Open Archive TOULOUSE Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in: http://oatao.univ-toulouse.fr/
Eprints ID: 17174

The contribution was presented at ICDIM 2016:


Any correspondence concerning this service should be sent to the repository administrator: staff-oatao@listes-diff.inp-toulouse.fr
Enabling OLAP Analyses on the Web of Data

Franck Ravat
IRIT-Université Toulouse I Capitole
UMR5505, CNRS
Toulouse, France
ravat@irit.fr

Jiefu Song
IRIT-Université Toulouse I Capitole
UMR5505, CNRS
Toulouse, France
song@irit.fr

Abstract— This paper describes a business-oriented analysis environment facilitating analyses of coherent data from Data Warehouses (DWs) and Linked Open Data (LOD) datasets. Specifically, we present a multidimensional modeling solution, named Unified Cube, which provides a single, comprehensive representation of data from multiple sources. Unified Cubes include both concepts close to business terms and user-friendly graphical notations. An implementation framework is proposed to enable unified analyses of warehoused data and LOD. The feasibility of the proposed concepts is illustrated with examples based on real-world datasets.

Keywords— Data Warehouses; Linked Open Data; Unified Conceptual Model; On-line Analytical Processing;

I. INTRODUCTION

For over a decade, Data Warehouses (DWs) have been widely used to support decision-making processes. In a DW, operational data are periodically extracted, transformed and loaded in a multidimensional structure called data cube. A data cube represents warehoused data according to a fact (analysis subject composed of numeric indicators) and dimensions (analysis axes composed of attributes organized according to hierarchies) [1].

Coming from the inside of an organization, warehoused data do not provide all useful information for decision-making. In today's highly dynamic business context, decision-makers need to access several information sources, especially the Web, to obtain comprehensive analytical perspectives [2]. Among the data publically available on the Web, Linked Open Data (LOD)2, especially analysis-oriented LOD2, provide promising opportunities to enhance business analyses with semantically interconnected and machine-readable data [3], [4].

Motivating example. In an organization studying the social housing market, a DW contains quarterly and annual submitted and accepted applications for affordable housings by districts and counties (cf. Fig. 1(a)). The DW follows a Relational On-Line Analytical Processing (ROLAP) schema which organizes data according to a fact table (e.g., Affordable Housing) composed of numeric indicators (e.g., Applications and Acceptance) and several dimension tables (e.g., Area and Period) including an ordered set of analytical granularities (e.g., Quarter-Year and District-County). The warehoused data only provide a partial view of the demand for affordable housings. The decision-maker also wants to study the other factors (e.g., supply, cost etc.) which affect the affordable housing market to obtain complementary points of view during analyses. To do so, she/he searches on the Web and finds two LOD datasets3 published by the UK Department for Communities and Local Government. The first dataset (LOD1) reveals the yearly additional supply of affordable housings by districts and types, while the other dataset (LOD2) presents the evolution of the average rental charges of affordable housings according to districts and years. For the sake of readability, extracts of both LOD datasets are presented in tabular form in Fig. 1(b) and (c).

Problem Statement. The efficiency of analyzing data from multiple sources is low, since decision-makers have to manually relate information scattered in several analysis results represented in different modeling languages. Carrying out analyses of both warehoused data and LOD is also difficult. Not to mention most LOD are schema-less or schema-light [5], all DWs and LOD datasets involved in analyses do not include the same information at the same analytical granularities: (a) some analysis axes are only available in certain sources, e.g. the analysis axis of the affordable housings' type TypeAS only exists in LOD1; (b) the same analysis axes present in different sources may include data at different analytical granularities. e.g., for the temporal analysis axis, the source ROLAP contains two analytical granularities Year-Quarter, whereas the sources LOD1 and LOD2 only include one analytical granularity Year; (c) the same data may have different labels in different sources (e.g., 2013 is labeled as Year in the source ROLAP and PeriodRC in the source LOD2); (d) one analytical granularity may group several attributes from heterogeneous sources, e.g., the analytical granularity about district is described by a simple string in ROLAP (e.g., Adur), a classification code in LOD1(e.g., Adur E07000223) and an URI in LOD2 (e.g., http://statistics.data.gov.uk/id/statistical-geography/E07000223), (e) each source contains different indicators. Some indicators can be analyzed together starting from certain analytical granularities. For instance, Fig. 1 shows independent analysis results revealing the Applications and Acceptance of affordable housings with the supply of affordable housings (i.e. Dwellings) and average rental Charges in the same district named Adur for the same year of 2013.

http://linkeddata.org

2 Analysis-oriented LOD refer to a type of Linked Open Data describing numeric indicators according to a set of analysis axes. Without specification, all LOD discussed in this paper belong to analysis-oriented LOD.

3 http://opendatacommunities.org/data
Our aim is to provide decision-makers with a business-oriented environment which allows facilitating analyses of coherent data from multiple sources. To this end, we propose a unified view including, in a single schema, all the indicators along with all available analysis axes as well as all the attributes and aggregation paths (coming from the heterogeneous sources). In the previous example, a unified view would enable decision-makers to more easily build the following dashboard (cf. Fig. 1(d)). This dashboard allows better explaining the lower rental charges in the district Adur in 2014 results in more applications for affordable housings, while the greater number of accepted applications is related to the additional supply of affordable housings in 2014.

In this paper, we describe a novel multidimensional model, named Unified Cube, which unifies related warehoused data and LOD in a business-oriented way. First, we discuss related work on combining warehoused data with LOD (cf. section II). Second, we present conceptual definitions and graphical notations of Unified Cubes (cf. section III). Third, we describe an implementation framework for Unified Cubes (cf. section IV). At last, we detail how analyses of data from multiple sources are carried out through an implemented Unified Cube (cf. section V).

II. RELATED WORK

In the scientific literature, we can find many state-of-art papers [2], [6] that put forward the idea of including data from DWs and LOD datasets in one analysis. Among these papers, one of the consensuses is that a generic modeling solution should be proposed to unify warehoused data with LOD. The existing work with regards to a unified model can be categorized into three approaches.

Firstly, the DW community tries to build a classical OLAP schema based on warehoused data combined with LOD. More specifically, the authors of [7]–[9] treat LOD as other external data which should be stored in a DW through Extract, Transform and Load (ETL) processes. However, due to the resource-consuming ETL processing, it is difficult to guarantee the freshness of LOD materialized in a local, stationary repository. Moreover, since warehoused LOD are queried in an offline way, decision-maker can hardly obtain up-to-date information during analyses.

Secondly, the LOD community aims at transforming warehoused data into a RDF graph through a customizable mapping language such as R2RML. The resulted schema is expressed in an RDF-based modeling vocabulary which describes the multidimensional structure of a LOD dataset. Among the existing vocabularies, we can cite the RDF Data Cube Vocabulary (QB) as the current W3C standard to publish multidimensional statistical data. In [10], the authors propose the QB4OLAP vocabulary by adding more multidimensional characteristics to QB, like multiple analytical granularities and aggregation functions associated with a measure. Yet, datasets published in QB4OLAP and QB vocabularies still become standalone LOD sources on the Web. No solution has been proposed to enable interoperability between independent data sources.

Thirdly, a joint effort between the two communities consists of combining standalone sources together to enable simultaneous analyses of both warehoused data and LOD. The authors of [11] discuss research challenges and application prospects of merging data from multiple sources without any concrete proposal. The first solution was proposed by the LOD community. [12] proposes IGOLAP vocabulary to represent data from both DWs and LOD datasets according to a multidimensional structure. However, the work of [12] requires warehoused data to be transformed into a stationary RDF dataset before being analyzed with LOD, which reminds us of the drawbacks of the work based on ETL processes. Moreover, IGOLAP is an RDF-based vocabulary which represents data at the logical level. A business-oriented modeling solution is still missing to enable non-expert users to exploit coherent data at a higher level, i.e., conceptual level.

4 https://www.w3.org/TR/r2rml/
5 http://www.w3.org/TR/vocab-data-cube
To overcome these drawbacks, Unified Cubes should be generic enough to be independent of specific modeling solutions in the domains of DW and LOD. Moreover, a Unified Cube model should support on-the-fly analyses of both warehoused data and LOD. To facilitate the analysis tasks of non-expert users, only concepts close to business terms should be included in a Unified Cube.

III. CONCEPTUAL MODELING OF UNIFIED CUBES

The objective of Unified Cubes is to provide decision-makers with a single, comprehensive representation of all useful data. This representation should be oriented to business analyses, so that non-expert users can easily interact with a unified representation of all data. To facilitate the analysis of warehoused data and LOD. Moreover, a generic enough to be independent of specific modeling solutions in the domains of DW and LOD. To facilitate the analysis of a Unified Cube without the need for specific knowledge in DW and LOD domains. To this end, a Unified Cube should fix a generic modeling language which

- organizes warehoused data and LOD according to (a) one analysis subject containing numeric indicators and (b) some analysis axe including analytical granularities according to one or several aggregation paths;
- allows gathering several aggregation paths from different sources together in one analysis axis, even if these aggregation paths do not start from the same analytical granularity;
- describes an analytical granularity with several attributes from multiple sources while managing the relationships between heterogeneous attribute instances;
- supports on-the-fly and automatic extraction of detailed data from the sources;
- associates a numeric indicator with only a set of summarizable analytical granularities instead of all analytical granularities on all analysis axes;
- provides user-friendly graphical notations to facilitate the data exploitation task of decision-makers.

A. Analysis Axis: Dimension

A dimension corresponds to a unified vision of attributes which are related to one analysis axis. The organization of these attributes consists of the definition of one or several analytical granularities within a dimension.

The analytical granularities of a dimension may form one or several aggregation paths (i.e., hierarchies). Due to the unification of attributes belonging to different analytical granularities, and contrary to classical multidimensional model, a dimension of Unified Cube may contain two hierarchies sharing no common lowest analytical granularities. For instance, in a temporal dimension, the union of analytical granularities week-year and month-quarter-year results in two aggregation paths without a common starting point. The definition of dimension should be generic enough to include this specificity.

Notations. In the remainder of the paper, a superscript represents an element that the base belongs to, e.g., \( n^{\text{Adur}} \) means the name of Adur. A subscript indicates the index of an element in a set, e.g., \( \text{district}_k \) corresponds to the \( k \)-th one in a set of districts.

Definition 1. A dimension representing an analysis axis composed of a sequence analytical granularities is denoted as \( D_\alpha = \{ n^{D_\alpha}; L^{D_\alpha}; \ll^{D_\alpha} \} \), where:

- \( n^{D_\alpha} \) is the name dimension;
- \( L^{D_\alpha} = \{ l_1; \ldots; l_k \} \) is a set of levels, each level represents a distinct analytical granularity;
- \( \ll^{D_\alpha} \) is a reflexive binary relation which associates a child level \( l_i (l_i \in L^{D_\alpha}) \) with a parent level \( l_0 (l_0 \in L^{D_\alpha}) \), such as \( l_i \ll^{D_\alpha} l_0 \).

Example. We identify a geographical dimension denoted \( D_{\text{Geography}} \) from the three datasets in Fig. 2. The dimension \( D_{\text{Geography}} \) includes two levels, such as \( L_{\text{Geography}} = \{ l_{\text{District}}, l_{\text{County}} \} \). The binary relation \( \ll_{\text{Geography}} \) connects the level \( l_{\text{District}} \) to its parent level \( l_{\text{County}} \), such as \( l_{\text{District}} \ll_{\text{Geography}} l_{\text{County}} \).

Remark. By removing the constraint on the unique lowest level, a hierarchy including month, quarter and year from one source (i.e., \( L^{S1} = \{ l_{\text{month}}, l_{\text{quarter}}, l_{\text{year}} \} \)) and another one containing week and year from a different source (i.e., \( L^{S2} = \{ l_{\text{week}}, l_{\text{year}} \} \)) can be unified in a dimension \( D_{S1S2} \) which includes a set of levels \( L_{D_{S1S2}} = \{ l_{\text{week}}, l_{\text{month}}, l_{\text{quarter}}, l_{\text{year}} \} \). The same hierarchies can be found within the new dimension \( D_{S1S2} \), such as \( l_{\text{month}} \ll_{D_{S1S2}} l_{\text{quarter}} \ll_{D_{S1S2}} l_{\text{year}} \), which do not share a unique lowest level.

We define a sub-dimension as a part of dimension containing only a subset of levels.

Definition 2. A sub-dimension of \( D_\alpha \), denoted \( D_{\alpha \backslash S} = \{ n^{D_{\alpha \backslash S}}; L^{D_{\alpha \backslash S}}; \ll^{D_{\alpha \backslash S}} \} \), corresponds to the part of the dimension \( D_\alpha \) started with the level \( l_\alpha \), where

- \( n^{D_{\alpha \backslash S}} \) is the name of the sub-dimension;
- \( L^{D_{\alpha \backslash S}} \) is the subset of levels, \( L^{D_{\alpha \backslash S}} = \{ l_\alpha \in L^{D_{\alpha \backslash S}} \} \); \( l_i \ll^{D_{\alpha \backslash S}} l_\alpha \);
- \( \ll^{D_{\alpha \backslash S}} \) is the same binary relation of the one on the dimension \( D_\alpha \).

Example. A sub-dimension of the temporal dimension may be \( D_{\text{Time}\backslash \text{Year}} \) named Time-Year with \( L_{\text{Time}\backslash \text{Year}} = \{ l_{\text{Year}} \} \), which represent the subpart of the dimension \( D_{\text{Time}} \) that measures from the datasets LOD1 and LOD2 can be calculated along.

B. Analytical Granularity: Level

A level indicates a distinct analytical granularity by grouping together a set of attributes from different sources. For each attribute, the level keeps its name and an extraction formula allowing querying the attribute instances on-the-fly. Some mappings are also included within a level to manage the correlation and child-parent relations between heterogeneous attributes instances.

Definition 3. A level represents a distinct analytical granularity on a dimension. A level is denoted as \( l_\beta = \{ n^\beta; \alpha^\beta; c^\beta; \alpha^{\beta'} \} \), where:
• $n^d$ is the name of level;

• $\mathcal{A}^d = \{a_1, \ldots, a_n\}$ is a finite set of attributes. Each attribute $a_i (a_i \in \mathcal{A}^d)$ is a pair $(n^d, E^d)$, where $n^d$ is the name of the attribute and $E^d$ is an extraction formula through query algebra (i.e., relational algebra [13] and SPARQL algebra) indicating the instances of $a_i$. The domain of an attribute is denoted as $dom(a_i)$;

• $\mathcal{C}^d : dom(a_i) \rightarrow 2^{dom(a_i)} (a_i \in \mathcal{A}^d, a_j \in \mathcal{A}^d \setminus a_i)$ is a symmetric correlative mapping which associates an attribute $a_i$ with its related ones at the same level;

• $\mathcal{R}^d : 2^{dom(a_i)} \rightarrow dom(a_i)$ ($a_i \in \mathcal{A}^d, a_j \in \mathcal{A}^d$ and $\mathcal{R}^d$) is a rollup mapping implementing the binary relation between two levels. It connects the instances of child attributes with the instances of a parent attribute at an adjacent level.

Remark. TABLE I shows the algebraic form of commonly used SPARQL queries.

### TABLE I. SPARQL QUERY AND ALGEBRAIC REPRESENTATION

<table>
<thead>
<tr>
<th>SPARQL Query</th>
<th>Algebra</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT ?s1 ?s2 WHERE {?s1 ?p1 ?o1; ?s2 ?p2 ?o2}</td>
<td>(PROJECT (?s1 ?s2))</td>
</tr>
<tr>
<td>SELECT ?s1 ?s2 WHERE {?s1 ?p1 ?o1; ?s2 ?p2 ?o2}</td>
<td>(FILTER (?s1 ?s2))</td>
</tr>
<tr>
<td>FILTER (?s1 &lt;= ?o1 5)</td>
<td>(FILTER (?s1))</td>
</tr>
<tr>
<td>OPTIONAL {?s12 ?p22 ?o22.}</td>
<td>(FILTER (?s12 ?p22 ?o22))</td>
</tr>
<tr>
<td>FILTER (?s1 &gt; ?o1 10)</td>
<td>(FILTER (?s1))</td>
</tr>
</tbody>
</table>

Example. The level $D_{\text{District}}$ of the dimension $D_{\text{Geography}}$ contains a set of attributes (i.e., $\mathcal{A}^d_{\text{District}} = \{\text{a} \_\text{District}, \text{a} \_\text{Description}, \text{a} \_\text{AreaAS}, \text{a} \_\text{AreaRC}\}$) from different sources: the attributes $\text{a} \_\text{District}$ and $\text{a} \_\text{Description}$ come from the ROLAP DW, while the $\text{a} \_\text{AreaAS}$ and $\text{a} \_\text{AreaRC}$ are extracted from the LOD1 and LOD2 datasets respectively (cf. Fig. 2). To indicate how attribute instances can be retrieved from sources, an extraction formula is defined for each attribute within a level. For instance, the attribute $\text{a} \_\text{District}$ is associated with an extraction formula $E^d_{\text{District}} = \pi_{\text{District}}(\text{Area})$, while the attribute $\text{a} \_\text{AreaAS}$ is connected with an extraction formula as follows:

The correlative mapping $\mathcal{C}^d_{\text{District}}$ associates the instances of the attribute $\text{a} \_\text{District}$ with the ones of its descriptive attribute

---

- $LM: 2^{L_{\text{dim}} \times \cdots \times L_{\text{dim}}} \rightarrow m_k$ is a level-measure mapping which associates a set of summarizable levels with a measure $m_k (m_k \in M)$, such that $\forall i \in [1..n]$, $L_{\text{dim}} (k \leq L_{\text{dim}})$ is the set of levels on the sub-dimension $D_{\text{dim}}$ of $D_i (D_i \in D)$ starting from $l_k$.

We propose a set of user-friendly graphical notations for a Unified Cube based on the fact-dimension model [14] with minor modifications (cf. Fig. 2). The graphical notation aims at facilitating data exploitation at the schema level for non-expert users. Measures sharing the same related dimensions at the same levels are grouped together. For readability purposes, the graphical notations do not include concepts involving data such as hierarchies and rollup mapping.

**Example.** Fig. 2 shows a Unified Cube which is built upon the warehoused data and the two LOD datasets in the motivating example. It contains three dimensions $D=\{D\text{Geography}; D\text{Time}; D\text{Type}\}$. Each measure is associated with its related levels. For the sake of readability, in the graphical notation the level-measure mappings are represented only between the lowest levels of sub-dimension and measures:

$LM: \{ \{L_{\text{dim}} \text{Time}, L_{\text{dim}} \text{Geography}\} \rightarrow \{m\text{Charges}\}; \}
\{ L_{\text{dim}} \text{Time}, L_{\text{dim}} \text{Geography} \rightarrow \{m\text{Applications}, m\text{Acceptance}\}; \}
\{ L_{\text{dim}} \text{Time}, L_{\text{dim}} \text{Geography}, L_{\text{dim}} \text{Type} \rightarrow \{m\text{Dwellings}\}\}

IV. IMPLEMENTATION OF A UNIFIED CUBE

In this section, we describe a generic, reusable implementation framework for Unified Cubes (cf. Fig. 3). The objective of the framework is to enable analyses of both warehoused data and LOD by representing different components of a unified schema and relationships among data from multiple sources. To do so, the implementation framework should include:

- a metamodel for Unified Cubes with a compatible instantiation algorithm. (cf. section IV.A Schema module);
- tables of correspondences allowing keeping related data in a coherent environment (cf. section IV.B Relationships module).

### A. Schema Module

The Schema module aims at representing the overall structure of data from multiple sources. A metamodel representing the concepts related to the multidimensional schema of a Unified Cube is included within this module. Fig. 4 shows the class diagram representation of the metamodel.

!!![](image)

**Fig. 4. Metamodel of Unified Cubes' schema.**

In this metamodel, each class implements one concept of Unified Cubes. Note that binary relations between level (i.e., $\equiv$) and level-measure mappings (i.e., $LM$) are implemented through associations. Moreover, each attribute and each measure are associated with an executable query (i.e., $queryM$ or $queryA$) translated from the extraction formulae. By executing a query in a corresponding source whose query endpoint (i.e., $URLM$ and $URLA$) is recorded, data can be extracted in real-time, so that decision-makers can always obtain up-to-date information during analyses.

In our previous work [15], we describe the designing of a conceptual Unified Cube through a schema definition process. In the remainder of this section, we complete our previous work by detailing how a Unified Cube is implemented. To automate implementations of Unified Cubes' schemas, we propose an algorithm which instantiates the metamodel with a conceptual Unified Cube (cf. Algorithm 1).

We apply the algorithm to the conceptual Unified Cube in Fig. 2. The obtained instantiated metamodel is composed of (a) 1 instance of the Fact class, (b) 4 instances of the Measure class, (c) 3 instances of the Dimension class, (d) 5 instances of the Level class with 12 associations implementing the level-measure mapping and 4 associations representing the binary relation between levels, and (e) 10 instances of the Attribute class. Fig. 3 shows an extract of the instantiated metamodel related to the measure $m\text{Charges}$.

### Algorithm 1. Unified Cube Schema Implementation

**Input:** A Unified Cube $= \{F; D; LM\}.$

**Output:** An instantiated metamodel.
Begin
1. For each $D_i \in \mathcal{D}$
2. For each $l_i \in \mathcal{L}^d$
3. Instantiate the Level class: $m_i^\text{meta} = \text{new Level}(n^d)$
4. For each $a_j \in \mathcal{A}^d$
5. Translate the query algebra of the extraction formula $E^{n^d}$ into a query $Q_{a_j}$
6. Get the attribute's query endpoint URL$_n$
7. Instantiate the Attribute class: $a_j^\text{meta} = \text{new Attribute}(n^x, Q_{a_j}, \text{URL}_n)$
8. Instantiate the association between $a_j^\text{meta}$ and $m_i^\text{meta}$
9. End for
10. End for
11. Instantiate the Fact class $F^\text{meta} = \text{new Fact}(n^f)$
12. End for
13. Instantiate the Fact class $F^\text{meta} = \text{new Fact}(n^f)$
14. For each $m_j \in \mathcal{M}^f$
15. Translate the query algebra of the extraction formula $E^{n^f}$ into a query $Q_{m_j}$
16. Get the measure's query endpoint URL$_m$
17. Instantiate the Measure class: $m_j^\text{meta} = \text{new Measure}(n^m, Q_{m_j}, \text{URL}_m)$
18. Instantiate the association between $m_j^\text{meta}$ and $F^\text{meta}$
19. For each level $l_h \in \mathcal{L}^f$ associated with $m_j$
20. Instantiate the Level class
21. End for
22. End for
23. End for

End

B. Relationships Module

The Relationships module handles related data from different sources. It includes (a) a toolkit identifying related attribute instances and (b) tables of correspondences keeping the obtained relationships between attribute instances.

a) Step 1: Identification of Related Data

In the context of Unified Cubes, identifying the relationships between data consists of assessing the correlation and the child-parent relations between attribute instances from different sources. Numerous techniques have been proposed in the scientific literature to identify the relationships between warehoused data and LOD. Two comprehensive surveys of automatic matching between related schemas and data instances can be found in [16], [17]. The authors of [18] focus on some semi-automatic techniques which take users’ needs into account.

Among all above-mentioned work, one consensus is that there is no "one-size-fits-all" method for aligning data of all types [19], and such a universal method does not fall within the scope of our work. One objective of the Relationships module is to provide schema designers with a toolkit facilitating the task of discovering related data. A designer can simply specify an appropriate method implemented within the toolkit. Then the implementation framework automatically searches for related data in corresponding sources according to the chosen method.

In the context of the Unified Cube of our running example, we implement some of the most effective methods allowing identifying the correlation and child-parent relationships between data. These methods can be categorized into three groups.

- The first group is based on an intermediate ontology with a comprehensive coverage of the related concepts in several datasets (i.e., containing enough matches between related data).
- The second group is applicable to data instances sharing related labels by calculating the string-based similarity. Normalization techniques and external thesaurus are often used to improve the obtained result.
- The third group includes methods use reasoning techniques to deduce the relations between data. This group of methods is especially useful to obtain rollup mappings between attributes at two adjacent levels. By referring to existing correlative and/or rollup mappings within a Unified Cube, the reasoning process is as follows:

$$\text{let } a_1, a_2 \in \mathcal{A}^d, \ a_1, a_2 \in \mathcal{A}^d \ (l_1 \leq l_2):$$

$$\mathcal{C}^d: \text{dom}(a_1) \rightarrow \text{dom}(a_2) \land \mathcal{R}^d: \text{dom}(a_2) \rightarrow \text{dom}(a_3)$$

$$\Rightarrow \exists \mathcal{C}^d: \text{dom}(a_1) \rightarrow \text{dom}(a_2) \land \mathcal{R}^d: \text{dom}(a_2) \rightarrow \text{dom}(a_3)$$

TABLE II. shows five correlative and six rollup mappings identified after applying suitable methods implemented in the Relationships module. Note that no method is needed to identify mappings already existing in the sources. For instance, mappings 1, 6 and 9 are obtained by directly referring to the ROLAP DW, while mapping 4 is directly identified because the instances of the attributes of PeriodAS and aPeriodRC are denoted by using the same identifier (e.g., the year of 2013 is denoted as <http://reference.data.gov.uk/id/government-year/2013-2014> for both aPeriodAS and aPeriodRC).

The relations between attribute instances involved in the above-mentioned mappings are called direct, since it can be identified by directly querying the corresponding sources. Relations embedded in other mappings are called correlative, because it is identified by using additional processing methods. For instance, the correlative mapping between the attributes aDistrict and aAreaAS (cf. mapping 2 in TABLE II.) is obtained by calculating the substring test similarity $\bar{\sigma}(s_1, s_2)$, such as $\bar{\sigma}(s_1, s_2) = \frac{2|s|}{|s_1| + |s_2|}$, where $s_1$ and $s_2$ are two attribute instances, $s$ is the longest common substring between $s_1$ and $s_2$. 
which can be possibly materialized, so that we shall know to calculate the maximum number of attribute instances that should be materialized in the database during analyses, mappings involving correlative attribute instances. Therefore, for a n-dimensional Unified Cube, the Relationships module materializes at most the following number of attributes instances involved in correlative mappings.

$$\sum_{i=1}^{n} \sum_{d=1}^{\left\lceil \frac{k}{d} \right\rceil -1} \min(\{\text{dom}(a_i), \text{dom}(a_k)\})$$

A rollup mapping associates one or several instances of a child attribute with at most one instance of a parent attribute. The maximum number of attributes instances possibly being materialized for a rollup mapping between a pair of attributes $a_i$ and $a_j (a_i \in \mathbb{A}^k, a_j \in \mathbb{A}^l, l \leq k)$ is $\text{dom}(a_i)$. Therefore, for a n-dimensional Unified Cube, the Relationships module materializes no more than the following number of attributes instances for the implementation of rollup mappings.

$$\sum_{i=1}^{n} \sum_{d=1}^{\left\lceil \frac{k}{d} \right\rceil -1} \sum_{k=1}^{\left\lceil \frac{l}{d} \right\rceil} \text{dom}(a_i)$$

In both formulae, only dimensions are involved. For a multidimensional dataset, the size of dimension is much smaller than the size of fact. For instance, in a test set of SSB benchmark [20], dimensions represent basically 1% to 6% of the total data volume, and the larger the data scale becomes, the lower the proportion of dimensions accounts for [21].

From the above discussions, it is apparent that the materialization of correlative mappings allows managing related data from different sources with minimum cost. Within the implementation framework, two types of table of correspondences are used to materialize correlative mappings and rollup mappings. As shown in Fig. 6, the one implementing correlative mappings associates the instances of an attribute (i.e., INSTANCE) of an attribute (i.e., ATTRIBUTE) with the related ones (i.e., COR_INS) of a correlative attribute (i.e., COR_ATT) within the same level, while the other one managing rollup mappings connects a set of instances (i.e., CHILD_INS) of a child attribute (i.e., CHILD_ATT) with an instance (i.e., PARENT_INS) of a parent attribute (i.e., PARENT_ATT). The content of a table of correspondence is preprocessed and updated at the beginning of an analysis process to keep the materialized data up-to-date.

---

**Table II. List of Mappings and Identification Methods**

<table>
<thead>
<tr>
<th>Mapping</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $\ell^{\text{access}}: \text{dom}(a_{\text{access}}) \rightarrow \text{dom}(a_{\text{access}})$</td>
<td>n/a substring test similarity</td>
</tr>
<tr>
<td>2. $\ell^{\text{access}}: \text{dom}(a_{\text{access}}) \rightarrow \text{dom}(a_{\text{access}})$</td>
<td>intermediate ontology$^7$</td>
</tr>
<tr>
<td>3. $\ell^{\text{access}}: \text{dom}(a_{\text{access}}) \rightarrow \text{dom}(a_{\text{access}})$</td>
<td>n/a substring test similarity</td>
</tr>
<tr>
<td>4. $\ell^{\text{access}}: \text{dom}(a_{\text{access}}) \rightarrow \text{dom}(a_{\text{access}})$</td>
<td>n/a reasoning (cf. mappings 2 and 6)</td>
</tr>
<tr>
<td>5. $\ell^{\text{access}}: \text{dom}(a_{\text{access}}) \rightarrow \text{dom}(a_{\text{access}})$</td>
<td>n/a reasoning (cf. mappings 3 and 7)</td>
</tr>
<tr>
<td>6. $\ell^{\text{access}}: \text{dom}(a_{\text{access}}) \rightarrow \text{dom}(a_{\text{access}})$</td>
<td>n/a reasoning (cf. mappings 5 and 9)</td>
</tr>
<tr>
<td>7. $\ell^{\text{access}}: \text{dom}(a_{\text{access}}) \rightarrow \text{dom}(a_{\text{access}})$</td>
<td>n/a reasoning (cf. mappings 4 and 10)</td>
</tr>
</tbody>
</table>

---

$^7$ http://opendatacommunities.org/data/ons-geography-administration

---

**Fig. 5. Extract of metamodel instantiated with the ROLAP DW and the two LOD datasets**
After implementing the **Unified Cube** of our running example, 1474 attribute instances are materialized in the **Relationships** module, which corresponds to about 5.1% of data in the sources. Selective materialization allows reducing significantly the amount of data stored in the implementation framework, which minimizes the cost of updating and storing data from the sources.

**V. ANALYSIS PROCESSING**

Besides the two modules presented in section IV, the implementation framework also includes a third module, named **Mediator**, whose aim is to enable decision-makers to analyze both warehoused data and LOD. It (a) deals with an analytical need expressed through the graphical notations of a **Unified Cube** (cf. arrow 1 in Fig. 7) and (b) yields an analysis result based on data extracted from multiple sources (cf. arrow 5 in Fig. 7).

![Fig. 7. Analysis processing within the implementation framework](image)

Since most decision-makers are not expert in neither DW nor LOD domain, the **Mediator** module should facilitate analysis tasks by including functionality to:

- automatically extract data related to the analysis from the sources. To do so, we propose an algorithm which generates a set of executable queries based on the measures and attributes chosen by a decision-maker (cf. arrows 2 and 3);
- include all data in one unique analysis result to offer comprehensive analytical perspectives. To this end, we present an algorithm which combines all partial results together by referring to the related attribute instances materialized in the implementation framework (cf. arrows 4.1, 4.2 and 5).

### A. Partial Query Generation

By referring to the extraction formulae implemented in the metamodel, the **Mediator** module can directly extract the instances of one measure or one attribute from one source. In the case of a unified analysis including several measures and attributes, a simple concatenation of the extraction formulae from the metamodel is not enough. The **Mediator** module must enrich and/or rewrite the queries hosted in the metamodel to adapt to the specificities of data sources.

- When an analysis involves attributes at different levels, the **Mediator** module complete queries by adding joins and grouping predicates.
- When an analysis includes measures and attributes from multiple sources, the **Mediator** module generates queries containing the corresponding attributes and measures from the same source.

To automate the generation of queries within the **Mediator** module, we propose the following algorithm.

**Algorithm 2. Partial Query Generation**

**Input:** A set of chosen attributes $A_{sub}$ ($A_{sub} \subseteq \bigcup_{D \in D} \bigcup_{k \in E_k} \mathcal{A}^k$), a set of chosen measures $M_{sub}$ ($M_{sub} \subseteq M'$)

**Output:** A set of partial queries $Q$

**Begin**

1. For each $m_e \in M_{sub}$
   - Get associated levels of $m_e$: $L_{m_e} = m_e.getLvl()$;
   - Get the query of the measure $q_{m_e} = m_e.getQueryM()$;
   - Remove irrelevant attributes from $q_{m_e}$
2. For each $a_e \in A_{sub}$
   - Get the level of $a_e$: $P_a = a_e.getLevel()$;
   - If $m_e.getURLA() \neq a_e.getURLA()$
     - Find an attribute $a_e'$ such as $a_e'.getLvl() = P_a$ and $m_e.getURLM() = a_e'.getURLM()$;
     - Let $a_e = a_e'$;
   - End if
3. Get the query $q_{p_a}$ of the attribute $a_e$;
4. Add $q_{p_a}$ in $q_{m_e}$;
5. Find a set of level $L_{sub} (L_{sub} \subseteq L_m)$ such as $\forall l \in L_{sub}$ $l_i.getDim() = P_a \land l_i.getLvl() \land l_i \neq P_a$;
6. If $|L_{sub}| > 1$
   - Add join predicates between $m_e$ and $a_e$ in $q_{m_e}$;
   - Add a grouping predicate at the end of $q_{m_e}$;
   - End if
7. End for
8. $Q = Q \cup q_{m_e}$;
9. End for
10. End for
11. End.

**End**

To better explain how the algorithm works, we illustrate its execution process with an example. An analysis need about the number of applications (i.e., $m_{app}$) for social housings with the average rental charges (i.e., $m_{char}$) by year

$$\forall D_i \in D, U_{k \in E_k} \mathcal{A}^k = \mathcal{A}^1 \cup \ldots \cup \mathcal{A}^{|E|}$$ is the set of attributes within the dimension $D$. Thus, $U_{D \in D} U_{k \in E_k} \mathcal{A}^k$ represents all attributes of a **Unified Cube**.
B. Generation of Analysis Result

At the end of an analysis, the Mediator module returns a global result unifying all related information to decision-makers. Since n-dimensional cubes are difficult to be exploited by decision-makers [22], [23], we model the analysis result in tabular form, called a Multidimensional Table.

Definition 6. A Multidimensional Table includes one analysis subject containing measures and up to three analysis axes. A Multidimensional Table is denoted as MT=\{MT_S, MT_A\}, where:

- \(S^{MT}\) is the analysis subject containing a set of measures \(M^{MT}\) (\(M^{MT} \subseteq M\));
- \(A^{MT}\) is the analysis subject containing a set of attributes.

\(Ax^{MT}_{1} = \{a^{MT}_{1}, \ldots, a^{MT}_{m}\} \ (1 \leq m \leq 3)\) is the set of analysis axes. Each analysis axis \(Ax^{MT}_{i}\) (\(Ax^{MT}_{i} \in A^{MT}\)) includes a set of attributes denoted \(A^{Ax^{MT}_{i}}\).

An example of Multidimensional Table can be found in Fig. 1(d). From this example, we can see measures and attributes from different sources are displayed together in one Multidimensional Table, so that decision-makers can analyze all related data in a unified way. We propose the following algorithm to produce a Multidimensional Table based on the partial query results received from the Mediator module and the mappings of related attributes instances obtained from the Relationships module.

**Algorithm 3. Generation of Analysis Result**

**Input:** A set of partial results \(Res\), each result \(r_i (r_i \in Res)\) contains a set of measures \(M^f\) and a set of attributes \(A^f\) from one data source; A set of mappings \(Map\), each mapping, \(f_{pq}: dom(a_p) \rightarrow dom(a_q)\) represents either the correlative mapping or the rollup mapping between the attributes \(a_p\) and \(a_q\).

**Output:** A Multidimensional Table \(MT=\{S^{MT}, A^{MT}\}\)

**Begin**

1. For each \(r_i \in Res\)
2. For each \(a_p \in A^f\)
3. Get the dimension of \(a_p\): \(D^{pe}=a_p.getLvl().getDim()\);
4. Add \(a_p\) in the set on the corresponding analysis axis of \(MT\), such as \(3Ax^{MT}\) is \(Ax^{MT}=D^{pe}_{Ax^{MT}} = A^{Ax^{MT} \cup \{a_p\}}\);
5. For each attribute instance \(i^pe_{pq} \in dom(a_p)\)
6. If \(\exists f_{pq}: dom(a_p) \rightarrow dom(a_q) (f_{pq} \in Map)\) and \(a_q \in A^{Ax^{MT}}\)
7. Associate \(i^pe_{pq}\) with the related instances of the attribute \(a_q\);
8. Else
9. Create a new header \(i^pe_{pq}\) in the MT
10. End if
11. End For
12. End For
13. For each \(m_i \in M^f\)
14. Add \(m_i\) in \(M^{MT}\);
15. For each measure value \(v^{m_i}\) in \(val(m_i)\)
16. If there exists a measure value \(v^{m_k}\) (\(m_k \in M^{MT}\)) described by the same related attributes instances as \(v^{m_i}\) in the MT
17. Display \(v^{m_i}\) in the same cell as \(v^{m_k}\);
18. Else
19. Display \(v^{m_i}\) in a new cell
20. End if
21. End For
22. End For
23. End For

**End.**

During the execution of the algorithm, each analysis axis in a Multidimensional Table is built by (a) combining different attributes referring to the same dimension (cf. lines 3 and 4) and (b) organizing related attribute instances into the same headers (cf. lines 5 to 11). The analysis subject in a
Multidimensional Table is obtained by (a) unifying the measures from all partial results of query (cf. line 14) and (b) displaying multiple measure values together if they are described by related attribute instances involved in correlative mappings or rollup mappings (cf. lines 15–21).

VI. CONCLUSION AND DISCUSSION

Under today’s highly competitive business environment, warehoused data alone do not provide enough information during analyses. External data, especially analysis-oriented LOD, should also be included in a decision-making context to offer multiple perspectives to decision-makers. In this paper, we describe a novel modeling solution, named Unified Cube, which represents data from DWs and LOD datasets in a generic and business-oriented way.

As an extension of classical multidimensional models, a Unified Cube organizes warehoused data and LOD according to one analysis subject (i.e., fact) and a set of analysis axes (i.e., dimensions). Due to the unification of analytical granularities (i.e., levels) from different sources, we generalize the definition of dimension to allow including several hierarchies sharing no common lowest level. The concept of level is also extended to regroup multiple attributes if they refer to the same analytical granularity. Heterogeneity among attribute instances from multiple sources is managed by correlative mappings, while the child-parent relations between attribute instances at adjacent levels are implemented through rollup mappings. The concept of level-measure mapping is also included in Unified Cubes to associate a numeric indicator (i.e., measure) with a set of summarizable levels.

To enable non-expert users to analyze warehoused data with LOD, we propose an implementation framework compatible with Unified Cubes. Three modules are included within the framework. The Schema module hosts a metamodel implementing the multidimensional structure of a Unified Cube. The instantiation of the metamodel yields a non-materialized Unified Cube supporting on-the-fly analyses among multiple data sources. The Relationships module provides a toolkit allowing identifying related attributes instances from heterogeneous sources. A set of tables of correspondences materializes the mappings between related data to avoid repetitive relationship processing during analyses. The third module named Mediator automatically translates a unified analysis need into a set of executable queries. At the end of an analysis, the Mediator module provides one unique result including all useful data for an analysis.

In the future, we intend to include a graphical querying language within our implementation framework to further facilitate the analysis tasks of non-expert users. We also intend to extend correlative and rollup mappings to more generic ones, e.g., n-ary mappings between three or more attributes. A more long-term objective consists of maximizing the analysis efficiency by finding an optimized selective materialization strategy according to different data types.

References


