A Report-Driven Approach to Design Multidimensional Models

Antonia Azzini, Stefania Marrara, Andrea Maurino, Amir Topalović

A Report-driven Approach to design Multidimensional Models

Antonia Azzini[1], Stefania Marrara[1], Andrea Maurino[2], and Amir Topalović[1]

[1] Consorzio per il Trasferimento Tecnologico, C2T, Milano, Italy
name.surname@consorzioc2t.it,
[2] Universitá degli studi di Milano Bicocca
Dipartiment of Informatics, Systemistics and Communication
Milano, Italy
maurino@disco.unimib.it

Abstract. Today, large organisations and regulated markets are subject to the control of external audit associations, which require the submission of a huge amount of information in the form of predefined and rigidly structured reports. The compilation of these reports requires the extraction, transformation and integration of data from different heterogeneous operational databases. This task is usually performed by developing a software ad hoc for each report, or by adopting a data warehouse and analysis tools, which are now established technologies. Unfortunately, the data warehousing process is notoriously long and error prone, and is therefore particularly inefficient when the output of the data warehousing is represented by a limited number of reports. This article presents "MMBR", an approach that can generate a multidimensional model from the structure of expected reports as data warehouse output. The approach is able to generate the multidimensional model and populate the data warehouse by defining a knowledge base specific to the domain. Although the use of semantic information in data storage is not new, the novel contribution of our approach is represented by the idea of simplifying the design phase of the data warehouse, making it more efficient, by using an industry-specific knowledge base and a report-based approach.

Keywords: multidimensional design, knowledge base, report driven methodology

1 Introduction

Business reporting is a strategic but heavy activity defined as "the public reporting of operating and financial data by a business enterprise," [1] or the regular provision of information to support decision-makers within organizations. Reporting is a fundamental part of the business intelligence and knowledge management activity and it is strongly required by audit organizations. Reporting activity can be realized in an ad hoc way by means of specific and complex

softwares, or by involving typical operations of extracting, transforming, and loading (ETL) procedures in coordination with a data warehouse.

Reports can be distributed in printed form, via email or accessed via a corporate intranet. In sectors as banking, reports are required by both National and European Central Bank organizations on regular basis, and all required reports must comply specific templates provided by the organizations themselves.

In particular, reports for auditing are often very specific, and their structure is usually imposed by the supervising organizations (e.g. Europan Central Bank, or the rating agency Moodys). The data included in the report are, in most cases, not useful for decision making activities due to the "control" nature of these reports. As a consequence, companies are forced to develop complex systems to compute data that are not useful for their business activities. In this context, it is clear that the need to develop a new approach able to support, in a fast and efficient way, the generation of reports is compelling.

In this scenario we propose to adopt a data warehouse as storage system for data, but we introduce a new approach aimed at designing the multidimensional models on the basis of the structure of the report itself in a (semi-)automatic way, in order to reduce the time needed to produce the report. A data warehouse essentially combines information from several heterogeneous sources into one comprehensive database. By combining all of this information in one place, a company can analyze its data in a more holistic way, ensuring that it has considered all the information available. Data warehousing also makes possible data mining, which is the task of searching for patterns in the data that could disclose hidden knowledge.

At the basis of a data warehouse lies the concept of the *multidimensional (MD) conceptual view of data*. The main characteristic of the multidimensional conceptual view of data is the fact/dimension dichotomy, which represents the data in an n-dimensional space. This representation facilitates the data interpretation and analysis in terms of *facts* (the subjects of analysis and related measures) and *dimensions* that represent the different perspectives from which a certain object can be analyzed.

Even if data warehousing benefits are well recognized by enterprises, it is well known that the warehousing process is time consuming, complex and error prone. Today the increasing reduction of the time-to-market of products forces enterprises to dramatically cut down the time devoted to the design ad the development of MD models, which support the evaluation of the key performance indicators of services and products.

There are different ways to design a data warehouse and many tools are available to help different systems to "upload" their data into a data warehouse for analysis purposes. However, all techniques are based on first extracting data from all the individual sources, by then removing redundancies and finally organizing the data into a format that can be interrogated.

As use case for presenting our approach we propose *securitization*, which is known by the literature as the financial practice of pooling various types of contractual debt such as residential mortgages, commercial mortgages, auto loans

or credit card debt obligations (or other non-debt assets which generate receivables) and selling their related cash flows to third party investors as securities [2]. Mortgage-backed securities, which are the case study presented in this paper, are a perfect example of securitization. By combining mortgages into one large pool, the issuer can divide the large pool into smaller parts based on each individual mortgage's inherent risk of default and then sell those smaller pieces to investors.

In this scenario we propose the MMBR (Multidimensional Model By Report) approach, which is able to automatically create the structure of a multi dimensional model (*MD* in the follow) and fill it on the basis of a knowledge base enriched with mapping information that depend on the specific application context. The preprocessing phase of the report (often a quite complex Excel file) is based on a table identification algorithm, which is able to extract the information needed to define the MD structure of the data warehouse. The approach has been tested in the context of financial data with the aim to automatically create the reports required by the Italian National Bank and by the European Central Bank.

The term "by report" refers to the capability of our solution to create a multidimensional model starting from a given report (typically expressed as Microsoft Excel files) that must to be filled with real data. MMBR is also able to generate the relational data structure related to the created MD, and it is also in charge of filling both fact and dimensional tables supported by domain ontologies and by mapping information to the operational sources.

In the literature there are many methodologies for creating MDs starting by requirements, but this is the first attempt to define an approach for creating a MD model starting directly from the structure of the final reports only.

The remaining of the paper is organized as follows: Section 2 introduces the state of the art. Section 3 presents the proposed approach, while Section 4 describes the knowledge base that is a key element in the MMBR methodology, and the report graph. In Section 5, the table identification algorithm is presented, while Section 6 describes the creation of the MD model. A real example taken from the financial domain is then reported in Section 7. Conclusions and final remarks are reported in Section 8.

2 Related Work

In the literature several approaches for creating conceptual MD schema from heterogeneous data sources have been presented. According to [3], these approaches can be classified into three broad groups:

- *Supply-driven*: starting from a detailed analysis of the data sources these techniques try to determine the MD concepts. By using this approach, it is possible to waste resources by specifying unnecessary information structures, and by not being able to really involve data warehouse users. See for instance [4–6].

– *Demand-driven*: These approaches focus on determining the MD requirements based on an end-user point of view (as typically performed by other information systems), and mapping them to data sources in a subsequent step (see for example [7, 8]).
– *Hybrid approaches*: Some authors (see for example [9–11]) propose to combine the two previously presented approaches in order to harmonize, in the design of the data warehouse, the data sources information with the end-user requirements.

All the methodologies available in literature, however, have the goal to create a MD model as general as possible in order to allow the generation of any report. This assumption requires a lot of effort in both the warehouse conceptualization phase and in the ETL procedure design and development. In several industrial contexts, there is the need to produce a limited number of reports only and, sometimes, with a very strict and well defined structure due to auditing rules or for specific business requirements. In the finance domain, for example, banks are required by central authorities and rating agencies to produce very specific reports related to the securization activities they perform.

In the field of the Semantic Web, Bontcheva and colleague [12] present an approach for the automatic generation of reports from domain ontologies encoded in Semantic Web standards like OWL. The novel aspects of their so-called "MIAKT generator" are in the use of the ontology, mainly the property hierarchy, in order to make it easier to connect a generator to a new domain ontology.

Another interesting approach is presented in [13], where the authors propose a framework for designing a semantic data warehouse. They represent the topic of analysis, measures and dimensions in the requirements. In such an approach they derive the MIO (Multidimensional Integrated Ontologies) along with the knowledge from external ontology sources and domain ontologies. Nebot and colleagues [13] propose an approach in which a Semantic Data Warehouse is considered as a repository of ontologies and other semantically annotated data resources. Then, they propose an ontology-driven framework to design multidimensional analysis models for Semantic Data Warehouses. This framework provides means for building an integrated ontology, called the Multidimensional Integrated Ontology (MIO), including the classes, relationships and instances representing the analysis developed over dimensions and measures.

Romero and colleague [14] introduce a user-centered approach to support the end-user requirements elicitation and the data warehouse multidimensional design tasks. The authors explain how the feedback of a user is needed to filter and shape results obtained from analyzing the sources, and eventually produce the desired conceptual schema. In this scenario, they define the AMDO (Automating Multidimensional Design from Ontologies) method, aimed at discovering the multidimensional knowledge contained in the data sources regardless of the users requirements.

The implemented process derives the multidimensional schema from a conceptual formalization of the domain, by defining a fully automatic supply-driven approach working at the conceptual level. Differently from the idea implemented

in this work, based on the report as starting point, they consider the queries as first. Such an identification comes from the categorization they introduced from a first analysis, that divides different contributions within a so-called demand "driven", "supply-driven" or "hybrid" framework. The first one focuses on determining the end-user multidimensional requirements to produce a multidimensional schema; the second one starts from a detailed analysis of the data sources to determine the multidimensional concepts in a re-engineering process. The latter refers to the approaches that combine the two previous frameworks.

Another interesting work aimed at supporting the multidimensional schema design is given by [15], in which the authors propose an extension of their previous work [16]. They follow a hybrid methodology where the data source and the end-user requirements are conciliated at the early stage of the design process, by deriving only the entities that are of interest for the analysis. The requirements are converted from natural language text into a logical format. The concepts in each requirement are matched to the source ontology and tagged. Then, the multidimensional elements such as fact and dimensions are automatically derived using reasoning.

On the other hand, Benslimane and colleague [17] define a contextual ontology as an explicit specification of a conceptualization, while Barkat [18] proposes a complete and comprehensive methodology to design multi-contextual semantic data warehouses. This contribution is aimed to provide a context meta model (language) that unifies the definitions provided in Database literature. This language is considered as an extension of OWL, which is the standard proposed by the W3C Consortium [19] to define ontologies. It is defined by the authors in order to provide a contextual definition of the used concepts, by offering an externalization of the context from the ontology side.

Pardillo and colleagues [20] present an interesting approach aimed at describing several shortcomings of the current data warehouse design approaches, showing the benefits of using ontologies to overcome them. This work is a starting point for discussing the convenience of using ontologies in the data warehouse design. In particular the authors present a set of situations in which ontologies may help data warehouse designers with respect to some critical aspects. Examples are the requirement analysis phase, where new concepts and techniques meaning should be clarified to be used by stakeholders, or the phase of reconciling requirements and data sources.

As also considered in this approach, it is important to underline that a domain specific ontological knowledge allows to enrich a multidimensional model in aspects that have not been taken into account during the requirement analysis or data-source alignment phases, as well as other aspects, like for example the application of statistic functions in order to aggregate data. Table 1 summarizes the main concepts explained into the literature reported by the above mentioned contributions.

Author	Reference	Principal Explained Topics	Work Description
Bontcheva et al.	[12]	Semantic Web ontologies	Report automatic generation from property hierarchy encoded in Semantic Web (OWL).
Nebot et al.	[13]	Semantic Multidimensional	Multidimensional Integrated Onto. definition with Semantic DW as repository.
Romero et al.	[14]	user-centered AMDO	User-centered app. to user support and automated multidim. design from ontologies.
Thenmozhi et al.	[15]	hybrid approach	Matching among req. concepts to the source ontology and tagging process.
	[16]	data source end-user req.	Conciliation among sources and user req. for interesting entity extraction.
Benslimane et al.	[17]	contextual ontoloty	Contextual ontology definition and explicit specification of a conceptualization.
Barkat et al.	[18]	multi-context semantic DW	Context meta model language as OWL extension for semantic DW definition.
Pardillo et al.	[20]	Ontologies DW design	Use ontologies for DW design to overcome the shortcomings of DW design.

Table 1. Related Work Summary.

3 Description of the approach and outline of the architecture

The MMBR approach main phases are shown in Figure 1: 1) Table Processing (TP), 2) Row and Column Header Identification and Extraction (RCHIE), 3) Ontology Annotation (OA), 4) Management of Non-Identified labels (MNL), 5) creation of the MD model, 6) ETL Schema Generation (ETL), and, finally, 7) the Report Generation (RG). The input of the TP phase is the template file that has to be filled with the data extracted from an Operational Data Base (ODB). In the TP phase the preprocessing of the template is performed by removing icons and other figures, moreover all terms in the schema are lowered and comment and description fields are removed.

The RCHIE phase is based on the *table identification algorithm* aimed at identifying and extracting the row and column headers in the template. The details of the table identification algorithm are presented in Section 5.

The list of terms recognized in the reports by the table identification algorithm is then annotated on the basis of a knowledge base (see Section 4). This phase produces two lists; the first one is the list of identified terms annotated w.r.t. the knowledge base, the second one is the list of terms that are not annotated. There are several possible reasons of failure for the annotation activity. The most frequent reason is that a given term may be not included in the knowledge base because it is not relevant to the domain (e.g "Total"). It is also possible that a term is not annotated because it is a composition of different terms (such as "MortageLoan" or "DelinquentLoan")[3]. Moreover some terms are written in a language different from English (e.g. "garantito" that means guaranteeded in Italian). In all these cases, not annotated terms are manually checked and, if relevant, added to the ontology by defining the corresponding rdf:label prop-

[3] the description of these terms is reported in Section 7

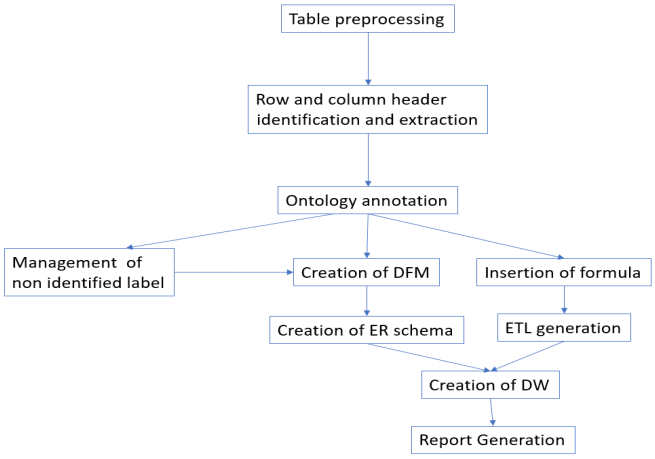

Fig. 1. Overall representation of the approach.

erty. The annotated list of terms is the input for the creation of the dimensional fact model (see Section 6). This logical model is translated into a relational star schema. In this phase the relational database is filled with data coming from the ODB. This activity is performed on the basis of the mapping rules included in the knowledge base. This activity is fully described in Section 4. Once the data warehouse is filled, the report generation phase is in charge of populating the report template by translating annotation of the report graph into SQL queries executed over the data warehouse. Query results are then inserted in the report template to generate the final output.

The architecture supporting the MMBR approach is represented in Figure 2. The Annotation Editor is in charge of the first three phases of the MMBR approach, by removing non relevant strings and images from the input file (e.g. logo, comments), and by identifying the terms that are annotated w.r.t. the KB and by creating the report Graph. The Schema builder is the software component aimed at creating the logical relational description of the MD model. The ETL generator is in charge of extracting, on the basis of the Report Graph and the KB, the information necessary to create the extraction-transformation-load data from the ODB to the data warehouse. The Knowledge base manager is in charge of managing and evolving the knowledge base. Any popular tool as, for instance, Protege[4] may be used for the KB creation. The Report generator finally allows to fill the report template by capturing the data from the DW according to the queries build on the base of the annotation included in the Report Graph.

[4] `https://protege.stanford.edu/`

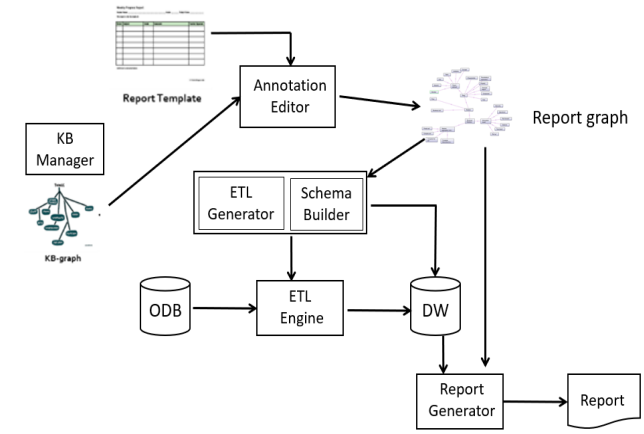

Fig. 2. Representation of the Overall architecture.

4 The MMBR Knowledge Base and the Report Graph

At the core of the proposed approach lies the creation of the knowledge base KB, which includes:

- the set of MD concepts and relations (fact, dimensions, measures, attributes);
- the list of terms adopted in the specific application domain (eg. ecommerce, bank securitization,...);
- the Operational DataBase (ODB) schema.

In order to create a knowledge base that could be easily shared in the financial domain we started by using an already existing ontology and only in case of need we created new concepts.

The ontology we used as starting point for creating new MD concepts in the KB is a simplified version of the *data cube vocabulary*[5], i.e., a W3C recommendation for modeling multidimensional data. The top level representation of the defined KB is shown in Figure 3.

The MD concepts are organized as follows. A fact (the event that is the target of a report, e.g., a sell in a e-commerce domain, a loan in the bank domain) is described by a set of measures and can by analyzed by considering its dimension and descriptive attributes. In the data cube vocabulary dimensions, measures and descriptive attributes are described by the concept *component properties instances*. Dimensions, measures and descriptive attributes are *terms* of the application domain and they are defined by the human (domain) expert trough the Knowledge Base. In fact, the KB annotation specifies if a KB component refers

[5] `https://www.w3.org/TR/vocab-data-cube/`

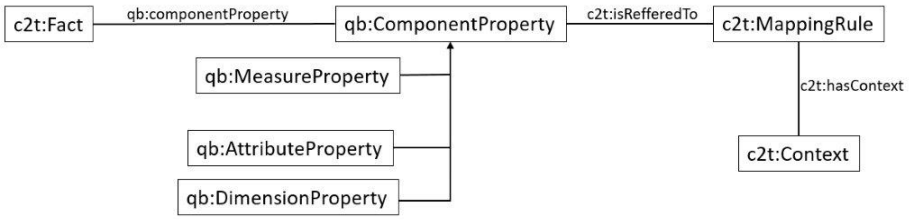

Fig. 3. Top level representation of the Knowledge Base.

to a fact, a measure or to a dimension. Such elements are then compared with each label extracted from the Excel file in order to define fact, measures and dimensions of the corresponding model. In order to build a KB related to the e-commerce domain, it is possible, for example, to use concepts described in the *good relation* section of the vocabulary [6]. In this scenario instances of *Dimension-Properties* are *gr:ProductOrService, gr:Brand*, while instances of *dq:Measure* are *gr:UnitPriceSpecification, gr:amountOfThisGood.* If no vocabulary is available, a new, ad-hoc, vocabulary has to be defined as first (as also reported in Section 7).

The concept *qd:ComponentProperty* can have one or more *rdf:label properties* associated to, that represent the references to the instances of the target concept. For example the dimension *gr:Brand* may be labeled as *"NameOfProduct"* or *"BrandName"*. During the annotation phase, labels are used to associate terms of the report to the application domain concepts.

In order to populate the MD model it is necessary to know how the *qb:componentProperties* are described in the operational DB (ODB). This mapping is described in the KB itself, by means of the *c2t:mappingRule* concept, which associates a *c2t:mappingFormula* related to a given instance of the *qb:ComponentProperties*. The *c2t:mappingFormula* contains a reference to some tables of the ODB and a query predicate over their tuples.

For example, in a bank scenario we can assume that the TLoan table of the ODB contains all information related to loans. A loan with a *fixed rate* (i.e., a loan where the interest rate on the note remains the same through the term of the loan) can be represented in the ODB by the predicate *InterestRate=1*, while a floating rate can be described by the predicate *InterestRate>1*. The formula *c2t:mappingFormula* includes the references to the TLoan table and the predicate regarding the *InterestRate*.

The concept *c2t:context* in Figure 3 assumes value when the reports provided by different audit authorities contain different mapping formulas for the dimension *dq:componentProperties*. For example, a given audit authority may classify a company as "small" if the employee number does not reach 10, while

[6] `http://www.heppnetz.de/projects/goodrelations/`

for another authority a company is small if it has less than 15 people employed. In this case we will have two different *c2t:MappingFormula*.

4.1 Report Graph

Starting from such a schema the ontology has been defined by using the Protégé editor [21]. Protégé is a free, open source, ontology editor and a knowledge management system with an user friendly graphic interface. It also includes some classifiers to validate that models are consistent and to infer new information based on the analysis of an ontology [21, 22].

The report graph is a rdf representation of the report template, which includes the annotation of each value cell in terms of KB elements.

The top level of the ontology represents the description of the structure of the reports. It includes a set of *qd:observation* elements (i.e., each *cell* of the report), each of them characterized by the two properties (*c2t:hasPosX* and *c2t:hasPosY*) representing their coordinates in the report. An observation element may contain a measure (*c2t:hasValue*) if it contains values from the ODB aggregated by means of an aggregation operator (e.g. *sum, avg...*) as in the traditional data warehouse. Moreover an observation element includes the dimensions, which are used for the analysis. Both dimensions and measure are fully described in the KB. An example of a report graph annotation is as follows:

```
eg:o1 a qb:Observation;
 c2t:hasValue fibo:outstandingPrincipal;
     c2t:isAggregatedBy "Sum";
 c2t:hasPosX "C";
 c2t:hasPosY "9";
     c2t:hasDimensionalProperty ontoLoan:Performing;
     c2t:hasDimensionalProperty ontoLoan:Mortage;
    .
```

In the example, the cell whose coordinates are column "c" and row "9" will contain the sum of *Outstanding Principal* extracted from the loans that are at the same time *Performing* and *with a mortgage guarantee*.

5 Table Identification

Reports required by audit organizations are usually structured documents represented by tables. Each table can be divided into different areas or sections, according to their structure. Thus, being able to correctly identify the inner structure of the table is important to find the concepts relevant to the MD models generation. As discussed in the introduction, a multidimensional model represents the data into a n-dimensional space; under this perspective each report can be considered as one of the possible hyperplanes slicing the n-dimensional cube of data. To represent this hyperplane into a bi dimensional table it is necessary

Time	Thailand		Japan		Total
	Food	NonFood	Food	NonFood	
2006	2400	2200	11000	5000	20600
2007	4500	3200	12000	6000	25700
2008	5600	2900	10000	5500	24000
Total	12500	8300	33000	16500	70300

Fig. 4. Example of report.

to reduce the dimensions. In Figure 4 the MD is composed by three dimensions (time, nations and type of sold goods) that are "flattened" into a bi-dimensional space by associating the values of type of sold goods (Food and non Food) to the nation dimension. According to this assumption, rows and columns header may contain dimensions, values of dimensions and measures of the MD.

In our approach, during the RCHIE phase a table was assumed as composed by three types of cell: respectively *textual*, *data* and *schema* ones. Figure 5 shows the general schema. The cell identifiers are represented by the couple $< X, Y >$, as reported in the table shown in Figure 5.

<1,1>	abc	abc				<1,n>
			C_Header	C_Header	...	C_Header
abc			C_Header	C_Header	...	C_Header
abc	R_Header	R_Header	<4,4>			
	R_Header	R_Header			...	
	...	...				
<m,1>	R_Header	R_Header				<m,n>

Fig. 5. Example of table used by the Table Identification algorithm.

The table may contain several types of cells, as defined in the following way:

- **textual-cell**: this cell is not used for table annotation, these cells are shown in grey in Figure 5, and they may contain simple text.
- **data-cell**: it contains data that are computed on the basis of the MD model. These cells are represented by the white colour in the figure.
- **schema-cell**: it specifies properties over a set of data-cells. It is shown in dark grey in the figure. This cell defines the header $h =< x, y >$ of a set of data cells, by specifying some semantic aspects (i.e., the measure or a value on a dimension).

Rows and columns are identified in order to extract the labels corresponding, respectively, to measures, dimensions, instances of the dimensions, etc. (for instance not relevant information as the $TOTAL$ value shown in Figure 6). These

labels represent the input of the annotation phase, which produces the annotated list of terms as output.

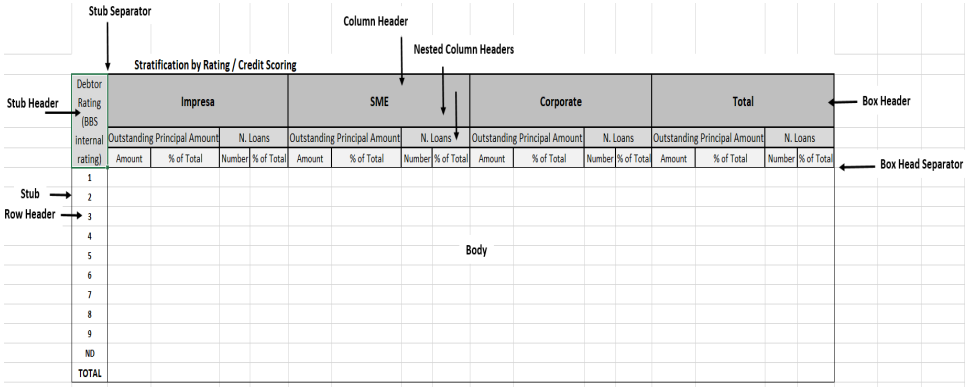

Fig. 6. Example of Table.

In the literature different table identification algorithms aimed at handling the tables structure have been proposed [23]; in our work the focus is identifying and removing multi spanning cells. An example is reported in Figure 6, where one of the reports related to securitization is shown. The *Stub Header* cell details information w.r.t. the measures *Loan* and *Oustanding Principal* of different types of companies, as Corporate, SME and "Impresa" (it refers to retail companies in the Italian jargon). Measures, names and instances of dimensions are placed in the *Box Header* and/or the *Stub* areas as headers, and they are used to index the elements located in the *Body* area of the table. The *Stub Header* may also contain a header naming or describing the dimensions located in the stub. The outcomes of the table identification algorithm are shown in Figure 7 where all data-cells are semantically associated to their row and column headers.

Finally, the RCHIE phase extracts a list of unique terms that are in the column and row headers. These terms are then annotated by means of the knowledge base, by evaluating the labels related to the application domain concepts and the terms extracted from the report table. In this task, the domain expert are forced to take action only for those text strings that do not have the corresponding ontology term (an example is given by the string `performing` and the term `in Bonis`).

6 The Dimensional Fact Model

Generally speaking, the Dimensional Fact Model (DFM) [24] is a graphical conceptual model for data warehouse aimed to:

	Impresa	Impresa	Impresa	Impresa	SME	SME	SME	SME	Corporate	Corporate	Corporate	Corporate	Total	Total	Total	Total
	Outstanding Principal Amount	Outstanding Principal Amount	N. Loans	N. Loans	Outstanding Principal Amount	Outstanding Principal Amount	N. Loans	N. Loans	Outstanding Principal Amount	Outstanding Principal Amount	N. Loans	N. Loans	Outstanding Principal Amount	Outstanding Principal Amount	N. Loans	N. Loans
Debtor Rating (BBS internal rating)	Amount	% of Total	Number	% of Total	Amount	% of Total	Number	% of Total	Amount	% of Total	Number	% of Total	Amount	% of Total	Number	% of Total
1	posx=3, poy=6	posx=4, poy=6	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2	posx=3, poy=7	posx=4, poy=7	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3	posx=3, poy=8	posx=4, poy=8	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4	posx=3, poy=9	posx=4, poy=9	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5	...	...	...	...	...	...	...	...	Body	...	...	...	...	...	...	...
6	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
ND	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
TOTAL	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Fig. 7. Example of flattened table.

- effectively support the conceptual project,
- define an environment over which intuitively define the queries of a user,
- allow the interaction between the designer and the final user for specific request refinements,
- produce useful and non ambiguous documentation.

The conceptual representation deriving from DFM is defined by a set of fact schema. The basic elements modeled by such a schema are the so called fact, measures and dimensions. A fact is useful for the decisional process: it models a set of events coming from the analysis context; it needs to be time evolving. A measure represents a numeric property of a fact, and it describe a quantitative aspect useful for further analysis. Finally, a dimension is a property with a finite domain of a fact and it describes one of the analysis coordinates.

According to the literature [4] a dimensional scheme consists of a set of fact schemes. The components of fact schemes are facts, measures, dimensions and hierarchies. A fact is a focus of interest for the decision-making process; typically, it models an event occurring in the enterprise world (e.g., sales and shipments). Measures are continuously valued (typically numerical) attributes which describe the fact from different points of view; for instance, each sale is measured by its revenue. Dimensions are discrete attributes which determine the minimum granularity adopted to represent facts; typical dimensions for the sale fact are product, store and date. Hierarchies are made up of discrete dimension attributes linked by one-to-one relationships, and determine how facts may be aggregated and selected significantly for the decision-making process. The dimension in which a hierarchy is rooted defines its finest aggregation granularity; the other dimension attributes define progressively coarser granularities. Hierarchies may also include non-dimension attributes. A non-dimension attribute contains additional information about a dimension attribute of the hierarchy, and is connected by a one-to-one relationship (e.g., the address); unlike dimension attributes, it cannot be used for aggregation. At a conceptual level, distinguishing

between measures and dimensions is important since it allows the logical design to be more specifically aimed at the efficiency required by data warehousing applications.

6.1 Queries Representation

In general, querying an information system means linking different concepts through user defined paths in order to retrieve some data of interest; in particular, for relational databases this is done by formulating a set of joins to connect relation schemes. On the other hand, a substantial amount of queries on DWs are aimed at extracting summary data to fill structured reports to be analysed for decisional or statistical purposes. Thus a typical DW query can be represented by the set of fact instances, at any aggregation level, whose measure values have to be retrieved.

The sets of fact instances can be denoted by writing fact instance expressions. The simple language proposed in the literature [24] is aimed at defining, with reference to a dimensional scheme, the queries forming the expected workload for the DW, to be used for logical design; thus, it focuses on which data must be retrieved and at which level they must be consolidated.

A fact instance expression has the general form:

```
<fact instance expression> ::= <fact name>
              (<pattern clause> ; <selection clause>)
<pattern clause> ::= comma-list of <pattern elements>
<pattern elements> ::= <dimension name> |
                       <dimension name>.<attribute name>
<selection clause> ::= comma-list of <predicate>
```

The pattern clause describes a pattern. The selection clause contains a set of Boolean predicates which may either select a subset of the aggregated fact instances or affect the way fact instances are aggregated. If an attribute involved either in a pattern clause or in a selection clause is not a dimension, it should be referenced by prefixing its dimension name.

6.2 MD creation and population on MMBR Approach

The Dimensional Fact Model (DFM)[4] approach has been used in our solution to describe the MD model. The list of annotated terms and the KB are the only two elements necessary to design and populate the MD. Each annotated term of the list is enriched by its type or subclass in order to understand if it is a measure, a dimension or an instance of a dimension. This can be realized by means of a set of SPARQL[7] queries over the KB (an example of query is shown in Section 7) generated by the Schema builder component. With this information it is possible to create the DFM and the corresponding logical relational schema

[7] https://www.w3.org/TR/rdf-sparql-query/

by means of the original methodology proposed in [4]. The relational schema is then populated according to the mapping information defined in the knowledge base.

All dimensional tables are populated with the instances defined in the KB, while the fact table is defined in a two steps procedure. In the first step all instances of the facts (e.g. sell or loan) are selected from the ODB by taking into account only the measures available in the annotated list. The second step is in charge of connecting the fact table with the dimensional tables. An Update query is executed to associate each instance of the fact table with the instances of the dimensions tables. Even in this case the KB plays a strategic role since it allows to extract the mapping formula at the basis of the SPARQL queries (see Section 7).

7 Case Study

After a brief introduction over the ontology defined in this work, an example of two financial reports that have to be filled is reported, together with an example of the mapping rules and the defined sparql queries. The implemented software prototype, supporting the MMBR approach, and a brief discussion about the methodology are finally presented.

7.1 The Considered Reports

The scenario motivating the definition of a report driven approach for the design of multidimensional models is related to the financial domain. In particular, the reporting activity of securitization was analyzed.

Applying the MMBR approach in this context, the first activity to be faced is the generation of the domain KB and vocabulary. The literature proposes two different vocabularies that partially describe the loan domain: FIBO [8] and Schema.org [9]. FIBO, a Financial Industry Business Ontology, contains the loan terms definitions without any further specification. Schema.org does not contain a full exhaustive specification of the securitization domain, but it includes the LoanOrCredit concepts[10] only. The KB defined in this work to describe the securitization domain is the ontology *OntoLoan*. During the KB definition, domain experts are in charge to define the main terms and concepts. The OntoLoan ontology is not freely available, since it is covered by the company's intellectual properties. However, the top level of OntoLoan is shown in Figure 8, while Figure 9 shows an example of securization report. Note that all private data related to the bank owning the report are removed for privacy issues, while the values for different kinds of loans are reported.

The term *Performing Loans* refers to those loans that have no overdue interest payments, or with unpaid installments due, even if under the maximum

[8] `https://www.edmcouncil.org/financialbusiness`
[9] `https://schema.org`
[10] `https://schema.org/LoanOrCredit`

number of delay days outstanding (which changes according to the securitization contract terms). *Delinquent Loan* refers to the loans close to default, i.e., to unpaid installments due to a delay in payments close to the limit on the delay of days overdue. *Defaulted Loans* refers then to loans with significant delays in payments.

Any kind of loan is further divided according to other features, generating the definition of *Mortgage Loan*, *Guaranteed Loan*, i.e. loans insured not by mortgages but by other guarantees (e.g., pledges), and, finally, *Unguaranteed Loan*, i.e. not insured.

The first phase of the MMBR approach removes text fields that do not carry relevant information from the report. An example of removed test is the string "A. PORTFOLIO OUTSTANDING BALANCE". The annotation tool removes the cell spanning from the table of Figure 9, creating the table structure shown in Figure 10. The data-cell in position $< 3, 3 >$ represents the aggregation of the values of *Outstanding Principal* of loans that are both performing and able to pay off the loan even in case of default of the borrower. The value in the cell with position $< 3, 4 >$ represents the aggregation of the *Outstanding Principal* of loans that are both performing and guaranteed. The mapping rule MR1 is described as follows:

```
MR1  :mappingRule1 rdf:type :MappingRule ;
  :hasContext :context1;
              :hasTargetDimension :defaulted ;
              :hasMappingFormula "rating_34=10 and
               rating5 between 1 and 7"^^xsd:string
          .
```

The rule MR1 indicates that for the context *context1* the *defaulted* value (instance of performance category) is associated to the loans having a *rating34* equals to 10 and a *rating5* between 1 and 7.

7.2 SPARQL Queries Definition

With this first activity the following list of terms related to the domain is extracted *loan:Performing, loan:Mortgage, loan:Guaranteed, loan:Unguaranteed, loan:Delinquent, loan:Defaulted, loan:DelinquentInstalments, loan:OutstandingPrincipal, loan:AccruedInterest, loan:PrincipalInstalment, loan:InterestInstalment.*. For each element of the list, MMBR retrieves from the KB the name of the dimensions or measures related to it, by means of *SPARQL* queries.

An example of query is the following.

```
SELECT distinct  ?x, ?p
WHERE {
loan:Guarantee rdf:type ?x.
?x rdfs:subClassOf ?p
 }
```

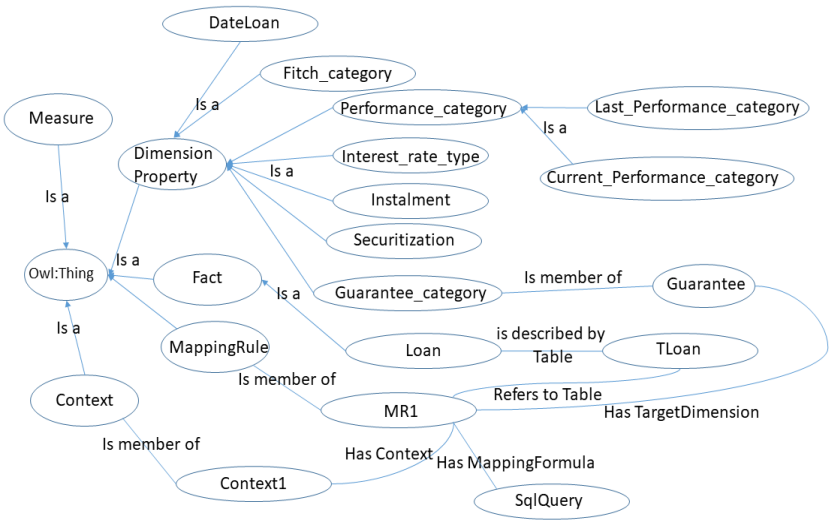

Fig. 8. The top level representation of OntoLoan.

				Delinquent instalments			Total
		Outstanding Principal	Accrued Interest	Principal inst.	Interest inst.	Total	
Performing Loans	Mortgage Loans						
	Guaranteed Loans						
	Unguaranteed Loans						
	Total Unsecured						
Delinquent Loans	Mortgage Loans						
	Guaranteed Loans						
	Unguaranteed Loans						
	Total Unsecured						
Defaulted Loans	Mortgage Loans						
	Guaranteed Loans						
	Unguaranteed Loans						
	Total Unsecured						
	TOTAL						

Fig. 9. An example of report template.

				Delinquent instalments	Delinquent instalments	Delinquent instalments	Total
		Outstanding Principal	Accrued Interest	Principal inst.	Interest inst.	Total	
Performing Loans	Mortgage Loans	pos=3, pos=3					
Performing Loans	Guaranteed Loans	pos=3, pos=4					
Performing Loans	Unguaranteed Loans	pos=3, pos=5					
Performing Loans	Total Unsecured						
Delinquent Loans	Mortgage Loans						
Delinquent Loans	Guaranteed Loans						
Delinquent Loans	Unguaranteed Loans						
Delinquent Loans	Total Unsecured						
Defaulted Loans	Mortgage Loans						
Defaulted Loans	Guaranteed Loans						
Defaulted Loans	Unguaranteed Loans						
Defaulted Loans	Total Unsecured						
	TOTAL						

Fig. 10. An example of flattened report.

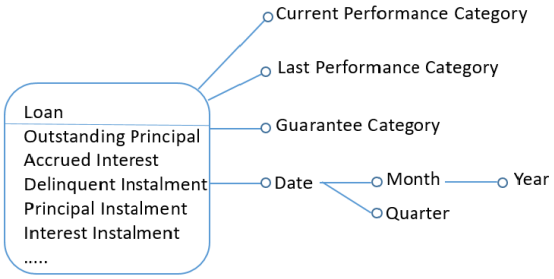

Fig. 11. Dimensional Fact Model Schema Example.

The example query is able to recognize, as shown in Figure 8, that the *loan:Guarantee* element is member of an entity named *Guarantee_Category*, which is a subclass of *qb:DimensionProperty*. Figure 8 also shows the query properties. After the identification of the measures and the dimensions, the DFM is designed as shown in Figure 11, according to Literature (see [25]) for the schema definition.

The DFM is then translated into a relational schema, whose instance is created in a relational DBMS as described in Section 6, and shown in Figure 12.

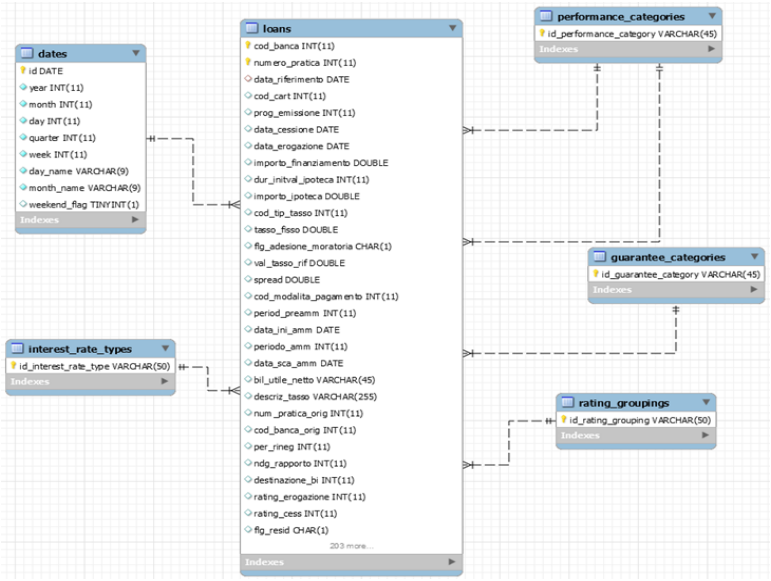

Fig. 12. Data Warehouse Schema.

In order to update the fact table, it is possible to retrieve the mapping formula in the KB, by means of a SPARQL query. For example to update the *guaranteed loan*, first we retrieve from the KB the corresponding mapping formula by using the following query:

```
SELECT ?table, ?rule
WHERE {
?s rdf:type loan:MappingRule.
?s loan:hasContext loan:context1.
?s  loan:hasTargetDimension loan:Guarantee.
?s  loan:refersToTable ?table.
    ?s  loan:hasMappingFormula ?rule.
}
```

The result is the following predicate:

```
TLoan
VAL_IPOTECA = 0 and (flag_garanzia_confidi='Y' or
(importo_pegno + importo_garan_pers) > 0)^^string
```

The corresponding update query using IBM DB2 SQL is:

```
UPDATE Fact
SET id_Guarantee_category=
(SELECT Guarantee
     FROM fact join odb.TLoan
     WHERE fact.id=obd.TLoan.id and
     VAL_IPOTECA = 0 and
     (flag_garanzia_confidi='Y' or (importo_pegno + importo_garan_pers) > 0)
)
```

The last phase is related to the generation of the report. Let us assume to generate a report where the cell corresponding to the coordinates F and 22 contains the sum of interest installments of all defaulted and guarantee loans. In the report graph the cell F22 is annotated as follows:

```
#Cell F22
eg:o36 a qb:Observation;
    c2t:hasDimensionalProperty ontoLoan:Defaulted;
    c2t:hasDimensionalProperty ontoLoan:Unguaranteed;
    c2t:hasValue ontoloan:InterestInstalment;
    c2t:isAggregatedBy "Sum"
    c2t:hasPosY "22";
    c2t:hasPosX "F";
```

The ReportGenerator module creates the aggregate SQL query able to compile the cell. First of all, it retrieves the correct aggregation operator to be used, i.e. here it is a *sum*, then by querying the rdf fragment of the report

graph it discovers that the values to be aggregated belong to the attribute *InterestInstalment*. To create the FROM statement of the query the ReportGenerator interrogates the ontology finding that both *ontoLoan:Defaulted* and *ontoLoan:Unguaranteed* concepts are instances of *ontoLoan:DimensionalProperty* and that both attributes are included in the Loans table. Thus, the FROM condition includes the Loans table only. The WHERE condition is composed creating a conjunctive predicates of *current_performance_category=Defaulted* and *guarantee_category=Unguaranteed*. The final SQL query for computing the value of the Cell F22 is the following:

```
SELECT sum(InterestInstalment)
FROM Loans
WHERE current_performance_category="Defaulted" AND
  guarantee_category="Unguaranteed"
```

The report generated is show in Figure 13.

		Outstanding Principal	Accrued Interest	Delinquent instalments			Total
				Principal inst.	Interest inst.	Total	
		a	b	c	d	e = c + d	f = a + b + e
1	**Performing Loans**	**784.388.232,49**	**2.117.994,43**	**1.186.477,11**	**271.017,76**	**1.457.494,87**	**787.963.721,79**
	Mortgage Loans	436.453.328,80	1.490.251,88	487.789,26	143.702,04	631.491,30	438.575.071,98
	Guaranteed Loans	256.920.537,40	406.690,11	574.366,19	95.721,59	670.087,78	257.997.315,29
	Unguaranteed Loans	91.014.366,29	221.052,44	124.321,66	31.594,13	155.915,79	91.391.334,52
	Total Unsecured	347.934.903,69	627.742,43	698.687,85	127.315,72	826.003,57	349.388.649,81
2	**Delinquent Loans**	**10.015.697,28**	**55.716,45**	**370.642,23**	**101.403,52**	**472.045,75**	**10.543.459,48**
	Mortgage Loans	5.677.758,73	47.864,60	26.801,82	74.087,28	100.889,10	5.826.512,43
	Guaranteed Loans	3.463.351,91	6.496,19	236.242,02	21.191,46	257.433,48	3.727.281,58
	Unguaranteed Loans	874.586,64	1.355,66	107.598,39	6.124,78	113.723,17	989.665,47
	Total Unsecured	4.337.938,55	7.851,85	343.840,41	27.316,24	371.156,65	4.716.947,05
3	***Collateral Portfolio (1+2)***	**794.403.929,77**	**2.173.710,88**	**1.557.119,34**	**372.421,28**	**1.929.540,62**	**798.507.181,27**
4	**Defaulted Loans**	**9.468.456,24**	**53.897,77**	**606.766,09**	**141.486,02**	**748.252,11**	**10.270.606,12**
	Mortgage Loans	5.709.138,55	40.931,49	180.150,10	64.062,84	244.212,94	5.994.282,98
	Guaranteed Loans	3.236.549,82	9.426,95	313.545,45	67.325,72	380.871,17	3.626.847,95
	Unguaranteed Loans	522.767,87	3.539,32	113.070,54	10.097,46	123.168,00	649.475,19
	Total Unsecured	3.759.317,69	12.966,28	426.615,99	77.423,18	504.039,17	4.276.323,14
	TOTAL (3+4)	**803.872.386,01**	**2.227.608,65**	**2.163.885,43**	**513.907,30**	**2.677.792,73**	**808.777.787,39**

Fig. 13. Report generated.

7.3 Discussion

The MMBR methodology and related techniques supporting the creation of multidimensional model able to produce a given (set of) report(s). The term "by report" refers to the capability of our solution to create a multidimensional model starting from a given report (typically expressed as Microsoft Excel file) that must to be filled with real data. MMBR is also able to generate the relational data structure related to the created a MD model and it is also in charge of filling both fact and dimensional tables thanks to the use of domain ontologies enriched with mapping information to the operational sources. According to the

literature, this is the first attempt to define a methodological approach for creating MD starting directly from the reports only. The methodology starts with the acquisition of the excel file and, thanks to an table identification algorithm, then it extracts rows and headers representing domain concept from the report, and the extracted terms are annotated by using a domain ontology enriched with md concepts. The ontological terms are finally used to design the MD.

One of the most important elements in our methodology is the use of a domain ontology, applied in order to annotate terms available in the report. Such ontological terms are used to identify fact, dimensions and instances of dimension that allow the creation and population of the MD model, by generating the Dimensional Fact model and the Entity Relational schema.

A prototype supporting the proposed methodology has been developed in the experimental session. The report graph is created by using *Protegé*, which is adopted to support the RCHIE phase in a semiautomated way. The phases of the methodology involved into the creation and population of the data warehouse are supported by a custom tool we named "CreDaW", (Create a Data Warehouse).

The tool, as described in Section 6, creates the DW schema by querying the Report graph. The relational tables implementing the DW schema are populated by querying the KB as reported in Section 7. The prototype is developed and tested on an Intel I7-6700 personal computer with 3.4GHz, 16GB ram and 1TB hdd and it is able to create and populate a DW in two different relational database management systems, MySQL and IBM DB2.

The data warehouse population phase requires around 13 seconds for loading more than 400.000 loans.

The last phase (the "Report generation") of the methodology is totally automated by the tool *ReGe*. The tool is able to read the report graph and, by using the Apache POI library (`https://poi.apache.org/`), the report template. For each observation in the report graph a SQL query is created and executed; the result fills the corresponding cell of the report template. A report generation requires less than 2 seconds.

8 Conclusions and Future Work

This work presents a "Multidimensional Model By Report" (MMBR) approach supporting the creation of multidimensional models able to produce a given (set of) report(s). The term "by report" refers to the ability to create a multidimensional (MD) model starting from a given report (typically expressed as Microsoft Excel file) that has to be filled with data extracted from a set of heterogeneous sources.

Important contributions refer to the automatic generation of the relational data structure correlated to the MD models generated by the approach, and to the ability to fill both fact and dimensional tables on the basis of domain ontologies enriched with mapping information related to the data sources.

There may be several future directions of research. The first one is related to the definition of an approach for the automatic computation of aggregates

of data according to the topological position of the cells that contain them, by taking into account row and column headers.

Another interesting research activity will study how to enrich the table identification algorithm. The aim is to allow the management of a larger (w.r.t., the actual algorithm) number of types of report, improving the efficiency of the presented approach.

References

1. Lymer, A., Debreceny, R., Gray, G.: Business reporting on the internet (1999)
2. Simkovic, M.: Competition and crisis in mortgage securitization. Indiana Law Journal **88** (2013) 213
3. Winter, R., Strauch, B.: A method for demand-driven information requirements analysis in data warehousing projects. In: 36th Hawaii International Conference on System Sciences (HICSS-36 2003), CD-ROM / Abstracts Proceedings, January 6-9, 2003, Big Island, HI, USA, IEEE Computer Society (2003) 231
4. Golfarelli, M., Maio, D., Rizzi, S.: The dimensional fact model: A conceptual model for data warehouses. Int. J. Cooperative Inf. Syst. **7**(2-3) (1998) 215–247
5. Golfarelli, M., Graziani, S., Rizzi, S.: Starry vault: Automating multidimensional modeling from data vaults. In Pokorný, J., Ivanovic, M., Thalheim, B., Saloun, P., eds.: Advances in Databases and Information Systems - 20th East European Conference, ADBIS 2016, Prague, Czech Republic, August 28-31, 2016, Proceedings. Volume 9809 of Lecture Notes in Computer Science., Springer (2016) 137–151
6. Blanco, C., de Guzmán, I.G.R., Fernández-Medina, E., Trujillo, J.: An architecture for automatically developing secure OLAP applications from models. Information & Software Technology **59** (2015) 1–16
7. Jovanovic, P., Romero, O., Simitsis, A., Abelló, A., Mayorova, D.: A requirement-driven approach to the design and evolution of data warehouses. Inf. Syst. **44** (2014) 94–119
8. Prat, N., Akoka, J., Comyn-Wattiau, I.: A uml-based data warehouse design method. Decision Support Systems **42**(3) (2006) 1449–1473
9. Nabli, A., Feki, J., Gargouri, F.: Automatic construction of multidimensional schema from OLAP requirements. In: 2005 ACS / IEEE International Conference on Computer Systems and Applications (AICCSA 2005), January 3-6, 2005, Cairo, Egypt, IEEE Computer Society (2005) 28
10. Giorgini, P., Rizzi, S., Garzetti, M.: Grand: A goal-oriented approach to requirement analysis in data warehouses. Decision Support Systems **45**(1) (2008) 4–21
11. Blanco, C., de Guzmán, I.G.R., Fernández-Medina, E., Trujillo, J.: An MDA approach for developing secure OLAP applications: Metamodels and transformations. Comput. Sci. Inf. Syst. **12**(2) (2015) 541–565
12. Bontcheva, K., Wilks, Y. In: Automatic Report Generation from Ontologies: The MIAKT Approach. Springer Berlin Heidelberg, Berlin, Heidelberg (2004) 324–335
13. Nebot, V., Berlanga, R., Pérez, J., Aramburu, M., Pedersen, T.: Multidimensional integrated ontologies: A framework for designing semantic data warehouses. Journal on Data Semantics XIII (2009) 1–36
14. Romero, O., Abelló, A.: A framework for multidimensional design of data warehouses from ontologies. Data & Knowledge Engineering **69**(11) (2010) 1138–1157
15. Thenmozhi, M., Vivekanandan, K.: An ontology based hybrid approach to derive multidimensional schema for data warehouse. International Journal of Computer Applications **54**(8) (2012)

16. Thenmozhi, M., Vivekanandan, K.: A framework to derive multidimensional schema for data warehouse using ontology. In: Proceedings of National Conference on Internet and WebSevice Computing, NCIWSC. (2012)
17. Benslimane, D., Arara, A., Falquet, G., Maamar, Z., Thiran, P., Gargouri, F. In: Contextual Ontologies. Springer Berlin Heidelberg, Berlin, Heidelberg (2006) 168–176
18. Barkat, O., Khouri, S., Bellatreche, L., Boustia, N.: Bridging context and data warehouses through ontologies. In: Proceedings of the Symposium on Applied Computing, ACM (2017) 336–341
19. W3C: W3c standard consortium. `http://www.w3.org`
20. Pardillo, J., Mazón, J.N.: Using ontologies for the design of data warehouses. International Journal of Database Management Systems, (IJDMS) **3**(2) (2011) 73–87
21. Protégé: Protégé ontology editor. `https://protege.stanford.edu/`
22. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1) (2007) 3–26
23. Zanibbi, R., Blostein, D., Cordy, J.R.: A survey of table recognition. Document Analysis and Recognition, Models, observations, transformations , and inferences **7**(1) (Mar 2004) 1–16
24. Golfarelli, M., Maio, D., Rizzi, S.: The dimensional fact model: A conceptual model for data warehouses. International Journal of Cooperative Information Systems **7**(02n03) (1998) 215–247
25. Sugumaran, V., Storey, V.C.: Ontologies for conceptual modeling: their creation, use, and management. Data & Knowledge Engineering **42**(3) (2002) 251–271