OLAP for Multidimensional Semantic Web Databases

Adriana Matei(✉), Kuo-Ming Chao, and Nick Godwin

Faculty of Engineering and Computing, Coventry University,
Priory Street, Coventry CV1 5FB, UK
{adriana.matei,k.chao,csx0l4}@coventry.ac.uk

Abstract. Semantic Web (SW) and web data have become increasingly important sources to support Business Intelligence (BI), but it is difficult to manage due to its scalability in their volumes, inconsistency in semantics and complexity in representations. On-Line Analytical Processing (OLAP) is an important tool in analysing large and complex BI data, but it lacks the capability of processing disperse SW data due to the nature of its design. A new concept with a richer vocabulary than the existing ones for OLAP is needed to model distributed multidimensional semantic web databases. In this paper we proposed a new OLAP framework with multiple layers including additional vocabulary, extended OLAP operators, and SPARSQL to model heterogeneous semantic web data, unify multidimensional structures, and provide new enabling functions for interoperability. We present the framework with examples to demonstrate its capability to unify RDF Data Cube (QB) [2] and QB4OLAP [1] with additional vocabulary elements to handle both informational and topological data [3] in Graph OLAP. It is also able to compose multiple databases (e.g. energy consumptions and property market values etc.) to generate observations through semantic pipe-like operators.

Keywords: On-Line Analytical Processing · Business Intelligence · Semantic web · Data management · RDF vocabulary

1 Introduction

In today's business, the data (e.g. web data) obtained over the Internet and their semantics can play an important role as resources in enhancing data analysis, when used in combination with internal enterprise business information systems. The Semantic Web (SW) technologies provide the capability of annotating web data with semantics hence generating Semantic Web data.

The information and activities in a typical BI scenario can be modelled by three different layers [5]: the data source layer, the integration layer and the analysis layer. The combination of Data Warehouses (DWs) and On-Line Analytical Processing (OLAP) covers these layers in order to support BI efficiently. OLAP tools and

This research is carried out as a part of CASSANDRA Project, funded by the European Community's Seventh Framework Program FP7-ICT-2011-7 under grant agreement No. 288429.

M. Castellanos et al. (Eds.): BIRTE 2013 and 2014, LNBIP 206, pp. 81–96, 2015.
DOI: 10.1007/978-3-662-46839-5_6

algorithms have been used successfully in BI to query large multidimensional (MD) databases or DWs for supporting decision making. In the middle layer the multidimensional model is used for normalizing and formatting the data, gathered from other sources, for subsequent analysis. The MD dataset representation is done through the OLAP Cube which is built from the data source using the ETL (extract, transform and load) process.

The evolution of the data management on SW data has recently showed an increase in the use of the OLAP approaches to improve efficiency. In order to perform OLAP over SW data, the data has to be modelled with a specific vocabulary and structure to comply with the facilities or engines that OLAP requires. Various MD models for enabling OLAP to operate over Semantic Web data resulted in the development of different structures and vocabularies which form autonomous and heterogeneous OLAP databases for handling semantic (linked) data [1, 2, 6, 7, 11, 12]. As a consequence, different OLAP databases based on proprietary structures with inconsistent query languages making it hard for individuals to communicate and share data with each other when joined datasets are required from multiple individuals.

There was no research conducted so far towards a method that enables multiple SW OLAP databases to be simultaneously accessible over the Internet, even though such demand is increasing. To respond to such queries is a complex task which needs a middleware with OLAP facilities and Semantic Web features to realize it. Furthermore, this type of system should provide explicit, expressive and consistent vocabulary for modelling data and offer full support for OLAP.

2 Research Context

An increasing number of large repositories containing semantically annotated data is available over Internet, but summarising the semantic data to support decision making is not a trivial task due to its scalability and complexity. The utilisation of OLAP capability in organising semantic web data into statistical or concise information can increase efficiency in analysis and visualisation. The implementation of OLAP analysis over a semantic web (SW), however, was understood differently and as such two main types of approach were adopted. Firstly, OLAP is performed after retrieving multidimensional information from the Semantic Web and stored in traditional databases. The second targets OLAP operations directly over RDF data. As for the first approach, storage of semantic web data in local DWs conflicts with the dynamic nature of web data, as OLAP is designed for static and batch offline processing. In addition, the manually built DWs cannot automatically reflect changes in the sources so it is hard to maintain the consistency between them.

On the other hand in order to perform OLAP over SW data there are a set of key aspects needed in the modelling process. There is a need for a precise, explicit describing vocabulary in order to represent OLAP data consistently. The key concepts of dimension and measure need to be introduced to support OLAP operations since they employ measures such as AVG, MIN, SUM etc. and dimension related actions such as roll-up, dice, slice, and drill.

SW data are, however, often published on the web in different cube representations for OLAP operations. As a consequence these generated multidimensional semantic web databases become standalone databases, so they only offer limited OLAP capabilities and only work with their own query languages. The information contained in these web databases can be incomplete for complex applications which may require information from multiple databases. Their proprietary specifications do not give the possibility to communicate or share with each other to compose appropriate responses. This is complicated when queries need to be performed over disparate data sources for new multidimensional semantic web databases.

2.1 Household Energy Consumption Profile Example

There are situations in which it is beneficial and desirable that multiple databases can be accessed simultaneously by complex queries in order to provide adequate answers. Below we introduce such an example in which we consider two different databases with complementary information, which if they are able to communicate, they can provide the data consumers with complete and valuable information.

One large Semantic Web OLAP database *DB1* contains detailed energy consumption information of households from different countries as well as properties of households like household income, accommodation size/layout (number of rooms), number of inhabitants, appliances and so on. A separate Semantic OLAP database *DB2* contains information about the historical value on the market of a specific property and its layout.

Energy consumption for households can be viewed in conjunction with different factors such as: number of inhabitants; household income; house size, or, house value, in order to analyse correlations in energy consumption. House energy efficiency profile can be a factor in a house acquisition or renting process, so it is desirable to have access to multiple databases. For example, a natural language version of a query relating to average energy consumption for houses within a selling price range and having a set of other characteristics may be issued by the users as follows:

The average electricity consumption per year of households in a specific area and with a specific layout, based on the number of occupants and the property market value is between a specific ranges (meaning both actual and historical).

This new aggregation can be materialized and stored as a new observation in the queried OLAP or in an independent OLAP structure. From the example, the following features are essential in order to satisfy the requests from users:

- Perform OLAP operations over the data
- Access to both databases without changing their structure but being able to generate the results
- Both databases have an OLAP structure in which basic OLAP operations such as AVG, SUM, COUNT can be applied as multi-level and multi-dimensionally
- Build OLAP observations in a common format
- Be able to perform data merging for building the response or to materialise it in a new database

In order to offer a solution for the above example, we need to provide a way in which the query is able to distribute to multiple databases, perform OLAP over each database and compose the results retrieved from them.

2.2 Related Work

On-Line Analytical Processing (OLAP) has been undeniably proven a successful approach to analysing large sets of data [5]. Furthermore OLAP is an approach that can be built on top of different database models and respond to multi-dimensional queries as long as they fall under some evaluation criteria regarding, but not limited to, multidimensionality, accessibility, transparency, dimensions and aggregation levels. Recently, a considerable stream of works [1–5, 9–11] was directed towards online analytical processing on informational network and mostly focusing on the Semantic Web data. Chen et al. [3, 9, 10] and Zhao et al. [4] both take the first step to introduce graphs in a multidimensional and level context by proposing conceptual frameworks for graph data cubes and a data warehousing model able to support graph OLAP queries. They both consider attribute aggregations and structure summarization, where the authors in [3] classify their framework into topological and informational OLAP based on the dimension. They proposed different aggregation functions to build summarisations and these cannot be mutually applied.

Kämpgen and Harth [12] introduce linked data transformations for OLAP analysis and they [6] try to map statistical Linked Data to an OLAP to conform to the RDF Data Cube Vocabulary [2] but they did not provide sufficient semantics that are required from the topological elements to build parts of the multiple dimensions. Etcheverry and Vaisman [11] introduce Open Cubes which focus on the publication of multidimensional cubes on the Semantic Web and they found the limitation of the RDF Data Cube [2] which can only address statistical data. Their work revolves around informational OLAP aggregations. Furthermore they revisit RDF Data Cube (DC) by extending DC's capabilities to support multidimensional levels to build hierarchies and to implement other OLAP operators beside the sole *Slice* operator offered by DC. Beheshti et al. [7] continue the work from [3, 4, 9, 10] and offered a graph data model for OLAP informational networks. The approach supports the description of entities and relationships between them and provides both topological aggregations. They use three levels of partitioning conditions to implement their proposed model as well as an adapted query language extended from SPARQL in order to support necessary n-dimensional computations. The aforementioned works do not only show the diversity of the approaches towards online analytical processing of Semantic Web but also the rapid change in the research direction.

RDF Data Cube Vocabulary (QB) [2] focuses on the adherence to Linked Data principles while publishing statistical data and metadata using RDF. QB4OLAP [1] introduces an extended vocabulary of QB in order to support OLAP operators directly over RDF representation. As it will be seen in Sect. 3.1. QB4OLAP introduces levels, members and aggregated functions in order to represent OLAP dimension structure which is not offered by QB vocabulary.

With all these, QB4OLAP, however, does not support a vocabulary to model online analytical processing on graphs introduced by Chen et al. [3]. Zhao et al. [4] introduced a data warehousing model that supports OLAP queries on graph and Graph Cube. None of these [7] provides a semantic-driven framework considering both informational and topological dimensions of graphs.

Beheshti et al. concentrate their approach on topological graphs without considering informational graphs. This is an important factor as semantic data is usually found in a mix of topological and informational graphs. Furthermore, in order to address topological dimensions constrains for OLAP, they use partitioning and an adapted SPARQL query to operate over the data. This approach hinders the published datasets being reused or being queried by applications and users against other datasets offering automated OLAP observations.

In order to reuse and extend existent implementations while extending OLAP capabilities to both topological and informational dimensions, we used the vocabulary in QB and QB4OLAP as basis to form a new vocabulary. Furthermore we introduced new elements and relationships able to model the topological OLAP. By describing topological and informational elements in the same vocabulary and identifying the relationships between entities we enable OLAP to operate over both aspects.

Research on bringing the pipe concept to the Semantic Web was introduced by Morbidoni et al. [8], where their focus was to build RDF-mashups by fetching RDF models on the Web and producing an accessible output. While the Semantic Pipes operators can access different RDF graphs and produce outputs to be consumed by other pipes, they do not offer means to access summary data or support OLAP operations.

The brief introduction of the up-to-date research in the Semantic Web's data management shows that a new model is required to answer computational intensive semantically queries but no existing OLAP system is capable of accessing, retrieving, and reusing semantic OLAP databases efficiently. In order to address this challenge we introduce a new model which can interpret a query based on the OLAP concept. The model offers standard OLAP functionalities with a built-in Pipe concept by extending existing OLAP systems with observations generated from individual RDF graphs or other SW OLAP. This new model is equipped with facilities for composing multiple queries to operate on multiple OLAP databases. It also provides an extended vocabulary for modelling semantic data for OLAP operations.

3 Conceptual Framework

A key factor in successfully performing OLAP over Semantic data is to acknowledge the characteristics and the relationship of data. The relationships between the data can be divided in two categories: *informational* (dimensions are coming from node attributes) and *topological* (when dimensions are coming from node and edge attributes). Some databases may be structured using one type (e.g. DB2) while others may have a mix of structures with the information offered from different dimensions (e.g. DB1). An example of a mix structure can be found in Fig. 1.

A middleware system is needed to perform collective OLAP operations over multiple databases to store the newly generated views in a multidimensional database as well as having an expressive vocabulary to model both topological and informational structure. Even though multiple semantic OLAP databases are accessible, composing retrieved data from them is a complex process. A pipe architectural style can be designed to handle RDF and summary data that can be fed into OLAP functions to support decision making.

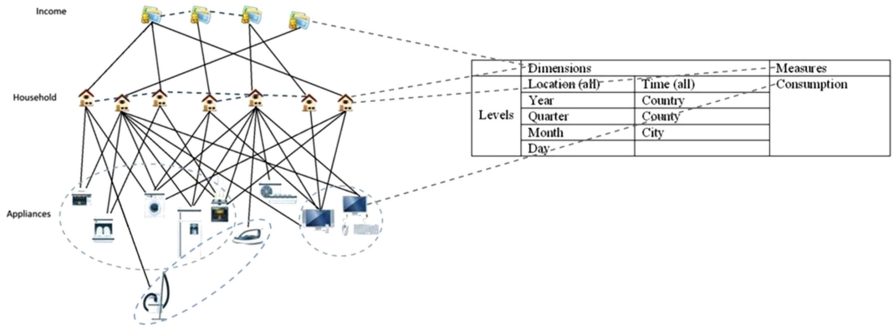

Fig. 1. Connections between topological and informational dimensions

This paper asserts that the key elements for composing such complex results are not yet fully available. Some related work, presented in the next subsection, has been proposed but each approach has limitations.

In order to provide OLAP functionality over multiple Semantic databases, our proposed model (IGOLAP) in Fig. 2 presents a three level contribution:

- An integrated system for collective querying over multiple multidimensional databases
- An extended vocabulary for multidimensional data representation
- A materialization of semantic OLAP database capability

The proposed conceptual framework with multiple layers is to address the issues identified and discussed in the previous sections.

On the bottom layer we have the raw data from relational databases and web data in different forms. In the case of data stored in relational databases, the layer on top of it provides multidimensional modelling of data. For the web data, there is an intermediate layer between raw data and data modelling for OLAP. This layer is described by linked data, which is a specific type of the semantic web data. This layer can also be an intermediate layer between data in relational databases and the modelling layer, when data is transformed from relational databases to linked data [12] before further OLAP analysis. Regarding the multidimensional modelling layer, the data is transformed into cubes for multidimensional models. It contains a series of different vocabularies which trigger different semantic OLAP databases, so this layer can have different representations of data for OLAP. We introduce an extended representation with an enhanced vocabulary and functionalities lacking in other existing vocabularies on the layer to support it. The proposed vocabulary and the functionalities are presented in subsection 3.1.

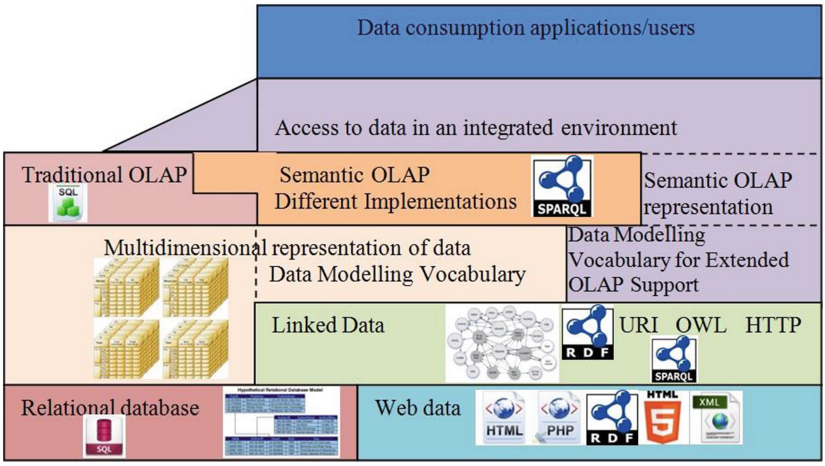

Fig. 2. Conceptual framework

The proposed framework is a multiple layer Semantic OLAP database which is able to handle data in dimensions, levels and measures in order to respond to OLAP related queries.

The top layer in the framework provides users with interfaces to specify queries and visualise the retrieved information in relation to business intelligence or decision making. The other layers provide necessary mechanisms and functions to transform the requests into executable syntax. The framework also increases interoperability among different semantic OLAP databases. So, a query can be executed to locate datasets, retrieve data, summarise information and compose semantics from various semantic OLAP databases. In order to support this functionality we introduce a pipe architecture and distributed query processing as detailed in Sect. 3.2.

3.1 Vocabulary for Modelling Multidimensional Graph Data

As mentioned in Sect. 2, there is existing work regarding a vocabulary for multidimensional data modelling for OLAP support. We consider that the QB vocabulary does not have sufficient capabilities to handle OLAP, but it has adequate structure. The QB4OLAP vocabulary is an extended version of QB, offering more functionality to support OLAP. Both vocabulary sets have missing facilities in relation to modelling two groups of data: Informational (dimensions are coming from node attributes) and Topological (when dimensions are coming from node and edge attributes). Their vocabulary needs to be extended and altered in order to provide full OLAP capabilities. Since an informational graph is modelled by dimensions and hierarchical levels and the topological graph is modelled in dimensions, members and defined relationships, the type of aggregations over their measures are very different. On the informational graphs the standard measure aggregations such as SUM, AVG, and COUNT are used to summarise the data, but the topological graphs require relationship type of aggregations. To design a unified semantic OLAP to handle both graphs is not trivial.

Considering that the dimensions in the topological structure do not have levels but direct members we introduced two different classes to model it: ***igolap:InfoDimension*** and ***igolap:TopoDimension*** as subclasses of the ***qb:DimensionProperty*** class. The property that connects these two classes to their superclass is: ***igolap:dimensionType***. The new vocabulary is presented in Fig. 3 and the comparison of the vocabularies' capabilities is presented in Table 1.

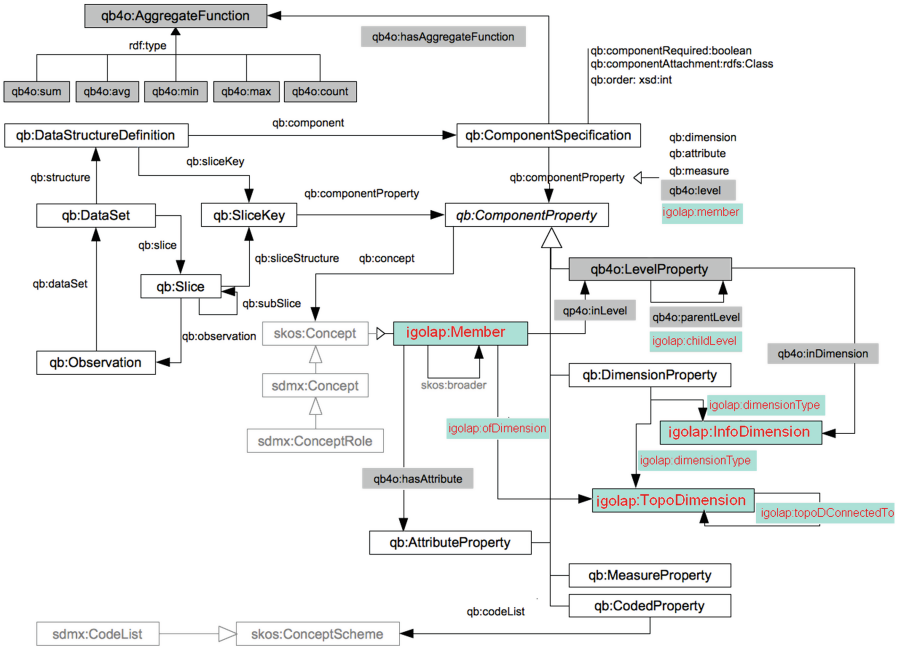

Fig. 3. IGOLAP vocabulary

The existing ***qb4o:LevelMember*** has to be altered in order to handle the topological dimension. We introduced the modified ***igolap:Member*** which while keeping the connection with qb4o:LevelProperty it also has a new connecting property to the topological dimension: ***igolap:ofDimension***. Since both topological and informational dimension have attributes, the property ***qb4o:hasAttribute*** had to be altered to reflect this. The informational dimension has levels which have members and the topological dimension has direct members. In order for the property ***qb4o:hasAttribute*** to apply to both topological and informational dimension, it has to connect the ***igolap:Member*** to ***qb:AttributeProperty***.

The topological dimensions can be connected to each other through a topological property and each member of the dimension holds the property. We introduced the ***igolap:topoDConnectedTo*** property to define those connections.

QB4OLAP introduces ***qb4o:parentLevel*** property which connects levels and can support the roll-up operation, but in order to offer a better support the Drill-down

Table 1. The differences between these three vocabularies, on both classes and properties

	QB	QB4O	IGOLAP
New classes	Attachable; DataSet;	LevelProperty	Member
	CodedProperty; AttributeProperty;	LevelMember	InfoDimension TopoDimnsion
	ComponentProperty; ComponentSet;	AggregateFunction	
	ComponentSpecification; SliceKey; Observation; MeasureProperty;		
	DataStructureDefinition; Slice;		
	DimensionProperty		
Properties	attribute; codeList; componenet; componentAttachment; component Property; dimension;	level	dimensionType, of Dimension, topo DConnectedTo
		inLevel	
	componentRequired; measure;	inDimension	
	concept; dataSet; slice; sliceKey;	parentLevel	member
	measureDimension; subSlice;	hasAggregate Function	childLevel
	measureType; observation; order; sliceStructure; structure;		
Altered Classes	–	LevelMember	Member
Properties	–	has Attribute	hasAttribute
		inDimension	inDimension

operation we introduced the ***igolap:childLevel*** property which also connects levels and is an inverse function to qb4o:parentLevel.

3.2 Integrated System for Collective Query of Semantic OLAP Databases

The proposed framework is based on a set of specialized OLAP operators (Federated OLAP operators) that can operate over multiple semantic OLAP databases, merge the outputs into a common format and translate them according to the desired output which can be materialized or viewed.

The Federated OLAP operators need to interpret the requests according to a specific OLAP database in order to retrieve the data and convert it to a requested output format. The Federated OLAP operators represent an extension of the classic OLAP operations as: roll-up, dice, drill down or slice.

A **roll-up operation** assumes a data summarization inside a given cube alongside a given dimension such as a given Cube C, a dimension D ∈ C and a dimension level lu∈ D, the Roll-up (C,D, lu) will return a new cube C' where measures are aggregated along D up to the level lu.

In the **dice operation** a new cube C' is generated from a given cube C and a set of constrains along its dimensions. The emerging cube has the same schema as the initial cube C and the instances in C' are also instances of C.

Slice operation receives a cube C, and a dimension D ∈ C and returns a sub cube C', with the same schema except the dimension D.

Drill down is considered to be the reverse of Roll up and assumes the disaggregation on a previously stored aggregation.

The dimension operations that are used in our approach are defined as F_Operators. These include the standard dimension operators as F_ROLL_UP, F_DICE, F_SLICE and F_DRILL. They are derived from the standard OLAP dimension operations, but they are adapted to have the necessary functions to access multiple semantic databases.

The standard OLAP measure operations are used as restriction functions in the dimension operations that include AVG for retrieving the arithmetical mean of a set of numerical values, SUM for the sum of a set of numerical values, COUNT for the cardinality of a set of elements and MIN and MAX for the minimum and maximum element of a set of elements.

We briefly introduce the F_ROLL_UP operator in the following subsections.

3.2.1 F_ROLL_UP Overview

The F_ROLL_UP operator includes a set of processes. For the retrieval stage, the operator identifies the targeted databases, builds the SELECT operators for each database with given constraints and gathers information from multiple databases by applying the built operators to specific datasets. In the building stage, the CONSTRUCT operator is initiated to compose the response from the retrieved data. When the datasets retrieved by each SELECT operator are in the same format, the CONSTRUCT operator is applied directly, but if the datasets have different formats, data normalisation is performed before generating the output. In order to handle the data exchange, the F_ROLL_UP operator is described as a pipe architecture containing a CONSTRUCT operator and a number of SELECT operators. If data normalisation is required before the output is generated the third operator, the MERGE operator, is included in the F_ROLL_UP pipe construction. The MERGE operator is used to structure the partial RDF triple results from the SELECT operators using the same vocabulary for the output construction. Even though the MERGE concept has some similarity with the one in the semantic web pipes, the MERGE from semantic web pipes is a simple join of the CONSTRUCT and/or SELECT operators output without normalisation capabilities and facilities to support OLAP.

Since F_OPERATOR's are designed to access one or more than one OLAP database, they require a set of arguments in order to interpret the requests. Based on the arguments received, F_ROLL_UP distinguish between:

- single or multiple database access;
- formatted or unformatted output;
- request for view or request for materialization of the output, and, so on

This means that the parameters can be divided into two main categories: the mandatory and the optional ones (e.g. materialised or immaterialised output represents an optional parameter). The mandatory parameters that need to be passed on are: location of accessed SW OLAP implementation(s) (URIs or IRIs), dimensions (and dimension level for F_ROLL_UP) and some others.

Assuming the example of a request of a ROLL-UP operation across two databases, defined by a F_ROLL_UP, the following steps describe the full process:

Input: ***olapds1*** is the data set of an implemented OLAP data cube ***C***;
olapds2 is the data set of an implemented OLAP cube K
constructSet is the observation building requirements set;
constrs1is the observation constrains for building for C,
constrs2is the observation constrains for building for K,
swOimpl1 represents the first SW OLAP that needs to be queried
swOimpl2 represents the second SW OLAP that needs to be queried

Output:*o1* is an observation generated by roll-up OLAP operation, from a specific a given level which can be materialized or not. Since in this example there is no request for materialization, the observation is outputted as a onetime view.

```
1.     Validate the f_roll-up request, search for valid call
implementation (set of checks as:  if (Count(swOimpl)=i) then
(Count(olapds)=i)||(Count(constrs)=i))
2.     Call of roll-up operator as:f_rollup(swOimpl1, swOimpl2,
olapds1, olapds2, constrSet, constrs1, constrs2)
3.     If the number of swOimpl parameters is bigger than "1"
then the f_rollup call is for multiple database access.
4.     If swOimpl1 equals swOiml2 then {
         Build f_rollup pipe as:
5.        CONSTRUCT(constrSet) <-  (SELECT(olapds1,constrs1),
SELECT(olapds2,constrs2))}
6.      Else{ Build f_rollup pipe as:
7.       CONSTRUCT(constrSet) <-  MERGE((SELECT(olapds1,
constrs1), SELECT(olapds2,constrs2)))}
8.     If materlize_view request exists
9.      Then MATERIALIZE(graph_location, graph_name)
10.     Else step 11 is skipped and standard RDF Graph returned
```

Firstly the request is validated by verifying the number of parameters and their types. The second step is to determine if F_ROLL_UP needs to access a single database or a multiple semantic OLAP databases. According to the outcomes of the previous steps, the operators decide the next tasks, to which a set of given parameters is passed. Step 4 and 5 show the construction of roll-up operator accessing a single database. Step 6 and 7 describe the multi-database access. In the above example no parameter is given to instruct the production of a materialised output. In this case Step 9 and 10 are skipped and the response is produced only for visualisation.

3.3 Multidimensional and Multi-databases OLAP

In this section, we will use the scenario presented in Sect. 2 to show how both informational and topological structures can be implemented using our new vocabulary elements. The content and structure of DB1 in the scenario are described in Figs. 4 and 5 and it contained curated data of both type of structures. Due to space restrictions, we omit the dataset prefixes, only introduce F_ROLL_UP operator, and demonstrate its application to DB1. The descriptions of other operators will be covered in future publications.

Figure 4 shows the structure of informational dimensions of an energy consumption database, DB1. It shows the representation of time and location dimensions structure as well as instances of the location. The structure of informational dimension is very similar

```
e:location a golap:InfoDimension.

e:country a qb4o:LevelProperty;
      qb4o:inDimension  e:location;
      golap:childLevel  e:firstAdministrativeDivision.
e:firstAdministrativeDivision a qb4o:LevelProperty;
      qb4o:inDimension  e:location;
      golap:childLevel  e:secondAdministrativeDivision;
      qb4o:parentLlevel  e:country.
e:secondAdministrativeDivision  a qb4o:LevelProperty;
      qb4o:inDimension  e:location;
      golap:childLevel  e:city;
      qb4o:parentLlevel  e:firstAdministrativeDivision.
e:city a qb4o:LevelProperty;
      qb4o:inDimension  e:location;
      qb4o:parentLlevel  e:secondAdministrativeDivision.

e:time a golap:InfoDimension.

e:year a qb4o:LevelProperty;
      qb4o:inDimension  e:time;
      golap:childLevel  e:quarter;
e:quarter a qb4o:LevelProperty;
      qb4o:inDimension  e:time;
      golap:childLevel  e:month;
      qb4o:parentLevel  e:year.
e:month a qb4o:LevelProperty;
      qb4o:inDimension  e:time;
      golap:childLevel  e:day;
      qb4o:parentLevel  e:quarter.
e:day a qb4o:LevelProperty;
      qb4o:inDimension  e:time;
      qb4o:parentLevel  e:month.
```

```
gn:2635167 a golap:Member;
      qb4o:inLevel  e:country;
      rdfs:label "United Kingdom@en";
      golap:childLevel  gn:6269131.
gn:6269131 a golap:Member;
      qb4o:inLevel e: firstAdministrativeDivision;
      rdfs:label "England@en";
      golap:childLevel  gn:3333134;
      golap:childLevel  gn:3333125.
gn:3333134 a golap:Member;
      qb4o:inLevel e:secondAdministrativeDivision;
      rdfs:label "City of Bristol@en";
      golap:childLevel  gn:2654675.
gn:2654675 a golap:Member;
      qb4o:inLevel e:city;
      rdfs:label "Bristol@en";
      qb4o:parentLevel  gn:3333134.
gn:3333125 a golap:Member;
      qb4o:inLevel e:secondAdministrativeDivision;
      rdfs:label "City and Borough of Birmingham @en"
      golap:childLevel  gn:2655603.
gn:2655603 a golap:Member;
      qb4o:inLevel e:city;
      rdfs:label "Birmingham@en";
      qb4o:parentLevel  gn:3333125.
```

Fig. 4. Informational dimensions: time and location schema and location instances

to the QB4OLAP vocabulary, but the igolap:childLevel in association with qb4o:parentLevel property give the possibility of bidirectional navigation in order to support both roll-up and drill-down OLAP operations.

Figure 5 shows the representation of the topological dimensions that include three dimensions: income, household and appliances. These dimensions are connected by a connecting property. These dimensions do not have a well-defined hierarchical structure but they define aggregations based on common attributes (e.g. number of bedrooms, or number of inhabitants, in the household dimension). We present in Fig. 5 instances of each dimension, with the necessary attributes to populate DB1, producing possible observations.

```
te:incomeRange a golap:TopoDimension;
 golap:TopoConnectedTo  te:household.

te:ukI04 a golap:Member;
 golap:ofDimension  te:incomeRange;
 golap:TopoDConnectedTo  te:hhBrs01;
 golap:TopoDConnectedTo  te:hhBhm73;
 te:min 25000;
 te:max 35000;
 te:currency  "GBP".

te:household a golap:TopoDimension;
 golap:TopoDConnectedTo  te:incomeRange;
 golap:TopoDConnectedTo  te:appliances.

te:hhBrs01 a golap:Member;
 golap:ofDimension  te:household;
 golap:TopoDConnectedTo  te:ukI04;
 golap:TopoDConnectedTo  te:fridge_ZZRB934FW2Brs01;
 golap:TopoDConnectedTo  te:tv_SUE22D5003Brs01;
 te:hasCity gn:2654675;
 te:hasPostCode  "BS35";
 te:size    57;
 te:bedrooms  3;
 te:bathrooms  2;
 te:houseType     "detached";
 te:built   1987.
```

```
te:appliance a golap:TopoDimension;
 golap:TopoDConnectedTo  te:household.


te:fridge_ZZRB934FW2Brs01  a golap:Member;
 golap:ofDimension  te:appliances;
 golap:TopoDConnectedTo  te:hhBrs01;
 te:hasModel  "ZRB934FW2";
 te:hasMake   "Zanussi";
 te:consumption  e:fridge_ZZRB934FW2_20090522;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090523;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090527;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090612;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090719.

te:tv_SUE22D5003Brs01  a golap:Member;
 golap:ofDimension  te:appliances;
 golap:TopoDConnectedTo  te:hhBrs01;
 te:hasModel  "ZRB934FW2";
 te:hasMake   "Zanussi";
 te:consumption  e:fridge_ZZRB934FW2_20090522;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090523;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090527;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090612;
 te:consumption  e:fridge_ZZRB934FW2_electricCons20090719.
```

Fig. 5. Topological dimensions: income, household, appliance and instances

```
e:consumptionYCity a qb:DataStructureDefinition;
      qb:component [qb4o:level e:year];
      qb:component [qb4o:level e:city];
      qb:component [qb4o:measure e:consumption;
qb4o:hasAggregateFunction qb4o:sum];
      qb:component [qb:attribute e:electricMeasureUnit].

e:datasetYCC a qb:DataSet;
rdfs:label "Yearly consumption in a city@en";
qb:structure e:consumptionYCity.

e:consumptionMHAC a qb:DataStructureDefinition;
      qb:component [qb4o:level e:month];
      qb:component [golap:member te:household];
      qb:component [golap:member te:appliance];
      qb:component [qb4o:measure e:consumption;
qb4o:hasAggregateFunction qb4o:sum];
      qb:component [qb:attribute e:electricMeasureUnit].

e:datasetMHAC a qb:DataSet;
rdfs:label "Monthly consumption of household by appliance@en";
qb:structure e:consumptionYCity.
```

```
e:consumption a qb:MeasureProperty

e:electricMeasureUnit a qb:AttributeProperty.
e:kWh a e:electricMeasureUnit;
      rdfs:label "Kilowatt-hour@en".

e:o1 a qb:Observation;
qb:dataSet e:datasetYCC;
e:year db:2009;
e:city gn:2654675;
e:consumption 600000;
e:electricMeasureUnit e:kWh.

e:o2 a qb:Observation;
qb:dataSet e:datasetMHC;
e:month tl:05_2009;
te:household te:hhBrs01;
te:appliance te:fridge_ZZRB934FW2Brs01;
e:consumption 000;
e:electricMeasureUnit e:kWh.
```

Fig. 6. Observations structure definitions and instances

We define the measure for the database and include attribute property which are used in generating observations. In our scenario, we define a measure for consumption (line 23) and its attribute which is measurement unit (e.g. kWh, line 26).

Using the structures, instances, measures and attributes mentioned we can obtain different observations over both topological and informational dimensions, as shown in Fig. 6. We use additional constraints over topological dimensions' attributes and/or different levels of the informational dimensions to structure the observations.

Assume that the proposed system needs to satisfy a query on *yearly energy consumption* of *households* from a specific *city*, based on a set of constraints such as no. of bedrooms and property price range. DB1 can only provide partial information and DB2 with informational dimensions regarding properties' market values in different years, based on the location and property layout details, can offer the complimentary information. This requires federated OLAP operations to retrieve, summarise and compose data from multiple semantic databases.

But the dimensions from different databases can have mismatched structure such as different level of detail in modelling. The location dimension in DB2 has a level called *area* which is a smaller division of *city* in DB1. Another mismatch in structure among them is the existence of a redundant level secondAdministrationDivision from DB1. Figure 7 provides the structure and an instance of area level observation.

```
z:area a qb4o:LevelProperty;
      qb4o:inDimension z:location;
      qb4o:parentLevel z:city.
```

```
z:BS3 a golap:member;
      qb4o:inLevel z:area;
      rdfs:label "Area covering
         Bristol BS3 postcode@en";
      qb4o:parentLevelgn:2654675.
```

Fig. 7. Structure and instance of DB2's location dimension level

Other dimensions used in this database include time and property with the property price as measure. Figure 8a shows observations over the price of a certain type of property, based on the year and the layout of the property.

```
z:pricebyAreaProperty a qb:DataStructureDefinition;
      qb:component [qb4o:level z:area];
      qb:component [qb4o:level z:Year];
      qb:component [qb4o:level z:bedno];
      qb:component [qb4o:measure z:propertyPrice;
                  qb40:hasAggregateFunction qb4o:avg];
      qb:attribute [qb:attribute z:currency].

z:datasetPriceAreaProperty a db:DataSet;
      rdfs:label "Propertis prices by Area, number of
                  bedrooms and year@en";
      qb:structure z:pricebyAreaProperty.

z:obsBS3B1Y2009 a qb:Observation;
      qb:dataSet z:datasetPriceAreaProperty;
      z:area z:BS3;
      z:year db:2009;
      z:bedno 1;
      z:propertyPrice 120236;
      z:currency z:GBP.
```

Fig. 8a. Observation instance of DB2

```
d:consByYearHousehold a qb:DataStructureDefinition;
      qb:component [qb4o:level z:city];
      qb:component [qb4o:level e:year];
      qb:component [qb4o:topoDimension te:household];
      qb:component [qb4o:measure e:consumption;
            qb4o:hasAggregateFunction qb4o:sum];
      qb:component [qb:attribute e:electricMeasureUnit].

d:datasetConsByYearHouseholdCMBI a qb:DataSet;
      rdfs:label "Yearly consumption for households from
            a specific area by number of bedrooms and
            inhabitants and restricted by market
            value@en";
         qb:structure d:consByYearHousehold.
```

Fig. 8b. Observation structure over DB1 and DB2

These standalone observations from both databases give useful overviews of the data but their combination can provide the complete output to the query, for example, on yearly energy consumption of households by performing roll up operations over these databases from and to different levels.

The structure of the desired observation that reflects our query is showed in Fig. 8b and contains information extracted from both DB1 and DB2. The following is an example showing the use of the federated roll-up operator to execute on the semantic databases against a set of constraints.

```
F_roll_up(swOimpl1, swOimpl2, olapds1, olapds2, constructSet,
constrs1, constrs2, true, materialize_location)
```

The following parameters passed to F_ROLL_UP are:

swOimpl1 = “igolap”
swOimpl2 = “igolap”
olapds1 = “http://www.energydb.eu/YearlyByAppliances#”
olapds2 = “http://www.zooplainfodb.org/PriceByArea#”
constructSet = observationStructure.rdf
materialize_location = “http://www.energydb.eu/mixObservations#”

The target of the operator is to perform roll-up operation on *olapds1* to summarise energy consumption from yearly by-appliance to yearly by-household and from area to city level over *olapds2*. The *swOimpl* parameter represents the type of the database,

which also identifies the vocabulary used to structure the data. The *olapds1* and *olapds2* give the locations of data and *constructSet* sets the schema for the CONSTRUCT operator, describing how the observation needs to be built. The *constr1* and *constr2* represent a set of constrains for each database in order to build the appropriate SELECT pipe to retrieve data (e.g. city, price range and number of bedrooms for *olapds2* and household, year, consumption for *olapds1*).

The operator also has the capability to materialise the OLAP observations in datasets that can be used for more complex processing. These can be published with dataset attributes that show the level and type of summarisation performed.

The materialisation capability provides a big advantage when it comes to reanalysing data, or building a new database containing summarised information from different databases.

The F_ROLL_UP operator above is required through the boolean "true" value, to materialise the generated observations in a given location identified by the URI passed through materlize_location parameter.

One of the observations generated from the F_ROLL_UP operator above is:

```
d:oYHCMBI01 a qb:Observation;
       qb:dataSet d:datasetConsByYearHouseholdCMBI;
       z:city gn:2654675;
       e:year db:2009
       te:household te:hhBrs01;
       e:consumption 8800;
       e:electricMeasureUnit e:kWh.
```

This example gives a brief introduction about how the vocabulary can describe observations and use those observations or datasets to provide more meaningful information through OLAP operators over multiple semantic databases.

4 Conclusion and Future Work

The need for accessing multiple Semantic Web multidimensional databases increases, but there is a lack of effective and efficient solutions due to their complexity and inconsistency, and the difficulty in providing scalability. We have shown that there are cases in which standalone semantic web OLAP databases need to communicate with each other, but the diversity of the existing designs on their structures has complicated their interoperability. The incompatibility between informational and topological graph OLAP has deepened the issue of having unified OLAP semantic system.

We proposed an OLAP framework with a multiple-layers mechanism in order to address the challenges.

The requirements for designing the proposed framework have been identified and addressed with a set of new federated OLAP operators, vocabulary and semantic pipe like functions that can retrieve and merge the data from heterogeneous semantic web OLAP databases. We presented one of the operators, F_ROLL_UP, in detail and walked through the steps with an example to demonstrate how it executes queries, locates the data, retrieves the information, and composes the outputs for visualisation.

The proposed new operators and the extended interpreter capabilities to handle multiple OLAP databases have been partially implemented and tested. The rest of the outstanding operators will be implemented and applied to the complex case study in order to evaluate its effectiveness and efficiency.

We recognise the need for querying and analysing data from semantic web and relational multidimensional databases in the future. Our proposed framework is extendable to include retrieval and composition to meet this need and to provide a basis for further analysis. These functions can be realised from the composition of the OLAP operators and the translation of SPARQL queries into specific SQL queries in order to provide complex query responses.

References

1. Etcheverry, L., Vaisman, A.: QB4OLAP: a new vocabulary for OLAP cubes on the semantic web. In: Proceedings of COLD 2012: 3rd International Workshop on Consuming Linked Data, 11–12 November 2012
2. The RDF Data Cube Vocabulary. http://www.w3.org/TR/2012/WD-vocab-data-cube-20120405/. Accessed 5 April 2012
3. Chen, C., Yahn, X., Zhu, F., Han, J., Yu, P.S.: Graph OLAP: towards online analytical processing on graphs. In: Proceedings of the Eighth IEEE International Conference on Data Mining (2008)
4. Zhao, P., Li, X., Xin, D., Han, J.: Graph cube: on warehousing and OLAP multidimensional networks. In: Proceedings of the Sigmod Conference, Athens, Greece (2011)
5. Berlanga, R., Romero, O., Simitsis, A., Nebot, V., Pedersen, T.B., Abelló, A., Aramburu, M.J.: Semantic web technologies for business intelligence. In: Zorrilla, M.E., Mazón, J.-N., Ferrández, Ó., Garrigós, I., Daniel, F., Trujillo, J. (eds.) Business Intelligence Applications and the Web: Models, Systems and Technologies, pp. 310–339 (2012). doi:10.4018/978-1-61350-038-5.ch014
6. Kämpgen, B., O'Rain, S., Harth, A.: Interacting with statistical linked data via OLAP operations. In: Proceedings of the International Workshop on Interacting with Linked Data, pp. 36–49 (2012)
7. Beheshti, S.-M., Benatallah, B., Motahari-Nezhad, H.R., Allahbakhsh, M.: A framework and a language for on-line analytical processing on graphs. In: Wang, X., Cruz, I., Delis, A., Huang, G. (eds.) WISE 2012. LNCS, vol. 7651, pp. 213–227. Springer, Heidelberg (2012)
8. Morbidoni, C., Polleres, A., Tummarello, G., Phuoc, D.L.: Semantic Web pipes. Technical report DERI-TR-2007-11-07, DERI Galway. December 2007
9. Chen, C., Yan, X., Zhu, F., Han, J., Yu, P.S.: Graph OLAP: a multi-dimensional framework for graph data analysis. Knowl. Inf. Syst. **21**, 41–63 (2009). Springer, London
10. Qu, Q., Zhu, F., Yan, X., Han, J., Yu, P.S., Li, Hongyan: Efficient topological OLAP on information networks. In: Yu, J.X., Kim, M.H., Unland, R. (eds.) DASFAA 2011, Part I. LNCS, vol. 6587, pp. 389–403. Springer, Heidelberg (2011)
11. Etcheverry, L., Vaisman, A.A.: Enhancing OLAP analysis with Web cubes. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 469–483. Springer, Heidelberg (2012)
12. Kämpgen, B., Harth, A.: Transforming statistical linked data for use in OLAP systems. In: Proceedings of the 7th International Conference on Semantic Systems, Graz, Austria, pp. 33–40, 07–09 September 2011