ACOD-BI

Automatic Creation of Optimized Datamarts

User manual

Table of Contents

1 Introduction

1.1 Tool presentation

Tool nature

ACOD-BI automatically generates optimized datamarts in a limited space storage.
It identifies optimum aggregates from reporting statistics.

In its current version, ACOD-BI allows you to generate a datamart in a few hours instead of a few weeks for manual creation.

The generated datamart is provided in the form of two readable and editable SQL scripts. Once the datamart is generated you are no longer dependent on the tool.

Access to ACOD-BI

ACOD-BI is currently available for free in beta.
To create your datamart please contact contact@acod-bi.com.

Input data

ACOD-BI defines the target datamart structure mainly from the following data:

- Logical data model,
- Data statistics,
- Statistics on data utilization in reporting,
- Space storage granted to the datamart.

Structure of datamarts created by ACOD-BI

ACOD-BI generates star schema datamarts.
It adds the most pertinent aggregates to the detail fact tables.

For every detail fact table, a unique view gathers indicators[1] and attributes[2] from the fact table and its associated dimensions.
The user builds queries on these views without worrying about how the dimension tables are modeled and without being impacted in case of datamart model evolution.

The system creates the additional required structures to allow the Oracle QUERY_REWRITE mechanism to automatically use aggregates.

1.2 Document objective

This document explains the procedure to follow to:

1. Give to the tool the needed data,
2. Generate the datamart creation script and load script.

[1] An indicator is a data on which reporting applies an aggregation function (SUM, MIN and MAX).
[2] An attribute is a data used by reporting in filters or in group by clause.

2 Prerequisites

The source[3] data model is close to the third normal form.
The closer the modeling is to the 3rd normal form the better the tool knows the source logical data model and the more the generated datamart is relevant.

The source statistics are up to date.

The system administrator is able to provide a dump with statistics without data.

The system administrator is able to provide an estimation of the frequency of use in reporting for the most used data.
The more this estimation is complete and accurate the more the generated datamart is relevant.
These statistics can be difficult to estimate, especially for a front scratch project. If so refer to the chapter "Progressive approach".

The source is contained in a single schema.
If this is a blocking point an evolution of the tool is possible.

The datamart schema has a database link to the source.
If this point cannot be respected manual modifications of the datamart load script will be necessary.

The foreign keys making loops are not taken into account.
If this is a blocking point an evolution of the tool is possible.

The only DBMS supported for the time being is Oracle.
Porting on other DBMS is feasible if needed.
The tool has been tested with datamarts on the 12.2.0.1.0 version but should work on previous versions.

[3] In the most standard cases the source is a data warehouse.
Loading data directly from the operational system can be done if the following conditions are satisfied:

- The operational system data model is close to the third normal form,
- There is a long enough time window during which the operational system is unused to load the datamart.

3 Steps overview

Exchanges between the administrator and the system are done via CSV files[4].
For more details refer to the chapter « CSV files ».

Step 1 (Administrator)
The system administrator provides a dump of the source with statistics without data.
The system administrator provides various global settings. Some are optional others are mandatory.

Step 2 (ACOD-BI)
The source dump is loaded in one of the ACOD-BI environments.
ACOD-BI generates 2 CSV files:

- A file (named DField file) that allows the system administrator to select data corresponding to attributes and/or indicators.
- A file (named Entity file) that lists the data model entities.

Step 3 (Administrator)
The system administrator indicates in the DField file the source columns corresponding to attributes and/or to indicators.

Step 4 (ACOD-BI)
ACOD-BI loads the DField file and generate a CSV file (named Use case file) used by the system administrator to edit statistics about attribute usages in reporting.

Step 5 (Administrator)
The system administrator indicates in the Use case file the usage frequency of attributes in reporting.
The system administrator indicates in the Entity file the relative weight of each indicator group in reporting.

Step 6 (ACOD-BI)
ACOD-BI loads the Use case file and the Entity file.
It generates the best possible datamart based on information received (source logical model, data statistics, reporting statistics and space allocated to the datamart).
The creation script and load script of the datamart are available to the system administrator.

Step 7 (Administrator)
The system administrator integrates the datamart in his system.

[4] ACOD-BI generates and loads all CSV files at each step.
In order to ease the understanding, the chapter « Steps overview » describes only the necessary generations and loadings of each step.

4 *Progressive approach*

When statistics on data utilization in reporting are difficult to estimate it is possible to use ACOD-BI in two steps.

4.1 Step 1 – ACOD-BI runs without reporting statistics

ACOD-BI runs without statistics on data utilization in reporting and without pre-calculated aggregates.
Reporting is developed normally, using views provided by ACOD-BI.

4.2 Step 2 – ACOD-BI runs with reporting statistics

The system administrator deduce statistics on data utilization from existing reporting (with the current version the action is manual).

ACOD-BI runs with reporting statistics and with a target datamart disk space.
The tool can generate aggregates.

The datamart is rebuild and re loaded.

Reporting development is not impacted.
Reporting performance are improved.

5 Concepts handled during edition

In addition to the concepts of attribute and indicator mentioned above, three other concepts are handled in CSV files.

Entities
Entities correspond to the source tables[5].

DFields (for datamart fields)
DFields correspond to the selected attributes and indicators.
For a source column used both as an attribute and as an indicator, two DFields are created.

Hierarchy links
Hierarchy links correspond to the source foreign keys.

6 CSV files

Exchanges between the administrator and ACOD-BI are done via CSV files.

In order to simplify administrator input, ACOD-BI generates pre-populated CSV files.

ACOD-BI generates and loads all CSV files at each step.

In case of successive iterations, ACOD-BI re generates csv files integrating the entries already made by the administrator in the previous iteration.

The format of numbers is without thousand separator and with decimal separator "."

CSV files can be edited, for example in Excel or in OpenOffice Calc.

The system administrator must not create new lines and must only modify the columns whose title are prefixed with "NEW".

CSV files are transferred between ACOD-BI and the system administrator by FTP-S.

CSV files are described in the appendix A « Structure of CSV files ».

[5] There are rare exceptions: if, for a foreign key of the source, the columns of the key in the child table form a mandatory unique key then both tables share the same entity in ACOD-BI.

7 Details of steps

7.1 Step 1 (Administrator) - Initialization

7.1.1 (Mandatory) Source DUMP with statistics and without data

The command to get the dump is:

exp <username>/<password>@<sid> OWNER=<username> FILE=<username>.dmp compress=N LOG=exp_<username>.log rows=N

<username> is the source owner.

7.1.2 (Mandatory) Entering global settings

The system administrator enters in the Global settings file the settings specific to his system.

7.1.2.1 (Mandatory) Code identifying the source

The system administrator indicates a code that identifies the source (parameter CODE_DWH).
This code makes it easier to manage different environments.

7.1.2.2 (Mandatory) Datamart database link on the source

The system administrator enters the database link name which allows the datamart to read the source (parameter DBL_DTM2DWH).

7.1.2.3 (Mandatory) Datamart tablespace names

The system administrator enters the datamart tablespace names:

- The data fact table tablespace
  (Parameter TBS_DTM_FACT_DATA),
- The index fact table tablespace
  (Parameter TBS_DTM_FACT_INDEX),
- The data reference table tablespace
  (Parameter TBS_DTM_REF_DATA),
- The index reference table tablespace
  (Parameter TBS_DTM_REF_INDEX).

The datamart creation script must be manually modified if the proposed tablespace splitting is not sufficient.

7.1.2.4 (Mandatory) Target datamart disk space

The system administrator must indicate what disk space he would like the datamart to use.
It can be done with the parameter EXPLICIT_DTM_ADS which indicates in bytes, the desired target in terms of space used by the datamart in the tablespaces.

7.1.2.5 (Optional) Determination function for entity code model

The entity code model[6] serves as a base for ACOD-BI to construct the names of tables and associated objects in the datamart.
By default, the entity code model is the name of the corresponding table in the source.
In some cases it may be worthwhile to transform the name of the table, for example to remove a prefix or to transcode meaningless table names into more understandable names.
This is done by the determination function for entity code model.

The function name is indicated by the parameter FUNC_NTAB_CODE2ENTI_CODE.

The function must be defined in the source and its signature must be:

```
<function_name>(table_name IN VARCHAR2) RETURN VARCHAR2
```

7.1.2.6 (Optional) Determination function for DField code model

The DField code model[7] serves as a base for ACOD-BI to construct the names of datamart columns as well as the names of some tables.
The source column names can be used as DField code model but in some cases it may be worthwhile to transform the name of the column, for example to remove a prefix or to transcode meaningless column names into more understandable names.
This is done by the determination function for DField code model.

The function name is indicated by the parameter FUNC_NCOL_CODE2INFO_CODE.

The determination function can be an ACOD-BI predefined function or a function defined in the source.
For ACOD-BI predefined function the parameter value must be <function_name >@DBL_STARP.
The function signature must be:

```
< function_name>(entity_code IN VARCHAR2, table_name IN VARCJAR2, column_name IN VARCHAR2)
RETURN VARCHAR2
```

By default, if the column name in the source is prefixed by the entity code, the DField code model is the column name without prefix, otherwise the DField code model equals the column name.
The corresponding parameter value is NCOL_CODE2INFO_CODE_V1@STARP.

As of 07/03/2019 there is no other determination function for DField code model predefined in ACOD-BI.
Other predefined determination functions will be added along the way as needed.

[6] Refer to chapter "Datamart codes" to see the construction of a code from a code model.
[7] Refer to chapter "Datamart codes" to see the construction of a code from a code model.

7.1.2.7 (Optional) Determination function for hierarchy link differentiating code model

When, in the source, two foreign keys have the same child table and the same parent table the system is likely to create multiple columns in the datamart views for a single column of the source parent table.
It is better to differentiate columns by meaningful codes rather than by numbers.
With some naming standards, these meaningful codes can be extracted from foreign key names.
It is done by the determination function for hierarchy link differentiating code model[8].

The function name is indicated by the parameter FUNC_NFK_CODE2HILI_DIFF_CODE.

The determination function can be an ACOD-BI predefined function or a function defined in the source.
For ACOD-BI predefined function the parameter value must be <function_name>@DBL_STARP.
The function signature must be:

```
<function_name>(fk_name IN VARCHAR2) RETURN VARCHAR2
```

By default, to determine the differentiating code model of a foreign key, ACOD-BI checks if the name of the foreign key is in the form of:

```
[[FK_]<child table>_]<parent table>_<complementary code>
```

If this is the case the differentiating code model is <complementary code>.
If this is not the case the differentiating code model is null.
The corresponding parameter value is NFK_CODE2HILI_DIFF_CODE_V1@STARP.

As of 07/03/2019 there is no other determination function for hierarchy link differentiating code model predefined in ACOD-BI. Other predefined determination functions will be added along the way as needed.

[8] Refer to chapter "Datamart codes" to see the construction of a code from a code model.

Example - NFK_CODE2HILI_DIFF_CODE_V1@STARP use

Source
The source contains the tables SALES, CUSTOMER and COUNTRY.

The table COUNTRY contains a COUNTRY_CODE (identifier) and a COUNTRY_LABEL.

The table CUSTOMER contains a COUNTRY_CODE_BIRTH and a COUNTRY_CODE_HOME with foreign keys on COUNTRY named FK_CUSTOMER_COUNTRY_BIRTH and FK_CUSTOMER_COUNTRY_HOME.

SALES is a fact table. It allows, among other things, to track the number and amount of products sold. It holds a foreign key on CUSTOMER table.

ACOD-BI
The tool gathers in a unique view all sales information.

This view contains (among others):
- One column for the client country label home,
- One column for the client country label birth.

It is better to differentiate these two columns by meaningful codes (BIRTH and HOME) rather than numbers.

The naming standard for foreign keys in the source is FK_<child table>_<parent table> [<complementary code>] (as assumed by default by the system).

The two differentiating codes are extracted from the name of the foreign keys.
Column names obtained in the sales view are:
- COUNTRY_LABEL_HOME
- COUNTRY_LABEL_BIRTH

7.2 Step 2 (ACOD-BI)

The source dump is loaded in one of the ACOD-BI environments.
ACOD-BI loads all present CSV files from input directory.
ACOD-BI generates all CSV files to output directory.

7.3 Step 3 (Administrator) – Attributes and indicators selection

7.3.1 (Mandatory) DField file

(Mandatory) The system administrator indicates in the DField file the source columns corresponding to attributes and/or to indicators[9].

(Optional) The system administrator may modify codification of attributes and indicators[10] [11].

7.3.2 (Optional) Entity file

(Optional) The system administrator may modify the datamart entities code model[12] [13].

The relative weight of each indicator group in reporting, in the same file, will be entered at step 5.

7.3.3 (Optional) Hierarchy link file

(Optional) The system administrator can modify hierarchy link differentiating code model[14] [15].

7.4 Step 4 (ACOD-BI)

ACOD-BI loads all present CSV files from input directory.
ACOD-BI generates all CSV files to output directory.

[9] Columns "NEW_ATTR_ACTIVATION" and "NEW_INDI_ACTIVATION" in the DField file.
[10] For more details refer to chapter « Datamart codes ».
[11] Columns "NEW_EXPLICIT_ATTR_CODE_MODEL" and "NEW_EXPLICIT_INDI_CODE_MODEL" in the DField file.
[12] For more details refer to chapter « Datamart codes ».
[13] Column "NEW_EXPLICIT_CODE_MODEL" in the Entity file.
[14] For more details refer to chapter « Datamart codes ».
[15] Column "NEW_EXPLICIT_DIFF_CODE_MODEL" in the Hierarchy link file.

7.5 Step 5 (Administrator) – Entering usage frequencies

7.5.1 (Mandatory) Entity file

(Mandatory) The system administrator indicates in the Entity file the relative weight of each indicator group in reporting [16] [17].

7.5.2 (Mandatory) Use case file

(Mandatory) The system administrator indicates in the Use case file the usage frequency of attributes and/or attribute branches[18] [19].

7.6 Step 6 (ACOD-BI)

ACOD-BI loads all present CSV files from input directory.
ACOD-BI generates all CSV files to output directory.

ACOD-BI generates the datamart creation script and the datamart load script and drops them into the output directory.

7.7 Step 7 (Administrator) – Datamart creation and loading

The system administrator downloads via FTP-S the datamart creation script and the datamart load script from the ACOD-BI output directory.
He uses these scripts to create and load the datamart in his own system[20].
For more details refer to « Appendix B – Datamart creation and loading».

[16] For more details refer to chapter « Indicator usage frequencies ».
[17] Column "NEW EXPLICIT INDICATOR(S) WEIGHT" in the Entity file.
[18] For more details refer to chapter « Attribute usage frequencies ».
[19] Columns "NEW_EXPLICIT_ATTRIBUT_UF" and "NEW_EXPLICIT_BRANCH_UF" in the Use case file.
[20] The generated scripts are readable and manually editable.
For example, the load script works in delete all/insert all mode. It can be modified to work in another mode, at least on the largest fact tables.

8 Reporting statistics

8.1 Indicator usage frequencies

ACOD-BI does not need to know the usage frequency of each indicator in reporting.
Each attribute and each indicator belongs to an entity. Entities having at least one indicator are called « indicator group ».

ACOD-BI only needs the relative weight of each indicator group in reporting.
The system administrator indicates the weight of each indicator group in the Entity file[21].

Example

Source
The source contains two fact tables, ORDER and INVOICE. Each of these fact tables contains an amount and a quantity.
Analyzed values are amounts and quantities.

Reporting
During a standard day the following reports are executed:

- 3 reports about invoice amount,
- 1 report about invoice quantity,
- 5 reports about invoice amount and invoice quantity,
- 4 reports about order amount and order quantity.

ACOD-BI
The repository contains:

- An entity (Invoice) with two indicators (invoice amount and invoice quantity).
  The invoice entity is an indicator group.
- An entity (Order) with two indicators (order amount and order quantity).
  The order entity is an indicator group.

Usage frequency
The usage frequency of the invoice indicator group is:
(3+1+5)/(3+1+5+4)
The usage frequency of the invoice indicator group is:
4/(3+1+5+4)

To transpose it, the administrator gives to invoice the relative weight of 9 (3+1+5) and to order the relative weight of 4.

[21] By default, the weight of a group of indicators is the number of indicators of the group. However, this usually does not correspond to reality.

8.2 Attribute usage frequencies

8.2.1 Attribute level entering

ACOD-BI needs an estimate of usage frequency for each attribute and each indicator group.

With the relative weight of the indicator group ACOD-BI already knows the usage frequency of each indicator group.

The system administrator can enter a reporting usage frequency for each pair <attribute, indicator group> for a proven use of the indicator group.

8.2.2 (Optional) Branch level entering

For complex models, entering all usage frequencies <attribute, indicator group> can be both laborious and difficult to estimate.

This is why ACOD-BI allows the system administrator to enter the reporting usage frequency of an attribute hierarchy branch for an indicator group.

When the system administrator enters a branch-level usage frequency, the attributes of the branch are impacted as follows:

- Attributes on which the administrator HAS entered usage frequency.
  ➔ No impact.
- Attributes on which the administrator HAS NOT entered usage frequency.
  ➔ The system automatically adjusts the usage frequencies with the following rules:
    - The combination of the usage frequency of all the attributes of the branch corresponds to the usage frequency of the branch.
    - Each attribute has a default usage frequency. The distribution of the branch usage frequency on attributes is weighted by their default usage frequency.

Example

Source

The source contains two fact tables, ORDER and INVOICE. Each of this fact table contains an amount and a quantity.
Analyzed values are amounts and quantities.
Each order is associated with a date, a product and a customer.
Each invoice is associated with a date, a product and a customer.
Each customer has a lot of attributes. The most used in the reporting are customer age group and customer region.
Each product has two attributes: name and color.

Reporting

During a standard day 130 reports are executed with the following repartition:

- 40 reports about order amount and/or quantity,
- 100 reports about invoice amount and/or quantity,
  (10 reports are about order AND invoice)

Among the 100 reports about invoice:

- 100 reports use the date as a filter criteria or a group by criteria,
- 50 reports use product attributes as a filter criteria or a group by criteria,
- 50 reports use customer attributes as a filter criteria or a group by criteria.

Among the 50 reports customer attributes:

- 20 reports use the customer age group as a filter criteria or a group by criteria,
- 30 reports use the customer region as a filter criteria or a group by criteria.
  (Some use both)

Usage frequency

To transpose this case, the system administrator enters following information.

Indicator group weight:

- Order : 40
- Invoice : 100

Date attribute usage frequency with invoices: 1.00

Product branch usage frequency with invoices: 50/100=0.5
This usage frequency is distributed on all branches attributes using their default usage frequency as weighting criteria.

Customer branch usage frequency with invoices: 50/100=0.5

Age Group attribute usage frequency with invoices: 20/100=0.2

Region attribute usage frequency with invoices: 30/100=0.3

Usage frequencies with invoices of other attributes of the customer branch are adjusted so that the combination of usage frequencies of all attributes of the branch gives 0.5.
Their default usage frequency is used as weighting criteria.

Note
ACOD-BI considers the usage of two different attributes as unrelated events.

As a consequence there is a relation between a branch usage frequency and the usage frequency of each attribute of that branch[22].

Attention to inconsistencies
When a branch usage frequency is entered, there may be an inconsistency in two cases:

1) The system administrator enters a usage frequency on all attributes of the branch and the usage frequency of the branch is different from the combination of the usage frequency of the attributes. *This is why it is not recommended to enter usage frequency on a branch and on all its attributes.*
2) The branch usage frequency is less than the combination of the entered attribute usage frequencies.

In case of inconsistency, an error is thrown.
The line of the Use case file introducing the inconsistency is not loaded and the error is recorded in the log table.

Example

Line N of the Use case file.
Customer branch usage frequency for invoices: 0.7

Line N+1 of the Use case file.
Age group attribute usage frequency for invoices: 0.5

Line N+2 of the Use case file.
Region attribute usage frequency for invoices: 0.5

When taking into account the N+2 line of the Use case file, the combination usage frequencies entered at attribute level on the client branch = 1- (1-0.5) x (1-0.5) > 0.7

Loading the N+2 line of the Use case file would introduce an inconsistency.
The N+2 line of the Use case file is not loaded.
The region attribute usage frequency stay free.
An error message is inserted in the log table.

[22] Branch usage frequency = 1 - Product on all attributes (1 – Attribute usage frequency).

9 Datamart codes

3 types of code are used to construct datamart object names:

- Entity codes
  Used to build names of most tables.
- DField codes
  Used to build names of most columns as well as some tables.
  A DField corresponds to an attribute or to an indicator.
- Hierarchy link differentiating codes
  Used to differentiate hierarchy links having the same child entity and the same parent entity.
  Used as complement to DField codes to build some column names.

These codes must comply with the following rules:

- Uniqueness of entity codes,
- Uniqueness of DField codes,
- Uniqueness of differentiating codes for two hierarchy links with the same child entity and the same parent entity.
- Entity code size <= 4
- DField code size <= 19
- Hierarchy link differentiating code size <= 4

The 3 types of code have the same building method.

A function determines a default code model from objects names in the source (see global parameters FUNC_NTAB_CODE2ENTI_CODE, FUNC_NCOL_CODE2INFO_CODE and FUNC_NFK_CODE2HILI_DIFF_CODE).

This default value can be overridden by explicitly entering a code model (file « exc_enti.csv » column NEW_EXPLICIT_CODE_MODEL, file « exc_info.csv » column NEW_EXPLICIT_ATTR_CODE_MODEL and NEW_EXPLICIT_INDI_CODE_MODEL, file « exc_hili.csv » column NEW_EXPLICIT_DIFF_CODE_MODEL).

If required, a code model is transformed into a code that respects the rules of uniqueness and size.

10 Appendix A – Structure of CSV files

The format of numbers is without thousand separator and with decimal separator "."

10.1 Global settings file

File name is « exc_param.csv ».

File columns are:

CODE
Code to identify the global parameter.

VALUE_TYPE
Data type (String or Float)

OLD_VALUE
Parameter value before loading the file in the system.

DEFAULT_VALUE
Parameter default value.

OLD_VALUE_IS_DEFAULT
Indicates if the value before loading the file is the default value.

NEW_VALUE
New value to be considered by ACOD-BI when loading the file.

DESCRIPTION
Parameter description.

10.2 DField file

File name is « exc_info.csv ».

File columns are:

DWH_TABLE
Name of the source table containing the column corresponding to the DField.

DWH_COLUMN
Name of the source column corresponding to the DField.

OLD_ENTITY_DTM_CODE
Datamart entity name containing the DField.

ATTRIBUT_ENABLED
Indicates if the DField is allowed as an attribute.

OLD_ATTR_ACTIVATION
Indicates if the DField is an attribute.
Value before loading the file in the system.

(Mandatory) NEW_ATTR_ACTIVATION
Indicates if the DField is an attribute.
New value to be considered by ACOD-BI when loading the file.

OLD_ATTR_DEFAULT_UF
Default value of the usage frequency for this attribute[23].
Value before loading the file in the system.

(Optional) NEW_ATTR_DEFAULT_UF
Default value of the usage frequency for this attribute[24].
New value to be considered by ACOD-BI when loading the file.

OLD_ATTR_DTM_CODE
Attribute code.
Value before loading the file in the system.

OLD_ATTR_CODE_MODEL
Model used to build the attribute code.
Value before loading the file in the system.

OLD_EXPLICIT_ATTR_CODE_MODEL
Model used to build the attribute code entered by the administrator in a previous iteration.
Value before loading the file in the system.

[23] For more details refer to chapter « Reporting statistics ».
[24] For more details refer to chapter « Reporting statistics ».

(Optional) NEW_EXPLICIT_ATTR_CODE_MODEL
Model used to build the attribute code entered by the administrator[25].
Replaces the value found by the determination function.
New value to be considered by ACOD-BI when loading the file.

INDICATOR_ENABLED
Indicates if the DField is allowed as an indicator.

OLD_INDI_ACTIVATION
Indicates if the DField is an indicator.
Value before loading the file in the system.

(Mandatory) NEW_INDI_ACTIVATION
Indicates if the DField is an indicator.
New value to be considered by ACOD-BI when loading the file.

OLD_INDI_AGREG_RULE
Indicator aggregation rule.
Value before loading the file in the system.

(Mandatory) NEW_INDI_AGREG_RULE
Indicator aggregation rule.
Allowed values are MIN, MAX, and SUM.
New value to be considered by ACOD-BI when loading the file.

OLD_INDI_DTM_CODE
Indicator code.
Value before loading the file in the system.

OLD_INDI_CODE_MODEL
Model used to build the indicator code.
Value before loading the file in the system.

OLD_EXPLICIT_INDI_CODE_MODEL
Model used to build the indicator code entered by the administrator in a previous iteration.
Value before loading the file in the system.

(Optional) NEW_EXPLICIT_INDI_CODE_MODEL
Model used to build the indicator code entered by the administrator[26].
Replaces the value found by the determination function.
New value to be considered by ACOD-BI when loading the file.

[25] Go to chapter « Datamart Codes » to see code building from a code model.
[26] Go to chapter « Datamart Codes » to see code building from a code model.

10.3 Entity file

File name is « exc_enti.csv ».

File columns are:

DWH_MAIN_TABLE
Main source table corresponding to the entity.

DWH_TABLE(S)
List of source tables corresponding to the entity.

OLD_DTM_CODE
Entity code.
Value before loading the file in the system.

OLD_CODE_MODEL
Model used to build the entity code.
Value before loading the file in the system.

OLD_EXPLICIT_CODE_MODEL
Model used to build the entity code entered by the administrator in a previous iteration.
Value before loading the file in the system.

(Optional) NEW_EXPLICIT_CODE_MODEL
Model used to build the entity code entered by the administrator [27].
Replaces the value found by the determination function.
New value to be considered by ACOD-BI when loading the file.

OLD INDICATOR(S) WEIGHT
Relative weight of the indicator group in reporting.
Value before loading the file in the system.

OLD EXPLICIT INDICATOR(S) WEIGHT
Relative weight of the indicator group in reporting entered by the administrator in a previous iteration.
Value before loading the file in the system.

(Mandatory) NEW EXPLICIT INDICATOR(S) WEIGHT
Relative weight of the indicator group in reporting entered by the administrator[28].
New value to be considered by ACOD-BI when loading the file.

[27] Go to chapter « Datamart Codes » to see code building from a code model.
[28] For more details refer to chapter « Indicator usage frequencies ».

10.4 Hierarchy link file

File name is « exc_hili.csv ».

File columns are:

DWH_FK
Source foreign key corresponding to the hierarchy link.

CONFLICT
Indicates if at least one other hierarchy link exists with the same child entity and the same parent entity.

CONFLICT_LIST
List of hierarchy links with the same child entity and the same parent entity.

OLD_DTM_DIFF_CODE
Differentiating code.
Value before loading the file in the system.

OLD_DIFF_CODE_MODEL
Model used to build the hierarchy link differentiating code.
Value before loading the file in the system.

OLD_EXPLICIT_DIFF_CODE_MODEL
Model used to build the hierarchy link differentiating code entered by the administrator in a previous iteration.
Value before loading the file in the system.

(Optional) NEW_EXPLICIT_DIFF_CODE_MODEL
Model used to build the hierarchy link differentiating code entered by the administrator[29].
Replaces the value found by the determination function.
New value to be considered by ACOD-BI when loading the file.

DWH_TABLE
Child table of the foreign key in the source.

DWH_COLUMN(S)
List of foreign key child columns in the source.

DWH_R_UK
Unique key referenced by the foreign key in the source.

DWH_R_TABLE
Parent table referenced by the foreign key in the source.

DWH_R_COLUMN(S)
Parent columns referenced by the foreign key in the source.

[29] Go to chapter « Datamart Codes » to see code building from a code model.

10.5 Use case file

File name is « exc_uc.csv ».

In the following descriptions we use the abbreviations:

- "branch" for " attribute hierarchy branch for an indicator group",
- "usage frequency" for " usage frequency for an indicator group",

File columns are:

INDICATOR_ENTITY
Code of the entity containing the indicators (also called indicator group)

ATTRIBUT_ENTITY
Code of the entity containing the attributes.

ATTRIBUT_CODE
Attribute code.

HIERARCHY
Hierarchy linking the indicators to the attributes.

BRANCH_UF_ENABLED
Indicates if entering the branch usage frequency is allowed.

OLD_BRANCH_UF
Branch usage frequency.
Value before loading the file in the system.

OLD_EXPLICIT_BRANCH_UF
Branch usage frequency entered by the administrator in a previous iteration.
Value before loading the file in the system.

(Mandatory) NEW_EXPLICIT_BRANCH_UF
Branch usage frequency entered by the administrator[30].
New value to be considered by ACOD-BI when loading the file.

ATTRIBUT_UF_ENABLED
Indicates if entering the attribute usage frequency is allowed.

OLD_ATTRIBUT_UF
Attribute usage frequency.
Value before loading the file in the system.

OLD_EXPLICIT_ATTRIBUT_UF
Attribute usage frequency entered by the administrator in a previous iteration.
Value before loading the file in the system.

[30] For more details refer to chapter « Branch level entering ».

(Mandatory) NEW_EXPLICIT_ATTRIBUT_UF
Attribute usage frequency entered by the administrator[31].
New value to be considered by ACOD-BI when loading the file.

INDI_DWH_MAIN_TABLE
Main source table corresponding to the indicators group.

INDI_DWH_TABLE(S)
List of source tables corresponding to the indicators group.

FK_HIERARCHY_LIST
List of source foreign keys linking the indicators to the attributes.

ATTR_ENTI_DWH_MAIN_TABLE
Main source table corresponding to the attribute branch or to the attribute.

ATTR_ENTI_DWH_TABLE(S)
List of source tables corresponding to the attribute branch or to the attribute.

ATTR_DWH_TABLE
Source table corresponding to the attribute.

ATTR_DWH_COLUMN
Source column corresponding to the attribute.

[31] For more details refer to chapter « Attribute usage frequencies ».

11 Appendix B – Datamart creation and loading

11.1 Instance parameters

The Oracle instance on which the datamart will be installed must have the following parameters:

- QUERY_REWRITE_ENABLED=TRUE
- QUERY_REWRITE_INTEGRITY=STALE_TOLERATED
- SKIP_UNUSABLE_INDEXES=TRUE
- STAR_TRANSFORMATION_ENABLED=TRUE

11.2 Datamart owner privileges

The Oracle user who owns the datamart schema must have the following system privileges:

- CREATE SESSION,
- ALTER SESSION
- CREATE CLUSTER,
- CREATE DATABASE LINK,
- CREATE SEQUENCE,
- CREATE SYNONYM,
- CREATE TABLE,
- CREATE VIEW,
- CREATE PROCEDURE,
- CREATE TRIGGER,
- CREATE MATERIALIZED VIEW,
- CREATE DIMENSION.

He must also have sufficient quotas on datamart tablespaces[32].

11.3 ACOD-BI tools package

The administrator must create the tools package PKG_ACOD_TOOLS in the datamart schema.
The easiest way is to connect with the datamart owner and run the following scripts:

- PKG_ACOD_TOOLS_decl.sql
- PKG_ACOD_TOOLS_body.sql

11.4 Datamart creation

The administrator creates the datamart structures (tables, views …) in the datamart schema.
The easiest way is to connect with the datamart owner and run the script cre_dtm_<datamart unique code>.sql.

11.5 Datamart loading

The administrator loads datamart data.
The easiest way is to connect with the datamart owner and run the script load_dtm_<datamart unique code>.sql.

[32] For more details refer to chapters « Datamart tablespace names » and « Target datamart disk space ».