Boardwalktech Inc.

*User’s Manual*

Super Merge Guide

Table of Contents

[1 GENERAL INFORMATION 3](#_Toc391491504)

[1.1 Overview 3](#_Toc391491505)

[1.2 Purpose 3](#_Toc391491506)

[2 GETTING STARTED 4](#_Toc391491507)

[2.1 Pre – Requisites 4](#_Toc391491508)

[2.2 Create SQL Staging Table 4](#_Toc391491510)

[3 PROCESSING MERGE 5](#_Toc391491511)

[3.1 BRectDefinition Cuboid (Boardwalk Rectangle Definition) 5](#_Toc391491512)

[3.2 KeyStore Cuboid 6](#_Toc391491514)

[3.3 SQL Table to Boardwalk Cuboid 7](#_Toc391491515)

[3.3.1 Configure SQL Table to Boardwalk Cuboid Super Merge Rule 10](#_Toc391491516)

[3.4 Boardwalk Cuboid to SQL Table 11](#_Toc391491521)

[3.4.1 Configure Cuboid to SQL Table Super Merge Rule 13](#_Toc391491522)

[3.5 Boardwalk Cuboid to Boardwalk Cuboid 14](#_Toc391491519)

[3.5.1 Configure Cuboid to Cuboid Super Merge Rule 16](#_Toc391491520)

[3.6 Multiple Boardwalk Cuboids to One Boardwalk Cuboid 19](#_Toc391491517)

[3.6.1 Configure Multiple Cuboid to One Cuboid Super Merge Rule 20](#_Toc391491518)

GENERAL INFORMATION

This section explains in general terms, overview and the purpose for which Super Merge Process is intended.

Overview

In most of the applications we require data from external system to be pulled in Boardwalk. This data can be master or transactional data related to customer’s business and can be pulled from Data Feeds or Direct Integration.

To transmit this data from these external entities to Boardwalk Cuboids, there are 2 steps

1. Get Data in the SQL Staging Table 🡺 This can be achieved using various ways like SSIS Packages or Direct Integration
2. From Staging SQL Table pull data into Boardwalk Cuboids 🡺 This is achieved using Super Merge

The scope of this document is to cover point #2 from above.

Super Merge feature enables flow of data from

* SQL Table to BW Cuboid
* Multiple Cuboids to one Cuboid
* BW Cuboid to BW Cuboid &
* BW Cuboid to SQL Table

Purpose

The Super Merge is driven by a SQL procedures executing on the database. Data transfer takes place by setting Super Merge Rules for following methods.

* Merge huge dataset from SQL table into BW Cuboid
* Merge data from Boardwalk Cuboid to SQL Table
* Merge data from Boardwalk Cuboid to another Boardwalk Cuboid.
* Merge data from Multiple Cuboid (2 Cuboids) to One Cuboid

GETTING STARTED

Getting Started section presents briefly Pre-Requisite & Initial setup which is necessary to go ahead with execution of Super Merge.

## Pre – Requisites

To start with the Super Merge - below set up should be present:-

* The pre-configured Super Merge template which is having below Cuboids with Ids.

|  |  |  |
| --- | --- | --- |
| **Sr#** | **Boardwalk Cuboid Id** | **Boardwalk Cuboid Name** |
| 1 | 1000002 | BRectDefinition |
| 2 | 1000003 | KeyStore |
| 3 | 1000004 | S2C\_SuperMerge\_Rules |
| 4 | 1000005 | C2S\_SuperMerge\_Rules |
| 5 | 1000006 | C2C\_SuperMerge\_Rules |
| 6 | 1000007 | Multi\_C2C\_SuperMerge\_Rules |

The Super Merge rules defined for specific method needs to be inserted into these respective Boardwalk Cuboids.

* Super Merge Procedures

|  |  |  |
| --- | --- | --- |
| **Sr#** | **Super Merge Process** | **SQL Procedure Name** |
| 1 | SQL Table to Boardwalk Cuboid | BW\_SUPERMERGE\_SQL\_TO\_CUBOID |
| 2 | Boardwalk Cuboid to SQL Table | BW\_SUPERMERGE\_CUBOID\_TO\_SQL |
| 3 | Boardwalk Cuboid to Boardwalk Cuboid | BW\_SUPERMERGE\_CUBOID\_TO\_CUBOID |
| 4 | Multiple Boardwalk Cuboids(2) to One Boardwalk Cuboid | BW\_SUPERMERGE\_MULTI\_CUBOID\_TO\_CUBOID |

* BW\_SUPERMERGE\_MEASURE SQL table if Measures needs to be part of SQL Table to Boardwalk Cuboid Super Merge.
* SQL Staging tables in which data inserted from data source files.

## Create SQL Staging Table

This SQL table will be created as part of SSIS package/Integration and available once Integration implementation is complete; with some data from External system in this SQL table.

This Source SQL Staging Table Primary Key(s) should match with Target Boardwalk Cuboid.

SQL staging table is used as the source table in SQL table to BW Cuboid merge process.

PROCESSING MERGE

This section provides details about the Boardwalk Cuboids and SQL staging tables with column wise structure description.

And also defining the Super Merge rules and executing the procedure for specific type.

## BRectDefinition Cuboid (Boardwalk Rectangle Definition)

This Cuboid is the definition Cuboid i.e. it contains the definition of the Sources and Targets used for Super Merge activity along with the database name and table type respectively.

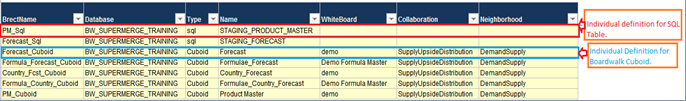
Unique BrectName is assigned to the individual definition which again used in merge rules.

The Cuboid Id of BRectDefinition is used while defining the merge rule as ‘BrectTableID’.

The columns description of BRectDefinition is as below

|  |  |  |
| --- | --- | --- |
| Sr# | Column Name | Description |
| 1 | BWID | System Generated Number assigned to the rule on link exporting Cuboid. |
| 2 | BrectName | Unique Name assign to the individual definition. This is used while defining the merge rule. |
| 3 | Database | Name of the database containing the Cuboid or SQL staging table. |
| 4 | Type | Cuboid or SQL : depends upon the type of table |
| 5 | Name | Name of the SQL table or Cuboid |
| 6 | WhiteBoard | * Column (6,7,8) Applicable for Cuboid only   Name of the WhiteBoard, Collaboration and Neighborhood in which the Cuboid is present.  These details can be obtained by logging in to the application. |
| 7 | Collaboration |
| 8 | Neighborhood |

The below image will give the brief idea about the BRectDefinition Cuboid.



## KeyStore Cuboid

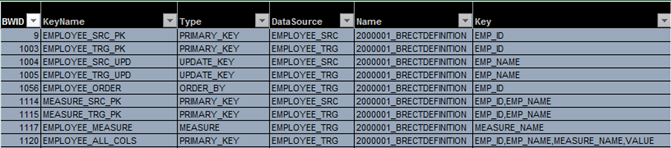
KeyStore Cuboid is used to define keys with respective Cuboid Ids and key columns.

Information present in KeyStore Cuboid is used in the rule definitions.

The columns description of KeyStore Cuboid is below

|  |  |  |
| --- | --- | --- |
| **Sr#** | **Column Name** | **Description** |
| 1 | KeyName | Name assigned to the Key definition |
| 2 | Type | The type of the key Eg. Primary Key, Update Key, Group By Key etc |
| 3 | DataSource | The BRect name to which the key belongs to |
| 4 | Name | BRectDefinition Cuboid information Eg. 1000002\_BRECTDEFINITION |
| 5 | Key | Comma separated Key columns |

The structure of KeyStore Cuboid is as shown below.



## SQL Table to Boardwalk Cuboid

S2C\_SuperMerge\_Rules Cuboid is used for defining the Super Merge rule whenever data needs to be merged from SQL Table to Boardwalk Cuboid.

The column wise description of Cuboid is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| Sr# | Column Name | Column Description | Possible Value  (If Applicable) |
| 1 | BWID | Unique Number assigned to the rule after submitted to the server |  |
| 2 | RuleName | Unique Name of Merge Rule for SQL Table to Boardwalk Cuboid Super Merge |  |
| 3 | AccessTableID | Cuboid Id of access control Cuboid use to restrict access for executing Super Merge Procedure.  *Note: Not in use* |  |
| 4 | BrectTableID | Cuboid Id of ‘BRectDefinition’ Cuboid.  *Note: Please refer 3.1 for details.* |  |
| 5 | SourceBrect | Name of the definition assigned to the Source table in BRectDefinition Cuboid. Source Table is, from which data to be transferred (for e.g. Staging table) |  |
| 6 | TargetBrect | Name of the definition assigned to the Target table in BRectDefinition Cuboid. Target Table to which data gets transferred (Cuboid) |  |
| 7 | KeystoreTableID | Cuboid id of KeyStore Cuboid. Key columns of source and target Brects are specified in this Cuboid.  *Note: Please refer 3.2 for details.* |  |
| 8 | SourcePrimaryKey | Key name specified in KeyStore Cuboid for the Primary Key columns of source SQL table. |  |
| 9 | TargetPrimaryKey | Key name specified in KeyStore Cuboid for the Primary Key columns of target Cuboid. |  |
| 8 | SourceUpdateKey | Key name specified in KeyStore Cuboid for the columns from source which are used as source for updating data in target. |  |
| 9 | TargetUpdateKey | Key name specified in KeyStore Cuboid for the columns of target Cuboid which are need to be updated from source. |  |
| 10 | InsertFlag | Need to set ‘1’ during insertion of new rows into the target table. else set to ‘0’. | * 1 * 0 |
| 10 | OrderByKey | Key name specified in KeyStore Cuboid for the columns by which data need to order or sequenced.  *Note: In case of Measures, this column is mandatory.* |  |
| 11 | AggregationFlag | Set to ‘1’ if aggregation on specific column required.  Else set to ‘0’. | * 1 * 0 |
| 12 | AggregationOperator | Operator used for aggregation operation.  SUM, AVG, COUNT, MAX, MIN can be used. |  |
| 13 | AggregationKey | Key name specified in KeyStore Cuboid for the columns from source table on which aggregation to be performed. |  |
| 14 | GroupByKey | Key name specified in KeyStore Cuboid for the column from Source on which grouping of data need to be done. |  |
| 15 | UpdateFlag | Need to set ‘1’ for updating cells in target table from source table Else set to ‘0’. | * 1 * 0 |
| 16 | DeleteFlag | When Delete flag set to ‘1’, this will delete rows from target table which are not present in source table.  If we do not need to delete earlier added rows and only keep on appending new rows in the Target then set this field as ‘0’ | * 1 * 0 |
| 17 | TransposeFlag | It can be set to ‘1’ or ‘0’ depending on the data structure in source table. (Refer - Transpose Flag Details) | * 1 * 0 |
| 18 | TransposeKey | Key name specified in KeyStore Cuboid for the columns which needs to be transposed in target. |  |
| 19 | MeasuresFlag | Set to ‘1’ if separate measure table is used.  Else set to ‘0’. | * 1 * 0 |
| 20 | TargetMeasureKey | Key name specified in KeyStore Cuboid for the Measure column used in target Cuboid. |  |
| 21 | FormulaFlag | Need to set ‘1’ if external Formula Cuboid is used.  Else set to ‘0’. | * 1 * 0 |
| 22 | FormulaBrect | Name of the definition assigned to the external Formula Cuboid in BRectDefinition Cuboid. |  |
| 23 | FormulaPrimaryKey | Key name specified in KeyStore Cuboid for the Primary Key columns of Formula Cuboid. |  |

Table 1: SQL Table to Cuboid Rule Column Description.

**Transpose Flag**

This flag is set depends upon the source file data structure. The image given below will give clear idea on this.

TRANSPOSE: This flag is set as per the data structure of source file as shown below

* If the Source Table matches with the TYPE-A structure then, SET Transpose as 0
* If the Source Table matches with the TYPE-B structure then, SET Transpose as 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PK Column Name 1 | PK Column Name 3 | PK Column Name 3 | Col name | Value |
| PK Value 1 | PK Value 2 | PK Value 3 | Cuboid Column Name 1 | value 1 |
| PK Value 1 | PK Value 2 | PK Value 3 | Cuboid Column Name 2 | value 2 |
| PK Value 1 | PK Value 2 | PK Value 3 | Cuboid Column Name n | value n |

Table 2 - TYPE A == Transpose Flag as 0

For Instance:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| D-Code | Geo Location | Country | Measure | Month Yr | Value |
| D1978 | EMEA | SWEDEN | Non-Revenue Consumption | Apr-13 | 100 |
| D1978 | EMEA | SWEDEN | Non-Revenue Consumption | May-13 | 100 |
| D1978 | EMEA | SWEDEN | Non-Revenue Consumption | Jun-13 | 100 |
| D1978 | EMEA | SWEDEN | Non-Revenue Consumption | Jul-13 | 100 |
| D1978 | EMEA | SWEDEN | Non-Revenue Consumption | Aug-13 | 100 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PK Column Name 1 | PK Column Name 3 | PK Column Name n | Cuboid Column Name1 | Cuboid Column Name2 |
| PK Value 1 | PK Value 2 | PK Value 3 | value | value |
| PK Value 1 | PK Value 2 | PK Value 3 | value | value |
| PK Value 1 | PK Value 2 | PK Value 3 | value | value |

Table 3 - TYPE B == Transpose Flag as 1

For Instance:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| D-Code | Geo Location | Country | B-Code | B-Code Description |
| D1978 | EMEA | SWEDEN | B9001 | Theken Spine |
| D1978 | EMEA | SWEDEN | B9001 | Theken Spine |
| D1978 | EMEA | SWEDEN | B9001 | Theken Spine |

Figure 1 - Transpose Flag Details

### Configure SQL Table to Cuboid Super Merge Rule

Super Merge Rule for SQL table to Boardwalk Cuboid can be created using (Table1 above) the column wise description.

Following details will clarify rule definition more precisely.

**Steps to perform Super Merge Operation**

**Step1:-** Add definition for source SQL table and target Boardwalk Cuboid in BRectDefintion Cuboid and submit the newly added data. These definition names are used in Super Merge rule.

**Step2:-** Set the KeyStore Cuboid for Source table and Target Cuboid Key columns definition.

**Step3:-** Set the proper Supermerge rule. Set BRectDefinition Cuboid Id and respective BrectNames for source and target while defining the Supermerge rule. Also use the KeyStore Cuboid Id and Key details in the Super Merge Rule.

* *Proper Rule definition is the important step of Super Merge process for successful execution. Please refer*

*Column-wise description for defining the other columns of Super Merge rule.*

**Step4:-** Confirm the flag setting for Insert, Update and Delete flag.

*Insert\_Flag:* Set to ‘1’ for inserting new rows in to Cuboid else it will be ‘0’.

Data present in SQL table but not in Cuboid will get added by setting this flag to ‘1’.

*Update\_Flag:* Set to ‘1’ for updating existing record of Cuboid else it will be ‘0’

Data present in SQL table and in Cuboid will be updated by setting this flag to ‘1’.

*Delete\_Flag:* Set to ‘1’ to delete rows from the Cuboid else it will be ‘0’.

Data present in Cuboid but not present in SQL table will be deleted by setting this flag to ‘1’.

**Step5:-** Execute the Super Merge procedure on database server.

**EXEC BW\_SUPERMERGE\_SQL\_TO\_CUBOID @RULE\_TBL\_ID, @RULE\_NAME, @USER\_NAME**

There are three parameter need to pass as shown above, there significance is as given below.

@RULE\_TBL\_ID 🡪 It is the Cuboid Id of the Super Merge Rule Cuboid

@RULE\_NAME 🡪 Super merge Rule Name defined in the Rule Cuboid S2C\_SuperMerge\_Rules

@USER\_NAME 🡪 Integration level User Name which exist in BW\_USER table.

Example:

*EXEC BW\_SUPERMERGE\_SQL\_TO\_CUBOID 1000004, 'Populate\_POD\_with\_FX', 'BOARDWALK\_INTEGRATION'*

## Boardwalk Cuboid to SQL Table

C2S\_SuperMerge\_Rules Cuboid is used for defining the Super Merge rule whenever data needs to be merged from Boardwalk Cuboid to SQL Table. This process is also known as Outbound process i.e. data is transferred from Boardwalk Cuboid to SQL Table for further processing.

The Column wise description for this Cuboid is as below.

|  |  |  |  |
| --- | --- | --- | --- |
| Sr# | Column Names | Description | Possible Value  (If Applicable) |
| 1 | BWID | System Generated Number assigned to the rule on link exporting Cuboid. |  |
| 2 | RuleName | Unique Name of Merge Rule for Cuboid to SQL merge. |  |
| 3 | AccessTableID | Cuboid ID of access control Cuboid used to restrict access for executing Super Merge Procedure.  *Note: Not in use* |  |
| 4 | BrectTableID | Cuboid Id of ‘BRectDefinition’ Cuboid.  *Note: Please refer 3.1 for details.* |  |
| 5 | SourceBrect | Name of the definition assign to the Source Cuboid in BRectDefinition Cuboid. Source Cuboid is, from which data to be transferred. |  |
| 6 | KeystoreTableID | Cuboid id of KeyStore Cuboid. Key columns of source and target Brects are specified in this Cuboid.  *Note: Please refer 3.2 for details.* |  |
| 7 | SourcePrimaryKey | Key name specified in KeyStore Cuboid for the Primary Key columns of source Cuboid. |  |
| 8 | TransposeKey | Key name specified in KeyStore Cuboid for the column names from source Cuboid which are transposed in target SQL Table. |  |
| 9 | GroupByKey | Key name specified in KeyStore Cuboid for the column from Source on which grouping of data need to be done. |  |
| 10 | AggregationFlag | Set to ‘1’ if aggregation on specific column required.  Else set to ‘0’. | * 1 * 0 |
| 11 | AggregationOperator | Operator used for aggregation operation.  SUM, AVG, COUNT, MAX, MIN can be used. |  |
| 12 | AggregationColumnType | Data type required of column used for aggregation. |  |
| 13 | AggregationKey | Key name specified in KeyStore Cuboid for the columns from source Cuboid on which aggregation to be performed. |  |
| 14 | TXIDFlag | Set to ‘1’ when using particular transaction id.  Set to ‘0’ for normal data transfer. | * 1 * 0 |
| 15 | TXID | Data update in the Cuboid after will this transaction id will be selected.  *Note: Please Refer Note Below.* |  |
| 16 | BWIDFlag | Set to ‘1’ if row ids also required in the target SQL Table.  Set to ‘0’ when row ids not required in final output. | * 1 * 0 |
| 17 | ColumnMapFlag | Set to ‘0’ when BW\_SUPERMERGE\_C2S\_SRC\_TRGT\_MAP SQL table not used for mapping.  Set to ‘1’ when BW\_SUPERMERGE\_C2S\_SRC\_TRGT\_MAP table is used for mapping target table in disaggregation process. | * 1 * 0 |
| 18 | ColumnMapSource | Column names of Source table need to be used in BW\_SUPERMERGE\_C2S\_SRC\_TRGT\_MAP as ‘SRC\_COLUMN’ |  |
| 19 | ColumnMapTarget | Column names of Target table need to be used in BW\_SUPERMERGE\_C2S\_SRC\_TRGT\_MAP as ‘TRG\_COLUMN’ |  |
| 20 | FilterString | Name of the Column Names (Separated by semicolon) and the value it contains used for filter the data.  Eg. (PRODUCT=|IPHONE|; MEASURE=|SALES|). |  |

Table2: Cuboid to SQL Table Merge Rule Column wise Description

* *Note: TX\_ID Transaction id is used whenever the data is required to apply on specific data*

*For e.g. data added after month April-2014 then that particular transaction id is specified in the rule for aggregation process.*

*Activities performed on Cuboid are maintained as a transaction in BW\_TXS SQL table which has timestamp of respective transaction.*

### Configure Boardwalk Cuboid to SQL Table Super Merge Rule

This is also known as outbound process in which data from source Cuboid is transferred into SQL table created.

Cuboid to SQL Super Merge can be used for 2 cases.

1. Normal Data transfer from Cuboid to SQL Table.

2. Data transfer by aggregating Cuboid data to SQL table.

SQL table column structure must be same as Cuboid columns. While defining the rule the sequence of columns mentioned in ‘SourcePrimaryKey’ as Primary Key column must be same as it is in SQL table design.

Super Merge rule can be defined with the help of Table 4 Column wise description. The details below shows rule defined.

**Steps to perform Super Merge operation**

**Step1:-** Add definitions for Source Boardwalk Cuboid and Target SQL table in BRectDefinition Cuboid as shown. These definitions are used in Super Merge rule.

**Step2:-** Set the KeyStore Cuboid for Source Cuboid and Target SQL Table Key columns definition.

**Step3:-** Set the proper Supermerge rule. Set BRectDefinition Cuboid Id and respective BrectNames for source and target while defining the Supermerge rule. Also use the KeyStore Cuboid Id and Key details in the Super Merge Rule.

**Step4:-** Execute the Super Merge procedure on database server.

**EXEC BW\_SUPERMERGE\_CUBOID\_TO\_SQL @RULE\_TBL\_ID, @RULE\_NAME, @USER\_NAME, @SQL\_TBL\_NAME**

There are four parameter need to pass as shown above, there significance is as given below.

@RULE\_TBL\_ID 🡪 It is the Cuboid Id of the Super Merge Rule Cuboid

@RULE\_NAME 🡪 Super merge Rule Name defined in the Rule Cuboid

@USER\_NAME 🡪 Integration level User Name which exist in BW\_USER table.

@SQL\_TBL\_NAME 🡪 Name of SQL Target table in which data need to be merged.

Example:

*EXEC BW\_SUPERMERGE\_CUBOID\_TO\_SQL 1000005, 'RulePOD', 'BOARDWALK\_INTEGRATION', 'Process\_Order\_Data'*

## Boardwalk Cuboid to Boardwalk Cuboid

C2C\_SuperMerge\_Rules Cuboid is used for defining the Super Merge rule whenever data needs to be merged from one Boardwalk Cuboid to another Boardwalk Cuboid.

The column wise description of Cuboid is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| Sr # | Column Name | Description | Possible Value  (If Applicable) |
| 1 | BWID | System Generated Number assigned to the rule on link exporting Cuboid. |  |
| 2 | RuleName | Unique Name of Merge Rule for Cuboid to Cuboid Super Merge. |  |
| 3 | AccessTableID | Cuboid Id of access control Cuboid use to restrict access for executing Super Merge Procedure.  *Note: Not in use* |  |
| 4 | BrectTableID | Cuboid Id of ‘BRectDefinition’ Cuboid.  *Note: Please refer 3.1 for details.* |  |
| 5 | SourceBrect | Name of the definition assigned to the Source Cuboid in BRectDefinition Cuboid. Source Cuboid is, from which data to be transferred. |  |
| 6 | KeystoreTableIDSource | Cuboid id of KeyStore Cuboid. Key columns of Source Brect is specified in this Cuboid.  *Note: Please refer 3.2 for details.* |  |
| 7 | SourcePrimaryKey | Key name specified in KeyStore Cuboid for the Primary Key columns of Source Cuboid. |  |
| 8 | TargetBrect | Name of the definition assigned to the Target Cuboid in BRectDefinition Cuboid. Target Cuboid is, to which data to be transferred. |  |
| 9 | TargetSampleBrect |  |  |
| 10 | KeystoreTableIDTarget | Cuboid Id of KeyStore Cuboid. Key columns of Target Brect is specified in this Cuboid.  *Note: Please refer 3.2 for details.* |  |
| 11 | TargetPrimaryKey | Key name specified in KeyStore Cuboid for the Primary Key columns of Target Cuboid. |  |
| 12 | KeystoreTableIDUpdateKey | Cuboid Id of KeyStore Cuboid. Update columns of Source and Target Brects are specified in this Cuboid. |  |
| 13 | SourceUpdateKey | Key name specified in KeyStore Cuboid for the columns from Source from which data needs to be updated except Primary Keys. |  |
| 14 | TargetUpdateKey | Key name specified in KeyStore Cuboid for the columns from Target to which data needs to be updated except Primary Keys. |  |
| 15 | IncludeTagName |  |  |
| 16 | AggregationEachSliceFlag |  |  |
| 17 | AggregationFunctionBySlice |  |  |
| 18 | AggregationSliceFlag | Set to ‘1’ if aggregation on specific column required.  Else set to ‘0’. | * 1 * 0 |
| 19 | AggregationFunction | Aggregate function use to aggregate data for.  SUM, AVG, COUNT, MAX, MIN can be used. |  |
| 20 | KeystoreTableIDGroupBy | Cuboid Id of KeyStore Cuboid. Definition for Key column names used for grouping data are specified in this Cuboid. |  |
| 21 | GroupByKey | Key name defined in KeyStore Cuboid used for grouping of data. |  |
| 22 | InsertFlag | Need to set ‘1’ during insertion of new rows into the target table. Else set to ‘0’. | * 1 * 0 |
| 23 | DeleteFlag | When Delete flag set to ‘1’, this will delete rows from target Cuboid which are not present in source Cuboid. If we do not need to delete earlier added rows and only keep on appending new rows in the Target then set this field as ‘0’. | * 1 * 0 |
| 24 | UpdateFlag | Need to set ‘1’ for updating cells in target Cuboid from source Cuboid.  Else set to ‘0’. | * 1 * 0 |
| 25 | BlankTargetFlag |  |  |
| 26 | StopOnNonFloat | Set to ‘1’ to stop the processing of merge if column used for aggregation contains non numeric value else set to ‘0’. Used in aggregation process. | * 1 * 0 |
| 27 | KeystoreTableIDOrderBy | KeyStore Cuboid id with definition for Key column names used for ordering (sequencing) data. |  |
| 28 | OrderByKey | Key name defined in KeyStore Cuboid; used for sequencing data. |  |
| 29 | TransposeTimeColumnTarget | It can be set to ‘1’ or ‘0’ depending on the data structure in source table. | * 1 * 0 |
| 30 | TransposeColumnSource | Column Name from source Cuboid need to be used for transposing data. |  |
| 31 | MapTableID | Cuboid id of mapping table. |  |
| 32 | MapSource | Key name defined in KeyStore Cuboid from Source used in mapping. |  |
| 33 | MapTarget | Key name defined in KeyStore Cuboid from Target used in mapping. |  |

Table3: Cuboid to Cuboid Rule Column wise Description

### Configure Boardwalk Cuboid to Boardwalk Cuboid Super Merge Rule

Super Merge Rule for one Boardwalk Cuboid to another Boardwalk Cuboid can be created using (Table3) the column wise description.

Following details will clarify rule definition more precisely.

**Steps to perform merge operation**

**Step1:-** Add definition for Source and Target Boardwalk Cuboids in BRectDefintion Cuboid and submit the newly added data. These definition names are used in Super Merge rule.

**Step2:-** Set the KeyStore Cuboid for Source and Target Cuboids Key columns definition.

**Step3:-** Set the proper Supermerge rule. Set BRectDefinition Cuboid Id and respective BrectNames for source and target while defining the Supermerge rule. Also use the KeyStore Cuboid Id and Key details in the Super Merge Rule.

**Step4:-** Confirm the flag setting for Insert, Update and Delete flag.

*Insert Flag:* Set to ‘1’ for inserting new rows in to target Cuboid else it will be ‘0’.

If new rows exist in source table and need to insert in target table based on primary keys, Insert flag is need to set to ‘1’.

*Update Flag:* Set to ‘1’ for updating existing record of target Cuboid else it will be ‘0’

If source table data updated then accordingly target table data needs to update w.r.t primary keys then set the update flag to ‘1’.

*Delete Flag:* Set to ‘1’ to delete rows from the target Cuboid else it will be ‘0’.

If rows form source table deleted and also need to delete from target table on the basis of primary key columns then need to set delete flag as ‘1’.

* *For other flag settings please refer column wise description table.*

**Step5:-** Execute the Super Merge procedure on database server.

**EXEC BW\_SUPERMERGE\_CUBOID\_TO\_CUBOID @RULE\_CUBOID, @RULE\_NAME, @SRC\_COLUMNS, @TGT\_COLUMNS, @MEMBER\_ID, @DEBUG\_FLAG, @TABLE\_NAME, @TABLE\_PURPOSE, @INCLUDE\_USER\_ID\_TGT, @TBL\_ID\_SRC\_FILTER, @TBL\_ID\_TRGT\_FILTER, @VIEW,@USER\_NAME**

There are thirteen parameters need to pass as shown above, there significance is as given below.

@RULE\_CUBOID 🡪 It is the Cuboid id of the Super merge Rule table.

@RULE\_NAME 🡪 Super merge Rule Name defined in the Rule Cuboid ‘C2C\_SuperMerge\_Rules’

@SRC\_COLUMNS 🡪 Blank or Source PK Columns defined in KeyStore Cuboid for PK reference.

Significance: Source Primary Key columns used for reporting.

@TGT\_COLUMNS 🡪 Blank or Target PK Columns defined in KeyStore Cuboid for PK reference.

Significance: Target Primary Key columns used for reporting.

@MEMBER\_ID 🡪 Member Id of respective user obtains from BW\_MEMBER table.

Significance: Each user is member of Neighborhood that Membership Id.

@DEBUB\_FLAG 🡪 Value ‘0’ or ‘1’

Set to ‘1’ to print debug messages during procedure execution.

Significance: Debug messages will be helpful while troubleshooting.

@TABLE\_NAME 🡪 Blank for Super Merge rule

Significance: Use for reporting.

@TABLE\_PURPOSE 🡪 Blank for Super Merge rule

Significance: Use for reporting.

@INCLUDE\_USER\_ID\_TGT 🡪 -1

@TBL\_ID\_SRC\_FILTER 🡪 -1

@TBL\_ID\_TRGT\_FILTER 🡪 -1

@VIEW 🡪 Blank value

@USER\_NAME 🡪 Integration level User Name which exist in BW\_USER table.

Example:

*EXEC BW\_SUPERMERGE\_CUBOID\_TO\_CUBOID 1000006, 'POD\_from\_PO', '', '', 2, 1, '', '', -1, -1, -1, '', 'BOARDWALK\_INTEGRATION'*

**Slicing Concept of one Boardwalk Cuboid to another Boardwalk Cuboid:**

There are 2 parameters named as @SRC\_SLICES and @TRG\_SLICES are of type ‘History\_Tag\_List’ which is a table type datatype. History\_Tag\_List has the below columns.

NAME 🡪 Name assigned for the set

CONDITION 🡪 Assign number as per slicing condition.

KEYNAME 🡪 Name of the column containing Measures

VALUE 🡪 Specific Measure Name on the basis of which data filtered.

TX\_ID 🡪 Default set to ‘-1’ or transaction id generated for slicing activity.

PK\_KEY 🡪 Primary Key Columns of Cuboid (Source or Target).

Optional Values can be inserted for slicing for

Case 1:- 1:1 matching condition i.e. one measure form source matching with one of target as

*INSERT INTO @SRC\_SLICES*

*SELECT 'CURRENT', 1, 'Measure', 'Total Manufacturing Cost', -1, 0*

*INSERT INTO @SRC\_SLICES*

*SELECT 'CURRENT', 2, 'Measure', 'Base Price', -1, 0*

*INSERT INTO @TGT\_SLICE*

*SELECT 'CURRENT', 1, 'Measure', 'Manufacturing Cost', -1, 0*

*INSERT INTO @TGT\_SLICE*

*SELECT 'CURRENT', 2, 'Measure', 'Base price', -1, 0*

Case 2:- Slicing for one source measure and two target measures.

*INSERT INTO @SRC\_SLICES*

*SELECT 'CURRENT', 1, 'FT/PT', 'FT', -1, 0*

*INSERT INTO @TGT\_SLICE*

*SELECT 'CURRENT', 1, 'Measure', 'Changes to FT FTE', -1, 0*

*INSERT INTO @TGT\_SLICE*

*SELECT 'CURRENT', 1, 'Account Sub Category', 'FULL TIME (FTE)', -1, 0*

## Multiple Boardwalk Cuboids to One Boardwalk Cuboid

Multi\_C2C\_SuperMerge\_Rules Cuboid is used for defining the Super Merge rule whenever data needs to be merged from 2 Boardwalk Cuboids into a single Boardwalk Cuboid.

It will form a structure in target Cuboid such that it will be ‘Cartesian product’ of source1 and source2.

The column wise description of Cuboid is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| Sr# | Column Names | Description | Possible Value  (If Applicable) |
| 1 | BWID | System Generated Number assigned to the rule on link exporting Cuboid. |  |
| 2 | RuleName | Unique Name of Merge Rule for multiple Cuboid to one Cuboid Super Merge. |  |
| 3 | BrectTableID | Cuboid Id of ‘BRectDefinition’ Cuboid.  *Note: Please refer 3.1 for details.* |  |
| 4 | SourceBrect1 | Name of the definition assigned to the Source Cuboid 1 in BRectDefinition Cuboid. Source Cuboid 1 is first Cuboid from which data to be transferred. |  |
| 5 | SourceBrect2 | Name of the definition assign to the Source Cuboid 2 in BRectDefinition Cuboid. Source Cuboid 2 is Second Cuboid from which data to be transferred. |  |
| 6 | TargetBrect | Name of the definition assign to the Target Cuboid in BRectDefinition Cuboid. Target Table is into which data gets transferred. |  |
| 7 | KeystoreTableID | Cuboid id of KeyStore Cuboid. Key columns of source and target Brects are specified in this Cuboid.  *Note: Please refer 3.2 for details.* |  |
| 8 | SourcePrimaryKey1 | Name of the source table1 primary key columns (Comma Separated). Order of the column names must be same as in TrgtPrimaryKeys. |  |
| 9 | SourcePrimaryKey2 | Name of the source table2 primary key columns (Comma Separated). Order of the column names must be same as in TrgtPrimaryKeys. |  |
| 10 | TargetPrimaryKey | Name of the target table primary key columns (Comma Separated). Order of the column names must be same as Source table1 and Source table2 columns. |  |
| 11 | SourceMatchKey1 | \*Not in use for current Version specify as ‘NULL’ |  |
| 12 | SourceMatchKey2 | \*Not in use for current Version specify as ‘NULL’ |  |
| 13 | SourceUpdateKey1 | Columns of source1 from which data is used for update. |  |
| 14 | SourceUpdateKey2 | Columns of source2 from which data is used for update. |  |
| 15 | TargetUpdateKey | Columns of target used for update. |  |
| 16 | InsertFlag | Need to set ‘1’ during insertion of new rows into the target table.  Else set to ‘0’. | * 1 * 0 |
| 17 | UpdateFlag | Need to set ‘1’ for updating cells in target table from source table1 and source table2 otherwise ‘0’ is set. | * 1 * 0 |
| 18 | DeleteFlag | When Delete flag set to ‘1’, this will delete rows from target table which are not present in source table1 and source table2. If we do not need to delete earlier added rows and only keep on appending new rows in the Target then set this field as ‘0’. | * 1 * 0 |

Table 4: Column Description for Multiple Cuboids to One Cuboid Rule

### Configure Multiple Boardwalk Cuboids to One Boardwalk Cuboid Super Merge Rule

Column wise description (Table 4) can be helpful to define Super Merge rule for Multiple Cuboid to one Cuboid Super Merge.

Source Cuboid primary key columns and target Cuboid primary key columns must be in same order and sequence.

**Steps to perform Super Merge Operation**

**Step1:-** Add definition for two source Boardwalk Cuboids and one target Boardwalk Cuboid in BRectDefintion Cuboid and submit the newly added data. These definition names are used in Super Merge rule.

**Step2:-** Set the KeyStore Cuboid for Source Cuboids and Target Cuboid Key columns definition.

**Step3:-** Set the proper Supermerge rule. Set BRectDefinition Cuboid Id and respective BrectNames for sources and target while defining the Supermerge rule. Also use the KeyStore Cuboid Id and Key details in the Super Merge Rule.

**Step4:-** Confirm the flag setting for Insert, Update and Delete flag.

*Insert Flag:* Set to ‘1’ for inserting new rows in to target Cuboid else it will be ‘0’.

If rows added in src1 or src2 Cuboid and need to insert rows in target Cuboid Insert Flag is set to ‘1’ which will add number of rows in target equal to Cartesian product of newly added rows in src1 and src2

*Update Flag:* Set to ‘1’ for updating existing record of target Cuboid else it will be ‘0’

If rows from src1 or src2 are updated and need to update target table then Update flag will be set to ‘1’

*Delete Flag:* Set to ‘1’ to delete rows from the target Cuboid else it will be ‘0’.

If any rows are deleted from src1 or src2 and need to delete rows from target table Delete flag set to ‘1’

**Step5:-** Execute the Super Merge procedure on database server.

**EXEC BW\_SUPERMERGE\_MULTI\_CUBOID\_TO\_CUBOID @BW\_RULE\_TBL\_ID, @BW\_RULE\_NAME, @DEBUB\_FLAG, @USER\_ID, @FORMULA\_TBL\_ID, @PK\_FORMULA**

There are six parameter need to pass as shown above, there significance is as given below.

@BW\_RULE\_TBL\_ID 🡪 It is the Cuboid Id of the Super Merge Rule Cuboid

@BW\_RULE\_NAME 🡪 Super merge Rule Name defined in the Rule Cuboid ‘Multi\_C2C\_SuperMerge\_Rules’

@DEBUB\_FLAG 🡪 Value ‘0’ or ‘1’

Set to ‘1’ to print debug messages during procedure execution.

Significance: Debug messages will be helpful while troubleshooting.

@USER\_ID 🡪 Integration level User Id which exist in BW\_USER table.

@FORMULA\_TBL\_ID 🡪 Formula Cuboid Id if formulas need to be applied.

@PK\_FORMULA 🡪 The Primary Key Columns based on which the formulas need to be applied

Example:

*EXEC BW\_SUPERMERGE\_MULTI\_CUBOID\_TO\_CUBOID 1000007, 'KPI\_metrix', 1, 3, -1, ''*