Phase III: Analysis, Design, and Implementation of the PresidentialQuery Database System

Andrew Butler
Justin Shen

2. Environment and Requirement Analysis

2.1 Purpose of the Document and Project

The purpose of this document is to describe the design and process of creating the database project *PresidentialQuery* as well as the result of the described implementation. This project is developed as part of the CMSC 424 course and this document is intended to present the work done for this project to the instructors of the course. As part of this project, a system will be created to allow users to retrieve information about past United States presidential election. The system is composed of an ETL (extract-Transform-Load) tool, a Presidential Election Database (PED), and a dynamic web interface. The ETL tool will extract relevant data concerning the election from specific web sources, standardize and organize all the data retrieved, and then load the sanitized data into the PED. The PED is a relational database containing diverse information and data regarding the United States presidential election from 1781 to 2008. Lastly, the web interface will serve as a platform to allow users to retrieve data from the PED. Inputs from the web interface is transformed into SQL queries in order to retrieve the relevant data to display back to the users. Relevant design diagrams such as the top-level information flow chart, task/ subtasks form, etc. will be included in the document.

The remainder of this document is organized as follows: The remainder of section 2 outlines the assumptions and limitations of the PED system and also assesses and address the information needs of the system. Section 3 will then address the preliminary tasks and subtasks for the implementation of the system and will provide the relevant task forms and data documents to show the preliminary design of the project.

2.2 Scope

The scope of this project will be limited to three major tasks. The first task consist primarily of researching web sources for data relating to the United States presidential election and designing relations suitable for organizing the data in the database. The second task requires implementing or using ETL tools to actually retrieve the data and load them into the PED. The last major task involves making the data stored in the PED accessible through a web interface. The web interface should allow users to search up presidential election information such as duration in office, political party affiliation, audio-visuals snippets, number of electoral votes won, etc. Creating a functional and easy-to-use interface as well as presenting the results of the searches in a well formatted manner will be the main goal of this task.

2.3 Assumptions

The following assumptions are made for the purpose of this project:

- The user read and understand English
- The user can navigate an average web page with minimal trouble
- Data pulled from the websites listed is accurate
- The database server can handle a reasonable number of requests

2.4 Problems and Solutions:

There are a number of technical, designs, and conceptual problems related to the implementation of the PED. These problems are detailed below along with possible solutions or process for dealing with the problems:

Problem: Gathering data
Solution: Research online sources for relevant data

Problem: Extracting the data
Solution: Create script to interface with the online source and select the data required for the database

Problem: Verifying the veracity and integrity of the data
Solution: Examine the metadata and description associated with the data source to and cross referencing multiple data sources to ensure the data extracted is of the highest quality.

Problem: Web server is needed to host the web interface.
Solution: Research possible web servers that will allow for efficient and flexible hosting of the web interface.

Problem: Identifying and create relevant results to produce from the data in the database
Solution: Research what information would most interest potential users. Then perform queries on the database to create the results.

Problem: Making the results accessible for the end user
Solution Format the results from the database in a way that is both concise and readable for the end user.

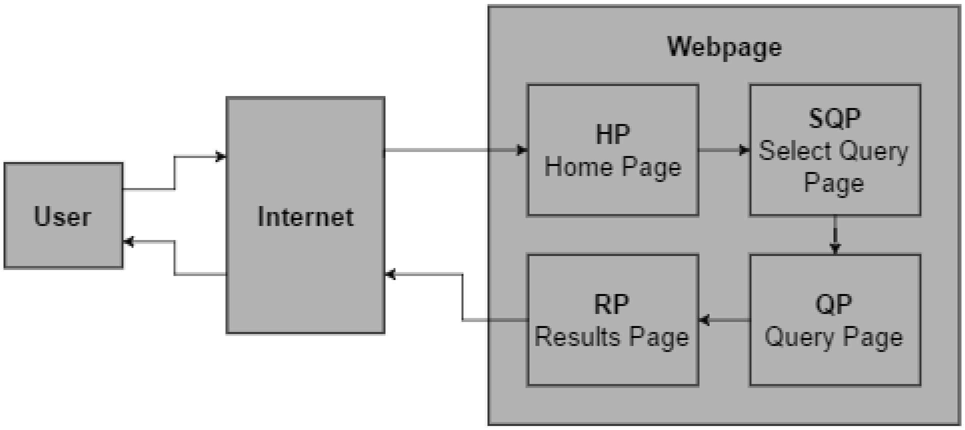
3. System Analysis and Specification

3.1 Description of Procedures

PresidentialQuery's system operates by allowing a user to connect the webpage and submit a query based on various criteria, which will allow them to find data relevant to the criteria they provided. The general description of information that can be found on the web page is as follows. Information on the most involved candidates in any presidential election season and on the election season itself from 1781 to the present.

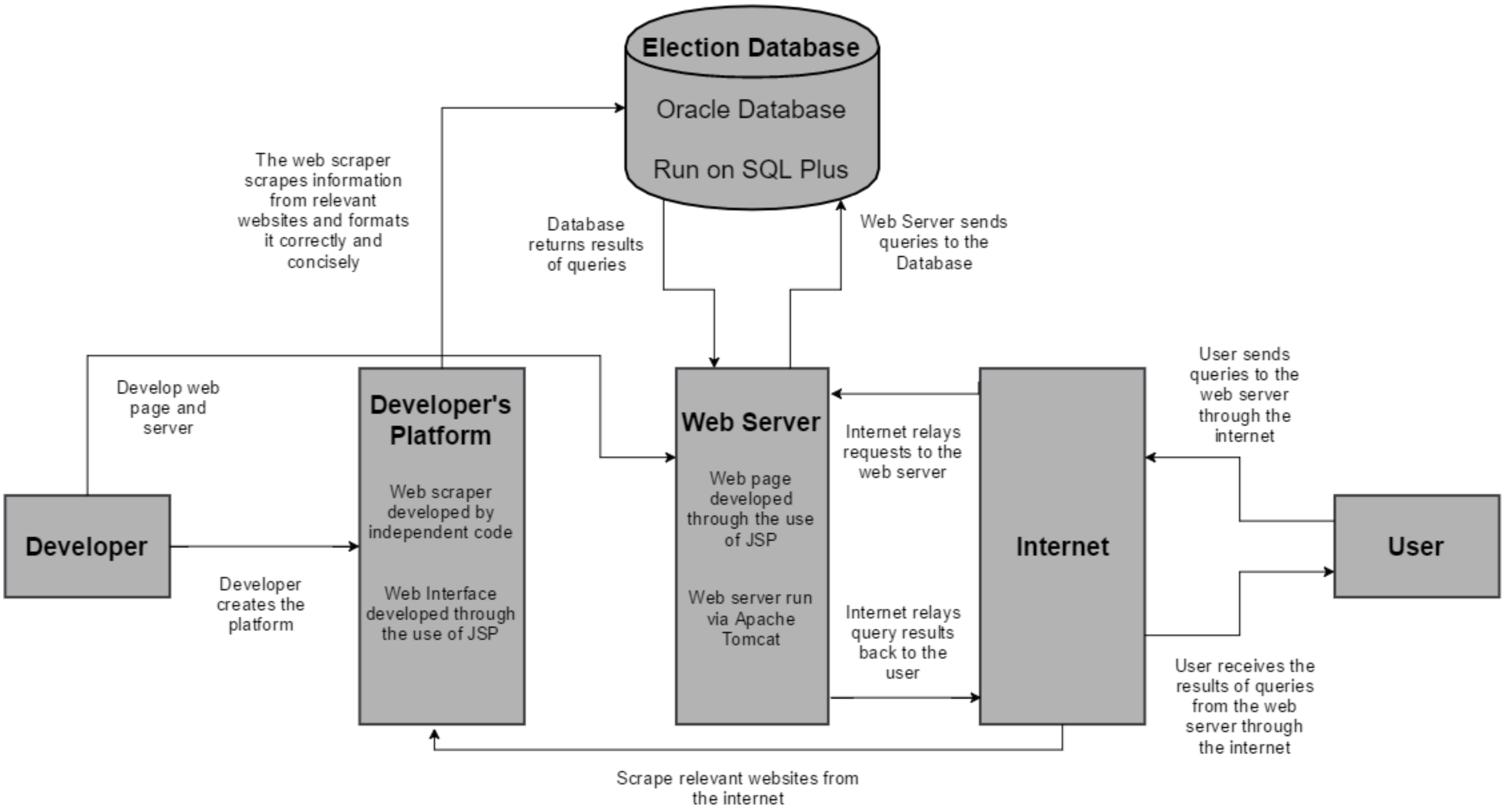
3.1.1 From the user's perspective:

The user will connect to *PresidentialQuery* over the internet and be received on the home page. On the home page the user will submit a query and *PresidentialQuery* will



3.1.2 From the developer’s perspective

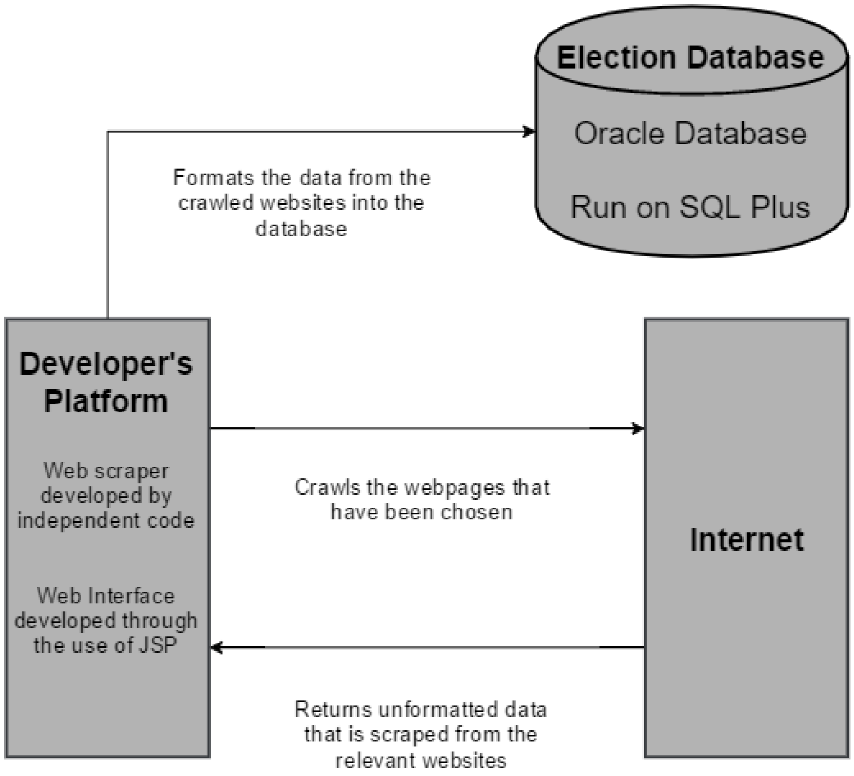
The developer will use an independently made web scraper, in order to scrape data from the websites list below. Below in the diagram, there is an outline of the communication between all the different parties and systems involved in the process.



Election Database
Oracle Database
Run on SQL Plus
The web scraper scrapes information from relevant websites and formats it correctly and concisely
Database returns results of queries
Web Server sends queries to the Database
Develop web page and server
Developer's Platform
Web scraper developed by independent code
Web Interface developed through the use of JSP
Web Server
Web page developed through the use of JSP
Web server run via Apache Tomcat
Internet relays requests to the web server
Internet relays query results back to the user
Internet
User sends queries to the web server through the internet
User receives the results of queries from the web server through the internet
Developer
Developer creates the platform
User
Scrape relevant websites from the internet

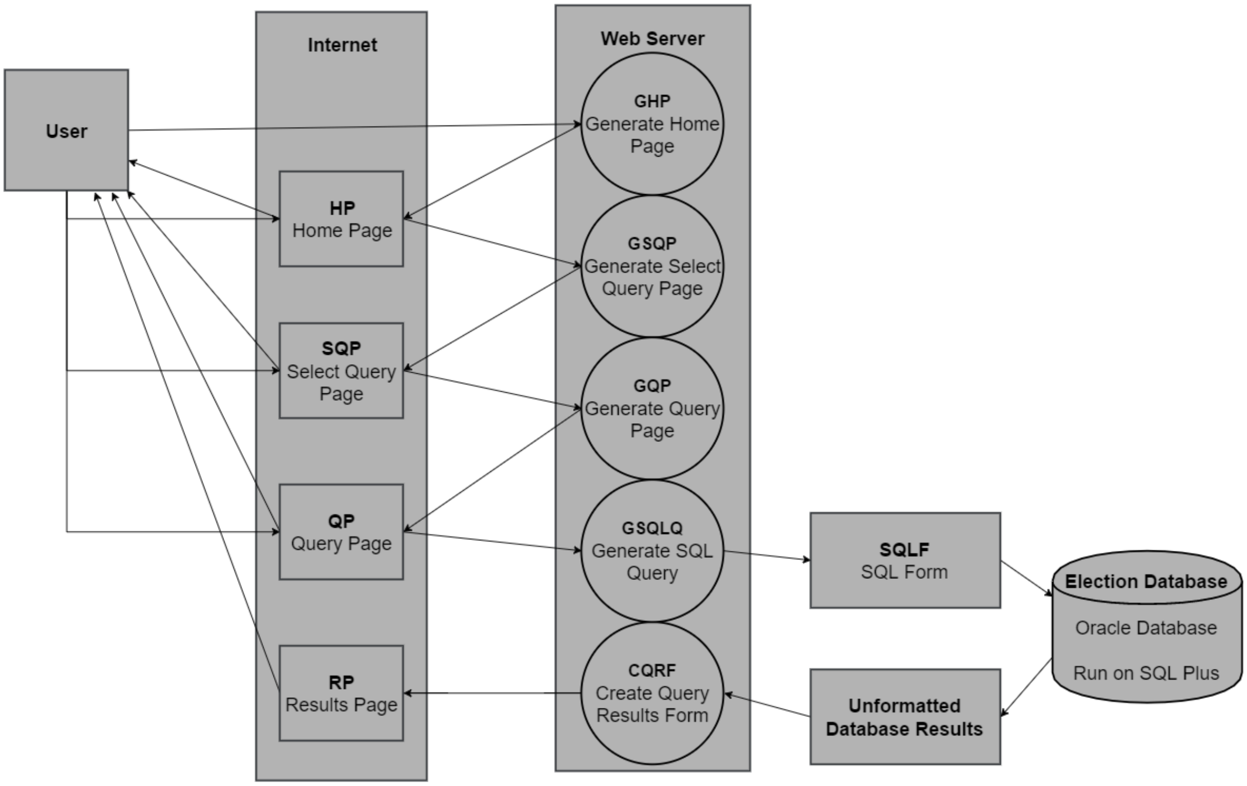
3.1.3 ETL Procedures

Websites that contain the most relevant and accurate data (as decided by the developers) will be found through careful research of the resources available. Once sufficient websites have been found, the data will be scraped and formated from the websites by a program independently designed by the developers. The formatted data will then be uploaded to the Election Database which will handle all future queries.


Election Database
Oracle Database
Run on SQL Plus
Formats the data from the crawled websites into the database
Developer's Platform
Web scraper developed by independent code
Web Interface developed through the use of JSP
Crawls the webpages that have been chosen
Returns unformatted data that is scraped from the relevant websites
Internet

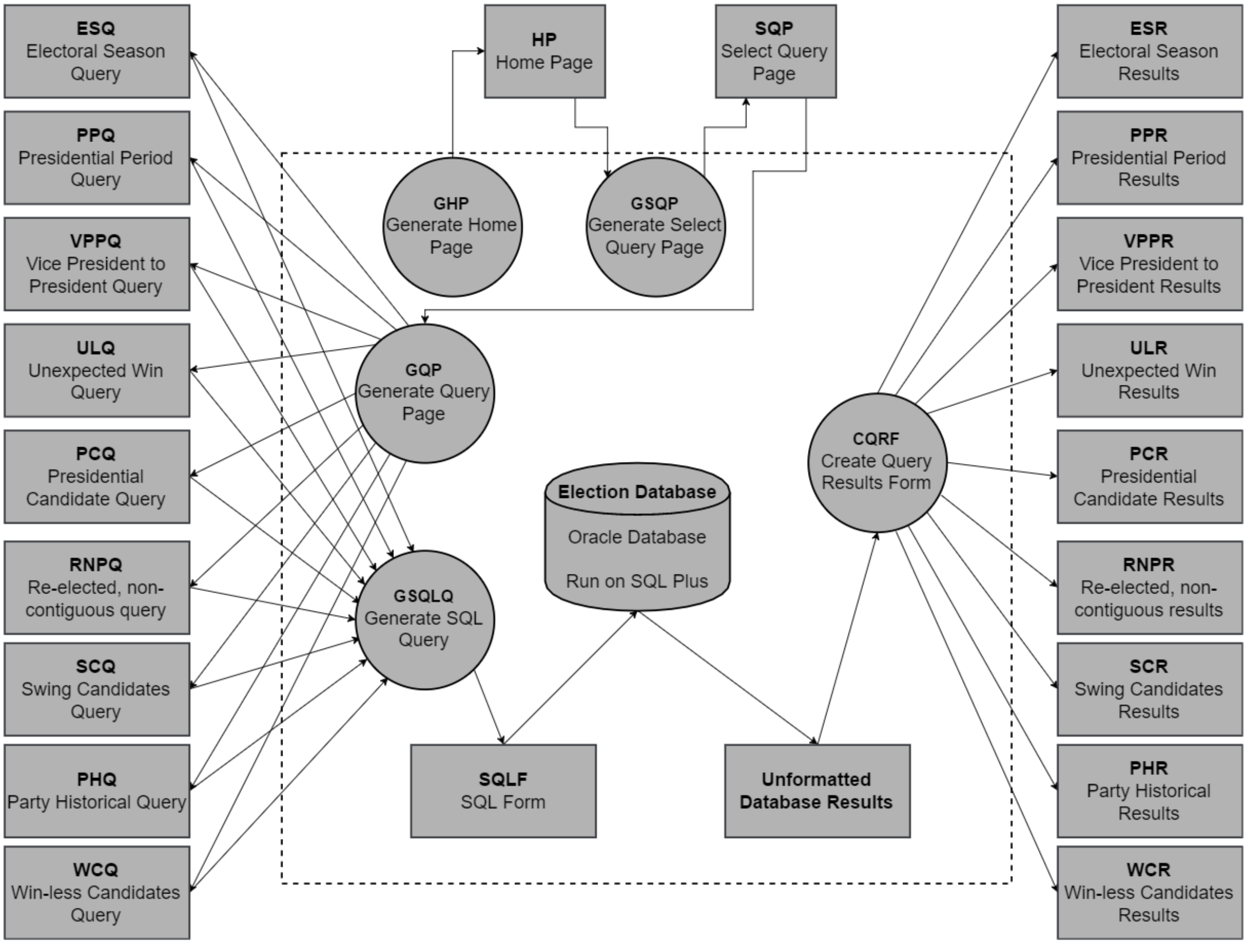
3.1.4 Web server Procedures

The user will connect to the website and the home page will be generated. The user will continue past the home page to the Select Query Page and will select a query they are interested in. This will bring them to the Query Page where they will further specify (if needed), the data they are interested in. Then an SQL Form will be created to use on the Elections Database. The results from this SQL Request will be formatted into a Results Form. The results page will then be generated and returned to the user.



3.2 Documentation

3.2.1 Top-Level Flow Documentation



3.2.2 Tasks, Subtasks, and Task Forms

3.2.2.1 Web Pages Research Task

TASK NUMBER: WPRT
TASK NAME: Web Pages Research
PERFORMER: *PresidentialQuery* Designers
PURPOSE: To locate web sources on the internet that provides election related information as well as population information in order to populate the database.
ENABLING COND: To populate the PED.
DESCRIPTION: Research the internet
FREQUENCY: As often as necessary but concentrated in the beginning part of the project
DURATION: 20 to 30 minutes per potential source investigated
IMPORTANCE: Critical
MAXIMUM DELAY: N/A
INPUT: Web queries
OUTPUT: Index of queried results
DOCUMENT USE: Web based search engines
OPS PERFORMED: Researching and keeping track of web sites and/or documents with Presidential election data.
SUBTASKS: None
ERROR COND: None

3.2.2.2 Extract, Transform, and Load Task

TASK NUMBER: ETLT
TASK NAME: Extract, Transform and Load Task
PERFORMER: *PresidentialQuery* Designers
PURPOSE: To extract data source web sources, transform it to a usable format, and then load the transformed data into the database.
ENABLING COND: Creation and updates to the PED
DESCRIPTION: Write custom code or utilize appropriate APIs to extract data, format them and then load them into the database via Oracle or other client.
FREQUENCY: However many web sources data are being extracted from.
DURATION: Varies
IMPORTANCE: Critical
MAXIMUM DELAY: N/A
INPUT: Web sources
OUTPUT: Formatted data organized into relations
DOCUMENT USE: HTML or other online TXT documents
OPS PERFORMED: Data extraction, data transformation, and data loading.
SUBTASKS: Develop data extraction procedure for each web sources
ERROR COND: None

3.2.2.3 Generate Home Page

TASK NUMBER: GHP
TASK NAME: Generate Home Page Task
PERFORMER: Apache Tomcat
PURPOSE: Generates the home page of the web server
ENABLING COND: A user connects to the web server created
DESCRIPTION: Utilize appropriate API’s to generate a homepage that the user can connect to
FREQUENCY: However often a user attempts to connect
DURATION: Short
IMPORTANCE: Critical
MAXIMUM DELAY: 30 seconds (The server will not allow any longer than a 30 second delay)
INPUT: User connection
OUTPUT: Home page formatted for the user
DOCUMENT USE: N/A
OPS PERFORMED: Creation of home page
SUBTASKS: N/A
ERROR COND: Assuming the website receives a reasonable number of requests, then there are no error conditions

3.2.2.4 Generate Query Select Page

TASK NUMBER: GQSP
TASK NAME: Generate Query Select Page
PERFORMER: Apache Tomcat
PURPOSE: Generates the page that allows the user to select among the various allowed queries
ENABLING COND: A user connects to the home page and continues to the GQSP
DESCRIPTION: Utilize appropriate API’s to generate a query select page that is easy to use and has the various queries to select from
FREQUENCY: Whenever a user continues past the home page
DURATION: Short
IMPORTANCE: Critical
MAXIMUM DELAY: 30 seconds
INPUT: User continuing past the home page
OUTPUT: A user selects one of the query options available
DOCUMENT USE: HTML Documents
OPS PERFORMED: Generating the Query Select Page and then taking the user’s input and relaying it to the next task
SUBTASKS: N/A
ERROR COND: Assuming the website receives a reasonable number of requests, then there are no error conditions

3.2.2.5 Generate Query Page Task

TASK NUMBER: GQP
TASK NAME: Generate Query Page
PERFORMER: Apache Tomcat
PURPOSE: Generates the page that allows the user to input data relevant to the query they selected during the GQSP task.
ENABLING COND: A user connects to the Generate Query Select Page and then selects a query
DESCRIPTION: Utilize appropriate API’s to generate a query page that is easy to use and allows users to input the data relevant to the query they have previously selected
FREQUENCY: Whenever a user selects a query in the Generate Query Select Page
DURATION: Short
IMPORTANCE: Critical
MAXIMUM DELAY: 30 seconds
INPUT: User selecting a query in the Generate Query Select Page
OUTPUT: A user creates an SQL Form that matches their needs
DOCUMENT USE: HTML Documents
OPS PERFORMED: Generating the Query Page and then taking the user’s input and relaying it to the next task
SUBTASKS: N/A
ERROR COND: Assuming the website receives a reasonable number of requests, then there are no error conditions

3.2.2.6 Generate SQL Query Task

TASK NUMBER: GSQLQ
TASK NAME: Generate SQL Query
PERFORMER: Apache Tomcat
PURPOSE: Takes the user’s input and formats it so that it is a proper SQL command
ENABLING COND: A user connects to the Generate Query Page and then inputs the data
DESCRIPTION: Formats the user’s input and makes it a proper SQL command for the Election Database to handle
FREQUENCY: Whenever a user inputs a new query at the Query Page
DURATION: Short
IMPORTANCE: Critical
MAXIMUM DELAY: N/A
INPUT: User’s input from the Generate Query Page Task
OUTPUT: SQL Commands
DOCUMENT USE: (SQLF) SQL Form
OPS PERFORMED: Formatting to an SQL Form and checks to ensure the user’s input is valid for the query selected
SUBTASKS: N/A

ERROR COND: Assuming the website receives a reasonable number of requests, then there are no error conditions. Also if the user's input does not match the expected input for the query selected, then throw a Data Mismatch error.

3.2.2.7 Create Query Results Form Task

TASK NUMBER: CQRF
TASK NAME: Create Query Results Form Task
PERFORMER: Apache Tomcat
PURPOSE: Generates a Query Results Form based on the SQL Form and Elections Database
ENABLING COND: A proper SQL Form is created
DESCRIPTION: Turns the results of an SQL Form from the Elections Database into a formatted Query Results Form
FREQUENCY: Whenever a correct SQL Form is created
DURATION: Short
IMPORTANCE: Critical
MAXIMUM DELAY: N/A
INPUT: SQL Form
OUTPUT: ESQ, PPQ, VPPQ, ULQ, PCQ, RNPQ, SCQ, PHQ, WCQ
DOCUMENT USE: ESQ, PPQ, VPPQ, ULQ, PCQ, RNPQ, SCQ, PHQ, WCQ
OPS PERFORMED: Generating the correct Query Results form
SUBTASKS: N/A
ERROR COND: Assuming the website receives a reasonable number of requests, then there are no error conditions. If the results from the SQL Form on the Elections Database can't be recognized, throw a Unrecognized Data error.

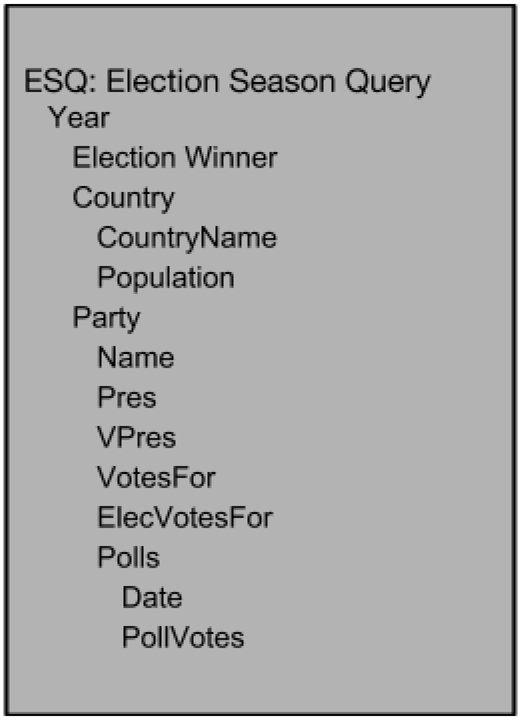
3.2.2.8 Generate Results Page Task

TASK NUMBER: GRP
TASK NAME: Generate Results Page
PERFORMER: Apache Tomcat
PURPOSE: Generates the page that displays the results from the query they selected during the GQSP task and the input they gave at the GQP task.
ENABLING COND: A proper Query Results Form has been created
DESCRIPTION: Utilize appropriate API's to generate a results page that is easy to use and allows users to view their results in an efficient manner.
FREQUENCY: Whenever a user has finished giving input on a query they have selected. Assuming that input matches the queries requirements
DURATION: Short
IMPORTANCE: Critical
MAXIMUM DELAY: 30 seconds
INPUT: One of the various Query Results Forms
OUTPUT: A results page generated from the Query Results Form given
DOCUMENT USE: ESQ, PPQ, VPPQ, ULQ, PCQ, RNPQ, SCQ, PHQ, WCQ
OPS PERFORMED: Generating the Results Page from the document provided from the Query Results Form

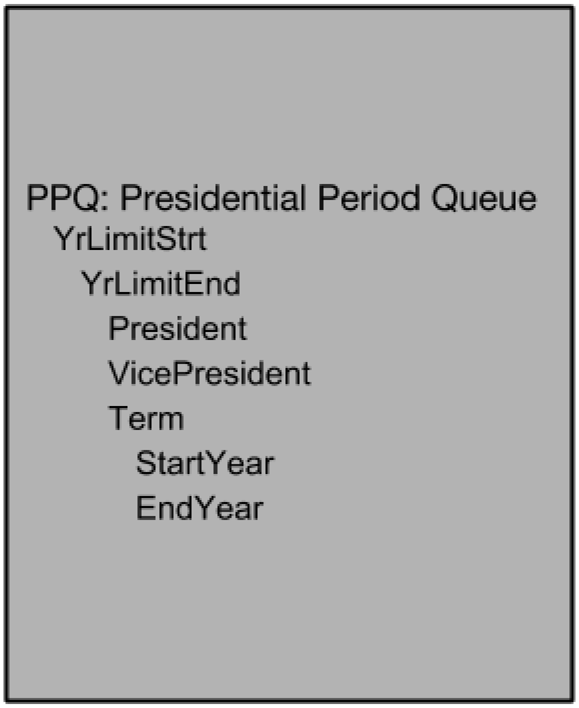
SUBTASKS: N/A

ERROR COND: Assuming the website receives a reasonable number of requests, then there are no error conditions.

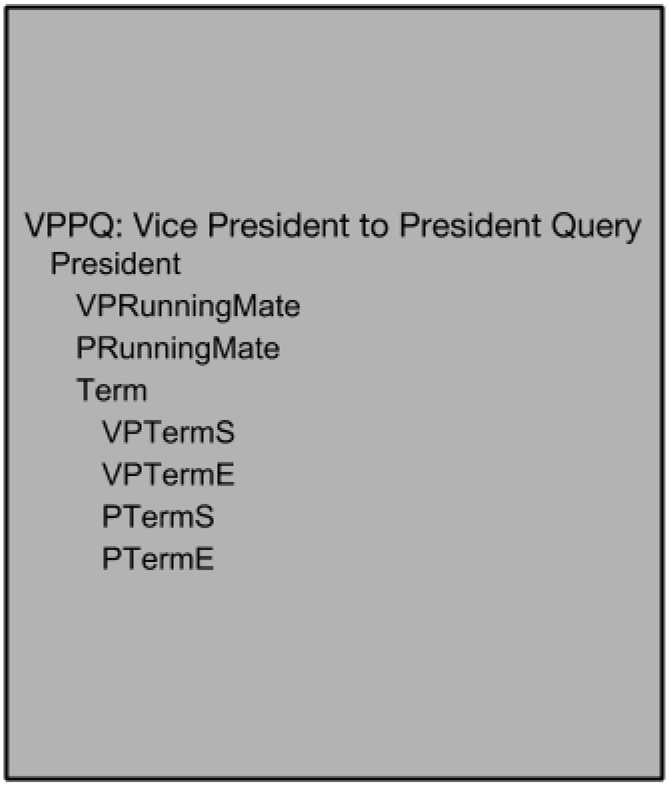
3.2.3 Document Forms



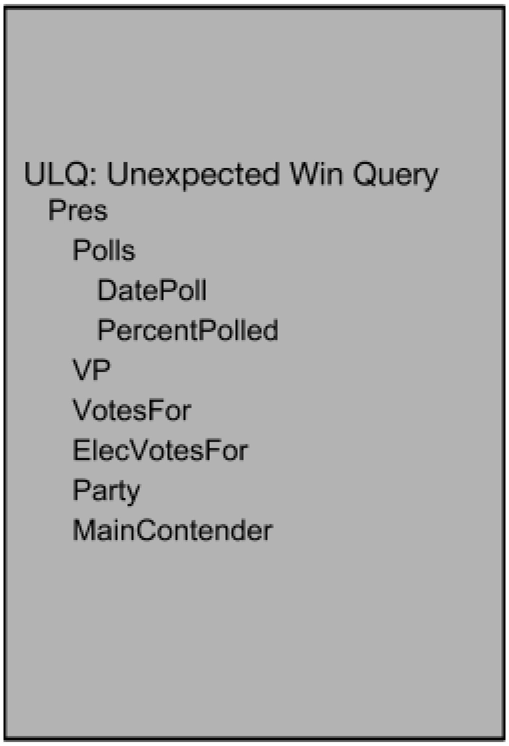
ESQ: Election Season Query
- Year
  - Election Winner
  - Country
    - CountryName
    - Population
  - Party
    - Name
    - Pres
    - VPres
    - VotesFor
    - ElecVotesFor
    - Polls
      - Date
      - PollVotes



PPQ: Presidential Period Queue
- YrLimitStrt
  - YrLimitEnd
    - President
    - VicePresident
    - Term
      - StartYear
      - EndYear



VPPQ: Vice President to President Query
- President
  - VPRunningMate
  - PRunningMate
  - Term
    - VPTermS
    - VPTermE
    - PTermS
    - PTermE



ULQ: Unexpected Win Query
- Pres
  - Polls
    - DatePoll
    - PercentPolled
  - VP
  - VotesFor
  - ElecVotesFor
  - Party
  - MainContender

PCQ: Presidential Candidate Query

- PresCand
  - YearsRan
    - VotesFor
    - ElecVotesFor
    - Results
    - Polls
      - DateOfPoll
      - PercentPolled
    - Party
    - VP

RNPQ: Re-elected, Non-contiguous President Query

- PresCand
  - YearElec
    - VotesFor
    - ElecVotesFor
    - Polls
      - DateOfPoll
      - PercentPolled
    - Party
    - VP

SCQ: Swing Candidates Query

- PresCand
  - YearRan
    - PartyRan
    - VP
    - VotesFor
    - ElecVotesFor
    - Results
    - Polls
      - DateOfPoll
      - PercentPolled

PHQ: Party Historical Query

- Party
  - PartyWins
  - PartyLosses
  - Year
    - PCand
    - VPCand
    - VotesFor
    - ElecVotesFor
    - Polls
      - DateOfPoll
      - PercentPolled

WCQ: Winless Candidates Query
- PCand
  - YearsRun
    - VP
    - CompetingCandidate
    - Year
    - VotesFor
    - ElecVotesFor
    - Polls
      - DateOfPolls
      - PercentPolled

SQLF: SQL Form
- Select
  - Attributes
- From
  - Relations
- Where
  - Conditions

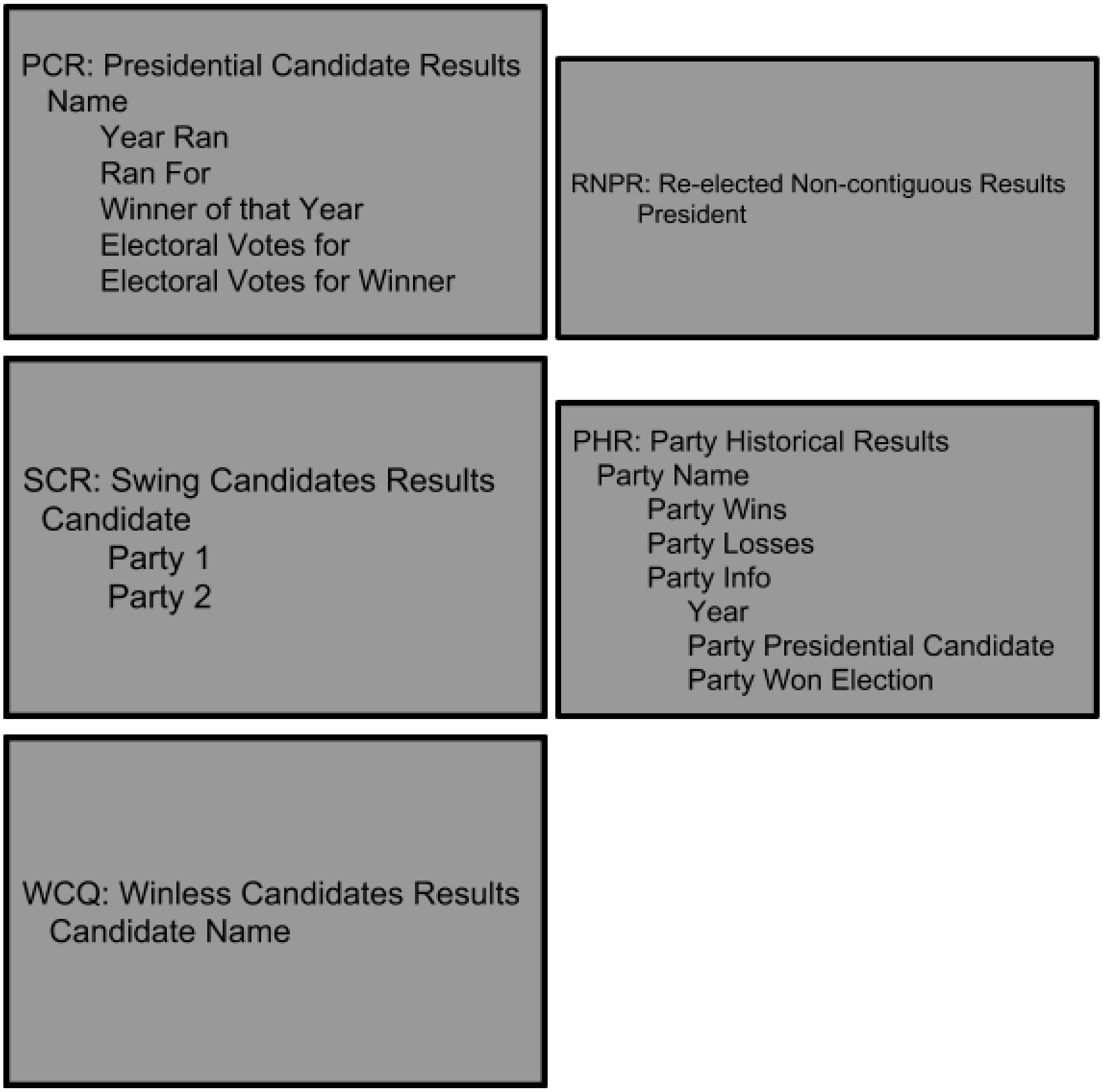
UDF: Unformatted Data Form
- Attribute List
  - Attribute Values

ESR: Election Season Results
- Year of Election
  - President
  - Vice President
  - Party
  - Popular Vote
  - Electoral Vote

PPR: Presidential Period Results
- Year of Election
  - President
  - Vice President
  - Party
  - Popular Vote
  - Electoral Vote

VPPR: VP to P Results
- Name
- Year of Vice-Presidency
- Year of Presidency

ULR: Unexpected Win Results
- President
  - Year of Election
  - President’s Elec Votes
  - President’s Pop Votes
  - Opponent
    - Opponent’s Elec Votes
    - Opponent’s Pop Votes



PCR: Presidential Candidate Results

- Name
  - Year Ran
  - Ran For
  - Winner of that Year
  - Electoral Votes for
  - Electoral Votes for Winner

RNPR: Re-elected Non-contiguous Results

- President

SCR: Swing Candidates Results

- Candidate
  - Party 1
  - Party 2

PHR: Party Historical Results

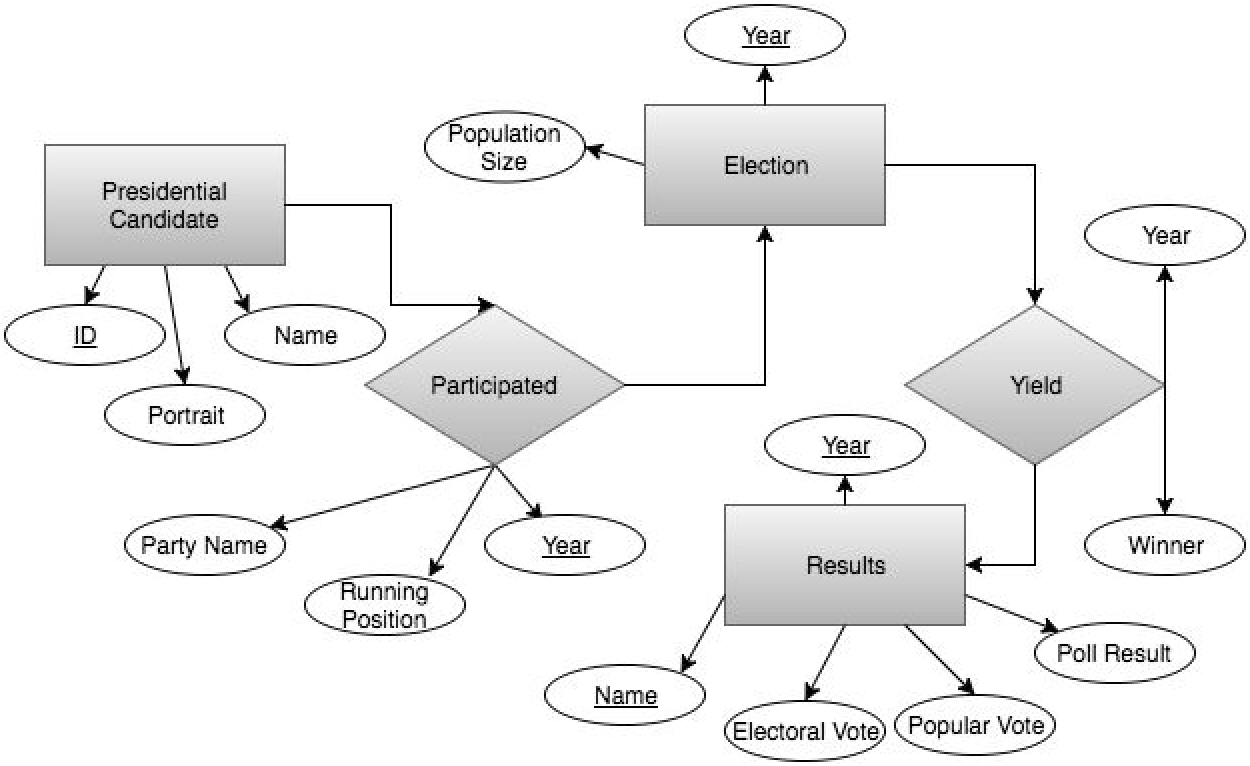
- Party Name
  - Party Wins
  - Party Losses
  - Party Info
    - Year
    - Party Presidential Candidate
    - Party Won Election

WCQ: Winless Candidates Results

- Candidate Name

4. Conceptual Modeling

4.1 Conceptual Schema



Based on the document forms the above high level conceptual schema is developed to represent the current design of the *PresidentialQuery* project. The different entities and relationships are extracted from the document forms created in the Requirements and Document section and these entities and relationships can subsequently be mapped into a logical schema.

4.2 Functional Dependency

The following functional dependencies are identified from the ER model:
For the Presidential Candidate Entity:

- ID -> Name
- ID -> Portrait

For the Participate Relationship:

- Year, Name -> Position
- Year, Name -> Political Party

For the Result Entity:

- Year, Name -> Popular vote
- Year, Name -> Electoral vote
- Year, Name -> Poll

From the Yield Relationship

- Year -> Winner

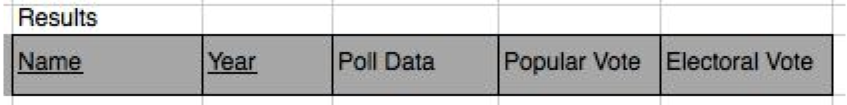
From the Election Entity

- Year -> Population Size

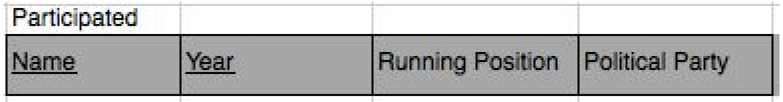
5. Logical Modeling

5.1 Logical Schema

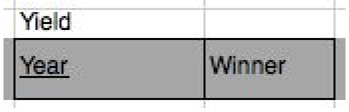
As stated previously, a logical schema for the *PresidentialQuery* project may be derived from the entities and relationships identified in the previous section. The logical schema will consists of several relational schemas represented below:



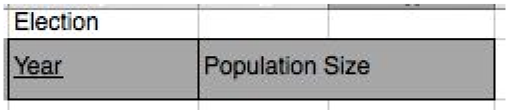
Results				
Name	Year	Poll Data	Popular Vote	Electoral Vote



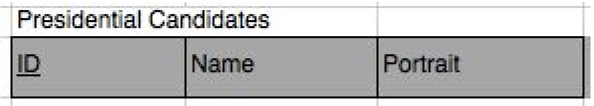
Participated			
Name	Year	Running Position	Political Party



Yield	
Year	Winner



Election	
Year	Population Size



Presidential Candidates		
ID	Name	Portrait

These relations are carefully designed to ensure that they are either in Boyce-Codd normal form (BCNF) or in Third normal form (3NF). These forms are ideal because they have nice properties that help make queries more simple and efficient. If any relations are found to not fit into these forms, the relations will be normalized through decomposition.

6. Task Emulation

6.1 Web Page Research Task:

```
{Use various web search engines to find websites related to the presidential election}
For each website found from searches
    For each webpage in the website
        If there is relevant, accurate information on the presidential election in the webpage
            Save the webpage for further use
```

6.2 Extract, Transform, and Load Task:

```
For each saved webpage from the Web Page Research Task
    Scrape the data from the webpage into a table using our independently developed Web Scraper
```

6.3 Generate Home Page:

```
{HTML for home page website}
{HTML for logo}
{HTML for button “Continue to Query Page”}
If button_is_clicked = true
    Send user to the Query Select Page
Else if logo_is_clicked = true
    Refresh Home Page
```

6.4 Generate Query Select Page:

```
{HTML for Query Select Page}
{HTML for logo}
{HTML for buttons of all query options (query option buttons)}
{HTML for button that locks in a query choice(lock choice button) default this button to not show}
{HTML for selection box}
If query_button_is_clicked = true
    {HTML change text in selection box to description of query selected}
    {HTML for lock choice button to appear}
    If lock_button_is_clicked = true
        Send user to Query Page
Else if logo_is_clicked = true
    Send user to the Home Page
```

6.5 Generate Query Page:

```
{HTML for general query page layout}
{HTML for logo}
If query_selected = Electoral Season Query
    {HTML for drop down menu labeled “Choose Election Year”}
If query_selected = Presidential Period Query
    {HTML for drop down menu labeled “Choose Start Election Year”}
    {HTML for drop down menu labeled “Choose End Election Year”}
If query_selected = Presidential Candidate Query
    {HTML for drop down menu labeled “Choose President/Candidate”}
If query_selected = Party Historical Query
    {HTML for drop down menu labeled “Choose Party”}
{HTML for submit button}
If submit_button_is_clicked = true
    Generate SQL query
    Send Query to PresidentialQuery’s database
    Format the results
    Generate Query Results Page
    Send user to Query Results Page
Else if logo_is_clicked = true
    Send user to the Home Page
```

6.6 Generate SQL Query:

```
Else if query_selected = Re-elected, Non-contiguous Query
    SELECT DISTINCT y.Winner
    FROM Yield y, Yield y2
    WHERE y.Winner = y2.Winner AND y.year > y2.year + 4
    AND y.Winner NOT IN (
        SELECT y3.Winner
        FROM Yield y3,Yield y4
        WHERE y3.Winner = y4.Winner AND y3.Year = y4.Year + 4);

Else if query_selected = Swing Candidates Query
    SELECT DISTINCT p.NAME
    FROM Participated p, Participated p2
    WHERE p.Name = p2.Name AND p.Party <> p2.party
        AND p.Party IS NOT NULL AND p2.party IS NOT NULL;

Else if query_selected = Vice President to President Query
    SELECT DISTINCT p.NAME
    FROM Participated p, Yield y
    WHERE p.Name = y.Winner AND p.Running_Position = 'vp' AND p.Year < y.Year;
```

Generate SQL Query (Continued):

```
Else if query_selected = Presidential Candidate Query
    SELECT cand.NAME, cand.YEAR, cand1.RUNNING_POSITION, year.WINNER,
    cand.ELECTORAL_VOTE, winner.ELECTORAL_VOTE
    FROM Results cand, Yield year, Results winner, Participated cand1, Participated cand2
    WHERE cand.NAME like '%"+cand[0]+"%' and cand.YEAR=year.YEAR
    and year.YEAR=winner.YEAR and year.WINNER=winner.NAME
    and cand.NAME=cand1.NAME and cand.YEAR=cand1.YEAR and
    cand1.YEAR=cand2.YEAR
    and cand1.RUNNING_POSITION like '%p%'
    and cand2.RUNNING_POSITION like '%p%' and cand2.NAME=winner.NAME;


Else if query_selected = Unexpected Win Query
    SELECT DISTINCT y.Winner
    FROM Results r1, Results r2, Yield y
    WHERE y.Year = r1.Year AND r1.Year = r2.Year AND
    y.Winner = r1.Name AND r1.popular_vote < r2.popular_vote AND r2.Name <> r1.Name;

Else if query_selected = Win-less Candidates Query
    SELECT DISTINCT part.NAME
    FROM Participated part
    WHERE part.Running_position ='p'
        AND part.NAME != all (SELECT y.Winner FROM Yield y);

Else if query_selected = Presidential Period Query
    SELECT y.YEAR, p.NAME, vp.NAME, p.PARTY, res.POPULAR_VOTE,
    res.ELECTORAL_VOTE
    FROM Results res, Yield y, Participated p, Participated vp
    WHERE p.Party=vp.Party and p.NAME!=vp.NAME and y.WINNER=p.NAME and
    y.YEAR<= $end_year and y.YEAR>= $start_year
    and res.YEAR=y.YEAR and p.running_position = 'p' AND vp.running_position = 'vp' and
    y.Year = p.year
    and p.year = vp.year and res.Name = p.Name
    ORDER BY y.YEAR;

Else if query_selected = Party Historical Query
    SELECT cand.PARTY, cand.YEAR, cand.NAME, yield.WINNER
    FROM Participated cand, Yield yield
    WHERE cand.PARTY='"+partyname[0]+"' AND cand.YEAR=yield.YEAR
    AND cand.RUNNING_POSITION='p'
    ORDER BY cand.YEAR;
```

6.7 Generate Results Form:

```
Table.create_table
If query_selected = Electoral Season Query
    table.add_columns(“Year”,”President”,”Portrait”,”Vice-President”,”Party Name”, “Popular Vote”, “Electoral Vote”)
    Populate table with results from PresidentialQuery’s database

Else if query_selected = Presidential Period Query
    table.add_columns(“Year”,”President”,”Portrait”,”Vice-President”,”Party Name”, “Popular Vote”, “Electoral Vote”)
    Populate table with results from PresidentialQuery’s database

Else if query_selected = Vice President to President Query
    table.add_columns(“President”)
    Populate table with results from PresidentialQuery’s database

Else if query_selected = Unexpected Win Query

Else if query_selected = Presidential Candidate Query
    table.add_columns(“Year”,”President”,”Vice-President”,”Party Name”, “Election Winner”, “Popular Vote”, “Electoral Vote”)
    Populate table with results from PresidentialQuery’s database

Else if query_selected = Re-elected, Non-contiguous Query
    table.add_columns(“President”)
    Populate table with results from PresidentialQuery’s database

Else if query_selected = Swing Candidates Query
    table.add_columns(“President”)
    Populate table with results from PresidentialQuery’s database

Else if query_selected = Party Historical Query
    Populate table with results from PresidentialQuery’s database

Else if query_selected = Win-less Candidates Query
    table.add_columns(“Candidate”)
    Populate table with results from PresidentialQuery’s database
```

6.8 Generate Results Page:

```
{HTML for general results page layout}
{HTML for logo}
{HTML for name of table (Name of query used)}
```

```
Show table returned from SQL query
If logo_is_clicked = true
        Send user to the Home Page
```

7. User Manual

7.1 Accessing the Database

To access the database, the user only require a basic computer and a web browser. The website may be accessed locally on the designer’s machine with the implemented code. In the future the website may be hosted so that any computer with access to the internet may interface with the database.

7.2 Reloading the Database

The data elements present in the PED were loaded from information extracted from various websites through the ETL tool developed in the course of the project. Since the data relevant to the database are not being generated on an hourly or even daily basis, the ETL tool does not need to be run continuously as a background process to maintain the database. However, new presidential election data do still get generated through the election process which occurs every four year so the Database will need reloading at least every four year in order to stay up to date. When such an occasion arise the ETL tool simply need to be run again to obtain the newly generated data to update the database. This will be possible provided that the web pages that the ETL tool obtained data from are themselves update as well.

7.3 Deploying the Database

The database was implemented on the MySQL platform, however this is not a necessity. Whatever platform you decide to use, the address should be updated in the ETL tool. Once, the address is updated the username and password must also be updated. Create a database on the platform chosen and then ensure the address points to that database. Then simply run the ETL Tool and the database will be filled with the needed tables.

7.4 Deploying the Web Server

The web server uses a mixture of Java Server Pages and HTML to implement its design. Apache Tomcat was used to host the web page. Enter the Command Prompt and change your directory to that of your Apache Tomcat’s bin folder. Then run the startup.bat file with the command “startup”.

7.5 Accessing the Web Server

The default address for the Web Server is “localhost/abc/index.html”. This address can be configured differently if needed.

8. Implementation Description

The implementation of PresidentialQuery is organized in the following way. The Web Server is hosted by Apache Tomcat and written in a combination of HTML and JSP (Java Server Pages). The ETL Tool is written in Java (in the Eclipse IDE) and is configured to upload the websites data onto the MySQL Database platform. MySQL is configured to hold our data and to respond to queries. The Apache Tomcat connects and queries the MySQL database through a library called MySQL-CONNECTOR (mysql-connector-java-5.1.40-bin.jar in the JDK library). Therefore, the Web Server queries data through the use of MySQL-CONNECTOR from a MySQL database (with queries dynamically written in SQL) which has its data uploaded from the independently written ETL Tool (written in Java).

9. System Limitations and Improvements

9.1 Limitations

There are several limitations on the current implementation of the system. One limitation is the lack of completeness of the data due to ambiguity in the web source or missing data.In particular, the web sources contains explicit description of each presidential candidate's political party but description of the vice-presidential candidates were more so implied based on the presidential candidates they were running with. This implicit representation of the vice-presidential candidate's political party completely breaks down for election years prior to 1844 since vice presidential candidates ran independent of the presidential candidates prior to 1844. Thus, political party affiliation could not be determined for vice presidential candidates prior to 1844. Similarly, the database do not currently account for presidents who became president through means other than political election. What this will impact is the queries dealing with the presidential status of some candidates. For example, in the vice president to president queries, the presidents who came to power through the death of the previous president would likely be missing in the query result. Another limitation of the system is the accuracy of the data loaded. The inaccuracies may be associated with the inaccuracies involved in the original data source or with the parsing and interpretation of the data. The latter reason for inaccuracy in the database is due to the inherent difficulties in parsing the web. As stated before, there are some ambiguity in the web sources so misinterpretation of the data may result in inaccuracies. Also the format of the web data are not fully consistent which presents difficulties to parsing and retrieving the data accurately.

9.2 Possible Improvements

- Parse multiple web sources on the same data type to improve accuracy (through cross verifications) and completeness (missing data from one source might be present in another)
- Optimize queries for faster results

10. Final Report on the End System

As mentioned before, the end system is composed of an ETL (extract-Transform-Load) tool, a Presidential Election Database (PED), and a dynamic web interface. The purpose of the final end system for PresidentialQuery is to provide users with an interface to access the PED in order to learn more about past United States Presidential Elections. The ETL tool (written in Java) is ran independent of the web interface in order to populate the PED in MySQL server with data retrieved from specific web sources. THE PED is composed of several relations as outlined in the Conceptual Modeling section. The relations are Presidential Candidates, Participated, Yield, Elections, and Results. These relations are designed to conform to at least conform to 3NF. Finally, the web interface is built with HTML and JSP and serve as a platform to allow users to perform certain queries on the PED. Inputs from the web interface is transformed into SQL queries in order to retrieve the relevant data to display back to the users in a formatted manner. While the results for each queries is not guaranteed to be fully accurate due to the aforementioned limitations, the end system is currently fully functional.

11. Web sources

Some preliminary web sources have been identified for extracting information relevant to the PED. Relevant data includes list of candidates, parties participated, electoral votes candidates received, popular votes candidates received, winner, population at time of election, polls, and photos of president. Additional data will be added to the database as research continues. Below is the list of some of the potential sources data can be extracted from:

http://www.gallup.com/poll/9442/election-polls-accuracy-record-presidential-elections.aspx

https://en.wikipedia.org/wiki/Historical_polling_for_U.S._Presidential_elections

https://en.wikipedia.org/wiki/United_States_Census

https://en.wikipedia.org/wiki/Presidential_portrait_(United_States)

http://www.infoplease.com/ipa/A0781450.html

http://ropercenter.cornell.edu/polls/us-elections/popular-vote/

http://2012election.procon.org/view.resource.php?resourceID=004332

The final web sources used for the project is listed below and is based on the actual information need of the database and queries:

https://en.wikipedia.org/wiki/Historical_polling_for_U.S._Presidential_elections

https://en.wikipedia.org/wiki/United_States_Census

https://en.wikipedia.org/wiki/Presidential_portrait_(United_States)

http://2012election.procon.org/view.resource.php?resourceID=004332