# 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 to answer queries on existing records and not for insertion or updates of records. Multidimensional cubes are often considered more appropriate for OLAP applications as opposed to 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 because dimensional models are easier to understand OLAP is meant for domain specialists and not IT professionals

In this project we build a small data warehouse on the locations mentioned in books. Similar work has been done on this area such as the LitOLAP project. The project consists in applying Business Intelligence techniques of Data warehousing and OLAP to the area of text processing. When a DW is built over literary data, not all the information is kept, but only information required to build the OLAP cubes. They build a word dimension that allows roll-up by stem suffix, and (several layers of) WordNet (cite here) hypernyms. Kaser et al. do not believe

where cubes are built for texts (cite here). Our project consists in downloading books from the Project Gutenberg Canada, a website in the Canadian public domain that offers ebooks at no charge.

To implement this project we used ROLAP where we used as our RDBMS MySQL and JasperServer which provides the Mondrian web application for the creation of OLAP cubes. For the ETL process we used ruby scripts as it was difficult to use Pentaho’s data integration tool to populate the data warehouse with the type of data we had.

## The Design of the Database Schema

One of the things we had to think about was the grain of the table we intended to use in our data warehouse.

I thought of putting sentences by time by author by sentence id and by locations. Also thought of the possibility of dividing the frequency of a word, when there was more than author for a book, among the number of authors.

I have created a DB schema that treats the sentences as facts, where they may be associated to the authors and book dimensions. However, because author and book have a many-to-many relationship. It is difficult to maintain these two dimensions. There are some possible solutions to this problem, which include:

1. Allocating. We give each author a fraction of the frequency of the locations. Thus, when you add up the sentences by book, you get a total, without double counting.

I also wondered on the possibility of allocating the frequency of each author by ½. Why not instead divide by the number of authors when aggregating by book.

## The ETL Process

Populating the datawarehouse from independent data sources involves a process of 3 main phases: Extracting the data from each source, transforming it to conform to the datawarehouse 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. For all the steps of the ETL process we used ruby scripts for the as the data was highly unstructured and the options weren't straightforward on how to move it from the sources to the RDBMS. In the guides provided by Pentaho, we found that it was meant for nicely structured data like the ones found in XML databases where the schema is already defined.

So there is an advantage of scripting the ETL Process using pentaho's ETL tool, Data Integration. When I look at the options for XML parsing in ETL, it does not support unstructured XML Data. I notice that the XML files that they use as examples are neatly organized and they are easy to transform to realational form. For the XML file I produce with gate, there isn't a straight-forward way to do this with the Data Integration tool.

In order to do 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 our 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.

## Natural Language Processing

Ok so far we have gate working alright but it is going to be too difficult to process those files without keeping my computer inactive for a while. The best approach is to have a few XML files from the gate thing. Now the next step should be to change the ruby script so that it works with more than one file. The relational tables should be changed to include the bookid from the sentence table. As these two form the primary key. I have been thinking of putting sentences by time by author by sentence Id by location. I was also thinking if instead of allocating the frequency of each author by 1/2. Why not instead divide by the number authors when aggregating by book.

s

Ask owen what pronoun to use whether I or we?