International Journal of Web Information Systems

H-SPOOL: A SPARQL-based ETL framework for OLAP over linked data with dimension hierarchy extraction

Takahiro Komamizu Toshiyuki Amagasa Hiroyuki Kitagawa

H-SPOOL
A SPARQL-based ETL framework for OLAP over linked data with dimension hierarchy extraction

Takahiro Komamizu, Toshiyuki Amagasa and Hiroyuki Kitagawa
University of Tsukuba, Tsukuba, Japan

Received 31 March 2016
Revised 18 May 2016
Accepted 18 May 2016

Abstract

Purpose – Linked data (LD) has promoted publishing information, and links published information. There are increasing number of LD datasets containing numerical data such as statistics. For this reason, analyzing numerical facts on LD has attracted attentions from diverse domains. This paper aims to support analytical processing for LD data.

Design/methodology/approach – This paper proposes a framework called H-SPOOL which provides series of SPARQL (SPARQL Protocol and RDF Query Language) queries extracting objects and attributes from LD data sets, converts them into star/snowflake schemas and materializes relevant triples as fact and dimension tables for online analytical processing (OLAP).

Findings – The applicability of H-SPOOL is evaluated using exiting LD data sets on the Web, and H-SPOOL successfully processes the LD data sets to ETL (Extract, Transform, and Load) for OLAP. Besides, experiments show that H-SPOOL reduces the number of downloaded triples comparing with existing approach.

Originality/value – H-SPOOL is the first work for extracting OLAP-related information from SPARQL endpoints, and H-SPOOL drastically reduces the amount of downloaded triples.

Keywords Dimension hierarchy extraction, ETL framework, OLAP for linked data

Paper type Research paper

1. Introduction

Linked data (Bizer *et al.*, 2009) (LD) is a method to publish and connect data using links. As open data movement is popularized, various organizations publish their knowledge and vocabularies on the Web. These individual knowledge and vocabularies are valuable for various applications. What is more, it is possible to increase their values by linking different LD data sets with each other.

LD has been popularized, and there are data sets from various fields published as LD data sets and connected each other. DataHub[1] (one of the most popular data management platforms) contains about 10,000 data sets[2]. Another observation is given by the Linked Open Data Cloud[3] which reports more than 1,000 data sets are published in various domain as LD[4].

Data sets in LD (LD data sets) are written in a standardized format, resource description framework (RDF) (W3C, 2014a). The LD data sets can be obtained in two

This research was partly supported by the program *Research and Development on Real World Big Data Integration and Analysis* of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

International Journal of Web Information Systems
Vol. 12 No. 3, 2016
pp. 359-378
© Emerald Group Publishing Limited
1744-0084
DOI 10.1108/IJWIS-03-2016-0014

major ways; one is to download a dump of the data set, and the other is to query by SPARQL (W3C, 2013) which is a standard dedicated query language for retrieving information in the RDF data. A SPARQL endpoint provides an interface which returns results for SPARQL queries over the back-end databases containing RDF data. Individual LD data set providers construct their own SPARQL endpoints to open their LD datasets. Thus, thanks to SPARQL endpoints, users do not need to download whole LD data sets to process, but the users can obtain a necessary subset of the data set using SPARQL queries.

Analyzing numerical data in LD data sets can expose important facts. Most of LD data sets comprise not only of textual data but also numerical data such as statistics (e.g. population, food consumption and money usage). Prototypical examples of such data sets are included in Data.gov[5] which publish data sets about agriculture, business, climate, education, energy and so forth. In many situations, analysts in general analyze individual data as well as aggregated data and/or finer data to discover some phenomena. However, it is troublesome for analysts to directly handle LD data sets because of their complex data structure (i.e. labeled multigraph). To analyze multidimensional numerical data, analysts typically use online analytical processing (OLAP).

There are three research issues on OLAP over LD data sets:

(1) *Issue 1*. The original structure of LD is not always suitable for OLAP. Even though there are dedicated vocabularies (e.g. QB vocabulary) to describe the OLAP cube structures, there are many data sets which are not written in the vocabularies because publishers may not be supposed to publish the data for OLAP. Thus, LD data sets must be transformed into suitable formats. However, this requires high-level knowledge about the data, which requires a large effort to learn on those who develop an OLAP system for a LD data set.

(2) *Issue 2*. LD data sets have unsuitable characteristics for such conversions, that is, large sizes and updates of the data sets. To convert an LD data set, in a naïve way, the data set itself should be downloaded on the local machine. Even once the OLAP system for the data set is constructed, when the data set has any update, the conversion must be taken again from scratch.

(3) *Issue 3*. Hierarchized dimensions are useful for OLAP, because such dimensions enable more complex OLAP operations such as roll-up and drill-down. However, extracting dimension hierarchy from LD data sets for corresponding dimension requires human burdens. Users who are responsible for designing dimension hierarchies need to observe a number of related properties for each target dimension. In general, choosing appropriate properties from such large candidates is bothersome.

The previous work (Inoue *et al.*, 2013) of this paper attempted to deal with the Issue 1 by transforming LD data sets into type-partitioned triple store (TPTS) and extracting OLAP cubes from the TPTS. The main difference between the previous work and this paper is whether the Issue 2 is taken into consideration. The previous work downloads an LD data set to a local server and performs the transformation. On the other hand, this paper proposes a scheme to download fragments of LD data sets which are smaller than whole data sets. Also, Inoue *et al.* (2013) have not given formal definitions related to TPTS, while this paper gives the formal definitions.

This paper attempts to solve the aforementioned issues by a SPARQL-based ETL framework for OLAP Over LD with dimension Hierarchy extraction (H-SPOOL)[6]. H-SPOOL transforms the LD data sets into TPTS which enables to extract attributive information about classes based on the rdf:type predicates [this solves the Issue 1 as the previous work Inoue *et al.* (2013) did]. H-SPOOL extracts necessary information for OLAP (i.e. star/snowflake schemas, and fact and dimension tables) through SPARQL endpoints without specific knowledge about the target LD datasets (this solves the Issue 2). H-SPOOL reduces the amount for downloading RDF data required to extract OLAP-related data. This is achieved by a series of SPARQL queries; thus, for each query, H-SPOOL only obtains a part of the target LD data set (the part is, typically, significantly smaller comparing with the whole dataset). In the experimental section, the reduction rate of downloaded triples is demonstrated. In addition, to tackle Issue 3, H-SPOOL uses a heuristic approach to extract a dimension hierarchy for a specified dimension.

This paper is organized as follows. Section 2 introduces related work to clarify differences and novelty of this paper comparing with existing researches. Section 3 shows a formal definition of the problem with which this paper is dealing. Then, Section 4 describes H-SPOOL framework including TPTS-based extraction scheme and SPARQL-based materialization strategy. To show H-SPOOL is a feasible approach to solve the problems, Section 5 showcases application scenarios using H-SPOOL framework as well as reduction ratio of downloaded triples comparing with the previous work. Finally, Section 6 concludes this paper and describes future directions of this research.

2. Related work

There has been lots of work studied OLAP or data warehousing for data on the Web as summarized in Pérez *et al.* (2008) and Abelló *et al.* (2015). The former trend (Pérez *et al.*, 2008) was integrating and making OLAP possible for data on the Web written in semi-structured or unstructured data (e.g. XML data and HTML data). The recent trend (Abelló *et al.*, 2015) is related to semantic Web technologies (e.g. RDF, ontologies and linked open data). The related work to this paper can be selected and categorized into:

- ontology-based OLAP; and
- ETL for OLAP.

The former includes dedicated vocabularies within LD data sets, and specific predicates indicate which part of data to be data cube. While, the latter attempts to extract OLAP-related information from LD data sets without any assumption for vocabularies of the data sets. Related work for each category is explained in the following sections.

2.1 Ontology-based online analytical processing

The ontology-based approaches (Kämpgen and Harth, 2011; Etcheverry and Vaisman, 2012, 2012; Kämpgen *et al.*, 2012; Kämpgen and Harth, 2014; Etcheverry *et al.*, 2014; Prat *et al.*, 2012) enable OLAP for LD data sets which are written in the dedicated vocabularies. In such vocabularies, a subgraph in the datasets are explicitly explained as facts, dimensions, hierarchies and others related to OLAP.

The ontology-based approaches attempt to define vocabularies and/or operations which enable OLAP for LD data sets. Kämpgen *et al.* (2011) have proposed an ETL system to apply OLAP for LD data sets including statistical information and written in the RDF data cube vocabulary (or QB vocabulary) (W3C, 2014b). Although QB vocabulary is a standardized vocabulary by W3C, QB vocabulary had some limitations. Thus, Etcheverry and Vaisman (2012) propose another vocabulary, open cube (or OC) vocabulary, which is an extension of data cube vocabulary. OC vocabulary enables OLAP operations (e.g. roll-up and drill-down) for data cube vocabulary. Afterward, (Etcheverry and Vaisman, 2012) propose a combined vocabulary of QB and OC vocabularies, QB4OLAP, to support OLAP operations such as roll-up. On the other hand, Kämpgen *et al.* (2012) give OLAP operation definitions over LD data sets written in QB vocabulary. Also, Kämpgen *et al.* (20140 demonstrate an OLAP system for LD data sets written in QB vocabulary. Coincidently, Etcheverry *et al.* (2014) propose a method to model data warehouses using QB4OLAP vocabulary for OLAP operations. On the other but related direction, Prat *et al.* (2012) propose a method to remodel a multidimensional model by OWL-DL (description logic version of OWL ontology) for verification of models and summarizabilites. Even though a multidimensional model written by OWL-DL can be converted into a normal multidimensional model, at first, the data must be prepared as the model in OWL-DL.

Although, there are lots of effort to realize OLAP for LD data sets by defining dedicated vocabularies, LD data set publishers are not necessarily acquainted with such vocabularies, so they fail to write their data sets in such dedicated vocabularies. For this situation, derivation of multidimensional data structures is necessary to enable OLAP over the LD data sets including numerical data.

2.2 ETL for online analytical processing over linked data

Several works (Niinimäki and Niemi, 2009; Nebot and Llavori, 2012; Inoue *et al.*, 2013; Ibragimov *et al.*, 2015) have tried to enable OLAP for LD data sets without dedicated vocabularies. The earlier works (Niinimäki and Niemi, 2009; Nebot and Llavori, 2012) require human efforts for mapping from LD data sets into OLAP-related schemas and tables. Niinimäki and Niemi (2009) propose ETL process for OLAP using OWL ontology (W3C, 2012). For each data source, mapping to OLAP schema (ROLAP) must be defined by manual in advance. While, (Nebot and Llavori, 2012) propose a way to construct data warehouses by asking users a mapping ontology.

The closest related work to this paper is Ibragimov *et al.* (2015) and the previous work (Inoue *et al.*, 2013) of this paper. The proposal of Ibragimov *et al.* (2015) is twofold: one is a novel multidimensional schema which behaves as a wrapper for an LD data set and the other is source discovery. The multidimensional schema is a combination of QB4OLAP vocabulary (Etcheverry and Vaisman, 2012) and VoID vocabulary (W3C, 2011); the former defines cubes for OLAP on RDF data, and the latter defines LD data sets such as locations of SPARQL endpoints. However, the generation process of the multidimensional schema requires large human efforts, even though the process is aided by machines. The previous work (Inoue *et al.*, 2013) provides ETL framework for downloaded LD data sets at begging; however, this takes time if the data sets are large. While, this paper attempts to provides OLAP analyses without downloading the LD data sets.

3. Basic knowledge and problem formulation

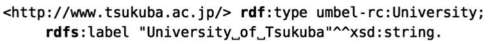
This section shares basic knowledge and gives a problem definition that this paper attempts to resolve. The compositions of the problem are SPARQL endpoints and OLAP. These compositions are introduced as in the following subsections: Sections 3.1 and 3.2. Thereafter, Section 3.3 explains the problem definition.

3.1 SPARQL endpoints

A SPARQL endpoint enables users to access data represented as RDF (W3C, 2014a) in its back-end database using the dedicated query language (W3C, 2013), for RDF data.

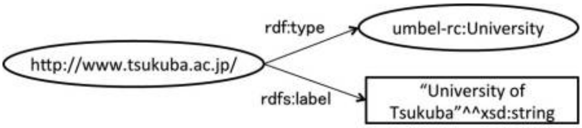
3.1.1 Resource description framework data. RDF data consist of a set of triples, each of which is represented by a triplet of subject, predicate and object. Subject and predicate are represented by uniform resource identifier (URI), and object is represented either URI or literal. RDF can be written by various formats such as Turtle, N-Triples, RDF/XML and so on. Figure 1 shows an RDF graph about "University of Tsukuba" using Turtle format. Table II shows a list of prefixes used in this paper. The Line 1 contains a URI of "University of Tsukuba" as a subject. The Line 2 represents the type (rdf:type) of the subject; the type is umbel-rc:University) which describes university in the UMBEL vocabulary (2016). The Line 3 shows a label (rdfs:label) of the subject. The graphical representation of the following RDF data is depicted in Figure 2.

3.1.2 SPARQL. SPARQL is a query language for RDF data, consisting of triple patterns corresponding to query demands. In SPARQL, query demands are represented by triple patterns [i.e. in the form of triplets (subject, predicate, object)]. Figure 3 showcases a sample SPARQL query. This SPARQL query describes to obtain concatenated predicates and objects on a resource, "www.tsukuba.ac. jp/". On the SELECT clause, desired variables are listed (?p and ?o in this example), while, on the WHERE clause, triple patterns are expressed, which include the desired variables. The results of the query over the RDF data in Figure 1 is as Table I. Note that here shows only a simple example, but, potentially, SPARQL can perform more complicated operations such as filtering on literals (i.e. FILTER), aggregations (e.g. GROUP BY and HAVING), subqueries and so on (Table II).



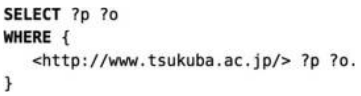
```
<http://www.tsukuba.ac.jp/> rdf:type umbel-rc:University;
    rdfs:label "University_of_Tsukuba"^^xsd:string.
```

Figure 1. RDF data example


rdf:type
umbel-rc:University
http://www.tsukuba.ac.jp/
rdfs:label
"University of Tsukuba"^^xsd:string

Notes: Circles mean resources referred by the URI, rectangles mean literals and arrows between resource and resource/literal mean relationships between them

Figure 2. A graphical view of a fragment of RDF data



```
SELECT ?p ?o
WHERE {
   <http://www.tsukuba.ac.jp/> ?p ?o.
}
```

Figure 3. SPARQL query example

3.1.3 SPARQL endpoints. SPARQL endpoints provide user interfaces (UIs) and application program interfaces (APIs) which enable users or application developers to access to back-end RDF data using SPARQL queries. On a UI, a user input a SPARQL query to the form and submit it, then the results to the query are displayed on the UI. While, through APIs, a program posts a SPARQL query to an API and handle the returned results on the program. The returned results can be formatted in various popular formats (e.g. XML, JSON, plain text, etc.). Such SPARQL endpoints are easily provided using Open Source Software such as Jena (Fuseki: Serving RDF data over HTTP, 2016; Virtuoso Open-Source Edition, 2016) as well as, enterprise software like Oracle Spatial and Graph (2006), Stardog (2016).

3.2 Online analytical processing

OLAP is an analytical processing mechanism for multidimensional data. The following sections explain basic data structure and operations for OLAP.

3.2.1 Data structure. OLAP uses a cube (referred to as OLAP cube) as a core component. An OLAP cube is composed of numeric facts (called measures) and hierarchical and categorical attributes for the measures (called dimensions). OLAP cubes are represented as star schemas or snowflake schemas, which a central table is called fact table, surrounding tables are called dimension tables and a fact table and dimension tables are connected by foreign key relationships. Consequently, a fact table $F = (m_{1\ldots}, m_k, d_1 \ldots, d_n)$ consists of measures $(m_1 \ldots, m_k$, where k is the number of measures) and dimension keys $(d_1 \ldots, d_n$, where n is the number of dimensions). Figure 4 depicts an example of star schemas representing sales data for days and locations. In this example, there are one measure, sales and two dimensions: time and location. The time dimension has a hierarchy consisting of year, month and day, while the location dimension has a hierarchy consisting of continent and country. Table III showcases example fact [Table III(a)] and dimensions tables [Table III(b) and (c)].

3.2.2 Operations. For analyses, users can apply operations to change target records and granularity of aggregations; the operations include slice, dice, roll-up and

Table I. Results of the query in Figure 3 over the RDF data in Figure 1

p	o
rdf:type	umbel-rc:University
rdfs:label	“University of Tsukuba”^^xsd:string

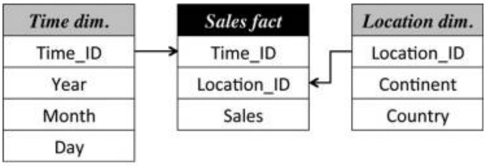
Table II. Prefix list used in this paper

Prefix	URI
rdf	www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	www.w3.org/2000/01/rdf-schema#
umbel-rc	http://umbel.org/umbel/rc/
dbp	http://dbpedia.org/property/
dbr	http://dbpedia.org/resource/
dbo	http://dbpedia.org/ontology/
xsd	www.w3.org/2001/XMLSchema#
fb	http://wifo5-04.informatik.uni-mannheim.de/factbook/ns#
db	http://wifo5-04.informatik.uni-mannheim.de/factbook/resource/

drill-down. Slice and dice operations are used for extracting a particular part of data for analyses. Slice operation is an operation taking out all data having a specified dimension-value. Dice operation is a similar operation with the slice operation, which extracts all data having a specified set of dimension-values. Besides, roll-up and drill-down operations are used for changing dimension granularities. On one hand, roll-up operation increases one level of a specified dimension, that is, the operation enables granular analyses. On the other hand, drill-down operation decreases one level of a specified dimension for finer analyses.

3.3 Problem definition

There are two options for enabling OLAP analyses over an LD data set. One option is to download whole data of the data set and then a processor extracts star/snowflake schemas as well as relations containing facts and dimensions. The other option is to extract these schemas and relations through the SPARQL endpoint of the dataset. As is the case when the size of the data set is so large that it takes long time to download, the former option is not realistic. In addition, if the data set update happens more frequently than the downloadable dumps have been created, the extracted schemas and relations do not reflect the latest state of the data set. Therefore, this work chooses the latter option. Formally, the problem this paper dealing with is as follows:



Notes: The arrow represents primary key foreign key relationships. The *Sales* attribute is the measure

Figure 4. A star schema example consisting of a Sales fact table and Time and Location dimension tables

(a) Fact table

Time_ID	Location_ID	Sales
time_1	Location_1	100
time_2	Location_1	200
time_2	Location_2	150

(b) Time dimension table

Time_ID	Year	Month	Day
time_1	2015	3	10
time_2	2015	3	11

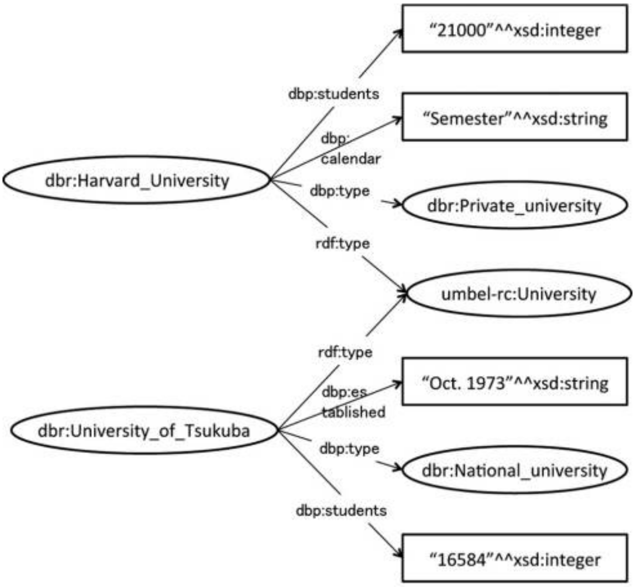
(c) Location dimension table

Location_ID	Continent	Country
location_1	Asia	Japan
location_2	Europe	Belgium

Table III. An example of fact table and dimension tables

- *Input*. SPARQL endpoint URL u which back-end database contains numeric values for analyses; and
- *Output*. OLAP cube for u consisting of star/snowflake schemas and fact and dimension tables.

The problem can be demonstrated using the example depicted in Figure 5 (suppose that the LD data set is inside of the SPARQL endpoint specified by a URL). The goal is to derive star/snowflake schemas like Figure 6, and fact and dimension tables like Table IV. There are two resources which are instances of “umbel-rc:University”, and each of the resources contain common and uncommon predicates and objects. In an OLAPing scenario, each university is considered to be a record, and the integer (i.e. numeric) values of a predicate “dbp:students” (which is the number of students in the university) can be measures, and other predicates can be dimensions. Figure 6 and Table IV display the corresponding star schema, and fact and dimension tables, respectively.



Notes: Two instances of umbel-rc:University class, and each of them is connected with four objects

Figure 5.
An LD data set sample including university information

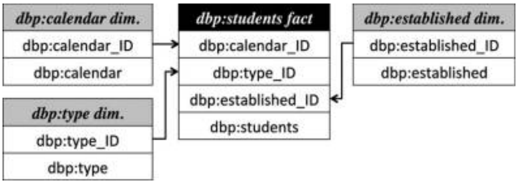


Figure 6.
Star schema extracted from LD data set in Figure 5

(a) dbp:students fact table

dbp:calendar_ID	dbp:type_ID	dbp:established_ID	dbp:students
dbp:calendar_1	dbp:type_1	*null*	21000
Null	dbp:type_2	dbp:established_1	16584

(b) dbp:calendar dimension table

dbp:calendar_ID	dbp:calendar
dbp:calendar_1	Semester

(c) dbp:established dimension table

dbp:established_ID	dbp:established
dbp:established_1	October 1973

(d) dbp:type dimension table

dbp:type_ID	dbp:type
dbp:type_1	dbr:Private_university
dbp:type_2	dbr:National_university

Notes: *Null* means "no value"

Table IV. Fact and dimension tables corresponding with the star schema in Figure 6

4. H-SPOOL: proposed framework

To solve the problem above, this paper proposes H-SPOOL, a SPARQL-based ETL framework for OLAP over LD with hierarchy extraction, which constructs OLAP cubes from remotely located SPARQL endpoints which contain numerical values. H-SPOOL derives fact and dimension tables by TPTS approach, which uses rdf:type information as a clue (Section 4.1). TPTS approach over SPARQL endpoints has several choices of timings for materializing triples for OLAP cube construction. When and how to materialize the triples is discussed in Section 4.2.

4.1 Type-partitioned triple store approach

TPTS approach uses rdf:type as a clue to determine which parts of LD data set are composed as units of information. This approach is based on our previous work (Inoue *et al.*, 2013). TPTS approach focuses upon rdf:type because of the following reasons:

- rdf:type is a most popularly used standardized property for defining class of resources; and
- such a class indicates a unit of information in the LD data set.

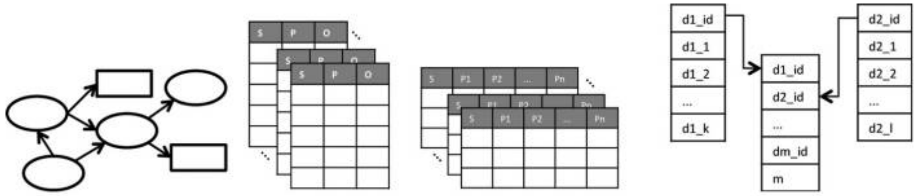
Therefore, TPTS approach first finds class information from LD data sets using rdf:type predicates as well as properties around the class resources, and then extracts schematic information (i.e. star/snowflake schemas) and related tables (i.e. fact and dimension tables).

Figure 7 displays an overview of the TPTS approach. In the first step [Figure 7(a) and (b)], the input LD data set is converted into TPTS to look over possible predicated on a particular class. In the second step [Figure 7(b) and (c)], a table corresponding with a class in TPTS is converted into property table (PT), which flattens the table so that it contains all possible predicates as attributes. The third step [Figure 7(c) and (d)] requires to determine which predicates in a PT to be measures for OLAP. Once measures are determined, other predicates become dimensions. According to the measures and dimensions, star/snowflake schemas are generated. The following sections give basic definitions of the TPTS approach (namely, type-partitioned triple store and property table), and show mechanisms deriving TPTS, PT, schemas and tables.

4.1.1 Type-partitioned triple store. A TPTS holds a set of tables (called type-partitioned tables), each of which is corresponding to a class and contains all triples which subjects are of the class. Thus, the structure of the table is to store the triples [i.e. (subject, predicate, object)] and the types of objects as object_type. Definition 1 gives a formal definition of TPTS. Table V shows an example of type-partitioned tables in TPTS from the LD data set in Figure 5:

D1 (TPTS): Given a set X of classes, a type-partitioned triple store TPTS = $\{T_x\}_{x \in X}$ is composed of a set of type-partitioned tables, each of which T_x corresponds with a class x $\in$ X. Each type-partitioned table $T_x \in$ TPTS has four attributes: subject, predicate, object and object_type, where object_type keeps a data type of object. A type-partitioned table T_x contains quads which subjects are of the class x.

A property table PT_x of a given class x $\in$ X is a flattened table of the type-partitioned table T_x and contains all possible predicates as its attributes. The all possible predicates can be extracted from the type-partitioned table T_x by obtaining distinct predicates in T_x. Definition 2 gives a formal definition of PT. Table VI depicts the PT of umbel-rc: University which type-partitioned table is shown in Table V. In Table VI, object_types



Notes: (a) Input LD data set; (b)TPTS; (c)PT; and (d)Star/snowflake Schema; First, from (a) to (b), the LD data set is converted into TPTS including several type-partitioned tables. Second, from (b) to (c), each type-partitioned table is converted into PT. Finally, from (c) to (d), star/snowflake schema is extracted from the PTs by specifying several predicates as measures and others as dimensions

Figure 7.
An overview of TPTS approach

Table V.
A type-partitioned table of umbel-rc: University from the LD data set in Figure 5

Subject	Predicate	Object	Object_type
dbr:Harvard_University	dbp:students	21000	xsd:integer
dbr:Harvard_University	dbp:calendar	Semester	xsd:string
dbr:Harvard_University	dbp:type	dbr:Private_university	resource
dbr:University_of_Tsukuba	dbp:established	Oct. 1973	xsd:string
dbr:University_of_Tsukuba	dbp:type	dbr:National_university	resource
dbr:University_of_Tsukuba	dbp:students	16584	xsd:integer

are shown in parentheses and corresponding values for each predicate are stored in the column:

D2 (Property table): Given a class x and its type-partitioned table T_x, a property table PT_x for x is defined as a table composed of attributes corresponding with subject and a set of predicate-object_type pairs (predicate, object_type) ϵ P [i.e. (subject,P)] where the set P of predicates are all distinct predicates in T_x.

Each PT is corresponding with a class; there are however relationships among PTs because in PTs may contain subjects of other PTs. Relationships among PTs are useful for expanding possible dimensions through the relationships. Thus, in TPTS approach, the relationship information should be kept. Definition 3 gives a definition of relationships among PTs. Table VII shows an example of relationships related to the example in Figure 5. There are two resources in the PT of umbel-rc:University (Table VI), namely, dbr:Private_university and dbr:National_university (Table VII only shows only two of them):

Definition 3 (Relationships between PTs): A relationship between PTs, PT_x and PT$_y$, is a presence of objects which belong to PT_x and they also belong to P Ty as subjects. The relationship is represented in the following triplet.

$$\langle PT_x, PT_x.attr, PT_y\rangle$$

where PT_x.attr ϵ PT_x.P refers an attribute attr of PT_x which has URIs for subjects in PT$_y$.

4.1.2 Schema generation. Star/snowflake schemas for OLAP are generated using PTs and relationships. To generate star/snowflake schemas, measures must be specified. There are several options for determining measures:

(1) a user specifies predicates from a list of candidate predicates; and

(2) a system automatically decides appropriate predicates.

Since the Option (2) is challenging and out of scope for this paper, this paper chooses the option (1). Therefore, in the assumption, predicates to be measures are given. For a given

Subject	dbp:students (xsd:integer)	dbp:calendar (xsd:string)	dbp:type (resource)	dbp:established (xsd:string)
dbr:Harvard_University	21000	Semester	dbr:Private_university	*null*
dbr:University_of_Tsukuba	16584	*null*	dbr:National_university	October 1973

Table VI. A property table of umbel-rc:University converted from the type- partitioned table in Table V

PT_x	$PT_x.attr$	PT_y
umbel-rc:University	dbp:type	yago:PrivateUniversities
umbel-rc:University	dbp:type	yago:NationalUniversities

Table VII. Relationship examples of PT in Table VI

measure, predicates in the same PT are extracted as dimensions. In addition, if any relationship exists with the PT, predicates in the related PTs are used as dimensions. The procedure to generate a star/snowflake schema is summarized as follows:

- given a predicate p as measure and its property table PT_x, find PTs Y in relation with PT_x;
- join all PTs in Y with PT by the join conditions kept in the relationships, that is:

$$\bigwedge_{PT_y \in Y} PT_y.subject = PT_x.attr$$

  where PT_x.attr is a predicate which is connected with PT_y by referring the relationships. The joined PT is referred to as J_x:
- predicates in J_x.P except the measure p are derived as dimensions and form dimension tables. If a subset of the predicates forms a dimension hierarchy, the subset forms a dimension table. Although there are several choices for constructing dimension hierarchies like Subsumption (Sanderson and Croft, 1999), this paper deals with extracting dimension hierarchies within the LD data sets (the detail is discussed later section); and
- fact table is formed as a tuple of the measure p and dimensions keys.

Take the PT in Table VI and the measure as dbp:students as an example. For simplicity, suppose that no related PTs for the PT. Thus, Step.1 and Step.2 in the procedure are skipped. In Step.3, the predicates other than dbp:students are extracted as dimensions: dbp:calendar, dbp:established and dbp:type. Each dimension table is formed as (id, value) as is depicted in Figure 6.

4.1.3 Dimension hierarchy extraction. To find appropriate predicates for constructing dimension hierarchies, H-SPOOL uses a heuristic approach. Formally, a given class c which is a dimension in the OLAP schema, H-SPOOL finds a predicate which connects c and any higher concepts.

The basic idea of the approach is the highly common predicates among the instances of a dimension (i.e. a class) include higher concepts of dimensions. The closest idea is lowest common ancestor (Eckhardt *et al.*, 2007) on directed acyclic graph, which find a nearest vertex from a given set of vertices. The procedure of finding the predicates for dimension hierarchy is as follows:

- Find the most popular predicate p connected from instances of c. Note that standard predicates defined as rdf, rdfs and owl[7] (e.g. rdf:type and owl:sameAs) are excluded.
- Find common classes which instances are directed by the predicate p. The commonality of the classes can be evaluated in various means (e.g. set overlaps and cosine similarity of TF-IDF). In this paper, we use a TF-IDF-based commonality measure to exclude globally common classes.

The procedure is processed over the relationship table by grouping and aggregation operations.

We take dbo:city as an example to introduce the procedure. First, find the predicates connected with instances of dbo:city using grouping and aggregation operations over the relationship table. The partial results of the aggregation are (dbo:isPartOf, 37,351), (dbp: subdivisionName, 36,501)[…]. Thus, we pick the dbo:isPartOf as the predicate for dimension hierarchy extraction. Second, calculate the commonality of classes of objects of the predicate, dbo:isPartOf. Then, dbo:Region is obtained as the highest commonality score. Hence, we obtain a two-level hierarchy, that is, city/region. We continue the procedure until any loop detected (this can be easily done by checking reoccurrence of classes).

4.1.4 Table generation. Materializing the generated star/snowflake schemas is converting corresponding PTs into the fact and dimension tables. The procedure to materialize the star/snowflake schemas is summarized as follows:

- For each dimension, H-SPOOL extracts unique values of the dimension and assigns an identifier (i.e. primary key) on each value. The values are extracted from the joined PT J_x.
- Fact table is materialized by joining J_x and all dimension tables, and extract the measure p as well as dimension keys.

Take the same example above, in Step.1, dimension tables are materialized as Table IV (b)-(d). In Step.2, with join of the materialized dimension tables, the fact table is constructed as tuples of the measures (i.e. dbp:students) and dimension keys as Table IV(a).

4.2 Triple materialization

In TPTS, there are three possible timings of downloading necessary triples:

(1) in the beginning;

(2) when to construct TPTS; and

(3) when to materialize fact and dimension tables.

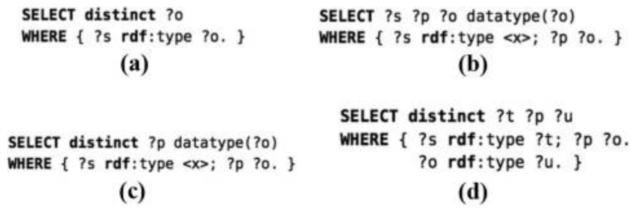
Due to the Issue 2 mentioned in the introduction, Choice (1) is out of consideration in this paper. Choice (2) can reduce the amount of downloaded triples if a user chooses the limited number of classes. In other words, if downloading triples related to all classes, the amount of downloaded triples is almost equal to Choice (1). FINALLY, Choice (3) is more beneficial than Choice (2) because the amount of downloading RDF data in LD data sets is smaller or close to that of Choice (2). Consequently, H-SPOOL provides the timing for downloading RDF data in LD data sets as Choice (3).

TPTS approach extracts three sorts of tables, namely type-partitioned tables, PTs, and relationships between PTs and H-SPOOL generates a series of SPARQL queries to extract them (summarized in Figure 8). First, TPTS requires what classes exist in the given LD data set. To obtain a list of classes, H-SPOOL uses the SPARQL query in Figure 8(a). For each class, H-SPOOL then extracts triples which subject is of the class and stores the triples in a type-partitioned table for the class. This is achieved by the query in Figure 8(b). Note that type-partitioned tables require type information about objects, datatype function[8] is responsible to obtain the type. However, the purpose of obtaining type-partition tables is to extract all possible predicates for each class, hence only predicates and types of objects are necessary to obtain. Therefore, the query [in Figure 8(b)] is converted into the query in Figure 8(c). As a result of this query, PTs for classes are generated. Next step on H-SPOOL is to obtain the relationships among the PTs (equivalent to classes). The query in Figure 8(d) obtains pairs of related classes with predicates between instances of the respective classes.

Given predicates as measures on a class, H-SPOOL generates the star/snowflake schemas and fact and dimension tables. According to the previous section, the schemas are generated from PTs and relationships. On the other hand, to materialize fact and dimension tables, the joined PTs must be materialized beforehand. A SPARQL query q to materialize a joined PT J_x is generated by Algorithm 1. In the algorithm, variables containing objects of the predicates in J_x are kept in the set S, a set of conditions (i.e. graph pattern) is kept in the set W, and then the query q is generated by concatenating the variables S and the conditions W. Note that implode function[9] converts an input set (i.e. the second argument) to string by concatenating each element with a specific string (i.e. the first argument). An example generated query for the PT in Table VI is as Figure 9. Table VI is a part of the results for this query, and the whole results include more universities. From the materialized joined PT, according to the table generation in the previous selection, fact and dimension tables are generated:

Algorithm 1 A SPARQL query generation algorithm for materializing a joined property Table
Input: Joined PT J_x
Output: Query q

```
$S \leftarrow \{“?s”\}$, $W \leftarrow \{“?s \text{ rdf:type}” + x\}$
for $i = 0$ to $|J.P|$ do
    $S \leftarrow S \cup \{“?v” + i\}$
    $W \leftarrow W \cup \{“?s” + J.P[i] + “?v” + i\}$
end for
$q \leftarrow$ “SELECT” + $implode(“ ”, S)$ + “WHERE {” + $implode(“.”, W)$ + “}”
```



```
SELECT distinct ?o
WHERE { ?s rdf:type ?o. }
```
(a)

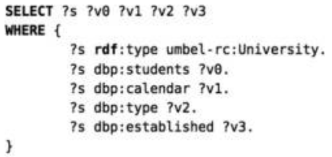
```
SELECT ?s ?p ?o datatype(?o)
WHERE { ?s rdf:type <x>; ?p ?o. }
```
(b)

```
SELECT distinct ?p datatype(?o)
WHERE { ?s rdf:type <x>; ?p ?o. }
```
(c)

```
SELECT distinct ?t ?p ?u
WHERE { ?s rdf:type ?t; ?p ?o.
        ?o rdf:type ?u. }
```
(d)

Notes: (a) Classes; (b) related triples with class x; (c) properties and data types of objects on class x; and (d) relationships among classes

Figure 8. A series of SPARQL queries for the TPTS approach



```
SELECT ?s ?v0 ?v1 ?v2 ?v3
WHERE {
        ?s rdf:type umbel-rc:University.
        ?s dbp:students ?v0.
        ?s dbp:calendar ?v1.
        ?s dbp:type ?v2.
        ?s dbp:established ?v3.
}
```

Notes: SPARQL query for getting instances of Figure 6

Figure 9. Generated SPARQL query by Algorithm 1

5. Experimental evaluation

This experimental evaluation tries to prove applicability and efficiency of H-SPOOL. The applicability is shown by applying H-SPOOL to real data sets, namely, CIA World FactBook and DBpedia. The efficiency is evaluated by reduction rate of downloaded triples comparing with dump downloading. Respective evaluations are shown in the following subsections.

5.1 Empirical study

To prove the applicability of the proposed framework H-SPOOL, H-SPOOL is applied for various real LD data sets published as SPARQL endpoints. The data sets include CIA World FactBook[10] and DBpedia[11]. The main part of H-SPOOL is to extract necessary information for OLAP from the remotely located SPARQL endpoints (or LD data sets) using SPARQL queries. Thus, this empirical study shows the results of each query to present the feasibility of OLAP for the given LD data sets.

The rest of this section displays the results for the series of queries for CIA World FactBook and DBpedia, respectively. For CIA World FactBook, the obtained possible classes in the data set is one class, namely, fb:Country. The following queries are based on the class, fb:Country. A set of triples which subjects is of fb:Country class is obtained as Table VIII. The direct query to provide PT for the class gives results as Table IX. The results of the queries indicate that there are various sorts of predicates which can be measures and dimensions. As the CIA World FactBook data set contains only one class, there exists no relationship.

For the DBpedia data set, there are lots of classes and relationships among them (cf. classes in Table X and relationships in Table XIII); only a part of whole list is shown due to the space limitation. In Table XI and Table XII, because of space limitation, the extracted triples and predicates for class umbel-rc:University are shown. Figures 5, 6 and 4 demonstrate parts of the extracted triples, start/snowflake schema and fact and dimension tables. These figures indicate that H-SPOOL successfully extract the necessary information for OLAP through SPARQL endpoints and is able to construct OLAP system for LD data sets available on the Web Table X-XIII.

5.2 Quantitative evaluation: reduction of the number of downloaded triples

H-SPOOL downloads only schematic triples as well as triples related to fact and dimension tables. This experiment shows that how H-SPOOL effectively reduces the

s	*p*	o	datatype
db:Afghanistan	fb:populationbelowpovertyline	53.0E0	xsd:double
db:Algeria	fb:populationbelowpovertyline	25.0E0	xsd:double
db:Azerbaijan	fb:populationbelowpovertyline	49.0E0	xsd:double
db:Albania	fb:populationbelowpovertyline	25.0E0	xsd:double
. . .	. . .	. . .	. . .
db:Aruba	fb:agestructure_15_64years_male	33553	xsd:long
db:Antigua_and. . .	fb:agestructure_15_64years_male	24137	xsd:long
db:United_Arab. . .	fb:agestructure_15_64years_male	2558029	xsd:long
db:Afghanistan	fb:agestructure_15_64years_male	8668170	xsd:long
. . .	. . .	. . .	. . .

Table VIII. A set of triples which subject is fb:Country with data types of objects on the CIA world FactBook data set

number of downloaded triples compering with the previous work (Inoue *et al.*, 2013). The previous work (Inoue *et al.*, 2013) requires the whole data set to extract OLAP-related information (i.e. type-partitioned triple store, PT and relationship table), that is, it requires the whole set of triples. In contrast, H-SPOOL downloads only necessary triples. Then, H-SPOOL is expected to reduce the number of downloaded triples.

In this experiment, we calculate the reduction ratio of downloaded triples comparing with the previous work. The target LD data set is DBpedia. The reduction ratio r is calculated as the proportion of the number of downloaded triples by H-SPOOL over that of DBpedia.

Figure 10 displays the reduction ratios for corresponding classes. In H-SPOOL, a class is selected as the target for OLAP; thus, reduction ratio is calculated for each class. In the figure, horizontal axis represents classes in descending order to the reduction ratios, and the vertical axis represents reduction ratios. Reduction ratios of most of classes are close to one, and those of all classes are above 0.6. The average reduction

Table IX.
A set of properties and data types of objects for fb:Country class on the CIA world FactBook data set

p	Datatype
factbook:populationbelowpovertyline	xsd:double
factbook:sexratio_totalpopulation	xsd:double
factbook:elevationextremes_lowestpoint_place	xsd:string
factbook:agestructure_15_64years_male	xsd:long
factbook:HIVorAIDS_peoplelivingwithHIVorAIDS	xsd:long
factbook:countryname_locallongform	xsd:string
factbook:lifeexpectancyatbirth_male	xsd:double
factbook:factbookcode	xsd:string
. . .	. . .

Table X.
A set of classes on the DBpedia data set

o
dbo:Scientist
dbo:Country
dbo:Olympics
dbo:Company
umbel-rc:University
. . .

Table XI.
A set of triples which subjects are umbel-rc:University on the DBpedia data set

s	*p*	o	Datatype
dbr:Tulane_University_Sch. . .	dbp:city	dbr:New_Orleans	resource
dbr:Nara_Women's_University	dbp:city	dbr:Nara,Nara	resource
dbr:University_of_Tsukuba	dbp:city	dbr:Tsukuba, Ibaraki	resource
. . .	. . .	. . .	. . .
dbr:Tulane_University_Sch. . .	dbp:country	dbr:United_States	resource
dbr:Nara_Women's_University	dbp:country	dbr:Japan	resource
dbr:University_of_Tsukuba	dbp:country	dbr:Japan	resource
. . .	. . .	. . .	. . .

ratio is about 0.9999. Only a small fraction of classes has lower reduction ratios; thus, in most cases, H-SPOOL downloads a very few triples to construct OLAP systems.

6. Conclusion

This paper has proposed H-SPOOL, a SPARQL-based ETL framework for OLAP over LD. H-SPOOL enables OLAP over LD data sets which contain numerical values even if they are impractical to download to process. To extract star/snowflake schemas as well as fact and dimension tables, this paper proposes TPTS approach which utilizes rdf:type as a clue in LD data sets. The empirical study successfully shows the applicability of H-SPOOL.

As H-SPOOL is still in an early stage to realize OLAP over arbitrary LD data sets, for the future, H-SPOOL has several directions for its extension:

1. update tolerant mechanism for OLAP;
2. supporting more complex aggregation functions [e.g. holistic functions called in Nandi *et al.* (2012) like top-k];
3. SPARQL only OLAPing; and
4. missing rdf:type.

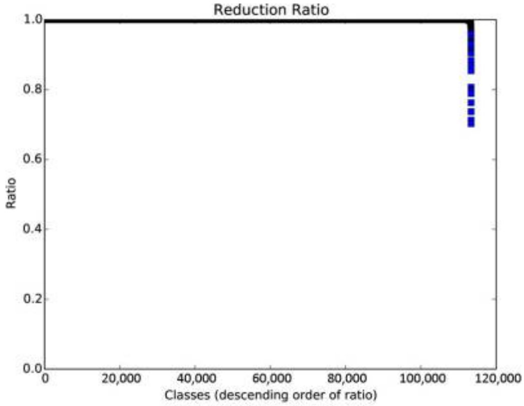
For (1), even current H- SPOOL is able to extract immediately from LD data sets, incremental update scheme of the star/snowflake schemas, and fact and dimension tables are also an important issue. For (2), because of heterogeneity and complicatedness of LD data sets, not only simple analyses but also complex analyses are highly expected. Supporting such complex analyses via SPARQL is challenging. For (3), current H-SPOOL stores fact and dimension tables in a local server, directly OLAPing through SPARQL endpoints is also expected situation in LD (note that some works depending upon dedicated vocabularies are able to do OLAP through SPARQL queries, but they are not applicable if LD datasets are note written in the vocabularies). For (4), even though rdf:type is very powerful to ensure units of information, some LD data sets miss rdf:type or contain inappropriate rdf:type. Thus, another strategy is necessary to determine the targets RDF subgraphs in LD data sets for OLAP.

p	datatype
dbp:country	resource
dbp:type	resource
dbp:lat	xsd:long
dbp:long	xsd:long
dbp:established	xsd:integer
. . .	. . .

Table XII. A set of properties and data types of their objects on the DBpedia data set

t	*p*	u
umbel-rc:University	dbp:country	dbo:Country
umbel-rc:University	dbp:chancellor	dbo:Scientist
umbel-rc:University	dbp:dean	dbo:Scientist
. . .	. . .	. . .

Table XIII. A set of relationships between classes on the DBpedia data set



Notes: Horizontal axis represents classes in descending order of ratio, and vertical axis represents the reduction ratio of each class. Most of dots are close to one and all dots are more than 0.6, that is, a large number of downloaded triples required for OLAP are reduced. Even if users choose class of the smallest reduction ratio, H-SPOOL down oads 60 per cent less number of triples

Figure 10.
Reduction ratio of downloaded triples for each class

Notes

1. available at: http://datahub.io/
2. Available at: http://datahub.io/dataset (accessed 23 July 2015)
3. Available at: http://lod- cloud.net/
4. Available at: http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/ (accessed 23 July 2015)
5. Available at: www.data.gov/
6. This work is an extension of the paper (Komamizu *et al.*, 2015)
7. Available at: www.w3.org/2002/07/owl#
8. Available at: www.w3.org/TR/rdf- sparql- query/#func- datatype
9. Implode function is a function in PHP which concatenates an array of elements with a specified string, and is equivalent to join function in Python and Ruby.
10. Available at: http://wifo5- 03.informatik.uni- mannheim.de/factbook/snorql/
11. Available at: http://dbpedia.org/sparql

References

Abelló, A., Romero, O., Pedersen, T.B., Llavori, R.B. Nebot, V., Cabo, M.J.A. and Simitsis, A. (2015), "Using semantic web technologies for exploratory OLAP: a survey", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 27 No. 2, pp. 571-588.

Bizer, C., Heath, T. and Berners-Lee, T. (2009), "Linked data – the story so far", *Journal on Semantic Web and Information Systems*, Vol. 5 No. 3, pp. 1-22.

Eckhardt, S., Mühling, A.M. and Nowak, J. (2007), "Fast lowest common ancestor computations in dags", *In ESA*, Vol. 4698, pp. 705-716.

Etcheverry, L., Vaisman, A.A. and Zimányi, E. (2014), "Modeling and querying data warehouses on the semantic web using QB4OLAP", *Proceedings of the Data Ware- Housing and Knowledge Discovery – 16th International Conference, DaWaK*, Munich, *2-4 September 2014*, pp. 45-56.

Etcheverry, L. and Vaisman, A.A. (2012), "Enhancing OLAP analysis with web cubes", *Proceedings of the Semantic Web: Research and Applications – 9th Extended Semantic Web Conference*, ESWC, *Heraklion, Crete*, *27-31 May 2012*, pp. 469-483.

Etcheverry, L. and Vaisman, A.A. (2012), "QB4OLAP: a vocabulary for OLAP cubes on the semantic web", *Proceedings of the Third International Workshop on Consuming Linked Data, COLD*, Boston, MA, 12 November, Vol. 12.

Fuseki: serving RDF data over HTTP (2016), available at: http://jena.apache.org/documentation/serving_data/?

Ibragimov, D., Hose, K., Pedersen, T.B. and ZimÂanyi, E. (2015), "Enabling real-time business intelligence", *Towards Exploratory OLAP over Linked Open Data – A Case Study*, pp. 114-132.

Inoue, H., Amagasa, T. and Kitagawa, H. (2013), "An ETL framework for online analytical processing of linked open data", *Web-Age Information Management – 14th International Conference*, WAIM, *Beidaihe*, *14-16 June 2013*, pp. 111-117.

Kämpgen, B. and Harth, A. (2011), "Transforming statistical linked data for use in OLAP systems", *Proceedings the 7th International Conference on Semantic Systems, I-SEMANTICS*, *Graz*, *7-9 September 2011*, pp. 33-40.

Kämpgen, B., Riain, S.O and Harth, A. (2012), "Interacting with statistical linked data via OLAP operations", *The Semantic Web: ESWC 2012 Satellite Events - ESWC 2012 Satellite Events*, *Heraklion, Crete*, *27-31 May 2012*, Revised Selected Papers, pp. 87–101.

Kämpgen, B. and Harth, A. (2014), "OLAP4LD - a framework for building analysis applications over governmental statistics", *The Semantic Web: ESWC 2014 Satellite Events – ESWC 2014 Satellite Events*, Anissaras, Crete, *25-29 May 2014*, Revised Selected Papers, pp. 389-394.

Komamizu, T., Amagasa, T. and Kitagawa, H. (2015), "SPOOL: a SPARQL-based ETL framework for OLAP over linked data", *iiWAS 2015*, *Brussels*, *11-13 December*, pp. 1-10,

Nandi, A., Yu, C., Bohannon, P. and Ramakrishnan, R. (2012), "Data cube materialization and mining over mapreduce", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24 No. 10, pp. 1747-1759.

Nebot, V. and Llavori, R.B. (2012), "Building data warehouses with semantic web data", *Decision Support Systems*, Vol. 52 No. 4, pp. 853-868.

Niinimäki, M. and Niemi, T. (2009), "An ETL process for OLAP using RDF/OWL ontologies", *Journal on Data Semantics XIII*, Vol. 13, pp. 97-119.

Oracle Spatial and Graph (2016), available at: www.oracle.com/database/spatial/index.html

Pérez, J.M., Llavori, R.B., Aramburu, M.J. and Pedersen, T.B. (2008), "Integrating data warehouses with web data: a survey", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 20 No. 7, pp. 940-955.

Prat, N., Megdiche, I. and Akoka, J. (2012), "Multidimensional models meet the semantic web: defining and reasoning on OWL-DL ontologies for OLAP", *Proceedings on the DOLAP, ACM 15th International Workshop on Data Warehousing and OLAP*, Maui, HI, 2 November 2012, pp. 17-24.

Sanderson, M. and Croft, W.B. (1999), "Deriving concept hierarchies from text", *SIGIR Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, pp. 206-213.

Stardog (2016), available at: http://stardog.com/

UMBEL Vocabulary (2016), available at: http://techwiki.umbel.org/index.php/UMBEL_Vocabulary

Virtuoso Open-Source Edition (2016), available at: http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/

W3C (2011), Describing Linked Datasets with the VoID Vocabulary, available at: www.w3.org/TR/void/

W3C (2012), Web Ontology Language (OWL), available at: www.w3.org/2001/sw/wiki/OWL

W3C (2013), SPARQL 1.1 Overview, available at: www.w3.org/TR/sparql11-overview/

W3C (2014a), Resource Description Framework (RDF), available at: www.w3.org/RDF/

W3C (2014b), The RDF data cube vocabulary, available at: www.w3.org/TR/vocab-data-cube/

Corresponding author
Takahiro Komamizu can be contacted at: taka-coma@acm.org