$SETL_{BI}$: An Integrated Platform for Semantic Business Intelligence

Rudra Pratap Deb Nath
Aalborg University
Aalborg, Denmark
rudra@cs.aau.dk

Katja Hose
Aalborg University
Aalborg, Denmark
khose@cs.aau.dk

Torben Bach Pedersen
Aalborg University
Aalborg, Denmark
tbp@cs.aau.dk

Oscar Romero
Universitat Politècnica de Catalunya
Barcelona, Spain
oromero@essi.upc.edu

Amrit Bhattacharjee
University of Chittagong
Chittagong, Bangladesh
amritcsecu@gmail.com

ABSTRACT

With the growing popularity of Semantic Web technologies, more and more organizations natively manage data using Semantic Web standards, in particular RDF. This development gives rise to new requirements for Business Intelligence tools to enable analyses in the style of On-Line Analytical Processing (OLAP) over RDF data. In this demonstration, we therefore present the $SETL_{BI}$ (Semantic Extract-Transform-Load and Business Intelligence) integration platform that brings together the Semantic Web and Business Intelligence technologies. $SETL_{BI}$ covers all phases of integration: target definition, source to target mappings generation, semantic and non-semantic source extraction, data transformation, and target population and update. It facilitates Data Warehouse designers to build a semantic Data Warehouse, either from scratch or by defining a multi-dimensional view over existing RDF data sources, and further enables OLAP-style analyses.

1 INTRODUCTION

Business Intelligence tools allow integrating and analyzing data from multiple sources to facilitate business decisions, often in the style of On-Line Analytical Processing (OLAP). The integrated data are organized in a Data Warehouse, typically designed following the Multidimensional Model (MD), which represents data in terms of facts and dimensions. As source data can be structured, semi-structured, unstructured, or semantic, it is important to consider semantic issues in the integration process, which is typically ignored by traditional (RDBMS-centric) data integration tools [1]. On the other hand, initiatives such as Open Government Data (https://opengovdata.org/) encourage organizations to publish their data using standards and non-proprietary formats [15]. Semantic Web standards fulfill these needs as they allow adding semantics on both data and schema level in the integration process and publish data in RDF using Linked Data principles [5]. To bridge this gap, we present $SETL_{BI}$, a tool that combines Semantic Web and Business Intelligence technologies to define, process, integrate, and query semantic data.

In the remainder of this paper, we refer to a semantically annotated Data Warehouse as a *semantic* DW and represent it as a knowledge base in RDF composed of two components: ABox and TBox [11]. The TBox defines a domain in terms of concepts, properties, and terminological axioms whereas the ABox consists of assertions of the TBox. To define the expressivity of the knowledge base with MD semantics, $SETL_{BI}$ uses the Data Cube (QB) [4] and QB4OLAP [6] vocabularies. In doing so, we can elegantly define a TBox with essential Data Warehouse concepts, such as cube structures, dimensions, levels, level attributes, OLAP operations, and complex hierarchies. On the other hand, $SETL_{BI}$ also allows to enrich the TBox with RDFS/OWL classes and properties. To create the ABox from (non-)semantic sources, we introduce a set of basic semantic ETL operations that can be connected and pipelined to orchestrate an ETL flow from the data sources to the *semantic* DW. Finally, $SETL_{BI}$ provides an interactive interface to enable self-service OLAP analysis over the *semantic* DW.

In summary, $SETL_{BI}$ enables (i) users with a background in Data Warehousing but little-to-no background in Semantic Web technologies to semantically integrate semantic and/or non-semantic data and analyze it in OLAP-style, and (ii) users with basic background in Semantic Web and Data Warehouse technologies to define multi-dimensional views over semantic data and run OLAP-like analysis. Additionally, users can enrich the generated TBox with RDFS and OWL constructs.

2 BACKGROUND AND RELATED WORK

Nowadays, the fusion of Semantic Web and Data Warehouse technologies has become popular. There are two lines of research in this direction: 1) use of ontologies as a canonical data model to physically integrate heterogeneous sources, and 2) specific approaches enabling OLAP analysis over semantic data.

A prominent study [13] related to the first line presents an ontology-based approach for facilitating the construction of an ETL flow. At first, each schema of (structured or semi-structured) data sources and the data warehouse is described by a common graph-based model named the datastore graph. Then, an (OWL-based) application ontology is generated to describe the semantics of the datastore graphs of data sources and Data Warehouse, and

WWW '20 Companion, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7024-0/20/04.
https://doi.org/10.1145/3366424.3383533

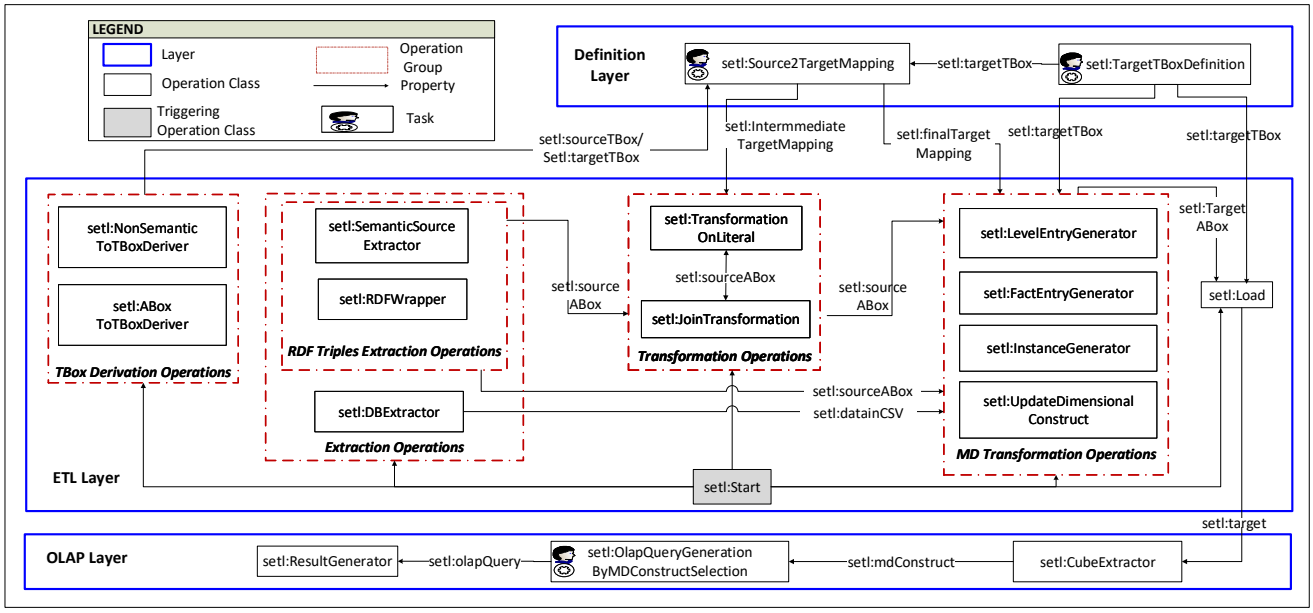

Figure 1: The ontology shows the components of each layer and their connections. The arrows from/to an operation group represent the arrows from/to each individual operation of the group.

mappings between the sources and the Data Warehouse are formulated through the ontology. In this way, they have resolved the heterogeneity issues among the source and target schemata and finally showed how the use of ontologies enables a high degree of automation of ETL flows. However, the integrated data are not annotated with MD semantics to enable OLAP queries.

Related work [12] addressing the second line of research presents a semi-automatic method for inclusion of Semantic Web data into a traditional MD data management system for OLAP analysis. The authors present a methodology which allows *a*) the analyst to design the MD schema from the TBox of an RDF dataset, *b*) to populate the MD fact table after extracting the facts from the ABox of the dataset, *c*) to generate the dimension hierarchies from instances of the fact tables and the TBox so that it enables MDX queries over the data warehouse. However, they have not shown how to update the Data Warehouse with the change of data in a RDF data source. Most importantly, the generated Data Warehouse no longer maintains the Semantic Web data principles; therefore, OLAP-style analysis directly over an RDF dataset remains unaddressed. To address this issue, [3] has introduced a notion of lens called the analytical schema over the RDF dataset, which is a graph of classes and properties. Each node of the schema presents a set of facts that can be analyzed by exploring the reachable nodes.

To enable self-service OLAP analysis over RDF datasets, an OLAP endpoint has been presented in [7]. At first, they superimpose an MD schema over an RDF dataset. Then, semantic analysis graphs are built on top of the MD schema where each node of the graph presents an analysis situation corresponding to a MD query and an edge corresponds to a set of OLAP operations. However, it does not allow end-users to create their own OLAP queries. In [3] and [7], analysts need to create either a lens or a semantic analysis graph to define MD view over a RDF dataset. As a major portion of the published (statistical) Linked Data contains facts and figures, W3C recommends QB [4] vocabulary as a standard to describe data in a MD fashion.

[9] has investigated OLAP operations on a single data cube published with QB vocabulary and shown the applicability of their OLAP-to-SPARQL mapping in answering business questions. Although QB is appropriate to publish statistical data, it has limitations to represent MD semantics properly. QB4OLAP [6] extends QB by providing constructs to define a dimension structure (in terms of levels, the relationship between the levels, and hierarchies of the dimension), a cube structure in terms of different levels of dimensions, measures, and attaching aggregate functions with measures. In [14] and [15], the authors presented a method to semi-automatically enrich the QB dataset with QB4OLAP constructs. However, to run OLAP queries using the OLAP interface of their system, it requires end users to be familiar with either QL [2] or complex SPARQL queries.

[8] has proposed a set of query processing strategies for executing OLAP-like SPARQL queries over a federation of SPARQL endpoints. Here, they use QB4OLAP constructs to annotate the conceptual global schema with MD semantics. However, participating sources in a federation might be unavailable at some point. Data and schemata of the sources might have evolved since the federation was created; thus, integration rules might no longer be valid or history of the data will be lost. Therefore, the standard approach is to avoid federation and have a local copy of the data which is the focus of this research.

The related approaches discussed in this section address one or more parts of our aimed problem, but there is no single solution that supports all the steps (target definition, mappings, ETL generation, target population, evolution and update) necessary to integrate heterogeneous data semantically in a *semantic* DW and enabling OLAP queries on it. $SETL_{BI}$ bridges the two lines of research by defining the TBox of a *semantic* DW with MD semantics using QB

and QB4OLAP constructs, providing RDF-based semantic integration operations to populate/evolve/update the ABox of the *semantic* DW from heterogeneous sources and allowing users to create OLAP queries by using different MD constructs of the *semantic* DW.

3 SYSTEM ARCHITECTURE

The data integration process using $SETL_{BI}$ requires the following steps: 1) defining a target TBox with MD semantics using QB and QB4OLAP constructs, 2) generating mappings from sources to target, 3) populating the target ABox from the available data sources, and 4) issuing OLAP queries on the *semantic* DW. Based on the integration steps, we organize $SETL_{BI}$ into three layers: the *Definition Layer*, *ETL Layer*, and *OLAP Layer*, see in Figure 1. Each layer has a set of tasks and/or operations to accomplish certain integration steps. A task requires user interactions with the system's interface to produce an output while from the given inputs, an operation automatically produces an output. Intuitively, one may consider tasks as defining the required metadata to automate ETL operations. The *Definition Layer* covers the first two steps of the integration process and allows users to define the metadata (target schemas with MD semantics and semantic-aware mappings between the sources and target) of the integration process. The *ETL Layer* covers the third step of the integration process and includes a set of ETL operations to create data flows from sources to target. The *OLAP Layer* allows users to analyze *semantic* DW cubes using a GUI (the final step of the integration process). The inter- and intra-layer connections among the components (operations/tasks) are shown in the ontology in Figure 1 where each component is considered as a class and relationships between components are presented with arrows. In the following, we describe the layers in more detail.

The *Definition Layer* includes two tasks: `setl:TargetTBoxDefinition` and `setl:Source2TargetMapping`. The former supports designing the TBox of a knowledge base with MD semantics and is implemented with a GUI where users can define/edit new/existing cubes, cuboids, dimensions, levels, level attributes, and measures. Internally, the operation annotates the user's input with QB, QB4OLAP and OWL constructs and generates an RDF file. Hence, it relieves TBox designers from the need to learn QB and QB4OLAP vocabularies. The latter also provides a GUI to create mappings across the constructs of (intermediate) source and target TBoxes. Intermediate mappings are required when there is a need of (string/numerical/date) transformation on RDF literals or join of the sources. Hence, it relieves the user from the burden of manual mappings at the ETL operation level.

Figure 1 shows how the intermediate mappings or final mappings are connected to the *ETL Layer* operations. The overall workflow of this operation is illustrated in Figure 2 (A); internally, it creates a mapping file from the user's input in RDF format with our own OWL-based mapping vocabulary S2TMAP (https://github.com/bi-setl/SETL) that allows defining a property-level mapping under a concept-level mapping, which is in turn defined under a mapping dataset. Different to other ETL orchestration tools, this layer proposes a new paradigm: we characterize the ETL flow transformations at the *Definition Layer* instead of independently within each ETL operation (in the ETL layer). This way, the user has an overall view of the process, which generates metadata (the mapping file) that the ETL operators will read and parametrize themselves with automatically.

The *ETL Layer* is composed of a set of ETL operations. Based on their functionality, we categorize the operations into five groups: *TBox Derivation Operations*, *Extraction Operations*, *Transformation Operations*, *MD Transformation Operations*, and *Load operation*. In the *TBox Derivation Operations* group, `setl:NonSemanticToTBoxDeriver` derives TBoxes from non-semantic (CSV, XML, JSON and Database) sources and `setl:ABoxToTBoxDeriver` derives a TBox from an RDF file containing only assertions. In *Extraction Operations*, `setl:RDFWrapper` wraps up data from non-semantic sources to RDF triples; `setl:SemanticSourceExtractor` extracts RDF triples from an RDF data source, and `setl:DBExtractor` extract data in CSV format from a Database source. The *Transformation Operations* group supports numeric, string and date based transformation on the values of RDF properties according to the intermediate mappings using `setl:TransformationOnLiteral` and joins between two RDF sources using `setl:JoinTransformation`. The *MD Transformation Operations* group includes `setl:LevelEntryGenerator`, `setl:FactEntryGenerator`, and `setl:InstanceGenerator` to create level members, observations and instances to create the ABox of a *semantic* DW according to the semantics encoded in the TBox. Those operations support both RDF and CSV input. To reflect the changes in a source to the target, the `setl:UpdateDimensionalConstruct` operation updates the target level members accordingly. This operation supports three types of updates (Type 1, Type 2, and Type 3) defined by Ralph Kimball in [10] for a *semantic* DW. `setl:Load` loads RDF data into either a local RDF file or Jena TDB triple store. Users can drag and drop operations to create ETL flows.

The *OLAP Layer* takes a local RDF file or SPARQL endpoint containing a *semantic* DW, and allows users to create OLAP queries using a GUI. Users first extract the cube structure composed of dimensions, hierarchies, levels, measures and aggregate functions. Then, users create and issue OLAP queries to explore and aggregate measures at various level of details. Figure 2 (B) shows how to create an OLAP query, similar to any traditional OLAP tool. Users can create slice and dice queries adding conditions on selected levels. Internally, we translate the OLAP query generated from the selections into an equivalent SPARQL query. Hence, the users are released of the burden of learning SPARQL. The system is developed in Java 8. All GUIs are implemented in SWT. To process, store and query RDF, we use Jena 3.4.0. As a triple store, we use Jena TDB.

A comprehensive video of $SETL_{BI}$ and its functionality is available at https://youtu.be/b3UdqgLI2Ag. The source code is also available in https://github.com/bi-setl/SETL.

4 DEMONSTRATION

Inspired by the Linked Data principles, the Bangladesh Bureau of Statistics (BBS) wants to publish the 2011 population census in an OLAP-compliant Linked Open Data-format to enable decision making. The dataset consists of 12 smaller datasets where each of them contains approx. 130,000 observations. In this demonstration, we show how a user uses $SETL_{BI}$ layers to accomplish the integration steps (discussed in Section 3), starting with extracting census data in PDF format from the http://203.112.218.65:

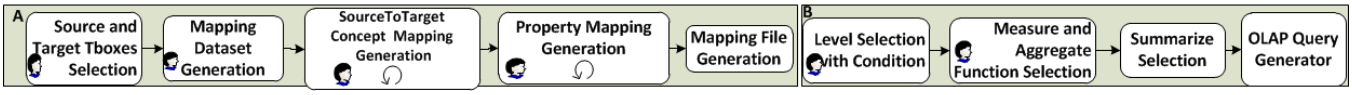

Figure 2: The workflows for Source2Target Mapping(A) and OLAP Query Generation (B) operation. The rectangles, user icons and curved arrows represent automatic, user-required and iterative tasks.

8008/Census.aspx?MenuKey=89 BBS website and converting to CSV. Conference demo users can interact with the system as outlined below.

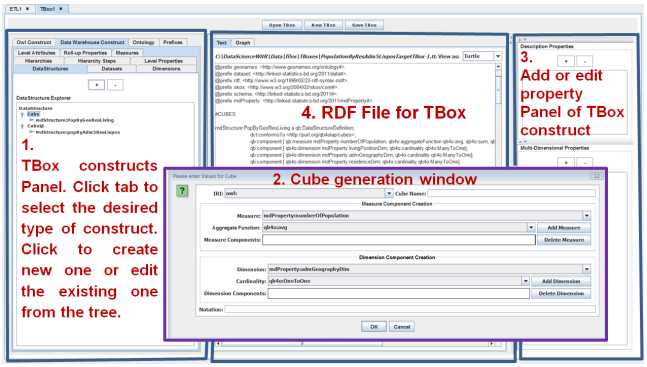

Figure 3: The interface to define a Target TBox

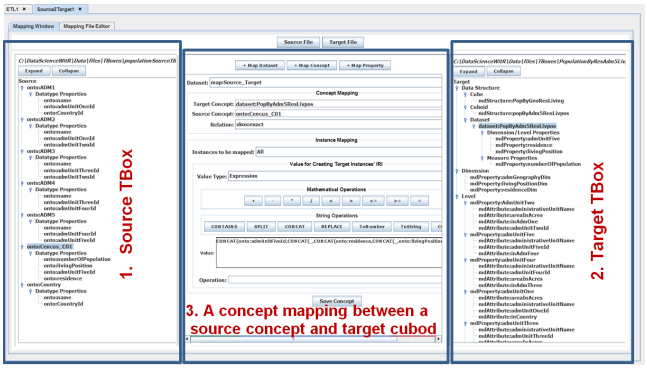

Figure 4: The interface to define mappings

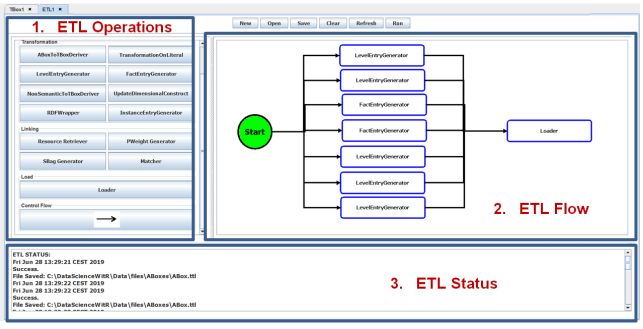

Figure 5: The interface to create ETL flows

The *Definition Layer*: First, the user generates a TBox using the GUI shown in Figure 3 which allows to create either a new TBox or edit an existing one. (S)he creates any TBox constructs

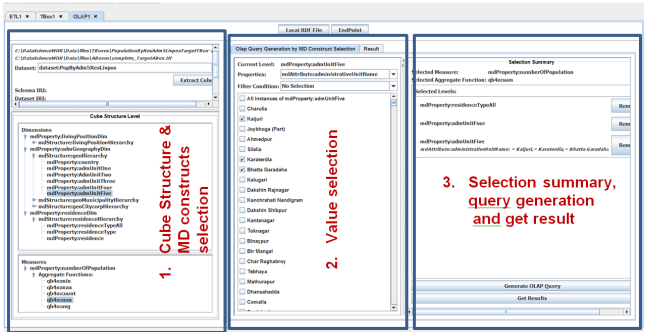

Figure 6: The interface to create and run OLAP queries

using panel 1 and a cube using panel 2. The rightmost panel is the edit panel where any property with its corresponding values can be added or deleted. The middle panel shows the generated RDF model for the TBox in Turtle format.

The next task is to map the source and target TBoxes. First, the user creates a source TBox from the CSV files using `setl:NonSemanticToTBoxDeriver` (*ETL Layer*). Figure 4 shows how to map between source and target TBoxes. The left panel shows the tree of source TBox constructs while the right panel shows that of the target. The middle panel shows how to create a mapping between a source concept and a target cube. In short, this concept mapping tells under which map dataset it is, whether source instances are fully or partially mapped with the target, and how to generate target observation IRIs. Then, (s)he map properties of the source and target concepts by using *Property Mapping* window.

The *ETL Layer*: Figure 5 shows how to create an ETL flow for populating the target ABox. *LevelEntryGenerator* and *FactEntryGenerator* are used to create Level members with their corresponding property values and observations according to the target semantics. The leftmost panel encapsulates the ETL operations. The user can drag and drop the operations in the ETL flow panel. The lowermost panel shows the ETL status.

The *OLAP Layer*: Then the user uses Figure 6 to load the *semantic* DW from a SPARQL endpoint or a local RDF file. The leftmost panel shows the cube structure of the corresponding dataset. (S)he can roll-up, drill-down by clicking the desired dimension hierarchy levels. The middle panel allows slicing and dicing according to the property values. The right panel shows the summary of the selections. Then the user generate the equivalent OLAP query and finally click *Get Result* to show the result panel.

ACKNOWLEDGEMENTS

This research is partially funded by the European Commission through the Erasmus Mundus Joint Doctorate Information Technologies for Business Intelligence (EM IT4BI-DC), the Poul Due Jensen Foundation, and the Danish Council for Independent Research (DFF) under grant agreement no. DFF-4093-00301B.

REFERENCES

[1] Alberto Abelló, Oscar Romero, Torben Bach Pedersen, Rafael Berlanga, Victoria Nebot, Maria Jose Aramburu, and Alkis Simitsis. 2014. Using Semantic Web Technologies for Exploratory OLAP: A Survey. *IEEE TKDE* 27, 2 (2014), 571–588.

[2] Cristina Ciferri, Ricardo Ciferri, Leticia Gómez, Markus Schneider, Alejandro Vaisman, and Esteban Zimányi. 2013. Cube algebra: A Generic User-Centric Model and Query Language for OLAP Cubes. *IJDWM* 9, 2 (2013), 39–65.

[3] Dario Colazzo, François Goasdoué, Ioana Manolescu, and Alexandra Roatiş. 2014. RDF Analytics: Lenses over Semantic Graphs. In *WWW*. ACM, 467–478.

[4] Richard Cyganiak, Dave Reynolds, and Jeni Tennison. 2014. The RDF Data Cube Vocabulary. *W3C Recommendation, W3C (Jan. 2014)* (2014).

[5] Rudra Pratap Deb Nath, Katja Hose, and Torben Bach Pedersen. 2015. Towards a Programmable Semantic Extract-Transform-Load Framework for Semantic Data Warehouses. In *Proceedings of the ACM Eighteenth International Workshop on Data Warehousing and OLAP*. 15–24.

[6] Lorena Etcheverry and Alejandro A Vaisman. 2012. QB4OLAP: A New Vocabulary for OLAP Cubes on The Semantic Web. *COLD* (2012).

[7] Median Hilal, Christoph G Schuetz, and Michael Schrefl. 2017. An OLAP Endpoint for RDF Data Analysis Using Analysis Graphs.. In *ISWC*.

[8] Dilshod Ibragimov, Katja Hose, Torben Bach Pedersen, and Esteban Zimányi. 2014. Towards Exploratory OLAP over Linked Open Data–A Case Study. In *Enabling Real-Time Business Intelligence*. Springer, 114–132.

[9] Benedikt Kämpgen, Seán O'Riain, and Andreas Harth. 2012. Interacting with Statistical Linked Data via OLAP Operations. In *ESWC*. Springer, 87–101.

[10] Ralph Kimball. 1996. *The data warehouse toolkit: practical techniques for building dimensional data warehouses*. John Wiley & Sons, Inc.

[11] Rudra Pratap Deb Nath, Katja Hose, Torben Bach Pedersen, and Oscar Romero. 2017. SETL: A Programmable Semantic Extract-Transform-Load Framework for Semantic Data Warehouses. *Information Systems* 68 (2017), 17–43.

[12] Victoria Nebot and Rafael Berlanga. 2012. Building Data Warehouses with Semantic Web Data. *Decision Support Systems* 52, 4 (2012), 853–868.

[13] Dimitrios Skoutas and Alkis Simitsis. 2007. Ontology-Based Conceptual Design of ETL Processes for Both Structured and Semi-Structured Data. *IJSWIS* 3, 4 (2007), 1–24.

[14] Jovan Varga, Lorena Etcheverry, Alejandro A Vaisman, Oscar Romero, Torben Bach Pedersen, and Christian Thomsen. 2016. QB2OLAP: Enabling OLAP on Statistical Linked Open Data. In *ICDE*. IEEE, 1346–1349.

[15] Jovan Varga, Alejandro A Vaisman, Oscar Romero, Lorena Etcheverry, Torben Bach Pedersen, and Christian Thomsen. 2016. Dimensional Enrichment of Statistical Linked Open Data. *Web Semantics: Science, Services and Agents on the World Wide Web* 40 (2016), 22–51.