Transforming Statistical Linked Data for Use in OLAP Systems

Benedikt Kämpgen
Institute AIFB
Karlsruhe Institute of Technology
76128 Karlsruhe, Germany
benedikt.kaempgen@kit.edu

Andreas Harth
Institute AIFB
Karlsruhe Institute of Technology
76128 Karlsruhe, Germany
harth@kit.edu

ABSTRACT

The amount of available Linked Data on the Web is increasing, and data providers start to publish statistical datasets that comprise numerical data. Such statistical datasets differ significantly from the currently predominant network-style data published on the Web. We explore the possibility of integrating statistical data from multiple Linked Data sources. We provide a mapping from statistical Linked Data into the Multidimensional Model used in data warehouses. We use an extract-transform-load (ETL) pipeline to convert statistical Linked Data into a format suitable for loading into an open-source OLAP system, and thus demonstrate how standard OLAP infrastructure can be used for elaborate querying and visualisation of integrated statistical Linked Data. We discuss lessons learned from three experiments and identify areas which require future work to ultimately arrive at a well-interlinked set of statistical data from multiple sources which is processable with standard OLAP systems.

Categories and Subject Descriptors

D.2.12 [**Software**]: Interoperability—*Data mapping*; H.2.8 [**Information Systems Applications**]: Database Management—*Database applications*

General Terms

Management, Performance, Experimentation

Keywords

Linked Data, OLAP, ontologies, statistics, integration

1. INTRODUCTION

Businesses are constantly struggling with the amount of information they need to process and leverage to their competitive advantage. Large amounts of useful data can be attained through the Web. Such data also comprises metadata and raw data from business operations that provide statistics. A commonly-used method for accessing and analysing statistical data is to build a data warehouse and to create reports and to allow exploration of the data via Online Analytical Processing (OLAP) systems[11]. Basic OLAP provides operations to integrate data in one view (selection), calculate meaningful metrics (projection), explore the data in various granularities and aggregations (drill-down/roll-up), and filter for certain information (slice/dice). However, providing an integrated view on disparate statistical data from multiple sources – a requirement for elaborate query and analysis scenarios – is a labour-intense task.

Semantic technologies promise concepts and technologies for making data and systems interoperable. Some datasets containing statistics are already published using Linked Data principles,[1] which provide a common access method and data model. Examples include the UK Open Government Data and the CIA World Factbook.[2] Our aim in this work is to use such statistical Linked Data from the Web in OLAP systems. We aim at collecting statistical Linked Data and transforming and integrating the data into a suitable format for storage in a data warehouse and analysis using common OLAP operations. The process has to work in an automated and scalable manner to cater for the amount of data that potentially can be accessed online. We need to tackle several challenges to enable elaborate OLAP operations over Linked Data:

- Single information pieces about datasets may be distributed over servers and files and published by different parties. Permanent availability is not guaranteed. Statistical Linked Data may be updated or refined continuously.
- Several heterogeneous schemas are in use. Still, there is no common agreement on how to make important aspects of statistical data self-descriptive, e.g., mathematical aggregation functions[12]. Similarly, data quality may be varying.
- With these challenges, direct querying and analysis of statistical Linked Data using SPARQL is not useful; OLAP typically is based on a specific conceptual model which is implemented in a data warehouse in order to store, integrate and analyse the statistical data. However, automatically building and evolving data warehouses has long been a topic of research[9].

[1] `http://linkeddata.org/`
[2] `http://www4.wiwiss.fu-berlin.de/factbook/`

I-SEMANTICS 2011, 7th Int. Conf. on Semantic Systems, Sept. 7-9, 2011, Graz, Austria

The conceptual model typically used for OLAP is a Multidimensional Model (MDM). Although there is no standard MDM[7, 3], all models have in common that they treat data as n-dimensional *Hypercubes* or *Cubes*. Data is divided into *Facts* and *Dimensions*. Facts are single data points in the Cube. Dimensions indicate the axes of a Cube. Dimensions have *Members*, which are possible values of the Dimension. Members are grouped along *Hierarchies* of one or more *Levels*. Sets of Facts with common values of their Dimensions return aggregated *Measures*. Several Cubes can use the same Dimension and can be put together in *Multicubes*. Figure 1 shows a common MDM.

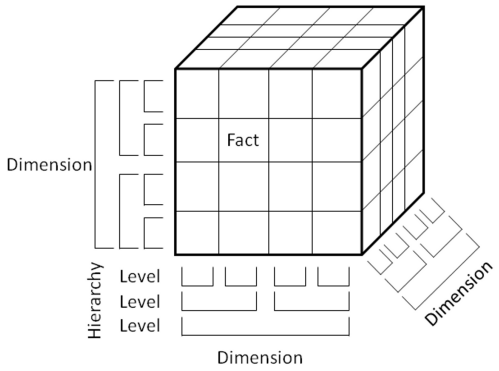

Figure 1: Common Multidimensional Model

Some formats for statistical data can directly be mapped to an MDM. For instance, in Excel, CSV, or relational database tables, a table corresponds to a Cube, rows to Facts, and columns – possibly keys for joins with other tables – to Dimensions. XBRL,[3] e.g., used by the U.S. Security and Exchange Commission, consists of items that are Facts, already described by Dimensions. SDMX[4] consists of observations which also amount to Facts described with Dimensions. However, disparate statistical data from multiple sources over the web are difficult to integrate using these formats.

For Linked Data, there are several vocabularies available that can be used to publish raw or aggregated data (e.g.,[2, 12]). we have selected the RDF Data Cube vocabulary (QB)[5] which mimics SDMX[1]. As a further development of the SCOVO vocabulary, QB intends to be easier to handle and better able to capture the semantics of the statistical Linked Data. There are already several statistical data published in QB format, e.g., issues from the U.S. Security and Exchange Commission,[6] and spendings by the UK district council Lichfield.[7] Also, datasets using SCOVO vocabulary can be re-expressed in the QB vocabulary. However, the RDF Data Cube vocabulary intends to be generally suitable to model all kinds of statistical datasets. Thus, the mapping of datasets using QB to an MDM is not as straightforward as the name suggests. For instance, QB allows to add attributes to statistical units for which there is no direct correspondence in a common MDM.

We use the following scenarios throughout the paper to illustrate our approach:

Unemployment fear and GDP growth Consider a dataset with survey data about German employees' fear of unemployment in the last few years, already published as Linked Data; and the European Commission's publication of the Gross Domestic Product growth of all European countries per year as provided by Eurostat. We want to integrate and make comparable both metrics over time for Germany to get insights about the relation between GDP and employees' perceived situation.

Number of death by illness and of hospitals Consider a dataset from the World Health Organisation comprising the number of people dying from a certain illness in certain countries, in conjunction with the number of hospitals as provided by Eurostat. For all European countries we want to compare the difference between people dying from a cause treated at hospitals and the number of hospitals.

Comparing EU 2020 - indicators Consider datasets containing several Eurostat metrics, such as the employment rate, the gross domestic expenditure on R&D, the energy intensity of the economy, and greenhouse gas emissions. We want to aggregate by average for all countries and to show the aggregated numbers per year, so that we can spot trends of important indicators for European countries.

Our contributions are as follows:

- We present a mapping between statistical Linked Data expressed in the QB vocabulary and a common MDM.
- We have developed an ETL pipeline that uses the mapping to fulfil our requirements for transforming statistical Linked Data into a Multidimensional Model in an automatic and scalable manner, and for allowing common OLAP operations over the transformed data.
- We have used the pipeline as part of a system which demonstrates the mentioned scenarios, to retrieve lessons learned and to evaluate whether our envisioned method fulfils our requirements.

We present the mapping between Linked Data and the MDM in Section 2. In Section 3, we present our system and apply it to the scenarios mentioned. In Section 4 we discuss our results and present lessons learned which point to possible future work. In Section 5, we describe related work, after which, in Section 6 we conclude.

2. MAPPING STATISTICAL LINKED DATA TO A MULTIDIMENSIONAL MODEL

In the following, we map Linked Data as described with QB to a common MDM. For that, we will describe our MDM with more detail and identify correspondences in QB. We use URIs in abbreviated form with common prefixes as listed by *prefix.cc*.[8]

Our MDM and the RDF Data Cube vocabulary provide constructs to model statistical data. Figure 2 shows a UML class diagram of an MDM which comprises common elements found in literature[7, 3], and which we use in this work. In this section we want to map identical elements in

[3] http://www.xbrl.org/Home/
[4] http://sdmx.org/
[5] http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html
[6] http://edgarwrap.ontologycentral.com/
[7] http://spending.lichfielddc.gov.uk/download
[8] http://prefix.cc/.

both representations as a basis for transformation from one representation to the other.

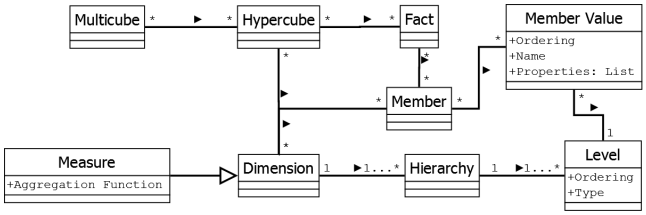

Figure 2: Class diagram of common Multidimensional Model

Table 1 gives an overview of the mapping, which we now explain in more detail. The central element of a common MDM is the Data Hypercube or often referred to as Cube. An MDM consists of one or more Cubes which are thematically related. In QB there is the concept of *qb:DataSet*. A number of instances of *qb:DataSet* chosen for integration and analysis predefine an MDM. A Cube in an MDM is uniquely identified by an instance of *qb:DataSet* related by the property *qb:structure* to an instance of *qb:DataStructureDefinition*. The property *rdfs:label* of the dataset and the data structure definition can be concatenated to form a unique name of a Cube. The property *rdfs:comment* can be used for a description of a Cube. Both properties can generally be used to name and describe multidimensional elements.

A Cube contains Facts representing the actual statistical data. In QB a Fact is an instance of *qb:Observation*. Such instances are connected to a *qb:DataSet* via the property *qb:dataset*.

A Cube uses certain Dimensions to describe its Facts. In this sense a Cube is a multidimensional coordination system and Facts are its data points. For instance, a Hypercube with three dimensions can be seen as a 3-dimensional coordination system with three axes. In QB Dimensions are predefined by the data structure definition of the dataset. In QB, Dimensions are represented as instances of *qb:ComponentProperty*. Note, in QB there is *qb:DimensionProperty* which however does not fully correspond to Dimensions; for an MDM *qb:AttributeProperty* and *qb:MeasureProperty* need to be considered as Dimensions. For each predefined Dimension a Fact has a Member. Members form the possible values of a Dimension. In QB, these Dimension Members can be given explicitly via *qb:codeList* by instances of *skos:ConceptScheme*, or implicitly by the Members used by actual Facts in the data. Additionally, the *rdfs:range* of a *qb:ComponentProperty* can state the type of the Members. Members can be resources or literal values. As an example, the data structure definition of "Real GDP growth rate" contains as *qb:DimensionProperty dc:date* with a literal of type *xsd:date* such as "2008", and *eurostat:geo* has a code list with resources as Members that represent countries.

Members of a Dimension are grouped along one or more Hierarchies of one or more Levels of granularity. In QB, Hierarchies of Levels depend on the actual Members of the Dimension. For instance, if we have *xsd:date* as range, we can have the natural hierarchy of year, month, day. Or, Members of type *foaf:Person* might be put into job roles such as academics, professors, and students so that we can have a hierarchy of job field and job role. On each Level of a Hierarchy, Members have a Member Value. A Level defines the type of the values, e.g., Boolean, Decimal, Integer, and String, Date, Time, Timestamp. For instance, year would be Integer and job role would be String. Member Values can have an ordering. If no Hierarchies and no Levels can be derived, this corresponds to one Hierarchy with one Level that is of type String and contains as values either the literal values or the labels of the Dimension Members. For a given Hierarchy and Level, also additional Properties can be given for a Dimension Member. For instance, Level properties can be used for giving the telephone number of a company or a measure unit of a numerical dimension. Such properties are not stored in a separate Dimension as they fully depend on the Dimension, Hierarchy, Level, and Member. In QB, such information can be taken from the resources representing the Member values of Dimensions. Also, a subproperty of *qb:ComponentProperty*, *qb:AttributeProperty*, can be used. Note, Facts that have different attribute component values may not be able to be directly compared, because aggregations of Facts that have different underlying semantics may be wrong. Facts with different attribute component values may nevertheless be put into the same Cube if Measures are translated into a common format, or clearly indicated by taking the attribute component as a special type of Dimension. Otherwise, Facts may be put into different Cubes.

The actual statistics are given by Measures – certain Dimensions of Facts that contain metrics. A Cube can define one or more Measures. Each Measure has an aggregation function, e.g., sum, min, max, avg, count, and distinct-count, that defines how the Measure is calculated from a set of Facts. In QB, for Measures, another subproperty of *qb:ComponentProperty*, *qb:MeasureProperty*, is available. QB does not describe how to model aggregation functions. They need to be derived automatically. If aggregation information is not given for a *qb:MeasureProperty* we create one Dimension and a Measure for each aggregation function possibly correct, e.g. sum, avg, min, max, count, count, and distinct-count for numerical measures and count, distinct-count for nominal measures such as string and date. In the case of the Eurostat dataset "Total length of motorways" with a Measure for the length of motorways for a certain country at a certain point in time, we automatically create a Measure, that gives the average length of motorways for a set of Facts.

If Cubes share Dimensions, they are put together into a Multicube. In QB, a Multicube corresponds to Cubes that use instances of *qb:ComponentProperty* which are found equivalent (e.g., linked by *owl:sameAs*). Similarly, Members might be equivalent. For instance, dataset "Real GDP growth rate"[9] and the GESIS dataset "ZA4570 ALLBUS/GGSS 1980-2008" (corresponds to "Fear of unemployment") [10] have a Dimension denoting a geopolitical entity, and both have a Member denoting Germany. Also, they both use the same time Dimension with literal values denoting the same time points. If there are *owl:sameAs* statements between the geo Dimensions and the geo Dimension values the two Cubes can be represented in one Multicube. See Figure 3 for an illustration. Note, in order to compare metrics from different cubes – different from Dimensions and Members – Measures always need to denote different metrics,

[9] `http://estatwrap.ontologycentral.com/id/tsieb020#ds`

[10] `http://lod.gesis.org/lodpilot/ALLBUS/ZA4570v590.rdf#ds`

MDM	RDF (QB)
Data Hypercube (Cube)	?ds, ?dsd: ?ds a qb:DataSet. ?ds qb:structure ?dsd
Fact	?fact: ?fact a qb:Observation. ?fact qb:dataSet ?dataset
Dimension	?dimension: ?dsd qb:componentSpecification ?cs. ?cs qb:componentProperty ?dimension
Member	?dimMem: ?componentProperty qb:codeList ?codeList. ?componentProperty a qb:ComponentProperty. ?codeList skos:hasTopConcept ?dimMem. UNION ?dimMem a ?range. ?componentProperty rdfs:range ?range. ?componentProperty a qb:ComponentProperty
Hierarchy	depends on Members of Dimension
Level	depends on Hierarchy and Members
Level Type	depends on Hierarchy and Level
Member Value	depend on Hierarchy and Level
Measure	?measure: ?dsd qb:componentSpecification ?cs. ?cs qb:measure ?measure
Aggregation Function	depends on Measure
Multicube	Cubes sharing Dimensions and Members

Table 1: Mapping terms of common MDM to SPARQL queries on RDF using QB

even though they may be described by the same property or linked by *owl:sameAs*.

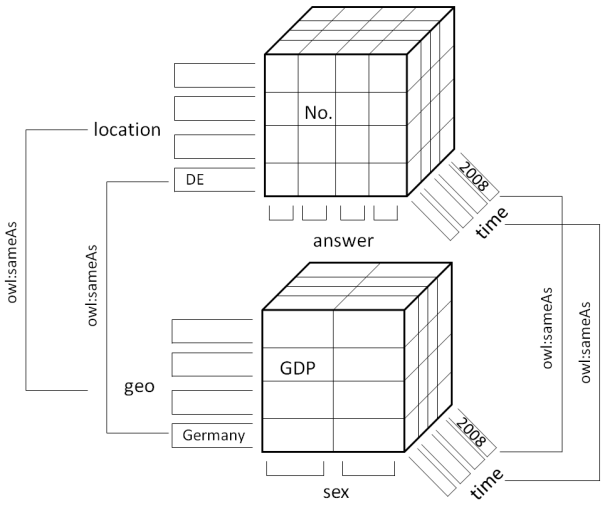

Figure 3: Example of a Multicube

In Linked Data, the MDM's data may be stored in a distributed manner; useful information about a Cube can be found by resolving URIs of interesting entities related to the Cube. For example, we need information about instances of *qb:DataSet*, instances of *qb:DataStructureDefinition*, and instances of *skos:TopConcept*. A *qb:DataSet* may give further information about its actual data, e.g., use the voiD vocabulary to indicate that the actual data can be retrieved from a certain SPARQL endpoint.

3. EVALUATION

In this section we evaluate whether our mapping helps to fulfil our requirements. We describe an implementation of the mapping that automatically transforms statistical Linked Data conforming to QB to enable OLAP operations. We apply this system to the scenarios introduced earlier. We will first describe our system and then the application to the scenarios.

3.1 System architecture

For evaluation, we have implemented our mapping in a system which is a web application written in PHP 5.3.0. Figure 4 shows the architecture of our system. Our system consists of two parts: an extract-transform-load (ETL) pipeline that creates a data warehouse based on datasets given by their URIs; and a runtime part, where any number of OLAP queries can be issued to the data after an ETL process has been finished. Running an experiment with the system includes the following steps, also indicated in Figure 4.

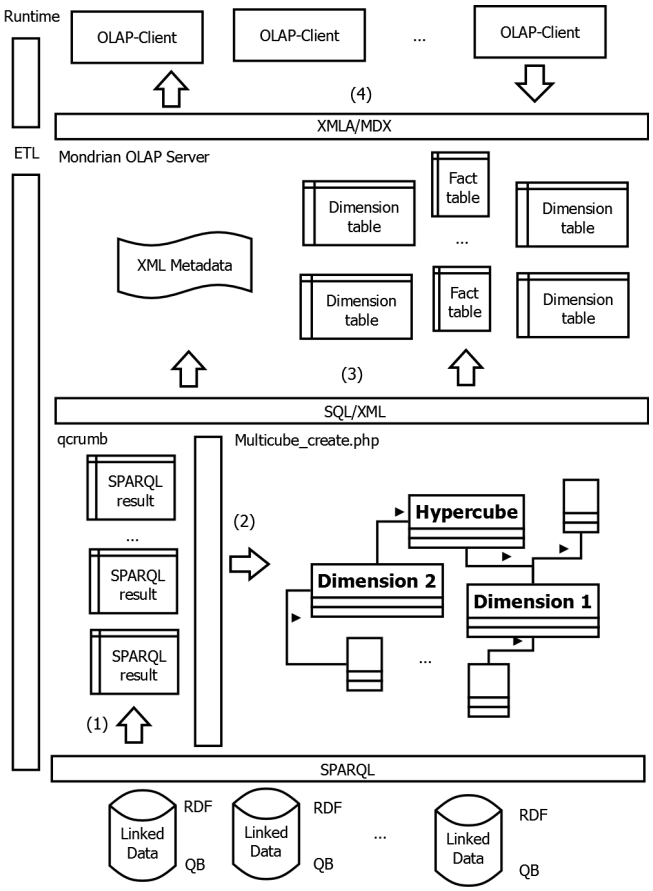

Figure 4: Architecture

(1) The user defines the datasets to be integrated. To retrieve the information for creating an MDM, the system issues SPARQL queries on metadata and the actual statistical data described by the datasets. Building on the Linked Data principles the system needs to retrieve the relevant RDF files from the Web, store the data in a triple store, and then issue the queries to the store. Our system uses qcrumb[11] which allows to specify the location of the files in the from clause to issue SPARQL queries to the entire RDF graph as defined by the files' content.

(2) Based on our mapping, the system creates an MDM and populates the MDM with the retrieved data.

(3) This MDM then is serialised for usage by XML for Analysis (XMLA).[12] We have chosen XMLA for several reasons. It provides a web-based interface to create and query an MDM. XMLA realises an MDM that corresponds to our common MDM, thus, the serialisation is not complex. XMLA is said to be most widely adopted in industry[8]. There are OLAP servers that provide XMLA interfaces, e.g., Palo OLAP Server and Mondrian.[13] There are client programs that connect to XMLA interfaces and allow user-friendly OLAP, e.g., JPivot and Palo Client. They allow OLAP operations and visualise the results as tables or graphs. Also, there are programming libraries, e.g., OLAP4J and xmla4js to extend existing applications. As an implementation of XMLA, we use Pentaho Mondrian OLAP Server, which uses XML for serialising the metadata about an MDM and SQL (we use MySQL) for populating it with data about Dimensions and Facts. Mondrian uses the well-known star schema to populate the MDM as relational data. There, we have a Fact table for Facts that for each Measure and Dimension contains a column and for each Dimension joins to a Dimension table with the Members and Values.

(4) XMLA implements the Multidimensional Query Language[14] (MDX) to issue common OLAP operations. Any number of MDX queries can be issued to the data in the warehouse. For that, we use a small JavaScript page based on xmla4js, that allows to connect to XMLA, to issue an MDX query and to display the results as a table. Listing 1 shows a basic MDX query.[15] Here, a Multicube of two datasets is queried to retrieve the percentage of negative answers by survey participants and the average Real GDP growth rate given for Germany at every available point in time.

Listing 1: MDX to query for employment fear metric and GDP growth

```
SELECT
{[Measures].[Percentage of Nos], [Measures
    ].[avg Real GDP growth rate]} ON
    COLUMNS,
{[Date].Children} ON ROWS
FROM [ALLBUSGGSS GDP growth rate]
WHERE {[Federal State].[Germany]}
```

[11] http://qcrumb.com/
[12] http://xmla.org/
[13] http://mondrian.pentaho.com/
[14] http://msdn.microsoft.com/en-us/library/Aa216767
[15] We have simplified the names. Our system gives longer names to make sure that all elements are uniquely identified.

3.2 Setup and experiments

Here we first describe our setup and then the experiments. We run our experiments on a Microsoft Windows 7 workstation with Intel(R) Core(TM) i5 CPU, M520, 2.40GHz, 4 GB RAM, 64-bit. There, our system is run together with Apache web server, Apache Tomcat, and Mondrian OLAP Server. Figure 5 shows how our system is controlled from a JavaScript-based web page.

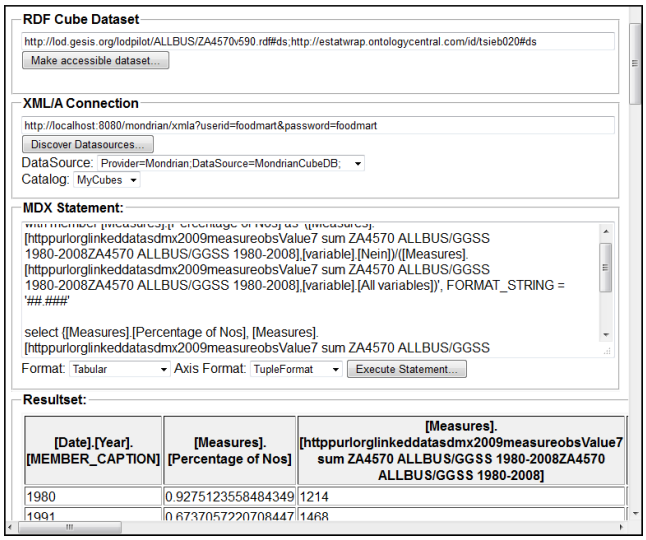

Figure 5: System interface

In correspondence to our architecture, we measure the time for each step in an experiment. The scenarios are 1) Unemployment fear and GDP growth 2) Number of death by illness and of hospitals, and 3a) and 3b) Comparing EU 2020 - indicators. Table 2 summarises the results from our experiments which we will then explain. It is not our intention to focus on the analyses results but rather on the process to prepare the statistical Linked Data.

3.2.1 *Unemployment fear and GDP growth*

The German federal state North Rhine-Westphalia publishes results from the Cumulated German General Social Survey which is also translated into Linked Data[16] using QB. Among others, we can retrieve a dataset `http://lod.gesis.org/lodpilot/ALLBUS/ZA4570v590.rdf#ds` with survey results. In the survey, people were asked about their fear of becoming unemployed. The description of these datasets are distributed over several files, e.g. for the actual data, the data structure definition, or the Member Values. The actual data of the survey dataset consists of 30 instances of *qb:Observation*. In total we are querying 1547 triples.

The European Commission publishes many different datasets about European countries. This Eurostat data also is available as Linked Data,[17] e.g., offering the dataset Real GDP growth rate. QB is not correctly used: Observations in the dataset do not fulfil the data structure definition they follow. We have made our system robust to this error so that such dimensions are ignored. The GDP growth dataset

[16] http://multiweb.gesis.org/gesis-lod-pilot/
[17] http://estatwrap.ontologycentral.com/table_of_contents.html

Experiment	Datasets	Triples	SPARQL (sec)	MDM (sec)	XML/SQL (sec)	MDX (sec)
1)	2	20268	234	27	12	0.073
2)	n/a	n/a	n/a	n/a	n/a	n/a
3a)	4	24636	580	36	38	0.161
3b)	8	35482	1417	116	105	0.473

Table 2: Performance evaluation with metrics corresponding to system architecture

contains 320 observations. Here, we are querying 18.721 triples.

The survey data and the Eurostat data are partly linked. The survey data links its values of the geo dimension[18] to values of the geo dimension of Eurostat. Which is missing, however, is a link between both dimensions. Both dimensions describe the same dimension – a location where metrics have been taken from. For this experiment, we have created an RDF file containing the missing link and added it to the queried resources.[19] For the time Dimension, both datasets use the same property so that the links are automatically given.

Both datasets we run through our system. In total, the program ran for 273 seconds. Split up, the SPARQL queries on the datasets took 234sec; it took 27sec to create the MDM; and it took 12sec to serialise the data model for XMLA. Our system creates Dimensions for Federal State, Variable, Date, and Observation value and a Cube for each dataset. One Cube contains Measures aggregating the survey answers, e.g., by sum, the other Cube contains aggregated Measures about the GDP growth. Both Cubes share the Federal State, the Data, and the Observation value dimension. A Multicube is created consisting of both Cubes.

Afterwards, we run an MDX query that asks for the percentage of people saying that they had no fear of becoming unemployed. Calculating the percentage is an example of a more complex measure as mentioned in a previous chapter. The MDX we have already shown in Listing 1. Listing 2 specifies the percentage calculation. We need this complex measure because the fear of unemployment dataset defines a Dimension *gesis:variable* that indicates the type of answer given by survey participants. The possible answers are "no fear", "fear to need to switch job", and "fear to become unemployed". The percentage of "no fear" among all answers is calculated, which means to divide the sum of negative answers by the number of all answers given.

Listing 2: Complex measure to query for employment fear metric

```
WITH MEMBER [Measures].[Percentage of Nos]
    AS
'([Measures].[sum survey results],[variable
    ].[Nein])/
([Measures].[sum survey results],[variable
    ].[All variables])'
```

The MDX took 0.073sec to run. The result shows a table with 16 rows and two columns of aggregated measures. Each row indicates a year. The first column contains the percentage, the second column the GDP growth. Unfortunately, only for 2006 and 2008 both metrics are available, making the table very sparse. Yet, we have successfully integrated both datasets and made available for querying using OLAP.

[18] http://lod.gesis.org/lodpilot/ALLBUS/geo.rdf

[19] http://people.aifb.kit.edu/bka/Public/cube_additionalRDF.rdf

3.2.2 *Number of death by illness and of hospitals*

The World Health Organisation publishes in its Global Health Observatory Data Repository various datasets on important health topics. Among others there is a dataset reporting about mortality and burden of disease for different countries. There is also a representation in QB available.[20]

Integrating metrics from this dataset to metrics from Eurostat, e.g., the number of hospitals, promises useful information. However, the data has turned out to be not sufficiently self-descriptive to be automatically used by our system. For instance, different from the vocabulary's guidelines, observations are not linked to a *qb:DataSet* from where an application can find a *qb:DataStructureDefinition* as a description.

3.2.3 *Comparing EU 2020 - indicators*

We integrate four different datasets from Eurostat: Employment rate by gender, age group 20-64; Gross domestic expenditure on R&D; Greenhouse gas emissions, base year 1990; and Energy intensity of the economy. Altogether these datasets contain 1247 observations. Our ETL pipeline finished in 654sec on 24636 triples that are related to these datasets. Afterwards, we ran a query to retrieve the metrics for all created measures showing their numbers over time and aggregating by average for all countries. To get a better impression about scalability of our system, we have run a fourth experiment with the same datasets plus another four, e.g., the population at-risk-of-poverty or exclusion. Altogether, these datasets include 2682 observations, twice as many as in the experiment before. The SPARQL queries took 2.5 times as long as with four datasets. Creating and serialising the MDM took almost three times as long. Also, the same MDX query issued on these eight datasets took three times as long.

4. DISCUSSION AND LESSONS LEARNED

In this section we discuss the results of the experiments from the previous section and give lessons learned. Our aim is to evaluate whether our system fulfils the requirements of automatically transforming statistical Linked Data conforming to QB to an MDM to allow common OLAP operations.

4.1 Scalable transformation to MDM

After an MDM has been successfully serialised into a data warehouse, the actual OLAP operations only take an instant. The performance of our mapping is assessed best by the time it takes to create the MDM and to serialise it.

[20] http://aksw.org/Projects/Stats2RDF

The amount of data we integrate does not allow conclusions about scalability of the system. Our system terminated at every experiment. Still, the performance can be improved: implicit knowledge of subclass or equivalence relationships are hard-coded so far, e.g., from *owl:sameAs* between dimensions a closure table is computed that assigns to each dimension URI a canonical value. For each canonical value, then, a dimension in the multidimensional model is created. For very large datasets, reasoning techniques, e.g., directly built in the triple store, may simplify the implementation. So far we built the entire MDM as described by the Linked Data about the input datasets; instead, modelling could be more directed by user queries as sometimes is done in Web warehousing[9]. The bottleneck of our system are the SPARQL queries; per run of the ETL pipeline, there are $ds+6*dim+4$ SPARQL queries issued, with ds as the number of datasets and dim as the number of dimensions. For example, we do not consider distributed storage of the actual data and its metadata and always issue the queries to the entire set of triples related to the datasets. So far, our datasets have not had very large numbers of observations and dimensions, so that we cannot analyse their influence on the process. However, as most modelling happens with the data structure definition, at least for the actual observations we do not expect an overly negative impact on the performance.

4.2 Automatic transformation to MDM

We agree with Vrandečić et al.[12] that statistical Linked Data benefits from grounding to its semantics. The RDF Data Cube vocabulary seems to provide a suitable trade-off between convenience to publish and expressivity to make statistics self-descriptive. Yet, as seen with our experiments we have found several cases where the vocabulary is not used correctly, so that fully automatic transformation was not possible. Also, more complex features of the vocabulary, e.g., slices of datasets, predefined component properties with a well-defined semantic (Content Oriented Guidelines), and hierarchies of *skos:TopConcept* are not used.

4.3 Common OLAP operations available

We were not able to test all common OLAP operations in our experiments. For instance, the vocabulary recommends ways to model Hierarchies and Levels. However, these possibilities are not much used, yet. We believe that there are more possibilities to find meaningful Hierarchies in statistical Linked Data. Questions remain, e.g., of how to identify redundant information on various levels of detail, and how to handle Facts that show too much or too less granular detail than used by other Facts in the Cube.

Another open question is related to automatically finding aggregation functions and complex Measures that can be used upon statistical Linked Data. Mathematical functions can be explicitly stated with the statistical data, for which there are various ways[12]. We expect that aggregation functions can also be found automatically from the semantic description of the domain. However, the ability to summarise the values of multiple measures [4] needs to be considered, as not all aggregations make sense, e.g., to use as aggregation function the sum operator for a Measure giving the current stock of a product at a certain point in time.

4.4 Mapping of query languages

Our mapping is implemented as an ETL pipeline to store the RDF in a data warehouse. A possible direction for future work is to have OLAP operations directly on the RDF for which we would need a mapping of query languages used for OLAP and RDF. SQL is based on relational data and is often used as the underlying technology to store multidimensional data (ROLAP). SQL can also be used to store and query graph based data, as well as SPARQL can be used to query relational data. Thus, in principle SQL and SPARQL can be used to run queries on multidimensional data. We will not focus on the differences between the languages, here, but try to give an impression of how a basic query for multidimensional data corresponds to a query on Linked Data. Figure 3 gives an overview of the mapping between OLAP and SPARQL.

OLAP	**SPARQL**
Selection	Query for data with certain Dimensions.
Projection	Query for aggregated Measures.
Drill-down/Roll-up	Querying more/less fine grained values of Members.
Slice/Dice	Filtering on Facts with certain Members.

Table 3: Mapping of query languages

One can issue basic OLAP operations on Cubes described by our MDM (taken from [11]). OLAP allows to select Dimensions, Hierarchies, and Levels to query for data from a Cube. It allows to also select Measures to aggregate. For instance, we can query a Multicube of two datasets to retrieve the percentage of no answers by survey participants and the average Real GDP growth rate given for Germany at every available point in time.

Listing 3 shows a SPARQL query that tries to query for such information from the Eurostat Linked Data. Here, projection is done to select and aggregate the percentage and the growth rate. However, in Section 3, we have seen that the percentage of negative answers forms a complex measure calculated from several aggregated metrics, which would not be as easy as indicated in this example, and would require SPARQL subqueries.

Listing 3: SPARQL to query for employment fear metric and GDP growth

```
SELECT ?time ?geo avg(?nos) avg(?grorate)
WHERE {
?s qb:dataset <http://estatwrap.
    ontologycentral.com/id/tsieb020#ds> .
?s dcterms:date ?time .
?s eus:geo ?g .
?g rdfs:label ?geo FILTER(?geo = "Germany")
?s eus:nos ?nos .
?s eus:growthrate ?grorate
}
ORDER BY ?geo
```

Drill-down or roll-up lead to more granular or less granular results. E.g., if we would drill-down the geopolitical entity from country to city, each country would be split up in its cities resulting in more fine grained information regarding the projected metrics. In SPARQL, one would instead of directly query for the label of the Member, use a query pattern that groups the Members and then query for the

label of the grouping resource. Slice (and dice) fix dimensions on one member (several members) and denote subsets of the data. In our example, we use this functionality to filter for information about Germany. Slices correspond to filter patterns in SPARQL.

5. RELATED WORK

There is much work on creating Multidimensional Models from web data using XML[8]. Google Public Data Explorer allows expressive analyses. These approaches do not use Semantic Web concepts; they have difficulties to ground statistical data to its domain, to find the most meaningful conceptual model, and to easily integrate datasets.

There is recent work on creating Multidimensional Models from ontologies[10, 4, 6]; Niinimäki and Niemi[5] describe an ETL approach to first transform data into an ontology for a Multidimensional Model and then serialise the MDM for use with the Mondrian OLAP server. They put much focus on their ontology, which directly models a Multidimensional Model. Our approach is based on a vocabulary that already has been adopted by different parties and we focus on statistical Linked Data that is grounded to the domain of the statistics to automatically map the statistical data to a meaningful Multidimensional Model.

Other related approaches retrieve statistical information from the Web, automatically integrate the data and let the user analyse it. Google Squared, Google Refine, and Needlebase use keyword searches and structured background information to structure data from the web in tables. They rely more on concepts and techniques from Information Retrieval, Machine Learning, NLP and Pattern Matching, and less on ontologies and Linked Data. Also, they do not allow OLAP operations such as drill-down/roll-up, aggregations, and complex measures. With respect to functionality and ease of use Gartner ranks Tableau Software highest among Business Intelligence platforms.[21] Tableau does not provide much ways to analyse web data directly. Semantic Web browsers provide an opportunity to directly analyse statistical Linked Data. Examples include Exhibit Faceted Search, OpenLink PivotViewer, and Freebase Parallax that, however, do not support views on numerical data such as aggregations and complex measures.

6. CONCLUSIONS

Analysing statistical Linked Data with OLAP promises useful decision-support. We have presented a mapping from statistical Linked Data that conforms to the RDF Data Cube vocabulary to a common Multidimensional Model. We have implemented the mapping in a system and applied it in three experiments which show that the requirements of an automatic and scalable transformation into a Multidimensional Model and a provision of common OLAP operations are partly met. We were able to automatically transform datasets and issue OLAP operations on the transformed dataset. A thorough performance evaluation of our ETL pipeline is still to be done. Some datasets were not self-descriptive enough to be automatically analysed. Also, current statistical Linked Data do not fully exploit the expressivity the RDF Data Cube vocabulary provides or functionalities a basic MDM supports. Therefore, not all OLAP operations could be tested. We will try to find more use cases to further evaluate our system with statistical Linked Data, to continuously improve the performance and functionality, and to eventually make a packaged version of the system available as open-source.

[21] `http://www.gartner.com/technology/media-products/reprints/tableau/vol2/article4/article4.html`

Acknowledgements

This work was supported by the German Ministry of Education and Research (BMBF) within the SMART project (Ref. 02WM0800) and the European Community's Seventh Framework Programme FP7/2007-2013 (PlanetData, Grant 257641).

7. REFERENCES

[1] R. Cyganiak, S. Field, A. Gregory, W. Halb, and J. Tennison. Semantic Statistics: Bringing Together SDMX and SCOVO. *Proceedings of the WWW2010 Workshop on Linked Data on the Web*, 2010.

[2] M. Hausenblas, W. Halb, Y. Raimond, L. Feigenbaum, and D. Ayers. SCOVO: Using Statistics on the Web of Data. In *ESWC*, volume 5554 of *Lecture Notes in Computer Science*, 2009.

[3] S. Lujanmora, J. Trujillo, and I.-Y. Song. A UML profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering*, 59(3), 2006.

[4] T. Niemi and M. Niinimäki. Ontologies and summarizability in OLAP. *Proceedings of the 2010 ACM Symposium on Applied Computing SAC 10*, 2010.

[5] M. Niinimäki and T. Niemi. An ETL Process for OLAP Using RDF/OWL Ontologies. *Journal on Data Semantics XIII*, 5530:97–119, 2009.

[6] J. Pardillo and J.-N. Mazón. Using Ontologies for the Design of Data Warehouses. *Journal of Database Management*, 3(2), 2011.

[7] T. Pedersen, T. B. Pedersen, and C. E. Dyreson. A foundation for capturing and querying complex multidimensional data. *Information Systems Journal*, 26(5), 2001.

[8] J. M. Perez Marti, R. Berlanga, M. J. Aramburu, and T. B. Pedersen. Integrating Data Warehouses with Web Data: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 20(7), 2008.

[9] S. Rizzi, A. Abelló, J. Lechtenbörger, and J. Trujillo. Research in data warehouse modeling and design: dead or alive? *Proceedings of the 9th ACM international workshop on Data warehousing and OLAP*, 2006.

[10] O. Romero and A. Abelló. Automating Multidimensional Design from Ontologies. *Proceedings of the ACM tenth international workshop on Data warehousing and OLAP DOLAP 07*, 2007.

[11] O. Romero and A. Abelló. On the Need of a Reference Algebra for OLAP. *Data Warehousing and Knowledge Discovery*, 2007.

[12] D. Vrandečić, C. Lange, M. Hausenblas, J. Bao, and L. Ding. Semantics of Governmental Statistics Data. *Proceedings of the WebSci10*, 2010.