDate (Custom Function) .....                       | 256        |
| extractTime (Custom Function) .....                       | 257        |
| extractTimestamp (Custom Function) .....                  | 258        |
| getTimestampInterval (Custom Function) .....              | 259        |
| intervalDay2Seconds (Custom Function) .....               | 259        |
| period2IntervalDay (Custom Function) .....                | 260        |
| DateUtils .....   | 260        |
| BigintToTimestamp (Custom Function) .....                 | 261        |
| DateUtils/DateAddDate (Custom Function) .....             | 261        |
| DateUtils/DateAddTimestamp (Custom Function) .....        | 262        |
| DateUtils/DateDiffDate (Custom Function) .....            | 262        |
| DateUtils/DateDiffTimestamp (Custom Function) .....       | 263        |

|  |            |
|--|------------|
| DateUtils/GetServerTimezone (Custom Function) .....          | 263        |
| TimestampToBigint (Custom Function) .....                    | 264        |
| DateUtils/TZConverter (Custom Function) .....                | 265        |
| <b>HOW TO USE ‘XML’ PROCEDURES.....</b>                      | <b>266</b> |
| Introduction.....  | 266        |
| castXMLTextNodeAsVarchar (Custom Function) .....             | 266        |
| CreateXmlString2CursorXForm .....                            | 266        |
| escapeXML (Custom Function) .....                            | 267        |
| getNodeFromXML .....   | 268        |
| getValueFromXML (Custom Function) .....                      | 269        |
| parseAndModifyXML .....                                      | 270        |
| pruneXML .....   | 271        |
| reverseXML (Custom Function) .....                           | 273        |
| stripInvalidXMLChars (Custom Function) .....                 | 273        |
| unescapeXML (Custom Function) .....                          | 273        |
| XMLUtils.....  | 274        |
| XMLUtils/CSVFromXMLToFile.....                               | 274        |
| XMLUtils/DeleteElement (Custom Function) .....               | 276        |
| XMLUtils/DeleteElementSpareChildren (Custom Function) .....  | 277        |
| XMLUtils/FixedFromXMLToFile .....                            | 277        |
| XMLUtils/HTMLtoXML .....                                     | 279        |
| XMLUtils/InsertElementDemoteChildren (Custom Function) ..... | 280        |
| <b>HOW TO SUBMIT NEW PROCEDURES.....</b>                     | <b>282</b> |
| Introduction.....  | 282        |
| Documentation.....   | 282        |
| Regression Test Cases.....                                   | 283        |
| Source Code Control.....                                     | 283        |
| Peer Review .....  | 283        |
| Team Members .....   | 283        |

## DOCUMENT CONTROL

### Version History

| Version  | Date       | Author  | Description  |
|----------|------------|---|--|
| 1.0      | 08/06/2010 | Mike Tinius,<br>Calvin Goodrich,<br>Gordon Rose | Initial revision   |
| 1.1      | 08/11/2010 | Calvin Goodrich                                 | Added documentation for string, time, and repository functions.<br>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   |

### Related Documents

| Document | Date | Author |
|----------|------|--------|
|          |      |        |
|          |      |        |

### Data Virtualization Business Unit (DVBU) Products Referenced

| DVBU Product Name        | Version |
|--------------------------|---------|
| Cisco Information Server | 6.2.x   |

## Third Party Software Licenses

| Product Name       | Product Description and Download Site   | License Type    |
|--------------------|---|-----------------|
| jTidy              | <a href="http://jtidy.sourceforge.net/">http://jtidy.sourceforge.net/</a>                                 | MIT Open Source |
| Apache Commons Net | <a href="http://commons.apache.org/proper/commons-net/">http://commons.apache.org/proper/commons-net/</a> | Apache          |
|                    |   |                 |

## RELEASE NOTES FOR VERSION 2014 Q4

### ***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.

### ***New Resources***

`encoding/CIS_JCE_PROVIDERS_VIEW` – A wrapper view for `encoding/EncodingCJP/CISSecurityProviders()`.

`encoding/EncodingCJP/CISSecurityProviders` – Lists all JCE providers, services, and algorithms configured in the CIS JVM.

`encoding/EncodingCJP/DecryptFrom3DES` – Decrypts a symmetrical Triple DES encrypted string.

`encoding/EncodingCJP/DecryptWithCISPrivKey` – Decrypts a string encrypted using CIS's built in SSL certificate.

`encoding/EncodingCJP/EncryptWith3DES` – Encrypts a string using symmetrical Triple DES encryption.

`encoding/EncodingCJP/EncryptWithCISPrivKey` – Encrypts a string using CIS's built in SSL certificate.

`log/auditLogger` – Provides audit logging services to log files, databases, emails, and/or print to the Studio console.

`reintrospectCJPs` – Walks the tree of Utilities resources and reintrospects any CJP data sources it finds.

`repository/getResourceListChildren` – Returns information about the immediate children of a container resource.

`repository/removePathQuotes` – Removes the quote characters from a resource path string.

`repository/user/getGroup` – Returns the system rights of a group.

`repository/user/getUser` – Returns the system rights of a user.

`string/addQuotesInList` – Quotes the value in a CSV string.

`string/concatNotNull` – Concatenates two string values together according to a specified rule.

`string/getConstant` – Extracts the value of a constant defined in a procedure resource.

`string/modifyConstant` – Updates the value of a constant defined in a procedure resource.

`string/TextUtils/HexToRaw` – Converts a VARCHAR hexdecimal value to a VARBINARY value.

`string/TextUtils/RawToHex` – Converts a VARBINARY value to a VARCHAR hexdecimal value.

## ***Updated Resources***

`conversions/convertBit` – Modified to accept 'Y', 'y', 'N', and 'n' as arguments.

`repository/getResourceListRecursive` – Updated to use `repository/getResourceListChildren()` while recursing.

`repository/getResourcePrivileges` – Fixed an issue where result filtering was not being applied properly.

`repository/updateResourceCacheConfiguration` – Updated to better handle period and interval configuration.

`repository/updateTrigger` – Updated to handle configuration of all current trigger features.

`repository/user/createGroup` – Updated to support a group annotation.

`repository/user/createUser` – Updated to support a user annotation.

`repository/user/deleteGroup` – Fixed an issue where some groups would not be located in the repository.

`repository/user/deleteUser` – Fixed an issue where some users would not be located in the repository.

`time/DateUtils/GetServerTimezone` – Updated to support the XML timezone format.

## ***Deprecated Resources***

### **2014 Q4**

`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 Q3**

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

## **2014 Q1**

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

## **2012 Q4**

`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 Q1**

`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 Q3**

`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.

## ***Regression Test Versions***

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

6.2.6.00.76

## 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. **Calculations** – general calculations.
3. **Conversions** – general conversions.
4. **Documentation** – tools for documenting resources.
5. **Encoding** – encoding conversions.
6. **Examples** – example resources that illustrate the usage of many of the utilities.
7. **File** – full Create, Read, Update, and Delete for files plus many other useful file capabilities.
8. **Logging** – general-purpose logging and error notification.
9. **Net** – tools for accessing the network.
10. **PDTool** – tools for generating PDTool deployment plans.
11. **Repository** – general-purpose repository API interaction functions.
12. **Request** – functions for accessing request source code.
13. **Security** – functions for encrypting and decrypting text.
14. **String** – general-purpose string manipulation functions.
15. **Time** – general-purpose time manipulation functions.
16. **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 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.

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 CIS and not likely to ever be rolled into the CIS 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 **FRESH** installs of the Utilities where no other AS assets have been installed yet:

- Create a folder in /shared called “ASAssets”. Spelling and capitalization are important here so please use this exact spelling and case.
- Right click on the new ASAssets folder and select “Import ...”, choose the Utilities distribution CAR file in the resulting dialog, and click the “Import>” button.

For **EXISTING** installs of the Utilities where an ASAssets folder **IS** desired:

- 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/RepositoryDefinitionsRe  
cursive
- 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 the new ASAssets folder and select “Import …”, choose the Utilities distribution CAR file in the resulting dialog, check the “Overwrite” checkbox, and click the “Import>” button.
- Using the “Overwrite” option should only overwrite the Utilities folder and leave everything else in ASAssets intact.
- Execute the procedure /shared/ASAssets/reintrospectCJP to reintrospect all the CJP data sources in the Utilities distribution.

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

- Right click on the ASAssets folder and select “Import …”, choose the Utilities distribution CAR file in the resulting dialog, check the “Overwrite” checkbox, and click the “Import>” button.
- Using the “Overwrite” option should only overwrite the Utilities folder and leave everything else in “ASAssets” intact.
- Execute the procedure /shared/ASAssets/reintrospectCJP to reintrospect all the CJP data sources in the Utilities distribution.

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

- Rename PSAssets to ASAssets
- Right click on the ASAssets folder and select “Import …”, choose the Utilities distribution CAR file in the resulting dialog, check the “Overwrite” checkbox, and click the “Import>” button.
- Using the “Overwrite” option should only overwrite the Utilities folder and leave everything else in “ASAssets” intact.
- Execute the procedure /shared/ASAssets/reintrospectCJP to reintrospect all the CJP data sources in the Utilities distribution.

For **EXISTING** installs of the Utilities where no other AS assets have been installed yet and an ASAssets folder **IS NOT** desired:

- Right click on the /shared folder and select “Import …”, choose the Utilities distribution CAR file in the resulting dialog, check the “Overwrite” checkbox, and click the “Import>” button.
- Using the “Overwrite” option should only overwrite the Utilities folder and leave everything else in “Shared” intact.
- After import, it may be necessary to manually edit some of the Utilities and correct paths that still point to an “ASAssets” folder.
- Execute the procedure /shared/ASAssets/reintrospectCJPs to reintrospect all the CJP data sources in the Utilities distribution.

### 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.)

## 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() < 2014.4) THEN
    RAISE System.NotSupportedException
        VALUE '/shared/ASAssets/Utilities must be version 2014.4';
END IF;
```

#### **1. Parameters:**

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

#### **2. Examples:**

##### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| OUT       | result         | 2014.4          |

### **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            |

#### **2. Examples:**

## 2.1. Assumptions: none

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

## TypeDefinitions

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

## 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           |

#### **2. Examples:**

##### **2.1. Assumptions: none**

| Direction | Parameter Name   | Parameter Value     |
|-----------|------------------|---------------------|
| IN        | ADInt8           | 1292960160000000000 |
| IN        | GMTOffsetInHours | 6                   |
| OUT       | ADDate           | ‘2010-09-21’        |

##### **2.2. Assumptions: none**

| Direction | Parameter Name   | Parameter Value     |
|-----------|------------------|---------------------|
| IN        | ADInt8           | 1292960160000000000 |
| 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      |

### 2. Examples:

#### 2.1. 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         |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | Num            | 3567            |

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

## 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        |

#### **2. Examples:**

##### **2.1. Assumptions:**

**2.1.1. 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                                       |

#### **2. Examples:**

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | query          | 'SELECT FreightCharge FROM /shared/examples/ds_orders/orders ORDER BY FreightCharge' |
| OUT       | result         | 26.0   |

## 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            |

##### **2. Examples:**

###### **2.1. 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)   |

##### **2. Examples:**

###### **2.1. 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        |

### **2. Examples:**

#### **2.1. 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          |

### **2. Examples:**

#### **2.1. 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)   |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | request        | '0'             |
| OUT       | response       | 'N'             |

## 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        | <b>in_resourcePath</b> – The starting CIS path for which to introspect resources and generate documentation.   | pathType       |
| IN        | <b>in_resourceType</b> – The starting CIS resource type for the resource path.   | VARCHAR        |
| IN        | <b>in_filePath</b> – The full file system path to generate the documentation file to.  | pathType       |
| IN        | <b>in_docPreambleImpl</b> – The CIS path to the procedure that represents the preamble of the documentation. If left null, the default “geDocPreambleImpl1” is used.   | pathType       |
| IN        | <b>in_docResourceFormatImpl</b> – The CIS path to the procedure that performs the documentation formatting. If left null, the default “getDocResourceFormatImpl1” is used.   | pathType       |
| IN        | <b>in_constantPath</b> – This is the path to the constants file  | pathType       |
| IN        | <b>in_switches</b> – Provides guidance on what documentation to print. The format is<br>print_switch=[option1 {default_option2} option3]<br><br>This is a space separate list with no spaces before or after the equal sign.<br><br>Print_containers=[{none} all] – print the resource container (folder)<br><br>print_annotations=[none {all} nonblank blank] – print all annotations whether they are blank or not | LONGVARCHAR    |

| Direction | Parameter Name   | Parameter Type |
|-----------|--|----------------|
|           | <p>print_resource_projections=[none {all}] – print the resource projections</p> <p>print_resources_used=[none {all}] – print the immediate child resources used by the parent resource</p> <p>print_datasource_accessed=[none {all}] – print the data source accessed list</p> <p>print_datasource_lineage=[none {all}] – print the data source lineage</p> <p>print_time=[{no} yes] – print the time it takes to retrieve the full documentation for each resource and the final time</p> <p>save_file=[{no} yes] – save the results to a file</p> <p>save_file_intermediate=[{no} yes] – save the file intermediately after each resource is completed</p> <p>Example:</p> <ol style="list-style-type: none"> <li>1) switches: when left blank or null then the defaults are taken<br/>result: all documentation modules are printed</li> <li>2) switches: print_annotations=nonblank<br/>print_resource_projections=none<br/>print_resources_used=none<br/>print_datasource_lineage=none<br/>result: only non-blank annotations are printed and nothing else</li> </ol> |                |
| IN        | <b>in_excludeKeywordsInPathList</b> – 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        | <b>in_excludePathsList</b> – 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    |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name  | Parameter Value  |
|-----------|-----------------|------------------|
| IN        | in_resourcePath | /shared/examples |

| Direction | Parameter Name                | Parameter Value  |
|-----------|-------------------------------|--|
| 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<br>save_file_intermediate=yes     |
| IN        | in_excludeKeywordsInPathsList | 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        | <b>resourcePath</b> – The starting CIS path for which to introspect resources and generate documentation.   | pathType       |
| IN        | <b>resourceType</b> – The starting CIS resource type for the resource path.   | VARCHAR        |
| IN        | <b>docPreambleImpl</b> – The CIS path to the procedure that represents the preamble of the documentation. If left null, the default "geDocPreambleImpl1" is used.   | pathType       |
| IN        | <b>docResourceFormatImpl</b> – The CIS path to the procedure that performs the documentation formatting. If left null, the default "getDocResourceFormatImpl1" is used.   | pathType       |
| IN        | <b>constantPath</b> – This is the path to the constants file  | pathType       |
| IN        | <b>switches</b> – 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.<br><br>print_containers=[{none} all] - print the resource | LONGVARCHAR    |

| Direction | Parameter Name   | Parameter Type |
|-----------|--|----------------|
|           | <p>container (folder)</p> <p>print_annotations=[none {all} nonblank blank] - print all annotations whether they are blank or not</p> <p>print_resource_projections=[none {all}] - print the resource projections</p> <p>print_resources_used=[none {all}] - print the immediate child resources used by the parent resource</p> <p>print_datasource_accessed=[none {all}] - print the data source accessed list</p> <p>print_datasource_lineage=[none {all}] - print the data source lineage</p> <p>print_time=[{no} yes] - print the time it takes to retrieve the full documentation for each resource and the final time</p> <p>save_file=[{no} yes] - save the results to a file</p> <p>save_file_intermediate=[{no} yes] - save the file intermediately after each resource is completed</p> <p>Example:</p> <p>3) switches: when left blank or null then the defaults are taken<br/>result: all documentation modules are printed</p> <p>2) switches: print_annotations=nonblank<br/>print_resource_projections=none<br/>print_resources_used=none<br/>print_datasource_lineage=none<br/>result: only non-blank annotations are printed and nothing else</p> |                |
| IN        | <b>excludeKeywordsInPathList</b> – 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        | <b>excludePathsList</b> – 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        | <b>filePath</b> – The full file system path to generate the documentation file to.   | pathType       |

| Direction | Parameter Name  | Parameter Type        |
|-----------|---|-----------------------|
| OUT       | <b>formattedText</b> – The formatted text is returned for all documentation | /lib/util/System.Text |

## 2. Examples:

### 2.1. 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<br>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 Documentation

Generated on 2012-11-11 08:10:34.614

---

=====  
 Resource Name: CompositeView  
 Resource Path: /shared/examples/CompositeView  
 Resource Type: TABLE  
 SubType: SQL\_TABLE

Description:

-----  
 None

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 |

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     | 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
```

Resource 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=yes

```
print_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): 0
Number of Folders (CONTAINER): 0
Number of Views (TABLE): 15
Number of Procedures (PROCEDURE): 3
Number of Data Sources (DATA_SOURCE): 3
Number of XML (TREE): 1
Number of Triggers (TRIGGER): 0
Number of Connectors (CONNECTOR): 0
Number of Def. Sets (DEFINITION_SET): 1
Number 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 |
|-----------|----------------|----------------|
|-----------|----------------|----------------|

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

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name  | Parameter Value   |
|-----------|-----------------|---|
| CONSTANT  | docPreambleImpl | /shared/ASAssets/Utilities/documentation/impl<br>ementations/geDocPreambleImpl1 |

| Direction | Parameter Name            | Parameter Value  |
|-----------|---------------------------|--|
| CONSTANT  | docResourceFormatImpl     | /shared/ASAssets/Utilities/documentation/impl<br>ementations/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<br>,/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        | <b>Procedure Path</b> – The path to the documentation driver procedure.                                   | pathType       |
| IN        | <b>Parameter Values</b> – refer to the procedure “getDocumentationDriver” for details on what to pass in. | VARCHAR        |

## 2. Examples:

### 2.1. 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<br>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        | <b>constantPath</b> – The path to the constants file ending with procedure parenthesis () . | pathType       |
| IN        | <b>constantName</b> – The name of the constant.   | VARCHAR        |
| OUT       | <b>outValue</b> – The value of the constant   | LONGVARCHAR    |

#### 2. Examples:

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

## 2. Examples:

### 2.1. 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        | <b>switches</b> – A space separated list of switches with no embedded spaces. Definitions:<br><b>print_containers</b> = [ {none} all] - print the resource container (folder)<br><b>print_annotations</b> = [none {all} nonblank blank] - print all annotations whether they are blank or not<br><b>print_resource_projections</b> = [none {all}] - print the resource projections<br><b>print_resources_used</b> = [none {all}] - print the immediate child resources used by the parent resource<br><b>print_datasource_accessed</b> = [none {all}] - print the data source accessed list<br><b>print_datasource_lineage</b> = [none {all}] - print the data source lineage<br><b>print_time</b> = [ {no} yes] - print the time it takes to retrieve the full documentation for each resource and the final time<br><b>save_file</b> = [ {no} yes] - save the results to a file<br><b>save_file_intermediate</b> = [ {no} yes] - save the file intermediately after each resource is completed | LONGVARCHAR    |
| IN        | <b>command</b> – One of the following:<br><b>print_containers</b>   <b>print_annotations</b>  <br><b>print_resource_projections</b>   <b>print_resources_used</b>  <br><b>print_datasource_accessed</b>   <b>print_datasource_lineage</b>  <br><b>print_time</b>   <b>save_file</b>   <b>save_file_intermediate</b>  | VARCHAR        |
| OUT       | <b>commandOptionValue</b> – The result command value [0,1,2,3] – none no=0, all yes=1, nonblank=2, blank=3   | INTEGER        |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value   |
|-----------|--------------------|---|
| IN        | switches           | <b>print_time=yes</b> <b>save_file=yes</b><br><b>save_file_intermediate=yes</b> |
| IN        | command            | <b>print_time</b>   |
| 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        | <b>resourcePath</b> – (optional) Full resource path which includes the path and the resource name .   | pathType                                  |
| IN        | <b>resourceType</b> – The starting CIS resource type for the resource path.                           | VARCHAR                                   |
| IN        | <b>constantPath</b> – This is the path to the constants file.   | pathType                                  |
| OUT       | <b>formattedText</b> – formatted text is out output complete with a separator at the beginning of the | PIPE (formattedText /lib/util/System.Tex) |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
|           | resource.      |                |

## 2. Examples:

### 2.1. 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        | <b>startingResourcePath</b> – The starting folder path from which to start formatting the documentation. It is always of type CONTAINER. | pathType       |
| IN        | <b>startingResourceType</b> – The starting CIS resource type for the resource path.  | VARCHAR        |
| IN        | <b>constantPath</b> – This is the path to the constants file.  | pathType       |
| IN        | <b>switches</b> – Provides guidance on what to print for the documentation and save files.   | LONGVARCHAR    |
| IN        | <b>excludeKeywordsInPathList</b> – Provides guidance on what to print documentation for  | LONGVARCHAR    |

| Direction | Parameter Name  | Parameter Type                             |
|-----------|---|--|
| IN        | <b>excludePathsList</b> – Exclude actual paths. Double quotes are not required. Comma separated list.           | LONGVARCHAR                                |
| IN        | <b>filePath</b> – The full file system path to generate the documentation file to.                              | pathType                                   |
| OUT       | <b>formattedText</b> – formatted text is out output complete with a separator at the beginning of the resource. | PIPE (formattedText /lib/util/System.Text) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name            | Parameter Value  |
|-----------|---------------------------|--|
| IN        | startingResourcePath      | /shared/examples   |
| IN        | startingResourceType      | CONTAINER  |
| IN        | constantPath              | /shared/ASAssets/Utilities/documentation/consts()          |
| IN        | switches                  | print_time=yes save_file=yes<br>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.

## 3. Parameters:

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
|-----------|----------------|----------------|

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

#### 4. Examples:

##### 4.1. Assumptions: none

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | resourcePath   | /shared/examples/CompositeView                             |
| IN        | resourceType   | TABLE  |
| IN        | constantPath   | /shared/ASAssets/Utilities/documentation/constan<br>ts()   |
| IN        | switches       | print_time=yes save_file=yes<br>save_file_intermediate=yes |
| OUT       | formattedText  | See below:   |

```
=====
Resource Name: CompositeView
Resource Path: /shared/examples/CompositeView
Resource Type: TABLE
    Subtype: SQL_TABLE

Description:
-----
None

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 |

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] |       |       |       |   |
| 4 -                    | 20679 | 20658 | 2     | [DS] /shared/examples/ds_orders<br>[CS] /shared/examples/ds_orders/orderdetails         |
| [TABLE.DATABASE_TABLE] |       |       |       |   |
| 5 -                    | 20711 | 20658 | 2     | [DS] /shared/examples/ds_orders<br>[CS] /shared/examples/ds_orders/orders               |
| [TABLE.DATABASE_TABLE] |       |       |       |   |
| 6 -                    | 20670 | 20711 | 3     | [DS] /shared/examples/ds_orders<br>/shared/examples/ds_orders                           |
| 7 -                    | 20729 | 20670 | 4     | [CS] /shared/examples/ds_orders/cache_status  |
| [TABLE.DATABASE_TABLE] |       |       |       |   |
| 8 -                    | 20671 | 20670 | 4     | [DS] /shared/examples/ds_orders<br>[CS] /shared/examples/ds_orders/cache_tracking       |
| [TABLE.DATABASE_TABLE] |       |       |       |   |
| 9 -                    | 20689 | 20711 | 3     | [DS] /shared/examples/ds_orders<br>[CS] /shared/examples/ds_orders/orders_cache         |
| [TABLE.DATABASE_TABLE] |       |       |       |   |
| 10 -                   | 20774 | 20587 | 1     | /shared/examples/ViewSales  |
| 11 -                   | 20679 | 20774 | 2     | [CS] /shared/examples/ds_orders/orderdetails  |
| [TABLE.DATABASE_TABLE] |       |       |       |   |
| 12 -                   | 20786 | 20774 | 2     | [DS] /shared/examples/ds_orders<br>/shared/examples/productCatalog_Transformation       |
| 13 -                   | 20757 | 20786 | 3     | [CS] /shared/examples/ds_XML/productCatalog.xml   |
| [TREE.XML_FILE_TREE]   |       |       |       |   |
| 14 -                   | 20763 | 20587 | 1     | /shared/examples/ViewSupplier   |
| 15 -                   | 20606 | 20763 | 2     | [CS] /shared/examples/ds_inventory/inventorytransactions                                |
| [TABLE.DATABASE_TABLE] |       |       |       |   |
| 16 -                   | 20619 | 20763 | 2     | [DS] /shared/examples/ds_inventory<br>[CS] /shared/examples/ds_inventory/purchaseorders |

```

[TABLE.DATABASE_TABLE]
17 - 20644 20763      2
[TABLE.DATABASE_TABLE]           [DS] /shared/examples/ds_inventory
                                [CS] /shared/examples/ds_inventory/suppliers
                                [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]

9 - 20689 20711      3      [DS] /shared/examples/ds_orders
[TABLE.DATABASE_TABLE]      [CS] /shared/examples/ds_orders/orders_cache
                           [DS] /shared/examples/ds_orders

```

### Parameters:

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

### 1. Examples:

#### 1.1. Assumptions: none

| Direction | Parameter Name               | Parameter Value                                   |
|-----------|------------------------------|---|
| IN        | resourcePath                 | /shared/examples/CompositeView                    |
| IN        | resourceType                 | TABLE   |
| IN        | constantPath                 | /shared/ASAssets/Utilities/documentation/consts() |
| 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                |
|                        |         |       |       | [DS] /shared/examples/ds_orders                          |
| [TABLE.DATABASE_TABLE] |         |       |       | [CS] /shared/examples/ds_orders/orderdetails             |
| 4                      | - 20679 | 20658 | 2     | [DS] /shared/examples/ds_orders                          |
| [TABLE.DATABASE_TABLE] |         |       |       | [CS] /shared/examples/ds_orders/orders                   |
| 5                      | - 20711 | 20658 | 2     | [DS] /shared/examples/ds_orders                          |
| [TABLE.DATABASE_TABLE] |         |       |       | [CS] /shared/examples/ds_orders/cache_status             |
| 6                      | - 20670 | 20711 | 3     | /shared/examples/ds_orders                               |
| 7                      | - 20729 | 20670 | 4     | [CS] /shared/examples/ds_orders/cache_tracking           |
| [TABLE.DATABASE_TABLE] |         |       |       | [DS] /shared/examples/ds_orders                          |
| 8                      | - 20671 | 20670 | 4     | [CS] /shared/examples/ds_orders/orders_cache             |
| [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        | <b>resourcePath</b> – Full resource path which includes the path and the resource name .                        | pathType              |
| IN        | <b>resourceType</b> – Type of CIS resource to be created  | VARCHAR               |
| IN        | <b>constantPath</b> – This is the path to the constants file.   | pathType              |
| OUT       | <b>formattedText</b> – formatted text is out output complete with a separator at the beginning of the resource. | /lib/util/System.Text |

## 2. Examples:

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

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
|           | resource.      |                |

## 2. Examples:

### 2.1. 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 |

## 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 (<br>Provider<br>Algorithm<br>"Service Description"<br>) |

#### **2. Examples:**

##### **2.1. Assumptions: none**

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

### ***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) |

#### **2. Examples:**

## 2.1. 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) |

#### 2. Examples:

## 2.1. 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 consumers, app servers, ESBs, etc.) when these failures are caused by unsupported security algorithms.

#### 1. Parameters:

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

#### 2. Examples:

## 2.1. Assumptions: none

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

## 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    |

### 2. Examples:

#### 2.1. 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/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    |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name         | Parameter Value                           |
|-----------|------------------------|---|
| IN        | "encrypted hex string" | '1232F6C78E9CE0D3EEB9CE05463942458C7689F' |

| Direction | Parameter Name      | Parameter Value   |
|-----------|---------------------|---|
|           |                     | 19CF45664B9BA8FFADDBFBE075880BAE9CE<br>BA93FF560368D96896E28E100FC79604CC75175<br>E4286B3B3D4BFDA6EA5A32E0568ECFB4C745D<br>8FC8A58CFCBF3BBE1C00BC55A97E23C7571705<br>2BFE51131E38D8504FB35E8393C277E7BEF9E5E<br>36CDEF5D19F6769CA673F2AD65EE' |
| 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    |

### 2. Examples:

#### 2.1. 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/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    |

### 2. Examples:

## 2.1. 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" | '1232F6C78E9CE0D3EEB9CE05463942458C7689F1<br>9CF45664B9BA8FFADDDBFBE075880BAE9CEBA<br>93FF560368D96896E28E100FC79604CC75175E428<br>6B3B3D4BFDA6EA5A32E0568ECFB4C745D8FC8<br>A58CFCBF3BBE1C00BC55A97E23C75717052BFE<br>51131E38D8504FB35E8393C277E7BEF9E5E36CD<br>EF5D19F6769CA673F2AD65EE' |

## 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) |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                   |
|-----------|----------------|-----------------------------------|
| IN        | inputString    | 'foo'                             |
| OUT       | result         | 'acbd18db4cc2f85cedef654fcc4a4d8' |

## 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) |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                            |
|-----------|----------------|--|
| IN        | inputString    | 'foo'                                      |
| OUT       | result         | '0beec7b5ea3f0fdbc95d0dd47f3c5bc275da8a33' |

## 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             |

#### **2. Examples:**

##### **2.1. 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)  |

## 2. Examples:

### 2.1. 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)        |

## 2. Examples:

### 2.1. 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             |

## 2. Examples:

### 2.1. 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 |                     |

#### **2. Examples:**

##### **2.1. 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 |                     |

#### **2. Examples:**

##### **2.1. Assumptions: none**

| Direction | Parameter Name        | Parameter Value             |
|-----------|-----------------------|-----------------------------|
| IN        | filePath              | '/files/incoming/file1.txt' |
| OUT       | archivalDirectoryPath | '/files/archive'            |

## FileProcessingCJP/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 |                     |

### 2. Examples:

#### 2.1. 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                              | SMALLINT            |
|           | 0=do not append file, 1=append file |                     |
| IN        | fileContent                         | LONGVARCHAR         |
| OUT       | None: Throws exception upon failure |                     |

### 2. Examples:

#### 2.1. 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<br>0=do not append file, 1=append file | SMALLINT            |
| IN        | fileContent                                   | LONGVARBINARY       |
| OUT       | None: Throws exception upon failure           |                     |

### 2. Examples:

#### 2.1. 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             |

### 2. Examples:

#### 2.1. 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             |

## 2. Examples:

### 2.1. 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         |

## 2. Examples:

### 2.1. 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       |

## 2. Examples:

### 2.1. 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            |

### 2. Examples:

#### 2.1. 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           |

### 2. Examples:

#### 2.1. 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             |

### 2. Examples:

#### 2.1. 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 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             |

### 2. Examples:

#### 2.1. 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<br>Y=remove all files and directories.<br>N=remove only files and leave the directory structure in place. | VARCHAR             |
| OUT       | success (1=true, 0=false)  | BOOLEAN             |

### 2. Examples:

#### 2.1. 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             |

### **2. Examples:**

#### **2.1. 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.

### **3. Parameters:**

| Direction | Parameter Name            | Parameter Type      |
|-----------|---------------------------|---------------------|
| IN        | filePath                  | VARCHAR(2147483647) |
| OUT       | success (1=true, 0=false) | BOOLEAN             |

### **4. Examples:**

#### **4.1. Assumptions: none**

| Direction | Parameter Name            | Parameter Value             |
|-----------|---------------------------|-----------------------------|
| IN        | filePath                  | '/files/incoming/files.zip' |
| OUT       | success (1=true, 0=false) | 1                           |

## HOW TO USE 'LOGGING' PROCEDURES

### ***Introduction***

This section will show how to use the 'Logging' procedures.

#### **auditLogger**

Provides the developer with a way in which to programmatically send error messages to a File, Email, Database and/or Studio console.

The "attributeVect" parameter governs how the procedure will deal with messages. An example of declaring a variable to contain the attribute vector:

```
DECLARE attributeVect  
    VECTOR(/shared/ASAssets/Utilities/log/auditLogger.AttributeType  
) ;
```

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): <ul style="list-style-type: none"><li>• LOG - Write to the &lt;CIS_HOME&gt;/logs/cs_server.log</li><li>• EMAIL - Send an email (Email settings must be configured on the CIS instance in the Configuration panel.)</li><li>• DB - Insert the message into the AUDIT_LOG table. (created and introspected by developer during initialization)</li><li>• PRINT - Print to the Studio console tab</li></ul> |
| notificationType | The type of notification that is being logged. One and only one of: <ul style="list-style-type: none"><li>• ERROR - Output error message with severity level ERROR.</li><li>• AUDIT - Output audit message with severity level INFO.</li><li>• INFO - Output info message with severity level INFO.</li><li>• DEBUG - Output debug message with severity level INFO.</li></ul>  |
| 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.  |

| Attribute        | Description  |
|------------------|--|
|                  | 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        | attributeVect  | VECTOR[(VARCHAR(255), LONGVARCHAR)] |

## 1. Examples:

### 1.1. Assumptions: none

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | messageText    | 'This is a test message'   |
| IN        | attributeVect  | [<br>('loggingType', 'LOG, DATABASE'),<br>('notificationType', 'DEBUG'), |

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
|           |                | ('auditTablePath', '/shared/myDB/audit_tab'), ('moduleName', '/shared/myTestProcedure') } |

## 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<br>The name of the invoking procedure  | /shared/ASAssets/Utilities/TypeDefinitions.moduleNameType |
| IN        | debug<br>Defines the debugging options for this procedure.<br>Values: Y or T = debugging turned on, N or F = debugging turned off | CHAR(1)   |
| IN        | messageText<br>The message to be sent or stored   | /shared/ASAssets/Utilities/TypeDefinitions.messageType    |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| IN        | moduleName     | Any procedure name |
| IN        | debug          | Y/N or T/F         |
| IN        | messageText    | Message text       |

## 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        |

##### **2. Examples:**

###### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value             |
|-----------|----------------|-----------------------------|
| IN        | fileName       | 'test.xml'                  |
| IN        | hostIp         | 'ftp.cisco.com'             |
| IN        | userId         | 'anonymous'                 |
| IN        | userPass       | 'cgoodric@cisco.com'        |
| IN        | ftpDirName     | 'pub'                       |
| IN        | dirName        | 'C:\Users\cgoodric\Desktop' |
| OUT       | success        | true                        |

## 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  $\geq$  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 (Composite Professional Services Promotion and Deployment Tool) is used to execute generated plan files. If you do not have PDTool on site, you may obtain it through a Composite/Cisco Professional Services engagement.

#### **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        |

#### **2. Examples:**

##### **2.1. Assumptions: none**

| <b>Direction</b> | <b>Parameter Name</b>     | <b>Parameter Value</b>   |
|------------------|---------------------------|--|
| IN               | planFilePath              | 'D:/PDTool62/resources/plans/examples.dp'  |
| IN               | bufferSize                | 100  |
| IN               | PDToolPlanTemplateVCS     | 'PASS TRUE ExecuteAction vcsCheckout2<br>\$SERVERID \$VCONN "\$RESOURCE_PATH"<br>"\$RESOURCE_TYPE" HEAD<br>"\$MODULE_HOME/VCSModule.xml"<br>"\$MODULE_HOME/servers.xml"' |
| IN               | PDToolPlanTemplateFolders | 'PASS TRUE ExecuteAction createFolder<br>\$SERVERID "\$RESOURCE_PATH"<br>"\$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 (Composite Professional Services Promotion and Deployment Tool) is used to execute generated plan files. If you do not have PDTool on site, you may obtain it through a Composite/Cisco Professional Services engagement.

#### **1. Parameters:**

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Type</b> |
|------------------|-----------------------|-----------------------|
| IN               | planFilePath          | VARCHAR(1024)         |

| Direction | Parameter Name               | Parameter Type |
|-----------|------------------------------|----------------|
| IN        | bufferSize                   | INTEGER        |
| IN        | PDTToolPlanTemplateVCS       | LONGVARCHAR    |
| IN        | PDTToolPlanTemplateFolders   | LONGVARCHAR    |
| IN        | resourcePathList             | LONGVARCHAR    |
| IN        | includeDependentTriggers     | INTEGER        |
| IN        | includebasePath              | LONGVARCHAR    |
| IN        | excludePathsList             | LONGVARCHAR    |
| IN        | inIgnoreResourceDoesNotExist | INTEGER        |
| IN        | debug                        | CHAR(1)        |
| OUT       | success                      | INTEGER        |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name               | Parameter Value  |
|-----------|------------------------------|--|
| IN        | planFilePath                 | 'C:\PDTTool62\resources\plans'   |
| IN        | bufferSize                   | 100  |
| IN        | PDTToolPlanTemplateVCS       | 'PASS TRUE ExecuteAction vcsCheckout2<br>\$SERVERID \$VCONN "\$RESOURCE_PATH"<br>"\$RESOURCE_TYPE" HEAD<br>"\$MODULE_HOME/VCSModule.xml"<br>"\$MODULE_HOME/servers.xml"' |
| IN        | PDTToolPlanTemplateFolders   | 'PASS TRUE ExecuteAction createFolder<br>\$SERVERID "\$RESOURCE_PATH"<br>"\$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        |

### **2. Examples:**

#### **2.1. 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 (Composite Professional Services Promotion and Deployment Tool) is used to execute generated plan files. If do not have PDTool on site, you may obtain it through a Composite/Cisco Professional Services engagement.

### **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         |

## 2. Examples:

### 2.1. 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.

PDTTool (Composite Professional Services Promotion and Deployment Tool) is used to execute generated plan files. If do not have PDTTool on site, you may obtain it through a Composite/Cisco Professional Services engagement.

## 1. Parameters:

| Direction | Parameter Name           | Parameter Type |
|-----------|--------------------------|----------------|
| IN        | resourcePathList         | LONGVARCHAR    |
| IN        | includeDependentTriggers | INTEGER        |
| IN        | includebasePath          | LONGVARCHAR    |
| IN        | excludePathsList         | LONGVARCHAR    |
| IN        | debug                    | INTEGER        |

| Direction | Parameter Name               | Parameter Type |
|-----------|------------------------------|----------------|
| IN        | inIgnoreResourceDoesNotExist | INTEGER        |
| OUT       | resourceTreeList             | CURSOR         |

## 2. Examples:

### 2.1. 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 (Composite Professional Services Promotion and Deployment Tool) is used to execute generated plan files. If do not have PDTool on site, you may obtain it through a Composite/Cisco Professional Services engagement.

## 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         |

## 2. Examples:

### 2.1. 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 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 (Composite Professional Services Promotion and Deployment Tool) is used to execute generated plan files. If do not have PDTool on site, you may obtain it through a Composite/Cisco Professional Services engagement.

## 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         |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|           |                |                 |

| <b>Direction</b> | <b>Parameter Name</b>        | <b>Parameter Value</b> |
|------------------|------------------------------|------------------------|
| IN               | resourcePathList             | '/shared/examples'     |
| IN               | includeDependentTriggers     | 1                      |
| IN               | includebasePath              | NULL                   |
| IN               | excludePathsList             | NULL                   |
| IN               | inIgnoreResourceDoesNotExist | 1                      |
| IN               | debug                        | 0                      |
| OUT              | resourceTreeList             | <row set>              |

## 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.

### **addRemoveDataSourceChildren**

This procedure performs the same function as the "Add/Remove Resources ..." context menu item for a data source. It takes as input a data source path (i.e. '/shared/examples/ds\_orders') and a list of child resources (the child path specification is an absolute path that begins at the data source container) and/or containers and their introspection state (one of 'SELF', 'SELF\_AND\_CHILDREN', or 'IGNORED'.) This procedure has been deprecated in CIS 6.2. See [introspectResourcesTask](#) below.

#### 1. Parameters:

| Direction | Parameter Name   | Parameter Type  |
|-----------|------------------|---|
| IN        | inDataSourcePath | /lib/resource/ResourceDefs.ResourcePath                         |
| IN        | inChildInfos     | /shared/ASAssets/Utilities/TypeDefinitions.ChildInfosVectorType |
| OUT       | success          | BIT   |
| OUT       | faultResponse    | XML   |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value  |
|-----------|------------------|--|
| IN        | inDataSourcePath | '/shared/examples/ds_orders'   |
| IN        | inChildInfos     | VECTOR [('/customers', 'TABLE', 'IGNORED'), ('/orderdetails', 'TABLE', 'IGNORED')] |
| OUT       | success          | 1  |
| OUT       | faultResponse    | [NULL]   |

## **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 (<br>operation     VARCHAR(255)<br>prevStatus    VARCHAR(255)<br>currStatus    VARCHAR(255)<br>resourceType    ResourceType<br>resourcePath    ResourcePath<br>) |

## 2. Examples:

### 2.1. 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          | (<br>'R',<br>'ENABLED',<br>'ENABLED',<br>'TABLE',<br>'/shared/examples/ds_orders/orders'<br>) |

## 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 confirmPassword 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)   |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value                  |
|-----------|--------------------|----------------------------------|
| IN        | oldPassword        | 'Old p4ssw0rd'                   |
| IN        | newPassword        | 'n3w p4ssw0rd'                   |
| IN        | confirmNewPassword | 'n3w p4ssw0rd'                   |
| OUT       | result             | 'Password successfully updated.' |

## 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        |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name    | Parameter Value |
|-----------|-------------------|-----------------|
| IN        | baseCisVersion    | '6.1.0.01.09'   |
| IN        | currentCisVersion | '6.1.0.01.14'   |

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| OUT       | status         | 1               |

## 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   |

### 2. Examples:

#### 2.1. 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 |

| Direction | Parameter Name | Parameter Type                         |
|-----------|----------------|--|
| IN        | inDestType     | /lib/resource/ResourceDfs.ResourceType |

## 2. Examples:

### 2.1. 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

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<br>0 (false) or null - only update the given resource and not the children.<br>1 (true) - update children recursively   | BIT  |
| IN        | copyPrivilegeMode<br>null (default) - do not set any privileges at all<br>0 - set mode to "OVERWRITE_APPEND" - merges and does not update privileges for users or groups not mentioned.<br>1 - set the mode to "SET_EXACTLY" - makes privileges look exactly like those provided in the call. | INTEGER  |
| IN        | copyPrivilegeEntry  | RepositoryDefinitions.copyPrivilegeEntryVector |
| OUT       | success   | BIT  |
| OUT       | faultResponse   | XML  |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value  |
|-----------|--------------------|--|
| IN        | updateRecursively  | 1  |
| IN        | copyPrivilegeMode  | 0  |
| IN        | copyPrivilegeEntry | VECTOR[ {('shared/test00','CONTAINER')}, {('shared/test01','CONTAINER')} ] |
| 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   |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| OUT       | faultResponse  | XML            |

## 2. Examples:

### 2.1. 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 are considered children and must be created.

## 1. Parameters:

| Direction | Parameter Name              | Parameter Type  |
|-----------|-----------------------------|---|
| IN        | sourceFolderPath            | /shared/ASAssets/Utilities/TypeDefinitions.pathType   |
| IN        | updatePrivilegesRecursively | BIT<br>0 (false) or null - only update privileges for the given resource and not the children.<br>1 (true) - update privileges for children recursively   |
| IN        | copyPrivilegeMode           | BIT<br>null (default) - do not set any privileges at all<br>0 - set mode to "OVERWRITE_APPEND" - merges and does not update privileges for users or groups not mentioned.<br>1 - set the mode to "SET_EXACTLY" - makes privileges look exactly like |

| Direction | Parameter Name              | Parameter Type |
|-----------|-----------------------------|----------------|
|           | those provided in the call. |                |
| OUT       | success                     | BIT            |
| OUT       | faultResponse               | XML            |

## 2. Examples:

### 2.1. 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                    |

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Type</b> |
|------------------|-----------------------|-----------------------|
| IN               | poolTimeout           | INTEGER               |

## 2. Examples:

### 2.1. Assumptions: none

| <b>Direction</b> | <b>Parameter Name</b>  | <b>Parameter Value</b>        |
|------------------|------------------------|-------------------------------|
| 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                           |

### 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.

| <b>Data Source Type</b> | <b>Valid Location</b>  | <b>Notes</b> |
|-------------------------|------------------------|--------------|
| 'COMPOSITE_DATABASE'    | /services/Databases    |              |
| 'COMPOSITE_SERVICE'     | /services/Web Services |              |
| 'Custom Java Procedure' | /shared/...            |              |
| 'DB2 v7 (Type 2)'       | /shared/...            |              |

| <b>Data Source Type</b>      | <b>Valid Location</b> | <b>Notes</b>  |
|------------------------------|-----------------------|---|
| '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:<br>java.lang.String cannot be cast to<br>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:<br>java.lang.String cannot be cast to<br>java.lang.Integer |
| 'Netezza 4.5'                | /shared/...           | This one causes a problem when used:<br>java.lang.String cannot be cast to<br>java.lang.Integer |

| Data Source Type                 | Valid Location | Notes   |
|----------------------------------|----------------|---|
| 'Netezza 5.0'                    | /shared/...    | This one causes a problem when used:<br>java.lang.String cannot be cast to<br>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                                     |

## 2. Examples:

### 2.1. 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/TypeDefinations.pathType |
| OUT       | success          | BIT   |
| OUT       | faultResponse    | XML   |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|-----------|----------------|-----------------|

| 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                    |

### 2. Examples:

#### 2.1. 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>' |

| <b>Direction</b> | <b>Parameter Name</b>  | <b>Parameter Value</b>        |
|------------------|------------------------|-------------------------------|
| 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:**

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Type</b>                               |
|------------------|-----------------------|---|
| IN               | resourcePath          | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN               | resourceName          | VARCHAR(255)  |
| IN               | resourceType          | VARCHAR(255)  |

| Direction | Parameter Name  | Parameter Type |
|-----------|-----------------|----------------|
| IN        | resourceSubType | VARCHAR(255)   |
| OUT       | success         | BIT            |
| OUT       | createResponse  | XML            |
| OUT       | faultResponse   | XML            |

## 2. Examples:

### 2.1. 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<br><br>C=Copy resource and leave original,<br>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   |

## 2. Examples:

## 2.1. 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                                     |

### 2. Examples:

#### 2.1. 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' |

| Direction | Parameter Name             | Parameter Value                           |
|-----------|----------------------------|---|
| 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' |

### 2. Examples:

#### 2.1. 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)               |

### 2. Examples:

#### 2.1. 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   |

## 2. Examples:

### 2.1. 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   |

## 2. Examples:

### 2.1. 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        |

## 2. Examples:

## 2.1. Assumptions: none

| Direction | Parameter Name      | Parameter Value |
|-----------|---------------------|-----------------|
| IN        | outputDirectory     | 'C:\Tmp'        |
| IN        | outputFileExtension | '.sql'          |
| OUT       | result              | 'SUCCESS'       |

## 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 |

### 2. Examples:

#### 2.1. 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"' |

## 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),                  |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
|           |                | resourcePath VARCHAR(1024),<br>resourceType VARCHAR(255),<br>subtype VARCHAR(255),<br>enabled BIT,<br>isNullable VARCHAR(255),<br>columnName VARCHAR(255),<br>columnType VARCHAR(255),<br>nativeBaseType VARCHAR(255),<br>nativeType VARCHAR(255) ) |

## 2. Examples:

### 2.1. 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 ( <table> <tr> <td>name</td> <td>VARCHAR,</td> </tr> <tr> <td>resPath</td> <td>TypeDefinitions.pathType,</td> </tr> <tr> <td>resType</td> <td>VARCHAR,</td> </tr> <tr> <td>subType</td> <td>VARCHAR)</td> </tr> </table> | name | VARCHAR, | resPath | TypeDefinitions.pathType, | resType | VARCHAR, | subType | VARCHAR) |
| name      | VARCHAR,                  |   |      |          |         |                           |         |          |         |          |
| resPath   | TypeDefinitions.pathType, |   |      |          |         |                           |         |          |         |          |
| resType   | VARCHAR,                  |   |      |          |         |                           |         |          |         |          |
| subType   | VARCHAR)                  |   |      |          |         |                           |         |          |         |          |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value    |
|-----------|------------------|--------------------|
| IN        | resourcePath     | '/shared/examples' |
| OUT       | resourceTreeList | See chart below    |

### 2.2. 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        | <b>resourcePath</b> - CIS source path to an actual resource                                   | pathType   |
| IN        | <b>resourceType</b> - Type of CIS resource to be created. It is null on the first invocation. | VARCHAR  |
| OUT       | result  | CURSOR lineageTreeType ( resourceName VARCHAR(255), resourcePath pathType, resourceType VARCHAR(255), subtype VARCHAR(255), enabled BIT, id INTEGER, changeid INTEGER, ownerDomain VARCHAR(255), ownerName VARCHAR(255), impactLevel VARCHAR(255), childCount INTEGER ); |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                      |
|-----------|----------------|--------------------------------------|
| IN        | resourcePath   | /shared/examples/ds_orders/customers |
| IN        | resourceType   | TABLE                                |
| OUT       | result         | See charts below                     |

#### 2.2. 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 |

### 2.3. 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          |

### 2.4. 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 |

### 2.5. Chart 4: Columns (16-17)

|           |              |
|-----------|--------------|
| dsEnabled | dsChildCount |
| [NULL]    | [NULL]       |
| 1         | 4            |

## 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 ) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | datasourcePath | '/shared/examples/ds_orders'   |
| IN        | cisType        | 'LONGVARCHAR'  |
| OUT       | result         | (<br>'LONGVARCHAR',<br>'LONGVARCHAR',<br>'LONGVARCHAR',<br>'-1',<br>'-1',<br>'-983',<br>'LONGVARCHAR',<br>'varchar(2147483647)',<br>'varchar',<br>'2147483647',<br>'-1'<br>) |

## getBasicResourceCursor

This procedure retrieves the basic resource metadata for a given resource. 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 - 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)  |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
| OUT       | resourceCursor | CURSOR (       name VARCHAR(255),       "path" VARCHAR(32768),       "type" VARCHAR(32768),       subtype VARCHAR(255),       enabled BIT     ) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value  |
|-----------|------------------|--|
| IN        | fullResourcePath | /shared/ASAssets/Utilities/repository/examples/source/proc1  |
| IN        | resourceType     | PROCEDURE  |
| OUT       | resourceCursor   | name: 'proc1'<br>path: '/shared/ASAssets/Utilities/repository/examples/source/proc1'<br>type: 'PROCEDURE'<br>subtype: 'SQL_SCRIPT_PROCEDURE'<br>enabled: 1 |

## getBasicResourceCursor\_All

This procedure retrieves the "all" basic resource metadata for a given resource. 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 - this procedure takes the XML response from getBasicResourceXML() 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)  |

| Direction | Parameter Name | Parameter Type   |
|-----------|----------------|--|
| OUT       | resourceCursor | CURSOR (       name VARCHAR(255),       "path" VARCHAR(32768),       "type" VARCHAR(32768),       subtype VARCHAR(255),       enabled BIT,       id INTEGER,       changeId INTEGER,       ownerDomain VARCHAR(255),       ownerName VARCHAR(255),       impactLevel VARCHAR(255),       annotation LONGVARCHAR,       explicitlyDesigned BIT,       tableType VARCHAR(255),       sqlText LONGVARCHAR,       scriptText LONGVARCHAR     ) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value  |
|-----------|------------------|--|
| IN        | fullResourcePath | '/shared/ASAssets/Utilities/repository/examples/source/trigger1' |
| IN        | resourceType     | TRIGGER  |
| OUT       | resourceCursor   | (Result too large to display)                                    |

## getBasicResourceCursor\_ActionAttributes

This procedure retrieves the resource metadata for a given resource that 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 |
|-----------|----------------|----------------|
|           |                |                |

| Direction | Parameter Name   | Parameter Type  |
|-----------|------------------|---|
| IN        | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType   |
| IN        | resourceType     | VARCHAR(255)  |
| OUT       | resourceCursor   | CURSOR (<br>actionType VARCHAR(255),<br>name      VARCHAR(255),<br>"type"    VARCHAR(255),<br>"value"   VARCHAR(255)<br>) |

## 2. Examples:

### 2.1. 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<br>PROCEDUR      PARAMETER<br>E              S           STRING    'string2'<br>PROCEDUR      PATH       STRING    '/shared/ASAssets/Utilities/repository/examples/source/proc1'<br>E |

## 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)  |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
| 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) ) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value                                     |
|-----------|------------------|---|
| IN        | fullResourcePath | '/shared/examples/LookupProduct'                    |
| IN        | resourceType     | 'PROCEDURE'   |
| OUT       | resourceCursor   | (See table below for resourceCursor example output) |

### 2.2. resourceCursor example output:

|               |                    |               |                      |         |            |               |               |        |
|---------------|--------------------|---------------|----------------------|---------|------------|---------------|---------------|--------|
| resourceName  | resourcePath       | Resource Type | subtype              | enabled | annotation | parameterName | parameterType | pa...  |
| LookupProduct | \$1/ LookupProduct | PROCEDURE     | SQL_SCRIPT_PROCEDURE | 1       | [NULL]     | [NULL]        | [N...]        | [N...] |
| LookupProduct | \$1/ LookupProduct | PROCEDURE     | SQL_SCRIPT_PROCEDURE | 1       |            | ProductID     | INTEGER       | IN...  |
| LookupProduct | \$1/ LookupProduct | PROCEDURE     | SQL_SCRIPT_PROCEDURE | 1       |            | result        | CURSOR(...)   | OU...  |

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/output parameter to return. A cursor of metadata is then 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 | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096))  |
| IN        | resourceType     | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40))  |
| IN        | cursorNum        | INTEGER  |
| OUT       | resourceCursor   | CURSOR ( resourceName VARCHAR(255), resourcePath VARCHAR(1024), resourceType VARCHAR(255), subtype VARCHAR(255), enabled BIT, annotation LONGVARCHAR, tableType VARCHAR(255), sqlText VARCHAR(32768), columnName VARCHAR(255), columnType VARCHAR(255), nativeBaseType VARCHAR(255), nativeType VARCHAR(255) ) |

## 2. Examples:

### 2.1. 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     ) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value  |
|-----------|------------------|--|
| IN        | fullResourcePath | '/shared/examples/ds_orders'   |
| IN        | resourceType     | 'DATA_SOURCE'  |
| OUT       | resourceCursor   | ('autoAddChildren', 'BOOLEAN', 'true', NULL, NULL),<br>('login', 'STRING', 'tutorial', NULL, NULL),<br>... |

## 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 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 and turns it into a cursor which is more usable by other CIS procedures.

## 1. Parameters:

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
|-----------|----------------|----------------|

| Direction | Parameter Name   | Parameter Type  |
|-----------|------------------|---|
| IN        | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType   |
| IN        | resourceType     | VARCHAR(255)  |
| OUT       | resourceCursor   | CURSOR (resourceName VARCHAR(255),<br>resourcePath VARCHAR(1024),<br>resourceType VARCHAR(255),<br>subtype VARCHAR(255),<br>enabled BIT,<br>tableType VARCHAR(255),<br>sqlText VARCHAR(32768),<br>columnName VARCHAR(255),<br>columnType VARCHAR(255) ) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value   |
|-----------|------------------|---|
| IN        | fullResourcePath | /shared/ASAssets/Utilities/repository/examples/source/PROD_UCT_VIEW |
| IN        | resourceType     | TABLE   |
| OUT       | resourceCursor   | See table below for resourceCursor example output                   |

### 2.2. resourceCursor example output:

#### 2.2.1. Note: Path \$1 = /shared/ASAssets/Utilities/repository/examples/source

| resourceName | resourcePath     | Resource Type | subtype   | enabled | Table Type | SQL Text Not | columnName         | columnType    |
|--------------|------------------|---------------|-----------|---------|------------|--------------|--------------------|---------------|
| PRODUCT_VIEW | \$1/PRODUCT_VIEW | TABLE         | SQL_TABLE | 1       | VIEW       | 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\_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),<br>sqlIndexName VARCHAR(255),<br>sqlIndexType VARCHAR(255),<br>sqlIndexUnique BIT,<br>sqlIndexColName VARCHAR(255),<br>sqlIndexColOrder VARCHAR(255) ) |

### **2. Examples:**

#### **2.1. 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                    |

#### **2.2. resourceCursor example output:**

##### **2.2.1. 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),<br>"path" VARCHAR(32768),<br>"type" VARCHAR(255),<br>subtype VARCHAR(255),<br>enabled BIT,<br>explicitlyDesigned BIT,<br>transformSourcePath VARCHAR(32768),<br>transformSourceType VARCHAR(255),<br>xsltText VARCHAR(32768) ) |

#### **2. Examples:**

##### **2.1. Assumptions: none**

| Direction | Parameter Name   | Parameter Value   |
|-----------|------------------|---|
| IN        | fullResourcePath | '/shared/ASAssets/Utilities/repository/lowerLevelProcedures/resourceExists' |
| IN        | resourceType     | 'PROCEDURE'   |
| OUT       | resourceCursor   | See table below for resourceCursor example output                           |

##### **2.2. 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>

```

### **get ChildResourcesCursor**

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)  |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
| OUT       | resourceCursor | CURSOR (resourceName VARCHAR(255),<br>resourcePath VARCHAR(1024),<br>resourceType VARCHAR(255),<br>subtype VARCHAR(255),<br>enabled BIT,<br>isNullable VARCHAR(255),<br>columnName VARCHAR(255),<br>columnType VARCHAR(255),<br>nativeBaseType VARCHAR(255),<br>nativeType VARCHAR(255) ) |

## 2. Examples:

### 2.1. 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<br>BaseType | naive<br>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]        |
| folder1      | \$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)    |

## 2. Examples:

### 2.1. 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 (<br>name                               VARCHAR(1024),<br>annotation                        VARCHAR(1024),<br>connectorType                      VARCHAR(1024),<br>groupName                          VARCHAR(1024),<br>jmsClientID                       VARCHAR(1024),<br>jndiContextFactory               VARCHAR(1024),<br>jndiProperties                   LONGVARCHAR,<br>jndiProviderUrl                  VARCHAR(1024),<br>jndiUser                          VARCHAR(50),<br>jndiPassword                      VARCHAR(50),<br>maxPool                           INTEGER,<br>minPool                           INTEGER,<br>poolTimeout                      INTEGER,<br>queueConnectionFactory        VARCHAR(1024),<br>useJNDI                          VARCHAR(50)<br>) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                     |
|-----------|----------------|-------------------------------------|
| IN        | debug          | 'N'                                 |
| IN        | name           | 'myMQ'                              |
| OUT       | resourceCursor | (Result too large to display here.) |

## **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) |

### **2. Examples:**

#### **2.1. 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' |

## **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,<br>useEnabled BIT,<br>tableGatherDefault VARCHAR,<br>numThreads INTEGER,<br>maxTime INTEGER,<br>refreshMode VARCHAR,<br>scheduleMode VARCHAR, |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
|           |                | startTime TIMESTAMP,<br>fromTimeInADay BIGINT,<br>endTimeInADay BIGINT,<br>recurringDay INTEGER,<br>interval INTEGER,<br>period VARCHAR,<br>count INTEGER,<br>isCluster BIT,<br>) |

## 2. Examples:

### 2.1. 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) ) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                  |
|-----------|----------------|----------------------------------|
| IN        | defSetPath     | '/lib/utils/System'              |
| OUT       | result         | ('CannotExecuteSelectException', |

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
|           |                | <pre>'EXCEPTION_DEFINITION', NULL, NULL), ... ('CANCELED', 'CONSTANT_DEFINITION', 'VARCHAR(255)', 'CANCELED'), ... ('CONTENT', 'TYPE_DEFINITION', 'VARCHAR(65536)', NULL), ...</pre> |

### 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<br>(VARCHAR(4096) as of CIS 5.1)  |
| IN        | resourceType     | /lib/resource/ResourceDefs.ResourceType<br>(VARCHAR(40) as of CIS 5.1)  |
| OUT       | resourceCursor   | CURSOR ( (resourceName VARCHAR(255), resourcePath VARCHAR(4096), resourceType VARCHAR(40), subtype VARCHAR(40), ownerDomain VARCHAR(255), ownerName VARCHAR(255), impactLevel VARCHAR(255), impactMessage VARCHAR(32768), enabled BIT ) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value                     |
|-----------|------------------|-------------------------------------|
| IN        | fullResourcePath | '/shared/examples/ds_orders/orders' |
| IN        | resourceType     | 'TABLE'                             |

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| OUT       | resourceCursor | (‘ViewOrder’,<br>‘/shared/examples/ViewOrder’,<br>‘TABLE’,<br>‘SQL_TABLE’,<br>‘composite’,<br>‘admin’,<br>‘NONE’,<br>NULL,<br>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<br>(VARCHAR(4096) as of CIS 5.1)   |
| IN        | resourceType     | /lib/resource/ResourceDefs.ResourceType<br>(VARCHAR(40) as of CIS 5.1)   |
| OUT       | resourceCursor   | CURSOR (<br>resourceName VARCHAR(255),<br>resourcePath VARCHAR(4096),<br>resourceType VARCHAR(40),<br>subtype VARCHAR(40),<br>ownerDomain VARCHAR(255),<br>ownerName VARCHAR(255),<br>impactLevel VARCHAR(255),<br>impactMessage VARCHAR(32768),<br>enabled BIT<br>) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|-----------|----------------|-----------------|

| Direction | Parameter Name   | Parameter Value   |
|-----------|------------------|---|
| IN        | fullResourcePath | '/shared/examples/ds_orders/orders'   |
| IN        | resourceType     | 'TABLE'   |
| OUT       | resourceCursor   | ('ViewOrder',<br>'/shared/examples/ViewOrder',<br>'TABLE',<br>'SQL_TABLE',<br>'composite',<br>'admin',<br>'NONE',<br>NULL,<br>1),<br>...) |

### 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<br>(VARCHAR(4096) as of CIS 5.1)   |
| OUT       | result         | CURSOR (<br>ResourcePath VARCHAR(4096),<br>ResourceType VARCHAR(40),<br>ImpactLevel VARCHAR(32768),<br>ImpactMessage VARCHAR(32768)<br>) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value                                      |
|-----------|------------------|--|
| IN        | fullResourcePath | '/shared/examples'                                   |
| OUT       | resourceCursor   | (<br>'/shared/examples/NewView',<br>'TABLE',<br>...) |

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
|           |                | 'UNKNOWN',<br>'View is newly created and has not been saved.'<br>)<br>... |

### 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 (<br>totalResults INTEGER,<br>completed BIT,<br>lastUpdate TIMESTAMP,<br>"path" VARCHAR(4096),<br>"type" VARCHAR(40),<br>subtype VARCHAR(40)<br>) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | taskId         | '1000'   |
| IN        | block          | 1  |
| IN        | pageSize       | NULL   |
| IN        | pageStart      | NULL   |
| OUT       | result         | (<br>8,<br>1,<br>'2012-11-01 10:45:00',<br>'cache_status', |

| Direction | Parameter Name | Parameter Value                          |
|-----------|----------------|--|
|           |                | 'TABLE',<br>'DATABASE_TABLE'<br>)<br>... |

### 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 |

| Direction | Parameter Name     | Parameter Type                          |
|-----------|--------------------|---|
| 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                                 |

## 2. Examples:

### 2.1. 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), |

| Direction | Parameter Name | Parameter Type          |
|-----------|----------------|-------------------------|
|           |                | "type" VARCHAR(40)<br>) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | taskId         | '1000'   |
| IN        | block          | 1  |
| IN        | pageSize       | NULL   |
| IN        | pageStart      | NULL   |
| OUT       | result         | (<br>8,<br>1,<br>'2012-11-01 10:45:00',<br>'cache_status',<br>'TABLE'<br>),<br>... |

## 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                                 |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value              |
|-----------|----------------|------------------------------|
| IN        | dsPath         | '/shared/examples/ds_orders' |
| OUT       | taskId         | '1000'                       |

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| 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 columns 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) ) |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name                 | Parameter Value    |
|-----------|--------------------------------|--------------------|
| IN        | detail                         | 'FULL'             |
| IN        | includeOnlyUnlockableResources | 'FALSE'            |
| OUT       | result                         | ( 'CompositeView', |

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
|           |                | '/shared/examples/CompositeView',<br>'TABLE',<br>'SQL_TABLE',<br>21874,<br>14124,<br>NULL,<br>NULL,<br>'composite',<br>'admin',<br>'NONE',<br>NULL,<br>1,<br>NULL<br>) |

## 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 columns 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)                     |

### 3. Examples:

#### 3.1. 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<br>VARCHAR(50), ContactFirstName<br>VARCHAR(30), ContactLastName<br>VARCHAR(50), BillingAddress |

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
|           |                | 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              |

### 2. Examples:

#### 2.1. Assumptions: none.

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| IN        | resourcePath   | '/shared/examples'  |
| IN        | resourceType   | 'CONTAINER'   |
| OUT       | result         | columnName: NULL<br>columnType: NULL<br>annotation: 'This folder contains pre-created resources.' |

## getResourceByDate

This procedure is used to return a list of resources from a given starting folder based on the options described 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        | optionType     | INTEGER                  |
| IN        | resourcePath   | TypeDefinitions.pathType |
| IN        | resourceType   | VARCHAR                  |
| IN        | resourceNum    | INTEGER                  |
| IN        | resourceDate   | TIMESTAMP                |
| OUT       | result         | CURSOR                   |

## 2. Examples:

### 2.1. Assumptions: none.

| Direction | Parameter Name | Parameter Value       |
|-----------|----------------|-----------------------|
| 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>             |

## 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   |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value   |
|-----------|------------------|---|
| IN        | fullResourcePath | '/shared/ASAssets/Utilities/repository/examples/source' |

| Direction | Parameter Name  | Parameter Value    |
|-----------|-----------------|--------------------|
| 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<br>(VARCHAR(4096) as of CIS 5.1)   |
| IN        | inType         | /lib/resource/ResourceDefs.ResourceType<br>(VARCHAR(40) as of CIS 5.1)   |
| OUT       | result         | CURSOR ( configured BIT,<br>enabled BIT,<br>storageMode VARCHAR(32768),<br>storageDataSourcePath VARCHAR(32768),<br>storageTargetName VARCHAR(32768),<br>storagePath VARCHAR(32768),<br>storageType VARCHAR(32768),<br>refreshMode VARCHAR(32768),<br>scheduleMode VARCHAR(32768),<br>startTime TIMESTAMP,<br>fromTimeInADay BIGINT,<br>endTimeInADay BIGINT,<br>recurringDay INTEGER,<br>"interval" INTEGER,<br>period VARCHAR(32768),<br>"count" INTEGER,<br>isCluster BIT,<br>expirationPeriod BIGINT,<br>clearRule VARCHAR(32768)<br>) |

#### **2. Examples:**

##### **2.1. Assumptions: none**

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Value</b>  |
|------------------|-----------------------|---|
| IN               | inResourcePath        | '/shared/examples/ds_orders/orders'   |
| IN               | inType                | 'TABLE'   |
| OUT              | result                | configured: 1<br>enabled: 1<br>storageMode: DATA_SOURCE<br>storageDataSourcePath : /shared/examples/ds_orders<br>storageTargetName: result<br>storagePath: /shared/examples/ds_orders/orders_cache<br>storageType: TABLE<br>refreshMode: MANUAL<br>scheduleMode: [NULL]<br>startTime: [NULL]<br>fromTimeInADay: [NULL]<br>endTimeInADay: [NULL]<br>recurringDay: [NULL]<br>interval: [NULL]<br>period: [NULL]<br>count: [NULL]<br>isCluster: [NULL]<br>expirationPeriod: 0<br>clearRule: NONE |

### **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:**

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Type</b>   |
|------------------|-----------------------|---|
| 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)   |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
| OUT       | result         | CURSOR ( impactLevel VARCHAR(32768), impactMessage VARCHAR(32768) ) |

## 2. Examples:

### 2.1. 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<br>impactMessage: View is newly created and has not been saved. |

## 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.

## 1. Parameters:

| Direction | Parameter Name  | Parameter Type |
|-----------|---|----------------|
| IN        | <b>resourcePath</b> – CIS source resource path to being assessing the parent data lineage     | pathType       |
| IN        | <b>resourceType</b> – Type of CIS resource to be created.                                     | VARCHAR        |
| IN        | <b>excludePathsList</b> – comma separated list of resource paths or partials paths to exclude | LONGVARCHAR    |
| IN        | <b>datasourceAncestor</b> – flag to get the data source ancestors. 1=yes, 0=no                | INTEGER        |
| IN        | <b>inIgnoreResourceDoesNotExist</b> – flag to ignore missing resources. 1=yes, 0=no           | INTEGER        |
| OUT       | <b>datasourceResource</b>   | CURSOR (       |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
|           |                | id INTEGER,<br>resourceName VARCHAR(255),<br>resourcePath pathType,<br>resourceType VARCHAR(255),<br>subtype VARCHAR(255),<br>enabled BIT,<br>childCount INTEGER) |

## 2. Examples:

### 2.1. 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             |

### 2.2. 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   |

### 2.3. Chart 2: Columns (6-7)

| enabled | childCount |
|---------|------------|
| 1       | 1          |
| 1       | 7          |

## 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, 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        | <b>inSeqNum</b> - sequence number starting with 1.  | INTEGER  |
| IN        | <b>inParentID</b> - the id of the parent.   | INTEGER  |
| IN        | <b>inResourceDepth</b> - the depth of the parent (number of levels which recursion has occurred). | INTEGER  |
| IN        | <b>resourcePath</b> - CIS source path to an actual resource                                       | pathType   |
| IN        | <b>resourceType</b> - Type of CIS resource to be created. It is null on the first invocation.     | VARCHAR  |
| IN        | <b>excludePathsList</b> - comma separate list of resource paths or partials paths to exclude      | LONGVARCHAR  |
| IN        | <b>datasourceAncestor</b> - flag to get the data source ancestor                                  | INTEGER  |
| IN        | <b>inIgnoreResourceDoesNotExist</b> – flag to ignore missing resources                            | INTEGER  |
| OUT       | resourceTreeList  | CURSOR lineageTreeType (<br>seqNum INTEGER, -- generated sequence number<br>resourceID INTEGER, -- resource id from CIS<br>parentID INTEGER, -- how this row relates to resourceID<br>resDepth INTEGER, -- depth as related to the start<br>treeType VARCHAR(255), -- Parent, Child<br>resName VARCHAR(255), -- resource name<br>resPath pathType, -- resource path<br>resType VARCHAR(255), -- resource type<br>subType VARCHAR(255), -- resource         ) |

| Direction | Parameter Name   | Parameter Type |
|-----------|--|----------------|
|           | sub type<br>enabled BIT, -- enabled or not (1 or 0)<br>dsID INTEGER, -- datasource<br>ancestor id<br>dsResName VARCHAR(255), -- datasource<br>ancestor name<br>dsResPath pathType, -- datasource<br>ancestor path<br>dsResType VARCHAR(255), -- datasource<br>ancestor type<br>dsResSubType VARCHAR(255), -- datasource<br>ancestor sub type<br>dsEnabled BIT, -- datasource ancestor<br>enabled (1 or 0)<br>dsChildCount INTEGER -- datasource<br>ancestor number of children<br>); |                |

## 2. Examples:

### 2.1. 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               |

### 2.2. Chart 1: Columns (1-6)

| seqnum | resourceID | parentID | resDepth | treeType | resName       |
|--------|------------|----------|----------|----------|---------------|
| 1      | 20760      | [NULL]   | 0        | Parent   | LookupProduct |
| 2      | 20633      | 20760    | 1        | Child    | products      |

### 2.3. 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       |

#### 2.4. 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 |

#### 2.5. Chart 4: Columns (16-17)

| dsEnabled | dsChildCount |
|-----------|--------------|
| [NULL]    | [NULL]       |
| 1         | 4            |

### 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 dependency 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        |

| <b>Direction</b> | <b>Parameter Name</b>   | <b>Parameter Type</b>  |
|------------------|---|--|
| IN               | <b>resourcePath</b> - CIS source path to an actual resource                                   | pathType   |
| IN               | <b>resourceType</b> - Type of CIS resource to be created. It is null on the first invocation. | VARCHAR  |
| IN               | <b>excludePathsList</b> - comma separate list of resource paths or partials paths to exclude  | LONGVARCHAR  |
| IN               | <b>datasourceAncestor</b> - flag to get the data source ancestor                              | INTEGER  |
| IN               | <b>inIgnoreResourceDoesNotExist</b> – flag to ignore missing resources                        | INTEGER  |
| OUT              | resourceTreeList  | CURSOR lineageTreeType (<br>seqNum INTEGER, -- generated sequence number<br>resourceID INTEGER, -- resource id from CIS<br>parentID INTEGER, -- how this row relates to resourceID<br>resDepth INTEGER, -- depth as related to the start<br>treeType VARCHAR(255), -- Parent, Ancestor<br>resName VARCHAR(255), -- resource name<br>resPath pathType, -- resource path<br>resType VARCHAR(255), -- resource type<br>subType VARCHAR(255), -- resource sub type<br>enabled BIT, -- enabled or not (1 or 0)<br>dsID INTEGER, -- datasource<br>ancestor id<br>dsResName VARCHAR(255), -- datasource<br>ancestor name<br>dsResPath pathType, -- datasource<br>ancestor path<br>dsResType VARCHAR(255), -- datasource<br>ancestor type<br>dsResSubType VARCHAR(255), -- datasource<br>ancestor sub type         ) |

| Direction | Parameter Name | Parameter Type   |
|-----------|----------------|--|
|           |                | dsEnabled BIT, -- datasource ancestor enabled (1 or 0)<br>dsChildCount INTEGER -- datasource ancestor number of children<br>); |

## 2. Examples:

### 2.1. 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           |

### 2.2. Chart 1: Columns (1-6)

| seqnum | resourceID | parentID | resDepth | treeType | resName   |
|--------|------------|----------|----------|----------|-----------|
| 1      | 22063      | [NULL]   | 0        | Parent   | ViewSales |
| 1      | 1          | [NULL]   | 0        | Ancestor | [NULL]    |
| ...    |            |          |          |          |           |

### 2.3. Chart 2: Columns (7-10)

| resPath                    | resType   | subType   | enabled |
|----------------------------|-----------|-----------|---------|
| /shared/examples/ViewSales | TABLE     | SQL_TABLE | 1       |
| /                          | CONTAINER | NONE      | 1       |
| ...                        |           |           |         |

### 2.4. Chart 3: Columns (11-15)

| dsID   | dsResName | dsResPath | dsResType | dsResSubType |
|--------|-----------|-----------|-----------|--------------|
| [NULL] | [NULL]    | [NULL]    | [NULL]    | [NULL]       |
| [NULL] | [NULL]    | [NULL]    | [NULL]    | [NULL]       |
| ...    |           |           |           |              |

## 2.5. Chart 4: Columns (16-17)

```

dsEnabled      dsChildCount
[NULL]         [NULL]
[NULL]         [NULL]
...

```

### **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       | resourceTreeList | CURSOR (           name          VARCHAR,           resPath       TypeDefinitions.pathType,           resType       VARCHAR,           subType       VARCHAR,           creationDate  TIMESTAMP,           creationDateBignum BIGINT,           creatorUserDomain VARCHAR(255),           creatorUserName VARCHAR(255)           lastModifiedDate  TIMESTAMP,           lastModifiedDateBignum BIGINT,           lastModifiedUserDomain VARCHAR(255),           lastModifiedUserName VARCHAR(255)         ) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value              |
|-----------|------------------|------------------------------|
| IN        | resourcePath     | '/shared/ASAssets/Utilities' |
| IN        | resourceType     | 'CONTAINER'                  |
| OUT       | resourceTreeList | See chart below              |

##### 2.2. Chart showing example output for resourceTreeList:

|              |   |           |                |
|--------------|---|-----------|----------------|
| name         | resPath                                 | resType   | subType        |
| calculations | /shared/ASAssets/Utilities/calculations | CONTAINER | FOLDER_CONTAIN |
| conversions  | /shared/ASAssets/Utilities/conversions  | CONTAINER | FOLDER_CONTAIN |
| file         | /shared/ASAssets/Utilities/file         | CONTAINER | FOLDER_CONTAIN |
| log          | /shared/ASAssets/Utilities/log          | CONTAINER | FOLDER_CONTAIN |
| repository   | /shared/ASAssets/Utilities/repository   | CONTAINER | FOLDER_CONTAIN |
| Etc...       |   |           |                |

### getResouceListRecursive

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                    | resourceTreeList          | CURSOR ( <table> <tr> <td>name</td> <td>VARCHAR,</td> </tr> <tr> <td>resPath</td> <td>TypeDefinitions.pathType,</td> </tr> <tr> <td>resType</td> <td>VARCHAR,</td> </tr> <tr> <td>subType</td> <td>VARCHAR,</td> </tr> <tr> <td>creationDate</td> <td>TIMESTAMP,</td> </tr> <tr> <td>creationDateBigint</td> <td>BIGINT,</td> </tr> <tr> <td>creatorUserDomain</td> <td>VARCHAR(255),</td> </tr> <tr> <td>creatorUserName</td> <td>VARCHAR(255)</td> </tr> <tr> <td>lastModifiedDate</td> <td>TIMESTAMP,</td> </tr> <tr> <td>lastModifiedDateBigint</td> <td>BIGINT,</td> </tr> <tr> <td>lastModifiedUserDomain</td> <td>VARCHAR(255),</td> </tr> <tr> <td>lastModifiedUserName</td> <td>VARCHAR(255)</td> </tr> </table> ) | name | VARCHAR, | resPath | TypeDefinitions.pathType, | resType | VARCHAR, | subType | VARCHAR, | creationDate | TIMESTAMP, | creationDateBigint | BIGINT, | creatorUserDomain | VARCHAR(255), | creatorUserName | VARCHAR(255) | lastModifiedDate | TIMESTAMP, | lastModifiedDateBigint | BIGINT, | lastModifiedUserDomain | VARCHAR(255), | lastModifiedUserName | VARCHAR(255) |
| name                   | VARCHAR,                  |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| resPath                | TypeDefinitions.pathType, |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| resType                | VARCHAR,                  |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| subType                | VARCHAR,                  |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| creationDate           | TIMESTAMP,                |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| creationDateBigint     | BIGINT,                   |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| creatorUserDomain      | VARCHAR(255),             |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| creatorUserName        | VARCHAR(255)              |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| lastModifiedDate       | TIMESTAMP,                |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| lastModifiedDateBigint | BIGINT,                   |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| lastModifiedUserDomain | VARCHAR(255),             |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |
| lastModifiedUserName   | VARCHAR(255)              |   |      |          |         |                           |         |          |         |          |              |            |                    |         |                   |               |                 |              |                  |            |                        |         |                        |               |                      |              |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value              |
|-----------|------------------|------------------------------|
| IN        | resourcePath     | '/shared/ASAssets/Utilities' |
| IN        | resourceType     | 'CONTAINER'                  |
| OUT       | resourceTreeList | See chart below              |

## 2.2. Chart showing example output for resourceTreeList:

| name                   | resPath   | resType     | subType          |
|------------------------|---|-------------|------------------|
| TypeDefinitions        | /shared/ASAssets/Utilities/TypeDefinitions                    | PROCEDURE   | SQL_SCRIPT_PROC  |
| calculations           | /shared/ASAssets/Utilities/calculations                       | CONTAINER   | FOLDER_CONTAINER |
| calculateAge           | /shared/ASAssets/Utilities/calculations/calculateAge          | PROCEDURE   | SQL_SCRIPT_PROC  |
| conversions            | /shared/ASAssets/Utilities/conversions                        | CONTAINER   | FOLDER_CONTAINER |
| convertBoolean         | /shared/ASAssets/Utilities/conversions/convertBoolean         | PROCEDURE   | SQL_SCRIPT_PROC  |
| convertDoubleToInteger | /shared/ASAssets/Utilities/conversions/convertDoubleToInteger | PROCEDURE   | SQL_SCRIPT_PROC  |
| convertYN              | /shared/ASAssets/Utilities/conversions/convertYN              | PROCEDURE   | SQL_SCRIPT_PROC  |
| 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_PROC  |
| logDebugMessage        | /shared/ASAssets/Utilities/log/logDebugMessage                | PROCEDURE   | SQL_SCRIPT_PROC  |
| repository             | /shared/ASAssets/Utilities/repository                         | CONTAINER   | FOLDER_CONTAINER |
| _debug                 | /shared/ASAssets/Utilities/repository/_debug                  | PROCEDURE   | SQL_SCRIPT_PROC  |
| 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 (<br>name /lib/resource/ResourceDefs.ResourceName,<br>resPath /lib/resource/ResourceDefs.ResourcePath,<br>resType /lib/resource/ResourceDefs.ResourceType,<br>subtype /lib/resource/ResourceDefs.ResourceType,<br>creationDate TIMESTAMP,<br>creationDateBignum BIGINT,<br>creatorUserDomain VARCHAR(255),<br>creatorUserName VARCHAR(255),<br>lastModifiedDate TIMESTAMP, |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
|           |                | lastModifiedDateBigint<br>lastModifiedUserDomain<br>lastModifiedUserName<br>) |

## 2. Examples:

### 2.1. 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   |

| <b>Direction</b> | <b>Parameter Name</b>     | <b>Parameter Type</b>   |
|------------------|---------------------------|---|
| IN               | includePrvsEqualFilter    | VARCHAR   |
| IN               | includePrvsLikeFilter     | VARCHAR   |
| IN               | excludePrvsNotEqualFilter | VARCHAR   |
| IN               | excludePrvsNotLikeFilter  | VARCHAR   |
| IN               | includeColumnPrvs         | BIT   |
| IN               | debug                     | CHAR(1)   |
| OUT              | result                    | CURSOR (<br>resourceID INTEGER,<br>treeType VARCHAR(255),<br>resName VARCHAR(255),<br>path VARCHAR(4096),<br>type VARCHAR(40),<br>subtype VARCHAR(40),<br>enabled BIT,<br>name VARCHAR(255),<br>nameType VARCHAR(255),<br>domain VARCHAR(255),<br>privilegeStatus VARCHAR(255),<br>privilegeRunTimeAnalysis VARCHAR(255),<br>privilegeDesignTimeAnalysis VARCHAR(255),<br>prvs VARCHAR(255),<br>combinedPrvs VARCHAR(255),<br>inheritedPrvs VARCHAR(255),<br>p_N BIT,<br>p_R BIT,<br>p_W BIT,<br>p_E BIT,<br>p_S BIT,<br>p_U BIT,<br>p_I BIT,<br>p_D BIT,<br>p_G BIT,<br>c_N BIT,<br>c_R BIT,<br>c_W BIT,<br>c_E BIT,<br>c_S BIT, |

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Type</b>  |
|------------------|-----------------------|--|
|                  |                       | c_U<br>c_I<br>c_D<br>c_G<br>i_N<br>i_R<br>i_W<br>i_E<br>i_S<br>i_U<br>i_I<br>i_D<br>i_G<br>) |

## 2. Examples:

### 2.1. Assumptions: none

| <b>Direction</b> | <b>Parameter Name</b>        | <b>Parameter Value</b>  |
|------------------|------------------------------|---|
| IN               | resourcePath                 | '/shared/examples'  |
| IN               | resourceType                 | 'CONTAINER'   |
| IN               | excludePathsList             | '/shared/examples/not_this_folder,<br>/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               | includePrvsEqualFilter       | NULL  |
| IN               | includePrvsLikeFilter        | NULL  |
| IN               | excludePrvsNotEqualFilter    | NULL  |
| IN               | excludePrvsNotLikeFilter     | NULL  |

| Direction | Parameter Name     | Parameter Value        |
|-----------|--------------------|------------------------|
| 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<br>(VARCHAR(4096))   |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | resourceType              | /lib/resource/ResourceDefs.ResourceType<br>(VARCHAR(4096))   |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | nameTypeFilter            | VARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | domainFilter              | VARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | includeNameEqualFilter    | LONGVARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | includeNameLikeFilter     | LONGVARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | excludeNameNotEqualFilter | LONGVARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | excludeNameNotLikeFilter  | LONGVARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | includePrvsEqualFilter    | VARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | includePrvsLikeFilter     | VARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | excludePrvsNotEqualFilter | VARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | excludePrvsNotLikeFilter  | VARCHAR  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | includeColumnPrivs        | BIT  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| IN            | debug                     | CHAR(1)  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| OUT           | result                    | CURSOR ( <table> <tr> <td>name</td> <td>VARCHAR(255),</td> </tr> <tr> <td>path</td> <td>VARCHAR(4096),</td> </tr> <tr> <td>type</td> <td>VARCHAR(40),</td> </tr> <tr> <td>nameType</td> <td>VARCHAR(255),</td> </tr> <tr> <td>domain</td> <td>VARCHAR(255),</td> </tr> <tr> <td>prvs</td> <td>VARCHAR(255),</td> </tr> <tr> <td>combinedPrvs</td> <td>VARCHAR(255),</td> </tr> <tr> <td>inheritedPrvs</td> <td>VARCHAR(255),</td> </tr> </table> | name | VARCHAR(255), | path | VARCHAR(4096), | type | VARCHAR(40), | nameType | VARCHAR(255), | domain | VARCHAR(255), | prvs | VARCHAR(255), | combinedPrvs | VARCHAR(255), | inheritedPrvs | VARCHAR(255), |
| name          | VARCHAR(255),             |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| path          | VARCHAR(4096),            |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| type          | VARCHAR(40),              |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| nameType      | VARCHAR(255),             |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| domain        | VARCHAR(255),             |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| prvs          | VARCHAR(255),             |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| combinedPrvs  | VARCHAR(255),             |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |
| inheritedPrvs | VARCHAR(255),             |  |      |               |      |                |      |              |          |               |        |               |      |               |              |               |               |               |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
|           | 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            |
|           | )              |                |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| IN        | resourcePath   | '/shared/examples' |
| IN        | resourceType   | 'CONTAINER'        |
| IN        | nameTypeFilter | 'GROUP'            |
| IN        | domainFilter   | 'composite'        |

| Direction | Parameter Name            | Parameter Value        |
|-----------|---------------------------|------------------------|
| IN        | includeNameEqualFilter    | 'all'                  |
| IN        | includeNameLikeFilter     | NULL                   |
| IN        | excludeNameNotEqualFilter | NULL                   |
| IN        | excludeNameNotLikeFilter  | NULL                   |
| IN        | includePrvsEqualFilter    | NULL                   |
| IN        | includePrvsLikeFilter     | NULL                   |
| IN        | excludePrvsNotEqualFilter | NULL                   |
| IN        | excludePrvsNotLikeFilter  | NULL                   |
| IN        | includeColumnPrvs         | 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<br>(VARCHAR(4096))  |
| IN        | resourceType   | /lib/resource/ResourceDefs.ResourceType<br>(VARCHAR(4096))  |
| OUT       | result         | CURSOR (<br>"path" VARCHAR(32768),<br>"type" VARCHAR(32768),<br>"name" VARCHAR(32768),<br>"domain" VARCHAR(32768),<br>prvs VARCHAR(32768),<br>combinedPrvs VARCHAR(32768),<br>inheritedPrvs VARCHAR(32768)<br>) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| IN        | resourcePath   | '/shared/examples' |
| IN        | resourceType   | 'CONTAINER'        |
| OUT       | result         | See chart below    |

## 2.2. Chart showing example output for result:

|                  |           |        |           |                   |                   |                   |
|------------------|-----------|--------|-----------|-------------------|-------------------|-------------------|
| Path             | Type      | Name   | Domain    | Prvs              | CombinedPrvs      | InheritedPrvs     |
| /shared/examples | CONTAINER | admin  | composite | READ WRITE EXE... | READ WRITE EXE... | READ WRITE EXI... |
| /shared/examples | CONTAINER | nobody | composite | NONE              | NONE              | NONE              |
| Etc.             |           |        |           |                   |                   |                   |

### 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   |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value   |
|-----------|------------------|---|
| IN        | fullResourcePath | '/shared/ASAssets/Utilities/repository/examples/source/proc2' |

| Direction | Parameter Name | Parameter Value               |
|-----------|----------------|-------------------------------|
| 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         ) |

#### 2. Examples:

##### 2.1. 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'                           |

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|------------------|
| OUT       | maxTime        | -1               |
| OUT       | columnSettings | See table below: |

## 2.2. 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                    |
| ...          |        |           |           |                |                      |

### 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)         ) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                  |
|-----------|----------------|----------------------------------|
| IN        | resourcePath   | '/shared/examples/CompositeView' |

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|-------------------|
| IN        | resourceType   | 'TABLE'           |
| OUT       | usedResCursor  | <See table below> |

## 2.2. 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 |

## 2.3. 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                                     |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
| 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) ) |

## 2. Examples:

### 2.1. 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>                |

### 2.2. 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 |
| ...          |                               |              |           |         |       |

### 2.3. 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 recursively 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 (<br>resPath     /lib/resource/ResourceDefs.ResourcePath,<br>privRead    CHAR(1),<br>privWrite   CHAR(1),<br>privExecute CHAR(1),<br>privSelect   CHAR(1),<br>privDelete   CHAR(1),<br>privInsert   CHAR(1),<br>privDelete   CHAR(1),<br>privGrant    CHAR(1)<br>) |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| IN        | userName       | 'admin'            |
| IN        | domainName     | 'composite'        |
| IN        | beginFolder    | '/shared/examples' |
| OUT       | result         | See chart below    |

#### **2.2. 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 (<br>resourcePath<br>/lib/resource/ResourceDefs.ResourcePath,<br>resourceType<br>/lib/resource/ResourceDefs.ResourceType,<br>subType<br>/lib/resource/ResourceDefs.ResourceType,<br>impactLevel       VARCHAR(1024),<br>impactMessage     VARCHAR(32768),<br>scriptText        LONGVARCHAR<br>) |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name         | Parameter Value   |
|-----------|------------------------|---|
| IN        | debug                  | 'N'   |
| IN        | inStartingFolders      | '/shared/examples'  |
| IN        | inExcludePathsKeywords | NULL  |
| OUT       | result                 | (<br>'/shared/examples/NewView',<br>'TABLE',<br>'SQL_TABLE',<br>'UNKNOWN',<br>'View is newly created and has not been saved.',<br>'SELECT * FROM' |

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|           |                | )               |

## introspectResourcesResult

This procedure gathers the results from a call to introspectResourcesTask. 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.

The result output will only be a single row if the introspection is successful. Otherwise, the additional result rows will indicate what went wrong.

### 1. Parameters:

| Direction               | Parameter Name  | Parameter Type   |              |          |           |      |        |                 |                     |          |           |            |         |            |            |          |              |          |              |          |              |          |                     |          |                |          |                  |          |                  |          |                         |          |              |          |            |          |        |                |        |              |         |              |          |                 |            |          |             |                 |      |                 |      |                 |         |                 |
|-------------------------|-----------------|--|--------------|----------|-----------|------|--------|-----------------|---------------------|----------|-----------|------------|---------|------------|------------|----------|--------------|----------|--------------|----------|--------------|----------|---------------------|----------|----------------|----------|------------------|----------|------------------|----------|-------------------------|----------|--------------|----------|------------|----------|--------|----------------|--------|--------------|---------|--------------|----------|-----------------|------------|----------|-------------|-----------------|------|-----------------|------|-----------------|---------|-----------------|
| IN                      | taskId          | LONGVARCHAR  |              |          |           |      |        |                 |                     |          |           |            |         |            |            |          |              |          |              |          |              |          |                     |          |                |          |                  |          |                  |          |                         |          |              |          |            |          |        |                |        |              |         |              |          |                 |            |          |             |                 |      |                 |      |                 |         |                 |
| IN                      | block           | BIT  |              |          |           |      |        |                 |                     |          |           |            |         |            |            |          |              |          |              |          |              |          |                     |          |                |          |                  |          |                  |          |                         |          |              |          |            |          |        |                |        |              |         |              |          |                 |            |          |             |                 |      |                 |      |                 |         |                 |
| IN                      | pageSize        | INTEGER  |              |          |           |      |        |                 |                     |          |           |            |         |            |            |          |              |          |              |          |              |          |                     |          |                |          |                  |          |                  |          |                         |          |              |          |            |          |        |                |        |              |         |              |          |                 |            |          |             |                 |      |                 |      |                 |         |                 |
| IN                      | pageStart       | INTEGER  |              |          |           |      |        |                 |                     |          |           |            |         |            |            |          |              |          |              |          |              |          |                     |          |                |          |                  |          |                  |          |                         |          |              |          |            |          |        |                |        |              |         |              |          |                 |            |          |             |                 |      |                 |      |                 |         |                 |
| OUT                     | result          | CURSOR ( <table> <tr><td>totalResults</td><td>INTEGER,</td></tr> <tr><td>completed</td><td>BIT,</td></tr> <tr><td>status</td><td>VARCHAR(32768),</td></tr> <tr><td>introspectorVersion</td><td>INTEGER,</td></tr> <tr><td>startTime</td><td>TIMESTAMP,</td></tr> <tr><td>endTime</td><td>TIMESTAMP,</td></tr> <tr><td>addedCount</td><td>INTEGER,</td></tr> <tr><td>removedCount</td><td>INTEGER,</td></tr> <tr><td>updatedCount</td><td>INTEGER,</td></tr> <tr><td>skippedCount</td><td>INTEGER,</td></tr> <tr><td>totalCompletedCount</td><td>INTEGER,</td></tr> <tr><td>toBeAddedCount</td><td>INTEGER,</td></tr> <tr><td>toBeRemovedCount</td><td>INTEGER,</td></tr> <tr><td>toBeUpdatedCount</td><td>INTEGER,</td></tr> <tr><td>totalToBeCompletedCount</td><td>INTEGER,</td></tr> <tr><td>warningCount</td><td>INTEGER,</td></tr> <tr><td>errorCount</td><td>INTEGER,</td></tr> <tr><td>"path"</td><td>VARCHAR(4096),</td></tr> <tr><td>"type"</td><td>VARCHAR(40),</td></tr> <tr><td>subtype</td><td>VARCHAR(40),</td></tr> <tr><td>"action"</td><td>VARCHAR(32768),</td></tr> <tr><td>durationMs</td><td>INTEGER,</td></tr> <tr><td>entryStatus</td><td>VARCHAR(32768),</td></tr> <tr><td>code</td><td>VARCHAR(32768),</td></tr> <tr><td>name</td><td>VARCHAR(32768),</td></tr> <tr><td>message</td><td>VARCHAR(32768),</td></tr> </table> ) | 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 | VARCHAR(32768), |
| 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                 | VARCHAR(32768), |  |              |          |           |      |        |                 |                     |          |           |            |         |            |            |          |              |          |              |          |              |          |                     |          |                |          |                  |          |                  |          |                         |          |              |          |            |          |        |                |        |              |         |              |          |                 |            |          |             |                 |      |                 |      |                 |         |                 |

| Direction | Parameter Name          | Parameter Type                    |
|-----------|-------------------------|-----------------------------------|
|           | detail<br>severity<br>) | VARCHAR(32768),<br>VARCHAR(32768) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value               |
|-----------|----------------|-------------------------------|
| IN        | taskId         | '1000'                        |
| IN        | block          | 1                             |
| IN        | pageSize       | NULL                          |
| IN        | pageStart      | NULL                          |
| OUT       | result         | (result too large to display) |

## introspectResourcesTask

This procedure begins an asynchronous thread that introspects a data source and adds objects to the CIS repository. This asynchronous thread will survive server restarts. Use `repository/introspectResourcesResult()` to retrieve the results.

See `repository/getIntrospectableResourceIdsTask()` for the list of resources in a data source that CAN be introspected (whether they're in the CIS repository or not.)

See `repository/getIntrospectedResourceIdsTask()` for the list of resources in a data source that already ARE introspected.

### Input:

dsPath - The path to the data source to be modified.

Values: Any path to a data source

updateAllIntrospectedResources - Indicates whether all currently introspected resources should be updated.

Values: 1 or 0 (NULL indicates FALSE)

failFast - Indicates whether the introspection task should halt on first error or continue on a best effort basis.

Values: 1 or 0 (NULL indicates FALSE)

commitOnFailure - Indicates whether the introspection should commit whatever it can. If fastFail is also TRUE, then only the successfully introspected resources, up to that point, will be committed.

Values: 1 or 0 (NULL indicates FALSE)

autoRollback - Indicates whether the introspection task will rollback rather than committing. This supersedes all commit options. This allows you to perform a dry run of resource introspection. The "introspectResourcesResult" procedure is usable if autoRollback is TRUE. If autoRollback is FALSE or NULL, then the introspection will not automatically be rolled back.

Values: 1 or 0 (NULL indicates FALSE)

scanForNewResourcesToAutoAdd - Indicates whether the introspection task will scan for native resources that have been newly added to the data source. If newly added resources are found and their parent container has the "autoAddChildren" introspection attribute set, then that child will automatically be introspected.

Values: 1 or 0 (NULL indicates FALSE)

runInBackgroundTransaction - Indicates that the introspection task should run as a background transaction.

Values: 1 or 0 (NULL indicates FALSE)

entries - The list of resources to introspect and their respective introspection actions. Optionally (rare) includes introspection attributes. (See /services/webservices/system/admin/resource/operations/getIntrospectionAttributeDefs for details.)

Values: A vector of type

/shared/ASAssets/Utilities/TypeDefinitions.IntrospectionPlanVectorType. For example, when introspecting /shared/examples/ds\_orders the entities vector might look like:

```
VECTOR [
    ('orders',      'TABLE',   'DATABASE_TABLE', 'ADD_OR_UPDATE',  NULL),
    ('customers',   'TABLE',   'DATABASE_TABLE', 'REMOVE',        NULL)
]
```

For an Oracle database, the entities vector might look like (notice the attribute vector on the SCOTT schema entry):

```
VECTOR [
    ('SCOTT',       'CONTAINER', 'SCHEMA_CONTAINER', 'ADD_OR_UPDATE', VECTOR
     [ ('tablePatterns', 'STRING', 'E%, D%') ]),
    ('SCOTT/EMP',   'TABLE',    'DATABASE_TABLE',   'ADD_OR_UPDATE',  NULL),
    ('SCOTT/DEPT',  'TABLE',    'DATABASE_TABLE',   'ADD_OR_UPDATE',  NULL)
]
```

dsAttributes - The list of introspection attributes to set at the data source level.

Values: A vector of type /shared/ASAssets/Utilities/TypeDefinitions.AttributesVectorType. See /services/webservices/system/admin/resource/operations/getIntrospectionAttributeDefs for detail. For example, when introspecting /shared/examples/ds\_orders the dsAttributes vector might look like:

```

VECTOR [
    ('autoAddChildren', 'BOOLEAN', 'true'),
    ('patternSeparator', 'STRING', ',')
]

```

### **Output:**

taskId - The introspection task ID. Use this with repository/introspectResourcesResult().

Values: A task ID

totalResults - Total size of the result set (if known.)

Values: A positive integer or NULL.

completed - Indicates whether the introspection task has already finished (if known.)

Values: 1, 0, or NULL (indicating completion status unknown.)

### **1. Parameters:**

| Direction | Parameter Name                 | Parameter Type  |
|-----------|--------------------------------|---|
| IN        | dsPath                         | /lib/resource/ResourceDefs.ResourcePath                 |
| IN        | updateAllIntrospectedResources | BIT   |
| IN        | failFast                       | BIT   |
| IN        | commitOnFailure                | BIT   |
| IN        | autoRollback                   | BIT   |
| IN        | scanForNewResourcesToAutoAdd   | BIT   |
| IN        | runInBackgroundTransaction     | BIT   |
| IN        | entries                        | VECTOR<br>(TypeDefinitions.IntrospectionPlanVectorType) |
| IN        | dsAttributes                   | VECTOR (TypeDefinitions.AttributesVectorType)           |
| OUT       | taskId                         | VARCHAR(32768)  |
| OUT       | totalResults                   | INTEGER   |
| OUT       | completed                      | BIT   |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name                 | Parameter Value              |
|-----------|--------------------------------|------------------------------|
| IN        | dsPath                         | '/shared/examples/ds_orders' |
| IN        | updateAllIntrospectedResources | NULL                         |

| Direction | Parameter Name             | Parameter Value      |
|-----------|----------------------------|----------------------|
| IN        | failFast                   | 1                    |
| IN        | commitOnFailure            | 0                    |
| IN        | autoRollback               | 1                    |
| IN        | scanForNewResourcesToAdd   | 0                    |
| IN        | runInBackgroundTransaction | 0                    |
| IN        | entries                    | (see examples above) |
| IN        | dsAttributes               | (see example above)  |
| OUT       | taskId                     | '1000'               |
| OUT       | totalResults               | NULL                 |
| OUT       | Completed                  | 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   |

### 2. Examples:

#### 2.1. 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)  |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
| IN        | rebindVector   | VECTOR (rebindRow ROW ( oldPath TypeDefinitions.pathType,<br>oldType VARCHAR(255),<br>newPath TypeDefinitions.pathType,<br>newType VARCHAR(255)<br>)) |
| OUT       | success        | BIT   |
| OUT       | createResponse | XML   |
| OUT       | faultResponse  | XML   |

## 2. Examples:

### 2.1. 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',<br>'TABLE',<br>'/shared/ASAssets/Utilities/repository/examples/source/ds_inventory/<br>products',<br>'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)                          |

## 2. Examples:

### 2.1. 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   |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value  |
|-----------|------------------|--|
| IN        | fullResourcePath | '/shared/ASAssets/Utilities/repository/examples/source/ds_inventor_y/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   |

### 2. Examples:

#### 2.1. 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                |

### 2. Examples:

## 2.1. 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    |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                         |
|-----------|----------------|---|
| IN        | resourcePath   | '/shared/"XML"/"path with spaces"/test' |
| OUT       | resultPath     | '/shared/XML/path with spaces/test'     |

## 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.

### 1. Parameters:

| Direction | Parameter Name    | Parameter Type                          |
|-----------|-------------------|---|
| IN        | startPath         | /lib/resource/ResourceDefs.ResourcePath |
| IN        | startResourceType | /lib/resource/ResourceDefs.ResourceType |
| IN        | removeStr         | LONGVARCHAR                             |
| IN        | replaceStr        | LONGVARCHAR                             |
| OUT       | success           | BIT                                     |
| OUT       | updateResponse    | XML                                     |
| OUT       | faultResponse     | XML                                     |

### 2. Examples:

**2.1. Assumptions: The folder /shared/examples has an annotation of “This folder contains pre-created resources.”**

| Direction | Parameter Name    | Parameter Value    |
|-----------|-------------------|--------------------|
| IN        | startPath         | ‘/shared/examples’ |
| IN        | startResourceType | ‘CONTAINER’        |
| IN        | removeStr         | ‘folder’           |
| IN        | replaceStr        | ‘container’        |
| OUT       | success           | 1                  |
| OUT       | updateResponse    | [XML response]     |
| OUT       | faultResponse     | NULL               |

**resourceExists (Custom Function)**

Determine if a resource exists or not.

**1. Parameters:**

| Direction | Parameter Name   | Parameter Type                                      |
|-----------|------------------|---|
| IN        | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN        | resourceType     | VARCHAR(255)  |
| OUT       | success          | BIT   |

**2. Examples:**

**2.1. Assumptions: none**

| Direction | Parameter Name   | Parameter Value   |
|-----------|------------------|---|
| IN        | fullResourcePath | ‘/shared/ASAssets/Utilities/repository/examples/source/proc2’ |
| IN        | resourceType     | ‘PROCEDURE’   |
| OUT       | success          | 1   |

**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 |
|-----------|----------------|----------------|
|           |                |                |

| 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 |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value                         |
|-----------|------------------|---|
| IN        | fullResourcePath | '/shared/ASAssets/Utilities/repository' |
| IN        | mode             | 'B'                                     |
| OUT       | folderName       | 'shared'                                |
| OUT       | FolderPath       | '/Utilities/repository'                 |

## 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 backslash "\\". For example: \\, \\\(, \\\), \\\[, \\\{, \\\?, \\\\*, \\\+, \\\'

### 3. Parameters:

| Direction | Parameter Name  | Parameter Type  |
|-----------|---|---|
| IN        | <b>startingFolders</b> - comma separated list of starting folder to begin searching.  | LONGVARCHAR   |
| IN        | <b>keywordList</b> - 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 [i]nteger in the keyword list. The following characters in the text require an escape using a backslash "\\" character: '\', '(', ')', '[', '{', '?', '*', '+', '\"' | LONGVARCHAR   |
| IN        | <b>keywordOccurrence</b> - 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  | <p>OUT result PIPE (</p> <p><b>startOrder</b> INTEGER, -- The starting path execution order</p> <p><b>startPath</b> VARCHAR(1024), -- The starting path extracted from the comma separated startingFolderss</p> <p><b>resourcePath</b> VARCHAR(1024), -- The full path to the resource</p> <p><b>resourceType</b> VARCHAR(255), -- The type of resource</p> <p><b>subtype</b> VARCHAR(255), -- The sub-type of the resource</p> |

| Direction | Parameter Name | Parameter Type   |
|-----------|----------------|--|
|           |                | <p><b>impactLevel</b> VARCHAR(255), --<br/>The impact level of the resource<br/>NONE and SYNTAX_ERROR are<br/>permitted.</p> <p><b>pos</b> INTEGER, -- The position of the<br/>first occurrence of any of the<br/>keywords</p> <p><b>keyword</b> LONGVARCHAR, --<br/>The keyword that was found from the<br/>keyword list</p> <p><b>keywordNum</b> INTEGER, -- The<br/>keyword position number within the<br/>keyword list</p> <p><b>occurrence</b> INTEGER – The<br/>occurrence of the keyword</p> <p>)</p> |

#### 4. Examples:

##### 4.1. Assumptions: none

| Direction | Parameter Name           | Parameter Value  |
|-----------|--------------------------|--|
| IN        | <b>startingFolders</b>   | /shared/examples, /shared/ASAssets/Utilities/repository/examples |
| IN        | <b>keywordList</b>       | “ProductID,”, [li]nsert, ProductName                             |
| IN        | <b>keywordOccurrence</b> | 1,2,-1   |
| OUT       | <b>result</b>            | See Chart 1 below  |

##### 4.2. 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                 |

##### 4.3. 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   | [li]nsert   | 2          | 2          |
| PROCEDURE    | SQL_SCRIPT_PROCEDURE       | 919   | ProductID,  | 1          | 1          |
| PROCEDURE    | SQL_SCRIPT_PROCEDURE       | 2300  | ProductName | 3          | -1         |

## 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   |

### 2. Examples:

#### 2.1. 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' |

| <b>Direction</b> | <b>Parameter Name</b>  | <b>Parameter Type</b> |
|------------------|------------------------|-----------------------|
| 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               |

## 2. Examples:

### 2.1. Assumptions: none

| <b>Direction</b> | <b>Parameter Name</b>  | <b>Parameter Value</b>        |
|------------------|------------------------|-------------------------------|
| 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<br>values: 'INSERT', 'UPDATE', or 'DELETE'                                       | VARCHAR(255)   |
| IN        | defName   | VARCHAR(255)   |
| IN        | defType<br>values: 'EXCEPTION_DEFINTION',<br>'TYPE_DEFINITION', or<br>'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   |                |

### **2. Examples:**

#### **2.1. 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    |

### **2. Examples:**

#### **2.1. 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   |

### **2. Examples:**

#### **2.1. 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  |

### 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            |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name         | Parameter Value            |
|-----------|------------------------|----------------------------|
| IN        | debug                  | 'N'                        |
| IN        | inStartingFolders      | '/shared/myImportedFolder' |
| IN        | inExcludePathsKeywords | NULL                       |
| OUT       | success                | 1                          |

## **updateResourceAnnotations**

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        | columnAnnotations | VECTOR(TypeDefinitions.ColumnAnnotationRow) |
| OUT       | success           | BIT   |
| OUT       | updateResponse    | XML   |
| OUT       | faultResponse     | XML   |

### **2. Examples:**

#### **2.1. 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        | columnAnnotations | (Vector of column specifications and annotations.) |
| 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   |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| OUT       | createResponse | XML            |
| OUT       | faultResponse  | XML            |

## 2. Examples:

### 2.1. 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

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 | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN        | inResourceType | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1)   |
| IN        | inConfigured   | BIT   |
| IN        | inEnabled      | BIT   |
| IN        | inStorageMode  | VARCHAR(255)  |

| <b>Direction</b> | <b>Parameter Name</b>     | <b>Parameter Type</b>   |
|------------------|---------------------------|---|
| IN               | inStorageDataSourcePath   | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN               | inStorageTargetName       | /lib/resource/ResourceDefs.ResourceName (VARCHAR(255) as of CIS 5.1)  |
| IN               | inStorageTargetPath       | /lib/resource/ResourceDefs.ResourcePath (VARCHAR(4096) as of CIS 5.1) |
| IN               | inStorageTargetType       | /lib/resource/ResourceDefs.ResourceType (VARCHAR(40) as of CIS 5.1)   |
| IN               | inRefreshMode             | VARCHAR(255)  |
| IN               | inScheduleMode            | VARCHAR(255)  |
| IN               | inStartTime               | TIMESTAMP   |
| IN               | inInterval                | INTEGER   |
| IN               | inExpirationPeriod        | BIGINT  |
| IN               | inClearRule               | VARCHAR(255)  |
| IN               | inIncremental             | BIT   |
| IN               | inStorageBucketMode       | VARCHAR(255)  |
| IN               | inStorageBucketCatalog    | VARCHAR(255)  |
| IN               | inStorageBucketSchema     | VARCHAR(255)  |
| IN               | inStorageBucketPrefix     | VARCHAR(255)  |
| IN               | inStorageBucketNumBuckets | INTEGER   |
| IN               | inStorageDropCreateIdx    | BIT   |
| IN               | inFirstRefreshCallback    | VARCHAR(32768)  |
| IN               | inSecondRefreshCallback   | VARCHAR(32768)  |
| OUT              | success                   | BIT   |
| OUT              | updateResponse            | XML   |
| OUT              | faultResponse             | XML   |

## 2. Examples:

- 2.1. **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.

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Value</b>                 |
|------------------|-----------------------|--|
| IN               | inResourcePath        | ‘/shared/examples/ds_orders/customers’ |

| <b>Direction</b> | <b>Parameter Name</b>     | <b>Parameter Value</b>                       |
|------------------|---------------------------|--|
| IN               | inResourceType            | 'TABLE'                                      |
| IN               | inConfigured              | 1  |
| IN               | inEnabled                 | 1  |
| IN               | inStorageMode             | 'DATA_SOURCE'                                |
| IN               | inStorageDataSourcePath   | '/shared/examples/ds_orders'                 |
| IN               | inStorageTargetName       | 'result'                                     |
| IN               | inStorageTargetPath       | '/shared/examples/ds_orders/customers_cache' |
| IN               | inStorageTargetType       | 'TABLE'                                      |
| IN               | inRefreshMode             | 'MANUAL'                                     |
| IN               | inScheduleMode            | NULL   |
| IN               | inStartTime               | NULL   |
| IN               | inInterval                | NULL   |
| 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:**

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Type</b>                               |
|------------------|-----------------------|---|
| IN               | fullResourcePath      | /shared/ASAssets/Utilities/TypeDefinitions.pathType |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| IN        | resourceType   | VARCHAR(255)   |
| IN        | cacheEnabled   | VARCHAR(255)   |
| OUT       | success        | BIT            |
| OUT       | updateResponse | XML            |
| OUT       | faultResponse  | XML            |

## 2. Examples:

### 2.1. 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

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     | /shared/ASAssets/Utilities/TypeDefinitions.pathType  |
| IN        | resourceType         | VARCHAR(255)   |
| IN        | resourceSubType      | VARCHAR(255)   |
| IN        | dataSourceType       | VARCHAR(255)   |
| IN        | dataSourceAttrVector | AttributeType ROW (<br>attrName        VARCHAR,<br>attrType        VARCHAR,<br>attrValue      VARCHAR<br>) |
| OUT       | success              | BIT  |
| OUT       | createResponse       | XML  |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| OUT       | faultResponse  | XML            |

## 2. Examples:

### 2.1. 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        | dataSourceAttrVector | {('noNamespaceSchemaLocation','STRING',<br>'file:/CompositeSoftware/Advisory.xsd'),<br>('root','STRING','file:/CompositeSoftware')} |
| OUT       | success              | 1   |
| OUT       | createResponse       | XML not shown here  |
| OUT       | faultResponse        | XML not shown here  |

## updateResourcesSqlTable

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 array of the columns and their types along with an array of the SQL indexes 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 | /shared/ASAssets/Utilities/TypeDefinitions.pathType  |
| IN        | scripttext - SQL Table text to be updated   | LONGVARCHAR  |
| IN        | columnList - a vector array of sql columns and definitions                          | childResourceType ROW (<br>resourceName VARCHAR,<br>resourcePath TypeDefinitions.pathType,<br>resourceType VARCHAR,<br>columnName VARCHAR, |

| Direction | Parameter Name                               | Parameter Type   |
|-----------|--|--|
|           |  | columnType VARCHAR );  |
| IN        | sqlIndexList - a vector array of sql indexes | sqlIndexType ROW (<br>sqlIndexName                   VARCHAR(255),<br>sqlIndexType                   VARCHAR(255),<br>sqlIndexUnique                 BIT,<br>sqlIndexColName              VARCHAR(255),<br>sqlIndexColOrder             VARCHAR(255)<br>); |
| OUT       | success                                      | BIT  |
| OUT       | createResponse                               | XML  |
| OUT       | faultResponse                                | XML  |

## 2. Examples:

### 2.1. 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        | columnList       | ('products','\$products','TABLE','ProductID','INTEGER'),<br>(('products','\$products','TABLE','ProductName','VARCHAR(50)'),<br>(('products','\$products','TABLE','ProductDescription','VARCHAR(255)'),<br>(('products','\$products','TABLE','CategoryID','INTEGER'),<br>(('products','\$products','TABLE','SerialNumber','VARCHAR(50)'),<br>(('products','\$products','TABLE','UnitPrice','DECIMAL(12,2)'),<br>(('products','\$products','TABLE','ReorderLevel','INTEGER'),<br>(('products','\$products','TABLE','LeadTime','VARCHAR(30)') |
| IN        | sqlIndexList     | ('productsPK','PRIMARY_KEY',1,'ProductID','ASCENDING')   |
| OUT       | success          | 1  |
| OUT       | createResponse   | XML not shown here   |

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| OUT       | faultResponse  | XML not shown here |

\$products =  
 /shared/ASAssets/Utilities/repository/examples/source/ds\_inventory/products

## 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   |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value   |
|-----------|------------------|---|
| IN        | fullResourcePath | '/shared/ASAssets/Utilities/repository/examples/source/proc2' |
| IN        | scripttext       | 'PROCEDURE proc2()<br>BEGIN<br>DECLARE var varchar;<br>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 |
|-----------|----------------|----------------|
|           |                |                |

| Direction | Parameter Name   | Parameter Type                                      |
|-----------|------------------|---|
| IN        | fullResourcePath | /shared/ASAssets/Utilities/TypeDefinitions.pathType |
| IN        | inScripttext     | LONGVARCHAR   |
| OUT       | success          | BIT   |
| OUT       | createResponse   | XML   |
| OUT       | faultResponse    | XML   |

## 2. Examples:

### 2.1. 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   |

## 2. Examples:

### 2.1. 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   |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name      | Parameter Value                              |
|-----------|---------------------|--|
| IN        | resourcePath        | '/shared/examples/streamingXSLT'             |
| IN        | transformSourcePath | '/shared/examples/ds_XML/productCatalog.xml' |
| IN        | transformSourceType | 'TREE'                                       |

| Direction | Parameter Name   | Parameter Value                  |
|-----------|------------------|----------------------------------|
| 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(<br>name                    VARCHAR(1000),<br>flags                    VARCHAR(1000),<br>columnMin                DOUBLE,<br>columnMax                DOUBLE,<br>columnDistinct            DOUBLE<br>)) |
| OUT       | responseXML         | XML  |
| OUT       | faultXML            | XML  |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                     |
|-----------|----------------|-------------------------------------|
| IN        | resourcePath   | '/shared/examples/ds_orders/orders' |
| IN        | cardinalityMin | 36                                  |

| Direction | Parameter Name      | Parameter Value |
|-----------|---------------------|-----------------|
| 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   |

### 2. Examples:

#### 2.1. 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   |

### **2. Examples:**

#### **2.1. 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 |

### **2. Examples:**

#### **2.1. 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 |

## 2. Examples:

### 2.1. 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) |

## 2. Examples:

### 2.1. Assumptions: Dependency on configureReservedList

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| IN        | inString       | 'P4ssw0rd'         |
| OUT       | result         | '5AC41E7EC9AC80CE' |

## 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) |

| Direction | Parameter Name | Parameter Type   |
|-----------|----------------|--|
| OUT       | result         | <pre>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 )</pre> |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| IN        | datasourcePath | '/shared/examples/ds_orders'  |
| IN        | cisType        | 'LONGVARCHAR'   |
| OUT       | result         | <pre>(   'LONGVARCHAR',   'LONGVARCHAR',   'LONGVARCHAR',   -1,   -1,   -983,   'LONGVARCHAR',   'varchar(2147483647)',   'varchar',   '2147483647',   '-1' )</pre> |

## 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 ) |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
|           | )              |                |

## 2. Examples:

### 2.1. 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(<br>Property VARCHAR(2147483647),<br>Value    VARCHAR(2147483647)<br>) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| OUT       | result         | (‘path.separator’, ‘;’),<br>(‘user.home’, ‘C:\Users\bob’),<br>... |

## 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(<br>DomainName VARCHAR(2147483647),<br>GroupName  VARCHAR(2147483647)<br>) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                                 |
|-----------|----------------|---|
| IN        | DomainName     | 'composite'                                     |
| IN        | UserName       | 'admin'   |
| OUT       | result         | ('composite', 'all'),<br>(‘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                           |

## 2. Examples:

### 2.1. 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 |
|-----------|----------------|----------------|
|           |                |                |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| IN        | column_path    | LONGVARCHAR    |
| IN        | annotation     | LONGVARCHAR    |
| OUT       | result         | LONGVARCHAR    |

## 2. Examples:

### 2.1. 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<br>(VARCHAR(4096)) |
| IN        | resourceParams | LONGVARCHAR  |

#### 2. Examples:

##### 2.1. 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   |

**2. Examples:**

**2.1. 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)              |

**2. Examples:**

**2.1. Assumptions: Calling user has MODIFY\_ALL\_CONFIG right**

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| IN        | debug          | 'N'                |
| IN        | licenseText    | XML not shown here |

### server/getServerAttribute (Custom Function)

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    |

#### 2. Examples:

##### 2.1. Assumptions: Calling user has READ\_ALL\_CONFIG right

| Direction | Parameter Name | Parameter Value                         |
|-----------|----------------|---|
| IN        | absolutePath   | '/server/webservices/timezoneBaseOnGMT' |
| OUT       | keyValue       | 'true'                                  |

### server/getServerAttributeList

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 ) |

#### 2. Examples:

##### 2.1. Assumptions: Calling user has READ\_ALL\_CONFIG right

| Direction | Parameter Name | Parameter Value            |
|-----------|----------------|----------------------------|
| IN        | absolutePath   | '/studio/data/examplelist' |
| OUT       | result         | (no rows)                  |

## **server/getServerAttributeMap**

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,<br>keyValue LONGVARCHAR ) |

### **2. Examples:**

#### **2.1. Assumptions: Calling user has READ\_ALL\_CONFIG right**

| Direction | Parameter Name | Parameter Value           |
|-----------|----------------|---------------------------|
| IN        | absolutePath   | '/studio/data/examplemap' |
| OUT       | result         | (no rows)                 |

## **server/getServerAttributeMapByKey (Custom Function)**

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    |

### **2. Examples:**

#### **2.1. 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/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)              |

### **2. Examples:**

#### **2.1. Assumptions: Calling user has MODIFY\_ALL\_CONFIG right**

| Direction | Parameter Name | Parameter Value                      |
|-----------|----------------|--------------------------------------|
| IN        | debug          | 'N'                                  |
| IN        | ATTR           | '/server/sql/language/caseSensitive' |

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | ATTR_TYPE      | 'BOOLEAN'       |
| IN        | NEWVAL         | 'true'          |

## 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)               |

#### 2. Examples:

##### 2.1. 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)  |

### 2. Examples:

#### 2.1. 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                |

## 2. Examples:

### 2.1. 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)               |

## 2. Examples:

### 2.1. 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)               |

### **2. Examples:**

#### **2.1. 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 (<br>name VARCHAR(32768),<br>domainName VARCHAR(32768),<br>id INTEGER,<br>explicitRights VARCHAR(32768),<br>effectiveRights VARCHAR(32768),<br>inheritedRights VARCHAR(32768),<br>annotation VARCHAR(32768)<br>) |

### **2. Examples:**

#### **2.1. Assumptions: Calling user has READ\_ALL\_USERS right**

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | inDomainName   | 'composite'     |

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| OUT       | result         | (<br>'admin',<br>'composite',<br>1,<br>'ACCESS_TOOLS, ...',<br>'ACCESS_TOOLS, ...',<br>'NONE',<br>'Administrator group'<br>),<br>... |

## 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)   |

### 2. Examples:

#### 2.1. 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 MODIFY_ALL_CONFIG<br>MODIFY_ALL_RESOURCES<br>MODIFY_ALL_STATUS<br>MODIFY_ALL_USERS<br>READ_ALL_CONFIG<br>READ_ALL_RESOURCES<br>READ_ALL_STATUS READ_ALL_USERS' |

| Direction | Parameter Name       | Parameter Value   |
|-----------|----------------------|---|
|           |                      | 'UNLOCK_RESOURCE'   |
| OUT       | groupEffectiveRights | 'ACCESS_TOOLS MODIFY_ALL_CONFIG<br>MODIFY_ALL_RESOURCES<br>MODIFY_ALL_STATUS<br>MODIFY_ALL_USERS<br>READ_ALL_CONFIG<br>READ_ALL_RESOURCES<br>READ_ALL_STATUS READ_ALL_USERS<br>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)   |

#### **2. Examples:**

##### **2.1. 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 MODIFY_ALL_CONFIG<br>MODIFY_ALL_RESOURCES<br>MODIFY_ALL_STATUS<br>MODIFY_ALL_USERS<br>READ_ALL_CONFIG<br>READ_ALL_RESOURCES' |

| <b>Direction</b> | <b>Parameter Name</b> | <b>Parameter Value</b>  |
|------------------|-----------------------|---|
|                  |                       | 'READ_ALL_STATUS READ_ALL_USERS<br>UNLOCK_RESOURCE'   |
| OUT              | userEffectiveRights   | 'ACCESS_TOOLS MODIFY_ALL_CONFIG<br>MODIFY_ALL_RESOURCES<br>MODIFY_ALL_STATUS<br>MODIFY_ALL_USERS<br>READ_ALL_CONFIG<br>READ_ALL_RESOURCES<br>READ_ALL_STATUS READ_ALL_USERS<br>UNLOCK_RESOURCE' |
| OUT              | userInheritedRights   | 'ACCESS_TOOLS MODIFY_ALL_CONFIG<br>MODIFY_ALL_RESOURCES<br>MODIFY_ALL_STATUS<br>MODIFY_ALL_USERS<br>READ_ALL_CONFIG<br>READ_ALL_RESOURCES<br>READ_ALL_STATUS READ_ALL_USERS<br>UNLOCK_RESOURCE' |

## HOW TO USE 'REQUEST' PROCEDURES

### ***Introduction***

This section describes the routines for examining the SQL of calling requests.

#### **terminatIdleSessions**

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,<br>success BIT,<br>faultXML XML<br>) |

##### **2. Examples:**

###### **2.1. 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            |

##### **2. Examples:**

###### **2.1. 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            |

### 2. Examples:

#### 2.1. 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 |
|-----------|----------------|----------------|
|           |                |                |

| Direction | Parameter Name | Parameter Type      |
|-----------|----------------|---------------------|
| OUT       | requestSrc     | VARCHAR(2147483647) |

## 2. Examples:

### 2.1. 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) |

## 2. Examples:

### 2.1. 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) |

| Direction | Parameter Name | Parameter Type                            |
|-----------|----------------|---|
| OUT       | result         | CURSOR(<br>value VARCHAR(2147483647)<br>) |

## 2. Examples:

### 2.1. 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'),<br>(‘2’),<br>(‘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) |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| OUT       | requestSrc     | 'SELECT * FROM<br>/shared/examples/procedure_that_eventually_calls_TopSqlRequest()' |

## HOW TO USE 'STRING' PROCEDURES

### ***Introduction***

This section will show how to use the 'String' manipulation procedures.

#### **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             |

##### **2. Examples:**

###### **2.1. 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 |

| Direction | Parameter Name | Parameter Type  |
|-----------|----------------|---|
|           |                | (VARCHAR(4096) as of CIS 5.1)   |
| OUT       | outputString   | /lib/resource/ResourceDefs.ResourceName<br>(VARCHAR(255) as of CIS 5.1) |

## 2. Examples:

### 2.1. 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    |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|           |                |                 |

| 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) |

#### 2. Examples:

##### 2.1. 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    |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | inputString    | null            |

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| 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:

- 1.1. **Assumptions:** Depends on 4 vectors defined as constants define in the procedure string/entityConstants(). Review that procedure and make modifications as required.
  - 1.1.1. **symbols1Vector** – Single value symbols and punctuation such as '!,@,#,\$,%' and etc. that get replaced with a non-null, empty character ''.
  - 1.1.2. **symbols2Vector** – Multi-character value symbols such as ' - ' that get replaced with a single space, ''.
  - 1.1.3. **symbols3Vector** – Hidden character symbols such as tabs that get replaced with a single space, ''.
  - 1.1.4. **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<br>any text.  | LONGVARCHAR    |
| IN        | inDelimiter<br>space, pipe or some single character delimiter                      | CHAR(1)        |
| IN        | inNumWords<br>the number of words to produce. If 0 or null then produce all words. | INTEGER        |
| IN        | random<br>true(1)=random-produce a list randomly                                   | BIT            |

| Direction | Parameter Name  | Parameter Type  |
|-----------|---|-----------------|
|           | from the incoming text<br>false(0)=sequential-produce a list sequentially from the incoming text  |                 |
| IN        | removeSymbols<br>true(1)=remove symbols and punctuation prior to extraction<br>false(0)=do not remove symbols and punctuation prior to extraction | BIT             |
| OUT       | entityVector  | VECTOR(VARCHAR) |

## 2. 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<br>any text.  | LONGVARCHAR    |
| IN        | inDelimiter<br>space, pipe or some single character delimiter                      | CHAR(1)        |
| IN        | inNumWords<br>the number of words to produce. If 0 or null then produce all words. | INTEGER        |
| IN        | random<br>true(1)=random-produce a list randomly from the incoming text            | BIT            |

| Direction | Parameter Name  | Parameter Type             |
|-----------|---|----------------------------|
|           | false(0)=sequential-produce a list sequentially from the incoming text  |                            |
| IN        | removeSymbols<br>true(1)=remove symbols and punctuation prior to extraction<br>false(0)=do not remove symbols and punctuation prior to extraction | BIT                        |
| OUT       | keywordsPipe  | Pipe (keyword LONGVARCHAR) |

## 2. Examples:

### 2.1. 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<br>string<br>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<br>any text   | LONGVARCHAR    |
| IN        | inDelimiter<br>space, pipe or some single character delimiter                      | CHAR(1)        |
| IN        | inNumWords<br>the number of words to produce. If 0 or null then produce all words. | INTEGER        |

| Direction | Parameter Name  | Parameter Type |
|-----------|---|----------------|
| IN        | random<br>true(1)=random-produce a list randomly from the incoming text<br>false(0)=sequential-produce a list sequentially from the incoming text | BIT            |
| IN        | removeSymbols<br>true(1)=remove symbols and punctuation prior to extraction<br>false(0)=do not remove symbols and punctuation prior to extraction | BIT            |
| OUT       | outWords<br>comma separated list of words   | LONGVARCHAR    |

## 2. Examples:

### 2.1. 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    |

## 2. Examples:

### 2.1. 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  | VARCHAR        |
|           | a word with no spaces that represents the keyword search term to locate   |                |
| IN        | openingDelim  | VARCHAR        |
|           | The beginning delimiter which directly follows the searchTerm. Allowed: Single: '[', '(', '{', '<' Doubles: '[]', '()', '{}', '<<'  |                |
| IN        | closingDelim  | VARCHAR        |
|           | The ending delimiter which directly follows the content. Allowed: Single: ']', ')', '}', '>' Doubles: ']]', ')()', '}}', '>>'   |                |
| IN        | inOccurrence  | INTEGER        |
|           | 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. |                |
| IN        | trimText  | INTEGER        |
|           | 0=do not trim result, 1=do trim result text (default=0)   |                |

| Direction | Parameter Name  | Parameter Type |
|-----------|---|----------------|
| IN        | caseSensitive<br>0=no search term case sensitivity, 1=case sensitive search term. | INTEGER        |
| OUT       | result  | LONGVARCHAR    |

## 2. Examples:

### 2.1. 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         |  |

## 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<br><br>The separator value (typically a comma) that will be used to define the boundary of text expressions. Because 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 | VARCHAR        |

| <b>Direction</b> | <b>Parameter Name</b>  | <b>Parameter Type</b>               |
|------------------|--|-------------------------------------|
|                  | separator escaped with a backslash [\\] - this is a caret separator escaped with a backslash.  |                                     |
| IN               | preserveDoubleQuotes<br><br>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<br><br>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<br><br>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<br><br>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:<br><br>"orders,customers"<br><br>orders<br><br>This example would not remove the qualifier:<br>text "more text" text,text<br><br>text "more text" text<br><br>text | BIT                                 |
| IN               | trimResults<br><br>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<br>LONGVARCHAR) |

## 2. Examples:

### 2.1. 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"<br>orders<br>customers |

### 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        |

## 2. Examples:

### 2.1. 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        |

## 2. Examples:

### 2.1. 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    |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value           |
|-----------|----------------|---------------------------|
| IN        | inMessage      | this is a 'quoted' string |
| OUT       | outMessage     | this is a "quoted" string |

### getConstant (Custom Function)

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              |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                                      |
|-----------|----------------|--|
| IN        | constantPath   | '/shared/ASAssets/Utilities/documentation/constants' |
| IN        | constantName   | 'resourcePath'                                       |
| 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    |

#### 2. Examples:

##### 2.1. 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 |
|-----------|----------------|----------------|
|           |                |                |

| Direction | Parameter Name       | Parameter Type |
|-----------|----------------------|----------------|
| IN        | inText               | LONGVARCHAR    |
| IN        | delimiter            | VARCHAR        |
| IN        | inStartingOccurrence | INTEGER        |
| OUT       | result               | DECIMAL(32,2)  |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name       | Parameter Value             |
|-----------|----------------------|-----------------------------|
| IN        | inText               | '1, 2, 3, 4, 5, 6, 7, 8, 9' |
| IN        | delimiter            | ','                         |
| IN        | inStartingOccurrence | 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    |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                            |
|-----------|----------------|--|
| IN        | depthNum       | 2 [the depth is 2 – 2 <sup>nd</sup> level] |
| IN        | indent         | ‘ ’ [4 spaces]                             |
|           | inValue        | ‘Text’ [the word Text]                     |
| OUT       | outValue       | ‘ Text’ [8 spaces precede the word 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            |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | request        | ''              |
| OUT       | response       | 1               |

## **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)   |

### **2. Examples:**

#### **2.1. 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    |

## 2. Examples:

### 2.1. 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)  |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value                       |
|-----------|----------------|---------------------------------------|
| IN        | curs           | {‘Words’,<br>‘Extracted’,<br>‘Words’} |
| OUT       | pipeStr        | {‘Words’,<br>‘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         | LONGVARCHAR         |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value                       |
|-----------|----------------|---------------------------------------|
| IN        | curs           | {‘Words’,<br>‘Extracted’,<br>‘Words’} |

| Direction | Parameter Name | Parameter Value    |
|-----------|----------------|--------------------|
| OUT       | result         | 'Words, Extracted' |

### **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 ) |

#### **2. Examples:**

##### **2.1. 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 ) |

#### **2. Examples:**

## 2.1. 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))     |

### 2. Examples:

#### 2.1. 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    |

### 2. Examples:

## 2.1. 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    |

### 2. Examples:

## 2.1. 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 |
|---------|---------|-----------------|
|         |         |                 |

| 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)<br>2 – Area code<br>3 – Central office code<br>4 – Subscriber number |
| XML_DATE_TIME                             | Validates or parses an XML dateTime string   | 1 – Date component<br>2 – Time component<br>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<br><br>any single delimiter character.<br>Values: Common separators include: ' | CHAR(1)        |

| Direction | Parameter Name     | Parameter Type       |
|-----------|--------------------|----------------------|
|           | ' or '' or ','     |                      |
| IN        | debug - Y/N or T/F | CHAR(1)              |
| OUT       | outString          | VECTOR(VARCHAR(255)) |

## 2. Examples:

### 2.1. 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    |

## 2. Examples:

### 2.1. 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) |

## 2. Examples:

### 2.1. 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) |

## 2. Examples:

### 2.1. 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<br>1, UPS Ground'        |

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|           |                | ...'            |

## 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             |

### 2. Examples:

#### 2.1. 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) |

### 2. Examples:

#### 2.1. 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<br>1 UPS Ground<br>...’                  |

## 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             |

### 2. Examples:

#### 2.1. 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) |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                                 |
|-----------|----------------|---|
| IN        | UnformattedXML | '<a><b>b text</b><c>c text</c></a>'             |
| OUT       | FormattedXML   | '<a><br><b>b text</b><br><c>c text</c><br></a>' |

## TextUtils/GenerateGUID

Generates a random GUID value.

### 1. Parameters:

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

### 2. Examples:

#### 2.1. 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  |

**2. Examples:**

**2.1. 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<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )   | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see <a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue   | VARCHAR        |

**2. Examples:**

**2.1. Assumptions: none**

| Direction | Parameter Name   | Parameter Value |
|-----------|------------------|-----------------|
| IN        | inValue          | 12345.6789      |
| IN        | inFractionLength | 2               |

| Direction | Parameter Name     | Parameter Value |
|-----------|--------------------|-----------------|
| 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<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )   | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see <a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue   | DOUBLE         |

#### 2. Examples:

##### 2.1. 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           |

| Direction | Parameter Name   | Parameter Type |
|-----------|--|----------------|
| IN        | inStyle<br>‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’  | VARCHAR        |
| IN        | ISO639LangCode<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )   | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see <a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue   | VARCHAR        |

## 2. Examples:

### 2.1. 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<br>‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’  | VARCHAR        |
| IN        | ISO639LangCode<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )   | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see <a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| OUT       | outValue       | DATE           |

## 2. Examples:

### 2.1. 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<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )   | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see <a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue   | VARCHAR        |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value |
|-----------|------------------|-----------------|
| IN        | inValue          | 12345.6789      |
| IN        | inFractionLength | 2               |
| IN        | ISO639LangCode   | 'EN'            |

| Direction | Parameter Name     | Parameter Value |
|-----------|--------------------|-----------------|
| 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<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )   | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see <a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue   | DOUBLE         |

#### 2. Examples:

##### 2.1. 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        | VARCHAR        |

| Direction | Parameter Name  | Parameter Type |
|-----------|---|----------------|
|           | ‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’  |                |
| IN        | ISO639LangCode<br><br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )  | VARCHAR        |
| IN        | ISO3166CountryCode<br><br>(see<br><a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue  | VARCHAR        |

## 2. Examples:

### 2.1. 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<br><br>‘SHORT’, ‘MEDIUM’, ‘LONG’, or ‘FULL’   | VARCHAR        |
| IN        | ISO639LangCode<br><br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )  | VARCHAR        |
| IN        | ISO3166CountryCode<br><br>(see<br><a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue  | TIME           |

## 2. Examples:

### 2.1. 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<br>'SHORT', 'MEDIUM', 'LONG', or 'FULL'   | VARCHAR        |
| IN        | inTimeStyle<br>'SHORT', 'MEDIUM', 'LONG', or 'FULL'   | VARCHAR        |
| IN        | ISO639LangCode<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )  | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see<br><a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue  | VARCHAR        |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value     |
|-----------|----------------|---------------------|
| IN        | inValue        | 2010-08-10 12:34:56 |
| IN        | inDateStyle    | 'FULL'              |

| Direction | Parameter Name     | Parameter Value                               |
|-----------|--------------------|---|
| IN        | inTimeStyle        | 'FULL'  |
| IN        | ISO639LangCode     | 'EN'  |
| IN        | ISO3166CountryCode | 'US'  |
| OUT       | outValue           | 'Tuesday, August 10, 2010<br>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<br>'SHORT', 'MEDIUM', 'LONG', or 'FULL'   | VARCHAR        |
| IN        | inTimeStyle<br>'SHORT', 'MEDIUM', 'LONG', or 'FULL'   | VARCHAR        |
| IN        | ISO639LangCode<br>(see <a href="http://www.loc.gov/standards/iso639-2/php/English_list.php">http://www.loc.gov/standards/iso639-2/php/English_list.php</a> )  | VARCHAR        |
| IN        | ISO3166CountryCode<br>(see<br><a href="http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm">http://www.iso.org/iso/country_codes/iso_3166_code_lists/english_country_names_and_code_elements.htm</a> ) | VARCHAR        |
| OUT       | outValue  | TIMESTAMP      |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value                               |
|-----------|--------------------|---|
| IN        | inValue            | 'Tuesday, August 10, 2010<br>12:34:56 PM PDT' |
| IN        | inDateStyle        | 'FULL'  |
| IN        | inTimeStyle        | 'FULL'  |
| IN        | ISO639LangCode     | 'EN'  |
| IN        | ISO3166CountryCode | 'US'  |

| Direction | Parameter Name | Parameter Value     |
|-----------|----------------|---------------------|
| 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) |

#### 2. Examples:

##### 2.1. 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    |

**2. Examples:**

**2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | rawVal         | 1F2E3D4C        |
| OUT       | hexVal         | '1f2e3d4c'      |

### 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        |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| IN        | Occurrence     | INTEGER        |
| OUT       | result         | VARCHAR        |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value                 |
|-----------|--------------------|---------------------------------|
| IN        | Input Text         | 'abaabaaaabaaaa'                |
| 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(<br>groupNumber INTEGER,<br>matchedGroup VARCHAR<br>) |

## 2. Examples:

## 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value   |
|-----------|--------------------|---|
| IN        | Input Text         | '(650) 227-8200'  |
| IN        | Regular Expression | '^(?(\d{3}))\)?[\s\-\-]*(\d{3})[\s\-\-]* (\d{4})'<br>Parses a U.S. phone number in several formats. |
| IN        | Occurrence         | 1   |
| OUT       | result             | (0, '(650) 227-8200'),<br>(1, '650'),<br>(2, '227'),<br>(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        |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value            |
|-----------|--------------------|----------------------------|
| IN        | Input Text         | 'abaabaaaabaaaa'           |
| IN        | Regular Expression | 'a+' (matches at least one |

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|           |                | '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        |

#### 2. Examples:

##### 2.1. 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 |

| Direction | Parameter Name | Parameter Value     |
|-----------|----------------|---------------------|
|           |                | 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(<br>splitElement VARCHAR<br>) |

#### 2. Examples:

##### 2.1. 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'),<br>(‘aa’),<br>(‘aaa’),<br>(‘aaaa’)        |

#### TextUtils/SSNumberFormatter (Custom Function)

Provides standard formatting of a Social Security number.

#### 3. Parameters:

| Direction | Parameter Name | Parameter Type      |
|-----------|----------------|---------------------|
| IN        | inSSNumber     | VARCHAR(2147483647) |
| OUT       | outSSNumber    | VARCHAR(2147483647) |

#### 4. Examples:

##### 4.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | inSSNumber     | '111223456'     |
| OUT       | outSSNumber    | '111-22-3456'   |

## 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

## HOW TO USE 'TIME' PROCEDURES

### ***Introduction***

This section will show how to use the 'Time' manipulation procedures.

#### **ADD\_MONTHS (Custom Function)**

This procedure takes an input timestamp and adds (or subtracts) an input number of months to it. If any of the inputs are null a null will be returned.

##### **1. Parameters:**

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| IN        | startTimestamp | TIMESTAMP      |
| IN        | numMonths      | INTEGER        |
| OUT       | result         | TIMESTAMP      |

##### **2. Examples:**

###### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value       |
|-----------|----------------|-----------------------|
| IN        | startTimestamp | '2011-03-01 00:00:00' |
| IN        | numMonths      | -2                    |
| OUT       | result         | '2011-01-01 00:00:00' |

### **DefaultValues**

This procedure contains a vector of valid date, time and timestamp formats which is used by `time/extractDate()`, `time/extractTime()`, and `time/extractTimestamp()` to extract a date, time or timestamp from a string. Add additional formats to the vector as needed. Place the most common formats towards the top of the vector so that the extraction procedure works as efficiently as it can.

##### **1. Parameters: none**

##### **2. Examples: none**

#### **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. The time formats supported in this procedure are defined as constants in `time/DefaultValues()`.

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<br>Y/N or T/F | CHAR(1)        |
| OUT       | extractDate         | DATE           |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value |
|-----------|--|-----------------|
| IN        | inDateStr  | Aug 7, 2010     |
| IN        | isMandatory<br>0 means not mandatory,<br>1 means mandatory.  | 0               |
| IN        | debug  | Y               |
| OUT       | extractDate<br><br>If isMandatory=0 and no format could be found then return null.<br><br>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. The time formats supported in this procedure are defined as constants in `time/DefaultValues()`.

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       |

| Direction | Parameter Name      | Parameter Type |
|-----------|---------------------|----------------|
| IN        | debug<br>Y/N or T/F | CHAR(1)        |
| OUT       | extractTime         | TIMESTAMP      |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name   | Parameter Value |
|-----------|--|-----------------|
| IN        | inTimeStr  | '12:00:00'      |
| IN        | isMandatory<br>0 means not mandatory,<br>1 means mandatory.  | 0               |
| IN        | debug  | Y               |
| OUT       | extractTimeIf isMandatory=0 and no<br>format could be found then return null.<br><br>If isMandatory=1 and no format could be<br>found then an exception is thrown. | 12:00:00        |

## extractTimestamp (Custom Function)

Extract a timestamp from a string using a vector of default timestamp formats found in `time/DefaultValues()`. 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<br>Y/N or T/F | CHAR(1)        |
| OUT       | extractTimestamp    | TIMESTAMP      |

## 2. Examples:

## 2.1. Assumptions: none

| Direction | Parameter Name  | Parameter Value            |
|-----------|---|----------------------------|
| IN        | inTimestampStr  | 'Aug 7, 2010 12:00:00.101' |
| IN        | isMandatory<br>0 means not mandatory,<br>1 means mandatory.   | 0                          |
| IN        | debug   | Y                          |
| OUT       | extractTimestamp<br><br>If isMandatory=0 and no format could be found then return null.<br><br>If isMandatory=1 and no format could be found then an exception is thrown. | 2010-08-07 12:00:00.101    |

## 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<br><br>measured in hours | INTEGER        |
| IN        | inTimestamp                         | TIMESTAMP      |
| OUT       | endDatetime                         | TIMESTAMP      |

### 2. Examples:

#### 2.1. 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                 |

## 2. Examples:

### 2.1. 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 |

## 2. Examples:

### 2.1. 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.

## **BignumToTimestamp (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        | inBignum       | BIGINT         |
| OUT       | result         | TIMESTAMP      |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value                                       |
|-----------|----------------|---|
| IN        | inBignum       | 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           |

### **2. Examples:**

#### **2.1. 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      |

### 2. Examples:

#### 2.1. 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        |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
|           |                |                 |

| 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        |

#### 2. Examples:

##### 2.1. 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        |

## 2. Examples:

### 2.1. 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         |

## 2. Examples:

### 2.1. 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      |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name  | Parameter Value       |
|-----------|-----------------|-----------------------|
| IN        | sourceTimestamp | ‘2010-01-01 00:00:00’ |
| IN        | fromTimeZone    | ‘UTC’                 |
| IN        | toTimeZone      | ‘PST8PDT’             |
| OUT       | targetTimestamp | ‘2009-12-31 16:00:00’ |

## 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<br>(VARCHAR(2147483647)) |

##### **2. Examples:**

###### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value       |
|-----------|----------------|-----------------------|
| IN        | inStr          | XML('this is a test') |
| OUT       | outStr         | 'this is a test'      |

###### **2.2. 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'            |

#### **CreateXmlString2CursorXForm**

This script can be used to create XSLT transforms that take a VARCHAR (with XML content) as input instead of an XML data source. The iXSLT input variable takes an XSLT document that produces the same XML as the standard XSLT transform does. In fact, the easiest way to create the XSLT is probably to use an XSLT modeler to create a transformation against a sample XML flat file then copy and paste the XSLT code.

The iColumnDefList input variable takes a vector of column definition rows (see the ColumnDefRow and ColumnDefList type definitions below.) This describes the transformation's output cursor. See

`xml/examples/createProcedureFromProductCatalogXForm()` for an example of how this script is used.

### 1. Parameters:

| Direction | Parameter Name | Parameter Type                          |
|-----------|----------------|---|
| IN        | iXFormProcName | /lib/resource/ResourceDefs.ResourceName |
| IN        | iParentFolder  | /lib/resource/ResourceDefs.ResourcePath |
| IN        | iXSLT          | XML                                     |
| IN        | iColumnDefList | ColumnDefList                           |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| IN        | iXFormProcName | 'ProductCatalogTransform'   |
| IN        | iParentFolder  | '/shared/ASAssets/Utilities/"xml"/examples'   |
| IN        | iXSLT          | XSLT script not shown here. See examples folder.  |
| IN        | iColumnDefList | VECTOR[<br>('ProductID', 'INTEGER'),<br>('ProductName', 'VARCHAR(32768)'),<br>('CategoryID', 'INTEGER'),<br>('CategoryName', 'VARCHAR(32768)'),<br>('ProductDescription',<br>'VARCHAR(32768)'),<br>('SerialNumber', 'VARCHAR(32768)'),<br>('UnitPrice', 'DECIMAL(32,2)'),<br>('ReorderLevel', 'INTEGER'),<br>('LastReorder', 'TIMESTAMP'),<br>('LeadTime', 'VARCHAR(32768)')] |

### escapeXML (Custom Function)

Change xml tags to their corresponding escape sequences.

### 1. Parameters:

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
|           |                |                |

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| IN        | inStr          | LONGVARCHAR    |
| OUT       | outStr         | LONGVARCHAR    |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value                    |
|-----------|----------------|------------------------------------|
| IN        | inStr          | '<tag>somevalue</tag>'             |
| OUT       | outStr         | '&lt;tag&gt;somevalue&lt;/tag&gt;' |

## 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    |

## 2. Examples:

### 2.1. 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'  |

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | iteration      | 1  |
| IN        | inXml          | '<Book<br>xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"><br><sam:Name sam:isbn="12-3456-123">Test</sam:Name><br><Chapter>Test Data</Chapter><br></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    |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | debug          | 'N'  |
| IN        | namespaces     | 'xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"<br>xmlns:xsd="http://www.w3.org/2001/XMLSchema"<br>xmlns:xlink="http://www.w3.org/1999/xlink"'            |
| IN        | inXpath        | '/Book/sam:Name/@sam:isbn'   |
| IN        | inXml          | '<Book<br>xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"><br><sam:Name sam:isbn="12-3456-123">Test</sam:Name><br><Chapter>Test Data</Chapter><br></Book>' |
| OUT       | outValue       | '12-3456-123'  |

## **parseAndModifyXML**

Parse and XML document given a vector 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        | ElemAttrVector     | VECTOR<br>(/shared/ASAssets/Utilities/TypeDefinitions.ElemAttrValuesType) |
| IN        | inXml              | XML   |
| OUT       | outXml             | XML   |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| IN        | debug          | 'N'   |
| IN        | namespaces     | 'xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"<br>xmlns:xsd="http://www.w3.org/2001/XMLSchema"<br>xmlns:xlink="http://www.w3.org/1999/xlink"' |
| IN        | ElemAttrVector | {   |

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
|           |                | <pre>-- Use Case 1: Modify the attribute for an element ('sam:Name','sam:isbn','00-0000-000'), -- Use Case 2: Modify the value of the the element with attributes ('sam:Name',NULL,'Joe Author'), -- Use Case 3: Modify the element with no attributes ('Chapter',NULL,'Chapter 1'), -- Use Case 4: Modify the element with no attributes and set it to null ('Paragraph',NULL,NULL), -- Use Case 5: Retain the attribute ('sam:Author','sam:firstname','Joe'), -- Use Case 5: Retain the attribute ('sam:Author','sam:lastname','Author'), -- Use Case 5: Modify the element with attributes and set it to null ('sam:Author',NULL,NULL) }</pre> |
| IN        | inXml          | <pre>'&lt;Book xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"&gt; &lt;sam:Name sam:isbn="12-3456-123"/&gt; &lt;Chapter&gt;Test Data&lt;/Chapter&gt; &lt;Paragraph&gt;Paragraph 1&lt;/Paragraph&gt; &lt;/Book&gt;'</pre>  |
| OUT       | outXml         | <pre>'&lt;Book xmlns:sam="http://www.compositesw.com/samples/mynamespace/v1.0"&gt;  &lt;sam:Name sam:isbn="00-0000-000"&gt;Joe Author&lt;/sam:Name&gt; &lt;Chapter&gt;Chapter 1&lt;/Chapter&gt; &lt;Paragraph xmlns:xsi="http://www.w3.org/2001/XMLSchema- instance" xsi:nil="true"/&gt; &lt;/Book&gt;'</pre>   |

## pruneXML

Prune out empty XML nodes except those in an retainXPathVector list.

### 1. Parameters:

| Direction | Parameter Name                         | Parameter Type |
|-----------|--|----------------|
| IN        | Debug - Y/N or T/F                     | CHAR(1)        |
| IN        | namespaces - any string with the valid | LONGVARCHAR    |

| Direction | Parameter Name  | Parameter Type        |
|-----------|---|-----------------------|
|           | namespaces for the incoming XML   |                       |
| IN        | retainXPathVector - contains a vector 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 vector. The invoker will determine the XPath statement by consulting the XML Schema for required elements and the determine 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'} | VECTOR(varchar(2048)) |
| IN        | inXml – any valid XML document  | XML                   |
| OUT       | outXml – any valid XML document   | XML                   |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name    | Parameter Value  |
|-----------|-------------------|--|
| IN        | debug             | 'N'  |
| IN        | namespaces        | 'xmlns:tns="http://mynamespace"<br>xmlns:xsd="http://www.w3.org/2001/XMLSchema"<br>xmlns:xlink="http://www.w3.org/1999/xlink"'           |
| IN        | retainXPathVector | {'/rootnode/XYZ/A/@foo','/rootnode/XYZ/C'}   |
| IN        | inXml             | '<rootnode xmlns:tns="http://mynamespace"><br><XYZ abc="def" ghi=""><br><A foo="">aa</A><br><B>bbb</B><br><C/><br></XYZ><br></rootnode>' |
| OUT       | outXml            | '<rootnode xmlns:tns="http://mynamespace"><br><XYZ abc="def"><br><A foo="">aa</A><br><B>bbb</B><br><C/><br></XYZ><br></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            |

### 2. Examples:

#### 2.1. 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            |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | inXml          |                 |
| OUT       | outXml         |                 |

## unescapeXML (Custom Function)

Change xml entities to their corresponding individual characters.

### 1. Parameters:

| Direction | Parameter Name | Parameter Type |
|-----------|----------------|----------------|
| IN        | inStr          | LONGVARCHAR    |
| OUT       | outStr         | LONGVARCHAR    |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value |
|-----------|----------------|-----------------|
| IN        | inStr          | 'A & B < C & D' |
| 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             |

## 2. Examples:

### 2.1. 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'                |

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|-------------------|
| 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) |

### 2. Examples:

#### 2.1. Assumptions: none

| Direction | Parameter Name     | Parameter Value   |
|-----------|--------------------|---|
| IN        | inXML              | '<xmldoc xmlns="uri:mynamespace"><br><parent><br><child><br><grandchild/><br></child><br></parent><br></xmldoc> |
| IN        | inElementName      | 'child'   |
| IN        | inElementNamespace | 'uri:mynamespace'   |
| IN        | occurrence         | 1   |
| OUT       | result             | '<xmldoc xmlns="uri:mynamespace"><br><parent/><br></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) |

### **2. Examples:**

#### **2.1. Assumptions: none**

| Direction | Parameter Name     | Parameter Value  |
|-----------|--------------------|--|
| IN        | inXML              | '<xmldoc xmlns="uri:mynamespace"><br><parent><br><child><br><grandchild/><br></child><br></parent><br></xmldoc>' |
| IN        | inElementName      | 'child'  |
| IN        | inElementNamespace | 'uri:mynamespace'  |
| IN        | occurrence         | 1  |
| OUT       | result             | '<xmldoc xmlns="uri:mynamespace"><br><parent><br><grandchild/><br></parent><br></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 |
|-----------|----------------|----------------|
|-----------|----------------|----------------|

| 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             |

## 2. Examples:

### 2.1. 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    |

## 2. Examples:

### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| IN        | inHTML         | '<html><body><table><tr><td></td></tr></table></body></html>' |

| Direction | Parameter Name | Parameter Value   |
|-----------|----------------|---|
| 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) |

#### 2. Examples:

##### 2.1. Assumptions: none

| Direction | Parameter Name | Parameter Value  |
|-----------|----------------|--|
| IN        | inXML          | '<xmldoc xmlns="uri:mynamespace"> <parent> <grandchild/> </parent> |

| <b>Direction</b> | <b>Parameter Name</b>  | <b>Parameter Value</b>  |
|------------------|------------------------|---|
|                  |                        | </xml/doc>'   |
| IN               | parentElementName      | 'parent'  |
| IN               | parentElementNamespace | 'uri:mynamespace'   |
| IN               | occurrence             | 1   |
| IN               | ElementName            | 'child'   |
| IN               | ElementNamespace       | 'uri:mynamespace'   |
| OUT              | result                 | '<xml/doc xmlns="uri:mynamespace"><br><parent><br><child><br><grandchild/><br></child><br></parent><br></xml/doc> |

## 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.)

Some of the folders under `/shared/ASAssets/Utilities` have examples in them. 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.

## ***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 hawaii.composite.com:9400. All official distributions of the Utilities are created from this instance. Please do not modify this distribution without notifying one of the development utilities team members.

The source code for the distribution will be managed using the field SVN repository. You can see the source code tree at the following URL (inside the Composite firewall):

[http://kauai.composite.com/viewvc/field/PS\\_Assets/dev\\_utils/](http://kauai.composite.com/viewvc/field/PS_Assets/dev_utils/)

CJP source code will be checked into the other\_resources/java\_src folder (details of folder structure TBD.) CIS resources will be exported using the VCS system and checked into the CIS\_Resources folder in one of the sub folders under /shared/ASAssets/Utilities.

## ***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. Source code will be initially checked in by one of these reviewers. 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. Please submit new procedures to the team members below.

## ***Team Members***

The following are members of the development utilities team:

- Calvin Goodrich ([cgoodrich@compositesw.com](mailto:cgoodrich@compositesw.com))
- Michael Tinius ([mtinius@compositesw.com](mailto:mtinius@compositesw.com))



---

**Americas Headquarters**  
Cisco Systems, Inc.  
San Jose, CA

**Asia Pacific Headquarters**  
Cisco Systems (USA) Pte. Ltd.  
Singapore

**Europe Headquarters**  
Cisco Systems International BV Amsterdam,  
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at [www.cisco.com/go/offices](http://www.cisco.com/go/offices).

---

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: [www.cisco.com/go/trademarks](http://www.cisco.com/go/trademarks). Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)