Enabling OLAP Analyses on the Web of Data

Franck Ravat, Jiefu Song

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 Cube*s 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 Warehouse*s (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)[1], especially analysis-oriented LOD[2], 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 datasets[3] 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://statisticss.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*.

[1] 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

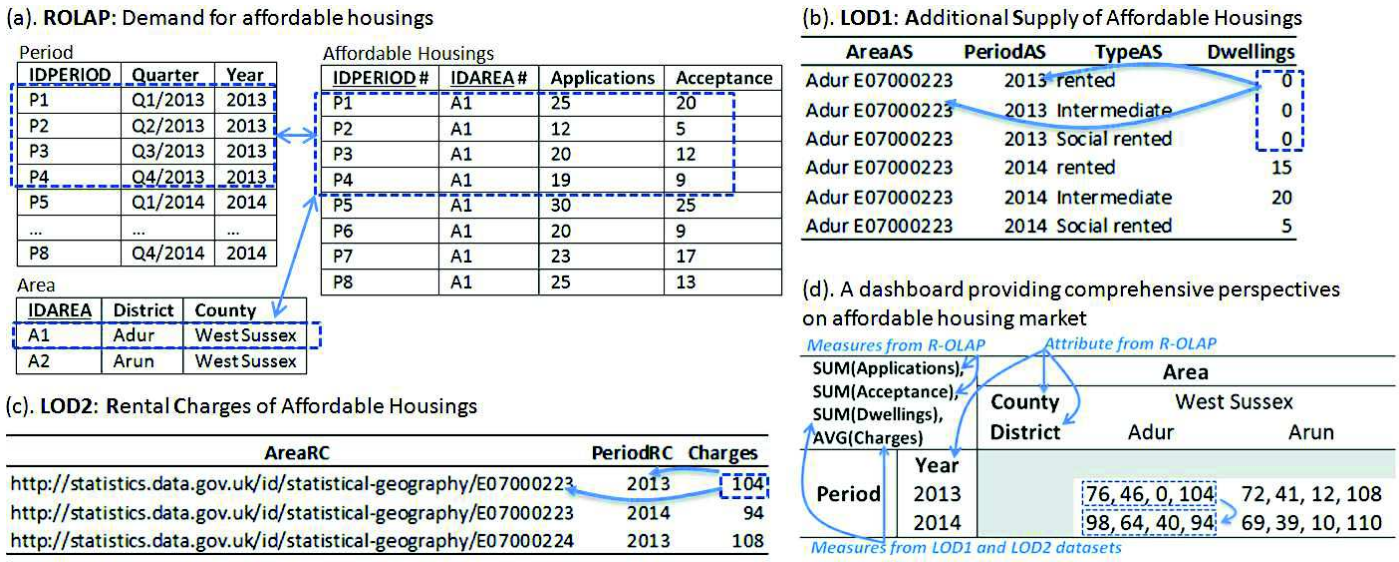

Fig. 1. A ROLAP DW and two LOD datasets about the social housing market.

Contribution. 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 Cube*s (cf. section III). Third, we describe an implementation framework for *Unified Cube*s (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 an RDF graph through a customizable mapping language such as R2RML[4]. 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)[5] 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 Cube*s 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 Cube*s 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 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 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., $name^{\text{Adur}}$ means the *name* of *Adur*. A subscript indicates the index of an element in a set, e.g., $district_{\text{k}}$ corresponds to the *k-th* one in a set of *district*s.

Definition 1. A *dimension* representing an analysis axis composed of a sequence analytical granularities is denoted as $\text{D}_\text{i}=\{\text{n}^{\text{D}_\text{i}}; \mathcal{L}^{\text{D}_\text{i}}; \preccurlyeq^{\text{D}_\text{i}}\}$, where:

- $\text{n}^{\text{D}_\text{i}}$ is the dimension name;
- $\mathcal{L}^{\text{D}_\text{i}}=\{l_1;\ldots; l_\text{k}\}$ is a set of levels, each level represents a distinct analytical granularity;
- $\preccurlyeq^{\text{D}_\text{i}}$ is a reflexive *binary* relation which associates a *child* level l_a ($l_\text{a}\in\mathcal{L}^{\text{D}_\text{i}}$) with a *parent* level l_b ($l_\text{b}\in\mathcal{L}^{\text{D}_\text{i}}$), such as $l_\text{a}\preccurlyeq^{\text{D}_\text{i}} l_\text{b}$.

Example. We identify a geographical dimension denoted $\text{D}_{\text{Geography}}$ from the three datasets in Fig. 2. The dimension $\text{D}_{\text{Geography}}$ named *Geography* includes two levels, such as $\mathcal{L}^{\text{Geography}}=\{l_{\text{District}}; l_{\text{County}}\}$. The *binary* relation $\preccurlyeq^{\text{Geography}}$ connects the level l_{District} to its parent level l_{County}, such as $l_{\text{District}}\preccurlyeq^{\text{Geography}} l_{\text{County}}$.

Remark. By removing the constraint on the unique lowest level, a hierarchy including *month*, *quarter* and *year* from one source (i.e., $\mathcal{L}^{S1}=\{l_{\text{month}}; l_{\text{quarter}}; l_{\text{year}}\}$) and another one containing *week* and *year* from a different source (i.e., $\mathcal{L}^{S2}=\{l_{\text{week}}; l_{\text{year}}\}$) can be unified in a dimension D_{S1S2} which includes a set of levels $\mathcal{L}^{\text{D}_{\text{S1S2}}}=\{l_{\text{week}}; l_{\text{month}}; l_{\text{quarter}}; l_{\text{year}}\}$. The same hierarchies can be found within the new dimension D_{S1S2}, such as $l_{\text{month}}\preccurlyeq^{\text{D}_{\text{S1S2}}} l_{\text{quarter}}\preccurlyeq^{\text{D}_{\text{S1S2}}} l_{\text{year}}$ and $l_{\text{week}}\preccurlyeq^{\text{D}_{\text{S1S2}}} l_{\text{year}}$ 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_i, denoted $\text{D}_{\text{i}\backslash l_\text{s}}=\{\text{n}^{\text{D}_{\text{i}\backslash l\text{s}}}; \mathcal{L}^{\text{D}_{\text{i}\backslash l\text{s}}}; \preccurlyeq^{\text{D}_\text{i}}\}$, corresponds to the part of the dimension D_i started with the level l_s, where

- $\text{n}^{\text{D}_{\text{i}\backslash l\text{s}}}$ is the name of the sub-dimension;
- $\mathcal{L}^{\text{D}_{\text{i}\backslash l_\text{s}}}$ is the subset of levels, $\mathcal{L}^{\text{D}_{\text{i}\backslash l_\text{s}}}\subseteq\mathcal{L}^{\text{D}_\text{i}}$, $\forall l_\text{i}\in\mathcal{L}^{\text{D}_{\text{i}\backslash l_\text{s}}}$, $l_\text{s}\preccurlyeq^{\text{D}_\text{i}} l_\text{i}$;
- $\preccurlyeq^{\text{D}_\text{i}}$ is the same binary relation of the one on the dimension D_i.

Example. A sub-dimension of the temporal dimension may be $\text{D}_{\text{Time}\backslash l_{\text{Year}}}$ named *Time-Year* with $\mathcal{L}_{\text{Time}\backslash l_{\text{Year}}}=\{l_{\text{Year}}\}$, which represent the subpart of the dimension D_{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 $l_\text{d}=\{\text{n}^{l_\text{d}}; \mathcal{A}^{l_\text{d}}; C^{l_\text{d}}; \mathcal{R}^{l_\text{d}}\}$, where:

- n^{l_d} is the name of level;
- $\mathcal{A}^{l_d} = \{a_1; \ldots; a_e\}$ is a finite set of *attributes*. Each attribute a_x ($a_x \in \mathcal{A}^{l_d}$) is a pair $\langle n^{a_x}, E^{a_x} \rangle$, where n^{a_x} is the name of the attribute and E^{a_x} is an *extraction formula* through query algebra (i.e., *relational* algebra[13] and *SPARQL* algebra[6]) indicating the instances of a_x. The domain of an attribute is denoted as $dom(a_x)$;
- $\mathcal{C}^{l_d} : dom(a_x) \rightarrow 2^{dom(a_y)}$ ($a_x \in \mathcal{A}^{l_d}$, $a_y \in \mathcal{A}^{l_d} \setminus a_x$) is a symmetric *correlative* mapping which associates an attribute a_x with its related ones at the same level;
- $\mathcal{R}^{l_d}: 2^{dom(a_x)} \rightarrow dom(a_z)$ ($a_x \in \mathcal{A}^{l_d}$, $a_z \in \mathcal{A}^{l_e}$ and $l_d \preccurlyeq l_e$.) 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

SPARQL query	SPARQL Algebra
SELECT * WHERE { ?s ?p ?o}	(BGP (TRIPLE ?s ?p ?o))
SELECT ?s ?p WHERE {?s ?p ?o}	(PROJECT(?s ?p) (BGP (TRIPLE ?s ?p ?o)))
SELECT ?o1 ?o2 WHERE{?s ?p ?o1. FILTER (?o1 < 5) OPTIONAL {?s ?p2 ?o2 . FILTER (?o2 > 10)}}	(PROJECT(?o1 ?o2) (FILTER (< ?o1 5) (LEFTJOIN (BGP (TRIPLE ?s ?p ?o1)) (BGP (TRIPLE ?s ?p2 ?o2)) (> ?o2 10))))
SELECT ?s (COUNT(?o) as ?nb) WHERE {?s ?p ?o} GROUP BY ?s HAVING (COUNT(?o)=2)	(PROJECT(?s ?nb) (FILTER (> ?.0 10) (EXTEND((?nb ?.0)) (GROUP (?s) ((?.0 (COUNT ?o))) (BGP (TRIPLE ?s ?p ?o))))))

Example. The level $l_{District}$ of the dimension $D_{Geography}$ contains a set of attributes (i.e., $\mathcal{A}^{l_{District}} = \{a_{District}; a_{Decription}; a_{AreaRC}; a_{AreaAS}\}$) from different sources: the attributes $a_{District}$ and $a_{Decription}$ come from the ROLAP DW, while the a_{AreaRC} and a_{AreaAS} 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 $a_{District}$ is associated with an *extraction formula* $E^{a_{District}} = \pi_{District}$ (Area), while the attribute a_{AreaAS} is connected with an *extraction formula* as follows:

```
prefix ex: <http://opendatacommunities.org/data/housing-market/supply/additions/>
prefix geo: <http://opendatacommunities.org/def/ontology/geography/>
E^aAreaAS=(PROJECT(?AreaAS)
        (BGP (TRIPLE ?ob qb:dataSet ex:affordablehousingtype)
            (TRIPLE ?ob geo:refArea ?AreaAS))).
```

The *correlative* mapping $\mathcal{C}^{l_{District}}$ associates the instances of the attribute $a_{District}$ with the ones of its descriptive attribute $a_{Decription}$ coming from the same ROLAP DW and the related attributes a_{AreaRC} and a_{AreaAS} in LOD datasets, for instance,

```
C^lDistrict:{Adur}→
        {{A local government district of West Sussex in UK}; {Adur E07000223};
        {http://statistics.data.gov.uk/id/statistical-geography/E07000223}}
```

The *rollup* mapping $\mathcal{R}^{l_{District}}$ aggregates the instances of $a_{District}$, a_{AreaAS} and a_{AreaRC} at the level $l_{District}$ to the ones of a_{County} at the level l_{County}, for instance,

```
R^lDistrict: {{Adur; Arun}; {Adur E07000223; Arun E07000224};
        {http://statistics.data.gov.uk/id/sta-tistical-geography/ E07000223;
        http://stati-stics.data.gov.uk/id/statistical-geography/E07000224}}
        →{West Soussex}.
```

C. Analysis Subject: Fact

To always provide up-to-date information to decision-makers, the definition of analysis subject should enable on-the-fly querying of numeric indicators from the sources. To this end, each numeric indicator is associated with an *extraction formula*.

Definition 4. A *Fact* corresponds to an analysis subject composed of a set of *measures*. A fact is denoted as $F = \{n^F; \mathcal{M}^F\}$ where:

- n^F is the name of the fact;
- $\mathcal{M}^F = \{m_1; \ldots; m_p\}$ is a finite set of numeric indicators called *measures*. Each measure m_e ($m_e \in \mathcal{M}^F$) is a pair $\langle n^{m_e}, E^{m_e} \rangle$, where n^{m_e} is the name of the measure, E^{m_e} is an *extraction formula* pointing to the images of the measure in the source. The set of values of a measure m_e is denoted as $val(m_e)$.

Example. The fact named *Affordable Housing* contains four measures, namely $m_{Applications}$, $m_{Acceptance}$, $m_{Dwelling}$ and $m_{Charges}$. The measure $m_{Applications}$ has an *extraction formula*: $E^{m_{Applications}} = \mathcal{F}_{SUM(Applications)}$ (AffordableHousing). The measure $m_{Dwellings}$ has an *extraction formula* as follows:

```
prefix m: <http://opendatacommunities.org/def/ontology/housing-market/>
E^mDwellings=(PROJECT (?Dwellings)
        (EXTEND ((?Dwellings ?.0)) (GROUP() ((?.0 (sum ?nb)))
        (BGP (TRIPLE qb:dataSet ?ob ex:affordablehousingtype)
            (TRIPLE ?ob m:supplyObs ?nb) )))).
```

D. Unified Cube

A *Unified Cube* contains an analysis subject (i.e., fact) described by a set of analysis axes (i.e., dimensions). A measure from one source may only be computed with regards to a subset of levels on a dimension (i.e., a sub-dimension). A generalization of the *dimension-measure* relationship is needed to associate a measure with a sub-dimension starting from any level in *Unified Cube*.

Definition 5. A *Unified Cube* is a n-dimensional finite space describing a fact with some dimensions. A *Unified Cube* is denoted as $UC = \{F; \mathcal{D}; \mathcal{LM}\}$, where

- F is a *fact* containing a set of *measures*;
- $\mathcal{D} = \{D_1; \ldots; D_n\}$ is a finite set of *dimensions*;

[6] https://www.w3.org/2001/sw/DataAccess/rq23/rq24-algebra.html

- $\mathcal{LM}$: $2^{\mathcal{L}^1_{\setminus l_p} \times \ldots \times \mathcal{L}^n_{\setminus l_q}} \rightarrow m_e$ is a *level-measure* mapping which associates a set of summarizable levels with a measure m_e ($m_e \in \mathcal{M}$), such as $\forall i \in [1..n]$, $\mathcal{L}^i_{\setminus l_k}$ ($k \leq |\mathcal{L}^i|$) is the set of levels on the sub-dimension $D_{i\setminus l_k}$ of D_i ($D_i \in \mathcal{D}$) 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 instances (e.g., *correlative* mapping and *rollup* mapping).

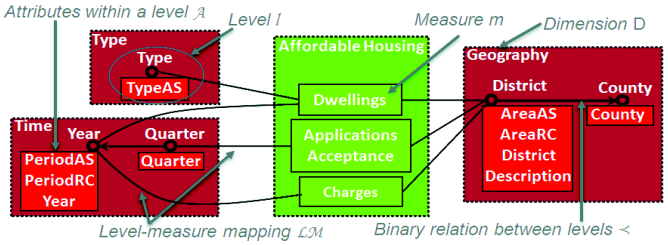

Fig. 2. A *Unified Cube* about affordable housings.

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 $\mathcal{D}=\{D_{Geography}; D_{Time}; D_{Type}\}$. 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:

$$\mathcal{LM}: \{ \{\mathcal{L}^{Time}_{\setminus l_{Year}}; \mathcal{L}^{Geography}\} \rightarrow \{m_{Charges}\}; \{\mathcal{L}^{Time}; \mathcal{L}^{Geography}\} \rightarrow \{m_{Applications}; m_{Acceptance}\}; \{\mathcal{L}^{Time}_{\setminus l_{Year}}; \mathcal{L}^{Geography}; \mathcal{L}^{Type}\} \rightarrow \{m_{Dwellings}\}\}$$

IV. Implementation of a Unified Cube

In this section, we describe a generic, reusable implementation framework for *Unified Cube*s (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:

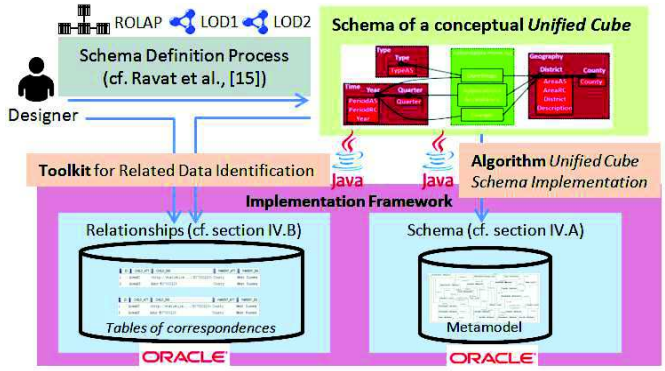

Fig. 3. Implementation framework for *Unified Cube*s

- a metamodel for *Unified Cube*s 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.

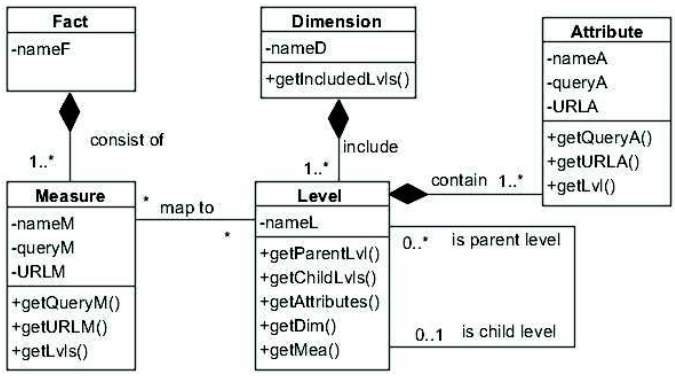

Fig. 4. Metamodel of *Unified Cube*s' schema.

In this metamodel, each class implements one concept of *Unified Cube*s. Note that *binary* relations between level (i.e., $\preccurlyeq$) and *level-measure* mappings (i.e., $\mathcal{LM}$) 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 Cube*s' 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 $m_{Charges}$.

Algorithm 1. *Unified Cube Schema Implementation*
Input: A ***Unified Cube***$=\{F; \mathcal{D}; \mathcal{LM}\}$.
Output: An instantiated metamodel.

Begin

For each $D_i \in \mathcal{D}$
  For each $l_d \in \mathcal{L}^{D_i}$
    Instantiate the Level class: l_d^{meta} = new Level(n^{l_d});
    For each $a_x \in \mathcal{A}^{l_d}$
      Translate the query algebra of the extraction formula E^{a_x} *into a query* Q_{a_x};
      Get the attribute's query endpoint URL_{a_x};
      Instantiate the Attribute class: a_d^{meta} = new *Attribute* (n^{a_x}, Q_{a_x}, URL_{a_x});
      Instantiate the association between a_d^{meta} *and* l_d^{meta}
    End for
  End for
  For each $l_e \preccurlyeq^{D_i} l_f$ (l_e, $l_f \in \mathcal{L}^{D_i}$)
    Instantiate the association between child level l_e^{meta} *and the parent level* l_f^{meta};
  End for
End for
Instantiate the Fact class F^{meta} = new Fact(n^F);
For each $m_g \in \mathcal{M}^F$
  Translate the query algebra of the extraction formula E^{mg} *into a query* Q_{m_g};
  Get the measure's query endpoint URL_{m_g};
  Instantiate the Measure class: m_g^{meta} = new *Measure* (n^{mg}, Q_{m_g}, URL_{m_g});
  Instantiate the association between m_g^{meta} *and* F^{meta};
  Find the set of levels $\mathcal{L}^r{}_{\setminus l_h} \times \ldots \times \mathcal{L}^t{}_{\setminus l_k}$ *associated with* m_g, $\mathcal{L}^r{}_{\setminus l_h} \times \ldots \times \mathcal{L}^t{}_{\setminus l_k} \subseteq \mathcal{L}^1 \times \ldots \times \mathcal{L}^n$, $\mathcal{LM}$: $\mathcal{L}^r{}_{\setminus l_h} \times \ldots \times \mathcal{L}^t{}_{\setminus l_k} \rightarrow m_g$
  For each level $l_h \in \mathcal{L}^r{}_{\setminus l_h} \times \ldots \times \mathcal{L}^t{}_{\setminus l_k}$
    Instantiate the association between l_h^{meta} *and* m_g^{meta};
  End for
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 I: Identification of Related Data.

In the context of *Unified Cube*s, 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:

let $a_{i1}, a_{i2} \in \mathcal{A}^{l_i}$, $a_{j1}, a_{j2} \in \mathcal{A}^{l_j}$ ($l_i \preccurlyeq l_j$):

$\mathcal{C}^{l_i}$: $dom(a_{i1}) \rightarrow dom(a_{i2}) \wedge \mathcal{R}^{l_i}$: $dom(a_{i2}) \rightarrow dom(a_{j2})$
$\Rightarrow \exists \mathcal{R}^{l_i}$: $dom(a_{i1}) \rightarrow dom(a_{j2})$

$\mathcal{R}^{l_i}$: $dom(a_{i2}) \rightarrow dom(a_{j2}) \wedge \mathcal{C}^{l_j}$: $dom(a_{j2}) \rightarrow dom(a_{j1})$
$\Rightarrow \exists \mathcal{R}^{l_i}$: $dom(a_{i2}) \rightarrow dom(a_{j1})$

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 $a_{PeriodAS}$ and $a_{PeriodRC}$ 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 $a_{PeriodAS}$ and $a_{PeriodRC}$). 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 *deductive*, because it is identified by using additional processing methods. For instance, the *correlative* mapping between the attributes $a_{District}$ and a_{AreaAS} (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.

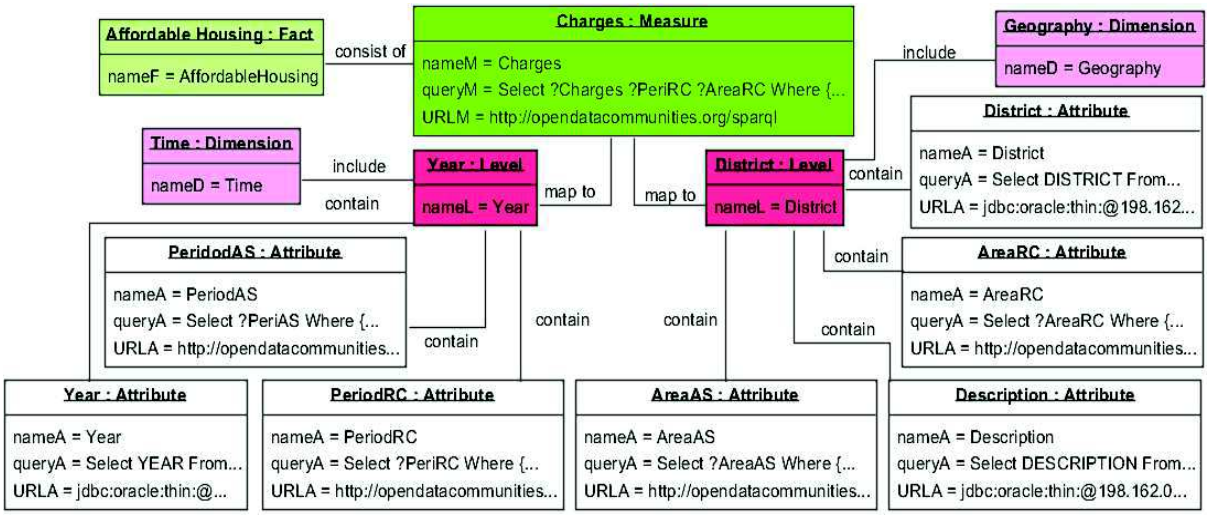

Fig. 5. Extract of metamodel instantiated with the ROLAP DW and the two LOD datasets

TABLE II. LIST OF MAPPINGS AND IDENTIFICATION METHODS

Mapping	Method
1. $\mathcal{C}^{l_{\text{District}}}$: $dom(a_{\text{District}}) \rightarrow dom(a_{\text{Description}})$	n/a
2. $\mathcal{C}^{l_{\text{District}}}$: $dom(a_{\text{District}}) \rightarrow dom(a_{\text{AreaAS}})$	*substring test* similarity
3. $\mathcal{C}^{l_{\text{District}}}$: $dom(a_{\text{AreaAS}}) \rightarrow dom(a_{\text{AreaRC}})$	intermediate ontology[7]
4. $\mathcal{C}^{l_{\text{Year}}}$: $dom(a_{\text{PeriodAS}}) \rightarrow dom(a_{\text{PeriodRC}})$	n/a
5. $\mathcal{C}^{l_{\text{Year}}}$: $dom(a_{\text{Year}}) \rightarrow dom(a_{\text{PeriodAS}})$	*substring test* similarity
6. $\mathcal{R}^{l_{\text{District}}}$: $dom(a_{\text{District}}) \rightarrow dom(a_{\text{County}})$	n/a
7. $\mathcal{R}^{l_{\text{District}}}$: $dom(a_{\text{AreaAS}}) \rightarrow dom(a_{\text{County}})$	reasoning (cf. mappings 2 and 6)
8. $\mathcal{R}^{l_{\text{District}}}$: $dom(a_{\text{AreaRC}}) \rightarrow dom(a_{\text{County}})$	reasoning (cf. mappings 3 and 7)
9. $\mathcal{R}^{l_{\text{Quarter}}}$: $dom(a_{\text{Quarter}}) \rightarrow dom(a_{\text{Year}})$	n/a
10. $\mathcal{R}^{l_{\text{Quarter}}}$: $dom(a_{\text{Quarter}}) \rightarrow dom(a_{\text{PeriodAS}})$	reasoning (cf. mappings 5 and 9)
11. $\mathcal{R}^{l_{\text{Quarter}}}$: $dom(a_{\text{Quarter}}) \rightarrow dom(a_{\text{PeriodRC}})$	reasoning (cf. mappings 4 and 10)

b) Step II: Materialization of Related Data.

To avoid repetitive execution of processing methods during analyses, mappings involving *deductive* relations should be materialized in the *Relationships* module for future uses. The point at issue here is whether it is feasible to materialize *correlative* mappings and *rollup* mappings in a *Unified Cube* without saturating the implementation framework. To answer this question, we show an extreme case by calculating the maximum number of attribute instances which can be possibly materialized, so that we shall know to what extent the volume of the materialized data can reach.

A *correlative* mapping associates one attribute instance with at most one instance of another attribute. Therefore, for a pair of attributes a_x and a_y (a_x, $a_y \in \mathcal{A}^{l_d}$), attributes instances that can possibly be materialized corresponds to the smaller set of attribute instances (i.e., $\min\{|dom(a_x)|, |dom(a_y)|\}$). Within a level l_d ($l_d \in \mathcal{L}^{D_i}$), there exist at most $\frac{|\mathcal{A}^{l_d}| \times (|\mathcal{A}^{l_d}|-1)}{2}$ non-redundant pairs of attributes that can possibly be associated together through *correlative* mappings. Let $\mathcal{C}_k^{l_d}$: $dom(a_x) \rightarrow dom(a_y)$ be a *correlative* mapping between attributes a_x and a_y, for a n-dimensional *Unified Cube*, the *Relationships* module materializes at most the following number of attributes instances involved in *correlative* mappings.

$$\sum_{i=1}^{n} \sum_{d=1}^{|\mathcal{L}^{D_i}|} \sum_{k=1}^{\frac{|\mathcal{A}^{l_d}| \times (|\mathcal{A}^{l_d}|-1)}{2}} min\{|dom(a_x)|, |dom(a_y)|\}$$

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_x and a_z ($a_x \in \mathcal{A}^{l_d}$, $a_z \in \mathcal{A}^{l_e}$, $l_d \preccurlyeq l_e$) is $|dom(a_x)|$. 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.

$$\sum_{i=1}^{n} \sum_{d=1}^{|\mathcal{L}^{D_i}|-1} \sum_{x=1}^{|\mathcal{A}^{l_d}|} |dom(a_x)|$$

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 *deductive* 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 (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.

[7] http://opendatacommunities.org/data/ons-geography-administration

ID	CHILD_ATT	CHILD_INS	PARENT_ATT	PARENT_INS
1	AreaRC	<http://statistics..../E07000223>	County	West Sussex
2	AreaAS	Adur E07000223	County	West Sussex

(a) Table of correspondences for *correlative* mappings.

ID	ATTRIBUTE	INSTANCE	COR_ATT	COR_INS
1	AreaAS	Adur E07000223	AreaRC	<http://statistics..../E07000223>
2	District	Adur	AreaAS	Adur E07000223

(b) Table of correspondences for *rollup* mappings.

Fig. 6. Two types of tables of correspondences.

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).

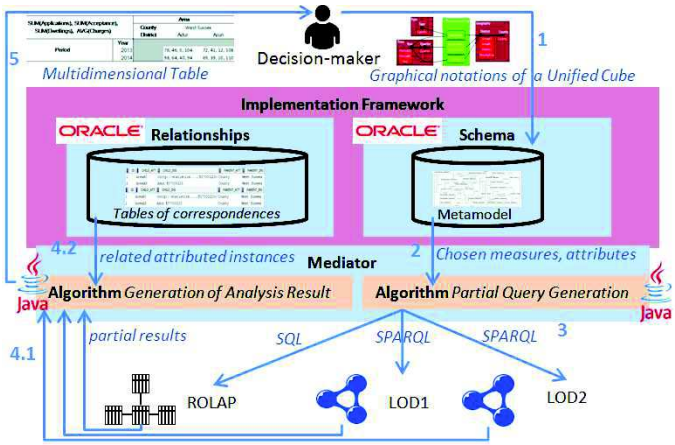

Fig. 7. Analysis processing within the implementation framework

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 $\mathcal{A}_{sub}$ ($\mathcal{A}_{sub} \subseteq \bigcup_{D_i \in \mathcal{D}} \bigcup_{l_k \in \mathcal{L}_i} \mathcal{A}^{l_k}$ [8]), a set of chosen measures $\mathcal{M}_{sub}$ ($\mathcal{M}_{sub} \subseteq \mathcal{M}^F$)

Output: A set of partial queries Q

Begin

1. For each $m_e \in \mathcal{M}_{sub}$
2. *Get associated levels of* m_e: $\mathcal{L}^{m_e} = m_e$.getLvls();
3. *Get the query of the measure* $q^{m_e} = m_e$.getQueryM();
4. *Remove irrelevant attributes from* q^{m_e}
5. For each $a_x \in \mathcal{A}_{sub}$
6. *Get the level of* a_x: $l^{a_x} = a_x$.getLevel();
7. If m_e.getURLM() $\neq a_x$.getURLA()
   Find an attribute a_y, *such as* a_y.getLvl()$=l^{a_x}$ $\wedge$ m_e.getURLM() $= a_y$.getURLA();
8. *Let* $a_x = a_y$;
9. End if
10. *Get the query* q^{a_x} *of the attribute* a_x;
11. *Add* q^{a_x} *in* q^{m_e};
12. *Find a set of level* $\mathcal{L}_{sub}$ ($\mathcal{L}_{sub} \subseteq \mathcal{L}^{m_e}$) such as $\forall l_j \in \mathcal{L}_{sub}$, l_j.getDim()$=l^{a_x}$.getDim() $\wedge$ $l_j \preccurlyeq l^{a_x}$;
13. If $|\mathcal{L}_{sub}| > 1$
14. *Add join predicates between* m_e *and* a_x *in* q^{m_e};
15. *Add a grouping predicate at the end of* q^{m_e};
16. End if
17. End for
18. $Q = Q \cup q^{m_e}$;
19. End for

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., $m_{Applications}$) for social housings with the average rental *charges* (i.e., $m_{Charges}$) by *year*

[8] $\forall D_i \in \mathcal{D}$, $\bigcup_{l_k \in \mathcal{L}_i} \mathcal{A}^{l_k} = \mathcal{A}^{l_1} \cup \ldots \cup \mathcal{A}^{l_{|\mathcal{L}_i|}}$ is the set of attributes within the dimension D_i. Thus, $\bigcup_{D_i \in \mathcal{D}} \bigcup_{l_k \in \mathcal{L}_i} \mathcal{A}^{l_k}$ represents all attributes of a *Unified Cube*.

(i.e., a_{Year})" consists of a cross-source analysis. It calculates two measures from ROLAP DW and LOD2 dataset according to an attribute from the ROLAP DW. Two queries are generated for this analysis after the execution of the algorithm (cf. Fig. 8). The first query Q1 is sent to the ROLAP DW. It contains the measure $m_{Applications}$ and the attribute a_{Year} with corresponding join and grouping predicates. For the second query Q2, the algorithm firstly refers to the metamodel to find a correlative attribute (i.e., $a_{PeriodRC}$) in the LOD2 dataset which is at the same level of a_{Year}. Then a query is directly generated from the *extraction formulae* of $m_{Charges}$ and $a_{PeriodRC}$ by simply adding a grouping predicate.

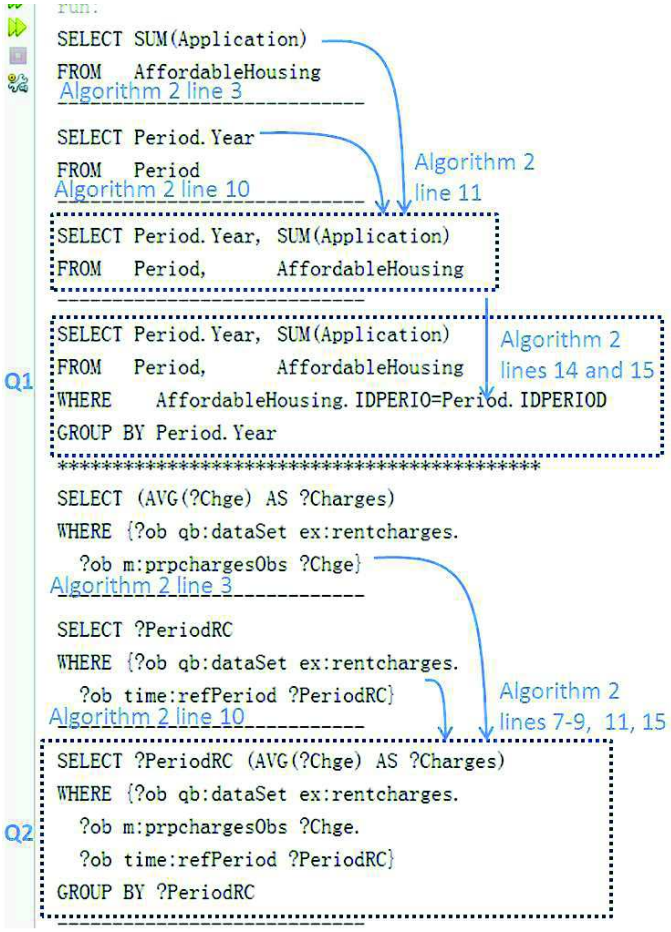

Fig. 8. Generated partial queries

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 *Multidimensional Table*.

Definition 6. A *Multidimensional Table* includes one analysis subject containing measures and up to three analysis axes composed of attributes. A *Multidimensional Table* is denoted as $MT=\{S^{MT}; \mathcal{AX}^{MT}\}$, where:

- S^{MT} is the analysis subject containing a set of measures $\mathcal{M}^{S^{MT}}$ ($\mathcal{M}^{S^{MT}} \subseteq \mathcal{M}$);
- $\mathcal{AX}^{MT}=\{ax_1^{MT};\ldots; ax_m^{MT}\}$ $(1\leq m\leq 3)$ is the set of analysis axes. Each analysis axis ax_i^{MT} ($ax_i^{MT} \in \mathcal{AX}^{MT}$) includes a set of attributes denoted $\mathcal{A}^{ax_i^{MT}}$.

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 $\mathcal{Res}$, each result r_i ($r_i \in \mathcal{Res}$) contains a set of measures $\mathcal{M}^{r_i}$ and a set of attributes $\mathcal{A}^{r_i}$ from one data source; A set of mappings $\mathcal{Map}$, 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* $MT=\{S^{MT}; \mathcal{AX}^{MT}\}$

Begin

1. For each $r_i \in \mathcal{Res}$
2. For each $a_e \in \mathcal{A}^{r_i}$
3. *Get the dimension of* a_e: $D^{a_e}=a_e$.getLvl().getDim();
4. *Add* a_e *in the set of attribute on the corresponding analysis axis of* MT, *such as* $\exists ax_i^{MT} \in \mathcal{AX}^{MT} \wedge ax_i^{MT}= D^{a_e}$: $\mathcal{A}^{ax_i^{MT}} = \mathcal{A}^{ax_i^{MT}} \cup \{a_e\}$;
5. For each attribute instance i^{a_e}, $i^{a_e} \in dom(a_e)$
6. If $\exists f_{ef}$: $dom(a_e) \rightarrow dom(a_f)$ ($f_{ef} \in \mathcal{Map}$) $\wedge$ $a_f \in \mathcal{A}^{ax_i^{MT}}$
7. *Associate* i^{a_e} *with the related instances of the attribute* a_f;
8. Else
9. *Create a new header* i^{a_e} *in the* MT
10. End if
11. End for
12. End For
13. For each $m_j \in \mathcal{M}^{r_i}$
14. *Add* m_j *in* $\mathcal{M}^{S^{MT}}$;
15. For each measure value v^{m_j}, $v^{m_j} \in val(m_j)$
16. If there exists a measure value v^{m_k} ($m_k \in \mathcal{M}^{S^{MT}}$) described by the same related attributes instances as v^{m_j} in the MT
17. *Display* v^{m_j} *in the same cell as* v^{m_k};
18. Else
19. *Display* v^{m_j} *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 Discussions

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 Cube*s 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 Cube*s. 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

[1] R. Kimball and M. Ross, *The data warehouse toolkit: the complete guide to dimensional modeling*, 2nd ed. New York: Wiley, 2002.

[2] A. Abelló, O. Romero, T. B. Pedersen, R. Berlanga, V. Nebot, M. J. Aramburu, and A. Simitsis, "Using Semantic Web Technologies for Exploratory OLAP: A Survey," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 2, pp. 571–588, Feb. 2015.

[3] M. E. Zorrilla, J.-N. Mazón, Ó. Ferrández, I. Garrigós, F. Daniel, and J. Trujillo, Eds., *Business Intelligence Applications and the Web: Models, Systems and Technologies*. IGI Global, 2012.

[4] F. Bauer and M. Kaltenböck, *Linked open data: the essentials: a quick start guide for decision makers*. Wien: ed. mono/monochrom, 2012.

[5] C. Bizer, P. Boncz, M. L. Brodie, and O. Erling, "The meaningful use of big data: four perspectives -- four challenges," *ACM SIGMOD Rec.*, vol. 40, no. 4, p. 56, Jan. 2012.

[6] S. Laborie, F. Ravat, J. Song, and O. Teste, "Combining Business Intelligence with Semantic Web: Overview and Challenges," in *INFormatique des Organisations et Systemes d'Information et de Decision (INFORSID 2015)*, Biarritz, France, 2015.

[7] R. P. Deb Nath, K. Hose, and T. B. Pedersen, "Towards a Programmable Semantic Extract-Transform-Load Framework for Semantic Data Warehouses," 2015, pp. 15–24.

[8] V. Nebot, R. Berlanga, J. M. Pérez, M. J. Aramburu, and T. B. Pedersen, "Multidimensional Integrated Ontologies: A Framework for Designing Semantic Data Warehouses," in *Journal on Data Semantics XIII*, vol. 5530, Springer Berlin Heidelberg, 2009, pp. 1–36.

[9] O. Romero and A. Abelló, "Automating multidimensional design from ontologies," in *international workshop on Data warehousing and OLAP*, 2007, pp. 1–8.

[10] L. Etcheverry, A. Vaisman, and E. Zimányi, "Modeling and Querying Data Warehouses on the Semantic Web Using QB4OLAP," in *Data Warehousing and Knowledge Discovery*, vol. 8646, Cham: Springer International Publishing, 2014, pp. 45–56.

[11] A. Abelló, J. Darmont, L. Etcheverry, M. Golfarelli, J.-N. Mazón, F. Naumann, T. Pedersen, S. B. Rizzi, J. Trujillo, P. Vassiliadis, and G. Vossen, "Fusion Cubes: Towards Self-Service Business Intelligence," *Int. J. Data Warehous. Min.*, vol. 9, no. 2, pp. 66–88, 32 2013.

[12] A. Matei, K.-M. Chao, and N. Godwin, "OLAP for Multidimensional Semantic Web Databases," in *Enabling Real-Time Business Intelligence*, vol. 206, Springer Berlin Heidelberg, 2015, pp. 81–96.

[13] R. Ramakrishnan and J. Gehrke, *Database management systems*, 3rd ed. Boston: McGraw-Hill, 2003.

[14] M. Golfarelli and S. Rizzi, *Data warehouse design: modern principles and methodologies*. New York: McGraw-Hill, 2009.

[15] F. Ravat, J. Song, and O. Teste, "Designing Multidimensional Cubes from Warehoused Data and Linked Open Data," in *IEEE International Conference on Research Challenges in Information Science*, Grenoble, France, 2016, pp. 171–182.

[16] E. Rahm and P. A. Bernstein, "A survey of approaches to automatic schema matching," *VLDB J.*, vol. 10, no. 4, pp. 334–350, Dec. 2001.

[17] P. A. Bernstein, J. Madhavan, and E. Rahm, "Generic schema matching, ten years later," *Proc. VLDB Endow.*, vol. 4, no. 11, pp. 695–701, 2011.

[18] A. DOAN and A. Y. HALEVY, "Semantic integration research in the database community: A brief survey.," *AI Mag.*, vol. 26, no. 1, pp. 83–94, 2005.

[19] S. Duan, A. Fokoue, and K. Srinivas, "One Size Does Not Fit All: Customizing Ontology Alignment Using User Feedback," in *The Semantic Web – ISWC 2010*, vol. 6496, P. F. Patel-Schneider, Y. Pan, P. Hitzler, P. Mika, L. Zhang, J. Z. Pan, I. Horrocks, and B. Glimm, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 177–192.

[20] P. O'Neil, E. O'Neil, X. Chen, and S. Revilak, "The Star Schema Benchmark and Augmented Fact Table Indexing," in *Performance Evaluation and Benchmarking*, vol. 5895, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 237–252.

[21] P. Wang, B. Wu, and B. Wang, "TSMH Graph Cube: A novel framework for large scale multi-dimensional network analysis," 2015, pp. 1–10.

[22] M. Gyssens and L. V. S. Lakshmanan, "A Foundation for Multi-dimensional Databases," in *VLDB '97 Proceedings of the 23rd International Conference on Very Large Data Bases*, Athens, Greece, 1997, pp. 106–115.

[23] A. Maniatis, P. Vassiliadis, S. Skiadopoulos, Y. Vassiliou, G. Mavrogonatos, and I. Michalarias, "A Presentation Model & Non-Traditional Visualization for OLAP," *Int. J. Data Warehous. Min.*, vol. 1, no. 1, pp. 1–36, 2005.