Early analysis and debugging of linked open data cubes

Enrico Daga[1], Mathieu d'Aquin[1], Aldo Gangemi[2], and Enrico Motta[1]

[1] Knowledge Media Institute (KMI),
The Open University
Walton Hall, Milton Keynes, Buckinghamshire MK7 6AA, United Kingdom
{enrico.daga,mathieu.daquin,enrico.motta}@open.ac.uk
http://kmi.open.ac.uk
[2] Institute of Cognitive Science and Technology (ISTC),
Italian National Research Council (CNR)
Via S. Martino della Battaglia 44, 00185 Roma (RM), Italy
aldo.gangemi@cnr.it
http://istc.cnr.it

Abstract. The release of the Data Cube Vocabulary specification introduces a standardised method for publishing statistics following the linked data principles. However, a statistical dataset can be very complex, and so understanding how to get value out of it may be hard. Analysts need the ability to quickly grasp the content of the data to be able to make use of it appropriately. In addition, while remodelling the data, data cube publishers need support to detect bugs and issues in the structure or content of the dataset. There are several aspects of RDF, the Data Cube vocabulary and linked data that can help with these issues. One of the features of an RDF dataset is to be "self-descriptive". Here, we attempt to answer the question "How feasible is it to use this feature to give an overview of the data in a way that would facilitate debugging and exploration of statistical linked open data?" We present a tool that automatically builds interactive facets as diagrams out of a Data Cube representation without prior knowledge of the data content to be used for debugging and early analysis. We show how this tool can be used on a large, complex dataset and we discuss the potential of this approach.

1 Introduction

Statistical linked data can be very complex for both publishers and consumers. On the one hand data publishers need to check that the data they publish are consistent with the standard specifications and with regards to the source model and documentation. On the other hand consumers of open data need to effortlessly understand how to get value out of a given linked open data source. What is particularly difficult in the early stage of analysis is to obtain assessable representations without having a deep understanding of the domain. Data analysts need to quickly grasp the value that they can get out of the data with small effort.

The release of the Data Cube Vocabulary specification introduces a standardised method for publishing statistical linked data. In particular, by giving a standardized method for describing the structure of the data, the Data Cube Vocabulary can be exploited to automatically generate SPARQL queries to support the user on the early

analysis of the dataset. Following this assumption, we argue that it is possible to develop a web based tool to explore and debug linked data cubes. We developed Vital[3], a tool that automatically builds interfaces for statistical linked open data and that

1. enables an effortless exploration of the information (data sets, dimensions, measures), and
2. supports publishers in the detection of bugs in the remodelled data.

In this paper we report on our experience with the HESA Unistats data[4] that we published as Linked Open Data in the context of the LinkedUp project[5]. We curated the translation and publishing of the Unistats data and we analysed, validated, and debugged the resulting RDF data. Vital automatically builds compact facets as diagrams out of a Data Cube representation without prior knowledge of the data content. We show how an intuitive insight on relevant content can be achieved without significant effort thanks to RDF and the Data Cube vocabulary, and we discuss the potential of this approach as well as lessons learnt.

In the following Section we introduce the RDF Data Cube Vocabulary and other related work. Section 3 reports on the remodelling of the Unistats data, that is the scenario that we will refer to in this paper. In Section 4 we describe our methodology and the associated tool, showing how it can support debugging and early analysis of linked data cubes. Finally, we discuss our experience and derive some conclusions in Section 5.

2 Background

The W3C RDF Data Cube Vocabulary [1] specification provides a means to publish multi-dimensional data on the web following the linked data paradygm. While we refer to the official specification, it is useful to introduce its basic concepts because they will be used extensively in the rest of this paper. Its conceptual model inherits the cube model that underlies SDMX (Statistical Data and Metadata eXchange)[6], that is an ISO standard for sharing statistical data and metadata among different organizations. In particular, it is based on the core of the the SDMX 2.0 Information Model[7]. The core class of Data Cube is `qb:DataSet`, that embeds a collection of `qb:Observations`. Observations are the actual multidimensional data. Each observation has a number of *dimensions*, one or more *measures*, and optionally some *attributes*. Dimensions serve the purpose of identifying observations. In practice, the set of dimension values identify uniquely an observation in the data set. Measures are the observed values. For example, an observation can measure the amount of students from non-UK countries that in year 2013 studied at The Open University. Listing 1 shows how this could be modelled with Data Cube.

[3] `http://data.open.ac.uk/demo/vital`
[4] `https://www.hesa.ac.uk/index.php?option=com_content&view=article&id=2609&Itemid=1576`
[5] `http://linkedup-project.eu/`
[6] See also ISO 17369:2013 `http://www.iso.org/iso/catalogue_detail.htm?csnumber=52500`
[7] `http://sdmx.org/?page_id=16`

```
[] a qb:Observation ;
   ex:organization ou:the_open_university ;
   ex:inUK false ;
   ex:amount "13227"^^xsd:int ;
   ex:year "2013"^^xsd:gYear .

ex:organization a qb:DimensionProperty .
ex:inUK a qb:DimensionProperty .
ex:year a qb:DimensionProperty .
ex:amount a qb:MeasureProperty .
```

Listing 1: A small Data Cube example.

Data Cube also provides mechanisms to specify the structure of observations in a given data set. This capability is provided by the means of a `qb:DataStructureDefinition`, that composes a number of `qb:ComponentSpecification`, each dedicated to the specification of a dimension, measure or attribute. Component specifications define the elements of the observations and can be exploited for several analysis and maintenance tasks, such as data validation[8].
Another useful feature is to specify meaningful ways to group subsets of the information. This is achieved by the means of `qb:Slices`. A slice fixes some dimensions and refers to all observations with those dimension values.

The LOD2 project developed an integrated set of tools for publishing Linked Data Cubes [2]. In particular, validation of translated data for automatic repair has been proposed in [3]. The approach is based on the application of SPARQL queries to check the compliance of observations with data structure specifications. This method allows the analyst to debug the coherence between data structure specifications and observations. However, especially when the data source does not have a cube-style format, many issues emerge only during data consumption, because they are related to remodelling issues and not only errors in the syntactic translation of the data. Remodelling problems will affect both the observations and the structure definition. Additionally, the task of evaluating how appropriate the documentation of dimensions and measures is cannot be easily automated. Both aspects become particularly hard to evaluate in cases where the data source does not come in a cube style model (it is the case of our use case scenario, see Section 3). We argue that data exploration lets emerge complementary issues, and helps improving the quality of the published data.

In the next sections we will often refer to the concept of *early analysis*. For the methodological aspects of data analysis we refer to [4]. Our scenario is focused on the initial phase of data analysis, where the analysist is mainly focused on assessing the quality of the data source in relation to its objectives.

3 Remodelling Unistats as Linked Data

The UK Higher Education Statistical Agency (HESA) collects a variety of data every year from UK universities. One of the most important collection is the Key Information

[8] The Data Cube Specification includes a section reporting SPARQL queries that can validate the data using the description in data structure definitions.

Set[9], that includes yearly information about full and part-time undergraduate courses as well as a wide range of data items. These include: course details, accommodation information, fee data and employment outcomes among others. The resulting data are published by HESA as the Unistats database. This database is accessible using a web API and can be downloaded as a XML single file or a set of CSV files. Documentation is offered through the HESA web site. However, the process of reusing these data can be hard. Unistats' data are pretty hard to understand, and exploring documentation is necessary and time consuming. The analysis of the source is not trivial especially because the documentation is not part of the data files, but is published aside as HTML or PDF.

Focusing in particular on the Employment data set, as we will use it as example in the remaining part of this paper, it includes information about the employment outcomes of students, taken from the Destination of Leavers data[10]. This information is provided in a section of the XML within the Course description of a given Institution. Listing 2 shows a snippet extracted from the XML.

```
<INSTITUTION>
    [...]
    <KISCOURSE>
        [...]
     <EMPLOYMENT>
          <EMPPOP>25</EMPPOP>
          <EMPAGG>24</EMPAGG>
          <WORKSTUDY>95</WORKSTUDY>
          <STUDY>25</STUDY>
          <ASSUNEMP>5</ASSUNEMP>
          <BOTH>5</BOTH>
          <NOAVAIL>5</NOAVAIL>
          <WORK>60</WORK>
     </EMPLOYMENT>
```

Listing 2: An excerpt of the XML file showing information about the Employment outcomes of a course.

Like all XML elements in the file, it is described in the related section of the documentation[11] Each Course might include more Employment sections, each showing the statistics considering one of the possible JACS[12] area under which the Course is located. Each Employment section (the actual Observation in Data Cube terms) contain the following measures (description as been extracted from the PDF documentation):

- "Work": the proportion of students in work six months after their course ended;
- "Study": the proportion of students who are undertaking further study six months after their course ended;
- "Work And Study": the proportion of students in work and study six months after their course ended;

[9] See https://www.hesa.ac.uk/index.php?option=com_studrec&Itemid=232&mnl=14061

[10] https://www.hesa.ac.uk/stats-dlhe

[11] See http://www.hesa.ac.uk/includes/C13061_resources/Unistats_checkdoc_definitions.pdf?v=1.12.

[12] The Joint Academic Coding System (JACS) system is used in the United Kingdom by the Higher Education Statistics Agency (HESA) and the Universities and Colleges Admissions Service (UCAS) to classify academic subjects.
https://www.hesa.ac.uk/jacs3

- "Work, Study, Work and Study": the proportion of students who are recorded as BOTH, WORK or STUDY six months after their course ended;
- "Assumed Unemployed": the proportion of students assumed to be unemployed six months after their course ended;
- "Not Available": the proportion of students who are not available for work or study six months after their course ended.

The analyst needs to build familiarity with the syntactic format of the data but also with its semantics, that can only be captured by comparing the data artifact with the documentation in the human readable files.

There are a number of reasons for publishing Unistats data as Linked Open Data:

- documentation is embedded in the data (as one major benefit of RDF is to allow the data to be self-descriptive);
- linked data help to make sense of the information with small effort, for example JACS codes can be materialized as resources reusing the existing SKOS based representation published by the University of Southampton[13];
- linked data can be queried using SPARQL, allowing for further analysis;
- universities can reuse and link this data to other linked data from the institution and from third parties;
- queries allow task-oriented tailored views
- representing dimensions as Linked Data resources, it is possible to derive new dimensions exploiting indirect connections[14]

The process of bringing Unistats data to RDF is not a mere translation, it is a remodelling effort that makes explicit the semantics of the data, hidden in the syntax of the input source and in the way it is referenced by the documentation documents.

As part of the LinkedUp project we made the effort to publish Unistats as Linked Open Data[15] to support learning analytics [5]. The data acquisition and publishing process takes into account not only the XML included in the downloadable package, but also reference data like the JACS codes published by the University of Southampton. In addition, the process includes the generation of links to known entities, such as the URIs of UKPRN institutions published at `http://learning-providers.data.ac.uk`.

The resulting RDF includes descriptions of datasets following the RDF Data Cube Vocabulary, but also embed well-formed documentation: a) all resources have at least one `rdfs:label` with language information specified; b) all resources have a single `rdfs:comment`; c) all resources have a single `skos:prefLabel`; d) schema elements have `rdfs:isDefinedBy` links to the Unistats vocabulary specification[16]; e) resources include links to HESA documentation using `rdfs:seeAlso`.

[13] See `http://data.southampton.ac.uk/dataset/jacs.html`.

[14] We will see in our example how most of the data is presented using the Course as primary dimension and not including the Institution by default. Obviously the Institution can be make explicit the relation between courses and institutions as additional dimension.

[15] The tool developed to remodel Unistats as RDF is published on `https://github.com/the-open-university/kiskit`.

[16] `https://raw.githubusercontent.com/the-open-university/kiskit/master/src/uk/ac/open/data/kiskit/v002/vocab/vocabulary.ttl`

The Unistats Linked Data contains 7.144.510 triples in total with 327.707 `qb:Observations` in 11 `qb:DataSets`. Table 1 illustrate the basic information about each data set. It is worth mentioning that the table has been obtained with a simple SPARQL query thanks to the fact that Linked Data embeds its documentation.

Table 1: DataSets in the Unistats Linked Data

Name	Observations	Description
Course Stages	82642	"Contains data relating to the learning, teaching and assessment methods for each course stage"@en
Continuation	30115	"The continuation data are derived by linking two successive years' of HESA student data."@en
Entry Qualifications	30013	"Contains data relating to the entry qualifications of students"@en
National Student Survey Results	29381	"DataSet containing the results of the National Student Survey"@en
Employment	29300	"Contains data relating to the employment outcomes of students"@en
Tariffs	27732	"Contains data relating to the entry tariff points of students"@en
Common Jobs	26849	"This is a DataCube DataSet including Common job types obtained by students taking a course. Jobs are described by course, percentage and order"@en
Job Types	26308	"Contains elements relating to the types of profession entered by students"@en
Degree Classes	22548	"Contains data relating to the degree classifications obtained by students"@en
Salaries	22430	"Contains data relating to the salary information of students"@en
National Student Survey NHS Results	389	"DataSet containing the results of the National Student Survey (NHS)"@en

Listing 3 shows the specifications of the Unistats Employment `qb:DataSets` and `qb:DataStrucuterDefinition`.

```
dset:employment
  a qb:DataSet;
  rdfs:label "Employment"@en ;
  skos:prefLabel "Employment" ;
  rdfs:comment "Contains data relating to the employment outcomes of students"@en;
  qb:structure dset:employmentStructure ;
  rdfs:seeAlso <http://www.hesa.ac.uk/includes/C13061_resources/
     Unistats_checkdoc_definitions.pdf?v=1.12> ;
  rdfs:isDefinedBy kis:
  .

dset:employmentStructure
  a qb:DataStructureDefinition;
  rdfs:label "Employment (structure)"@en ;
  skos:prefLabel "Employment (structure)" ;
  rdfs:comment "Information relating to the employment outcomes of students"@en ;
  # The dimensions
    qb:component [ a qb:ComponentSpecification ; qb:dimension  kis:course ; skos:
     prefLabel "Specification dimension: course" ];
    qb:component [ a qb:ComponentSpecification ; qb:dimension  dcterm:subject ;
     skos:prefLabel "Specification dimension: subject" ];
  qb:component [ a qb:ComponentSpecification ; qb:dimension  kis:institution ;
     skos:prefLabel "Specification dimension: institution" ];
  # The attributes
    qb:component [ a qb:ComponentSpecification ; qb:attribute  kis:population ;
     skos:prefLabel "Specification attribute: population"  ; qb:componentRequired
     true  ];
    qb:component [ a qb:ComponentSpecification ; qb:attribute  kis:aggregation ;
     skos:prefLabel "Specification attribute: aggregation" ; qb:componentRequired
     true ];
```

```
  # The measures
  qb:component [ a qb:ComponentSpecification ; qb:measure  kis:workAndStudy ;
   skos:prefLabel "Specification measure: workAndStudy" ];
  qb:component [ a qb:ComponentSpecification ; qb:measure  kis:assumedUnemployed
    ; skos:prefLabel "Specification measure: assumedUnemployed" ];
  qb:component [ a qb:ComponentSpecification ; qb:measure  kis:workOrStudy ;
   skos:prefLabel "Specification measure: workOrStudy" ];
  qb:component [ a qb:ComponentSpecification ; qb:measure  kis:notAvailable ;
   skos:prefLabel "Specification measure: notAvailable" ];
  qb:component [ a qb:ComponentSpecification ; qb:measure  kis:study ; skos:
   prefLabel "Specification measure: study" ];
  qb:component [ a qb:ComponentSpecification ; qb:measure  kis:work ; skos:
   prefLabel "Specification measure: work" ] ;
  rdfs:seeAlso <http://www.hesa.ac.uk/includes/C13061_resources/
   Unistats_checkdoc_definitions.pdf?v=1.12> ;
rdfs:isDefinedBy kis:
  .
```

Listing 3: The Employment dataset and data structure definition.

4 Consuming Data Cubes with Vital

Here, we attempt to answer the question *"How feasible is it to use self-descriptive RDF to give an overview of the data in a way that would facilitate debugging and exploration of statistical linked open data?"*.

Our aim is twofold: 1- We want to develop a supporting method for both the data publisher and the data consumer, 2- We want to provide linked data analysists:

- a suitable procedure in order to obtain a representative sample of the data, for early analysis and evaluation, by providing an overview of the datasets available in the data cube along with a description of their content by relying only on the SPARQL endpoint;
- a methodology and a tool to debug linked data cubes and assess possible data quality issues.

We decided to use bar graphs as a means to intuitively explore the data. Bar graphs are simple to manage and more intuitive to consume than other formats easy to produce, like data tables. A bar graph uses either horizontal or vertical bars to show comparisons among objects or categories; in our terms *dimension values*. One axis of the chart shows the dimension values being compared, and the other axis represents a discrete measure value. However a basic bar chart is only made up of two axis. This means that it can only show a single dimension (the categories observed) and a single measure (the values compared). For this reason we should define specific views on the dataset that aggregate the data in order to fit this shape. Data sets can be rather complex, but this complexity is formally expressed in RDF.

4.1 Automatic and dynamic query generation

Our work is then to automatically generate views to be used to aggregate data in a way to fit the shape needed by the diagram. We know `qb:DataSet` objects are described by the means of `qb:DataStructureDefinitions`, that include component specifications, each focusing on one of the following types of component properties: dimensions,

measures and attributes (as shown in the example in Listing 3). Following the data structure definition, we generate views definitions automatically by picking a single dimension and a single measure. Listing 4 shows the SPARQL query we use to extract a documented list of datasets, dimensions and measures. Each row in the result set is a mapping between a dataset, one of its dimensions and one of its measures. These mappings are then used to generate the views specifications. (At the current stage, we only consider dimensions and measures, we leave attributes to future work.) We aggregate

```
PREFIX qb:              <http://purl.org/linked-data/cube#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?dimension ?dataSet ?measure ?dimensionLabel ?dataSetLabel ?measureLabel ?
    measureDescription
$from
WHERE {
?dataSet qb:structure ?structure .
?structure qb:component/qb:dimension ?dimension .
?structure qb:component/qb:measure ?measure .
optional { ?dataSet skos:prefLabel ?dataSetLabel } .
optional { ?dimension skos:prefLabel ?dimensionLabel } .
optional { ?measure skos:prefLabel ?measureLabel } .
optional { ?measure rdfs:comment ?measureDescription } .
}
```

Listing 4: The SPARQL query that extracts the components used to generate the views.

the values by dimension by applying a SPARQL aggregation function to the measure values selected with follow heuristic:

- if the values datatype is numeric, we compute the average (we apply the SPARQL function AVG);
- otherwise, we apply the `COUNT` function to the set of `DISTINCT` values, thus obtaining the number of different values for all observations under the dimension.

According to the Data Structure Definition of the Employment dataset, 18 diagrams are generated. The DataSet is composed by 3 dimensions: Institution, Course and Subject; and 6 measures: Study, Work, Work and Study, Work or Study Assumed Unemployed, Not available. Listing 5 shows the generated query for the view with dimension Institution (aggregating data of any Course and any Subject) and measure Work. The measure value reports the percentage of students working six months after the completion of their course.

```
PREFIX skos:            <http://www.w3.org/2004/02/skos/core#>
PREFIX rdfs:            <http://www.w3.org/2000/01/rdf-schema#>
PREFIX qb:              <http://purl.org/linked-data/cube#>

SELECT (?dimension as ?cat) (AVG ( ?value ) as ?nb) ?label
WHERE {
[] a qb:Observation ;
   qb:dataSet <http://data.linkedu.eu/kis/dataset/employment> ;
   <http://data.linkedu.eu/kis/ontology/institution> ?dimension ;
   <http://data.linkedu.eu/kis/ontology/work> ?value .
   optional { ?dimension skos:prefLabel ?label } .
```

```
}
GROUP BY ?dimension ?label
ORDER BY DESC(?nb) ?label
LIMIT 20
```

Listing 5: The view Employment: Institution + Work.

4.2 Data exploration

Following this approach our system generates 230 diagrams. The information displayed to the user is organized by dataset (see Figure 1). For each dataset a number of panels are available, grouping diagrams *by dimension* or *by measure*. Figure 1 gives an overview of the Employment dataset organized by measure. Each box group diagrams covering a single measure. The user can browse the different dimensions using the selection tool at the bottom of each box. Diagrams covering different measures are shown in parallel, and the user can compare how a measure *performs* under the different dimensions. Similarly, the information can be browsed by dimension. Along with the diagram, links to downloadable CSV or HTML versions of the data are provided, as well as a link to inspect the SPARQL queries that materializes the dynamically generated views. For example, the analyst can observe the top 20 UK universities with the highest average proportion of students in study six months after their course ended, compared with the ones in work.

Along with the capability of browsing the information, Vital also allows the analyst to filter the data by a specific dimension value. By clicking on an item in the diagram, all diagrams of datasets with that dimensions are updated fixing the given dimension value. In Data Cube terms, the views is generated from the Slice with the selected dimension value.

An interesting early analysis might be to check the performance of academic subjects with respect to employment outcome capabilities in the context of a given institution. In other words the analyst might want to check whether this dataset is suitable for building a system to recommend institutions according to some topic of interest and employment outcome. By selecting the *subject* dimension in the *by measure* value we can have the overview of subjects and employment outcomes. Switching to the *by dimension* view we can access the same information and see the most performant institutions from the point of view of the employment outcome. One of them is the "Ravensbourne College of Design and Communication"[17]. By clicking on the related bar, the analyst reloads the information focusing on this institution. She can switch back to the *by measure* panel of the Employment dataset and compare the subject areas under which are classified the courses that have a higher average of students still studying with the ones that mostly have working ex-students (see Figure 1). She notes that *Electronic and Electrical Engineering* has around 11% of students still studying while *Cinematic and Photography* as the 100% of studying students. The analyst can add or remove dimension filters and compare how measures perform under different dimensions and vice versa. She can pick a specific course, or test a specific subject area across the different institutions. Finally, she can explore how specific dimension values affect the information in the different data sets, for example the one about *Common Jobs*.

[17] http://www.ravensbourne.ac.uk/

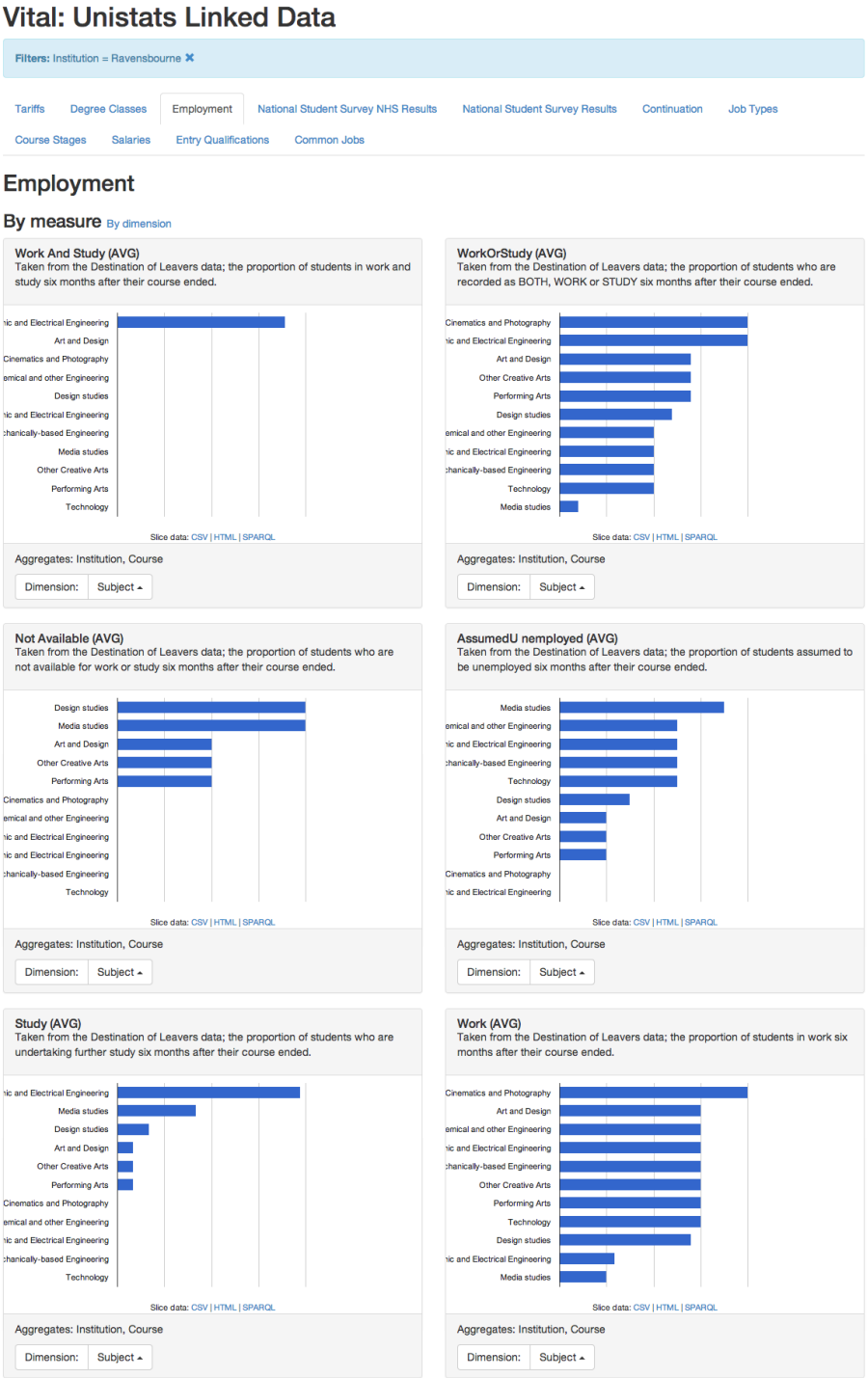

Fig. 1: This example shows an overview of the information of the Employment data set. In this picture diagrams are grouped by measure. Each diagram includes links to raw data as well as the SPARQL query. It is also possible to browse the same information grouped by dimension.

4.3 Debugging methodology

The debugging methodology is a continous iteration of the following steps:

1. browse the diagrams and identify a suspicious graphs (for example, an empty diagram)
2. inspect the SPARQL query and compare it with the obtained result sets
3. identify the error (for example, a wrong data type leading to a void result set)
4. perform the fix on the data remodelling tool

A similar process has been performed to review the documentation attached to dimensions and measures, that is also provided below each diagram. The issues found can be grouped in the following categories: a) insufficient documentation b) wrong data types c) syntax errors in URIs d) inconsistency between the data structure specifications and the observations. With our tool we have been able to discover all of them very easily and to fix the data remodelling procedure accordingly.

With this approach we have been able to solve many issues with the Unistats data. By exploring the diagrams and the generated queries we discovered that most entities did not have a human readable label (in this case the tools displays the local part of the URI). Another issue was that the datatype of many measures was not (or not properly) specified, thus leading to errors in query execution or void result sets. The data provider can have quick access to a number of meaningful and ready-made queries to run over the dataset. The generated queries are also easy to test because of their simplicity, thus allowing an effortless validation of the result sets.

5 Discussion and future work

The Vital approach takes into account with a unique simple tool the basic needs of data publishers and early adopters. The queries are generated automatically, covering core aspects of the data. The user interface allows the exploration of data sets, dimensions, measures and the assessment of their relations as well as an incremental exploration and assessment of the documentation attached to the data. It can contribute to the design of more relevant SPARQL queries from initial drafts. It effectively supported us in the debugging of the remodelling of Unistats data into RDF data cubes. Applying the tool to other data sources is straight forward, requiring minor configuration (basically, pointing to the SPARQL endpoint). During the development, we did some initial tests with third-parties SPARQL endpoint containing data implementing Data Cube. It made emerge immediately the lack of proper documentation and the incompleteness of the RDF structure, for example missing `qb:dataSet` links from observations to the container data set. However, we plan to do a more robust evaluation of both the methodology and the tool.

Future work includes the consideration of other Data Cube components to debug and evaluate, such as Code Lists specifying the possible values that a dimension can have and then use it to browse and test the generated views.

However, there are some limitations that we plan to tackle in future works. On large data sets the generated charts could be too many. We want to verify and refine the

methodology in these scenarions and experiment other ways of organizing the charts accordingly. At this stage we only use two simple aggregation functions: AVG for numeric values and COUNT for string values. We select an aggregation function by looking at the data type of the measure values. We are aware that aggregation rationales may vary depending on the measure itself, not only based on the data type. One option could be to provide the capability to choose an aggregation function to apply. For example counting distinct values of percentages gives a sight on the variability of that measure in the population of a given dimension.

Another aspect is related to the semantics of the measure, that may suggest specific aggregations. One approach could be to map aggregation functions to more abstract measures such as the ones defined in SDMX. The measures used could be declared as rdfs:subPropertyOf the SDMX ones, thus enabling more accurate computations.

An aspect that might influence the accuracy of the diagrams is that we do not consider attributes. The procedure assumes that they are always homogeneous in the observations of a given dataset. In Data Cube terms that they are "attached" at dataset level. We are aware that attribute values may vary between observations of the same data set. Attributes can affect the way measures need to be processed for aggregations. However the objective of our approach is not to produce accurate analytics for decision making but to help the analyist in gaining an early understainding of the core capabilities of a statistical linked open data endpoint.

References

[1] Dave Reynolds and Richard Cyganiak. The RDF data cube vocabulary. W3C recommendation, W3C, January 2014. http://www.w3.org/TR/2014/REC-vocab-data-cube-20140116/.

[2] Valentina Janeva, Bert Van Nuffelenb, Vuk Mijovića, Karel Kremerb, Michael Martinc, Uroš Miloševića, and Sanja Vraneša. Supporting the linked data publication process with the lod2 statistical workbench. *Semantic Web Journal (Submitted) http://www.semantic-web-journal.net/content/supporting-linked-data-publication-process-lod2-statistical-workbench*.

[3] Valentina Janev, Vuk Mijović, and Sanja Vraneš. Lod2 tool for validating rdf data cube models. In *Web Proceedings of the 5th ICT Innovations Conference*, pages 12–15, 2013.

[4] Herman J Ader. Phases and initial steps in data analysis. *Chapter*, 14:333–356, 2008.

[5] d'Aquin Mathieu, Dietze Stefan, Herder Eelco, Drachsler Hendrik, and Taibi Davide. Using linked data in learning analytics. *eLearning Papers. Publisher: openeducation.eu, ISSN: 1887-1542*, 2014.