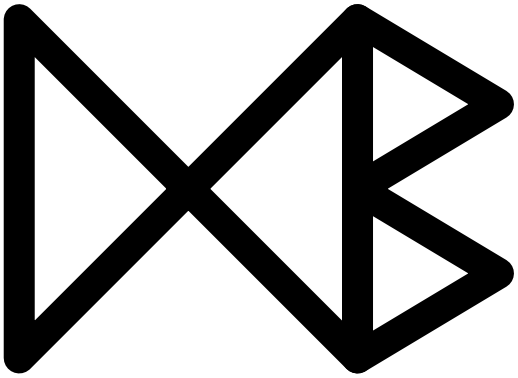

Publishing Danish Agricultural Government Data as Semantic Web Data

Alex B. Andersen, Nurefşan Gür, Katja Hose, Kim A. Jakobsen, and Torben Bach Pedersen

September 15, 2014

TR-35

A DB Technical Report

Title	Publishing Danish Agricultural Government Data as Semantic Web Data
Author(s)	Alex B. Andersen, Nurefşan Gür, Katja Hose, Kim A. Jakobsen, and Torben Bach Pedersen
Publication History	September 2014. A DB Technical Report

For additional information, see the DB TECH REPORTS homepage: ⟨dbtr.cs.aau.dk⟩.

The DB TECH REPORTS icon is made from two letters in an early version of the Rune alphabet, which was used by the Vikings, among others. Runes have angular shapes and lack horizontal lines because the primary storage medium was wood, although they may also be found on jewelry, tools, and weapons. Runes were perceived as having magic, hidden powers. The first letter in the logo is "Dagaz," the rune for day or daylight and the phonetic equivalent of "d." Its meanings include happiness, activity, and satisfaction. The second letter is "Berkano," which is associated with the birch tree. Its divinatory meanings include health, new beginnings, growth, plenty, and clearance. It is associated with Idun, goddess of Spring, and with fertility. It is the phonetic equivalent of "b."

Abstract

Recent advances in Semantic Web technologies have led to a growing popularity of the (Linked) Open Data movement. Only recently, the Danish government has joined the movement and published several data sets – formerly only accessible for a fee – as Open Data in various formats, such as CSV and text files. These raw data sets are difficult to process automatically and combine with other data sources on the Web. Hence, our goal is to convert such data into RDF and make it available to a broader range of users and applications as Linked Open Data. In this paper, we discuss our experiences based on the particularly interesting use case of agricultural data as agriculture is one of the most important industries in Denmark. We describe the process of converting the data and discuss the particular problems that we encountered with respect to the considered data sets. We additionally evaluate our result based on several queries that could not be answers based on existing sources before.

1 Introduction

In recent years, more and more structured data has become available on the Web, driven by the increasing popularity of both the Semantic Web and Open Data, which (unlike traditional for-purchase data sources) is publicly available on the Web and free of charge. Several governments have been driving forces of the Open Data movement, most prominently `data.gov.uk` (UK) and `data.gov` (USA), which publish Open Data from departments and agencies within, e.g., agriculture, health, education, employment, transport, and education. The goal is to enable collaboration, advanced technologies, and applications that would otherwise be impossible or very expensive, thus inspiring new services and companies. While publication of raw data is a substantial progress, the difficulty in interpreting the data as well as the heterogeneity of publication formats, such as spreadsheets, relational database dumps, and XML files, represent major obstacles that need to be overcome [12, 15, 18].

Thus, the Linked (Open) Data movement[1] encourages the publication of data following Web standards and along with *links* to other data sources to provide semantic context, easing the access and interpretation of structured data on the Web. Publishing data as Linked Data (LD) [9,11] entails the usage of certain standards such as HTTP, RDF, and SPARQL as well as HTTP URIs as entity identifiers that can be dereferenced using a browser, making LD easily accessible on the Web, and thus highly usable.

In late 2012, the Danish government joined the Open Data movement by making several raw digital data sets [2] freely available at no charge. These data sets span domains such as environmental data, geospatial data, business data from transport to tourism, fishery, forestry, agriculture etc. To the best of our knowledge, they are currently only available in their raw formats and have not yet been converted and made public as Linked Open Data. We choose agriculture as the main use case, as it is one of the main sectors in Denmark, with 61% of Denmark's land surface being farmland, and thus there is significant potential in providing free access to such data and enabling efficient answering of sophisticated (SPARQL) queries over it.

The goal of this paper is to make Danish governmental Open Data available as Linked Open Data and evaluate the challenges in doing so. As a starting point, we choose some agricultural datasets, transform them into RDF, and make explicit relationships among them using links. Furthermore, we integrate the agricultural data with company information, thus enabling queries on new relationships not contained in the original data.

This paper presents the process applied to transform and link the data, describes the challenges encountered and how they were met, and discusses how the experience can provide guidelines for similar projects. We develop our own ontology, while still making use of existing ontologies whenever possible. A particular challenge is deriving spatial containment relationships not encoded in the original datasets. The resulting LOD data sets are available as a SPARQL endpoint: `http://extbi.lab.aau.dk/sparql` as well as for download: `http://extbi.lab.aau.dk/`.

[1] `http://lod-cloud.net/`

The remainder of the paper is structured as follows, Section 2 provides background and motivation, Section 3 describes our use case datasets, and Section 4 discuss the main challenges. Then, Section 5 gives an overview of our system architecture and Section 6 describes the process and its application to the use case. Section 7 evaluates alternative design choices, while Section 8 concludes the paper and provides directions for future work.

2 Background and Motivation

In the past couple of years, more and more Open Data has become available on the Web. Many communities have invested great effort and many governments have joined the movement so that now various data sets are available on the Web free of charge and accessible for everyone. The goal, especially for governments, is to inspire novel applications and companies, which will eventually increase the wealth and prosperity of the country. Making raw data available in various heterogeneous formats, such as CSV, JSON, PDF, and XML, is convenient for the publisher but it is still relatively difficult to make use of it – especially because the schema is rarely well documented and explained for non-experts. Furthermore, it is not possible to evaluate queries over one or multiple of these data sets.

The quality of Open Data can be categorized by a "five star rating system" [11]:

⋆ *Data is available on the Web in whichever format but with an open license.*

⋆⋆ *Data is available as machine-readable structured data i.e. Excel instead of image scan of a table.*

⋆⋆⋆ *Data is available as ⋆⋆ but in a non-proprietary format i.e. CSV instead of Excel*

⋆⋆⋆⋆ *Data is available according to all the above, plus the use of open standards from W3C (RDF[2] and SPARQL[3]) to identify things, so that people can link to it.*

⋆⋆⋆⋆⋆ *Data is available according to all the above, plus outgoing links to other people's data to provide context.*

To make data given in a raw format as Open Data accessible to a broader range of users on the Web, it can be transformed into Linked Open Data (LOD) [11]. LOD uses unique HTTP URIs to identify entities/things, the usage of HTTP URIs of remote sources creates links between the datasets of multiple sources. Using standards such as RDF as data format and SPARQL as query language enables efficient access to the information. Data published as Linked Open Data corresponds to 5-star data according to the above rating system.

RDF allows to formulate statements about resources, each statement consists of subject, predicate, and object – referred to as a triple. Extending the dataset and adding new data is very convenient due to the self-describing nature of RDF and its flexibility. A triple indicates that there is a relationship between the subject and object. Predicates are also defined by URIs and describe the nature of the relationship. A good practice for choosing predicates is reusing existing ontologies and vocabularies available on the Web.

3 Use Case

Whereas several governments have joined the Open Data movement already several years ago, the Danish government has joined in only recently in 2012 [2]. So, the Danish Open Data has so far only been available in its raw format. Hence, our goal is to make it available as Linked Open Data so that a broad range of users have the opportunity to find and use it. As a starting point for our efforts, we have found the agricultural domain to be particularly interesting as it represents a non-trivial use case that covers spatial attributes and

[2] http://www.w3.org/TR/rdf-primer/

[3] http://www.w3.org/TR/rdf-sparql-query/

can be extended with temporal information. The agriculture industry plays an important role in Denmark, 66% of Denmark's land surface is farmland[4]. By combining the agricultural data with business data, we can process and answer queries that were not possible before as the information was neither freely available nor efficiently queryable.

3.1 Datasets

Late 2012 the Ministry of Food, Agriculture, and Fisheries of Denmark (FVM)[5] made geospatial data of all fields in Denmark freely available – henceforth we denote this collection of data as *agricultural data*. This combined with data about all Danish companies from the Central Company Register (CVR)[6] allows for evaluating queries about fields and the companies owning them. In the following, we will refer to the collection of data from the CVR as *business data*. In total, we have converted 5 data sets provided by FVM and CVR into Linked Open Data. We downloaded the data on October 1, 2013 from FVM [13] and direct download from CVR. In the meantime, the direct file download from CVR has been replaced with a web service.

3.2 Agricultural Data

The agricultural data collection is available in Shape format [5], this means that each field, field block, and organic field is described by several coordinate points forming a polygon.

3.2.1 Field

The Field dataset has 9 attributes and contains all registered fields in Denmark. In total, this dataset contains information about 641,081 fields. Each field has an ownerId that is unique for the owner of the field, an owner may own several fields. To identify individual fields of the same owner, each field has a fieldIdentifier that is chosen by the owner of the field. Other important attributes define the field's polygon, the field's area in hectare, and the crop type.

3.2.2 Organic Field

This data set has 12 attributes and contains information about 52,060 organic fields. Hence, the attributes are very similar to those of the Field data set. It has a few more attributes, among which the CVR attribute is most interesting as it is unique for the owner of the field and references the CVR data set that we explain in Section 3.3. The fieldBlockId attribute describes to which field block it belongs – we explain the concept of a field block in the following paragraph.

3.2.3 Field Block

The Field Block dataset has 12 attributes for 314,648 field blocks and in its original use contributes to calculating and regulating the funds the farmers receive in EU area support. Each field block is defined by a polygon, identified by a unique identifier, and contains a number of fields. Other important attributes [14] describe the area of a field block as grossArea, netArea, and taraArea. Additionally, field blocks have an attribute that defines the year when the most recent aerial photo was taken of it.

[4] http://www.dst.dk/en/%20Statistik/emner/areal/arealanvendelse.aspx
[5] http://en.fvm.dk/
[6] http://cvr.dk/

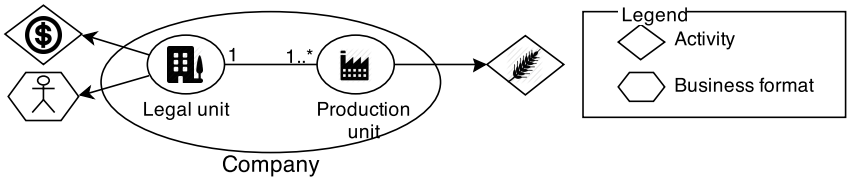

Figure 1: Illustration of legal units and production units

3.3 CVR - Company Registry Data

As mentioned above, the CVR is the central registry of all Danish companies and provides its data in CSV format. There are two datasets available that we refer to as Company and Participant.

3.3.1 Company

This data set has 59 attributes [4] and contains information about 603,667 companies and 659,639 production units. The list of attributes covers a companies' name, contact details, business format, activity, etc. The company dataset contains all legal units in Denmark as well as their production units. A legal unit together with its production units is called a *company*. Each legal unit has a unique identifier, which is also known as CVR number, and each legal unit has at least one production unit. Figure 1 illustrates the relationship between legal units and production units.

3.3.2 Participant

This dataset describes the relations that exist between a participant and a legal unit. A participant is a person or legal unit that is responsible for a legal unit of the company dataset, i.e., a participant is an owner of a company. The Participant data set describes 359,929 participants with 7 attributes. There are three types of participants:

1. Persons who have a Danish CPR number,
2. Companies which have a CVR number, and
3. Entities that have neither CPR nor CVR number.

Figure 2 illustrates the relationship between legal units and participants. Participant 1 is a person (type 1) who owns legal unit A. Participant 2 (type 2) is a legal unit that owns legal units B and C. As participant 2 is a legal unit, it is also described in the company dataset itself – indicated by the dotted line between participant 2 and legal unit D. Participant 3 (type 3) owns legal unit E.

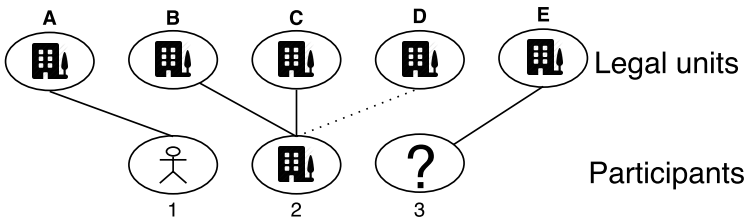

Figure 2: Relationship between legal units and participants

The five datasets mentioned above are connected through foreign keys and can be further connected by spatial joins. Figure 3 illustrates these foreign key relationships as well as natively contained primary keys. The Field dataset is connected to Field Block and Organic Field based on the spatial coordinates of

the polygons that define them. Section 6 describes in detail how we derived this spatial relationships and how we represent it in the Linked Open Data that we created.

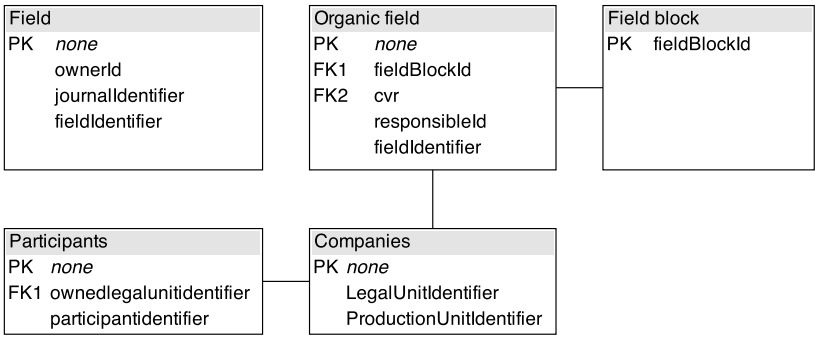

Figure 3: Compact schemas of the 5 datasets showing primary and foreign keys

4 Objectives and Challenges

Our work is driven and inspired by semantic heterogeneity and the lack of technical and structural integrity of government data published in raw format. Hence, currently similar data is stored in very heterogeneous ways and not explicitly related to sources with similar content – data is provided in various file formats, different access protocols and query languages are used, etc. Furthermore, expert knowledge is often needed to understand the data. So, combining multiple similar data sources and querying them efficiently is not only possible at the moment with huge efforts by the user.

The main goal of our work in this paper is making such data available for a greater audience, integrate the data sources by data cleansing and defining cross-references, developing and using ontologies to enhance the structure with meaning, and using Web standards to allow easy access. In particular, to enable queries that have not been possible before, we cleanse and link (Organic) Field data sets to the Company data set so that we can query fields and crops of companies related to agriculture.

To achieve these goals, the particular challenges that we address in this paper are:

- Disparate data sources without common format
- Lack of unique identifiers to link different but related data sources
- Language (Danish)
- Lack of ontologies and their use

As mentioned in Section 3, in our use case data is available in different formats and in some cases does not come with keys that we could use as identifiers. Further, there is only little cross-reference and links between the data sets and no links to Web sources in general. Spatial relationships are even more difficult to represent in the data and querying data based on the available polygons is a complex problem. Furthermore, attribute names in the original datasets are mostly Danish, which makes language a non-trivial challenge that makes linking accurately to external sources difficult.

To make the schema of the data understandable to a greater public and to organize the data accordingly, it is necessary to develop an appropriate ontology that properly describes the structure of the data and gives the structure meaning. Hence, on the ontology level our goal is to create an appropriate oenology and using common predicates from existing ontologies, such as `owl:sameAs`, `rdfs:subClassOf`, `rdfs:subPropertyOf`, `dc:location`. Furthermore, we want to establish links to other datasets and ontologies by defining links to related entities and concepts.

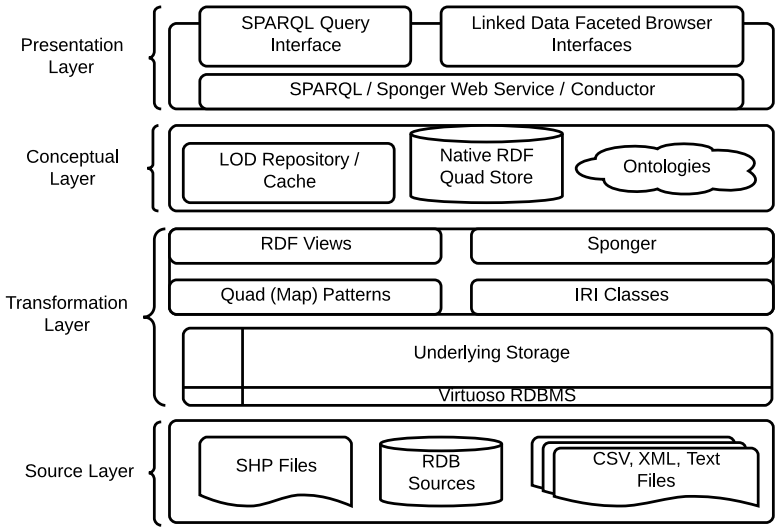

Figure 4: System architecture

5 System Architecture

The system architecture is illustrated in Figure 4. It is organized in four layers: *Source, Transformation, Conceptual, and Presentation*. The Source layer contains the data sources in their original, native format, e.g., XML, CSV, free text, or even RDF. The Transformation layer then transforms the source data into RDF format. A large part of the system is based on the Openlink Virtuoso universal server[7], which we chose a) due to its support for both native and RDBMS-based RDF storage, and its ability to ingest a wide range of data types/formats, as required by our use case; b) for its integrated support for mapping tabular data to RDF with built-in ontology generation and SPARQL Endpoint creation; and c) for its relative ease of use. The transformed data is initially stored in the underlying storage in relational format using Virtuoso's own vtRDBMS. IRI classes are used for converting raw literal values from relational tables into URIs and vice versa, RDF views specify how to rewrite SPARQL queries into SQL queries based on Quad (Map) Patterns and IRI Classes. The Sponger is an "RDFizer" that (semi-)automates the extraction of source data and the mapping of it into Linked Data. In the Conceptual Layer, the LOD repository/cache receives SPARQL queries/updates and tries to match them with the cache. Queries with no cache hit or updates are passed on to the Quad Store, which either executes the statements directly (if the requested data is available in the Native RDF Quad Store) or passes the statements to the RDF views layer. If a query can be answered by the quad store, Virtuoso provides the ontologies to specify and rewrite the query using inference in the given ontology. Finally, the Presentation Layer provides a SPARQL endpoint interface, a Linked Data iFaceted Browser Interface, and an interface based on the Sponger web service. SPARQL 1.1 queries and updates are supported. The SPARQL endpoint is accessible via `http://extbi.lab.aau.dk/sparql`.

6 Data Annotation and Reconciliation

In this section, we outline the process that we followed to publish the datasets described in Section 3. The complete procedure with its main activities is depicted in Figure 5. The initial step is to import the data into the data repository (Import activity) – this is mainly based on scripts that need to be written in dependence on the raw datasets that shall be imported. We refer to the raw data imported into the data repository as

[7] `http://virtuoso.openlinksw.com/`

imported data. All data in the data repository undergoes an iterative integration process consisting of three activities:

Analyze: Gain an understanding of the data and create an ontology
Refine: Refine the source data by cleansing it and converting it to RDF
Link: Link the data to internal and external data

Data that has been through the integration process at least once is referred to as *integrated data*. Integrated data may be published and thus become Linked Open Data that others can use and link to. In the remainder of this section, we will discuss these individual steps in more detail, in particular regarding how we solved the challenges discussed in Section 4 for our use case data. Although we made use of the tools offered by Virtuoso (Section 5), the same results can be achieved by using other frameworks and tools.

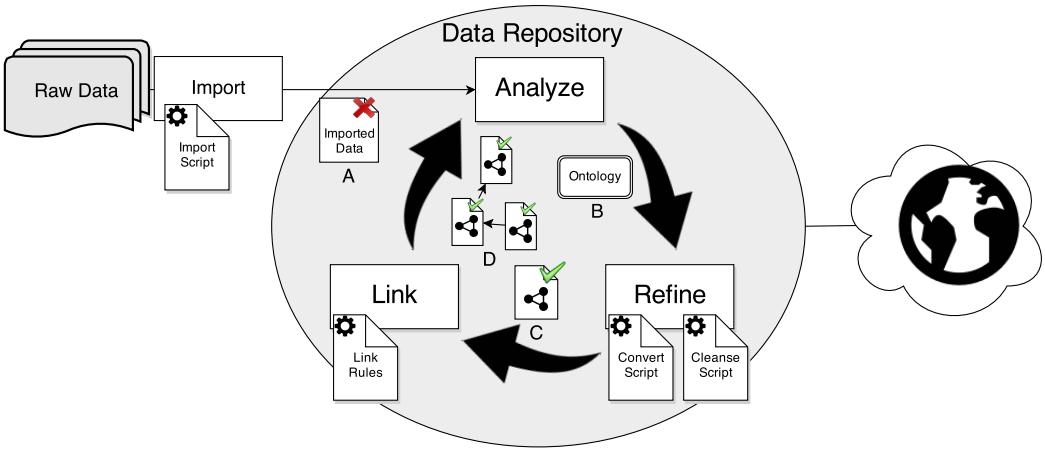

Figure 5: Process overview

6.1 Raw Data Import and Analysis

The concrete method used for importing a dataset depends on the formats of the raw data and how compatible this format is to the one used in the data repository. The *Import Script* is the result of this initial step.

The use case data introduced in Section 3, is available in different raw formats (Shape and CSV) – the agriculture datasets in Shape format and the business data as CSV files. Before importing the data, the attribute fields of the datasets are converted to UTF-8 and illegal characters are removed or escaped. Then, the data is imported into the triple store, which in our architecture corresponds to the Virtuoso RDBMS (Figure 4). We chose to import the data into a relational system first out of convenience because there are formal standardized specifications available for this case [6, 7] and we could benefit from all the tools that are provided for this problem [16] – a direct conversion into triples would also have been possible [8, 10] but is less well supported and sophisticated.

File	Description	Size (in MB)	Tuples
ABO_117_TOTAL	Company dataset	315.2	1,263,306
ABO_117_TOTAL_FAD	Participant dataset	19.1	353,929
ABO_117_DELTA	Company dataset changes	0.85	3,599
ABO_117_DELTA_FAD	Participant dataset changes	0.024	488

Table 1: Statistics for the Company Registry Data

6.1.1 Company Registry Data

As mentioned in Section 3, we used two datasets in this category: Company and Participant. Every day, change sets to these datasets are published that contain updated data entries (companies and production units) that are new, have been changed, or deleted. Table 1 shows an overview of the sizes of the datasets that we considered; the sizes of the change datasets are the average over all datasets that we collected – in total we considered 67 change datasets collected on October 1, 2013, i.e., these change datasets together with the main dataset represent the state of October 1, 2013.

6.1.2 Agricultural Data

As mentioned in Section 3, the three datasets in this category are related based on their spatial shapes and geographical locations. Hence, entities in one of these datasets can reference entities in other datasets if they overlap in their geographic regions. As this information is not directly given in the data by use of foreign keys, we needed to link them using a tool that can determine the spatial overlap. For this purpose, we imported the Shape files into ArcGIS[8], a geographical information system.

After ArcGIS had computed the spatial join, we removed the polygons from the data and only kept the spatial coordinates (longitude and latitude) of a polygon's centroid. In addition, we added another attribute to the tables that represents the spatial relationship as a foreign key. For instance, the field table is extended by an attribute representing a foreign key referencing the primary key of the field block table. The same procedure is applied to the spatial relationships between fields and organic fields as well as organic fields and field blocks.

6.1.3 Analyze

The analyze activity as shown in Figure 5 is the first activity in the iterative process that additionally involves the refine and link activities. The goal of this step is to acquire a deeper understanding of the data and formalize it as an ontology.

The data in our use case is mostly in Danish language. Hence, attribute names were translated to English so that further processing and especially linking to other sources on the Web becomes easier. The formalized ontology represents the understanding of all the data in the repository. If new aspects are discovered in future iterations, then the ontology is updated and the following steps are repeated.

When constructing or improving an ontology, URIs must be chosen with care. The URIs must uniquely identify each class or concept and should remain static so that external sources can link to the data. This true for both the resources in the RDF ontology (classes and properties) as well as resources in the integrated dataset (entities in the instance data). The URIs should be human readable when possible and be accessible through HTTP as described in the Linked Data Principles.

In our use case, we prefix all our ontology-related URIs with `http://extbi.lab.aau.dk/ontology/` and start class names with an upper case character (class Field: `http://extbi.lab.aau.dk/ontology/agriculture/Field`) and properties with lower case (property hasProductionUnit `http://extbi.lab.aau.dk/ontology/business/hasProductionUnit`). It is also important to create meaningful and human-readable URIs for entities in the instance data. We prefix such URIs with `http://extbi.lab.aau.dk/resource/` and create URIs by concatenating candidate key values. An example URI of an entity for Aalborg municipality from the business domain, for instance, is `http://extbi.lab.aau.dk/resource/business/municipality/aalborg`.

[8] `http://www.esri.com/software/arcgis`

6.1.4 Ontology

A crucial part of creating an ontology is to identify sets of attributes that uniquely identify a resource such that a URI can be constructed – this is similar to determining candidate keys in relational database systems. In general, we strive to use existing ontologies to base our own ontologies on. To do this, we make use of predicates such as `rdfs:subClassOf`, `rdfs:subPropertyOf`, and `owl:equivalentClass`, which can link our classes and properties to known ontologies.

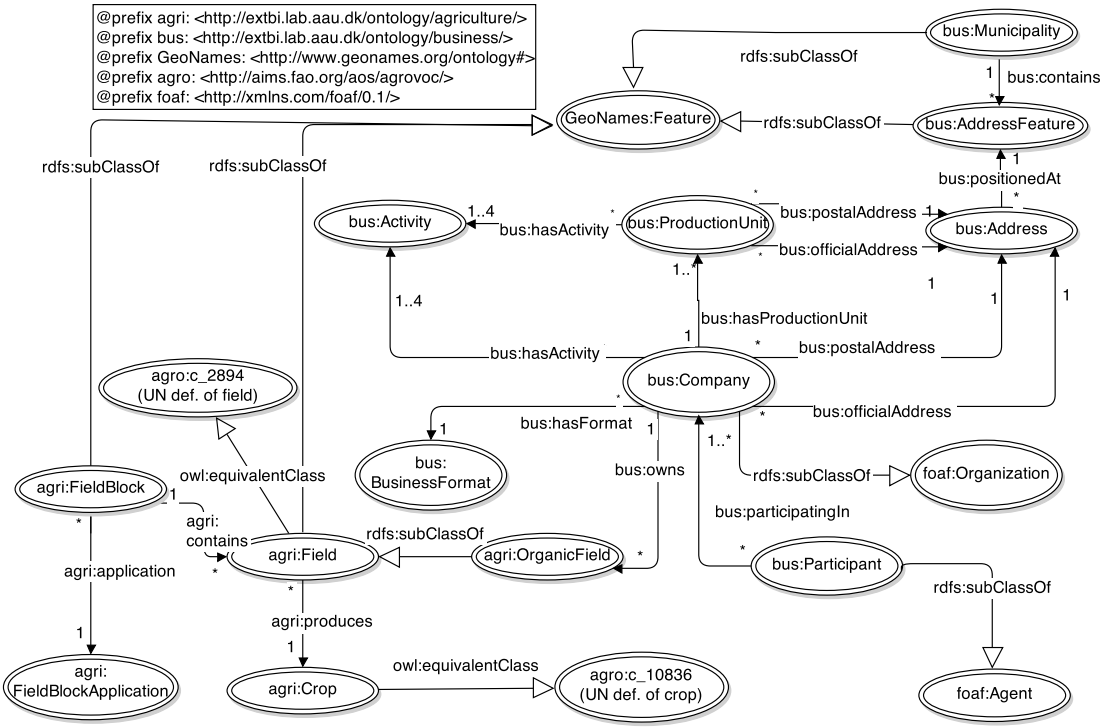

Figure 6: Overview of the ontology for our use case

Figure 6 provides an overview of of the ontology that we developed for our use case with all classes and properties. All arrows are annotated with predicates. The arrows with black tips represent relations between the data instances. The arrows with white tips represent relations between the classes. In creating the ontology we group data into classes and use candidate keys to create a unique URI for each instance of a class. The Field is contained within a FieldBlock, which is expressed with the property `agri:contains` and is determined by a spatial join of the data. OrganicField is a subclass of Field and therefore transitively connects Field to Company. Field is also defined as being equivalent to the UN's definition of European fields from the AGROVOC [17] vocabulary. AGROVOC is a vocabulary covering descriptions and relations for domains of food, nutrition, agriculture, fisheries, forestry, environment, etc.

In total, we make use of 4 external ontologies or vocabularies, respectively.

AGROVOC We have chosen to create and use our own classes for fields and crops even though AGROVOC [17] already defines these, because the URIs used by AGROVOC are poor in human readability. Instead, we define our own and link them to the definitions in AGROVOC using the `owl:equivalentClass` predicate.

GeoNames and WGS The ontologies GeoNames [19] and WGS (World Geodetic System) [3] are used as they offer spatial definitions. We do not link to concepts of WGS in our ontology but use some of the properties it defines; specifically longitude and latitude of fields, organic fields, and field blocks. GeoNames' Feature class is a central part of our ontology that describes a spatial object. We also

link municipality names from our local data to names of places from the GeoNames geographical database.

FOAF The last ontology we are using is FOAF (Friend of a Friend) [1], which we link the business data. As in ontology Figure 6, *Company* is an `rdfs:subClassOf-foaf:Organization`.

6.2 Refinement and Linking

6.2.1 Refine

The Refine activity involves data cleansing and conversion of the imported data. The process is based on the understanding gained in the Analyze activity and consists of data cleansing and conversion. Figure 5 illustrates the data cleansing process where imported data and ontologies are used to produce cleansed data. The adaptations in this corrects the data, creates unique identifiers, and structures the data as defined by the ontology.

In our use case, we implemented data cleansing by using views that filter out inconsistent data as well as correct invalid key attribute values, inconsistent string, and invalid coordinates. In particular, we defined one view for each class defined in the ontology, which eases the task of creating mappings.

After cleansing the data convert scripts are created that transform the cleansed data into RDF. In our use case, we can use mapping tables to convert the data stored as relational tables to RDF dynamically during runtime. For big datasets, this might cause runtime problems so that an alternative is to transform the data into native RDF. The URI's are created during process of mapping to RDF in *Transformation Layer* by IRI Classes in which `iri` function converts a string into a URI(Figure 4).

6.2.2 Link

The Link activity also consists of two steps: internal linking and external linking, which converts the refined data into integrated data that in turn might undergo another iteration or be published as the result of the process. The Link activity materializes the relationships between concepts and classes identified in the Analyze activity as explicit information (triples) in the data.

Internal linking means that we introduce triples that represent the foreign key (and spatial) relationships illustrated in Figure 3 using the URIs that we created for the entities and classes. The example below shows our internal linking of the Field and the FieldBlock classes using the `geonames:contains` predicate.

```
agri:contains rdf:type owl:ObjectProperty ;
        rdfs:domain agri:FieldBlock ;
        rdfs:range agri:Field ;
        rdfs:subPropertyOf geonames:contains .
```

External linking involves linking to remote sources on instance and ontology level. On the ontology level, it is primarily the insertion of triples using predicates such as `rdfs:subClassOf`, `rdfs:subPropertyOf`, and `owl:equivalentClass` that link URIs from our local ontology to URIs from remote sources. We link places mentioned in our company registry data to equivalent places in GeoNames [19] using triples with the `owl:sameAs` predicate – illustrated in Figure 7.

The overall process has provided us with analyzed, refined, and linked data; in total *8,657,607* triples were created. In case the data of the integrated sources changes or we would like to add more sources, we simply need to reiterate through the process (Figure 5). As import, cleansing, and conversion is mostly automated via scripts, updates can be handled efficiently. The linkage of known concepts and data, however, needs to be done manually in the first iteration and become part of the scripts for future iterations.

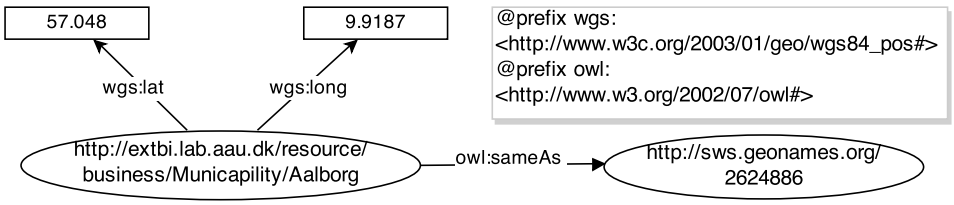

Figure 7: Example: external linking on instance level

7 Experiments

We run an OpenLink Virtuoso 07.00.3203 server on a 3.4 GHz Intel Core i7-2600 processor with 8 GB RAM operated by Windows 7 Enterprise 64-bit SP 1. In the following, we first describe three alternative design choices differing in the materialization of data. They represent tradeoffs between load time and query runtime. We then, present the results of our experimental evaluation in these setups.

7.1 Materialization

The materialization strategies that we considered are: *virtual*, *relational materialization*, and *native*. Figure 8 shows the different paths that data is traveling on starting from its raw format and ending at the user who issued a query. The solid lines represent data flow during the integration process whereas dashed lines represent data flow at query time.

7.1.1 Virtual

In the virtual strategy we import the raw data into a relational database and perform data cleansing based on SQL views (Section 6.2). The RDF mappings that translate the relational data to RDF are formulated on top of these cleansing views. To increase performance, we create a number of indexes on primary keys, foreign keys, and spatial attributes. In Figure 8, using this strategy data flows through the arrows marked with 1, 2, and 3 at query time with no flow during load time.

7.1.2 Relational Materialization

In this strategy, we materialize the above mentioned SQL views as relational tables. We create similar indexes as above but on the obtained tables. In Figure 8, data flows through arrows 4, 5, and 3 – with 4 during load time and 5 and 3 during query time.

7.1.3 Native RDF

In this strategy, we store the data natively in RDF, i.e., we extract all RDF triples from the materialized views and mappings and load them into a triple store. In Figure 8, data flows through arrows 4, 5, and 6 during load time and arrow 7 during query time.

7.2 Performance Measurements

To test our setup, we created a number of query templates that we can instantiate with different entities and that are based on insights in agricultural contracting gained from field experts. Some of them contained aggregation and grouping (Aggregate Query Templates, AQT) others only standard SPARQL 1.0 constructs (Standard Query Templates, SQT).

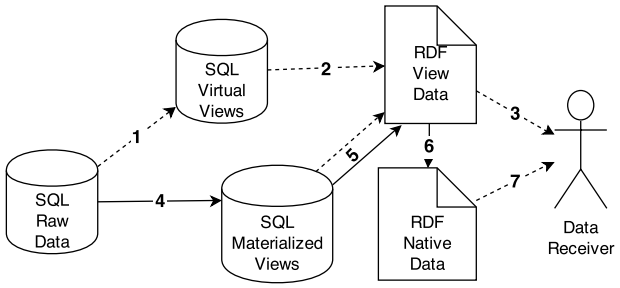

Figure 8: Data flow for the materialization strategies

Listing 1 shows an example AQT query that counts the fields based on the crop they produce. To limit result size, we restrict the area that we are interested in based on a range around a pair of coordinates.

```
SELECT ?crop COUNT(*) AS ?cnt
FROM <http://extbi.lab.aau.dk/resource/agriculture>
WHERE {
  ?field agri:produces ?crop .
  ?field wgs:long ?long .
  ?field wgs:lat ?lat .
  FILTER(?long > [x - 0.5] && ?long < [x + 0.5] &&
    ?lat > [y - 0.5] && ?lat < [y + 0.5]) .
} GROUP BY ?crop
```

Listing 1: AQT 1

For the virtual and relational materialization strategies we measured the load time for each step during loading – the results are shown in Table 2. Table 3 shows the execution times for our query temples on the three materialization strategies. Queries that could not be evaluated because they ran into timeouts are marked by a dash. As we can see, the native RDF strategy is faster than the two other and relational materialized is generally faster than virtual. There is obviously a notable overhead when using views and mappings. On the other hand, the virtual strategy has very fast load time compared to the other strategies since no data has to be moved or extracted – in fact, the cleansing is delayed until query time. The relational materialized is one order of magnitude faster in load time than the native strategy as it again has less overhead during loading.

We can therefore conclude that virtual strategy is well suited for rapidly changing data as it has minimum load time, the materialized strategy represents a tradeoff between load time and query time and is suitable for data with low update rates, and the native strategy decouples the RDF data from the relational data and is very suitable for static data.

8 Conclusion and Future Work

Motivated by the increasing popularity of both Semantic Web and Open Data, and the recent availability of interesting open government data from Denmark, this paper investigated how to make Danish agricultural data available as Linked Open Data and what lessons could be learned from that. Specifically, the paper chose the most interesting agricultural datasets among a range of options, transformed them into RDF format, and made explicit links between the data items to represent the relationships. Furthermore, the agricultural data was integrated with data from the public company register, thus enabling queries on new relationships that were not contained in the original data. Doing this, the paper presented the process for transforming and linking the data. It also describes the challenges encountered and how they were met.

Step	Virtual	Materialized	Native
Data Cleansing	74.92	603.35	603.35
Load Ontology	1.01	1.01	1.01
Load Mappings	8.76	12.35	12.35
Dump RDF	0.00	0.00	4684.82
Load RDF	0.00	0.00	840.04
Total	84.68	616.70	6141.56

Table 2: Load times in seconds

Query	Virtual	Materialized	Native
AQT 1	5.92	3.39	1.04
AQT 2	13.32	7.00	0.23
AQT 3	10.81	7.70	0.05
AQT 4	–	–	0.14
AQT 5	–	20.37	0.86
SQT 1	–	–	2.35
SQT 2	0.09	0.12	0.10
SQT 3	2188.85	1.81	0.40
SQT 4	6.57	2.35	1.63
SQT 5	–	23.79	3.29
Average	370.93	8.31	1.01

Table 3: Runtimes in seconds

Finally, it discussed how to generalize our experiences to provide guidelines for other projects. A new ontology was developed, with reuse of existing ontologies where it was feasible. A particularly interesting challenge was how to derive spatial containment relationships that were not encoded in the original datasets, where existing LOD standards did not provide sufficient support. The resulting LOD data sets were made available as a SPARQL endpoint and for download.

Several directions are interesting for future research. First, new LOD concepts and standards for easily and efficiently creating spatial and spatio-temporal relationships should be developed. Second, novel LOD concepts and standards, including SPARQL support, for enabling business intelligence analytics over Linked Open spatio-temporal data are needed. Finally, we will continue our work of publishing relevant Danish government data as Linked Open Data.

Acknowledgment

This research is partially funded by "The Erasmus Mundus Joint Doctorate in Information Technologies for Business Intelligence – Doctoral College (IT4BI-DC)".

References

[1] The Friend of a Friend (FOAF) Project. `http://www.foaf-project.org/`.

[2] Jakob Bøving Arendt. Denmark releases its digital raw material. `http://uk.fm.dk/news/press-releases/2012/10/denmark-releases-its-digital-raw-material/`, Ministry of Finance of Denmark, October 2012.

[3] W3C-Dan Brickley. W3C Semantic Web Interest Group: Geo. `http://www.w3.org/2003/01/geo/wgs84_pos`, `www.wgs84.com`.

[4] Erhvervsstyrelsen. Record layout: Juridiske enheder og P-enheder. `http://www.cvr.dk/Site/Resources/Files/Media/RecordlayoutABO110.pdf`.

[5] ESRI. Shapefile technical description. *An ESRI White Paper*, 1998. `http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf`.

[6] Marcelo Arenas et al. (eds.). A Direct Mapping of Relational Data to RDF. `http://www.w3.org/TR/rdb-direct-mapping/`.

[7] Souripriya Das et al. (eds.). R2RML: RDB to RDF Mapping Language. `http://www.w3.org/TR/r2rml/`.

[8] Lushan Han, Tim Finin, Cynthia Parr, Joel Sachs, and Anupam Joshi. RDF123: From Spreadsheets to RDF. In *ISWC*, pages 451–466, 2008.

[9] Tom Heath and Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web. Morgan & Claypool Publishers, 2011.

[10] Andreas Langegger and Wolfram Wöß. XLWrap – Querying and Integrating Arbitrary Spreadsheets with SPARQL. In *Proceedings of the 8th International Semantic Web Conference*, ISWC, pages 359–374, 2009.

[11] Tim Berners Lee. Design issues. `http://www.w3.org/DesignIssues/LinkedData.html`, July 2006.

[12] Fadi Maali, Richard Cyganiak, and Vassilios Peristeras. A publishing pipeline for linked government data. In *The Semantic Web: Research and Applications*, pages 778–792. Springer, 2012.

[13] Agriculture Ministry of Food and Fisheries of Denmark. FVM Geodata Download. `https://kortdata.fvm.dk/download/index.html`.

[14] Danish Ministry of the Environment. Markblokkort (datasæt). `http://www.geodata-info.dk/Portal/ShowMetadata.aspx?id=1eb89ebb-f674-4ad1-9e53-d1e252226596`.

[15] Martin G. Skjæveland, Espen H. Lian, and Ian Horrocks. Publishing the Norwegian Petroleum Directorate's FactPages as Semantic Web Data. In *ISWC*, pages 162–177, 2013.

[16] Dimitrios-Emmanuel Spanos, Periklis Stavrou, and Nikolas Mitrou. Bringing Relational Databases into the Semantic Web: A Survey. *Semant. web*, 3(2):169–209, 2012.

[17] Agricultural Information Management Standards. AGROVOC Linked Open Data. `http://aims.fao.org/aos/agrovoc/`.

[18] Boris Villazón-Terrazas, Luis.M. Vilches-Blázquez, Oscar Corcho, and Asunción Gómez-Pérez. Methodological Guidelines for Publishing Government Linked Data. In *Linking Government Data*, pages 27–49. Springer New York, 2011.

[19] Marc Wick. GeoNames Ontology. `http://www.geonames.org/ontology/documentation.html`.