Case Study: Load Transaction Data in SAP BW and create Data Store Object

BUSINESS CASE

David Woods is data warehouse architect at Global Bike. In a preceding case study he has loaded master data. Now David will take care of transaction data. In this process David will create an operational data store as well.

GOOD TO KNOW

Data Warehouse Architecture is discussed in numerous publications. The model shown below is the "hub-and-spokes" Data Warehousing architecture that is popular in many organizations. In short, data is moved from databases used in operational systems into a data warehouse staging area, then into a data warehouse layer and finally into a set of conformed data marts. Data is copied from one database to another using a technology called ETL (Extract, Transform, Load).

Source: http://data-warehouses.net/architecture/overview.html

SAP has defined a reference architecture supporting large data warehouse implementations called Layered Scalable Architecture (LSA) which uses semantic partitioning. Instead of storing data in a single InfoCube or DSO, it means distributing the data into several identical InfoCubes or DataStore objects, thus splitting the affected flow. Either organizational or time characteristics serve as semantic partitioning criteria. (Source: http://scn.sap.com/people/juergen.haupl/blog)

1 Create Data Source and Load PSA

TASK

Create a data flow object and create a DataSource for loading transaction data. Load transaction data in Persistent Staging Area (PSA). Loading data into Infoproviders including InfoCube will be done in a subsequent task.

GOOD TO KNOW

- As explained in a preceding case study a DataSource is the SAP BW system view to data in the source system.
- When loading data from source systems the requested data normally is at first stored unchanged in an inbound storage layer.
called Persistent Staging Area (PSA). When loading master data we made use of a simplified approach bypassing PSA. Now we will use PSA.

**NOTE**

When working with the case study, please replace in all names `<course-ID>` with the Id of your course, and `<user-ID>` with your user ID (three digits) provided by your lecturer.

1.1 Open Data Warehousing Workbench

Open Data Warehousing Workbench and navigate to functional area for modeling Data Flows. Look for the InfoArea assigned by your lecturer which has the technical name `<course-ID><user-ID>` and description `Student <user-ID>`.

1.2 Create Data Flow

Create data flow for transaction data. Use `<course-ID>DFSD<user-ID>` and `<course-ID><user-ID> Data Flow Sales Data` for the technical name and description. Make use of data flow template delivered by UCC with technical name `MUDDATAFLOWSALESDATA` and description `Data Flow Template Sales Data`.

**GOOD TO KNOW**

A data flow template describes a data flow scenario with all the required objects and support best practice modeling knowledge. Following the simple data warehouse architecture described in the beginning of this case the data flow template provides objects for DataSource with PSA (1), a DataStoreObject (2) representing the data warehouse layer and the InfoCube (3) as part of the data mart layer. However the data flow template is just a blueprint which has to filled with real objects in the following.

1.3 Check Transaction Data Load File

Check the transaction data load file, `SalesdataV01.csv`, to determine the data separator (e.g., comma) and number format being used. You will need this information later in the exercise.
1.4 Create DataSource

Use source system PC_FILE as well as \( <\text{course-ID}>\text{DSSD}<\text{user-ID}> \) and \( <\text{course-ID}><\text{user-ID}> \) Sales Data for the technical name and description. Use \( Z<\text{course-ID}><\text{user-ID}> \) as application component.

GOOD TO KNOW

- In some data warehouse systems the structure of the inbound storage layer is defined by character fields of variable length. Then there is no conversion necessary at all in order to receive the data from source systems.
- In SAP BW however it is common to define the structure of DataSource and PSA by appropriate data types to ease further processing. E.g. amounts like revenue are stored as currency fields eliminating decimal separator and thousands. Date information is stored in internal format YYYYMMDD. Therefore some rules for initial conversions have to be defined.
- Another example to deal with field types is the sales order number. Since this can potentially contain characters and digits (e.g. SO10001) the SAP data type is CHAR. However in our case we only have digits (e.g. 100001) so that NUMC (character with numerical values only) is a good choice.
- NUMC is also appropriate for year, month and day in order to concatenate this to date format YYYYMMDD later on.
- For amount and quantity fields appropriate fields for units have to be assigned.

1.5 Assign DataSource to Data Flow

Go back to your Data Flow and assign your DataSource.

1.6 Create InfoPackage and Load PSA

Create a InfoPackage and load data from flat file to PSA. Use \( <\text{course-ID}><\text{user-ID}> \) Sales Data for description of InfoPackage.

2 Create and Load DataStore Object

**TASK**

After loading the data into the staging layer in the previous task we now want to take care of the data warehouse layer (cf. architecture overview at the beginning of this case study). For this we will create a DataStore Object and pass data from PSA. Loading data into InfoCube will be done in a subsequent task.

GOOD TO KNOW

- There are different reasons to build a data warehouse layer. Following Bill Inmons approach (cf. Inmon: Building the Data Warehouse) all data should be consolidated in a common layer before passing it to data marts for reporting.
- In SAP BW InfoCubes are optimized for multidimensional reporting and therefore used to build data mart layer. Data
warehouse layer is build by DataStoreObjects which are relational tables with key fields and data fields.

- In SAP BW reporting can be based on DataStoreObjects as well. Therefore it is good practice to store data in InfoCubes on an aggregated level (in our case: monthly sales data) and store granular data (in our case: sales orders) in DataStoreObject to enable special purpose reporting.
- There are several ways to add InfoObjects to the key and data fields of a DataStore Object. Since there is a big overlap we will use InfoCube as a template when creating DataStoreObject.

2.1 Open Data Flow

Open data flow in change mode you created in previous task which has technical name <course-ID>DFSD<user-ID> and description <course-ID><user-ID> Data Flow Sales Data.

2.2 Create DataStoreObject

Create DataStoreObject. Use technical name <course-ID>D1<user-ID> and description <course-ID><user-ID> Sales Items.

2.3 Choose Key fields

Your DataStoreObject should have the following key fields:

<table>
<thead>
<tr>
<th>Name of key field</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales Order</td>
<td>characteristic delivered by UCC</td>
</tr>
<tr>
<td>Order Item</td>
<td>characteristic delivered by UCC</td>
</tr>
</tbody>
</table>

2.4 Choose Data Fields

Your DataStoreObject should have the following data fields. For most fields you can choose your InfoCube as template.

<table>
<thead>
<tr>
<th>Name of key field</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calendar Day</td>
<td>time characteristic delivered by SAP</td>
</tr>
<tr>
<td>Calendar Month</td>
<td>characteristic used in InfoCube</td>
</tr>
<tr>
<td>Calendar Year</td>
<td>characteristic used in InfoCube</td>
</tr>
<tr>
<td>Material</td>
<td>characteristic used in InfoCube</td>
</tr>
<tr>
<td>Customer</td>
<td>characteristic used in InfoCube</td>
</tr>
<tr>
<td>Sales Quantity</td>
<td>key figure used in InfoCube</td>
</tr>
<tr>
<td>Revenue in local currency</td>
<td>key figure used in InfoCube</td>
</tr>
<tr>
<td>Discount in local currency</td>
<td>key figure used in InfoCube</td>
</tr>
</tbody>
</table>

GOOD TO KNOW

- *Net Sales* and *Cost of Goods Manufactured* is not delivered by source system and not stored in DataStoreObject. They will be calculated when transferring the data to InfoCube
- *Calendar Month* and *Calendar Year* can be derived from *Calendar Day* and therefore are redundant. However it is a good idea to store these fields in DataStoreObject anyway to have them available for selections in reporting.

2.5 Create Transformation

Create a transformation from your data source to the DataStoreObject.

GOOD TO KNOW

In our DataStoreObject we have a field 0CALDAY which expects internal date format YYYYMMDD. The DataSource has day month
and year in three separate fields and no special date field. Since we defined the fields YEAR, MONTH and DAY as SAP data type NUMC of appropriate length the target field CALDAY can be defined by concatenation.

2.6 Create Data Transfer Process and Load Data

Create and execute a data transfer process to load the sales data from PSA to DataStoreObject.

3 Load InfoCube

**TASK**

Now we will take care of data mart layer (cf. architecture overview at the beginning of this case study). We will make use of the InfoCube created in a preceding case study *Creating a Data Mart in SAP BW* and assign this to our data flow. Then we will define appropriate transformation rules and load the data into the InfoCube. This will finish load of transaction data.

3.1 Open Data Flow

Open data flow in change mode you created in previous task which has technical name `<course-ID>DFSD<user-ID>` and description `<course-ID><user-ID> Data Flow Sales Data`.

3.2 Assign InfoCube

Assign your InfoCube that you created in a previous module which has technical name `<course-ID>R1<user-ID>` and description `<course-ID><user-ID> Reporting`.

3.3 Create Transformation

Create a transformation from your DataStoreObject to InfoCube.

**GOOD TO KNOW**

- In our InfoCube we have the fields *Net Sales* and *Cost of Goods Manufactured* which are not delivered by source sales system. These fields have to be calculated by transformation rules.
- *Net Sales* can be calculated by a simple formula as `Revenue - Discount`.
- *Cost of Goods Manufactured* is calculated by `Sales Quantity * Transfer Price` where `Transfer Price` is an attribute of Material.
- However the lookup of master data attributes like `Transfer Price` can not be done within a formula. Therefore this calculation has to be done by a ABAP routine.
- ABAP is the programming language developed by SAP and is the foundation for most of the SAP software.
- We will use two ABAP statements, the first one reading `Transfer Price` from master data table and the second one multiplying the result by `Sales Quantity`. The first statement looks similar to the following:

  \[
  \text{SELECT SINGLE /BIC/MU0TRPRI FROM /BIC/PMU0MATRL INTO RESULT WHERE /BIC/MU0MATRL = SOURCE_FIELDS-/BIC/MU0MATRL AND OBJVERS = 'A'}.\]

  Your code may differ since you have used your own InfoObject for Material. You can have a look at the master data table as follows. Open an additional session and display your InfoObject Material (2). On the Master data /texts tab (3) double-click on Mater data tab (4) to navigate to the table definition in SAP data dictionary.
We need the technical names of the table itself (1), the technical field names of Material (2) and Transfer Price (3).

Moreover you might want to have a look at the table content by choosing Contents and then Execute at the following screen.

Take into account that the source fields might have different technical names as well.

The first ABAP statement fills the result of the table select (which is the Transfer Price) into a local variable RESULT. So in order to get Cost of Goods Manufactured all we have to do is multiply this by Sales Quantity which is a source field. So you can use a statement like \texttt{RESULT = RESULT \ast SOURCE\_FIELDS\=/BIC/MU0SLQTY}. Again the source field might have a different technical name.
3.4 Create Data Transfer Process and Load Dataa

Create and execute a data transfer process to load the sales data from DataStoreObject to InfoCube.

3.5 Check Calculated Key figures

Display cube data to check the calculations at transformation time of Net Sales and Cost of Goods Manufactured. Display master data of Material and check that Cost of Goods Manufactured is calculated properly as Sales Quantity * Transfer Price.