# Introduction

Data warehousing involves gathering data from various sources, conforming it and storing it as multidimensional cubes to allow OLAP (Online Analytical Processing) or data mining. A data warehouse is a database specifically used for reporting; thus it is usually optimized for answering queries on existing records and not for the insertion or updating of records. Data warehouses tend to be much larger than operational databases, often hundreds of gigabytes to terabytes in size. A dimensional model is often used to provide the logical design used in a data warehouse. The data warehouse generally consists of an ETL tool, a database, a reporting tool and other facilitating tools, such as a Data Modeling tool.

Multidimensional cubes are often considered more appropriate for OLAP applications than schemas normalized to the third normal form (3NF). Multidimensional cubes have important benefits for business intelligence which include understandability and query performance. In query performance, the number of join operations is greatly reduced when using a multidimensional cube as opposed to a relational schema. Furthermore, the query plan can be improved through “star joins” which may be performed faster through indexing or result set size prediction. Finally, dimensional models are generally easier to understand (cite Kimball) allowing the domain specialists and not the IT professionals to perform the analysis of the data (cite litOLAP).

Data warehouses are often used as one of the main components of Decision Support Systems. These systems allow a business to make better decisions by allowing efficient analyses on historical data that has been accumulated over time. However Data warehouses are also used in other fields that generate large amounts of records. In this project, I build a data warehouse over literary texts and places mentioned within them. An OLAP cube is built with the frequencies at which locations are mentioned in sentences of a particular book. Similar work has been done on this area, such as the LitOLAP project (cite here). LitOLAP involves applying Business Intelligence techniques and OLAP in the area of text processing. The data warehouse found in OLAP contains only the information required to build OLAP cubes that facilitate the analysis of specific measures of a given aspect of the texts. For instance, a cube could be built on the frequencies of co-occurring words, word n-grams or analogies. The OLAP cube generated then allows a literary researcher to answer questions over an author’s style, or particularities about the book among others.

My project consists in downloading books from the Project Gutenberg Canada, a website in the Canadian public domain that offers ebooks at no charge. I store the OLAP cube in a relational database. For the ETL process, I used: GATE to perform Natural Language Processing (NLP), Nokogiri an XML parser for ruby, and Pentaho’s ETL tool, Kettle. As my RDBMS, I used MySQL. Finally for my reporting tool and the construction of OLAP cubes I used JasperServer 4.0 which provides an installation of Mondrian and JPivot.

## The Star Schema Design

To build OLAP cubes I used the multidimensional model which is a technique for structuring data so that it is intuitive to business users and delivers fast query performance. The multidimensional model divides the data between measures and context. The measurements are captured by the organization’s business processes and are usually numeric; they are also called facts. The context is represented by the dimensions which help answer the questions who, what, when, where, why and how of a measurement. Dimensional models may be stored as star schemas or cubes. When stored in a relational database platform, they are called star schemas, and when stored in an OLAP structure they are called cubes. To illustrate with an example, consider the multidimensional model for a store that sells fish (figure x). Each cell in this model of three dimensions contains a measure. The measure is referenced by specifying a value in each of the three dimensions.

Multidimensional models are often considered more appropriate for OLAP applications than traditional normalized relational models. Because industry also refers to them as entity-relationship (ER) models, I will refer to them as such. In contrast to multidimensional models, ER models seek to reduce redundancies as much as possible, and are considered better for transactional processing or OLTP applications. The key difference between them is the degree of normalization. While normalized models are completely normalized to third normal form (3NF), the star schema consists of a fact table that is normalized to 3NF, and dimension tables that are normalized to 2NF. Fact tables are normalized to 3NF because the related context is moved to dimension tables. In contrast, the dimension tables are kept denormalized as flat dimensional tables. Often, dimension tables resemble 2NF tables with many low cardinality descriptors. The dimension tables connect to the fact table via foreign keys thus providing the glue of the measures to their context. The tables for my OLAP cube are stored in a relational database and therefore I am essentially designing a star schema.

In my project I picked as my fact table, a table of sentences and places, where a fact is the occurrence of a given place in a sentence. The measure of the fact is the frequency at which the place appears in the sentence. My star schema consists of two dimensions tables of place and sentence, and a single fact table of facts measuring the number of times a place is mentioned in a sentence (see figure x).

### The Place Dimension

As can be observed from (figure x), the place dimension table has a hierarchy that goes from finest granularity at city and coarsest at continent. However, one of the issues of the star schema I propose is that the place mentioned could be at a level different from the lowest level of the hierarchy. Kimball recommends that every fact in a fact table should be at the finest grain of the hierarchy of all its respective dimensions. The nature of the problem cannot comply with this advice. A sentence may mention a place that could be a continent or a country without specifying the country or city in question. Thus, I have some facts that cannot be drilled down upon to finer details. The issue may be resolved by creating a dummy value, namely, *unspecified* to fill in spots where a member would be missing. While you can still drill down to the value of unspecified values, this should indicate that a fact is not meant to be interpreted at that level (see figure x).

Yet another issue I encounter with the place dimension is that there could be ambiguity into what country should be the parent of a city name that could be associated to different countries. For instance, London is the name of a city in England and in Canada. In which of these two countries is the mentioned city located? To answer this question more context from the book would be necessary. This job could potentially be done through further natural language processing. Yet how much context do we need to figure out to which country does the mentioned city pertain? The answer to this question is relative to how much context the book provides and the process used for determining that context. Another method to deal with this issue is through allocation. I could assign a fraction of the measure of each sentence to each of the possible rollups where a given member could belong. For example, if given the city London in a sentence, two facts would be created. One would roll up to England, and the other would roll up to Canada. The fraction of frequency Canada and England would get could be determined through their probabilities. This approach, however, could only be meaningful for certain queries where the data is summarized at the higher levels of the hierarchy. Querying for which specific books mention London in Canada, and which mention London in England would not be possible.

## The sentence dimension

The sentence dimension table also has a hierarchy that goes from finest granularity at sentence and coarsest at author occupation. For the table of sentences I maintain a surrogate key to provide better time performance.

Sentences can be rolled up to books and books can be rolled up to authors. However, because two or more authors may write a particular book, a many-to-many relationship exists between both attributes. Ideally, a hierarchy consists of a single parent member and multiple children members forming a one-to-many relationship. Because two or more author members are parents of a particular book member this relationship becomes a many-to-many relationship. Such a relationship between attributes could lead to double counting and other unwanted side effects. There are various ways to address this problem, some of which I address in this report:

Repeating attribute: This method involves creating an additional column or attribute for each additional parent member that a child member has. For instance, if I have two authors for a particular book then I would have a field for the first author, and a field for the second author. One inherent problem from this approach is that we do not know how many fields we need to ensure that all authors are accounted for. There is also a lot of space wasted because most books have only one author.

Allocation: This method involves repeating each fact for every author of a given book with an additional Sentence ID. Then each author gets a fraction of the frequency at which a location is found in a sentence. For instance, suppose a given book has four authors and the city Saint John is mentioned in three different sentences of the book. Each fact of the table for the three sentences mentioning Saint John would be represented in four different facts, each having a fourth of the frequency. When adding up the frequencies at which Saint John occurred in this book, I would get a total of three sentences. Yet if the measure is not divided by the number of authors, this would lead to counting each fact 4 times, leading to a misleading 12 times where the city Saint John is mentioned.

Bridge Table: This method consists in creating

The table of places has an issue, in which the place that is mentioned may be a continent, a country or a city. If all the places mentioned are cities, then one could have a nice hierarchy, but because in a sentence, one may refer to a specific level of the hierarchy. I am not sure how to best deal with this in my star schema.

Adding another dimension: This last method evades the many-to-many relationship that the Book could have with

When you have the author as a separate dimension we can better understand manage the many to many relationship between authors and books. However because most books are not written by the same author the cube would become even sparser. Additionally, we would have a cube that wouldn't really respond to the query of how many sentences mention places in a book with a table that lists the number of sentences per author. Logically we would have to have a way to determine which sentences were written by one author and which by the other. If we don't know this then we could give both authors credit for their sentences.

When I have author as a separate dimension the many to many relationship between authors and books I managed better. However because most books are not written by the same author the cube would become even sparser. Additionally, we would have a cube that wouldn't really respond to the query of how many sentences mention places in a book with a table that lists the number of sentences per author. Logically we would have to have a way to determine which sentences were written by one author and which by the other. If we don't know this then we could give both authors credit for their sentences.

This was one of the things I had thought about. I explored two possibilities for it:

* Having the finest grain be the frequency of a place in a given sentence by a given author: AuthorID x SentenceID xPlaceID 🡪 Frequency
* Having the finest grain be the frequency of a place in a given sentence: SentenceID xPlaceID 🡪 Frequency

I have created a star schema that builds a fact table of the presence of a place of a given sentence. The fact table is associated to a sentence and place dimension.

## The ETL Process

Populating the data warehouse from data sources involves a process of three main phases: Extracting the data from each source, transforming it to conform to the warehouse schema and cleaning it, and loading it into the warehouse. This process is known as ETL (Extracting, Transforming and Loading).

The data extraction step consists in bringing data from different sources into a database where they can be modified and incorporated into the warehouse. In my project, I extracted the information from a single data source, the Gutenberg Canada website (cite). Before I extracted books from this website, I first downloaded the HTML file onto local disk and cleaned it of unnecessary text and tags that could confuse my pattern matching regular expressions for eventually extracting information about the books from it. To extract the books from the website I used an HTML parser that identified text files links in the website and downloaded them into a location on disk.

The transformation process uses a set of rules and scripts to transform the data from an input schema to a destination schema representation. Most of the work in my ETL process was done on this stage. Data cleaning is also an important part of the transformation stage and it consists in fixing errors and differences in schema conventions. These differences may result in inaccurate query responses and consequently inaccurate mining models. In my project, I had to remove some irrelevant content from the books. I did most of this with ruby scripts that used pattern matching rules to gather only the necessary information. Some preliminary common information found in the text files was also eliminated as it was not particularly relevant to the book.

## Natural Language Processing

One of the most critical steps in my transformation process was that of doing Natural Language Processing (NLP) for the texts I had downloaded. I did part of this transformation process with GATE (cite) , and part of it with the WhatLanguage ruby gem (cite). Both are open source free software projects. Initially, I use WhatLanguage to determine the language a text is written in. If the text is written in English I keep it, and if it is not I discard it. Once I have all texts that are written in English, I use GATE to annotate the texts. GATE builds an XML file of the English texts where sentences and places are tagged. Sentences are tagged using GATE’s default sentence splitter, and the places are annotated using a Gazetteer. I illustrate the transformation of a text to xml annoted text by GATE in the following listing:

Hello this is a test file for ANNIE. I went to Halifax last week. I live

in St John, and I am studying at the University of New Brunswick. I also

have a supervisor whose name is Daniel in Montreal. My supervisor's name

in St John is Owen.

<Root>

<Sentence>Hello this is a test file for ANNIE.</Sentence>

<Sentence>I went to <Location>Halifax</Location> last week.</Sentence>

<Sentence>I live in <Location>St John</Location>, and I am studying at the

University of New Brunswick.</Sentence> <Sentence>I also have a supervisor

whose name is Daniel in <Location>Montreal</Location>.</Sentence>

<Sentence>My supervisor's name in <Location>St John</Location> is

Owen.</Sentence>

</Root>

## Incremental Updates to the Data Warehouse

Though I have not planned implementing this part of the ETL process, I describe it in this section. In order to perform ETL of the database I would first take the latest html file and do a diff with the previous html file that was added to the database. Using diff I would determine what contents are new in my file, and generate a new file with only the new content. After that, I would attempt to run the fixFile.rb script to fix any manual inconsistencies that could hurt the ETL process. Then, getSources can obtain the relevant information on the authors.

Abstract

A data warehouse is a database specifically used for reporting; it is optimized for reducing query response time and not for the insertion, updating or deletion of records. In businesses, they are used as one of the principal components of decision support systems. Decision support systems often involve OLAP cubes that allow the efficient analysis of the historical data acquired from the data warehouse. Data warehouses and business intelligence techniques such as OLAP cubes are also used in other fields where there is a large amount of data. The purpose of this project was to apply data warehousing and OLAP in the field of literature. A data warehouse of places mentioned in books gathered from the Gutenberg Canada website is built along with a corresponding OLAP cube that facilitates analysis.