EMC® Documentum® System

Version 6.5

Object Reference Manual

P/N 300-007-224–A01

EMC Corporation
Corporate Headquarters:
Hopkinton, MA 01748-9103
1-508-435-1000
www.EMC.com

Copyright © 1992 - 2008 EMC Corporation. All rights reserved.

Published July 2008

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED AS IS. EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

All other trademarks used herein are the property of their respective owners.

Table of Contents

Preface 13

Chapter 1 Object Basics 15

Computed properties 15

Property datatypes 24
Note on string datatype nomenclature 26

Referencing properties 26

Identifiers 27
Names 27
Object type and property names 28
User and group names 28
Table and column names 28
Object identifiers 28
Property identifiers 29
Collection identifiers 29
Type identifiers 29
Session identifiers 29

Relationships 29
User-defined relationships 30
Creating a relationship type 30
Creating a relationship between two objects 31
Security and relationships 32
SYSTEM security 32
PARENT and CHILD security 32
None 33
The permanent_link and copy_child properties 33
Compatibility with prior releases 34
Defining delete behavior 34
Destroying a relationship type 36
Destroying a relationship between objects 36
Relationships and object operations 37
Checkin, Saveasnew, and Branch 37
Save 37
Destroy 37
Object replication 37

Chapter 2 Core Persistent Object Types 39

Chapter 3 Lightweight Object Type and Properties for Shareable Object Types and the Property Bag 479

Chapter 4 Non-persistent Object Types 485

Chapter 5 Documentum Process Builder Object Types 499

Chapter 6 Collaboration Object Types 521

Chapter 7 Document Submission Manager Object Types 543

Chapter 8 Retention Policy Services Object Types 567

Chapter 9 Site Caching Services Object Types 593

Chapter 10 Forms Builder Object Types 607

Chapter 11 Editorial Publishing Services Types 615

Appendix A Language and Country Codes 639

Appendix B RDBMS Tables for Documentum Types 643

Overview 643
The tables 643
The views 644
Returning database table information 644
The user type tables in the Oracle RDBMS 644

Appendix C Obsolete, Deprecated, or Unused Object Types 647

List of Tables

Table 1. The computed properties 15
Table 2. Computed properties for lifecycles 23
Table 3. The property datatypes 24
Table 4. Effects of setting permanent_link and copy_child properties 33
Table 5. Deletion behavior when deleting a parent object 35
Table 6. Deletion behavior when deleting a child object 36
Table 7. Properties defined for the ACL type 40
Table 8. Properties defined for the ACS config type 44
Table 9. Properties defined for the activity type 49
Table 10. Properties defined for the activity group instance type 62
Table 11. Properties defined for the aggr domain type 64
Table 12. Properties defined for the alias set type 66
Table 13. Property defined for the app ref type 68
Table 14. Property defined for the application type 69
Table 15. Properties defined for the aspect relation type 72
Table 16. Properties defined for the aspect type object type 73
Table 17. Properties defined for the assembly type 74
Table 18. Properties defined for the audit trail type 77
Table 19. Properties defined for the audit trail acl type 83
Table 20. Properties defined for the audit trail attrs type 85
Table 21. Properties defined for the audit trail group type 86
Table 22. Properties defined for the auth config type 89
Table 23. Properties defined for the blob store type 90
Table 24. Properties defined for the bocs config type 91
Table 25. Property defined for the builtin expr type 94
Table 26. Properties defined for the CA store type 95
Table 27. Properties defined for the cabinet type 98
Table 28. Properties defined for the cache config type 99
Table 29. Properties defined for the category type 101
Table 30. Properties defined for the category assign type 104
Table 31. Properties defined for the category class type 107
Table 32. Properties defined for the change record type 110
Table 33. Properties defined for the CI config type 112
Table 34. Properties defined for the class type 115
Table 35. Default aspect properties defined for the class type 116
Table 36. Properties defined for the client registration type 117
Table 37. Properties defined for the client rights type 118

Table 38. Properties defined for the completed workflow type 120
Table 39. Properties defined for the completed workitem type 124
Table 40. Properties defined for the component type 129
Table 41. Properties defined for the composite predicate type 130
Table 42. Properties defined for the cond expr type 131
Table 43. Properties defined for the cond ID expr type 132
Table 44. Properties defined for the config scope relation type 133
Table 45. Properties defined for the constraint set object type 134
Table 46. Properties defined for the cont transfer config type 135
Table 47. Properties defined for the containment type 138
Table 48. Properties defined for the content type 141
Table 49. Properties defined for the cryptographic key type 149
Table 50. Properties defined for the DD attr info type 151
Table 51. Properties defined for the DD common info type 156
Table 52. Properties applicable to object types and properties 161
Table 53. Properties applicable only to object types 163
Table 54. DD info properties applicable to properties 164
Table 55. Properties defined for the DD type info type 167
Table 56. Properties defined for the display config type 170
Table 57. Properties defined for the distributed store type 172
Table 58. Properties defined for the dms config type 174
Table 59. Properties defined for the docbase config type 176
Table 60. Properties defined for the docset type 186
Table 61. Properties defined for the docset run type 187
Table 62. Properties defined for the domain type 190
Table 63. Properties defined for the dump object record type 192
Table 64. Properties defined for the dump record type 193
Table 65. Properties defined for the email message type 196
Table 66. Properties defined for the esign template type 198
Table 67. Properties defined for the expr code type 200
Table 68. Properties defined for the expression type 201
Table 69. Properties defined for the external file store type 203
Table 70. Properties defined for the external store type 205
Table 71. Properties defined for the federation type 208
Table 72. Properties defined for the file store type 211
Table 73. Properties defined for the folder type 212
Table 74. Properties defined for the foreign key type 213
Table 75. Properties defined for the format type 214
Table 76. Properties defined for the ft index agent config type 217
Table 77. Properties defined for the FT engine config type 221
Table 78. Properties defined for the fulltext index type 222
Table 79. Properties defined for the func expr type 224
Table 80. Properties defined for the group type 227

Table 81. Properties defined for the index type 231
Table 82. Properties defined for the jar type 233
Table 83. Properties defined for the java library type 234
Table 84. Properties defined for the job type 236
Table 85. Properties defined for the job request type 241
Table 86. Properties defined for the job sequence type 243
Table 87. Properties defined for the key type 245
Table 88. Properties defined for the LDAP config type 246
Table 89. Properties defined for the linked store type 253
Table 90. Properties defined for the link record type 254
Table 91. Properties defined for the load object record type 257
Table 92. Properties defined for the load record type 259
Table 93. Properties defined for the location type 261
Table 94. Properties defined for the locator type 263
Table 95. Properties defined for the media profile type 264
Table 96. Properties defined for the message address type 266
Table 97. Properties defined for the message archive type 268
Table 98. Properties defined for the message attachment type 270
Table 99. Properties defined for the message container type 271
Table 100. Properties defined for the message route user data type 273
Table 101. Properties defined for the message user data type 274
Table 102. Properties defined for the method type 276
Table 103. Properties defined for the module type 282
Table 104. Properties defined for the module config type 285
Table 105. Properties defined for the mount point type 287
Table 106. Properties defined for the network location map type 289
Table 107. Properties applicable to object type and properties 291
Table 108. Properties applicable only to properties 292
Table 109. Properties defined for the other file type 295
Table 110. Properties defined for the output device type 296
Table 111. Properties defined for the package type 297
Table 112. Properties defined for the plugin type 300
Table 113. Properties defined for the policy type 302
Table 114. Properties defined for the preset info type 310
Table 115. Properties defined for the preset package type 311
Table 116. Properties defined for the procedure type 312
Table 117. Properties defined for the process type 313
Table 118. Properties defined for the process parameter type 320
Table 119. Properties defined for the process correlation set type 322
Table 120. Atributes defined for the public key certifcate type 323
Table 121. Properties defined for the qual comp type 325
Table 122. Properties defined for the queue item type 328
Table 123. Properties defined for the reference type 335

Table 124. Properties defined for the registered type 340
Table 125. Properties defined for the registry type 342
Table 126. Properties defined for the relation type 345
Table 127. Properties defined for the relation SSA policy type 347
Table 128. Properties defined for the relation type object type 348
Table 129. Properties defined for the relationship def type 352
Table 130. Default aspect properties defined for the relationship def type 355
Table 131. Properties defined for the replica record type 357
Table 132. Properties defined for the retainer type 359
Table 133. Properties defined for the routecase condition type 364
Table 134. Properties defined for the scope config type 368
Table 135. Properties defined for the sequence type 370
Table 136. Properties defined for the server config type 371
Table 137. Properties defined for the Smart List type 382
Table 138. Properties defined for the state extension type 384
Table 139. Properties defined for the state type object type 385
Table 140. Properties defined for the SSA policy type 386
Table 141. Properties defined for the store type 387
Table 142. Properties defined for the subcontent type 393
Table 143. General properties of SysObjects 396
Table 144. SysObject folder-related properties 399
Table 145. Virtual document-related properties 400
Table 146. SysObject content-related properties 401
Table 147. Web-content properties 402
Table 148. SysObject security-related properties 404
Table 149. SysObject version-related properties 405
Table 150. SysObject event-related property 406
Table 151. SysObject lifecycle-related properties 407
Table 152. SysObject aspect-added properties 407
Table 153. Properties defined for the taxonomy type 409
Table 154. Properties defined for the TCF activity type 410
Table 155. Properties defined for the TCF activity template type 411
Table 156. Properties defined for the transition condition type 412
Table 157. Properties defined for the type object type 416
Table 158. Properties defined for the type info type 419
Table 159. Properties defined for the user type 422
Table 160. Properties defined for the validation descriptor type 430
Table 161. Properties defined for the validation module type 431
Table 162. Properties defined for the validation relation type 433
Table 163. Properties defined for the value assist type 435
Table 164. Properties defined for the value func type 437
Table 165. Properties defined for the value list type 438
Table 166. Properties defined for the value query type 439

Table 167. Properties defined for the vstamp type 441
Table 168. Properties defined for the WF attachment type 442
Table 169. Property defined for the WF package report parent type 443
Table 170. Property defined for the WF package schema type 445
Table 171. Properties defined for the wf package skill type 446
Table 172. Properties defined for the wf package type info type 448
Table 173. Properties defined for the WF timer type 449
Table 174. Properties defined for the workflow type 451
Table 175. Properties defined for the work item type 456
Table 176. Properties defined for the work queue type 461
Table 177. Properties defined for the work queue doc profile type 463
Table 178. Properties defined for the work queue policy type 464
Table 179. Properties defined for the work queue user profile type 466
Table 180. Properties defined for the wq skill info object type 467
Table 181. Properties defined for the wq task skill object type 470
Table 182. Properties defined for the wq user skill object type 471
Table 183. Properties defined for the xml application type 472
Table 184. Properties defined for the xml config type 474
Table 185. Properties defined for the xml custom code type 475
Table 186. Properties defined for the xml style sheet type 476
Table 187. Properties defined for the xml zone type 477
Table 188. Properties defined for the lightweight object type 480
Table 189. Properties added to a type in specific circumstances 481
Table 190. Properties defined for the connection config type 487
Table 191. Properties defined for the docbase locator type 489
Table 192. Properties defined for the docbroker locator type 491
Table 193. Properties defined for the server locator type 492
Table 194. Properties defined for the session config type 494
Table 195. Properties defined for the wfsd element object type 501
Table 196. Properties defined for the wfsd element boolean object type 502
Table 197. Properties defined for the wfsd element date object type 503
Table 198. Properties defined for the wfsd element double object type 504
Table 199. Properties defined for the wfsd element integer object type 505
Table 200. Properties defined for the wfsd element string object type 507
Table 201. Properties defined for the wfsd type info object type 508
Table 202. Properties defined for the wfsdrp boolean object type 513
Table 203. Properties defined for the wfsdrp date object type 514
Table 204. Properties defined for the wfsdrp double object type 515
Table 205. Properties defined for the wfsdrp integer object type 516
Table 206. Properties defined for the wfsdrp parent object type 517
Table 207. Properties defined for the wfsdrp string object type 519
Table 208. Properties defined for the calendar event type 523
Table 209. Properties defined for the comment type 526

Table 210. Properties defined for the datatable type 528
Table 211. Properties defined for the datatable schema type 530
Table 212. Properties defined for the datatable schema ex type 531
Table 213. Properties defined for the datatable settings type 532
Table 214. Properties defined for the notepage type 534
Table 215. Properties defined for the readcomment type 535
Table 216. Properties defined for the richtext type 536
Table 217. Properties defined for the room type 537
Table 218. Properties defined for the topic type 540
Table 219. Properties defined for the dmc dsm application type 544
Table 220. Properties defined for the dmc dsm backbone type 545
Table 221. Properties defined for the dmc dsm doc properties type 546
Table 222. Properties defined for the dmc dsm drug product type 548
Table 223. Properties defined for the dmc dsm drug substance type 549
Table 224. Properties defined for the dmc dsm excipient type 550
Table 225. Properties defined for the dmc dsm facilities equip type 551
Table 226. Properties defined for the dmc dsm indication type 552
Table 227. Properties defined for the dmc dsm safety eval type 554
Table 228. Properties defined for the dmc dsm sect doc attributes type 555
Table 229. Properties defined for the dmc dsm section type 557
Table 230. Properties defined for the dmc dsm section extension type 559
Table 231. Properties defined for the dmc dsm stf backbone type 560
Table 232. Properties defined for the dmc dsm stf section type 562
Table 233. Properties defined for the dmc dsm study attributes type 563
Table 234. Properties defined for the dmc dsm study report type 564
Table 235. Properties defined for the dmc dsm submission type 565
Table 236. Properties defined for the rps action type 568
Table 237. Properties defined for the rps action rel type 569
Table 238. Properties defined for the rps authority type 570
Table 239. Properties defined for the rps base date type 571
Table 240. Properties defined for the rps child strategy type 572
Table 241. Properties defined for the rps condition type 574
Table 242. Properties defined for the rps contact type 575
Table 243. Properties defined for the rps disposition method type 576
Table 244. Properties defined for the rps event type 578
Table 245. Properties defined for the rps execution rule type 580
Table 246. Properties defined for the rps hold type 581
Table 247. Properties defined for the rps notification type 583
Table 248. Properties defined for the rps phase rel type 584
Table 249. Properties defined for the rps retainer Type 586
Table 250. Properties defined for the rps retainer event rel type 589
Table 251. Properties defined for the rps retention policy type 590
Table 252. Properties defined for the scs admin config type 594

Table 253. Properties defined for the webc config type 596
Table 254. Properties defined for the webc target type 602
Table 255. Properties defined for the xfm adaptor config type 608
Table 256. Properties defined for the xfm form type 610
Table 257. Properties defined for the xfm instance type 612
Table 258. Properties defined for the xfm schema type 613
Table 259. Properties defined for the eps config cache type 616
Table 260. Properties defined for the eps elem type mapping type 617
Table 261. Properties defined for the eps element type 618
Table 262. Properties defined for the eps issue type 620
Table 263. Properties defined for the eps mapping type 621
Table 264. Properties defined for the eps metadata mapping type 622
Table 265. Properties defined for the eps page detail type 623
Table 266. Properties defined for the eps performer mapping type 624
Table 267. Properties defined for the eps placement type 625
Table 268. Properties defined for the eps pubinfo cache type 626
Table 269. Properties defined for the eps publication type 627
Table 270. Properties defined for the eps relation type 628
Table 271. Properties defined for the eps schedule mapping type 629
Table 272. Properties defined for the eps section type 630
Table 273. Properties defined for the eps state type 631
Table 274. Properties defined for the eps state mapping type 632
Table 275. Properties defined for the eps state perm mapping type 633
Table 276. Properties defined for the eps state transition type 634
Table 277. Properties defined for the eps template mapping type 635
Table 278. Properties defined for the eps transition override type 636
Table 279. Properties defined for the eps wf template mapping type 637
Table 280. Properties defined for the eps workflow data type 638
Table 281. Recommended language codes 639
Table 282. Recommended country codes 640
Table 283. Columns of the dm_user_s table 644
Table 284. Columns of the dm_user_r table 645

Preface

This manual is the reference manual for the EMC Documentum object hierarchy. It is a companion to *Content Server DQL Reference Manual*, *Content Server Fundamentals*, *Content Server Administration Guide*, and *Documentum Distributed Configuration Guide*.

Intended audience

This manual is written for application developers and system administrators and any others who want to build a content or work-group management application that accesses and uses repository objects. It assumes that you are familiar with the concepts of document processing, object-oriented programming, and client-server applications. It also assumes working knowledge of SQL.

Conventions

This manual uses the following conventions in the syntax descriptions and examples.

Syntax conventions

Convention	Identifies
italics	A variable for which you must provide a value.
[ ] square brackets	An optional argument that may be included only once
{ } curly braces	An optional argument that may be included multiple times

Revision history

The following changes have been made to this document.

Revision history

Revision date	Description
July 2008	Initial release

Chapter 1

Object Basics

This chapter provides reference information about objects and properties that is not specific to a particular object type. The topics in this chapter are:

- Computed properties, page 15
- Property datatypes, page 24
- Referencing properties, page 26
- Identifiers, page 27
- Relationships, page 29

Computed properties

This section lists the computed properties supported by Content Server. Computed property values are not stored in the repository with an object's description but are computed at runtime when needed.

Table 1, page 15, lists all computed properties except those that are specific to lifecycles. The lifecycle-specific computed properties are found in Table 2, page 23.

Table 1. The computed properties

Property	Single or repeating	Description
_accessor_app_permit	R	Application_permit values of the entries for an ACL. The value at each index position represents the application_permit property value for the entry represented by the corresponding index position in _accessor_name.

Property	Single or repeating	Description
_accessor_name	R	The list of users and groups for whom some level of access to the selected object is defined. This property has valid values only if the repository is using ACLs for security. This property reflects the current status of an object's ACL. Therefore, if you are changing an object's permissions, this value reflects those changes, regardless of whether you have saved the changes. Check the _acl_ref_valid property to determine whether the values in _accessor_name have been saved.
_accessor_permit	R	The permission levels assigned to each user or group defined in _accessor_name. The values, expressed as integers, are associated with the user or group at the corresponding index position. For example, the permission level at _accessor_permit[4] is assigned to the user or group specified by _accessor_name[4]. This property reflects the current status of an object's ACL. Therefore, if you are changing an object's permissions, this value always reflects those changes, regardless of whether you have saved them. Check the _acl_ref_valid property to determine whether the values in _accessor_permit have been saved.
_accessor_permit_type	R	Permit types of the entries. The value at each index position is the permit type of the entry represented by the corresponding index position in _accessor_name.
_accessor_xpermit	R	Contains the integer value of the extended permissions assigned to each user or group returned by _accessor_name. The values, expressed as integers, are associated with the user or group at the corresponding index position. For example, the permission level at _accessor_xpermit[4] is assigned to the user or group specified by _accessor_name[4].

Property	Single or repeating	Description
_accessor_xpermit_ names	R	A list of the extended permissions, in string form, assigned to each user or group defined in _accessor_name.
_acl_ref_valid	S	A Boolean property that indicates whether the values of _accessor_name and _accesssor_permit have been saved. If they have been saved, this property is FALSE. If the values have not been saved, _acl_ref_valid is TRUE.
_alias_set	S	The name of the alias set object bound to an object. This property is applicable to SysObjects, user objects, and server config objects.
_all_users_names	R	A list of all users directly or indirectly contained in a group. This property is cached on the server and client the first time it is queried in a session. Subsequent queries against it in the session return the cached values. To ensure that the returned values are correct, you can force the server to refresh the cache by issuing a IDfPersistentObject.revert method on the group object. Changing the group membership also invalidates the cache and forces a refresh.
_allow_change_location	S	A Boolean value indicating whether the user has the Change Location permission.
_allow_change_permit	S	A Boolean value indicating whether the user has the Change Permission permission.
_allow_change_state	S	A Boolean value indicating whether the user has the Change State permission.
_allow_execute_proc	S	A Boolean value indicating whether the user has the Execute Procedure permission.
_allow_change_owner	S	A Boolean value indicating whether the user has the Change Ownership permission.
_change_folder_links	S	A Boolean value indicating whether the user has the Change Folder Links permission

Property	Single or repeating	Description
_attribute_list_values	R	A string property containing the list of audited properties and their values recorded in an audit trail entry. The first index position stores the value of dm_audittrail.attribute_list. The remaining index positions stores the values from the dmi_audittrail_attrs object associated with the audittrail object, if any. The length of the property is the maximum length allowed by the underlying RDBMS for character data types.
_cached	S	A Boolean property indicating whether the object is in the object cache. It returns TRUE if the object is in the cache and FALSE if not.
_changed	S	A Boolean property that indicates whether the object has been changed since it was last saved. The server checks the object cache and if the object is not in the cache or is in the cache but hasn't been changed, server returns FALSE. TRUE is returned if the object is present in the cache and has been changed.
_componentID	R	The chronicle IDs of the component documents that make up a virtual document. The ID value at each index position is the chronicle ID for the component at that position in the virtual document. The zero position, _containID[0], represents the chronicle ID for the virtual document. A virtual document is always considered a component of itself.

Property	Single or repeating	Description
_containID	R	The object IDs of the containment objects that link the component document to its containing virtual document. The ID value at each index position represents the containment object for the component at that position in the virtual document. The zero position, _containID[0], represents the containment object for the virtual document. A virtual document is always considered a component of itself.
_content_state	R	An integer value representing the state of a content for a given SysObject. State values and their meanings are: 0, meaing available and online 1, meaning archived 2, meaning offline 3, meaning error fetching content from the repository For example, if _content_state[1] = 1, then the first content for the object is archived.
_count	S	The number of properties in the object
_current_state	S	The current lifecycle state of the object.
_docbase_id	S	That portion of a specified object ID that corresponds to the repository ID.
_dump	S	A description of the object, including all properties and their values
_has_config_audit	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.
_has_create_type	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.
_has_create_group	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.
_has_create_cabinet	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.

Property	Single or repeating	Description
_has_purge_audit	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.
_has_superuser	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.
_has_sysadmin	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.
_has_view_audit	S	A Boolean property indicating whether the user has the privilege. T (TRUE) means the user has the privilege.
_id	S	A unique object identifier that contains the object's r_object_id value
_is_restricted_session	S	A Boolean property that indicates whether the current session is a restricted session in which the only valid operation is changing the user's password. T means that the current session is such a session. F means that the session is not restricted to a change password operation.
_isdeadlocked	S	A Boolean property that indicates whether the current session was chosen as the victim of a deadlock. This property is reset when the next method that issues an RPC call is executed.
_isnew	S	A Boolean property that indicates whether the object is a new object, that is, whether the object has been saved in the repository.
_isreplica	S	For use in a distributed storage environment, this indicates whether an object is a replica (read-only copy) or the original object. This property is valid only for cabinets, folders, documents, notes, and smart lists.
_istransactionopen	S	A Boolean property that indicates whether an explicit transaction is open. TRUE means a transaction is open. FALSE means there is no transaction open.

Property	Single or repeating	Description
_lengths	R	Lengths of an object's properties. Property length is measured in the number of bytes required by its datatype, such as a 4-byte integer. However, this information is only useful for character string columns.
_masterdocbase	S	The repository ID of the repository that owns an object. Unless the object is a replica, this value is the ID of the local repository. If the object is a replica, the value is the ID of the foreign repository that owns the object. This property is an integer property.
_names	R	Names of an object's properties.
_permit	S	An integer number that corresponds to the permissions for an object of a specified user. If the user is unspecified in the query (for example: get,c,object_id,_permit), the current user is assumed. The permission returned represents the most permissive permission the user has for the specified object. A value of 0 indicates that the object is not a SysObject. Values of 2 to 7 correspond to the object-level permissions Browse through Delete, respectively. The *Content Server Administration Guide* has details about the permissions defined for objects.
_policy_name	S	Name of the lifecycle to which the object is attached.
_repeating	R	Whether properties are single or repeating. Values are 0 for single-valued properties and 1 for repeating properties.
_resume_state	S	Name of the lifecycle's normal state which immediately preceded the current state of the object.

Property	Single or repeating	Description
_sign_data	S	An XML file that contains the properties used to generate the hashed signature for an audit trail entry. The output is well-formed XML. It includes the values of the entry's associated dmi_audittrail_attrs object, if one exists.
_status	S	Level of the error (if any) resulting from a method call against an object. The level is represented as a numeric value. The values are: 0, for informational messages (OK, for example) 1, for trace messages 2, for warning messages 3, for error messages 4, for fatal error messages If two or more errors occur, only the first error's level is stored unless the subsequent error is more severe. In such cases, the number associated with the more severe error is stored. If no errors occur, this property contains 0 (for OK). To clear this property, use the reset method.
_type_id	S	The object ID of the dm_type object for the object type of the specified object. This is returned as a string value.
_type_name	S	The internal name of the object's type.
_types	R	Datatypes of an object's properties. The datatypes are specified as numbers: 0, for Boolean 1, for Integer 2, for String 3, for ID 4, for Time 5, for Double

Property	Single or repeating	Description
_typestring	S	Returns the number of supertypes in a specified object's hierarchy and type information about each supertype.
_values	R	Total number of values of an property (useful for repeating properties).
_xpermit	R	Contains the integer value of the extended permissions the current user or the specified user has on the object.
_xpermit_list	S	Contains a full list of the extended permissions, in string form, currently supported by server. Note that this computed property returns the same list regardless the object ID.
_xpermit_names	s	Contains the list of the extended permissions, in string form, the current user or the specified user has on the object.

Table 2, page 23, lists additional computed properties that apply only to lifecycles (dm_policy objects).

Table 2. Computed properties for lifecycles

Property	Single or repeating	Description
_alias_sets	R	List of the object names of the valid alias sets for the lifecycle. The value at each index position corresponds to the alias set specified at the same index position in the alias_set_ids property.
_entry_criteria	R	List of expressions that must evaluate to TRUE before an object can be promoted to a state. The value at each index position corresponds to the conditions set for the state defined at that index position. This computed property is read and write. Setting _entry_criteria sets the entry_criteria_id property in the policy object. The *Content Server Fundamentals* has details.
_included_types	S	The list of acceptable object types for this policy, in lower case, separated by commas.
_next_state	R	The name of next state. Exception states and the terminal state do not have a next state.

Property	Single or repeating	Description
_previous_state	R	The name of previous state. Exception states and the base state do not have a previous state.
_state_extension_obj[[*index*]]	R	The object ID of the state extension object associated with a particular lifecycle state. The index value identifies the lifecycle state by identifying the state's state number. For example, _state_extension_obj[0] returns the object ID of the state extension object associated with the lifecycle state number 0. If an index value is not included, the default is 0.
_state_type	R	Identifies whether the state is an exception state or, if is not an exception state, it's place in the sequence of lifecycle states. The information at each index position is applicable to the state whose state number corresponds to the index position number. Valid values are: 1, for Exception 0, for Base (first normal state) 1, for Terminal (last normal state) 2, for Step (normal between base and terminal states)

Property datatypes

Table 3, page 24, lists the Documentum datatypes, their RDBMS equivalents, their range of values, and their default values. Note that the length specifications for the string datatype are expressed as number of bytes.

Table 3. The property datatypes

Documentum Property Datatypes	DQL Type	Oracle	DB2, Sybase, and MS SQL Server	Value Range	Default Value
DM_INTEGER	integer	number(10)	integer	-2147483647 to +2147483647	0

Documentum Property Datatypes	DQL Type	Oracle	DB2, Sybase, and MS SQL Server	Value Range	Default Value
DM_ BOOLEAN	boolean, bool	number(6)	integer	1 (TRUE) or 0 (FALSE)	0 (FALSE)
DM_ STRING For values for non-qualifiable string properties, refer to the footnote **	string, character, char	char(64) max length is 4000	varchar, char for DB2 and Sybase. On DB2: maximum length is 2000 For Sybase, refer to the footnote * nvarchar and nchar on SQL Server, with maximum length of 4000 (Unicode) or 7000 (non-Unicode)	0 (blank string) to maximum for RDBMS	a null string, specified ""
DM_ DOUBLE	double, float	number 1 x 10-129 to 1 x 10+129	float 1.7e-308 to 1.7e+308	specific to RDBMS	0.0
DM_TIME	time, date	date	timestamp (DB2) datetime (MS SQL Server and Sybase)	01/01/1753 to 12/31/9999	current time

Documentum Property Datatypes	DQL Type	Oracle	DB2, Sybase, and MS SQL Server	Value Range	Default Value
DM_ID	ID	char(16)	char(16)	not applicable	an ID of all zeros

* The maximum length of a string property on a Sybase RDBMS is dependent on the configuration of the underlying database tables; specifically, on the page size of the tables and whether they are configured as APL or DOL tables. For allowed sizes for the various sizes and configurations, refer to the Sybase installation and configuration documentation.

** The maximum length of a non-qualifiable property defined as a string datatype is the value in the max_nqa_string key in the server.ini file. If not set, the maximum length of a non-qualifiable string property is the maximum length defined by the underlying relational database.

Note on string datatype nomenclature

The lengths defined for a string-datatype property in Table 3, page 24, and in the reference descriptions of object types are expressed as bytes. For example, the dm_sysobject.title property is string(255). This means that it is a string datatype that accepts a value that is a maximum of 255 bytes long.

Referencing properties

You can reference persistent properties in DFC method calls or DQL statements. Computed properties can be referenced only in DFC calls; you cannot reference them in DQL statement.

DFC provides get and set methods for most properties. For those that do not have specific get and set methods, you can use the get*Datatype* and set*Datatype* methods in the IDfTypedObect interface. For example, if you were retrieving the value of a string property, you could call IDfTypedObject.getString with the property name as the argument. There is a get and set method for each supported datatype in the IDfTypedObject interface.

When you reference a property in DQL, use the property's name. For example:

```
SELECT "title","subject" FROM "dm_document"
WHERE ANY "keywords" IN ('floorplan','cubicle')
```

If the property is a property defined for an aspect, qualify the property name with the aspect name. For example:

```
SELECT "title","editingAspect.editor_name" FROM "dm_document"
WHERE ANY "keywords" IN ('floorplan','cubicle')
```

If the name matches a DQL reserved word, you must double-quote the name when you reference it in a method call or DQL statement.

Identifiers

Content Server recognizes the following kinds of identifiers:

- Names (Names, page 27)
- Object identifiers (Object identifiers, page 28)
- Property identifiers, page 29
- Collection identifiers Collection identifiers, page 29)
- Type identifiers (Type identifiers, page 29)
- Session identifiers (Session identifiers, page 29)

Names

Names identify types, properties, tables, columns, users, and groups.

Names fall into three groups, depending on which naming rules apply to them. The groups are:

- Object type and property names

  The rules for these are described in Object type and property names, page 28.
- User and group names

  The rules for these are described in User and group names, page 28.
- Table and column names

  The rules for these are described in Table and column names, page 28.

Three naming rules apply to all names:

- Do not begin a name with the prefix dm. Documentum reserves this prefix for its own use.
- The name must consist of ASCII characters.
- If a name matches a DQL reserved word, enclose the name in double quotes whenever you use it in a DQL query.

  Note: It is good practice not to choose names that conflict with DQL reserved words.

In addition to these rules, other rules apply specifically to each group of names.

Object type and property names

Object type and property names must observe the following rules:

- The maximum allowed length is 27 characters.
- The first character must be a letter. The remaining characters can be letters, digits, or underscores.
- The name cannot contain spaces or punctuation.
- The name cannot be any of the words reserved by the underlying RDBMS.

In addition, object type names cannot end in an underscore (_).

Type and property names are not case sensitive.

User and group names

User and group names must observe the following rules:

- The maximum allowed length is 32 characters.
- If the name is enclosed in single quotes when referenced, it can contain any character allowed in a character string literal (spaces, apostrophes, and so forth).
- If the name is enclosed in double quotes when referenced, it can be a DQL reserved word.

User names are case sensitive. Group names are not case sensitive. All group names are stored in lowercase in the repository regardless of the case used for the name when creating the group.

Table and column names

Table and column names must observe the following rules:

- The maximum length allowed by Content Server is 64 characters. The RDBMS or operating system may impose stricter limits on the name's length.
- The first character must be a letter. The remaining characters can be letters, digits, or underbars.
- The name cannot contain spaces or punctuation.
- The name cannot be any of the words reserved by the underlying RDBMS.

Table and column names may or may not be case sensitive, depending on the rules imposed by the RDBMS or operating system.

Object identifiers

Object identifiers, or object IDs, are assigned by Content Server when an object is created. You cannot assign them yourself nor can you change them. Object IDs are expressed as a 16-character string.

Property identifiers

Property identifers are assigned by Content Server when the property is defined for an object type. That can occur when the object type is created or when a type is altered to add a property.

Property identifiers are unique within the object type's type hierarchy. The identifiers cannot be changed. They are used instead of property names, to identify properties when properties are stored in a property bag.

Collection identifiers

A collection identifier represents the non-persistent object (collection) that contains the results of a query or a method call. When the server executes the query or method that returns a collection, it places the results in a collection and returns an identifier for the collection to the user or application. The user or application must use that identifier to retrieve the results.

Collection identifiers are two characters. The first is the letter q and the second is a numeric character.

Type identifiers

A type identifier is a two-character string that appears as the first two characters of an object's object ID. Each system-defined object type has a unique type identifier. For example, the type identifier for the dm_document type is 09, and all objects of type dm_document have an object ID that begins with 09.

When you create an object type, it inherits its type identifier from the system-defined supertype. For example, any user-defined subtypes of dm_document also have a type identifier of 09.

User-defined types that have no supertype have a type identifier of 00.

Session identifiers

Session identifiers are used in trace file entries to identify the session associated with an entry. A session identifier has the format s*n*, where *n* is an integer number.

Relationships

A relationship is an association between two objects in the repository. Some system-defined relationships are installed with Content Server. (For example, annotations are implemented as

a system-defined relationship between a SysObject, generally a document, and a note object.) Documentum also allows you to define your own kinds of relationships.

Each kind of relationship is described by a dm_relation_type object. The properties of a relation type object name the relationship, define the security that applies to instances of that relationship, and define the behavior when an object participating in an instance of that relationship is copied, versioned, or deleted.

Each instance of a particular relationship is described by a dm_relation object. A relation object identifies the two objects involved in the relationship and describes the type of relationship. Relation objects also have several properties that you can use to manage and manipulate the relationship.

Both types, dm_relation_type and dm_relation, are persistent and, consequently, objects of those types have object IDs and are stored in the repository. However, you cannot version relation type or relation objects, nor are object-level permissions applied to these object types. (Security for relationships is set by the value defined for the security_type property of the relation type object. Refer to Security and relationships, page 32 for more information.)

User-defined relationships

You must provide the procedures that manage and manipulate the objects in a user-defined relationship. Content Server enforces only security for user-defined relationships. If the nature of the relationship requires actions other than security management, you must write the procedures for those actions. For example, suppose you define a relationship between two documents that requires one document to be updated automatically when the other document is updated. The server does not perform this kind of action. You must write a procedure that determines when the first document is updated and then updates the second document.

Creating a relationship type

A relation type object describes a relationship. Before you can create relation objects representing instances of a relationship, the relationship must be described by a relation type object. Creating a relation type object requires Sysadmin or Superuser privileges.

To define a type of relationship:

1. Create a relation type object.
2. Set the object's child_type, parent_type, relation_name, and security_type properties.
   Security and relationships, page 32, describes the possible settings for security_type.
3. Set the permanent_link and child_copy properties.

These properties control how instances of the relationship are handled when the parent object is copied or versioned using a 5.3 (or higher) DFC-based application. For complete information about how these properties control behavior and compatibility with client library versions, refer to The permanent_link and copy_child properties, page 33.

4. To define non-default delete behavior, set the direction_kind and integrity_kind properties.

   Defining delete behavior, page 34, describes how the settings of these properties affect behavior when users delete objects participating in relationships.

5. Use the Save method to save the object.

Creating a relationship between two objects

To create a relationship between two objects, you must create the relation object that describes their relationship. A relation object represents an instance of the relationship described by a relation type object.

To create a relationship between two objects:

1. Create a dm_relation object.
2. Set the object's attributes.

   You must set the following attributes:

   - relation_name

     This identifies the type of relationship between the two objects. It must match the value of the relation_name attribute of an existing relation type object.

   - parent_id

     This identifies the object that is considered the parent in the relationship. Use the object's object ID.

   - child_id

     This identifies the object that is considered the child in the relationship. Use the object's object ID.

   Additionally, you may want to set the optional attribute, child_label. Setting the child_label attribute defines which version of the child participates in the relationship. The version is not required to be the same version you identified in the child_id attribute. If the object specified in child_id does not carry the correct version label, the server searches the version tree containing the version identified in the child_id attribute for the version specified in child_label and uses that version in the relationship.

   Note: If the applications that will manage and manipulate the objects in the relationship are running against the DMCL or a pre-5.3 DFC, you may also want to set the permanent_link attribute. In the DMCL or a pre-5.3 DFC, this attribute controls whether the relationship is

maintained when the parent object is copied or versioned. (In a 5.3 DFC, this behavior is controlled by attributes in the relationship's definition instead of by the permanent_link attribute at the relation instance level. For more information, refer to The permanent_link and copy_child properties, page 33.)

3. Save the object.

Security and relationships

Documentum manages security for all relationships, system- or user-defined.

The security level defined for a relation type object determines who can create, modify, or drop instances of that relationship. You define security by setting the security_type property for the relation type object. There are four valid settings:

- SYSTEM
- PARENT
- CHILD
- NONE

SYSTEM security

SYSTEM security means that only superusers or system administrators can directly create, modify, or drop the relation objects that represent instances of the relationship.

SYSTEM-level security does not prevent an owner of an object from destroying an object that participates in the relationship even though destroying the object automatically destroys all relation objects associated with the object. The object owner is not required to be a superuser or system administrator.

PARENT and CHILD security

PARENT and CHILD security levels mean that the permissions to create, modify, or drop instances of a relationship depend on the object type of either the parent or child, whichever is specified. If you specify PARENT, the server uses the object type specified as the parent in the relationship to determine the permissions. If you specify CHILD, the server uses the object type specified as the child to determine the permissions. In either case, the rules that the server applies are the same:

- If the object type is dm_user or dm_group, only a superuser or system administrator can create, modify, or drop instances of the relationship.

- If the object type is dm_sysobject or a SysObject subtype, a user must have at least Relate permission on the object to create, modify, or drop instances of the relationship.
- If the object type is a persistent type that is not covered by either of these two rules, security is not enforced. For example, objects of type dm_store fall into this category.

None

NONE security means that any user can create, modify, or drop instances of the relationship. If you do not want the server to enforce security for a particular type of relationship, set the security_type property for the relation type object to NONE.

The permanent_link and copy_child properties

The permanent_link and copy_child properties control how instances of the relationship are handled by a 5.3 DFC-based application when the parent object is copied or versioned. They do not effect behavior if the application is based on pre-5.3 DFC or any version of the DMCL. (For more details, refer to Compatibility with prior releases, page 34.)

The permanent_link property determines whether the relationship is maintained when the parent is copied or versioned. The property is T (TRUE) by default, meaning that the relationship is maintained. The copy_child property is used only when permanent_link is true and when the parent is copied. copy_child is not used if the parent is versioned.

If permanent_link is true and the parent is copied, then copy_child is used to determine whether the relationship with the new parent is established with the old child or a new copy of the child. Table 4, page 33, describes the behavior for all combinations of permanent_link and copy_child.

Table 4. Effects of setting permanent_link and copy_child properties

	permanent_link=T	permanent_link=F
copy_child=0	The child object is not copied. A new relation object is created that associates the new copy or version of the parent object with the old child.	The relationship is not maintained when the parent is copied or versioned.
copy_child=1	If the parent is copied (using IDfCopyOperation), the child object is copied. A new relation object is created that associates the new copy or version of the parent	The relationship is not maintained when the parent is copied or versioned.

	permanent_link=T	permanent_link=F
	object with the new copy of the child object. If the parent object is versioned, the child object is not copied and a new relation object is created that associates the new version of the parent object with the old child.	

Compatibility with prior releases

The dm_relation_type.permanent_link and dm_relation_type.copy_child properties are recognized and supported only by DFC versions 5.3 and later. The DMCL and pre-5.3 versions of DFC do not recognize or support these properties.

The DMCL and pre-5.3 versions of DFC will continue to use dm_relation.permanent_link to determine whether to maintain an instance of a relation when the parent object is copied or versioned.

In DFC 5.3 and later, the value of dm_relation_type.permanent_link is the default value for the dm_relation.permanent_link property. Consequently, if a user or application creates an instance of a relationship using an application that is using a 5.3 or later version of DFC and the user or application does not set the dm_relation.permanent_link property, the DFC sets the property's value to the value of the dm_relation_type.permanent_link property by default. This ensures that the relationship is correctly handled whether the parent object is manipulated by an application using DFC 5.3 or later, the DMCL, or a pre-5.3 DFC application.

Defining delete behavior

How Content Server behaves when a user deletes an object that participates in a relationship depends on how the direction_kind and integrity_kind properties are set in the relationship's definition (the dm_relation_type object). These properties are integer properties.

The direction_kind property defines the direction of the relationship. Valid values are:

Value	Meaning
0	From the parent to the child (default)
1	From the child to the parent
2	Bidirectional (the objects are treated as siblings)

The integrity_kind property defines what deletion rule is applied to objects participating in instances of the relationship. The possible values are:

Value	Meaning
0	Allow delete (default)
1	Restrict delete
2	Cascade delete

The allow delete option means that a user with appropriate permissions can delete an object referenced as parent or child in the relationship without restriction. All relation objects that reference the object are also deleted.

The restrict delete option means that an object participating in the relationship cannot be deleted until the relation object is deleted. You must destroy the relation object before you can delete the participating object.

The cascade delete option means that when a user deletes an object participating in a relationship instance, Content Server may also destroy the participating partner in the relationship, depending on the direction of the relationship. For example, if the relationship is parent to child and a user deletes the parent, Content Server automatically removes the child also. The deletions may cascade to other relationships in which the partner is participating.

Table 5, page 35, describes the behavior for all combinations of direction_kind and integrity_kind when a parent object is deleted from the repository.

Table 5. Deletion behavior when deleting a parent object

direction_kind	integrity_kind		
	0 (allow delete)	1 (restrict delete)	2 (cascade delete)
0 (parent to child)	Deletion succeeds; child object is not deleted.	Deletion fails	Deletion succeeds; child object is also deleted.
1 (child to parent)	Deletion succeeds; child object is not deleted.	Deletion succeeds; child object is not deleted.	Deletion succeeds; child object is not deleted.
2 (bidirectional -siblings)	Deletion succeeds; child object is not deleted.	Deletion fails	Deletion succeeds; child object is also deleted.

Table 6, page 36, describes the behavior for all combinations of direction_kind and integrity_kind when a child object is deleted from the repository.

Table 6. Deletion behavior when deleting a child object

direction_kind	integrity_kind		
	0 (allow delete)	1 (restrict delete)	2 (cascade delete)
0 (parent to child)	Deletion succeeds; parent object is not deleted.	Deletion succeeds; parent object is not deleted.	Deletion succeeds; parent object is not deleted.
1 (child to parent)	Deletion succeeds; parent object is not deleted.	Deletion fails	Deletion succeeds; parent object is also deleted.
2 (bidirectional -siblings)	Deletion succeeds; parent object is not deleted.	Deletion fails	Deletion succeeds; parent object is also deleted.

Destroying a relationship type

To remove a particular kind of relationship from your repository, you destroy the relation type object that describes that relationship. Destroying a relation type object requires either Superuser or Sysadmin privilege.

You cannot destroy a relation type object if there are relation objects that reference that relationship. You must first destroy the relation objects.

Destroying a relationship between objects

To destroy a user-defined relationship between two objects, destroy the relation object representing their relationship. Destroying a relation object does not remove the participating objects from the repository. It only removes the connection between the objects that was established by the relation object.

The security level defined for a relationship (in the relation type object) determines who can destroy the relation objects that represent instances of the relationship. (For information about security levels, refer to Security and relationships, page 32.)

Use a destroy method to remove a relation object from the repository. Specify the relation object's object ID.

Relationships and object operations

This section describes how relationships and common operations interact.

Checkin, Saveasnew, and Branch

How a relationship is handled when the parent object is checked in, saved as new, or branched depends on the setting of the two attributes, permanent_link and copy_child, in the relationship's definition (the dm_relation_type object defining the relationship type). For information about how the settings of these attributes are used, refer to The permanent_link and copy_child properties, page 33.

Note: In the DMCL and in a pre-5.3 DFC, the permanent_link attribute in the dm_relation object itself determines how the relationship is handled when the parent is versioned or copied. If permanent_link is set to T, when the parent object is versioned or copied, a new relation object is created that associates the new copy or version with the child object.

Save

Saving an object does not change or remove any of its user-defined relationships.

However, saving an object does not automatically keep its annotations, a system-defined relationship. Refer to *Content Server Fundamentals* for information about this.

Destroy

When you destroy an object, the server also destroys all the relation objects that reference the destroyed object, regardless of whether the object participates in the relationship as parent or child.

Content Server behavior when you attempt to destroy an object that participates in a user-defined relationship is determined by the settings of the direction_kind and integrity_kind attributes in the relationship's definition (the dm_relation_type object). For details, refer to Defining delete behavior, page 34.

Object replication

If a replicated object is a parent in any user-defined relationship in the source repository, the child object and the associated relation and relation type objects are replicated.

If the replicated object has annotations, the annotations are replicated.

Chapter 2

Core Persistent Object Types

This chapter contains the reference information for object types that are either built in to Content Server or installed through a script when a repository is configured. Most of the types listed in this chapter are direct or indirect subtypes of the persistent object type. The persistent object type has three properties that it passes to all of its subtypes:

- r_object_id

  The r_object_id property contains a 16-character hexadecimal string that is assigned by the system when an object is created. This value uniquely identifies the object in the repository.

- i_vstamp

  The i_vstamp property contains an integer value that represents the number of committed transactions that have changed an object. This value is used for versioning, as part of the locking mechanism, to ensure that one user does not overwrite the changes made by another.

- i_is_replica

  The i_is_replica property indicates whether the object is a local replica of an object in a remote repository.

The non-persistent types described in this chapter are not stored in the repository. Consequently, they do not have object IDs or i_vstamp values, and they are never replicated. They are objects that are created and used during a session and vanish when the session is terminated.

This chapter lists the properties defined for the persistent and non-persistent types in Content Server. For the persistent types, the information also includes the type's supertype, subtypes, and internal name.

ACL

Purpose

An ACL object represents an Access Control List.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_acl
Object type tag: 45

For SysObjects, the entries in the repeating properties control who can access the object to which the ACL is attached. If the repository security mode is set to acl, Content Server enforces the permissions granted by the ACL entries.

Properties

Table 7, page 40, lists the properties defined for the type.

Table 7. Properties defined for the ACL type

Property	Datatype	Single or repeating	Description
acl_class	integer	S	Indicates whether the ACL is a regular ACL, a template, an instance of a template, or a public ACL. Valid values are: 0, Regular ACL 1, Template ACL 2, Template instance 3, Public ACL
description	string(128)	S	User-defined description of the ACL.
globally_ managed	Boolean	S	Indicates whether the acl object is managed globally or locally. The default is FALSE, meaning that it is locally managed.

Property	Datatype	Single or repeating	Description
i_has_access_ restrictions	Boolean	S	Whether the ACL has one or more entries with a permit_type value of 2.
i_has_required_groups	Boolean	S	Whether the ACL has one or more entries with a permit_type value of 3.
i_has_required_ group_set	Boolean	S	Whether the ACL has one or more entries with a permit_type value of 4.
i_partition	integer	S	Identifies the partition where the object resides.
object_name	string(32)	S	Names the ACL object. The name, if provided, must be unique among the ACLs in the repository. This property can be NULL.
owner_name	string(32)	S	Identifies the owner of the ACL. This will be the user who created the ACL or, for system-level ACLs, the name of the repository owner or the alias dm_dbo.
r_accessor_name	string(32)	R	List of those for whom access to the ACL's attached object is defined. Valid entries in the list are individual user names, group names, and aliases. This must be or resolve to a group name if the r_permit_type is RequiredGroup or RequiredGroupSet.

Property	Datatype	Single or repeating	Description
r_accessor_permit	integer	R	Identifies the access level granted to the user or group at the corresponding index level. Valid values are: 0, for NULL, 1, for None 2, for Browse 3, for Read 4, for Relate 5, for Version 6, for Write 7, for Delete Refer to the *Content Server Administration Guide* for more information about these levels.
r_accessor_ xpermit	integer	R	Identifies the extended permission level granted to the user or group at the corresponding index level. Extended permissions are: execute_proc change_location change_state change_permit change_owner delete_object
r_application_permit	string(128)	R	Specifies a permission recognized by an application. An application permit is not recognized or enforced by Content Server. The application is responsible for enforcing the permission. The permission defined at a particular index position is applicable to the user or group identified in the corresponding position in r_accessor_name.

Property	Datatype	Single or repeating	Description
r_is_group	Boolean	R	Indicates whether the r_accessor_names at the corresponding index levels are individual users or groups.
r_has_events	Boolean	S	Whether somone has registered the ACL for auditing.
r_is_internal	Boolean	S	Indicates whether the ACL was created explicitly by a user or implicitly by the server.
r_permit_type	integer	R	Identifies the permit type for the entry. Valid values are: 0, meaning AccessPermit 1, meaning ExtendedPermit 2, meaning ApplicationPermit 3, meaning AccessRestriction 4, meaning ExtendedRestriction 5, meaning ApplicationRestriction 6, meaning RequiredGroup 7, meaning RequiredGroupSet To set this to either 6 or 7, the value in r_accessor_name at the corresponding index position must be a group name. The default value is 0.

ACS Config

Purpose

Stores the configuration information for an ACS Server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_acs_config
Object type tag: 08

An acs config object describes the configuration of an Accelerated Content Services (ACS) server. One ACS server is installed with each Content Server. The installation automatically creates the associated acs config object. To change or modify the settings, use Documentum Administrator.

Properties

Table 8, page 44, lists the properties defined for the type.

Table 8. Properties defined for the ACS config type

Property	Datatype	Single or repeating	Description
acs_base_url	string(240)	R	Base URL to use when accessing the ACS using the protocol identified at the corresponding index position in acs_supported_protocol.
acs_network_locations	string(80)	R	The network identifiers of the network locations served by the ACS.
acs_rw_capability	integer	S	Defines whether the ACS is enabled and whether it can use surrogate get. Valid values are: 0, meaning the ACS is disabled 1, meaning the ACS is enabled 3, meaning SurrogateGet is enabled

Property	Datatype	Single or repeating	Description
acs_supported_ protocol	string(6)	R	Set of network protocols supported by the ACS Values in this property must be all lowercase. For example: http or https.
config_type	integer	S	Whether the accessible storage areas and defined projection targets are configured in the server config object or this acs config object. Valid values are: 1, meaning use the configuration values in the server config object (identified in srv_config_id) 2, meaning use the configuration values in this acs config object (defined in the near_stores and projection-related properties)
delivery_type	string(4)	S	Whether the DMS server can send messages to the ACS server or must wait for the ACS server to pull messages from the DMS server. Valid values are: push, meaning the DMS can send messages to this server pull, meaning DMS must wait for the server to pull messages The default value is push.
is_cache_acs	Boolean	S	The value is F (FALSE) for ACS servers. A value of T (TRUE) is not supported for ACS servers.
near_stores	string(32)	R	Storage areas that are accessible for this ACS server The default is a single blank.

Property	Datatype	Single or repeating	Description
projection_enable	Boolean	R	Indicates whether projection to the connection broker specified at the corresponding index position in projection_targets is enabled.
projection_netloc_ enable	Boolean	R	Indicates whether the server is allowed to service users in the network location specified at the corresponding index position in projection_netloc_ident.
projection_netloc_ ident	string(80)	R	Names of the network locations that this server can service. The corresponding index position in projection_netloc_enable must be set to T (TRUE) to allow the server to actually handle requests from users in a particular network location specified in this property.
projection_ports	integer	R	Identifies the port on which the connection broker is listening. The value at each index position is matched to the connection broker specified at the corresponding level in projection_targets. The default is 1489.
projection_proxval	integer	R	Proximity values representing the ACS server's proximity to the network locations identified at the corresponding index position in projection_netloc_ident.
projection_targets	string(80)	R	Names of the host machines on which the connection brokers reside.
server_major_version	string(16)	S	Used internally. For ACS servers associated with a Version 6 Content Server, this is set to 2.

Property	Datatype	Single or repeating	Description
server_minor_version	string(16)	S	Used internally. For ACS servers associated with a Version 6 Content Server, this is set to 0.
supported_ compression_mode	integer	R	Specifies whether the ACS supports compression for the storage type identified in the corresponding index position of supported_store_types. Valid values are: 0, meaning compression is not supported 1, meaning compression is supported
supported_crypto_ mode	integer	R	Specifies whether the ACS supports encryption for the storage type identified in the corresponding index position of supported_store_types. Valid values are: 0, meaning encryption is not supported 1, meaning encryption is supported
supported_store_types	integer	R	Set of storage area types supported by the ACS. Valid values are: • 1, for file store • 3, for linked store **Note:** Linked storage areas are deprecated in Documentum 6 . • 4, for distributed store • 5, for blob store • 7, for external store • 8, for external file store • 9, for external URL store

Property	Datatype	Single or repeating	Description
			• 10, for external free store • 11, for ca store
srv_config_id	ID	S	Object ID of the server config object of the ACS server's associated Content Server.

Activity

Purpose

An activity object defines a workflow activity.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_activity
Object type tag: 4c

The properties in an activity object define who performs the activity and the packages and work items generated from the activity.

Properties

Table 9, page 49, lists the properties defined for the activity type.

Table 9. Properties defined for the activity type

Property	Datatype	Single or repeating	Description
activity_group_flag	integer	S	reserved for future use
activity_group_id	ID	S	reserved for future use
control_flag	integer	S	Controls who receives a task if automatic delegation fails. Valid values are: 0, meaning that the task is reassigned to the workflow supervisor 1, meaning the task is reassigned to the original task performer.

Property	Datatype	Single or repeating	Description
exec_err_handling	integer	S	Indicates how to handle execution failure for a runtime instance of this activity. Valid values are: 0, meaning stop if the activity fails 1, meaning proceed with the transition even if the activity fails 2, meaning terminate the workflow The default is 0.
exec_method_id	ID	S	Contains the ID of the dm_method object that represents the application. This is required if exec_type is automatic.
exec_retry_interval	integer	S	The length of time, in minutes, to wait between each attempt to execute an automatic activity. The default is 0, meaning retry immediately.
exec_retry_max	integer	S	The maximum number of times to attempt to execute an automatic activity before taking the failure action defined in exec_err_handling. The default is 0, meaning do not make any execution retries.
exec_save_results	Boolean	S	Indicates whether to save results generated by the application. The default is F (FALSE). If set to T (TRUE), the server does not verify the execution results and assumes that the method executed properly. Execution results are saved for later verification in the object whose ID is recorded in the

Property	Datatype	Single or repeating	Description
			r_exec_result_id property of the work item.
exec_subtype	integer	S	Indicates whether the activiity waits for an inbound message to proceed at runtime. Valid values are: 0, meaning that is a Begin, Step or End activity and no inbound message is required to start the activity 1, meaning that this is not an Initiate activity but it is triggered by an inbound message 2, meaning that this is an Initiate activity that is triggered by an inbound message. 3, meaning that this is an Initiate activity but it is not triggered by an inbound message. Typically, these activities have forms associated with them.
exec_time_out	integer	S	Defines how long an automatic activity's method is allowed to execute before timing out. You must set this to a positive number if the activity is an automatic activity (exec_type = 1). The value is interpreted in seconds. This is not used for manual activities.
exec_type	integer	S	Indicates the execution type. Valid values are: 0, for manual 1, for automatic

Property	Datatype	Single or repeating	Description
is_private	Boolean	S	Indicates whether the definition is private or public. The default is F (private).
performer_flag	integer	S	Indicates whether delegation or extension is allowed. Valid values are: 0, No delegation or extension 1, Allow delegation 2, Allow extension 3, Allow both delegation and extension The default is 0.
performer_name	string(66)	R	Identifies the activity's performer or performers. Valid values are a user name, a group name, the string dm_world, representing all users in the repository, a work queue name, or an alias. The alias format is `%[alias_set_name.]alias_name` **Note:** Do not specify a user or group name that begins with %. The server interprets that as an alias.
performer_type	integer	S	Identifies the category of the activity performer. Valid values are: 0, Workflow supervisor 1, repository owner 2, Last performer 3, A user 4, All members in a group 5, Any user in a group 6, The group member who is least loaded 8, Some members of a group or some users in the repository

Property	Datatype	Single or repeating	Description
			9, Some members of a group or some users in the repository sequentially 10, A user from a work queue
post_timer	integer	S	Defines the length of time, in hours, in which the activity must be completed after it is started. The default value is 0, which means the timer is not in use. This property is only set when Workflow Manager is used to create a post-timer. Using Process Builder to define a post-timer sets the post_timer_increment property instead.
post_timer_action	ID	R	Object IDs of the module config objects that identify the actions associated with the timer's firing. The ID at a particular index position represent the action associated with the triggering associated with the corresponding index position in post_timer_increment.
post_timer_calendar_flag	integer	S	reserved for future use
post_timer_calendar_id	ID	S	reserved for future use

Property	Datatype	Single or repeating	Description
post_timer_increment	integer	R	Value in the first index position defines a time interval after which the timer is triggered if the activity has not been completed. The interval is expressed in minutes and is counted from the start of the activity. Values in the following index positions define intervals for second and subsequent triggerings of the timer if the activity has not been completed. The intervals in these index positions are counted from the end of the previous interval.
post_timer_name	string(64)	S	Contains names of the post-timers defined for this activity
post_timer_repeat_last	Boolean	S	Whether to repeat the last action specified in the post_timer_action property.
pre_timer	integer	S	Defines the length of time, in hours, in which the activity must be started after the workflow starts. The default value is 0, which means the timer is not in use. This property is only set when Workflow Manager is used to create a pre-timer. Using Process Builder to define a pre-timer sets the pre_timer_increment property instead.

Property	Datatype	Single or repeating	Description
pre_timer_action	ID	R	Object IDs of the module config objects that identify the actions associated with the timer's firing. The ID at a particular index position represent the action associated with the triggering associated with the corresponding index position in pre_timer_increment.
pre_timer_calendar_ flag	integer	S	reserved for future use
pre_timer_calendar_id	ID	S	reserved for future use
pre_timer_increment	integer	R	Value in the first index position defines a time interval after which the timer is triggered if the activity has not been started. The interval is expressed in minutes and is counted from the start of the workflow. Values in the following index positions define intervals for second and subsequent triggerings of the timer if the activity has not been started. The intervals in these index positions are counted from the end of the previous interval.
pre_timer_name	string(64)	S	Contains names of the pre-timers defined for this activity
pre_timer_repeat_last	Boolean	S	Whether to repeat the last action specified in the pre_timer_action property.
property_name	string(256)	R	Contains names of extra properties of the activity
property_value	string(2000)	R	Value of the extra property defined at the corresponding index of property_name

Property	Datatype	Single or repeating	Description
r_condition_id	ID	S	Contains the object ID of a compound condition (a dm_cond_expr object).
r_condition_name	string(16)	R	Contains the names of route cases.
r_condition_port	string(16)	R	Contains the output port names. The ports are a particular index position are the ports associated with the route case condition at the corresponding index position in r_condition_name.
r_definition_state	integer	S	Identifies the state of this activity definition. Valid values are: 0, Draft 1, Validated 2, Installed
r_package_flag	integer	R	Indicates whether the package is visible and whether it is allowed to be empty. Valid values are: 0, meaning the package is invisible but cannot be empty 1, meaning the package is visible and cannot be empty 2, meaning the package is invisible but may be empty 3, meaning the package is visible and may be empty The default is 1.
r_package_id	ID	R	Contains the ID of a single package component. Not used if the package has more than one component.
r_package_label	string(32)	R	Contains a version label for the package.

Property	Datatype	Single or repeating	Description
r_package_name	string(128)	R	Contains the package name. The package name must be unique within the scope of the containing port.
r_package_oprtn	string(64)	R	Contains the operation to be performed by performers.
r_package_report_flag	integer	R	Indicates whether to report the package defined at the corresponding index position of the r_package_name
r_package_type	string(40)	R	Contains the type name of the package object.
r_performer_cond_id	ID	S	Object ID of dm_cond_expr object that records the conditions used to resolve the performer
r_performer_cond_ name	string(16)	R	Names of the conditions used to choose a performer
r_performer_cond_ user	string(2000)	R	Contains a user or group name, an alias that resolves to a user or group name, or a select query that returns a user or group name. The value at a particular index position applies to the condition specified at the corresponding index position in r_performer_condition_name.
r_port_name	string(16)	R	Contains the unique identifier of a port.
r_port_type	string(16)	R	Specifies the port type, one of: INPUT, OUTPUT, REVERT, or EXCEPTION.
r_predicate_id	ID	R	Object ID of a persistent predicate, to be evaluated for the corresponding route case.

Property	Datatype	Single or repeating	Description
repeatable_invoke	Boolean	S	Indicates whether this activity can be included multiple times or only once within a process definition. The default value is T (TRUE), meaning the activity may be included multiple times.
resolve_pkg_name	string(128)	S	Contains the name of the package whose associated alias set will be used to resolve the activity's performer alias. This property is used only if resolve_type is set to 1. If resolve_type is 1 and resolve_pkg_name is not set, then all incoming packages are checked in the order in which they are defined.
resolve_type	integer	S	Defines the resolution algorithm for the performer aliases. Valid values are: 0, Normal resolution path (default) 1, Use alias set associated with incoming package 2, Use alias set associated with user or group
sd_element_flag	integer	R	Indicates whether to report the structured data element at runtime for this activity. The value at a particular index position applies to the structured data element identified in sd_element_name property at the corresponding index position. Valid values are: 0, do not report (default) 1, report

Property	Datatype	Single or repeating	Description
sd_element_name	string(255)	R	Names of the structured data elements that are visible to the activity. The names must be a subset of the structured data element names specified in dm_process.sd_element_name.
sign_off_required	Boolean	S	Whether the activity requires a signoff. If this is set to T, the user or application must execute a Signoff method to complete the activity. The default is F (FALSE).
task_subject	string(255)	S	A user-defined string describing the task generated for this activity. The string may contain key phrases that are interpreted at runtime. The key phrases have the format: `$type.property` Type may be one of: dm_workflow dmi_workitem dmi_queue_item dmi_package dmi_package[x] If type is dm_workflow, dmi_workitem or either package option, you can specify any property of the type. If type is dmi_queue_item, you can specify any queue item property except task_subject.

Property	Datatype	Single or repeating	Description
transition_eval_cnt	integer	S	Defines how many tasks must be completed to trigger activity completion. This property is not valid for activities with a performer type of 9, Some Users Sequentially. Valid values are 0 and any positive integer. Setting this to 0 means that all tasks must be completed before the activity is completed. The default value is 0.
transition_flag	integer	S	Determines which output links are actually triggered when an activity completes. This value is used only if transition_type is set to 1. Valid values are: • 0, meaning trigger all selected ouptut links. • 1, Trigger only revert links if any are selected and ignore forward links. If there are no revert links selected, trigger selected forward links. • 2, Trigger only forward links if any are selected and ignore any selected revert links. If there are no forward links selected, trigger revert links. • 3, meaning trigger a revert link immediately if one is selected • 4, meaning trigger a forward link immediately if one is selected.

Property	Datatype	Single or repeating	Description
transition_max_ output_cnt	integer	S	Defines the maximum number of output ports that a performer can select. This property is valid only when transition_type is set to 1. A valid value is 0 or any positive integer. 0 means that there are no limits on the number of output ports the user can select. The default is 0.
transition_type	integer	S	Specifies the type of the transition condition. Valid values are: 0, Prescribed (Packages are forwarded to output ports when the post-condition is met.) 1, Manual (Activity performers specify the output ports with the Setoutput method.) 2, Automatic (Conditional routing logic computes the output ports.) The default value is 0.
trigger_event	string(64)	S	Contains the expected event type that can trigger this activity instance.
trigger_threshold	integer	S	Indicates the threshold of the trigger condition; any number between 1 and the total number of input ports. The default value is 0. 0 is only allowed for begin activities, which have no input ports.

Activity Group Instance

Purpose

Identifies an activity group instance of a workflow

Description

Supertype:
Subtypes:
Internal name: dmc_act_group_instance
Object type tag: 00

Properties

Table 10, page 62, lists the properties defined for the type.

Table 10. Properties defined for the activity group instance type

Property	Datatype	Single or repeating	Description
act_group_flag	integer	R	Contains the activity group flag of the activity The flag at each index position corresponds to the activity identified at the corresponding index position in act_name.
act_group_id	ID	S	ID of the activity group (dm_activity object)
act_group_name	string(128)	S	Name of the activity group
act_group_sequenceno	integer	S	The unique sequence number of the activity group instance

Property	Datatype	Single or repeating	Description
act_group_state	integer	S	State of the activity group instance Valid values are: 0, dormant 1, active 2, finished
act_name	string(128)	R	Contains the activity identifier as defined in the dm_process object
act_sequenceno	integer	R	Contains the unique sequence number The flag at each index position corresponds to the activity identified at the corresponding index position in act_name.
act_state	integer	R	Records the activity's current state Valid values and their corresponding states are: 0, dormant 1, active 2, finished
end_date	Time	S	Represents the end time of the activity group instance; set by the server when the workflow exits the activity group
start_date	Time	S	Represents the start time of the activity group instance; set by the server when the workflow enters the activity group
workflow_id	ID	S	ID of the workflow (dm_workflow object) the activity instance belongs to

Aggr Domain

Purpose

An aggr domain object describes the set of properties in an object type for which data dictionary information is defined.

Description

Supertype: Domain
Subtypes: None
Internal name: dm_aggr_domain
Object type tag: 51

Aggr domain objects are created and managed by the server. Users cannot create an aggr domain object.

Properties

Table 11, page 64, lists the properties defined for the type.

Table 11. Properties defined for the aggr domain type

Property	Datatype	Single or repeating	Description
attr_domain_id	ID	R	Object IDs of the domains that describe the properties.
attr_domain_name	string(32)	R	Currently unused.
attr_name	string(32)	R	The names of properties that have data dictionary information in their definitions.
code_objects	ID	R	Object IDs of the expr code objects holding source code that implements the expressions associated with the type, if any. Currently, only one value is allowed in this property.

Property	Datatype	Single or repeating	Description
i_dd_flags	integer	R	Used internally.
type_name	string(32)	S	Name of the object type to which the properties belong. This value is NULL for lifecycle state or other non-type overrides.

Alias Set

Purpose

Records aliases and their actual values.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dm_alias_set
Object type tag: 66

An alias set object contains one or more alias names that can be used as placeholders in template ACLs, in certain SysObject properties, in workflow definitions, and in Link and Unlink methods. Any user can create alias sets.

Properties

Table 12, page 66, lists the properties defined for the type.

Table 12. Properties defined for the alias set type

Property	Datatype	Single or repeating	Description
alias_category	integer	R	Identifies the category of the value defined at the corresponding index position in alias_value. Valid values and their associated categories are: 0, Unknown 1, User 2Group 3, User or Group 4, Cabinet path 5, Folder path 6, ACL name
alias_description	string(255)	R	User-defined description of the alias name at the corresponding index position in alias_name.

Property	Datatype	Single or repeating	Description
alias_name	string(32)	R	List of alias for the alias set. An alias_name value cannot contain a period (.).
alias_usr_ category	integer	R	User-defined category values for the values in the alias_value property.
alias_value	string(255)	R	The actual user or group name associated with the alias that appears at the corresponding index level in alias_name.
object_description	string(128)	S	User-defined description of the alias set.
object_name	string(32)	S	Name of the alias set. The name must be unique among all alias set objects in the repository and cannot contain a period (.) or be composed entirely of numerical digits.
owner_name	string(32)	S	Name of the user who owns the alias set.

App Ref

Purpose

An app ref object is used internally to reference an object used in an application.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_app_ref
Obect type tag: 07

Objects of this type reference objects used in a DAR file. Users cannot create app ref objects directly. They are created and managed by the server or Documentum Application Builder.

Properties

Table 13, page 68, lists the only property defined for the type

Table 13. Property defined for the app ref type

Property	Datatype	Single or repeating	Description
application_obj_id	ID	S	Object ID of the object referenced by the application.

Application

Purpose

An application object represents a packaged application that can be installed by the Documentum Application Builder.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_application
Object type tag: 08

Application objects represent packaged DAR files. Users cannot create application objects directly. They are created through the Documentum when a user creates a DAR file.

Properties

Table 14, page 69, lists the property defined for the type.

Table 14. Property defined for the application type

Property	Datatype	Single or repeating	Description
app_version	string(16)	S	Application version label.
application_object _id	ID	R	Object IDs of the objects in the application.
content_transfer _option	integer	R	Defines how to handle cabinets or folders when copying them to a target repository. Valid values are: 0, Copy all contained objects 1, Copy only the hierarchy (the directly or indirectly contained folders, but not the documents in the folders) 2, Copy just the cabinet or folder itself, but none of its contained objects 3, Copy only the cabinet or

Property	Datatype	Single or repeating	Description
			folder and its directly contained documents. The value at an index position applies to the object at the corresponding index position in application_object_id
copyright_string	string(255)	S	User-defined.
def_alias_set_id	ID	S	The object ID of the default alias set for the application. This is set through Developer Studio
post_install_proc _id	ID	S	Chronicle ID of the post-installation procedure
post_install_proc _label	string(32)	S	Version label of the post-installation procedure
pre_install_proc_id	ID	S	Chronicle ID of the pre-installation procedure
pre-install_proc _label	string(32)	S	Version label of the pre-installation procedure
target_loc_alias	string(64)	R	An alias that resolves to the location to which to copy the object in the application. The value at an index position applies to the object at the corresponding index position in application_object_id.
target_perm_alias	string(64)	R	An alias that resolves to the name of a template ACL. The value at an index position applies to the object at the corresponding index position in application_object_id.
target_owner_alias	string(64)	R	An alias that resolves to the name of the owner of the object. The value at an index position applies to the object at the corresponding index position in application_object_id.

Property	Datatype	Single or repeating	Description
upgrade_option	integer	S	Defines how to handle the object when the application is upgraded. The value at an index position applies to the object at the corresponding index position in application_object_id. Valid values are: 0, Overwrite the object 1, Ignore the object 2, Version the object
user_runnable	Boolean	S	Reserved for future use.

Aspect Relation

Purpose

Describes the compatibility of two aspects.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_aspect_relation
Object type tag: 37



Caution: This object type is for internal use only. It is not currently supported for external use.

An aspect relation object type describes the compatibility of two aspects.

Properties

Table 15, page 72, lists the properties defined for the type.

Table 15. Properties defined for the aspect relation type

Property	Datatype	Single or repeating	Description
child_aspect_name	string(64)	S	Reserved for internal use
parent_aspect_name	string(64)	S	Reserved for internal use
relation	integer	S	Reserved for internal use

Aspect Type

Purpose

Stores the files that implement an aspect.

Description

Supertype: Module
Subtypes: None
Internal name: dmc_aspect_type
Object type tag: 0b

An aspect type object is a folder that contains the implementation classes and other information about one aspect. (An aspect is a custom behavior implemented for a particular object type class.) Aspect type folders are stored in the repository under System/Business Objects/Aspects.

Properties

Table 16, page 73, lists the properties defined for the type.

Table 16. Properties defined for the aspect type object type

Property	Datatype	Single or repeating	Description
aspect_category	string(32)	R	Used internally
attaching_aspect_policy	string(128)	R	Used internally
copy_policy	integer	S	Used internally
detaching_aspect_policy	string(128)	R	Used internally
i_attr_def	string(27)	S	Used internally
is_data_shared	Boolean	S	Used internally
target_object_type	string(27)	R	Used internally
version_policy	integer	S	Used internally

Assembly

Purpose

An assembly object describes a component of a virtual document.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_assembly
Object type tag: 0d

Assembly objects are created when a user creates an assembly, a snapshot of the virtual document at particular point in time. Users must have at least Version permission for the object identified in the book_id property to modify an assembly object.

Properties

Table 17, page 74, lists the properties defined for the type.

Table 17. Properties defined for the assembly type

Property	Datatype	Single or repeating	Description
a_contain_desc	string(255)	S	Used by Documentum clients to manage XML documents in assemblies. For more information, refer to *Content Server Fundamentals*.
a_contain_type	string(32)	S	Used by Documentum clients to manage XML documents in assemblies. For more information, refer to *Content Server Fundamentals*.
book_id	ID	S	Object ID of the topmost containing document in the virtual document structure that contains the component described by this assembly object.

Property	Datatype	Single or repeating	Description
component _chron_id	ID	S	Chronicle ID of the component represented by the assembly object.
component_id	ID	S	Object ID of the component of the virtual document
depth_no	integer	S	Depth of the component within the levels of the virtual document identified by book_id.
i_partition	integer	S	Identifies the partition where the object resides.
order_no	integer	S	The ordering of the component within the virtual document identified by the book_id.
parent_id	ID	S	Object ID of the document that directly contains the virtual document described by this assembly object.
path_name	Oracle: string(740) SQLServer: string(765) Sybase string(600) DB2: string(740)	S	The path from the top-most containing document (identified in the book_id property) to the component. Path names are comprised of the object names of the nodes delimited by forward slashes. If the path is longer than the property, the path is truncated from the end of the path.

Attachments Folder

Purpose

Used to organize email message attachments.

Description

Supertype: Folder
Subtypes: None
Internal name: dm_attachments_folder
Object type tag: 0b

The attachments folder object serves, in conjunction with folders and cabinets, to organize the email attachments stored in your repository.

Properties

Attachments Folders inherit all of their properties from their supertype, the Folder type.

Audit Trail

Purpose

An audit trail object stores information about one audited event.

Description

Supertype: Persistent Object
Subtypes: Audittrail Acl, Audittrail Group
Internal Name: dm_audittrail
Object type tag: 5f

Audit trail objects are created automatically by the server after a user initiates auditing for a particular event. Automatic auditing is provided for system events, such as a checkin or checkout, workflow events, and lifecycle events.

Users and applications can also create audit trail objects to record the occurrence of user-defined events.

Audit trail objects have ten properties whose use is dependent on the event. These properties are the id_*n* and string_*n* properties. The *Content Server Administration Guide* provides a complete list of auditable events and describes the use of the id_*n* and string_*n* properties for server-generated audit trail objects.

Properties

Table 18, page 77, lists the properties defined for the type.

Table 18. Properties defined for the audit trail type

Property	Datatype	Single or repeating	Description
acl_domain	string(32)	S	Owner of the ACL associated with the object being audited. This is null for objects that are not SysObjects or SysObject subtypes.
acl_name	string(32)	S	Name of the ACL associated with the object being audited. This is null for objects that are not SysObjects or SysObject subtypes.

Property	Datatype	Single or repeating	Description
application_code	string(64)	S	The application code or codes defined for the session in which the event occurred. Multiple codes are separated by commas. **Note:** The list of codes is truncated if necessary.
attribute_list	For all except DB2: string(2000) For DB2: string(1300)	S	Comma-separated list of audited properties and their values at the time of the event. If the list is longer than the length of the property, the overflow is stored in the dmi_audittrail_attrs object identified in the attribute_list_id property.
attribute_list_id	ID	S	Object ID of the dmi_audittrail_attrs object that contains the overflow values from the attribute_list property.
attribute_list_old	string(1000), if the RDBMS (Sybase or DB2) page size is 8K or greater string(1) if RDBMS (Sybase or DB2) page size is 4K	S	Comma separated list of audited properties and their old values if the values changed. If the list is longer than the length of the property, the overflow is stored in the dmi_audittrail_attrs object identified in the attribute_list_id property.
audit_signature	string(255)	S	Signed hash of the audit trail entry.

Property	Datatype	Single or repeating	Description
audit_version	integer	S	Indicates how the signature was created if the entry is signed. The only valid value is 1, to indicate that 5.2 Content Server created the value in audit_signature. Otherwise, this is not set.
audited_obj_id	ID	S	Object ID of the object being audited For a checkin event, this is the object ID of the new version of the object. For workflow events, this is always the object ID of the process object. For dm_move_content events, this is always the object ID of the content object.
audited_obj_vstamp	integer	S	The i_vstamp value of the object identified in the audited_obj_id property.
chronicle_id	ID	S	Chronicle ID of the object being audited if the object is a SysObject or SysObject subtype. Otherwise, this is the object ID of the object being audited.
controlling_app	string(32)	S	The object's controlling application. This is the value in the a_controlling_app property of the object being audited.
current_state	string(32)	S	Name of the current lifecycle state of the object being audited.
event_description	string(64)	S	Name of the event being audited in a user-friendly form.

Property	Datatype	Single or repeating	Description
event_name	string(64)	S	Actual name of the event being audited.
event_source	string(64)	S	Source of the event. Some possible values are: lifecycle electronic signature system unspecific workflow
host_name	string(128)	S	Name of the host on which the Content Server that generated the audit trail entry resides
i_audited_obj_class	integer	S	Identifies what type of object is identified in audited_obj_id. Values are: -1, meaning the event has no audited object (for example, dm_audit or dm_unaudit events) 0, dm_sysobject and subtypes 1, for dm_acl 2, for dm_user and subtypes 3, for dm_group 4, for dmr_content
i_is_archived	Boolean	S	T means that the audit trail entry has been archived. This is set to F, meaning that the entry has not been archived, when the audit trail entry is created and reset to T when the MARK_AS_ARCHIVED administration method is executed against the entry.
id_1	ID	S	event-dependent object ID

Property	Datatype	Single or repeating	Description
id_2	ID	S	event-dependent object ID
id_3	ID	S	event-dependent object ID
id_4	ID	S	event-dependent object ID
id_5	ID	S	event-dependent object ID
object_name	string(255)	S	Name of the object being audited. For SysObjects and ACLs, this is the object_name property value. For groups, this is the group_name value.
object_type	string(32)	S	Object type of the object being audited.
owner_name	string(32)	S	Name of the owner of the object being audited. The value may be an individual user or a group name.
policy_id	ID	S	Object ID of the lifecycle associated with the object being audited.
r_gen_source	integer	S	The source of the audit trail. Values are: 1, for a system-generated audit trail 0, for a user-generated audit trail
registry_id	ID	S	Object ID of the registry object that stores the registration for the event that generated this audit trail entry.
session_id	ID	S	ID of the session in which the event occurred.
string_1	string(200)	S	event-dependent string
string_2	string(200)	S	event-dependent string
string_3	string(200)	S	event-dependent string
string_4	string(200)	S	event-dependent string
string_5	string(200)	S	event-dependent string

Property	Datatype	Single or repeating	Description
time_stamp	date	S	The local time at which the audit trail was generated
time_stamp_utc	date	S	The UTC (or GMT) time at which the event occurred
user_id	ID	S	Object ID of the dm_user object representing the user whose task caused the event to occur.
user_name	string(32)	S	Name of the user whose task caused the event to occur.
version_label	string(16)	S	The implicit (numeric) version label of the object being audited. This is null for objects that are not SysObjects or SysObject subtypes.
workflow_id	ID	S	Object ID of the dm_workflow for workflow events.

Audit Trail ACL

Purpose

An audit trail acl object records information about one audited event for an ACL.

Description

Supertype: Audit Trail
Subtypes: None
Internal Name: dm_audittrail_acl
Object type tag: 5f

Audit trail acl objects are created when a dm_save, dm_saveasnew, or dm_destroy registered event occurs on an ACL. (The audited ACL is identified by the inherited audited_obj_id property.)

Properties

Table 19, page 83, lists the properties defined for the type.

Table 19. Properties defined for the audit trail acl type

Property	Datatype	Single or repeating	Description
accessor_name	string(32)	R	The ACL user or group entry on which the operation at the corresponding index position in accessor_operation was performed.
accessor _operation	string(1)	R	The operation performed on the user or group entry identified in the corresponding position in accessor_name. Valid values are: U, for update I, for insert D, for delete

Property	Datatype	Single or repeating	Description
accessor_permit	integer	R	Value in the ACL's r_accessor_permit property for the ACL entry identified in the dm_audittrail_acl. accessor_name property at the corresponding index position.
accessor _xpermit	integer	R	Value in the ACL's r_accessor_xpermit property for the ACL entry identified in the dm_audittrail_acl. accessor_name property at the corresponding index position.
acl_class	integer	S	Value in the audited ACL's acl_class property.
application_permit	string(128)	R	Value in the ACL's application_permit property for the ACL entry identified in the dm_audittrail_acl. accessor_name property at the corresponding index position.
description	string(128)	S	Value in the audited ACL's description property.
globally _managed	Boolean	S	Value in the audited ACL's globally_managed property.
is_group	Boolean	R	Value in the ACL's r_is_group property for the ACL entry identified in the dm_audittrail_ acl.accessor_name property at the corresponding index position.
is_internal	Boolean	S	Value in the ACL's r_is_internal property.
permit_type	integer	R	Value in the ACL's r_permit_type property for the ACL entry identified in the dm_audittrail_ acl.accessor_name property at the corresponding index position.

Audit Trail Attrs

Purpose

An audit trail attrs object stores the overflow property name and value pairs from the attribute_list property of an audit trail object.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dmi_audittrail_attrs
Object type tag: 00

Audit trail attr objects are created automatically, when needed, by Content Server. They record overflow property names and their values if the list of properties to be recorded is too long to fit into the attribute_list property of the associated audit trail object.

Properties

Table 20, page 85, lists the properties defined for the type.

Table 20. Properties defined for the audit trail attrs type

Property	Datatype	Single or repeating	Description
attribute_list	string(2000)	R	Comma-separated list of audited properties and their values at the time of the event.
attribute_list_old	string(1000) if RDBMS (Sybase or DB2) page size is 8K or greater string(1) if RDBMS (Sybase or DB2) page size is 4K	R	Comma separated list of audited properties and their old values if the values changed.
audit_obj_id	ID	S	Object ID of the associated audit trail object.

Audit Trail Group

Purpose

An audit trail group object records information about an audited event for a group.

Description

Supertype: Audit Trail
Subtypes: None
Internal Name: dm_audittrail_group
Object type tag: 5f

Audit trail group objects are created when a dm_save, saveasnew, or dm_destroy registered event occurs on a group. (The audited group is identified in the inherited audited_obj_id property.)

Properties

Table 21, page 86, lists the properties defined for the object type.

Table 21. Properties defined for the audit trail group type

Property	Datatype	Single or repeating	Description
alias_set_id	ID	S	Value in the audited group's alias_set_id property.
description	string(128)	S	Value in the audited group's description property.
globally _managed	Boolean	S	Value in the audited group's globally_managed property.
group_admin	string(32)	S	User or group allowed to modify the audited group.
group_address	string(80)	S	Value in the audited group's group_address property.
group_class	string(32)	S	Value in the audited group's group_class property.

Property	Datatype	Single or repeating	Description
group_source	string(16)	S	Identifies the source of the audited group. The only valid value is LDAP, meaning the group was created by importing an LDAP group.
groups_names	string(32)	R	Names of groups that are users in the audited group on which an operation occured. The operation is recorded in the groups_names_operation property in the corresponding index position.
groups_names _operation	string(1)	R	The operation that occurred on the group identified in the groups_name property at the corresponding index position. Valid values are: U, for update I, for insert D, for delete
is_dynamic	Boolean	S	T (TRUE) means the audited group is a dynamic group. F (FALSE) means the group is not a dynamic group.
is_dynamic_default	Boolean	S	Controls whether users in the audited group's list of potential users are considered members of the group by default when they connect to the respository. T (TRUE) means that users are treated as members of the group. F (FALSE) means that users are not treated as group members when they connect.
is_private	Boolean	S	Value in the group's is_private property .

Property	Datatype	Single or repeating	Description
users_names	string(32)	R	Names of individual users in the audited group on which an operation occured. The operation is recorded in the groups_names_operation property in the corresponding index position.
users_names _operation	string(1)	R	The operation that occurred on the user identified in the users_name property at the corresponding index position. Valid values are: U, for update I, for insert D, for delete

Auth Config

Purpose

An auth config object contains the names of the primary and backup domain controllers for a particular domain.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dm_auth_config
Object type tag: 00

The values in an auth config object are used when you authenticate users in repository on a UNIX platform against a Windows domain. A repository can have only one auth config object. It is created and its values set through Documentum Administrator, when unix_domain_used is defined in the auth_protocol property of the docbase config object.

Properties

Table 22, page 89, lists the properties defined for the type.

Table 22. Properties defined for the auth config type

Property	Datatype	Single or repeating	Description
domain_name	string (32)	R	The name of a Windows domain
primary _controller	string(32)	R	The name of the machine that is the primary domain controller for the domain named at the corresponding index position in domain_name.
backup _controller	string(32)	R	The name of the machine that is the backup domain controller for the domain named at the corresponding index position in domain_name.

Blob Store

Purpose

A blob store object represents a blob storage area.

Description

Supertype: Store
Subtypes: None
Internal name: dm_blobstore
Object type tag: 40

Blob store objects represent blob store storage areas. Content stored in blob stores in stored in tables in the repository. You can store a maximum of 64 Kbytes of data in a blob store storage area.

Properties

Table 23, page 90, lists the properties defined for the type.

Table 23. Properties defined for the blob store type

Property	Datatype	Single or repeating	Description
ascii	Boolean	S	Indicates whether the contents are ASCII strings or arbitrary sequences of 8-bit characters.
selectable	Boolean	S	Not currently used.

BOCS config

Purpose

Defines the configuration of a BOCS server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_bocs_config
Object type tag: 08

A bocs config object defines the configuration of a BOCS (Branch Office Caching Services) server. All bocs config objects reside in the global registry, in the /System/NetworkLocation folder. The objects are created and managed using Documentum Administrator.

Properties

Table 24, page 91, lists the properties defined for the type.

Table 24. Properties defined for the bocs config type

Property	Datatype	Single or repeating	Description
base_url	string(240)	R	Base URL to use when accessing the BOCS server using the protocol identified at the corresponding index position in supported_protocol.
delivery_type	string(4)	S	Whether the DMS server can send messages to the BOCS server or must wait for the BOCS server to pull messages from the DMS server. Valid values are: push, DMS can send messages pull, DMS must wait for BOCS to pull messages The default value is push.

Property	Datatype	Single or repeating	Description
docbase_names	string(255)	R	List of repository names. The list is either an inclusion list or exclusion list depending on the value of is_inclusion_list.
external_url	string(240)	R	BOCS external URL
external_url_type	string(32)	R	BOCS external URL type. The only value currently supported is *ws_push*. This value indicates a proxy URL for the BOCS message URL. This is used by DMS to deliver messages to BOCS when BOCS is in push mode behind a proxy.
is_inclusion_list	Boolean	S	T (TRUE) means that the list of repositories in docbase_names is a list of repositories that the BOCS server may service. F (FALSE) means that the list of repositories in docbase_names is an exclusion list; that is, a list of repositories that this BOCS server may not service.
network_locations	string(80)	R	The network identifiers of the network locations served by the BOCS server. The order of the identifiers in this property defines the relative closeness of the location to the BOCS server.

Property	Datatype	Single or repeating	Description
rw_capability	integer	S	Indicates whether the BOCS server is enabled or disabled and whether it is capable of asynchronous write. Valid values are: 0, meaning disabled 1, meaning enabled but no asynchronous write 9, meaning enabled with asynchronous write capabilities
server_major_version	string(16)	S	Indicates the major version of the BOCS server. For a BOCS 6.0 server, this must be set to 2.
server_minor_version	string(16)	S	Indicates the minor version level of the BOCS server. For a BOCS 6.0 server, this must be set to 0.
supported_protocol	string(6)	R	Set of network protocols supported by the BOCS server Values in this property must all lowercase. For example: http or https.

Builtin Expr

Purpose

A builtin expr object stores information needed to execute built-in server functions.

Description

Supertype: Expression
Subtypes: None
Internal name: dm_builtin_expr
Object type tag: 54

Builtin expr objects store information needed to execute built-in server functions. Content Server creates and manages builtin expr objects. Users cannot create them.

Properties

Table 25, page 94, lists the property defined for the type.

Table 25. Property defined for the builtin expr type

Property	Datatype	Single or repeating	Description
builtin_tag	integer	S	Specifies which server function to invoke. Valid values are: 1, for NULL 2, for NOW 3, for TODAY 4, for TOMORROW 5, for YESTERDAY 6, for USER None of the functions require passed arguments.

CA Store

Purpose

A ca store object represents a retention—enabled storage system known to Content Server.

Description

Supertype: Store
Subtypes: None
Internal name: dm_ca_store
Object type tag: 6d

A ca store object represents a retention—enabled storage system. A retention-enabled storage system enforces retention policies at the device level, providing an additional level of content retention above that provided by Content Server alone. The object internal name, dm_ca_store, is retained for historical reasons, as the first supported retention-enabled storage system was a content-addressable storage system. A content addressable storage system uses a content address instead of a directory path to locate content stored in the system.

Properties

Table 26, page 95, lists the properties defined for the type.

Table 26. Properties defined for the CA store type

Property	Datatype	Single or repeating	Description
a_content_attr_description	string(128)	R	User-defined description of the property named at the corresponding position in a_content_attr_name
a_content_attr_name	string(64)	R	Names of the content metadata fields whose values will be set using a Setcontentattrs method. The names may not contain spaces. This property can have a maximum of 62 values. Do not include the metadata field identified in a_retention_attr_name, if any.

Property	Datatype	Single or repeating	Description
a_default_ retention_date	Date	S	The default retention value for content stored in this storage system. The value is ignored if a_retention_ attr_required is set to T (TRUE), a_retention_attr_name is set, or if default_retention_days is set. **Note:** default_retention_days is a property inherited from dm_store.
a_plugin_id	ID	S	Object ID of the plugin object whose content is the plugin library that implements this ca storage system.
a_retention_attr_ name	string(64)	S	Identifies the content metadata field that stores the retention value for the content. The metadata field name may not contain spaces. Do not use a metadata field defined in a_content_attr_name. The value of this property may not be changed after the storage area is created.
a_retention_attr_ required	Boolean	S	T (TRUE) means that the content stored in this system must have a retention period. If set to T, you must set a_retention_attr_name also. The default is F (FALSE).
a_storage_params	string(1024)	R	List of parameters specific to the retention—enabled storage system. For Centera systems: The first index position ([0])is reserved for the IP address or addresses for the Centera host machine or machines. The value may include the path to a Centera profile. The second index position ([1]) is reserved for embedded blob configuration. If set, the value is in the format: `pool_option:embedded_blob:`*`size_ in_KB`

Property	Datatype	Single or repeating	Description
			This instructs the Centera system to store all content files less than or equal to the specified size as embedded blobs. (Refer to the *Content Server Administration Guide* for a complete description of this feature.)

Cabinet

Purpose

The cabinet object represents the highest level of organization visible to end users in a repository.

Description

Supertype: Folder
Subtypes: None
Internal name: dm_cabinet
Object type tag: 0c

The cabinet object represents the highest level of organization visible to end users in a repository. All folders, documents, and other objects (except cabinets) that are subtypes of the SysObject type are stored in cabinets. A cabinet is basically a folder that cannot be placed inside another folder or a cabinet. Users must have Superuser, Sysadmin, or Create Cabinet user privilege to create or destroy a cabinet. However, users can change a cabinet's properties if they have Write permission for the cabinet.

Properties

Table 27, page 98, lists the only property defined for the type.

Table 27. Properties defined for the cabinet type

Property	Datatype	Single or repeating	Description
is_private	Boolean	S	Indicates whether the cabinet is private or public. If set to TRUE, the cabinet is private. If set to FALSE, the cabinet is public. The default is FALSE. This property is not used by Content Server for security or any other use. It is intended for use by client applications.

Cache Config

Purpose

A cache config object identifies a group of queries, objects, or both to be cached on the client and refreshed at defined intervals.

Description

Supertype: SysObject
Subtypes: None
Internal Name: dm_cache_config
Object type tag: 08

A cache config object identifies a group of queries, objects, or both to be cached on the client and refreshed at defined intervals.

Properties

Table 28, page 99, lists the properties defined for the type.

Table 28. Properties defined for the cache config type

Property	Datatype	Single or repeating	Description
cache_element_query	string(1500)	R	The queries that return the query results, objects or both to be cached. The value can be any valid query. Each index position can contain one query.
cache_element_type	string(32)	R	Indicates whether the value at the corresponding position in cache_element_query identifies a query or object to be cached. Valid values are: query object

Property	Datatype	Single or repeating	Description
client_check _interval	integer	S	Defines how often the client calls the server to obtain updated information about the cache config object. The default value is 0, meaning that the client requests updated information every time an API call specifies the cache config object. The value is interpreted in seconds.
i_query_result _hash	string(100)	S	Last computed hash of the query results. This value is used to compare executions of the cached queries.
r_last_changed _date	Date	S	Time at which the server last validated the queries and found a change.
r_last_checked _date	Date	S	Time at which the server last validated the queries.
server_check _interval	integer	S	Interval, in seconds, at which the server revalidates the queries in the cache configuration. The default value is 0, meaning that the server reruns all the queries every time an API call specifies the cache config object and the client_check_interval has expired.

Category

Purpose

Represents a taxonomy in a repository.

Description

Supertype: Folder
Subtypes: Taxonomy
Internal Name: dm_category
Object type tag: 0b

A category object is a folder in the folder structure that represents a taxonomy in a repository. Every category object must be linked to at least one taxonomy object or other category object. (For more information about categories and taxonomies, refer the Content Intelligence Services documentation.)

Properties

Table 29, page 101, lists the properties defined for the category type and the inherited properties whose use is specific to the category object type.

Table 29. Properties defined for the category type

Property	Datatype	Single or repeating	Description
a_status	string(16)	S	Describes the development and deployment status of the category. Valid values are: offline online
allowed_ operations	string(32)	R	List of allowed operations for the category. Valid values are: user_browse user_subscribe user_subscribe_mandatory
candidate_ threshold	integer	S	Relevence level at which a document is automatically routed for approval to the category owner. Values are interpreted as a percentage, from 1 to 100.

Property	Datatype	Single or repeating	Description
category_evidence	string(255)	R	Identifies other categories whose evidence will be considered as evidence for this category. Values for this property are set using the Content Intelligence Client.
category_owner	string(32)	R	The owner of the category. **Note:** Only one owner is currently supported.
child_count	integer	S	Number of categories linked directly to this category.
class_id	ID	S	Object ID of the category class to which this category belongs. Defaults to the value of the class_id property in the taxonomy.
description	string(255)	S	User-defined description of the category.
definition_type	string(32)	S	Identifies the type of the definition. The definition is comprised of the values in the category_evidence, keyword_evidence, candidate_threshold, and on_target_threshold properties. These values determine which documents can be automatically assigned to the category. Valid values are: **none**, which indicates that the category has no definition or the definition is to be ignored, meaning documents cannot be automatically a ssigned to the category **simple**, which indicates that the category has a simple definition
keyword_evidence	string(255)	R	List of keywords in document text considered to be evidence for the category. The values in this property are set using Content Intelligence Client.
object_name	string(255)	S	The name of the category. The name must be unique among all categories within the category class.

Property	Datatype	Single or repeating	Description
on_target_ threshold	integer	S	The relevance level at which a document is automatically assigned to the category. The value is interpreted as a percentage, from 1 to 100.
qualifiers	string(255)	R	List of initial conditions that a document must meet to qualify for testing against the evidence for the category. The values in this property are set using Content Intelligence Client.
supported_ language	string(32)	R	List of language codes for all supported languages. The entry at each position must have a category name in the specified language at the matching index position in translated_name.
translated_name	string(255)	R	List of the category name in all supported languages. The name at each index position must be in the language specified by the language code in the matching index position in supported_language.

Category Assign

Purpose

Records the assignment of an object to a particular category.

Description

Supertype: Relation
Subtypes: None
Internal Name: dm_category_assign
Object type tag: 37

A category assign object records the assignment of an object to a particular category. Category assign objects are created automatically when an object is assigned or proposed for assignment to a category folder.

Note: The data dictionary label for this type is Category Assignment.

Properties

Table 30, page 104, lists the properties defined for the type and any inherited properties that have a use specific to the object type.

Table 30. Properties defined for the category assign type

Property	Datatype	Single or repeating	Description
active_assignment	Boolean	S	Whether the object represented by this category assign object is visible to users. T (TRUE) means the object is available to end users. The value is T when assign_type value is active and the assign_status value begins with "assigned". F (FALSE) means the object is not available.

Property	Datatype	Single or repeating	Description
assign_status	string(32)	S	Describes the assignment operation. Valid values are: assigned_auto assigned_manual assigned_approved assigned_pending_remove assigned_final pending_assign removed_auto removed_manual removed_ approved removed_final
assign_type	string(32)	S	Identifies whether the assignment took place during a test or production run. Valid values are: text active
child_id	ID	S	The i_chronicle_id value of the assigned object.
document_id	ID	S	Object ID of the assigned object.
modifier	string(32)	S	Name of the user or process that last modified the assignment. For all automatic assignment operations, this value is CIS.
parent_id	ID	S	Object ID of the category to which the object is assigned.
permanent_link	Boolean	S	Whether to maintain the assignment across versions of the assigned object. The default is F (FALSE).

Property	Datatype	Single or repeating	Description
prev_assign_status	string(32)	S	Describes the assigned object's previous assignment, if any. Valid values are: assigned_auto assigned_manual assigned_approved assigned_pending_remove assigned_final pending_assign removed_auto removed_manual removed_ approved removed_final
prev_modifier	string(32)	S	Name of the user or process that performed the object's previous assignment. For all automatic assignments, this value is CIS.
relation_name	string(32)	S	Name of the relationship between the assigned object and the category. The only legal value is dm_category_assign.
relevance	integer	S	The relevance number of the assigned object. Values range from 1 to 100, with 100 meaning fully relevant.
run_id	ID	S	The object ID of the dm_docset_run object that identifies the run that classified and assigned the object. This property can be a NULL value.

Category Class

Purpose

Stores the definition of a class of categories and the default property values and behaviors for the included categories.

Description

Supertype: SysObject
Subtypes: None
Internal Name: dm_category_class
Object type tag: 08

A category class object stores the definition of a class of categories and the default property values and behaviors for the included categories. Category class objects are created using Content Intelligence Client.

Properties

Table 31, page 107, lists the properties defined for the type and those inherited properties that have a use specific to the object type.

Table 31. Properties defined for the category class type

Property	Datatype	Single or repeating	Description
cat_evidence_conf	string(32)	S	Default confidence level for all explicit (not propagated) category evidence for categories in this class. Valid values are: certain high medium low auxilliary negative off *numeric* *numeric* is any value from 0 to 100.

Property	Datatype	Single or repeating	Description
description	string(255)	S	User-defined desrcription of the category class.
evidence_prop_conf	string(32)	S	Default confidence level for all propagated category evidence for categories in this class. Valid values are: certain high medium low auxilliary negative *numeric* *numeric* is any value from 0 to 100.
evidence_prop_type	string(32)	S	Default value for all automatic (propagated) category evidence generation for categories in the class. Valid values are: **off**, which turns off evidence propagation **@child**, which propagates evidence from any child to the parent category **@parent**, which propagates any evidence for the parent category to immediate children
implied_keyword_conf	string(32)	S	Default keyword confidence level for all categories in this class. Valid values are: certain high medium low auxilliary negative off *numeric* *numeric* is any number from 0 to 100.

Property	Datatype	Single or repeating	Description
implied_keyword_ style	string(64)	S	Default implied keyword style for categories in this class. Valid values are combinations of the following: active stem phrase_order_exact
keyword_conf	string(32)	S	Default keyword confidence level for all categories in the class. Valid values are: certain high medium low auxiliary negative off *numeric* *numeric* is an integer from 0 to 100.
keyword_style	string(64)	S	Default keyword style for categories in this class. Valid values are combinations of: active stem phrase_order_exact
object_name	string(255)	S	Category class name. The name must be unique among the categories that belong to the class.
source	string(32)	S	User-defined text identifying the source of the taxonomy and category objects in this class. For example: DCTM or Semio.
target_attribute	string(32)	S	Name of the property in a repository object type to be used for storing all document category assignments. The property must be a repeating property.

Change Record

Purpose

Stores information used internally to ensure consistency of cached information.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dmi_change_record
Object type tag: 33

A change record object stores information used internally to ensure consistency of cached information. There is one change record object for each repository.

Properties

Table 32, page 110, lists the properties defined for the type.

Table 32. Properties defined for the change record type

Property	Datatype	Single or repeating	Description
cache_change _count	integer	S	Number of changes affecting the server's global cache.
dd_change _count	integer	S	Number of times the data dictionary has been changed.
group_change_count	integer	S	Number of times any group object has been changed.
reinit_change _count	integer	S	Number of times the main server thread (parent server) has been reinitialized.
sys_change _count	integer	S	Number of changes made to object types and format objects.
storage_change _count	integer	S	Number of times that a storage area has changed state.

Property	Datatype	Single or repeating	Description
type_change _count	integer	S	Number of times that a type in the global cache has been changed.
user_change_count	integer	S	Number of times any user object has been changed.

CI Config

Purpose

Stores Content Intelligence Services configuration options.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_ci_config
Object type tag: 08

A CI config object stores Content Intelligence Services configuration options. CI config objects are created when a repository is enabled for Content Intelligence Services. Enabling occurs when Content Intelligence Client is installed or the DAR file for Web Publisher is installed.

Properties

Table 33, page 112, lists the properties defined for the type.

Table 33. Properties defined for the CI config type

Property	Datatype	Single or repeating	Description
assign_as_attribute	Boolean	S	Whether reflecting the category assignments as document properties is enabled in the repository.
assign_as_link	Boolean	S	Whether assigning objects to folders is enabled in the repository.
auto_proc_enabled	Boolean	S	Whether automatic assignment is enabled in the repository.
auto_user	string(32)	S	User account that is used to generate and process the automatic document processing queue.
auto_workflow	Boolean	S	Whether automated object assignment initiates a confirmation workflow.
cat_owner_suggest	Boolean	S	Not currently used.

Property	Datatype	Single or repeating	Description
category_object_type	string(32)	S	Name of the object type used to manage categories. This is set to dm_category by default.
ci_client_enabled	Boolean	S	Whether the Content Intelligence Client is enabled on the repository.
ci_enabled	Boolean	S	Whether Content Intelligence functionality is enabled on the repository.
ci_server_host_dev	string(80)	S	Host on which the Content Intelligence Services development server resides.
ci_server_host_prod	string(80)	S	Host on which the Content Intelligence Services production server resides.
impersonate_assign	Boolean	S	T (TRUE) allows both automatic and manual document assignments to be performed using the designated superuser account. **Note:** The superuser account is identified using Content Intelligence Client.
manual_proc_enabled	Boolean	S	Whether manual processing is enabled on the repository.
manual_user	string(32)	S	User account that is used to generate and process the manual document processing queue.
manual_workflow	Boolean	S	Whether manual object assignment requires a confirmation workflow.
root_admin_path	string(255)	S	Folder path of the root folder from which all administration information managed.
root_category_path	string(255)	S	Folder path of the root folder from which all categories and taxonomies are managed.

Property	Datatype	Single or repeating	Description
sync_mode	string(32)	S	Identifies how data synchronization with the Content Intelligence server occurs. The only valid vaue is auto.
taxonomy_object_type	string(32)	S	Name of the object type used to manage taxonomies in the repository. This is set to dm_taxonomy by default.

Class

Purpose

Represents a model for a Smart Container.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_class
Object type tag: 08

A class object represents a model for a Smart Container. The content of the object is the XML representation of the Smart Container model. Every object of this type has an associated dm_logical_version aspect. This object type is installed with the Smart Container DAR file during repository configuration.

Properties

Table 34, page 115, lists the properties defined for the type.

Table 34. Properties defined for the class type

Property	Datatype	Single or repeating	Description
class_category	string(255)	S	Category of the model
class_description	string(1024)	S	Description of the model
metamodel_id	ID	S	Object ID of the dmc_metamodel object from which the model represented by this class object was derived.

Table 35, page 116, lists the properties added by a system-defined default aspect for the type.

Table 35. Default aspect properties defined for the class type

Property	Datatype	Single or repeating	Description
logical_version	string(32)	S	Supports logical versioning of persistent objects From the aspect named dm_logical_aspect

Client Registration

Purpose

Records registration information about client applications.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_client_registration
Object type tag: 08

A client registration object records the identity of a particular privileged DFC instance. Client registration objects are stored in the global registry, in the /System/Client Registration folder. They are protected by the dm_acl_registration ACL. This ACL grants Browse permission to dm_world and Delete permission to dm_owner and the admingroup group.

Properties

Table 36, page 117, lists the properties defined for the type.

Table 36. Properties defined for the client registration type

Property	Datatype	Single or repeating	Description
client_id	string(40)	S	Unique identifier for the client instance
host_name	string(128)	S	Name of host on which the client is installed
is_validated	Boolean	S	Identifies whether the entry has been checked for authenticity. T means that the entry has been checked. F means that the entry has not been checked.
public_key_identifier	string(40)	S	Points to the public key certificate object that holds the public key for this client.

Client Rights

Purpose

Records the roles that a particular instance of a privileged client is allowed to use.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_client_rights
Object type tag: 08

A client rights object records the roles that a particular instance of a client application is allowed to use within a repository. There must be a client rights object in each repository in which a privileged DFC instance will invoke the roles. Client rights objects are created using Documentum Administrator. You must have Superuser privileges to create a client rights object. The objects are stored in the /System/Client Rights folder, and are assigned the dm_acl_superusers ACL.

Properties

Table 37, page 118, lists the properties defined for the type.

Table 37. Properties defined for the client rights type

Property	Datatype	Single or repeating	Description
allow_all_roles	Boolean	S	T means the client may use any role. Setting this to T is the equivalent of listing all roles in allowed_roles.
allow_all_priv_ modules	Boolean	S	T means the client may use any role invoked by an installed module.
allowed_roles	string(32)	R	List of the roles that the client is allowed to use
client_id	string(40)	S	Unique identifier for the client instance

Property	Datatype	Single or repeating	Description
host_name	string(128)	S	Name of host on which the client is installed
principal_auth_priv	Boolean	S	T means that the client is allowed to create sessions for users without user credentials. This enables Principal Mode support for a DFC instance.
public_key_identifier	string(40)	S	Points to the public key certificate object that holds the public key for this client.

Completed Workflow

Purpose

Records information about completed workflows

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_completed_workflow
Object type tag: 00

A completed workflow object stores information from audit trail records about a completed workflow. The objects are created by the dm_WfReporting job and used by Webtop's aggregrate workflow reporting tool. To fully populate the properties in these objects, you must be auditing all workflow events.

The object type is created by a script when Content Server is installed.

Properties

Table 38, page 120, lists the properties in the type.

Table 38. Properties defined for the completed workflow type

Property	Datatype	Single or repeating	Description
active_duration	double	S	Length of time the workflow was in the active state. This value is computed from the audit trail entries for the dm_startworkflow, dm_changestateworkflow, dm_finishworkflow, and dm_abortworkflow events.

Property	Datatype	Single or repeating	Description
complete_date_utc	Date	S	The date and time on which the workflow is finished or aborted. This value is taken from the audit trail entries for the dm_finishworkflow or dm_abortworkflow events.
complete_type	integer	S	Whether the workflow was finished or aborted. Valid values are: 0, meaning the workflow was finished 1, meaning the workflow was aborted The value is taken from the audit trail entries for the dm_finishworkflow or dm_abortworkflow events.
component_id	ID	S	Object ID of the first component in the first package. The value is taken from audit trail entries for the dm_addpackage event.
component_name	string(80)	S	Name of the first component in the first package. The value is taken from audit trail entries for the dm_addpackage event.
creator_name	string(32)	S	Name of the workflow's creator. The value is taken from the audit trail entries for the dm_createworkflow event.

Property	Datatype	Single or repeating	Description
paused_duration	double	S	Length of time the workflow was in the paused state. The value is computed by subtracting the value of active_duration from total_duration.
process_id	ID	S	Object ID of the template used to create the workflow. The value is taken from the audit trail entries for the dm_createworkflow event.
process_name	string(255)	S	Name of the workflow template used to generate the workflow instance.
start_date_utc	Date	S	Date and time of the workflow's start. The value is taken from the audit trail entries for the dm_startworkflow event.
supervisor_name	string(32)	S	Name of the workflow's supervisor The value is taken from the audit trail entries for the dm_startworkflow and dm_changeworkflowsupervisor events.
total_duration	double	S	Total length of the time the workflow existed. The value is computed by subtracting start_date_utc from complete_date_utc.

Property	Datatype	Single or repeating	Description
total_user_cost	double	S	Sum of the user_cost values for the completed work items in the workflow. The value is computed from the dmc_completed_workitem objects.
total_user_time	integer	S	Sum of the user_time values for the completed work items in the workflow. The value is computed from the dmc_completed_workitem objects.
workflow_id	ID	S	Object ID of the completed workflow. The value is taken from dm_createworkflow events.
workflow_name	string(128)	S	Name of the workflow. The value is taken from the audit trail entries for the dm_createworfklow event.

Completed Workitem

Purpose

Records information about completed work items in workflows.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_completed_workitem
Object type tag: 00

A completed workitem object stores information from audit trail records about a completed work item. The objects are created by the dm_WfReporting job and used by Webtop's aggregrate workflow reporting tool. To fully populate the properties in these objects, you must be auditing all workflow events.

The object type is created by a script when Content Server is installed.

Properties

Table 39, page 124, lists the properties of the type.

Table 39. Properties defined for the completed workitem type

Property	Datatype	Single or repeating	Description
act_name	string(128)	S	Name of the activity that generated the work item The value is taken from the audit trail entries for dmd_startedworkitem events
act_seqno	integer	S	Sequence number of the activity that generated the work item The value is taken from the audit trail entries for dmd_startedworkitem events

Property	Datatype	Single or repeating	Description
active_duration	double	S	Length of time that the work item was in the active state The value is computed from the audit trail entries for the dm_selectedworkitem, dm_completedworkitem, and dm_changestateactivity events.
complete_action	integer	S	Identifies the action chosen by the user when completing the work item. Valid values are: -1, meaning unknown 0, meaning forward 1, meaning reject The value is derived from the audit trail entries for the dm_setoutput event..
complete_date_utc	Date	S	Date and time at which the work item was completed The value is taken from the audit trail entries for dm_completedworkitem events.
complete_state	integer	S	State of the work item when the workflow was finished or aborted. Valid values are: 0, meaning dormant 1, meaning acquired 2, meaning finished
creation_date_utc	Date	S	Date and time when the work item was created The value is taken from the audit trail entries for dm_startedworkitem events.

Property	Datatype	Single or repeating	Description
dormant_duration	double	S	Length of time, in seconds, that the work item was in the dormant state. The value is derived from the audit trail entries for dm_startedworkitem and dm_selectedworkitem events.
paused_duration	integer	S	Length of time, in seconds, that the work item was in the paused state The value is derived from the values for total_duration, dormant_duration, and active_duration.
performer_name	string(32)	S	Name of the task performer The value is taken from the audit trail entries for the dm_startedworkitem event.
process_id	ID	S	Object ID of the workflow template used to create the workflow that generated this work item. The value is taken from audit trail entries for the dm_startedworkitem event.
process_name	string(255)	S	Name of the workflow template used to generate the workflow instance that contained this work item. The value is taken from audit trail entries for the dm_startedworkitem event.

Property	Datatype	Single or repeating	Description
task_priority	integer	S	The final priority of the work item. The value is taken from the audit trail entries for the dm_startedworkitem and dm_changepriorityworkitem events.
total_duration	integer	S	Total length of time, in seconds, that the work item existed This is computed using start_date_utc and complete_date_utc values.
user_cost	double	S	The user cost argument value specified in the Complete method. The value is taken from audit trail entries for the dm_completedworkitem event.
user_time	integer	S	Amount of time the user spent on the work item (specified as an argument to the Complete method) The value is taken from audit trail entries for the dm_completedworkitem event.
workflow_id	ID	S	Object ID of the workflow that generated the work item The value is taken from audit trail entries for the dm_startedworkitem events.
workitem_id	ID	S	Object ID of the work item The value is taken from audit trail entries for the dm_startedworkitem events.

Property	Datatype	Single or repeating	Description
wq_doc_profile	string(64)	S	Value of the a_wq_doc_profile property of the work item. The value is taken from audit trail entries for the dm_completedworkitem events.
wq_flag	integer	S	Value of the a_wq_flag property of the work item The value is taken from audit trail entries for the dm_completedworkitem events.
wq_name	string(32)	S	Value of the a_wq_name property of the work item The value is taken from audit trail entries for the dm_completedworkitem events.

Component

Purpose

Represents a component, a set of related functionality that is used by client applications.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_component
Object type tag: 08

A component object represents a component, a set of related functionality that is used by client applications.

Properties

Table 40, page 129, lists the properties defined for the type.

Table 40. Properties defined for the component type

Property	Datatype	Single or repeating	Description
build_technology	integer	S	The build technology used to create the component. Valid values are: 1, for ACX 16, for WIN 32 Exes 32, for Data module
com_class_id	string(38)	S	The component's COM class ID. This is required for desktop applications. It is not required for Web-based applications.
component _version	string(16)	S	The component's version.
uniq_cont_ticket	string(128)	S	Used internally.

Composite Predicate

Purpose

Represents one route case condition for a workflow activity.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_composite_predicate
Object type tag: 00

A composite predicate object records a route case condition for an activity that has a transition type of automatic. Composite predicate objects are created internally when the addConditionRouteCase method is used to add route case definitions to an activity. You cannot create these objects directly.

Properties

Table 41, page 130, lists the properties defined for the type.

Table 41. Properties defined for the composite predicate type

Property	Datatype	Single or repeating	Description
predicate_id	ID	R	Object IDs of transition condition objects representing the individual comparison expressions that make up the route case.
r_aspect_name	string(64)	R	Names of the TBO associated with the predicate.

Cond Expr

Purpose

Stores a list of one or more Boolean expressions that are called by IF conditions in a routine or containing expression.

Description

Supertype: Func Expr
Subtypes: Cond ID Expr
Internal name: dm_cond_expr
Object type tag: 56

A cond expr object stores a list of one or more Boolean expressions that are called by IF conditions in a routine or containing expression. Content Server creates and manages cond expr objects. Users cannot create them.

Properties

Table 42, page 131, lists the property defined for the cond expr type.

Table 42. Properties defined for the cond expr type

Property	Datatype	Single or repeating	Description
expression_list	ID	R	List of object IDs representing dm_func_expr objects for expressions that return a Boolean result.

Cond ID Expr

Purpose

Stores a list of one or more Boolean expressions and a list of corresponding object IDs.

Description

Supertype: Cond Expr
Subtypes: None
Internal name: dm_cond_id_expr
Object type tag: 57

A cond id expr object stores a list of one or more Boolean expressions and a list of corresponding object IDs. (The list of expressions is stored in the inherited expression_list property.) The information stored in a cond ID expr object is part of the data dictionary. A cond id expr object is created when a routine or larger expression includes one or more if conditions that return an object ID. Content Server creates and manages cond id expr objects. Users cannot create them.

Properties

Table 43, page 132, lists the properties defined for the type.

Table 43. Properties defined for the cond ID expr type

Property	Datatype	Single or repeating	Description
default_id	ID	S	Object ID to return if none of the expressions evaluate to TRUE.
id_list	ID	R	Object IDs to be returned when the corresponding expression evaluates to TRUE. For example, if the expression specified in expression_list[3] evaluates to TRUE, the object ID specified in id_list[3] is returned.

Config Scope Relation

Purpose

Describes a configuration scope and relates a preset package object to its scope.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_config_scope_relation
Object type tag: 37

A config scope relation object is used to relate a preset package to the preset package's scope. The properties of the config scope relation object record information about the scope. The name of the relationship represented by objects of type config scope relation is dmc_config_scope_rel_type. This object type is installed with the Preset DAR file during repository configuration.

Properties

Table 44, page 133, lists the properties defined for the type.

Table 44. Properties defined for the config scope relation type

Property	Datatype	Single or repeating	Description
scope_name	string(32)	S	Name of the scope This value is defined by the application.
scope_values	string(64)	R	Specifies the values to which the related preset configuration applies

Constraint Set

Purpose

Records a collection of constraints applied to relationships.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_constraint_set
Object type tag: 08

A constraint set object records one or more predicates that define constraints applied to the parent or child in a relationship defined by a dmc_relationship_def object. Constraint set objects are created and managed internally. Users should not directly manipulate these objects. The object type is installed with the Smart Container DAR file during repository configuration.

Properties

Table 45, page 134, lists the properties defined for the type.

Table 45. Properties defined for the constraint set object type

Property	Datatype	Single or repeating	Description
constraint_name	string(255)	R	Name of the constraint
constraint_metadata	string(255)	R	Metadata with which to configure the module identified in module_name prior to invoking the module.
module_name	string(255)	R	Name of the BOF module that implements the constraint

Cont Transfer Config

Purpose

Controls the capabilities of the ACS servers that service the repository and whether content in the repository may be pre-cached on a BOCS server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_cont_transfer_config
Object type tag: 08

A cont transfer config object defines a top-level of control for the read and write capabilities of the ACS servers associated with the repository. In addition, it controls whether content in the repository can be pre-cached on a BOCS server.

A cont transfer config object is created during repository installation. Although a Superuser is allowed to create an object of this type, there should be only one cont transfer config object in a repository.

Properties

Table 46, page 135, lists the properties defined for the type.

Table 46. Properties defined for the cont transfer config type

Property	Datatype	Single or repeating	Description
acs_read_mode	integer	S	Whether ACS servers associated with the repository can transfer content to users for reading. Valid values are: 0, meaning disabled 1, meaning enabled The default is 1. The value in this property overrides any capability defined for individual ACS servers.

Property	Datatype	Single or repeating	Description
acs_write_mode	integer	S	Whether ACS servers associated with the repository can transfer content from users to the repository. Valid values are: 0, meaning disabled 1, meaning enabled 3, meaning enabled and asynchronous write enabled also 7, meaning enabled and asynchronous write with retention enabled also The default is 0. The value in this property is disregarded for individual ACS servers that are disabled.
bocs_precaching_mode	integer	S	Indicates whether content in the repository may be pre-cached on a BOCS server or not. Valid values are: 0, pre-caching content is not allowed 1, pre-caching content is allowed The default is 1.
bocs_encryption_mode	string(32)	S	This attribute is the default value for the encryption_mode parameter of the BOCS content URL. If you use a default type-based object (TBO) and the content is in an encrypted store, *require* is used regardless of the value of this attribute. If you write your own TBO, the content will be encrypted

Property	Datatype	Single or repeating	Description
			on BOCS based on what the doGetContentEncryptionMode method returns, regardless of the value of this attribute. Valid values are: *allow* – encrypt content on BOCS only if the encryption.mode parameter of the acs.properties file on BOCS is set to *require*. *require* – encrypt content on BOCS. Content will not be stored on BOCS if the BOCS version does not support encryption or if the encryption.mode parameter of the acs.properties file on BOCS is set to *disable* *disable* – do not encrypt content on BOCS The default value is *allow*.

Containment

Purpose

Stores information about a component of a virtual document.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmr_containment
Object type tag: 05

A containment object stores information about a component of a virtual document. Each time a user adds a component to a virtual document, the server creates a containment object for that component. The properties for this type are set by the Appendpart, Insertpart, and Updatepart methods. Users can query these properties using DQL.

Properties

Table 47, page 138, lists the properties defined for the type.

Table 47. Properties defined for the containment type

Property	Datatype	Single or repeating	Description
a_contain_desc	string(255)	S	User-defined. Used by Documentum clients to manage XML documents. For more information, refer to *Content Server Fundamentals*.
a_contain_type	string(255)	S	User-defined. Used by Documentum clients to manage XML documents. For more information, refer to *Content Server Fundamentals*.

Property	Datatype	Single or repeating	Description
component_id	ID	S	Chronicle ID of the component. (The chronicle ID is the object ID of the original version of an object. If the object has no versions, then its object ID and chronicle ID are the same.)
copy_child	integer	S	Defines client behavior when the document containing the component is copied. Valid values are: 0, meaning the decision whether to copy or reference the component is left to the user when the document is copied. 1, meaning when the document is copied, the component is referenced in the new copy rather than actually copied. 2, meaning when the document is copied, the component is also copied.
follow_assembly	Boolean	S	If set to TRUE, directs the system to resolve a component using the component's assembly (if the component has an assembly).
i_partition	integer	S	Identifies the partition where the object resides.
order_no	double	S	Number representing the component's position within the components of the virtual document identified by parent_id.
parent_id	ID	S	Object ID of the object that directly contains the component.

Property	Datatype	Single or repeating	Description
use_node_ver _label	Boolean	S	If set to TRUE for early-bound components, the server uses the early-bound symbolic label to resolve late-bound descendants of the component during assembly.
version_label	string(32)	S	Version label for the component.

Content

Purpose

Stores information about a content file.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmr_content
Object type tag: 06

A content object stores the information about the format and location of a content file. It also contains the information that links the content to an object. A content object also has five properties used by Content Transformation Services to store metadata values generated by a Content Transformation Services server. If the object is stored in a content-addressed storage area, these properties are used to record metadata values to be stored in the storage system with the content.

Properties

Table 48, page 141, lists the properties defined for the type.

Table 48. Properties defined for the content type

Property	Datatype	Single or repeating	Description
content_attr _data_type	integer	R	Data type of the media property in the corresponding index position in content_attr_name. Possible values are: 2, for a string datatype 4, for a date/time datatype 5, for a double datatype

Property	Datatype	Single or repeating	Description
content_attr _date_value	date	R	The value of the metadata field identified in the corresponding index position in content_attr_name if the property is a date data type. For all other data types, the value in the corresponding index position in content_attr_date_value is NULLDATE.
content_attr _name	string(64)	R	Names of the metadata fields generated by a Content Transformation Services server or specified in a Setcontentattrs or SET_CONTENT_ATTRS method for the content.
content_attr _num_value	integer	R	The value of the metadata field identified in the corresponding index position in content_attr_name if the property is an integer data type. For all other data types, the value in the corresponding index position in content_attr_num_value is NULLINT.
content_attr value	string(255)	R	The value of the metadata field identified in the corresponding index position in content_attr_name if the property is an string data type. For all other data types, the value in the corresponding index position in content_attr_value is NULLSTRING.

Property	Datatype	Single or repeating	Description
content_size	integer	S	Size, in bytes, of the content file This property cannot record a size greater than 2GB. If the file is large, examine the full_content_size property to obtain the full size. The property is not set if the content is stored in external storage.
data_ticket	integer	S	Value used internally to retrieve the content. An actual data ticket is created for content stored in file stores and blob stores. If the content is stored in turbo storage, in the content object, data_ticket is 0. If the content is stored in turbo storage and is too large for the i_contents property in the content object, the content is stored in repeating values of the i_contents property of a subcontent object. The data_ticket value in the content object contains the number of values in that properties used to store the content. If the content is stored in external storage, data_ticket is 0. If the content is stored in a distributed storage area component, the data ticket value identifies a dmi_replica_record object for the content. If the content is stored in content-addressed storage, the data_ticket is a number used

Property	Datatype	Single or repeating	Description
			internally only and has no external meaning.
encoding	string(10)	S	The encoding for the format, if any
format	ID	S	Object ID of the format object describing this content's format
full_content _size	double	S	Size, in bytes, of the content file The property is not set if the content is stored in external storage.
full_format	string(64)	S	Full format specification for the content
fulltext_index	ID	R	obsolete
i_parked_state	integer	S	Identifies the inbound transfer status of the content. Valid values are: 0, meaning the content is in the storage area 1, meaning the content is temporarily residing on a BOCS server 2, meaning the content is temporarily residing on an ACS server 3, meaning that the content is not available because this content object was replicated from a repository whose content is not yet available. That is, the source content has a non-zero value in i_parked_state.
i_partition	integer	S	Identifies the partition where the object resides.
index_format	ID	S	obsolete
index_formats	ID	R	obsolete

Property	Datatype	Single or repeating	Description
index _operations	integer	R	obsolete
index_pages	integer	R	obsolete
index_parent	ID	S	obsolete
index_parents	ID	R	obsolete
index_set_times	DATE	R	obsolete
index_subtypes	string(27)	R	obsolete
is_archived	Boolean	S	Indicates whether the content has been archived.
is_offline	Boolean	S	Indicates whether the content is in the storage area.
i_contents	For all databases except Sybase: string(2000) For Sybase: string(255)	S	If the content is stored in turbo storage, this property contains the actual content file. If the content is too large for this property, the content is stored in a dmi_subcontent object and this property is unused. If the content is stored in content-addressed storage, this property contains the content address. If the content is stored in external storage, this property contains the token used to retrieve the content. The property cannot be selected using the API or DQL.
i_encoding	string(10)	R	Contains format information used internally in the management of distributed repositories.
i_format	ID	R	Contains the information from the page and format properties

Property	Datatype	Single or repeating	Description
i_full_format	string(64)	R	Contains format information used internally in the management of distributed repositories.
i_index_format	ID	R	obsolete
i_px	integer	R	Contains format information used internally in the management of distributed repositories.
i_py	integer	R	Contains format information used internally in the management of distributed repositories.
i_pz	integer	R	Contains format information used internally in the management of distributed repositories.
i_rendition	integer	R	Contains the information in the rendition property. Used internally in the management of distributed repositories.
loss	integer	S	The transformation loss for the format, if any.
other_file_size	double	S	Reserved for future use
other_ticket	integer	S	Value used internally to retrieve the content
page	integer	R	Position of the content in each of the objects that contain it
page_modifier	string(16)	R	User-defined string to disambiguate renditions having the same format that are associated with a particular content page of a document.
parent_count	integer	S	Total number of objects that contain this content
parent_id	ID	R	Object IDs of the objects that contain the content represented by this content object

Property	Datatype	Single or repeating	Description
r_content_hash	string(256)	S	Hashed value of the associated content file. This is only set if content is stored in a file store storage area and the storage area's content_hash_mode property is set to 1.
rendition	integer	S	Provides information about a rendition. Valid values are: 0, for original content 1, for a rendition generated by the server 2, for a rendition generated by the client 3, meaning keep the rendition when the content with which it is associated is updated or removed from the document or repository
resolution	integer	S	The resolution specification for the content's format.
set_client	string(64)	S	Name of the client machine on which the setfile was executed.
set_file	string(255)	S	Source file on the client machine that contained the content.
set_time	DATE	S	The initial value is the time at which the Setfile was executed. The time value is the time on the server machine.
storage_id	ID	S	Object ID of the store object representing the storage area that contains the content.
transform_path	string(32)	S	The transformation path for this format
update_count	integer	R	Number of the update operation in which this content will be indexed.
x_range	integer	S	The format's range along the x axis.

Property	Datatype	Single or repeating	Description
y_range	integer	S	The format's range along the y axis.
z_range	integer	S	The format's range along the z axis.

Cryptographic Key

Purpose

Stores a private cryptographic key.

Description

Supertype: SysObject
Subtypes: None
Internal Name: dm_cryptographic_key
Object type tag: 08

A cryptographic key object stores a private cryptographic key. The cryptographic key object is created automatically by Content Server. It is used by Content Server to encrypt instructions regarding content file availability for an ACS server or BOCS server. There is only one cryptographic key object in a repository. The permissions on this object give Delete permission to the owner and the members of the dm_superusers group.



Caution: These objects are for internal use only. Do not modify, remove, or add these objects.

Properties

Table 49, page 149, lists the properties defined for the type.

Table 49. Properties defined for the cryptographic key type

Property	Datatype	Single or repeating	Description
key_identifier	string(40)	S	Base 64-encoded SHA1 digest of the DER private key
key_type	integer	S	Identifies what the key used for by Content Server. The only valid value is: 1, meaning used for ACS encryption

Property	Datatype	Single or repeating	Description
key_value	string(1000)	S	Base 64-encoded and DBK-encrypted DER key
public_key_identifier	string(40)	S	Base 64-encoded SHA1 digest of the DER public key value in the dm_public_key_certificate.key_identifier property.

DD Attr Info

Purpose

Contains the published data dictionary information for a property.

Description

Supertype: DD Common Info
Subtypes: None
Internal Name: dmi_dd_attr_info
Object type tag: 6a

A dd attr info object contains the published data dictionary information for a property. You cannot create dd attr info objects directly. They are created or modified as needed when data dictionary information is published.

With one exception (attr_name), each property defined for the type has a corresponding property in the type named i_*property_name*. The i_ properties are used internally by Content Server.

Properties

Table 50, page 151, lists the basic properties of a dd attr info object.

Table 50. Properties defined for the DD attr info type

Property	Datatype	Single or repeating	Description
allowed_search _ops	integer	R	The search operators available for the property. Valid values depend on the property's data type. For string: 1, meaning = 2, meaning <> 3, meaing > 4, meaning < 5, meaning >= 6, meaning <= 7, meaning begins with 8, meaning contains 9, meaning does not contain 10, meaning ends with

Property	Datatype	Single or repeating	Description
			11, meaning in 12, meaning not in 14, meaning is null 15, meaning is not null
allowed_search _ops			For Integer:
(continued)			1, meaning = 2, meaning <> 3, meaning > 4, meaning < 5, meaning >= 6, meaning <= 11, meaning in 12, meaning not in 13, meaning between 14, meaning is null 15, meaning is not null For Date: 1, meaning = 2, meaning <> 3, meaning > 4, meaning < 5, meaning >= 6, meaning <= 13, meaning between 14, meaning is null 15, meaning is not null For Boolean and ID: 1, meaning = 2, meaning <> 14, meaning is null 15, meaning is not null
attr_name	string(32)	S	Name of the property described by the dd attr info object. This property has no corresponding i_ property in the type.

Property	Datatype	Single or repeating	Description
category_name	string(64)	S	User-defined.
computed_dep _usr	Boolean	S	Indicates whether the values in computed_expr_dep were set by the user or calculated. TRUE means the values are user defined; FALSE means they are calculated.
computed_expr _dep	string(32)	R	List of properties on which the computed expression referenced in cond_computed_expr depends.
cond_computed _expr	ID	R	Object ID of the dm_cond_id_expr object that contains the conditional computed expressions and the associated object IDs.
cond_value_assist	ID	S	Object ID of the cond ID expr object that contains the conditional expressions for the property's value assistance.
def_value_length	integer	S	Length of the property's default value. A zero indicates that the value is unspecified.
default_expr _builtin	integer	R	Used internally.
default_expr_kind	integer	R	Used internally.
default_expr _value	string(255)	R	Used internally to optimize default value handling.
default_search_arg	integer	S	Default value to display in conjunction with the default search operator.
default_search_op	string(255)	S	Default search operator to display to users when they search on the property.

Property	Datatype	Single or repeating	Description
default_value	ID	R	Object ID of an expression object that resolves to the default value for the property. For single-valued properties, only one value can be defined. For repeating properties, multiple values can be defined.
domain_length	integer	S	For string properties, the maximum length of a value.
domain_type	integer	S	Identifies the datatype of the property. Valid values are: 0, Boolean 1, Integer 2, String 3, ID 4, Time/Date 5, Double
format_pattern	string(64)	S	For date properties, the pattern used to interpret values for the property.
format_pattern _tag	integer	S	Currently unused. (This value is always 1.)
ignore_immutable	Boolean	S	Indicates whether the property is changeable even if the containing object is immutable. This setting is effective only for objects of type dm_sysobject or its subtypes.
is_hidden	Boolean	S	For use by client applications.
is_required	Boolean	S	Indicates whether users must provide a value for the property.
map_data_string	string(128)	R	List of possible data values for the property.
map_description	string(255)	R	Descriptions of the data values at the corresponding index levels in map_data_string.

Property	Datatype	Single or repeating	Description
map_display _string	string(128)	R	The character string to display for the data value at the corresponding index level in map_data_string.
not_null	Boolean	S	Indicates whether the property has a NOT NULL constraint defined for it.
not_null_enf	integer	S	Indicates who is responsible for enforcing a NOT NULL constraint. Valid values are the same as for foreign_key_enfs.
not_null_msg	string(255)	S	Error message to display when the NOT NULL constraint is violated.
read_only	Boolean	S	Indicates whether the property is read only.
reference_kind	integer	S	Used internally by Collaboration Services
super_domain_id	ID	S	Object ID of the domain from which the domain of this property is derived, if any.
super_domain _name	string(32)	S	Name of the domain from which the domain of this property is derived. If blank, the domain is the built-in domain indicated by domain_type.
value_assist_dep	string(32)	R	Properties on which the value assistance expressions depend.
value_assist_dep _usr	Boolean	S	Indicates whether the user defined the list of values in value_assist_dep. TRUE means the user defined the values; FALSE means the values were calculated.

DD Common Info

Purpose

Contains the published data dictionary information that an object type and property have in common.

Description

Supertype: Persistent Object
Object Subtypes: DD Attr Info; DD Type Info
Internal Name: dmi_dd_common_info
Object type tag: 68

A dd common info object contains the published data dictionary information that an object type and property have in common. You cannot create a dd common info object directly. They are created as needed when data dictionary information is published.

With four exceptions, each property listed in Table 51, page 156 has a corresponding property named i_*property_name* in the object type definition. The i_ properties are used internally by Documentum products. The four exceptions are the properties that identify the object type or property with which the dd common info object is associated. These four properties are:

- business_policy_id
- nls_key
- state_name
- type_name

Properties

Table 51, page 156, lists the properties defined for the object type.

Table 51. Properties defined for the DD common info type

Property	Datatype	Single or repeating	Description
business_policy_id	ID	S	Object ID of a dm_policy object This property has a value only if the information in the dd common info object is associated with a particular lifecycle and state.

Property	Datatype	Single or repeating	Description
			This property has no corresponding i_ property in the type.
comment_text	string(255)	S	User-defined.
constraint_dep _usr	Boolean	S	TRUE indicates that the values in val_constraint_dep are user-defined. FALSE means that the values are computed by Content Server.
foreign_keys	ID	R	Object IDs of any foreign key objects associated with the property.
foreign_key_enfs	integer	R	Indicates who is responsible for enforcing the corresponding foreign key. Valid values are: 1, Enforcement disabled 2, Enforcement by application
foreign_key_msgs	string(255)	R	Error message to display when the corresponding foreign key constraint is violated.
help_text	string(255)	S	Help text to use for the property.
i_dd_flags	integer	R	Reserved for future use.
ignore_constraints	Boolean	S	TRUE indicates that constraints inherited from parent types are ignored. FALSE means the inherited constraints are enforced. The default is FALSE.
is_searchable	Boolean	S	Indicates whether the property is searchable. The default is TRUE, meaning the property can be searched.

Property	Datatype	Single or repeating	Description
label_text	string(64)	S	Label for the property or object type. Documentum provides default labels for all properties and object types. For numerous properties, the default is defined in a data dictionary population file. If no default is defined in the data dictionary, the default for an object type is its name. For example, for dm_sysobject, the default is dm_sysobject. The default for a property is the property's name.
life_cycle	integer	S	Indicates the current life cycle status of the type or property. Valid values are: 1, Currently in use 2, For future use 3, Obsolete The default is 1 (currently in use).
nls_key	string(5)	S	Locale of the information in the dd common info object. This property has no corresponding i_ property in the type.
primary_key	ID	S	Object ID of dm_key object that contains primary key constraints for the type or property.
primary_key_enf	integer	S	Identifies who is responsible for enforcing the corresponding primary key. Valid values are: 1, Enforcement disabled 2, Enforcement by application

Property	Datatype	Single or repeating	Description
primary_key_msg	string(255)	S	Error message to display when the primary key is violated.
resync_needed	Boolean	S	Indicates whether there are unpublished changes to the object or property. TRUE indicates that there are unpublished changes.
state_name	string(32)	S	Name of the lifecycle state in which this data dictionary information applies. This property has a value only if the information in the dd common info object is associated with a particular lifecycle and state. This property has no corresponding i_ property in the type.
type_name	string(32)	S	Name of the object type with which the information in the dd common info object is associated. This property has no corresponding i_ property in the type.
unique_keys	ID	R	Object IDs of the key objects representing any unique keys that apply to the type or property. The unique keys defined for the type or property are listed first, followed by those inherited from the supertype or super domain.

Property	Datatype	Single or repeating	Description
unique_key_enfs	integer	R	Indicates who is responsible for enforcing the corresponding unique key. Valid values are the same as for foreign_key_enfs.
unique_key_msgs	string(255)	R	Error message to display when the corresponding unique key is violated.
val_constraint	ID	R	Object IDs of the expression objects representing any check constraints that apply to the property.
val_constraint _dep	string(32)	R	Names of the properties referenced by the constraint at the corresponding index level in val_constraint.
val_constraint _enf	integer	R	Indicates who is responsible for enforcing the corresponding validation constraint. Valid values are the same as for foreign_key_enfs.
val_constraint _msg	string(255)	R	Error message to display when the corresponding validation constraint is violated

DD Info

Purpose

Contains data dictionary information for an object type or property that is not dependent on locale.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_dd_info
Object type tag: 4e

A dd info object contains data dictionary information for an object type or that is not dependent on locale. Content Server creates and manages dd info objects. Properties in dd info objects are set when users add or change data dictionary information.

Properties

Table 52, page 161, lists the properties defined for dd info that are applicable to both object types and properties. Table 53, page 163, lists the properties that apply only to object types. Table 54, page 164, lists the dd info properties that apply only to properties.

Table 52. Properties applicable to object types and properties

Property	Datatype	Single or repeating	Description
auditable _appevents	string(64)	R	The application-defined events that can be audited.
auditable _sysevents	string(64)	R	The system-defined events that can be audited.
foreign_keys	ID	R	Object IDs of the foreign key objects for the type or property.
foreign_key_enfs	integer	R	Indicates how the corresponding foreign keys are enforced. Valid values are: 1, Enforcement disabled 2, Enforcement by application

Property	Datatype	Single or repeating	Description
fulltext_support	integer	S	Indicates how full-text indexing is supported for the object type. Valid values are: 0, no support 1, light support 2, full support **Note:** Currently, this is only used by lightweight object subtypes.
ignore_constraints	integer	S	Indicates whether the validation, key, and NOT NULL constraints defined for the type or are ignored. Valid values are: 0, Follow the constraints 1, Ignore the constraints -1, Inherit the value from the corresponding property in the type or type's supertype
is_searchable	integer	S	For client use. Indicates whether the or type should appear in pick lists for Find dialogs. Valid values are: 0, FALSE 1, TRUE -1, Inherit the value from the corresponding property in the type or type's supertype
life_cycle	integer	S	Indicates the state of the object type or in the repository. Valid values are: 1, Currently in use 2, For future use 3, Obsolete
unique_keys	ID	R	Objects IDs of the key objects that define unique keys for this type or property.

Property	Datatype	Single or repeating	Description
unique_key_enfs	integer	R	Indicates how the corresponding unique keys are enforced. Valid values are the same as for foreign_key_enfs.
val_constraint	ID	R	Object IDs of the expression objects corresponding to the constraints applied to the type or property.
val_constraint_dep	string(32)	R	The properties on which the expression validation constraints depend.
val_constraint _enf	integer	R	Indicates how the corresponding validation constraints are enforced. Valid values are the same as for foreign_key_enfs.

Table 53. Properties applicable only to object types

Property	Datatype	Single or repeating	Description
comp_classifier	string(128)	R	Qualified components that can be called for this type.
default_policy_id	ID	S	Chronicle ID of the default policy object for the type.
icon_index	integer	S	The index that locates the type's icon in the icon resource file.
policy_ver_label	string(32)	S	Version label that identifies which version of the default policy object to use for the type.
primary_key	ID	S	Object ID of the primary key for the type.The primary key must be one of the unique keys defined in unique_key.
primary_key_enf	integer	S	Indicates how the primary key is enforced. Valid values are the same as for foreign_key_enfs.
qual_comp_id	ID	R	Object IDs corresponding to the qualified components identified in comp_classifier.

Table 54. DD info properties applicable to properties

Property	Datatype	Single or repeating	Description
allowed_search _ops	integer	R	A list of integers representing the valid search operators for this property. Valid values are: 1, for = 2, for <> 3, for > 4, for < 5, for >= 6, for <= 7, for begins with 8, for contains 9, for does not contain 10, for ends with 11, for in 12, for not in 13, for between 14, for is null 15, for is not null 16, for not
computed_expr _dep	string(32)	R	Currently unused.
cond_computed _expr	ID	S	Currently unused.
cond_value_assist	ID	S	Object ID of the cond id expr object that contains the conditional value assistance.
default_search _arg	string(255)	S	The default value to use in conjunction with the default search operator (defined in default_search_op)
default_search_op	integer	S	The default search operator for the property. The operator specified must be in allowed_search_ops.

Property	Datatype	Single or repeating	Description
default_value	ID	R	Object IDs of the expression objects representing the property's default values. For single-valued properties, default_value[0] must be the object ID of a dm_expression object corresponding to a literal of the appropriate datatype for the property or the NULL value appropriate for the type (for example, NULLSTRING). For repeating properties, multiple expression object IDs are allowed, representing multiple default values for the property. However, object IDs representing expressions that evaluate to NULL are not allowed.
ignore_immutable	integer	S	Indicates whether the value in r_immutable_flag controls the property's changeability. This setting affects only properties of objects of type dm_sysobject or SysObject subtypes. If set to 1 (TRUE), the is changeable regardless of the object's r_immutable_flag setting. Valid values are: 0, FALSE 1, TRUE -1, Inherit value from super domain or type
is_hidden	integer	S	Provided for use by client applications. Valid values are the same as for ignore_immutable.

Property	Datatype	Single or repeating	Description
is_required	integer	S	Indicates whether a value is required for the property. If set to 1 (TRUE), the must have a value before saving the object. Valid values are the same as for ignore_immutable.
not_null	integer	S	Indicates whether the has the NOT NULL constraint defined for it. If set to 1 (TRUE), the NOT NULL constraint is specified for the property. Valid values are the same as for ignore_immutable.
not_null_enf	integer	S	Indicates how the NOT NULL constraint is enforced. Valid values are the same as for the foreign_key_enfs .
parent_id	ID	S	Object ID of the aggr domain object that references the dd info object.
read_only	integer	S	Indicates whether users can read and write the property or only read it. If set to 1 (TRUE), users can only read this property, they cannot write to it. Valid values are the same as for ignore_immutable.
reference_kind	integer	S	Used internally by Collaboration Services
value_assist_dep	string(32)	R	Properties on which the value assistance depends.

DD Type Info

Purpose

Contains the published data dictionary information for an object type.

Description

Supertype: DD Common Info
Subtypes: None
Internal Name: dmi_dd_type_info
Object type tag: 69

A dd type info object contains the published data dictionary information for an object type. You cannot create or modify dd type info objects directly. They are created or modified as needed when data dictionary information is published.

Properties

Table 55, page 167, lists the properties of the object type.

Table 55. Properties defined for the DD type info type

Property	Datatype	Single or repeating	Description
attr_domain_name	string(32)	R	Currently unused.
attr_domain_id	ID	R	Currently unused.
auditable _appevents	string(64)	R	The application-defined events that can be audited.
auditable _sysevents	string(64)	R	The system-defined events that can be audited.
comp_classifier	string(128)	R	The classifiers for the qualified components that can be executed against instances of the type.
default_policy_id	ID	S	Object ID of the dm_policy object representing the default lifecycle for the type.
i_attr_domain _id	integer	S	For internal use.
i_attr_domain _name	integer	S	For internal use.

Property	Datatype	Single or repeating	Description
i_comp_classifier	integer	S	For internal use.
i_default_policy _id	Boolean	S	For internal use.
i_icon_index	Boolean	S	For internal use.
i_policy_version _label	Boolean	S	For internal use.
i_qual_comp_id	integer	S	For internal use.
icon_index	integer	S	The index that locates the icon for the type in the icon resource file.
map_data_string	string(128)	R	List of possible data values for the .
map_description	string(255)	R	Descriptions of the data values at the corresponding index levels in map_data_string.
map_display _string	string(128)	R	The character string to display for the data value at the corresponding index level in map_data_string
policy_ver_label	string(32)	S	Version label of the object identified in default_policy_id. **Note:** This value is provided by the user, not derived from the object by Content Server
qual_comp_id	ID	R	Object IDs of the qual comp objects representing the component routines that can be executed against instances of the type.
r_has_check	Boolean	R	For internal use
r_has_constraint	Boolean	R	For internal use
r_has_default	Boolean	R	For internal use
r_has_dependency	Boolean	R	For internal use
r_has_foreign_key	Boolean	R	For internal use
r_has_ignore _immutable	Boolean	R	For internal use
r_has_not_null	Boolean	R	For internal use

Property	Datatype	Single or repeating	Description
r_has_primary _key	Boolean	R	For internal use
r_has_unique_key	Boolean	R	For internal use
r_has_value_assist	Boolean	R	For internal use
scope_config	ID	R	Object IDs of the scope config object for the object type.

Display Config

Purpose

Defines the display configuration for a group of properties.

Description

Supertype: None
Subtypes: None
Internal name: dm_display_config
Object type tag: 6b

A display config object defines the display configuration for a group of properties.Display config objects are referenced by scope config objects, which define the context within which the display configuration is used. Both display config objects and scope config objects are used by client applications. Content Server does not use these objects.

Properties

Table 56, page 170, lists the properties defined for the type.

Table 56. Properties defined for the display config type

Property	Datatype	Singe/ repeating	Description
attribute_display_ hint	integer	R	Controls the display of properties in the user interface. These values are interpreted by client application, not Content Server. Values used by Desktop Client and Webtop are: • **0**, meaning do not display a separator before a property • **1**, meaning display a separator before a property Values used by Webtop only are: • **2**, meaning display the property if the user requests it and do not use a separator • **3**, meaning display the property if the user requests it and use a separator

Property	Datatype	Singe/ repeating	Description
			The hint at a particular index position is applied to the property named at the corresponding index position in attribute_name.
attribute_name	string(40)	R	List of properties names. The properties must be defined for the object type identified in property source. The order in which the properties are listed determines the order in which they are displayed in the client application.
attribute_source	string(27)	S	The object type for which the properties listed in attribute_name are defined.
fixed_display	Boolean	S	Indicates whether properties can be added to attribute_name or display_hints changed. T means that the property list and hint cannot be changed. F means that the property list and hint can be changed. The default is F.
i_config_identifier	string(20)	S	Uniquely identifies a display config object across repositories. The format of the value is dm_*r_object_id*, where r_object_id is the object ID of the display config object. This is used internally to manage load operations.
object_name	string(64)	S	Name of the display config object. This must be unique within the scope in which the display config object is used. **Note:** Uniqueness is enforced by the client, not Content Server.

Distributed Store

Purpose

Contains information about a storage area.

Description

Supertype: Store
Subtypes: None
Internal name: dm_distributedstore
Object type tag: 2c

A distributed store object contains information about a storage area. A distributed storage area points to component storage areas and is used to implement a repository that has distributed content. All but three of its properties are inherited from its supertype, dm_store.

Note: For information about distributed storage areas, refer to the *EMC Documentum Distributed Configuration Guide*.

Properties

Table 57, page 172, lists the properties defined for the distributed store type.

Table 57. Properties defined for the distributed store type

Property	Datatype	Single or repeating	Description
change_record_id	ID	S	Used internally to manage structural changes.
epoch_number	integer	S	Records a count of how many structural changes (adding or removing component areas) have been made to the distributed storage area.
only_fetch_close	Boolean	S	Indicates if a server only directly fetches files from component storage areas in the distributed storage area that are defined as close to the server. Typically set to false for distributed stores that use shared content. Typically set

Property	Datatype	Single or repeating	Description
			to true for distributed stores that replicate content. When set to true, the surrogate get method replicates the content to the local store

DMS Config

Purpose

Records configuration information about a DMS server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_dms_config
Object type tag: 08

A dms config object records the configuration of a DMS server. A dms config object is stored in the global registry repository. There can be only one dms config object in each global registry in the installation. Use Documentum Administrator to create and manage a dms config object.

Properties

Table 58, page 174, lists the properties defined for the type.

Table 58. Properties defined for the dms config type

Property	Datatype	Single or repeating	Description
dms_capability	integer	S	Defines the capabilities of the DMS server. Valid values are: 0, meaning disabled 1, meaning enabled The default is 1.
message_consume_url	string(240)	S	The URL used by a BOCS to retrieve messages from a DMS server.
message_post_url	string(240)	S	The URL used by DFC to post messages to the DMS.

Property	Datatype	Single or repeating	Description
server_major_version	string(16)	S	Indicates the major version of the DMS server. For a DMS 6.0 server, this must be set to 1.
server_major_version	string(16)	S	Indicates the minor version level of the DMS server. For a DMS 6.0 server, this must be set to 0.

Docbase Config

Purpose

Contains configuration information about a repository.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_docbase_config
Object type tag: 3c

A docbase config object contains configuration information about a repository Each repository must have a single docbase config object whose object name matches the name of the repository.

Properties

Table 59, page 176, lists the properties defined for the type.

Table 59. Properties defined for the docbase config type

Property	Datatype	Single or repeating	Description
a_bpaction_run_as	string(32)	S	Defines which user account is used to run lifecyle (business policy) actions. Valid values are: • session_user (default) • superuser • lifecycle_owner • *user name* The first three values are keywords that must be entered as shown. For the last, *user name*, specify the user name of a repository user.
approved_clients_only	Boolean	S	Indicates whether clients connecting to this repository must have an entry in the Client Rights database.

Property	Datatype	Single or repeating	Description
audit_old_values	Boolean	S	Indicates whether to include old as well as new values of audited properties in the audit trail entry. TRUE means to include the old values. FALSE means to exclude them. The default is TRUE. For repositories created on DB2 or Sybase platforms with a page size of 4K, this property value is always FALSE.
auth_deactivation_ interval	integer	S	Length of time between a user's account deactivation and automatic re-activation. If this is 0, the account is not automatically re-activated. The value is specified in minutes. The default is 0.
auth_failure_interval	integer	S	Length of time, in minutes, in which consecutive failed login authorizations will cause a user's account to be deactivated. The number of failed attempts that must occur within the interval to trigger deactivation is determined by the max_auth_attempts property. The default is 0, meaning that deactivation always occurs when the maximum number of consecutive failed login attempts is reached, regardless of how long that takes.

Property	Datatype	Single or repeating	Description
auth_protocol	string(32)	S	On Windows platforms, if set to domain_required, it indicates that the repository is running in domain-required mode. If the repository is not using domain-required mode, this is blank. On UNIX platforms, if you are authenticating users against a Windows domain, set this to unix_domain_used Otherwise, this property is blank for repositories running on a UNIX platform.
check_client_version	Boolean	S	T means that the repository servers will not accept connections from clients older than the version level specified in the oldest_client_version property. F means that the servers accept connections from any client version. The default is F.
client_pcaching_change	integer	S	Controls persistent client cache flushing. Incrementing this value forces clients to flush all persisent caches on start up.
client_pcaching_disabled	Boolean	S	T (TRUE) disables persistent client caching for sessions with the repository. The default is F (caching is allowed).

Property	Datatype	Single or repeating	Description
dd_locales	string(5)	R	Data dictionary locales recognized by the server. Setting this property requires you to execute a Reinit method on the server to make the change visible.
default_app _permit	integer	S	Default user permission level for application-controlled objects accessed through an application that doesn't own the object. Values are: 2, Browse permission 3, Read permission 4, Relate permission 5, Version permission 6, Write permission 7, Delete permission The default value is 3, Read permission.
dir_user_sync_on_ demand	Boolean	S	Controls whether LDAP directory users can be synchronized on demand in the repository. T means that such users can be synchronized on demand. F means that they cannot be synchronized on demand. The default value is F.
docbase_roles	string(32)	R	Used internally
effective_date	date	S	Obsolete in 5.2 repositories. In pre-5.2 repositories, this is used to force persistent client cache flushes. The default value is NULLDATE.

Property	Datatype	Single or repeating	Description
folder_security	Boolean	S	Indicates whether the repository is running with folder security on or off. The default is T, meaning that security is turned on.
fulltext_install_locs	string(32)	R	Name of the location object that points to a fulltext installation.
i_crypto_key	string(255)	S	The encryption key for the repository
i_ticket_crypto_key	string(255)	S	The encrypted login ticket key
index_store	string(80)	S	Name of the RDBMS tablespace where you want to store type indexes.
login_ticket_cutoff	Date	S	Defines the earliest possible creation date for valid login tickets. Tickets issued before this date are not valid in this repository. The default value is NULLDATE, meaning there is no cutoff date for login tickets.
mac_access_protocol	string(32)	S	Identifies the type of file sharing protocols in use for Macintosh clients. Valid values are nt, double, ushare, or none. This is set to nt when a repository is configured to enable “other file” use. Use none only if you have no Macintosh clients. Double and ushare are meaningful only to Macintosh clients.

Property	Datatype	Single or repeating	Description
macl_security_ disabled	Boolean	S	If T (TRUE), Content Server enforces only the AccessPermit and ExtendedPermit entries in an ACL. If the server is installed with a Trusted Content Services license, the default is F (FALSE), meaning the server enforces all entries an ACL If the server is not installed with a Trusted Content Services license, the default is T.
max_auth_attempt	integer	S	Maximum number of unsuccessful login attempts allowed. The default is 0, which means the feature is disabled.
object_name	string(255)	S	Contains the name of the repository. (This property is inherited from dm_sysobject.) A repository name must be all ASCII characters and <=32 characters in length. The name docu is reserved by Documentum.
offline_checkin_flag	integer	S	Used by some Documentum clients to determine whether a client dialog box is used to check in content or the user's local check-in setting takes precedence. Valid values are: 0, meaning use a client dialog box for check ins 1, meaning allow the user's local check-in setting to take precedence

Property	Datatype	Single or repeating	Description
			The default value is 0.
offline_sync_level	integer	S	Identifies the level of repository access for offline synchronization. This is used primarily by some Documentum clients. Valid values are: 0, meaning none 1, meaning one-way access 2, meaning role-based access The default value is 0. **Note:** Refer to the client documentation for information about the actual use of this property and role-based synchronization.
oldest_client _version	string(32)	S	Version number of the oldest Documentum client that will access this repository. This must be set manually. It is used by the DFC to determine how to store chunked XML documents. If check_client_version is set to T, then this value is also used to identify the oldest client version level that may connect to the repository.
r_address _partitions	integer	S	This is no longer used.

Property	Datatype	Single or repeating	Description
r_dbms_name	string(32)	S	Name of the RDBMS. Valid values on Windows platforms are: Oracle, SQL Server, and DB2. Valid values on Unix platforms are: Oracle, Sybase, and DB2.
r_docbase_id	integer	S	The repository ID as a decimal value.
r_ending _partition	integer	S	This is no longer used.
r_federation _name	string(120)	S	Name of the federation to which the repository belongs, if any. The name is taken from the value of the object_name property in the dm_federation object.
r_module_mode	integer	R	Used internally
r_module_name	string(32)	R	Used internally
r_normal_tz	integer	S	Indicates the storage mode used for date values. 0 directs Content Server to store all date values in UTC time. A value other than 0 represents the time zone offset from UTC time used by Content Server to normalize all date values for storage. A time zone offset must be expressed as seconds. For example, if the offset is for PST (Pacific Standard Time), the value is -28800 (-8*60*60). This defaults to 0 for a new 6.0 or later repository and to an offset representing the server local time

Property	Datatype	Single or repeating	Description
			for a repository that is upgraded from a pre-6.0 release.
r_starting _partition	integer	S	This is no longer used.
r_storage_mode	integer	S	Used internally
richmedia _enabled	Boolean	S	Indicates whether the Content Server for this repository can process rich-media content. This value is set to TRUE during installation of a Content Transformation Services product.
security_mode	string(32)	S	Defines the security level for the repository. Valid values are acl or none.
trust_by_default	Boolean	S	Whether the repository accepts login tickets and application access control tokens from other repositories that have the same login ticket key (LTK) as this repository. T (TRUE) means the repository accepts login tickets and tokens from other repositories that share its LTK. F (FALSE) means that it only accepts login tickets and tokens from other repositories sharing its LTK if they appear in the trusted_docbases list. The default value is F.

Property	Datatype	Single or repeating	Description
trusted_docbases	string(255)	R	Name of the repositories from which login tickets and tokens are accepted. The respositories must have the same login ticket key as the repository represented by this docbase config object.
wf_package_control_enabled	Boolean	S	Controls whether Content Server exposes the object names of components in workflow packages. F allows Content Server to set the r_component_name property in dmi_package objects. T disallows setting the r_component_name property in package objects. If this property is T, the setting at the workflow level, in the package_control property of the dm_process object is ignored. The default value is F.

Docset

Purpose

Represents a set of documents to be processed by Content Intelligence Services.

Description

Supertype: Document
Subtypes: None
Internal name: dm_docset
Object type tag: 09

A docset object represents a set of documents to be processed by Content Intelligence Services. The documents are defined by a query recorded in the docset content. The content's format is crtext.

Properties

Table 60, page 186, lists the properties defined for the type.

Table 60. Properties defined for the docset type

Property	Datatype	Single or repeating	Description
description	string(255)	S	User-defined description of the document set
last_run_id	ID	S	Object ID of the docset run object last used to process this document set.
object_name	string(255)	S	Document set name
qualifiers	string(255)	R	List of conditions that documents must meet for inclusion in the docset.
queue_bound	Boolean	S	Controls whether documents placed on the processing queue can be considered for inclusion in this docset. Setting this to T allows documents on the processing queue to be considered for inclusion in this docset.
query_type	string(32)	S	Identifies the type of query. The only valid value is dql.

Docset Run

Purpose

Represents an executable collection of documents.

Description

Supertype: Document
Subtypes: None
Internal name: dm_docset_run
Object type tag: 09

A docset run object represents an executable collection of documents. Docset run objects are used by Content Intelligence Services.

Properties

Table 61, page 187, lists the properties defined for the type and those inherited properties that have a meaning specific to this type.

Table 61. Properties defined for the docset run type

Property	Datatype	Single or repeating	Description
docset_id	ID	S	Object ID of the document set associated with the processing run
estimated_count	integer	S	Estimated number of documents to process in the run
object_name	string(255)	S	Name of the docset run
processed_count	integer	S	Actual number of documents processed by the run
run_interval	integer	S	Not currently used
run_mode	integer	S	Not currently used
run_now	Boolean	S	T (TRUE) invokes processing immediately. The default is F (FALSE).

Property	Datatype	Single or repeating	Description
run_owner	string (32)	S	Name of the category manager who started the processing.
run_status	string(32)	S	Status of the processing run. Valid values are: not_started started scheduled completed failed
run_type	string(32)	S	Indicates whether this is a test run or an actual run. Valid values are: test active
start_time	Date	S	Not currently used
taxonomy_id	ID	R	Object IDs of all taxonomies included in the processing. Currently, only one taxonomy in a run is supported.
title	string(255)	S	Description of status state. The description may be user or system-defined.

Document

Purpose

Contains information about a document.

Description

Supertype: SysObject
Subtypes: Email Message, Esign Template, XML Config, XML Custom Code, XML Style Sheet, XML Zone, Notepage, TCF Activity, TCF Activity Template, XFM Form, XFM Instance
Internal name: dm_document
Object type tag: 09

Documents are the objects most commonly manipulated by users in the Documentum system. Documents can be simple documents or virtual documents. In a simple document, the content generally seen by a user is in one or more content files associated with the document. A virtual document is a document that is composed of components that are either simple documents or other virtual documents, and the content that users see is the content files associated with these components.

A virtual document can also have associated content files. That is, in addition to its components, a virtual document can have an associated content file (or files). However, when you open a virtual document for viewing or editing through WorkSpace, WorkSpace opens the component.

All content files associated directly with a document must have the same file format. The components of virtual documents can have a mixture of formats.

Properties

Documents inherit all of their properties from their supertype, the SysObject type.

Domain

Purpose

Describes the properties of a property's domain.

Description

Supertype: Persistent Object
Subtypes: Aggr Domain
Internal name: dm_domain
Object type tag: 50

A domain object describes the properties of a property's domain. The information contained in a domain object is part of the data dictionary. The domain information includes the property's datatype, length (if a string type), localized label text, and any expressions used to populate the property. Domain objects are created and managed by the server and cannot be created by users.

Properties

Table 62, page 190, lists the properties defined for the type.

Table 62. Properties defined for the domain type

Property	Datatype	Single or repeating	Description
dd_info	ID	S	Object ID of the dd info object containing the domain's information that does not depend on locale.
domain_length	integer	S	For string domains, the maximum length of the domain.
domain_name	string(32)	S	Currently unused.
domain_type	integer	S	Indicates the domain's datatype. Valid values are: 0, Boolean 1, Integer 2, String 3, ID 4, Time/Date 5, Double

Property	Datatype	Single or repeating	Description
nls_dd_info	ID	R	Object IDs of the nls dd info objects containing the domain's information that is dependent on locale.
nls_default	integer	S	Currently unused.
nls_keys	string(64)	R	Identifies the locales for which localized information is available. The values are the standard locale codes; for example, en or ja. The localized domain information for each locale is recorded in the nls dd info object identified in the corresponding index position of the nls_dd_info property. This property is set when a locale is populated.
parent_id	ID	S	Object ID of the aggr domain object that references this domain in its attr_domain_id property.
re_sync_dd	Boolean	S	Indicates whether the domain's property values accurately reflect the values in the associated type definition. If TRUE, indicates that the domain may not accurately reflect the information in the type definition.
super_domain_id	ID	S	Object ID of the domain object from which this domain is derived, if any.
super_domain _name	string(32)	S	Currently unused.

Dump Object Record

Purpose

Contains information about an object that has been copied from a repository into an external file using the Dump utility.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_dump_object_record
Object type tag: 30

A dump object record object contains information about an object that has been copied from a repository into an external file using the Dump utility. The information in a dump object record object is primarily useful if the dump process is interrupted and must be restarted.

Properties

Table 63, page 192, lists the properties defined for the type.

Table 63. Properties defined for the dump object record type

Property	Datatype	Single or repeating	Description
extra_data _dumped	Boolean	S	Used internally by the dmarchive utility.
dump_object	ID	S	Object ID of the dump record object associated with the dump procedure that generated this dump object record object.
object_id	ID	S	Object ID of the object that was copied from the repository.
version	integer	S	Version stamp of the dumped object.

Dump Record

Purpose

Contains information about a specific dump execution.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_dump_record
Object type tag: 2f

A dump record object contains information about a specific dump execution. The server uses this information to start a dump execution. Additionally, some of the properties contain recovery information if the dump process is interrupted and must be restarted.

Note: To start a dump process, you create a dump record object and save it. Saving a dump record object automatically starts the dump process. You must have Sysadmin or Superuser privileges to create and save a dump record object.

Properties

Table 64, page 193, lists the properties defined for the type.

Table 64. Properties defined for the dump record type

Property	Datatype	Single or repeating	Description
dump_operation	string(255)	S	Dumps the entire repository if set to full_docbase_dump. If NULL, dumps only the types specified by the type property.
dump_parameter	string(255)	R	Defines parameters for the dump operation, such as the cache size or content compression. Valid values are: • compress_content=T|F T directs the operation to compress the content. The default is F. • cache_size=*integer* *integer* defines the cache size in megabytes

Property	Datatype	Single or repeating	Description
file_name	string(255)	S	Name of the external file that the system is writing the dump into.
include_content	Boolean	S	Indicates whether or not to include the actual content files in the dump file. The default is false.
predicate	string(255)	R	Predicate expression that specifies which objects of the specified types are dumped.
predicate2	string(255)	R	Extends the predicate expression defined in the predicate property at corresponding index levels. For example, predicate2[0], if defined, is concatenated at the end of predicate[0] when the dump is executed.
r_current_object _count	integer	S	Number of objects dumped. This is updated continuously throughout the process, as objects are copied to the file.
r_current_pos	integer	S	Position in the dump file at which you wrote the last entry, if file is less than 2 GB.
r_current_root _count	integer	S	Indicates how many of the objects qualified by the predicate have been dumped. This is constantly refreshed during the dump process as the types specified in the predicate are dumped.
r_end_time	TIME	S	Time at which the dump ended.
r_is_complete	Boolean	S	Set at the completion of the dump operation. Values are: T (TRUE), meaning the operation completed successfully F (FALSE), meaning the operation terminated due to an error **Note:** The value is NULL if the dump record object was created prior to upgrading to version 5.3.

Property	Datatype	Single or repeating	Description
r_is_more	Boolean	S	This value is set when r_is_complete is set to T. Valid values are: T (TRUE), meaning the dump operation completed because it hit the objects_per_transfer limit, but there are more objects to dump F (FALSE), meaning that the dump operation completed and there are no more object to dump. The value of this property is undefined if r_is_completed is set to F (FALSE).
r_root_count	integer	S	Number of objects qualified by the predicate.
r_start_time	TIME	S	Time at which the dump was started.
type	string(33)	R	Identifies the types to be dumped. (Note that the predicate and type repeating properties are associated in that the predicate in position x is applied to the type in position x, the predicate in position y is applied to the type in position y, and so forth.)
r_current_offset	string(20)	S	Position in the dump file at which the server wrote the last entry if file is 2 GB or greater.

Email Message

Purpose

Stores the content of an email message.

Description

Supertype: Document
Subtypes: None
Internal Name: dm_email_message
Object type tag: 09

An email message object stores an electronic message as content. Any user can create an email message object.

Note: This is a deprecated object type.

Properties

Table 65, page 196, lists the properties defined for the type.

Table 65. Properties defined for the email message type

Property	Datatype	Single or repeating	Description
date_received	Date	S	The date and time at which the message was received.
date_published	Date	S	Date and time on which the message was sent.
media_type	string(32)	S	Identifies what kind of message this is. For example, possible values are email, IM (instant message), note, or calendar.
message_identifier	string(32)	S	The unique message identifier provided by the messaging system.
other_recipients	string(48)	R	Message recipients identified in the CC list.

Property	Datatype	Single or repeating	Description
originating_org	string(128)	S	The sender of the message
recipients	string(48)	R	Message recipients identified in the To list.

Esign Template

Purpose

Stores a signature page template as content

Description

Supertype: Document
Subtypes: None
Internal name: dm_esign_template
Object type tag: 09

An esign template object is used to store a signature page template in the repository. A default signature page template is provided with Content Server, and you can create custom signature page templates. Each template is represented in the repository by one esign template object.

Properties

Table 66, page 198, lists the properties defined for the type.

Table 66. Properties defined for the esign template type

Property	Datatype	Single or repeating	Description
append_to_body	Boolean	S	Whether to append or pre-pend the signature page to the content. T (TRUE) means to append the content. F (FALSE) means to pre-pend the content. The default is F.
begin_tag	string(1)	S	Identifies the beginning delimiter for tags on the signature page template. The default is <.
document_type	string(32)	R	Names of the object types that can be signed using the template.

Property	Datatype	Single or repeating	Description
end_tag	string(1)	R	Identifies the ending delimiter for tags on the signature page template. The default is >.
max_signatures	integer	S	Defines the maximum number of signatures that may added to a particular version of the object. The default is 0. There is no upper limit.

Expr Code

Purpose

Stores generated expression source code and pcode.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmi_expr_code
Object type tag: 58

An expr code object stores generated expression source code and pcode. Content Server creates and manages expr code objects. Users cannot create them.

Properties

Table 67, page 200, lists the property defined for the type.

Table 67. Properties defined for the expr code type

Property	Datatype	Single or repeating	Description
i_type_name	string(32)	S	Name of the type on whose behalf the source code and pcode are collected.
parent_id	ID	S	Object ID of an aggrdomain object. Used internally by dump and load operations.

Expression

Purpose

Stores information needed to execute expressions defined in the data dictionary.

Description

Supertype: Persistent
Object Subtypes: Func Expr, Literal Expr, Builtin Expr
Internal name: dm_expression
Object type tag: 52

An expression object stores information needed to execute expressions defined in the data dictionary. Expression objects are created and managed by Content Server and cannot be created by users.

Properties

Table 68, page 201, lists the properties defined for the type.

Table 68. Properties defined for the expression type

Property	Datatype	Single or repeating	Description
expression_name	string(32)	S	Currently unused. For the current release, this is always NULL.
expression_text	string(255)	R	The text of the expression. If the expression text is greater than 255 characters, the first 255 characters are put in expression_text[0], the second 255 in expression_text[1] and so forth.

Property	Datatype	Single or repeating	Description
expression_type	integer	S	The data type of the expression. If the expression subtype is literal expr or builtin expr, valid values are: 0, Boolean 1, Integer 2, String 3, ID 4, ,Time/Date 5, Double For the func expr subtype, valid values are: 0, Boolean 1, Integer 3, ID
parent_id	ID	S	Object ID of the aggr domain or policy object that references the expression object.

External File Store

Purpose

Represents an external file store.

Description

Supertype: External Store
Subtypes: None
Internal name: dm_extern_file
Object type tag: 61

An external file store object represents an external file store. The server uses external file store objects to locate content stored in external file stores.

Properties

Table 69, page 203, lists the properties defined for dm_extern_file.

Table 69. Properties defined for the external file store type

Property	Datatype	Single or repeating	Description
a_config_name	string(64)	R	Name of the server config object to use for a_location[position].
a_location	string(64)	R	Name of a location object to be used as root_name for a_config_name[position].
def_client_root	string(64)	S	Name of the client-specific location object. Default is NULL.
def_server _root	string(64)	S	Name of the server-specific location object. Default is NULL.

External Free Store

Purpose

Represents an external storage area accessed through a user-defined content token.

Description

Supertype: External Store
Subtypes: none
Internal name: dm_extern_free
Object type tag: 63

The external free store object type allows you to specify a token for content retrieval and storage that does not follow the standard for external file store or external URL store. You define your own token standard and means of retrieving the content associated with the token.

Configuration of plug-in properties for the external free store type is fully under the discretion of users. Depending on the accessibility of the content, users decide whether to run the plug-in on the server or client.

Properties

The external free store object type inherits all its properties from its supertype. No properties are defined specifically for it.

External Store

Purpose

Represents an external storage area accessible to the server but whose physical contents are stored outside the server.

Description

Supertype: Store
Subtypes: External File Store, External URL Store, External Free Store
Internal name: dm_extern_store
Object type tag: 60

An external store object represents an external storage area accessible to the server but whose physical contents are stored outside the server. An external store object eliminates the need to transfer the content from client to server during a save operation. All subtypes of dm_extern_store operate in token mode.

Properties

Table 70, page 205, lists the properties defined for the external store object type.

Table 70. Properties defined for the external store type

Property	Datatype	Single or repeating	Description
a_content_static	Boolean	S	Controls how getFile methods behave when fetching content if the storage area is an external storage area. For an explanation of the behavior, refer to the *Content Server Administration Guide*. The default is F.

Property	Datatype	Single or repeating	Description
a_exec_mode	Boolean	S	Controls where to execute the plug-in. FALSE directs the DFC to execute the plug-in on the server host. This must be set to FALSE if the DFC is at Version 6 or later. TRUE is a deprecated value for this property. It executes the plug-in on the client host. DFC Version 6 does not support executing the plug-in on the client host. The default is FALSE
a_plugin_id	ID	R	ID of the document whose content is the plug-in DLL (Windows) or shared library (UNIX). The storage type for this document must be dm_filestore.
a_platform	integer	R	Identifies the client platform. Valid values are: 1, for Windows 2, for Solaris 3, for AIX 4, for HP-UX 5, for Macintosh 6, for Linux
a_storage_param_name	string(64)	R	User-defined name of the attribute used to hold specific information pertaining to the store
a_storage_param_value	string(1024)	R	User-defined description of the attribute named at the corresponding position in a_storage_param_name
is_writable	Boolean	S	Designates if this external store supports writable operations
storage_class	string(64)	S	Identifies what the store will be used for

External URL Store

Purpose

Represents an external storage area whose content is accessed using a token that follows the URL standard.

Description

Supertype: External Store
Subtypes: None
Internal name: dm_extern_url
Object type tag: 62

An external URL store object represents objects whose content is stored externally and accessed using token-mode operation, in which the token follows the URL standard. For the external URL store object type, it is best to configure the plug-in for content retrieval to be executed on the client side. URLs are universal and can be accessed from anywhere. Content Server does not validate the URL.

Properties

The external URL store object type inherits all its properties from its supertype. No properties are defined specifically for it.

Federation

Purpose

Contains the description of a federation.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_federation
Object type tag: 5e

A federation object contains the description of a federation. Federation objects are created and managed by Content Server when you create a federation using Documentum Administrator. Each repository belonging to a federation will have a federation object that describes the federation.

Properties

Table 71, page 208, lists the properties defined for the type.

Table 71. Properties defined for the federation type

Property	Datatype	Single or repeating	Description
r_govern_docbase	string(120)	S	Name of the governing repository in the federation. This value serves as the default name of the federation unless the object's object_name property is explicitly set. This property is set in the federation objects for all members of the federation.

Property	Datatype	Single or repeating	Description
r_govern_is_active	Boolean	S	Indicates whether the governing repository is active. If this is FALSE (the default), federation-related jobs cannot be executed. This value is set only in the federation object in the governing repository.
r_govern_ suspended	Boolean	S	Indicates whether transactions or changes to configuration objects can occur in the governing repository. The default is FALSE, meaning that transactions are allowed. (This is used to manage synchronization and integrity between repositories in the federation while jobs are executing.) This value is set only in the federation object in the governing repository.
r_member_ docbases	string(120)	R	Names of the repositories that belong to the federation. Each name must be unique within the list. This value is set only in the federation object in the governing repository.

Property	Datatype	Single or repeating	Description
r_member_is_ active	Boolean	R	Indicates whether the member repository is active in the management of configuration objects. The value at each index level is associated with the repository at the corresponding index level in r_member_docbases. The default is TRUE, meaning that the repository participates. This value is set only in the federation object in the governing repository.
r_member_refresh	date	R	Indicates the last refresh date of the configuration objects in the member repositories. The date at each index level is associated with the repository at the corresponding index level in r_member_docbases. This value is set only in the federation object in the governing repository.
user_subtypes	string(32)	R	dm_user subtypes to propagate. The federation update job propagates users represented by the users subtypes listed in this property (in addition to users represented by dm_user objects). The subtypes must exist on all member repositories.

File Store

Purpose

Contains information about a file store storage area.

Description

Supertype: Store
Subtypes: none
Internal name: dm_filestore
Object type tag: 28

A file store object contains information about a file store storage area. A storage area of type dm_filestore is one of the most common types of file storage areas in a Content Server installation. It is used to store files that have a wide variety of formats.

Properties

Table 72, page 211, lists the properties defined for the type.

Table 72. Properties defined for the file store type

Property	Datatype	Single or repeating	Description
is_public	Boolean	S	Indicates if the area is accessible to the public with no restrictions. Valid values are: 1, for is accessible 0, for is not accessible
root	string(64)	S	Object name of the location object representing this storage area
use_extensions	Boolean	S	Indicates whether server should append a DOS extension to the file when writing it into the storage area.

Folder

Purpose

Used, in conjunction with cabinets, to organize the contents of your repository.

Description

Supertype: SysObject
Subtypes: Cabinet, Category, Room, Module, Topic, DSM Application, DSM Submission, XML Application
Internal name: dm_folder
Object type tag: 0b

The folder object serves, in conjunction with cabinets, to organize the contents of your repository. All SysObjects and SysObject subtypes (except cabinets) must be stored either directly in a cabinet or in a folder. Folders, in turn, are stored in cabinets or other folders. Ultimately, every SysObject or subtype is stored in a cabinet.

Properties

Table 73, page 212, lists the properties defined for the folder type.

Table 73. Properties defined for the folder type

Property	Datatype	Single or repeating	Description
i_ancestor_id	ID	R	Object ID of the folders or cabinets that contain this folder directly or indirectly.
r_folder_path	With Oracle and DB2: string(740)	R	Folder paths for all locations to which the folder is linked.
	With Sybase: string(600)		The length of this property will vary for SQL Server customers:
	With SQL Server: see Description		For a new SQL Server repository, the length is 450.
			For an upgraded SQL Server repository not using Unicode, the length is 765.

Foreign Key

Purpose

Describes the set of properties that define a foreign key.

Description

Supertype: Relation Type
Subtypes: None
Internal name: dm_foreign_key
Object type tag: 65

A foreign key object describes the set of properties that define a foreign key. Content Server creates foreign key objects when users define foreign keys for a type. Users cannot create foreign key objects.

Properties

Table 74, page 213, lists the properties defined for the type.

Table 74. Properties defined for the foreign key type

Property	Datatype	Single or repeating	Description
child_attributes	string(32)	R	Names of the child key properties for the foreign key.
parent_attributes	string(32)	R	Names of the parent key properties for the foreign key.

Format

Purpose

Records information about a file format recognized by Content Server.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_format
Object type tag: 27

A format object contains information about a file format recognized by Content Server. A predefined set of file formats is installed by default when a repository is configured.

Properties

Table 75, page 214, lists the properties defined for the type.

Table 75. Properties defined for the format type

Property	Datatype	Single or repeating	Description
a_page_plugin_name	string(255)	S	Name of the plug-in that implements the ability to retrieve designated pages of content in this format. **Note:** In this context, "pages" refers to the numbered pages of the content, not the content's position within the files associated with an object.
asset_class	string(32)	S	Identifies the kind of asset (video, audio, and so forth) represented by this format. This is used by applications.
can_index	Boolean	S	Indicates whether this format can be full-text indexed. This property is set by an entry in a configuration file.

Property	Datatype	Single or repeating	Description
com_class_id	string(38)	S	The class ID (CLSID) recognized by the Windows registry for a content type.
default_storage	ID	S	Identifies the default storage area for contents having this format.
description	string(64)	S	User-defined description of the format.
dos_extension	string(10)	S	The DOS extension to use when copying a file in the format into the common area, client local area, or storage.
filename _modifier	string(16)	S	The modifier to append to a file name, to create a unique file name.
format_class	string(32)	R	Content Server may set this to either: • ft_always, meaning that content files in this format are always indexed • ft_preferred, meaning if there are multiple renditions of the content, this format is the preferred format for indexing May be also be set to a user-defined value to identify the class or classes of formats to which the format belongs. For example, the xml, xsd, and xsl formats belong to the XML and MSOffice classes.
icon_index	integer	S	The index that locates the icon for the format in the icon resource file.
is_hidden	Boolean	S	Used by some Documentum clients.

Property	Datatype	Single or repeating	Description
mac_creator	string(4)	S	Information used internally for managing Macintosh resource files.
mac_type	string(4)	S	Information used internally for managing Macintosh resource files.
mime_type	string(256)	S	The Multimedia Internet Mail Extension (MIME) for the content type.
name	string(64)	S	Name of the format; for example, doc or tiff.
richmedia _enabled	Boolean	S	Indicates whether thumbnails, proxies, and metadata are generated for content in this format. This is set to TRUE by default for the following formats: jpeg, mpeg, pdf, gif, and avi. It is FALSE for all other formats by default.
topic_filter	string(64)	S	Obsolete
topic_format	ID	S	Obsolete
topic_format _name	string(64)	S	Obsolete
topic_transform	Boolean	S	Indicates if it is necessary to transform the format before indexing
win31_app	string(12)	S	Used by Documentum clients. It identifies the application to launch when users select a document in the format represented by the format object.

FT Index Agent Config

Purpose

Stores information about an index agent.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_ftindex_agent_config
Object type tag: 08

An ft index agent config object stores information about one index agent.

Properties

Table 76, page 217, lists the property defined for the type plus two that are inherited and used in a specific manner by this type.

Table 76. Properties defined for the ft index agent config type

Property	Datatype	Single or repeating	Description
active_connectors	string(32)	S	Whether the index agent is running in normal or migration mode. Valid values are; **normal**, indicating normal mode **reindex**, indicating migration mode If the property is not set or is blank, the default is normal.
checkpoint_interval	integer	S	Reserved for future use
connectors_batch_size	integer	S	Number of queue items picked up by the queue reader in each polling interval. The default is 1000.

Property	Datatype	Single or repeating	Description
docbase_connector_wait_time	integer	S	Frequency, in seconds, at which the index agent polls for events The default is 60.
exporter_queue_threshold	integer	S	Used to control additions to the index agent's internal, in-memory queue. If the number of objects in the queue is equal to or exceeds the value in this property, then no more objects may be added to the queue. The default value is 500.
exporter_thread_count	integer	S	Number of concurrent exporter threads being run by the index agent The default is 3.
force_inactive	integer	S	Indicates whether the agent is operating normally or shutdown and waiting for a manual start. Valid values are: 0, meaning the agent is operating normally 1, meaning the server will not start the agent until this property is set to 0 manually. The default is 0.
index_name	string(64)	S	Name of the fulltext index object associated with this index agent.

Property	Datatype	Single or repeating	Description
indexer_queue_ threshold	integer	S	Used to control additions to the index agent's internal, in-memory queue. If the number of objects in the queue is equal to or exceeds the value in this property, then no more objects may be added to the queue. The default value is 500.
object_name	string(255)	S	This is an inherited property. For ft index agent config objects, the value must be unique among all ft index agent config objects in the repository. The default value for the index agent installed with Content Server is *hostname_indexagentname*.
queue_user	string(64)	S	User name used by the index agent to look for queued items The default is dm_fulltext_ index_user.
r_fail_time	Date	S	Reserved for future use
r_is_active	integer	S	Whether the index agent is active or inactive. Valid values are 0, meaning the agent is inactive, and 1, meaning the agent is active.
r_last_done_time	Date	S	Reserved for future use
r_last_work_time	Date	S	Reserved for future use
r_modify_date	Date	S	This is an inherited property. For ft index agent config objects, this is set to the last time the associated index agent updated the ft index agent config object.
r_start_time	Date	S	Reserved for future use
r_stop_time	Date	S	Reserved for future use

Property	Datatype	Single or repeating	Description
runaway_item_ timeout	integer	S	How long a queue item remains in the internal queue before it is abandoned. The value is interpreted in seconds. The default value is 600.
save_queue_items	integer	S	Whether successfully processed queue items are saved or destroyed. 0 means to mark successfully processed queue items “done” and save them. 1 means to destroy successfully processed queue items. The default is 1.
shut_down_requested	integer	S	Reserved for future use

FT Engine Config

Purpose

Records configuration information for an index server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_ftengine_config
Object type tag: 08

An ft engine config object stores the parameters necessary to use a particular index server for indexing or querying. There is one ft engine config associated with each fulltext index object in a repository. Permissions to modify objects of this type are controlled by the ACL named dm_fulltext_admin_acl.

Caution: Changing the values in the param_name or param_value properties manually may result in undefined behavior during indexing or querying. It is recommended that you do not change these properties unless told to do so by technical support.

Properties

Table 77, page 221, lists the properties defined for the type.

Table 77. Properties defined for the FT engine config type

Property	Datatype	Single or repeating	Description
param_name	string(64)	R	Names of the parameters passed to the engine.
param_value	string(255)	R	Values of the parameters passed to the engine. The value at each index position is associated with the name at the same index position in param_name.

Fulltext Index

Purpose

Represents a full-text index for the repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_fulltext_index
Object type tag: 3b

A fulltext index object represents the index associated with the repository. The object is created internally when a 'normal mode' index agent is configured for the repository. Only the ft_engine_id property can be modified by a user. Superuser or Sysadmin user privileges are required to change this property.

Properties

Table 78, page 222, lists the properties defined for the type.

Table 78. Properties defined for the fulltext index type

Property	Datatype	Single or repeating	Description
codepage	string(64)	S	Identifies the code page in use for strings sent to and received from the Index Agent. Typically, this is UTF-8.
ft_engine_id	ID	S	Object ID of the dm_ftengine_config object used for this index.
index_name	string(64)	S	Name of the fulltext index object. This must consist of ASCII characters.

Property	Datatype	Single or repeating	Description
install_loc	string(32)	S	Name of the location object that identifies the location of the fulltext installation for this index. The default name is the value of the fulltext_location property in the server config object.
is_standby	Boolean	S	Reserved for internal use.
r_content_count	integer	S	Reserved for future use
r_index_type	integer	S	Indicates the type of index. The default value is 4.
r_last_clean	integer	S	Reserved for future use
r_last_update	DATE	S	Time and date of the last update for this index.

Func Expr

Purpose

Stores the data dictionary information needed to execute expressions written in a third-party expression language.

Description

Supertype: Expression
Subtypes: Cond Expr
Internal name: dm_func_expr
Object type tag: 55

A func expr object stores the data dictionary information needed to execute expressions written in a third-party expression language. Content Server creates and manages func expr objects. Users cannot create them directly.

Properties

Table 79, page 224, lists the properties defined for the type.

Table 79. Properties defined for the func expr type

Property	Datatype	Single or repeating	Description
attr_object_index	integer	R	For each property in attribute_name, provides an index into the type_name list that identifies the source type of the property. The value in attr_object_index[0] is the index value to use for the property specified in attribute_name[0]. For example, if attr_object_index[0] is 3, then the property specified in attribute_name[0] is taken from the type specified in type_name[3}.

Property	Datatype	Single or repeating	Description
attribute_name	string(32)	R	List of property names that are passed to the expression. The names are passed in the order in which they are listed.
code_page	integer	R	Content page number in the object specified in routine_id that contains the generated code. Currently, this value is always 1.
expression_kind	integer	S	Indicates whether the expression is a complete routine or an expression. Valid values are: 1, Expression 2, User routine
expression_lang	integer	S	Identifies the language of the expression or routine. The only valid value is 1, meaning Docbasic.
object_alias	string(128)	R	Identifies an alias for each type named in type_name. The aliases are used to map properties in the expression_text to their associated types. The alias at each index level is associated with the type specified at the corresponding index level in type_name.
repeat_attr_index	integer	R	For each repeating property in attribute_name, this provides the index into the property's list of values, to identify the desired value. Valid values are: 0 or greater, Index into the repeating property -2, Any value in the property -3, All values in the property -4, First value in the property -5, Last value in the property

Property	Datatype	Single or repeating	Description
			For single-valued properties specified in attribute_name, the corresponding value in repeat_attr_index is always -1.
routine_id	ID	S	Object ID of the SysObject containing the source and pcode content pages.
routine_name	string(255)	S	Name of the entry point in the routine.
source_page	integer	S	Content page number in the object specified in routine_id containing the source code. Currently, this value is always 0.
type_name	string(32)	R	Names of the object types that are the sources of the properties specified in attribute_name.

Group

Purpose

Contains information about a group in the repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_group
Object type tag: 12

A group object contains information about a group in the repository.

Properties

Table 80, page 227, lists the properties defined for the group type.

Table 80. Properties defined for the group type

Property	Datatype	Single or repeating	Description
alias_set_id	ID	S	The object ID of an alias set associated with the group.
description	string (255)	S	User-defined description of the group.
globally_ managed	Boolean	S	Indicates whether the group is globally or locally managed. The default is FALSE, meaning that it is locally managed. Requires at least Sysadmin privileges to change.
group_address	string(80)	S	Electronic mail address for the group.
group_admin	string(32)	S	Name of a user or group who can modify the group.

Property	Datatype	Single or repeating	Description
group_class	string(32)	S	Indicates what kind of group this group is. Valid values are: • group • role • module role • privilege group • domain
group_directory_id	ID	S	Object ID of the LDAP config object representing the LDAP directory used to synchronize this group.
group_display_name	string(255)	S	Name of the group as it appears in a room. The name must be unique within the room. The default is the group_name.
group_global_ unique_id	string(255)	S	Reserved for future use
group_name	string(32)	S	Name of the group. If group_class is set to role, then group_name is the name of the role. If group_class is set to domain, then group_name is set the name of the domain. The name must consist of characters compatible with the server os code page of the Content Server and must be unique among the user and group names in the repository.
group_native_room_id	ID	S	Object ID of the room that contains this private group.

Property	Datatype	Single or repeating	Description
group_source	string(16)	S	Identifies the source of the group. The only valid value is LDAP, meaning the group was created by importing an LDAP group.
groups_names	string(32)	R	Names of any groups that are users in this group.
i_all_users _names	string(32)	R	List of all users in the group, including those users that belong to groups contained within the group. This property value is computed when the property is first queried in a session and cached on the server and client.
i_nondyn_ supergroups_names	string(32)	R	Names of the non-dynamic groups to which this group belongs
i_supergroups _names	string(32)	R	Name of the group and all groups that contain this group.
is_dynamic	Boolean	S	T (TRUE) means the group is a dynamic group. F (FALSE) means the group is not a dynamic group. The default is F.
is_dynamic_default	Boolean	S	Controls whether users in the group's list of potential users are considered members of the group by default when they connect to the respository. T (TRUE) means that users are treated as members of the group. F (FALSE) means that users are not treated as group members when they connect. The default is F (FALSE).

Property	Datatype	Single or repeating	Description
is_module_only	Boolean	S	T means that this is a dynamic group to which only privileged BOF modules may add members.
is_protected	Boolean	S	T means that this is a dynamic group to which only authorized client installations may add members. **Note:** Client installation authorizations are recorded in the client rights database.
is_private	Boolean	S	Indicates whether the group is private (T) or public (F).
owner_name	string(32)	S	Name of the user or group who owns the group.
r_has_events	Boolean	S	Whether someone has registered the group for auditing.
r_object_id	ID	S	Object ID of the group.
users_names	string(32)	R	Names of the directly contained users in the group.

Index

Purpose

Stores information about an RDBMS index created in the repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_index
Object type tag: 1f

An index object stores information about an RDBMS index created in the repository.

Properties

Table 81, page 231 lists the properties defined for the type.

Table 81. Properties defined for the index type

Property	Datatype	Single or repeating	Description
attribute	integer	R	Position of the properties being indexed in the type's definition.
attr_count	integer	S	Number of properties (columns) in the index
data_space	string(64)	S	Name of the Oracle tablespace or MS SQL Server or Sybase database in which the index resides.
index_type	ID	S	Object ID of the dm_type object that represents the type being indexed.
is_unique	Boolean	S	Indicates if the rows in the index are unique
name	string(64)	S	Name of the index. This is generated internally.

Property	Datatype	Single or repeating	Description
rebuilding	Boolean	S	This is TRUE if the index is currently being moved from one location to another.
repeating	Boolean	S	Indicates whether the index is built on the type's single-valued property table or the repeating property table. Values are F for the single-valued property table and T for the repeating property table.
use_id_col	Boolean	S	Used by server for server-generated indexes.
use_pos_col	Boolean	S	Used by server for server-generated indexes.
use_tag	integer	R	Used by server for server-generated indexes.

Jar

Purpose

Represents a jar file stored in the repository.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_jar
Object type tag: 08

A jar object contains information about a jar file stored in the repository as the object's content. The object type is installed by a script when Content Server is installed. Instances of the type are created through Documentum Application Builder.

Properties

Table 82, page 233, lists the properties defined for the type.

Table 82. Properties defined for the jar type

Property	Datatype	Single or repeating	Description
jar_type	integer	S	Identifies what type of jar file this object represents. Valid values are: 1, meaning interface 2, meaning implementation 3, meaning both interface and implementation
minimum_vm_version	string(32)	S	Specifies the VM version level required by this file.

Java Library

Purpose

Represents a third-party Java library stored in the repository.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_java_library
Object type tag: 0b

Java library objects represent java libraries stored in the repository. Each object represents one library. The object type is installed by a script when Content Server is installed. Instances of the type are created through Documentum Application Builder.

Properties

Table 83, page 234, lists the property defined for the type.

Table 83. Properties defined for the java library type

Property	Datatype	Single or repeating	Description
sandbox_library	Boolean	S	Indicates how the library is to be loaded T (TRUE) means the library is loaded by the Business Object class loader. F (FALSE) means the library loaded by the Shared class loader.

JMS Config

Purpose

Reserved for internal use.

Description

Supertype: Sysprocess Config
Subtypes: None
Internal name: dm_jms_config
Object type tag: 08

Do not use this type. It is reserved for internal use only.

Job

Purpose

Used to run a program at regularly scheduled intervals.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_job
Object type tag: 08

A job object represents a program that you want to run at regularly scheduled intervals. A job object stores information about the program, such as the name of the method object for the program, how often or when to run it, the number of times the program has been run, the date of its last execution, and so forth. Programs represented by job objects are launched by a special utility called the dm_agent_exec utility, which is itself started by Content Server.

Properties

Table 84, page 236, lists the properties defined for the type.

Table 84. Properties defined for the job type

Property	Datatype	Single or repeating	Description
a_continuation _interval	integer	S	Indicates how often to re-invoke the job, in minutes. If set to a number greater than zero, the job is automatically re-started after the specified interval. The default is 0.
a_current _status	string(255)	S	Contains status information provided by the application or program run by the job.
a_is_continued	Boolean	S	Indicates whether the current execution of the job is a result of an automatic re-invocation of the job. The default is FALSE.
a_iterations	integer	S	Number of times the job has been invoked and completed.

Property	Datatype	Single or repeating	Description
a_last_completion	date	S	Contains the date and time when the last invocation of the job was completed.
a_last_document _id	ID	S	Object ID of the status document created for the last invocation of the job.
a_last_invocation	date	S	Contains the date and time when the last invocation of the job was started.
a_last_process_id	ID	S	The process id of the last-launched dm_method.
a_last_return_code	integer	S	Return status of the last dm_method executed.
a_next _continuation	date	S	Date of next automatic invocation of the job. The default is NULLDATE.
a_next_invocation	date	S	Computed time of next invocation.
a_special_app	string(32)	S	This property is inherited from dm_sysobject. It is set by the server when the job is checked out by the dm_agent_exec utility for execution.
expiration_date	date	S	Date and time on which the job expires. The job will not be executed after the specified date and time. The default is NULLDATE.
inactivate_after _failure	Boolean	S	Indicates if the job is inactive due to failure. The default is FALSE.
is_inactive	Boolean	S	Indicates whether the job is unavailable for execution. The default is FALSE, which indicates that the job can be executed.

Property	Datatype	Single or repeating	Description
max_iterations	integer	S	Maximum number of times to execute the job. The default is 1. A value of zero (0) indicates that there is no maximum.
method _arguments	string(255)	R	Arguments to pass to the method. These arguments are passed only if the pass_standard_arguments property is set to F.
method_data	string(255)	R	Used by procedure associated with job. This property is available for the program to write/read as needed during its execution.
method_name	string(255)	S	Name of the method to invoke when the job is executed. This is a required property.
method_trace _level	integer	S	Controls tracing for the method. Any value other than 0 turns on tracing. 0, the default, turns off tracing.
object_name	string(255)	S	Name of the job object. This is inherited from dm_sysobject.
pass_standard _arguments	Boolean	S	T means that the standard arguments are passed to the method and the arguments in method_arguments are not passed. (The standard arguments are: -docbase_name *repositoryname.serverconfig* -user_name *username* -job_id *jobid* -method_trace_level *methodtracelevel*) F means that the standard arguments are not passed to the method, but the arguments in

Property	Datatype	Single or repeating	Description
			the method_arguments value are passed instead. The default is F.
run_interval	integer	S	Used in conjunction with run_mode to determine how often to invoke the job. For example, if you enter 3 for this property and 4 for run_mode, the job is invoked every three weeks. The default is zero (0).
run_mode	integer	S	Used in conjunction with run_interval to determine how often to invoke the job. Run_ mode specifies the unit of measure for the value in run_interval. Valid values are: 1, for every x minutes 2, for every x hours 3, for every x days 4, for every x weeks 5, for every x months 7, for every x day of week 8, for every x day of month 9, for every x day of year x is the value in run_interval The default value is zero (NULL).
run_now	Boolean	S	If set to T, the job is executed the next time the agent exec program polls the repository.

Property	Datatype	Single or repeating	Description
start_date	date	S	Date and time of the first required invocation. This is a required property.
target_server	string(255)	S	Specifies a particular server to run the job. Specify the server name using the format: *repository*[.*server_config_name*] [@*machine*] The default value is NULLSTRING.

Job Request

Purpose

Used internally by Documentum Administrator.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_job_request
Object type tag: 08

A job request object records information needed by Documentum Administrator to execute certain jobs. Users cannot create these objects manually.



Caution: Users, or applications, must not modify these objects. However, they may be deleted if request_completed is T, indicating that the associated job has been invoked.

Properties

Table 85, page 241, lists the properties defined for the object type.

Table 85. Properties defined for the job request type

Property	Datatype	Single/rRepeating	Description
arguments_keys	string(255)	R	Internal keys used in conjunction with arguments_values. The key at any particular index position is used in conjunction with the value at the corresponding index position in arguments_values.
arguments_values	string(255)	R	Argument values passed to the job.
job_name	string(255)	S	Name of the requested job

Property	Datatype	Single/rRepeating	Description
method_name	string(255)	S	Name of the method invoked by the job
priority	integer	S	A priority value internally assigned by Documentum Administrator
request_completed	Boolean	S	Whether the job has been executed. T means the job has been invoked. F means the job is awaiting execution.

Job Sequence

Purpose

Identifies a job that belongs to a set of jobs executed in a particular sequence.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_job_sequence
Object type tag: 08

A job sequence object stores information about a job that belongs to a set of jobs executed in a particular sequence. Sequenced jobs are executed using the dm_run_dependent_jobs method, which can be invoked by another, controlling job or on the command line. Job sequence objects are created using Documentum Administrator.

Properties

Table 86, page 243, lists the properties defined for the object type and the one inherited property that has a specific use for job sequences.

Table 86. Properties defined for the job sequence type

Property	Datatype	Single or repeating	Description
job_display_name	string(255)	S	Reserved for use by Documentum Administrator
job_docbase_name	string(255)	S	Identifies the respository in which the job resides. This string is used to match an entry in the repository connection file if that file is used. Valid values for this string are: *repository_name* *repository_name.content_server_name* *repository_name@host_name*

Property	Datatype	Single or repeating	Description
			repository.content_server_name@host_name
job_id	ID	S	Object ID of the job
job_login_domain	string(255)	S	Login domain of the user identified in job_login_user_name
job_login_user_name	string(32)	S	Login name of the user as whom the job identified in job_id will run
object_name	string(255)	S	This inherited property must be set to the name of the job sequence that includes the job identified in this job sequence object.
predecessor_id	ID	R	Identifies the job or jobs that must complete successfully before the job specified in job_id is run. The jobs are identified by their object IDs. This property may be empty.

Key

Purpose

Contains information describing a key for an object type.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_key
Object type tag: 59

A key object contains information describing a key for an object type. Content Server creates key objects when users define keys for a type. Users cannot create key objects.

Properties

Table 87, page 245, lists the properties defined for the type.

Table 87. Properties defined for the key type

Property	Datatype	Single or repeating	Description
is_unique	Boolean	S	Indicates whether the key is a unique key.
key_attributes	string(32)	R	Names of the properties in the key definition.
type_name	string(32)	S	Name of the object type containing the key properties.
parent_id	ID	S	Object ID of an aggrdomain object. Used internally by dump and load operations.

LDAP Config

Purpose

Stores the set-up values that configure a repository's use of an LDAP-compliant directory server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_ldap_config
Object type tag: 08

An ldap config object stores the set-up values that configure a repository's use of an LDAP-compliant directory server. The set-up values are defined through Documentum Administrator when the repository is set up to use the LDAP directory server. Setting the values in an ldap config object requires Superuser user privileges.

Properties

Table 88, page 246, lists the properties defined for the ldap config type.

Table 88. Properties defined for the LDAP config type

Property	Datatype	Single or repeating	Description
a_last_run	date	S	Used internally
a_last_no	string(20)	S	Used internally
bind_dn	string(255)	S	A distinguished name for binding This is a required property.
bind_pwd	string(32)	S	Password for the binding name This is a required property.

Property	Datatype	Single or repeating	Description
bind_type	string(16)	S	Defines the binding mode used to authenticate LDAP users. Valid values are: *bind_search_dn*, which uses the user's operating system name to find the Distinguished name for authentication. *bind_by_dn*, which uses the Distinguished name stored in the user's user_ldap_dn property to perform authentication.
certdb_location	string(255)	S	Name of the location object that points to the location of the certificate database directory.
deactivate_user_option	Boolean	S	TRUE causes the synchronization job to set users to inactive in the repository if they are inactivated in the Directory Server; FALSE disables the repository update. The default is FALSE.
failover_ldap_config_ids	ID	R	Object IDs of the ldap config objects representing the LDAP directory servers used as secondary (failover) servers for this LDAP directory server.
failover_use_interval	time	S	Defines the length of time during which Content Server uses a secondary LDAP directory server for user authentication before attempting to contact the primary LDAP directory server. The default is 5 minutes.

Property	Datatype	Single or repeating	Description
first_time_sync	Boolean	S	Whether the LDAP directory must perform a first-time synchronization of the users and groups in the respository that match users and groups in the directory. T means to perform such a synchronization. F means not to perform the synchronization. The default is F.
group_obj_class	string(64)	S	Identifies the object class of groups, as defined in the LDAP directory server. This is optional. The default is groupofuniquemembers.
grp_search_base	string(256)	S	Defines the starting point in the directory server's schema for group searches. (Refer to the documentation provided by the directory server vendor for information about search bases.)
grp_search _filter	string(256)	S	Defines a filter for group searches. (Refer to the documentation provided by the directory server vendor for information about the format of a filter specification.)
import_mode	string(7)	S	Controls whether LDAP synchronization is performed for users, groups, or both. Valid values are: user, group, and all. The default is all.
ldap_host	string(128)	S	The host name of the machine on which the LDAP directory server is running. This is required.

Property	Datatype	Single or repeating	Description
map_attr	string(32)	R	Identifies object properties that are set when the user or group is imported into the repository. The property at a particular index position is set to the value identified at the same index position in map_val.
map_attr_type	string(32)	R	Object type for which the property identified at the corresponding index position in map_attr is defined.
map_const_attr	string(32)	R	currently unused
map_const_val	string(32)	R	currently unused
map_rejection	integer	R	Define the rejection rule for a mapped property. If the value mapped to the property fails to satisfy the rule associated with the property, the user or group entry is rejected and is not synchronized in the repository. Valid values are: • 0, meaning synchronization always occurs even if an LDAP attribute referenced in the mapping is missing, empty, or does not have sufficient characters. • 1, meaning that synchronization does not occur if an LDAP attribute in the mapping is missing or empty. However, synchronization will occur even if an LDAP attribute in the mapping has fewer characters than specified in the mapping. • 2, meaning that synchronization does not occur if an LDAP attribute

Property	Datatype	Single or repeating	Description
			referenced in the mapping is missing, empty, or has fewer characters than specified in the mapping. **Note:** For mandatory properties (both users and groups), the map rejection value is automatically set to 2 and cannot be changed.
map_val	string(64)	R	The name of a Directory server property or a hard-coded value to which the property identified in the corresponding index position in map_attr is mapped.
map_val_type	string(1)	R	Indicates whether the value in the corresponding index position in map_val is a Directory Server attribute name, an expression, or an actual value. Valid values are: A, meaning the value is a Directory Server property E, meaning the value is an expression V, meaning the value an actual value
per_search_base	string(256)	S	Defines the starting point in the directory server's schema for user searches. (Refer to the documentation provided by the directory server vendor for information about search bases.)
per_search _filter	string(256)	S	Defines a filter for user searches. (Refer to the documentation provided by the directory server vendor for information about the format of a filter specification.)

Property	Datatype	Single or repeating	Description
person_obj _class	string(64)	S	Identifies the object class of users as defined in the LDAP directory server. This is optional. The default is person.
port_number	integer	S	The port number on the LDAP directory server's host machine which Content Server will use to communicate with the directory server.
rename_group _option	Boolean	S	TRUE updates group names in the repository during synchronization if they have changed in the Directory Server; FALSE disables this option. The default is FALSE.
rename_user _option	Boolean	S	TRUE updates user names in the repository during synchronization if they have changed in the Directory Server; FALSE disables this option. The default is FALSE.
retry_count	integer	S	Defines the number of times Content Server will attempt to contact the LDAP directory server before reporting that it cannot connect. The default is 3.
retry_interval	time	S	Defines the time interval that must elapse before Content Server tries to contact the LDAP directory server again. The default is 5 seconds.

Property	Datatype	Single or repeating	Description
ssl_mode	integer	S	Whether LDAP user authentication takes place over a secure socket layer (SSL). Values are 0, meaning SSL is not used and 1, meaning that SSL with server authentication is used. The default is 0.
ssl_port	integer	S	Port number of the LDAP SSL port. The default is 636.
use_ext_auth _prog	Boolean	S	TRUE directs Content Server to use an external password checking program to authenticate LDAP users. FALSE directs the server to authenticate LDAP users itself. The default is FALSE.
user_subtype	string(64)	S	Identifies the user's object type if the user is a subtype of dm_user.

Linked Store — Deprecated

Purpose

A linked store object represents a storage area that links a common directory accessible to all clients and an actual file directory.

Description

Supertype: Store
Subtypes: None
Internal name: dm_linkedstore
Object type tag: 2a

A linked store object contains information about the link between a common directory accessible to all clients and an actual file directory.

Caution: The linked store object type is deprecated as of Version 6.0. DFC Version 6 does not support linked store storage areas.

Properties

Table 89, page 253, lists the properties defined for the type.

Table 89. Properties defined for the linked store type

Property	Datatype	Single or repeating	Description
link_location	string(64)	S	Name of the location object that represents the directory that contains the logical link.
symbolic_links	Boolean	S	Indicates if the links defined in the link directory are symbolic or hard links. TRUE indicates symbolic links; FALSE indicates hard links.

Link Record — Deprecated

Purpose

Contains information about established links to content storage areas.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_linkrecord
Object type tag: 2b

A link record object contains information about established links to content storage areas.

Note: The dmi_linkrecord object type is deprecated. DFC Version 6 does not support linked storage areas nor, consequently, does it support or manage link record objects.

Properties

Table 90, page 254, lists the properties defined for the type.

Table 90. Properties defined for the link record type

Property	Datatype	Single or repeating	Description
parent_id	ID	S	Object ID of the object with which the content is associated.
component_id	ID	S	Object ID of the content object associated with the content.
format_id	ID	S	Object ID of the format object representing the format of the content for which the link was established.
data_ticket	integer	S	Used internally to fetch and save the content.
other_ticket	integer	S	Used internally to fetch and save the content.

Property	Datatype	Single or repeating	Description
session_count	integer	S	Number of sessions that are accessing the content storage area identified by this link record object.
session_id	ID	R	Session IDs of the sessions accessing the content storage area.

Literal Expr

Purpose

Describes a literal value that is an integer, string, Boolean, ID, or date datatype.

Description

Supertype: Expression
Subtypes: None
Internal name: dm_literal_expr
Object type tag: 53

A literal expr object describes a literal value that is an integer, string, Boolean, ID, or date datatype. Literal expr objects are created and managed by Content Server and cannot be created by users.

Note: A date literal value specified in a literal expr object cannot require a pattern definition.

Properties

The literal expr type has no defined properties. All of its properties are inherited from its supertype, dm_expression.

Load Object Record

Purpose

Records information about objects that have been loaded from a dump file into a repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_load_object_record
Object type tag: 32

A load object record object contains information about one object that has been loaded from a dump file into a repository.

Properties

Table 91, page 257, lists the properties defined for the type.

Table 91. Properties defined for the load object record type

Property	Datatype	Single or repeating	Description
extra_offset	string(20)	S	Position in the dump file at which the server wrote the last entry, if file is 2 GB or greater.
extra_pos	integer	S	Specifies the position in the dump file where you can find additional information about the object. This is expressed as the number of characters from the beginning of the file.
file_offset	string(20)	S	Position in the dump file at which the server wrote the last entry, if the file is less than 2 GB.

Property	Datatype	Single or repeating	Description
is_synonym	Boolean	S	Indicates if the new_id already exists in the new repository. This is used for objects such as content storage areas and partitions, where you do not want to copy the old object into the new repository, but merely use a matching object in the new repository.
load_object	ID	S	Object ID of the load record object that contains the information about the dump file containing the loaded object.
new_id	ID	S	Object ID of the loaded object in the new repository.
old_id	ID	S	Object ID of the loaded object in the old repository.

Load Record

Purpose

Stores information about dump files that are being loaded into a new repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_load_record
Object type tag: 31

A load record object stores information about a dump file that is being loaded into a new repository. You can use the information in a load record object, in conjunction with the information in the associated load object record objects, to restart the loading process if it is interrupted.

Properties

Table 92, page 259, lists the properties defined for the type.

Table 92. Properties defined for the load record type

Property	Datatype	Single or repeating	Description
file_name	string(255)	S	Name of the dump file that you are loading.
load_operation	string(255)	S	Used internally.
load_parameter	string(255)	R	Defines parameters for the load operation. Valid parameters are: preserve_replica=T|F generate_event=T|F generate_event is TRUE by default. Setting it to F (FALSE) turns off notification to the fulltext index user for the save event when an object is loaded into the target repository.

Property	Datatype	Single or repeating	Description
predicate	string(255)	R	Used internally.
r_offset	string(20)	S	Used internally to manage the load operation and objects created during the load.
relocate	Boolean	S	If set to TRUE, this directs the server to assign new object IDs to all the objects in the dump file during the load process. If set to FALSE, this directs the server to use the objects' previous IDs. The default is TRUE.
r_end_time	date	S	Date and tBooleanime at which the load process ended.
r_first_phase	Boolean	S	Indicates whether the load process has completed the first phase.
r_position	integer	S	Indicates how far into the dump file the load process has gone. This value is updated continually as the load process proceeds.
r_start_time	TIME	S	Time at which the load process was started.
type	string(33)	R	Used internally.

Location

Purpose

Records a file system location for a specific file or directory.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_location
Object type tag: 3a

A location object contains a file system location for a specific file or directory. The server uses the information in location objects to find the files and directories that it needs for successful operation.

Properties

Table 93, page 261, lists the properties defined for the type.

Table 93. Properties defined for the location type

Property	Datatype	Single or repeating	Description
file_system_path	string(255)	S	Specifies the actual location of the directory or file represented by the location object. The specification's syntax must be appropriate for the machine on which the server resides. For example, if the server is on a Windows host, the specification must be expressed as a Windows path specification. If on a UNIX host, the specification must be a UNIX path. The directory path and name must consist of ASCII characters.

Property	Datatype	Single or repeating	Description
mount_point _name	string(32)	S	Identifies the mount point underneath which this location resides. Use the name of the mount point object that describes the mount point.
no_validation	Boolean	S	Identifies whether applications create a location object. If set to TRUE, the server does not validate the directory path specified by file_system_path and does not create the directory. The default is FALSE.
object_name	string(255)	S	Name of the location object. The name must be unique among the location objects in a repository. This property is inherited from its SysObject supertype, but is included here because its setting has specfiic requirements for the location type.
path_type	string(16)	S	Indicates whether the location represents a directory or a file. Legal values are directory and file.
security_type	string(32)	S	Defines the security level applied to the directory or file. Legal values are: public_open, public, and private. The default setting is private.

Locator

Purpose

Stores a Documentum Resource Locator (DRL), which points to an object in a repository.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_locator
Object type tag: 08

A locator object stores a Documentum Resource Locator (DRL), which points to an object in a repository. The DRL is stored in the one property defined for the type.

Properties

Table 94, page 263, lists that property.

Table 94. Properties defined for the locator type

Property	Datatype	Single or repeating	Description
resource_locator	string(255)	S	Contains a text string that resolves to an object in a repository.

Media Profile

Purpose

Stores a transformation profile as content.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_media_profile
Object type tag: 08

A media profile object stores a transformation profile as content. Transformation profiles are XML documents that describe the transformations that can be performed on content, to change one format to another. A default set of profiles is provided when a Content Transformation Server product is configured for the repository. Additional profiles can be created by system administrators.

Properties

Table 95, page 264, lists the properties defined for the type.

Table 95. Properties defined for the media profile type

Property	Datatype	Single or repeating	Description
cmd_file_paths	string(255)	R	Used internally by Documentum Content Transformation Services
filter_names	string(32)	R	Identifies the filter used internally, within Content Transformation Services. Valid values are: CTSProduct Public

Property	Datatype	Single or repeating	Description
filter_values	string(64)	R	Specifies whether the transformation is viewable in the WDK transformation wizard. Valid values are: An acronym representing a CTS product System Public
related_objects_only	Boolean	S	Indicates whether a specific transformation can only store the target file as a new object or whether it can store the target as either a new object or as a rendition. T means that the target may only be stored as a new object. F means that the target may be stored as either a new object or a rendition.
source_formats	string(64)	R	Names of the source formats identified in the transformation profile.
src_object_type	string(32)	R	Object types of the sources
target_formats	string(64)	R	Names of the target formats identified in the transformation profile. The format named in any given index position in source_formats can be transformed to the format identified in the corresponding index position in target_formats.

Message Address

Purpose

Records all unique, full email addresses for an email message.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dm_message_address
Object type tag: 00

A message address object records a unique email address found in the header of an email message. A unique address is an address not currently represented in the repository by another message address object. Each time a unique address is found in an archived message, a message address object is created to record the address. Message address objects are primarily used by personal and compliance archiving applications.

Properties

Table 96, page 266, lists the properties defined for the type.

Table 96. Properties defined for the message address type

Property	Datatype	Single or repeating	Description
addr_id	string(16)	S	Hash value of the email address
addr_type	string(2)	S	Identifies the kind of email message. Valid values are: 0, meaning Exchange email message 1, meaning SMTP email message 2, Lotus Notes email message
email_addr	string(1024)	S	Fully qualified email address of the user

Property	Datatype	Single or repeating	Description
friendly_name	string(128)	S	User's name as it appears prepended to the full email address. This name typically appears in double quotes prior to the actual email address in angle brackets.
mail_user	string(128)	S	User's name as it appears in the actual email address.
primary_domain	string(512)	S	Primary domain of the email address
sub_domain	string(512)	S	Subdomains in the email address

Message Archive

Purpose

Stores an email message.

Description

Type category: lightweight type

Supertype: Message Container (shareable parent type)
Subtypes: None
Internal name: dm_message_archive
Object type tag: 09

A message archive object is used to store an email message. The header information in the message is stored in the object's properties and the actual content of the message is stored as content. Message archive objects are primarily used by personal and compliance archiving applications. Beginning in release 6.5, this type became a lightweight type that uses Message Container as its shareable parent type. Some of the type properties were moved to the parent type.

Properties

Table 97, page 268, lists the properties defined for the type.

Table 97. Properties defined for the message archive type

Property	Datatype	Single or repeating	Description
attachment_count	integer	S	Total number of message attachments in the email message
conversation_id	string(32)	S	Reserved for internal use
is_encrypted	string(1)	S	Whether the message is encrypted, and which encryption method. Valid values are: 0, no encryption 1, MIME encryption 2, other type of encryption

Property	Datatype	Single or repeating	Description
message_id	string(42)	S	Email message identifier
message_id_ timestamp	string(8)	S	Reserved for internal use
message_importance	string(1)	S	The level of importance of the email message. Valid values are: 0, normal 1, low 3, medium 5, high
message_link_count	integer	S	Link number in a SysObject chain
message_sensitivity	string(1)	S	Specifies the sensitivity of the message. Valid values are 1 to 255.
message_size	integer	S	Size of the complete email message, including headers and routing information, in bytes
message_subject	string(256)	S	Subject line of the message
parent_message_id	string (24)	S	Object ID of the root message, if this message is an embedded message.
receive_date	Date	S	GMT date and time at which the message was received
sent_date	Date	S	GMT date and time at which themessage was sent.
tracking_id	string(2000)	S	Used internally to track message source

Message Attachment

Purpose

Records names of attachments on email messages.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_message_attachment
Object type tag: 00

A message attachment object records the names of attachments sent with email messages. The attachments represented by this object type are bound to the parent email message by the value in the message_object_id property. Message attachment objects are created when an email message that has an attachment is archived. Message attachment objects are primarily used by personal and compliance archiving applications.

Properties

Table 98, page 270, lists the properties defined for the type.

Table 98. Properties defined for the message attachment type

Property	Datatype	Single or repeating	Description
attachment_name	string(512)	S	Attachment file name
attachment_position	integer	S	Order of the attachment entry
format_name	string(64)	S	Name of the attachment's format
message_object_id	string(16)	S	Object ID of the message archive object representing the email message to which this attachment was attached

Message Container

Purpose

Stores multiple email messages.

Description

Type category: shareable type

Supertype: Document
Subtypes: Message Archive (lightweight)
Internal name: dm_message_container
Object type tag: 09

A message container object is used as the shareable parent of message archive objects that store email messages. Common information for groups of messages is stored in the object's properties; the information for individual messages is stored in individual message archive objects, and the actual content of the messages is stored as content. Message container objects are primarily used by personal and compliance archiving applications.

Properties

Table 99, page 271, lists the properties defined for the type.

Table 99. Properties defined for the message container type

Property	Datatype	Single or repeating	Description
datamodel_version	string(6)	S	Used internally to manage data model changes
is_archive_message	string(1)	S	Identifies whether the message is an "archive" message. Valid values are: 0, meaning the message is not an archive message 1, meaning the message is an archive message, and the message_id is deterministic. Archive messages are immutable. Non-archive messages may be edited, and the

Property	Datatype	Single or repeating	Description
			message_id is set internally and is unique to this message.
message_class	string(1)	S	Identifies the class of message represented by the mail message object. Valid values are: 0, meaning email message 1, meaning contact 2, meaning appointment
message_doctype	string(1)	S	Identifies the kind of email message. Valid values are: 0, meaning MAPI mail message 1, meaning SMTP message 2, Lotus Notes email message
retention_start_date	Date	S	GMT date and time for retention start date

Message Route User Data

Purpose

Stores route-specific user information for a particular email message.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_message_route_user_data
Object type tag: 00

A message route user data object records route-specific user information for an email message. Message route user data objects are primarily used by personal and compliance archiving applications.

Properties

Table 100, page 273, lists the properties defined for the object type.

Table 100. Properties defined for the message route user data type

Property	Datatype	Single or repeating	Description
hash_id	string(16)	S	Hash value generated for the user_data and user_id pairs for a given route
i_partition	integer	S	Identifies the partition where the object resides
user_data	string(256)	R	Key name
user_tag	string(256)	R	Key value The value at a particular index position is associated with the key name at the corresponding index position in user_data.

Message User Data

Purpose

Records application-specific information about a user referenced by an archived message.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_message_user_data
Object type tag: 00

A message user data object records application-specific information about a user referenced in an archived message. Message user data objects are created when an email message is archived. The objects are primarily used by personal and compliance archiving applications.

Properties

Table 101, page 274, lists the properties defined for the type.

Table 101. Properties defined for the message user data type

Property	Datatype	Single or repeating	Description
message_object_id	string(16)	S	Object ID of the message archive object that contained the email address.
user_data	string(2000)	S	Application-specific information about the user identified in user_tag.
user_tag	string(64)	S	Tag name assigned to the user by the application

Metamodel

Purpose

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_metamodel
Object type tag: 08

A metamodel object records information about a Smart Container model. The content of a metamodel object is the XML representation the metamodel. These objects are the building blocks for Smart Container, and must never be deleted from a repository. Each metamodel object has an associated dm_logical_version aspect. Each also has an aspect named dm_metamodel_*version* where *version* is the version number of the metamodel contained as content.

This object type is installed with the Smart Container DAR file during repository configuration.

Properties

There are no properties defined for the type. It inherits all properties from its supertype.

Method

Purpose

Represents an external procedure that is invoked through the DO_METHOD function using either the DQL EXECUTE statement or the Apply method.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_method
Object type tag: 10

A method object represents an external procedure that is invoked through the DO_METHOD function using either the DQL EXECUTE statement or the Apply method.

Properties

Table 102, page 276, lists the properties defined for the type.

Table 102. Properties defined for the method type

Property	Datatype	Single or repeating	Description
is_restartable	Boolean	S	Reserved for internal use.
launch_async	Boolean	S	Indicates whether the procedure is to run asynchronously or not. If this is set to TRUE and the method is launched on the application server, setting SAVE_RESPONSE on to TRUE on the command line is ignored. This property setting is ignored if the method is launched on the method server or Content Server and SAVE_RESULTS is set to TRUE on the command line. The method is always launched synchronously.

Property	Datatype	Single or repeating	Description
launch_direct	Boolean	S	Used by Content Server. This is ignored when the method executed on the application server or method server. Controls whether the program is executed by the operating system's system or exec API call. If set to TRUE, the server uses the exec call to execute the procedure. In such cases, the method_verb must be a fully qualified path name. If set to FALSE, the server uses the system call to execute the procedure. **Note:** If SAVE_RESULTS is set to TRUE on the command line, the server ignores the setting of launch_direct and always uses the system call to execute the procedure.
method_args	string(255)	R	List of command line arguments for the procedure. Not currently used.
method_type	string(32)	S	If you are executing the method using Content Server or the method server and the executable is stored as content for the method, then setting this to dmawk or dmbasic, directs the server to add -f in front of the file name and to pass all arguments specified on the DO_METHOD command line to the program. If you are executing the method on an application server, set this to java.

Property	Datatype	Single or repeating	Description

method_verb	string(255)	S	Command-line name of the procedure or the name of the interpretive language that will execute the program file.
			
			Use dmbasic if the program is written in Docbasic.
			
			On Windows, use dmawk32 if the program is written in dmawk. On UNIX, use dmawk if the program is written in dmawk.
			
			Use the namespace in which to launch the method if the method is executing on the application server. For example:
			
			`com.foo.<package_name>.<class>`
			
			or
			
			`fooPayMethod`

run_as_server	Boolean	S	If set to TRUE, it indicates that you want the method to run as the server account. The default is FALSE. run_as_server must be TRUE to execute a method on the method server or application server.
success_return_codes	integer	R	Valid values for the a_last_return_code property of the job. If this is set, the a_last_return_code property must be set to one of these values or the dm_run_dependent_jobs method assumes that the job failed. The property is set to 0 for replication jobs.

Property	Datatype	Single or repeating	Description
			The property is ignored if it is not set.
success_status	string(255)	S	Defines the valid value for a_current_status in the completed job. If set, the job's a_current_status property value must match the success_status value after the job completes. If it does not, the dm_run_dependent_jobs method considers the job to have failed. For replication jobs, the value is set to "Replicate Operation Complete." The property is ignored if it is not set.
timeout_default	integer	S	The default time-out for this procedure. Used if no other time-out is specified on the command line. The default is 60 seconds. This value cannot be greater than the value of timeout_max.
timeout_max	integer	S	The maximum time-out that you can specify on the command line for this procedure. The default is 300 seconds.
timeout_min	integer	S	The minimum time-out that you can specify on the command line for this procedure. The default is 30 seconds. This value cannot be greater than the value of timeout_default.

Property	Datatype	Single or repeating	Description
trace_launch	Boolean	S	Controls whether internal trace messages generated by the executing program are logged. If set to TRUE, the messages are put in the session log. If set to FALSE, the messages are not logged. This is FALSE by default. **Notes:** If trace_launch is TRUE and SAVE_RESULT is set to TRUE on the command line, the trace output is put into the result document. If the method is executing on the application server, the trace records information about the method launch. Internal trace messages are stored in a result document if SAVE_RESULTS is set to TRUE.
use_method _content	Boolean	S	Indicates whether the program that you want to run is stored as content in the method object or as an external file. Set this to TRUE if you have specified an interpretive language for method_verb and the program you want to run is stored as the method's content. Set this to FALSE if method_verb contains a full file path to a program or script. The value is ignored if the method is executing on an application server.
use_method _server	Boolean	S	Whether to use the Method Server or Application Server

Property	Datatype	Single or repeating	Description
			to execute Dmbasic or Java methods. TRUE means to use the Method Server or application server. FALSE means to use the Content Server. If this is TRUE, the value is used in conjunction with the value of method_type to determine which server to use. Setting this property requires Superuser privileges.

Module

Purpose

Represents a business object module.

Description

Supertype: Folder
Subtypes: Aspect Type, Validation Module
Internal name: dmc_module
Object type tag: 0b

A module object stores information about one business object module. The name of the module is recorded in the object_name property of the module object. The name must be unique. Module objects are stored in the repository in /System/Modules/*module_type*, where module_type is the value also found in the a_module_type property of the module.

The object type is installed by a script when Content Server is installed. Instances of the type are created and managed through Documentum Application Builder.

Properties

Table 103, page 282, lists the properties defined for the type.

Table 103. Properties defined for the module type

Property	Datatype	Single or repeating	Description
a_bof_version	string(32)	S	Version level of the BOF module. This is defined by the module's creator.
a_interfaces	string(255)	R	Fully qualified class names of the interfaces implemented by the BOF module
a_is_privileged	Boolean	S	Indicates whether the module is a privileged module. T means that the module is privileged. F means that the module is not a privileged module.

Property	Datatype	Single or repeating	Description
a_module_type	string(32)	S	Identifies the type of the module. For user-defined modules, the a_module_type value is also user defined. Modules provided by EMC Documentum typically have one of the following values: TBO SBO Aspect **Note:** Module objects that have Aspect as the module type are created and used internally only.
a_privilege_roles	string(32)	R	List of the roles that can use this module.
a_req_module_ interfaces	string(255)	R	Object IDs of the dmc_jar objects associated with the modules identified in req_module_names. The IDs at a particular index position are associated with the module at the corresponding index position in req_module_names. Multiple object IDs at one index position are separated by commas.
contact_info	string(256)	S	Contact information about a party who can provide information about this module. This value is user-defined.
implementation_ technology	string(32)	S	Identifies the technology used to create the module.

Property	Datatype	Single or repeating	Description
min_dfc_version	string(32)	S	Identifies the minimum DFC version level required to implement this module. The minimum value is 5.3.
module_description	string(255)	S	User-defined description of the module's functionality
primary_class	string(255)	S	The main implementation class. For .NET implementations, the value may consist of multiple comma-separated values.
req_module_names	string(64)	R	Names of modules on which this module depends

Module Config

Purpose

Identifies a business object module to the workflow Timer servlet.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_module_config
Object type tag: 00

A module config object identifies a business object module to a workflow Timer servlet and stores argument values used by the servlet. The object type is installed using a script when Content Server is installed.

Properites

Table 104, page 285, lists the properties defined for the type.

Table 104. Properties defined for the module config type

Property	Datatype	Single or repeating	Description
id_attr_name	string(40)	R	Names of properties having object ID values to be used by the servlet.
id_attr_value	ID	R	Object IDs stored in the properties in the corresponding index position in id_attr_name.
r_aspect_name	string(64)	R	List of BOF module names

Property	Datatype	Single or repeating	Description
string_attr_name	string(40)	R	Names of properties having string values to be used by the servlet
string_attr_value	string(2000)	R	Values of the properties identified in string_attr_name. The value in a particular index position is the vaue of the property identified in the corresponding index position in string_attr_name.

Mount Point

Purpose

A mount point object describes a mounted directory in a Content Server installation.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_mount_point
Object type tag: 3e

A mount point object describes a mounted directory in a Content Server installation.

Properties

Table 105, page 287, lists the properties defined for the type.

Table 105. Properties defined for the mount point type

Property	Datatype	Single or repeating	Description
file_system_path	string(255)	S	Specifies the full path location of the mount point. The specification's syntax must be appropriate for the machine on which the location resides. For example, if the location is on a Windows host, the specification must be a Windows path specification.
host_name	string(128)	S	Identifies the machine on which this location resides. Use the machine's host name.
mac_preferred _alias	string(32)	S	Defines the alias that you prefer the system to use when a Macintosh client references the mount point. For example, a valid value might be Documentum.

Property	Datatype	Single or repeating	Description
object_name	string(255)	S	Names the mount point object. This property is actually inherited from its sysobject supertype. However, it is included here because you are required to set this property when you create a mount point object.
security_type	string(32)	S	Specifies the security level applied to this directory location. Legal values are: public_open, public, and private. Defaults to private if not explicitly set.
unix_preferred _alias	string(32)	S	Defines the alias that you prefer the system to use when the client references the mount point. When the server is a UNIX server, you are not required to set this property.
win_preferred _alias	string(32)	S	Defines the alias that you prefer the system to use when a Windows client references the mount point. For example, valid values might be t:\ or k:\.

Network Location Map

Purpose

Records information about network locations of client hosts.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_network_location_map
Object type tag: 08

A network location object describes the network location of client sites. Instances of the object type are used to determine a user's location relative to the Content Servers that might provide them with content.

Properties

Table 106, page 289, lists the properties defined for the type.

Table 106. Properties defined for the network location map type

Property	Datatype	Single or repeating	Description
begin_near_ip_address	string(80)	R	The beginning of the IP address range representing IP addresses near to this network location Each index position represents the beginning of one range. The end of that particular range is recorded in the corresponding index position in end_near_ip_address.
default_netloc	Boolean	S	Whether this network location can serve as a default network location for a client, to be used when no other mapping for a user's IP address exists.

Property	Datatype	Single or repeating	Description
end_near_ip_address	string(80)	R	The end of the IP address range representing IP addresses near to this network location Each index position represents the end of one range. The beginning of that particular range is recorded in the corresponding index position in begin_near_ip_address.
ip_address_range	string(240)	R	Defines an IPv6 address range for the network location. You can use either the prefix method or the IPv6address1 — IPv6address2 method to indicate the range. For example, the following two examples represent the same IPv6 range: 2001:0db8:1234::/48 2001:0db8:1234::-2001:0db8:1234:ffff:ffff:ffff:ffff:ffff
netloc_display_name	string(80)	S	User-defined display name or description of this network location.
netloc_ident	string(80)	S	User-defined unique identifier for this network location.

NLS DD Info

Purpose

Records locale-dependent data dictionary information for an object type or property.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_nls_dd_info
Object type tag: 4f

An nls dd info object holds locale-dependent data dictionary information for an object type or property. NLS DD info objects are created and managed by the server. Properties in nls dd info objects are set when users add or change data dictionary information. Each nls dd info object contains the data dictionary information specific to one locale.

Not all nls dd info properties apply to both object types and properties. Some apply only to properties.

Properties

Table 107, page 291, lists the properties that are applicable to both object types and properties. Table 108, page 292, lists the nls dd info properties that are applicable only to properties.

Table 107. Properties applicable to object type and properties

Property	Datatype	Single or repeating	Description
comment_text	string(255)	S	User defined.
foreign_key_msgs	string(255)	R	The error message for foreign key constraint violations.
help_text	string(255)	S	User-defined help text for the property or type.
label_text	string(64)	S	User-defined label for the property or type.
map_data_string	string(128)	R	List of possible data values for the property.
map_description	string(255)	R	Descriptions of the data values at the corresponding index levels in map_data_string.

Property	Datatype	Single or repeating	Description
map_display _string	string(128)	R	The character string to display for the data value at the corresponding index level in map_data_string.
primary_key_msg	string(255)	S	The error message for primary key constraint violations.
unique_key_msgs	string(255)	R	The error message for unique key constraint violations. The message at any one index level is displayed when the constraint at the corresponding index level is violated.
val_constraint _msg	string(255)	R	The error message for value constraint violations. The message at any one index level is displayed when the constraint at the corresponding index level is violated.

Table 108. Properties applicable only to properties

Property	Datatype	Single or repeating	Description
category_name	string(64)	S	User defined.
def_value_length	integer	S	Length, in characters, of a typical value for the property. Zero indicates an unspecified (NULL) length.
format_pattern	string(64)	S	For date properties, the pattern to use for date interpretation.
format_pattern _tag	integer	S	Currently unused. (Currently, this is always 1.)
map_data_string	string(128)	R	List of possible data values for the property.
map_description	string(255)	R	Descriptions of the data values at the corresponding index levels in map_data_string.

Property	Datatype	Single or repeating	Description
map_display _string	string(128)	R	The character string to display for the data value at the corresponding index level in map_data_string.
not_null_msg	string(255)	S	The error message for NOT NULL constraint violations.
parent_id	ID	S	Object ID of the aggr domain object that references the nls dd info object.

Note

Purpose

Represents an annotation for a document or other SysObject.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_note
Object type tag: 41

A note object represents an annotation for a document or other SysObject. The content of the annotation is stored in a content file associated with the note object.

Properties

Note objects inherit all of their properties from their supertype, SysObject.

Other File

Purpose

Contains information about resource fork files for Macintosh.

Description

Supertype: Document
Subtypes: None
Internal name: dmi_otherfile
Object type tag: 23

An other file object contains information about resource fork files for Macintosh.

Properties

Table 109, page 295, lists the properties defined for the type.

Table 109. Properties defined for the other file type

Property	Datatype	Single or repeating	Description
data_ticket	integer	S	Value that tells the server how to access the file.
i_partition	integer	S	Identifies the partition where the object resides
protocol	integer	S	Value indicating the Macintosh access protocol

Output Device

Purpose

Contains a description of a printer that is accessible to users of the repository.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_outputdevice
Object type tag: 17

An output device object contains a description of a printer that is accessible to users of the repository.

Properties

Table 110, page 296, lists the properties defined for the type.

Table 110. Properties defined for the output device type

Property	Datatype	Single or repeating	Description
image_format	string(64)	S	Specifies the format that the printer uses to print images
os_name	string(32)	S	Name of the printer as defined in the /etc/printcap file.
output_command	string(128)	S	Contains the Unix print command sent to the printer
output_formats	string(128)	R	Contains the formats supported by the device.
output_mask	string(64)	S	Contains the job number of a queued print job, in a platform-dependent mask.
remove_command	string(128)	S	Contains the command that the server sends to the printer to cancel a print job
status_command	string(128)	S	Contains the command that the server sends when the user issues a Unix lpq command

Package

Purpose

Serves as a container for one or more documents (or other SysObjects) moving through a workflow.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_package
Object type tag: 49

A package object is used within workflows to deliver one or more documents (or other SysObjects) from the output port of a source activity to the input port of a destination activity. Packages are generated by the server from the properties of the activity definition. Users cannot create or modify package objects.

Properties

Table 111, page 297, lists the properties defined for the type.

Table 111. Properties defined for the package type

Property	Datatype	Single or repeating	Description
i_acceptance_date	date	S	Records the date and time when a package arrived and was accepted by an input port.
i_package_oprtn	string(64)	S	Contains a value copied from the package definition upon creation of a package.
i_package_order	integer	S	Represents the order of where its associated package definition appears in the containing port definition.
r_act_seqno	integer	S	Contains the sequence number (r_act_seqno) of the activity in which the package is being handled.

Property	Datatype	Single or repeating	Description
r_component _chron_id	ID	R	Chronicle ID of the object identified at the corresponding index position in r_component_id.
r_component_id	ID	R	Object IDs of the bound objects.
r_component_name	string(80)	R	Object names of the bound objects. The name at a particular index position is the name of the object identified in r_component_id at the same index position. Whether this property is set is configurable. It is not set by default. Refer to *Content Server Fundamentals* for more information.
r_note_flag	integer	R	Identifies when a note was added to the package and how to handle the note in activity transitions. Valid values are: 0, Note added prior to activity; discard at next transition 1, Note added prior to activity; retain in next transition 2, Note added during this activity; send to next activity and then discard during transition fromthat activity 3, Note added during this activity; retain in subsequent transitions
r_note_id	ID	R	Contains the ID of a dm_note object.
r_note_writer	string(32)	R	Contains the name of the user who made notes.

Property	Datatype	Single or repeating	Description
r_package_flag	integer	S	Indicates whether the package is visible to the activity and whether it is allowed to be empty. The value is copied from the r_package_flag property in the activity object. Valid values are: 0, meaning the package is invisible but cannot be empty 1, meaning the package is visible and cannot be empty 2, meaning the package is invisible but may be empty 3, meaning the package is visible and may be empty
r_package_label	string(32)	R	Contains a version label, if any.
r_package_name	string(16)	S	Contains the package name.
r_package_type	string(40)	S	Contains the real type name of the bound components.
r_port_name	string(16)	S	Contains the name (port_name) of the port of the activity run-time instance in which the package is being handled.
r_workflow_id	ID	S	Contains the object ID of the containing workflow.

Plugin

Purpose

A plugin object represents a plugin library used by Content Server.

Description

Supertype: Document
Subtypes: None
Internal name: dm_plugin
Object type tag: 67

Content Server uses plug-in libraries to access external storage areas, including content-addressed storage, retention-enabled storage, and for some forms of user authentication.

Properties

Table 112, page 300, lists the properties defined for the type.

Table 112. Properties defined for the plugin type

Property	Datatype	Single or repeating	Description
a_hw_platform	string(255)	R	Hardware architecture on which the plugin is running. The value is user defined. For example, a value might be Intel x86 or Sun SPARC Power PC
a_op_sys	string(255)	R	Identifies the operating system on which the plugin is running.

Property	Datatype	Single or repeating	Description
a_usage	string(255)	S	Description of the plugin's use.
a_plugin_type	string(255)	S	Indicates the type of plugin. The value is user-defined. Some example values are: batch file Bourne shell script C-shell script DLL docbasic executable java java script sl so

Policy

Purpose

A policy object contains the definition of a lifecycle.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_policy
Object type tag: 46

Policy objects record lifecycle definitions.

Properties

Table 113, page 302, lists the properties defined for the type.

Table 113. Properties defined for the policy type

Property	Datatype	Single or repeating	Description
action_object_id	ID	R	Object ID of the dm_procedure that implements state actions. All properties specified in the procedure must be defined in the primary type. This is unused if java_methods is T.
alias_set_ids	ID	R	List of the object IDs of the alias set objects representing acceptable alias sets for the lifecycle.
allow_attach	Boolean	R	Indicates whether a document can be attached to this state.The default value is false.
allow_demote	Boolean	R	Indicates whether the object can be demoted from this state. The default value is False.

Property	Datatype	Single or repeating	Description
allow_schedule	Boolean	R	Indicates whether a state transition event (such as promote, demote, suspend, or resume event) can be scheduled when this state is the current state of the sysobject. The default value is true.
app_validation_id	ID	S	Identifies the procedure object that validates the lifecycle. If app_validation_ver is set, then app_validation_id contains the chronicle ID of the produre object. This is unused if java_methods is T.
app_validation_ver	string(32)	S	Version of the object identified in app_validation_id. This is unused if java_methods is T.
entry_criteria_id	ID	R	Object ID of the dm_func_expr that implements the entry criteria. All properties specified in the expression must be defined in the primary type of this policy object. The source code for all expressions defined for this policy object are stored on page 0 of the policy's content, and all p-code is stored on page 1.
exception_state	string(32)	R	Name of the exception state, if any. Optional for normal policy states. Ignored for exception states.
extension_type	string(32)	S	Name of the extension type used for the lifecycle.

Property	Datatype	Single or repeating	Description
i_state_no	integer	R	A 0-base internal integer number which uniquely identifies a state. Every time a new state is added, the highest number in the state chain plus one is used. This value, not the state name, is used in sysobject to record current state and resume state.
include_subtypes	Boolean	R	Indicates whether all subtypes of the type specified in the corresponding position of included_type are included. Possible values are: • false (all subtypes are NOT included) • true (all subtypes are included) The property is always true when the object is created.
included_type	string(27)	R	The acceptable object types. The first position stores the primary type associated with this policy object. The later positions are valid subtypes of the primary type. At least one value is required.
java_methods	Boolean	S	Whether the entry criteria, actions, and validation methods are implemented in Java. T means they are written in Java. F means they are written in Docbasic.

Property	Datatype	Single or repeating	Description
r_definition_state	integer	S	Status of the policy object. Possible values are: 0, meaning draft 1, meaning validated 2, meaning installed The property is always 0 when the object is created.
return_condition	integer	R	Used when return_to_base is TRUE to identify which operations return an object to the policy's base state. Valid values are: • 0, return to base state on Checkin or Save • 1, return to base state on checkin • 2, return to base state on Save • 4, return to base state on Saveasnew • 8, return to base state on Branch The default value is 0. To identify multiple operations, sum the integers. For example, if you want to return to the base state on Checkin or Saveasnew, set the property to 5.
return_to_base	Boolean	R	TRUE returns an object to the base state when the operations identified in the corresponding index position in return_condition are performed. The setting in each index position applies to the state in the corresponding index position in state_name. For a Checkin or Branch method, the affected object is the new

Property	Datatype	Single or repeating	Description
			version. For a Saveasnew, the affected object is the new copy. For a Save, the affected object is the saved object. The base state's entry criteria and actions are applied to the object when it is returned to base. If the entry criteria are not met, the method fails. If the actions fail, the object remains in the base state. FALSE means to leave the object in the current state. The default value is FALSE.
state_class	integer	R	Indicates whether the state is a normal state (with value 0), or an exception state (with value 1). The default value is 0.
state_description	string(128)	R	Purpose of the state.
state_name	string(32)	R	Name of the state. Unique within the policy definition to which the state belongs. A state name must start with a letter and cannot contain colons, periods, or commas.
state_type	string(32)	R	Name of the state type describing the state at the corresponding index position in state_name.
system_actions	ID	R	Object ID of tcf activity object describing a sequence of actions through modules implementing the IDfLifecycleAction interface This is only set if java_methods is T.

Property	Datatype	Single or repeating	Description
type_override_id	ID	R	Object ID of dm_aggr_domain which contains the type override information prepared by data dictionary. This property controls properties of the document (whether properties are read-only, or hidden, and so on) when the document is in this state.
user_action_id	ID	R	Object ID of the dm_procedure that implements user-defined actions, if the user_action_ver is unspecified; chronicle ID, otherwise. All properties specified in the procedure must be defined in the primary type. This is unused if java_methods is T.
user_action_service	string(128)	R	Names of the module objects implementing the user actions This is only set if java_methods is T.
user_action_ver	string(32)	R	The version label of user_action_id. If unspecified, the user_action_id is the id of the dm_procedure that contains user-defined actions. Otherwise, the user_action_id is the chronicle id. Together with user_action_id, this property allows users to associate a late bound user action object with a lifecycle. This is unused if java_methods is T.

Property	Datatype	Single or repeating	Description
user_criteria_id	ID	R	Object ID of the dm_procedure that implements the user-defined entry criteria for the state identified at the corresponding index position in state_name. All properties specified in the procedure must be defined in the primary type. This is unused if java_methods is T.
user_criteria_service	string(128)	R	Names of the module objects implementing the entry criteria This is only set if java_methods is T.
user_criteria_ver	string(32)	R	The version label of user_criteria_id. If specified, the specified version of the procedure identied in user_criteria_id is used. If a version lable is not specified, the version identified in user_criteria_id is used. Together with user_criteria_id, this property allows users to late bind an entry procedure with a lifecycle. This is unused if java_methods is T.
user_postproc_id	ID	R	Object ID of the dm_procedure that implements the user-defined procedure executed after entry into the state identified in the corresponding index position in state_name. This property may be used in conjunction with user_postproc_ver, to late-bind a particular version of the procedure to the lifecycle

Property	Datatype	Single or repeating	Description
			state. For information about late-binding, refer to *Content Server Fundamentals*. This is unused if java_methods is T.
user_postprocessing_service	string(128)	R	Names of the module objects implementing the post-processing actions This is only set if java_methods is T.
user_postproc_ver	string(32)	R	Version labels identifying which versions of the procedures specified in user_postproc_id are to be executed. The label at a particular index position is applied to the procedure identified at the corresponding index position in user_postproc_id. Setting this property for a particular procedure allows you to late-bind a particular version of a procedure to the lifecycle state. For more information, refer to *Content Server Fundamentals*. This is unused if java_methods is T.
user_validation_service	string(128)	R	Names of the module objects implementing user validation This is only set if java_methods is T.

Preset Info

Purpose

Relates a preset to a folder.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_preset_info
Object type tag: 37

A preset info object relates a preset to a folder and records information about how and when the preset should be updated when changes occur to the folder. This object type is installed with the Preset DAR file during repository configuration.

Properties

Table 114, page 310, lists the properties defined for the type.

Table 114. Properties defined for the preset info type

Property	Datatype	Single or repeating	Description
copy_source_id	ID	S	If the related folder was created by a copy operation, this property records the object ID of the source folder
is_dirty	Boolean	S	Indicates that a change was made to the folder that requires a change to the preset of the folder
preset_package_id	ID	S	Object ID of the preset for the folder
preset_repository_path	string(255)	S	Specifies the location of the presets for this folder

Preset Package

Purpose

Description

Supertype: Document
Subtypes: None
Internal name: dmc_preset_package
Object type tag: 09

A preset package object records the preset configurations for a particular scope combination. This object type is installed with the Preset DAR file during repository configuration.

Properties

Table 115, page 311, lists the properties defined for the type.

Table 115. Properties defined for the preset package type

Property	Datatype	Single or repeating	Description
last_modifier	string(32)	S	User name of the user who last modified the object. This is the name of the actual user who initiated the change through the PresetService.
last_modifier_ repository	string(32)	S	Name of the repository to which the user was connected when the preset was modified
locked_by	string(32)	S	If a preset is checked out, this property records the name of the user who initiated the change to the preset through PresetService.
scope_key	string(32)	S	Key that represents the unique scope combination to which the preset configuration applies

Procedure

Purpose

Procedure objects store Docbasic procedures that extend the behavior of the Documentum clients.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_procedure
Object type tag: 08

Docbasic is an interpretive language that you can use to write applications to extend or customize the behavior of Documentum clients or Content Server. Depending on where they are stored in the repository, procedures can be executed automatically, when a user connects to a repository, or on demand, when users select a menu item. Users must have at least Read permission on the procedure object to execute it.

Properties

Table 116, page 312, lists the property defined for the type.

Table 116. Properties defined for the procedure type

Property	Datatype	Single or repeating	Description
user_runnable	Boolean	S	Indicates whether the procedure is ready for use.

Process

Purpose

A process object contains the definition of a workflow process.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_process
Object type tag: 4b

A process object is created when a user creates a workflow definition in either Workflow Manager or Process Builder. There are three inherited properties that are reserved for internal use for process objects: a_special_app, a_status, and a_application_type.

Properties

Table 117, page 313, describes the properties of the process type.

Table 117. Properties defined for the process type

Property	Datatype	Single or repeating	Description
act_choose_by	string(128)	R	Identifies by name the activity whose performer will choose the performer for the activity in the corresponding index position in act_choose_for.
act_choose_for	string(128)	R	Identifies by name the activity whose performer will be chosen by the performer of the activity named at the corresponding index position in act_choose_by.
act_choose_name	string(128)	R	A user-defined name that links the performers chosen by the performer of the activity in the corresponding index position in act_choose_by to other activities.

Property	Datatype	Single or repeating	Description
act_group_id	ID	R	Contains the object IDs of the tactivity groups (dm_activity objects) defined for this process
act_group_name	string(128)	R	Contains the activity group names The name at each index position corresponds to the activity group identified at the corresponding index position in act_group_id.
act_performer_from	string(128)	R	Name of the activity from which the performer or performers of the activity in act_performer_to are selected.
act_performer_rule	integer	R	Indicates whether the activity identified in act_performer_to is performed by the last performer or all performers of the activity identified in act_performer_from. Valid values are: 0, meaning the activity named in act_performer_to is performed by the last performer of the activity identified in act_performer_from. 1, meaning the activity named in act_performer_to is performed by all the performers of the activity identified in act_performer_from.
act_performer_to	string(128)	R	Name of the activity whose performers are selected according to the rule defined in act_performer_rule.
calendar_id	ID	S	reserved for future use
current_environment	string(64)	S	Contains the deployment environment that is currently in use. The value of current_environment must be among the values defined in the deploy_environment attribute.

Property	Datatype	Single or repeating	Description
deploy_environment	string(64)	R	Contains the deployment environments defined for this process.
execution_flag	integer	S	reserved for future use
is_private	Boolean	S	Indicates whether the definition is private or public. The default is private.
package_control	integer	S	Controls whether Content Server sets the r_component_name property of a package object when the component name is identified in the Addpackage method. Valid values are: • 0, to allow the server to set the r_component_name property to the component name • 1, to require the server to set the property to a blank The default value is 0. The value in package_control is only effective if the docbase config property named wf_package_control_enabled is set to T.
parameter_name	string(255)	R	Contains the parameter names. The name at each index position corresponds to the parameter identified at the corresponding index position in parameter_id.
parameter_id	ID	R	Contains the object IDs of the parameters (dmc_process_ parameter objects) defined for this process.

Property	Datatype	Single or repeating	Description
perf_alias_set_id	ID	S	Contains the object ID of the alias set object used to resolve performer aliases when a workflow based on this process definition is created.
r_act_def_id	ID	R	Contains the object IDs of the activities (dm_activity objects) defined for this process.
r_act_name	string(128)	R	Contains the activity names. The name at each index position corresponds to the activity identified at the corresponding index position in r_act_def_id.
r_act_priority	integer	R	Indicates the importance of the activity compared to other activities in this process. The priority at each index position is assigned to the activity defined in r_act_def_id at the corresponding index position. The default value is 0.
r_act_type	integer	R	Indicates the type of the activity identified at the corresponding index position in r_act_def_id. Valid values are: 0, for step (default) 1, for begin 2, for end 3, for exception 4, for initiate
r_definition_state	integer	S	Indicates the state of the process definition. Valid values are: 0, for draft 1, for validated 2, for installed
r_link_dest_act	string(128)	R	Identifies the destination activity for the link.

Property	Datatype	Single or repeating	Description
r_link_dest_port	string(16)	R	Indicates the input port of the destination activity.
r_link_name	string(16)	R	Uniquely identifies the link.
r_link_src_act	string(128)	R	Identifies the source activity for the link.
r_link_src_port	string(16)	R	Indicates the output port of the source activity.
sd_element_acl	ID	R	Object IDs of the ACLs associated with the structured data element objects for this process. The ACL identified at each index position is associated with the structured data element instance identified at the corresponding index position in sd_element_name.
sd_element_default_acl	ID	S	The default ACL used for an structured data element when there is no ACL specified in sd_element_id for the element.
sd_element_default_value	string(2000)	R	Default value for simple structured date element types. The value in this property at a particular index position corresponds the type identified in the corresponding position in the sd_element_type property. If the type is a simple type but is not a string datatype, the value is converted to a string before being stored in this property. If the type is not a simple type, the value in this property is an empty string.

Property	Datatype	Single or repeating	Description
sd_element_name	string(255)	R	Names of the structured data elements. Each name must be unique among the names of other structured data elements and packages within the process.
sd_element_options	integer	R	Indicates whether to remove the runtime instance of the structured data element when the workflow is finished. Valid values are: 0, remove the structured data element instance (default) 1, keep the instance The value at each index position is associated with the structured data element instance identified at the corresponding index position in sd_element_name.
sd_element_parent_id	ID	R	List of the object IDs of the wfsd element parent objects that serve as the parents of structured data elements for this process. The object ID at each index position is the parent for the structured data element at the corresponding index position in workflow definition.
sd_element_type	ID	R	Object IDs of the wfsd type info objects that identify the datatypes of the structured data element. The ID at each index position is associated with the structured data element instance identified at

Property	Datatype	Single or repeating	Description
			the corresponding index position in sd_element_name.
target_namespace_uri	string(2000)	S	Contains the target namespace uri used by the inbound web-service activities contained in the process.

Process Parameter

Purpose

Identifies a parameter object of a process definition.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_process_parameter
Object type tag: 08

The process parameter object contains the parameters associated with process objects (dm_process).

Properties

Table 118, page 320, describes the properties of the process type.

Table 118. Properties defined for the process parameter type

Property	Datatype	Single or repeating	Description
boolean_value	Boolean	S	Value of the parameter if the data type identified in parameter_type is boolean
date_value	Date	S	Value of the parameter if the data type identified in parameter_type is date
deploy_environment	string(64)	S	Deployment environments of the parameter
double_value	double	S	Value of the parameter if the data type identified in parameter_type is double
int_value	integer	S	Value of the parameter if the data type identified in parameter_type is int
parameter_description	string(255)	S	Description of the parameter
parameter_name	string(255)	S	Name of the parameter

Property	Datatype	Single or repeating	Description
parameter_type	integer	S	Data type of the parameter Valid values are: 0: Boolean 1: Int 2: String 3: ID (not used) 4: Date 5: Double
process_id	ID	S	ID of the process (dm_process object) for which the parameter is defined
string_value	string(2000)	S	Value of the parameter if the data type identified in parameter_type is string The value at each index position corresponds to the deploy environment identified at the corresponding index position in deploy_environment. This is not set unless the datatype is string.

Process Correlation Set

Purpose

Stores information about a correlation set.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_process_correlation_set
Object type tag: 37

A process correlation set object records information about a correlation set defined for a workflow definition. A correlation set defines what data will be used to communicate between the Process engine and external applications at runtime. The object type is installed by a script when Content Server is installed.

Properties

Table 119, page 322, lists the properties defined for the type.

Table 119. Properties defined for the process correlation set type

Property	Datatype	Single or repeating	Description
correlation_set_alias	string(255)	R	Alias names used for this correlation set. The name can be either a valid package name or a structured data element defined for the process.
correlation_set_attr	string(2000)	R	Names of the properties referenced in the correlation set. The property specified at a particular index position must be referenced by the correlation set identified in the corresponding index position in correlation_set_alias.

Public Key Certificate

Purpose

Used to decrypt instructions encrypted using a private cryptographic key.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_public_key_certificate
Object type tag: 08

A public key certificate object stores a public key certificate. A public key certificate is used by ACS and BOCS servers to decrypt the instructions sent by Content Server in response to a request for content from an ACS or BOCS server. A public key certificate is also used to authenticate DFC instances.

Public key certificate instances are created automatically by Content Server. Public key certificate objects are stored in the /System/Cryptographic/Keys folder, and are protected by the dm_acl_superusers ACL.



Caution: Public key certificate objects are for internal use only. Do not modify, remove, or add these objects.

Properties

Table 120, page 323, lists the properties defined for the type.

Table 120. Atributes defined for the public key certifcate type

Property	Datatype	Single or repeating	Description
certificate_value	string(1900)	S	Base 64-encoded DER certificate
key_identifier	string(40)	S	Base 64-encoded SHA1 digest of the DER public key

Property	Datatype	Single or repeating	Description
key_type	integer	S	Identifies what the key is used for by Content Server. The valid values are: 1, meaning used for ACS encryption 2, meaning used for DFC authentication 3, meaning used for BOCS authentication
private_key_identifier	string(40)	S	Base 64-encoded SHA1 digest of the DER public key value in the dm_cryptographic_key. key_identifier property. This property is empty if the key_value is 2.

Qual Comp

Purpose

A qual comp object represents a component routine.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_qual_comp
Object type tag: 08

Component routines are written in a language compatible with the platform on which they will be executed. They are used by client applications.

Properties

Table 121, page 325, lists the properties defined for the type.

Table 121. Properties defined for the qual comp type

Property	Datatype	Single or repeating	Description
available_offline	Boolean	S	Currently unused.
class_name	string(64)	S	Specifies the functionality provided by the component; for example, checkin or checkout.
client_capability	integer	S	Indicates who can use the component. Valid values are: 0, meaning no one can use 1, for consumer 2, for contributor 3, for consumer and contributor 4, for coordinator 5, for coordinator and consumer 6, for coordinator and contributor 7, for consumer, contributor, and coordinator 8, for administrator

Property	Datatype	Single or repeating	Description
			9, for administrator and consumer 10, for administrator and contributor 11, for administrator, contributor, and consumer 12, for administrator and coordinator 13, for administrator, coordinator, and consumer 14, for contributor, coordinator, and administrator 15, for all users If there is no value specified for this property, then any user can use the component.
component_obj_id	ID	S	Object ID of the component object associated with the qual comp object.
is_global	Boolean	S	TRUE indicates that the component is executable at all times. The user is not required to select an object before executing the component. FALSE indicates that there must be a currently selected object to execute the component.
nls_label	string(255)	S	Currently unused.
valid_groups	string(32)	R	Names of groups whose users can use this component. If there are no values, any user may use this component. If there are one or more groups named, users must belong to one of the groups to use the component.

Query

Purpose

A query object stores a DQL query as content.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_query
Object type tag: 0a

Query objects are used by Desktop Client™. Their content files are saved DQL queries that a user can execute by selecting them from a menu. Users can create query objects in Desktop Client or through the API. To execute a saved query, a user must have at least Read permission on the query object.

Properties

The Query type has no defined properties. It inherits all of its properties from its supertype, SysObject.

Queue Item

Purpose

A queue item object stores information about an object placed on a user's queue.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_queue_item
Object type tag: 1b

A queue item object records information about tasks and other objects queued to a specific user's Inbox. Tasks are typically queued to a user when that user is selected as the performer of a workflow task. Users and applications can query queue item objects directly.

Properties

Table 122, page 328, lists the properties defined for the type.

Table 122. Properties defined for the queue item type

Property	Datatype	Single or repeating	Description
a_content_type	string(32)	S	File format to use when opening the object associated with the task from the inbox.
a_operations	string(16)	S	User-defined
actual_start_date	Date	S	Actual date that the task is started. This is set when the task is acquired.
content_type	string(32)	S	Not used for work items.
date_sent	Date	S	The date the object was sent.

Property	Datatype	Single or repeating	Description
delete_flag	Boolean	S	Indicates whether the peer work item has been completed or if the event has been delivered or processed. 0 (false) if not completed or delivered or 1 (true) if completed or processed.
dependency_type	string(10)	S	Records the circumstances of the work item's creation. Valid values are: 0, meaning Normal. Standard work item creation (default) 1, meaning Revert. Item created by reverting a previous task 2, meaning a resolution error occurred while trying to resolve an alias for a normal activity. 3, meaning Revert and resolution error. Item was reverted and resolution error occurred for revert activity. 4, meaning Delegated. The item was created when performer of previous task delegated the task or when Content Server automatically delegates the task to the user identified in the performer's delegated_user property. 5, meaning Repeated. Item was created when activity performer chose to repeat the activity, with another performer. 6, meaing Failed auto_delegation: Performer doesn't exist; performer is not available and has no indicated replacement; the indicated replacement does not exist; or

Property	Datatype	Single or repeating	Description
			the replacement availability turned off.
dequeued_by	string(32)	S	Contains the name of the user who completes the peer work item or the name of the user who dequeued the item, either explicitly or implicitly.
dequeued_date	Date	S	Contains the date and time that the item was dequeued or when an event was delivered and processed.
due_date	Date	S	Date by which the task or work item should be completed.
event	string(64)	S	A system- or user-defined event If dependency_type is 4 and the task was delegated by the performer, this is set to dm_delegatedworkitem. If the dependency_type is 4 and the task was automatically delegated by the server, the event is “dm_startedworkitem.
event_detail	string(2000)	S	For dm_batch_flush events, this attribute contains a list of IDs created or deleted in each batch flush. The list is formatted as objID1(op), objID2(op), ... where op is C for creation and D for deletion.
i_event_flags	integer	S	Used internally
i_partition	integer	S	Identifies the partition where the object resides.
instruction_page	integer	S	Not used for work items

Property	Datatype	Single or repeating	Description
item_id	ID	S	For a workflow task, the object ID of the peer work item. For events, the object ID of the object that generated the event. For checkin events, this is the object ID of the new version. For events queued to workflows, this is the workflow ID. If the queue item was generated by an Archive method, this is set to the object ID of the object selected by the method's predicate. If the predicate selects multiple objects, item_id is set to NULL.
item_name	string(255)	S	For a non-workflow event, the object name of the object that generated the event. For a workflow event, the object name of the process or workflow object that generated the event. Unused For workflow tasks or workflow-related events.
item_type	string(32)	S	For a workflow task, the object type of the peer work item. For an event, the object type of the object that generated the event.
message	string(255)	S	Message from the task's sender to the person receiving the queued item.
name	string(32)	S	Contains the name of the user to whom the queued item was sent.
next_tasks_type	string(10)	S	Obsolete. Not used for work items.

Property	Datatype	Single or repeating	Description
plan_start_date	Date	S	Obsolete. Not used for work items.
position	integer	S	If the queue item represents a work item for a work queue and the package associated with the work item has a defined skill level, this property is set to the integer value of that skill level.
priority	integer	S	User or application-interpreted value that defines the item's priority.
read_flag	Boolean	S	Not used for work items.
remote_pending	Boolean	S	Indicates whether the queue item must be distributed to other repositories. TRUE means that the object associated with the queue item must be sent to the repository defined in target_docbase.
router_id	ID	S	Object ID of the workflow that contains the queued task.
sent_by	string(32)	S	Name of the user that sent the queued item. This is the user's Content Server user name.
sign_off_date	Date	S	Not used for work items.
sign_off_required	Boolean	S	Indicates if the task's user must sign off the task before forwarding the task.
sign_off_user	string(32)	S	Not used for work items.
source_docbase	string(120)	S	Name of the repository in which the queue item originated. This property has no value if the queue item is not a distributed item.
source_event	ID	S	Object ID of the queue item in the source repository.

Property	Datatype	Single or repeating	Description
source_stamp	integer	S	The i_vstamp value of the source queue item object.
stamp	ID	S	Object ID of the queue item object If the queue item object is a replica, this is the object ID of the replica.
supervisor_name	string(32)	S	Name of the workflow's supervisor.
target_docbase	string(120)	S	Name of the repository to which the queue item is sent. This property has no value if the queue item is not a distributed item.
task_name	string(128)	S	Contains the string event, if this is an event, or contains the activity identifier if this is a work item.
task_number	string(10)	S	For work items, the sequence number within the workflow of the activity that generated the work item.
task_state	string(10)	S	Indicates the status of the event, work item, or object queued for indexing. The status of the event or work item. One of: acquired, paused, or finished. If the queue item represents an object queued for indexing, the possible values are: • blank, meaning the item has been queued but not yet picked up by the index agent • acquired, meaning the index agent has picked up the

Property	Datatype	Single or repeating	Description
			object but not yet completed processing • failed, meaning the item failed indexing and will not be available for searching • warning, meaning content indexing failed, but metadata was indexed • done, meaning indexing of content and metadata was successful.
task_subject	string(512)	S	For workflow events, constructed from the value in dm_activity.task_subject. For dm_save events, this records the object IDs of the old and new storage areas if the object is saved with a change to a_storage_type. For dm_move_content events, this records the object IDs of the object and the old and new storage areas. For dm_destroy events, this records the object ID of the object's storage area.
task_type	string(10)	S	Not used for work items.

Reference

Purpose

A reference object contains the information that links an object in a remote repository to the local object that mirrors the remote object.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_reference
Object type tag: 47

Users never work directly with reference objects. Content Server or client applications create them when users perform operations on remote objects.

Properties

Table 123, page 335, lists the properties defined for the type.

Table 123. Properties defined for the reference type

Property	Datatype	Single or repeating	Description
binding_condition	string(32)	S	Identifies which version of the remote object to fetch when the object is accessed through its reference object. Valid values are: • EXACT_ID, which directs the server to fetch the version that has the object ID specified in reference_by_id. • VERSION_LABEL, which directs the server to fetch the version carrying the version label defined in binding_label. VERSION_LABEL is the default.

Property	Datatype	Single or repeating	Description
binding_label	string(32)	S	Identifies the version to fetch for operations on the remote object when the value of binding_condition is VERSION_LABEL. The default version label is CURRENT.
i_external_rep	Boolean	S	Indicates whether the replication mode was external. TRUE indicates that external replication was used.
i_global_vstamp	integer	S	Version stamp of the remote object.
i_null_ref_flag	integer	S	Indicates the status of the remote object. Valid values are: 0, meaning the object exists 1, meaning the object does not exist 2, the object exists but no version matches the binding condition
i_ref_acl_domain	string(32)	S	Owner of the remote object's ACL.
i_ref_acl_name	string(32)	S	Name of the remote object's ACL.
i_ref_acl_vstsamp	integer	S	Version stamp of the remote object's ACL.
i_replica_job_id	ID	S	Object ID of the replication job that created the replica. This only has a value if the reference object is the result of a replication job.
i_replica_master	ID	S	Object ID of the original (non-replica) object.

Property	Datatype	Single or repeating	Description
i_replica_source	ID	S	Object ID of the replicated object. This value will be the same as that in i_replica_master unless the replicated object was itself a replica.
i_usage_count	integer	S	The number of system-created reference objects.
local_folder_link	string(200)	S	Identifies the local folder to which the mirror object is linked. If unspecified, this defaults to the user's default folder.
r_last_refresh	DATE	S	The time of the mirror object's last refresh.
r_mirror_object_id	ID	S	Object ID of the mirror object associated with the reference object.
r_next_refresh	DATE	S	Next scheduled refresh time for the mirror object.
r_ref_creator	string(32)	S	Name of the reference object's creator.
r_ref_object_id	ID	S	Object ID of the actual remote object to which the reference object points.
r_ref_version_id	ID	S	Object ID of the exact object that last satisfied the binding condition.
r_reference_info	string(32)	S	Indicates how the reference is used. Valid values are: LINK_REFERENCE CHECKOUT_REFERENCE WORKFLOW_REFERENCE ASSEMBLY_REFERENCE ANNOTATION_REFERENCE REPLICA
r_refresh_status	string(255)	S	Status of the last mirror object refresh.

Property	Datatype	Single or repeating	Description
reference_db _name	string(120)	S	Name of the repository that contains the remote object. This property must be defined if reference_by_name is used to define the version tree containing the remote object. The property is optional if reference_by_id is used to define the object's version tree.
reference_by_id	ID	S	Object ID of the remote object. This is used to identify the version tree containing the remote object. If the binding_condition's value is EXACT_ID, then this identifies the exact object on the tree to which the reference points. If binding_condition is VERSION_LABEL, then this value only identifies the version tree containing the version specified in binding_label. If reference_by_id is set, you cannot set reference_by_name.
reference_by _name	string(255)	S	Full folder path of the remote object. If you set reference_by_name, you must also set reference_db_name. Reference_by_name and reference_by_id are mutually exclusive. You cannot set both.
refresh_interval	integer	S	Defines how often to poll the repository to determine if the mirror object needs to be

Property	Datatype	Single or repeating	Description
			refreshed. The value is specified in minutes. The default is one day.

Registered

Purpose

A registered object contains information about an underlying RDBMS table that has been registered with Content Server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_registered
Object type tag: 19

The information includes the names and datatypes of some or all of the columns in the table and a list of those columns belonging to indexes. To create a registered object, you use the DQL REGISTER statement. To destroy a registered object, use the UNREGISTER statement.

Note: Registered objects are not updated automatically if the definition of their corresponding RDBMS table changes.

Properties

Table 124, page 340, lists the properties defined for the type.

Table 124. Properties defined for the registered type

Property	Datatype	Single or repeating	Description
column_count	integer	S	Number of columns in the table.
column_datatype	string(64)	R	List of the datatypes of the columns.
column_length	integer	R	Lengths of the columns that have a string data type.
column_name	string(64)	R	List of the names of the columns in the table. The names must consist of ASCII characters.

Property	Datatype	Single or repeating	Description
group_table _permit	integer	S	Defines the RDBMS table permit level assigned to the registered table's group.
is_key	Boolean	R	Indicates if an index is built on the column.
owner_table _permit	integer	S	Defines the RDBMS table permit level assigned to the registered table's owner.
synonym_for	string(254)	S	Name of the table in the underlying RDBMS (can be an Oracle table synonym, or an MS SQL Server or Sybase table alias)
table_name	string(64)	S	Name of the table. The name must consist of ASCII characters.
table_owner	string(64)	S	Name of the owner of the RDBMS table (the person who created the RDBMS table).
world_table _permit	integer	S	Defines the RDBMS table permit level assigned to the world.

Registry

Purpose

A registry object contains information about a registered event.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_registry
Object type tag: 26

Users can register to receive notification of an event's occurrence. Users can also initiate auditing of events. Both actions generate registry objects.

Properties

Table 125, page 342, lists the properties defined for the type.

Table 125. Properties defined for the registry type

Property	Datatype	Single or repeating	Description
a_authentication	integer	S	Defines whether an application should authenticate the user before the event. Valid values are: • 0, meaning do not authenticate • 1, meaning authentication required This setting must be enforced by the application. Content Server ignores this property value.

Property	Datatype	Single or repeating	Description
a_esignature_required	integer	S	Whether the event requires an electronic signature created with Addesignature. The default value is 0, meaning FALSE, a signature is not required. A value of 1 means TRUE, a signature is required. This property value is not used by Content Server. It is for use by client applications.
audit_attr_names	string(40)	R	Names of the properties to store in the attribute_list property of the audit trail entry.
audit_subtypes	Boolean	S	Whether to audit subtypes of the audited object type. The audit_subtyes setting is used only if the registered_id property identifies an object type.
controlling_app	string(32)	S	Identifies a particular application. If set, only objects controlled by that application are audited.
event	string(64)	S	Name of the event.
message	string(255)	S	For notifications, defines a message to be sent to the user with the notification. For events, defines a user-friendly event name.
oneshot	Boolean	S	Currently unused.
policy_id	ID	S	Chronicle ID of a lifecycle. If set, only objects attached to the lifecycle are audited.

Property	Datatype	Single or repeating	Description
policy_state	string(32)	S	Name of a particular state in the lifecycle identified in policy_id. If set, only objects in that state are audited.
priority	integer	S	The event priority level. This is user-defined.
is_audittrail	Boolean	S	Defines whether this is a notification or audit trail event. Valid values are 0, for a notification event, and 1, for an audit event.
registered_id	ID	S	Identifies the object or object type being audited or for which a user has registered interest. For individual objects, this is the chronicle ID of the object. For object types, this is the object ID of the type's dm_type object.
sendmail	Boolean	S	Indicates whether to send email notification of the event.
sign_audit	Boolean	S	Whether Content Server must sign entries generated from this registration.
user_name	string(32)	S	Name of the user that registered for the event or initiated auditing.

Relation

Purpose

Describes a relationship between two objects.

Description

Supertype: Persistent Object
Subtypes: Category Assign, Aspect Relation, State Extension, Validation Relation, DSM Sect Doc Attributes, DSM Doc Properties, DSM Study Report
Internal name: dm_relation
Object type tag: 37

A relation object describes a relationship between two objects. The properties identify the objects and some behaviorial characteristics of the relationship.

Properties

Table 126, page 345, lists the properties defined for the type.

Table 126. Properties defined for the relation type

Property	Datatype	Single or repeating	Description
child_id	ID	S	Identifies the object that is the child in the relationship.
child_label	string(32)	S	Version label of the specified child_id. This is optional. If set, the child_id must be the chronicle ID for the child.
description	string(255)	S	Not system defined. Provided for the user's convenience.
effective_date	date	S	Not system defined. Provided for the user's convenience.
expiration_date	date	S	Not system defined. Provided for the user's convenience.
i_partition	integer	S	Identifies the partition where the object resides.

Property	Datatype	Single or repeating	Description
order_no	integer	S	Not system defined. This is provided for the user's convenience. For example, this could be used to order a set of relationships.
parent_id	ID	S	Identifies the object which is the parent in the relationship.
permanent_link	Boolean	S	Indicates if you want to maintain the relationship across versions of the parent object. If the relation is created by a 5.3 DFC, this defaults to the value of dm_relation_type.permanent_link. If the relation is created by the DMCL or a pre-5.3 DFC, the default is FALSE. **Note:** This property is deprecated in version 5.3. For more information, refer to The permanent_link and copy_child properties, page 33.
relation_name	string(32)	S	Identifies a valid relation type object, which defines the type of relationship existing between the two objects.

Relation SSA Policy

Purpose

Describes the relationship between a SysObject object and a storage policy.

Description

Supertype: Relation
Subtypes: None
Internal name: dm_relation_ssa_policy
Object type tag: 37

A relation ssa policy object associates an ssa policy object, representing a content assignment policy, with an object type. The object type must be a SysObject or SysObject subtype. Relationships between object types and ssa policy objects are created through Documentum Administrator.

Properties

Table 127, page 347, lists the properties defined for the type.

Table 127. Properties defined for the relation SSA policy type

Property	Datatype	Single or repeating	Description
target_object_type	string(32)	S	Identifies the object type of the related object.
ssa_policy_id	ID	S	Object ID of the ssa policy object enforced for the related object.

Relation Type

Purpose

Describes a relationship that can exist between two objects in the repository.

Description

Supertype: Persistent Object
Subtypes: dm_foreign_key
Internal name: dm_relation_type
Object type tag: 38

A relation type object describes a relationship that can exist between two objects in the repository.

When a user sets up a relationship between two objects, for example, by creating an annotation and attaching it to a document, the server creates an object of type dm_relation. The relation object links the annotation (child) to the document (parent) and describes the relationship between them by referring to the associated dm_relation_type object.

A user must have Sysadmin or Superuser privileges to create or destroy a relation type object.

For information about defining relationships or relationships between two objects, refer to Relationships, page 29.

Properties

Table 128, page 348, lists the properties defined for the type.

Table 128. Properties defined for the relation type object type

Property	Datatype	Single or repeating	Description
a_controlling_kind	string(32)	S	Set to a value to indicate whether relationships of this type are created by users or internally. The setting is used by EMC Documentum clients to determine whether to allow users to create and destroy instances of the relation type.
child_parent_label	string(255)	R	User-defined label for a child-to-parent relationship.

Property	Datatype	Single or repeating	Description
child_type	string(32)	S	The object type of valid child objects in the relationship.
copy_child	integer	S	Specifies whether to copy the child in a relationship when permanent_link is T and the parent is copied. Valid values are: 0, meaning do not copy the child 1, meaning copy the child This property is used only by DFC when copying a parent object in a relationship and dm_relation_type.permanent_link is T. The DMCL and pre-5.3 versions of DFC do not use this property.
description	string(250)	S	User-defined description of the relationship.
direction_kind	integer	S	The direction of the relationship. Valid values are: • 0, meaning from parent to child • 1, meaning from child to parent • 2, meaning bidirectional — objects are treated as siblings The default is 0.
i_partition	integer	S	Identifies the partition where the object resides.
integrity_kind	integer	S	Indicates how referential integrity is enforced. Valid values are: • 0, meaning allow delete • 1, meaning restrict delete • 2, meaning cascade delete The default is 0.
parent_child_label	string(255)	R	User-defined label for a parent-to-child relationship.

Property	Datatype	Single or repeating	Description
parent_type	string(32)	S	Defines the object type of a valid parent object in the relationship.
permanent_link	Boolean	S	Whether the relationship is maintained when the parent is copied or versioned. T means that the relationship is maintained with the new parent object. F means that the relationship is not maintained. The default value is F. This property is used only by DFC 5.3 when copying or versioning a parent object in a relationship. The DMCL and pre-5.3 versions of DFC do not use this property. **Note:** When an instance of the relationship is created using a 5.3 DFC-based application, this value is the default value assigned to dm_relation.permanent_link if that property is not set explicitly.
relation_name	string(32)	S	The name of the relationship. Names must be unique. The names of system-defined relationships begin with dm_.
security_type	string(10)	S	Defines the security applied to objects of type dm_relation whose relation_name property value matches the relation_name property for the dm_relation_type object. Valid values are: **SYSTEM** Only users with Superuser or Sysadmin privileges can create, modify, or drop dm_relation objects having this relation_name **PARENT**

Property	Datatype	Single or repeating	Description
			Security is determined by the object type of the parent object participating in this kind of relationship. **CHILD** Security is determined by the object type of the child object participating in this relationship. **NONE** No security is applied to the dm_relation objects representing instances of this relationship.

Relationship Def

Purpose

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_relationship_def
Object type tag: 08

A relationship def object records the definition of a relationship. These objects are created using the IDfRelationshipInfo interface in DFC. Each relationship def object must have a binding aspect attached that describes the specific physical binding that characterizes the relationship.

The object type is installed with the Smart Container DAR file when a repository is configured.

Properties

Table 129, page 352, lists the properties defined for the type.

Table 129. Properties defined for the relationship def type

Property	Datatype	Single or repeating	Description
is_composition	Boolean	S	Indicates whether the relationship is a strict composition, meaning the related children have a single parent and that destroying the parent also destroys the children. T means that the relationship is a strict composition. F means that it is not a strict composition. The default is F.

Property	Datatype	Single or repeating	Description
is_private	Boolean	S	Indicates whether the relationship is available for external use. T means that the relationship is private, not available for public use. F means that the relationship may be used by external applications.
operation_name	string(255)	R	Name of an operation to be applied after some triggering event. The name at any particular index position is associated with the module and metadata in the corresponding index positions in operation_module and operation_module metadata. The name should be a fully qualified name, such as a Java package name, to guarantee uniqueness.
operation_module	string(255)	R	Module that implements the operation identified in operation_name. The module at a particular index position is the module that implements the operation identified at the same index position in operation_name.
operation_module_ metadata	string(1024)	R	Metadata used to configure the operation module. The metadata at a particular index position is the metadata used to configure the module identified at the same index position in operation_module.
rel_description	string(1024)	S	User-defined description of the relationship.

Property	Datatype	Single or repeating	Description
rel_degree	integer	S	Indicates the degree of the relationship from parent to child. Valid values are: 0, meaning a one-to-one relationship 1, meaning a many-to-one relationship 2, meaning a many-to-many relationship
required_source_ aspects	string(64)	R	List of aspects that will be associated with an instance of the source (parent) type when participating in this relationship.
required_target_ aspects	string(64)	R	List of aspects that will be associated with an instance of the target (child) type when participating in this relationship.
source_constraint_ set_id	ID	S	Object ID of the dmc_constraint_set object that records the set of constraints applied to the source object in the relationship.
source_role	string(255)	S	Name for the role played by the source object in the relationship. For example, if the relationship relates a customer with a vendor, the role for the customer may 'buyer'.
source_type	string(27)	S	Identifies the top-most type in the hierarchy of types that may participate as parent in this relationship. This type or any of its subtypes may be a parent in the relationship.

Property	Datatype	Single or repeating	Description
target_constraint_set_id	ID	S	Object ID of the dmc_constraint_set object that records the set of constraints applied to the target object in the relationship.
target_role	string(255)	S	Name for the role played by the target object in the relationship. For example, if the relationship relates a customer with a vendor, the role for the vendor may 'seller'.
target_type	string(27)	S	Identifies the top-most type in the hierarchy of types that may participate as child in this relationship. This type or any of its subtypes may be a child in the relationship.

Table 130, page 355, lists the properties added by a system-defined default aspect for the type.

Table 130. Default aspect properties defined for the relationship def type

Property	Datatype	Single or repeating	Description
logical_version	string(32)	S	Supports logical versioning of persistent objects From the aspect named dm_logical_aspect
annotation_name	string(255)	R	A non-qualifiable property that stores the names of the annotation From the aspect named dm_annotable

Property	Datatype	Single or repeating	Description
module_name	string(255)	R	A non-qualifiable property that stores module names From the aspect named dm_annotable
annotation_ namemetatdata	string(1054)	R	A non-qualifiable property that stores annotation metadata values From the aspect named dm_annotable

Replica Record

Purpose

Contains information about replicated content in distributed storage areas.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_replica_record
Object type tag: 2d

A replica record object contains information about replicated content in distributed storage areas. The object is created and used by Content Server to manage distributed content storage areas.

Properties

Table 131, page 357, lists the properties defined for the type.

Table 131. Properties defined for the replica record type

Property	Datatype	Single or repeating	Description
component_id	ID	R	Object IDs of the components storage areas that contain replicas of the content file
data_ticket	integer	R	Information used internally to fetch and save the content.
epoch_number	integer	S	Used internally to manage distributed storage areas.
format_id	ID	S	Object ID of the format object representing the format of the content file associated with the replica record.
i_partition	integer	S	Identifies the partition where the object resides.
other_ticket	integer	R	Information used internally to fetch and save the content.

Property	Datatype	Single or repeating	Description
owner_id	ID	S	Object ID of the distributed storage area that contains the content file.
r_storage_ticket	integer	S	Used internally to manage the content

Retainer

Purpose

Describes a retention policy.

Description

Supertype: SysObject
Subtypes: dmc_rps_retainer
Internal name: dm_retainer
Object type tag: 08

A retainer object records information about one retention policy. Retention policies are created through Retention Policy Services (accessed through Documentum Administrator) and require a Retention Policy Services license to create and use. Retainer objects are owned by a member of the dm_retention_managers group or by the group itself. Retainer objects must be uniquely named within the repository and cannot be versioned.

Properties

Table 132, page 359, lists the properties defined for a retainer object.

Table 132. Properties defined for the retainer type

Property	Datatype	Single or repeating	Description
aging_method	integer	S	Identifies the type of the retainer. Valid values are: 0, meaning under fixed retention value 1, meaning under conditional (event-based) retention 2, meaning under retention hold

Property	Datatype	Single or repeating	Description
disposition_rule	integer	S	Defines how RPS disposes of an object. Valid values are: 0, meaning DESTROY CHILDREN 1, meaning DESTROY_VDM 2, meaning DESTROY_ROOT
enforcement_rule	integer	S	Defines how objects under this policy are managed when the retention period expires. Valid values are: 0, meaning objects are retained until the retention rule is removed using RPS. 1, meaning Content Server disallows deletion if the objects are linked to an active retainer. 2, meaning objects are retained until their associated structural retainer or retainers are removed **Note:** Disposition of objects associated with an active retainer is controlled by the retention policy's disposition rules defined within Retention Policy Services.
immutability_rule	integer	S	Defines whether to mark documents under the control of the policy immutable. Valid values are: 0, meaning objects are not marked immutable 1, meaning objects are marked immutable The default is 0. Notes on use: • When the retainer is container-based (dm_retainer. retainer_strategy=1), the

Property	Datatype	Single or repeating	Description
			container is not affected by this setting. • When an object that is marked immutable due to its retention policy or policies is detached from all retainers, its r_immutable_flag property is not reset to F unless the object is the CURRENT version and is not a cabinet or folder.
r_retention_status	integer	S	Identifies the status of the retainer. Valid values are: 0, meaning active—objects under the control of the retainer may not be deleted and new objects may be added to the retainer 1, meaning locked—the retainer is active but no new objects may be placed under its control 2, meaning inactive—objects under the control of the retainer may be deleted from the repository The default is 0.
rendition_rule	integer	S	Defines how multiple content files for one document are handled. Valid values are: 0, meaning all content files associated with a document under the control of the policy are under retention 1, meaning only the primary content files of a document under the control of the policy are under retention

Property	Datatype	Single or repeating	Description
retainer_root_id	ID	S	Identifies the object to which this retainer is applied. If dm_retainer.retainer_strategy is set to 0 (object-based retention), retainer_root_id is the object ID of the individual object under retention. If retainer_strategy is set to 1 (container-based retention), retainer_root_id is the object ID of the folder containing the objects under retention.
retainer_strategy	integer	S	Defines whether the retainer controls a single object or multiple objects placed in a container. Valid values are: 0, meaning the retainer controls a single object 1, meaning the retainer controls multiple objects in a container The default is 0.
retention_date	date	S	Retention value when dm_retainer.retention_rule_type is set to 1, meaning date.
retention_interval	integer	S	Retention value when dm_retainer.retention_rule_type is set to 2, meaning interval. The value is interpreted in seconds.

Property	Datatype	Single or repeating	Description
retention_rule_type	integer	S	Specifies how the retention period is defined. Valid values are: 0, meaning there is no specified retention period 1, meaning the retention period is defined as a specific date 2, meaning the retention period is defined as an interval The default is 0.
retention_storage_class	string(40)	S	Reserved for future use
version_rule	integer	S	Defines how retention is applied to new versions of documents under the control of the retainer. The only valid value is 0, meaning that the retention period applies only to the specific version to which the retainer was attached.

Routecase Condition

Purpose

Records a route case condition expression for an automatic transition of a workflow activity.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_routecase_condition
Object type tag: 00

A routecase condition object records a conditional expression in a route case condition. You cannot create these objects directly. They are created when an addConditionRouteCase method (defined for the IDfActivity interface) is executed from Process Builder to save an activity's route case conditions when one or more of the route cases contains an XPath expression.

Properties

Table 133, page 364, lists the properties defined for the type.

Table 133. Properties defined for the routecase condition type

Property	Datatype	Single or repeating	Description
a_attribute_name	string(32)	S	Name of the property referenced in the expression, if any
a_boolean_value	Boolean	S	Value to be used in the comparison if the datatype identified in a_value_type is Boolean. This is not set if unless the datatype is Boolean.
a_double_value	double	S	Value to be used in the comparison if the datatype identified in a_value_type is double. This is not set if unless the datatype is double.

Property	Datatype	Single or repeating	Description
a_id_value	ID	S	Value to be used in the comparison if the datatype identified in r_value_type is ID. This is not set if unless the datatype is ID.
a_int_value	integer	S	Value to be used in the comparison if the datatype identified in a_value_type is integer. This is not set if unless the datatype is integer.
a_object_alias	string(128)	S	Name of the package, or manifest values referring to the workflow or work item
a_relational_op	integer	S	The relation operator in the condition. Valid values are: 0, meaning = 1, meaning <> 2, meaning < 3,meaning > 4, meaning <= 5, meaning >=
a_repeating_attr_flag	integer	S	Indicates whether the property named in a_attribute_name is a repeating property and if it is a repeating property, which values to examine when evaluating the condition. Valid values are: -1, meaning the property is not a repeating property 0, meaning ANY 1, meaning ALL 3, meaning LAST 2, meaning FIRST

Property	Datatype	Single or repeating	Description
a_string_value	string(1024)	S	Value to be used in the comparison if the datatype identified in a_value_type is string. This is not set if unless the datatype is string.
a_time_value	date	S	Value to be used in the comparison if the datatype identified in a_value_type is Date. This is not set if unless the datatype is Date.
a_value_type	integer	S	Data type of the value in the relational expression. The data type is expressed as an IDfValue constant.
a_xpath_datatype	string(64)	S	The xschema's built-in datatype name. This is set only if the condition includes an XPath expression.
a_xpath_def_ namespace_uri	string(255)	S	The schema URI for the default XPath namesapce This is set only if the condition includes an XPath expression.
a_xpath_expression	string(1024)	S	An XPath expression. This is set only if the condition includes an XPath expression.
a_xpath_namespace_ prefix	string(64)	R	The prefixes that alias the namespace URIs within the XPath expression This is set only if the condition includes an XPath expression.

Property	Datatype	Single or repeating	Description
a_xpath_namespace_uri	string(255)	R	Namespace URIs of the XPath expression components This is set only if the condition includes an XPath expression.
a_xpath_value	string(1024)	S	Literal value used in the XPath transition condition evaluation. The value is in the format in which it is found in the XML document. This is set only if the condition includes an XPath expression.
r_aspect_name	string(64)	R	Used internally

Scope Config

Purpose

Defines the context for the referenced display config objects.

Description

Supertype: None
Subtypes: None
Internal name: dm_scope_config
Object type tag: 6c

A scope config object defines the context for the referenced display config objects. Each display config object defines a set of properties and display hint for the properties. Each scope config object identifies one or more display config objects and one or more contexts in which the display config is used. Both scope config and display config objects are used by client applications. Content Server does not use these objects.

Properties

Table 134, page 368, lists the properties defined for the type.

Table 134. Properties defined for the scope config type

Property	Datatype	Single or repeating	Description
display_config	ID	R	Object IDs of the display config objects used in the scopes defined in the scope config object.
scope_class	string(32)	R	The category of the scope. The only valid value is application.
scope_value	string(32)	R	The specific scope within the scope class. For example, if scope_class is application, scope_value might be Webtop or a DAR file name. The scope at a particular index position is the specific instance of the class at the corresponding index position in scope_class.

Script

Purpose

Stores a script as content.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_script
Object type tag: 08

A script object stores a script as content. Script objects are used by Documentum clients. Their content files contain the scripts. Depending on where the script object is stored, the script is executed automatically when users connect to a repository or on demand, when users select the script from a menu. Users must have at least Read permission for the script object to execute its associated script.

Properties

The script type has no defined properties. It inherits all of its properties from its supertype, SysObject.

Sequence

Purpose

Used by Content Server to generate object IDs.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_sequence
Object type tag: 20

A sequence object is used by Content Server to generate object IDs. There is one sequence object for each object type in a repository, up to a maximum of 255.

Properties

Table 135, page 370, lists the properties defined for the type.

Table 135. Properties defined for the sequence type

Property	Datatype	Single or repeating	Description
i_current_mask	integer	S	Used internally to manage object IDs
i_high_mask	integer	S	Used internally to manage object IDs
i_high_water _mark	integer	S	The highest number that can be assigned
i_in_use	Boolean	S	Indicates if any of the numbers in the allowed range have been assigned.
i_last_no	integer	S	Contains the last number assigned
i_low_mask	integer	S	Used internally to manage object IDs
i_low_water _mark	integer	S	The first or lowest number that can be assigned

Server Config

Purpose

Describes a Content Server.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_server_config
Object type tag: 3d

A server config object contains information that the server uses to define its operation and operating environment, such as the number of allowed concurrent sessions, maximum cache sizes, and the storage area locations, and the locations of executables that the server calls.

Properties

Table 136, page 371, lists the properties defined for the type.

Table 136. Properties defined for the server config type

Property	Datatype	Single or repeating	Description
a_silent_login	Boolean	S	Used internally
agent_launcher	string(32)	S	The name of the method object that launches the agent exec process. The default is agent_exec_method.
alias_set_id	ID	S	The object ID of the alias set object representing the system-level default alias set.
app_server_name	string(32)	R	Name of a Java servlet. The name do_method identifies the servlet to which DO_METHOD functions are directed.

Property	Datatype	Single or repeating	Description
app_server_uri	string(255)	R	URI for the servlet at the corresponding index position in app_server_name. Contains the host, port, and servlet_path in the following format: http://*host*:*port*/*servlet_path* *host* is the IP address or name of the machine hosting the application server. *port* is the port number on which the application server is listening. *servlet_path* is the path to the servlet to which an HTTP_POST request is directed. The URI can contain only ASCII characters.
application_ access_control	Boolean	S	T (TRUE) means users without Superuser privileges must use an application access token to connect to the server. F (FALSE) means any user can connect without an access token. The default is F.
assume_user _location	string(32)	S	Name of the location object for the directory containing the assume user program. The default is assume_user.
auth_plugin_ location	string(32)	S	Name of the location object that identifies the directory location of authentication plugins. This is set to auth_plugin by default.
cached_types	string(32)	R	Names of the user-defined object types that you want to cache on server startup. The default is a single blank.
certdb_location	string(32)	S	Name of the location object pointing to the location of the certificate database. The default is ldapcertdb_loc.
change_password _location	string(32)	S	Name of the location object for the directory containing the change password program. The default is change_password.

Property	Datatype	Single or repeating	Description
checkpoint_ interval	integer	S	Defines how often the server broadcasts its service information. The default is 300 (seconds).
client_cache_size	integer	S	Maximum allowed size of the client cache, expressed as the number of objects. The default is 50. The number of objects may exceed the specified maximum size if an object must be added to the cache and no candidate for deletion is found. Also, persistently cached objects are not counted towards the maximum.
common_location	string(32)	S	Name of the location object for the common directory. The default is common.
compound_ integrity	Boolean	S	Indicates if the server is enforcing referential integrity for virtual documents. The default is TRUE.
concurrent_ sessions	integer	S	Number of concurrent sessions. The default is 100.
dba_location	string(32)	S	Name of the location object that identifies the location of the dba directory in the Documentum installation. The default is dm_dba.
default_acl	integer	S	Specifies which ACL the server will use as the default ACL when creating a new object if an ACL is not explicitly associated with the object. Valid values are: • 1, to specify the ACL associated with the object's primary folder • 2, to specify the ACL associated with the object's type • 3, to specify the ACL associated with the user who created the object • 4, meaning no default ACL is specified The default is 3.

Property	Datatype	Single or repeating	Description
default_client _codepage	string(64)	S	Identifies the default code page for clients. Legal values are: US-ASCII UTF-8 ISO_8859-1 Shift_JIS EUC-JP EUC-KR ISO-10646-UCS-2 The value is determined programmatically and set during Content Server installation.
events_location	string(32)	S	Name of the location object that identifies the events directory. The default is events.
extra_directory_ config_id	ID	R	Object IDs of the ldap config objects representing LDAP directories from which users and groups are synchronized.
far_stores	string(32)	R	Names of the storage areas that are considered far for the server. The default is a single blank.
fulltext_location	string(64)	S	Name of the location object that identifies the location of the fulltext configuration file (dmfulltext.ini).
ignore_pre _processing	Boolean	S	Not currently used
keep_entry_ interval	integer	S	Defines how long a connection broker keeps a server entry in the absence of a checkpoint message from the server. The default is 1440 minutes (24 hours).
ldap_config_id	ID	S	Object ID of the ldap config object representing the default LDAP directory used by Content Server.

Property	Datatype	Single or repeating	Description
locale_name	string(32)	S	Content Server's locale. Legal values are: en, for English es, for Spanish fr, for French de, for German it, for Italian sv, for Swedish ja, for Japanese ko, for Korean zh, for Chinese The value is determined programmatically and set during Content Server installation.
log_location	string(32)	S	Name of the location object for the logs directory. The default is log.
login_ticket_ timeout	integer	S	Defines the length of time for which a login ticket is valid. The value is interpreted in minutes. The minimum accepted value is 1 minute. The default is 5 minutes.
mail_method	string(32)	S	Name of the method that Content Server uses to send email messages. Valid values are dm_event_sender, an empty string, or dm_event_template_sender. The default value is dm_event_sender. "dm_event_sender_template" is a valid value only if you have Business Process Designer installed and are using the workflow email template feature. **Note:** Setting this to an empty string does not disable server email. If you want to disable server email, you must set the mail_notification key in the server.ini file.
max_login_ticket_ timeout	integer	S	Maximum length of time, in minutes, that a login ticket can remain valid. The minimum value is 1 minute. The default value is 43200 minutes (30 days).

Property	Datatype	Single or repeating	Description
nfs_enabled	Boolean	S	Indicates if the server is using NFS for file sharing. The default is FALSE.
nls_location	string(32)	S	Name of the location object for the nls directory. The default is a single blank.
object_name	string(255)	S	Identifies the server config object. The default is the name of the repository for which the server was started. Server config names for a Content Server at a primary site is <=32 characters in length. Server config names for content-file servers (remote Content Servers) are 25 characters or less. Server config names are created automatically when the server is installed.
operator_name	string(32)	S	Identifies the operator for the repository (for archiving and restore operations). The default is the value in the server config object's owner_name property.
projection_enable	Boolean	R	Indicates whether projection to the connection broker specified at the corresponding index position in projection_targets is enabled.
projection_netloc_enable	Boolean	R	Indicates whether projection to the network location specified at the corresponding index position in projection_netloc_id is enabled.
projection_netloc_id	string(80)	R	User-defined identifiers of network locations.
projection_notes	string(80)	R	User-defined
projection_ports	integer	R	Identifies the port on which the connection broker is listening. The value at each index level is matched to the connection broker specified at the corresponding level in projection_targets. The default is 1489.

Property	Datatype	Single or repeating	Description
projection_ proxval	integer	R	Proximity values projected to the connection brokers, the network locations, or both. The value at each index position is projected to the connection broker specified at the corresponding position in projection_targets if projection_enable is TRUE in the corresponding index position. Similarly, the value at each index position is projected to the network location specified at the corresponding index position in projection_netloc_id if projection_netloc_enable is TRUE in the corresponding index position.
projection_targets	string(80)	R	Names of the host machines on which the connection brokers reside.
r_castore_enabled	Boolean	S	Whether the server was installed with a Content Services for EMC Centera license. Valid values are: 1, meaning the server was installed with a Content Services for EMC Centera license 2, meaning the server was not installed with a Centera license.
r_host_name	string(64)	S	Name of the host machine on which the server resides.
r_install_domain	string(16)	S	Name of the Windows domain
r_install_owner	string(32)	S	Name of the installation owner
r_process_id	integer	S	Contains the process id of the main server thread (parent server).
r_server_version	string(32)	S	Version number of the server.

Property	Datatype	Single or repeating	Description
r_trusted_mode	integer	S	Whether the server is a trusted server. 0 means that the server is not a trusted server. 1 means the server is a trusted server. This property is set during server installation, depending on whether the user provides a trusted server license key.
rend_backing_ store	string(32)	S	obsolete
restrict_su_ticket_ login	Boolean	S	T (TRUE) means a user with Superuser privileges cannot connect to the server using a global login ticket. F (FALSE) allows users with Superuser privileges to connect to the server using a global login ticket. The default is F.
rightsite_image	string(255)	S	Used to construct the URL when a user is sending a document to a mail recipient. Valid values are: • webtop • a RightSite imageRightSite images are found in the Windows registry. An example of a Rightsite image is /rs-bin/RightSite.dll The default is webtop.
secure_connect _mode	string(16)	S	The type of port on which the server is listening. Valid values are: native, meaning an unsecure port secure, meaning a secure port (using SSL) dual, meaning both a native and a secure port Non-trusted servers do not use this property. They accept only native connections.

Property	Datatype	Single or repeating	Description
secure_writer _location	string(32)	S	Name of the location object for the directory containing the secure writer program. The default is secure_common_area_writer.
server_cache_size	integer	S	Maximum allowed size, expressed as the number of objects. The default is 200.
server_os_ codepage	string(64)	S	The code page used by the operating system of the Content Server host. Legal values vary by operating sytsem. Valid options are: US-ASCII UTF-8 ISO_8859-1 Shift_JIS EUC-JP EUC-KR ISO-10646-UCS-2 The value is determined programmatically and set during Content Server installation.
sibling_ checkpoint _interval	integer	S	Defines how often the server broadcasts sibling information (Not currently used)
sibling_export _enabled	Boolean	S	Indicates if the server can broadcast information about its sibling servers. (Not currently used)
signature_chk_loc	string(32)	S	Name of the location object for the directory containing the signature validation program. The default is validate_signature.
smtp_server	string(64)	S	Name of the SMTP server host. This is used only on Windows platforms, by the email notification system.
stage_destroyer _location	string(32)	S	Not currently used
system_converter _location	string(32)	S	Name of the location object for the directory containing the convert.tbl file and the system-supplied transformation scripts. The default is convert.

Property	Datatype	Single or repeating	Description
temp_location	string(32)	S	Name of the location object for the temp directory. The default is temp.
turbo_backing_ store	string(32)	S	Contains the name of the file store object for the file store storage area where the server puts renditions generated by indexing blob and turbo content. The default is filestore_01.
user_converter _location	string(32)	S	Name of the location object that identifies the directory containing the user-defined transformation scripts. The default is a single blank.
user_validation _location	string(32)	S	Name of the location object for the directory containing the user validation program. The default is validate_user.
verity_location	string(32)	S	Obsolete
web_server_loc	string(255)	S	Name of the Web server's host machine, and optionally, its domain and protocol. Examples: `cinderella cinderella.mycompany.com http://cinderella.mycompany.com` The default protocol is http.
web_server_port	integer	S	Identifies the port which the Web server is using. The default value is 80.
wf_agent_worker_ threads	integer	S	Number of workflow agent worker sessions. The maximum value is 1000. The default value is 3. Setting this to 0 disables the workflow agent.
wf_sleep_interval	integer	S	Length of time, in seconds, that the master workflow agent master session waits before querying the repository for activities awaiting execution in the absence of a notification of an activity waiting for execution. The default is 5 seconds. There is no maximum value.

Session

Purpose

Not currently used.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_session
Object type tag: 01

The session object type is constructed by the server during server start-up. It is not currently used by Content Server, and there are no objects of that type stored in the repository. However, you can execute a describe command through IDQL to view its properties, and it is visible as a dm_type object.

Smart List

Purpose

Represents the set of objects determined by the SELECT query defined for the Smart List.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_smart_list
Object type tag: 08

A Smart List object represents the set of objects determined by the SELECT query defined for the Smart List. When a user opens a Smart List, the query is executed and the retrieved items are displayed to the user in a window that allows the users to manipulate the objects.

A Smart List object can only be created using Desktop Client or Webtop. On Webtop, the Smart List object contains the saved search object created by the search service. This object contains the search definition and search results. The Smart List object created by Desktop Client can be executed in Webtop but not edited nor listed as a saved search.

If the Smart List is created from Desktop Client, the query is saved as DQL. If the Smart List is created from Webtop, the query is saved as an XML file. Smart Lists created in Desktop Client can be read by Webtop. However, Smart Lists created in Webtop cannot be read by Desktop Client.

Properties

Table 137, page 382, lists the properties defined for the type.

Table 137. Properties defined for the Smart List type

Property	Datatype	Single or repeating	Description
has_results	Boolean	S	TRUE if the saved search was saved with results

Property	Datatype	Single or repeating	Description
query_type	string(32)	S	Identifies the search type. Valid values are: passthrough_query query_builder or blank when query type is unknown
results_count	integer	S	Number of results the search had when the search was saved
selected_sources	string(256)	R	List of the sources searched by the search. These can be repositories or external sources.

State Extension

Purpose

Identifies the lifecycle state with which a state extension object is associated.

Description

Supertype: Relation
Subtypes: RPS Phase Rel, RPS Retainer Event Rel
Internal name: dm_state_extension
Object type tag: 37

A state extension object identifies the lifecycle state with which the state extension object is associated. The state extension object type is the base object type for state extension object subtypes. Typically, the state extension object type is subtyped to add properties to store the application-specific information needed for particular lifecycle states.

Properties

Table 138, page 384, lists the property defined for the type.

Table 138. Properties defined for the state extension type

Property	Datatype	Single or repeating	Description
state_no	integer	S	Number of the state with which the state extension object is associated. The number is the value in the dm_policy object's state_no property.

State Type

Purpose

Identifies a particular lifecycle state type name and application pair.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_state_type
Object type tag: 00

A state type object identifies a particular lifecycle state type name and application pair. State type objects can be used to associate a state type name with a particular application. Some Documentum client applications require specific state type names be assigned to lifecycle states. User applications may also make use of state types. (For more information, refer to *Content Server Fundamentals*.)

Properties

Table 139, page 385, lists the properties defined for the type.

Table 139. Properties defined for the state type object type

Property	Datatype	Single or repeating	Description
application_code	string(32)	S	Name of the application that recognizes the state type.
state_type_name	string(32)	S	Name of a lifecycle state type.

SSA Policy

Purpose

Defines a content assignment policy for new content files.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_ssa_policy
Object type tag: 08

An ssa policy object defines storage rules for content files. The rules are stored as content of the object. A policy is associated with one or more object types through a relationship, recorded in an relation ssa policy object.

SSA policy objects are created using Documentum Administrator. Existing policies can be copied or modified, but not versioned. You must have installed Content Server with a Content Storage Services license to create and use assignment policies.

Properties

Table 140, page 386, lists the properties defined for the type.

Table 140. Properties defined for the SSA policy type

Property	Datatype	Single or repeating	Description
is_activated	Boolean	S	Whether the policy is active (being enforced by the policy engine). T (TRUE) means the policy is enforced. F (FALSE) means the policy is inactive. The default is T.

Store

Purpose

Represents a content storage area.

Description

Supertype: Persistent Object
Subtypes: Distributed Store, File Store, Linked Store, Blob Store, External Store, CA Store
Internal name: dm_store
Object type tag: 0e

A store object represents a content storage area. The properties in store objects may be changed only by users with Sysadmin or Superuser user privileges.

Properties

Table 141, page 387, lists the properties defined for the type.

Table 141. Properties defined for the store type

Property	Datatype	Single or repeating	Description
base_url	string(255)	S	The basic URL used to retrieve contents directly from a storage area.
capacity	integer	S	Not currently used
component	string(64)	R	The name of the storage objects representing component storage areas. (Used by linked store and distributed stores.)

Property	Datatype	Single or repeating	Description
compression_mode	integer	S	Applicable only to file store and ca store storage areas and if the server is installed with a Content Storage Services license, this property indicates whether content written to the storage area is compressed. Valid values are 0, meaning the content in the storage area is not compressed, and 1, meaning the content is compressed. If not explicitly set when the storage area is created, the default value is 0. The setting may not be changed after the storage area is created.
content_dupl_pref	integer	S	Used only if the server is installed with a Content Storage Services license, this property indicates the content duplication preference for the storage area. Valid values are: 0, meaning Content Server does not check the storage area for duplicate content files when saving content files to the storage area 1, meaning Content Server checks the storage area for duplicate content files when saving content files to the storage area The default is 0. Refer to the *Content Server Administration Guide* for information about how this property is used.

Property	Datatype	Single or repeating	Description
content_hash_mode	integer	S	Used only if the server is installed with a Content Storage Services license, this property configures hash generation for content files in the storage area. This property setting is applicable only to file store storage areas. Valid values are: 0, meaning do not generate hashes for the content files in the storage area. 1, meaning generate a SHA1 hash for each content file in the storage area If content_dupl_pref is set to a non-zero value, the value of content_hash_mode must be 1. Content Server sets the value to 1 automatically under that condition. If content_dupl_pref is 0, the default value for content_hash_mode is 0. Refer to the *Content Server Administration Guide* for information about how this property is used.
crypto_key	string(256)	S	Contains the symmetric key used to encrypt the content files stored in the storage area. You cannot create indexes on this property.

Property	Datatype	Single or repeating	Description
crypto_mode	integer	S	Whether the contents of the storage area are encrypted. 0 means the contents are not encrypted. 1 means the contents are encrypted. This value is set when the storage area is created and cannot be changed afterwards. You can only set this value for file store storage areas. The default value is 0.
current_use	integer	S	For internal use
default_retention_days	integer	S	Number of days to retain content in the storage area. The count of days begins when the content is saved to the storage area. This property is used only for ca store (retention-enabled) storage areas. If this property is set, its value takes precedence over a value set in dm_ca_store.a_default_retention_date.
digital_shredding	Boolean	S	Whether digital shredding is enabled for the storage area. Valid values are: 0, meaning shredding is disabled 1, meaning shredding is enabled The default is 0. This property only affects file store storage areas and may only be set to 1 if the server is installed with a Trusted Content Services license.

Property	Datatype	Single or repeating	Description
get_method	string(32)	S	Contains the name of the dm_method object representing the user-defined program that retrieves a specified content.
i_use_mask _after	integer	S	Used internally to manage storage ticket numbers for content files.
media_type	integer	S	Identifies the media type of content stored in this storage area. Valid values are: • 0, for any type other than thumbnail or streaming • 1, for thumbnail • 2, for streaming The property cannot be changed after the storage area is created.
name	string(64)	S	Contains the name of the store object. This must be a unique name within the repository.
offline_get _method	Boolean	S	Indicates whether the application should regard the retrieved content as immediately available or awaiting restoration (from archiving).
r_component	ID	R	The object IDs of any component storage areas.
r_component _count	integer	S	The total number of component storage areas.
r_status	integer	S	Records the state of the storage area. Valid values are: 0, for on-line 1, for off-line 2, for read-only

Property	Datatype	Single or repeating	Description
require_ticket	Boolean	S	Indicates whether Content Server generates a ticket when returning a URL to a content file. TRUE means a ticket is returned. FALSE means a ticket is not returned. The default is FALSE.
store_type	integer	S	Value indicating the subtype of the storage area. This can be one of the following values: • 1, for file store • 3, for linked store • 4, for distributed store • 5, for blob store • 7, for external store • 8, for external file store • 9, for external URL store • 10, for external free store • 11, for CA store **Note:** 2 and 6 are currently unused values.

Subcontent

Purpose

Stores a content file in turbo storage or the content addresses of content stored in content-addressed storage systems.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_subcontent
Object type tag: 64

A subcontent object stores a content file in turbo storage or the content addresses of content stored in content-addressed storage systems. When a content file in turbo storage is too big to store in a content object, the content is stored in subcontent object. If a content file stored in a content-addressed system is modified or if its metadata properties in the storage system are modified, the storage system creates a new address for the content file. This address is stored in a subcontent object.

Subcontent objects are internal objects that users and applications cannot access directly. However, users and applications can query subcontent objects using DQL.

Properties

Table 142, page 393, lists the properties defined for the subcontent object type.

Table 142. Properties defined for the subcontent type

Property	Datatype	Single or repeating	Description
parent_id	string(16)	S	Object ID of the dmr_content object that represents the content file or content address stored in this object.

Property	Datatype	Single or repeating	Description
i_contents	For all databases except Sybase: string(2000) For Sybase: string(255)	R	For objects in turbo storage, this contains the object's content file if the content is too big for i_contents in the content object. This property cannot be selected using either the API or DQL when it used as a storage area for turbo content. For objects in content-addressed storage, this contains additional content addresses generated by operations on the object or its storage metadata values. All addresses point to the same content, however the metadata and retention period information associated with a particular address typically differs.
i_contents_size	integer	R	Actual size of the content, including trailing spaces, stored in the corresponding index position of i_contents. **Note:** Subcontent objects created by server versions prior to 5.0 will have no value in this atribute. This property is not used if the subcontent object records a content address.
i_partition	integer	S	Identifies the partition where the object resides.

SysObject

Purpose

Serves as the parent type of the most commonly used objects in the Documentum system.

Description

Supertype: Persistent Object
Subtypes: Document, Folder, Output Device, Category Class, and so forth
Internal name: dm_sysobject
Object type tag: 08

The SysObject type is the parent type of the most commonly used objects in the Documentum system. The SysObject type has three important properties, represented by properties, that it passes on to all its subtypes. These properties are:

- A SysObject accepts security permissions. Properties defined for the SysObject allow you to set permissions on the object.
- A SysObject, unless it is a cabinet, can belong to a folder.
- A SysObject can own one or more content objects.

Properties

Tables Table 143, page 396, through Table 151, page 407, list the properties defined for the type. The tables group the SysObject properties by use or function:

- General properties, described in Table 143, page 396
- Folder-related properties, described in Table 144, page 399
- Virtual document-related properties, described in Table 145, page 400
- Content-related properties, described in Table 146, page 401
- Version-control properties, described in Table 149, page 405
- Event-related property, described in Table 150, page 406
- Lifecycle-related properties, described in Table 151, page 407
- Properties added by an system-defined aspect, described in Table 152, page 407

Table 152, page 407, lists the properties attached to the objects of the type through system-defined aspects.

General properties

The general properties provide a general description of the object.

Table 143. General properties of SysObjects

Property	Datatype	Single or repeating	Description
a_application_type	string(32)	S	Reserved for use by Documentum products.
a_archive	Boolean	S	Used internally.
a_extended _properties	string(32)	R	Values of the relation_name property in a dm_relation object or subtype of a dm_relation. This is used in conjunction with the r_object_id of the sysobject to identify the relation object that contains the SysObject's extended properties.
a_is_hidden	Boolean	S	Indicates if this object is visible to end users
a_is_template	Boolean	S	Indicates whether the object is a template
a_last_review_date	Date	S	Date of the last review of the object. This value is set automatically by some Documentum client products.
a_retention_date	date	S	Used internally.
a_special_app	string(32)	S	Reserved for use by Documentum products.
a_status	string(16)	S	Reserved for use by Documentum products.
authors	string(48)	R	List of the authors for the object. This is user-defined.
i_is_deleted	Boolean	S	If the object is the root version of a version tree, deleting the object sets this property to TRUE.
i_is_reference	Boolean	S	Indicates whether the object is a mirror object for a reference link to a remote object. TRUE means that the object is a mirror object.

Property	Datatype	Single or repeating	Description
i_contents_size	integer	R	Actual size of the content, including trailing spaces, stored in the corresponding index position of i_contents. **Note:** Subcontent objects created by server versions prior to 5.0 will have no value in this atribute. This property is not used if the subcontent object records a content address.
i_contents_size	integer	R	Actual size of the content, including trailing spaces, stored in the corresponding index position of i_contents. **Note:** Subcontent objects created by server versions prior to 5.0 will have no value in this attribute. This property is not used if the subcontent object records a content address.
i_partition	integer	S	Identifies the partition where the object resides.
i_property_bag	string(2000)	S	The property stores the names and values of non-qualifiable properties. It is also used to store the names and values of aspect properties if the properties are added to the aspect with the OPTIMIZEFETCH option.
i_retainer_id	ID	R	Object IDs of the retainer representing the retention policy that controls the object.
keywords	string(48)	R	List of user-defined keywords for the object

Property	Datatype	Single or repeating	Description
language_code	string(5)	S	Five-character code, in the format *xx_yy*, indicating the language in which the document is written and the country of origin. The first two characters (*xx*) contain the language code and the final two (*yy*), the country code. Appendix A, Language and Country Codes contains recommended language and country codes.
object_name	string(255)	S	Name of the object. This is user-defined. **Note:** If you are using a Documentum client, this is the name that appears under the object's icon.
r_access_date	date	S	Contains the date and time when this object was last accessed by a Getfile or Print method
r_alias_set_id	ID	S	Object ID of the alias set associated with the sysobject.
r_aspect_name	string(64)	R	Reserved for internal use Manipulation of this property by users or external applications is not supported.
r_creation_date	date	S	Date the object was created. This is set by the server.
r_modifier	string(32)	S	Name of the user who made the last modification.
r_modify_date	time	S	Date the object was last modified. This is set by the server.
r_object_type	string(32)	S	The object's type, for example, dm_sysobject or dm_document, set when the object is created.

Property	Datatype	Single or repeating	Description
r_property_bag	string(2000)	R	This property stores overflow from the i_property_bag property.
subject	string(192)	S	Subject of the object. This is user-defined.
title	string(400)	S	Title of the object. This is user-defined.

Folder-related properties

The folder properties provide information for managing folders.

Table 144. SysObject folder-related properties

Property	Datatype	Single or repeating	Description
governing_room_id	ID	S	Object ID of the room that governs this object.
i_folder_id	ID	R	Object IDs of all folders linked to the object.
i_has_folder	Boolean	S	Flag indicating whether this object is the CURRENT object in the version tree. T means the object has the CURRENT version label. F means it does not.
r_link_cnt	integer	S	Number of objects linked to the folder. **Note:** This property has two uses in the SysObject object type. One, described here, for folders and one, described in Table 145, page 400 , for virtual documents.
r_order_no	integer	R	No longer used. Preserved for backwards compatibility.
i_reference_cnt	integer	S	Number of folder references made to this object.
r_component_label	string(32)	R	No longer used. Preserved for backwards compatibility.

Property	Datatype	Single or repeating	Description
r_composite_id	ID	R	No longer used. Preserved for backwards compatibility.
r_composite_label	string(32)	R	No longer used. Preserved for backwards compatibility.

Virtual document-related properties

These properties provide information for the management of virtual documents.

Table 145. Virtual document-related properties

Property	Datatype	Single or repeating	Description
a_compound _architecture	string(16)	S	Used by Virtual Document Manager to determine if the object can be structurally changed through VDM. An empty string indicates that the object can be changed.
a_link_resolved	Boolean	S	Used internally.
resolution_label	string(24)	S	Contains the default version label used to resolve late-bound nodes of a virtual document.
r_assembled _from_id	ID	S	Object ID of the virtual document that was the source of the assembly associated with a document.
r_frzn_assembly_cnt	integer	S	Contains a count of the number of frozen assemblies that contain this object.
r_has_frzn_ assembly	Boolean	S	Indicates that the document's assembly is frozen.
r_is_virtual_doc	integer	S	Indicates whether the SysObject is a virtual document. If the value is 1, the object is a virtual document. If the value is 0, the object is not a virtual document unless the r_link_cnt is greater than 0.

Property	Datatype	Single or repeating	Description
r_link_cnt	integer	S	Number of components in the virtual document.
r_link_high_cnt	integer	S	Records the current maximum order number assigned to a component.

Content-related properties

The content-related properties provide information about the content of sysobjects.

Table 146. SysObject content-related properties

Property	Datatype	Single or repeating	Description
a_content_type	string(32)	S	File format of the object's content. **Note:** For some system-created renditions, Content Server sets this value to dm_internal.
a_full_text	Boolean	S	Indicates whether the document is marked for full-text indexing.
a_is_signed	Boolean	S	Indicates whether the object has been digitally signed. The property is set to T by the client application after the user signs the object.
a_storage_type	string(32)	S	Identifies the storage area for content files associated with the object. This is the name of the storage object representing the storage area.
i_contents_id	ID	S	Object ID of the content object for an object that has only one content. This is not used if an object has multiple content. In such cases, the Content Facility manages the relationships between the object and its multiple content.

Property	Datatype	Single or repeating	Description
r_content_size	integer	S	Size, in bytes, of the first content file associated with the document. This property cannot record content sizes greater than 2GB. Examine r_full_content_size to obtain the size of content larger than 2GB.
i_retain_until	date	S	Most restrictive retention date among all content associated with this object. This property is only set if the content has a retention period. Retention periods are set by a retention policy or through storage in a retention—enabled storage area with a retention period.
r_full_content _size	double	S	Size, in bytes, of the first content file added to the SysObject.
r_page_cnt	integer	S	Number of content files associated with the object.

Web-content properties

The Web-content properties are used by the Documentum applications that manage SysObjects that are published on Web sites. These properties are not intended for use by user applications.

Table 147. Web-content properties

Property	Datatype	Single or repeating	Description
a_category	string(64)	S	Used internally to manage the application.

Property	Datatype	Single or repeating	Description
a_effective_ date	date	R	The date on which the document can be published to the Web site. The value at a particular index position applies to the Web site named at the corresponding index position in a_effective_label.
a_effective_ flag	Boolean	R	Indicates whether a pending expiration notice was sent to the document's owner. The value at a particular index position applies to the Web site named at the corresponding index position in a_effective_label.
a_effective_ label	string(32)	R	User-defined, typically a symbolic label. The WebCache export operation examines the values in the a_effective_date and a_expiration_date properties of all documents whose a_effective_label property value matches the webc config effective_label property. If unspecified, effective labels are not enforced.
a_expiration_ date	date	R	The date at which the document is to be removed from the Web site. The value at a particular index position applies to the Web site named at the corresponding index position in a_effective_label.
a_publish_ formats	string(32)	R	Lists the object's renditions that are to be exported to the WebCache repository. The system will export these renditions plus those defined in the source_formats property

Property	Datatype	Single or repeating	Description
			of the webc config object. This is an optional property. If unspecified, only renditions defined in the webc config object are published.

Security-related properties

The security-related properties provide information that allows the server to enforce security on the object. Changes to security-related properties may create a new ACL for the object.

Table 148. SysObject security-related properties

Property	Datatype	Single repeating	Description
a_controlling _app	string(32)	S	Identifies the application or applications that can modify this object. If NULL, then any application can modify the object.
acl_domain	string(32)	S	Identifies the domain of the ACL associated with the object. The value will be either the user who created the ACL or, for system-level ACLs, the name of the repository owner.
acl_name	string(32)	S	The object name of the associated ACL.
group_name	string(32)	S	Group to which this object belongs.
group_permit	integer	S	Object-level permission assigned to the object's group for this object.
i_cabinet_id	ID	S	Object ID of the cabinet that is the object's primary storage location.
owner_name	string(32)	S	Name of the object's owner. This can be a user or group name.
owner_permit	integer	S	Object-level permission assigned to the owner for this object.

Property	Datatype	Single repeating	Description
r_creator_name	string(32)	S	Name of the object's creator.
r_is_public	Boolean	S	Value is computed based on the object's ACL. TRUE means that dm_world has at least Read permission. FALSE indicates that dm_world does not have Read or higher permission.
r_lock_machine	string(80)	S	Name of the client machine on which a user is working when he or she locks an object (by a checkout or branch operation). If the server is unable to resolve the name, r_lock_attribute is set to the client machine's IP address instead.
world_permit	integer	S	Object-level permission assigned to the world (all users except owner and group members) for this object.

Version-related properties

The version-related properties provide information that lets the server manage and track the versions of an object.

Table 149. SysObject version-related properties

Property	Datatype	Single or repeating	Description
i_antecedent_id	ID	S	Object ID of the object's parent version.
i_branch_cnt	integer	S	Number of branches on the version tree that contains the object.
i_chronicle_id	ID	S	Object ID of the root object of the version tree that contains this object.

Property	Datatype	Single or repeating	Description
i_direct_dsc	Boolean	S	Indicates whether the object has any direct descendants (versions derived directly from this object). The default is FALSE.
i_latest_flag	Boolean	S	Indicates whether this version is the most recent version of the object on a particular branch in the version tree.
log_entry	string(120)	S	Comments specified by the user.
r_frozen_flag	Boolean	S	Indicates whether the object is unchangeable because it was specifically frozen. The default is FALSE.
r_immutable_flag	Boolean	S	Indicates whether the object can be changed. The default is FALSE.
r_lock_date	date	S	Date that this object was locked.
r_lock_owner	string(32)	S	Name of the user who locked the object.
r_version_label	string(32)	R	List of the version labels associated with the object. The first position in this property holds the object's implicit version label. The remaining rows contain the object's symbolic version labels.

Event-related properties

The SysObject type has the following event-related property.

Table 150. SysObject event-related property

Property	Datatype	Single or repeating	Description
r_has_events	Boolean	S	Indicates whether any users have registered to receive events for this object. The default is FALSE.

Lifecycle-related properties

The lifecycle-related properties provide information about associated lifecycles (policy objects).

Table 151. SysObject lifecycle-related properties

Property	Datatype	Single or repeating	Description
r_policy_id	ID	S	Object ID of the associated policy object
r_current_state	integer	S	State number of the current state of the object in the lifecycle.
r_resume_state	integer	S	The state number to which the object is resumed if r_current_state identifies an exception state.

Attributes added by system-defined aspects

The properties in this table are added to Sysobject instances or the object type by a system-defined aspect. Whether the properties are available for use and examination will depend on whether your environment has installed the EMC Documentum product that adds the aspect and its attributes.

Table 152. SysObject aspect-added properties

Property	Datatype	Single or repeating	Description
a_gov_room_id	ID	S	Object ID of the dmc_room in which this object is stored, and thereby governed by.

Sysprocess Config

Purpose

Reserved for internal use.

Description

Supertype: SysObject
Subtypes: JMS Config
Internal name: dm_sysprocess_config
Object type tag: 08

Do not use this type. It is reserved for internal use only.

Taxonomy

Purpose

Records the default values for the properties in the categories of a taxomony structure in the repository.

Description

Supertype: Category
Subtypes: None
Internal name: dm_taxonomy
Object type tag: 0b

A taxonomy object records the default values for the properties in the categories of a taxomony structure in the repository.

Properties

Table 153, page 409, lists the property defined for a taxonomy.

Table 153. Properties defined for the taxonomy type

Property	Datatype	Single or repeating	Description
taxonomy_ version	string(32)	S	Current version of the installed taxonomy.

TCF Activity

Purpose

Records a sequence of actions to be executed on entry for a lifecycle state.

Description

Supertype: Document
Subtypes: None
Internal name: dmc_tcf_activity
Object type tag: 09

A tcf activity object represents a sequence of actions to execute when an object enters a particular state. TCF activity objects are created when a user identifies one or more actions for a state when defining a lifecycle state in Lifecycle Editor. You cannot create these objects manually. The actions and their parameters are recorded in an XML file that is stored as the content of the tcf activity object.

Properties

Table 154, page 410, lists the properties defined for the type.

Table 154. Properties defined for the TCF activity type

Property	Datatype	Single or repeating	Description
act_identifier	string(256)	S	Character string identifier that identifies this tcf activity object.
act_template_ identifier	string(256)	S	Used internally

TCF Activity Template

Purpose

Represents one lifecylce state action.

Description

Supertype: Document
Subtypes: None
Internal name: dmc_tcf_activity_template
Object type tag: 09

A tcf activity template object represents one action that may be executed when an object enters a lifecycle state. A suite of TCF activity template objects is installed with the Lifecycle Editor. The content files associated with the objects are XML files that contain the template's string identifier and the parameters that must be supplied to execute the actions. You cannot create these objects manually.

Properties

Table 155, page 411, lists the properties defined for the type.

Table 155. Properties defined for the TCF activity template type

Property	Datatype	Single or repeating	Description
act_template_ identifier	string(256)	S	Character string identifier that identifies this tcf activity template object.
template_groups	string(32)	S	Reserved for future use

Transition Condition

Purpose

Records a route case condition expression for an automatic transition of a workflow activity.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_transition_condition
Object type tag: 00

A transition condition object records a conditional expression in a route case condition. You cannot create these objects directly. They are created when an addConditionRouteCase method (defined for the IDfActivity interface) is executed by a DFC at version level 5.3 FCS. This method is executed by Process Builder to save an activity's route case conditions when any one or more of the route cases contains an XPath expression.

Note: When the addConditionRouteCase method is called by a DFC 5.3 SP1 or later, the method creates routecase condition objects, instead of transition condition objects, to record the expression.

Properties

Table 156, page 412, lists the properties defined for the type.

Table 156. Properties defined for the transition condition type

Property	Datatype	Single or repeating	Description
r_aspect_name	string(64)	R	Used internally
r_attribute_name	string(32)	S	Name of the property referenced in the expression, if any
r_boolean_value	Boolean	S	Value to be used in the comparison if the datatype identified in r_value_type is Boolean. This is not set if unless the datatype is Boolean.

Property	Datatype	Single or repeating	Description
r_double_value	Double	S	Value to be used in the comparison if the datatype identified in r_value_type is double. This is not set if unless the datatype is double.
r_id_value	ID	S	Value to be used in the comparison if the datatype identified in r_value_type is ID. This is not set if unless the datatype is ID.
r_int_value	integer	S	Value to be used in the comparison if the datatype identified in r_value_type is integer. This is not set if unless the datatype is integer.
r_object_alias	string(128)	S	Name of the package, or manifest values referring to the workflow or work item
r_relational_op	integer	S	The relation operator in the condition. Valid values are: 0, meaning = 1, meaning <> 2, meaning < 3,meaning > 4, meaning <= 5, meaning >=

Property	Datatype	Single or repeating	Description
r_repeating_attr_flag	integer	S	Indicates whether the property named in r_attribute_name is a repeating property and if it is a repeating property, which values to examine when evaluating the condition. Valid values are: -1, meaning the property is not a repeating property 0, meaning ANY 1, meaning ALL 2, meaning FIRST 3, meaning LAST
r_string_value	string(1024)	S	Value to be used in the comparison if the datatype identified in r_value_type is string. This is not set if unless the datatype is string.
r_time_value	Date	S	Value to be used in the comparison if the datatype identified in r_value_type is Date. This is not set if unless the datatype is Date.
r_value_type	integer	S	Data type of the value in the relational expression. The data type is expressed as an IDfValue constant.
r_xpath_datatype	string(64)	S	The xschema's built-in datatype name. This is set only if the condition includes an XPath expression.

Property	Datatype	Single or repeating	Description
r_xpath_expression	string(1024)	S	An XPath expression. This is set only if the condition includes an XPath expression.
r_xpath_value	string(1024)	S	Literal value used in the XPath transition condition evaluation. The value is in the format in which it is found in the XML document. This is set only if the condition includes an XPath expression.

Type

Purpose

Stores structural information about an object type in the repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_type
Object type tag: 03

A type object stores structural information about an object type in the repository. The object types in Content Server are themselves represented as types (that is, each type is an object of type dm_type). Content Server does not allow you to create new objects of type dm_type directly. Instead, when you create a new type or subtype, Content Server automatically creates an object of type dm_type to describe your new type. (To create a new type, use the DQL CREATE TYPE statement.)

The values in the repeating properties across one index position represent one property defined for the object type.

Properties

Table 157, page 416, lists the properties defined for the type.

Table 157. Properties defined for the type object type

Property	Datatype	Single or repeating	Description
attr_count	integer	S	Number of properties in the type (includes defined and inherited)
attr_identifier	integer	R	Uniquely identifies the property within the type hierarchy. Values start from 1.
attr_length	integer	R	Indicates the length of those properties that are string-valued.
attr_name	string(40)	R	Names of the type's properties
attr_repeating	Boolean	R	Indicates if the properties are repeating

Property	Datatype	Single or repeating	Description
attr_restriction	integer	R	Identifies any restrictions on the property. Valid values are: 0, meaning no special restrictions 1, meaning the property is a non-qualifiable property
attr_type	integer	R	Contains integer values representing the datatype of the properties. Valid values are: 0, Boolean 1, Integer 2, String 3, ID 4, Date and Time 5, Double
i_type	integer	S	Used internally
info	ID	S	Object ID of the type's associated type info object
name	string(27)	S	Name of the type
next_attr_identifier	integer	S	Next available property identifier in the type hierarchy
owner	string(40)	S	Name of the type's owner (creator)
r_object_id	ID	S	Object ID of the type
r_index_attr	ID	S	Object ID of the index object describing any indexes built on the type's repeating properties.
shared_parent_name	string(40)	S	For lightweight object types, this is the name of the parent object type. The parent is the type with whose instances the lightweight type instances are sharing properties. For all other types, this is the same as the value in super_name.

Property	Datatype	Single or repeating	Description
start_pos	integer	S	Position of the first non-inherited property
super_name	string(40)	S	Name of the type's supertype
s_index_attr	ID	S	Object ID of the index object describing any indexes built on the type's single-valued properties.
type_category	integer	S	Identifies the category to which the type belongs. Valid values are: • 0, standard object type • 1, aspect property object type • 2, shareable object type • 4, lightweight object type • 8, data table type
views_valid	Boolean	S	Indicates if the views for the type are valid. This is used internally during recovery operations.

Type Info

Purpose

Sstores non-structural information about an object type.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_type_info
Object type tag: 2e

A type info object stores non-structural information about an object type. Storing non-structural information separately from the type's structure definition (the dm_type object) enhances the performance when a type is altered.

Properties

Table 158, page 419, lists the properties defined for the type.

Table 158. Properties defined for the type info type

Property	Datatype	Single or repeating	Description
acl_domain	string(32)	S	The domain (owner) of the ACL associated with the type definition. The value is either the name of the user who created the ACL or, for system ACLs, the name of the repository owner.
acl_name	string(32)	S	The object name of the associated ACL.
default_aspects	string(64)	R	List of aspects defined for the object type. These are identified by the name of the aspect as defined in the aspect's object_name property of its dmc_aspect_type object.
default_group	string(27)	S	Default group defined for the type.

Property	Datatype	Single or repeating	Description
default_group _permit	integer	S	Default object-level permission defined at the group level for the type.
default_owner _permit	integer	S	Default object-level permission defined at the owner level for the type.
default_storage	ID	S	Object ID of a storage object of type dm_store. This identifies the default storage type for the contents associated with any object of this type.
default_world _permit	integer	S	Default object-level permission defined at the world level for the type.
ftindex_attrs	string(27)	R	For lightweight object types, contains a list of properties that are indexed. The list does not contain the basic SysObject properties that are indexed by default (acl_name, acl_domain, owner_name, and r_object_type) if the object type has fulltext support. This property is not currently used for any object type except lightweight object types.
i_type_features	integer	S	The value in this property is a bitmask used internally by Content Server to identify several characteristics of the object type.
locally _managed	Boolean	R	Indicates whether a given property is locally or globally managed if the repository is participating in a federation. The value at each index level corresponds to the property named in attr_names in the type's dm_type object.

Property	Datatype	Single or repeating	Description
r_object_id	ID	S	Object ID of the type info object
r_orig _declaration	Boolean	R	Indicates whether a given property is defined for the type or inherited. TRUE means it is defined for the type. The value at each index level corresponds to the property named in attr_names in the type's dm_type object.
r_supertype	string(27)	R	List of all supertypes of the type.
r_type_id	ID	S	Object ID of the type object that contains the definition of the type.
r_type_name	string(27)	S	Name of the type represented by the type object identified in r_type_id.
type_cache_vstamp	integer	S	Records the value of the dm9_change_record.type_ change_count when the type was changed. When a type is changed, the value of type_change_count is first incremented, and then the new value is written to the type_cache_vstamp property.
type_override	ID	S	Object ID of the dm_aggr_domain object, if any, for the object type.
type_version	integer	S	Version number of the object type. This is incremented when type properties are added, changed, or deleted.

User

Purpose

Contains information about a user in the repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_user
Object type tag: 11

A user object contains information about a user in the repository. The information includes the user's default permissions, electronic mail address, default folder, and inbox object identifier. You must have Sysadmin or Superuser user privileges to create or drop a user or to activate or deactivate a user.

Properties

Table 159, page 422, lists the properties defined for the type.

Table 159. Properties defined for the user type

Property	Datatype	Single or repeating	Description
acl_domain	string(32)	S	Identifies the domain (owner) of the ACL associated with the user. The value is either the user who created the ACL or, for system ACLs, the name of the repository owner.
acl_name	string(32)	S	The object name of the associated ACL.
alias_set_id	ID	S	Object ID of the alias set object representing the user-level default alias set.

Property	Datatype	Single or repeating	Description
client_capability	integer	S	Indicates what level of use is expected of the user. Valid values are: • 0 or 1, for consumer • 2, for contributor • 4, for coordinator • 8, for system administrator The default value is 0.
deactivated_ip_addr	string(64)	S	Reserved for future use
deactivated_utc_time	Date	S	Reserved for future use
default_folder	string(200)	S	Identifies the user's default folder. The default is Temp.
description	string(255)	S	User-defined description of the user.
failed_auth_attempt	integer	S	Number of unsuccessful authentication attempts on behalf of the user. With the exception of the installation owner, this is set to 0 when a user is created. The default value for the installation owner is -1. Setting this to -1 for any user disables the feature for that user. If the feature is enabled for a user, the value is reset to 0 whenever the user is authenticated successfully.
first_failed_auth_utc_time	Date	S	Reserved for future use

Property	Datatype	Single or repeating	Description
globally_ managed	Boolean	S	Indicates whether the user object is managed globally or locally. The default is FALSE, meaning that it is locally managed. Requires at least Sysadmin privileges to change.
group_def _permit	integer	S	Default group permit. This is used to assign object-level permission at the group level to any object the user creates if no permission at that level is explicitly assigned.
home_docbase	string(120)	S	The user's home repository.
last_login_utc_time	Date	S	The time, in UTC format, of the user's last login to a repository. This property is not set if the user is the dmadmin user or if the value in the property has been set to NULLDATE . **Note:** Only a superuser can set this to NULLDATE.
owner_def _permit	integer	S	Default owner permit. This is used to assign object-level permission at the owner level to any object the user creates if no permission at that level is explicitly assigned.
r_has_events	Boolean	S	Whether someone has registered the user for auditing.
r_is_group	Boolean	S	Indicates if the user represents a group or an individual user.
r_modify_date	date	S	Time and date of the last change to the user object.
r_object_id	ID	S	Object ID of the user.

Property	Datatype	Single or repeating	Description
restricted_folder_ids	ID	R	Object IDs of the cabinets or folders which the user can access. The user can access these objects and their subfolders. If set, the user may access only these folders. This is a local property, not a global property.
user_address	string(80)	S	User's electronic mail address. This is a required property.
user_admin	string(32)	S	Identifies who is allowed to modify the user object. Valid values are an individual user name or a group name.
user_db_name	string(32)	S	User's user name in the underlying RDBMS. This must consist of ASCII characters.
user_delegation	string(32)	S	The name of a user to whom to delegate work items.
user_global_ unique_id	string(255)	S	Reserved for future use
user_group _name	string(32)	S	The default group. This is used when the user creates an object as part of the determination of the object's default group.
user_initials	string(16)	S	Reserved for future use
user_ldap_dn	string(255)	S	User's Distinguished Name in LDAP.

Property	Datatype	Single or repeating	Description
user_login_domain	string(255)	S	Name of the Windows domain against which this user is authenticated. If the user is an LDAP user, the property stores the object name of the LDAP config object representing the LDAP directory against which the user is authenticated. This may be blank.
user_login_name	string(80)	S	Name used to authenticate the user. This property must be set. If a domain is not required in the repository, user_login_name must be unique within the repository. If a domain is required, the combination of user_login_name and user_login_domain must be unique.
user_name	string(32)	S	Content Server user name. This can be an individual user or a group name. It must be unique among the user and group names in the repository. The name must consist of characters compatible with the server_os_codepage of the Content Server. This is a required property.
user_os_domain	string(15)	S	Windows domain name associated with user.

Property	Datatype	Single or repeating	Description
user_os_name	string(32)	S	The name of the user's operating system account, if any. This must consist of ASCII characters if specified.
user_password	string(256)	S	Encrypted password for the user. This is set only if the user's user_source is set to "inline password". When displayed, it displays as 16 asterisks (*).
user_privileges	integer	S	Identifies the user's user privileges. This value is the sum of the values corresponding to the user privileges defined for the user. For example, if a user has Create Type and Create Cabinet user privileges, the value is 3. The privileges and their values are: None (0) Create Type (1) Create Cabinet (2) Create Group (4) Sysadmin (8) Superuser (16) For more information about these levels, refer to the *Content Server Administration Guide*.

Property	Datatype	Single or repeating	Description
user_source	string(16)	S	Indicates the source of the user's authentication. Valid values are: **LDAP**, meaning the user is authenticated through the LDAP directory server. **unix only**, meaning the user is authenticated by standard UNIX mechanism; the domain isn't used. **domain only**, meaning the user is authenticated against the Windows domain; the UNIX password file isn't used. **unix first**, meaning the user is authenticated first by the standard UNIX mechanism. If that fails, the user is authenticated against the Windows domain. **domain first**, meaning the user is authenticated first against the Windows domain. If that fails, the user is authenticated using the standard UNIX mechanism. **inline password**, meaning that the user is authenticated against the password stored in the user_password property. *plugin_identifier*, meaning the user is authenticated using the plug-in identified by the identifier. The identifier for the plug-ins provided with Content Server are dm_rsa and dm_netegrity. **Note:** The UNIX- and domain-related values are effective only if an auth config object exists in the repository.
user_state	integer	S	Indicates the user's activation state. Valid values are:

Property	Datatype	Single or repeating	Description
			0, indicating a user who can log in 1, indicating a user who cannot log in 2, meaniing a user who is locked 3, meaning a user who is locked and inactive
user_web_page	string(255)	S	Reserved for future use
user_xprivileges	integer	S	The user's extended user privileges. The value is the sum of the values corresponding to the extended user privileges defined for the user. For example, if a user has Config Audit and Purge Audit privileges, the value is 24. The privileges and their values are: 8, Config Audit 16, Purge Audit 32, View Audit
workflow_ disabled	Boolean	S	Indicates the user's availability for work item assignment. Valid values are: 0, meaning available 1, meaning not available The default is 0 (available).
world_def _permit	integer	S	Default world permit. This is used to assign object-level permission at the world level to any object the user creates if no permission at that level is explicitly assigned.

Validation Descriptor

Purpose

Stores configuration information used by DFC and ACS or BOCS servers.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_validation_descriptor
Object type tag: 08

Validation descriptor objects record configuration information that is shared between DFC and ACS or BOCS servers. These objects are created and managed internally by Content Server.



Caution: Do not modify this type or instances of this type.

Properties

Table 160, page 430, lists the properties defined for this type.

Table 160. Properties defined for the validation descriptor type

Property	Datatype	Single or repeating	Description
descriptor_version	string(16)	S	Used internally
parameter_names	string(255)	R	Used internally

Validation Module

Purpose

Represents a business module that corresponds to a Docbasic expression defined in a func expr object.

Description

Supertype: Module
Subtypes: None
Internal name: dmc_validation_module
Object type tag: 0b

A validation module represents the Java equivalent of the Docbasic expressions identified in func expr objects. Each validation module stores the Java equivalents for all Docbasic expressions defined for check constraints or for conditional value assistance for a particular object type. A validation module object is associated with the equivalent func expr objects by a relationship whose name is dmc_expr_to_module. The validation module is related to the dmc_jar object that contains the compiled code through a relationship named dmc_module_to_jar.

These objects are created only when a Docbasic expression is manually converted to a Java expression. Documentum 6 and later does not require manual conversion, as the conversion happens at runtime automatically. Manual conversion is required only for clients prior to Documentum 6.

Properties

Table 161, page 431, lists the properties defined for the type.

Table 161. Properties defined for the validation module type

Property	Datatype	Single or repeating	Description
expr_code_object_id	ID	S	Object ID of the dmi_expr_code object that contains the original Docbasic expressions file and P-code file for the expressions. (The expressions are compiled into one file and the P-code file generated from that file. Both the source file and the P-code file are stored as content of the expr code object.)

Property	Datatype	Single or repeating	Description
implementation_ enabled	Boolean	S	Whether Java evaluation is enabled for the expressions implemented by this module. T means that Java evaluation is enabled; F means that Java evaluation is not enabled. The value in this property takes precedence over the expr_enabled setting in the associated dmc_validation_relation object.
referenced_type_name	string(27)	R	Name of the object type to which the expression is bound. Currently, only one value in this property is supported.

Validation Relation

Purpose

Relates a func expr object to a validation module object.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_validation_relation
Object type tag: 37

A validation relation object relates a func expr object to the validation module that represents the Java equivalent of the Docbasic expression in the func expr object. The relation_name value for all validation relation objects is dmc_expr_to_module.

Properties

Table 162, page 433, lists the properties defined for the type.

Table 162. Properties defined for the validation relation type

Property	Datatype	Single or repeating	Description
expr_enabled	Boolean	S	Whether Java evaluation is enabled for the expression represented by the related validation module. T means Java evaluation is enabled; F means that Java evaluation is not enabled. The setting in this property can be overridden by the setting of the implementation_enabled property in the dmc_validation_module object.
has_implementation	Boolean	S	Indicates whether actual Java code exists for the expression. This is set to F if the Docbasic expression was not successfully

Property	Datatype	Single or repeating	Description
			migrated to Java code for the validation module. If this is F, then expr_enabled is F and cannot be reset to T.

Value Assist

Purpose

Describes the value assistance provided through the data dictionary for an property.

Description

Supertype: Persistent Object
Subtypes: Value List, Value Query, Value Func
Internal name: dm_value_assist
Object type tag: 5a

A value assist object describes the value assistance provided through the data dictionary for an property. Value assist objects are created and managed by Content Server and cannot be created by users.

Properties

Table 163, page 435, lists the properties defined for the type.

Table 163. Properties defined for the value assist type

Property	Datatype	Single or repeating	Description
complete_list	Boolean	S	Indicates whether the values returned by value assistance are the only acceptable values for the property. TRUE means that they are the only acceptable values. FALSE means that other values are acceptable.

Property	Datatype	Single or repeating	Description
domain_type	integer	S	Indicates the datatype for the values returned by value assistance. Valid values are: 0, meaning Boolean 1, meaning Integer 2, meaning String 3, meaning ID 4, meaning Time/date 5, meaning Double
object_name	string(32)	S	Currently unused.
parent_id	ID	S	Object ID of the aggr domain object to which the property belongs.
use_as_constraint	Boolean	S	Currently unused.
value_estimate	integer	S	The number of values expected to be returned by value assistance.

Value Func

Purpose

Describes the procedural form of value assistance.

Description

Supertype: Value Assist
Subtypes: None
Internal name: dm_value_func
Object type tag: 5d

A value func object describes the procedural form of value assistance. Value func objects are created and managed by Content Server and cannot be created by users.

Properties

Table 164, page 437, lists the properties defined for the type.

Table 164. Properties defined for the value func type

Property	Datatype	Single or repeating	Description
func_expression	ID	S	Object ID of the func expr object representing the user function to call.
value_separator	string(1)	S	Defines the character used as a separator in the value list returned by the user function. The default is a comma (,).

Value List

Purpose

Contains the valid values for the list form of value assistance.

Description

Supertype: Value Assist
Subtypes: None
Internal name: dm_value_list
Object type tag: 5b

A value list object contains the valid values for the list form of value assistance. Value list objects are created and managed by Content Server and cannot be created by users.

Properties

Table 165, page 438, lists the property defined for the type.

Table 165. Properties defined for the value list type

Property	Datatype	Single or repeating	Description
valid_values	string(255)	R	List of string values representing literals of the appropriate data type. Each literal must be unique within the list. The list values are displayed to users in the order in which they appear in the property.

Value Query

Purpose

Contains information for the query form of value assistance.

Description

Supertype: Value Assist
Subtypes: None
Internal name: dm_value_query
Object type tag: 5c

A value query object contains information for the query form of value assistance. Value query objects are created and managed by Content Server and cannot be created by users.

Properties

Table 166, page 439, lists the properties defined for the type.

Table 166. Properties defined for the value query type

Property	Datatype	Single or repeating	Description
allow_caching	Boolean	S	Indicates whether the queries defined for value assistance can be cached. TRUE means that the queries can be cached. FALSE means that the query should be rerun each time it is needed.
query_attribute	string(32)	S	The property from the query's selected values list that will provide the data to be displayed. The default is the first selected value.
query_string	string(255)	R	The DQL SELECT statement that returns the value assistance values. If it is longer than 255 characters, the first 255 characters appear in query_string[0], the second 255

Property	Datatype	Single or repeating	Description
			characters in query_string[1], and so forth. The continuation character is an underscore (_).

Vstamp

Purpose

Used internally by the system, at start-up, to validate the consistency of the server and repository.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_vstamp
Object type tag: 1e

A vstamp object is used internally by the system, at start-up, to validate the consistency of the server and repository.

Properties

Table 167, page 441, lists the properties for the object type.

Table 167. Properties defined for the vstamp type

Property	Datatype	Single or repeating	Description
i_application	string(64)	S	Name of the client program
i_stamp	integer	S	Value used internally to validate repository consistency

WF Attachment

Purpose

Describes an attachment to a workflow.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_wf_attachment
Object type tag: 00

A wf attachment object describes an object added at runtime to a workflow or workitem as an attachment.

Properties

Table 168, page 442, lists the properties defined for the type.

Table 168. Properties defined for the WF attachment type

Property	Datatype	Single or repeating	Description
r_component_id	ID	S	Object Id of the object attached to the workflow
r_component_name	string(80)	S	Name of the object attached to the workflow
r_component_type	string(40)	S	Object type (actual or supertype) of the object attached to the workflow
r_creation_date	Date	S	Date and time the attachment was attached to the workflow
r_creator_name	string(32)	S	Name of the user who attached the object to the workflow
r_workflow_id	ID	S	Object ID of the workflow to which the object is attached.

WF Package Report Parent

Purpose

The parent type of workflow package report types.

Description

Supertype: None
Subtypes: None
Internal name: dmc_wpr_parent
Object type tag: 00

Properties

Table 169, page 443, lists the property defined for the type.

Table 169. Property defined for the WF package report parent type

Property	Datatype	Single or repeating	Description
act_name	string(128)	S	Contains the name of the activity where the package is defined
act_sequenceno	integer	S	Contains the sequence number of the activity instance
component_name	string(255)	S	Contains the name of the component contained in the package
package_name	string(128)	S	Contains the name of the package
performer_name	string(32)	S	Contains the name of the performer who completes the activity instance
process_id	ID	S	Contains the ID of the process (dm_process object) where the package is defined
process_name	string(255)	S	Contains the name of the process defined in process_id

Property	Datatype	Single or repeating	Description
time_stamp_utc	Time	S	Contains the time the report entry is created
type_info_id	ID	S	Contains the ID of the type info object (dmc_wf_package_type_info object) of the type used in the workflow package
workflow_id	ID	S	Contains the ID of the workflow instance
workitem_id	ID	S	Contains the ID of the workitem instance

WF Package Schema

Purpose

Stores the URI of a workflow package schema.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_wf_package_schema
Object type tag: 37

A wf package schema object stores the URI of a schema associated with a package whose component is an XML file. It is used to validate any XPath expression referencing that file in a transition condition for an activity. WF package schema objects are created when the package is defined using Process Builder. The object type is installed by a script when Content Server is installed.

Properties

Table 170, page 445, lists the property defined for the type.

Table 170. Property defined for the WF package schema type

Property	Datatype	Single or repeating	Description
schema_uri	string(255)	S	URI of a schema

WF Package Skill

Purpose

Associates a skill level required for a workflow package with a workflow.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_wf_package_skill
Object type tag: 37

A wf package skill object identifies the skill level a user must have to acquire or be assigned the task associated with a particular package. This feature is effective only when applied to tasks on workqueues. WF package skill objects are created at runtime and associated with the workflow.



Caution: Do not modify this object type nor instances of the object. These objects are created, maintained, and destroyed internally, as needed.

Properties

Table 171, page 446, lists the properties defined for the type and the inherited properties whose settings are particular to instances of this type.

Table 171. Properties defined for the wf package skill type

Property	Datatype	Single or repeating	Description
child_id	ID	S	This is not set.
child_label	string(32)	S	There is no label set for instances of this type.
description	string(250)	S	There is no description provided for instances of this type.
package_name	string(128)	S	Name of the package that requires the skill level identified in skill_level.
parent_id	ID	S	Object ID of the workflow with which the package is associated

Property	Datatype	Single or repeating	Description
permanent_link	Boolean	S	Set to T (TRUE).
relation_name	string(32)	S	Set to dmc_wf_package_skill
skill_level	integer	S	An integer value identifying the required skill level. Refer to the Process Builder documentation for information about the skill levels.

WF Package Type Info

Purpose

Identifies a type info object for a type used in a workflow package

Description

Supertype: dm_sysobject
Subtypes: None
Internal name: dmc_wf_package_type_info
Object type tag: 08

A wf package type info

Properties

Table 172, page 448, lists the properties defined for the type and the inherited properties whose settings are particular to instances of this type.

Table 172. Properties defined for the wf package type info type

Property	Datatype	Single or repeating	Description
attr_is_filterable	integer	R	Indicates whether the attribute defined at the corresponding index of reportable_attribute can be filtered or not
package_type_name	string(27)	S	Contains the name of the data type used in workflow package
report_type_name	string(27)	S	Contains the name of the report type corresponds to the type defined in package_type_name
reportable_attribute	string(40)	R	Contains names of reportable attributes in the type defined in package_type_name

WF Timer

Purpose

Records the configuration of a timer for a workflow activity.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_wf_timer
Object type tag: 00

A wf timer object describes a timer for a workflow activity. The wf timer objects are created automatically when the timer is instantiated. The object type is installed by a script when Content Server is installed.

Properties

Table 173, page 449, lists the properties defined for the type.

Table 173. Properties defined for the WF timer type

Property	Datatype	Single or repeating	Description
i_calendar_status	integer	S	reserved for future use
r_act_id	ID	S	Object ID of the activity for which this timer is defined
r_act_name	string(128)	S	Name of the activity for which this timer is defined
r_act_seqno	integer	S	Sequence number of the activity instance
r_action_id	ID	S	Object ID of the module config object associated with the timer
r_action_index	integer	S	The index to the r_pre_timer_action or r_post_timer_action property of the activity
r_calendar_id	ID	S	reserved for future use

Property	Datatype	Single or repeating	Description
r_timer	Date	S	Absolute date and time at which to trigger the timer
r_timer_name	string(64)	S	Name of the timer object as defined in the dm_activity object
r_timer_type	integer	S	Identifies what kind of timer this is. Valid values are: 0, meaning a pre-timer 1, meaning a post-timer 2, meaning a suspend-timer
r_workflow_id	ID	S	Object ID of the workflow that contains the activity instance.

Workflow

Purpose

Contains the runtime information about a workflow.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dm_workflow
Object type tag: 4d

A workflow object contains the runtime information about a workflow.

Properties

Table 174, page 451, describes the properties defined for the workflow type.

Table 174. Properties defined for the workflow type

Property	Datatype	Single or repeating	Description
correlation_identifer	string(128)	S	A string value that uniquely identifies the workflow instance
i_next_act_group_ seqno	integer	S	Indicates the next sequence number of an activity group instance For internal use only.
i_next_seqno	integer	S	Indicates the next sequence number of an activity instance.
i_performer_flag	integer	R	Indicates special run-time conditions such as extension.
initiate_act	string(128)	S	Name of the Initiate activity, if any, from which the workflow was initiated

Property	Datatype	Single or repeating	Description
instructions	string(255)	R	User-defined string displayed to users who are using the sendToDistributed workflow. Applications can reference this property. It must be set before the first Save operation on the workflow object.
object_name	string(128)	S	Contains the workflow name.
parent_act_name	string(128)	S	Name of the activity in the workflow identified in parent_id from which this workflow was started
parent_act_seqno	integer	S	Sequence number of the activity identified in parent_act_name.
parent_id	ID	S	Object ID of the workflow that contains the activity that started this workflow.
process_id	ID	S	Contains the object ID of process definition (the dm_process object) on which this workflow is based.
r_act_def_id	ID	R	Contains the object ID of the activity definitions (dm_activity objects) included in the process definition identified in process_id.
r_act_errorno	integer	R	Records the failing operation, if any. Used for error recovery.
r_act_name	string(128)	R	Contains the activity identifier as defined in the dm_process object.
r_act_seqno	integer	R	Contains the unique sequence number of an activity instance.

Property	Datatype	Single or repeating	Description
r_act_state	integer	R	Records the activity's current state. Valid values and their corresponding states are: 0, meaing dormant 1, meaning active 2, meaning finished 3, meaning halted 4, meaning failed
r_alias_set_id	ID	S	Records the object ID of the alias set used to resolve performer aliases when the workflow is created. This is a runtime copy of the alias set identified in perf_alias_set_id of the dm_process object.
r_complete_witem	integer	R	Records the number of completed work items.
r_creator_name	string(32)	S	Indicates the creator. Automatically set by the server.
r_last_performer	string(32)	R	Contains the activity performer who last marks his or her own work item as complete. Before any work item of an activity instance completes, this may contain the last performer of the previous activities.
r_last_witem_id	ID	R	Records the ID of the work item that was most recently marked as complete.

Property	Datatype	Single or repeating	Description
r_perf_act_name	string(128)	R	Contains the names of activities whose performers were chosen at workflow initiation or upon completion of another activity. An activity name appears once for each performer of the activity. The activity's performer appears at the corresponding index position in r_performers.
r_performers	string(32)	R	Contains the name of a user or group chosen at workflow initiation or upon completion of another activity as a performer for the activity at the corresponding index position in r_perf_act_name. If there are multiple performers for an activity, each performer appears at a different index position, and r_perf_act_name records the name of the activity in the corresponding index positions.
r_post_timer	Date	R	This property is obsolete in version 5.3. If the workflow and activity were started prior to upgrading to 5.3, this value is the absolute date and time when the activity should finish.
r_pre_timer	Date	R	This property is obsolete in version 5.3. If the workflow was started prior to upgrading to 5.3, this value is the absolute date and time when the activity should start.
r_repeate_invoke	Boolean	R	Indicates whether this activity can be triggered multiple times.

Property	Datatype	Single or repeating	Description
r_runtime_state	integer	S	Indicates the current state of the workflow. Values are: 0, meaing dormant 1, meaning running 2, meaning finished 3, meaning halted 4, meaning terminated
r_start_date	date	S	Represents the start time of the instance, set by the server when the workflow enters the running state.
r_target_act_seqno	integer	R	For fault handling activities, this is the sequence number of the activity of the failed automatic activity for which this fault handling activity was invoked. For all other activity types, this is set to -1.
r_total_witem	integer	R	Records the total number of generated work items.
r_trigger_revert	integer	R	Records the number of revert ports triggered (either 0 or 1).
r_trigger_input	integer	R	Records the number of input ports triggered.
r_trigger_thresh	integer	R	Indicates the triggering threshold.
supervisor_name	string(32)	S	Indicates the workflow supervisor. The default is the creator.

Work Item

Purpose

Stores information about a task for a human or automatic performer.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmi_workitem
Object type tag: 4a

Work items are generated by Content Server from an activity object. Users cannot create work items. Users can modify only the following properties: a_held_by, a_wq_doc_profile, a_wq_flag, a_wq_name, and a_wq_policy_id.

Properties

Table 175, page 456, lists the properties for the work item type.

Table 175. Properties defined for the work item type

Property	Datatype	Single or repeating	Description
a_held_by	string(32)	S	Name of the user who has acquired the task. This is only set if the workflow tasks are managed using the work queue feature available through Process Builder.
a_wq_doc_profile	string(64)	S	Name of the work queue doc profile associated with this task. **Note:** This is only set if the workflow tasks are managed using the work queue feature available through Process Builder.

Property	Datatype	Single or repeating	Description
a_wq_flag	integer	S	Indicates whether the task is pushed to a queue member or pulled by a queue member. Valid values are: 0, meaning the task is pulled by a queue member 1, meaning the task is pushed to a queue member The default is 1. **Note:** This is only set if the workflow tasks are managed using the work queue feature available through Process Builder.
a_wq_name	string(32)	S	Name of the work queue to which this work item is assigned if the task is assigned to a work queue. For tasks generated by automatic activities, this may be set to an error message if Content Server cannot process the task. For details, refer to *Content Server Fundamentals*.
a_wq_policy_id	ID	S	Object ID of the work queue policy that controls how the work item is handled on the queue. **Note:** This is only set if the workflow tasks are managed using the work queue feature available through Process Builder.
r_act_def_id	ID	S	Refers to the activity definition object.

Property	Datatype	Single or repeating	Description
r_act_seqno	integer	S	Records the sequence number of the activity in which the package is being handled.
r_auto_method_id	ID	S	Contains the object ID of an application to be invoked (a dm_method instance).
r_creation_date	datetime	S	Records date and time of when a work item is generated.
r_due_date	datetime	S	Indicates the date and time when a work item is expected to complete.
r_exec_launch	Boolean	S	Indicates whether the work item is currently executing. TRUE means the work item is executing; FALSE means it is not executing.
r_exec_os_error	string	S	Contains the operating system error string, if any.
r_exec_result_id	ID	S	Contains the document ID of the saved results of the application execution.
r_exec_retried_count	integer	S	For automatic tasks, the number of times execution has been attempted
r_exec_timed_out	Boolean	S	Indicates if an execution times out.
r_ext_performer	string(32)	R	Lists new performers to repeat the same activity. If a group name, only one work item is generated and any group member can acquire the work item.

Property	Datatype	Single or repeating	Description
r_handling_instruction	integer	S	Used internally by the workflow agent to handle automatic tasks. Valid values are: 0, meaning handle in the regular manner 1, meaning TBD
r_launch_timeout	Date	S	Records the date and time at which at work item execution times out. This is set when the work item begins execution if the work item is generated by an automatic activity. Otherwise, its value is NULLDATE.
r_next_retry_date	Date	S	Time at which a failed task will be retried
r_output_port	string(16)	R	Allows a performer to specify a set of output ports by their unique names.
r_performer_name	string(32)	S	Contains the name of an activity performer, or the owner name of the method if this is an automatic work item.
r_priority	integer	S	Represents the priority assigned by a performer or an application. The default value comes from the activity_priority property of dm_process of the activity by which the work item is generated.
r_queue_item_id	ID	S	Refers to a peer dmi_queue_item object.

Property	Datatype	Single or repeating	Description
r_runtime_state	integer	S	Contains the current state of the work item. Valid value are: 0, meaning dormant 1, meaning acquired 2, meaning finished 3, meaning paused 4, meaning Dpaused (a work item in the dormant state is paused) 5, meaning Apaused (a work item in the acquired state is paused) 6, meaning Ppaused (a work item in the paused state is paused) 7, meaning Faulted Values 4-6 occur when a workflow containing the work item is halted.
r_target_task_id	ID	S	For fault-handling tasks, this is the object ID of the automatic activity for which this fault-handling task was created.
r_workflow_id	ID	S	Contains the object ID of the workflow that generated this item.
return_value	integer	S	Contains the returned value set by a performer or an application.
user_cost	double	S	Actual user cost spent to complete the work item. The default value is 0.
user_time	integer	S	Actual amount of time spent by the user to complete the work item. The default is 0.

Work Queue

Purpose

Represents a work queue for work items generated from a workflow.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_workqueue
Object type tag: 00

A work queue object represents a work queue for workflow tasks. The workqueue object type is installed by a script when Content Server is installed. However, work queues (instances of the type) are created and managed using Webtop.

Properties

Table 176, page 461, lists the properties defined for the type.

Table 176. Properties defined for the work queue type

Property	Datatype	Single or repeating	Description
skill_info_ids	ID	R	Object IDs of the wq skill info objects that identify the skills required to complete the tasks in this work queue
wq_category_id	ID	S	Object ID of the workqueue category object representing the category to which this work queue belongs
wq_name	string(32)	S	Name of the work queue
wq_policy_id	ID	S	Object ID of the workqueue policy used to handle work items in this queue

Work Queue Category

Purpose

Defines a category for a work queue.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_workqueue_category
Object type tag: 0b

A work queue category object represents a category of work queues.

The object type is installed by a script when Content Server is installed. However, instances of the type are created and managed through Webtop when work queues are set up and managed.

Properties

The object type has no properties defined for it. It inherits all its properties from its supertype, dm_folder. It uses only the object_name, acl_name, and acl_domain properties. The object_name property stores the work queue category name. The acl_name and acl_domain properties are set to specify the ACL named “Work Queue User Default ACL” for all work queue category objects.

Work Queue Doc Profile

Purpose

Stores information about a particular kind of document.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_workqueue_doc_profile
Object type tag: 00

A work queue doc profile contains information that describes a particular kind of document. The objects are used to manage work items placed on work queues. Work queues, and consequently, work queue doc profiles, are created using Webtop. The object type is installed using a script at the time Content Server is installed.

Properties

Table 177, page 463, lists the properties defined for the type.

Table 177. Properties defined for the work queue doc profile type

Property	Datatype	Single or repeating	Description
doc_profile_name	string(64)	S	Name of the doc profile.
owner_name	string(32)	S	Name of the user who owns the work queue doc profile object.
wq_name	string(32)	R	Names of the work queues to which documents that use this profile may be assigned.
wq_policy_id	ID	R	Object IDs of work queue policy objects. The work queue policy object identified at a particular index position is associated with the work queue specified at the corresponding index position in wq_name.

Work Queue Policy

Purpose

Defines configuration information for a work queue.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_workqueue_policy
Object type tag: 00

A work queue policy object defines configuration parameters for handling a task in a workqueue. The parameters control how the items are handled. Each work queue has one associated work queue policy. If a document associated with a task has a defined work queue policy, that policy overrides the work queue's policy.

The work queue policy object type is installed by a script when Content Server is installed. You must have installed Documentum Process Builder to use work queues and their associated work queue policies.

Properties

Table 178, page 464, lists the properties defined for the type.

Table 178. Properties defined for the work queue policy type

Property	Datatype	Single or repeating	Description
calendar_id	ID	S	reserved for future use
increment_priority	integer	S	Value by which the task priority is incremented
increment_priority_method	string(256)	S	reserved for future use
increment_priority_mode	integer	S	reserved for future use
initial_priority	integer	S	Initial priority value of tasks in the work queue

Property	Datatype	Single or repeating	Description
max_priority	integer	S	Maximum allowed priority value for an unfinished task on the queue
max_threshold	ID	S	Maximum number of unfinished tasks allowed in the queue
owner_name	string(32)	S	Name of the user who owns this work queue policy object
percent_quality_check	integer	S	Value, interpreted as percentage, that determines whether the quality assurance is performed on the task when it is completed
policy_name	string(255)	S	Name of the work queue policy
policy_type	integer	S	Identifies whether the policy defines the policy of a work queue or a document. Valid values are: 0, meaning it is a work queue policy 1, meaning it is a document policy

Work Queue User Profile

Purpose

Records information about a work queue user.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_workqueue_user_profile
Object type tag: 00

A work queue user profile describes a user who performs tasks taken from a work queue. The information includes the user's skill level. The user profiles are created when the work queue is created.

The object type is installed by a script when Content Server is installed. You must have installed Documentum Process Builder to use work queues and the associated user profiles.

Properties

Table 179, page 466, lists the properties defined for the type.

Table 179. Properties defined for the work queue user profile type

Property	Datatype	Single or repeating	Description
doc_profile_name	string(64)	R	Names of doc profile objects
owner_name	string(32)	S	Name of the owner of this workqueue user profile object
skill_level	integer	R	Defines skill levels for the user. The level identified at a particular index position applies to the workqueue and doc profile identified in the corresponding index positions in workqueue_name and doc_profile_name.
user_name	string(32)	S	Name of the user
workqueue_name	string(32)	R	Names of the work queues

Wq Skill Info

Purpose

Describes a skill required for the performance of a task on a workqueue.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dmc_wq_skill_info
Object Type Tag: 00

An object of type wq skill info records information about a skill required to complete a workflow task. These skill descriptions are used in conjunction with user skill descriptions to determine whether a user can perform a task on from a workqueue.

Properties

Table 180, page 467, lists the properties defined for wq skill info.

Table 180. Properties defined for the wq skill info object type

Property	Datatype	Single or repeating	Description
display_text	string(255)	R	Text displayed for the skill value at the corresponding index position in one of the valid_values_*datatype* properties.
multiple_select	integer	S	Indicates whether a user can have multiple values for this skill. Valid values are: 0, the user may not have multiple values for this skill (default) 1, the user may have multiple values for this skill
skill_description	string(2000)	S	Description of the skill
skill_name	string(128)	S	Name of the skill

Property	Datatype	Single or repeating	Description
skill_operator	integer	S	Operator to use when checking a user's actual skill level meets the required skill for a workflow task. Valid values depend on the datatype of the skill. For skills of type integer, valid values are: 0, meaing != 1, meaning = 2, meaning > 3, meaning >= 4, meaning < 5, meaning <= 6, meaning "bit and" 7, meaning "bit or" **Note:** The value 6 (bit and) is not supported on DB2. For skills of type double, valid values are: 0, meaing != 1, meaning = 2, meaning > 3, meaning >= 4, meaning < 5, meaning <= For skills of type string, valid values are: 0, meaning not equals 1, meaning equals
skill_type	integer	S	Datatype of the skill. Valid values are: 0, integer (default) 1, double 2, string
valid_values_double	double	R	Valid values of the skill. This property is used only when the skill_type is 1.

Property	Datatype	Single or repeating	Description
valid_values_int	integer	R	Valid values of the skill. This property is used only when the skill_type is 0.
valid_values_string	string(255)	R	Valid values of the skill. This property is used only when the skill_type is 2.

Wq Task Skill

Purpose

Associates a required skill with a workflow task.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dmc_wq_task_skill
Object Type Tag: 00

A wq task skill object associates a skill with a workflow task on a workqueue.

Properties

Table 181, page 470, lists the properties defined for the type.

Table 181. Properties defined for the wq task skill object type

Property	Datatype	Single or repeating	Description
skill_info_id	ID	S	Object ID of the dmc_wq_skill_info object.
skill_values_double	double	R	The required skill values for the task. This is set only when the skill is a double datatype skill.
skill_values_int	integer	R	The required skill values for the task. This is set only when the skill is a integer datatype skill.
skill_values_string	string(255)	R	The required skill values for the task. This is set only when the skill is a string datatype skill.
workitem_id	ID	S	Object ID of the workitem

Wq User Skill

Purpose

Records the skill values for a user for a specific skill.

Description

Supertype: Persistent Object
Subtypes: None
Internal Name: dmc_wq_user_skill
Object Type Tag: 00

A wq user skill object records the skill values for a particular user for a specific skill.

Properties

Table 182, page 471, lists the properties defined for the type.

Table 182. Properties defined for the wq user skill object type

Property	Datatype	Single or repeating	Description
skill_info_id	ID	S	Object ID of the dmc_wq_skill_info object
skill_values_double	double	R	The required skill values for the task. This is set only when the skill is a double datatype skill.
skill_values_int	integer	R	The required skill values for the task. This is set only when the skill is a integer datatype skill.
skill_values_string	string(255)	R	The required skill values for the task. This is set only when the skill is a string datatype skill.
user_name	string(32)	S	Name of the user

XML Application

Purpose

Stores the information that defines a particular kind of XML document.

Description

Supertype: Folder
Subtypes: None
Internal name: dm_xml_application
Object type tag: 0b

The information in an xml application object includes the document type definition (DTD) used by the document, the name space identified in the document's header, and the document's root element.

Properties

Table 183, page 472, lists the properties defined for the type .

Table 183. Properties defined for the xml application type

Property	Datatype	Single or repeating	Description
dtd_public_id	string(255)	S	The public ID of the document type definition identified in dtd_system_id.
dtd_system_id	string(255)	S	The system ID of a document type definition.
namespace	string(80)	S	The name space identified in the XML document's header.

Property	Datatype	Single or repeating	Description
root_elements	string(32)	R	Identifies the type of XML document.
root_object _types	string(32)	R	Names of the object types used by the XML application. These are the formal names of the object types; for example, dm_document. Custom object types may be included. EMC Documentum Desktop uses this property to determine which object types to display in user interface fields for operations on objects handled by an XML application.

XML Config

Purpose

Stores a document type definition file (DTD) as content.

Description

Supertype: Document
Subtypes: None
Internal name: dm_xml_config
Object type tag: 09

An xml config object stores, as content, a document type definition file (DTD), the XML file that describes the rules for chunking an XML document in an XML application.

Properties

Table 184, page 474, lists the property defined for the XML config object type.

Table 184. Properties defined for the xml config type

Property	Datatype	Single or repeating	Description
config_locator	ID	S	Used internally to manage XML documents.

XML Custom Code

Purpose

Stores the Java code that implements a custom Java interface as content.

Description

Supertype: Document
Subtypes: None
Internal name: dm_xml_custom_code
Object type tag: 09

An xml custom code object stores, as content, the Java code that implements a custom Java interface.

Properties

Table 185, page 475, lists the properties defined for the type.

Table 185. Properties defined for the xml custom code type

Property	Datatype	Single or repeating	Description
class_path	string(255)	S	Java class path of the code.
code_type	integer	S	Identifies the Java interface implemented by the code. Valid values are: 1, meaning code is a DfDTDHandler subclass 2, meaning code is a DfDTDDocument Handler subclass 3, meaning Code implements IDfLinkDetector
com_class_id	string(80)	S	Currently unused

XML Style Sheet

Purpose

Stores an XSL file as content.

Description

Supertype: Document
Subtypes: None
Internal name: dm_xml_style_sheet
Object type tag: 09

An xml style sheet object stores an XSL file as content. An XSL file is a style sheet that defines how to format an XML document on output.

Properties

Table 186, page 476, lists the properties defined for the type.

Table 186. Properties defined for the xml style sheet type

Property	Datatype	Single or repeating	Description
transformed _format	string(80)	S	Defines the output file format of the XML document.

XML Zone

Purpose

Contains the information used to populate the Zone tab on a Find dialog.

Description

Supertype: Document
Subtypes: None
Internal name: dm_xml_zone
Object type tag: 09

An xml zone object contains the information used to populate the Zone tab on a Find dialog.

Properties

Table 187, page 477, lists the properties defined for the type.

Table 187. Properties defined for the xml zone type

Property	Datatype	Single or repeating	Description
description	string(32)	S	User defined.
zone_name	string(32)	R	Identifies one or more zones that can be searched in a fulltext index search. A zone is the name of an XML tag.

Chapter 3

Lightweight Object Type and Properties for Shareable Object Types and the Property Bag

This chapter contains reference information for lightweight objects. It also contains the reference information for the properties that are only added to an object type that is declared shareable or that has non-qualifiable properties.

When first introduced, the lightweight types and the properties supporting shareable types were only used by EMC Documentum client products. Users could not create lightweight or shareable object types. From version 6.5 forward, users can take advantage of lightweight and shareable types. There are additional properties defined for dm_type and dmi_type_info that also support these object types. Refer to the description of dm_type and dmi_type_info object types to learn about those properties.

The creation and use of non-qualifiable properties is supported for external use, since the version 6.0 release of lightweight objects. When you create an object type, you may define one or more of the type's properties as non-qualifiable. *Content Server Fundamentals* describes non-qualifiable properties and the supporting property bag property. The *Documentum DQL Reference Manual* describes how to create a type with a non-qualifiable property.

Lightweight

Purpose

Description

Supertype: dm_sysobject
Subtypes: User-defined
Internal name: N/A
Object type tag: inherited from supertype (or, for EMC Documentum client types released in 6.0, 6E)

For EMC Documentum client types this object type serves as the supertype of any lightweight object type. Like the persistent object type, it serves only to pass on its properties to all of its subtypes. For version 6.5, the data model was changed so that lightweight types are subtypes of SysObjects.

Properties

Table 188, page 480, lists the properties defined for this object type in addition to the user-defined properties.

Table 188. Properties defined for the lightweight object type

Property	Datatype	Single or repeating	Description
i_vstamp	integer	S	Version stamp of the object
i_sharing_parent	ID	S	Object ID of the parent object with which the lightweight object is sharing If the object is materialized, this value is the object ID of the lightweight object.
object_name	string(255)	S	Name of the object
r_object_id	ID	S	Object ID of the object
r_page_cnt	integer	S	Number of primary content files associated with the object

Properties supporting lightweight and shareable object types and non-qualifiable properties

Purpose

Provide support for lightweight subtypes and shareable types and non-qualifiable properties.

Description

The properties listed here are added to an object type only under specific conditions. The description of each property explains the property's use and the conditions under which it appears in a object type definition.

Table 189, page 481, lists the properties that are added to an object type only in specific circumstances.

Table 189. Properties added to a type in specific circumstances

Property	Datatype	Single or repeating	Description
i_shared_status	integer	S	The property indicates whether an instance of the object type is shared by any lightweight object. Valid values are: 0, the object is not shared 1, the object is shared This property is automatically added to the definition of an object type if that type is created as a shareable type. Only dm_sysobject or its subtypes can be shareable types.

Property	Datatype	Single or repeating	Description
i_sharing_type	ID	S	This property is added to any object type defined as a shareable object type. The property records the object ID of the dm_type object representing the top-most lightweight object type in the type hierarchy of the lightweight object instance that shares the shareable instance. The property is NULL if no lightweight objects share the shareable instance.
i_orig_parent	ID	S	This property is added to any object type defined as a shareable object type. The property records the object ID of the shareable instance of the sharing, parent object. Consequently, for a materialized lightweight object, this value records the original sharing instance with which the lightweight object shared values.
allow_propagating_changes	Boolean	S	Indicates whether any changes applied to the sharing instance are propagated to the lightweight objects that share the instance. The default is F.

Property	Datatype	Single or repeating	Description
i_property_bag	string(2000)	S	The property stores the names and values of non-qualifiable properties. It is also used to store the names and values of aspect properties if the properties are added to the aspect with the OPTIMIZEFETCH option. This property is automatically added to the definition of a lightweight object type if the type definition contains a non-qualifiable property. The property can also be explicitly added by altering the object type.
r_property_bag	string(2000)	R	This property stores overflow from the i_property_bag property. This property is added to an object type definition if the definition includes the i_property_bag property.

Chapter 4

Non-persistent Object Types

This chapter contains the reference information for the non-persistent object types. These object types represent objects that are created as needed at runtime and are destroyed when the session is terminated.

Client Config

Purpose

An client config object describes the configuration parameters for sessions.

Description

The client config object type is a non-persistent type.

A client config object is created when DFC is initialized. Its properties are the keys of the dfc.properties file. If a key is set in the file, its value is reflected in the client config object. If a key is not specifically set, the client config object reflects the key's default value. The values are applied to all sessions opened by that DFC. This object is intended for use by system administrators.

Properties

To access the object and retrieve the property values, use the IDfSession.getClientConfig method.

Connection Config

Purpose

Describes a session's connection to a single repository.

Description

The connection config object type is a non-persistent type.

A connection config object describes a session's connection to a single repository. This object is intended for use by system administrators.

Properties

Table 190, page 487, lists the properties defined for the type.

Table 190. Properties defined for the connection config type

Property	Datatype	Single or repeating	Description
client_cache_size	integer	S	Defines the size of the client cache size for the subconnection. The value is taken from the session config's client_cache_size property setting
client_cache_write_interval	integer	S	Controls how often periodic refreshes of the client persistent cache occurs. The default value is 60 minutes.
connection_id	ID	S	Object ID of the session object for the connection.
connection_name	string(5)	S	Connection identifier for the connection
force_coherency_checks	Boolean	S	T disables the use of consistency check rules (for client persistent caches) defined in queries or a cache config object. The default is F (FALSE).

Property	Datatype	Single or repeating	Description
network_requests	integer	S	Number of RPC calls sent by the client session to the server. This updates throughout the session, whenever an RPC call occurs.
r_date_format	string(40)	S	The date format that will be used to return dates to the user.
r_docbase_id	ID	S	repository ID of the repository to which the subconnection is connected.
r_docbase_name	string(120)	S	Name of the repository to which the subconnection is connected.
r_events_location	string(32)	S	Name of the location object in the repository that points to the repository's events directory.
r_mac_protocol	string(32)	S	Identifies the Macintosh file-sharing protocol used by the repository.
r_persistent_caching	Boolean	S	Whether persistent client caching is enabled for the session. T (TRUE) indicates that caching is enabled. F (FALSE) indicates that it is disabled.
r_security_mode	string(32)	S	Identifies the security mode under which the repository is running.
r_user_name	string(32)	S	Name of the current user. This value is always the same as that found in the session config object.
secure_channel	Boolean	S	Whether the session is using a secure connection for its repository connection. T (TRUE) means the connection is secure. F (FALSE) means it is not.

Docbase Locator

Purpose

Contains information about the repositories known to a connection broker.

Description

The docbase locator type is a non-persistent type.

A docbase locator object is an object that is constructed and returned by a connection broker in response to a Getdocbasemap method call. The information for a single repository appears at corresponding index levels in the repeating properties. For example, the name of the repository whose ID appears in r_docbase_id[0] is found in r_docbase_name[0], and its description is found in r_docbase_description[0].

Properties

Table 191, page 489, lists the properties defined for the type.

Table 191. Properties defined for the docbase locator type

Property	Datatype	Single or repeating	Description
auth_protocol	string(32)	S	On Windows platforms, if set to domain required, indicates that the repository is running in domain-required mode. If the repository is not using domain-required mode, this property is blank. On UNIX platforms, if you are authenticating users against a Windows domain, this is set to unix_domain_used Otherwise, this property is blank for

Property	Datatype	Single or repeating	Description
			repositories running on a UNIX platform.
i_docbroker _version	string(32)	S	Version number of the responding connection broker.
i_host_addr	string(32)	S	The IP address of the host on which the responding connection broker resides.
i_host_name	string(128)	S	Name of the host machine on which the responding connection broker resides.
i_port_number	integer	S	Port number of the responding connection broker.
r_docbase _description	string(128)	R	Verbose name or description of the repository.
r_docbase_id	ID	R	The internal ID of the repository.
r_docbase _name	string(32)	R	Name of the repository.
r_federation _name	string(32)	R	Name of the federation to which this docbase belongs, if any.
r_govern _docbase	string(32)	R	For each repository participating in a federation, the name of the governing repository in the federation.
r_object_id	ID	S	Object ID of the docbase locator object.
r_server_version	string(32)	R	Version numbers of the servers.

Docbroker Locator

Purpose

Contains information about each connection broker that the client DMCL can access.

Description

The docbroker locator type is a non-persistent type.

A docbroker locator object contains information about each connection broker that the client DMCL can access. The object is constructed and returned by the client library in response to a Getdocbrokermap method call. The information for a single connection broker appears at corresponding index levels in the properties. For example, the protocol for the connection broker whose host_name appears in host_name[0] is found in network_protocol[0] and its port number is found in port_number[0].

Properties

Table 192, page 491, lists the properties defined for the type.

Table 192. Properties defined for the docbroker locator type

Property	Datatype	Single or repeating	Description
host_name	stringstring((128)	R	Name of the host machine on which the connection broker resides
network_protocol	string(12)	R	Protocol for a given connection broker
port_number	integer	R	Number of the port on the host machine that the connection broker uses for communication
time_out	integer	R	Time, in seconds, that a server waits for a response from the connection broker before forwarding the request to a backup connection broker.

Server Locator

Purpose

Contains information about the servers known to a connection broker.

Description

The server locator type is a non-persistent type

A server locator object contains information about the servers known to a connection broker. A server locator object is constructed and returned by the connection broker in response to a Getservermap method call.

Properties

Table 193, page 492, lists the properties defined for the type.

Table 193. Properties defined for the server locator type

Property	Datatype	Single or repeating	Description
a_silent_login	Boolean	S	Used internally by the Documentum clients.
i_connection _protocol	string(12)	R	Used internally by the dmcl to establish a network connection.
i_docbase_id	string	S	Decimal value of the DocBase identifier assigned during installation.
i_docbroker _version	string(32)	S	Version number of the responding connection broker.
i_host_addr	string(32)	S	IP address of the host machine on which the responding connection broker resides.
i_host_name	string(128)	S	Name of the host machine on which the responding connection broker resides.
i_port_number	integer	S	Port number of the port on the host machine that the connection broker is using for communications.

Property	Datatype	Single or repeating	Description
i_server _connection _address	string(40)	R	Used internally by the dmcl to establish a network connection.
i_logon_support	string(64)	S	Type of logon (Will be NT_Unified_logon for servers running under Windows)
r_client_proximity	integer	R	Indicates how far the server is from the client. The value rises directly in relation to the distance.
r_host_name	string(128)	R	Name of the host machine on which the server resides.
r_keep_entry _interval	integer	R	How long the connection broker will keep the server's entry in the absence of a checkpoint message from the server.
r_last_checkpoint	TIME	R	The time that the server last reported to the connection broker.
r_last_status	string(24)	R	Status of the server process. The value is one of: starting, open, stopped, or presumed down.
r_next_checkpoint	TIME	R	The time that the server is next expected to report to the connection broker.
r_object_id	string	S	Object ID of the server locator object.
r_process_id	integer	R	Process ID of the server.
r_server_name	string(128)	R	Name of the server.
r_server_version	string(32)	R	Version number of the server.

Session Config

Purpose

Contains information about an open repository session.

Description

The session config object type is a non-persistent type.

A session config object contains information about an open repository session. Each repository session has one associated session config object. Access to this object is through the IDfSessionManager interface.

Properties

Table 194, page 494, lists the properties defined for the type.

Table 194. Properties defined for the session config type

Property	Datatype	Single or repeating	Description
alias_set	string(32)	S	The session-level default alias set.
api_exec_count	integer	S	Number of dmAPIExec function calls issued during the session.
api_get_count	integer	S	Number of dmAPIGet function calls issued during the session.
api_set_count	integer	S	Number of dmAPISet function calls issued during the session.
application_code	string(32)	R	Identifies, by application, the application-controlled objects that this session can modify. If this is NULL, the session cannot modify any application-controlled objects. Application codes can contain only alphanumeric characters and the underspace character. They cannot contain spaces nor can they start with the characters

Property	Datatype	Single or repeating	Description
			dm_. Codes beginning with dm_ are reserved for use by Documentum.
batch_hint_size	integer	S	Defines the number of rows returned to Content Server by the RDBMS in a single call to the RDBMS. The default is 20.
client_cache_size	integer	S	obsolete
docbase_scope	string(120)	S	obsolete
dynamic_groups	string(32)	R	List of the dynamic groups to which the session belongs.
ignore_pre _processing	Boolean	S	Used for customized workflows. It directs the server not to execute external applications defined as pre-processing. The default is FALSE.
ignore_post _processing	Boolean	S	Used for customized workflows. It directs the server not to execute external applications defined as post-processing. The default is FALSE.
local_clean_on _init	Boolean	S	Indicates whether the server should automatically purge the client local area whenever the client is started. The default is T.
local_diskfull _check	integer	S	Defines how often the server checks the free space on the disk containing the client's local area. The default is 1, meaning that the space is checked with each getFile. Setting this to a number greater than 1, n, means that the space is checked every n getFiles. Setting it to 0 means the space is never checked.

Property	Datatype	Single or repeating	Description
local_diskfull _limit	integer	S	Specifies the maximum amount of space that the client local area may use. The value is interpreted in megabytes. A value of 0 means no limit.
local_purge_on _diskfull	Boolean	S	Indicates whether the server should automatically purge the client local area when the diskfull limit is reached. The default value is T.
local_path	string(255)	S	Specifies the location of the local client area. Use a full path specification.
network_requests	integer	S	Number of RPC calls made during the session. This is a cumulative number, incrementing by one each time an RPC call is made in the session.
r_cache_query	Boolean	S	Indicates whether the session is enabled for query caching. The default value is FALSE.
r_date_format	string(40)	S	The date format the server will use to return dates to the client. The value is derived from the setting in the connectionconfig object for the first repository connection of the session.
r_events_location	string(32)	S	Identifies the location object in the repository representing the events directory. The value is derived from the setting in the connection config object for the first repository connection of the session.
r_host_name	string(128)	S	Contains the host name of the machine on which the current session was initiated.

Property	Datatype	Single or repeating	Description
r_mac_protocol	string(32)	S	Specifies the file-sharing protocol if this is a Macintosh client session. The value is derived from the setting in the connection config object for the first repository connection of the session.
r_security_mode	string(32)	S	Contains the security level enforced for the repository you are using. The value is derived from the setting in the connection config object for the first repository connection of the session.
r_user_name	string(32)	S	Contains the user name of the currently logged-in user.
r_working _directory	string(255)	S	Contains the path of the user's working directory.
ref_binding_label	string(32)	S	Defines which document version to fetch when accessing a remote document. If this property is defined, its value overrides the binding specified in the dm_reference object that points to the remote object.
session_alias	string(12)	S	Contains the session identifier, as a string literal.
session_codepage	string(64)	S	The code page in use for the repository session. By default, the value is obtained from the client_codepage property of the api config object. If client_codepage is unset, the value is determined programmatically based on the value in session_locale.

Property	Datatype	Single or repeating	Description
			For all session locales except ja (Japanese) and ko (Korean), the default setting is ISO-8859-1. For Japanese, the default setting is Shift_JIS. For Korean, the default setting is EUC-KR.
session_id	ID	S	Contains the ID of the session.
session_locale	string(32)	S	Locale of the repository session. By default, the value is obtained from the client_locale property of the api config object. If client_locale is unset, the setting is determined programmatically based on the locale of the client's host machine. Valid values are: en, for English de, for German fr, for French ja, for Japanese ko, for Korean

Chapter 5

Documentum Process Builder Object Types

This chapter contains the reference information for the object types installed with the Documentum Process Builder product.



Caution: Instances of these types are created and managed internally. Do not modify or delete the instances. Additionally, do not modify, delete, or subtype these object types.

In addition to the object types described in this chapter, there are several object types supporting workflows that are installed with Content Server. Those object types, described in Chapter 2, Core Persistent Object Types, are:

- Activity
- Completed Workflow
- Completed Work Item
- Package
- Routecase Condition
- Process
- WF Attachment
- WF Package Schema
- WF Package Skill
- WF Timer
- Workflow
- Work Item
- Work Queue
- Work Queue Category
- Work Queue Doc Profile
- Work Queue Policy
- Work Queue User Profile

- Wq Skill Info
- Wq Task Skill
- Wq User Skill

Wfsd Element

Purpose

Description

Supertype: dm_lightweight
Shares: dmc_wfsd_element_parent
Subtypes: wfsd element boolean, wfsd element date, wfsd element double, wfsd element integer, wfsd element string
Internal Name: dmc_wfsd_element
Object Type Tag: 00

The wfsd element object type serves as the supertype for all the object types used to model structured data elements for the business process data model functionality. The wfsd element object type is a lightweight object type. Instances of wfsd element share property values with an instance of dmc_wfsd_element_parent.

Properties

Table 195, page 501, lists the properties defined for the type.

Table 195. Properties defined for the wfsd element object type

Property	Datatype	Single or repeating	Description
object_name	string(255)	S	Name of the structured data element. This is from its shared wfsd element parent instance.
process_id	ID	S	Object ID of the workflow template
sd_type_info_id	ID	S	Object ID of the wfsd type info object that identifies a structured data type.
workflow_id	ID	S	Object ID of the workflow to which the structured data element belongs

Wfsd Element Boolean

Purpose

Represents structured data elements of the Boolean datatype.

Description

Supertype: Wfsd Element
Subtypes: None
Internal Name: dmc_wfsd_element_boolean
Object Type Tag: 00

An object of type wfsd element boolean represents a structured data element whose datatype is Boolean.

Properties

Table 196, page 502, lists the property defined for the type.

Table 196. Properties defined for the wfsd element boolean object type

Property	Datatype	Single or repeating	Description
boolean_value	Boolean	S	Value of the structured data instance

Wfsd Element Date

Purpose

Represents structured data elements of the date datatype.

Description

Supertype: Wfsd Element
Subtypes: None
Internal Name: dmc_wfsd_element_date
Object Type Tag: 00

An object of type wfsd element date represents a structured data element whose datatype is date.

Properties

Table 197, page 503, lists the property defined for the type.

Table 197. Properties defined for the wfsd element date object type

Property	Datatype	Single or repeating	Description
date_value	Date	S	Value of the structured data instance

Wfsd Element Double

Purpose

Represents structured data elements of the double datatype.

Description

Supertype: Wfsd Element
Subtypes: None
Internal Name: dmc_wfsd_element_double
Object Type Tag: 00

An object of type wfsd element double represents a structured data element whose datatype is double.

Properties

Table 198, page 504, lists the property defined for the type.

Table 198. Properties defined for the wfsd element double object type

Property	Datatype	Single or repeating	Description
double_value	Double	S	Value of the structured data instance

Wfsd Element Integer

Purpose

Represents structured data elements of the integer datatype.

Description

Supertype: Wfsd Element
Subtypes: None
Internal Name: dmc_wfsd_element_integer
Object Type Tag: 00

An object of type wfsd element integer represents a structured data element whose data type is integer.

Properties

Table 199, page 505, lists the property defined for the type.

Table 199. Properties defined for the wfsd element integer object type

Property	Datatype	Single or repeating	Description
int_value	integer	S	Value of the structured data instance

Wfsd Element Parent

Purpose

Serves as the supertype for the wfsd element object type.

Description

Supertype: SysObject
Subtypes: Wfsd Element
Internal Name: dmc_wfsd_element_parent
Object Type Tag: 08

The wfsd element parent object type is defined as shareable object type. If a workflow template has structured data elements in the workflow, there is one wfsd element object that is the parent (shared instance) for all the workflow's structured data element instances

Properties

This object type has no properties defined for it. It inherits all of its properties from its supertype.

Wfsd Element String

Purpose

Represents structured data elements of the string datatype.

Description

Supertype: Wfsd Element
Subtypes: None
Internal Name: dmc_wfsd_element_string
Object Type Tag: 00

An object of type wfsd element string represents a structured data element whose data type is string.

Properties

Table 200, page 507, lists the property defined for the type.

Table 200. Properties defined for the wfsd element string object type

Property	Datatype	Single or repeating	Description
string_value	string(2000)	S	Value of the structured data instance

Wfsd Type Info

Purpose

Records the type definition of structured data types

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_wfsd_type_info
Object type tag: 08

A wfsd type info object records the type definition of a structured data element object type.

Properties

Table 201, page 508, lists the properties defined for the type.

Table 201. Properties defined for the wfsd type info object type

Property	Datatype	Single or repeating	Description
attr_default_value	string(255)	R	Default value of the property identified in the corresponding index position in attr_name. **Note:** When accessing this property for properties of integer, double, and date datatypes, cast the value to the appropriate type.
attr_description	string(255)	R	Description of the property. The value at a particular index position is associated with the property identified at the corresponding index position in attr_name.

Property	Datatype	Single or repeating	Description
attr_display_name	string(255)	R	Display name of the property. The name at a particular index position is associated with the property identified at the corresponding index position in attr_name.
attr_element_type_name	string(40)	R	Name of the property in the wfsd element subtype that models the structured data element. The name at a particular index position is associated with the property identified at the corresponding index position in attr_name.
attr_is_filterable	integer	R	It indicates whether the attribute defined at the corresponding index of attr_display_name can be filtered or not
attr_is_repeating	integer	R	Indicates whether the property is repeatable. Valid values are: 0, the property is not repeatable (default) 1, the property is repeatable The value at a particular index position is associated with the property identified at the corresponding index position in attr_name.

Property	Datatype	Single or repeating	Description
attr_is_reportable	integer	R	Indicates whether the property is reportable. The workflow engine may report the values of reportable properties when an activity is completed. Valid values are: 0, the property is not reportable (default) 1, the property is reportable The value at a particular index position is associated with the property identified at the corresponding index position in attr_name.
attr_is_searchable	integer	R	Indicates whether the property may be used in search screens for searching active workflows. Valid values are: 0, the property is not searchable (default) 1, the property is searchable The value at a particular index position is associated with the property identified at the corresponding index position in attr_name.
attr_sdt_name	string(2000)	R	Name of the property in the structured data element object type. The name at a particular index position is associated with the property identified at the corresponding index position in attr_name.

Property	Datatype	Single or repeating	Description
attr_type	integer	R	Identifies the datatype of the property. Valid values are: 0, Boolean 1, Integer 2, String 3, ID 4, Time and Date 5, Double The value at a particular index position is associated with the property identified at the corresponding index position in attr_name.
element_type_name	string(27)	S	Name of the wfsd element subtype that represents the structured data element. This name must be unique within the repository.
object_name	string(255)	S	Display name of the structured data element object type. This is the display name used by Process Builder and Webtop. The property is inherited from dm_sysobject.
report_type_name	string(27)	S	Name of the object type that models the report table for the structured data element This property is empty if none of the attributes of the structured data table are reportable.

Property	Datatype	Single or repeating	Description
sdt_name	string(255)	S	Name of the structured data element type. This value is used to exchange data between EMC Documentum processes and other systems. If the structured data type is created by importing a structured data type schema, the name is the name of the complexType or elementDeclaration of the schema element.
target_namespace_uri	string(2000)	S	Target namespace URI of the schema associated with the structured data object type.
type_description	string(1024)	R	Description of the property.
xml_config_id	ID	S	Object ID of the dm_xml_config object that contains the storage mapping information for this structured data object type.
xml_schema_id	ID	S	Object ID of the XML schema object that contains the XML schema information for this structured data object type.

Wfsdrp Boolean

Purpose

Represents the reporting table for structured data elements of type Boolean.

Description

Supertype: Wfsdrp Parent
Subtypes: None
Internal Name: dmc_wfsdrp_boolean
Object Type Tag: 00

An object of type wfsdrp boolean records the values in the reporting table for one structured data element of type Boolean.

Properties

Table 202, page 513, lists the properties defined for the type.

Table 202. Properties defined for the wfsdrp boolean object type

Property	Datatype	Single or repeating	Description
boolean_value	Boolean	S	Value of the structured data element

Wfsdrp Date

Purpose

Represents the reporting table for structured data elements of type Date.

Description

Supertype: Wfsdrp Parent
Subtypes: None
Internal Name: dmc_wfsdrp_date
Object Type Tag: 00

An object of type wfsdrp date records the values in the reporting table for one structured data element of type date.

Properties

Table 203, page 514, lists the properties defined for the type.

Table 203. Properties defined for the wfsdrp date object type

Property	Datatype	Single or repeating	Description
date_value	Date	S	Value of the structured data element

Wfdsrp Double

Purpose

Represents the reporting table for structured data elements of type double.

Description

Supertype: Wfsdrp Parent
Subtypes: None
Internal Name: dmc_wfsdrp_double
Object Type Tag: 00

An object of type wfsdrp double records the values in the reporting table for one structured data element of type double.

Properties

Table 204, page 515, lists the properties defined for the type.

Table 204. Properties defined for the wfsdrp double object type

Property	Datatype	Single or repeating	Description
double_value	double	S	Value of the structured data element

Wfsdrp Integer

Purpose

Represents the reporting table for structured data elements of type integer.

Description

Supertype: Wfsdrp Parent
Subtypes: None
Internal Name: dmc_wfsdrp_integer
Object Type Tag: 00

An object of type wfsdrp integer records the values in the reporting table for one structured data element of type integer.

Properties

Table 205, page 516, lists the properties defined for the type.

Table 205. Properties defined for the wfsdrp integer object type

Property	Datatype	Single or repeating	Description
int_value	integer	S	Value of the structured data element

Wfsdrp Parent

Purpose

Serves as the supertype for all reporting subtypes for reportable attributes.

Description

Supertype: None
Subtypes: Wfsdrp Boolean, Wfsdrp Date, Wfsdrp Double, Wfsdrp Integer, Wfsdrp String,
Internal Name: dmc_wfsdrp_parent
Object Type Tag: 00

This object type is used as the supertype for all subtypes created at runtime for structured data types that have reportable attributes. These subtypes are used for reporting. The subtype inherits the properties of this type.

Properties

Table 206, page 517, lists the properties defined for the type.

Table 206. Properties defined for the wfsdrp parent object type

Property	Datatype	Single or repeating	Description
act_name	string(128)	S	Name of the activity associated with this element
act_seqno	integer	S	Sequence number of the activity associated with this element
performer_name	string(32)	S	Name of the work item's performer
process_id	ID	S	Object ID of the dm_process object of the workflow's template.
process_name	string(255)	S	Name of the workflow template
sd_element_name	string(255)	S	Name of the Boolean structured data element
time_stamp_utc	Date	S	UTC time of the reporting

Property	Datatype	Single or repeating	Description
workflow_id	ID	S	Object ID of the workflow instance
workitem_id	ID	S	Object ID of the work item.

Wfsdrp String

Purpose

Represents the reporting table for structured data elements of type string.

Description

Supertype: Wfsdrp Parent
Subtypes: None
Internal Name: dmc_wfsdrp_string
Object Type Tag: 00

An object of type wfsdrp string records the values in the reporting table for one structured data element of type string.

Properties

Table 207, page 519, lists the properties defined for the type.

Table 207. Properties defined for the wfsdrp string object type

Property	Datatype	Single or repeating	Description
string_value	string(2000)	S	Value of the structured data element

Chapter 6

Collaboration Object Types

This chapter contains the reference information for object types supporting Documentum Collaborative Services. The object types are packaged in the Documentum Collaborative Services DAR file. Instances of the types are created and managed internally when you use features of Documentum Collaborative Services. You must have installed Content Server with a Documentum Collaborative Services license to use the features these types support.

Calendar

Purpose

Serves as the base type for calendars.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_calendar
Object type tag: 0b

A calendar object contains the calendar event objects that comprise the calendar. Any objects in the calendar folder that are not calendar events are considered attachments to the calendar. These objects are created and managed internally when users create calendars.

Properties

The calendar object type has no properties defined for itself. All its properties are inherited from its supertype.

Calendar Event

Purpose

Records an individual event on a calendar.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_calendar_event
Object type tag: 08

A calendar event object stores the information about an event on a calendar. These objects are created and managed internally when users create and modify calendars. Calendar event objects are stored in a calendar folder representing the calendar on which the event appears.

Properties

Table 208, page 523, lists the properties defined for the type.

Table 208. Properties defined for the calendar event type

Property	Datatype	Single or repeating	Description
change_number	integer	S	Value that is incremented whenever a change is made that invalidates attendee responses
end_date	date	S	Event's end date
event_attendees	ID	R	Object ID of the dm_user objects of the event participants who have user objects in the repository
event_description	string This is Non-qualifiable property, so it has no defined length.	S	Rich text description of the event.
event_location	string(128)	S	Location of the event

Property	Datatype	Single or repeating	Description
event_organizer	ID	S	Object ID of the repository user who organized the event
event_priority	integer	S	Represents the priority of the vent. Valid values are from 1 (highest priority) through 9 (lowest priority)
event_uid	string(255)	S	Globally unique identifier for the event
exception_dates	date	R	For recurring events, the date of an exception to the recurrence rule. These are the dates on which the event did not occur as scheduled.
has_date_only	Boolean	S	T means that the start and end dates of the event specify only the date, with no time component. F means that the start and end dates include a time component.
is_transparent	Boolean	S	Whether the event appears to consume time on the calendar. T means that event does appear to consume time. F means that the event does not appear to consume time on the calendar.
other_attendees	string(255)	R	List of email addresses of those participants who are not users in the repository
recurrence_dates	date	R	Dates of recurrence instances
recurrence_end	date	S	The upper boundary of the recurrence pattern
recurrence_id	date	S	Date of recurrence which this event overrides

Property	Datatype	Single or repeating	Description
recurrence_rule	string	S	Recurrence rule
	This is Non-qualifiable property, so it has no defined length.		The recurrence rule is specified in the iCal RRULE format.
start_date	date	S	The starting date of the event

Comment

Purpose

Records a comment in a discussion

Description

Supertype: Richtext
Subtypes: None
Internal name: dmc_comment
Object type tag: 08

A comment object represents a single comment in discussion. Comments are created and managed through Webtop.

Properties

Table 209, page 526, lists the properties defined for the type.

Table 209. Properties defined for the comment type

Property	Datatype	Single or repeating	Description
comment_creation_date	date	S	Original date and time at which a comment was created. This is only set if the comment object represents a comment copied from the original comment.
comment_creator	string (128)	S	Name of the user who originally made the comment. This is only set if the comment object represents a comment copied from the original comment.
comment_id	integer	S	Identifying value for the comment within the topic.

Property	Datatype	Single or repeating	Description
comment_modtag	integer	S	Value of the topic's last_update_ modtag property after the most recent modification of this comment
comment_parentid	integer	S	The identifying value of the comment to which this comment is a reply or response. If the comment is not a reply to another comment, this is 0.

Datatable

Purpose

Contains the objects that comprise a data table.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_datatable
Object type tag: 0b

A datatable object contains all the row objects, the datatable schema object, and the folders for row attachments that together define a data table. The properties of the datatable record the basic metadata about the datatable, such as its name and owner. These objects are created internally when a user creates a database in Collaboration Services.

Properties

Table 210, page 528, lists the properties defined for the type.

Table 210. Properties defined for the datatable type

Property	Datatype	Single or repeating	Description
schema_id	ID	S	Object ID of the dmc_datatable_ schema object associated with this datatable

Datatable Row

Purpose

Serves as the supertype for the custom types created to record the definition of a row in a datatable.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_datatable_row
Object type tag: 08

The datatable row object type is the supertype of all custom object types created to represent rows in a datatable. These custom types are created when a user creates a datatable. Each row has a custom object type that records the row's characteristics.

Properties

There are no properties defined for this object type. It inherits all properties from its supertype.

Datatable Schema

Purpose

Records metadata about a datatable.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_datatable_schema
Object type tag: 08

A datatable schema object records metadata about a datatable in its defined properties and stores the XML schema that contains information about the table fields as content. These objects are created and managed internally, as users create, modify, and delete datatables.

Properties

Table 211, page 530, lists the properties defined for the type.

Table 211. Properties defined for the datatable schema type

Property	Datatype	Single or repeating	Description
column_counter	integer	S	Counter that provides a unique suffix to the row type attribute name, in the format: column_*x*, where x is an integer.
is_enterprise	Boolean	S	Indicates whether a schema can be shared among multiple instances of a data table.
row_type	string(32)	S	Name of the Content Server type used for the row
title_column	string(32)	S	Identifier of the column which represents the name of the row.

Datatable Schema Ex

Purpose

Records information required to manage datatable schemas.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_datatable_schema_ex
Object type tag: 00

A datatable schema ex object is used internally to manage datatables. Do not modify, delete, or change this type or one of these objects directly.

Properties

Table 212, page 531, lists the properties defined for the type.

Table 212. Properties defined for the datatable schema ex type

Property	Datatype	Single or repeating	Description
a_column_id	string(32)	S	Identifies the column to which the data belongs. The value is the column_id.
a_next_autonumber	integer	S	Next automatically generated number to use for an auto-number field.
a_schema_id	ID	S	Object ID of the schema

Datatable Settings

Purpose

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_datatable_settings
Object type tag: 37

A datatable settings object links a datatable and its datatable schema object or a datatable and a user object. The properties of the datatable settings object store basic information about the datatable's settings.

Properties

Table 213, page 532, lists the properties defined for the type.

Table 213. Properties defined for the datatable settings type

Property	Datatype	Single or repeating	Description
fulltext_search_data	string(255)	S	Full-text search string data value
grouping_column	string(32)	S	Id of the group column
grouping_columns	string(32)	R	Ids of the columns that can be grouped
search_column	string(32)	R	List of the columns that are searchable
search_column_data	string(255)	R	Search parameters or data used to build a search query
search_column_data2	string(255)	R	Additional search parameters or data used to build a search query
search_column_ opcode	integer	R	Specifies the operation code used to build the query
sort_column	string(32)	S	Search column identifier
summary_column	string(32)	R	List of summary columns

Discussion

Purpose

Contains one or more topics or topic subtypes.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_discussion
Object type tag: 0b

A discussion object serves as the container for topics relating to a particular discussion. Only objects of type dm_topic or its subtypes may be linked to a discussion.

Properties

The discussion object type has no properties defined for the type. It inherits all its properties from its supertype.

Notepage

Purpose

Represents richtext content that is authored and read online.

Description

Supertype: Document
Subtypes: None
Internal name: dmc_notepage
Object type tag: 09

A notepage object stores a content file in rich text format. You cannot create renditions of notepage objects. The content represented by notepage objects is created and used in the context of collaboration in Webtop.

Properties

Table 214, page 534, lists the property defined for the type.

Table 214. Properties defined for the notepage type

Property	Datatype	Single or repeating	Description
anchor_id	ID	R	Object IDs of any repository objects within the content file associated with the object.

Readcomment

Purpose

Used to manage unread comments in a discussion.

Description

Supertype: Persistent Object
Subtypes: None
Internal name: dmc_readcomment
Object type tag: 00

A readcomment object contains information that allows the system to determine which comments in a particular discussion are unread by a particular viewer.

Properties

Table 215, page 535, lists the properties defined for this type.

Table 215. Properties defined for the readcomment type

Property	Datatype	Single or repeating	Description
topic_id	ID	S	Object ID of topic object.
topic_viewed_modtag	integer	S	Date and time at which the user in user_name last viewed the discussion represented by the topic. The value is copied from the topic object's last_update_modtag property.
user_name	string(32)	S	Name of user

Richtext

Purpose

Represents rich text content associated with a SysObject

Description

Supertype: SysObject
Subtypes: dmc_comment
Internal name: dmc_richtext
Object type tag: 08

A richtext object represents richtext content associated with a SysObject either directly, through a relationship, or indirectly as a comment in a topic thread. The actual content is stored in either the content_value property or as an actual content file associated with the richtext object.

Properties

Table 216, page 536, lists the properties defined for the type.

Table 216. Properties defined for the richtext type

Property	Datatype	Single or repeating	Description
anchor_id	ID	R	Object IDs of repository objects found in the content
content_value	string(2000)	S	Stores the first 2000 bytes of the content
has_content	Boolean	S	Whether the content is fully contained in the content_value property or whether there is also a content file attached to the object. T means that there is a content file. F means that the content is fully contained in content_value.

Room

Purpose

Provides an additional access management layer for SysObjects.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_room
Object type tag: 0b

A room is a special folder that provides additional, optional functionality to control access to the objects in the folder when the objects are accessed through the room.

Properties

Table 217, page 537, lists the properties defined for the type.

Table 217. Properties defined for the room type

Property	Datatype	Single or repeating	Description
builtin_groups	string(32)	R	Names of private groups in the room. The first four index positions are reserved for the system-defined groups in room: [0]=Members [1]=Owners [2]=Contributors [3]=Visitors
default_contrib_permit	integer	S	Default base object-level permission for the contributor's group applied to objects governed by this room
default_contrib_ xpermit	string(32)	S	Default extended permission for the contributor's group applied to objects governed by this room

Property	Datatype	Single or repeating	Description
default_owner_permit	integer	S	Default base object-level permission for the owner's group applied to objects governed by this room
default_owner_ xpermit	string(32)	S	Default extended permission for the owner's group applied to objects governed by this room
is_public	Boolean	S	Whether the room is a public or private room. T means the room is a public room. F means the room is a private room and only members of the Members group (builtin_groups[0]) can access objects in the room. The only valid value is currently F.
only_owners_ ungovern	Boolean	S	Whether membership in the Owners groups is an additional requirement for users trying to remove an object from the room. T means that a user must be a member of the Owners group in addition to having Write and Change_permit permissions on an object to remove that object from the room. F means that any user in the room with Write and Change_permit permissions on the object can remove it from the room.

Property	Datatype	Single or repeating	Description
status_text	string (*maximum*)	S	Text description of the room's status. The length of this property is the maximum allowed by the underlying database.
status_value	integer	S	Application-defined status value of the room.

Topic

Purpose

Used to manage a single discussion.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_topic
Object type tag: 0b

A topic object is used to manage a discussion thread—the set of comments about a single topic. Discussions are supported in Webtop if you have installed Content Server with a Documentum Collaborative Services license.

Properties

Table 218, page 540, lists the properties defined for the type.

Table 218. Properties defined for the topic type

Property	Datatype	Single or repeating	Description
is_disabled	Boolean	S	Whether the topic is disabled. T means the topic is disabled. F means the topic is not disabled. The default is F. **Note:** Users cannot create, view or edit comments in disabled topics.

Property	Datatype	Single or repeating	Description
next_comment_id	integer	S	Value to be used as the object ID of the next new comment.
last_update_modtag	integer	S	Value used internally to track each time a topic is modified

Chapter 7

Document Submission Manager Object Types

This chapter contains the reference information for the object types that support Document Submissions Manager. The object types are installed when you install the DSM DAR file in a repository.



Caution: These object types are used by EMC Documentum Submissions Manager. You must not modify the object types, nor can you create instances of the types directly.

DSM Application

Purpose

Contains a eCTD application.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_dsm_application
Object type tag: 0b

A dsm application object contains all the objects that comprise an eCTD application submitted and managed by EMC Documentum Submissions Manager.

Properties

Table 219, page 544, lists the properties defined for the type.

Table 219. Properties defined for the dmc dsm application type

Property	Datatype	Single or repeating	Description
last_published_name	string(255)	S	Name of the last published submission

DSM Backbone

Purpose

Represents the root document of the virtual document that represents a submission.

Description

Supertype: dsm section
Subtypes: DSM Stf Backbone, DSM M1 Backbone
Internal name: dmc_dsm_backbone
Object type tag: 09

A dsm backbone object stores the index.xml file for a submission. This file is the root document of the virtual document that represents the submission in the repository.

Properties

Table 220, page 545, lists the properties defined for the type.

Table 220. Properties defined for the dmc dsm backbone type

Property	Datatype	Single or repeating	Description
document_id	ID	R	Used internally
dsm_data_model_version	string(32)	S	Used internally
is_import_post_processed	Boolean	S	Used internally
is_published	Boolean	S	Used internally
last_used_leaf_id	string(128)	S	Used internally
leaf_id	string(128)	R	Used internally

DSM Doc Properties

Purpose

Stores the values of the eCTD properties for an eCTD content file.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_dsm_doc_properties
Object type tag: 37

A dsm doc properties object records the eCTD properties of an eCTD content file. The values are version-specific, so there is one dsm doc properties object for each version of the document that contains the file.

Properties

Table 221, page 546, lists the properties defined for the type.

Table 221. Properties defined for the dmc dsm doc properties type

Property	Datatype	Single or repeating	Description
application_version	string(128)	S	Version of the software application used to create the file
checksum_type	string(128)	S	Algorithm used to generate the checksum
checksum_value	string(128)	S	Checksum value for the file
document_keywords	string(4000)	S	Keywords specified for the file
document_title	string(128)	S	Title of the document
font_library	string(128)	S	Commercial name of the font or font set used to create the document

Property	Datatype	Single or repeating	Description
last_checksum_time	Date	S	Timestamp of the last checksum for this document
version_identifier	string(128)	S	Submitter's internal version number or version identification for the report

DSM Drug Product

Purpose

Records information about a drug product specified in an eCTD section element.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_drug_product
Object type tag: 09

A dsm drug product object records the property values specified in a <m2-3-drug-product> or <m3-2-p-drug-product> section element.

Properties

Table 222, page 548, lists the properties defined for the type.

Table 222. Properties defined for the dmc dsm drug product type

Property	Datatype	Single or repeating	Description
dosage_form	string(128)	S	The form in which the drug was given to study subjects.
manufacturer_ name	string(128)	S	Name of the drug's manufacturer
product_name	string(128)	S	Name of the product

DSM Drug Substance

Purpose

Records information about a drug substance specified in an eCTD section element.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_drug_substance
Object type tag: 09

A dsm drug substance object records the property values specified in a <m2–3–s-drug-substance> or <m3–2–s-drug-substance> section element.

Properties

Table 223, page 549, lists the properties for the type.

Table 223. Properties defined for the dmc dsm drug substance type

Property	Datatype	Single or repeating	Description
manufacturer_ name	string(128)	S	Name of the drug's manufacturer
substance_name	string(128)	S	Name of the substance

DSM Excipient

Purpose

Records information about an excipieint substance specified in an eCTD section element.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_excipient
Object type tag: 09

A dsm excipient object records the property values specified in a <m3-2-a-1-control-of-excipients> section element.

Properties

Table 224, page 550, lists the properties defined for the type.

Table 224. Properties defined for the dmc dsm excipient type

Property	Datatype	Single or repeating	Description
excipient_substance	string(128)	S	Name of the excipient substance

DSM Facilities Equip

Purpose

Records the property values specified in an eCTD section element.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_facilities_equip
Object type tag: 09

A dsm facilities equip object records the property values in a <m3-2-a-1-facilities-and-equipment> eCTD section element.

Properties

Table 225, page 551, lists the properties defined for the type.

Table 225. Properties defined for the dmc dsm facilities equip type

Property	Datatype	Single or repeating	Description
dosage_form	string(128)	S	The form in which the drug was given to study subjects.
manufacturer_ name	string(128)	S	Name of the drug's manufacturer
product_name	string(128)	S	Name of the product
substance_name	string(128)	S	Name of the substance

DSM Indication

Purpose

Records the property values specified in an eCTD section element.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_indication
Object type tag: 09

A dsm indication object records the property values specified in a <m5-3-5-reports-of-efficacy-and-safety-studies> or <m2-7-3-summary-of-clinical-efficacy> eCTD section element.

Properties

Table 226, page 552, lists the properties defined for the type.

Table 226. Properties defined for the dmc dsm indication type

Property	Datatype	Single or repeating	Description
indication_name	string(128)	S	Name of the indication

DSM M1 Backbone

Purpose

Represents as regional backbone.

Description

Supertype: DSM Backbone
Subtypes: None
Internal name: dmc_dsm_m1_backbone
Object type tag: 09

A dsm m1 backbone object represents a regional backbone file.

Properties

This type has no defined properties. It inherits all its properties from its parent type.

DSM Safety Eval

Purpose

Records the property values specified in an eCTD section element.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_safety_eval
Object type tag: 09

A dsm safety eval object records the property values in a <m3-2-a-2-adventitious-agents-safety-evaluation> eCTD section element.

Properties

Table 227, page 554, lists the properties defined for the type.

Table 227. Properties defined for the dmc dsm safety eval type

Property	Datatype	Single or repeating	Description
dosage_form	string(128)	S	The form in which the drug was given to study subjects.
manufacturer_ name	string(128)	S	Name of the drug's manufacturer
product_name	string(128)	S	Name of the product
substance_name	string(128)	S	Name of the substance

DSM Sect Doc Attributes

Purpose

Stores the values of attributes specific to a section-document relationship.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_dsm_sect_doc_attributes
Object type tag: 37

A dsm sect doc attributes object stores the values of properties specific to a section-document relationship. The relation_name for objects of this type is always dmc_dsm_section_to_doc.

Properties

Table 228, page 555, lists the properties defined for the type.

Table 228. Properties defined for the dmc dsm sect doc attributes type

Property	Datatype	Single or repeating	Description
folder_path	string(1024)	S	Repository location of the document
leaf_id	string(128)	S	Unique identifier for this file in the XML instance.
modified_file	string(128)	S	Identifies the location of the document being modified. The value in this property points the the index.xml file and the leaf ID of the file being altered.
modified_file_id	string(32)	S	Object ID of the document specified in modified_file

Property	Datatype	Single or repeating	Description
operation_name	string(32)	S	Identifies the operation to be performed on the file to be modified. Valid values are: new append replace delete
submission_id	string(32)	S	Object ID of the submission folder to which the document belongs.

DSM Section

Purpose

Records information used to display a submission in Virtual Document Manager.

Description

Supertype: Document
Subtypes: DSM Backbone, DSM Drug Product, DSM Drug Substance, DSM Excipient, DSM Facilities Equip, DSM Safety Eval, DSM Indication, DSM Section Extension, DSM Stf Section
Internal name: dmc_dsm_section
Object type tag: 09

A dsm section object contains information used by EMC Documentum Submissions Manager to manage the display of the submission in the Virtual Document Manager.

Properties

Table 229, page 557, lists the properties defined for the type.

Table 229. Properties defined for the dmc dsm section type

Property	Datatype	Single or repeating	Description
ectd_profile	string(32)	S	Identifies the associated DSM configuration profile governing DSM processing This property may be empty. If so, the value is derived from the first parent in the virtual document that contains a value for this property.

Property	Datatype	Single or repeating	Description
section_id	string(128)	S	Name of the section. If the section has properties, the property values are concatenated with the section name.
section_name	string(128)	S	The eCTD section name. The exact value depends on whether the section has properties.

DSM Section Extension

Purpose

Records eCTD element extensions.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_section_extension
Object type tag: 08

A dsm section extension object records eCTD element extensions. Each pair at the same index position in the properties represents a nested level of <node-extension>.

Properties

Table 230, page 559, lists the properties defined for the type.

Table 230. Properties defined for the dmc dsm section extension type

Property	Datatype	Single or repeating	Description
extension_id	string(128)	R	List of extension identifiers.
extension_title	string(128)	R	Titles of the extensions. The title at any particular index position is associated with the identifier in the corresponding position in extension_id.

DSM Stf Backbone

Purpose

Represents the backbone of a study tagging (stf.xml) file.

Description

Supertype: DSM Backbone
Subtypes: None
Internal name: dmc_dsm_stf_backbone
Object type tag: 09

A dsm stf backbone object represents the root document of the virtual document that represents an stf.xml file in the repository. That virtual document is a component of the virtual document whose root is represented by a dsm backbone object.

Properties

Table 231, page 560, lists the properties defined for the type. For the repeating properties, the values recorded at a particular index position represent one category name attribute element.

Table 231. Properties defined for the dmc dsm stf backbone type

Property	Datatype	Single or repeating	Description
category_name	string(128)	R	Category names identified in the <category name> properties in a <study-identifier> element.
category_info_type	string(32)	R	Information type identified in <category name> properties in a <study-identifier> element.
category_value	string(128)	R	The value specified in <category name> properties in a <study-identifier> element.

Property	Datatype	Single or repeating	Description
study_id	string(128)	S	Value of the <study-id> attribute in a <study-identifier> element.
study_title	string(128)	S	Value of the <title> attribute in a <study-identifier> element.

DSM Stf Section

Purpose

Records the attributes of an eCTD content block within an STF document.

Description

Supertype: DSM Section
Subtypes: None
Internal name: dmc_dsm_stf_section
Object type tag: 09

A dsm sft section object stores the values of attributes of an eCTD content block from an sft.xml document.

Properties

Table 232, page 562, lists the properties defined for the type.

Table 232. Properties defined for the dmc dsm stf section type

Property	Datatype	Single or repeating	Description
block_title	string(128)	S	The title value of the content block.
property_info_type	string(32)	R	The information types specified in property attributes.
property_name	string(128)	R	The names specified in property attributes.
property_value	string(128)	R	The values specified in property attributes.
subject_name	string(128)	R	Name of the subject
subject_info_type	string(32)	R	The information type for the subject.

DSM Study Attributes

Purpose

Records the study-related attributes for an STF study report.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_dsm_study_attributes
Object type tag: 08

A dsm study attributes object records the vaues of the study attribute for an STF study report. The objects are related to the document that stores the report. The relation name is always dmc_dsm_doc_to_study_attri.

Properties

Table 233, page 563, lists the properties defined for the type.

Table 233. Properties defined for the dmc dsm study attributes type

Property	Datatype	Single or repeating	Description
property_info_type	string(32)	R	The information types specified in property attributes
property_name	string(128)	R	The names specified in n property attributes
property_value	string(128)	R	The values specified in property attributes
study_title	string(128)	S	The title of the study
subject_name	string(128)	R	Names of the subjects
subject_info_type	string(32)	R	The information types for the subjects.

DSM Study Report

Purpose

Relates a study report within an STF to a logical content file in the main backbone.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_dsm_study_report
Object type tag: 37b

A dsm study report object establishes a relationship between a study report within an STF and a logical content fiel in the main backbone. The objects are related to the document that stores the report. The relation name is always dmc_dsm_doc_to_stf.

Properties

Table 234, page 564, lists the properties defined for the type.

Table 234. Properties defined for the dmc dsm study report type

Property	Datatype	Single or repeating	Description
study_report_status	string(32)	S	Only valid value is Delete, to indicate that the logical content has been deleted from the STF.

DSM Submission

Purpose

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_dsm_submission
Object type tag: 0b

A dsm submission object contains all the documents related to a particular submission.

Properties

Table 235, page 565, lists the property defined for the type.

Table 235. Properties defined for the dmc dsm submission type

Property	Datatype	Single or repeating	Description
previous_name	string(32)	S	Previous submission name before “lock”

Chapter 8

Retention Policy Services Object Types

This chapter contains the reference information for the object types used by Retention Policy Services.



Caution: These types are installed with the Retention Policy Services product. Instances of these types are created and maintained through the Retention Policy Services product. Neither the types nor instances of the types can be changed or modified by users or client applications other than RPS.

RPS Action

Purpose

Identifies the Java class used to execute an action.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_action
Object type tag: 08

An rps action object identifies the class and, optionally the interface within the class, that is executed to perform an action.

Properties

Table 236, page 568, lists the properties defined for the type and those inherited properties that are used for a specific purpose by the type.

Table 236. Properties defined for the rps action type

Property	Datatype	Single or repeating	Description
action_type	integer	S	Identifies what kind of action is represented by the rps action object. The only valid value is 0, meaning a notification.
class	string(128)	S	Path to the Java class that executes the action
interface	string(128)	S	Path to the Java class interface that executes the action
object_name	string(255)	S	Name of the action
subject	string(192)	S	Description of the action

RPS Action Rel

Purpose

Relates an rps phase rel object to an rps action object.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_rps_action_rel
Object type tag: 37

An rps action rel object associates a particular rps phase with an action and identifies the execution rule for the action.

Properties

Table 237, page 569, lists the properties defined for the type and those inherited properties that are used for a specific purpose by the type.

Table 237. Properties defined for the rps action rel type

Property	Datatype	Single or repeating	Description
arg_object_id	ID	R	Used internally. Value dependent on the type of action identified in the associated rps action object.
child_id	ID	S	Object ID of the rps action object
execution_rule_id	ID	S	Object ID of the execution rule that is the criterion for action
object_name	string(255)	S	Name of the object
parent_id	ID	S	Object ID of the rps phase rel object representing the phase associated with the action

RPS Authority

Purpose

Identifies the persons or agencies that are authorized to determine whether an object has fulfilled its phase conditions.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_authority
Object type tag: 08

An rps authority object records information about the authorities authorized to determine whether an object has fulfilled its phase conditions. The rps authority objects are related to objects of type dmc_rps_phase_rel objects (representing phases in a retention policy lifecycle). The relationship is named dmc_rps_phase_authority_rel_type.

Properties

Table 238, page 570, lists the properties defined for the type and those inherited properties that are used for a specific purpose by the type.

Table 238. Properties defined for the rps authority type

Property	Datatype	Single or repeating	Description
authorizers_id	ID	R	Object IDs of the rps contact objects representing users or agencies.
is_authority_valid	Boolean	S	Whether this authority is valid. T means that this authority can determine phase fulfillment; F means that it cannot be used to determine phase fulfillment.
object_name	string(255)	S	Name of the authorizing body or person. This property is required.
title	string(400)	S	User-defined description of the authority.

RPS Base Date

Purpose

Records which property in an object type is used to populate the retention_base_date property in the dmc_rps_retainer objects for instances of the type.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_base_date
Object type tag: 08

An rps base date object records the property in an object type whose value is used to populate the retention_base_date property in the dmc_rps_retainer objects with which instances of object type are associated.

Properties

Table 239, page 571, lists the properties defined for the type.

Table 239. Properties defined for the rps base date type

Property	Datatype	Single or repeating	Description
object_type	string(32)	S	Name of the object type. The object type must be dm_sysobject or one of its subtypes. Use the type's internal name. This is a required property.
base_date_name	string(32)	S	Name of an property defined for or inherited by the object type identified in object_type. The property must be of date/time datatype. This is a required property.
title	string(32)	S	User-defined description of the property mapping.

RPS Child Strategy

Purpose

Records the name of the class called to apply an rps retainer object to the objects associated with a dm_retainer object.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_child_strategy
Object type tag: 08

An rps child strategy object defines the Java class called to attach an rps retainer to objects that are stored in a container that is associated with an rps retainer object.

Properties

Table 240, page 572, lists the properties defined for the type and those inherited properties that have a specific use in the type.

Table 240. Properties defined for the rps child strategy type

Property	Datatype	Single or repeating	Description
class	string(128)	S	Path to the Java class that executes based on the selected child strategy. This is a required property.
interface	string(128)	S	Path the interface for the class specified in class.
is_container_aging	Boolean	S	Whether the rps retainer ages. T means the container ages; F means it does not.

Property	Datatype	Single or repeating	Description
object_name	string(255)	S	Name of the child strategy. Valid values are: ONE_FOR_ALL ONE_FOR_EACH RECORDS_STRATEGY EMAIL_STRATEGY This inherited property is required.
retainer_strategy	integer	S	Inherited from rps retainer, this indicates whether all objects are to use the same retainer or each object must have its own retainer. Valid values are: 0, meaning that each object must have its own retainer 1, meaning that all objects are to use the same retainer

RPS Condition

Purpose

Represents a template from which rps event objects are generated.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_condition
Object type tag: 08

An rps condition object defines a condition associated with a particular phase of retention policy lifecycle. RPS condition objects are related to phases (dmc_rps_phase_rel objects) through a relationship named dmc_rps_phase_condition_rel_type.

Properties

Table 241, page 574, lists the properties defined for the type and the inherited properties used for a specific purpose by the type.

Table 241. Properties defined for the rps condition type

Property	Datatype	Single or repeating	Description
category	string(32)	S	Identifies the category to which the condition is assigned. This is a required property.
object_name	string(255)	S	The name of the condition. For example: Project Closed or Employee Retired This inherited property is a required property.
title	string(400)	S	User-defined description of the condition.

RPS Contact

Purpose

Records information about a person defined in Retention Policy Manager.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_contact
Object type tag: 08

RPS contact objects contain information about persons defined in Retention Policy Manager. The objects are referenced by rps hold objects, rps authority objects, and rps event objects.

Note: The persons defined by rps contact objects are not necessarily repository users.

Properties

Table 242, page 575, lists the properties defined for the type and those inherited properties used for a specific purpose by the type.

Table 242. Properties defined for the rps contact type

Property	Datatype	Single or repeating	Description
email	string(32)	R	Email addresses for the contact
phone	string(64)	R	Telephone numbers for the contact
object_name	string(255)	S	Name of the contact
title	string(400)	S	User-defined description of the contact
user_id	ID	S	Object ID of the contact's user object in the repository if the contact is a repository user.

RPS Disposition Method

Purpose

Records the name of the Java class called to dispose of a retained object.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_disposition_method
Object type tag: 08

An rps disposition method object identifies the Java class called to dispose of an object at the end of retention. Which class is called is dependent on the selected disposition option.

Properties

Table 243, page 576, lists the properties defined for the type and the inherited property used for a specific purpose by the type.

Table 243. Properties defined for the rps disposition method type

Property	Datatype	Single or repeating	Description
class	string(128)	S	Path to the Java class invoked to dispose of an object.
is_container_destroyed	Boolean	S	Whether the container was destroyed on disposition. T means the container is destroyed; F means that it was not.
object_name	string(255)	S	Name of the disposition method. Valid values are: UNKNOWN REVIEW TRANSFER/EXPORT DESTROY
priority_no	integer	S	Defines the action on a retained object when two of its retainers

Property	Datatype	Single or repeating	Description
			qualify for disposition during the same qualification period. Valid values are: 10, undefined action 30, review 50, transfer or export 70, destroy

RPS Event

Purpose

Represents one instance of a dmc_rps_condition object.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_event
Object type tag: 08

An rps event object is an instance of a condition defined in an rps condition object. It contains data specific to that instance of the condition. There is one rps event object generated for each rps condition on each rps phase rel object. RPS event objects are associated with the generating rps condition object through a relation object named dmc_rps_condition_event_rel_type.

Properties

Table 244, page 578, lists the properties defined for the type and those inherited properties used for a specific purpose by the type.

Table 244. Properties defined for the rps event type

Property	Datatype	Single or repeating	Description
a_last_review_date	Date	S	The most recent date and time at which the condition was fulfilled.
condition_id	ID	S	Object ID of the rps condition object from which this event was generated This is a required property.
fulfilled_by	ID	R	Object IDs of the rps contact objects representing contacts who have indicated that the event has been fulfilled.
fulfillment_message	string(255)	S	User-defined message to be displayed when this event is fulfilled.

Property	Datatype	Single or repeating	Description
object_name	string(255)	S	Name for the event. This property is required.
title	string(400)	S	User-defined description of the event.

RPS Execution Rule

Purpose

Defines a criterion for execution of an action.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_execution_rule
Object type tag: 08

An rps execution rule object defines when a particular action is to be executed against an obejct controlled by a retention policy. The definition is recorded in the object name. For example, if object_name is “Phase Entry”, the rule is executed only when the target object experiences phase entry.

The object type has no defined properties.

Properties

Table 245, page 580, lists the inherited properties that it uses for a specific purpose.

Table 245. Properties defined for the rps execution rule type

Property	Datatype	Single or repeating	Description
object_name	string(255)	S	Identifies the criterion for execution of the associated action
subject	string(192)	S	Description of the execution rule

RPS Hold

Purpose

Records information about a hold defined in Retention Policy Services.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_hold
Object type tag: 08

An rps hold object represents a hold defined in RPS. Holds prevent deletion of an object when the object is available for disposition at the end of its retention. A hold is associated with the object through a relationship named dmc_rps_hold_object_rel_type.

Properties

Table 246, page 581, lists the properties defined for the type and those inherited properties that are used for a specific purpose by the type.

Table 246. Properties defined for the rps hold type

Property	Datatype	Single or repeating	Description
a_last_review_date	Date	S	Date on which the status of the hold must be reviewed
a_retention_date	Date	S	Date when the hold was applied to the object
approvers_id	ID	R	Object IDs of the rps contact objects representing users who have approved the hold
object_name	string(255)	S	Name of the hold This is a required property.

Property	Datatype	Single or repeating	Description
requestors_id	ID	R	Object IDs of the rps contact objects representing the users who requested the hold This is a required property.
title	string(400)	S	User-defined description of the hold

RPS Notification

Purpose

Identifies the recipients of notifications

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_notification
Object type tag: 08

An rps notification object records the contacts to be sent notifications for a particular action and those contacts who acknowledge the notification.

Properties

Table 247, page 583, lists the properties defined for the type and those inherited properties that are used for a specific purpose by the type.

Table 247. Properties defined for the rps notification type

Property	Datatype	Single or repeating	Description
a_last_review_date	Date	S	Date and time at which the notification was sent
acknowledged_by_id	ID	R	Object IDs of the rps contact objects representing contacts who have acknowledged the notification
action_rel_id	ID	S	Object ID of the rps action rel object that identifies the action for a given phase
contact_id	ID	R	Objects IDs of the rps contact objects representing contacts to be notified
number_sent	integer	S	Number of times the notification has been sent
object_name	string(255)	S	Application-defined identifier for the notification
subject	string(192)	S	Description of the notification

RPS Phase Rel

Purpose

Represents one phase of an RPS retention policy.

Description

Supertype: State Extension
Subtypes: None
Internal name: dmc_rps_phase_rel
Object type tag: 37

An rps phase rel object represents one phase in an rps retention policy. The object associates a lifecycle definition (dm_policy object) with the dmc_rps_retention_policy. The parent_id property identifies the dm_policy object on which the retention policy is based and the child_id property identifies the rps retention policy. The properties defined for the rps phase rel object define the retention phase.

The relationship represented by the rps phase rel object is named dmc_rps_phase_rel_type.

Properties

Table 248, page 584, lists the properties defined for the type.

Table 248. Properties defined for the rps phase rel type

Property	Datatype	Single or repeating	Description
cutoff_day	integer	S	The day in a month up to which cutoff will be applied. The day is specified by its calendar number.
cutoff_month	integer	S	The month in the year in which cutoff will be applied. The month is specified by its numeric position in the year.

Property	Datatype	Single or repeating	Description
cutoff_period	string(32)	S	Defines an interval used to round-up cutoff values during processing. Values are: MONTHLY QUARTERLY SEMI_ANNUALLY ANNUALLY DISABLED
duration_days	integer	S	Number of days in the duration of this phase.
duration_months	integer	S	Number of months in the duration of this phase
duration_years	integer	S	Number of years in the duration of this phase
is_final_phase	Boolean	S	Whether this phase is the final phase in the retention lifecycle. T means that this is the final phase; F means that it is not.
phase_name	string(32)	S	Name of the phase. This is a derived value, not user-defined.
storage_id	ID	S	Object ID of the storage area to which to move objects when they complete this phase.

RPS Retainer

Purpose

Records information about the state of an object's retention.

Description

Supertype: Retainer
Subtypes: None
Internal name: dmc_rps_retainer
Object type tag: 08

An rps retainer object mirrors an object's association with a dm_retainer object and includes additional information about the object's status in the retention lifecycle.

Properties

Table 249, page 586, lists the properties defined for the type and those inherited properties used for a specific purpose by the type.

Table 249. Properties defined for the rps retainer Type

Property	Datatype	Single or repeating	Description
current_phase_id	ID	S	Object ID of the rps phase rel object representing the phase with which this rps retainer is associated
entry_date	Date	S	Date on which the current phase was entered
event_date	Date	S	The fulfillment date furthest in the future for the current phase's conditions.

Property	Datatype	Single or repeating	Description
immutability_rule	integer	S	Whether objects under control of this retention policy are immutable. Valid values are: 0, the object is changeable 1, the object is immutable This value is copied from the dm_retainer object.
object_name	string(255)	S	The object ID of the retainer. (An rps retainer's name is its object ID.)
parent_ancestor_id	ID	S	Reserved for future use
phase_name	string(32)	S	Name of the phase identified in current_phase_id
qualification_date	Date	S	Date on which objects in the current phase may be qualified for promotion
r_policy_id	ID	S	Object ID of the lifecycle applied to this rps retainer.
retainer_root_id	ID	S	Object ID of the object attached to this rps retainer object. This is blank if the child strategy for the retainer is ONE_FOR_ALL. This is an inherited property.
retention_base_date	Date	S	Date from which all qualifications, promotions, and dispositions are calculated This is as required property.
retention_date	Date	S	Earliest time at which the object attached to this retainer ceases to be retained. The date is calculated by adding sum of all phase durations to the value in retention_base_date.
retention_policy_id	ID	S	Object ID of the retainer object on which this rps retainer is based.

Property	Datatype	Single or repeating	Description
retention_rule_type	integer	S	Inherited from dm_retainer. This value must be 1, meaning retention is specified as a date.
ultimate_ancestor_id	ID	S	Object ID of the retainer from which all other rps retainer objects were generated. This value is applicable when the child strategy is ONE_FOR_EACH.
vdm_retention_rule	integer	S	Defines how virtual documents and their elements are retained. Valid values are: 0, meaning the policy may not be applied to virtual documents 1, meaning apply the policy only to the root document 2, meaning apply the policy to all virtual document components The value is copied from the retention policy when the retainer is created.

RPS Retainer Event Rel

Purpose

Associates rps event objects with rps retainer objects.

Description

Supertype: State Extension
Subtypes: None
Internal name: dmc_rps_retainer_event_rel
Object type tag: 37

An rps retainer event rel object associates rps event objects with a rps retainer object. The name of the relationship defined by rps retainer event rel objects is dmc_rps_retainer_event_rel_type.

Properties

Table 250, page 589, lists the properties defined for the type.

Table 250. Properties defined for the rps retainer event rel type

Property	Datatype	Single or repeating	Description
phase_id	ID	S	Object ID of the rps phase rel object to which this relationship applies
state_no	integer	S	Identifies the phase to which the event applies

RPS Retention Policy

Purpose

Defines a rule set from which rps retainer objects are generated.

Description

Supertype: SysObject
Subtypes: None
Internal name: dmc_rps_retention_policy
Object type tag: 08

An rps retention policy object defines the rules by which rps retainer objects are generated from the dm_retainer objects. The rules represent the influence of the retention lifecycle definition on the retention policy.

Properties

Table 251, page 590, lists the properties defined for the type.

Table 251. Properties defined for the rps retention policy type

Property	Datatype	Single or repeating	Description
aging_method	string(32)	S	Determines whether the promotion out of phases is based on conditions or dates. Valid values are: CONDITIONAL, meaning that fulfillment of conditions is used to advance through the phases CHRONOLOGICAL, meaning that dates are used to advance through the phases. This must be set to CONDITIONAL to define conditions for phases.

Property	Datatype	Single or repeating	Description
child_strategy_id	ID	S	Object ID of the rps child strategy object that defines the Java class invoked to apply retention.
child_strategy_class	string(128)	S	Java class that executes based on the selected child strategy. The value is a fully specified class name.
disposition_date	Date	S	Date on which objects associated with retention policy are eligible for disposition
disposition_method_class	string(128)	S	Java class that executes based on the selected disposition method. The value is a fully specified class name.
disposition_method_id	ID	S	Object ID of the rps disposition method object that defines how to dispose of objects controlled by this retention policy.
disposition_rule	integer	S	Defines how RPS disposes of an object. Valid values are: 0, meaning DESTROY CHILDREN 1, meaning DESTROY_VDM 2, meaning DESTROY_ROOT
immutability_rule	integer	S	Whether objects controlled by rps retainers based on this policy are set to immutable. Valid values are: 0, do not set the objects to immutable 1, set the objects immutable
is_enabled	Boolean	S	Whether the policy may be applied to objects.
object_name	string(255)	S	Name of the retention policy. This value is derived from the dm_policy on which the retention policy is based.

Property	Datatype	Single or repeating	Description
rendition_rule	integer	S	Whether the retention policy applies to all content (primary and renditions) of the object or only to primary content. Valid values are: 0, applies to all content 1, applies only to primary content
retainer_lifecycle_id	ID	S	Object ID of the lifecycle definition that is applied to all rps retainers generated from this retention policy object.
title	string(400)	S	User-defined description of the retention policy.
vdm_retention_rule	integer	S	Defines how virtual documents and their elements are retained. Valid values are: 0, meaning the retention policy may not be applied to a virtual document 1, meaning the policy is applied only to the root document 2, meaning the policy is applied to all the virtual document components

Chapter 9

Site Caching Services Object Types

The object types in this chapter support Documentum Site Caching Services (SCS). These types are installed when the SCS DocApp archive file is installed in a repository.

SCS Admin Config

Purpose

Records the common properties for all SCS configurations in the repository and the JDBC connection information for the source repository

Description

Supertype: SysObject
Subtypes: None
Internal name: scs_admin_config
Object type tag: 08

Properties

Table 252, page 594, lists the properties defined for the type.

Table 252. Properties defined for the scs admin config type

Property	Datatype	Single or repeating	Description
content_caching_ enabled	Boolean	S	Indicates whether source-side caching of content is enabled. The default is F, meaning that caching is not enabled.
database_connection	string(255)	S	The JDBC URL with which to connect to the source database.
extra_arguments	string(255)	R	Additional arguments to be applied to all SCS configurations. The arguments are specified in a name and value pair format. For example: `store_log TRUE`

Property	Datatype	Single or repeating	Description
object_name	string(255)	S	This property is inherited from dm_sysobject. However, because it is required for scs admin config objects, it is included in this table. For scs admin config objects, object_name is the name of the repository in which the object resides.
jdbc_driver	string(255)	S	The JDBC driver class name for the source database connection
method_trace_level	integer	S	Specifies the method trace level for all SCS configurations in the repository. Valid values range from 0 to 10.
product_verson	string(64)	S	Version level of the SCS product

Webc Config

Purpose

Describes the documents and properties that are to be exported to a Site Caching Services repository.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_webc_config
Object type tag: 08

A webc config object describes the documents and properties that are to be exported to a Site Caching Services repository.

Properties

Table 253, page 596, lists the properties defined for the type.

Table 253. Properties defined for the webc config type

Property	Datatype	Single or repeating	Description
a_event_number	integer	S	Unique number generated internally for each publish operation
a_increment_cnt	integer	S	Number of successful incremental refreshes since the initial publication or the last full refresh
a_initial_publish	date	S	Date of the initial publication
a_last_increment	date	S	Date of the last successful incremental refresh
a_publish_status	string(128)	S	Status of the last operation
a_refresh_date	date	S	Date of the last successful full refresh

Property	Datatype	Single or repeating	Description
content_caching_ enabled	Boolean	S	Indicates whether source-side caching of content is enabled. The default is F, meaning that caching is not enabled.
content_meta_tags	Boolean	S	If TRUE , the system inserts the document properties as metatags into the HTML files of the ContentDB. The default is FALSE.
effective_label	string(32)	S	Used to determine which documents are valid for publishing. The export operation examines the values in the a_effective_date and a_expiration_date properties of all documents whose a_effective_label property value matches the webc config effective_label property. If unspecified, effective labels are not enforced.
export_directory	string(255)	S	Local directory where exported files are placed for pickup by the transfer agent. This is a required property.
export_properties	Boolean	S	If TRUE, the system creates the PropertiesDB on the target site. If FALSE, the PropertiesDB is not created. The default is FALSE.
extra_arguments	string(255)	R	Additional arguments for this specific SCS configuration. The arguments are specified in a name and value pair format. For example: `store_log TRUE`

Property	Datatype	Single or repeating	Description
is_active	Boolean	S	Indicates whether the system will publish documents described by this webc config object. TRUE (the default) directs the system to publish the documents; FALSE suspends publication.
method_trace_level	integer	S	Controls the trace level for operations. Valid values are the same as those for setServerTraceLevel method. The default is 0.
notification_user	string(32)	S	Reserved for future use
object_name	string(255)	S	This property is inherited from dm_sysobject. However, because it is required for webc config objects, it is included in this table. For webc config objects, object_name is the logical name of the WebCache repository.
owner_name	string(32)	S	This property is inherited from dm_sysobject. For the webc config subtype, the user defined as the owner must have Superuser user privileges.
publish_arguments	string(255)	S	Specifies arguments to pass to the publishing method. Currently, there is one valid argument: -system_validate This is a Boolean argument that determines, in conjunction with the target_validate_required flag in agent.ini, how authentication is conducted. -system_validate is set to T (TRUE) by default.

Property	Datatype	Single or repeating	Description
			Refer to the WebCache documentation for details.
publish_contentless_ props	Boolean	S	Indicates whether to publish documents with no content. The default is F, meaning documents without content are not published.
publish_folder_props	Boolean	S	Indicates whether to publish folder properties. The default is F, meaning that folder properties are not published.
send_notification	Boolean	S	Reserved for future use
source_attr_isrep	Boolean	R	Indicates whether the property named in the corresponding index position in source_attrs is a repeating property. TRUE means the property is repeating. FALSE means the property is single-valued.
source_attr_length	integer	R	Length of the property named in the corresponding index position in source_attrs.
source_attr_type	integer	R	Identifies the datatype of the property named in the corresponding index position in source_attrs. Valid values are: 0, for Boolean 1, for Integer 2, for String 3, for ID 4, for Date 5, for Double

Property	Datatype	Single or repeating	Description
source_attrs	string(80)	R	Identifies additional properties to export to the PropertyDB. The specification syntax is: *type_name.property_name* If this property has no values, then only the default properties are exported.
source_folder_id	ID	S	Object ID of the local folder that is the root of the repository structure that matches the WebCache repository. This is a required property.
source_formats	string(64)	R	Document formats to publish. If formats are specified in this property, only documents in the specified formats are published. If a document has multiple formats (a primary format and renditions), only those that are included in the list are published. If this property is unspecified, all existing renditions, including the one in the primary format, are published by default.
source_version	string(32)	S	Version of the WebCache software running on the source. (Informational only)
target_id	ID	R	Object ID of the webc target object that references this webc config object. This is a required property. **Note:** This is a repeating property to allow for future enhancements.

Property	Datatype	Single or repeating	Description
transfer_method	string(32)	S	Reserved for future use.
version_labels	string(32)	R	Identifies the repository version you want to publish. Only documents carrying the specified version are published. Specify only one version label. If you set this to ANY VERSION, the system publishes whichever version is found in the source folder (the source folder is identified in source_folder_id). It is assumed that only one version of any particular document will be in the source folder at a given time. The default is CURRENT. **Note:** This is a repeating property to allow for future enhancements.

Webc Target

Purpose

Describes a Site Caching Services repository on a Site Caching Services target host.

Description

Supertype: SysObject
Subtypes: None
Internal name: dm_webc_target
Object type tag: 08

A webc target object describes a Site Caching Services repository on a Site Caching Services target host.

Properties

Table 254, page 602, lists the properties defined for the type.

Table 254. Properties defined for the webc target type

Property	Datatype	Single or repeating	Description
app_server	string(32)	S	Name of the Web site delivery engine. (Informational only)
app_server_ version	string(32)	S	Version level of the Web site delivery engine. (Informational only)
extra_arguments	string(255)	R	Reserved for future use
global_locales	string(5)	R	Specifies the locales for global publishing. This is only set if global_publishing is set to T.
global_publishing	Boolean	S	Indicates whether global publishing is enabled. The default is F, meaning global publishing is not enabled.

Property	Datatype	Single or repeating	Description
object_name	string(255)	S	This property is inherited from dm_sysobject. However, because it is required for webc target objects, it is included in this table. For webc target objects, object_name is the logical name of the WebCache repository.
online_sync	Boolean	S	Indicates whether the data in the WebCache repository is accessible to users during the synchronization operation. If set to TRUE, the synchronization agent minimizes the possibility that users will be interrupted by repository updates. The default is FALSE.
online_sync_dir	string(255)	S	Remote directory used for the backup copy of the WebCache repository during online updates. This is required if online_sync is TRUE.
owner_name	string(32)	S	This property is inherited from dm_sysobject. For the webc target subtype, the user defined as the owner must have Superuser user privileges.
post_sync_script	string(48)	S	Name of a script to be executed after synchronization. The script must be stored in the WebCache bin directory on the target machine. Refer to the WebCache documentation for details.

Property	Datatype	Single or repeating	Description
pre_sync_script	string(48)	S	Name of a script to be executed prior to synchronization. The script must be stored in the WebCache bin directory on the target machine. Refer to the WebCache documentation for details.
probdb_flags	string(64)	R	Reserved for future use.
propdb_ tablename	string(64)	S	Name to use when creating the PropertyDB tables. Three tables are created using the name in this property: *propdb_tablename*_s *propdb_tablename*_r *probdb_tablename*_m You must specify a name if the export_properties property in the webc config object is set to TRUE.
propdb_ dbversion	string(32)	S	Version level of the target RDBMS. (Informational only)
secure_ connection	string(16)	S	Identifies which agent port to use for the transfer. Valid valus are: • ssl, meaning use the target_ssl_port • raw, meaning use the target_raw_port • both, meaning use the target_ssl_port if available; if not, use the target_raw_port **Note:** Documentum Administrator will not allow you to specify the both option; it must be set through the API. If you publish using the both

Property	Datatype	Single or repeating	Description
			option, you will receive warning messages in the log file.
sync_arguments	string(128)	R	Reserved for future use
target_host	string(128)	S	Host name or IP address of the WebCache host machine.
target_is_active	Boolean	S	Reserved for future use
target_raw_port	integer	S	Port number at which the target Agent process resides. This port is used for unencrypted transmissions. Either this property or target_ssl_port must be set.
target_root_ directory	string(255)	S	The physical directory on the Web server where the files will be placed. This is a required property.
target_ssl_port	integer	S	Port number at which the target Agent process resides. This port is used for encrypted transmissions. Either this property or target_raw_port must be set.
target_version	string(32)	S	Version level of the WebCache software running on the host server. (Informational only)
target_virtual_dir	string(255)	S	The URL for the WebCache installation. This is a required property if you are using WebPublisher™.
transfer_ arguments	string(128)	R	Reserved for future use

Property	Datatype	Single or repeating	Description
transfer_ directory	string(255)	S	Remote directory where exported files are placed for pickup by the Sychronization agent. This property is required if the transfer_method property in the webc config object is set.
transfer_ domain	string(16)	S	For Windows NT, the domain of the user identified in transfer_user. For other platforms, this is unused.
transfer_ protocol	string(16)	S	Reserved for future use
transfer_user	string(32)	S	User name to use to access the remote host in the transfer operation. The default is the local repository owner.
transfer_user_ password	string(255)	S	Password for the user identified in transfer_user.

Chapter 10

Forms Builder Object Types

This chapter contains the reference information for the object types supporting EMC Documentum Forms Builder. The object types are installed when the DAR file provided with EMC Documentum Forms Builder is installed.

XFM Adaptor Config

Purpose

Records information about a forms adaptor.

Description

Supertype: Document
Subtypes: None
Internal name: dmc_xfm_adaptor_config
Object type tag: 09

An xfm adaptor config object stores information about a forms adaptors. An adaptor is executable Java code invoked by the Forms engine, which allows a form to interact with external data sources at runtime. Instances of the type are created internally when EMC Documentum Forms Builder is used to create a form template.

Description

Table 255, page 608, lists the properties defined for the type.

Table 255. Properties defined for the xfm adaptor config type

Property	Datatype	Single or repeating	Description
adaptor_type	string(16)	S	Identifies the type of adaptor. Valid values are: • datasource • item-intializer • item-validator • doc-validator • doc-processor
impl_class	string(255)	S	The implementaion class, with a fully qualified namespace
impl_type	string(16)	S	Class name of the SBO
input_name	string(64)	R	Name of an input

Property	Datatype	Single or repeating	Description
input_required	Boolean	R	Whether an input is required. T means an input is required. The value at a particular index position is applied to the input specified at the corresponding position in input_name.
input_type	string(16)	R	Type of an input. The type specified at a particular index position defines the input specified at the corresponding index position in input_name.
is_public	Boolean	S	Whether this adaptor is public. T means the adaptor is public. F means the adaptor is not public.
output_type	string(64)	S	If the output is a primitive type, it can be string, Boolean, integer, date, and so forth. If the output is a complex type, the value is the name of the complex type.
param_name	string(64)	R	Names of the initialization parameters
param_value	string(255)	R	Values of the initialization parameters. The value at a particular index position is the value of the initialization parameter identified at the corresponding index position in param_name.

XFM Form

Purpose

Contains information about a form template.

Description

Supertype: Document
Subtypes: None
Internal name: dm_xfm_form
Object type tag: 09

An xfm form object stores information about a form template. Instances of the type are created internally when EMC Documentum Forms Builder is used to create a form template.

Properties

Table 256, page 610, lists the properties defined for the type.

Table 256. Properties defined for the xfm form type

Property	Datatype	Single or repeating	Description
definition_state	integer	S	Indicates the status of the form. Valid values are: 0, meaning draft 1, meaning validated 2, meaning installed 3, meaning obsolete
description	string(300)	S	User-defined description of the form

Property	Datatype	Single or repeating	Description
display_mode	integer	S	Indicates the display mode of the form. Valid values are: 0, meaning dialog 1, meaning publish
relationship_to_base	integer	S	Identifies the relationship between the current form and the base form. Valid values are: 0, meaning the form is not related to the base form 1, meaning the form is related to the base form

XFM Instance

Purpose

Represents an instance of a form.

Description

Supertype: Document
Subtypes: None
Internal name: dm_xfm_instance
Object type tag: 09

A xfm instance object contains information about one form. Instances of the type are created internally when EMC Documentum Forms Builder is used to create a form template.

Properties

Table 257, page 612, lists the property defined for the type.

Table 257. Properties defined for the xfm instance type

Property	Datatype	Single or repeating	Description
is_default	Boolean	S	Indicates the form is a form default instance. T (TRUE) means the form is a default instance. F (FALSE) means the form is not a default instance.

XFM Schema

Purpose

Stores information about an XML schema for a form.

Description

Supertype: dm_document
Subtypes: None
Internal name: dm_xfm_schema
Object type tag: 09

An xfm schema object represents an XML schema used with a form. Instances of the type are created internally when EMC Documentum Forms Builder is used to create a form.

Properties

Table 258, page 613, lists the properties defined for the type.

Table 258. Properties defined for the xfm schema type

Property	Datatype	Single or repeating	Description
data_model	string(128)	S	Identifies the root element of the schema.
storage_option	integer	S	Specifies the storage mode for the schema. Valid values are: 0, meaning store as content 1, meaning store as content and use an XML Application to

Property	Datatype	Single or repeating	Description
			populate object properties with the content.
storage_type	string(32)	S	Name of the object type of the object with which the content of the form instance is associated

Chapter 11

Editorial Publishing Services Types

The object types in this chapter are added when the EMC Documentum Content Services for Woodwing product is installed. This product allows users of the Woodwing Smart Connection product to interact with a Documentum Content Server repository.

Some of these types are for internal use by the product only, and are not useful or visible to users. These types are noted within their individual descriptions.

EPS Config Cache

Purpose

Records configuration caching information

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_config_cache
Object type tag: 00

An eps config cache object records configuration caching information. This information is used internally to manage cached information about specific object type configurations.



Caution: Do not modify or delete this type or instances of this type.

Properties

Table 259, page 616, lists the properties defined for the type.

Table 259. Properties defined for the eps config cache type

Property	Datatype	Single/repeating	Description
eps_object_type	string(32)	S	Name of the EPS object type
eps_time_stamp	Time	S	Time stamp of the object type's last modification

EPS Elem Type Mapping

Purpose

Records the object types to use with a particular element type

Description

Supertype: EPS Mapping
Subtypes: None
Internal name: dmc_eps_elem_type_mapping
Object type tag: 0b

An object type eps elem type mapping records the object types that may be used with a particular element type. Objects of this type are created and managed internally.

Properties

Table 260, page 617, lists the properties defined for the type.

Table 260. Properties defined for the eps elem type mapping type

Property	Datatype	Single/repeating	Description
eps_elem_type	string(32)	S	Element type to which the object type is mapped
eps_object_type_name	string(32)	S	Object type mapped to the element type identified in eps_elem_type

EPS Element

Purpose

Serves as the base object type for all element subtypes.

Description

Supertype: Document
Subtypes: None
Internal name: dmc_eps_element
Object type tag: 09

This object type serves as the base object type for all element object types. All element object types are subtypes of eps element and inherit its properties. Those properties record basic information about an element.

Users may create subtypes of the object type through the Content Services for Woodwing object type editor in the Publication Manager.

Properties

Table 261, page 618, lists the properties defined for the type.

Table 261. Properties defined for the eps element type

Property	Datatype	Single/repeating	Description
eps_creation_action	string(32)	S	Creation action type used to create the element
eps_ element_type	string(32)	S	Type of element
eps_issue_id	ID	S	Object ID of the issue that contains the element
eps_publication_id	ID	S	Object ID of the publication that contains the element
eps_route_to_owners	string(2000)	S	Names of users who have the element in their inbox
eps_section_id	ID	S	Object ID of the section that contains the element
eps_state_name	string(255)	S	Name of the workflow state the element is in

Property	Datatype	Single/repeating	Description
eps_workflow_id	ID	S	Object ID of the workflow
eps_workflow_ template_id	ID	S	Object ID of the workflow template that was the source of the workflow in eps_workflow_id

EPS Issue

Purpose

Serves as the base object type for issue object types

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_eps_issue
Object type tag: 0b

This object type serves as the base object type for all issue object types. All issue object types are subtypes of eps issue and inherit its properties. Those properties record basic information about an issue.

Users may create subtypes of the object type through EMC Documentum Application Builder.

Properties

Table 262, page 620, lists the properties defined for the type.

Table 262. Properties defined for the eps issue type

Property	Datatype	Single/repeating	Description
eps_archived	Boolean	S	Whether the object is archived
eps_issue_deadline	Date	S	Date on which issue is to be released
eps_publication_id	ID	S	Object ID of the publication to which this issue belongs

EPS Mapping

Purpose

Serves as the base object type for all mapping tables

Description

Supertype: None
Subtypes: EPS Metadata Mapping, EPS State Perm Mapping, EPS Performer Mapping, EPS Template Mapping, EPS Schedule Mapping
Internal name: dmc_eps_mapping
Object type tag: 00

This object type records mapping information common to all publications. It serves as the base type for all other mapping related object types.



Caution: Do not modify or delete this object type or instances of the type.

Properties

Table 263, page 621, lists the properties defined for the type.

Table 263. Properties defined for the eps mapping type

Property	Datatype	Single/repeating	Description
eps_issue_id	ID	S	Issue to which the mapping applies
eps_publication_id	ID	S	Publication to which the mapping applies
eps_section_name	string(255)	S	Name of the section to which this mapping applies

EPS Metadata Mapping

Purpose

Records the properties visible on each Woodwing dialog box

Description

Supertype: EPS Mapping
Subtypes: None
Internal name: dmc_eps_metadata_mapping
Object type tag: 0b

An eps metadata mapping object records the properties of an element that are exposed on a Woodwing dialog box. Objects of this type are created and managed internally.

Properties

Table 264, page 622, lists the properties defined for the type.

Table 264. Properties defined for the eps metadata mapping type

Property	Datatype	Single/repeating	Description
eps_attribute_names	string(255)	R	Names of the visible properties
eps_element_type	string(32)	S	Element type to which this object applies
eps_woodwing_dialog	string(32)	S	Name of the Woodwing dialog box on which these properties are visible

EPS Page Detail

Purpose

Records information about a page in an element in a publication.

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_page_detail
Object type tag: 00

An eps page detail object records information about a page within an element.

Caution: Do not modify or delete this type or instances of this type.

Properties

Table 265, page 623, lists the properties defined for the type.

Table 265. Properties defined for the eps page detail type

Property	Datatype	Single/repeating	Description
eps_element_id	ID	S	Object ID of the element for which this object records page details
eps_page_height	string(32)	S	Page height
eps_page_number	string(32)	S	Page number
eps_page_order	string(32)	S	Page order
eps_page_width	string(32)	S	Page width

EPS Performer Mapping

Purpose

Records performer information

Description

Supertype: EPS Mapping
Subtypes: None
Internal name: dmc_eps_performer_mapping
Object type tag: 0b

An eps performer mapping object records information about a particular performer. Objects of this type are created and managed internally.

Properties

Table 266, page 624, lists the properties defined for the type.

Table 266. Properties defined for the eps performer mapping type

Property	Datatype	Single/repeating	Description
eps_performer	string(255)	S	Identifes the performer to whom this mapping applies
eps_performer_can_revert	Boolean	S	Whether the performer can revert object states
eps_performer_group_name	string(255)	S	Name of the group that records performer users

EPS Placement

Purpose

Records information about placed articles and the metadata associated with the link.

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_placement
Object type tag: 00

An eps placement object records information about placed articles and the metadata associated with the link.

Caution: Do not modify or delete this type or instances of this type.

Properties

Table 267, page 625, lists the properties defined for the type.

Table 267. Properties defined for the eps placement type

Property	Datatype	Single/repeating	Description
eps_child_id	ID	S	Object ID of the child in the placement
eps_parent_id	ID	S	Object ID of the parent in the placement
eps_placement_props	string(255)	R	Properties of the placement

EPS Pubinfo Cache

Purpose

Records the last modification date of a publication

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_pubinfo_cache
Object type tag: 00

An eps pubinfo cache object records the last modification time of a publication. Objects of this type are used internally to manage publications.



Caution: Do not modify or delete this type or instances of this type.

Properties

Table 268, page 626, lists the properties defined for the type.

Table 268. Properties defined for the eps pubinfo cache type

Property	Datatype	Single/repeating	Description
eps_publication_id	ID	S	Object ID of the publication
eps_last_mod_time	integer	S	The time of the publication's last modification

EPS Publication

Purpose

Serves as the base object type for all publication object types.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_eps_publication
Object type tag: 0b

This object type serves as the base object type all publication object types. All publication object types are subtypes of eps publication and inherit its properties. Those properties record basic information about a publication.

Users may create subtypes of the object type through EMC Documentum Application Builder.

Properties

Table 269, page 627, lists the properties defined for the type.

Table 269. Properties defined for the eps publication type

Property	Datatype	Single/repeating	Description
eps_archived	Boolean	S	Whether the publication is archived
eps_section_names	string(255)	R	Names of the sections in the publication

EPS Relation

Purpose

Associates articles, layouts, and images.

Description

Supertype: Relation
Subtypes: None
Internal name: dmc_eps_relation
Object type tag: 37

An eps relation object is used to establish the links between an article and any images it contains and between the article and its layout. The objects are created automatically as needed.

Properties

Table 270, page 628, lists the properties defined for the type.

Table 270. Properties defined for the eps relation type

Property	Datatype	Single/repeating	Description
eps_child_version	string(32)	S	Version of the child object used in the relationship
eps_parent_version	string(32)	S	Version of the parent object used in the relationship
eps_placement_type	string(255)	S	Whether this represents a native relationship or a placeholder relationship.

EPS Schedule Mapping

Purpose

Records calendar information

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_schedule_mapping
Object type tag: 00

An eps schedule mapping object records calendar information, including workdays and holidays for each year.

Caution: Do not modify or delete this type or instances of this type.

Properties

Table 271, page 629, lists the properties defined for the type.

Table 271. Properties defined for the eps schedule mapping type

Property	Datatype	Single/repeating	Description
eps_holidays	string(32)	R	List of the holidays in the year
eps_year	integer	S	Calendar year to which the information applies
eps_weekend_days	integer	R	List of the weekends in the year
eps_workday_ overrides	string(32)	R	Workday overrides

EPS Section

Purpose

Serves as the base object type for all section object types.

Description

Supertype: Folder
Subtypes: None
Internal name: dmc_eps_section
Object type tag: 0b

This object type serves as the base object type for all section object types. All section object types are subtypes of eps section and inherit its properties. Those properties record basic information about a sections.

Users may create subtypes of the object type through EMC Documentum Application Builder.

Properties

Table 272, page 630, lists the properties defined for the type.

Table 272. Properties defined for the eps section type

Property	Datatype	Single/repeating	Description
eps_issue_id	ID	S	Object ID of the issue that contains this section.
eps_publication_id	ID	S	Object ID of the publication that contains this section.

EPS State

Purpose

Records information about states

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_state
Object type tag: 00

An eps state object records information about states in an editorial workflow process.



Caution: Do not modify or delete this type or instances of this type.

Properties

Table 273, page 631, lists the properties defined for the object type.

Table 273. Properties defined for the eps state type

Property	Datatype	Single/repeating	Description
eps_produce_pdf_flag	Boolean	S	Default value for the produce PDF flag for this element
eps_required_ permission	integer	S	Permission level required to perform the task associated with this state
eps_state_color	string(32)	S	State color
eps_state_duration	integer	S	Expected duration of the state, in days
eps_state_name	string(255)	S	Name of the state
eps_state_permissions	string(300)	R	List of performer types and their permissions

EPS State Mapping

Purpose

Records state configuration information for publications

Description

Supertype: EPS Mapping
Subtypes: None
Internal name: dmc_eps_state_mapping
Object type tag: 0b

An eps state mapping object records configuration information about a state for a particular element in a publication. Objects of this type are created and managed internally.

Properties

Table 274, page 632, lists the properties defined for the type.

Table 274. Properties defined for the eps state mapping type

Property	Datatype	Single/repeating	Description
eps_creation_action	string(32)	S	The creation action type to which this state information applies
eps_element_type	string(32)	S	The element type to which this state information applies
eps_produce_pdf	Boolean	S	Whether to generate a PDF preview of the elements in this state
eps_state_color	string(32)	S	Color to associate with elements in this state
eps_state_duration	integer	S	Expected length of time, from start to finish, for the completion of the state
eps_state_name	string(255)	S	Name of the state

EPS State Perm Mapping

Purpose

Records the permissions held by performers for each state.

Description

Supertype: EPS Mapping
Subtypes: None
Internal name: dmc_eps_state_perm_mapping
Object type tag: 0b

An eps state perm mapping object records the permission held by a performer for a particular state in an editorial workflow process. Objects of this type are created and managed internally.

Properties

Table 275, page 633, lists the properties defined for the type.

Table 275. Properties defined for the eps state perm mapping type

Property	Datatype	Single/repeating	Description
eps_creation_action	string(32)	S	The creation action type to which the permission applies
eps_element_type	string(32)	S	The element type to which the permission applies
eps_performer	string(255)	S	Performer who holds this permission
eps_performer_ permission	integer	S	Permission level assigned to the performer identified in epss_performer
eps_state_name	string(255)	S	Name of the state to which this permission applies

EPS State Transition

Purpose

Records all state transitions

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_state_transition
Object type tag: 00

An eps state transition object records a state transition. These objects are used internally, for reporting on issue and element status.



Caution: Do not modify or delete this type or instances of this type.

Properties

Table 276, page 634, lists the properties defined for the type.

Table 276. Properties defined for the eps state transition type

Property	Datatype	Single/repeating	Description
eps_element_id	ID	S	Object ID of the element participating in the state transition
eps_end_state	string(32)	S	Name of the end state in the transition
eps_start_state	string(32)	S	Name of the start state in the transition
eps_time_stamp	string(32)	S	Time stamp of the transition

EPS Template Mapping

Purpose

Records the templates to use for element types

Description

Supertype: EPS Mapping
Subtypes: None
Internal name: dmc_eps_template_mapping
Object type tag: 0b

An eps template mapping object records the template to be used for a particular element type. Objects of this type are created and managed internally.

Properties

Table 277, page 635, lists the properties defined for the type.

Table 277. Properties defined for the eps template mapping type

Property	Datatype	Single/repeating	Description
eps_element_type	string(32)	S	Name of the element type
eps_template_id	ID	S	Object ID of the template obect representing the template to used with the element identified in eps_element_type

EPS Transition Override

Purpose

Records scheduling changes to the master schedule configured for an issue

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_transition_override
Object type tag: 00

An eps transition override object records a schedule change for an element within an issue.



Caution: Do not modify or delete this type or instances of this type.

Properties

Table 278, page 636, lists the properties defined for the type.

Table 278. Properties defined for the eps transition override type

Property	Datatype	Single/repeating	Description
eps_element_id	ID	S	Object ID of the element participating in the state transition
eps_state	string(255)	S	Name of the state in the transition
eps_start_duration	integer	S	Duration of the state
eps_workflow_ template_id	ID	S	Object ID of the workflow template

EPS WF Template Mapping

Purpose

Identifies the workflow template to be used with an element and creation action type

Description

Supertype: EPS Mapping
Subtypes: None
Internal name: dmc_eps_wf_template_mapping
Object type tag: 0b

An eps wf template mapping object specifies which workflow template is to be used with a particular combination of element type and creation action type. Objects of this type are created and managed internally.

Properties

Table 279, page 637, lists the properties defined for the type.

Table 279. Properties defined for the eps wf template mapping type

Property	Datatype	Single/repeating	Description
eps_creation_action	string(32)	S	The creation action type with which the workflow template is associated
eps_element_type	string(32)	S	Name of the element type
eps_workflow_ template_id	ID	S	Object ID of the workflow template object representing the workflow template to used with the element identified in eps_element_type

EPS Workflow Data

Purpose

Records information to be passed to the Java workflow method

Description

Supertype: None
Subtypes: None
Internal name: dmc_eps_workflow_data
Object type tag: 00

An eps workflow data object records information that is passed to a Java workflow method.



Caution: Do not modify or delete this type or instances of this type.

Properties

Table 280, page 638, lists the properties defined for the type.

Table 280. Properties defined for the eps workflow data type

Property	Datatype	Single/repeating	Description
eps_element_id	ID	S	Object ID of the element to which this information applies
eps_performer_name	string(255)	S	Name of the performer to whom the task is assigned or delegated
eps_state_name	string(255)	S	Name of the state to set the object to
eps_workflow_id	ID	S	Object ID of the workflow to which this information is passed

Appendix A

Language and Country Codes

This appendix lists the recommended language codes and some of the recommended country codes for use in the language_code property of SysObjects. The language codes are taken from ISO 639. The country codes are taken from the larger set found in ISO 3116.

Table 281. Recommended language codes

Language	Code	Lanuage	Code
Albanian	sq	Arabic	ar
Armenian	hy	Azerbaijani	az
Basque	eu	Bulgarian	bg
Byelorussion	be	Cambodian	km
Catalan	ca	Chinese	zh
Croatian	hr	Czech	cs
Danish	da	Dutch	nl
English	en	Estonian	et
Finnish	fi	French	fr
Georgian	ka	German	de
Greek	el	Hebrew	he
Hungarian	hu	Icelandic	is
Indonesian	id	Italian	it
Japanese	ja	Kazakh	kk
Korean	ko	Latvian, Lettish	lv
Lithuanian	lt	Macedonian	mk
Malay	ms	Moldavian	mo

Language	Code	Lanuage	Code
Norwegian	no	Persian	fa
Polish	pl	Portuguese	pt
Romanian	ro	Russian	ru
Serbian	sr	Serbo-Croatian	sh
Slovak	sk	Slovenian	sl
Spanish	es	Swedish	sw
Thai	th	Turkish	tr
Ukranian	uk	Uzbek	uz
Vietnamese	vi		

Table 282. Recommended country codes

Country	Code	Country	Code
Argentina	AR	Austria	AT
Australia	AU	Azerbaijan	AZ
Belarus	BY	Belgium	BE
Belize	BZ	Bolivia	BO
Brazil	BR	Bulgaria	BG
Cambodia	KH	Cameroon	CM
Canada	DA	Chile	CL
China	CN	Colombia	CO
Costa Rica	CR	Croatia	HR
Czech Republic	CZ	Denmark	DK
Ecuador	EC	Egypt	EG
Estonia	EE	Finland	FI
France	FR	Gabon	GA
Gambia	GM	Georgia	GE
Germany	DE	Great Britain (UK)	GB
Greece	GR	Greenland	GL
Guatemala	GT	Haiti	HT
Honduras	HN	Hong Kong	HK

Country	Code	Country	Code
Hungary	HU	Iceland	IS
India	IN	Indonesia	ID
Iran	IR	Iraq	IQ
Ireland	IE	Israel	IL
Italy	IT	Japan	JP
Jordan	JO	Kazachstan	KZ
Kenya	KE	Korea (North)	KP
Korea (South)	KR	Kyrgyz Republic	KG
Laos	LA	Latvia	LV
Lebanon	LB	Libya	LY
Liechstenstein	LI	Lithuania	LT
Luxemborg	LU	Mexico	MX
Mozambique	MZ	Morocco	MA
Netherlands	NL	New Zealand	NZ
Nicaragua	NI	Norway	NO
Pakistan	PA	Panama	PK

Appendix B

RDBMS Tables for Documentum Types

All the objects in Content Server, and their property values, are stored in a repository. This repository is part of a larger database, the relational database (RDBMS). This appendix contains an explanation and an example of the RDBMS tables that describe a Documentum type.

Note: The example tables assume that the underlying RDBMS is Oracle for the datatype descriptions.

Overview

Within the relational database, each object type is represented by two tables and two views:

- A table and a view for the object's single-valued properties, and
- A table and a view for the object's repeating properties.

The tables

The name of the table describing an object's single-valued properties has the format:

*typename*_s

For example, the table that contains the single-valued properties for the document type is the dm_document_s table.

The names of the tables describing the repeating properties have the following format:

*typename*_r

For example, the table that contains the repeating properties of the document type is the dm_document_r table.

The views

The views join the a type's tables with the tables of its supertypes. For example, the view of the single-valued properties for folders joins the dm_folder_s and dm_sysobject_s tables. For cabinets, the view joins the dm_cabinet_s, dm_folder_s, and dm_sysobject_s tables.

The names of the views that are built on the single-valued property tables have the following format:

*typename*_sp

For example, the single-valued property view associated for dm_document type is named dm_document_sp.

Returning database table information

To obtain a listing of a table's columns and their datatypes in an Oracle database, you use the SQL Describe statement. For example, the following statement returns a description of the table that stores the SysObject type's single-valued properties:

```
sql> describe dm_sysobject_s;
```

The user type tables in the Oracle RDBMS

This section contains examples of the _s and _r tables, with the dm_user type as the demonstrative type. The views are not shown as they do not differ for this type, as it has no supertypes.

Table 283, page 644 lists the results of the following statement, which retrieves the columns of the dm_user_s table:

```
sql>describe dm_user_s
```

Table 283. Columns of the dm_user_s table

Column Name	Datatype	Nullable (Yes/No)
R_OBJECT_ID	CHAR(16)	N
GROUP_NAME	CHAR(32)	N
GROUP_ADDRESS	CHAR(80)	N
I_VSTAMP	NUMBER(10)	N
I_ATTR_EXTRA2	CHAR(64)	Y
I_ATTR_INT1	NUMBER(10)	Y

Column Name	Datatype	Nullable (Yes/No)
I_ATTR_INT2	NUMBER(10)	Y
I_VSTAMP	NUMBER(10)	Y

Table 284, page 645 shows the results of the following statement, which retrieves a description of the dm_user_r table:

```
sql>describe dm_user_r;
```

Table 284. Columns of the dm_user_r table

Column Name	Datatype	Nullable (Yes/No)
R_OBJECT_ID	NUMBER(10)	N
I_POSITION	NUMBER(6)	N
USER_NAMES	CHAR(32)	Y
GROUPS_NAMES	CHAR(32)	Y
I_ALL_USERS_NAMES	CHAR(16)	Y

Appendix C

Obsolete, Deprecated, or Unused Object Types

This appendix lists object types that are obsolete, deprecated, or unused. An obsolete type was used in prior releases to support some functionality. An unused object type was added to the hierarchy, but never used by Content Server or any EMC Documentum client products.

The following object types are obsolete or unused:

- dm_java
- dm_staged
- dm_router

The following object types are supported but deprecated because they have been superceded by another object type or are used by deprecated features or products:

- dm_format_preferences

  This object type is used by Desktop Client to record information about which rendition and application to use for editing and viewing a document in a particular format.

- dm_menu_system

  This object type is used by Desktop Client to record the Explorer Integration menu configuration.

- dm_email_message

  The reference information for this type is still documented. It is found in Email Message, page 196.

Index

_ (underbar)
    in table and column names, 28
_ (underscore), in type names, 28

A

_accessor_app_permit (computed property), 15
_accessor_name (computed property), 16
_accessor_permit (computed property), 16
_accessor_permit_type (computed property), 16
_accessor_xpermit (computed property), 16
_accessor_xpermit_names (computed property), 17
ACL object type, 40
_acl_ref_valid (computed property), 17
ACLs
    aliases in, 66
    dm_audittrail_acl type, 83
    dmi_audittrail_attrs type, 85
acs config object type, 44
ACS servers
    cont transfer config object type, 135
activity group instance, 62
activity object type, 49
agent exec process
    dm_agent_exec utility, 236
    method name in server config object, 371
aggr domain object type, 64
alias set object type, 66
_alias_set (computed property), 17
_alias_sets (computed property), 23
_all_users_names (computed property), 17
_allow_change_folder_links (computed property), 17
_allow_change_location (computed property), 17
_allow_change_owner (computed property), 17
_allow_change_permit (computed property), 17
_allow_change_state (computed property), 17
_allow_execute_proc (computed property), 17
annotations
    note object type, 294
app ref object type, 68
application object type, 69
aspect properties
    referencing in DQL, 26
aspect relation object type, 72
aspect type object type, 73
assembly object type, 74
attachments
    wf attachment object type, 442
attachments folder object type, 76
_attribute_list_values (computed property), 18
audit trail acl object type, 83
audit trail attrs object type, 85
audit trail entries
    _sign_data computed property, 22
audit trail group object type, 86
audit trail object type, 77
audit trail signatures
    _sign_data computed property, 22
auditing
    audit trail objects, 77
auth config object type, 89

B

backing up
    dumping repositories, 193
blob storage areas

object type, 90
renditions in, 380
blobstore object type, 90
bocs config object type, 91
BOCS servers
cont transfer config object type, 135
dm_bocs_config type, 91
BOF modules
module object type, 282
Branch method, 37
builtin expr object type, 94

C

CA store object type, 95
cabinet object type, 98
cabinets
privileges to change, 98
cache config object type, 99
_cached (computed property), 18
caches
client_cache_size property, 373
dump process, 193
query, 496
size described in server config object, 371
calendar object type, 522
case sensitivity
group names, 28
table and column names, 28
type and property names, 28
user names, 28
category assign object type, 104
category class object type, 107
category object type, 101
change record object type, 110
_changed (computed property), 18
Checkin method, 37
child security for relationships, 32
child_id attribute, 31
child_label attribute, 31
CI config object type, 112
class object type, 115
client applications
date formats, 496
client cache
maximum size, defining, 373
client caches
cache config object type, 99
client config object type, 486
client registration object type, 117
client rights object type, 118
client_cache_size property, 373
collection object identifiers, 29
columns
naming rules, 28
comment object type, 526
completed workflow object type, 120
completed workitem object type, 124
component object type, 129
component routine, 325
_componentID (computed property), 18
composite predicate object type, 130
computed properties
_accessor_app_permit, 15
_accessor_permit_type, 16
_is_restricted_session, 20
_istransactionopen, 20
lifecycles, 23
list of, 15
referencing, 26
_sign_data, 22
cond expr object type, 131
cond ID expr object type, 132
config scope relation object type, 133
configuration objects
client config, 486
connection config, 487
docbase config, 176
location, 261
mount point, 287
server config, 371
session config, 494
connection brokers
docbase locator object type, 489
docbroker locator object type, 491
server locator object type, 492
connection config objects, 487
constraint set object type, 134
cont transfer config object type, 135
_containID (computed property), 19
containment object type, 138
content assignment policies
dm_relation_ssa_policy type, 347
dm_ssa_policy type, 386
content object type, 141
Content Server
server config object type, 371
server locator object type, 492
content-addressed storage

dm_ca_store type, 95
_content_state (computed property), 19
copy_child property, 33
_count (computed property), 19
country codes, 639
_current_state (computed property), 19

D

data dictionary
aggr domain object type, 64
dd attr info object type, 151
dd common info object type, 156
dd info object type, 161
dd type info object type, 167
domain object type, 190
expr code object type, 200
nls dd info object type, 291
value assist object type, 435
value func object type, 437
value list object type, 438
value query object type, 439
datatable object type, 528
datatable row object type, 529
datatable schema ex object type, 531
datatable schema object type, 530
datatable settings object type, 532
datatypes
default values, 24
properties, 22, 24
units of length for string, 26
dates
format for clients, 496
dd attr info object type, 151
dd common info object type, 156
dd info object type, 161
dd type info object type, 167
default values
datatypes, 24
procedure timeout period, 279
destroying
relationships, effect on, 37
DFC
client config objects and, 486
direction_kind property, 34
discussion object type, 533
discussions
dmc_comment type, 526
dmc_readcomment type, 535
dmc_topic type, 540
display config type, 170
distributed store object type, 172
dm_ prefix, 27
dm_acl type, 40
dm_acs_config type, 44
dm_activity type, 49
dm_agent_exec utility, 236
dm_aggr_domain type, 64
dm_alias_set type, 66
dm_app_ref type, 68
dm_application type, 69
dm_assembly type, 74
dm_attachments_folder type, 76
dm_audittrail type, 77
dm_audittrail_acl type, 83
dm_audittrail_group type, 86
dm_auth_config type, 89
dm_blob_store type, 90
dm_bocs_config type, 91
dm_builtin_expr type, 94
dm_ca_store type, 95
dm_cabinet type, 98
dm_cache_config type, 99
dm_category type, 101
dm_category_assign type, 104
dm_category_class type, 107
dm_ci_config type, 112
dm_client_registration type, 117
dm_client_rights type, 118
dm_component type, 129
dm_cond_expr type, 131
dm_cond_id_expr type, 132
dm_cont_transfer_config type, 135
dm_dd_info type, 161
dm_display_config type, 170
dm_distributedstore type, 172
dm_dms_config type, 174
dm_docbase_config type, 176
dm_docset type, 186
dm_docset_run type, 187
dm_document type, 189
dm_domain type, 190
dm_dump_record type, 193
dm_email_message type, 196
dm_esign_template type, 198
dm_expression type, 201
dm_extern_file type, 203
dm_extern_free type, 204
dm_extern_store type, 205
dm_extern_url type, 207

dm_federation type, 208
dm_filestore type, 211
dm_folder type, 212
dm_foreign_key type, 213
dm_format type, 214
dm_format_preferences type, 647
dm_ftengine_config type, 221
dm_ftindex_agent_config type, 217
dm_fulltext_index type, 222
dm_func_expr type, 224
dm_group type, 227
dm_java type, 647
dm_jms_config type, 235
dm_job type, 236
dm_job_request type, 241
dm_job_sequence type, 243
dm_key type, 245
dm_ldap_config type, 246
dm_lightweight type, 480
dm_linkedstore type, 253
dm_literal_expr type, 256
dm_load_record type, 259
dm_location type, 261
dm_locator type, 263
dm_media_profile type, 264
dm_menu_system type, 647
dm_message_address type, 266
dm_message_archive type, 268
dm_message_attachment type, 270
dm_message_container type, 271
dm_message_route_user_data type, 273
dm_message_user_data type, 274
dm_method type, 276
dm_mount_point type, 287
dm_network_location_map type, 289
dm_nls_dd_info type, 291
dm_note type, 294
dm_outputdevice type, 296
dm_plugin type, 300
dm_policy type, 302
dm_procedure type, 312
dm_process type, 313
dm_public_key_certificate, 323
dm_qual_comp type, 325
dm_query type, 327
dm_reference type, 335
dm_registered type, 340
dm_relation type, 345
dm_relation_ssa_policy type, 347
dm_relation_type type, 348
dm_retainer type, 359
dm_router type, 647
dm_scope_config type, 368
dm_script type, 369
dm_server_config type, 371
dm_smart_list type, 382
dm_ssa_policy type, 386
dm_staged type, 647
dm_state_extension type, 384
dm_state_type type, 385
dm_store type, 387
dm_sysobject type, 395
dm_sysprocess_config type, 408
dm_taxonomy type, 409
dm_type type, 416
dm_user type, 422
dm_validation_descriptor type, 430
dm_value_assist type, 435
dm_value_func type, 437
dm_value_list type, 438
dm_value_query type, 439
dm_webc_config object type, 596
dm_webc_target type, 602
dm_WfReporting job
completed workflow object type, 120
dm_workflow type, 451
dm_xfm_form type, 610
dm_xfm_instance, 612
dm_xml_application type, 472
dm_xml_config type, 474
dm_xml_custom_code type, 475
dm_xml_style_sheet type, 476
dm_xml_zone type, 477
dmc dsm application object type, 544
dmc dsm backbone object type, 545
dmc dsm doc propertie sobject type, 546
dmc dsm drug product object type, 548
dmc dsm drug substance object typte, 549
dmc dsm excipient object type, 550
dmc dsm facilities equip object type, 551
dmc dsm indication object type, 552
dmc dsm m1 backbone object type, 553
dmc dsm safety eval object type, 554
dmc dsm sect doc attributes object
type, 555
dmc dsm section extension object
type, 559
dmc dsm section object type, 557
dmc dsm sft section object type, 562
dmc dsm stf backbone object type, 560

dmc dsm study attributes, 563
dmc dsm study report object type, 564
dmc dsm submission object type, 565
dmc rps authority object type, 568
dmc rps execution rule object type, 580
dmc rps notification object type, 583
dmc_act_group_instance, 62
dmc_aspect_relation type, 72
dmc_aspect_type type, 73
dmc_calendar type, 522
dmc_class type, 115
dmc_comment type, 526
dmc_completed_workflow type, 120
dmc_completed_workitem type, 124
dmc_composite_predicate type, 130
dmc_config_scope_relation type, 133
dmc_constraint_set type, 134
dmc_datatable type, 528
dmc_datatable_row type, 529
dmc_datatable_schema type, 530
dmc_datatable_schema_ex type, 531
dmc_datatable_settings type, 532
dmc_discussion type, 533
dmc_dsm_application type, 544
dmc_dsm_backbone type, 545
dmc_dsm_doc_properties type, 546
dmc_dsm_drug_product type, 548
dmc_dsm_drug_substance type, 549
dmc_dsm_excipient type, 550
dmc_dsm_facilities_equip, 551
dmc_dsm_indication type, 552
dmc_dsm_m1_backbone type, 553
dmc_dsm_sect_doc_attributes type, 555
dmc_dsm_section type, 557
dmc_dsm_section_extension type, 559
dmc_dsm_stf_backbone type, 560
dmc_dsm_stf_section type, 562
dmc_dsm_study_attributes, 563
dmc_dsm_study_report type, 564
dmc_dsm_submission type, 565
dmc_eps_config_cache type, 616
dmc_eps_elem_type_mapping type, 617
dmc_eps_element type, 618
dmc_eps_issue type, 620
dmc_eps_mapping type, 621
dmc_eps_metadata_mapping type, 622
dmc_eps_performer_mapping type, 624
dmc_eps_placement type, 625
dmc_eps_pubinfo_cache type, 626
dmc_eps_publication type, 627
dmc_eps_relation type, 628
dmc_eps_schedule_mapping type, 629
dmc_eps_section type, 630
dmc_eps_state type, 631
dmc_eps_state_mapping type, 632
dmc_eps_state_perm_mapping type, 633
dmc_eps_state_transition type, 634
dmc_eps_template_mapping type, 635
dmc_eps_transition_override type, 636
dmc_eps_wf_template_mapping type, 637
dmc_eps_workflow_data type, 638
dmc_jar type, 233
dmc_java_library type, 234
dmc_metamodel type, 275
dmc_module type, 282
dmc_module_config type, 285
dmc_notepage type, 534
dmc_page_detail type, 623
dmc_preset_info type, 310
dmc_preset_package type, 311
dmc_process_correlation_set type, 322
dmc_process_parameter type, 320
dmc_reaadcomment type, 535
dmc_relationship_def type, 352
dmc_richtext type, 536
dmc_room type, 537
dmc_routecase_condition type, 364
dmc_rps_authority type, 568, 570
dmc_rps_base_date type, 571
dmc_rps_child_strategy type, 572
dmc_rps_condition type, 574
dmc_rps_contact type, 575
dmc_rps_disposition_method type, 576
dmc_rps_event type, 578
dmc_rps_execution_rule type, 580
dmc_rps_hold type, 581
dmc_rps_notification type, 583
dmc_rps_phase_rel, 584
dmc_rps_retainer type, 586
dmc_rps_retainer_event_rel type, 589
dmc_rps_retention_policy type, 590
dmc_tcf_activity type, 410
dmc_tcf_activity_template type, 411
dmc_topic type, 540
dmc_transition_condition type, 412
dmc_validation_module type, 431
dmc_validation_relation type, 433
dmc_wf_package_schema type, 445
dmc_wf_package_skill type, 446

dmc_wf_package_type_info type, 448
dmc_wfsd_element type, 501
dmc_wfsd_element_boolean type, 502
dmc_wfsd_element_date type, 503
dmc_wfsd_element_double type, 504
dmc_wfsd_element_integer type, 505
dmc_wfsd_element_parent type, 506
dmc_wfsd_element_string type, 507
dmc_wfsd_type_info type, 508
dmc_wfsdrp_boolean type, 513
dmc_wfsdrp_date type, 514
dmc_wfsdrp_double type, 515
dmc_wfsdrp_integer type, 516
dmc_wfsdrp_string type, 519
dmc_workqueue type, 461
dmc_workqueue_category type, 462
dmc_workqueue_doc_profile type, 463
dmc_workqueue_policy type, 464
dmc_workqueue_user_profile type, 466
dmc_wpr_parent type, 443
dmc_wq_skill_info type, 467
dmc_wq_task_skill type, 470
dmc_wq_user_skill type, 471
dmc_xfm_adaptor_config type, 608
DMCL (client library)
    Docbroker locator object, 491
    server locator object, 492
dmi_audittrail_attrs type, 85
dmi_change_record type, 110
dmi_dd_attr_info type, 151
dmi_dd_common_info type, 156
dmi_dd_type_info type, 167
dmi_dump_object_record type, 192
dmi_expr_code type, 200
dmi_index object type, 231
dmi_linkrecord type, 254
dmi_load_object_record type, 257
dmi_otherfile type, 295
dmi_package type, 297
dmi_queue_item type, 328
dmi_registry type, 342
dmi_replica_record type, 357
dmi_sequence type, 370
dmi_session type, 381
dmi_subcontent type, 393
dmi_type_info type, 419
dmi_vstamp type, 441
dmi_wf_attachment type, 442
dmi_wf_timer type, 449
dmi_workitem type, 456
dmr_containment type, 138
dmr_content type, 141
dms config object type, 174
DO_METHOD procedures
    object type for, 276
docbase config object type, 176
docbase locator object type, 489
_docbase_id (computed property), 19
Docbasic
    method object and, 278
    plugins and, 301
    procedure objects and, 312
docbroker locator object type, 491
docset object type, 186
docset run object type, 187
document object type, 189
Document Set, *see* docset object type
documents
    language and country codes, 639
Documentum Collaborative Services
    room object type, 537
Documentum Resource Locator, object
    type for, 263
domain controllers
    identifying, 89
domain object type, 190
domain-required mode, 178
domains, 190
DQL (Document Query Language)
    names, 27
    naming rules, 27
    query objects and, 327
    referencing properties, 26
    reserved words in queries, 27
dsm_dmc_safety_eval type, 554
_dump (computed property), 19
dump object record object, 192
dump record object type, 193

E

electronic mail address
    group, 227
    user, 425
electronic signatures
    Esign template object type, 198
email message object type, 196
EMC Documentum Forms Builder
    xfm form object type, 610
    xfm instance object type, 612

_entry_criteria (computed property), 23
eps config cache object type, 616
eps elem type mapping object type, 617
eps element object type, 618
eps issue object type, 620
eps mapping object type, 621
eps metadata mapping object type, 622
eps page detail, 623
eps performer mapping type, 624
eps placement object type, 625
eps pubinfo cache object type, 626
eps publication object type, 627
eps relation object type, 628
eps schedule mapping object type, 629
eps section object type, 630
eps state mapping object type, 632
eps state perm mapping object type, 633
eps state transition object type, 634
eps state type object type, 631
eps template mapping object type, 635
eps transition override object type, 636
eps wf template mapping object type, 637
eps workflow data object type, 638
error level in _status property, 22
Esign template object type, 198
events
    registry object type, 342
    SysObject properties, 406
explicit transactions
    _isdeadlocked, 20
    _istransactionopen (computed property), 20
expr code object type, 200
expression object type, 201
extended permissions, computed
    properties for, 23
external applications
    app ref object type, 68
    application object type, 69
    qual comp object type, 325
external file store object type, 203
external free store object type, 204
external procedures
    method object type and, 276
external storage
    external free store object type, 204
    external store object type, 205
    external URL object type, 207
    plug-in libraries, 300
external store object type, 205
external URL object type, 207

F

federation object type, 208
file store object type, 211
folder object type, 212
foreign key object type, 213
format object type, 214
ft index agent config object type, 217
ftengine config object type, 221
full-text indexes
    fulltext index object type, 222
fulltext index object type, 222
fulltext indexes
    ftengine config object type, 221
func expr object type, 224

G

group object type, 227
groups
    aliases, 66
    _all_users_names (computed property), 17
    dm_audittrail_group type, 86
    group object type, 227
    names, case sensitivity, 28
    naming rules, 28

H

_has_config_audit (computed property), 19
_has_create_cabinet (computed property), 19
_has_create_group (computed property), 19
_has_create_type (computed property), 19
_has_purge_audit (computed property), 20
_has_superuser (computed property), 20
_has_sysadmin (computed property), 20
_has_view_audit (computed property), 20

I

i_is_replica property, 39
i_vstamp property, 39
_id (computed property), 20

identifiers
collection object, 29
names, 27
object, 28
property, 29
session, 29
type, 29
_included_types (computed property), 23
index agents
ft index agent config object type, 217
index object type, 231
integrity_kind property, 35
_is_restricted_session (computed property), 20
_isdeadlocked (computed property), 20
_isnew (computed property), 20
_isreplica (computed property), 20
_istransactionopen (computed property), 20

J

jar files
jar object type, 233
jar object type, 233
Java
java library object type, 234
java library object type, 234
JMS config object type, 235
job object type, 236
job request object type, 241
job sequence object type, 243
jobs
job object type, 236
job sequence object type, 243

K

key object type, 245
keys
foreign key object type, 213
key object type, 245

L

language codes, 639
ldap config object type, 246
LDAP directory server
ldap config object type, 246
_lengths (computed property), 21
lifecycle states
attachability, 302
lifecycles
computed properties, 23
object type for, 302
_policy_name (computed property), 21
state extension object type, 384
state type object type, 385
lightweight object type, 480
link record object type, 254
linked storage areas
link record object type, 254
linked store object type, 253
linked store object type, 253
links
reference object type, 335
literal expr object type, 256
load object record objects, 257
load record object type, 259
location object type, 261
location objects
events directory, 488
external file stores, 203
file stores, 211
linked stores, 253
locator object type, 263
locking
SysObject property for, 405
login ticket key
ticket_crypto_key property, 180
login tickets
login_ticket_cutoff property, 180
trust_by_default property, 184
trusted_docbases property, 185
login_ticket_cutoff property, 180

M

_masterdocbase (computed property), 21
maximums
cache size in server config object, 371
characters in table and column names, 28
characters in type and property names, 28
characters in user and group names, 28
sequence number, 370
media profile object type, 264
message address object type, 266

message archive object type, 268
message attachment object type, 270
message container object type, 271
message route user data object type, 273
message user data object type, 274
metamodel object type, 275
method object type, 276
minimum values
    login tickets, issuance date, 180
module config object type, 285
module object type, 282
mount point object type, 287

N

names
    column, 28
    described, 27
    DQL reserved words as, 27
    group, 28
    property, 28
    repository, 181
    rules for, 27
    server config objects, 376
    table, 28
    type, 28
    user, 28
_names (computed property), 21
network location map object type, 289
_next_state (computed property), 23
nls dd info object type, 291
non-persistent types
    client config, 486
    connection config, 487
    docbase locator, 489
    docbroker locator, 491
    server locator, 492
    session config, 494
none security for relationships, 33
note object type, 294
notepage object type, 534
notes
    note object type, 294
notification in registry object type, 342

O

object identifiers, 28
object IDs
    sequence numbers, 370
object replication
    relationships and, 37
object types
    dd type info, 167
    dm_type type, 416
    dmi_type_info type, 419
    naming rules, 28
    relation, 30
    relation type, 30
objects
    _cached (computed property), 18
    _current_state (computed property), 19
    _policy_name (computed property), 21
    relation type objects, 36
    relationships
        creating, 31
        destroying, 36
    relationships between, 29
other file object type, 295
output device object type, 296

P

package object type, 297
packages
    dmi_package type, 297
    information about, 448
    skill levels for, 446
    wf package report parent object type, 443
    wf package schema object type, 445
parent security for relationships, 32
parent_id attribute, 31
permanent_link attribute, 37
permanent_link property, 33
_permit (computed property), 21
persistent object type, 39
plug-in libraries
    plugin object type, 300
plugin object type, 300
policy object type, 302
_policy_name (computed property), 21
preset info object type, 310
preset package object type, 311
_previous_state (computed property), 24
primary session identifiers, 29
private
    activities, 52

cabinets, 98
groups, 230
mount point objects, 288
process definitions, 315
procedure object type, 312
procedures
timeout period, 279
process correlation set object type, 322
process object type, 313
process parameter object type, 320
properties
aggr_domain object type, 64
aspect properties, referencing, 26
_attribute_list_values (computed property), 18
computed, 15
computed, for lifecycles, 23
datatypes, 24
dd attr info object type, 151
dd common info object type, 156
dm_display_config type, 170
domain object type, 190
identifiers, 29
naming rules, 28
referencing, 26
public
activities, 52
cabinets, 98
groups, 230
mount point objects, 288
process definitions, 315
public key certificate object type, 323

Q

qual comp object type, 325
queries
caching, 496
query object type, 327
queue item object type, 328
quotes
enclosing reserved words in, 27

R

r_object_id property, 39
RDBMS
datatypes, 24
Documentum tables in, 643
index storage areas, 180
reserved words, 28
user name, 425
views, 644
readcomment object type, 535
reference object type, 335
registered object type, 340
registered tables
dm_registered type, 340
registry object type, 342
relation object type, 345
relation objects
creating, 31
described, 30
relation SSA policy object type, 347
relation type object type, 348
relation type objects
creating, 30
described, 30
destroying, 36
relation_name attribute, 31
relationship def object type, 352
relationships
Branch method, 37
Checkin method, 37
child security level, 32
copy behavior, release compatibility, 34
copy_child property, 33
creating instances, 31
delete behavior, defining, 34
described, 29
direction_kind property, 34
integrity_kind property, 35
none security level, 33
object removal, effect of, 37
object replication and, 37
parent security level, 32
permanent_link property, 33
relation object, 30
relation type objects, 30
relationship types, creating, 30
Save method and, 37
Saveasnew method and, 37
security, 32
system security level, 32
user-defined, 30
remote repositories
i_is_replica property, 39
renditions
storage of, 380

_repeating (computed property), 21
repeating properties
    number of in _values property, 23
replica record object type, 357
repositories
    docbase config object type, 176
    names, restrictions, 181
    RDBMS and, 643
    reference object type, 335
    ticket_crypto_key property, 180
    trust_by_default property, 184
    trusted_docbases property, 185
    underlying RDBMS table descriptions, 643
reserved words
    as property names, 27
    in queries, 27
    names and DQL, 27
    RDBMS, 28
_resume_state (computed property), 21
retainer object type, 359
retention policies
    dm_retainer type, 359
retention—enabled storage
    dm_ca_store type, 95
richtext content
    dmc_notepage type, 534
richtext object type, 536
room object type, 537
route cases
    dmc_composite_predicate type, 130
    dmc_transition_condition type, 412
routecase condition object type, 364
routines, component, 325
rps authority object type, 570
rps base date object type, 571
rps child strategy object type, 572
rps condition object type, 574
rps contact object type, 575
rps disposition method object type, 576
rps event object type, 578
rps hold object type, 581
rps phase rel object type, 584
rps retainer event rel object type, 589
rps retainer object type, 586
rps retention policy object type, 590

S

Save method
    effect on relationships, 37
Saveasnew method, 37
scope config object type, 368
script object type, 369
scs_admin_config type, 594
security
    ACL object type, 40
    client registration object type, 117
    client rights object type, 118
    directory in mount point object, 288
    directory or file in Location type object, 262
    folder, 180
    level for repository, 184
    properties controlling, 404
    public key certificate object type, 323
    relationships, 32
sequence object type, 370
server config object type, 371
server config objects
    names, restrictions, 376
server locator object type, 492
session config object type, 494
session identifiers, 29
session object type, 381
sessions
    client config type and, 486
    connection config objects and, 487
    identifiers, 29
_sign_data (computed property), 22
Smart List object type, 382
state extension object type, 384
state type object type, 385
_state_extension_obj (computed property), 24
_state_type (computed property), 24
states
    computed properties describing, 23
_status (computed property), 22
storage areas, 204 to 205, 386
    See also content assignment policies
    blob, 90
    distributed store object type, 172
    external URL object type, 207
    file store object type, 211
    linked store object type, 253
    store object type, 387
store object type, 387
string datatypes
    unit of length, 26

subcontent object type, 393
SysObject object type, 395
SysObjects
_current_state (computed property), 19
event-related property, 406
folder properties, 399
general properties, 395
lifecycle-related properties, 407
security-related properties, 404
version-related properties, 405
virtual document properties, 400
sysprocess config object type, 408
system security for relationships, 32

T

tables
naming rules, 28
tablespaces
index location, 231
type index storage, 180
taxonomy object type, 409
TCF Activity object type, 410
TCF activity template type, 411
timeouts
automatic activities, 51
procedure, 279
topic object type, 540
tracing
error level in _status property, 22
transactions
_isdeadlocked, 20
transformation profiles
media profile object type, 264
transition condition object type, 412
translation relationships
language and country codes, 639
trust_by_default property, 184
trusted_docbases property, 185
turbo storage
renditions in, 380
subcontent object type, 393
type identifiers, 29
type indexes
index object type, 231
type info object type, 419
type object type, 416
_type_id (computed property), 22
_type_name (computed property), 22
types
naming rules, 28
RDBMS table descriptions of, 643
_types (computed property), 22
_typestring (computed property), 23

U

underbar (_)
in table and column names, 28
underscore (_), in type names, 28
user authentication
UNIX users against domains, 89
user object type, 422
user privileges
user object, 427
user-defined types
identifiers, 29
naming rules, 28
users
aliases, 66
naming rules, 28
user_name, case sensitivity, 28
workqueue user profile object type, 466

V

validation descriptor object type, 430
validation module object type, 431
validation relation object type, 433
value assist object type, 435
value func object type, 437
value list object type, 438
value query object type, 439
_values (computed property), 23
versions
SysObject properties, 405
views
Documentum, 644
RDBMS, 643
virtual documents
assembly objects and, 74
SysObject properties, 400
vstamp object type, 441

W

webc config object type, 596
webc target object type, 602
wf attachment object type, 442

wf package report parent object type, 443
wf package schema object type, 445
WF package skill object type, 446
WF package type info type, 448
wf timer object type, 449
wfsd element booleanobject type, 502
wfsd element date object type, 503
wfsd element double object type, 504
wfsd element integer object type, 505
wfsd element object type, 501
wfsd element parent object type, 506
wfsd element string object type, 507
wfsd type info object type, 508
wfsdrp boolean object type, 513
wfsdrp date object type, 514
wfsdrp double object type, 515
wfsdrp integer object type, 516
wfsdrp string object type, 519
work item object type, 456
work items
    completed workitem object type, 124
    workqueue object type, 461
work queues
    workqueue doc profile object type, 463
    workqueue object type, 461
    workqueue policy object type, 464
    workqueue user profile object type, 466
workflow definitions
    wf timer object type, 449
workflow object type, 451
workflow object types
    wfsd element, 501
    wfsd element boolean, 502
    wfsd element date, 503
    wfsd element double, 504
    wfsd element integer, 505
    wfsd element string, 507
    wfsd type info, 508
    wq skill info, 467
    wq task skill, 470
workflows
    completed workflows object type, 120
    completed workitem object type, 124
    module config object type, 285
    process object type, 313
    process parameter object type, 320
    wf attachment object type, 442
    wf package report parent object type, 443
    wf package schema object type, 445
    work item object type, 456
    workqueue object type, 461
workqueue category object type, 462
workqueue doc profile object type, 463
workqueue object type, 461
workqueue policy object type, 464
workqueue user profile object type, 466
wq skill info object type, 467
wq task skill object type, 470
wq user skill object type, 471

X

xfm adaptor config object type, 608
xfm form object type, 610
xfm instance object type, 612
XML (Extensible Markup Language)
    xml application object type, 472
    xml config object type, 474
    xml custom code object type, 475
    xml style sheet object type, 476
    xml zone object type, 477
xml application object type, 472
xml config object type, 474
xml custom code object type, 475
xml style sheet object type, 476
xml zone object type, 477
_xpermit (computed property), 23
_xpermit_list (computed property), 23
_xpermit_names (computed property), 23