**600.315 Databases Final Project Phase II**

**Team Members**

Joon Hyuck Choi (jchoi100)

Joo Chang Lee (jlee381)

**Section: 315**

***Prior Note:*** *A lot of the information required for Phase II has already been written in the slides we attached. Please see the pdf slides “315\_jchoi100\_jlee381\_db\_slides.pdf” for more details.*

1. **Project Description**

This project focuses on extracting interesting trends and information about artwork stored/displayed at the Museum of Modern Art. This project uses additional information on countries and historical events (focused on wars) to compare and analyze artwork characteristics (such as size) during wars and on certain criteria of artwork (such as the golden ratio).

*(Slides on this topic is in the pdf slides)*

1. **Some Changes Since Phase I**

The initial goal of this project included making analysis on changes of prices of artwork over time, the correlation between the sizes of artworks and their prices, and many more queries related to artwork and their prices. However, after days of futile search, the team came to realize that finding auction data is relatively difficult as they can be valuable pieces of data for certain people. Thus, the team decided to focus on digging deeper with the artwork/artist/war/country tables and extract interesting facts from them.

1. **Data Source / Extraction Method**

The MoMA information (which we used to make our Artist, Artwork, Film, WorkedOn tables with) came from MoMA’s own GitHub page. Information on wars came from the Correlates of War Project website. Country demonyms came from Wikipedia.

For about half of the tables, we wrote our own code to write .sql files for the table within our Java parser programs. (To accomplish this, we made Java classes for each object that would occupy a tuple in a table—e.g. Artwork.java to populate the Artwork table—and wrote a method called **toSqlStatement() : String** inside each class.) For the other half, we used a web-based program provided by *http://www.convertcsv.com/csv-to-sql.htm*.

*(Slides on this topic is in the pdf slides)*

1. **Software/hardware platform**

MySQL dbase (JHU ugrad machine)

1. **User’s Guide**

The sql files required to make tables needed to run our queries have already been produced and lie within our Java project directory downloadable on the GitHub address mentioned below in Section 9. However, if one desires to run our parsers to test if the programs work, one can run the Java programs to create sql files for themselves. Please note that the team wrote some parsers without any sqlWrite operations. Thus, only a few of the parser programs will create .sql files for the tables they just created. Again, all the sql files required to create tables for this project are already given in the project directory.

In order to see the HTML/PHP websites on which the user can interact with our system, please open the *“index.html”* file in the project directory. We have written sample interfaces for three of our more complex queries. Code we wrote for this is copied into the project directory. However, the actual php and html files users are directed to when clicking the links in *index.html* actually sit inside jchoi100’s ugrad dbase.

1. **Areas of Specialization**

The project focused on 1) Parsing information from disorganized tables and data sources and 2) dealing with large amounts of data with UTF-8 encoded characters.

*(A slide on this topic is in the pdf slides)*

1. **Analysis of System’s Limitations / Possibilities for Improvement**

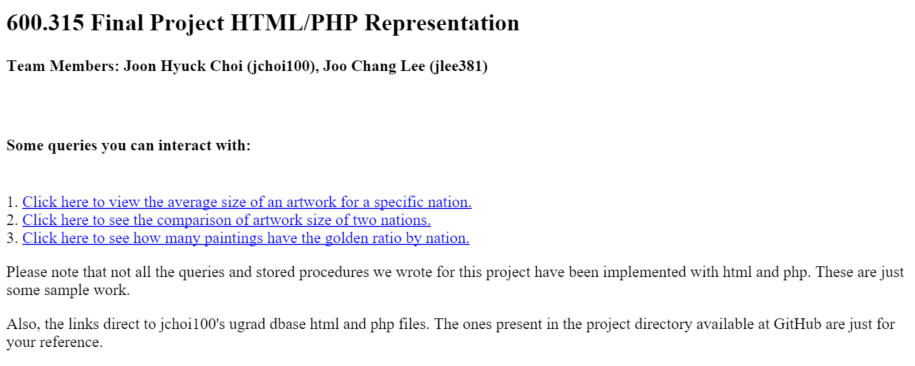
One possibility for improvement with this project is to further extend the system’s capabilities of recognizing certain words such as people’s names. For instance, given the query *“Who donated to MoMA the most?”* the program could look up the Artwork table’s CreditLine column and extract information on people’s names. Most instances in the CreditLine column contain wordy expressions that include lengthy descriptions of the context and method of donation/purchase. Thus one way of dealing with queries like the one mentioned above is to go with the natural language processing method in which the program will try to process a string such as “Gift of Max Underwood”, see given the word “Max Underwood”, and figure out what these words actually mean. Another way of processing such queries is to normalize the CreditLine column in the preprocessing stage so the information can be readily returned to the user.

*(A slide on this topic is in the pdf slides)*

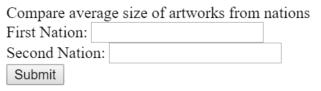
1. **Output Samples**

This is a sample from the index.html file in our project directory.

**1) index.html**



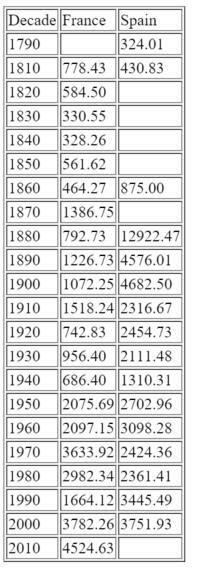
**2) CompareSizeByNationDecade.html**



**3) CompareSizeByNationDecade.php**

>> input1: France

>> input2: Spain

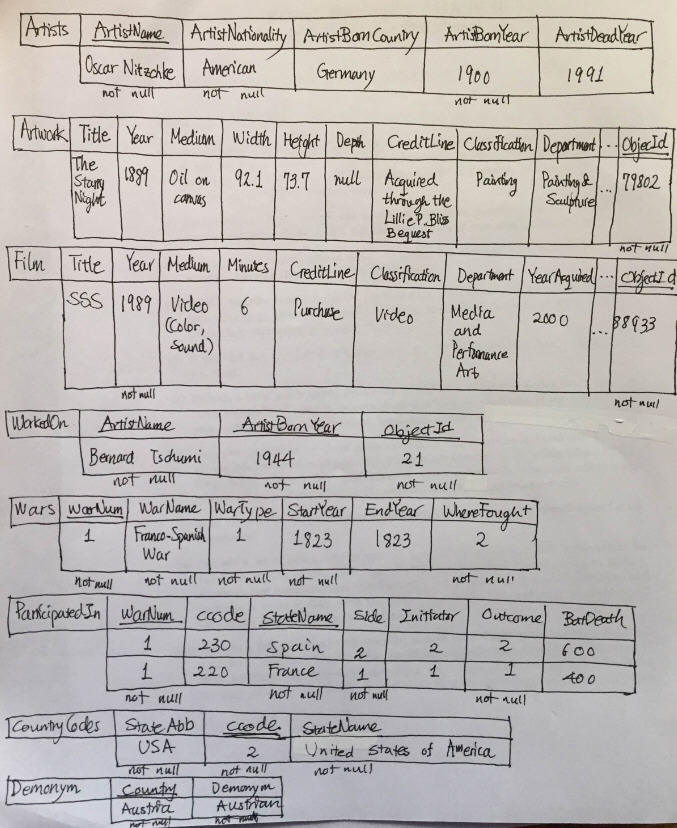


*(Slides on this topic is in the pdf slides)*

1. **Database Submission**

All Java parser files, raw data .csv files, and produced .sql files are in the GitHub repository: [*https://github.com/james1022/databases.git*](https://github.com/james1022/databases.git)*.* The demo slides which include some information requested for this document is also in the GitHub repository.

The following is the relational table specification of our database tables.



1. **SQL Code and Input**

The SQL code we have written for this project are all inside the “315\_jchoi100\_jlee381\_sql\_queires.txt”.

All the .sql files used to create tables are inside the project directory. (e.g. artists.sql, artwork.sql)

*(Slides on this topic is in the pdf slides)*

***Thank you!***