Karlsruhe Institute of Technology

Search Engine optimized Presentation of Statistical Linked Data

SEO4OLAP - An approach to create SEO-landingpages for all possible views of Linked Data Cubes

Masterarbeit von

Daniel Breucker

An der Fakultät für
Wirtschaftswissenschaften

In dem Studiengang
Wirtschaftsingenieurwesen

Eingereicht am 24. März 2016 beim
Institut für Angewandte Informatik
und Formale Beschreibungsverfahren
des Karlsruher Instituts für Technologie

Referent: Prof. Dr. Studer
Betreuer: Dr. Benedikt Kämpgen

Eidesstattliche Erklärung

Ich erkläre hiermit, dass ich die vorliegende Masterarbeit selbständig und ohne unerlaubte Hilfsmittel angefertigt, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form oder auszugsweise noch keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Karlsruhe, 24. März 2016 Daniel Breucker

Contents

List of Figures

List of Tables

1. Introduction

This chapter introduces the motivation and objectives of our research, followed by an introduction to the structure of the present thesis.

1.1. Motivation

Statistical data is published online by a variety of organizations, among them public agencies and governmental institutions. The statistical office of the European Union *Eurostat* is one of those institutions. Among others, they publish statistics about trade, population, agriculture, economy and finance inside the EU. The data can be accessed by web services, downloaded directly or explored by an online pivot table.

An ISO approved format for publishing statistical data is SDMX (Statistical Data and Metadata eXchange)[1]. It is an industry standard for expressing statistical data, used by *Eurostat* and a variety of other organizations, e.g. the *Organisation for Economic Cooperation and Development* (OECD), the *United Nations* or the *International Monetary Fund*.

SDMX is based on a Multidimensional Data Model (MDM)[CR14], i.e. a data cube, thus allows for Online Analytical Processing (OLAP). OLAP offers operations to analyse, explore and aggregate data cubes. Thereby new information can be derived and knowledge can be generated.

Recent work [CAN13, SMM+12, KOH12] has focused on leveraging statistical data on the Web by means of semantic web technologies in conjunction with OLAP. This

[1] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=52500, last accessed 2016-02-21

has been fostered by the W3C[2], since they recommended the RDF Data Cube Vocabulary (QB)[3] for modeling cube data (especially SDMX) in the Resource Description Framework (RDF).

Regarding the on-going research made in this scientific field, it can be assumed that in the near future organizations are going to publish statistical linked data directly. For the moment, linked data wrappers exist for some of those statistical datasets. In the case of *Eurostat*, the corresponding Linked Data wrapper is *Estatwrap* by *OntologyCentral*[4].

Linked data technologies and the RDF Data Cube Vocabulary allow to publish statistical data in a standardized format, interlink different datasets and retrieve information with the standard query language SPARQL. This has great potential for many different fields. Empirical experiments could be published and reused by other research parties. New services on open data may evolve, leading to new business opportunities. One can imagine a question answering system which processes statistical data from many different sources, thus generating new knowledge.

Despite the potential of statistical linked data, human friendly interfaces to explore and interact with it, are still suspect to ongoing research [Hoe13, PCH+12, MHT+14]. The usage of SPARQL, the query language for linked data, is the most common way to retrieve information from linked data. Since SPARQL can only be used by experts in the field of semantic technologies, the information published in the semantic web is not available for usual web surfers.

A common way to retrieve information from the Web is the usage of search engines. Since search engines are focusing on HTML-based content, linked data is usually not included in search results. It has to be noted that major search engines are improving their algorithms with semantic technologies by building their own knowledge graphs, such as the Google Knowledge Graph[5]. Nevertheless, included information of the graph is preprocessed and distinctly selected. Therefore, published linked data is not included by default.

This leads to the following problem: Even though statistical data is published by a variety of organizations, search engines are not able to retrieve corresponding results. As an example, *Eurostat* publishes a dataset about employment statistics in the European Union. It includes various facts, such as the employment rate of Germany in 2014. By asking Google for "employment rate germany 2014", one would expect this dataset among the search results, which is not directly the case. One has

[2]https://www.w3.org/, last accessed 2016-02-21
[3]https://www.w3.org/TR/vocab-data-cube/, last accessed 2016-02-21
[4]http://estatwrap.ontologycentral.com/, last accessed 2016-02-21
[5]https://www.google.com/intl/es419/insidesearch/features/search/knowledge.html, last accessed 2016-03-01

to search for ”employment statistics europe” in order to find the dataset and then manually retrieve the exact information from *Eurostat*.

This leads to the research question of the present thesis. We want to analyze how statistical linked data can be published in order to allow search engines to properly index a dataset. This provides benefits for both publishers and search engines. Publishers can attract more users to their website and services. Search engines can show relevant content in their search results. The concrete objectives of this thesis are presented in the following section.

1.2. Objectives

The present thesis analyzes how statistical linked data can be published in order to allow search engines to index the entire information. This includes base facts from the dataset, as well as all possible computed facts such as aggregations.

Our approach to solve this problem is to generate search engine optimized webpages for all possible facts of a dataset. Once those webpages are published, search engines should be able to crawl the content. From this approach the following research questions can be derived and will be answered in this thesis.

- Is it possible to automatically create search engine optimized webpages for all facts of a statistical dataset? How does the architecture of such a system look like? Is it improving the status quo?
- What is the computation complexity of this problem? How many webpages are created depending on the amount of dimensions, measures and dimension values?
- How can semantic markup technologies such as Schema.org contribute to this?
- Which requirements does a dataset have to fulfill? What problems exist in data modeling?

In order to answer these questions, we developed a system architecture called *SEO4OLAP* which is able to generate webpages for arbitrary datasets modeled in the RDF Data Cube Vocabulary. In order to evaluate our approach, we implemented a concrete system in Java and published two datasets from *Eurostat*.

1.3. Structure of the present Thesis

After explaining the motivation and objectives of this thesis, the following chapter introduces the theoretical background in the fields of Multidimensional Data Modeling, Online Analytical Processing and Search Engine Optimization. In chapter

3, the system architecture of our approach *SEO4OLAP* is presented. We explain the main system components, discuss the mathematical complexity of the problem and demonstrate an algorithm which computes all possible queries. The approach is evaluated in chapter 4, where we present our concrete implementation and evaluate the system based on two datasets from *Eurostat*. The succeeding chapter 5 discusses our findings and reacts to our research questions. We present related work in chapter 6 and summarize and conclude our findings in chapter 7.

2. Theoretical Background

In order to understand the concepts of our approach, which will be presented in the following chapter 3, this chapter introduces theoretical background.

First we explain the general concept of a Multidimensional Data Model (MDM), the model of our examined datasets. After that, we describe the functionality of OLAP (Online Analytical Processing), a methodology to analyze MDMs. In our approach we use OLAP-Operations in order to query datasets. Since our approach is based on semantic web technologies, we further explain core concepts, especially the Data Cube Vocabulary which is an RDF vocabulary designed to express MDMs as Linked Data. At last, we give an introduction to Search Engine Optimization (SEO).

2.1. Multidimensional Data Model

Statistical datasets are often multidimensional. As an example, one can imagine a dataset showing the employment rate of European countries by gender and date. This scenario has one measure, the employment rate, and three dimensions: country, gender and year. In practice, such datasets are often visualized by a pivot table (see section 2.2.3) which allows the presentation of multiple dimensions in a two-dimensional table.

An ISO approved format for publishing statistical data is SDMX (Statistical Data and Metadata eXchange)[1]. It is considered to be the industry standard for expressing statistical data, providing a highly structured mechanism to represent statistical observations, classifications and data structures [CAN13]. Among other organizations, *Eurostat*, the statistical agency of the European Commission, uses SDMX as

[1] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=52500, last accessed 2016-02-16

data exchange format. The dataset used for our evaluation (see chapter 4.3.1) is from *Eurostat*.

Although there is no standard MDM, SDMX and other formats or models, all have certain *Multidimensional Elements* in common. We adapt the definition of a MDM from Benedikt Kämpgen [Käm15] and illustrate a common MDM in figure 2.1. Afterwards we explain every element in an informal manner. For a formal definition in set notation, we refer to [Käm15].

Definition 1 (Multidimensional Data Model) *"A multidimensional data model treats data as n-dimensional data cubes. The independent attributes of a data cube are called dimensions, the dependent attributes measures. The possible values of dimensions are referred to as members. Members are grouped along hierarchies of one or more levels. The higher the level in the hierarchy, the less granular the members. Facts are the single data points in a cube. Facts have a value for every dimension and measure of the cube." [Käm15, p. 33]*

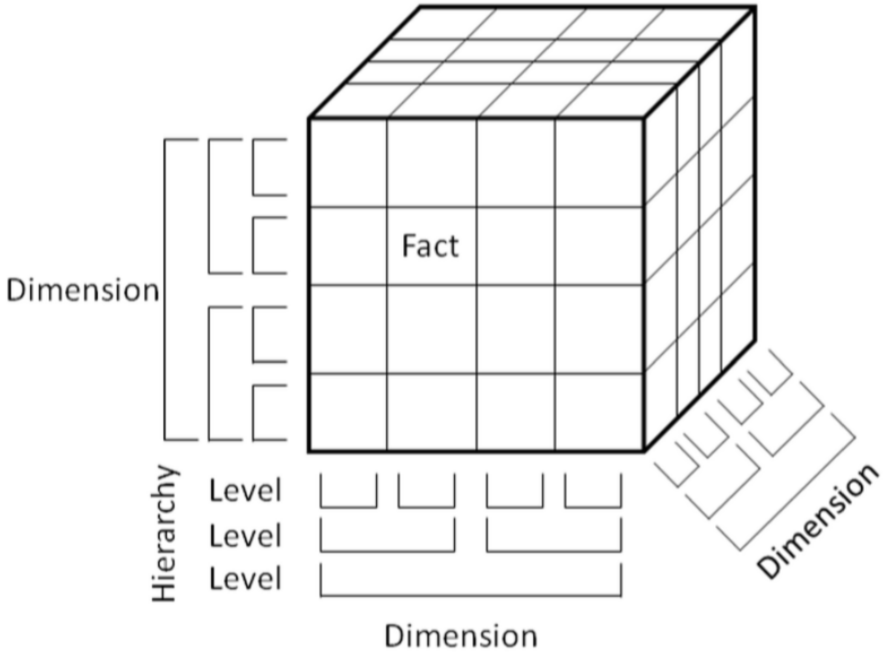

Figure 2.1.: Illustration of a common Multidimensional Data Model [Käm15]

Dimension and Members

A dimension is an independent variable consisting of a member-set (possible values), e.g. *country, gender, date* with possible members {Germany, Spain, Poland ...}, {female, male} and {2010, 2011, 2012, ...}.

Level and Hierarchy

Members can be grouped in hierarchies consisting of different levels with a certain depth. As an example, the dimension *date* could have the members *January 2010, Febuary 2010, ... , December 2010*. A possible aggregation of these members to a higher level would be the year *2010*. Every dimension has the implicit *ALL*-Member, which represents an aggregation to the highest level (with depth 0).

Measure

Measures are dependent variables describing an observation, such as the *Employment rate*. Its value depends on the dimensions attributes, e.g. the employment rate is different in every year. A data cube can consist of multiple measures.

Fact

A fact represents a single observation from a statistical dataset. For every fact, each dimension and measure has a maximum of one member and value. A fact from our Employment dataset would be: *The employment rate 2013 of women in Germany is 72,5 %*.

Data Cube Schema

A data cube schema consists of a set of measures, dimensions and their corresponding members, levels and hierarchies.

Data Cube

A data cube represents the set of all facts for a given Data Cube Schema. There are several assumptions on data cubes, which we adapt from Benedikt Kämpgen.

- "The measure value is fully dependent on the dimension members, thus, any two facts of a data cube need to have a different member on one of their dimensions.
- Each member needs to be contained in a level of a hierarchy of the dimension. A data cube may be sparse and not containing facts for each possible combination of dimension members.
- If not said otherwise, we assume the explicitly given facts of a data cube to be *base facts*. Base Facts have members only on the lowest level of each dimension.
- Implicitly, a data cube contains aggregate facts, facts with members on higher levels for dimensions that can be computed by aggregating lower-level facts, e.g., facts describing Male and Female can be aggregated to ALL, meaning the total of Male and Female [GCB+97]." [Käm15, p. 35]

2.2. Online Analytical Processing

Multidimensional data cubes can be analyzed by means of Online Analytical Processing (OLAP). An OLAP-system typically consists of an ETL pipeline which extracts, transforms and loads data from whichever source into a data warehouse, e.g. a relational or multidimensional database. OLAP-clients such as Saiku or JPivot enable users to run queries on data cubes and display results in pivot tables. OLAP-engines, such as Mondrian, transform OLAP-queries into the target query language of the data warehouse, e.g. SQL or SPARQL queries. [KOH12][EV12]

In the following, we define OLAP-operations and describe how these operations can be used to build OLAP-queries in subsection 2.2.2. In subsection 2.2.3 we explain how OLAP-results are displayed in pivot tables.

2.2.1. OLAP Operations

OLAP-operations are used to manipulate data cubes, e.g. to filter or aggregate values of a cube.

In the following, we define OLAP-operations adapted from [KOH12], [RMA+11] and [EV12]. Figure 2.2 illustrates the effect of common operations with inputs and outputs.

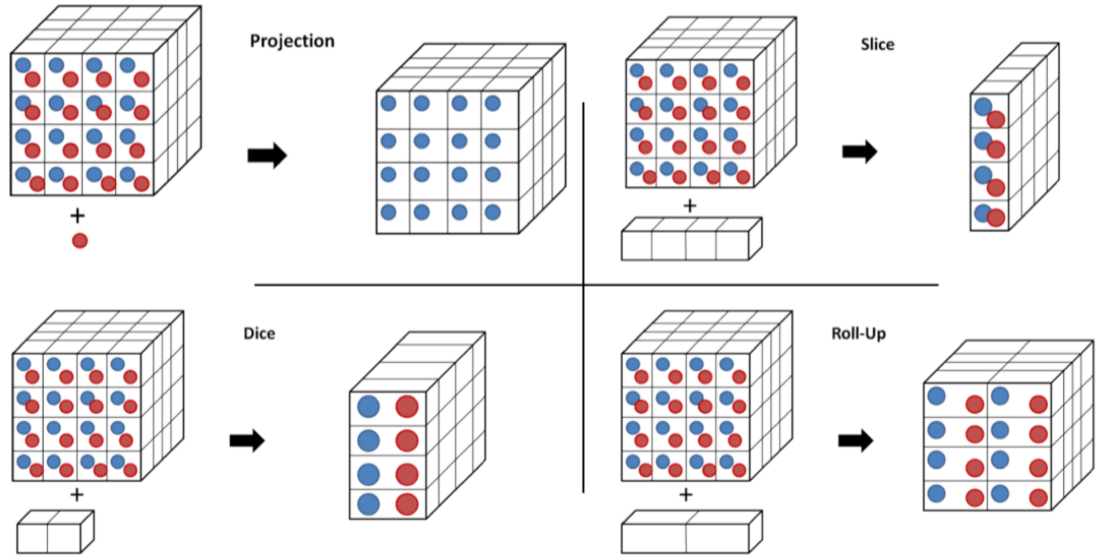

Figure 2.2.: Illustration of common OLAP-operations with inputs and outputs [KOH12]

Projection selects a subset of measures, e.g. select *Employment rate* and neglect all other measures.

Slice removes a dimension from the input cube by aggregating over all members of the corresponding dimension, e.g. remove the dimension *Gender*, thus aggregate *Female* and *Male* to *Total*.

Dice filters for selected members of a dimension, e.g. keep only those facts with *Germany* as member of the dimension *Country*.

Roll-Up summarizes data to a higher level in the hierarchies of a dimension, e.g. aggregates all months to the level of years in the dimension *Date*.

BaseCube returns the full Cube without any alterations through other OLAP-operations.

The output of any operation is called **SubCube**, since it is a subset of the input cube.

2.2.2. OLAP Queries

The input and output of any OLAP-operation is a data cube, which allows the consecutive execution. A nested set of OLAP-operations is called an OLAP-query [KOH12].

OLAP-queries can be expressed in various ways. As an example, Kämpgen et al. [KOH12] define an OLAP-query as a *subcube query*, which is a tuple of dimension members and selected measures for a predefined data cube. In section 3.2, we adapt this idea and develop a new query model.

Another way to express OLAP-queries are query languages. A widely adapted declarative query language is *Multidimensional Expressions* (MDX), which is among others part of Microsoft's OLAP product[2].

An OLAP-engine needs to transform an OLAP-query into a native database query, such as SQL or SPARQL. Independent from the concrete implementation of OLAP-engines, on a conceptual level, the OLAP-query first needs to be translated into a series of basic OLAP-operations. We define this as a *Logical OLAP Query Plan* in Definition 2.

Definition 2 (Logical OLAP Query Plan) *A Logical OLAP Query Plan is a series of OLAP-operations which are executed by an OLAP-engine in the defined order. The output of an operation is the input of the succeeding operation.*

[2]https://technet.microsoft.com/en-us/library/aa216767%28v=sql.80%29.aspx, last accessed 2016-02-16

As an example, consider a query for the *Employment rate in Germany per year* on our previously introduced dataset with the measures *Employment rate* and *Absolute Employment* and the dimensions *Country, Gender and Date*.

A possible Logical OLAP Query Plan is illustrated by figure 2.3. At first, the whole cube is derived by the BaseCube operation. With a Projection on the measure *Employment rate*, all other measures are excluded from the result. Since the query does not explicitly ask for *Female* or *Male* results, the dimension *Gender* has to be sliced, thus aggregated to *Total*. The following Dice-Operation filters the dimension *Country* for the member *Germany*. The Query is completed by a Roll-Up on the dimension *Date* in order to aggregated all dates to the higher level *Year*.

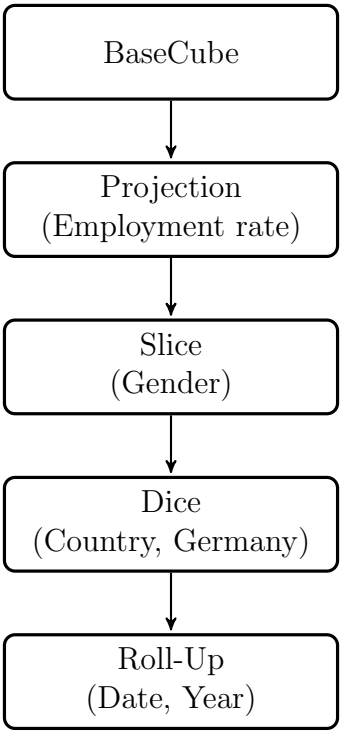

Figure 2.3.: Schematic illustration of a Logical OLAP Query Plan

A *Logical OLAP Query Plan* has to start with a BaseCube operation as first input for the succeeding operations. The order of these operations does not determine the final output. It can however have an effect on the computation time, since the amount of computed aggregations may vary.

2.2.3. Result Visualization

The previous sections explained how OLAP-operations can be composed to build an OLAP-query. This section introduces the standard visualization form of OLAP-results, the pivot table.

As previously stated, Multidimensional Expressions (MDX) is a widely adapted query language for OLAP. Whereas SQL queries return a relational table as result, MDX returns parts of a data cube, which can be displayed in a pivot table [GCB+97].

We define *pivot tables* as per Definition 3 and as illustrated in Figure 2.4, both adopted from Benedikt Kämpgen [Käm15].

Definition 3 (Pivot Table) *"Pivot tables display data from a data cube in a compact, two-dimensional, tabular form where both the number of rows and columns are variable depending on the multidimensional dataset represented in the cube [CGLG04]. The metadata of a pivot table describes a queried data cube, lists of member combinations (positions) from a fixed set of levels from different dimensions to be displayed on rows and columns, and member combinations from a fixed set of levels as filter conditions about which facts to summarise in the pivot table. The cells in a pivot table are populated with measure values of facts in the cube." [Käm15]*

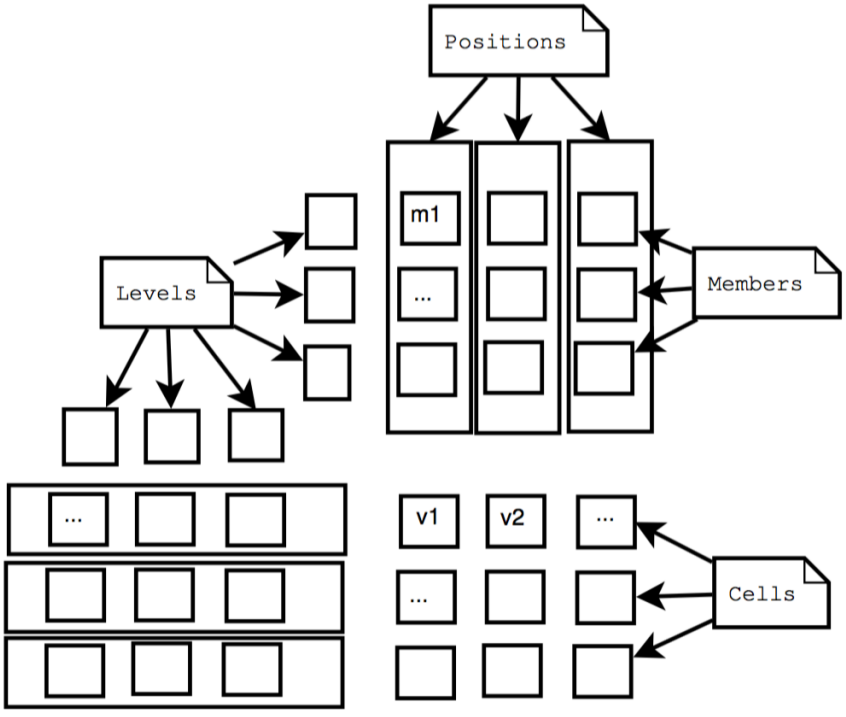

Figure 2.4.: "Schema of pivot table as generated by a typical MDX query; this pivot table displays six levels of different dimensions on columns and rows." [Käm15, p. 49]

Spreadsheet programs, especially Microsoft Excel[3], but also more advanced Business

[3]https://products.office.com/de-de/excel, last accessed 2016-02-16

Intelligence Software such as Tableau[4], Qlik[5] or Cubeware[6] display OLAP-results as pivot tables and provide intuitive interfaces for generating MDX queries.

An example visualization of a pivot table on the *Employment dataset* [7], generated by the *Linked Data Cubes Explorer*[8] is shown in Figure 2.5.

		1564 EMPRT		
		F	M	T
AT	2007	63.5	76.3	69.9
	2008	64.8	76.8	70.8
	2009	65.2	75.5	70.3
	2010	65.7	76.0	70.8
	2011	66.1	76.2	71.1
	2012	66.7	76.2	71.4
	2013	66.9	76.0	71.4
	2014	66.9	75.2	71.1
BE	2007	55.3	68.7	62.0
	2008	56.2	68.6	62.4
	2009	56.0	67.2	61.6
	2010	56.5	67.4	62.0
	2011	56.7	67.1	61.9
	2012	56.8	66.9	61.8
	2013	57.2	66.4	61.8
	2014	57.9	65.8	61.9
BG	2007	57.6	66.0	61.7
	2008	59.5	68.5	64.0
	2009	58.3	66.9	62.6
	2010	56.2	63.3	59.8
	2011	55.6	61.2	58.4
	2012	56.3	61.3	58.8
	2013	56.8	62.1	59.5
	2014	58.2	63.9	61.0

Figure 2.5.: Pivot table showing the Employment rate in Austria, Belgium and Bulgaria, per year and gender (Female, Male and Total)

2.3. Statistical Linked Data

In our evaluation, we use open statistical datasets modeled as *Statistical Linked Data* (SLD), defined as per Definition 4.

[4]http://www.tableau.com, last accessed 2016-02-16
[5]http://www.qlik.com/, last accessed 2016-02-16
[6]http://de.cubeware.com/, last accessed 2016-02-16
[7]http://appsso.eurostat.ec.europa.eu/nui/show.do?wai=true&dataset=lfsi_emp_a, last accessed 2016-03-18
[8]http://km.aifb.kit.edu/projects/ldcx/, last accessed 2016-02-16

Definition 4 (Statistical Linked Data) *"Statistical Linked Data are RDF data with multidimensional datasets properly modelled and published as Linked Data according to the RDF Data Cube Vocabulary." [Käm15, p.51]*

In order to understand the benefits of this data model, the following sections introduce core concepts and technologies. In section 2.3.1, the four principles of Linked Data are described, followed by a brief introduction to the *Semantic Web*. In section 2.3.3 we explain *RDF*, the Resource Description Framework, and the corresponding query language *SPARQL*. The chapter is completed by an overview of the RDF Data Cube Vocabulary.

2.3.1. Linked Data

Linked Data is an approach to connect data from different sources based on "Best Practices" introduced by Tim Berners-Lee in his Web architecture note [BL06]. These best practices have become known as the *Linked Data Principles* presented in the following:

1. "Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)
4. Include links to other URIs. so that they can discover more things."[BL06]

The **first principle** states that URIs should be used to identify not just Web documents or digital content, but also real world objects or concepts. This means that not only persons or things, but also abstract concepts, e.g. the relationship status *knowing someone* or a colorset should be identified by URIs. In consequence, the scope of the Web is extended from a source of electronic data by all concepts and objects of the real world.[HB11]

HTTP is the standard protocol for accessing data on the Web. By using HTTP URIs according to the **second principle**, users or machines are enabled to dereference (i.e. to *lookup*) any resource in order to retrieve a description or further information about it. Whereas human beings usually prefer a representation in HTML for better readability, machines require data in a machine-readable format such as RDF. This can be managed by a mechanism called *Content Negotiation*, which is possible through HTTP. [HB11]

In order to enable a wide range of different applications to process Web content, the **third principle** states that standard technologies, especially RDF and SPARQL

should be used. These graph-based technologies will be further explained in section 2.3.3. By agreeing on standard formats, an important factor for scaling Linked Data is created. [HB11]

By using links to other URIs, especially links from other namespaces, the reuse of vocabularies is possible. This allows the interconnection of different datasets and thus enables users to retrieve further information about other resources. As an example, a lookup on *Karlsruhe* can lead to the information *"Karlsruhe is a city in Germany"*. By interlinking *Germany* to Karlsruhe, a user is enabled to gather further information about Germany. [HB11]

During the last years, a growing amount of data has been published as Linked Data which resulted in an open available data graph. The following section introduces this data graph which we refer to as *Semantic Web*.

2.3.2. Semantic Web

Publicly available Linked Data is referred to as *Linked Open Data* (LOD). The origins of this lie in the efforts of the Semantic Web research community and particularly in the activities of the W3C Linking Open Data (LOD) project18, a grassroots community effort founded in January 2007. The original idea was to identify existing datasets available under open licenses, convert those to RDF and publish them under the four principles of Linked Data. [HB11]

Since then, numerous companies, individuals and organizations have contributed to this "Web of Data". As a result, a constantly growing graph of interlinked dataset has been created, which is visualized in Figure 2.6. The so called *Linked Open Data Cloud* represents the efforts of the LOD-community to make data openly available and is a symbol for the *Semantic Web*. In the graph, every node is a dataset and edges in between represent links.

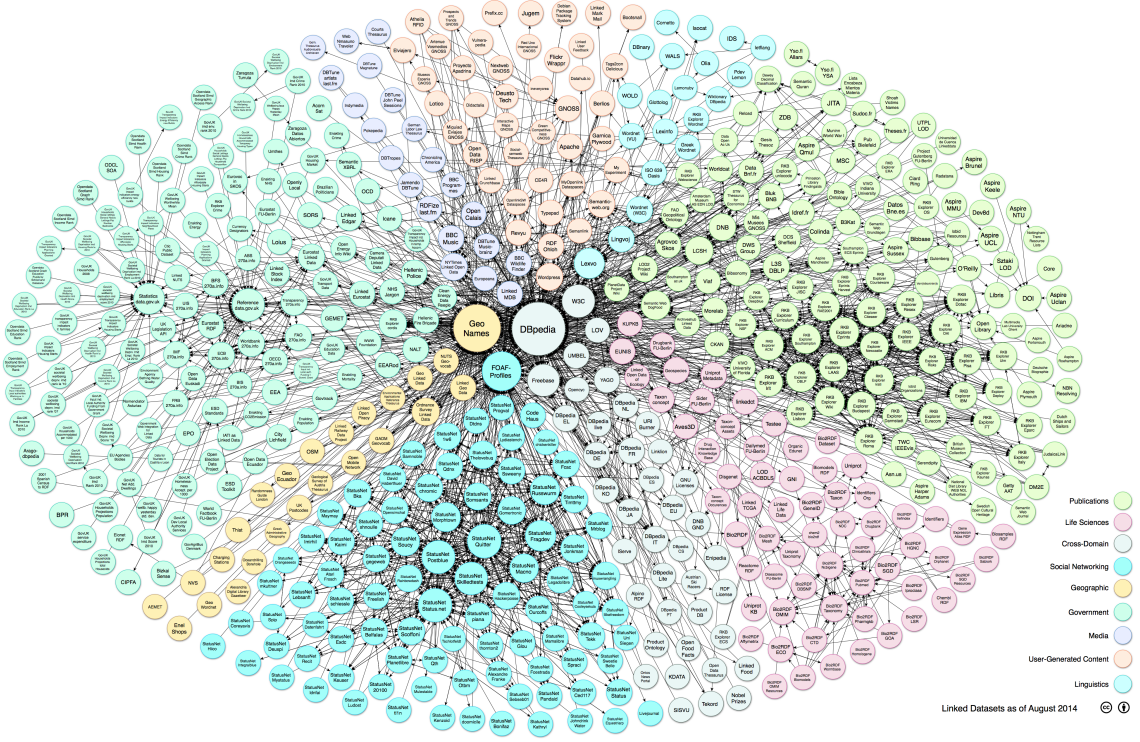

Figure 2.6.: Linked Open Data Cloud diagram as of August 2014 [SBJC14]

The graph includes various different topics such as public figures, companies, literature, movies, music, but also statistical datasets can be found. The following list demonstrates two examples of statistical linked datasets, which may be used as input data for our approach.

- **Eurostat wrapped by OntologyCentral:** Eurostat[9] is the statistical bureau of the European Commission. They publish a wide range of statistical datasets regarding the European Union such as statistics about population, employment, agriculture, financials and many more. The data is available as Linked Open Data in the QB-Vocabulary through OntologyCentral.[10].
- **EDGAR wrapped by OntologyCentral:** EDGAR[11] provides access to data of companies, who are required by law to file forms with the U.S. Securities and Exchange Commission. The data is available as Linked Open Data through OntologyCentral.[12].

[9]http://ec.europa.eu/eurostat/, last accessed 2016-02-16
[10]http://estatwrap.ontologycentral.com/, last accessed 2016-02-16
[11]http://edgar.sec.gov/, last accessed 2016-02-16
[12]http://edgarwrap.ontologycentral.com/, last accessed 2016-02-16

2.3.3. Semantic Technologies

RDF[13] and SPARQL[14] are the standard technologies used for Linked Data, which is why they are introduced in the following.

2.3.3.1. Resource Description Framework

RDF is a data model for expressing assertions over resources identified by a URI. Assertions are expressed as triples of subject, predicate and object [EV12]. A set of RDF triples forms a directed graph, defined as per Definition 5 and illustrated in Figure 2.7.

Definition 5 (RDF Graph with Terms and Triples) *"The set of terms in an RDF graph consists of the set of HTTP URIs $\mathcal{I}$, the set of blank nodes $\mathcal{B}$ and the set of literals $\mathcal{L}$. A triple (s,p,o) $\in \mathcal{T} = (\mathcal{I} \cup \mathcal{B}) \times \mathcal{I} \times (\mathcal{I} \cup \mathcal{B} \cup \mathcal{L})$ is called an RDF triple, where s is the subject, p is the predicate and o is the object." [Käm15, p. 52]*

Figure 2.7.: RDF triple with subject, predicate and object

There are several serialization formats to RDF, such as RDF/XML, Turtle, N-Triples or JSON-LD. The presented examples in this thesis use the Turtle representation, since it is easy to understand for humans.

A basic turtle triple has the following structure:

```
ex:subject ex:predicate ex:object.
```

A URI is divided in two parts, with the prefix before and the ID after the double dot. In this thesis, the used prefix `ex:` always refers to an example namespace. For better readability we abbreviate URIs with well-known prefixes as listed by prefix.cc[15].

The Resource Description Framework Schema (RDFS)[16] is a particular RDF vocabulary, with a set of reserved words for describing relationships or properties,

[13] https://www.w3.org/TR/rdf-concepts/, last accessed 2016-02-16
[14] https://www.w3.org/TR/sparql11-query/, last accessed 2016-02-16
[15] http://prefix.cc, last accessed 2016-02-04
[16] https://www.w3.org/TR/rdf-schema/, last accessed 2016-02-04

e.g., an attribute of a resource. Some of those reserved words are `rdfs:range`, `rdfs:domain`, `rdfs:type`, `rdfs:subClassOf` or `rdfs:subPropertyOf` [EV12]. By this, RDFS offers the possibility to semantically enrich RDF documents. It allows to model simple ontologies, so that common domains can be represented in a standardized way. Several of these domain specific vocabularies are widely adapted and reused by the Linked Data Community. One of these vocabularies is the *RDF Data Cube Vocabulary (QB)* which is used to describe multidimensional datasets and will be presented in the following section 2.3.4.

2.3.3.2. SPARQL

SPARQL[17] is the W3C standard query language for RDF. One can distinguish between three different types of queries:

- **SELECT** queries return results in a tabular representation, similar to SQL.
- **CONSTRUCT** queries allow to create new RDF documents.
- **ASK** queries return a boolean result, matching a statement against an RDF graph.

The query evaluation mechanism of SPARQL is based on subgraph matching [EV12]. The selection criteria is expressed as a graph pattern in the *WHERE* clause, which is matched against the data graph. An example query is shown in the following, asking for all objects and their corresponding label.

```
PREFIX rdfs:    <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?x, ?label
WHERE {
     ?x rdfs:label ?label.
}
```

With the new version *SPARQL 1.1* new functionalities such as aggregations and subqueries are possible, which is of particular interest to our work.

2.3.4. RDF Data Cube Vocabulary

The RDF Data Cube Vocabulary (QB) is a W3C recommendation for publishing statistical datasets as Linked Data. It is based on the cube model of SDMX, an ISO Standard for exchanging and sharing statistical data and metadata among organizations [CR14] and thus allows to easily map SDMX to QB. Even though there

[17] https://www.w3.org/TR/sparql11-query/, last accessed 2016-02-04

are several other vocabularies, e.g. *SCOVO*[18] or *Open Cubes* (see [EV12]), in the present thesis only datasets in the QB-Vocabulary are taken into account, due to the W3C recommendation.

The following Figure 2.8 provides an overview of core classes of the QB-Vocabulary.

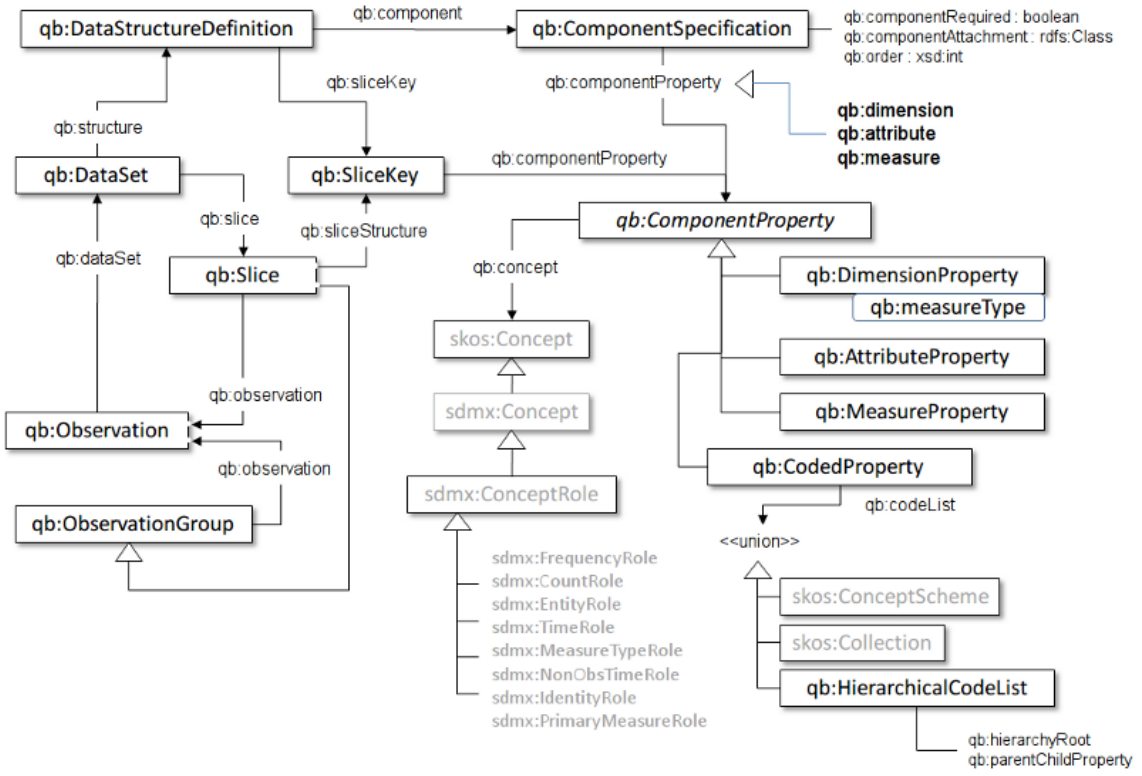

Figure 2.8.: Core components of the RDF Data Cube Vocabulary [CR14]

In QB every dataset (instance of `qb:DataSet`) is defined by a structure (instance of `qb:DataStructureDefinition`) consisting of components (`qb:ComponentProperty`) which define the cube. These components are especially the dimensions (`qb:DimensionProperty`), e.g., *Time* or *Region*, and measures (`qb:MeasureProperty`). A dataset consists of observations (instances of `qb:Observation`) which are the actual facts of the dataset. Every observation has a value for each of the measures and dimensions, with the measure values being functionally dependent on the dimensions [Käm15]. Dimension values, i.e. the members, are predefined as `skos:Concept` in a `skos:ConceptScheme` inside the `qb:DataStructureDefinition`. The following triples of Listing 2.1 show an example observation from our *Employment*[19] dataset, representing the fact that the employment rate in 2008 of women in Germany was 67,8%.

[18] http://vocab.deri.ie/scovo, last accessed 2016-02-04
[19] http://appsso.eurostat.ec.europa.eu/nui/show.do?wai=true&dataset=lfsi_emp_a, last accessed 2016-02-16

```
_:obs1 a qb:Observation ;
        qb:dataSet eurostat-employment:ds;
        estatwrap:sex estatwrap:F;
        estatwrap:indic estatwrap:EMP_RT_20_64;
        estatwrap:geo estatwrap:DE;
        dcterms:date "2008";
        sdmx-measure:obsValue "67.8"^^xsd:double.
```

Listing 2.1: Sample observation as RDF triples

2.4. Search Engine Optimization

In order for websites to be found by search engines, a series of techniques have evolved to influence the ranking algorithms. These methods are summarized by the term *Search Engine Optimization* (SEO). It is a rather new discipline, with constant changes and a high importance in the field of *Online Marketing*. Even though there are some academic papers targeting this topic ([BGW09], [SCC13], [Mal09]), the main discussion of trends and new techniques is driven by private companies and communities (e.g. Moz and others[20]).

The overall idea of SEO is to configure websites in an optimized way in order to get the best possible ranking by search engines. The techniques of SEO can be divided in two groups, *On-Page Optimization*, summarizing all methods that can directly be applied to a website itself, and *Off-Page Optimization*. The latter groups techniques raising the overall importance of a website. It has to be noted, that the underlying algorithms of search engines are disclosed and constantly updated. In consequence, SEO is a set of best practices derived from experiments. Techniques which worked in the past may not work in the near future.

For a better understanding of why SEO should be used for websites, we describe its significance in the following section 2.4.1. In order to understand the used techniques, section 2.4.2 briefly explains how search engines work. Afterwards, the most common methods of On-Page and Off-Page Optimization are presented.

2.4.1. Significance of SEO

For most websites, especially those with no brand, search engines are the major source of user traffic [Rog11]. One can distinguish between paid and organic traffic.

[20] https://moz.com/blog
http://searchengineland.com/
https://www.seo.com/blog/

Paid traffic is generated from advertising campaigns, which can be bought from all major search engines. As an example, Google AdWords[21] provides a straightforward process to set up an advertising campaign. Paid results are usually displayed on top and on the right of a Search Engine Result Page (SERP). The organic search results are displayed afterwards.

A recent study [Pet14] examined that 71% of all searches on Google end in a click on the first result page and that the first five results account for 67% of all clicks. Since users are also inclined to trust organic results more than paid ones [Mal09], it should be in the best interest of every website depending on user traffic to be as high in the organic search ranking as possible. This can be achieved with successful SEO. Its significance comes from the fact that it increases the volume of traffic directed to individual sites from search engines, thus enhancing a websites visibility and user interaction [SCC13].

During the last years, Google manifested its position as market leader. With a share of 94% in Germany[22] and similar allocations in other European countries, the main focus of every SEO activity should be to optimize for Google first. As for this reason, we focus on Google while evaluating our approach in chapter 4.

2.4.2. Components of Search Engines

In order to understand SEO techniques and how they influence search engine rankings, it is key to understand the general system architecture of search engines. Users interact with the graphical user interface, which is a search box for the query and a result page, displaying about ten organic results per page. In contrast to a widespread assumption, the browser does not display live-results. Before a query is analyzed, a variety of different system components have already preprocessed all potential results in order to retrieve a fast answer [Erl16, p. 201]. In general, one can assign three main functions to search engines, each being handled by one key component. In the following those functions are explained.

2.4.2.1. Data Collection

Before data can be analyzed and prepared for later retrieval, it has to be collected. In the case of web search engines, a Webcrawler-System is responsible for this function. The main task of such a system is done by a so called *spider* or *crawler* which visits websites, downloads the content and follows outgoing links to other webpages. In order to index the entire Web, a Webcrawler-System consists of many independent

[21] https://www.google.de/adwords/, last accessed 2016-02-05
[22] http://de.statista.com/statistik/daten/studie/167841/umfrage/marktanteile-ausgewaehlter-suchmaschinen-in-deutschland/, last accessed 2016-02-05

crawlers managed by a scheduler. With millions of websites generated every day, the system has to independently find new documents as well as check whether old content was updated [Erl16, p. 209].

In consequence, for new websites to be indexed by a search engine, a crawler must be told the URL to visit. There are two possible ways to achieve this. Some search engines such as Google have tools for webmasters[23] where new websites can be submitted. The more natural way is to link from other websites to the new domain. At some point, a crawler will follow these links and the new website will be indexed.

2.4.2.2. Data Analysis and Administration

After the documents were downloaded, they are converted into a searchable data structure. This is done by an *Information-Retrieval-System*. At first documents have to be converted into a normalized data representation structure which is done by a *Parser*. This normalization process has many steps, with the overall goal to extract the most relevant keywords of a document and thereby recognizing the documents content. Among others, this multilevel process consists of the following steps: [Erl16, p. 225]

- Data normalization
- Word identification
- Language identification
- Word stemming
- Word-group identification
- Stop-word identification
- Keyword extraction
- URL processing

After this process, the parser has tagged the keywords of a document. In order to identify the most relevant, a weighting has to be applied. A general assumption states that the more often a word occurs in a text, the more important it is. This can be calculated, e.g. with the *Keyword-Density*. But there are other factors besides the amount of occurrences of a keyword which influence the weighting. Text in a headline or bold text is often more important than regular text [Erl16, p 491 ff.]. As previously mentioned, the algorithms of different search providers are disclosed and differ from each other. At this point, it is important to understand, that many different factors influence the weighting of a keyword for a document. How this

[23] https://www.google.com/webmasters/tools/, last accessed 2016-02-05

can be used for optimization will be explained in the following section 2.4.3 about *On-Page Optimization*.

The extracted and weighted keywords are finally stored in an *Inverted Index*, a structure which allows for fast retrieval. It can be compared to a book index, where certain keywords are listed with corresponding pages of their occurrences [LM06, p. 19]. Even though the architecture of an index for the Web is more complex, the underlying principle remains the same.

2.4.2.3. Query Processing

Whereas the other two components work day and night to nurture the system with an up-to-date index, the query processor starts working at the moment it receives a new request, e.g. by a user through the searchbox. Its task is to analyze the query, request documents from the inverted index and generate a weighted list of results. In the end, it displays the result pages to the user. Again, the concrete weighting is influenced by many different factors [Erl16, p. 202]. As some of those factors are well known to have a great impact on the ranking, the following section explains the most common techniques.

2.4.3. On-Page Optimization

On-Page Optimization is a term describing all activities assigned to a site itself. On the contrary, *Off-Page Optimization* stands for all activities, which cannot directly be influenced by the site itself (see section 2.4.4).

In the following the different aspects of On-Page Optimization are presented.

2.4.3.1. Keyword Categorization

For effective SEO, one has to identify the right keywords. They are different in every domain and depend on the overall SEO-strategy. In order to assess the best keywords, different aspects, such as the monthly search volume, competition and available resources [Erl16, p. 92 ff.] have to be regarded. As an example, it is much easier to achieve a good ranking for the keyword ”restaurant karlsruhe” compared to ”restaurant”, since the competition is only regional and not global.

Keywords can be categorized according to the *Longtail-Principle*. The term *Longtail* was first mentioned by Chris Anderson [And08]. It describes the phenomenon, that the Web enables a market for niche-products, because also poor selling products can be offered due to the mass of potential customers. Adapted to keywords, this means that some keywords attract a lot of customers and thus competition (the *Shorttail*). An example keyword in the field of statistical datasets would be the

term ”Population”. On the other side, in the *Longtail*, many keyword combinations are possible which still attract some users, e.g. the query ”Population of women in Germany in 2015”. This categorization is illustrated by Figure 2.9 and explained in the following.

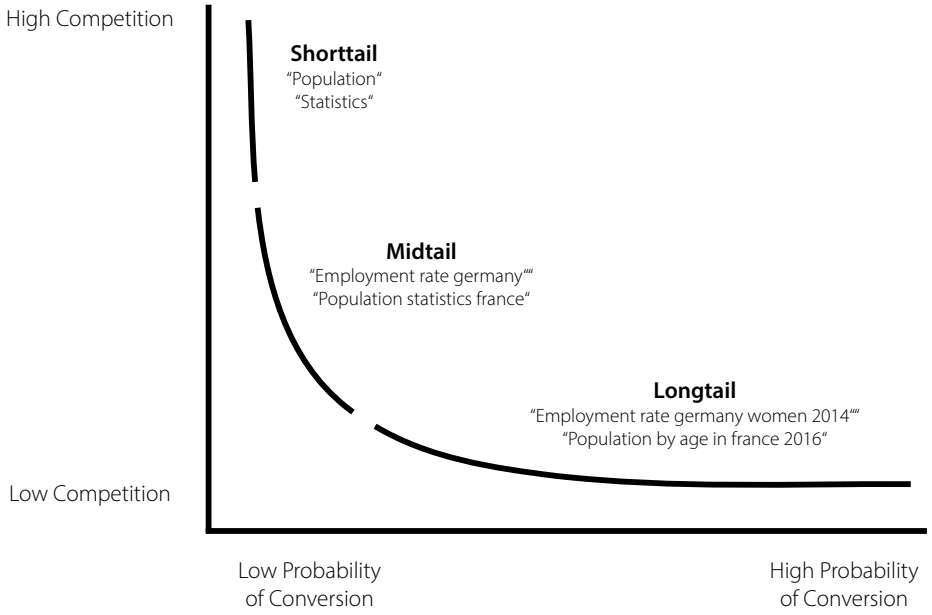

Figure 2.9.: The Longtail-Principle

- **Shorttail:** Usually one, maximum two keywords with high competition and many potential users, e.g., ”Population” or ”Employment rate”.
- **Midtail:** Two to three keywords with medium competition and users, e.g., ”Employment rate germany” or ”Population statistics france”.
- **Longtail:** More descriptive phrases with few competitors and potential users, e.g., ”Employment rate germany women 2015” or ”Population by age in france 2016”.

This categorization will be used in our evaluation (see chapter 4).

2.4.3.2. Landingpages

In order to achieve the best possible ranking, every webpage should be optimized only for one specific set of keywords. Such an optimized webpage is called *Landingpage*. The weighting algorithms count the amount of occurrences and take special interest in some website elements. Therefore it is important to repeat the same set of keywords especially in the elements as listed in the following [Erl16, p. 485 ff.].

- `<title>`: The title-tag is important, since the title summarizes the content of a webpage and is usually displayed in the SERPs.
- `<description>`: The meta-tag for description is displayed in the SERPs and should attract the users attention. Keywords should appear here at least once.
- `<h1> - <h6>`: Headlines are important signals to search engines and therefore keywords should appear in them.
- `Text`: Text ought to be informative and as natural as possible. Keywords should appear on a regular basis without spaming.
- `URL`: Even though it is not critical, a keyword within the URL-path can affect the ranking in positive ways.

The repetition of keywords in the listed elements is a very common approach. But there are other factors, such as the internal link structure which are explained in the following.

2.4.3.3. Site Organization

The content of a webpage has to be machine-readable, in order to be understood and thus indexed by search engines. As an example, crawlers are not able to understand text hidden in pictures. They also struggle with Flash and complex Javascript content. From an SEO-perspective, content should be offered in HTML-based files [ESFS12, p. 182].

As previously outlined, search engines follow links on websites in order to find new websites. In consequence, the link-structure of a website should allow a crawler to reach every webpage which ought to be indexed. Common reasons why pages may not be reachable are the following [ESFS12, p. 183]:

- **Links in submission-required forms:** Search engines will not fill out forms or follow search boxes.
- **Links in hard-to-parse Javascript:** Links embedded in Javascript may not be followed or given low weight by search engines
- **Links in Flash, Java or plug-ins:** Links embedded in plug-ins are invisible to crawlers

In 2006 Google, Yahoo and MSN Search agreed on the *Sitemaps Protocol*[24], which is now supported by all major search engines. Using the protocol, webmasters can submit a list of all pages to be crawled. However, this does not guarantee that all

[24]http://www.sitemaps.org/, lass accessed 2016-02-06

pages will be indexed. It is an aid to the crawler which may have positive influence on a website's ranking. From a SEO-perspective, it is recommended to provide a Sitemap according to the protocol. Possible formats are XML, plain text or RSS [ESFS12, p. 184 ff.].

Another factor with ranking influence is the page speed, as officially stated by Google in 2010. Page speed refers to the overall loading time of a website. A good average value is around 500 milliseconds. Anything above can lead to reduced crawling intensity. Any value above 1,5 seconds can have direct influence on the ranking [Beu10][Erl16, p. 402].

2.4.3.4. Markup - Schema.org

Schema.org[25] was launched on June 2, 2011 by a powerful consortium consisting of Google, Bing, Yahoo! and later joined by Yandex. Its purpose is to establish and document an extensive vocabulary which is meant to be used by webmasters to add structured metadata to their content. This allows search engines to better understand provided content and display more information about websites within the result pages. It may lead to more clicks and a higher ranking in the long run [VE13].

By June 2015, Schema.org contained 635 classes and 894 unique properties. The ontology contains classes to describe the most popular types of web content, including personal profiles, movie reviews, business listings, product offers, and more [Mik15]. As for the time of this writing, it only contains a limited type set to properly describe datasets and its facts. It offers the possibility to markup title, source, publisher and release date of a dataset. A further description of the contained facts is not possible with the core vocabulary.

Schema.org can be assigned to On-Page Optimization techniques. It can be concluded, that its importance for SEO will further increase in the future, since all major search engines apply semantic technologies to their algorithms.

The following section introduces methods from the field of Off-Page Optimization.

2.4.4. Off-Page Optimization

The main goal of Off-Page Optimization is to create incoming links from other websites, a process called *Link-Building*. There are many different strategies to create them, starting with posting links in blog comments to advanced social media campaigns.

[25] http://schema.org/, last accessed 2016-02-07

In order to get good rankings, especially for high competitive keywords, Off-Page Optimization is very important. That being said, it is difficult to measure the effects of link building efforts, since search engines take a long time to respect them in their algorithms. Even though links to a website can lead to faster indexing, it usually takes month for a website to rank well [Mal09].

Due to the time delay, it is difficult to derive causal correlation from experimental studies. For that reason, the approach of this thesis is evaluated without any link-building efforts. Nevertheless, it is essential to understand the basics of Off-Page Optimization, in order to evaluate SEO campaigns. For that reason, the following describes the *PageRank*, some link-building strategies and pitfalls, which can lead to penalties.

2.4.4.1. PageRank

During the last years, Google manifested its position as market leader. Besides high performance and user friendly design, this is due to the good quality of retrieved results. The main reason for this is the usage of the *PageRank*-Algorithm, which has its name from Google founder Lawrence Page [Erl16, p. 308] and was fist published as an academic paper in 1999 (see [PBMW99]).

The PageRank-Concept is based on the assumption that a websites importance is higher, the more incoming links from other websites it has. Since the pure amount of incoming links can easily be manipulated, PageRank also measures the link-popularity. This means that links from important websites have a higher effect on the PageRank than links from unknown domains. In general, PageRank is a measure for the probability that a random surfer reaches a website. [Erl16, p. 310]

The PageRank can be iteratively computed for the whole Web and was formerly published with regular updates by Google [Erl16, p. 309]. Obviously, the algorithm has evolved and all major search engines implemented updates in order to detect unnatural link-structures. There are a couple of new measures which gained importance during the last years, such as the TrustRank who indicates how trustworthy and thus spam-free a site is [GGMP04]. Regardless of the concrete implementation, all algorithms have in common that they analyze how a website is embedded in the overall web, in order to identify its popularity and importance.

2.4.4.2. Link-Building

According to the presented algorithms, a websites ranking is improved by accumulating links from other domains, especially from trusted ones with high authority. There are several ways to achieve this. Some of the most common approaches are briefly described in the following [Erl16, p. 545 ff.].

- **Web-Directories:** It is possible to submit a website to a web directory, where it is categorized and published. Some of these directories, such as DMOZ[26] are still people reviewed and thus trusted by search engines.
- **Links in comments:** It is possible to set links in forums or blog-comments. Even though they are usually not followed by search engines, this is part of a link-building campaign.
- **Social signals:** It is said, that links from social networks such as Facebook ar Twitter do not strongly affect the ranking. Nevertheless, it looks natural and thus sends positive signals.
- **Paid content:** It is officially prohibited by Google to pay for links and once detected, this can lead to penalties. It is still a common approach to pay other websites, e.g. blogs to write an article and link back to one's website.
- **Link-worthy content:** The most natural way to generate links is to have interesting content, so that other people want to link it. It can be concluded that this is difficult to achieve for most websites with standard topics.

This listing conveys an overview of how a link-building campaign can be done. Search engines improve in detecting unnatural link-structures which result from over optimization. For a successful strategy, it is important to generate a natural appearing link-network. Otherwise, penalties may be applied by search engines.

2.4.4.3. Penalties

By over optimizing a website's appearance, the risk of being penalized by search engines increases. This can lead to a temporary banning from the index until the website has recovered. This strong penalty is usually only applied for so called *Black Hat*-techniques, which are SEO methods officially declared as unfair by search engines and thus penalized [Mal08]. But also normal techniques may lead to a bad ranking, such as duplicate or stolen content from other websites.

For the future it can be concluded, that search engines will further improve their ranking algorithm to detect unnatural websites. It is therefore in the best interest of every webmaster to generate unique content with value to potential visitors.

[26] http://www.dmoz.org, last accessed 2016-02-10

3. SEO4OLAP - The Approach

As already stated in the introduction, the motivation of this thesis is the following situation. Even though statistical linked datasets are published on the Web, single facts are not found by search engines. In this chapter, we present our approach to solve this issue called *SEO4OLAP*.

The schematic process of *SEO4OLAP* is illustrated by Figure 3.1. The system receives a dataset modeled by the RDF Data Cube Vocabulary (QB) as input. The dataset is then transformed by common OLAP Operations, i.e. dimensions are sliced, members are diced and measures are projected. The goal is to generate custom search engine optimized landingpages for all possible facts of the cube. Depending on the cube's size, this can lead to a huge amount of different pages (see section 3.3.1).

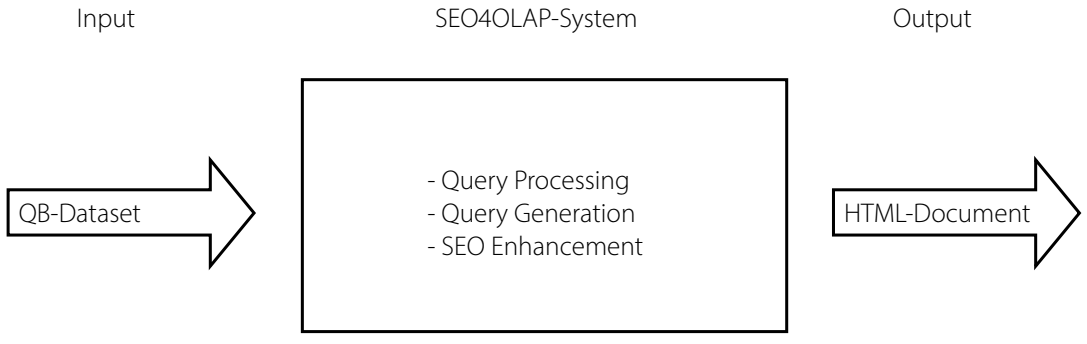

Figure 3.1.: Schematic process of *SEO4OLAP*

With every fact being represented by a landingpage, search engines are able to index them and thus retrieve results for very specific keywords. As an example, a dataset

published by *Eurostat* about the Employment rate in european countries may be found by search engines with the keywords "dataset employment rate europe". A search for a concrete fact, such as "employment rate in germany 2014" will not lead to the source of this dataset directly, since it is not optimized for these keywords. *SEO4OLAP* creates such landingpages and thus allows search engines to retrieve results.

From a conceptual view, a landingpage is a search engine optimized HTML representation of an OLAP request. In order to generate them for all possible views of a cube, we propose a system based on two main parts as described in the following.

- The **Query Processor** receives OLAP queries, computes results and displays them as search engine optimized HTML-pages. Since those are accessed via a browser, OLAP queries are expressed as HTTP-requests, e.g. as a link. In order to achieve this, we propose a query model as described in section 3.2.
- The **Query Generator** has to compute all possible requests (i.e. links) and make them accessible to search engines. This can either be done by a link structure or a sitemap, listing all links and thus all possible OLAP queries.

In the following, the system architecture of the Query Processor is presented with its main components. In section 3.2, a new Query Model for OLAP is introduced which allows its representation as HTTP-requests. In section 3.3 the Query Generation as second main part of *SEO4OLAP* is described. It explains how all possible facts of a datacube can be computed and points out the involved computation complexity. In the last section 3.4, possible methods of SEO are analyzed. The concrete implementation is presented and evaluated in the following chapter 4 *Evaluation*.

3.1. Query Processing

A generic system architecture for query processing consists of three main components: HTTP Interface, OLAP Engine and Result Transformation. The query processing workflow is illustrated by Figure 3.2. The components are further described in the following.

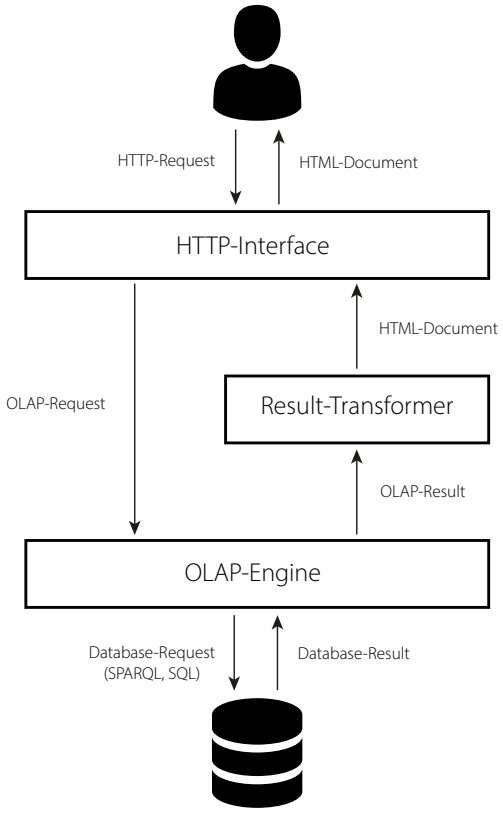

Figure 3.2.: Query processing of *SEO4OLAP*

HTTP Interface

OLAP queries are submitted to the system via HTTP which is the standard protocol of the Web. It allows for content negotiation, meaning that clients can request a specific data format. For our use case of SEO, this would preferably be HTML. But one could also think of other clients, such as mobile applications, requesting machine-readable representations, e.g. XML or JSON. Thereby, the system would interact as a generic endpoint for OLAP processing.

A request can be expressed as HTTP methods GET or POST. The concrete format depends on the API specification. In our implementation as presented in chapter 4, we use GET requests. Since clean and readable links are a ranking factor for search engines, we developed a URL-Schema for expressing GET requests (Section 3.2).

The interface receives incoming client requests and sends them to the *OLAP-Engine*. After the query has been processed, a result in the specified format is returned to the client.

OLAP-Engine

The OLAP-engine receives incoming requests from the HTTP interface. These requests first have to be transformed into a *Logical OLAP Query Plan* (see section 3.2.3) and are then processed into native database queries. Since in this thesis, we focus on statistical Linked Data, the OLAP-engine interacts with an RDF triple store via SPARQL queries. From a generic point of view, the engine could address any database.

Result Transformer

Once the OLAP result has been generated by the engine, it has to be enhanced with keywords and transformed into the specified format. In the normal case of HTML, the document is generated with reoccurring keywords in headline, title, description and other HTML-tags. Once the document has been generated, it is returned to the requesting client.

3.2. Query Model

There are various ways to define OLAP-queries and express them with query languages. As an example, MDX allows to specifically define how OLAP-results should be displayed in a pivot table. For our use case, this is too complex. Search engines prefer clean and readable URLs for their ranking. Therefore we developed a new query model which is presented in the following. First, the parameters are described and afterwards introduced to our URL-scheme.

3.2.1. Subcube Queries

We adapt the concept of subcube queries from Kämpgen et al. [KOH12] and define it as per Definition 6.

Definition 6 (Subcube Query) *A subcube query on a certain cube is represented as a tuple (Measures2Project, Dimensions2Keep, Members2Dice), with Measures2Project consisting of identifiers of measures to be projected, Dimensions2Keep consisting of identifiers of dimensions to be kept and Members2Dice consisting of identifiers of members to be diced. Dimensions that are not represented by a member in Members2Dice or part of Dimensions2Keep, are sliced.*

We propose this model because we believe it is a good fit for our use case of SEO. The main advantage is that only relevant parameters have to be set. This means

that the query directly defines which measures, dimensions and members should be retrieved. In comparison to Kämpgen et al., less parameters are needed.

As an example, consider a query on our employment cube asking for the absolute employment number and the employment rate of germany per year, disregarding the gender. A subcube query for this request would be the following:

$$(employment\_absolute, employment\_rate, date, germany)$$

We assume, that the dimension *Date* only has members on the level year. The dimension *Gender* is sliced, since neither men or women are defined as Member2Dice nor is it part of Dimensions2Keep.

3.2.2. HTTP Requests and URL Scheme

A subcube query as defined per Definition 6 can be submitted via HTTP by setting the cube identifier and the three parameters *Measures2Project*, *Dimensions2Keep* and *Members2Dice*. In order to create URLs, a GET-request is the most suitable approach. Depending on the application and the use case, the usage of POST may be a feasible alternative.

A typical HTTP GET request for our scenario has the following structure:

```
http://baseUri/endpoint?cube=id1&measure=id2&dimension=id3&member=id4
```

An API based on this scheme would work just fine. The problem is that search engines prefer clean and readable URLs consisting of keywords. Therefore, we developed a URL-scheme which enables the same functionality but has a clean appearance supposing that keywords are used as identifiers. It is presented in the following:

```
http://baseUri/cubeId/pattern/id1/id2/...
```

The pattern consists of three digits, defining the amount of used parameters per group. The following identifiers are slash-separated and in the order of the pattern. As an example, the pattern *122* means that the first identifier is a measure, the following two are dimensions and the last two are members. In the following some example URLs are presented for better understanding.

- `http://example.org/employment/211/absolute/rate/date/germany`:
  A URL for our previous example.

- `http://example.org/employment/112/rate/date/women/france`: A URL asking for the employment rate of women in France per year.
- `http://example.org/population/111/absolute/date/poland`: A URL asking for population numbers in Poland per year. This is a different datacube.

It should be noted that the order of identifiers within a group does not alter the query. In consequence, different URLs can lead to the same page. As an example, the following two URLs have the same content.

- `http://example.org/employment/102/rate/women/germany`
- `http://example.org/employment/102/rate/germany/women`

Duplicate content is penalized by search engines and should therefore be avoided, e.g. by ordering identifiers alphabetically. This has to be regarded for a concrete implementation.

3.2.3. Query Transformation

In order to be processed by an OLAP-engine, subcube queries have to be transformed into a Logical OLAP Query Plan. We propose the following procedure to achieve this.

1. **Input**: M, a set of measures2project; D, a set of dimensions2keep; V, a set of members2dice
2. **Set**: $P = \emptyset$, an empty queue, the Logical OLAP Query Plan.
3. Add `BaseCube` to P.
4. For every measure in M, add `Projection` to P.
5. For every dimension, that is not included in D or has no corresponding member in V, add `Slice` to P.
6. For every member in V, add `Dice` to P.
7. **Return** P

3.3. Query Generation

The previous sections explained, how OLAP-queries can be represented by URLs and how they are processed by the system to retrieve results. This section introduces concepts to make those links available to users and search engines. The natural way is to provide a link structure on the website which allows a crawler to reach every

available page. This is described in section 3.3.2. In order to help search engines crawl a website, a sitemap listing all available pages can be provided. In section 3.3.3 we explain how this can be done. Since big datasets can lead to huge amounts of potential webpages, we first address the issue of complexity management.

3.3.1. Complexity Management

A cube's size and thus the number of possible views is exponentially dependent on the amount of dimensions. The maximum amount of possible views can be computed with formula 3.1. Every dimension has d_i possible members plus 2 extra values: the implicit *ALL*-member, which is a slice and the implicit *Zero*-Member, when no value is selected. m is the amount of measures, n is the amount of dimensions. The meaning of d_i, m and n apply for all formulas in this chapter.

$$MaxViews = m \times \prod_{i=1}^{n}(d_i + 2) \tag{3.1}$$

Formula 3.1 underlies the following restrictions:

- Only one measure is displayed per view.
- A dimension can be set to one member, be sliced (All-Member) or not set (Zero-Member). Multiple members of the same dimension are not regarded.
- Implicit aggregated members of higher levels are neglected. Only explicit members are regarded.

Due to the exponential growth of the problem, the computation effort to generate separate webpages for every possible view can be enormous. From a SEO and a user experience perspective, it is questionable whether all possible views are necessary. Therefore, we propose two restrictions to our model:

- The **Dice Dimensionality** (DiceDim) restricts the maximum amount of dimensions which can be diced in one query.
- The **Amount of free Dimensions** (FreeDim) defines the maximum amount of dimensions that are neither diced or sliced, thus free.

Depending on those two restrictions, the total amount of possible views can be computed by formula 3.2.

$$Views_{DiceDim,FreeDim} = m \times \sum_{d=0}^{DiceDim} \Big(FreeDimFactor_{FreeDim} \times DiceDimFactor \Big) \tag{3.2}$$

with

$$FreeDimFactor_{FreeDim} = \begin{cases} \sum_{s=0}^{FreeDim} \frac{(n-d)!}{s!\,(n-d-s)!} & \text{with } \frac{(n-d)!}{s!\,(n-d-s)!} = 0 \\ & \text{for } (n-d-s) < 0 \end{cases} \tag{3.3}$$

and

$$DiceDimFactor = \begin{cases} 1 & \text{if DiceDim} = 0 \\ \sum_{i_{n-(d-1)}=1}^{n-(d-1)} d_{i_{n-(d-1)}} \times \cdots \times & \\ \sum_{i_{n-(d-1)+j}=i_{n-(d-1)+(j-1)}+1}^{n-(d-1)+j} d_{i_{n-(d-1)+j}} \times \cdots \times & \text{if DiceDim} > 0 \\ \sum_{i_n=i_{n-1}+1}^{n} d_{i_n} & \end{cases} \tag{3.4}$$

Die *DiceDimFactor* computes the amount of possible dice combinations for a given Dice Dimensionality. Every possible combination can be shown with a different set of free dimensions. Therefore, it is multiplied by the *FreeDimFactor*.

In order to convey a better understanding of how formula 3.2 has to be applied, we present it with both parameters *DiceDim* and *FreeDim* set to 2 in formula 3.5. The formula applies for all $n \geq 4$.

$$\begin{aligned} Views_{2,2} = m \times \Bigg(& \Big((\frac{(n-2)(n-3)}{2} + n - 1) \times \sum_{i=1}^{n-1} d_i \sum_{j=i+1}^{n} d_j \Big) \\ & \Big((\frac{(n-1)(n-2)}{2} + n) \times \sum_{i=1}^{n} d_i \Big) \\ & \Big((\frac{n(n-1)}{2} + n + 1) \Big) \Bigg) \end{aligned} \tag{3.5}$$

As we show in chapter 4.2, the maximum amount of possible views is massively decreased by the restrictions *DiceDim* and *FreeDim* for high dimensional datasets. In our URL-scheme, those restrictions can be regarded by defining a maximum pattern. The maximum pattern of formula 3.5 is "X22". It means that X measures2project, two dimensions2keep and two members2dice are possible.

3.3.2. Link Structure

The link structure of a website should allow a crawler to reach every page which ought to be indexed. One can think of various ways to achieve this, e.g. by categorizing pages per dimension, measure or member. We propose a structure as illustrated by Figure 3.3.

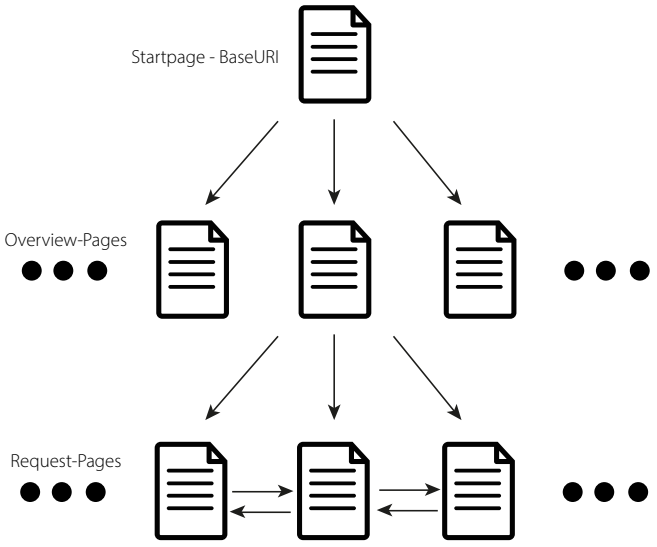

Figure 3.3.: Link structure of *SEO4OLAP*

The entry page at the baseUri is the start for a crawler. It can display general content about the project, but most importantly, it links to all available datasets. On the second level, we propose to display overview pages per dataset. From there, links to concrete facts are displayed. They can be categorized or ordered in any thinkable way. On the third level, the concrete facts, i.e. our landingpages for OLAP requests are shown. We propose, that every page links to near neighbors. A near neighbor is an OLAP-requests with only one parameter changed, e.g. a different member, dimension or measure. By this, it is guaranteed that every possible request can be reached.

3.3.3. Sitemap

A sitemap is a list of all available links of a webpage according to the *Sitemaps Protocol*[1]. It is an aid for a crawler to understand a website's structure and should therefore be offered [ESFS12, p. 184 ff.].

In order to generate a list of all links and thus of all OLAP-requests, we propose a procedure as listed in the following. It is based on our URL-scheme and regards the

[1]http://www.sitemaps.org/, lass accessed 2016-02-06

complexity restrictions as outlined in section 3.3.1. In order to reduce the maximum amount of possible views with formula 3.2, a maximum pattern is defined. As recap, the pattern of a URL defines the amount of used measures, dimensions and members. We propose the pattern "122" as default. It means, that one measure and a maximum of two dimensions and two members are used per query.

1. **Input**: M, a set of measures; D, a set of dimensions; V, a set of members; `MaxPattern`
2. **Set**: $L = \emptyset$, the List of Links; `CurrentPattern` = 100, `CurrentPath` = "/"
3. **Iterate** through either M, D or V, depending on the `CurrentPattern`. If restrictions do not apply, add $id \in M \cup D \cup V$ to `CurrentPath`, increment `CurrentPattern` and recursively start again. If `CurrentPattern` = `MaxPattern`, add `CurrentPath` to L.
4. **Cleanup** L by removing duplicates.
5. **Return** L

This listing conveys a general understanding of the procedure. A concrete algorithm in Java can be found in the appendix E.

3.4. SEO Enhancement

The previous sections explained the query model, the query processing architecture and how links are generated and made accessible for search engines. This last section describes how OLAP-results can be enhanced for SEO. Therefore, we first analyze how keywords are created and afterwards point out where they can be put within an HTML document.

We know from chapter 2.4.3.2 that a set of keywords is assigned to a landingpage. They are repeated several times in various HTML-tags, in order to be noticed as the most relevant content of a page. In our system architecture as illustrated by Figure 3.2, this is done by the Result Transformer which converts an OLAP-result into an SEO enhanced HTML document.

An OLAP-result is best described by the measure, displayed dimensions and filtered values. These are defined within an OLAP-request as *Measures2Project*, *Dimensions2Keep* and *Members2Dice*, thus are directly available. If a label is assigned to each request parameter within the RDF dataset, these labels can be used to define the set of keywords. If labels are not available, one can think of two possible solutions:

- Missing RDF statements can be added to the source.

- Labels can be defined in a separate file.

From a SEO perspective, it is important to use keywords that users would search for. As an example, both expressions *Female* and *Women* would fit as a label for the female value of the dimension *Gender*. But probably, more users search for *Women* and therefore this term is a better label. Because of this, we propose to check labels and alter them if necessary in order to attract more users.

Once the set of keywords is defined for a request and thus a landingpage, they have to be repeated several times within the document. The most important HTML tags for keyword placement were described in section 2.4.3.2. Since one cannot simply list keywords in a headline, we propose the following structure to create a sentence:

$$\text{(measure)}^{and} \text{ per (dimension)}^{and} \text{ of (member)}^{and}$$

Keywords of the same type are combined by an *and*. The following shows an example sentence, which could be used as headline:

Employment rate per gender and year of Germany

The repetition of keywords is the main method of SEO that can be applied directly to a webpage. Semantic markup such as Schema.org should be used, but the possibilities to markup facts of a dataset are limited. Other On-Page techniques are important, but regard the structure of the website. Therefore, our linkstructure guarantees that all webpages are accessible by users and crawlers. The sitemap helps crawlers by listing all possible requests of the site. The system should also guarantee a site speed of less than 1,5 seconds in order to avoid penalties by search engine providers. If a dynamic computation takes longer, the precomputation of all views may be a solution.

Off-Page optimization techniques, which strengthen the PageRank or TrustRank are not regarded within our approach, since they can not be influenced by the system directly and have to be done manually.

4. Evaluation

The previous chapter explained the conceptual model of our approach *SEO4OLAP*. In order to evaluate it, we implemented a concrete system and published two datasets from *Eurostat*. In this chapter, we present our findings. We analyze how our generated webpages rank in search engine results in comparison to the original website of *Eurostat*. We further evaluate our algorithm for generating OLAP-queries by comparing the amount of generated queries with our mathematical model of section 3.3.1. We also present challenges regarding real world dataset modeling.

The following section describes the characteristics of the implementation. In section 4.2 the involved mathematical complexity of our approach is evaluated. Section 4.3 presents the published datasets and describes our evaluation method and its findings.

4.1. Characteristics of the Implementation

We implemented a *SEO4OLAP* system in Java and deployed it on a Google App Engine. The source code is published as Open Source and can be used for further research (see appendix A). During the implementation, we were facing some challenges due to the modeling characteristics of real world datasets. In the following, we give a brief overview of the implementation and point out the challenges and the corresponding solutions.

4.1.1. Technical Overview

We implemented the system with a standard Java Servlet and JSP architecture. The system is deployed on a Google App Engine and therefore uses Google specific libraries for data storing, cache and task queuing. The packages with its most important classes are illustrated in Figure 4.1 and explained in the following.

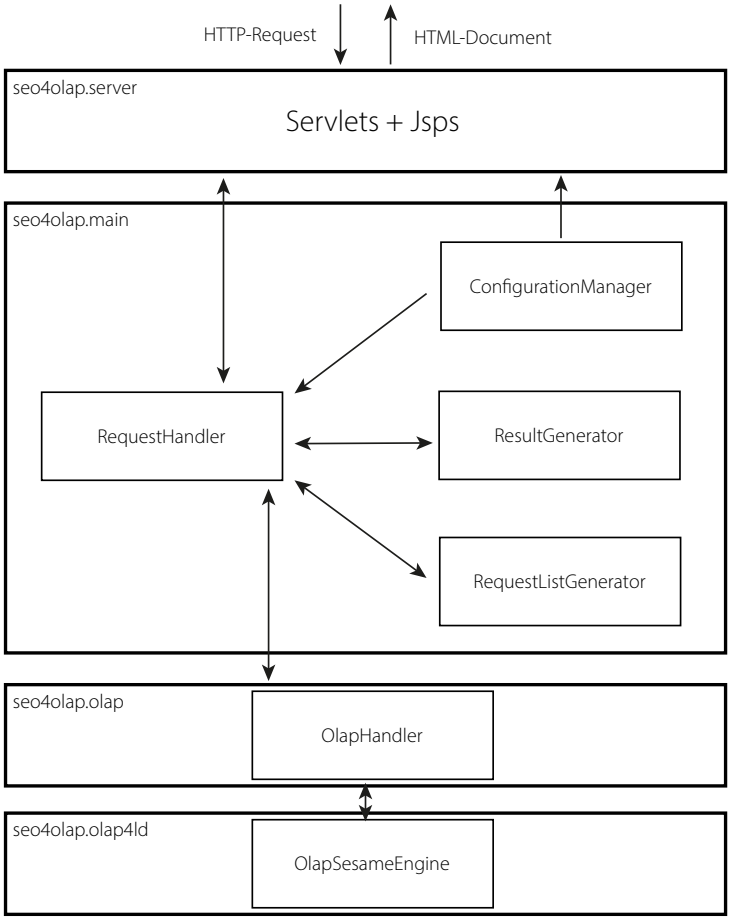

Figure 4.1.: Packages and main classes of *SEO4OLAP* implementation

Server package

The server package contains various public and private servlets which receive HTTP-requests and mostly respond with JSPs. Requests are forwarded to the RequestHandler for further evaluation. The following three servlets are most relevant:

- The **ResultServlet** receives OLAP requests per HTTP and returns results.
- The **SitemapServlet** generates the sitemap, thus triggers the RequestList-Generator.
- The **InitServlet** initializes a dataset by filling a task queue with all possible requests in order to precompute results.

Main package

The main package contains the logic for handling requests, i.e. storing, result transformation or link generation. The main classes are described in the following:

- The **RequestHandler** handles the entire interaction with servlets, especially incoming requests. It transforms HTTP-requests into OLAP-requests and forwards them to the OlapHandler. OLAP-results are then forwarded to the ResultGenerator and stored in a database.
- The **ResultGenerator** enhances plain OLAP-results by adding necessary information to be displayed by JSPs, e.g. title, headlines, description and other SEO specific information.
- The **ConfigurationManager** is the interface to a configuration file, which stores application and dataset specific information. The usage of it is further explained in section 4.1.4.
- The **RequestListGenerator** contains the algorithm which computes all possible queries for a dataset.

OLAP package

This package contains classes for handling OLAP-requests and their results. The OlapHandler is the main class of the package. It receives OLAP requests and transform them into Logical OLAP Query Plans, which are executed by the OlapSesameEngine.

Olap4ld package

The Olap4ld package is a library developed by Kämpgen and Harth [KH14], which handles OLAP-operations on datasets modeled in the RDF Data Cube Vocabulary. The included *EmbeddedSesameEngine* is an OLAP-engine with an integrated Sesame triple store. The engine receives a URI of a dataset, loads it into the triple store and allows to perform metadata and OLAP-queries.

We extended the *EmbeddedSesameEngine* with functionality to retrieve labels and other information from the dataset. We also optimized the computation time by storing the results of often used metadata requests, such as `getDimensions()` or `getMeasures()`. Our *OlapSesameEngine* is therefore an extension to the original library. Its main task is to execute Logical OLAP Query Plans generated by the OlapHandler, as well as providing information about the dataset which is used for SEO enhancement.

4.1.2. Datastore

Our OlapSesameEngine loads datasets into an in-memory triple store. On startup of a SEO4OLAP-instance, the dataset has to be loaded and metadata queries have to be performed before OLAP-requests are possible. Depending on the computation

power of the instance and the size of the dataset, this can take up to several minutes. Once the triple store is filled with data, OLAP-requests take several seconds of computation time.

In order to guarantee a sitespeed of less than 1.5 seconds, we decided to precompute all requests and store results in a database. This supports the user experience and avoids penalties by search engine providers. An initialization process fills a Google App Engine specific task queue with all requests, so that every request is computed and stored. In consequence, updates in the data source are only displayed after reinitializing the dataset. For our published datasets as described in section 4.3.1, updates are only expected occasionally.

4.1.3. Dataset Challenges

For our evaluation, we focused on datasets from *Eurostat* which were wrapped as Linked Data by *Estatwrap*. We discovered some challenges that have to be addressed for automated query generation. This is mainly due to the fact that real-world datasets differ from the modeling approaches as intended by the Data Cube Vocabulary (QB) or SDMX. The major issues are presented in the following.

- **Measure-Dimensions**: In many datasets, e.g. European employment statistics[1], a dimension is used to specify the indicator (e.g. employment rate or absolute employment number) of the measure. This means that a measure is modeled as a dimension, a *Measure-Dimension*. We understand that this may be due to a transformation process, e.g. by converting a table into SDMX, and therefore a convenient solution for data publishers. Nevertheless, from the conceptual point of view of a Multidimensional Data Model, this is not intended and causes problems for OLAP-operations.

  First, instead of a Projection on a measure, a Dice on the *Measure-Dimension* has to be performed. Second, the *Measure-Dimension* can not be sliced, since an aggregation would cause implausible values. Since QB offers the possibility to explicitly declare such *Measure-Dimensions*, we conclude that this is a common practice in real-world datasets. Nevertheless, the Linked Data from *Estatwrap* does not make use of this QB-feature.

- **Slice-Members**: Some dimensions contain members which are, from a conceptual point of view, aggregations to a higher level. As an example, at *Eurostat* we often find three values for the dimension *Gender*: *Female*, *Male* and *Total*. In consequence, a Slice on *Gender* would aggregate all three members

[1] http://appsso.eurostat.ec.europa.eu/nui/show.do?wai=true&dataset=lfsi_emp_a, last accessed 2016-02-04

and thus lead to wrong values. The member *Total* represents the correct values. Therefore, we define such members as *Slice-Members*. A *Slice-Member* is a member of a dimension, which represents the aggregation of the dimension.

We understand that publishers of datasets may have good reasons for this. Nevertheless, this is a challenge for automated OLAP-query generation, since such dimensions can not be sliced. Instead of a Slice, a Dice on the *Slice-Member* has to be performed in order to create correct values.

- **Missing measure units**: For *Estatwrap* data, we discovered that the units of some measures are missing. As an example, the values for absolute employment count[2] have to be multiplied by a thousand. This information is given by *Eurostat*, but not clearly assigned to a specific measure. Therefore, we understand the difficulty for an automated Linked Data wrapper, such as *Estatwrap*, to include such information. Nevertheless, the provided information of *Estatwrap* misses this information.

In order to handle these challenges, we decided to implement a dataset configuration file which is described in the following.

4.1.4. Dataset Configuration

In order to react to the modeling challenges of section 4.1.3, we implemented our algorithm based on a configuration file. For every dataset, the included measures, dimensions and members have to be defined. If labels are missing in the RDF, they can be added in the configuration file. Most importantly, dimensions can be declared as *Measure-Dimensions* and dimensions can declare a *Slice-Member*. By this, users of our *SEO4OLAP*-implementation can react in a flexible way to challenging data models in order to ensure the correctness of displayed data.

We are aware that we add information to the original data source by using a configuration file. The main intention is to ensure the correctness of data. If datasets from *Eurostat* were correctly modeled, this would not have been necessary. Since the information could have been provided in the RDF, we do not see any problems regarding our SEO evaluation of section 4.3.

4.1.5. Limitations

During the implementation, we were facing a programming specific problem which we were not able to solve. We did not manage to dice members that are modeled as literals. We were only able to dice members modeled as URI. At this point, we

[2] http://appsso.eurostat.ec.europa.eu/nui/show.do?wai=true&dataset=lfsi_emp_a, last accessed 2016-02-04

are not sure whether this is a limitation of the library *Olap4ld* or if we misused the library.

Our approach to solve this was to automatically convert such members into a URI by means of SPARQL Update Queries. In the following, a possible query for this task is presented:

```
INSERT {
        ?DIMENSION_UNIQUE_NAME rdfs:range skos:Concept.
        ?DIMENSION_UNIQUE_NAME qb:codeList ?newCodeList.
        ?newCodeList skos:hasTopConcept ?MEMBER_UNIQUE_NAME.

        ?obs ?DIMENSION_UNIQUE_NAME ?MEMBER_UNIQUE_NAME.
        ?MEMBER_UNIQUE_NAME a skos:Concept;
        skos:inScheme ?newCodeList;
        rdfs:label ?MEMBER_LABEL.
}

WHERE {
        ?CUBE_NAME qb:structure ?dsd.
        ?dsd qb:component ?compSpec.
        ?compSpec qb:dimension ?DIMENSION_UNIQUE_NAME.
        ?DIMENSION_UNIQUE_NAME rdfs:range ?range FILTER
        (?range != skos:Concept).
        ?obs qb:dataSet ?CUBE_NAME.
        ?obs ?DIMENSION_UNIQUE_NAME ?MEMBER_LABEL FILTER
        (ISLITERAL(?MEMBER_LABEL)).

        BIND(URI(CONCAT(STR(?DIMENSION_UNIQUE_NAME),
        CONCAT("XXX", STR(?MEMBER_LABEL)))) as ?MEMBER_UNIQUE_NAME).
        BIND(URI(CONCAT(STR(?range),"XXXNewCodelist"))
        as ?newCodeList).
}
```

Listing 4.1: SPARQL Update query for degenerated members. Prefixes are not shown

The Sesame library uses threads for SPARQL Update queries. Since the Google App Engine prohibits the usage of Java threads, this approach was not possible within the Google App Engine. Nevertheless, the approach should work, e.g. in a Tomcat environment.

In consequence, we are not able to dice members of the dimension *Date*, because these are modeled as literals in the RDF. This is unfortunate, since views such

as "Employment rate in Germany in 2014" are not possible. We are only able to generate views showing the entire *Date* dimension or a sliced version.

This is another restriction to our query generation algorithm and results in less possible views. Since the overall amount of generated views is still sufficient, this restriction does not affect our SEO evaluation.

4.2. Complexity Evaluation

In chapter 3.3.1, we presented two formulas which calculate the number of possible views of a dataset. Formula 3.1 on page 34 computes the maximum number of possible views, the upper bound. Formula 3.2 computes the number of possible views, depending on the restrictions *Dice Dimensionality* and *Max amount of Free Dimensions*. In the following, we present how these numbers evolve with added dimensionality in order to visualize the involved computing complexity of our approach.

We implemented both formulas in Java (see appendix D: `ComplexityCalculator`). In order to show the effect of both restrictions, we calculated the number of views depending on the amount of dimensions n. In the following, we present the results in two tables with corresponding diagrams. For both calculations, we set the amount of members per dimension to ten and the amount of measures to one.

Table 4.1 and Figure 4.2 show the restricted amount of views depending on the *Dice Dimensionality*. The maximum amount of free dimensions is fixed to two.

Dimensions n	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Max / Upper Bound	12	144	1.728	20.736	248.832	2.985.984	35.831.808	429.981.696	5.159.780.352	61.917.364.224
1-Dim Dices	12	44	127	291	566	982	1.569	2.357	3.376	4.656
2-Dim Dices	-	144	727	2.691	7.566	17.482	35.169	63.957	107.776	171.156
3-Dim Dices	-	-	1.727	10.691	47.566	157.482	420.169	959.957	1.955.776	3.651.156
4-Dim Dices	-	-	-	20.691	147.566	757.482	2.870.169	8.659.957	22.115.776	49.851.156
5-Dim Dices	-	-	-	-	247.566	1.957.482	11.270.169	47.859.957	160.715.776	453.051.156
6-Dim Dices	-	-	-	-	-	2.957.482	25.270.169	159.859.957	748.715.776	2.763.051.156
7-Dim Dices	-	-	-	-	-	-	35.270.169	319.859.957	2.188.715.776	11.163.051.156
8-Dim Dices	-	-	-	-	-	-	-	419.859.957	3.988.715.776	29.163.051.156
9-Dim Dices	-	-	-	-	-	-	-	-	4.988.715.776	49.163.051.156
10-Dim Dices	-	-	-	-	-	-	-	-	-	59.163.051.156

Table 4.1.: Amount of possible views per Dice Dimensionality. Amount of free dimensions = 2; Members per dimensions = 10; Amount of measures = 1

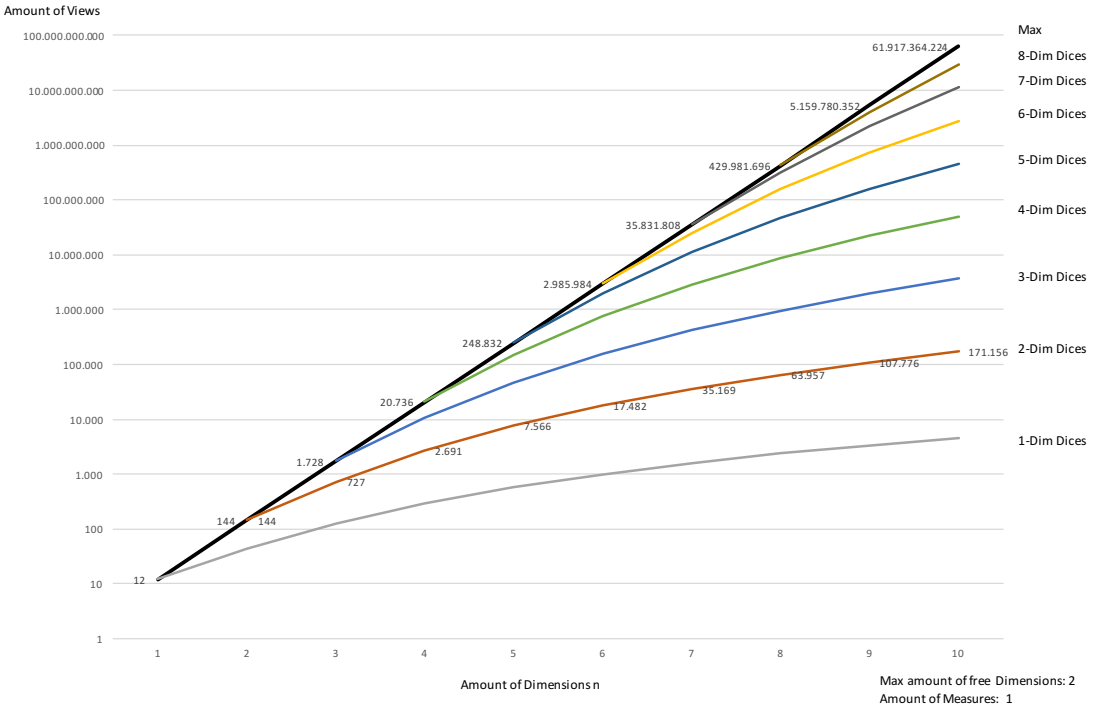

Figure 4.2.: Amount of possible views, depending on Dice Dimensionality

Table 4.2 and Figure 4.3 show the restricted amount of views depending on the *Amount of free Dimensions*. The Dice Dimensionality is fixed to two.

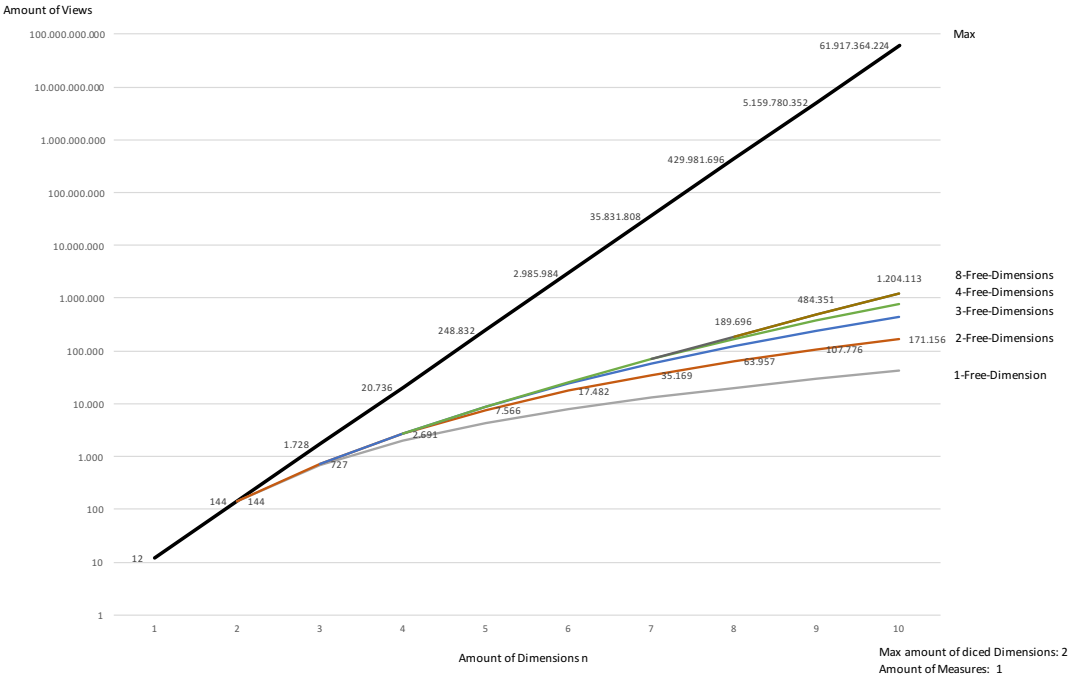

Figure 4.3.: Amount of possible views, depending on amount of free dimensions

Dimensions n	1	2	3	4	5	6	7	8	9	10
Max / Upper Bound	12	144	1.728	20.736	248.832	2.985.984	35.831.808	429.981.696	5.159.780.352	61.917.364.224
1 free Dimension	12	143	694	1.965	4.256	7.867	13.098	20.249	29.620	41.511
2 free Dimensions	-	144	727	2.691	7.566	17.482	35.169	63.957	107.776	171.156
3 free Dimensions	-	-	728	2.735	8.776	24.102	57.604	122.813	238.900	431.676
4 free Dimensions	-	-	-	2.736	8.831	25.917	69.189	167.683	371.326	759.486
5 free Dimensions	-	-	-	-	8.832	25.983	71.730	186.219	452.092	1.024.338
6 free Dimensions	-	-	-	-	-	25.984	71.807	189.607	479.896	1.158.948
7 free Dimensions	-	-	-	-	-	-	71.808	189.695	484.252	1.198.668
8 free Dimensions	-	-	-	-	-	-	-	189.696	484.351	1.204.113
9 free Dimensions	-	-	-	-	-	-	-	-	484.352	1.204.223
10 free Dimensions	-	-	-	-	-	-	-	-	-	1.204.224

Table 4.2.: Amount of possible views depending on free dimensions. Dice Dimensionality = 2; Members per dimensions = 10; Amount of measures = 1

By setting the *Dice Dimensionality* and the *Max amount of Free Dimensions* to n, we calculate the same result for both formulas 3.1 and 3.2 (see appendix B). In consequence, we conclude the correctness of our formulas, as well as of our implementation.

Figure 4.2 illustrates that the *Dice Dimensionality* has an exponential impact on the amount of possible views. In contrast, the effect of *Max amount of Free Dimensions* is much weaker, since it influences the *Dice Dimensionality* only as a factor. Therefore, in order to drastically decrease the amount of views, the *Dice Dimensionality* should be set to a low value. For our *SEO4OLAP* implementation, we set the *Dice Dimensionality* to two. Since normal tables can only show two dimensions, we also set the *Max amount of Free Dimensions* to two.

4.3. SEO Evaluation

This section evaluates our *SEO4OLAP* approach from an SEO perspective. We want to analyze how well datasets published on the Web by *SEO4OLAP* are found by search engines. Therefore, we published two datasets from *Eurostat* with our *SEO4OLAP*-implementation (see section 4.1). We measure the search engine rankings for a predefined set of keywords and compare these with the Eurostat website as benchmark.

In the following section 4.3.1, we describe the two datasets and how they were published. Section 4.3.2 describes the benchmark pages from Eurostat, to which we compare our own results. Section 4.3.3 describes our evaluation method and presents the results. The closing section 4.3.4 evaluates hypotheses against our results.

4.3.1. Dataset Description

We chose two datasets from *Eurostat* which are available as Linked Data from *Estatwrap*. Their characteristics are described in the following.

Employment Statistics

This dataset contains facts about employment statistics, especially in European countries. The data is grouped by three dimensions: year, gender and country. It contains multiple measures, such as the absolute employment number and the employment rate for various age groups.

The following table shows the source URL of this dataset.

Eurostat ID	lfsi_emp_a
Eurostat Source URL	http://ec.europa.eu/eurostat/en/web/products-datasets/-/lfsi_emp_a
Estatwrap Source URL	http://estatwrap.ontologycentral.com/id/lfsi_emp_a

Table 4.3.: Employment statistics dataset source

The dimensions and measures of this dataset are presented and described in table 4.4.

	Label	Unique Name	Member Count	Description
Dimensions	Date	http://purl.org/dc/terms/date	8	Contains years from 2007-2014
	sex	http://ontologycentral.com/2009/01/eurostat/ns#sex	3	Female, male and total
	geography	http://ontologycentral.com/2009/01/eurostat/ns#geo	41	Different countries, mostly from Europe, as well as aggregations such as European Union or Euro Area.
	null	http://ontologycentral.com/2009/01/eurostat/ns#indic_em	9	This is a measure dimension. It contains different measures, such as the employment rate and the absolute employment number by different age groups
Measures	Observation	http://purl.org/linked-data/sdmx/2009/measure#obsValue	-	This is just the observation value. The measure is defined by the measure dimension

Table 4.4.: Dimensions and Measures of lfsi_emp_a

We chose this dataset for our SEO evaluation due to the following reasons:

- **Sufficient search volume**: The *Google Keyword Planner*[3] reveals a monthly search volume of around 50.000 as sum of all relevant keywords.
- **Low competition**: The *Google Keyword Planner* values the competition for the dataset specific keywords as low. This is confirmed by our personal impression.

Gross Domestic Product

This dataset provides information about the gross domestic products per year in various countries, especially in the European Union.

[3]https://adwords.google.de/keywordplanner, last accessed 2016-03-13

The following table shows the source URL of this dataset:

Eurostat ID	tec00001
Eurostat Source URL	http://ec.europa.eu/eurostat/web/products-datasets/-/tec00001
Estatwrap Source URL	http://estatwrap.ontologycentral.com/id/tec00001

Table 4.5.: Employment statistics dataset source

The dimensions and measures of this dataset are presented and described in table 4.6.

	Label	Unique Name	Member Count	Description
Dimensions	Date	http://purl.org/dc/terms/date	8	Contains years from 2008-2015
	null	http://ontologycentral.com/2009/01/eurostat/ns#na_item	1	This is a measure dimension with only one member: "Gross domestic product at market prices"
	geography	http://ontologycentral.com/2009/01/eurostat/ns#geo	41	Different countries, mostly from Europe, as well as aggregations such as European Union or Euro Area.
	unit	http://ontologycentral.com/2009/01/eurostat/ns#unit	3	This is a measure dimension with three members: "Current prices, euro per capita", "Current prices, million euro", "Current prices, million Purchasing Power Standards"
Measures	Observation	http://purl.org/linked-data/sdmx/2009/measure#obsValue	-	This is just the observation value. The measure is defined by the measure dimension

Table 4.6.: Dimensions and Measures of tec00001

We chose this dataset for our SEO evaluation due to the following reasons:

- **Sufficient search volume**: The *Google Keyword Planner* reveals a monthly search volume of around 100.000 as sum of all relevant keywords.
- **Medium competition**: The *Google Keyword Planner* values the competition for the dataset specific keywords as low. Nevertheless, we value it as medium, since the Google Knowledge Graph provides direct results (e.g. by searching "gdp germany").

Both datasets have a sufficiently high search volume which is why we value the information as relevant for many people. In contrast to the employment statistic dataset, this dataset has a higher competition. We selected this dataset for evaluation in order to analyze, if this influences the ranking.

Dataset Configuration

As mentioned in section 4.1.4, it is necessary for our implementation to configure datasets in order to react to *Measure-Dimensions* and *Slice Members*.

The following table 4.7 shows how many measures and members per dimension were used per dataset in the configuration file. As mentioned in section 4.1.5, we were not able to handle members modeled as RDF literals. Therefore, there are zero members in the *year* dimension. A slice-member does not count as member, since it represents a slice. Since some measures were very similar to each other, we only selected two measures per dataset. We also neglected some country members, such as "Euro area (17 countries)" or "Euro area (18 countries)", because we have a high number of countries and wanted to reduce the overall count of generated websites.

ID	Type	Label	Member Count
lfsi_emp_a	Dimension	year	0
	Dimension	gender	2
	Dimension	country	29
	Measure	employment rate	-
	Measure	absolute employment	-
tec00001	Dimension	year	0
	Dimension	country	28
	Measure	gdp in million euros	-
	Measure	gdp per capita	-

Table 4.7.: Dimensions and Measures of tec00001 and lfsi_emp_a in configuration

We chose a *Max amount of free Dimensions* and *Dice Dimensionality* of two. By using formula 3.5, this results in 120 possible views for tec00001 and 494 views for lfsi_emp_a. For both datasets, our sitemap contains exactly this number of links (see appendix F). Therefore, we conclude the correctness of our formulas in chapter 3.3.1 and of our algorithm for generating all possible requests (chapter 3.3.3) and its implementation (appendix E).

4.3.2. Benchmark Description

As presented in the following section 4.3.3, we measure the search engine rank of our published datasets for various keyword queries. In order to compare the results, we define two benchmarks which are presented in the following.

The baseline is the dataset itself, as it is published by *Eurostat*. A web surfer can access and explore the datasets via an online pivot table which is available on the *Eurostat*-website.

The second benchmark are landingpages from *Eurostat* for their datasets which link to the baseline tables. *Eurostat* describes the information included in their datasets on manually created webpages. They provide a long text with various keywords. Therefore, these landingpages rank well for many different search queries.

Obviously, only a fraction of all facts can be described on those pages. In contrast to our published landingpages, the concrete information is not necessarily included on the page.

Table 4.8 lists the benchmark URLs depending on the dataset:

Benchmark ID	Dataset ID	URL
Baseline	lfsi_emp_a	http://appsso.eurostat.ec.europa.eu/nui/show.do?wai=true&dataset=lfsi_emp_a
Baseline	tec00001	http://ec.europa.eu/eurostat/tgm/table.do?tab=table&plugin=1&language=en&pcode=tec00001
Eurostat-Landingpage	lfsi_emp_a	http://ec.europa.eu/eurostat/statistics-explained/index.php/Employment_statistics
Eurostat-Landingpage	tec00001	http://ec.europa.eu/eurostat/statistics-explained/index.php/National_accounts_and_GDP

Table 4.8.: Benchmark URLs

4.3.3. Evaluation Method

For the purpose of this evaluation, we acquired the domain *http://open-statistics.org*. It is a neutral domain name in the field of statistics and is therefore well suited to host webpages with statistical data. On open-statistics.org, we published the two datasets as described in section 4.3.1. They were first published on 22nd of December 2015. The sitemap was submitted to Google on the same day. Since it takes a couple of weeks until new pages rank well, we waited until the 6th of March 2016 for this evaluation. Besides setting two links from other websites, we did not do any off-page optimization techniques in order to strengthen the PageRank or TrustRank of open-statistics.org.

In this evaluation, we measure the search engine rank depending on different keywords for our website *open-statistics.org* and the two benchmarks as described in section 4.3.2. Since Google is by far the most used search engine with a market share of 94% in Germany[4], we only measure the Google rank.

The rank assessment was done by the software *CuteRank*[5]. As search engine, we defined the German Google in English language. By using the software, we guarantee that the rank is not influenced by a personal search profile.

Ranks higher than 100 are not assessed by *CuteRank*. Since results above the second result page receive a *Click-Through-Rate* of only 1,6% [Pet14], we treat those cases as not found. For mathematical aggregation, we set the values to 100.

We defined a set of 84 different search queries, which are grouped by four main keywords. The main keywords are: ”employment”, ”employment rate”, ”gross domestic product” and ”gdp”. So there are two main keywords for each dataset. It has to be noted that we consider multiple words as one keyword. Since we want to analyze, if

[4]http://de.statista.com/statistik/daten/studie/167841/umfrage/marktanteile-ausgewaehlter-suchmaschinen-in-deutschland/, last accessed 2016-02-05

[5]http://cuterank.net/, last accessed 2016-03-14

the amount of keywords per query affects the search engine rank, we further grouped the query-set by the amount of included keywords. As an example, the set includes the query ”gdp per capita hungary per year”, which has the main keyword ”gdp” and three other keywords ”per capita”, ”hungary” and ”per year”, thus four keywords in total.

The results of our rank assessment are displayed in appendix C. We aggregated the values by building the average value per main keyword depending on the amount of keywords. Since queries with a value of 100 strongly influence the average value, we calculated two average values: a *normal* one, including the outliers and a *clean* one, excluding the outliers. The calculated average values including the standard deviation are shown in table 4.9 on page 55.

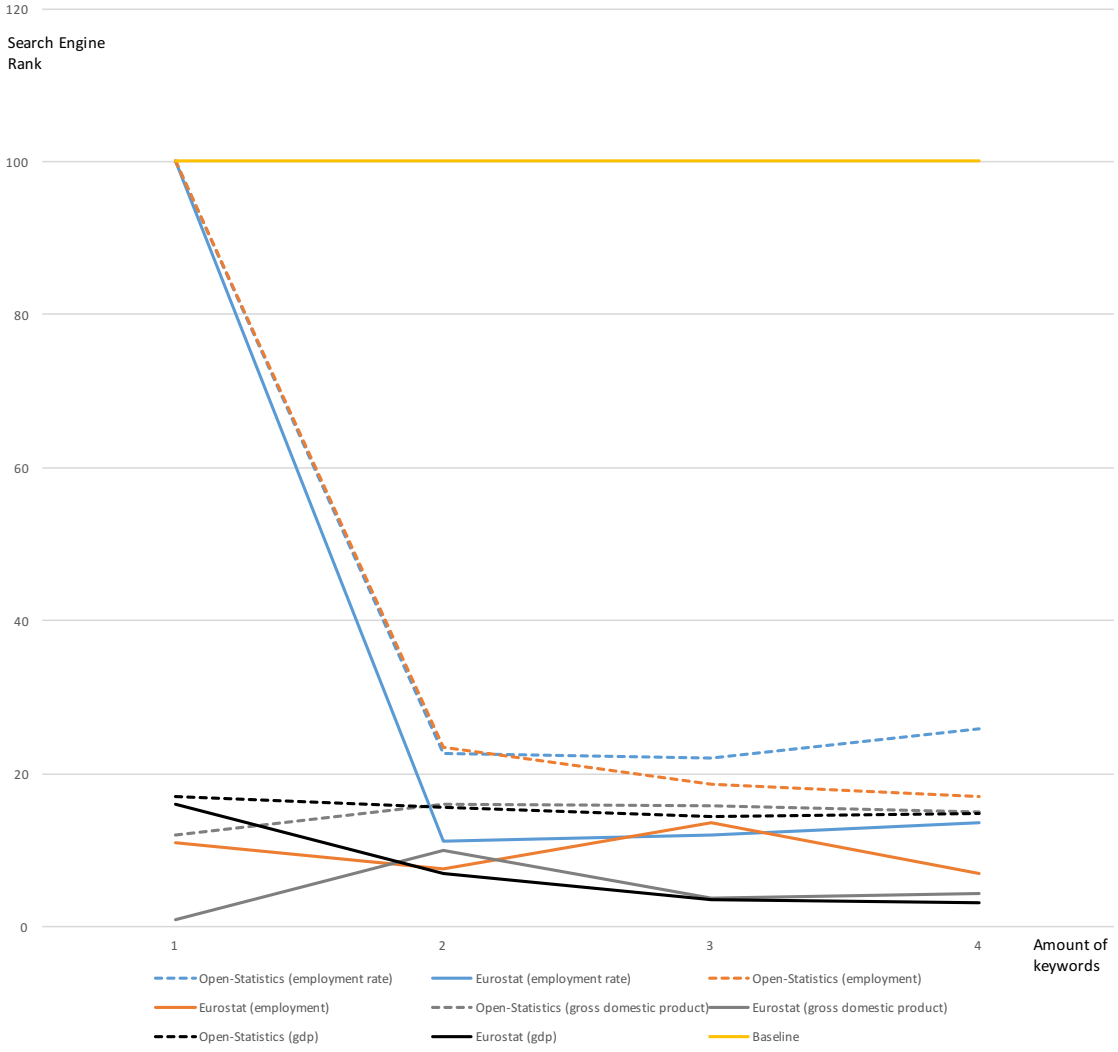

Figure 4.4.: Cleaned average Google ranks per main keyword for open-statistics.org and benchmarks

Figure 4.4 illustrates the Google ranks for the cleaned averages per main keyword

depending on the amount of keywords. Our own published datasets on *open-statistics.org* are marked with a dotted line. The corresponding *Eurostat* landingpages as benchmark are illustrated in the same color with a normal line. The baseline benchmark was never found for any query and is therefore always on top at rank 100.

Figure 4.5 shows the average Google rank for all keywords per amount of keywords. In this case, we aggregated the group of main keywords to the overall average. The figure compares the *normal* with the *clean* averages. The figure illustrates the strong effect of outliers on the aggregation function. This is why we used cleaned averages in figure 4.4.

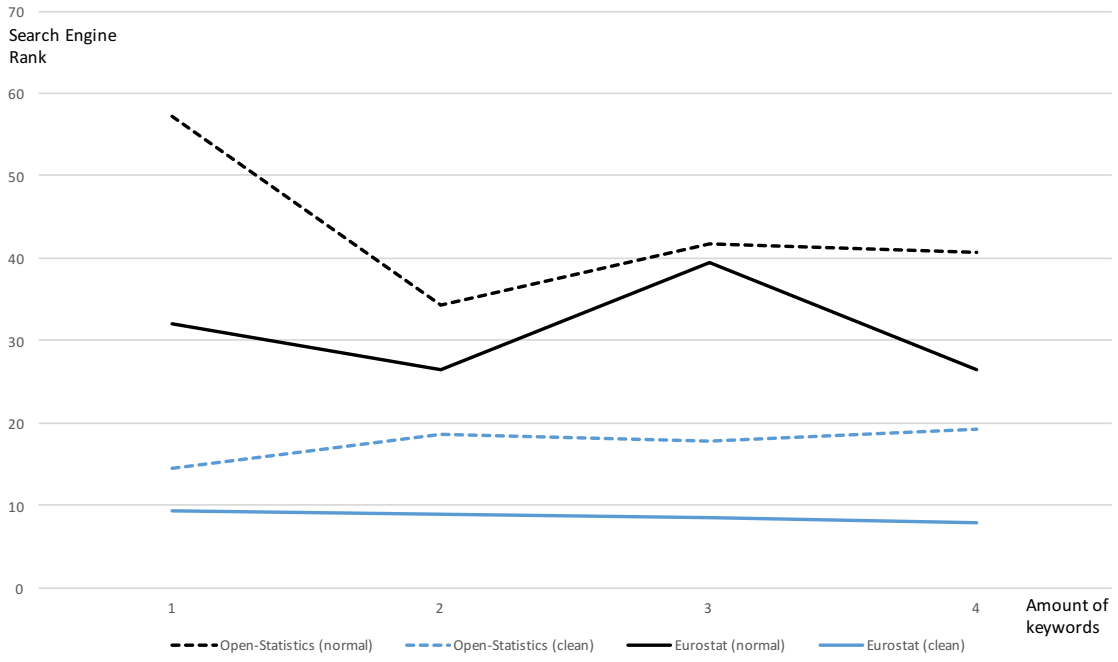

Figure 4.5.: Aggregated keywords (cleaned and normal) for open-statistics.org and benchmark

The findings of our assessment are discussed in the following section.

4.3.4. Findings

Prior to the assessment, we formulated hypotheses, that are presented and discussed in the following:

- *Webpages generated by SEO4OLAP are indexed by search engines and retrieved for specific queries*: The assessment shows that this thesis holds. Even though we created a lot of highly similar webpages, the according landingpage for

its specified keywords is in most cases retrieved by Google. This shows, that the approach to generate websites for a high number of views per dataset is feasible.

- *The more specific a query is, the better our search engine results are*: Our approach is to generate a lot of views per dataset and thereby produce very specific landingpages for longtail queries. Since a query consisting of more keywords is more specific and thereby has less competition, we assumed that our pages rank better with more keywords. This thesis can not be confirmed by the assessed data. As soon as two keywords are involved in a query, the rankings do not improve significantly with further keywords.

- *Datasets published by SEO4OLAP with higher competitive keywords, rank worse than datasets with low competitive keywords*: We assumed, that datasets of higher interest have more competition and therefore our pages will rank worse in comparison to less important datasets. In consequence, we assumed, that the GDP datasets will rank worse than the employment dataset. As our assessment shows, the contrary occurred. Therefore, our hypotheses can not be confirmed. It has to be noted, that this thesis is hard to verify with this evaluation method, since many different factors influence the search engine ranking.

- *SEO4OLAP pages rank better than the benchmark*: This thesis holds depending on the benchmark. The baseline, i.e. the pivot table containing the dataset, is never found by Google for our defined queries. In comparison to this, our pages rank significantly better. On the contrary, the manually generated landingpages by Eurostat rank very well and on average always better than our pages.

Domain	Type	Aggregator	Main keyword	Amount of keywords 1	2	3	4
Open-Statistics	normal	Average	employment rate	100,0	48,5	35,0	50,6
Open-Statistics	normal	Std. Deviation	employment rate	0,0	37,4	29,3	35,3
Open-Statistics	clean	Average	employment rate	100,0	22,8	22,0	25,8
Open-Statistics	clean	Std. Deviation	employment rate	0,0	10,5	4,0	6,2
Eurostat	normal	Average	employment rate	100,0	40,8	41,4	35,3
Eurostat	normal	Std. Deviation	employment rate	0,0	41,9	41,5	37,5
Eurostat	clean	Average	employment rate	100,0	11,3	12,0	13,7
Eurostat	clean	Std. Deviation	employment rate	0,0	1,5	2,2	2,8
Open-Statistics	normal	Average	employment	100,0	49,0	59,3	41,7
Open-Statistics	normal	Std. Deviation	employment	0,0	41,8	44,3	41,4
Open-Statistics	clean	Average	employment	100,0	23,5	18,7	17,0
Open-Statistics	clean	Std. Deviation	employment	0,0	25,8	25,0	16,6
Eurostat	normal	Average	employment	11,0	38,3	28,0	38,0
Eurostat	normal	Std. Deviation	employment	0,0	43,6	33,4	44,0
Eurostat	clean	Average	employment	11,0	7,5	13,6	7,0
Eurostat	clean	Std. Deviation	employment	0,0	1,7	9,8	4,9
Open-Statistics	normal	Average	gross domestic product	12,0	16,0	29,8	43,3
Open-Statistics	normal	Std. Deviation	gross domestic product	0,0	9,1	32,6	41,0
Open-Statistics	clean	Average	gross domestic product	12,0	16,0	15,8	15,0
Open-Statistics	clean	Std. Deviation	gross domestic product	0,0	9,1	9,5	10,7
Eurostat	normal	Average	gross domestic product	1,0	10,0	52,5	20,3
Eurostat	normal	Std. Deviation	gross domestic product	0,0	4,8	47,6	35,7
Eurostat	clean	Average	gross domestic product	1,0	10,0	3,8	4,4
Eurostat	clean	Std. Deviation	gross domestic product	0,0	4,8	4,3	2,7
Open-Statistics	normal	Average	gdp	17,0	27,7	43,0	14,8
Open-Statistics	normal	Std. Deviation	gdp	0,0	30,1	40,5	4,0
Open-Statistics	clean	Average	gdp	17,0	15,7	14,5	14,8
Open-Statistics	clean	Std. Deviation	gdp	0,0	6,6	5,1	4,0
Eurostat	normal	Average	gdp	16,0	20,3	35,7	3,2
Eurostat	normal	Std. Deviation	gdp	0,0	32,8	45,5	3,2
Eurostat	clean	Average	gdp	16,0	7,0	3,5	3,2
Eurostat	clean	Std. Deviation	gdp	0,0	4,3	0,5	3,2
Open-Statistics	normal	Average	[Aggregation of all]	57,3	34,3	41,8	40,8
Open-Statistics	normal	Std. Deviation	[Aggregation of all]	42,8	34,5	38,8	36,9
Open-Statistics	clean	Average	[Aggregation of all]	14,5	18,6	17,8	19,3
Open-Statistics	clean	Std. Deviation	[Aggregation of all]	2,5	14,2	12,5	10,7
Eurostat	normal	Average	[Aggregation of all]	32,0	26,4	39,4	26,4
Eurostat	normal	Std. Deviation	[Aggregation of all]	39,6	36,1	43,3	37,1
Eurostat	clean	Average	[Aggregation of all]	9,3	8,9	8,5	8,0
Eurostat	clean	Std. Deviation	[Aggregation of all]	6,2	4,1	7,4	5,6

Table 4.9.: Average search engine ranks and standard deviation depending on amount of keywords

5. Discussion

The idea of our approach *SEO4OLAP* is to generate custom SEO-landingpages for every possible view of a datacube. The evaluation of our implementation shows that, by doing this, *SEO4OLAP* was able to achieve better rankings than the baseline. In the case of *Eurostat*, the baseline was an online pivot table showing the dataset. In consequence, our approach is an improvement for data publishers who simply publish their data without further efforts to manually describe their data on landingpages. This leads to the following benefits:

- Webpages created by *SEO4OLAP* are found by Google for the corresponding keywords. This can be a new source of user traffic for dataset publishers.
- Single facts or views of a dataset are presented in a human readable representation. Thereby, facts of the Semantic Web are made accessible to normal web surfers.
- *SEO4OLAP* allows to reference specific facts in an HTML representation by a URL, whereas before, one could only refer to an entire dataset. As an example, this is an advantage for researcher who want to reference a statistic source.

In the following, some aspects regarding our approach and our evaluation are discussed.

Benchmark comparison

Besides the baseline, we also benchmarked our approach against manually created landingpages by *Eurostat*. Our evaluation shows that on average, we were not able to achieve better rankings than this benchmark. It has to be noted that *Eurostat* has a high authority domain which is trusted by Google. It can be assumed that our rankings would be better, if we had done SEO Off-Page Optimization in order to

gain domain trust and PageRank. Nevertheless, we can only speculate whether we could have beaten the benchmark by applying those techniques. But we can derive a recommendation for data publishers. If the goal of using *SEO4OLAP* is to gain a new source of traffic, we recommend to apply off-page optimization techniques in order to achieve rankings on the first search engine result page.

Longtail approach

In contrast to manually generating descriptive landingpages, our approach allows to automatically generate potentially thousands of webpages. From a SEO perspective, this is a longtail strategy, since those pages are optimized for very specific search queries with low competition. As mentioned in our evaluation findings (section 4.3.4), we assumed that the more keywords are added to a query, the more specific the query and thus the better the rank. We observed that, once a query includes two keywords, adding further keywords does not improve the rank. An explanation would be that queries consisting of two keywords are already very specific in the domain of statistical facts. Thus, the competition for these keywords is already very low. At this point, our pages compete against pages which are not optimized for this exact set of keywords, but instead have a higher PageRank or TrustRank. Therefore, a further specification is not improving the result rank.

Publishing high dimensional datasets

In chapter 4.2, we illustrated how the total amount of potential views evolves in high dimensional datasets. In order to reduce this amount, we recommended to apply restrictions, e.g. by restricting the Dice Dimensionality. In our evaluation, we only published small datasets with a maximum of three dimensions, in order to test our approach at first with reduced complexity. Therefore, we do not know how search engines react to high dimensional datasets with more than 100.000 pages for one dataset. We leave the SEO-evaluation of high dimensional datasets to future research.

Handling real world datasets

Our approach is designed to generate webpages automatically by analyzing the scheme of a data cube. In practice, we were forced to implement a configuration file in order to react to real world modeling issues, such as *Measure-Dimensions* and *Slice-Members* (see chapter 4.1.3). We are confident that such a configuration file can be generated automatically, if datasets were modeled as intended by the RDF Data Cube Vocabulary. The vocabulary already contains means to properly model *Measure-Dimensions*. Since *Slice-Members* often exist in real world datasets, we recommend to extend the vocabulary with means to mark members as *Slice-Member*.

Schema.org

SEO4OLAP converts machine readable data into human readable webpages. Search engines would benefit, if the data was also provided in a machine readable manner. Therefore, semantic markup technologies such as Schema.org were developed. In our implementation, we added Schema.org markup in order to semantically describe the content of our webpages. The main intention of doing this, was to achieve better search engine rankings. To the best of our knowledge, Schema.org does not provide means to properly describe statistic facts. It is possible to describe a dataset, the publisher, the publishing date and many more. However, single statistical facts can not be described. We think our approach and also statistical data itself would benefit, if such functionality would be added to Schema.org.

6. Related Work

This is, to the best of our knowledge, the first attempt to generate search engine optimized landingpages out of data cubes, especially statistical linked data. Nevertheless, there are a variety of studies related to our research approach.

The related work can be grouped in three different areas. The visualization and human-friendly interaction with linked data, approaches to enhance the machine-readability of websites through semantic annotation and studies about Search Engine Optimization. In the following, we discuss some of those studies.

Linked Data Visualization

Since Linked Data has a self-describing data format, it has the benefit of being machine-readable, thus allowing machine interpretation, e.g. by search engines such as the Google Knowledge Graph[1]. On the contrary, even though the amount of published linked data is growing on a fast scale [BHBL09], there is still a lot of research to be conducted on how non-experts can interact with this data.

One approach is described by Hoefler [Hoe13]. He introduces different tools, that allow the analysis and visualization of linked data without the knowledge of SPARQL or other semantic technologies. Those tools, further explained in [STV+14], are e.g. the *CODE Query Wizard* and the *CODE Visualization Wizard (Vis Wizard)*. *Vis Wizard* automatically matches visualizations, such as bar charts or pie charts with data represented in the RDF Data Cube Vocabulary.

A similar approach is presented by Salas et al. [SMM+12] with *CubeViz*. It also allows the visualization of RDF Data Cubes and is an extension to the semantic

[1] http://www.google.com/intl/es419/insidesearch/features/search/knowledge.html

Wiki Software *Ontowiki*. In contrast to *Vis Wizard*, it does not automatically suggest visualizations.

Another system allowing users to analyze Linked Data Cubes is the *Linked Data Cubes Explorer* by Kämpgen and Harth [KH14]. We have in common, that we use their OLAP-engine *OLAP4LD* as technological basis.

The presented studies analyze, transform or visualize RDF Data Cubes, just as we do. In contrast, we focus on a presentation optimized for search engines and at this stage, do not focus on user interaction. Other researchers try to make linked data in general, meaning not especially Data Cubes, more accessible to non-experts. The following approaches also have in common, that they do not focus on a search engine optimized presentation.

Steger et al. [SKS13] present a Javascript framework, that allows web designers with no background in semantic technologies to build Linked Data applications. Bergmann et al. [BBE+13] create a linked database about soccer games and players and build an interface for easy exploration. Popov et al. [PCH+12] present a system which supports the interconnection of semantic applications in order to build richer systems.

Semantic Annotation

Another related research area is the automatic annotation of documents, especially websites, with semantic information. A very interesting approach is presented by Ambiah and Lokuse [AL12]. They describe two case studies, one on generating new webpages with microdata and the other focusing on enriching existing webpages with microdata. In contrast to us, they evaluate the semantic annotation by comparing their system to other enriching systems with a qualitative benchmark. They do not evaluate whether their semantic annotation leads to improvements in search engine rankings.

A different study is presented by Kudelka et al. [KSL+09] who annotate websites by means of web patterns. Veres and Elseth [VE13] present the system *MaDaMe* which annotates websites with Schema.org concepts. Other studies try to generate ontologies from varying sources or through different means, but those do not have much in common with our research question.

A lot of research on how search engines can be improved by using semantic information is conducted by Peter Mika. In [Mik15] he describes how semantic markup technologies have evolved over time and how this led to Schema.org.

Search Engine Optimization

The last related field deals with Search Engine Optimization. A lot of research in this area is conducted by major SEO-Agencies, who drive experiments in order to further understand how website rankings can be improved. Since they rely on a competitive advantage, it can be concluded that only a fraction of those experiments are made public. Nevertheless, some academic studies are available and will be shortly presented in the following. Contrary to our approach, they analyze which techniques influence the search engine ranking, whereas we simply use these techniques for our approach.

Beel et al. [BGW09] analyze how the ranking of academic papers can be improved. They present some advice for optimization within papers, but do not evaluate whether their approach is successful. Shih et al. [SCC13] describe an empirical evaluation on how basic SEO-Techniques correspond to rankings and Malaga [Mal09] evaluates the effect of Web 2.0 techniques.

7. Conclusion

7.1. Summary

We have presented our approach *SEO4OLAP* to generate search engine optimized landingpages from arbitrary datacubes, modeled in the RDF Data Cube Vocabulary. We developed a new OLAP query model which can be represented by a clean URL-scheme and therefore allows to be submitted via HTTP. We illustrated a system architecture which is able to process such OLAP-requests by transforming them into Logical OLAP Query Plans, executes them on an OLAP-engine and enhances results with corresponding keywords.

We also presented a mathematical model to calculate the amount of possible requests depending on the amount of members per dimension. Therefore, two formulas were shown. The first one calculates the upper bound of possible views. The second one introduces two restrictions in order to decrease the overall amount. Both formulas were numerically verified.

We demonstrated and implemented an algorithm which creates a list of all possible OLAP-requests depending on the data cube schema and the restrictions. Our evaluation shows that the amount of generated requests corresponds to our mathematical model.

In order to evaluate our approach from a SEO perspective, we implemented a *SEO4OLAP* system in Java and published two datasets from *Eurostat* which resulted in 614 generated landingpages. We evaluated how well our pages are found by search engines in comparison to the dataset source, as well as to manually generated websites from *Eurostat*.

As a conclusion, it can be stated that our approach is feasible in practice and has benefits for data publishers, search engine providers and web surfers. First, single

facts of a dataset are represented as HTML and are thus referenceable and human readable; Second, facts can be found by search engines which results in a new traffic source for data publishers; Third, the system provides an interface for non-experts to Statistical Linked Data.

7.2. Future Work

Future work may be conducted in the following areas:

- So far, we evaluated our system on small datasets only. It would be interesting to analyze how search engines react to huge datasets with millions of generated webpages.
- The assessed search engine results were conducted two month after first publication. At this point, our domain has not earned trust or a high PageRank. It would be interesting to see how rankings evolve over time, after increasing the PageRank through SEO off-page optimization techniques.
- The possibilities of the Semantic Web could be further utilized, e.g. by dynamically acquiring relevant content and thereby enhancing the generated landing-pages.

Literature

[AL12] N. Ambiah and D. Lukose, "Enriching webpages with semantic information," in *Proceedings of the 2012 International Conference on Dublin Core and Metadata Applications*. Dublin Core Metadata Initiative, 2012, pp. 1–11.

[And08] C. Anderson, "Long tail, the revised and updated edition: Why the future of business is selling less of more," 2008.

[BBE+13] T. Bergmann, S. Bunk, J. Eschrig, C. Hentschel, M. Knuth, H. Sack, and R. Schüler, "Linked Soccer Data," in *I-SEMANTICS (Posters & Demos)*. Citeseer, 2013, pp. 25–29.

[Beu10] J. Beus, "Google macht die Ladegeschwindigkeit zum Rankingfaktor," 2010, https://www.sistrix.de/news/google-macht-die-ladegeschwindigkeit-zum-rankingfaktor/ [Online; accessed 05-February-2016].

[BGW09] J. Beel, B. Gipp, and E. Wilde, "Academic Search Engine Optimization (aseo) Optimizing Scholarly Literature for Google Scholar and Co." *Journal of scholarly publishing*, vol. 41, no. 2, pp. 176–190, 2009.

[BHBL09] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data-the story so far," *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, pp. 205–227, 2009.

[BL06] T. Berners-Lee, "Linked Data - Design Issues," 2006, https://www.w3.org/DesignIssues/LinkedData.html [Online; accessed 03-Febuary-2016].

[CAN13] S. Capadisli, S. Auer, and A.-C. N. Ngomo, "Linked SDMX data," *Semantic Web Journal*, pp. 1–8, 2013.

[CGLG04] C. Cunningham, C. A. Galindo-Legaria, and G. Graefe, "PIVOT and UNPIVOT: Optimization and Execution Strategies in an RDBMS," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 2004, pp. 998–1009.

[CR14] R. Cyganiak and D. Reynolds, "The RDF Data Cube Vocabulary," 2014, https://www.w3.org/TR/vocab-data-cube/ [Online; accessed 21-January-2016].

[Erl16] S. Erlhofer, *Suchmaschinenoptimierung*. Rheinwerk Computing, 2016.

[ESFS12] E. Enge, S. Spencer, R. Fishkin, and J. Stricchiola, *The art of SEO*. O'Reilly Media, Inc., 2012.

[EV12] L. Etcheverry and A. A. Vaisman, "Enhancing OLAP analysis with web cubes," in *The Semantic Web: Research and Applications*. Springer, 2012, pp. 469–483.

[GCB+97] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh, "Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals," *Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 29–53, 1997.

[GGMP04] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen, "Combating web spam with trustrank," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 2004, pp. 576–587.

[HB11] T. Heath and C. Bizer, "Linked data: Evolving the web into a global data space," *Synthesis lectures on the semantic web: theory and technology*, vol. 1, no. 1, pp. 1–136, 2011.

[Hoe13] P. Hoefler, "Linked data interfaces for non-expert users," in *The Semantic Web: Semantics and Big Data*. Springer, 2013, pp. 702–706.

[iPr06] iProspect, "iProspect Search Engine User Behavior Study," 2006, http://district4.extension.ifas.ufl.edu/Tech/TechPubs/WhitePaper_2006_SearchEngineUserBehavior.pdf [Online; accessed 21-January-2016].

[Käm15] B. Kämpgen, *Flexible Integration and Efficient Analysis of Multidimensional Datasets from the Web*. KIT Scientific Publishing, 2015.

[KH14] B. Kämpgen and A. Harth, "OLAP4LD–A Framework for Building Analysis Applications Over Governmental Statistics," in *The Semantic Web: ESWC 2014 Satellite Events*. Springer, 2014, pp. 389–394.

[KOH12] B. Kämpgen, S. O'Riain, and A. Harth, "Interacting with statistical linked data via OLAP operations," in *The Semantic Web: ESWC 2012 Satellite Events*. Springer, 2012, pp. 87–101.

[KSL+09] M. Kudelka, V. Snasel, O. Lehecka, E. El-Qawasmeh, and J. Pokornỳ, "Semantic Annotation of Web Pages Using Web Patterns," in *Advanced Internet Based Systems and Applications*. Springer, 2009, pp. 280–291.

[LM06] A. N. Langville and C. D. Meyer, *Google's PageRank and beyond: The science of search engine rankings*. Princeton University Press, 2006.

[Mal08] R. A. Malaga, "Worst practices in search engine optimization," *Communications of the ACM*, vol. 51, no. 12, pp. 147–150, 2008.

[Mal09] R. Malaga, "Web 2.0 Techniques for search engine optimization: Two case studies," *Review of Business Research*, vol. 9, no. 1, pp. 132–139, 2009.

[MHT+14] B. Mutlu, P. Hoefler, G. Tschinkel, E. Veas, V. Sabol, F. Stegmaier, and M. Granitzer, "Suggesting visualisations for published data," *Proceedings of IVAPP*, pp. 267–275, 2014.

[Mik15] P. Mika, "On Schema.org and Why It Matters for the Web," *Internet Computing, IEEE*, vol. 19, no. 4, pp. 52–55, 2015.

[PBMW99] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: bringing order to the web." 1999.

[PCH+12] I. Popov, G. Correndo, W. Hall, N. Shadbolt *et al.*, "Interacting with the web of data through a web of inter-connected lenses," 2012.

[Pet14] P. Petrescu, "Google Organic Click-Through Rates in 2014," 2014, https://moz.com/blog/google-organic-click-through-rates-in-2014 [Online; accessed 30-February-2016].

[RMA+11] O. Romero, P. Marcel, A. Abelló, V. Peralta, and L. Bellatreche, "Describing analytical sessions using a multidimensional algebra," in *Data Warehousing and Knowledge Discovery*. Springer, 2011, pp. 224–239.

[Rog11] A. Roggio, "Understanding Traffic Sources in Google Analytics," 2011, http://www.practicalecommerce.com/articles/2916-Understanding-Traffic-Sources-in-Google-Analytics [Online; accessed 30-February-2016].

[SBJC14] M. Schmachtenberg, C. Bizer, A. Jentzsch, and R. Cyganiak, "Linked Open Data Cloud diagram," 2014, http://lod-cloud.net/ [Online; accessed 03-February-2016].

[SCC13] B.-Y. Shih, C.-Y. Chen, and Z.-S. Chen, "An empirical study of an internet marketing strategy for search engine optimization," *Human Factors and Ergonomics in Manufacturing & Service Industries*, vol. 23, no. 6, pp. 528–540, 2013.

[SKS13] B. Steger, T. Kurz, and S. Schaffert, "Resource Description Graph Views for Configuring Linked Data Visualizations," in *I-SEMANTICS (Posters & Demos)*. Citeseer, 2013, pp. 30–34.

[SMM+12] P. E. R. Salas, M. Martin, F. M. D. Mota, S. Auer, K. Breitman, M. Casanova *et al.*, "Publishing statistical data on the web," in *Semantic Computing (ICSC), 2012 IEEE Sixth International Conference on*. IEEE, 2012, pp. 285–292.

[STV+14] V. Sabol, G. Tschinkel, E. Veas, P. Hoefler, B. Mutlu, and M. Granitzer, "Discovery and visual analysis of linked data for humans," in *The Semantic Web–ISWC 2014*. Springer, 2014, pp. 309–324.

[VE13] C. Veres and E. Elseth, "Schema.org for the Semantic Web with MaDaME." in *I-SEMANTICS (Posters & Demos)*. Citeseer, 2013, pp. 11–15.

Acronyms

Acronym	Meaning
API	Application Programming Interface
BI	Business Intelligence
HTTP	Hypertext Transfer Protocol
LOD	Linked Open Data
MDM	Multidimensional Data Model
MDX	Multidimensional Expressions
MVC	Model-View-Controller
OLAP	Online Analytical Processing
OWL	Web Ontology Language
QB	RDF Data Cube Vocabulary
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RSS	Rich Site Summary
SEO	Search Engine Optimization
SERP	Search Engine Results Page
SLD	Statistical Linked Data
SPARQL	SPARQL Protocol And RDF Query Language
SQL	Structured Query Language
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
XML	Extensible Markup Language

Table 7.1.: Acronyms

Appendices

A. Evaluation Sources

The source code of the *SEO4OLAP*-implementation (see chapter 4.1) is published as Open Source under the Apache License 2.0[1] at `https://github.com/dbreucker/seo4olap`.

The computed and assessed data used in chapter 4 can be downloaded from `https://github.com/dbreucker/seo4olap-evaluation`. It is also presented in the following chapters of the appendix.

[1] http://www.apache.org/licenses/

B. Complexity Evaluation

The following table shows results from formula 3.1 and 3.2 depending on different dataset parameters and restrictions. Some of these results were used in chapter 4.2.

Members per Dimension	Dimension count n	Measures m	Max Free Dimensions	Dice Dimensionality	Views	Upper Bound
10	1	1	1	1	12	12
10	2	1	1	1	43	144
10	2	1	1	2	143	144
10	2	1	2	1	44	144
10	2	1	2	2	144	144
10	3	1	1	1	94	1.728
10	3	1	1	2	694	1.728
10	3	1	1	3	1.694	1.728
10	3	1	2	1	127	1.728
10	3	1	2	2	727	1.728
10	3	1	2	3	1.727	1.728
10	3	1	3	1	128	1.728
10	3	1	3	2	728	1.728
10	3	1	3	3	1.728	1.728
10	4	1	1	1	165	20.736
10	4	1	1	2	1.965	20.736
10	4	1	1	3	9.965	20.736
10	4	1	1	4	19.965	20.736
10	4	1	2	1	291	20.736
10	4	1	2	2	2.691	20.736
10	4	1	2	3	10.691	20.736
10	4	1	2	4	20.691	20.736
10	4	1	3	1	335	20.736
10	4	1	3	2	2.735	20.736
10	4	1	3	3	10.735	20.736
10	4	1	3	4	20.735	20.736
10	4	1	4	1	336	20.736
10	4	1	4	2	2.736	20.736
10	4	1	4	3	10.736	20.736
10	4	1	4	4	20.736	20.736
10	5	1	1	1	256	248.832
10	5	1	1	2	4.256	248.832
10	5	1	1	3	34.256	248.832
10	5	1	1	4	134.256	248.832
10	5	1	1	5	234.256	248.832
10	5	1	2	1	566	248.832
10	5	1	2	2	7.566	248.832
10	5	1	2	3	47.566	248.832
10	5	1	2	4	147.566	248.832
10	5	1	2	5	247.566	248.832
10	5	1	3	1	776	248.832
10	5	1	3	2	8.776	248.832
10	5	1	3	3	48.776	248.832
10	5	1	3	4	148.776	248.832
10	5	1	3	5	248.776	248.832
10	5	1	4	1	831	248.832
10	5	1	4	2	8.831	248.832
10	5	1	4	3	48.831	248.832
10	5	1	4	4	148.831	248.832
10	5	1	4	5	248.831	248.832
10	5	1	5	1	832	248.832
10	5	1	5	2	8.832	248.832
10	5	1	5	3	48.832	248.832
10	5	1	5	4	148.832	248.832
10	5	1	5	5	248.832	248.832
10	6	1	1	1	367	2.985.984
10	6	1	1	2	7.867	2.985.984
10	6	1	1	3	87.867	2.985.984
10	6	1	1	4	537.867	2.985.984
10	6	1	1	5	1.737.867	2.985.984
10	6	1	1	6	2.737.867	2.985.984
10	6	1	2	1	982	2.985.984
10	6	1	2	2	17.482	2.985.984
10	6	1	2	3	157.482	2.985.984
10	6	1	2	4	757.482	2.985.984
10	6	1	2	5	1.957.482	2.985.984
10	6	1	2	6	2.957.482	2.985.984
10	6	1	3	1	1.602	2.985.984
10	6	1	3	2	24.102	2.985.984
10	6	1	3	3	184.102	2.985.984

10	6	1	3	4	784.102	2.985.984
10	6	1	3	5	1.984.102	2.985.984
10	6	1	3	6	2.984.102	2.985.984
10	6	1	4	1	1.917	2.985.984
10	6	1	4	2	25.917	2.985.984
10	6	1	4	3	185.917	2.985.984
10	6	1	4	4	785.917	2.985.984
10	6	1	4	5	1.985.917	2.985.984
10	6	1	4	6	2.985.917	2.985.984
10	6	1	5	1	1.983	2.985.984
10	6	1	5	2	25.983	2.985.984
10	6	1	5	3	185.983	2.985.984
10	6	1	5	4	785.983	2.985.984
10	6	1	5	5	1.985.983	2.985.984
10	6	1	5	6	2.985.983	2.985.984
10	6	1	6	1	1.984	2.985.984
10	6	1	6	2	25.984	2.985.984
10	6	1	6	3	185.984	2.985.984
10	6	1	6	4	785.984	2.985.984
10	6	1	6	5	1.985.984	2.985.984
10	6	1	6	6	2.985.984	2.985.984
10	7	1	1	1	498	35.831.808
10	7	1	1	2	13.098	35.831.808
10	7	1	1	3	188.098	35.831.808
10	7	1	1	4	1.588.098	35.831.808
10	7	1	1	5	7.888.098	35.831.808
10	7	1	1	6	21.888.098	35.831.808
10	7	1	1	7	31.888.098	35.831.808
10	7	1	2	1	1.569	35.831.808
10	7	1	2	2	35.169	35.831.808
10	7	1	2	3	420.169	35.831.808
10	7	1	2	4	2.870.169	35.831.808
10	7	1	2	5	11.270.169	35.831.808
10	7	1	2	6	25.270.169	35.831.808
10	7	1	2	7	35.270.169	35.831.808
10	7	1	3	1	3.004	35.831.808
10	7	1	3	2	57.604	35.831.808
10	7	1	3	3	582.604	35.831.808
10	7	1	3	4	3.382.604	35.831.808
10	7	1	3	5	11.782.604	35.831.808
10	7	1	3	6	25.782.604	35.831.808
10	7	1	3	7	35.782.604	35.831.808
10	7	1	4	1	4.089	35.831.808
10	7	1	4	2	69.189	35.831.808
10	7	1	4	3	629.189	35.831.808
10	7	1	4	4	3.429.189	35.831.808
10	7	1	4	5	11.829.189	35.831.808
10	7	1	4	6	25.829.189	35.831.808
10	7	1	4	7	35.829.189	35.831.808
10	7	1	5	1	4.530	35.831.808
10	7	1	5	2	71.730	35.831.808
10	7	1	5	3	631.730	35.831.808
10	7	1	5	4	3.431.730	35.831.808
10	7	1	5	5	11.831.730	35.831.808
10	7	1	5	6	25.831.730	35.831.808
10	7	1	5	7	35.831.730	35.831.808
10	7	1	6	1	4.607	35.831.808
10	7	1	6	2	71.807	35.831.808
10	7	1	6	3	631.807	35.831.808
10	7	1	6	4	3.431.807	35.831.808
10	7	1	6	5	11.831.807	35.831.808
10	7	1	6	6	25.831.807	35.831.808
10	7	1	6	7	35.831.807	35.831.808
10	7	1	7	1	4.608	35.831.808
10	7	1	7	2	71.808	35.831.808
10	7	1	7	3	631.808	35.831.808
10	7	1	7	4	3.431.808	35.831.808
10	7	1	7	5	11.831.808	35.831.808
10	7	1	7	6	25.831.808	35.831.808
10	7	1	7	7	35.831.808	35.831.808
10	8	1	1	1	649	429.981.696
10	8	1	1	2	20.249	429.981.696
10	8	1	1	3	356.249	429.981.696
10	8	1	1	4	3.856.249	429.981.696
10	8	1	1	5	26.256.249	429.981.696
10	8	1	1	6	110.256.249	429.981.696
10	8	1	1	7	270.256.249	429.981.696
10	8	1	1	8	370.256.249	429.981.696
10	8	1	2	1	2.357	429.981.696
10	8	1	2	2	63.957	429.981.696
10	8	1	2	3	959.957	429.981.696

10	8	1	2	4	8.659.957	429.981.696
10	8	1	2	5	47.859.957	429.981.696
10	8	1	2	6	159.859.957	429.981.696
10	8	1	2	7	319.859.957	429.981.696
10	8	1	2	8	419.859.957	429.981.696
10	8	1	3	1	5.213	429.981.696
10	8	1	3	2	122.813	429.981.696
10	8	1	3	3	1.578.813	429.981.696
10	8	1	3	4	12.078.813	429.981.696
10	8	1	3	5	56.878.813	429.981.696
10	8	1	3	6	168.878.813	429.981.696
10	8	1	3	7	328.878.813	429.981.696
10	8	1	3	8	428.878.813	429.981.696
10	8	1	4	1	8.083	429.981.696
10	8	1	4	2	167.683	429.981.696
10	8	1	4	3	1.903.683	429.981.696
10	8	1	4	4	13.103.683	429.981.696
10	8	1	4	5	57.903.683	429.981.696
10	8	1	4	6	169.903.683	429.981.696
10	8	1	4	7	329.903.683	429.981.696
10	8	1	4	8	429.903.683	429.981.696
10	8	1	5	1	9.819	429.981.696
10	8	1	5	2	186.219	429.981.696
10	8	1	5	3	1.978.219	429.981.696
10	8	1	5	4	13.178.219	429.981.696
10	8	1	5	5	57.978.219	429.981.696
10	8	1	5	6	169.978.219	429.981.696
10	8	1	5	7	329.978.219	429.981.696
10	8	1	5	8	429.978.219	429.981.696
10	8	1	6	1	10.407	429.981.696
10	8	1	6	2	189.607	429.981.696
10	8	1	6	3	1.981.607	429.981.696
10	8	1	6	4	13.181.607	429.981.696
10	8	1	6	5	57.981.607	429.981.696
10	8	1	6	6	169.981.607	429.981.696
10	8	1	6	7	329.981.607	429.981.696
10	8	1	6	8	429.981.607	429.981.696
10	8	1	7	1	10.495	429.981.696
10	8	1	7	2	189.695	429.981.696
10	8	1	7	3	1.981.695	429.981.696
10	8	1	7	4	13.181.695	429.981.696
10	8	1	7	5	57.981.695	429.981.696
10	8	1	7	6	169.981.695	429.981.696
10	8	1	7	7	329.981.695	429.981.696
10	8	1	7	8	429.981.695	429.981.696
10	8	1	8	1	10.496	429.981.696
10	8	1	8	2	189.696	429.981.696
10	8	1	8	3	1.981.696	429.981.696
10	8	1	8	4	13.181.696	429.981.696

Table B.2.: Amount of views depending on dataset parameters and restrictions

C. SEO Evaluation

The following table shows ranking results per keyword. The values were aggregated for our evaluation in chapter 4.3.3. The values were assessed on the 6th of March 2016 by the software *CuteRank*. The values on the right three columns represent the rank of the search engine Google in Germany with English language.

Values above 100 are set to 100, since *CuteRank* is not able to retrieve the exact number higher than 100.

Keyword	Amount of Keywords	Open-Statistics.org	Eurostat Landingpage	Eurostat Pivot Table
Employment	1	100	11	100
employment croatia	2	64	10	100
employment croatia women	3	100	13	100
employment croatia women per year	4	27	7	100
employment france	2	28	8	100
employment france women	3	54	10	100
employment france women per year	4	39	15	100
employment germany	2	1	6	100
employment germany women	3	1	3	100
employment germany women per year	4	1	2	100
employment hungary	2	1	6	100
employment hungary women	3	1	10	100
employment hungary women per year	4	1	4	100
employment ireland	2	100	100	100
employment ireland women	3	100	32	100
employment ireland women per year	4	100	100	100
employment poland	2	100	100	100
employment poland women	3	100	100	100
employment poland women per year	4	100	100	100
Employment rate	1	100	100	100
employment rate croatia	2	100	100	100
employment rate croatia men per year	4	100	100	100
employment rate croatia women	3	100	100	100
employment rate croatia women per year	4	100	100	100
employment rate france	2	100	100	100
employment rate france men per year	4	100	14	100
employment rate france women	3	29	13	100
employment rate france women per year	4	24	100	100
employment rate germany	2	17	9	100
employment rate germany men per year	4	29	13	100
employment rate germany women	3	23	100	100
employment rate germany women per year	4	100	12	100
employment rate hungary	2	39	12	100
employment rate hungary men per year	4	37	20	100
employment rate hungary women	3	21	15	100
employment rate hungary women per year	4	30	13	100
employment rate ireland	2	24	11	100
employment rate ireland men per year	4	17	11	100
employment rate ireland women	3	20	9	100
employment rate ireland women per year	4	29	16	100
employment rate poland	2	11	13	100
employment rate poland men per year	4	19	14	100
employment rate poland women	3	17	11	100
employment rate poland women per year	4	22	10	100
gdp	1	17	16	100
gdp croatia	2	12	13	100
gdp france	2	17	10	100
gdp germany	2	16	9	100
gdp hungary	2	28	2	100
gdp ireland	2	100	100	100
gdp per capita	2	15	7	100
gdp per capita croatia	3	20	3	100
gdp per capita croatia per year	4	19	3	100
gdp per capita france	3	19	4	100
gdp per capita france per year	4	17	2	100
gdp per capita germany	3	100	100	100
gdp per capita germany per year	4	18	2	100
gdp per capita hungary	3	8	3	100
gdp per capita hungary per year	4	16	10	100
gdp per capita ireland	3	11	4	100
gdp per capita ireland per year	4	11	1	100
gdp per capita poland	3	100	100	100

gdp per capita poland per year	4	8	1	100
gdp poland	2	6	1	100
Gross domestic product	1	12	1	100
gross domestic product croatia	2	4	2	100
gross domestic product france	2	25	14	100
gross domestic product germany	2	17	13	100
gross domestic product hungary	2	22	16	100
gross domestic product ireland	2	15	11	100
gross domestic product per capita	2	27	10	100
gross domestic product per capita croatia	3	22	11	100
gross domestic product per capita croatia per year	4	100	8	100
gross domestic product per capita france	3	100	100	100
gross domestic product per capita france per year	4	100	7	100
gross domestic product per capita germany	3	14	3	100
gross domestic product per capita germany per year	4	33	100	100
gross domestic product per capita hungary	3	30	100	100
gross domestic product per capita hungary per year	4	12	4	100
gross domestic product per capita ireland	3	11	100	100
gross domestic product per capita ireland per year	4	5	1	100
gross domestic product per capita poland	3	2	1	100
gross domestic product per capita poland per year	4	10	2	100
gross domestic product poland	2	2	4	100

Table C.3.: Google Ranking results per keyword

D. Complexity Computation

The following shows the Java code of the class `ComplexityCalculator` which implements formulas 3.1 and 3.2 of chapter 3.3.1.

```
public class ComplexityCalculator {

	private ComplexityCalculator() {}

	/**
	* Calculates the maximum possible amount of views
	* @param measureCount the amount of measures
	* @param dimensions an Array consisting of the amount of members
	* per dimension
	* @return
	*/
	public static long calculateMaxViews(int measureCount, int[] dimensions){
		final int n = dimensions.length;
		long result = 1;
		for(int i = 0; i < n; i ++){
			result *= (dimensions[i] + 2);
		}
		return measureCount * result;
	}

	/**
	* Calculates the amount of views
	* @param measureCount the amount of measures
	* @param dimensions an Array consisting of the amount of members
	* per dimension
	* @param maxFreeDimensions
	* @param maxMembers2Dice
	* @return
	*/
	public static long calculateViews(int measureCount, int[] dimensions, int maxFreeDimensions, int maxMembers2Dice){
		final int n = dimensions.length;
		long result = factor(maxFreeDimensions, 0, n);
		for(int i = 1; i <= maxMembers2Dice; i ++){
			result += factor(maxFreeDimensions, i, n) * diceCombinations(dimensions, i);
		}
		return measureCount * result;
	}

	private static long diceCombinations(int[] dimensions, int diceDimensionality){
		final int n = dimensions.length;
		int upperBound = n - diceDimensionality + 1;
		long sum = recursiveSum(dimensions, 1, upperBound);
		return sum;
	}

	private static long recursiveSum(int[] dimensions, int lowerBound, int upperBound){
		final int n = dimensions.length;
		long sum = 0;
		for(int i = lowerBound; i<=upperBound; i++){
			if(upperBound == n){
				sum += dimensions[i-1];
			}
			else{
				sum += dimensions[i-1] * recursiveSum(dimensions, i+1, upperBound + 1);
			}
		}
		return sum;
	}

	private static long factor(int maxFreeDim, int diceDim, int dimensionCount){
		long factor = 0;
		for(int i = 0; i <= maxFreeDim; i ++){
			long increment = 0;
			if((dimensionCount - diceDim - i) >= 0 ){
				increment = factorial(dimensionCount - diceDim) / (factorial(i) * factorial(dimensionCount
				- diceDim - i));
			}
			factor += increment;
		}
		return factor;
	}

	private static int factorial(int n){
```

```
                if( n <= 1)     // base case
                        return 1;
                else
                        return n * factorial( n - 1 );
        }

        private static int[] generateArray(int value, int size){
                int[] array = new int[size];
                for(int i = 0 ; i< size; i ++ ){
                        array[i] = value;
                }
                return array;
        }
}
```

E. Computation of all OLAP Queries

The following shows the Java code of the class `RequestListGenerator` which implements the algorithm of chapter 3.3.3.

```
class RequestListGenerator {

	private final ConfigurationManager configManager;
	private final SitemapConfiguration sitemap;
	private List<Request> requests;
	private final URL datasetUri;
	private final List<String> measures;
	private final List<String> dimensions;
	private final List<String> members;

	public RequestListGenerator(URL datasetUri) {
		if(datasetUri == null){
			throw new InvalidParameterException("datasetUri cannot be null");
		}
		this.datasetUri = datasetUri;
		this.configManager = ConfigurationManagerFactory.getConfigurationManager();
		SitemapConfiguration sc = configManager.getSitemapConfiguration(datasetUri);
		if(sc == null){
			sc = new SitemapConfiguration();
		}
		this.sitemap = sc;
		this.requests = new ArrayList<Request>();

		//get Measures
		List<String> measures = configManager.getMeasures(datasetUri);
		List<String> measures2add = new ArrayList<String>();
		List<String> measures2remove = new ArrayList<String>();
		for(String measure : measures){
			List<String> measureMembers = configManager.getMembers(datasetUri, measure);
			if(measureMembers != null){
				measures2add.addAll(measureMembers);
				measures2remove.add(measure);
			}
		}
		measures.addAll(measures2add);
		measures.removeAll(measures2remove);
		this.measures = measures;

		//get Dimensions
		List<String> dimensions = configManager.getDimensions(datasetUri);
		List<String> dimensions2remove = new ArrayList<String>();
		for(String dimension: dimensions){
			if(configManager.isMeasureDimension(datasetUri, dimension)){
				dimensions2remove.add(dimension);
			}
		}
		dimensions.removeAll(dimensions2remove);
		this.dimensions = dimensions;

		//get Members without SliceMembers
		List<String> members = new ArrayList<String>();
		for(String dimension : dimensions){
			List<String> dimensionMembers = configManager.getMembers(datasetUri, dimension);
			if(dimensionMembers != null){
				for(String member: dimensionMembers){
					if(!configManager.isSliceMember(datasetUri, member)){
						members.add(member);
					}
				}
			}
		}
		this.members = members;
	}

	/**
	* Get a list of URL-Requests, e.g. baseUri/endpoint/pattern/id, given the Configuration of the Dataset
	* @return
	*/
	public List<String> getURLRequestList(boolean absolutePath){
		List<String> requests = new ArrayList<String>();
		List<OlapRequest> olapRequests = getOlapRequestList();
		for(OlapRequest olapRequest: olapRequests){
			Link link = LinkGenerator.getLink(olapRequest, absolutePath);
```

```
                if(link != null && link.getUrl() != null){
                        requests.add(link.getUrl());
                }
        }
        return requests;
}

/*#############-----------###################
*
* Private Methods
*
*#############-----------###################*/

/**
* Get a list of OlapRequests, given the Configuration of the Dataset
* @return
*/
private List<OlapRequest> getOlapRequestList(){
        RequestPattern maxPattern = new RequestPattern(sitemap.getMaxMeasureCount(),
        sitemap.getMaxDimensionCount(), sitemap.getMaxMemberCount());
        List<Request> rootList = this.generateRootList(maxPattern);
        for(Request root : rootList){
                this.doNextStep(root);
        }
        cleanup(requests);
        List<OlapRequest> olapRequests = new ArrayList<OlapRequest>();
        for(Request request: requests){
                olapRequests.add(request.getOlapRequest());
        }
        return olapRequests;
}

/**
* Recursive generation of further Request-Objects. Stops when all Requests are created.
* @param request
*/
private void doNextStep(Request request){
        RequestPattern actualPattern = request.actualPattern;
        RequestPattern targetPattern = request.targetPattern;
        if(actualPattern.equals(targetPattern)){
                requests.add(request);
                return;
        }
        if(actualPattern.amountMeasures < targetPattern.amountMeasures){
                addMetadata(request, this.measures, MetadataType.MEASURE);
                return;
        }
        if(actualPattern.amountDimensions < targetPattern.amountDimensions){
                addMetadata(request, this.dimensions, MetadataType.DIMENSION);
                return;
        }
        if(actualPattern.amountMembers < targetPattern.amountMembers){
                addMetadata(request, this.members, MetadataType.MEMBER);
                return;
        }
}

/**
* Adds a Metadata Object to request, i.e. either a new Measure, a Dimension or a Member. Is called recursively
* by doNextStep()
* @param request to add Metadata
* @param metadata uniqueName of Measure/Dimension/Member
* @param type MetadataType has to fit with metadata uniqueName
*/
private void addMetadata(Request request, List<String> metadata, MetadataType type){
        ConfigurationManager configManager = ConfigurationManagerFactory.getConfigurationManager();

        for(String uniqueName: metadata){
                Request newRequest = request.copy();
                String dimensionUniqueName = null;
                if(type.equals(MetadataType.DIMENSION)){
                        dimensionUniqueName = uniqueName;
                }
                if(type.equals(MetadataType.MEMBER)){
                        dimensionUniqueName = configManager.getParent(datasetUri, uniqueName);
                }
                //a path may only contain one member or dimension per dimension
                //a path may not include the same path item twice
                if(!newRequest.isIncrementIncluded(uniqueName) && !newRequest.hasDimension(dimensionUniqueName))
                {
                        newRequest.addToRequest(uniqueName, type);
```

```
                        newRequest.increment(type);
                        if(dimensionUniqueName != null){
                                newRequest.putDimension(dimensionUniqueName);
                        }
                        doNextStep(newRequest);
                }
                else{
                        newRequest = null;
                }
        }
}

/**
* Generates a rootList to start recursive Request-Creation
* @param maxPattern The maximal pattern to be included in PathList
* @return
*/
private List<Request> generateRootList(RequestPattern maxPattern){
        List<Request> rootList = new ArrayList<Request>();
        for(int amountMeasures = 0; amountMeasures <= maxPattern.amountMeasures; amountMeasures++){
                for(int amountDimensions = 0; amountDimensions <= maxPattern.amountDimensions; amountDimensions++){
                        for(int amountMembers = 0; amountMembers <= maxPattern.amountMembers; amountMembers++){
                                RequestPattern actualPattern = new RequestPattern(0,0,0);
                                RequestPattern targetPattern = new RequestPattern(amountMeasures, amountDimensions, amountMembers);
                                OlapRequest olapRequest = new OlapRequest(datasetUri, new ArrayList<String>(),
                                new ArrayList<String>(), new ArrayList<String>());
                                Request path = new Request(olapRequest, actualPattern, targetPattern, new HashSet<String>());
                                rootList.add(path);
                        }
                }
        }
        return rootList;
}

/**
* Removes duplicate request. Duplicate request are e.g.
* 'mypath/path1/path2' and 'mypath/path2/path1'
* @param requests List of requests, in which duplicates should be removed
*/
private List<Request> removeDuplicates(List<Request> requests){
        Set<Integer> pathHashSet = new HashSet<Integer>();
        List<Request> paths2remove = new ArrayList<Request>();
        for(Request request : requests){
                boolean isUnique = pathHashSet.add(request.requestHash());
                if(!isUnique){
                        paths2remove.add(request);
                }
        }
        requests.removeAll(paths2remove);
        return requests;
}

/**
* Cleans List of requests from duplicates and removes all requests, that do not fit for
* the MinPattern from Configuration
* @param requests List of requests to clean
* @return cleaned List
*/
private List<Request> cleanup(List<Request> requests){
        requests = removeDuplicates(requests);
        RequestPattern minPattern = new RequestPattern(sitemap.getMinMeasureCount(),
        sitemap.getMinDimensionCount(), sitemap.getMinMemberCount());
        List<Request> requests2remove = new ArrayList<Request>();
        for(Request request: requests){
                if(request.targetPattern.isSmaller(minPattern)){
                        requests2remove.add(request);
                }
        }
        requests.removeAll(requests2remove);
        return requests;
}



/*#############-----------###################
*
* Inner classes
*
*#############-----------###################*/

private class Request{
```

```
private OlapRequest olapRequest;
private RequestPattern actualPattern;
private final RequestPattern targetPattern;
private final Set<String> includedDimensions;

public Request(OlapRequest olapRequest, RequestPattern actualPattern, RequestPattern targetPattern,
          Set<String> includedDimensions) {
        this.olapRequest = olapRequest;
        this.actualPattern = actualPattern;
        this.targetPattern = targetPattern;
        this.includedDimensions = includedDimensions;
}

public OlapRequest getOlapRequest() {
        return olapRequest;
}

public Request copy(){
        Set<String> newDimensions = new HashSet<String>();
        Iterator<String> it = this.includedDimensions.iterator();
        while(it.hasNext()){
                newDimensions.add(it.next());
        }
        return new Request(this.olapRequest.copy(), this.actualPattern.copy(), this.targetPattern.copy(), newDimensions);
}

public int requestHash(){
        return olapRequest.hashCode();
}

public void addToRequest(String uniqueName, MetadataType type){
        if(uniqueName != null){
                List<String> measures = olapRequest.getMeasures2project();
                List<String> dimensions = olapRequest.getDimensions2keep();
                List<String> members = olapRequest.getMembers2dice();
                if(type == MetadataType.MEASURE){
                        measures.add(uniqueName);
                }
                if(type == MetadataType.DIMENSION){
                        dimensions.add(uniqueName);
                }
                if(type == MetadataType.MEMBER){
                        members.add(uniqueName);
                }
                OlapRequest newRequest = new OlapRequest(this.olapRequest.getDatasetUri(),
                members, dimensions, measures);
                this.olapRequest = newRequest;
        }
}

public void putDimension(String dimensionUniqueName){
        includedDimensions.add(dimensionUniqueName);
}

public boolean hasDimension(String dimensionUniqueName){
return includedDimensions.contains(dimensionUniqueName);
}

public String toString(){
        String output = "olapRequest:'" + olapRequest + "', actualPattern:'" + actualPattern.toString()
        + "', targetPattern:'" + targetPattern.toString() + "'";
        return output;
}

public void increment(MetadataType type){
        switch (type){
                case MEASURE:
                        RequestPattern newActualPattern = new RequestPattern(actualPattern.amountMeasures + 1,
                        actualPattern.amountDimensions, actualPattern.amountMembers);
                        this.actualPattern = newActualPattern;
                        break;
                case DIMENSION:
                        RequestPattern newActualPattern1 = new RequestPattern(actualPattern.amountMeasures,
                        actualPattern.amountDimensions + 1, actualPattern.amountMembers);
                        this.actualPattern = newActualPattern1;
                        break;
                case MEMBER:
                        RequestPattern newActualPattern2 = new RequestPattern(actualPattern.amountMeasures,
                        actualPattern.amountDimensions, actualPattern.amountMembers + 1);
                        this.actualPattern = newActualPattern2;
```

```
                        break;
                }
        }

        public boolean isIncrementIncluded(final String uniqueName){
                if(uniqueName == null){
                        return false;
                }
                boolean isInMeasures = olapRequest.getMeasures2project().contains(uniqueName);
                boolean isInDimensions = olapRequest.getDimensions2keep().contains(uniqueName);
                boolean isInMembers = olapRequest.getMembers2dice().contains(uniqueName);

                return isInMeasures || isInDimensions || isInMembers ;
        }

        }

        private class RequestPattern{

        public final int amountMeasures;
        public final int amountDimensions;
        public final int amountMembers;

        public RequestPattern(int amountMeasures, int amountDimensions, int amountMembers){
                this.amountMeasures = amountMeasures;
                this.amountDimensions = amountDimensions;
                this.amountMembers = amountMembers;
        }

        public RequestPattern copy(){
                return new RequestPattern(this.amountMeasures, this.amountDimensions, this.amountMembers);
        }

        public String toString(){
                String output = "";
                output += amountMeasures;
                output += amountDimensions;
                output += amountMembers;
                return output;
        }

        public boolean equals(RequestPattern comparePattern){
                if(comparePattern == null){
                        return false;
                }
                return (amountMeasures == comparePattern.amountMeasures &&
                amountDimensions == comparePattern.amountDimensions &&
                amountMembers == comparePattern.amountMembers);
        }

        public boolean isSmaller(RequestPattern comparePattern){
                if(comparePattern == null){
                        return false;
                }
                return (amountMeasures < comparePattern.amountMeasures ||
                amountDimensions < comparePattern.amountDimensions ||
                amountMembers < comparePattern.amountMembers);
                }
        }

        enum MetadataType {
                MEASURE, DIMENSION, MEMBER
        }
}
```

F. Generated Sitemaps

The following tables show the generated sitemap links of dataset lfsi_emp_a and tec00001. The domain "http://open-statistics.org" is abbreviated with "..".

No.	URL	No.	URL
1	../employment/100/absolute	251	../employment/111/rate/year/austria
2	../employment/100/rate	252	../employment/111/rate/year/belgium
3	../employment/101/absolute/female	253	../employment/111/rate/year/bulgaria
4	../employment/101/absolute/male	254	../employment/111/rate/year/switzerland
5	../employment/101/absolute/austria	255	../employment/111/rate/year/cyprus
6	../employment/101/absolute/belgium	256	../employment/111/rate/year/cz
7	../employment/101/absolute/bulgaria	257	../employment/111/rate/year/germany
8	../employment/101/absolute/switzerland	258	../employment/111/rate/year/denmark
9	../employment/101/absolute/cyprus	259	../employment/111/rate/year/euro
10	../employment/101/absolute/cz	260	../employment/111/rate/year/estonia
11	../employment/101/absolute/germany	261	../employment/111/rate/year/greece
12	../employment/101/absolute/denmark	262	../employment/111/rate/year/spain
13	../employment/101/absolute/euro	263	../employment/111/rate/year/finland
14	../employment/101/absolute/estonia	264	../employment/111/rate/year/france
15	../employment/101/absolute/greece	265	../employment/111/rate/year/croatia
16	../employment/101/absolute/spain	266	../employment/111/rate/year/hungary
17	../employment/101/absolute/finland	267	../employment/111/rate/year/ireland
18	../employment/101/absolute/france	268	../employment/111/rate/year/iceland
19	../employment/101/absolute/croatia	269	../employment/111/rate/year/italy
20	../employment/101/absolute/hungary	270	../employment/111/rate/year/lithuania
21	../employment/101/absolute/ireland	271	../employment/111/rate/year/netherlands
22	../employment/101/absolute/iceland	272	../employment/111/rate/year/norway
23	../employment/101/absolute/italy	273	../employment/111/rate/year/poland
24	../employment/101/absolute/lithuania	274	../employment/111/rate/year/romania
25	../employment/101/absolute/netherlands	275	../employment/111/rate/year/sweden
26	../employment/101/absolute/norway	276	../employment/111/rate/year/slovenia
27	../employment/101/absolute/poland	277	../employment/111/rate/year/slovakia
28	../employment/101/absolute/romania	278	../employment/111/rate/year/uk
29	../employment/101/absolute/sweden	279	../employment/111/rate/year/us
30	../employment/101/absolute/slovenia	280	../employment/111/rate/gender/austria
31	../employment/101/absolute/slovakia	281	../employment/111/rate/gender/belgium
32	../employment/101/absolute/uk	282	../employment/111/rate/gender/bulgaria
33	../employment/101/absolute/us	283	../employment/111/rate/gender/switzerland
34	../employment/101/rate/female	284	../employment/111/rate/gender/cyprus
35	../employment/101/rate/male	285	../employment/111/rate/gender/cz
36	../employment/101/rate/austria	286	../employment/111/rate/gender/germany
37	../employment/101/rate/belgium	287	../employment/111/rate/gender/denmark
38	../employment/101/rate/bulgaria	288	../employment/111/rate/gender/euro
39	../employment/101/rate/switzerland	289	../employment/111/rate/gender/estonia
40	../employment/101/rate/cyprus	290	../employment/111/rate/gender/greece
41	../employment/101/rate/cz	291	../employment/111/rate/gender/spain
42	../employment/101/rate/germany	292	../employment/111/rate/gender/finland
43	../employment/101/rate/denmark	293	../employment/111/rate/gender/france
44	../employment/101/rate/euro	294	../employment/111/rate/gender/croatia
45	../employment/101/rate/estonia	295	../employment/111/rate/gender/hungary
46	../employment/101/rate/greece	296	../employment/111/rate/gender/ireland
47	../employment/101/rate/spain	297	../employment/111/rate/gender/iceland
48	../employment/101/rate/finland	298	../employment/111/rate/gender/italy
49	../employment/101/rate/france	299	../employment/111/rate/gender/lithuania
50	../employment/101/rate/croatia	300	../employment/111/rate/gender/netherlands
51	../employment/101/rate/hungary	301	../employment/111/rate/gender/norway
52	../employment/101/rate/ireland	302	../employment/111/rate/gender/poland
53	../employment/101/rate/iceland	303	../employment/111/rate/gender/romania
54	../employment/101/rate/italy	304	../employment/111/rate/gender/sweden
55	../employment/101/rate/lithuania	305	../employment/111/rate/gender/slovenia
56	../employment/101/rate/netherlands	306	../employment/111/rate/gender/slovakia
57	../employment/101/rate/norway	307	../employment/111/rate/gender/uk
58	../employment/101/rate/poland	308	../employment/111/rate/gender/us
59	../employment/101/rate/romania	309	../employment/111/rate/country/female
60	../employment/101/rate/sweden	310	../employment/111/rate/country/male
61	../employment/101/rate/slovenia	311	../employment/112/absolute/year/austria/female
62	../employment/101/rate/slovakia	312	../employment/112/absolute/year/belgium/female
63	../employment/101/rate/uk	313	../employment/112/absolute/year/bulgaria/female
64	../employment/101/rate/us	314	../employment/112/absolute/year/switzerland/female
65	../employment/102/absolute/austria/female	315	../employment/112/absolute/year/cyprus/female
66	../employment/102/absolute/belgium/female	316	../employment/112/absolute/year/cz/female
67	../employment/102/absolute/bulgaria/female	317	../employment/112/absolute/year/germany/female
68	../employment/102/absolute/switzerland/female	318	../employment/112/absolute/year/denmark/female
69	../employment/102/absolute/cyprus/female	319	../employment/112/absolute/year/euro/female
70	../employment/102/absolute/cz/female	320	../employment/112/absolute/year/estonia/female
71	../employment/102/absolute/germany/female	321	../employment/112/absolute/year/greece/female
72	../employment/102/absolute/denmark/female	322	../employment/112/absolute/year/spain/female

73	../employment/102/absolute/euro/female
74	../employment/102/absolute/estonia/female
75	../employment/102/absolute/greece/female
76	../employment/102/absolute/spain/female
77	../employment/102/absolute/finland/female
78	../employment/102/absolute/france/female
79	../employment/102/absolute/croatia/female
80	../employment/102/absolute/hungary/female
81	../employment/102/absolute/ireland/female
82	../employment/102/absolute/iceland/female
83	../employment/102/absolute/italy/female
84	../employment/102/absolute/lithuania/female
85	../employment/102/absolute/netherlands/female
86	../employment/102/absolute/norway/female
87	../employment/102/absolute/poland/female
88	../employment/102/absolute/romania/female
89	../employment/102/absolute/sweden/female
90	../employment/102/absolute/slovenia/female
91	../employment/102/absolute/slovakia/female
92	../employment/102/absolute/uk/female
93	../employment/102/absolute/us/female
94	../employment/102/absolute/austria/male
95	../employment/102/absolute/belgium/male
96	../employment/102/absolute/bulgaria/male
97	../employment/102/absolute/switzerland/male
98	../employment/102/absolute/cyprus/male
99	../employment/102/absolute/cz/male
100	../employment/102/absolute/germany/male
101	../employment/102/absolute/denmark/male
102	../employment/102/absolute/euro/male
103	../employment/102/absolute/estonia/male
104	../employment/102/absolute/greece/male
105	../employment/102/absolute/spain/male
106	../employment/102/absolute/finland/male
107	../employment/102/absolute/france/male
108	../employment/102/absolute/croatia/male
109	../employment/102/absolute/hungary/male
110	../employment/102/absolute/ireland/male
111	../employment/102/absolute/iceland/male
112	../employment/102/absolute/italy/male
113	../employment/102/absolute/lithuania/male
114	../employment/102/absolute/netherlands/male
115	../employment/102/absolute/norway/male
116	../employment/102/absolute/poland/male
117	../employment/102/absolute/romania/male
118	../employment/102/absolute/sweden/male
119	../employment/102/absolute/slovenia/male
120	../employment/102/absolute/slovakia/male
121	../employment/102/absolute/uk/male
122	../employment/102/absolute/us/male
123	../employment/102/rate/austria/female
124	../employment/102/rate/belgium/female
125	../employment/102/rate/bulgaria/female
126	../employment/102/rate/switzerland/female
127	../employment/102/rate/cyprus/female
128	../employment/102/rate/cz/female
129	../employment/102/rate/germany/female
130	../employment/102/rate/denmark/female
131	../employment/102/rate/euro/female
132	../employment/102/rate/estonia/female
133	../employment/102/rate/greece/female
134	../employment/102/rate/spain/female
135	../employment/102/rate/finland/female
136	../employment/102/rate/france/female
137	../employment/102/rate/croatia/female
138	../employment/102/rate/hungary/female
139	../employment/102/rate/ireland/female
140	../employment/102/rate/iceland/female
141	../employment/102/rate/italy/female
142	../employment/102/rate/lithuania/female
143	../employment/102/rate/netherlands/female
144	../employment/102/rate/norway/female
145	../employment/102/rate/poland/female
146	../employment/102/rate/romania/female
147	../employment/102/rate/sweden/female
148	../employment/102/rate/slovenia/female
149	../employment/102/rate/slovakia/female
150	../employment/102/rate/uk/female
151	../employment/102/rate/us/female
152	../employment/102/rate/austria/male
153	../employment/102/rate/belgium/male
323	../employment/112/absolute/year/finland/female
324	../employment/112/absolute/year/france/female
325	../employment/112/absolute/year/croatia/female
326	../employment/112/absolute/year/hungary/female
327	../employment/112/absolute/year/ireland/female
328	../employment/112/absolute/year/iceland/female
329	../employment/112/absolute/year/italy/female
330	../employment/112/absolute/year/lithuania/female
331	../employment/112/absolute/year/netherlands/female
332	../employment/112/absolute/year/norway/female
333	../employment/112/absolute/year/poland/female
334	../employment/112/absolute/year/romania/female
335	../employment/112/absolute/year/sweden/female
336	../employment/112/absolute/year/slovenia/female
337	../employment/112/absolute/year/slovakia/female
338	../employment/112/absolute/year/uk/female
339	../employment/112/absolute/year/us/female
340	../employment/112/absolute/year/austria/male
341	../employment/112/absolute/year/belgium/male
342	../employment/112/absolute/year/bulgaria/male
343	../employment/112/absolute/year/switzerland/male
344	../employment/112/absolute/year/cyprus/male
345	../employment/112/absolute/year/cz/male
346	../employment/112/absolute/year/germany/male
347	../employment/112/absolute/year/denmark/male
348	../employment/112/absolute/year/euro/male
349	../employment/112/absolute/year/estonia/male
350	../employment/112/absolute/year/greece/male
351	../employment/112/absolute/year/spain/male
352	../employment/112/absolute/year/finland/male
353	../employment/112/absolute/year/france/male
354	../employment/112/absolute/year/croatia/male
355	../employment/112/absolute/year/hungary/male
356	../employment/112/absolute/year/ireland/male
357	../employment/112/absolute/year/iceland/male
358	../employment/112/absolute/year/italy/male
359	../employment/112/absolute/year/lithuania/male
360	../employment/112/absolute/year/netherlands/male
361	../employment/112/absolute/year/norway/male
362	../employment/112/absolute/year/poland/male
363	../employment/112/absolute/year/romania/male
364	../employment/112/absolute/year/sweden/male
365	../employment/112/absolute/year/slovenia/male
366	../employment/112/absolute/year/slovakia/male
367	../employment/112/absolute/year/uk/male
368	../employment/112/absolute/year/us/male
369	../employment/112/rate/year/austria/female
370	../employment/112/rate/year/belgium/female
371	../employment/112/rate/year/bulgaria/female
372	../employment/112/rate/year/switzerland/female
373	../employment/112/rate/year/cyprus/female
374	../employment/112/rate/year/cz/female
375	../employment/112/rate/year/germany/female
376	../employment/112/rate/year/denmark/female
377	../employment/112/rate/year/euro/female
378	../employment/112/rate/year/estonia/female
379	../employment/112/rate/year/greece/female
380	../employment/112/rate/year/spain/female
381	../employment/112/rate/year/finland/female
382	../employment/112/rate/year/france/female
383	../employment/112/rate/year/croatia/female
384	../employment/112/rate/year/hungary/female
385	../employment/112/rate/year/ireland/female
386	../employment/112/rate/year/iceland/female
387	../employment/112/rate/year/italy/female
388	../employment/112/rate/year/lithuania/female
389	../employment/112/rate/year/netherlands/female
390	../employment/112/rate/year/norway/female
391	../employment/112/rate/year/poland/female
392	../employment/112/rate/year/romania/female
393	../employment/112/rate/year/sweden/female
394	../employment/112/rate/year/slovenia/female
395	../employment/112/rate/year/slovakia/female
396	../employment/112/rate/year/uk/female
397	../employment/112/rate/year/us/female
398	../employment/112/rate/year/austria/male
399	../employment/112/rate/year/belgium/male
400	../employment/112/rate/year/bulgaria/male
401	../employment/112/rate/year/switzerland/male
402	../employment/112/rate/year/cyprus/male
403	../employment/112/rate/year/cz/male

154 ../employment/102/rate/bulgaria/male
155 ../employment/102/rate/switzerland/male
156 ../employment/102/rate/cyprus/male
157 ../employment/102/rate/cz/male
158 ../employment/102/rate/germany/male
159 ../employment/102/rate/denmark/male
160 ../employment/102/rate/euro/male
161 ../employment/102/rate/estonia/male
162 ../employment/102/rate/greece/male
163 ../employment/102/rate/spain/male
164 ../employment/102/rate/finland/male
165 ../employment/102/rate/france/male
166 ../employment/102/rate/croatia/male
167 ../employment/102/rate/hungary/male
168 ../employment/102/rate/ireland/male
169 ../employment/102/rate/iceland/male
170 ../employment/102/rate/italy/male
171 ../employment/102/rate/lithuania/male
172 ../employment/102/rate/netherlands/male
173 ../employment/102/rate/norway/male
174 ../employment/102/rate/poland/male
175 ../employment/102/rate/romania/male
176 ../employment/102/rate/sweden/male
177 ../employment/102/rate/slovenia/male
178 ../employment/102/rate/slovakia/male
179 ../employment/102/rate/uk/male
180 ../employment/102/rate/us/male
181 ../employment/110/absolute/year
182 ../employment/110/absolute/gender
183 ../employment/110/absolute/country
184 ../employment/110/rate/year
185 ../employment/110/rate/gender
186 ../employment/110/rate/country
187 ../employment/111/absolute/year/female
188 ../employment/111/absolute/year/male
189 ../employment/111/absolute/year/austria
190 ../employment/111/absolute/year/belgium
191 ../employment/111/absolute/year/bulgaria
192 ../employment/111/absolute/year/switzerland
193 ../employment/111/absolute/year/cyprus
194 ../employment/111/absolute/year/cz
195 ../employment/111/absolute/year/germany
196 ../employment/111/absolute/year/denmark
197 ../employment/111/absolute/year/euro
198 ../employment/111/absolute/year/estonia
199 ../employment/111/absolute/year/greece
200 ../employment/111/absolute/year/spain
201 ../employment/111/absolute/year/finland
202 ../employment/111/absolute/year/france
203 ../employment/111/absolute/year/croatia
204 ../employment/111/absolute/year/hungary
205 ../employment/111/absolute/year/ireland
206 ../employment/111/absolute/year/iceland
207 ../employment/111/absolute/year/italy
208 ../employment/111/absolute/year/lithuania
209 ../employment/111/absolute/year/netherlands
210 ../employment/111/absolute/year/norway
211 ../employment/111/absolute/year/poland
212 ../employment/111/absolute/year/romania
213 ../employment/111/absolute/year/sweden
214 ../employment/111/absolute/year/slovenia
215 ../employment/111/absolute/year/slovakia
216 ../employment/111/absolute/year/uk
217 ../employment/111/absolute/year/us
218 ../employment/111/absolute/gender/austria
219 ../employment/111/absolute/gender/belgium
220 ../employment/111/absolute/gender/bulgaria
221 ../employment/111/absolute/gender/switzerland
222 ../employment/111/absolute/gender/cyprus
223 ../employment/111/absolute/gender/cz
224 ../employment/111/absolute/gender/germany
225 ../employment/111/absolute/gender/denmark
226 ../employment/111/absolute/gender/euro
227 ../employment/111/absolute/gender/estonia
228 ../employment/111/absolute/gender/greece
229 ../employment/111/absolute/gender/spain
230 ../employment/111/absolute/gender/finland
231 ../employment/111/absolute/gender/france
232 ../employment/111/absolute/gender/croatia
233 ../employment/111/absolute/gender/hungary
234 ../employment/111/absolute/gender/ireland
404 ../employment/112/rate/year/germany/male
405 ../employment/112/rate/year/denmark/male
406 ../employment/112/rate/year/euro/male
407 ../employment/112/rate/year/estonia/male
408 ../employment/112/rate/year/greece/male
409 ../employment/112/rate/year/spain/male
410 ../employment/112/rate/year/finland/male
411 ../employment/112/rate/year/france/male
412 ../employment/112/rate/year/croatia/male
413 ../employment/112/rate/year/hungary/male
414 ../employment/112/rate/year/ireland/male
415 ../employment/112/rate/year/iceland/male
416 ../employment/112/rate/year/italy/male
417 ../employment/112/rate/year/lithuania/male
418 ../employment/112/rate/year/netherlands/male
419 ../employment/112/rate/year/norway/male
420 ../employment/112/rate/year/poland/male
421 ../employment/112/rate/year/romania/male
422 ../employment/112/rate/year/sweden/male
423 ../employment/112/rate/year/slovenia/male
424 ../employment/112/rate/year/slovakia/male
425 ../employment/112/rate/year/uk/male
426 ../employment/112/rate/year/us/male
427 ../employment/120/absolute/gender/year
428 ../employment/120/absolute/country/year
429 ../employment/120/absolute/country/gender
430 ../employment/120/rate/gender/year
431 ../employment/120/rate/country/year
432 ../employment/120/rate/country/gender
433 ../employment/121/absolute/gender/year/austria
434 ../employment/121/absolute/gender/year/belgium
435 ../employment/121/absolute/gender/year/bulgaria
436 ../employment/121/absolute/gender/year/switzerland
437 ../employment/121/absolute/gender/year/cyprus
438 ../employment/121/absolute/gender/year/cz
439 ../employment/121/absolute/gender/year/germany
440 ../employment/121/absolute/gender/year/denmark
441 ../employment/121/absolute/gender/year/euro
442 ../employment/121/absolute/gender/year/estonia
443 ../employment/121/absolute/gender/year/greece
444 ../employment/121/absolute/gender/year/spain
445 ../employment/121/absolute/gender/year/finland
446 ../employment/121/absolute/gender/year/france
447 ../employment/121/absolute/gender/year/croatia
448 ../employment/121/absolute/gender/year/hungary
449 ../employment/121/absolute/gender/year/ireland
450 ../employment/121/absolute/gender/year/iceland
451 ../employment/121/absolute/gender/year/italy
452 ../employment/121/absolute/gender/year/lithuania
453 ../employment/121/absolute/gender/year/netherlands
454 ../employment/121/absolute/gender/year/norway
455 ../employment/121/absolute/gender/year/poland
456 ../employment/121/absolute/gender/year/romania
457 ../employment/121/absolute/gender/year/sweden
458 ../employment/121/absolute/gender/year/slovenia
459 ../employment/121/absolute/gender/year/slovakia
460 ../employment/121/absolute/gender/year/uk
461 ../employment/121/absolute/gender/year/us
462 ../employment/121/absolute/country/year/female
463 ../employment/121/absolute/country/year/male
464 ../employment/121/rate/gender/year/austria
465 ../employment/121/rate/gender/year/belgium
466 ../employment/121/rate/gender/year/bulgaria
467 ../employment/121/rate/gender/year/switzerland
468 ../employment/121/rate/gender/year/cyprus
469 ../employment/121/rate/gender/year/cz
470 ../employment/121/rate/gender/year/germany
471 ../employment/121/rate/gender/year/denmark
472 ../employment/121/rate/gender/year/euro
473 ../employment/121/rate/gender/year/estonia
474 ../employment/121/rate/gender/year/greece
475 ../employment/121/rate/gender/year/spain
476 ../employment/121/rate/gender/year/finland
477 ../employment/121/rate/gender/year/france
478 ../employment/121/rate/gender/year/croatia
479 ../employment/121/rate/gender/year/hungary
480 ../employment/121/rate/gender/year/ireland
481 ../employment/121/rate/gender/year/iceland
482 ../employment/121/rate/gender/year/italy
483 ../employment/121/rate/gender/year/lithuania
484 ../employment/121/rate/gender/year/netherlands

235	../employment/111/absolute/gender/iceland	485	../employment/121/rate/gender/year/norway
236	../employment/111/absolute/gender/italy	486	../employment/121/rate/gender/year/poland
237	../employment/111/absolute/gender/lithuania	487	../employment/121/rate/gender/year/romania
238	../employment/111/absolute/gender/netherlands	488	../employment/121/rate/gender/year/sweden
239	../employment/111/absolute/gender/norway	489	../employment/121/rate/gender/year/slovenia
240	../employment/111/absolute/gender/poland	490	../employment/121/rate/gender/year/slovakia
241	../employment/111/absolute/gender/romania	491	../employment/121/rate/gender/year/uk
242	../employment/111/absolute/gender/sweden	492	../employment/121/rate/gender/year/us
243	../employment/111/absolute/gender/slovenia	493	../employment/121/rate/country/year/female
244	../employment/111/absolute/gender/slovakia	494	../employment/121/rate/country/year/male
245	../employment/111/absolute/gender/uk		
246	../employment/111/absolute/gender/us		
247	../employment/111/absolute/country/female		
248	../employment/111/absolute/country/male		
249	../employment/111/rate/year/female		
250	../employment/111/rate/year/male		

Table F.4.: Sitemap links for dataset lfsi_emp_a

No.	URL	No.	URL
1	../gdp/100/gdp-capita	61	../gdp/110/gdp-total/dates
2	../gdp/100/gdp-total	62	../gdp/110/gdp-total/country
3	../gdp/101/gdp-capita/austria	63	../gdp/111/gdp-capita/dates/austria
4	../gdp/101/gdp-capita/belgium	64	../gdp/111/gdp-capita/dates/belgium
5	../gdp/101/gdp-capita/bulgaria	65	../gdp/111/gdp-capita/dates/bulgaria
6	../gdp/101/gdp-capita/switzerland	66	../gdp/111/gdp-capita/dates/switzerland
7	../gdp/101/gdp-capita/cyprus	67	../gdp/111/gdp-capita/dates/cyprus
8	../gdp/101/gdp-capita/cz	68	../gdp/111/gdp-capita/dates/cz
9	../gdp/101/gdp-capita/germany	69	../gdp/111/gdp-capita/dates/germany
10	../gdp/101/gdp-capita/denmark	70	../gdp/111/gdp-capita/dates/denmark
11	../gdp/101/gdp-capita/euro	71	../gdp/111/gdp-capita/dates/euro
12	../gdp/101/gdp-capita/estonia	72	../gdp/111/gdp-capita/dates/estonia
13	../gdp/101/gdp-capita/greece	73	../gdp/111/gdp-capita/dates/greece
14	../gdp/101/gdp-capita/spain	74	../gdp/111/gdp-capita/dates/spain
15	../gdp/101/gdp-capita/finland	75	../gdp/111/gdp-capita/dates/finland
16	../gdp/101/gdp-capita/france	76	../gdp/111/gdp-capita/dates/france
17	../gdp/101/gdp-capita/croatia	77	../gdp/111/gdp-capita/dates/croatia
18	../gdp/101/gdp-capita/hungary	78	../gdp/111/gdp-capita/dates/hungary
19	../gdp/101/gdp-capita/ireland	79	../gdp/111/gdp-capita/dates/ireland
20	../gdp/101/gdp-capita/iceland	80	../gdp/111/gdp-capita/dates/iceland
21	../gdp/101/gdp-capita/italy	81	../gdp/111/gdp-capita/dates/italy
22	../gdp/101/gdp-capita/lithuania	82	../gdp/111/gdp-capita/dates/lithuania
23	../gdp/101/gdp-capita/netherlands	83	../gdp/111/gdp-capita/dates/netherlands
24	../gdp/101/gdp-capita/norway	84	../gdp/111/gdp-capita/dates/norway
25	../gdp/101/gdp-capita/poland	85	../gdp/111/gdp-capita/dates/poland
26	../gdp/101/gdp-capita/romania	86	../gdp/111/gdp-capita/dates/romania
27	../gdp/101/gdp-capita/sweden	87	../gdp/111/gdp-capita/dates/sweden
28	../gdp/101/gdp-capita/slovenia	88	../gdp/111/gdp-capita/dates/slovenia
29	../gdp/101/gdp-capita/slovakia	89	../gdp/111/gdp-capita/dates/slovakia
30	../gdp/101/gdp-capita/uk	90	../gdp/111/gdp-capita/dates/uk
31	../gdp/101/gdp-total/austria	91	../gdp/111/gdp-total/dates/austria
32	../gdp/101/gdp-total/belgium	92	../gdp/111/gdp-total/dates/belgium
33	../gdp/101/gdp-total/bulgaria	93	../gdp/111/gdp-total/dates/bulgaria
34	../gdp/101/gdp-total/switzerland	94	../gdp/111/gdp-total/dates/switzerland
35	../gdp/101/gdp-total/cyprus	95	../gdp/111/gdp-total/dates/cyprus
36	../gdp/101/gdp-total/cz	96	../gdp/111/gdp-total/dates/cz
37	../gdp/101/gdp-total/germany	97	../gdp/111/gdp-total/dates/germany
38	../gdp/101/gdp-total/denmark	98	../gdp/111/gdp-total/dates/denmark
39	../gdp/101/gdp-total/euro	99	../gdp/111/gdp-total/dates/euro
40	../gdp/101/gdp-total/estonia	100	../gdp/111/gdp-total/dates/estonia
41	../gdp/101/gdp-total/greece	101	../gdp/111/gdp-total/dates/greece
42	../gdp/101/gdp-total/spain	102	../gdp/111/gdp-total/dates/spain
43	../gdp/101/gdp-total/finland	103	../gdp/111/gdp-total/dates/finland
44	../gdp/101/gdp-total/france	104	../gdp/111/gdp-total/dates/france
45	../gdp/101/gdp-total/croatia	105	../gdp/111/gdp-total/dates/croatia
46	../gdp/101/gdp-total/hungary	106	../gdp/111/gdp-total/dates/hungary
47	../gdp/101/gdp-total/ireland	107	../gdp/111/gdp-total/dates/ireland
48	../gdp/101/gdp-total/iceland	108	../gdp/111/gdp-total/dates/iceland
49	../gdp/101/gdp-total/italy	109	../gdp/111/gdp-total/dates/italy
50	../gdp/101/gdp-total/lithuania	110	../gdp/111/gdp-total/dates/lithuania
51	../gdp/101/gdp-total/netherlands	111	../gdp/111/gdp-total/dates/netherlands
52	../gdp/101/gdp-total/norway	112	../gdp/111/gdp-total/dates/norway
53	../gdp/101/gdp-total/poland	113	../gdp/111/gdp-total/dates/poland
54	../gdp/101/gdp-total/romania	114	../gdp/111/gdp-total/dates/romania
55	../gdp/101/gdp-total/sweden	115	../gdp/111/gdp-total/dates/sweden
56	../gdp/101/gdp-total/slovenia	116	../gdp/111/gdp-total/dates/slovenia
57	../gdp/101/gdp-total/slovakia	117	../gdp/111/gdp-total/dates/slovakia
58	../gdp/101/gdp-total/uk	118	../gdp/111/gdp-total/dates/uk
59	../gdp/110/gdp-capita/dates	119	../gdp/120/gdp-capita/country/dates
60	../gdp/110/gdp-capita/country	120	../gdp/120/gdp-total/country/dates

Table F.5.: Sitemap links for dataset tec00001