**DateTime Transformations in Data Warehouse**

**Last Edited 10/11/2012**

**Background.** The way that Vista stores dates and times allows for values that cannot be translated into other data systems that are “data typed”, i.e. that only allow values it thinks appropriate for that type of data. For example, dates in Vista can be stored that have no day-of-month or an hour value of 24 or greater. Since CDW is based on SQL Server, which is data typed, some date values and some time values from Vista cannot be stored in a date(time) field in CDW. CDW can and sometimes has stored the Vista date value as text (varchar) but the usefulness in text format is limited. It is difficult to perform date arithmetic (differences) and the performance from invoking custom functions can be poor. To store values that do not correspond to valid SQL Server dates, there must be a data transformation to a value that SQL Server recognizes as legitimate.

The CDW Governance Board on April 3, 2012 determined that incomplete or otherwise invalid dates from Vista should be transformed into valid SQL Server dates in the CDW when practical. Furthermore the type of transformation applied to a particular data element should be documented in the data. The governance board suggested that for each datetime column that requires some transformation that a separate column be created to indicate what transformation rule was applied.

Some datetime fields in Vista do not contain values that require any transformations to convert to SQL Server data types. The Governance board agreed that the domain team (perhaps recommended by the Architecture team) would determine if a particular datetime field requires the additional transformation architecture, described in this document.

The first release of tables containing this new architecture is expected July, 2012. All tables released after that will have the new datetime architecture.

**The Rules**

The desired outcomes of the new architecture are 1) the performance for queries that do not need to examine the specific transformation rule applied is maximized and 2) specific common transformation scenarios do not suffer any loss of information from the original Vista value (e.g. the fact there was no day of month) and 3) the amount of information redundancy written to the table is minimized and 4) the applicable transformation logic are exposed in a lookup table. The BISL team has implemented the following rules.

|  |  |
| --- | --- |
| TransformRule | TransformRuleDescription |
| 1 | If DayOfMonth is missing then set DayOfMonth=1 |
| 2 | If Month is missing then set Month=1 |
| 4 | If Date component is invalid for reasons other than missing DayOfMonth or missing Month then set Date to null |
| 8 | If Hour=24 and rest of time component is valid then set Hour =23 (minutes and seconds remain the same) |
| 16 | If time component is invalid for reason other than Hour=24 then set time component to 00:00:00 |

**Implementation Details**

For each source date field for which this architecture is necessary three columns are generated. The following table shows the types.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Column Type** | | |
|  | **Base column (transformed)** | **Vista formatted date** | **Foreign key to Dim.Transform** |
| **Content** | The actual datetime or date value, transformed when necessary | The Vista formatted string | SID column for the transform that was applied. |
| **Column name** | xxxDateTime or xxxDate | xxxVistaErrorDate | xxxDateTimeTransformSID  or xxxDateTransformSID |
| **Example column name** | EnteredOnDateTime | EnteredOnVistaErrorDate | EnteredOnDateTimeTransformSID |
| **SQL data type** | datetime2(0) or date | varchar(50) | bigint |
| **Which rows populated** | All rows that can be written to the SQL Server typed column with or without a transformation rule | Only rows for which a rule was applied, including the rule that says no value can be written to base column | Only rows for which a rule was applied, including the rule that says no value can be written to base column |
| **Source field mapping in MDR** | Required | Required to be same as base column | Required to be same as base column |

Following are some example data scenarios.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Original source Vista Date (not in DW table)** | **Base column (transformed)** | **VistaErrorDate** | **TransformSID** | **Explanation** |
| 3010617.2315 | 2001-06-17 23:15 | Null | Null | No rule applied |
| 3010600 | 2001-06-01 | 3010600 | 1 | Rule #1 applied for DayOfMonth |
| 3010000 | 2001-01-01 | 3010000 | 3 | Rules #1 and 2 both applied (add 1 + 2) |
| 301a028 | Null | 301a028 | 4 | Invalid format of date component |
| 3010617.2415 | 2001-06-17 23:15 | 3010617.2415 | 8 | Rule #8 applied. Notice this has same datetime value as the first example but together with transform rule original value can be derived. |
| 3010600.1265 | 2001-06-01 | 3010600.1265 | 17 | Rules #1 and #16 perhaps artificial in that these two rules may not ever be invoked together. |

The xxxTransformSID fields in the DW fact tables have data type bigint. Records requiring no transform are represented as null. The large majority of the records will have no transformation applied. Standard use of table compression in CDW should minimize the increase in tables’ sizes if these types of fields are null for most records. This xxxTransformSID foreign key references a view CDWWork.Dim.Transform. There is one record in Dim.Transform for each rule or possible combination of rules defined above. The actual transformations occur at the time of extract from Vista by the method employed by the BISL Vista Extract Team.

The conventional query to see descriptions of transform logic would look like:

Select m.MyTableSID, m.MyExampleDateTime, t.TransformDescription as MyExampleDateTimeTransformDescription

From MyTable m

left join Dim.Transform t

on m.MyExampleDateTimeTransformSID=t.TransformSID

Following are other possible queries for advanced users.

--Retrieve records where at least one transform rule was applied

Select MyTableSID, MyExampleDateTime

From MyTable

Where MyExampleDateTimeTransformSID is not null

--Retrieve records where no transform rule was applied

Select MyTableSID, MyExampleDateTime

From MyTable

Where MyExampleDateTimeTransformSID is null

The DWTechnicalDescription of each xxxTransformSID field contains the following text.

*This is a code indicating the tranformation(s) logic used to transform the Vista datetime value into a SQL date or datetime data type. A transformation is necessary when the Vista value does not translate directly, e.g. there is no day of month. The logic is applied and the result is written to the field of the same name as this field without the suffix "TransformSID". If the value is null no transformation rule was applied. See a complete description on how to use this code on the Metadata SharePoint page.*

**Cutoff Logic**

Each fact table must have a date field identified for comparison to an initial cutoff date (nominally 10/1/1999) when loading the data into the RDW database. Records having a value in that cutoff field that is earlier than the cutoff date are excluded from the load (although typically records having a null value are included). New table versions, having the datetime architecture described in this document, will use the transformed date/datetime field as the cutoff field. This is a departure from the old practice of using of a VistaRawDate as the cutoff field.