Towards a new automatic data warehouse design method

Vers une nouvelle méthode automatique de conception des entrepôts de données

Nawfal El Moukhi

MISC Laboratory, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco

elmoukhi.nawfal@gmail.com

Ikram El Azami

MISC Laboratory, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco

akram_elazami@yahoo.fr

Abdelaaziz Mouloudi

MISC Laboratory, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco

mouloudi_aziz@hotmail.com

Abdelali Elmounadi

LASTIMI, Mohammadia School of Engineers, Mohammed V University, Rabat, Morocco

a.elmounadi@gmail.com

Résumé

Les entrepôts de données sont actuellement reconnus comme étant un composant essentiel des systèmes d'aide à la décision dans la mesure où ils offrent la meilleure réponse aux problèmes de prise de décision des différents domaines fonctionnels des organisations. Cependant, la conception et la construction des entrepôts de données demeurent une tâche très complexe, difficile à accomplir. Cette complexité est essentiellement la conséquence de l'absence de techniques et de méthodes reconnues dans le domaine. Dans ce contexte, cet article identifie différentes règles pour la conception des entrepôts de données à partir de données relationnelles et introduit une nouvelle méthode pour l'automatisation de ce processus en se fondant sur les technologies MDA et XML.

Abstract

Nowadays, the data warehouse is recognized as the essential component of decision support systems since it ensures the best response to the decision problems of different functional areas of an organization. However, designing and building a data warehouse remain a very complex task, difficult to accomplish. This complexity is mainly due to the absence of technics and methods that are recognized in the field. Thus, the present paper identifies different rules for designing a data warehouse from relational data and introduces a new method that aims to automate this process using MDA techniques and XML.

Mots clés

Entrepôt de données, modèle relationnel, modèle multidimensionnel, conception d'entrepôts de données, Architecture orientée modèle

Keywords

Data warehouse, Relational model, Multidimensional model, designing data warehouses, Model Driven Architecture.

1. Introduction

The design of data warehouses is one of the most complicated issues in business intelligence. Its complexity is due to the proliferation of data types on the one hand, and on the other, to the abundance of information that gave rise to the new era of Big Data. Furthermore, the data warehouse design phase is the first and the most important step in the decision-making process, since all other steps of the process - data transformation, data analysis, information extraction, online analytical processing (OLAP), etc. - are largely dependent on the quality of the designed and adopted model. For all these reasons, this issue started quickly to arouse researchers' interest since the 1990s when the first research work appeared, which focused on the structure of data warehouses.

Much effort has been devoted to data warehouse design, and several methods automating the data warehouse modeling were developed but none of them has become a consensus (Gosain and Singh, 2015). Despite the lack of a standard model, it is widely assumed that the data warehouse design must follow the multidimensional paradigm (Kumari and Yadav, 2015) and it must be derived from the data sources, since a data warehouse is the result of homogenizing and integrating relevant data of the organization in a single and detailed view (Taniar and Chen, 2011). Other research considers that user requirement analysis is crucial in data warehouse design (Abai et al., 2013) and therefore some experts developed a new method that supports both approaches (Battaglia et al., 2011).

In this paper, we propose a new method that follows the data-driven paradigm to design a data warehouse from relational data sources. We opted for this approach because it permits significant time saving since the start of the data warehouse design project requires only the availability of the transactional data. The choice of relational data is explained by their widespread use in all types of organizations (Ghosh, 2010).

The rest of this paper is organized as follows: in the next section, we present a set of rules dedicated to perform the transformation from a relational data model to a data warehouse model. Section 3 describes the transformation method by applying the set of rules previously elaborated in the first part, before moving to describing the transformation engine. Section 4 comes as the conclusion part where we describe the perspectives of this work.

2. Related work

The design of data warehouses has been subject of several research projects. Generally, existing approaches can be categorized into three categories: Bottom-up, top-down and mixed approaches.

Bottom-up approaches start from a detailed analysis of data sources, but missed the decision-makers needs. The works of (Golfarelli and Rizzi, 1998), (Moody and Kortink, 2000), (Vrdoljak et al., 2003), (Varga, 2002) and (Sehgal and Ranga, 2016) present different approaches that allow to generate the data warehouse schema from Entity / Association diagrams of the data sources. The methods proposed by (Romero and Abelló, 2010) and (Jensen et al., 2004) exploit data mining techniques and ontologies to generate the multidimensional schema. There are even solutions suggested by big companies such as Oracle. They proposed a set of tools that allow the transformation of logical structure to relational structure and next transformation to Multidimensional Model of warehouse in star or snowflake schema (Drzymala et al., 2012).

Top-down approaches (Winter and Strauch, 2003), (Annoni et al., 2006) and (Jovanovic et al., 2014) allow building the data warehouse schema from a detailed analysis of the decision

makers needs. Verification of the correspondence between these needs and the data sources is done a posteriori.

Mixed approaches (Phipps and Davis, 2002), (Giorgini et al., 2005) and (Abdelhedi and Zurfluh, 2013) consider both the needs of decision makers and the source data. Therefore, they have the advantage of designing multidimensional schemas that respect the data source structure.

If we analyze these different methods, especially those following the data-driven approach (Bottom-up), we can see that they are all semi-automatic, and there are even some methods that just provide guidelines and recommendations to get suitable multidimensional schemas. In this context, our work consists of developing a new fully automatic method called X-ETL. This method will allow to transform a relational model into a multidimensional model without any human intervention.

3. Rules for data warehouse design from relational data

This section presents the set of rules that we have developed from previous work (Khouri et al., 2014) (Elmoukhi et al., 2015)(Khnaisser et al., 2015)(Santos et al., 2016) (Dahlan and Wibowo, 2016). These rules will form the foundation of our solution to standardize the data warehouses design.

3.1 Rules for Facts and Measures

- The fact tables are the concepts of main interest for the decision making process. They correspond to events that always occur in the organization or company (Chandwani and Uppal, 2015) ;
- The measures of the fact table should be numeric and additives (at worst semi-additives) (Akbar et al., 2013) ;
- The data of a fact table are fixed and cannot be changed (Bliujute et al., 1998) ;
- A fact table represents always a particular activity and should be interrogated from a particular context (one or a few dimensions) ;
- No line of the fact table can contain an empty value ;
- A fact table contains only the foreign keys which represent the primary keys of the dimensions and these keys must be numeric, to ensure that the fact table is more efficient (Rudra and Nimmagadda, 2005) ;
- Each combination of dimension values defines an instance of the fact table, which is characterized by one and only one value for each measure.

Below are the mathematical representations of the rules for facts and measures:

Let T_F be a fact table, M_{TF} a fact table measures, D_i a dimension of the fact table and m an instance of M_{TF}.

- $T_F = P(E_v)$
  with:
  P : Main interests
  E_v : Company events ;
- Let m_1 and m_2 be two instances of M_{TF}. If m_1 and m_2 are additives:

$\exists\ m_3 = m_1 + m_2$
With $m3$ an instance of the same measure M_{TF} ;

- Suppose that f is a change function on T_F
  $\forall\ m \in M_{TF}$
  $f(m)=\alpha$
  With α a constant ;
- Let F be the set of fact tables and A a particular activity of the organization
  For each $T_F \in F$ we have:
  $T_F = A$;
- $\forall\ T_F$there is at least one function f which applies at least one dimension D_i on T_F ;
- Let L_{TF} be the set of rows of a fact table and l a row of L_{TF}
  $\forall\ l \in L_{TF}$
  $l \neq \emptyset$;
- Let fbe a foreign key and p a primary key
  we have :
  $\{f_1,f_2,...,f_n\}=\{p_1,p_2,...,p_n\}$
  $\forall k \in \{1,2,...,n\}$
  with $f_k \in T_F$and $p_k \in D_i$
  and f_k and p_k of type Integer ;
- Let C be the set of combinations of dimension values, c a combination of C and f a function on M_{TF} :
  - For each instance m of M_{TF}, the combination $f(m) \in C$.
  - For each combination c of C, the equation $f(m) = c$ admits a unique solution (any combination c of C admits a unique antecedent M_{TF}) $f(m)$ is bijective.

3.2 Rules for Dimensions and Attributes

- The dimensions determine how fact instances can be aggregated significantly for decision making process ;
- A fact table must always contain the time dimension ;
- The dimensions should have numeric primary keys ;
- The primary key of each dimension table should be unique (preferably auto-increment), and fields should have an atomic value (not compound) ;
- The dimension hierarchies should preferably have a simple form of 1-n type, and avoid relationships of n-n type ;
- A non-dimensional attribute contains additional information on an attribute of the hierarchy, and is linked by to-one relationship (Golfarelli et al., 1998) ;
- The non-dimensional attributes cannot be used for aggregation (Golfarelli et al., 1998) ;
- The relationship between a fact table and a dimension is always many-to-one(Cavalheiro and Carreira, 2016).

Below are the mathematical representations of the rules for dimensions and attributes:

Let T_F be a fact table and D_i a dimension of the fact table.

- Let f be an aggregation function on T_F
  f is significant if and only if f applies one or more dimensions on the instances of T_F ;
- $\forall T_F (T_F \ni D_{it})$ with D_{it} a time dimension ;
- Let C_p be the set of primary keys of D_i
  $\forall D_i$
  $C_p \in \mathbb{N}$;
- Let C_p be the set of primary keys of D_i and p_1 and p_2 two instances of C_p
  $\forall p_1$ and p_2
  $p_1 \neq p_2$;
- Let R be a relationship between two dimensions
  $\forall R$
  $R \neq (n,n)$;
- Let f be an aggregation function, A the set of its attributes, and a_n a non dimensional attribute
  For any non-dimensional attribute a_n we have:
  $a_n \notin A$;
- Let R be a relationship between T_F and D_i
  $\square$ T_F and D_i
  R= (n,1).

3.3 Application of Rules

We take as an example the model below. It represents the sales activity of a set of products in a chain of stores located in several cities and countries. The model is composed of a purchase table related to three tables:

- The product table containing the reference, the description, the price and the type of the product sold. We assume that a purchase concerns one and only one product, but a product may concern several purchases;
- The customer table that contains the customers' information;
- The city table where the store is located (a city can only contain a single store).

The city table is also related to the department table to which belongs a set of cities, then the department table to the region table and finally the region table to the country table.

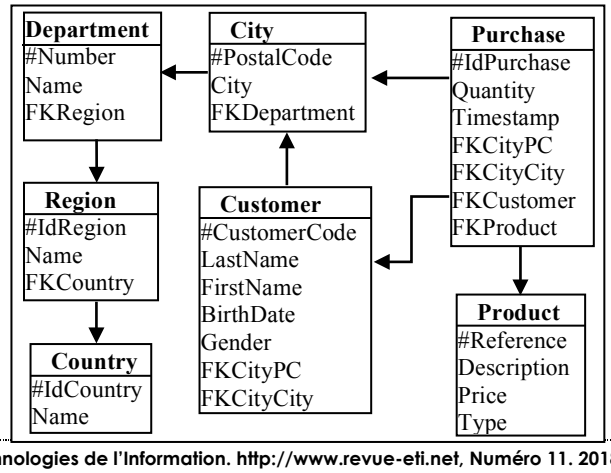

Figure 1. Example of a sales transactional model

By following the rules presented in the previous sections, the fact table will be the Purchase table. It is the table that represents a particular activity of the company and the main interest for decision makers. It contains a numeric and additive field which can be considered as the main measure of the fact table. In addition, this table contains the highest number of many-to-one relationships (the highest number of foreign keys), which is the privileged type for relations between fact tables and dimension tables (the last rule in the section 3.2).

Concerning the dimension tables, they will be respectively the Product table, the Customer table, and the City table since they are related directly to the fact table by a one-to-many relationship. All these tables represent the analysis contexts of the fact table and determine how fact instances can be aggregated significantly for the decision-making process. The City table is related to a tree of tables representing the location, and therefore they can all be grouped together in a single table (Dim_Place). Finally, our example does not contain any time table, so it is necessary to add one that will represent the time dimension (rule 2 in section 3.2).

The multidimensional model below is the final result of this transformation:

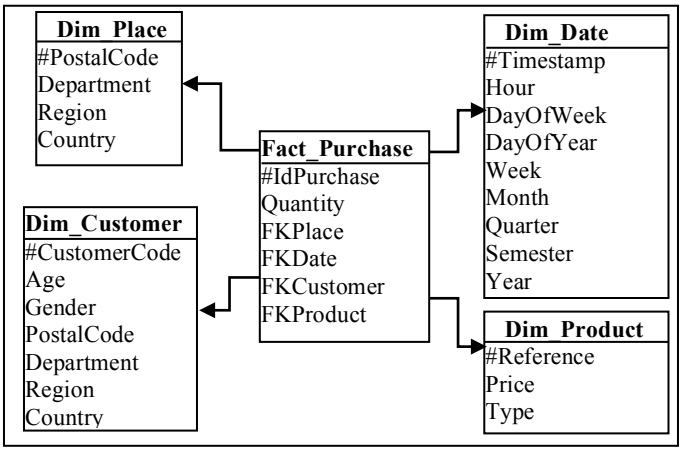

Figure 2. Sales transactional model example after rules application

Thus, we chose the Purchase table as fact table, as it represents a particular business activity and a main interest for the decision making process. This fact table contains a numeric and additive measure, and the foreign keys of the dimensions that are also numeric. We also added a dimension table that contains the different granularity of time, and we tried to choose the dimensions and attributes that will enable a significant aggregation of fact instances. In this sense, the attribute Description of the Product table was not retained as an attribute of the dimension since it contains only a description of the product and no data that can determine how instances of the fact table can be aggregated (cf. rule 1 for dimensions and attributes).

There is no doubt that these rules will facilitate the identification of facts, measures, dimensions and attributes from relational data. Therefore, we have the essential components for the construction of our own method while using the Model-driven Architecture (MDA) techniques. Once our method will be developed, we will apply it for building a data warehouse for the National Library of Morocco from their relational data.

4. A new method for transforming a relational model to multidimensional model

4.1 Model Driven Architecture

MDA (Model-Driven Architecture) is a standard of the OMG (OMG, 2001), which is based on the MDE (Model-driven engineering), providing a set of guidelines and an architecture for the design of software systems. The MDA approach provides the opportunity to understand complex systems and the real world through their abstraction. This abstract view of the system is elaborated in a conceptual framework as well as a number of standards provided by the OMG. These standards allow to define the models, their relations and their transformations (for example: UML -Unified Modeling Language, MOF and XMI -XML Metadata Interchange). In order to visually represent the MDA approach, the OMG has set up a framework, structured of several types of models. Figure 3 shows the development cycle in Y, which implements these models and their relationships:

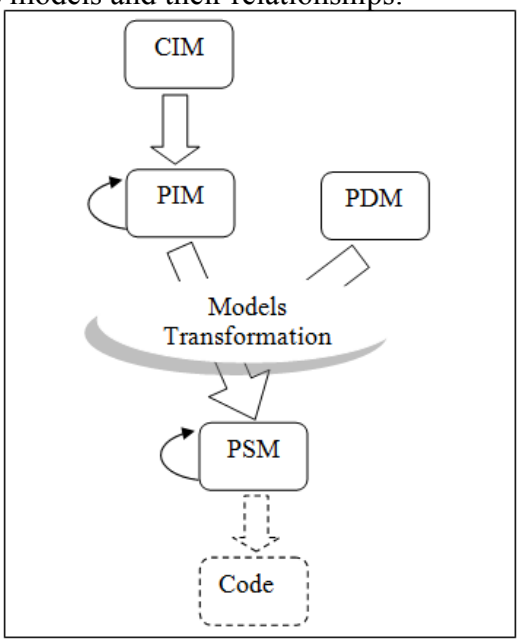

Figure 3. The MDA process (OMG, 2001)

- CIM (Computation Independent Model): he CIM allows a vision of the system and its environment, while hiding the details of structure and implementation. Models of the CIM level help narrow the gap between domain experts and designers. As a result, a CIM model is sometimes called a domain model;
- PIM (Platform Independent Model): Models of the PIM level represent a vision of system analysis and design, independently of any technological details concerning the platform (operating system, programming language, hardware, network performance, etc.);
- PSM (Platform Specific Model): The PSM level presents a projection of PIM level models to a specific platform. These models combine PIM specifications with platform specific details;

- PDM (Platform Description Model): These models describe the platform on which the system will be executed, by providing a set of technical data regarding the functionality and use of the platform.

4.2 Models transformation

Model transformation in MDA context consists to transform the PIM models to PSM models. This process is performed by a transformation engine that applies a set of rules to PIM to generate the PSM.

The concept of meta-model is omnipresent in this case. Thus, each model (PIM or PSM) is based on a meta-model used to describe it. When both models use the same meta-model, it's about an "endogenous" transformation; we talk about "exogenous" transformation in the opposite case.

There are mainly two types of transformations:

- Transformations M2M (Model to Model): used to transform models to other models, these transformations concern all tasks to be executed in order to get a model respecting the technical specifications of the target environment. The standard which technically represents this type of transformation in the MDA approach is the MOF 2.0 QVT;
- Transformations M2T (Model to Text): used to generate code or documentation. M2T transformations constitute the MOFM2T project which is one of many parts of MDA project.

In our case, we are interested in M2M transformation since we aim to transform a relational model to a multidimensional model.

Basically, the main steps to complete the transformation are as follows:

1. *To specify the source meta-model*: first, we have to specify the source meta-model, in our case it is the meta-model of relational schema;
2. *To specify the target meta-model*: we must also specify the meta-model representing the decisional concept;
3. *To build the transformation engine*: this step will be based on the rules presented in section 3.

The figure below illustrates these different steps:

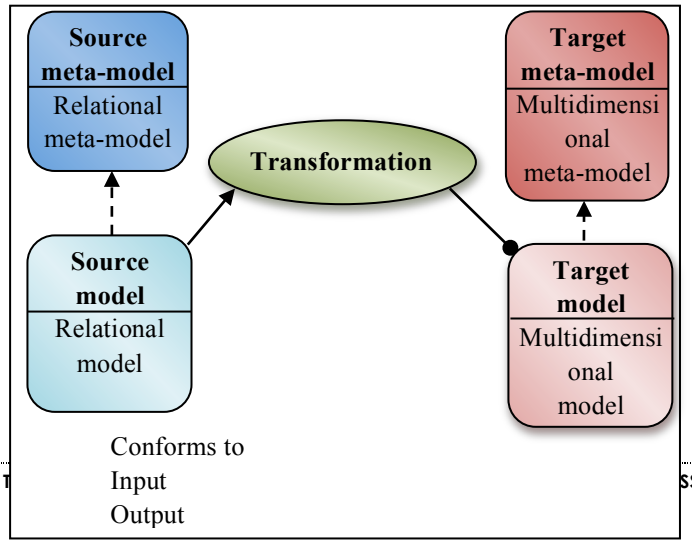

Figure 4. Transformation of a relational model into a multidimensional model (Blanc and Salvatori, 2005)

4.3 Eclipse Modeling Framework

EMF is a modeling and code generation platform that facilitates the construction of tools. It is about a set of development tools integrated into the Eclipse environment in the form of plugins among which we quote: the Ecore meta-model, the editor EMF.Edit, the generation model GenModel, etc. EMF was designed to open Eclipse to the model-driven development. It is an approach based on simplifying MOF. It allows to define meta-models then to derive an implementation in Java to build instance models (Budinsky et al., 2009).

Figure 5 shows the architecture of the EMF Framework. The main role of this structure is to accept models or files as input, and to generate code corresponding to tools (plug-in) manipulating the input data.

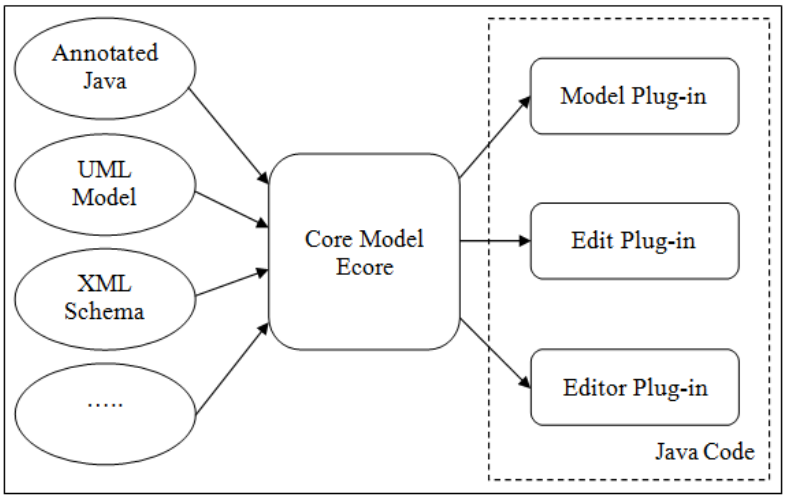

Figure 5. EMF Architecture (Budinsky et al., 2009)

4.4 Defining the source and target meta-models

We started by defining our two meta-models (source and target) by using Ecore which is considered as an EMF model (Eclipse Modeling Framework). In this regard, we note that all the meta-models presented in this section are our proposal: in the ISCRAM-med conference (El Moukhi et al., 2016), we have presented the relational metamodel and the multidimensional metamodel that covers all multidimensional models.

The relational meta-model consists of three essential elements: a database that contains tables which in turn contain columns. So we tried to resume these components in the figure below (El Moukhi et al., 2016):

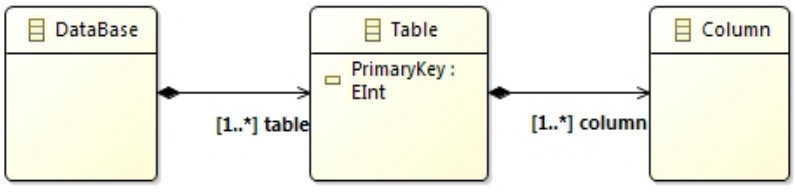

Figure 6. The relational meta-model

Concerning the multidimensional meta-model, it consists of a multidimensional schema that contains facts and dimensions. In order to perform a multidimensional analysis it is necessary to have at least two dimensions. Each fact contains measures and each dimension contains an hierarchy of attributes. The figure 7 below resumes these components and describes our multidimensional meta-model (target) (El Moukhi et al., 2016):

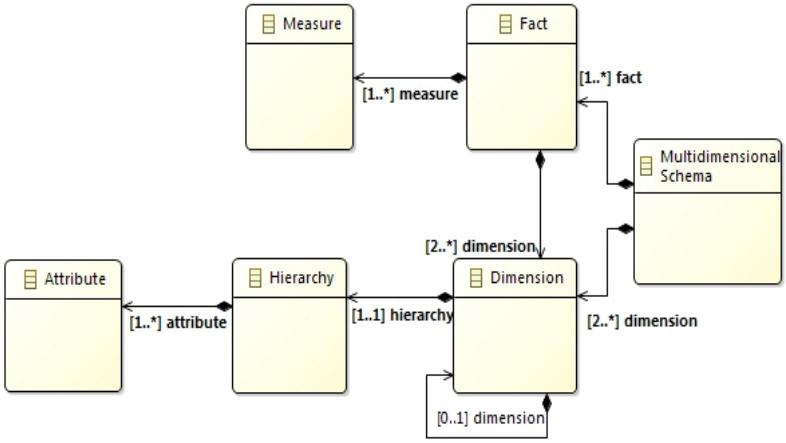

Figure 7. The multidimensional meta-model

Contrary to the previous work, this paper deals in detail with the metamodels of the various types of multidimensional model, (Figures 8, 10, 12) and introduces a new method that allows to transform the relational model into a multidimensional one. Thus, we have:

- The star schema which contains a single fact table directly linked to dimensions (see example of figure 2) and no dimension is related to another, that's why we removed the reflexive link on the dimension. Its meta-model is described below (figure 8):

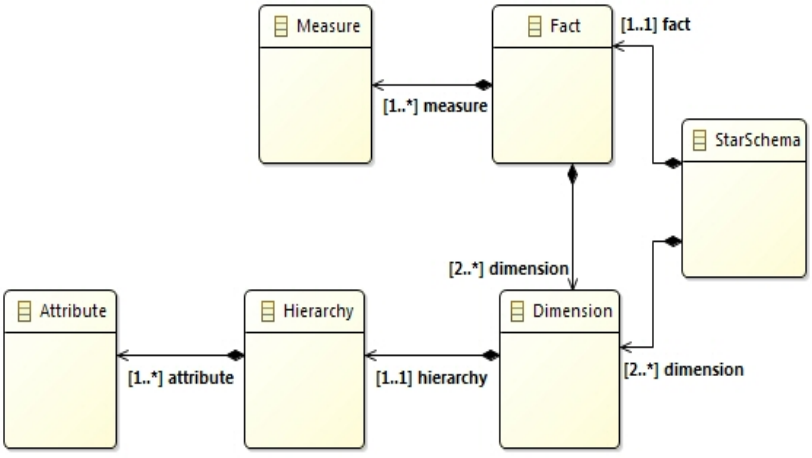

Figure 8. The multidimensional meta-model for star schema

- The snowflake schema which contains a single fact table with dimensions which may be linked to other dimensions. The example below (figure 9) illustrates this type of model.

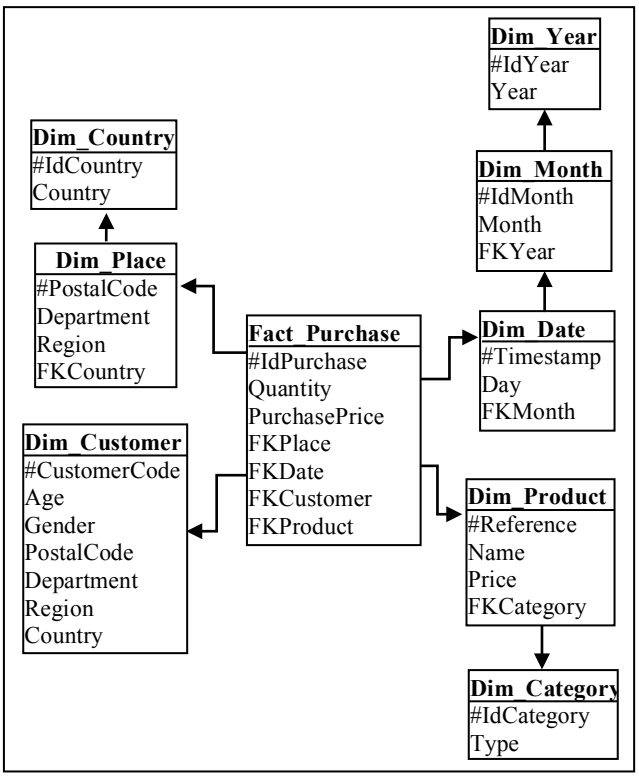

Figure 9. The snowflake schema for sales example

The meta-model corresponding to snowflake schema is shown in the figure 10:

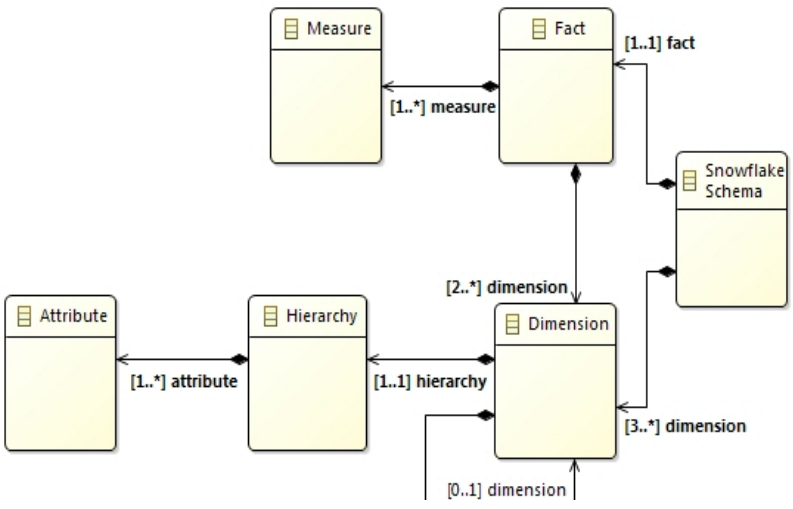

Figure 10. The multidimensional meta-model for snowflake schema

- The constellation schema which is the most complex. It may contain two or many fact tables with shared dimensions, as shown in the example below (Figure 11).

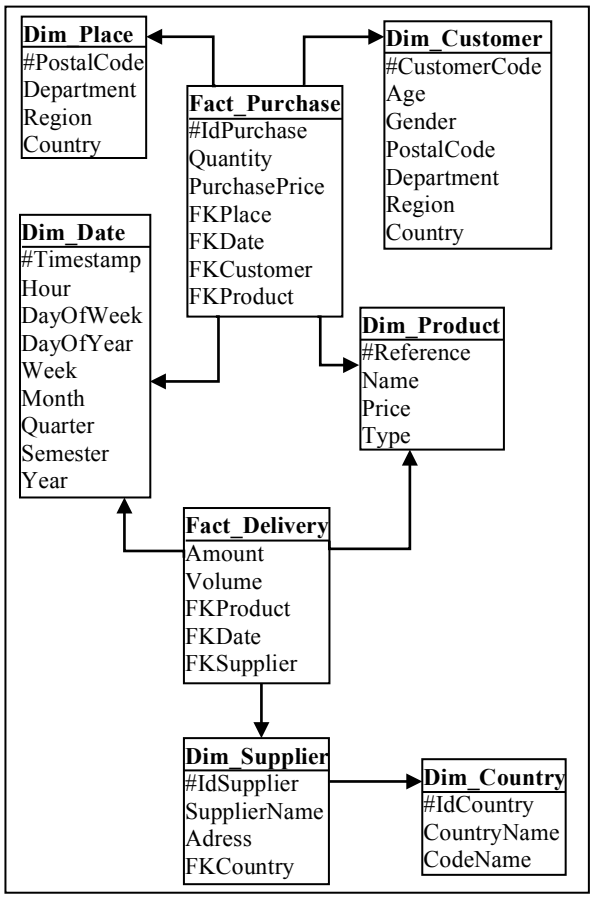

Figure 11. The constellation schema for sales example

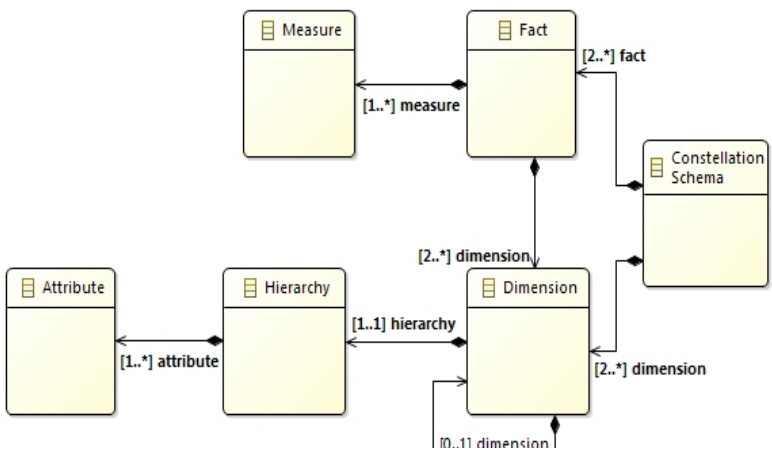

Figure 12. The multidimensional meta-model for constellation schema

In order to verify the conformity of our models to these meta-models, we created two other files with xsd language in which we described how the model structure should be to comply with its meta-model. Thus, we got an xsd file to validate the relational model and another one to validate the multidimensional model. The codes fragments below correspond to these two files.

The XML Schema Definition for relational models :

```
<?xmlversion="1.0"?>
<xs:schemaxmlns:xs="http://www.w3.org/2001/XMLSchema"attributeFormDefault="unquali
fied"elementFormDefault="qualified">
<xs:elementname="relationalSchema"type="relationalSchemaType"/>
<xs:complexTypename="columnType">
<xs:simpleContent>
<xs:extensionbase="xs:string">
<xs:attributetype="xs:string"name="name"use="optional"/>
<xs:attributetype="xs:string"name="isPk"use="optional"/>
<xs:attributetype="xs:string"name="type"use="optional"/>
<xs:attributetype="xs:string"name="isFk"use="optional"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:complexTypename="columnsType">
<xs:sequence>
<xs:elementtype="columnType"name="column"maxOccurs="unbounded"minOccurs="0"/>
</xs:sequence>
</xs:complexType>
<xs:complexTypename="associationType">
<xs:simpleContent>
<xs:extensionbase="xs:string">
<xs:attributetype="xs:string"name="multiplicity"use="optional"/>
<xs:attributetype="xs:string"name="target"use="optional"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:complexTypename="associationsType">
<xs:sequence>
<xs:elementtype="associationType"name="association"maxOccurs="unbounded"minOccurs=
"0"/>
</xs:sequence>
</xs:complexType>
<xs:complexTypename="tableType">
<xs:sequence>
```

The XML Schema Definition for multidimensional models

```
<?xmlversion="1.0"?>
<xs:schemaxmlns:xs="http://www.w3.org/2001/XMLSchema"attributeFormDefault="unquali
fied"elementFormDefault="qualified">
<xs:elementname="multidimensionalSchema"type="multidimensionalSchemaType"/>
<xs:complexTypename="measureType">
<xs:simpleContent>
<xs:extensionbase="xs:string">
<xs:attributetype="xs:string"name="name"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:complexTypename="foreignkeyType">
<xs:simpleContent>
<xs:extensionbase="xs:string">
<xs:attributetype="xs:string"name="name"use="optional"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:complexTypename="fieldsType">
<xs:sequence>
<xs:elementtype="measureType"name="measure"/>
<xs:elementtype="foreignkeyType"name="foreignkey"maxOccurs="unbounded"minOccurs="0
"/>
</xs:sequence>
</xs:complexType>
<xs:complexTypename="associationType">
<xs:simpleContent>
<xs:extensionbase="xs:string">
<xs:attributetype="xs:string"name="multiplicity"use="optional"/>
<xs:attributetype="xs:string"name="target"use="optional"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:complexTypename="associationsType">
<xs:sequence>
<xs:elementtype="associationType"name="association"maxOccurs="unbounded"minOccurs=
"0"/>
</xs:sequence>
</xs:complexType>
<xs:complexTypename="factType">
<xs:sequence>
<xs:elementtype="fieldsType"name="fields"/>
<xs:elementtype="associationsType"name="associations"/>
</xs:sequence>
<xs:attributetype="xs:string"name="name"/>
</xs:complexType>
<xs:complexTypename="attributeType">
<xs:simpleContent>
<xs:extensionbase="xs:string">
<xs:attributetype="xs:string"name="name"use="optional"/>
<xs:attributetype="xs:string"name="isPk"use="optional"/>
<xs:attributetype="xs:string"name="type"use="optional"/>
<xs:attributetype="xs:string"name="isFk"use="optional"/>
```

Many other meta-models have been proposed in the literature. We quote as examples (Hachaichi and Feki, 2013) (Srai et al., 2017) (Sapia et al., 1999) (Choura and Feki, 2011). Concerning the relational meta-model, it is a well-known standard that is repeated in almost

all research with some slight differences (Lämmel, 2005) (Chang et al., 2003) (Inria, 2005). As for the multidimensional meta-model, we find that each existing research work presents a new schema but the common point between all these works is that these meta-models remain too general and do not treat each type of multidimensional model separately. We take as an example the CWM schema (Figure 13) which represents the most known and the most used multidimensional meta-model in the field.

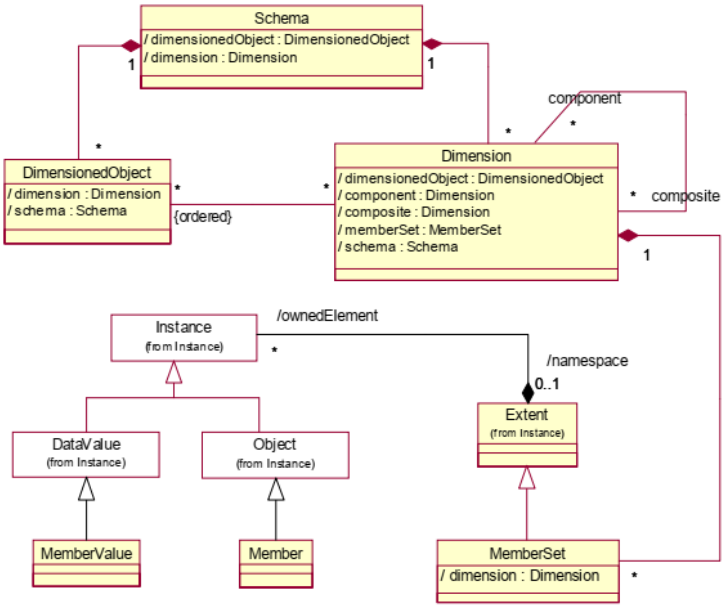

Figure 13. Multidimensional Metamodel of CWM (OMG, 2003)

As we can see, it is a meta-model that covers all types of multidimensional models and therefore there will be fewer restrictions when we choose to work with a specific type. In this context, our work comes to complete these research works by treating each type of multidimensional model (star-flocon-constellation) separately and presenting its meta-model. Thus, we obtained for the snowflake schema (figure 10) a cardinality of three for the dimension table, since we must have at least one more dimension related to another. For the constellation schema (figure 12), we must have at least two fact tables related by dimensions, which explains the cardinality of two for the fact table.

4.5 The transformation engine

After defining our meta-models, we built the transformation engine based on the rules presented in section 2 and principles of the MDA. Firstly, we developed a Java program that calculates the number of foreign keys in each table of the relational model, which allowed us to detect the (potential) fact tables.

Once the fact tables are identified, we generate a multidimensional model that contains only the fact table with the dimension tables that are directly related to it in the relational model. After that, we will also include the indirectly related dimensions in a future work.

The figure 14 shows the architecture of the X-ETL project.

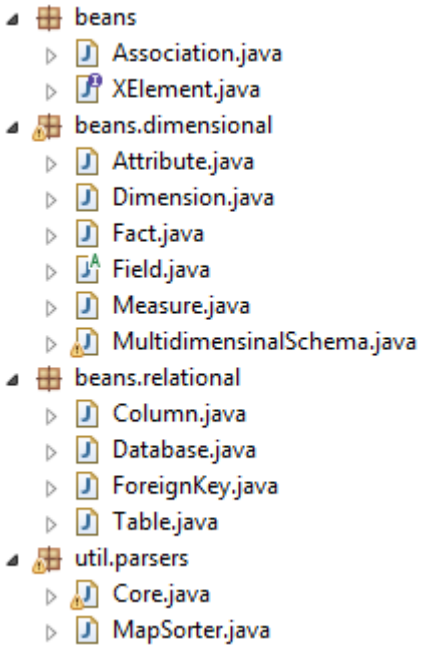

Figure 14. X-ETLproject architecture

The most important part of the project is the file that allows parsing tables in order to calculate the number of foreign keys and then identify potential fact tables. The figure 15 shows the edit window for this file.

```
NodeList tableList = doc.getElementsByTagName("table");
for (int i=0;i<tableList.getLength();i++) {
    Node t = tableList.item(i);
    if(t.getNodeType() == Node.ELEMENT_NODE) {
        Element xmltable = (Element) t;
        Table table = new Table(database);
        table.setName(xmltable.getAttribute("name"));
        database.getTables().put(table.getName(), table);
        NodeList tableChilds = xmltable.getChildNodes();
        int count = 0;
        for (int j = 0; j < tableChilds.getLength(); j++) {
            Node e = tableChilds.item(j);
            if(e.getNodeType() == Node.ELEMENT_NODE && e.getNodeName().equals("columns")) {
                NodeList columns = e.getChildNodes();
                for (int k = 0; k < columns.getLength(); k ++) {
                    Node n = columns.item(k);
                    if (n.getNodeType() == Node.ELEMENT_NODE && n.getNodeName().equals("column")) {
                        Element fk = (Element) n;
                        Column column = new Column();
                        column.setName(fk.getAttribute("name"));
                        column.setType(fk.getAttribute("type"));
                        if (fk.hasAttribute("isPk")) {
                            column.setPK(true);
                        }
                        if (fk.hasAttribute("isFk")) {
                            column.setFK(true);
                        }
                        table.getColumns().add(column);
                        String isfk = fk.getAttribute("isFk");
                        if(isfk.equals("true")) {
                            count ++;
                        }
                    }
                }
            }
```

Figure 15. Dom parsing of table element

Once the fact tables are detected, the second step consists of detecting the dimensions by parsing associations of the relational model to identify many-to-one relationships. The figure 16 represents the Dom parsing of association element.

```
NodeList associations = e.getChildNodes();
for (int k = 0; k < associations.getLength(); k ++) {
    Node n = associations.item(k);
    if (n.getNodeType() == Node.ELEMENT_NODE && n.getNodeName().equals("association")) {
        Element assoc = (Element) n;
        Association association = new Association();
        association.setMultiplicity(assoc.getAttribute("multiplicity"));
        ForeignKey fkey = new ForeignKey();
        fkey.setName(assoc.getAttribute("target"));
        association.setTarget(fkey);
        table.getAssociations().add(association);
```

Figure 16. Dom parsing of association element

The complex calculations that are made on the cardinalities to detect the dimensions justify the choice of the JAVA language, since the other transformation languages such as ATL (Atlas Transformation Language) or QVT (Query/View/Transformation) do not allow tomake this kind of complicated calculation.

After detecting the dimension tables, the file described in Figure 17 allows to create them in the target multidimensional model.

```
Element a = (Element) n;

ForeignKey fk = new ForeignKey();
fk.setName(a.getAttribute("target"));

fact.getFields().add(fk);

Association association = new Association();
association.setTarget(fk);
association.setMultiplicity("ManyToOne");
fact.getAssociations().add(association);

Dimension dim = new Dimension();
dim.setName(fk.getName());

Table tempTable = database.getTables().get(fk.getName());

for (int l = 0; l < tempTable.getColumns().size(); l++) {
    Column col = tempTable.getColumns().get(l);
    Attribute attr = new Attribute();
    attr.setName(col.getName());
    attr.setType(col.getType());
    attr.setPK(col.isPK());
    attr.setFK(col.isFK());

    dim.getAttributes().add(attr);
}

mds.getDimensions().put(dim.getName(), dim);
```

Figure 17. Creating a new dimension

The diagram below illustrates all the steps of the X-ETL transformation.

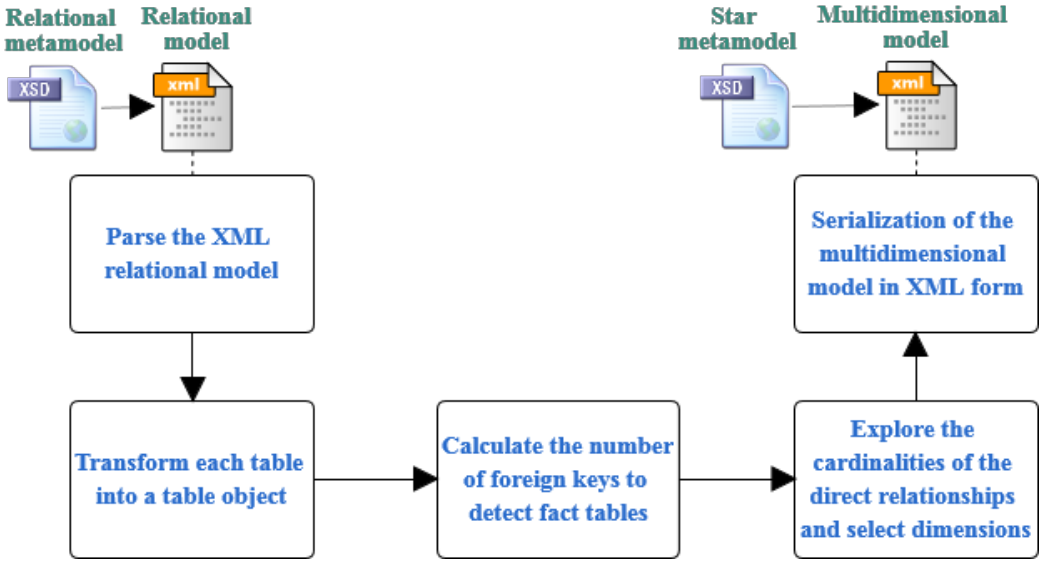

Figure 18. The steps of the X-ETL transformation

Below is a screenshot of the X-ETL application.

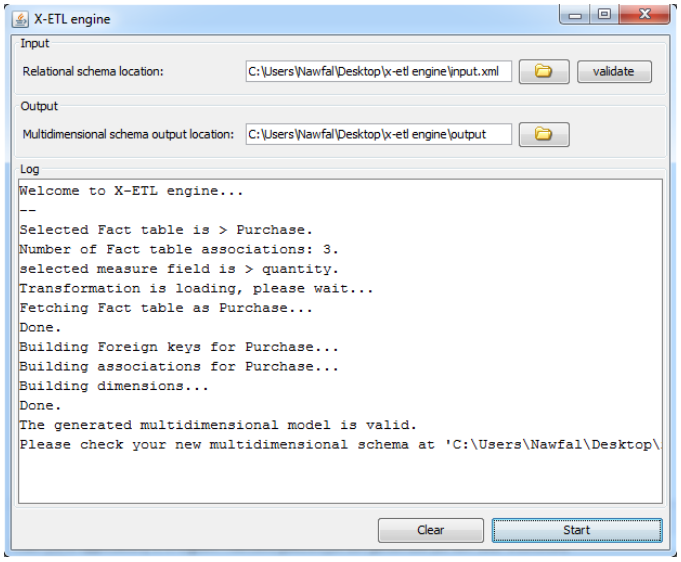

Figure 19. Screenshot of the X-ETL engine

5. Conclusion

In this paper, we presented a new data-driven method for designing a multidimensional model from a relational model. This method is mainly based on a list of rules to identify the different elements of the multidimensional schema and consists of two steps. The first one aims to identify fact tables by calculating the number of foreign keys in each table of the relational model, and the second one allows identifying dimensions that are directly related to the fact table, by analyzing the cardinalities of relations. At the end, several multidimensional models are generated in an automatic way. At this stage, it should be noted that the quality of these generated models depends greatly on the quality of the source model, and therefore it is very important to verify the relational source model before using the X-ETL engine. Our future work will consist of finding a method to select dimensions from tables that are indirectly related to the fact table in order to generate a complete multidimensional model.

References

Abai, N.H.Z., Yahaya, J.H., Deraman, A., (2013). User Requirement Analysis in Data Warehouse Design: A Review. Procedia Technology 11, pp. 801–806. doi:10.1016/j.protcy.2013.12.261

Abdelhédi, F., Zurfluh, G., (2013). User support system for designing decisional database, in : *Proceedings of the 6th International Conference on Advances in Computer-Human Interactions.* Nice, France, pp. 377–382.

Akbar, K., Krishna, S.M., Reddy, T.V.S., (2013). ETL process modeling in DWH using enhanced quality techniques. International Journal of Database Theory & Application 6, pp. 179–197.

Annoni, E., Ravat, F., Teste, O., Zurfluh, G., (2006). Towards multidimensional requirement design, in: *Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery*. Springer-Verlag, Krakow, Poland, pp. 75–84.

Battaglia, A., Golfarelli, M., Rizzi, S., (2011). Qbx: a case tool for data mart design, in: *International Conference on Conceptual Modeling*. Springer, pp. 358–363.

Blanc, X., Salvatori, O., (2005). MDA en action: ingénierie logicielle guidée par les modèles. Eyrolles.

Bliujute, R., Saltenis, S., Slivinskas, G., Jensen, G., (1998). Systematic change management in dimensional data warehousing. Citeseer.

Budinsky, F., Merks, E., Paternostro, M., Steinberg, D., (2009). EMF: Eclipse Modeling Framework. Addison-Wesley,

Cavalheiro, J., Carreira, P., (2016). A multidimensional data model design for building energy management. Advanced Engineering Informatics, Vol. 30, issue 4, pp. 619–632. doi: 10.1016/j.aei.2016.08.001

Chandwani, G., Uppal, V., (2015). Implementation of Star Schemas from ER Model. International Journal of Database Theory and Application 8, pp. 111–130. doi:10.14257/ijdta.2015.8.3.10

Chang, D., Mellor, D., Poole, J., Tolbert, D., (2003). Common Warehouse Metamodel: Developer's Guide. Wiley, p. 86.

Choura, H., Feki, J., (2011). MDA Compliant Approach for Data Mart Schemas Generation, in: *Proceedings of the First international conference on Model and data engineering*. Springer, pp. 262–269.

Dahlan, A., Wibowo, F. W., (2016). Design of Library Data Warehouse Using SnowFlake Scheme Method, in: *7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*. IEEE, pp. 318–322. doi: 10.1109/ISMS.2016.71

ElMoukhi, N., El Azami, I., Mouludi, A., (2016). X-ETL Engine: from relational model to a multidimensional model, in: *3rd International Conference on Information Systems for Crisis Response and Management in Mediterranean Countries (ISCRAM)*. Madrid, Spain, pp. 39–42.

El Moukhi, N., El Azami, I., Mouludi, A., (2015). Data warehouse state of the art and future challenges, in: *International Conference on Cloud Technologies and Applications (CloudTech)*. IEEE. Doi : 10.1109/CloudTech.2015.7337004.

Ghosh, D., (2010). Multiparadigm data storage for enterprise applications. IEEE software 27, pp. 57–60.

Giorgini, P., Rizzi, S., Garzetti, M., (2005). Goal-oriented requirement analysis for data warehouse design, in: *Proceedings of the 8th ACM International Workshop on Data Warehousing and OLAP*. ACM, Bremen, Germany, pp. 47–56.

Golfarelli, M., Maio, D., Rizzi, S., (1998). Conceptual design of data warehouses from E/R schemes, in: *Proceedings of the Thirty-First Hawaii International Conference on System Sciences*. IEEE, pp. 334–343.

Gosain, A., Singh, J., (2015). Conceptual Multidimensional Modeling for Data Warehouses: A Survey, in:*Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA)*. Springer International Publishing, pp. 305–316.

Hachaichi, Y., Feki, J., (2013). An automatic method for the design of multidimensional schemas from object oriented databases. International Journal of Information Technology & Decision Making, Vol. 12, issue 6, pp. 1223–1259.

Inria, (2005). ATL transformation example: class to relational. https://www.eclipse.org/atl/atlTransformations/Class2Relational/ExampleClass2Relational[v00.01].pdf

Jensen, M.R., Holmgren, T., Pedersen, T.B., (2004). Discovering Multidimensional Structure in Relational Data, in: *Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 138–148.

Jovanovic, P., Romero, O., Simitsis, A., Abelló, A., Mayorova, D., (2014). A requirement-driven approach to the design and evolution of data warehouses. Information Systems, Vol. 44, pp. 94–119. https://doi.org/10.1016/j.is.2014.01.004

Khnaisser, C., Lavoie, L., Diab, H., Ethier, J.-F., (2015). Data Warehouse Design Methods Review: Trends, Challenges and Future Directions for the Healthcare Domain, in : *East European Conference on Advances in Databases and Information Systems (ADBIS)*. Springer, pp. 76–87. doi: 10.1007/978-3-319-23201-0_10

Khouri, S., Bellatreche, L., Jean, S., Ait-Ameur, Y., (2014). Requirements Driven Data Warehouse Design: We Can Go Further, in: *International Symposium On Leveraging Applications of Formal Methods, Verification and Validation (ISoLA)*. Springer, pp. 588–603.

Kumari, S., Yadav, P., (2015). Study of Influence of Data Mining & Data Warehousing, in: *Proceedings of National Conference on Innovative Trends in Computer Science Engineering (ITCSE)*. IJRRA, pp. 138–140.

Lämmel, R., Saraiva, J., Visser, J., (2005). Generative and Transformational Techniques in Software Engineering. Springer, p. 41.

Moody, D.L., Kortink, M.A.R., (2000). From enterprise models to dimensional models: a methodology for data warehouse and data mart design., in: *Proceedings of the International Workshop on Design andManagement of Data Warehouses*.Stockholm, Sweden, p. 5.

OMG, (2003). Common Warehouse Metamodel (CWM) Specification. Vol. 1, Version 1.1, pp. 8-3.

OMG, (2001). MDA Specifications. https://www.omg.org/mda/specs.htm

Phipps, C., Davis, K.C., (2002). Automating data warehouse conceptual schema design and evaluation., in: *Proceedings of the 4th International Conference on Design and Management of Data Warehouses*. Toronto, Canada, pp. 23–32.

Romero, O., Abelló, A., (2010). A framework for multidimensional design of data warehouses from ontologies. Data & Knowledge Engineering, Vol. 69, pp. 1138–1157. https://doi.org/10.1016/j.datak.2010.07.007

Rudra, A., Nimmagadda, S.L., (2005). Roles of multidimensionality and granularity in warehousing Australian resources data, in: *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*. IEEE, pp. 216b–216b.

Santos, M. Y., Oliveira e Sá, J., (2016). A Data Warehouse Model for Business Processes Data Analytics, in: *International Conference on Computational Science and Its Applications (ICCSA)*. Springer, pp. 241–256. doi: 10.1007/978-3-319-42092-9_19

Sapia, C., Blaschka, M., Höfling, G., Dinter, B., (2017). Extending the E/R Model for the Multidimensional Paradigm, in: *Proceedings of the Workshops on Data Warehousing and Data Mining: Advances in Database Technologies*. Springer, pp. 105–116.

Sehgal, S., Ranga, K. K., (2016). Translation of Entity Relational Model to Dimensional Model. International Journal of Computer Science and Mobile Computing, Vol. 5, issue 5, pp. 439–447.

Srai, A., Guerouate, F., Berbiche, N., Drissi, H., (2017). An MDA approach for the development of data warehouses from Relational Databases Using ATL Transformation Language. International Journal of Applied Engineering Research, Vol. 12, issue 12, pp. 3532–3538.

Taniar, D., Chen, L. (Eds.), (2011). Integrations of Data Warehousing, Data Mining and Database Technologies: Innovative Approaches. IGI Global.

Varga, M., (2002). A Procedure of Conversion of Relational into Multidimensional Database Schema. Journal of Computing and Information Technology, Vol. 10, pp. 69–84.

Vrdoljak, B., Banek, M., Rizzi, S., (2003). Designing web warehouses from XML schemas, in: *International Conference on Data Warehousing and Knowledge Discovery*. Springer, pp. 89–98.

Wiak, S., Drzymala, P., Welfle, H., (2012). Using ORACLE tools to generate Multidimensional Model in Warehouse. Przegląd Elektrotechniczny, pp. 257–262.

Winter, R., Strauch, B., (2003). A method for demand-driven information requirements analysis in data warehousing projects, in: *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*. IEEE.