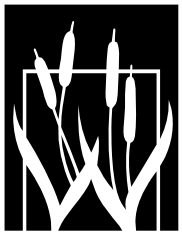

Whitemarsh
Information Systems Corporation

Reference Data Management

Whitemarsh Information Systems Corporation
2008 Althea Lane
Bowie, Maryland 20716
Tele: 301-249-1142
Email: Whitemarsh@wiscorp.com
Web: www.wiscorp.com

Table of Contents

1.0 Objective

The objective of this paper is to present the Whitemarsh approach to managing a very critical set of enterprise data: Reference Data. This short paper describes the rationale for reference data and its management, sets out the six most common reference data cases, a generalized data model for managing reference data, descriptions of each of the six reference data cases, a strategy for reference data updating and maintenance, and finally, the recasting of existing reference data.

2.0 Topics Covered

The topics is this paper include:

- Rationale for Reference Data
- Description of the Six Reference Data Cases
- Generalized Data Model
- Specifications of Six Reference Data Cases
- Strategy for Reference Data Updating and Maintenance
- Recasting Reference Data

3.0 Introduction

Reference data represents data-based object collection classification schemes. Each reference data value should be orthogonal in terms of its semantics. Reference data values are employed to uniquely distinguish one object collection from another.

Collections of objects can be characterized in multiple ways. Thus, there can be multiple reference-data based characterizations. Simple reference data consists of discrete values representing single, easily distinguished meaning. Complex reference data contains multiple inter-related facts that are intrinsically related to each other, and in many situations have their values changed in lock step.

An instance of reference data is almost always a row in a database table. As with all other rows, there are four kinds of columns for each table: uniqueness columns, metadata columns, content columns, and relationship columns. Uniqueness columns represent a set of values that causes the selection of just one row. Metadata columns represent the reference data's adjudication information. That is, the information that affirms the authoritativeness of the values. This normally contains who, what, when, where, why, and how context data. Content columns represent the "real" data, and for a single reference data content column that would be just the reference data value and the reference data value meaning. The final column set, relationship, represents the columns that, when valued, provide the uniqueness value for a different row in the same table (recursive relationship) or different table (child table for a parent-child relationship).

In this later situation, the uniqueness column set is often called the reference data table's primary key.

Any database column that has a severely restricted value domain can be seen as reference data. Additionally, whole collections of data, that is, multiple columns within a reference data table, can also be seen as reference data. Reference data is often employed as sort fields, selection fields, and as "control breaks" for the purpose of statistical calculations. Sometimes reference data exists as single database columns, and other times reference data contains multiple columns that act as collections, in concert. On occasion, the reference data column values must "progress" from one value to the next in a strict sequence.

4.0 Reference Data Cases

The six common cases for managing reference data are presented in Table 1. Specific examples for each of these cases are provided in the tables (Tables 4 through 9) and are described in the six subsections of this section. Table 2 of this section names and describes the database tables for the data model for reference data. Included in these subsections are descriptions of the data that needs to exist in the various data model tables from the generalized data model for reference data. Table 3 of this section describes the data that needs to be stored in the reference data tables to accomplish proper value domain governance.

Reference Data Case	Reference Data Type	Description
1	Single Content Column	A single column of reference data such as a person Gender (e.g., female and male), Gender. See current practice above.
2	Multiple Content Column	Multiple columns that should be treated together.
3	Sequenced	Single or multiple columns of reference data that proceed in a strict sequence. For example, Proposed, Draft, Final, and Revised.
4	Mapped Values	Single or multiple columns of reference data that have changed values and/or meanings across time, or that represent a "transformed in" value from a set of inward interface processes or "transformed out" value for a set out outward interface processes. Mapped values may also represent a reduction in the quantity of "mapped to" values from a set of "mapped from" values. The converse is also possible.
5	Discrete Values	Specific enumerated values that can be additionally allowed or disallowed. There can be multiple sets of discrete values, some allowed and other disallowed.

Reference Data Case	Reference Data Type	Description
6	Range Of Values	Specific ranges of values that can be additionally allowed or disallowed. There can be multiple sets of value ranges, some allowed and other disallowed.

Table 1. Reference Data Cases

These six reference data cases can be incorporated within the data model design set out in Figure 1. The database tables within the data model are set out in the Table 2.

Data Model for Reference Data	
Code Fact Table Code Assignment	A Code Fact Table Code Assignment stores the assignment of a specific code type (e.g. Gender) value for one or more columns in various database content table columns. This table also contains both effective start and end dates.
Code Type	A Code Type identifies the specific code type, such as Gender. Multiple content columns are addressed in the Code Type Structure and Code Type Structure Type tables. Code Type values are addressed in the Code Value, Code Value Structure and Code Value Structure Type tables.
Code Type Structure	The Code Type Structure table supports the interrelationship of different Code Types that are required for multiple content columns that contain values that must be modified in lock-step.
Code Type Structure Type	The Code Type Structure Type table's data contains the specification necessary to fully understand the Code Type Structure records that a Code Type Structure Type record governs. This sets out collections of Code Type columns that need to be addressed in combinations.
Code Value	The Code Value table contains the specific values associated with a given Code Type. Each Code Value record contains the code, values, meaning, description, and effective start and end dates. The relationships that exist between and among code values are managed by the Code Value Structure and Code Value Structure Type tables.
Code Value Structure	The Code Value Structure table contains the specific interrelationships among code values. This table enables two classes of interrelationships. Sequencing, and value mappings. Sequencing is needed for value sets that progress from one value to the next. Value mappings are needed support the changes in values for a given meaning across time. Value mappings support equivalent values, merged values from previous multiple values and diverged values from a common previous value.
Code Value Structure Type	The Code Value Structure Type table's data contains the specification necessary to fully understand the Code Value Structure records that a Code Type Structure Type record governs. This would set out collections of Code Values that need to be addressed in specific collections.

Data Model for Reference Data	
External Standard	An External Standard is the specification that governs identified collections of Code Types and their corresponding values. As the external standard changes over time, the values may have to migrate from one value set to the next. Value migration is managed by the appropriate Code Value tables.
External Standard Organization	An External Standard Organization is the value domain standards creation organization. This might be an organization managed by ISO, ANSI, an industry association, or a formal working group within an enterprise.
Fact Table	A fact table is a surrogate name for a content database table that contains one or more columns that have restricted value domains that require reference-based value domain management.
Policy Basis	The Policy Basis table contains the necessary information to understand the policy basis for a collection of Code Values from one or more Code Types. Essentially the Policy Basis data represents the study organization that performs the analysis and formulation of the initial, subsequent, and evolution of the various Code Type value sets.
Policy Basis Member	The Policy Basis Member table contains the names of the persons who participated in the development of the policy basis for the value sets appropriate for the various Code Types.
Policy Basis Study Membership	Policy Basis Study Membership represents the interrelationship between Policy Basis members and the Policy Basis studies that determine the Code Type value domains and subsequent evolutions of these value domains.

Table 2. Database Tables for the Reference Data Model

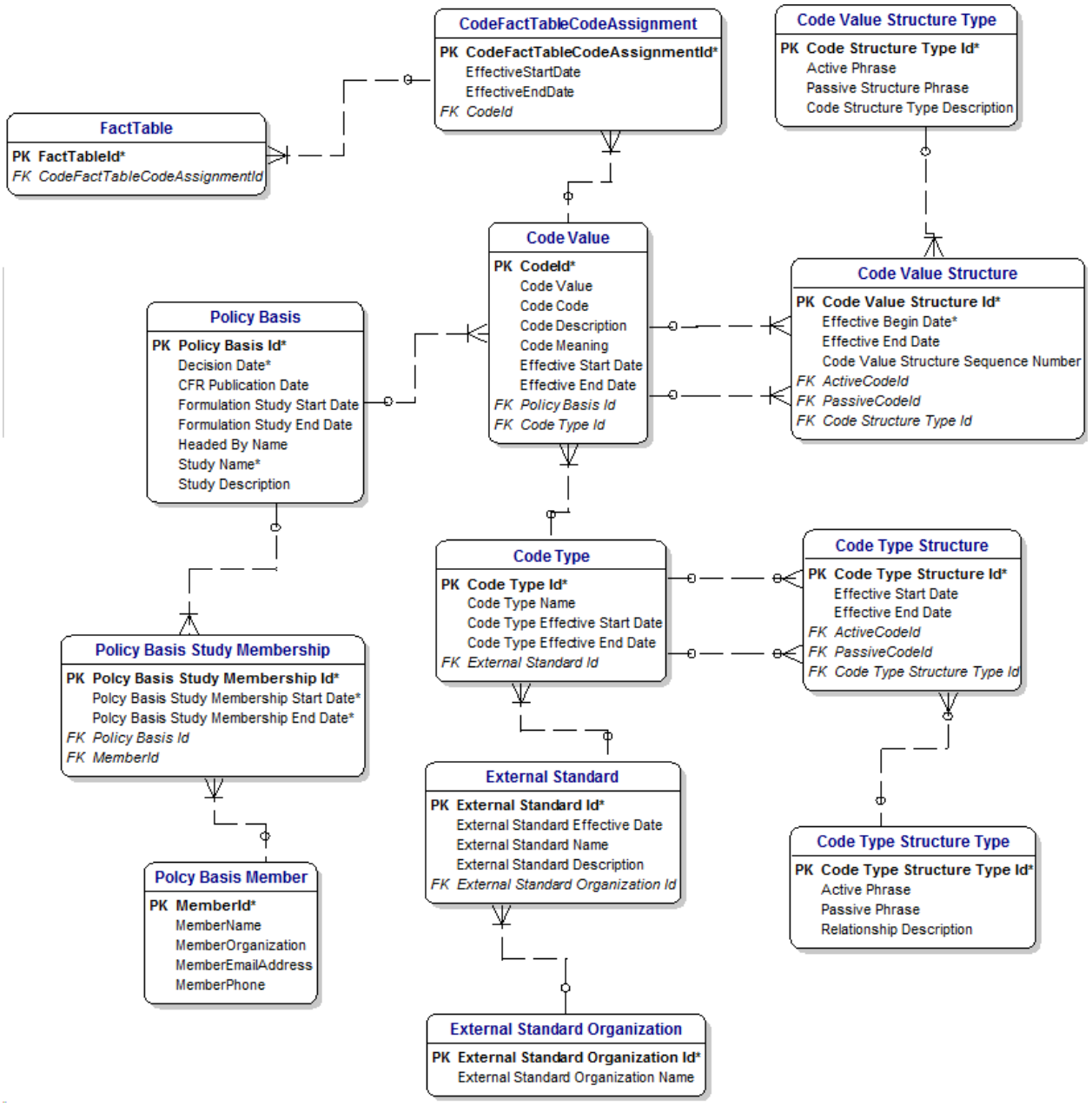

Figure 1. Generalized Data Model for Reference Data.

For all of the six cases the following tables need to be populated with an appropriate set of data that supports specific Code Types and Code Values. For the sequenced and mapped code values, this includes the policy basis for interrelationship among code values. For Code Types, this includes the relevant information within Code Type structures needed to support related Code Types. These tables are set out in Table 3.

Reference Data Table Involvement	
External Standard	The standard from the enterprise or an external organization that is to govern the existence of the code types and the code values associated with the code types.
External Standard Organization	The name and description of the organization charged with the creation of the reference data standard. This could also be the working group within the organization that has created the code types and associated code values.
Policy Basis	The description of the policy basis that governs the identification of the complete set of code values and by inference the code types that are to be managed.
Policy Basis Member	The names of the members that belong to the value domain policy group charged with identification and specification of the code types and code values.
Policy Basis Study Membership	The association of the given policy basis members with the specific policy basis that governs the Code Values and by inference Code Types.

Table 3. Database Tables for the Reference Data Model

4.1 Single Content Columns

An example of Case 1, Single Content Column reference data, is Gender. The involved Reference Data tables are identified and described in Table 4.

Reference Data Tables for Single Content Columns	
Code Fact Table Code Assignment	The row of data from within the reference data table that contains the primary key value that, in turn becomes the foreign key value of the reference data column from within the appropriate fact table.
Code Type	The row of data that contains the name of the code table column about which reference data values are created. This row contains the effective start and end date for the Code Type. This is to ensure that the code value that exists is from a current Code Type column.
Code Type Structure	This is not necessary for this reference case.
Code Type Structure Type	This is not necessary for this reference case.

Reference Data Tables for Single Content Columns	
Code Value	The row of data that contains the specific value, code, meaning and description that is represented by the appropriate column in the fact table. The effective start and end dates must be appropriately set so that the represented fact table value represents a current value.
Code Value Structure	This is not necessary for this reference case.
Code Value Structure Type	This is not necessary for this reference case.
Fact Table	The column from within the appropriate fact table that contains the reference data column. The value from this column acts as a foreign key value reference to the Code Fact Table Code Assignment row that, in turn, represents the reference data Code Value.

Table 4. Database Tables for Single Content Columns

4.2 Multiple Content Columns

An example of Case 2, Multiple Content Column reference data, is postal addresses. In each address there are valid triples: City, State, and Postal Code. It is often incorrect to only update one of these columns without also updating its adjacent column's value. The involved Reference Data tables are identified and described in Table 5.

Reference Data Tables for Multiple Content Columns	
Code Fact Table Code Assignment	The row of data from within the reference data table that contains the primary key value that, in turn, becomes the foreign key value of the reference data column from within the appropriate fact table.
Code Type	The row of data that contains the name of the code table column about which reference data values are created. This row contains the effective start and end date for the Code Type. This is to ensure that the code value that exists is from a current Code Type column.
Code Type Structure	This table contains multiple rows of data, one each for the different Code Types that must be associated with each other as a multiple content columns. The relationship that exists between the multiple columns essentially is: "Also involves" as the active relationship and "Is involved with" as the passive relationship.
Code Type Structure Type	This table contains the multiple content column subset collection data that supports the identification of collections of multiple content columns. In this example, the individual active and passive descriptive names are identified and defined. Defined also is the overall meaning of this Code Type Structure relationship type.
Code Value	The row of data that contains the specific value, code, meaning and description that is represented by the appropriate column in the fact table. The effective start and end dates

Reference Data Tables for Multiple Content Columns	
	must be appropriately set so that the represented fact table value represents a current value.
Code Value Structure	This is not necessary for this reference case.
Code Value Structure Type	This is not necessary for this reference case.
Fact Table	The column from within the appropriate fact table that contains the reference data column. The value from this column acts as a foreign key value reference to the Code Fact Table Code Assignment row that, in turn, represents the reference data Code Value.

Table 5. Database Tables for Multiple Content Columns

4.3 Sequenced Values

An example of Case 3, Sequenced Values reference data are the various values for a document's progression state. For example, Proposed, Draft, Final, and Revised. The involved Reference Data tables are identified and described in Table 6.

Reference Data Tables for Sequenced Values	
Code Fact Table Code Assignment	The row of data from within the reference data table that contains the primary key value that, in turn, becomes the foreign key value of the reference data column from within the appropriate fact table.
Code Type	The row of data that contains the name of the code table column about which reference data values are created. This row contains the effective start and end date for the Code Type. This is to ensure that the code value that exists is from a current Code Type column.
Code Type Structure	This is not necessary for this reference case.
Code Type Structure Type	This is not necessary for this reference case.
Code Value	The row of data that contains the specific value, code, meaning and description that is represented by the appropriate column in the fact table. The effective start and end dates must be appropriately set so that the represented fact table value represents a current value.
Code Value Structure	Rows of data from this table, ordered according to a specific sequence of values one to the other.
Code Value Structure Type	This table contains the sequenced values subset collection data that supports the identification of collections of sequences of values. In this example, the individual active

Reference Data Tables for Sequenced Values	
	and passive descriptive names are identified and defined. Defined also is the overall meaning of this Code Value Structure relationship type.
Fact Table	The column from within the appropriate fact table that contains the reference data column. The value from this column acts as a foreign key value reference to the Code Fact Table Code Assignment row that, in turn, represents the reference data Code Value.

Table 6. Database Tables for Sequenced Values

A variation on either the single content column or the multiple content column reference data type is sequenced reference data. In the case, the reference data values must either proceed in a forward or backward strict sequence, or can skip values forward or backward. Regardless, the general approach that is most commonly applicable is that any given value must be preceded by a previous value. For example, the values, Proposed, Draft, Final, and Revised. are essentially indicators of state-based on well-defined assessments of a document's progression.

In the example, the rules may be that the Draft state cannot exist except as a successor to a Proposed state. Or, that the Revised state cannot be set without the Final state having been achieved. Some states may exist in parallel and are only dependent on the prior state.

Representation of reference data sequence rules requires the employment of the Code Value Structure and Code Value Structure Type reference data tables. When these tables are properly valued, an application program can query an existing state and know what value is the allowed successor state. If an inappropriate value change is attempted, the data from these reference data tables can be used to support a rejection. When these states are represented as data within the Code Value Structure data, they can be changed without 1) finding all the different application programs in which they have been encoded, 2) propose and accomplish software changes, 3) perform testing, and 4) modify user-guide documentation and the like.

With this strategy, single or multiple column reference data that is to be sequenced can be represented. The value of managing this sequence as part of the reference data is that it prevents any required encoding of a sequence in the application programs. Additionally, rules can be established and placed in the database that provides triggering information to the application program whenever any sequence "transgression" occurs.

4.4 Mapped Values

The ability to map reference data values one to another is needed for a number of different reasons. The five most common cases are:

- Single or multiple content column value changes
- Single value equivalences
- Reduced quantity of reference data values

- Expanded quantity of reference data values
- External interfaces to other application systems and databases

An example of single value equivalences is again, "Gender." For example, 0 = Female = F = Mujer. The involved Reference Data tables are identified and described in Table 7.

Reference Data Tables for Mapped Values	
Code Fact Table Code Assignment	The row of data from within the reference data table that contains the primary key value that, in turn, becomes the foreign key value of the reference data column from within the appropriate fact table.
Code Type	The row of data that contains the name of the code table column about which reference data values are created. This row contains the effective start and end date for the Code Type. This is to ensure that the code value that exists is from a current Code Type column.
Code Type Structure	This is not necessary for this reference case.
Code Type Structure Type	This is not necessary for this reference case.
Code Value	The row of data that contains the specific value, code, meaning and description that is represented by the appropriate column in the fact table. The effective start and end dates must be appropriately set so that the each of the acceptably represented fact table value represents a current value.
Code Value Structure	Rows of data from this table represent the interrelationships across all the code values that are valid. The different cases that have to be mapped include: equivalent values, and former and new values. In this last case, there are two subcases, a reduction in the quantity of former values that is replaced by a single new value, and an expansion in the quantity of a single former value that is replaced by a set of multiple new values. The "from" value is identified through a Code Value retrieval based on the ActiveId value. The "to" value is identified through a Code Value retrieval based on the PassiveId value.
Code Value Structure Type	This table contains the mapped values subset collection data that supports the identification of collections of mapped of values. In this example, the individual active and passive descriptive names are identified and defined. Defined also is the overall meaning of this Code Value Structure relationship type. For this specific example, there would be different sets of Code Value Structure Type data for each of the five different cases. When a row of data from the Code Value Structure data is retrieved, the Code Value Structure Type value enables the retrieval of the "type" information and also for the retrieval of the other Code Value Structure rows that represent the "from" or "to" value for that particular case.
Fact Table	The column from within the appropriate fact table that contains the reference data column. The value from this column acts as a foreign key value reference to the Code Fact Table Code Assignment row that, in turn, represents the reference data Code Value.

Reference Data Tables for Mapped Values

Table 7. Database Tables for Mapped Values

In the first case, single or multiple content column value changes, whenever a single or multiple content column value changes, both the previous and current values may need to be interrelated within the Code Value Structure table in order to represent history. Additionally, if the values are "changed in place" within the fact tables themselves, databases that contain millions of records and significant quantities of reference data columns would require large quantities of computer resources to effect the data value change. Not only would this be a waste of computer resources, all the backups of the database would have to be brought back on-line and have their reference data values changed as well.

Through the strategy of a mapped Code Value Structure table, an existing value would be retrieved, the effective end date would be discovered to be no longer valid, and the now current value would be retrieved.

For an actual example, if the value (e.g. Gender = "F") is stored in the database and the value needs to be changed to "Female," then, when the change is needed, all the records with "F" have to be retrieved and the gender column value changed to "Female." This has two problems. First, the previous value of "F" is now lost if there hasn't been an audit trail record created for each of the changed records, and second, there can be a significant quantity of computing resources expended for this change. If there were a million records and half were "F" then there would have to be 500,000 records changed and also 500,000 audit trail records created.

The only practical alternative to this approach is to create the Code Value Structure records that interrelate the old value, "F," to the now current value, "Female," that has as its starting date and time the very same ending date and time for the "F" reference data record. The Code Value Structure record is the method that interrelates the "F" reference data record with the "Female" Code Value record. With this approach, none of the half-million records have to be changed. When the actual gender is needed for a person, the reference data value is retrieved. Because there is an "end date" value, the PassiveId value is employed to retrieve the current value for that reference data field. This technique works when there is the same quantity of valid values after a change as there was before.

In the second case, reduced quantity of reference data values, several values may be replaced by a single value. Suppose there were the values: Proposed, Draft, Final, and Revised. Suppose the new set was to combine the values Proposed and Draft into just one value, Initial. In this case the two discrete values would have to point to the same new value. This approach is straight forward if the mapping is always "forward." But if it's backwards, it would not be known what the prior value for "Initial" had been. That is, either "Proposed" or "Draft."This too can be addressed by "walking back" from the Code Value row to the Code Fact Table Code Assignment row.

In the third case, expanded quantity of reference data values, suppose the current values are: Proposed, Draft, Final, and Revised. Suppose the new values are to be: Proposed, Draft, Preliminary Acceptance, Accepted, Final, and Revised. In this case, business rules need to be

created or changed that enable a re-casting of the reference data values. In this situation, not only must these values be added to the Code Value table, the interrelationships among these values and also the proper sequence must be added to the Code Value Sequence table. There also must be the identification of the business rules that would provide the unambiguous knowledge to know the conditions supporting the proper discernment of these new values. These business rules must be properly added and/or modified within existing business information systems. If the business information systems had been properly designed these changes should be minor.

In this third case, where there are interfaces to external systems either for data importing or exporting, there has to be Code Value mappings from the various reference data based external system data fields to the current set of reference data. These fields have to be discovered and their reference data values and meanings uncovered and stored in the reference data tables. This effort is directly supported by the reference data tables in Figure 1.

In the fourth case, external interfaces to other application systems and databases, on importing, an external data record is read, its fields and values are known, and known as well are the transformed-to values. If all these values are stored in the Code Value tables, the data importing programs do not have to contain all these value transformation rules.

On exporting, this same general strategy applies for exporting data except in reverse. In either scenario, it is best to have the reference data tables contain all the value mappings and any necessary supporting rules for value mappings so that they are explicitly visible, are data not program based, and can be more easily changed than from within programs.

4.5 Discrete Values

An example of Case 5, Discrete Values, Zip Code, again serves as a good example. That's because the Code Values are able to be well defined and are discrete. The involved Reference Data tables are identified and described in Table 8.

Discrete values reference data simply means that the values are enumerated. For example, the values 20716, 20717, 20718, 20720, and 20721. A common capability for representing discrete values is the ability to indicate that certain discrete values are not allowed. So, if all between 20716 and 20718 but not 20719, then the list would either contain mechanisms to express both.

Discrete value management can exist on either single content or multiple columns, and also on sequenced value and mapped value columns.

Reference Data Tables for Discrete Values	
Code Fact Table Code Assignment	The row of data from within the reference data table that contains the primary key value that, in turn, becomes the foreign key value of the reference data column from within the appropriate fact table.
Code Type	The row of data that contains the name of the code table column about which reference

Reference Data Tables for Discrete Values	
	data values are created. This row contains the effective start and end date for the Code Type. This is to ensure that the code value that exists is from a current Code Type column.
Code Type Structure	If there is multiple code table columns involved, this table contains multiple rows of data, one each for the different Code Types that must be associated with each other as a multiple content columns. The relationship that exists between the multiple columns essentially is: “Also involves” as the active relationship and “Is involved with” as the passive relationship.
Code Type Structure Type	If there are multiple code table columns involved, this table contains the multiple content column subset collection data that supports the identification of collections of multiple content columns. In this example, the individual active and passive descriptive names are identified and defined. Defined also is the overall meaning of this Code Type Structure relationship type.
Code Value	The row of data that contains the specific value, code, meaning and description that is represented by the appropriate column in the fact table. The effective start and end dates must be appropriately set so that the represented fact table value represents a current value. **Note:** the data model design shown in Figure 1 does not properly reflect the ability to express NOT values for discrete values. It also does not express the ability to have a mixture of discrete and NOT sets of discrete values. Changes in the data model are necessary to accomplish this capability.
Code Value Structure	This is not necessary for this reference case.
Code Value Structure Type	This is not necessary for this reference case.
Fact Table	The column from within the appropriate fact table that contains the reference data column. The value from this column acts as a foreign key value reference to the Code Fact Table Code Assignment row that, in turn, represents the reference data Code Value.

Table 8. Database Tables for Discrete Values

4.6 Range of Values

The immediately preceding example demonstrates the need to express ranges of values, such as 20716 through 20718, or 20718 through 20721. Negative ranges are commonly expressible as well. So, 20716 through 20718, but not 20719 would be expressible. Again, as in the last case this could involve both single and multiple column cases. The involved Reference Data tables are identified and described in Table 9.

Reference Data Tables for Range of Values	
Code Fact Table Code Assignment	The row of data from within the reference data table that contains the primary key value that, in turn, becomes the foreign key value of the reference data column from within the appropriate fact table.
Code Type	The row of data that contains the name of the code table column about which reference data values are created. This row contains the effective start and end date for the Code Type. This is to ensure that the code value that exists is from a current Code Type column.
Code Type Structure	If there is multiple code table columns involved, this table contains multiple rows of data, one each for the different Code Types that must be associated with each other as a multiple content columns. The relationship that exists between the multiple columns essentially is: "Also involves" as the active relationship and "Is involved with" as the passive relationship.
Code Type Structure Type	If there are multiple code table columns involved, this table contains the multiple content column subset collection data that supports the identification of collections of multiple content columns. In this example, the individual active and passive descriptive names are identified and defined. Defined also is the overall meaning of this Code Type Structure relationship type.
Code Value	The row of data that contains the specific range of values, code, meaning and description that is represented by the appropriate column in the fact table. The effective start and end dates must be appropriately set so that the represented fact table value represents a current value. **Note:** the data model design shown in Figure 1 does not properly reflect the ability to express ranges of values. Nor does it properly express NOT values for value ranges. It does not contain the ability of mixtures of ranges of values and NOT sets of ranges of values Finally, it does not express the ability to have a mixture of discrete and ranges of values. Changes in the data model are necessary to accomplish this capability.
Code Value Structure	This is not necessary for this reference case.
Code Value Structure Type	This is not necessary for this reference case.
Fact Table	The column from within the appropriate fact table that contains the reference data column. The value from this column acts as a foreign key value reference to the Code Fact Table Code Assignment row that, in turn, represents the reference data Code Value.

Table 9. Database Tables for Range of Values

5.0 Reference Data Updating and Maintenance

Issues related to changes to reference data values parallels the Section 4 six reference data cases. Some of the update anomalies have already been addressed above and are cited. Of overall concern is what happens to a reference data value that was once valid, and through a change, has now become invalid? In this situation, there will have to be a thorough examination of how the formerly valid reference data values were stored in the database. How are they discovered? Can it be automatic? Or should it ever be automatic? Regardless, how will the already stored invalid values be changed? Will prior generated reports have to be re-executed and the difference in counts and other statistical and/or financial summaries have to represented along with explanation?

5.1 Single Content Column

The key issue with single content column value changes is whether history is to be retained or not. If not, there is no need to retain the previous value. In this case, there would be an automatic and complete recasting of one value and meaning to a different value and meaning. However, if history needs to be retained, this case immediately turns into a mapped value case where the old values are mapped to the new values.

Of especial concern is whether a reference data value's meaning has changed. If it has, and if the new meaning is to have the prior meanings retained, then again, this case immediately transforms to a mapped values case.

5.2 Multiple Content Columns

The updating concerns in this case are the same as for single content columns except that the value changes are possibly tracked as collection.

5.3 Sequenced

Sequenced values are inherently more complex. That's because, for example, two additional states might have been introduced prior to a given state. All the business rules that enable an exact determination of all the states need to be very carefully examined to be assured that when the rules are executed the new set of sequenced states are able to materialized.

5.4 Mapped Values

Mapped values are also inherently more complex. That is because there will be prior mappings and new mappings. Each changed mapping set has to have the same end-date and start-dates so as to avoid unmapped intervals. If the mappings change, the various statistics, especially as they relate to importing and exporting from external systems may change and need to be explained. This will be especially so in situations where the quantity of source and target mapped values have changed.

5.5 Discrete Values

Discrete values are similar to single content column values except if an existing value becomes no longer valid. For example, if there were the discrete values, 20716, 20717, 20718, 20720, and 20721 and under a new scheme 20718 is now seen as invalid, the 20718 Value Case transforms to a mapped value case because the value 20718 might be mapped to either 20717 or 20720. Otherwise the reference data value becomes invalid and cannot remain in the database as such. If a new value is allowed, it may be necessary to revisit prior business event executions to determine if previously excluded data should now be included. Similarly, if a value is no longer allowed, prior data inclusions would have to be discovered and examined to determine what to do about what now becomes invalid data.

5.6 Range Of Values

Ranges of values are similar to discrete values except if an existing begin or end value becomes no longer valid. For example, if there was the existing range, 20716 through 20717, and the new scheme was 20716 through 20721, an examination would have to be conducted to discover if previously invalided records with the value 20718 and now need to be included. Similarly, if the new range was 20716, through 20720, an examination of some existing records (e.g., 20721) that formerly passed a range test would have to be examined to determine appropriate actions.

6.0 Recasting Data

The most critical issue surrounding reference data is what to do when acceptable values change. Within the reference data community this is sometimes called "recasting." Simply, recasting means that when a reference data value and/or meaning are changed, the understanding resulting from effectively the same queries and/or reports may also change. The three common cases are:

- Single value change (with and without history).

- Organizational relationship change (with and without history)
- Meaning Change (with and without history)

For the first case, single value change (with and without history), some changes are simple and others are significantly more complex. Simple cases change, for example, the value of "F" to "Female." In this situation, there is no recasting of the meaning of reference data value, only the value itself. Reports that include historical data will have to be able to not only obtain reference data values via their surrogate reference data values, but also have to discover and access any changed values that are, by reference data design represented by different reference data surrogate Id values.

For the second case, organizational relationship changes (with and without history), zip codes provide a ready but simple example. Zip codes change from time to time. Some entirely new ones are added, and some existing ones are split. For example, the "city", Hershey, PA does not really exist. It's just a U.S. Postal Service designation. If the USPS changes "Hershey, PA" to "Derry Township, PA" what happens to all the existing addresses for "Hershey, PA?" This would cause certain existing addresses that were formerly valid to now be invalid. Additionally, it might appear that Deny Township doubled in size if the report was based on its zip code. More importantly is the possibility of invalid addresses that were once valid and are now stored in a database. How are they discovered? How are they corrected?

A more critical example of organizational relationship changes would be the history of certain organizations within a database. Organizations can change through either acquisitions or losses. How is that history of changes reported over time? From one year to the next a given organization can take on a completely different set of statistics. If longitudinal graphs are shown, there could be some real surprises.

For the third case, meaning change (with and without history), reference data recasting can be very problematic. For example, suppose there are five values before, such as Proposed, Draft, Final, Terminated and Revised. Suppose the meaning of Terminated is changed from its meaning, Administrative Reasons, to also be understood as Administrative and Fraud Reasons. Grant Applications that were formerly denied for simply administrative reasons may now be seen as being denied for a very different set of reasons. While such a change can be accomplished through the reference data case, mapping, the real issue is reporting.

All these cases require very careful analysis and assessment to understand the complete implications of making reference data changes. All three reference data recasting cases apply to all six of the reference data types.

7.0 Conclusions

The practical application of the points made in this paper include:

- There is a critical need to reference data management as reference data are often the key characteristics employed for selection, counting, sorting, and grouping.

- There exists a generalized data model that can be employed for reference data management.

- The six common cases that must be addressed to effect reference data management are: single content column, multiple content columns, sequenced values, mapped values, discrete values, and range of values.

- Strategies have to be created and set into motion to accomplish reference data management that is both efficient and effective, and that maintains to the maximum extent possible, the histories of values and meanings as the values were originally implemented.

8.0 References

The following references to Whitemarsh materials provide a more detailed exposition practical application of the significant content of this paper.

Paper	URL
Data Modeler Architecture and Concept Of Operations Reverse and Forward Engineering Guide Metabase Module User Guides	http://www.wiscorp.com/metabase_demo.html
Comprehensive Metadata Management (short paper).	http://www.wiscorp.com/featured_papers.html
DAMA 2002 - Metadata Architecture for Enterprise Wide Data Sharing - Problem Specification DAMA 2003 - Metadata Architecture for Enterprise Wide Data Sharing - Problem Solution	http://www.wiscorp.com/DatabaseDesignInformation.html

The following documents are available for Whitemarsh Website Members. The URLs that follow provide descriptions of the pages. Members should log in and proceed to the appropriate page, e.g., Enterprise Database, find the book, paper, or course and perform the download.

Paper	URL
Data Management Program - Data Standards Architectures And Implementation Data Management Program - Engineering Data Management Program - Metadata Architecture For Data Sharing Data Management Program - Tag And Post Vs Data Standardization	http://www.wiscorp.com/EnterpriseDatabase.htm
Iterations of Database Design	http://www.wiscorp.com/DatabaseDesign.htm
Data Is Executed Policy	http://www.wiscorp.com/DatabaseProjects.htm
A Column By Any Other Name is Not a Data Element Presentation (several paper and courses) Achieving Data Standardization (several papers, a book and courses) Achieving Enterprise Wide Data Semantics Standardization An Old Saw That Just Wont Cut An Old Saw That Just Wont Cut - Software Implementation Report Analysis of the DISA 8320 Data Standardization Approach Data Standardization Work Plan The Data Standardization Problem	http://www.wiscorp.com/DataQuality.htm