Data Dictionary in Documentum

As you all know Documentum is all about Objects and its attributes, Have you ever wondered where Documentum is storing information about its Object types and its attributes?
To get the answer for this question you should know more about Data Dictionary in Documentum. Let me walk you through some of the important aspects of Data Dictionary in this study note.

What is the use of Data Dictionary

The following is a partial collection of the information about an Object type that will be stored by Data Dictionary *Attribute label, help and other information (Localized), Default Attribute values, Value Assistance, Value Mapping, Constrains, and Default lifecycle of that object type*. A Documentum client application can leverage this information to build the Presentation layer for that Object type and also provide some business rule enforcements.

Another interesting point about Data Dictionary is that it supports multiple locales, which means you can have multiple locale configured for each object type. Each locale represents a geographical region.
Imagine an Organization that has offices in Paris, Spain and US , The Data Dictionary allows you to store each Attribute label information in all three languages and the Client application (e.g. : WebTop , or a Custom UI) can fetch for the specific labels in the language of that region and display it to the user.

More about Data Dictionary

Let's see some of the useful features of Data Dictionary here, above I mentioned some of the information that you can store in the Data Dictionary, lets see some of these in detail for a better understanding. First we will see all UI related and then we will see some business rules and functionalities that you can enforce through data dictionary

UI Related

1) Default Values for Attributes

During the Creation of an Object type or upon its modification you can specify the default value of that Attribute, which means if no value for that attribute has been specified by the user this default value will be set as the attribute value

2) Value Assistance

Value assistance is used to provide user with a drop down list of possible values for that attribute. This can be even conditional, means upon selected criteria the values in the value assistance can be changed (Conditional Value Assistance)
Another important point about Value assistance is the values used for Value Assistance can be a fixed list or based out of a DQL query that runs dynamically.

3) Value Mapping

Value mapping is another useful feature where a value can be mapped for another, this works as a Key Value pair, for an example consider this list used for Value mapping New Jersey - NJ, New York - NY, and New Hampshire - NH.

This option provides a possibility of user being displayed with the Complete State name in the UI and Value stored will be just the state code.

4) Internationalization of Various Texts

If you look at the WebTop or Any Documentum UI application (Not necessarily a custom build User Interface) you can see there are lot of information like labels, error messages, help information etc, these text bits can be stored for different locales in the Data Dictionary. Different locale means different languages. So this helps to build a single UI for a global application and support multiple languages.

Business Rule and Functionality

1) Lifecycles for an object type.

During the Creation or modification of an object type you can specify a lifecycle as the default lifecycle of an object type. This eliminates the pain of searching for a lifecycle name or its ID to attach it to the newly created object; User can do it by using keyword **default** at the time of attaching a new Object instance to a life cycle.
But the *important point to note here is just by specifying a default lifecycle a object will not be attached to its default lifecycle*. The Creator or the application has to specifically attach that object to the lifecycle.

2) Constrains

You can do validation of a property by adding constrains to it. The possible types of constrains are the following. *Important point to note here is Content server does not enforce these constrains even though you define in Data Dictionary*. Typically the Client application should read these constrains and enforces it. You can also specify the localized error messages in for the validation error in the Data Dictionary.

a) Primary key

Primary key should be added in combination with not- null constrain. Primary keys are inherited. One or more attributes can make primary key but only single value properties can be a part of it. One object type can have only one primary key definition, (But can have more if it inherit primary key from its super type). Primary key constraints can be either the object type level or the property level. If the key has more than one participating properties it should be defined at type level. If the key is a single property then it's a good idea to define it at property level.

b) Unique Key

Unique key constraint is used to enforce a property or combination of properties for which all the object of that type should have unique value. The key can be a combination of one or more single-valued properties or one or more repeating properties which is defined in that object type itself (Not Inherited).Another important point is the key for Unique constraint cannot be a combination of single-valued and repeating properties. These are inherited too.

c) Foreign key

Foreign key constraint identifies relationship between one or more properties for one object type and one or more properties in another. The number and data types of the properties in each set of properties must match. Foreign key constraints can be at object type level or at the property level. It should be defined at type level If the key has two or more participating properties. Also both object types must be set in the same repository, and corresponding parent and child Properties should be of same data type.

d) Not Null
A NOT NULL constraint sets on a property that will not allow having a null value. It can be defined only at the property level and only for single properties

e) Check
Check constraints are used for validating data. An expression or script can be provided in the constraint's definition that the client application can run to validate a given property's value. This can be on Object level or Attribute level

How is Data Dictionary modified

Data Dictionary modification can be either adding a new Object type information or can be modifying existing Object type information.
For adding a new Object type and its any of the above mentioned details can be done by either calling CREATE TYPE DQL Script, or by creating a new type in a new or Existing DAR and deploying it.
Modifying the Existing Object type information can be done by editing type information in the doc app or DAR or by calling Alter type DQL script.
Please note that DAR or Documentum Archive is applicable only for repositories those are running on Documentum 6 or higher

Data Dictionary Publishing Job

When you update the Data Dictionary it in essence updates the internal object types and you need to run Data Dictionary Publishing job. This job is responsible for creating the necessary dmi_dd_attr_info , dmi_dd_type_info and dmi_dd_common_info objects. You can configure and run this job from Documentum Administrator.

What makes a Data Dictionary

I had mentioned three object types that are getting created when the Data Dictionary publishing job publishes the Data Dictionary information. Lets see those objects briefly here

dmi_dd_common_info
This object type contains information about an object type or an attribute that are common. All the objects of this type will have r_object_id starting with **68**.

dmi_dd_type_info
This is a sub type of ***dmi_dd_common_info***. This object type contains information about an object type. (which has already been published to the data dictionary) All the objects of this type will have r_object_id starting with **69**.

dmi_dd_attr_info
This is a sub type of ***dmi_dd_common_info***. This object type has information about a property (which is already published to the data dictionary) All the objects of this type will have r_object_id starting with **6a**.

You would have noticed that all these object type starts with keyword ***dmi*** this means we cannot create or modify this object type, Only Data Dictionary publishing job can modify or create this type.