1. Introduction

The Integration Layer is the second layer in the reference Data Warehouse solution architecture. This Layer is not designed to be accessible by (end) users of the information but serves as the true Data Warehouse Layer, where information is maintained in such a way that it is both resilient and flexible. The Integration Layer sources its information from the Staging Area and stores it in a consistent and atomic way, without applying business logic. This data can then be presented in a consumable form in the Presentation Layer.

This document defines the Integration Layer and describes the involved steps and techniques. Various references to other documents of the ETL Framework will be made, including error handling and metadata management. Ultimately, this document provides the information to configure the Integration Layer in a flexible and modular way.

The design and approach for modelling the Integration Layer is a project specific decisions which needs to be captured in the Solution Architecture (including supporting reasoning). Examples are 3NF or Data Vault approaches, but other hybrid techniques can be applied as well as long as they are in sync with the framework guiding principles.

If the Solution Architecture for a project is defined as ‘2-tiered’ – the classic Kimball approach – the Integration Layer is not implemented.

1. Integration Layer overview

The Integration Layer, or the process from staging to integration, is comprised of two parts (or areas): the Integration Area and the Interpretation Area. The Integration Layer is a persistent Layer.

The Integration Area is the phase where data from the Staging Layer is re-modelled and changes in attributes are captured and tracked using the Slowly Changing Dimension (SCD) 2 technique. Surrogate keys for new records are also identified and assigned prior to the loading of the attributes.

An optional step is then available for the data to be manipulated and for business rules (or logic) to be applied. Data in the Integration Area can be de-duplicated, standardised, validated and/or cleansed, and stored away in the Interpretation Area. This design provides a flexible approach to data cleansing and interpretation at a granular level, while still keeping a flexible yet consistent design.

This is because in the Integration Area the data remains largely pristine and nothing gets changed or altered. In doing so, it offers the flexibility of presenting the data in more than one way, while the original data remains unchanged. Auditing of data is also made easy as original image of data (obtained from source system) is retained and therefore traceable.

* 1. High level Integration Layer overview

The position of the Integration Layer in the overall architecture is:

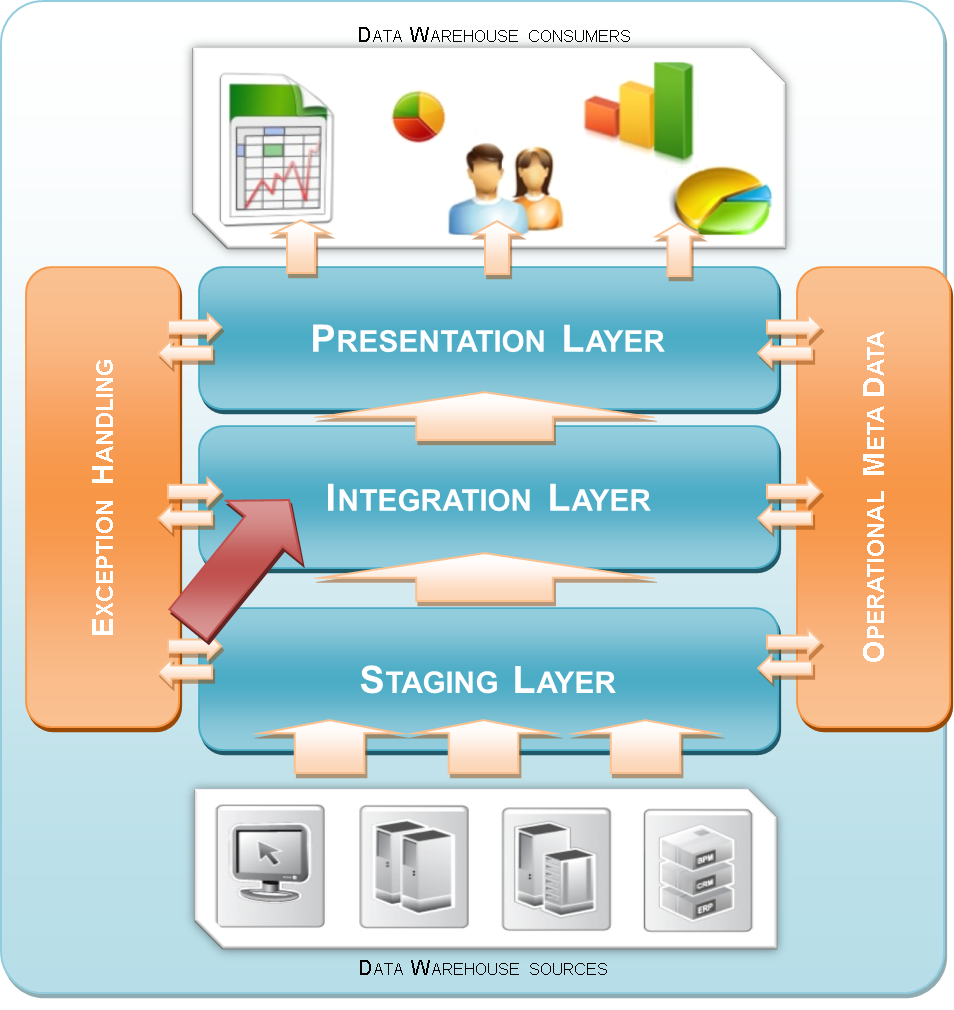


Figure 1: Positioning

* 1. Detailed Integration Layer overview

As documented in the detailed architecture overview (A100 - Outline Architecture), data from the Staging Area is loaded into an Integration Area with the option of being further transformed into an Interpretation Area. While each area serves a specific purpose, the underlying data model / structure in both the Integration and Interpretation Areas remain the same.

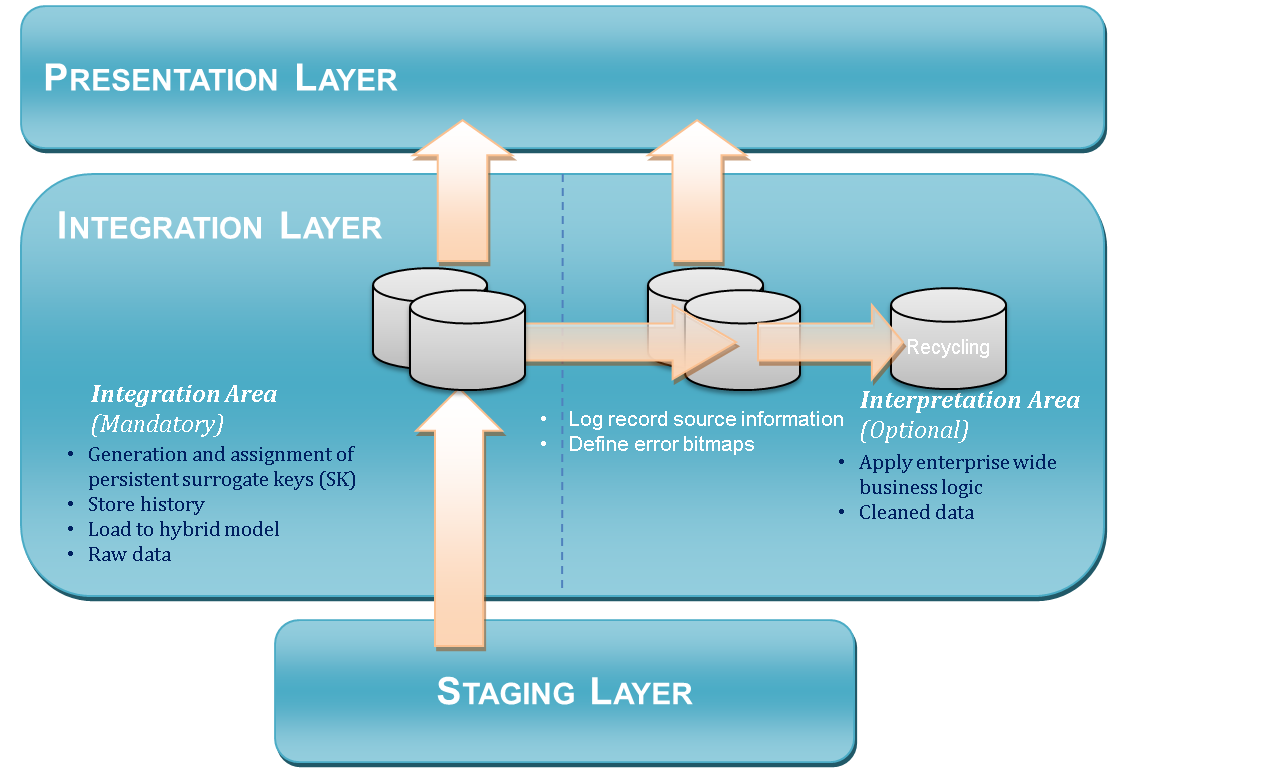


Figure 2: Integration Layer

In the Integration Area, data from the Staging Area is first processed by identifying new business keys and assigning surrogate keys for newly discovered business keys. This management of Data Warehouse keys is vital and it is the first ETL process that happens in the Integration Area. Data (from staging) is then further processed by loading them into the remaining integration (hybrid) model tables. Changes to the historical data are tracked by means of using Slowly Changing Dimension Type-2 (SCD2) approach. Effectively, this decouples key distribution from managing history.

It is worthwhile to note that data captured in the Integration Area remains in its pristine, raw form. Data taken from Staging Area is modelled to support the Data Warehouse, but the content does not change.

In the Interpretation Area, enterprise wide business rules are applied to the data residing in the Integration Area. This is done to accomplish any of the following objectives:

* Data cleansing / scrubbing
* Data enrichment
* Data standardisation
* Deduplication
* Validation

The intention of the Interpretation Area is to reduce replication of the rules in ETL processes towards the Presentation Layer. An example would be the formatting of addresses where abbreviations such as “st” or “rd” get expanded to “street” and “road”, and the validation of phone numbers. It also could be combining various data elements from Human Resource related tables into Payroll information.

In separating the state of the data into two physical forms (original and modified), it gives the flexibility of applying multiple sets of rule to a specific data set, in order to suit specific needs. An example is provided in Appendix A.

1. The Integration Area

The Integration Area is modelled differently from the Staging Area. In the Integration Area data is divided into common entities which form the core of the Data Warehouse model. Various modelling techniques can be applied for the Integration Layer (3NF, Data Vault, Anchor) as long as the same technique is used for both areas. Regardless of the approach the tables in the Integration Area are either:

* Entity / Surrogate key tables. The function of a surrogate key table is to create a unique list of instances for that particular entity. For example: all employee numbers. The source key / logical keys are the only attributes which are copied from the Staging Area. This includes 3NF key tables (if applied this way), Data Vault Hub tables and Anchor tables
* Other tables. The rest of the model depends on the chosen modelling technique, but examples of other Integration Area tables are:
  + History or reference type tables which contain all the attributes but the logical key. These tables inherit the surrogate key of the main entity. This is the most common other entity type and will be used as an example. This includes Data Vault Satellite tables, 3NF History tables and Anchor attributes.
  + Relationship tables which contain (information about) the relationships between main entities. This includes 3NF intersection tables and Data Vault Link tables
  1. Surrogate Key table structure

The following metadata attributes are mandatory for the Surrogate Key tables:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| <entity>\_SK | INTEGER or CHAR(32) / Hash | The Data Warehouse key; an unique identifier and also the primary key which is issued for each record in the table. It can be a meaningless key (sequence) or hashed value |
| OMD\_INSERT\_MODULE\_INSTANCE\_ID | INTEGER | Default OMD; logging which process has inserted the record |
| OMD\_FIRST\_SEEN\_DATETIME | DATETIME (high precision) | This is the time that the record has been presented to the Data Warehouse environment. This is not the system date/time for insert however, but the original processing time for the records to be loaded into the Staging Area.  The Insert Date/Time is the conceptual Event Date/Time; the date time when the source event was triggered or the change in the source has taken place. It can be the moment a user updated a record in a source system, or the trigger which caused a message to be sent. |
| OMD\_RECORD\_SOURCE\_ID | INTEGER | The relation to the OMD table which contains the identification of the source system that originally supplied the information. |
| <business key> | Depending | The business key value |

The following attributes are optional for the Surrogate Key tables depending on the approach for Data Modelling:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| OMD\_EFFECTIVE\_DATETIME | DATETIME (high precision) | Start of the validity period for the record. Equal to the OMD\_INSERT\_DATETIME; this is not the system date/time, but the information recorded during the Staging Area ETL process. |
| OMD\_EXPIRY\_DATETIME | DATETIME (high precision) | The date time when the record was closed. Records are closes based on changes in the history (alteration or deletion). The value of this attribute is the value of the valid start date time of the previous related. The default value is 99991231 23:59:59. |
| OMD\_CURRENT\_RECORD\_INDICATOR | VARCHAR(100) | The flag (Y/N) whether this record is active. This makes selection and querying easier, but is essentially twice redundant. If possible use the Expiry Date/Time for this purpose. |
| OMD\_UPDATE\_MODULE\_INSTANCE\_ID | INTEGER | The module ID of the ETL process which has updated the record. |

The use of a ‘valid period of time’ (start and end date time) including the current record indicator is optional. There can be sound reasons for including these metadata attributes in a surrogate key table when source systems can reuse their own keys and specific logic has to be created to determine if a reused key is in fact a new instance of an entity or that an old one has been reopened.

* 1. History table structure

The following metadata attributes are present in the rest of the integration area tables. This does depend on the chosen modelling technique. For this paragraph history or reference type tables are used as an example.

The following metadata attributes are mandatory for the history tables:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| <entity>\_SK | I INTEGER or CHAR(32) / Hash | The Data Warehouse key; an unique identifier and also the primary key which is issued for each record in the table. It can be a meaningless key (sequence) or hashed value. This is inherited from the parent table as Foreign Key |
| OMD\_EFFECTIVE\_DATETIME | DATETIME (high precision) | Start of the validity period for a record. Populated by the OMD\_INSERT\_DATETIME value from the Staging Area this is not the system date/time but the information recorded during the Staging Area ETL process. |
| OMD\_INSERT\_MODULE\_INSTANCE\_ID | INTEGER | Default OMD attribute for any table for logging which process has inserted the record. |
| OMD\_UPDATE\_MODULE\_INSTANCE\_ID | INTEGER | The module ID of the ETL process which has updated the record. |
| OMD\_RECORD\_SOURCE\_ID | INTEGER | The relation to the OMD table which contains the identification of the source system that originally supplied the information. |
| OMD\_SOURCE\_ROW\_ID | INTEGER | Copied from the Staging Area. The combination of OMD\_INSERT\_MODULE\_INSTANCE\_ID and OMD\_SOURCE\_ROW\_ID always relate back to a single History Area record |
| OMD\_DELETED\_RECORD\_INDICATOR | VARCHAR(100) | This flag (Y/N) indicates that the record has been deleted from the source system. |

The following attributes are optional for the history tables in the Integration Layer:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| OMD\_EXPIRY\_DATETIME | DATETIME (high precision) | The date time when the record was closed. Records are closes based on changes in the history (alteration or deletion). The value of this attribute is the value of the valid start date time of the previous related record minus 1 second. The default value is 99991231 23:59:59. |
| OMD\_CURRENT\_RECORD\_INDICATOR | VARCHAR(100) | The flag (Y/N) whether this record is active. This makes selection and querying easier. |
|  |  |  |
| OMD\_HASH\_FULL\_RECORD | CHAR(32) | A checksum for record comparison requires storing a checksum value as an attribute. |

In history tables the Primary Key is composed of the <entity\_SK> and the OMD\_EXPIRY\_DATETIME attributes.

The optional attributes include all reference data which relates to the entity Data Warehouse key. In the example of an employee record the person ID would lead to the generation of a new surrogate key, while all descriptive attributes are placed in the history table. Depending on considerations regarding volume or width of the table (in terms of records, bytes) different history records can be placed in different history tables, but always with the same structure as described in the above table.

* 1. Relationship table structure

The relationship table structure is largely dependent on the applied modelling technique, but in the most concise definition contains the following attributes:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| <relationship\_SK> | INTEGER or CHAR(32) / Hash | The Data Warehouse key; an unique identifier and also the primary key which is issued for each record in the table. It can be a meaningless key (sequence) or hashed value |
| <entity>\_SK (one side of the relationship) | INTEGER or CHAR(32) / Hash | A unique identifier; the Data Warehouse key obtained from the Surrogate Key table. |
| <entity>\_SK (other side of the relationship) | INTEGER or CHAR(32) / Hash | A unique identifier; the Data Warehouse key obtained from the Surrogate Key table. |
| OMD\_INSERT\_MODULE\_INSTANCE\_ID | INTEGER | Default OMD; logging which process has inserted the record |
| OMD\_FIRST\_SEEN\_DATETIME | DATETIME (high precision) | This is the time that the record has been presented to the Data Warehouse environment. This is not the system date/time for insert however, but the processing time for the records to be moved into the Staging Area. |
| OMD\_RECORD\_SOURCE\_ID | INTEGER | The relation to the OMD table which contains the identification of the source system that originally supplied the information. |

This structure essentially links two types of information together, and can be expanded to include relationship attributes or relations to history or properties specific for that relationship (including transactions or historical facts).

* 1. Integration Area relation to error handling

Once configured and tested properly there are almost no possible technical errors in the Integration Layer (except infrastructure issues). As all staging tables that contain business keys get processed or pre-processed in the Integration area, failure in the lookup of a business key against a surrogate key (SK) table can never occur.

In the event of timing issues (early arriving facts or later arriving dimensions) where factual data gets processed before dimensional data, pre-processing of the fact data will result in the business key itself being added to the surrogate key table. This means every workflow/Batch requires the capability to issue surrogate keys itself.

This record in the surrogate key table acts as a placeholder and can optionally be identified as being erroneous or incomplete. This is optional since the metadata attributes enable the system to view which interface has loaded the data. If this is deemed a serious error then the error bitmap is available to filter the records at the presentation layer if there is a requirement to do so.

* 1. Integration Area development guidelines
* Source keys, also known as business keys, natural keys or logical keys usually lead to surrogate key tables. A composite key might indicate separate surrogate key entities, but not necessarily so. Also, a business key might be composed of more than one source attribute
* If the ETL platform allows it, prefix the ‘area’ or ‘folder’ in the ETL tool with ‘200\_’ because this is the first area in the second layer in the architecture. This forces most ETL tools to sort the folders in the way the architecture handles the data, making the development environment better readable
* The Integration Area folder or module in ETL contains all table definitions, all mappings loading from any source (folder) to the integration area folder and all relevant objects for this process
* The Integration Area only loads data from the Staging Layer
* Everything is copied as-is, no transformations are done other than formatting data types. This may never lead to errors! Errors can only be caused by incorrect dependencies (i.e. loading history before distributing surrogate keys), this is not something that can occur in a properly tested environment
* The Staging Area may contain multiple loads of data which could do not necessarily have arrived in the correct order (messaging!) data has to be sorted based on the conceptual Event Date/Time before further processing

1. The Interpretation Area

The structure of the Interpretation Area closely follows the conventions of the Integration Area; the same modelling technique is used and the same type of tables, metadata attributes and rules apply. Because of this similarity, only the differences will be documented.

The Interpretation Area is an optional area where enterprise wide business rules can be effectuated and only sources its data from the Integration Area. From an ETL perspective there is no set ‘pattern’ to load data into the Interpretation Area. Because all data is derived (and therefore redundant) this area can even be virtualised and/or dropped and recreated. Additionally, 3rd party matching or fuzzy logic software can be implemented in addition to standard ETL processing.

The only real difference is that for a certain table not every sibling has to be present in this area. Depending on the applied business rules the Interpretation Area dataset can be a subset of the integration layer dataset, or a fully updated one.

The Interpretation Area contains derived data from the Integration Area and this can be done for specific tables or for complete or mixed entity sets. For instance, if only cleansing of a specific history table is necessary it will be sufficient to only store a new derived history table which still links to the integration area surrogate key table. If advanced algorithms like deduplication / match and survive are used this might require the creation of a new main entity with both a surrogate key table, history table and perhaps a relationship table.

* 1. Interpretation Area development guidelines

Integration Area ETL can be very specific because of the implementation of business logic. Because of this a typical Interpretation Area pattern cannot be defined. There are however some points of attention:

* If the ETL platform allows it, prefix the ‘area’ or ‘folder’ in the ETL tool with ‘250\_’ because this is the second step in the second layer in the architecture. This forces most ETL tools to sort the folders in the way the architecture handles the data
* The Interpretation Area folder contains all table definitions, all mappings loading from any source (folder) to the Interpretation Area folder and all relevant objects for this process
* The Interpretation Area only loads data directly from the Integration Area
* An Integration Area table can never be a target for Interpretation Area ETL
  1. Interpretation Area relation to error handling

The Interpretation Area can implement the error bitmap concept, or any error handling that might be needed for a specific purpose. As explained in the error handling document (A160 Error handling and recycling) the error bitmap can be used to store multiple errors for a single row and to provide a flexible way to load data to the Presentation Layer.

For this purpose the Interpretation Area tables may contain the following error handling attributes.

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| ERROR\_BITMAP | INTEGER | The numeric value representing the various errors. |
| ERROR\_DESCRIPTION | VARCHAR(100) | Additional explanation of the error, containing relevant values from the erroneous record. |
| ERROR\_RECYCLE\_COUNT | INTEGER | A counter recording how many times this error was selected from the recycling table and re-processed incorrectly. |

Appendix A: Data modelling example

The following example explains how the Integration Area and Interpretation Area work together in the same Layer.

Employee

(Surrogate key table)

Employee History with raw address data

Special employee

(Surrogate key table)

Relationship between Employees and Special Employees

Employee History with correct addresses

***Integration Area***

*(Operational Data Vault)*

***Interpretation Area***

*(Business Data Vault)*

In this example the dataset starts with employee records which are split between an employee surrogate key table and a table in which the history of the employee attributes are tracked. In this case attributes include address information. Since this is the raw data coming from the Staging Area the contents will be stored in the Integration Area.

After the data has been loaded certain enterprise wide business rules are effectuated; this is done in the Interpretation Area. This results in the creation of a subset of employees which have certain qualities into an entity ‘special employees’. The relationship of which employee is a special employee is stored in a relationship table or linked directly to the original table, depending on the modelling technique. Also, address information is updated for the employee. Technically this will result in a new history table with the updated values while still retaining the link to the original values from the Integration Area.

The same (simplified) example with data for the Integration Layer would be:

EMPLOYEE:

|  |  |  |
| --- | --- | --- |
| **ID** | **Logical Key** | **Source Row ID** |
| 1 | 201001 | 35 |
| 2 | 201004 | 36 |
| 3 | 201245 | 37 |

EMPLOYEE HISTORY (with raw address data):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **First Name** | **Last Name** | **Address** | **Effective Date** | **Expiry Date** |
| 1 | John | Doe | 40 George St. | 20080101 | 99991231 |
| 2 | John | Doe | 40 George Street | 20080101 | 99991231 |
| 3 | Peter | Smith | 70 York St. | 20080504 | 99991231 |

After applying a deduplication and cleansing business rule the following tables will be populated in the Interpretation Area. In this case two records are merged into one using a specific match and survive algorithm. The result is a set of two employees instead of the original three.

SPECIAL EMPLOYEE (selected by business rules):

|  |  |  |
| --- | --- | --- |
| **ID** | **Logical Key** | **Source Row ID** |
| 2 | 201004 | 36 |
| 3 | 201245 | 37 |

The relationship between the original data and the cleaned data has to be maintained for audit purposes.

This table shows the relationship between the original and the cleansed data:

RELATIONSHIP BETWEEN EMPLOYEES AND SPECIAL EMPLOYEES:

|  |  |
| --- | --- |
| **ID original employee** | **ID special employee** |
| 1 | 2 |
| 2 | 2 |
| 3 | 3 |

Separately the employee history has been corrected to show the correct address information. While this table resides in the cleansing area it still relates to the original surrogate key table EMPLOYEE. There is no need to copy all the historical data, only the corrected attribute(s).

EMPLOYEE HISTORY (with correct address data):

|  |  |  |  |
| --- | --- | --- | --- |
| **ID original employee** | **Updated Address** | **Effective Date** | **Expiry Date** |
| 1 | 40 George Street | 20080101 | 99991231 |
| 2 | 40 George Street | 20080101 | 99991231 |
| 3 | 70 York St. | 20080504 | 99991231 |

In a similar fashion business rules can be used to detect and tag errors using the error bitmap. This will result in (updated) error bitmap attribute which will be stored in a SCD2 fashion in the Interpretation Area, just as every other attribute. The process which selects the data from the Interpretation Area can select the data based on the type of errors (if this is a requirement).