An Architecture and Services for Constructing Data Marts From Online Data Sources. *

Suzanne McCarthy, Andrew McCarren, and Mark Roantree

Insight Centre for Data Analytics, School of Computing, Dublin City University, Dublin 9, Ireland.
suzanne.mccarthy@insight-centre.org, andrew.mccarren@dcu.ie, mark.roantree@dcu.ie

Insight Technical Report Ref. 2018-1, 20th April 2018.

Abstract. The Agri sector has shown an exponential growth in both the requirement for and the production and availability of data. In parallel with this growth, Agri organisations often have a need to integrate their in-house data with international, web-based datasets. Generally, data is freely available from official government sources but there is very little unity between sources, often leading to significant manual overhead in the development of data integration systems and the preparation of reports. While this has led to an increased use of data warehousing technology in the Agri sector, the issues of cost in terms of both time to access data and the financial costs of generating the Extract-Transform-Load layers remain high. In this work, we examine more lightweight data marts in an infrastructure which can support on-demand queries. We focus on the construction of data marts which combine both enterprise and web data, and present an evaluation which verifies the transformation process from source to data mart.

* Research funded by Science Foundation Ireland under grant number SFI/12/RC/2289

1 Introduction

The Agri (short for Agricultural) food industry in Ireland accounts for approximately 5.7% of gross value added, 9.8% of Ireland's trade exports and 8.5% of national employment [2]. Agri companies are increasingly making use of data analytics and data warehouse technologies. Start-ups and research groups are emerging for the purpose of addressing the need for data analytics unique to the Agri sector [1, 8]. Extract-Transform-Load (ETL) as a framework for data integration gained popularity in the 1970's and quickly became the standard process for data integration. Traditional ETL grabs a 'snapshot' of the source data at given intervals, be it daily, weekly etc and updates the data warehouse or data marts [15]. This happens pro-actively at these intervals and all the source data is extracted, transformed according to relevant rules and loaded into a warehouse where it remains until it is needed for user queries. The primary issue with this approach is that it is very costly in terms of both time and resources. As the integration effort is generally significant, it is not easy to modify the ETL system, meaning that the addition of new sources is a major task.

Motivation. The Kepak Group [7] is a leading Irish meat producer and our industry partner and provided the business requirements that drive our case study for this project. Using one of these requirements, we have specified a data mart to be built and populated using our ETL framework. It is often the case that only a small subset of all the data loaded is actually required for a user's query. However, the user must still wait for the loading of the data update in its entirety [6], a burden for data scientists who require near real-time data to make predictions. Traditional ETL is typically run on a batch basis, for example on a weekly or daily schedule. All of the data from the input data source(s) are transformed and loaded to the target warehouse, with no regard for how, when or if it will be used at query-time. This is less than ideal for two main reasons: it is heavy on resources and creates redundancy; this time-consuming process also means that the warehouse is constantly out of date. This is becoming increasingly unacceptable when business operations often require up-to-the-minute data. In [6], researchers focus on shortening the update interval using Active Warehousing. However, doing so increases the burden on the transactional and analytical databases, as it requires doing more frequent extractions, scheduled concurrently with the frequent loads, while still not addressing the issue of redundancy in the data.

Contribution. In this paper, we present a new framework which presents a more lightweight approach to the ETL process. Essentially, it captures source data in a Data Lake which involves a low level of processing. The framework also provides a methodology to integrate unknown data sources, from both enterprise and web environments. With the aid of an (Agri) ontology and transformation templates, the Transform and Load components are used to populate the data marts directly. As the process is largely automatic, it has a significant benefit to end users who can import new data into data marts for analysis and predictions. However, this fast transformation of unknown sources may introduce errors and thus, our evaluation contains validation routines to verify transformed data.

Paper Structure. The remainder of this paper is structured as follows: In §2, we present a review of related research in this area; In §3, the system framework is presented to illustrate the generic nature of this work and to provide an understanding of how new sources can be added and data marts constructed; Using a real world case study from a test application developed with our industry partner, §4 provides a detailed guide to the approach to importing new (enterprise or web) data sources; §5 continues the case study with a description of data mart construction and population; in §6, we present our evaluation which validates our transformation process; and finally, in §7, we present our conclusions.

2 Related Research

As the primary focus of this paper is to build an automated ETL process to combine web and enterprise data into data marts, we focus on similar work in this area. While data integration is now a well-understood problem and ETL architectures comprise mature processes, research into systems that attempt a form of auto-integration between enterprise and web sources, or web sources alone, remains topical.

In [11], the difficulty in automating a process to assign the mappings and transformations required for ETL is addressed. While information about the semantics of the data, the necessary constraints and requirements may be hard to find or unreliable, the design of an ETL process hinges on this information. Therefore, the authors propose making use of an ontology in order to provide this domain-specific information to drive the transformation stage of the ETL process. Our approach also requires a domain-specific ontology to drive the transformation stage of the ETL process but we propose a number of metadata structures drawn from the ontology for a more lightweight ETL process which can support on-demand querying.

In [14], the authors present an ontology-based agricultural knowledge fusion method based on recent advances in the area of data fusion such as the semantic web. Their ontology defines an integrated hierarchy of agricultural knowledge: definitions and relationships. The purpose was to use an ontology to drive a knowledge fusion method to resolve disparity when integrating Agri data sources to create knowledge as well as analyse the data. A tree structure is used to map the hierarchical structure of Agri concepts and their values. We also make use of an ontology but instead combine it with transformation templates to resolve the heterogeneities between enterprise and web sources.

Food and Agriculture Organization of the United Nations (FAO) are responsible for the AGROVOC ontology, a multilingual Agri thesaurus and one of the first initiatives towards the construction of the sophisticated ontology or terminology system in the area of agriculture. It covers areas of interest to the Agri sector such as food, nutrition, fisheries, forestry and environment. Researchers in [10] make use of the AGROVOC ontology by defining an ontology vocabulary to integrate heterogeneous datasets on the areas of climate, soil etc. Their

aim was to address the issue of data disparity and lack of availability of key information - such as soil properties, crop production and climate change - to farmers in Nepal. However, they do not provide a mechanism to integrate known (enterprise) sources with (unknown) web sources.

In [3], researchers focus on reducing latency in the data warehouse as a decision-making tool. Similar to our work, they regard the creation of a real-time data warehouse as a continuous data integration environment and end-to-end automation of the ETL process. They propose an architecture that facilitates a more streamlined approach to ETL by moving the data between the different layers of the architecture without any intermediate file storage. They make use of ETL container - Java applications that execute and monitor ETL tasks. The data is converted into the XML format and the architecture, which is based on J2EE, supports refreshing a data warehouse in close to real-time. However, this is quite similar to traditional ETL in that data is loaded as it becomes available and not as it is needed by the user. Our approach populates data marts from the Data Lake as they are required by the end user.

3 System Framework

There are three layers to the framework which closely adhere to typical Extract-Load-Transform architectures. In these systems, source data resides outside the main system; within the system itself, there are processes to Extract from data sources, Transform to the system's model format, and Load into the warehouse; and an application layer for queries and various forms of analyses. The focus of this work is to present the framework architecture which underpins the construction of data marts from previously unknown sources. This section will provide a high level overview of the different components and their roles in constructing the data marts.

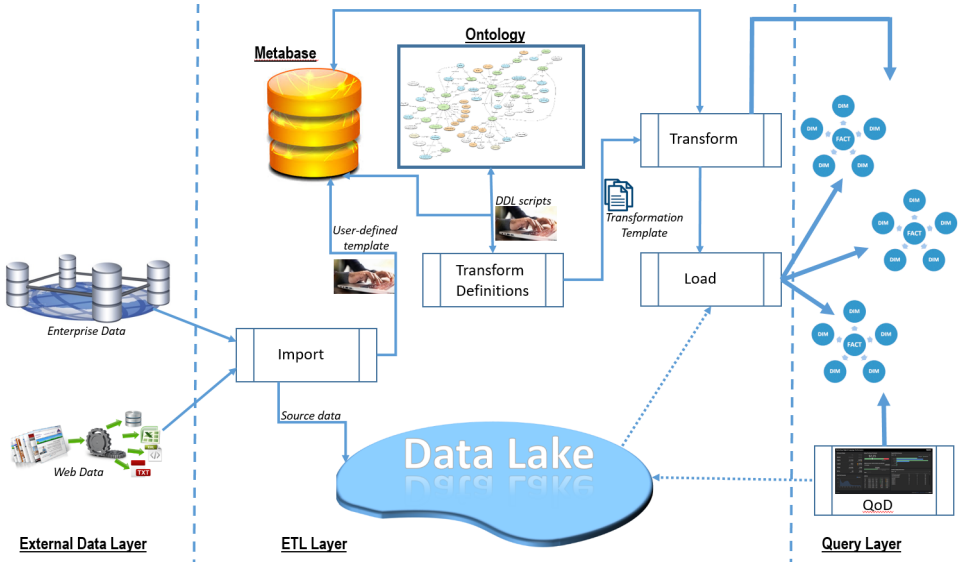

Fig. 1. Framework for Data Mart Construction

3.1 Storage and Transformers

The main components of our system are the Metabase, Ontology and Data Lake.

- A Data Lake is a repository for large amounts of data from multiple sources, wherein the data is stored in its native format. This makes a Data Lake a very lightweight and low-overhead data storage area. In our system, we adapt the concept slightly and perform a very simple transformation on each of the imported datasets. The goal is to prepare the data for loading into the target data marts.
- The Metabase functions as a management tool for the Data Lake. The components of the Metabase are described in the next section.
- Unlike the Metabase, which is generic, the Ontology is domain-specific and holds a complete set of terms and components for the purpose of transforming data unique to the Agri trade industry. Although the ontology is not part of the Transformation process, it supplies the set of terms that are used in the process.

3.2 Metabase Components

New data sources require the user to specify two tools for data transformation. When a new data source is imported, metadata (the user-defined template in Fig 1) is recorded as an Import Template, and updated again when the data is transformed, queried and/or loaded. The *Import Template* then can provide the system with basic details (metadata) of the source data and the *Transformer*

provides the information on mappings between the system's warehouse model and the data source. As the Import Template is a simple capture of the structure of the data source, its manual preparation typically requires less than 10 minutes although new data sources generally require between 1-2 hours of user analysis depending on the complexity of the source data. The Metabase holds the following structures:

- Our common Warehouse model is a star schema model with multiple facts and dimensions. Dimensions are subdivided into two types: static, where there is a finite number of possible values and these are pre-populated before any datasets are introduced; and dynamic, which are populated from the datasets. The Metabase stores the information about each of the facts and dimensions - the attributes of each and the connections between them.
- The Import Template captures metadata about a new data source. It stores information that facilitates the process of converting the data to attribute-value pairs as well as informing the system where to load the data. An example of an Import Template stored in the Metabase is seen in Table 1 in §5.
- Transformers are elements of the Metabase that direct the transformations that take place. A Transformer is created for each individual data source and it stores each of the native terms used in the data and maps each of them to a standard term. Currently a manual process and too large for inclusion here, a method for automating this Transformer will be presented in future work.

4 Data Importation Process

In traditional ETL systems, warehouse updates occur on a periodic basis through the ETL process and normally represents a significant overhead. The various components of Extract, Transform and Load are separated in our framework where the Extract process imports to the Data Lake only but involves the construction of an Import Template and a Transformer. The *Import Template* is necessary to capture new source data to the Data Lake and the *Transformer* is necessary to load the new data marts.

4.1 Initial Analysis

This process begins when a new data source is imported for the first time. To keep the explanation simple, we assume that web data is tabular in format (CSV style), though our system is designed for both JSON and XML models also. Once the web data has been extracted from source, an analysis is performed to determine those attributes necessary for the Metabase.

- Is the file columnar, i.e. are the data attributes at the top of the file with one piece of data per row? If not, does it need to be transformed?
- What is the grain of the data? Monthly? Daily?

- Which columns contain the dimension data and which contain the measure/fact?
- Do all columns contain useful data or are some redundant?
- Do all rows contain useful data or are some redundant?

4.2 Data Lake Storage

The Import process requires the Import Template has already been specified so as to manage Data Lake updates and maintain a record of all data source files along with when they were extracted from the original source (scraped). When data is written to the Data Lake, it is then stored as a set of attribute-value (A-V) pairs. Assuming the dataset is in a tabular format as in Example 1, this is converted to the A-V pairings shown in Example 2.

Example 1. Data in Tabular (CSV) format
A_1,A_2,A_3,A_4
V_{11},V_{21},V_{31},V_{41}
V_{12},V_{22},V_{32},V_{42}
V_{13},V_{23},V_{33},V_{43}

Example 2. Data Lake (A-V pairs) format
S=$\{(A_1, V_{11}), (A_1, V_{12}), (A_1,_{V13})...(A_4, V_{43})\}$

4.3 Transformation

The role of the Transformer is to convert source data into the format of the framework's common model. In this research, we are using the Agri warehouse model described in [9]. All data marts must conform with the common model: they are cubes derived from the base schema. The base schema is a star schema comprised of 24 dimensions and 26 fact tables.

The output of a 5-step process is a dataset which is *ready to load* into the data mart. Given a set S=$\{(A_1, V_{11}), (A_1, V_{12}), (A_1,_{V13})...(A_4, V_{43})\}$

1. Retrieve attribute-value pair
2. Convert(attribute_name)
3. Convert(attribute_value)
4. Create transformed attribute-value pair.
5. Update metadata record to indicate transformation status

After the Transformer has been used to convert the source into the Common Agri Model format, it remains in attribute-value format but will now be comprised of standard terms from the Ontology. Therefore, each of the data sources are ready to be integrated in a data mart.

4.4 Load

Both the Import Templates and the Transformers are used again in the Loading process. To fulfil the Load process, the set of attribute-value pairs are first split into batches, where each batch contains a *single item* of *each* measure and its associated dimension data. These are then used to populate a MySQL command to insert data first to the dimensions and subsequently to the fact table. This order is important for the foreign key relationship between dimensions and facts. The Import Template informs the process of the size of each batch and the target fact table.

The Transformer allocates a dimension to each term and so tells the system which dimension to populate. The Load process initiates a MySQL Insert statement and populates parameters before execution.

5 Data Mart Construction

5.1 Overview

Kepak, our Agri business partner, have a requirement to analyse trade prices of beef commodities and wish to do so using their own enterprise data, combined with selected web based information. The 5 data sources are described below and, in the following section, we provide details for the Eurostat import and transformation process.

- The Eurostat [5] website is the source for EU trade data which allows direct access.
- The United States Department of Agriculture (USDA) [13] publishes Agri trade data figures which can be downloaded in bulk.
- StatCan [12] is the Canadian National Statistics agency and publishes economic, social and census data.
- Comtrade [4] is the U.N. international trade statistics database.
- Kepak Group [7] have exported from their own operational databases.

The following dimensional data is drawn from the above sources:

- **reporter**: the country that is reporting this data
- **partner**: the country with whom the reporter is trading
- **product**: the commodity being traded
- **flow**: the direction of the trade, i.e. imports, exports, re-imports or re-exports

There are some exceptions: the output from the StatCan website does not have a **reporter** column as the reporter is always **Canada** and therefore, this is added to the **geo** dimension prior to loading. All sources additionally have a time attribute but the **date** dimension in the warehouse is static and pre-populated with a fixed range of dates before loading time. Therefore, it is not part of the evaluation process presented in §6. All web sources use **trade weight**, measured in either tons or kg, and **trade value**, in their local currency, as their measures.

The enterprise data uses both of these together with two additional measures: **offcut value** and **yield**.

Beef trade data is extracted from each of these sources. This data will then populate a beef trade data mart with the following dimensions: **geo**, **trade_product**, **trade_flow**, **date**; and measures: **trade_weight**, **trade_value**, **offcut_value** and **yield**. Both the reporter and partner attributes will populate the **geo** dimension. In the event that the source does not contain data for a certain attribute (e.g. Eurostat does not publish yield data), this attribute is recorded as null.

5.2 Details for Eurostat Import

The input for this process (Figure 2) is a file downloaded from the website. Performing the initial analysis on the file, we can see that all the rows contain data but not all of the columns need to be used as the site generates its own codes as well as values, which we do not need. Therefore, the Metabase can be appended with the Eurostat Import Template as seen in Table 1. Example 3 shows a single row of the Eurostat data in the attribute-value pairs which is the format for the Data Lake.

Table 1. Eurostat Import Template

meta-attribute	value
data_address	path/to/eurostat_file.csv
scrape_date	31_01_2018
source	eurostat
measure(s)	trade weight,trade value
grain	monthly
industry	meat
transformed	N
loaded	N
queried	N
num_cols	15
num_valid_cols	8
num_rows	4384
header_start	0
dim_col_list	10,1,3,7,4,5,12
fact_col_list	14
skip_rows	null

Example 3. Eurostat data in Data Lake
S={
(PERIOD,201708),
(DECLARANT_LAB,France),
(PARTNER_LAB,Netherlands),
(FLOW_LAB,EXPORT),

DEC_LAB	PARTNER_LAB	PRODUCT	PRODUCT_LAB	FLOW_LAB	PERIOD	INDICATORS	INDIC_VALUE
France	Netherlands	2011000	CARCASES OR HALF-CARCASES OF BOVINE...	EXPORT	201708	VALUE_1000EURO	828.68
France	Netherlands	2011000	CARCASES OR HALF-CARCASES OF BOVINE...	EXPORT	201708	QUANTITY_TON	198.6
France	Netherlands	2011000	CARCASES OR HALF-CARCASES OF BOVINE...	EXPORT	201709	VALUE_1000EURO	773.63
France	Netherlands	2011000	CARCASES OR HALF-CARCASES OF BOVINE...	EXPORT	201709	QUANTITY_TON	184.1

Fig. 2. Eurostat web output

(PRODUCT,2011000),
(PRODUCT_LAB,CARCASES OR HALF-CARCASES OF BOVINE ANIMALS, FRESH OR CHILLED),
(INDICATORS,QUANTITY_TON),
(INDICATOR_VALUE,198.6)
}

5.3 Eurostat Transformation Process

The input for this process is the Eurostat data in its Data Lake format, as seen in Example 3. For each attribute and value, the Eurostat Transformer supplies the standard term which should replace it (though it is not always the case that the source's original term is different from the standard term). The output of this process is seen in Example 4.

Example 4. Eurostat data after Transformation
S'={
(yearmonth,201708),
(reporter,FRANCE),
(partner,NETHERLANDS),
(flow,exports),
(product_code,2011000),
(product_desc,CARCASES OR HALF-CARCASES OF BOVINE ANIMALS, FRESH OR CHILLED),
(unit,ton),
(value,198.6)
}

5.4 Eurostat Load Process

The data mart constructed from this case study is presented in in Figure 3. The tables to be populated using the data from this data source are:

- fact_trade_monthly: containing the measure and foreign key links to each of the dimensions
- dim_geo
- dim_trade_product
- dim_trade_flow
- dim_unit: pre-populated
- dim_date: pre-populated
- dim_source: pre-populated

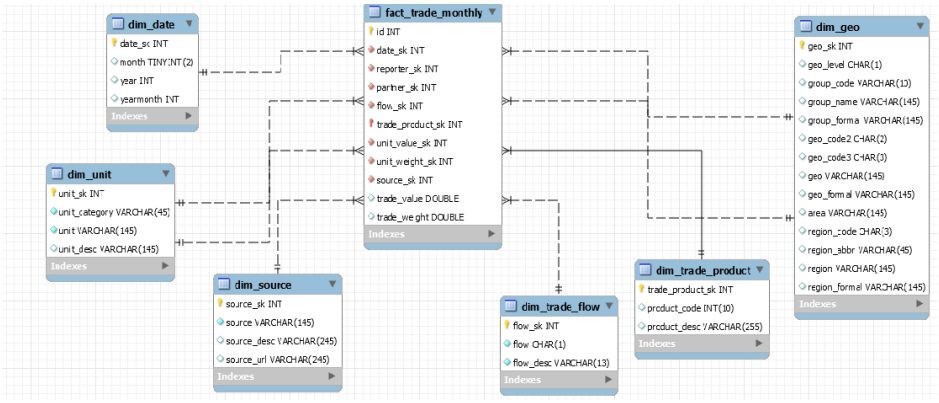

Fig. 3. Trade Price Data Mart

Example 5. INSERT INTO fact_trade_monthly (source_sk, date_sk, reporter_sk, partner_sk, flow_sk, trade_product_sk, unit_weight_sk, trade_weight) VALUES(. . .)

The Loading process begins with a MySQL prepared statement as seen in Example 5. Therefore, taking the output from the Transformation process as input, the Load process generates a batch of MySQL statements to Insert to each of these dimensions before creating the facts. The MySQL statement is populated with the foreign key values for each of the dimension values as well as the measure(s), and is executed as seen in Example 6.

Example 6. INSERT INTO fact_trade_monthly (source_sk, date_sk, reporter_sk, partner_sk, flow_sk, trade_product_sk, unit_weight_sk, trade_weight) VALUES(1,10,315,379,3,168,29,198.6);

6 Evaluation

As with any automatic transformation process, validation of the process itself is crucial. The goal of this paper is on the overall framework which moves data from source to data mart and the focus of the evaluation to validate transformations. This involves two sets of experiments: the first validates that all *measure* data was loaded correctly in its entirety from source to (warehouse model) data mart; the second validates that *dimensional* data underwent valid transformation. Possible states that may result during the Loading of both fact and dimension tables are:

- **Attribute error**: A mismatch between a data point from the source file and the data mart because dimensions were extracted from the Data Lake in the wrong order. This could occur with both dimensional and measure data.
- **Value error**: Missing values where a value is accidentally skipped during the Loading process means that the datasets have different cardinalities. This could occur with both dimensional and measure data.
- **Transformer error**: Errors during creation of the Transformer means that the data is incorrectly NULL. This can only occur for dimensional data.

– **True**: The validation test passes.

Measure Validation: Ordered List Test. The purpose of the Ordered List (OL) Test is to validate measures against 3 criteria: the data mart has not dropped any measure values; duplicate values were not loaded into the data mart; values that should remain unchanged, did so. The OL test runs against every measure in each of the data sources, with logic shown in Algorithm 1. The inputs for this process are OF - an ordered list of measure values obtained from the Fact table, and OS - an ordered list of measure values extracted from the source file. It tests for Attribute errors and Value errors, and returns True if none are found.

Algorithm 1 Ordered List Measure Test

```
function ORDERED LIST TEST(OF, OS)
    if |OF| ≠ |OS| then
        return "Value error"
    for i ∈ 0..|OF| do
        if OF_i - OS_i ≠ 0 then
            return "Attribute error"
    return True
```

Frequency Pattern (FP) Test for Dimension Values. Dimensional data is treated as non-numeric and values may be required to change during the Transformation process. The purpose of this test is to ensure that term transformations are *consistent*. This test is run on every populated dimension and each of the data sources used. Algorithm 2 details the *Frequency Pattern* test, which takes input in the form of a dimension name `d_name`. From here, two sorted lists are constructed: `DV` is the frequency count of all distinct dimensional values found within the source file, sorted in ascending order; `FKV` is the distinct frequency count of all foreign keys found within the fact table, also sorted in ascending order. If the frequencies of the values are different, then either two source original terms have been mapped to the same standard term or a single original term has been mapped to more than one standard term, i.e. term mappings have not been consistent.

The two sorted lists are compared and tested for all of the potential errors - attribute, value and transformer, and returns True if none are found.

Algorithm 2 Frequency Pattern Test

```
function FREQUENCY PATTERN TEST(d_name)
    DV = GET_DIM_VALUES(d_name)
    FKV = GET_FK_VALUES(d_name)
    SDV = SORT(DV)
    SFKV = SORT(FKV)
    if |SDV| ≠ |SFKV| then
        return "Value error"
    for i ∈ (0..|SDV|) do
        if SFKV_i == null then
            return "Transformer error"
        if SDV_i - SFKV_i ≠ 0 then
            return "Attribute error"
    return True
```

6.1 Results Overview

Table 2. Overall results of validation process

id	test	source	attribute	pass/fail	error_type
1	FP	eurostat	reporter	pass	none
2	FP	eurostat	partner	pass	none
3	FP	eurostat	product	pass	none
4	FP	eurostat	flow	pass	none
5	FP	usda	reporter	pass	none
6	FP	usda	partner	pass	none
7	FP	usda	product	fail	attribute
8	FP	usda	flow	fail	attribute
9	FP	comtrade	reporter	pass	none
10	FP	comtrade	partner	fail	transformer
11	FP	comtrade	product	pass	none
12	FP	comtrade	flow	pass	none
13	FP	statcan	partner	pass	none
14	FP	statcan	product	fail	transformer
15	FP	kepak	partner	pass	none
16	FP	kepak	product	fail	transformer
17	OL	eurostat	trade weight	pass	none
18	OL	eurostat	trade value	pass	none
19	OL	usda	trade weight	pass	none
20	OL	usda	trade value	pass	none
21	OL	statcan	trade weight	pass	none
22	OL	statcan	trade value	pass	none
23	OL	comtrade	trade weight	pass	none
24	OL	comtrade	trade value	fail	value
25	OL	kepak	trade weight	pass	none
26	OL	kepak	trade value	pass	none
27	OL	kepak	yield	pass	none
28	OL	kepak	offcut value	pass	none

The results of all validation tests are presented in Table 2, with 6 failures out of 28 tests giving a success rate of 78.57%. On investigating the causes for the failures, all the errors listed previously were found.

- **Attribute error**: For example with USDA, the *trade product* foreign key was assigned to the *trade flow* value. This shows the importance of importing the dimension values in the order in which they are expected to populate the dimensions.
- **Value error**: For example with Comtrade trade value, a data point was skipped as a result of an error during Loading.
- **Transformer error**: For the dimensional data of our business partner, there is a high number of dimensions and not all were imported as the Agri model did not capture this level of dimensionality.

Table 3 shows the events (error_type) that were triggered after failures were recorded, with the pass rate moving to 100% after each action (fix) was taken. The most common error was a transformer error, where the transformer either did not contain a standard term for the source term, or mapped two source terms to the same term.

Table 3. Actions taken for errors in validation

id	test	source	attribute	error_type	fix
7	FP	usda	product	attribute	alter import
8	FP	usda	flow	attribute	alter import
10	FP	comtrade	partner	transformer	manually update transformer
14	FP	statcan	product	transformer	manually update transformer
16	FP	kepak	product	transformer	manually update transformer
24	OL	comtrade	trade value	value	debug loading script

6.2 Detailed Eurostat results

The results for the validation of the Loading process for two Eurostat dimensions and one measure are given here in greater detail.

Fig 4 shows the Frequency Pattern of the *reporter* attribute as extracted from the **dim_geo** dimension for Eurostat. Fig 5 shows the Frequency Pattern of the product attribute as extracted from the **dim_trade_product** dimension for Eurostat. In both cases, it can be seen that the terms in the source file and the transformed data in the data mart have the same frequency pattern and that the source original terms were mapped correctly to a standard term.

For the sake of brevity, the entire results of the Ordered List tests on the measure will not be displayed. Instead we have summed the values of **trade_weight** from the source file (TWS) and the values of **trade_weight** from the fact table (TWF).

$$\sum(tws \in TWS) - \sum(twf \in TWF) = 0$$

The detailed results of the Eurostat case study data shows that the data retains its integrity throughout the ETL process and the original data could, if necessary, be

Frequency data from source file		Frequency data from data mart		
DECLARANT_LAB	frequency	geo_sk	geo	frequency
Netherlands	572	379	NETHERLANDS	572
Fr Germany	468	319	GERMANY	468
France	462	315	FRANCE	462
Spain	392	416	SPAIN	392
Italy	306	337	ITALY	306
Poland	286	393	POLAND	286
Ireland	276	335	IRELAND	276
Utd. Kingdom	248	1487	UNITED KINGDOM	248
Lithuania	208	355	LITHUANIA	208
Austria	180	265	AUSTRIA	180
Belgium	152	272	BELGIUM	152
Denmark	126	302	DENMARK	126
Portugal	120	394	PORTUGAL	120
Latvia	76	349	LATVIA	76
Czech Republic	76	301	CZECH	76
Slovenia	68	412	SLOVENIA	68
Greece	64	321	GREECE	64
Hungary	56	329	HUNGARY	56
Romania	56	396	ROMANIA	56
Luxembourg	44	356	LUXEMBOURG	44
Estonia	40	311	ESTONIA	40
Croatia	34	298	CROATIA	34
Slovakia	26	411	SLOVAKIA	26
Sweden	20	422	SWEDEN	20
Finland	16	314	FINLAND	16
Bulgaria	12	281	BULGARIA	12

Fig. 4. Eurostat reporter dimension Frequency Pattern

Frequency data from source file		Frequency data from data mart		
PRODUCT_LAB	freq.	trade_product_sk	product_desc	freq.
FRESH OR CHILLED BOVINE MEAT, BONELESS	1402	183	FRESH OR CHILLED BOVINE MEAT, BONELESS	1402
FRESH OR CHILLED BOVINE CUTS, WITH BONE IN (EXCL. CARCASES AND HALF-CARCASES, "COMPENSATED QUARTERS", FOREQUARTERS AND HINDQUARTERS)	944	179	FRESH OR CHILLED BOVINE CUTS, WITH BONE IN (EXCL. CARCASES AND HALF-CARCASES, "COMPENSATED QUARTERS", FOREQUARTERS AND HINDQUARTERS)	944
CARCASES OR HALF-CARCASES OF BOVINE ANIMALS, FRESH OR CHILLED	622	168	CARCASES OR HALF-CARCASES OF BOVINE ANIMALS, FRESH OR CHILLED	622
UNSEPARATED OR SEPARATED HINDQUARTERS OF BOVINE ANIMALS, WITH BONE IN, FRESH OR CHILLED	528	174	UNSEPARATED OR SEPARATED HINDQUARTERS OF BOVINE ANIMALS, WITH BONE IN, FRESH OR CHILLED	528
UNSEPARATED OR SEPARATED FOREQUARTERS OF BOVINE ANIMALS, WITH BONE IN, FRESH OR CHILLED	504	173	UNSEPARATED OR SEPARATED FOREQUARTERS OF BOVINE ANIMALS, WITH BONE IN, FRESH OR CHILLED	504
"COMPENSATED" QUARTERS OF BOVINE ANIMALS WITH BONE IN, FRESH OR CHILLED	384	169	"COMPENSATED" QUARTERS OF BOVINE ANIMALS WITH BONE IN, FRESH OR CHILLED	384

Fig. 5. Eurostat product dimension Frequency Pattern

re-created from the data mart by reversing the transforms by the same process. This supports our vision of a lightweight ETL process that makes use of metadata structures, as opposed to a high-overhead traditional data integration process.

7 Conclusions

Data warehouses provide the basis for powerful analytical tools but are expensive to build and support, and tend to be very slow to incorporate new data sources. In this work, we looked at creating more lightweight data marts in order to accommodate new data sources that are selected by end users. We presented a framework which uses a common Agri model to which data marts must conform, and a method to transform data sources into the conforming data mart. A real world case study was specified by our Agri business partner which used their own enterprise data together with 4 selected web sources. Our evaluation used a number of validation techniques to ensure that data extracted from source and loaded into data marts were accurate. An area of work not addressed in this research is the automatic construction of the Transformer which is part of current research. Our overall aim with this platform is to provide Query on Demand data marts which extract from the data lake as required by the user.

References

1. DIT Agriculture Analytics Research Group. http://www.agrianalytics.org/. 2018.
2. Central Statistics Office. http://www.cso.ie/en/index.html. 2018.
3. Bruckner R, List B, and Schiefer J. Striving towards near real-time data integration for data warehouses. In Data Warehousing and Knowledge Discovery, 4th International Conference, DaWaK 2002, Aix-en-Provence, France, September 4-6, 2002, Proceedings, pages 317-326, 2002.
4. UN Comtrade. https://comtrade.un.org//. 2018.
5. Eurostat: Your key to European statistics. http://ec.europa.eu/eurostat/about/overview, 2018.
6. Kargin Y, Pirk H, Ivanova M, Manegold S, and Kersten M. Instant-on scientific data warehouses - lazy ETL for data-intensive research. In Enabling Real-Time Business Intelligence - 6th International Workshop, BIRTE 2012, Held at the 38th International Conference on Very Large Databases, VLDB 2012, Istanbul, Turkey, August 27, 2012, Revised Selected Papers, pages 60-75, 2012.
7. Kepak Group. https://www.kepak.com/. 2018.
8. MCR Agri Analytics. http://www.mcragrianalytics.com/. 2018.
9. McCarren A, McCarthy S, O Sullivan C, and Roantree M. Anomaly Detection in Agri Warehouse Construction. Proceedings of the ACSW, ACM press, pages 1-17, 2017.
10. Pokharel S, Sherif M, and Lehmann J. Ontology based data access and integration for improving the effectiveness of farming in Nepal. In 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), pp. 319-326, 2014.
11. Skoutas D, Simitsis A, and Sellis T. Ontology-driven conceptual design of ETL processes using graph transformations. J. Data Semantics, 13:120-146, 2009.
12. Statistics Canada. https://www.statcan.gc.ca/eng/start. 2018.
13. United States Department of Agriculture. http://www.ers.usda.gov/data-products/chart-gallery/detail.aspx?chartId=40037. 2018.
14. Xie N, Wang W, Ma B, Zhang X, Sun W, and Guo F. Research on an agricultural knowledge fusion method for big data. In Data Science Journal, 2015.
15. Zhu Y, An L, and Liu S. Data updating and query in real-time data warehouse system. In International Conference on Computer Science and Software Engineering, CSSE 2008, 2008, Wuhan, China, pp 1295-1297, 2008.