Relationships in Documentum

Objects in the Documentum repository can be associated with Relationships. Lets explore the basics of Object Relationships here.

There are 2 major object types that you need to know in

1) dm_relation_type
This defines a relation, means this object type holds the information about the relation like its name, security type, parent type, child type etc. We will get into the details of this soon.
User must have SysAdmin or Super User privileges to create object of this type. All the objects of this type will have r_object_id starting with **38**

2) dm_relation
This defines each individual relationship between the objects. Means this type has ID's of the both parent and child object which makes the relation, the name of the relation etc. All the objects of this type will have r_object_id starting with **37**

Now lest see the attributes of each object type and understand what each attributes is meant for and how these objects are related.

Lets list the attributes of type **dm_relation_type**

Name	Info	Description
relation_name	Char (32) Single	Name of the relation, This must be a Unique name.
security_type	Char (10) Single	This defines who can add, delete, drop or modify a relationship. The possible values are SYSTEM PARENT CHILD NONE (I will explain this in detail below)
parent_type	Char (32) Single	The valid object type of parent in this relationship
child_type	Char (32) Single	The valid object type of child in this relationship
description	Char (250) Single	Description of the relationship (User defined)
direction_kind	Integer Single	The Direction of the relation There are 3 Possible values for it. And they are 0 – This relation is parent to child 1 – This relation is child to parent 2 – This relation is bidirectional – means these objects are siblings The default value is always 0
integrity_kind	Integer Single	This determines the referential

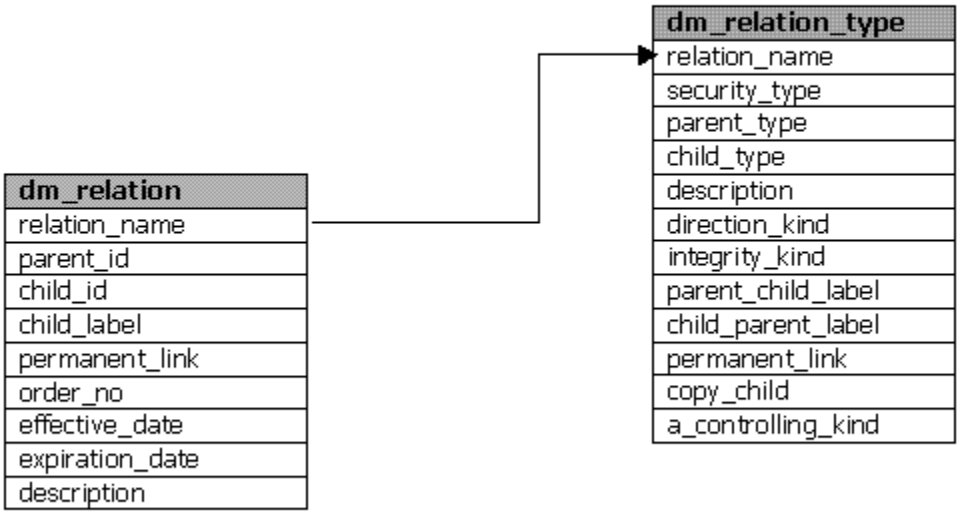
		integrity of this relation. There are 3 Possible values for it. And they are 0 – Allow Delete 1 – Disallow delete 2 – Cascade Delete (This means when a user deletes an object participating in a relationship instance, Server will also destroy the participating partner in the relationship) The default value is always 0
parent_child_label	Char (255) Repeating	User defined name for the parent child relationship
child_parent_label	Char (255) Repeating	User defined name for the child to parent relationship
permanent_link	Boolean Single	This defines what happens to the relation when the parent is copied or versioned. If set to True relation is maintained with new parent object. If set false the relation is not maintained. The Default value is False
copy_child	Integer Single	Specifies whether to copy the child in a relationship when the parent is copied and permanent_link is True. . There are 2 Possible values for it. And they are 0 – Do not copy Child 1 – Copy Child
a_controlling_kind	Char (32) Single	This indicates whether relationships of this type are created by users or internally.

Lets list the attributes of type **dm_relation**

Name	Info	Description
relation_name	Char (32) Single	Name of the relation that's specified for the dm_relation_type object
parent_id	ID Single	ID of the parent object
child_id	ID Single	ID of the child object.
child_label	Char (32) Single	Optional version label of the child object. If this value is provided then I_chronicle_id of the object should be given as the id of the child in the child_id attribute.
permanent_link	Boolean Single	<Deprecated>
order_no	Integer Single	Not used, but can be used for custom purpose
effective_date	Date Single	Not used, but can be used for

		custom purpose
expiration_date	Date Single	Not used, but can be used for custom purpose
description	Char (255) Single	Description of the relationship (User defined)

The relation between dm_relation_type and dm_relation type objects is illustrated below



Few bullets points about dm_relation and dm_relation_type

Niether dm_relation nor dm_relation_Tupe can be versioned
Object level security (ACL) cannot be set on these object types
You cannot delete a dm_relation_type object if any of the dm_relation with its relation_name exists.
You need super user privileges to delete a dm_relation_type object
Destroying a dm_relation object does not destroy the parent or child objects.
When you destroy an object all the dm_relation associated with that object also gets deleted.

Security and Relations

When we saw the attributes of dm_relation_type object I had mentioned about security_type. Lets see this in detail. As mentioned above security_type defines who can add, delete, drop or modify an object of dm_relation object, which has a relation name that of the relation_name in the dm_relation_type. Lets see what the individual possible values means here

1) **SYSTEM**
   This means only SuperUsers and SysAdmins can create, edit or delete a relation with the relation_name in this type. This does not prevent an owner of an object from destroying an object that participates in the relationship. When a Objectis destroyed content server also destroys all relation objects associated with the

object. The owner of that Object is not required to be a superuser or system administrator.

2) **PARENT**
   This means create, delete or edit a Relation will depend upon the users permissions on the parent object. I.e. if the object that's getting related is of object type dm_sysobject or its child user should at least have RELATE permission.
3) **CHILD**
   This means create, delete or edit a Relation will depend upon the users permissions on the Child object. I.e. if the object that's getting related is of object type dm_sysobject or its child user should at least have RELATE permission.
4) **NONE**
   This means Any user can create, delete or edit a Relation of this type.