1. Introduction

The Presentation Layer is the third and final layer in the Data Warehouse architecture. This layer contains the Reporting Structure Area which is the area which is the only area accessible by the (end) users of the information though Business Intelligence software. The Presentation Layer is the part of the Data Warehouse which can be modelled in any way as long as it suits the information and / or software requirements.

This document defines this layer and describes exactly what the required steps and solutions are. Many references to other parts of the ETL Framework will be made, including error handling and metadata management. The document aims to provide the information to configure the Presentation Layer in a modular way.

1. Presentation Layer overview

The Presentation Layer, or the process from integration to presentation, is comprised of two parts (or areas): the Helper Area and the Reporting Structure Area. It is the part of the Data Warehouse where the business logic is applied. The Presentation Layer is not defined as a persistent layer since it can be recreated from the Integration Layer at any time. However, due to performance considerations most Helper and Reporting Structure will be updated as opposed to a full recalculation.

The Presentation Layer does not necessarily have to be a sorted set of data, for instance it can be a set of views but with regards to performance there usually is a requirement for the information to be stored in the Presentation Layer permanently for quick access.

The Helper Area in an optional phase where calculations can be stored which are shared for every Data Mart. Using semi-aggregate tables, summaries or any shared calculation can prevent Data Marts from executing performance heavy aggregations more than once. The use of a helper area is very dependant of the way the Data Marts are modelled and what their purpose is.

The core area of the presentation layer is the Reporting Structure Area. Here all the specific subject areas of information are modelled from the integration layer model to a model which suits the requirements of the recipient of the information. Typically referred to as ‘Data Marts’, this area is the only part of the Data Warehouse that is accessible to end-users.

Considerations which affect the Presentation Layer in general and the Reporting Structure Area specifically are:

* Front-end analysis and reporting tooling used. Some reporting and analysis software performs better on a specific data model. In this case the Data Mart that is created for this tool has to conform to these requirements as much as possible
* Views on history. Not everyone needs a historical view on the data. For most reasons the current view (Type 1) might be sufficient whether other recipients might need to monitor how information changes over time
* Quality of data. All data is stored in the integration layer: its raw form in the integration area and the cleaned form in the cleansing area. By selecting information from the cleansing area, based on the error bitmap, varying levels of data can be selected. Alternatively there is the option of creating a data quality helper summary which uses the error bitmap to enable the selection of data with or without certain errors

* 1. High level Presentation layer overview

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

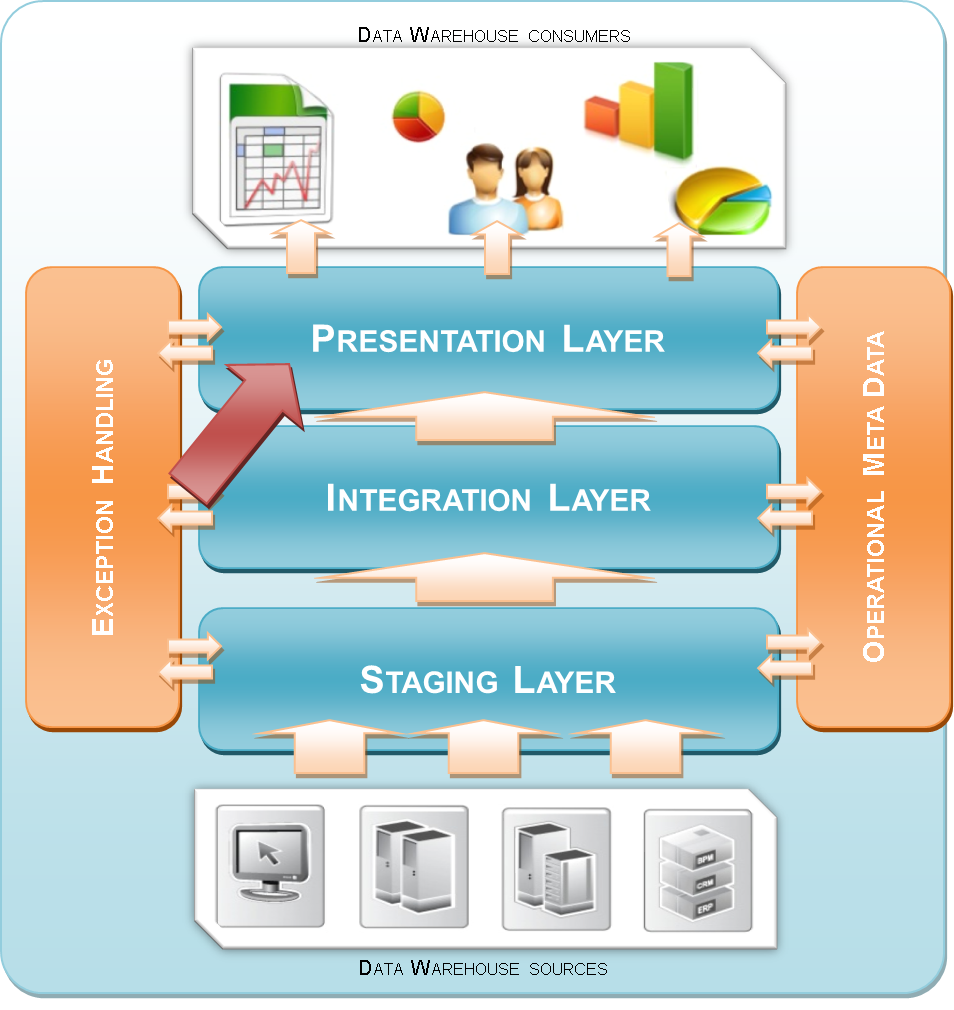


Figure 1: Positioning

* 1. Detailed Presentation Layer overview

As documented in the solution architecture overview, data from the Integration Layer is loaded into the Presentation Layer with the option of being temporarily (or semi-permanently) stored in a Helper Area. Because the Presentation Layer is defined to support the reporting software the model can differ substantially from the Integration Layer.

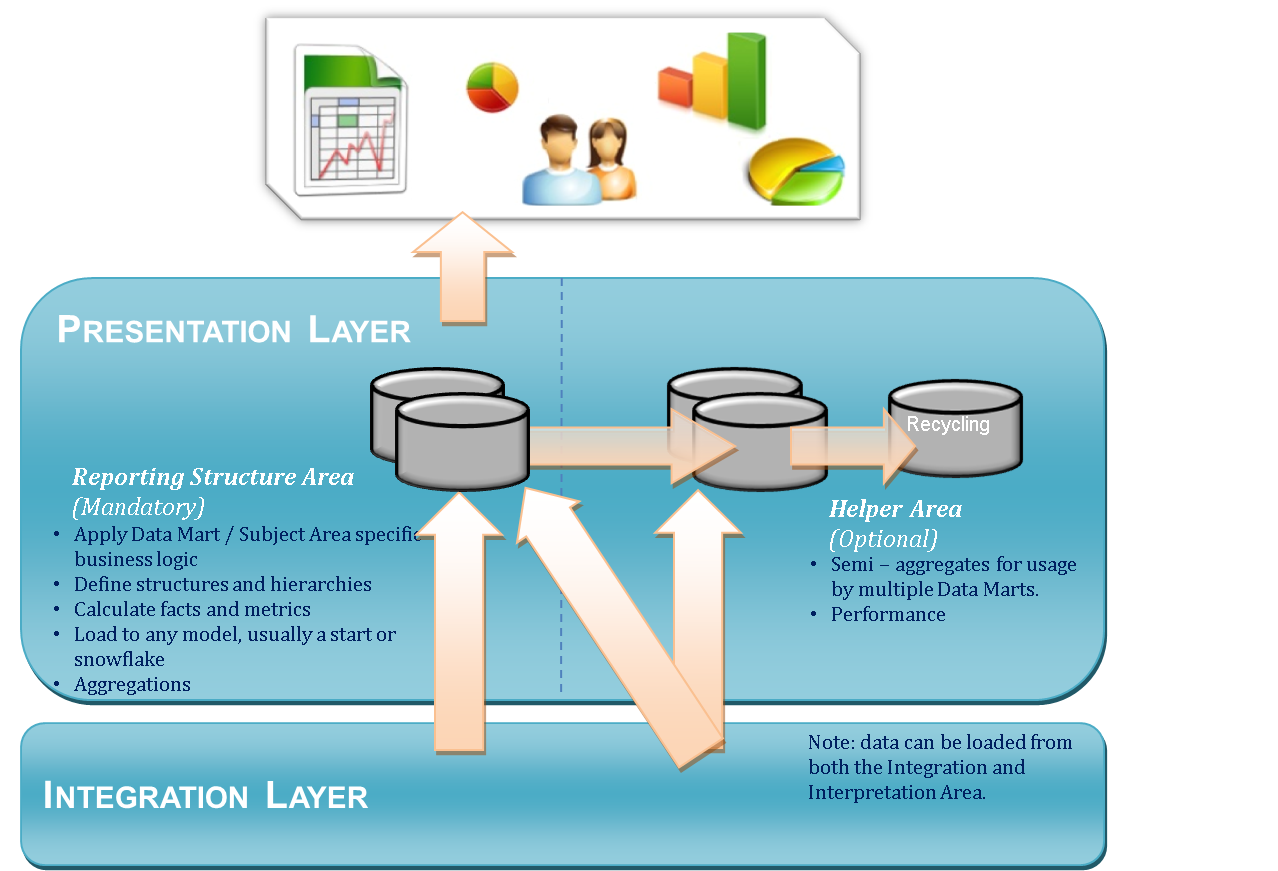


Figure 2: Presentation Layer

The most important aspect of the Presentation Layer is that it inherits its Data Warehouse keys from the Integration Layer, thus enabling backtracking of information. The detailed (raw or cleaned) data from the Integration Layer is still applicable to the keys in the Presentation Layer.

1. Load strategies
   1. Loading from Integration to Presentation

As the above diagram shows, both the Helper and Reporting Structure area processes can load from both the integration area and cleansing area. Do note that a Reporting Structure or Helper process almost always means implementing specific business rules to transform the data. The following combinations are possible:

**Combination 1:**

A Data Mart or helper table is created entirely based on the Integration Area. This may mean that there is no need for enterprise wide data cleansing (no deduplication for instance) and that the Data Mart is either too specific to share other aggregations or performance is not an issue. Eventually the Helper Area should support a Data Mart therefore the dashed arrow is displayed between the Helper and the Data Mart.

**Combination 2:**

A Data Mart or helper table is created entirely based on the Interpretation Area. This may mean that the Data Mart requires an indication of any errors in the enterprise wide business rules or that it requires an integrated data set. Not using the helper area again suggests an independent Data Mart or that performance is not an issue. Eventually the Helper Area should support a datamart therefore the dashed arrow is displayed between helper and Data Mart.

**Combination 3:**

A Data Mart is created both based on the Integration and the Interpretation Area and also uses a (shared) Helper table. This is a viable option if some treated data is used but other reference information might be usable directly.

**Combination 4:**

A Data Mart is based fully on the helper area. This suggests that several aggregations on a detailed level are available and that several Data Marts aggregate this to a higher level depending on their requirements. All business logic may already be done in the helper area. Of course the original data has to come from the Integration Layer therefore the dashed arrow is displayed as well.

**Combination 5:**

A Data Mart is based partly on the Helper Area and partly on the Integration and/or Interpretation Area. This could mean that certain Dimensions are shared between Data Marts but a Data Mart specific dimension has yet to be derived only for this particular Data Mart.

* 1. Loading and managing aggregate tables

The following load strategies are available for managing summary / aggregated tables. This is true for both the Helper and the Reporting Structure Area:

* Drop and create. Because all data is always present in the Integration Layer it will be possible to recalculate any summarised table
* Real time update. Select and modify existing records (or adding new ones) in small bits by recalculating the value at the defined grain. For instance a newly added record could lead to the subtraction of, say, 10 to the total value sum for a certain day and Dimension if the entire table would be recalculated. It will also be possible to update that specific fact record to the new value (existing value minus 10)
* Update by stacking (mutations). Another option would be to add new information to the Data Mart (usually the fact table) as a new record. In this case a new record would be added with the value difference. This is similar to calculation transactions from balance information. Summarisation in a front end will now display the actual result even though it technically consists of two records: the previous total sum and the change value
* Rollback and consolidate. This strategy is a refined approach to the update by stacking mechanism. Every set period of time the summary table could be recalculated in a quiet time period within the loading window. This will consolidate any change records into a new total value to improve performance

1. The Reporting Structure Area

The Reporting Structure Area is the final part of the Reference Architecture. A Data Mart is modelled for a specific purpose, audience and technical requirement. The complete Data Warehouse can contain very different data marts with different models and different ‘versions of the truth’ depending on the business needs. In the process from loading the data from the Integration layer to the Presentation layer most of the business logic is implemented.

* 1. Reporting Structure Area tables

Since this area aims to support the front-end tool or solution in the best possible way there is no fixed structure for the tables. The only guideline is the addition of the core OMD attributes.

The following attributes however are considered core, and therefore mandatory:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| <entity>\_SK | INTEGER | For Dimensions only; a Dimension should receive its own dedicated surrogate key since time-variant joins across Integration Layer tables create more records (as viewed from an Integration Area key perspective). This is a true surrogate key |
| OMD\_INSERT\_MODULE\_INSTANCE\_ID | INTEGER | Default OMD; logging which process has inserted the record |
| OMD\_EFFECTIVE\_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. |

Depending on the defined solution primary keys may be distributed here, for instance as part of creating a standard Dimension based on the information in the Integration Layer.

The following attributes are optional for the Presentation Layer:

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| <Integration Layer entity>\_SK | INTEGER | Any inherited surrogate keys from the Integration Layer essentially act as level keys in a Dimension. |
| 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\_DELETED\_RECORD\_INDICATOR | VARCHAR(100) | This flag (Y/N) indicates that the record has been deleted from the source system. |
| OMD\_UPDATE\_MODULE\_INSTANCE\_ID | INTEGER | The module ID of the ETL process which has updated the record. |
| OMD\_CHECKSUM (multiple may be required) | VARCHAR(100) | Using a checksum for record comparison requires storing a checksum value as an attribute. Multiple checksums may be implemented. |

* 1. Reporting Structure Area relation to error handling

Error handling for this area is documented part of the ‘A160 – Error handling and recycling’ document. All required error handling concepts (which lead to rejects of records) may be implemented here because information is stored in the Integration Layer as well.

* 1. Reporting Structure Area development guidelines
* If the ETL platform supports it, prefix the ‘area’ or ‘folder’ in the ETL tool with ‘350\_’ because this is the second step in the third layer in the architecture. This forces most ETL software to sort the folders in the way the architecture handles the data
* Create separate folders within this range for separate data marts because these can be viewed as stand-alone applications, with different purposes and for different audiences.
* The Reporting Structure Area folders contains all table definitions, all mappings loading from any source (folder) to the cleansing area folder and all relevant objects for this
* When designing Reporting Structure Area tables, especially typical Dimension tables, determine for every single attribute whether it is a Type 0, Type 1 or a Type 2 attribute

1. The Helper Area

The Helper Area is an optional area where semi-aggregates or any useful tables can be stored. These types of tables are usually added for either performance reasons or the wish to implement the same business logic in as few places as possible. Helper tables can be modelled in any way as long as they benefit the Reporting Structure Area. They are not accessible by users or front-end reporting and analysis tools.

By thoughtfully creating aggregate tables which can be shared by the Data Mart one could for instance create a Fact table on a certain aggregate level and have different Data Marts aggregate this table further depending on their needs. The business logic and performance demanding calculations only have to be done once this way.

* 1. Helper Area Structure

A Helper table can have any structure as long as it will support the connected Data Marts. Performance and limiting the duplicate implementation of business rules is the main goal. Usually though the Helper Area will follow the design of the Data Marts, which will in most cases lead to a star- or snowflake model. Regardless of this possible variety in table structures the audit trail always has to be preserved.

The following metadata attributes are mandatory for the Helper Area tables. The reasoning is that at the very least, the information should be related to an ETL process and date/time.

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Reasoning** |
| OMD\_INSERT\_MODULE\_INSTANCE\_ID | INTEGER | Default OMD; logging which process has inserted the record |
| OMD\_INSERT\_DATETIME | DATETIME | 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. |

* 1. Helper Area relation to error handling

Error handling for this area is documented part of the ‘A160 – Error handling and recycling’ document. All required error handling concepts (which lead to rejects of records) may be implemented here because information is stored in the Integration Layer as well.

* 1. Helper Area development guidelines
* If the ETL platform supports it, prefix the ‘area’ or ‘folder’ in the ETL tool with ‘300\_’ because this is the first area in the third layer in the architecture. This forces most ETL software to sort the folders in the way the architecture handles the data
* Reuse the Integration Area surrogate keys wherever possible. This further strengthens the audit capabilities and provides a standard level key in a Dimension