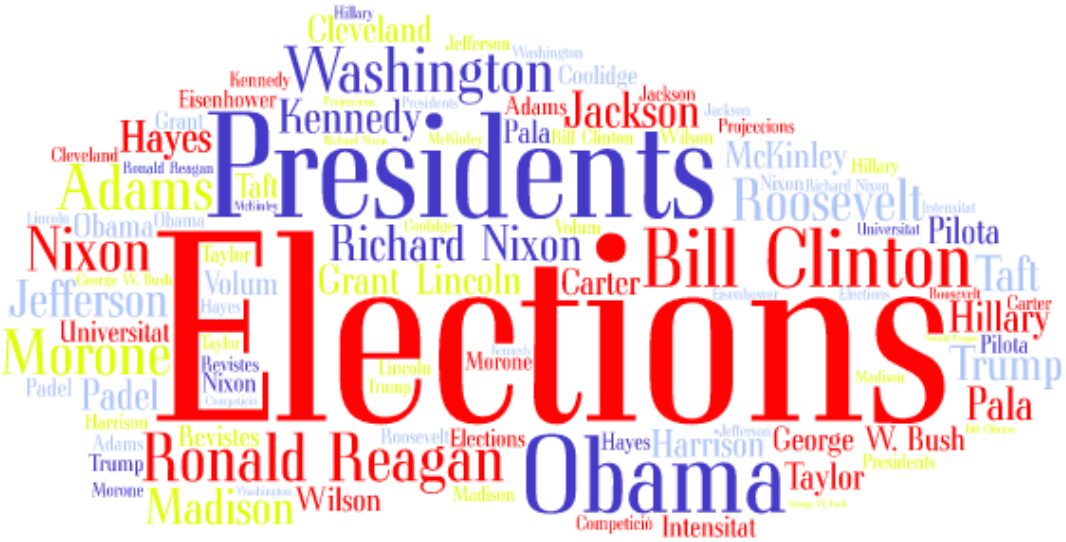


History of Presidential Elections
CMSC Database Design Project

Yuan Dong, Weiru Xie
University of Maryland

Contents

1 ENVIRONMENT AND REQUIREMENT ANALYSIS

1.1 Purpose of Document

The purpose of this document is to provide detailed requirements and design specifications as well as to describe the implementation process and result for the CMSC424 Database Design Project **History of Presidential Elections**. The document contains a description of limitations and assumptions about this model, a description of how to extract, transform and load data and how to build a web server. A top-level flow diagram is included in this document to show the logical flow of this project. Besides that, there are document forms as well as task forms to describe different tasks at each stage. For conceptual modeling, there is an ER model graphic schema and some functional dependencies derived. For logical modeling, there is a logical schema of the relational model. The document also has a list of design specification of each tasks, a user manual about each query that this project implements and some additional information.

1.2 Purpose of Project

The main goal of this project is to populate a relational database, called the Presidential Elections Database (PED), with data readily available on the Web. In this project, at ETL part, we need transform web HTML and XML data into a relational database that supports SQL querying and processing, format the data and integrate the data into a single relational database. Then we need to create a web-interface for users to interact and query the database. At last, a detailed report to describe the analysis, design and implementation process will be produced.

1.3 Scope

The scope of this project involves multiple tasks. The first task is to research and collect reliable data source related to the history of the president elections in United States(1789 to 2016). The second is extract data from these web sites and to do some "data cleaning" during transformation and loading. That is to discover and eliminate duplicate data, correct wrong data and transfer data into uniform format. The third task is to build a welcome page and a select query page. It provides an interface for users to create a query to search for information on a given election year, information on a given president/candidate, presidents who were re-elected on non-contiguous times, swing candidates, results of a party throughout the history of the elections, information on a given state, and so on. This task also needs us to write code to process and interpret these mentioned queries and provide results to users.

1.4 Assumption

The assumptions for this project are as follows:

- The data will be accurate, reliable and complete.
- The user has basic web browsing skills to access the web interface.
- It is assumed that the database server is configured appropriately to handle the user demands placed on the project.

1.5 Technical and Conceptual Problems and Solutions

1.5.1 Technical Problems and Solutions

Problem: Implementing a web server
Solutions: Research to acquire necessary knowledge to install and run a web server on a demo machine.

Problem: Gathering data
Solutions: Search online and compare data on different websites.

Problem: Extracting data
Solutions: For useful and reliable data, we choose different ways to extract data considering their format. Write a web crawler in Python to extract data from websites.

Problem: Transforming data
Solutions: Transform extracted data into uniform format.

Problem: Checking data
Solutions: Go through extracted data to clean and correct data.

Problem: Loading data
Solutions: Load the reformatted data into our database.

Problem: Lack of knowledge in creating interactive web pages
Solutions: Research and learn languages like HTML, CSS, PHP.

Problem: Lack of knowledge creating web server scripts
Solutions: Research and learn PHP.

Problem: Writing accurate and detailed pseudocode for each task and the embedded DML code
Solutions: Further research the technologies and starting the programming phase.

Problem: Building the SQL queries
Solutions: Research and follow examples.

Problem: Learning PHP to interact with mySQL
Solutions: Research and follow examples.

Problem: Returning results and display to the client side
Solutions: Research and follow examples.

1.5.2 Conceptual Problems and Solutions

Problem: Identifying a complete set of tasks of the project.
Solutions: Deeper analysis of this project. Look through the sample of a complete project and learn from it. Talk to the teaching assistant for advice.

Problem: Designing additional queries.
Solutions: Analyze our extracted data and their relationship to com up with our own queries. Ask teaching assistant for approval.

Problem: Designing the flow chart.
Solutions: Deeper analysis and understanding of this project. Figure out what we should do at each phrase of project.

Problem: Designing and building the E-R model.
Solutions: Review concepts of E-R model. Analyze extracted data and their relationship. Build a E-R model with less redundancy and more convenience for queries.

2 SYSTEM ANALYSIS AND SPECIFICATION

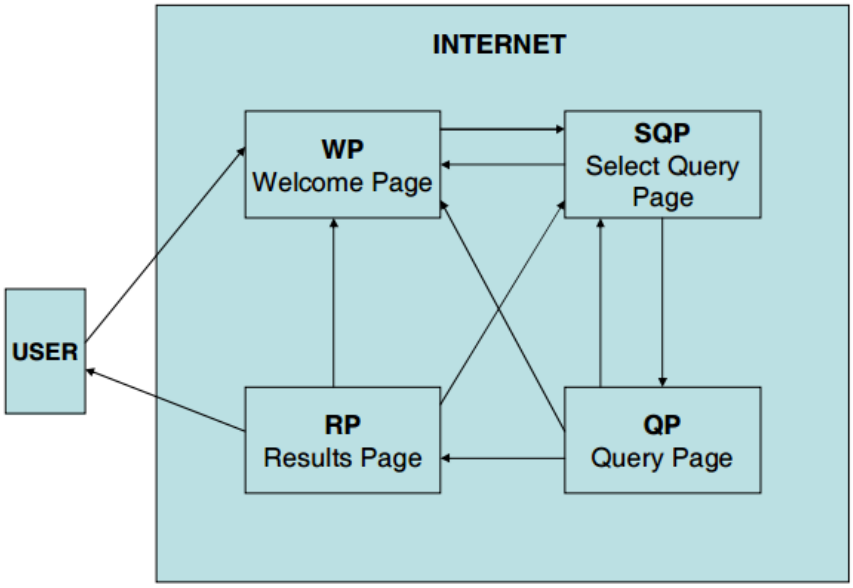
2.1 Description of Procedure

PED operates via a web browser that allows users to select various search criteria in researching facts about American presidential election from 1789 through 2016.

2.1.1 From the User's Perspective

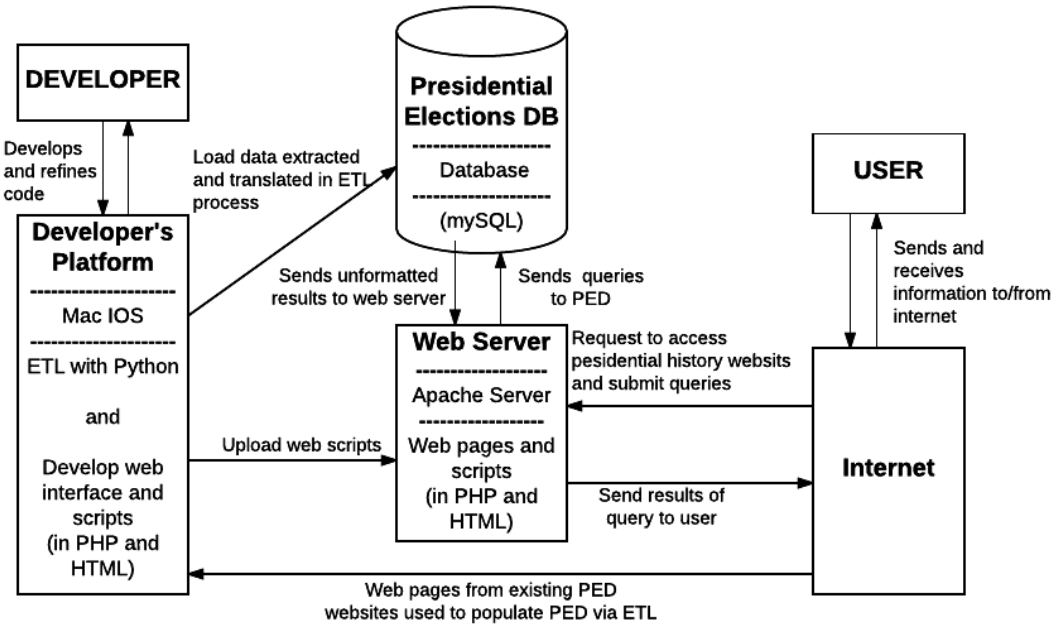
The first step in accessing our website PED is to navigate a predefined website. There the user will create a query and submit it to a process running on a remote server where the PED is stored. This process will create a form containing SQL commands for the specified query and will submit it to the database. After the data has been retrieved from the database it will be formatted and

presented through the user's web browser.



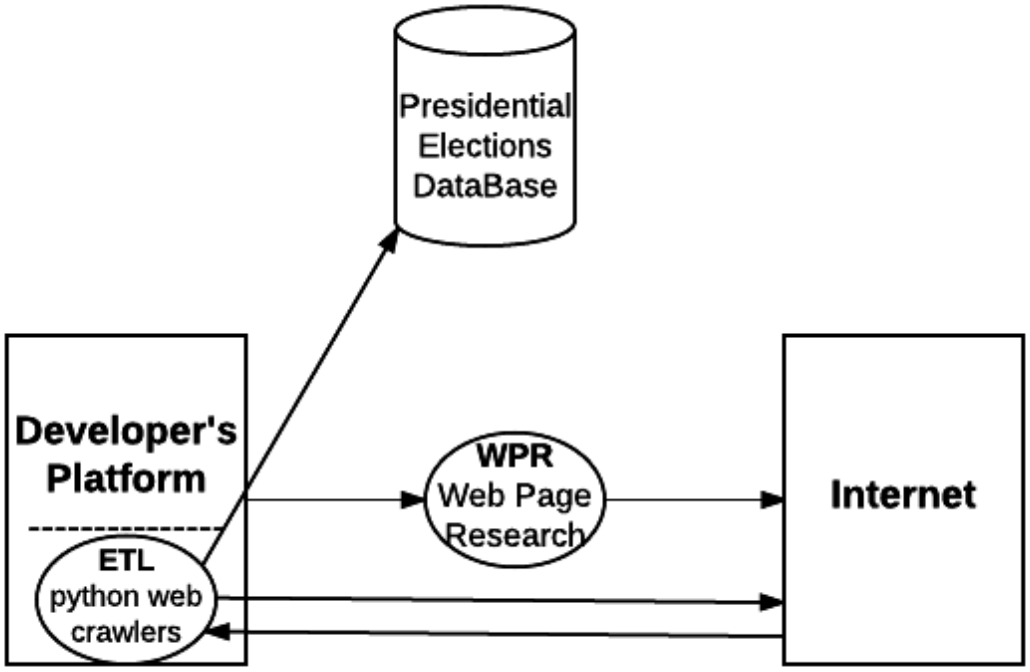
2.1.2 From the Developer's Perspective

The developers will be employing several technologies in order to implement the enterprise in its varying phases. The diagram below shows the main components of the system and indicates what responsibility to the system each component has as well as the general flow of information. Parts of this diagram will be elaborated upon in subsequent sections.



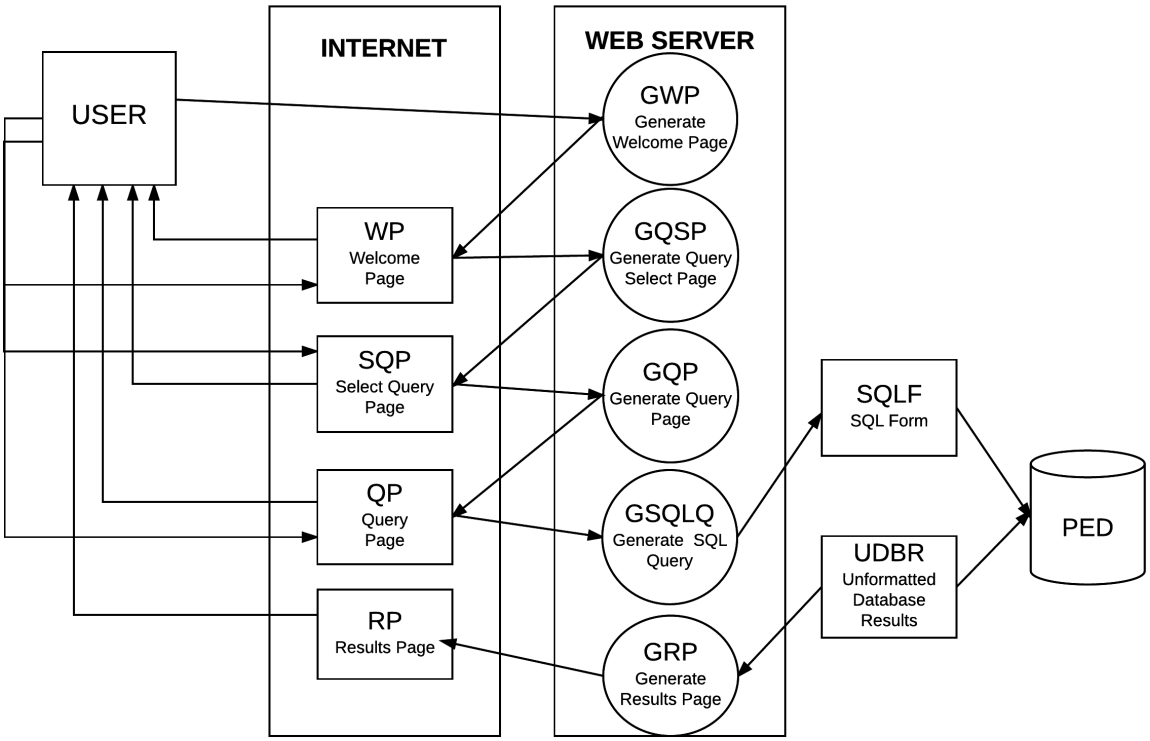
2.1.3 ETL Process

The designers research, analyze, and select the most relevant presidential election information websites. With the resulting websites bookmarked, Python scripts are used to extract useful data from the resulting tables, transform it into the required format. USe mySQL to load data into the Presidents Elections Database tables and connect oy to web server to answer the different user queries through a web interface.



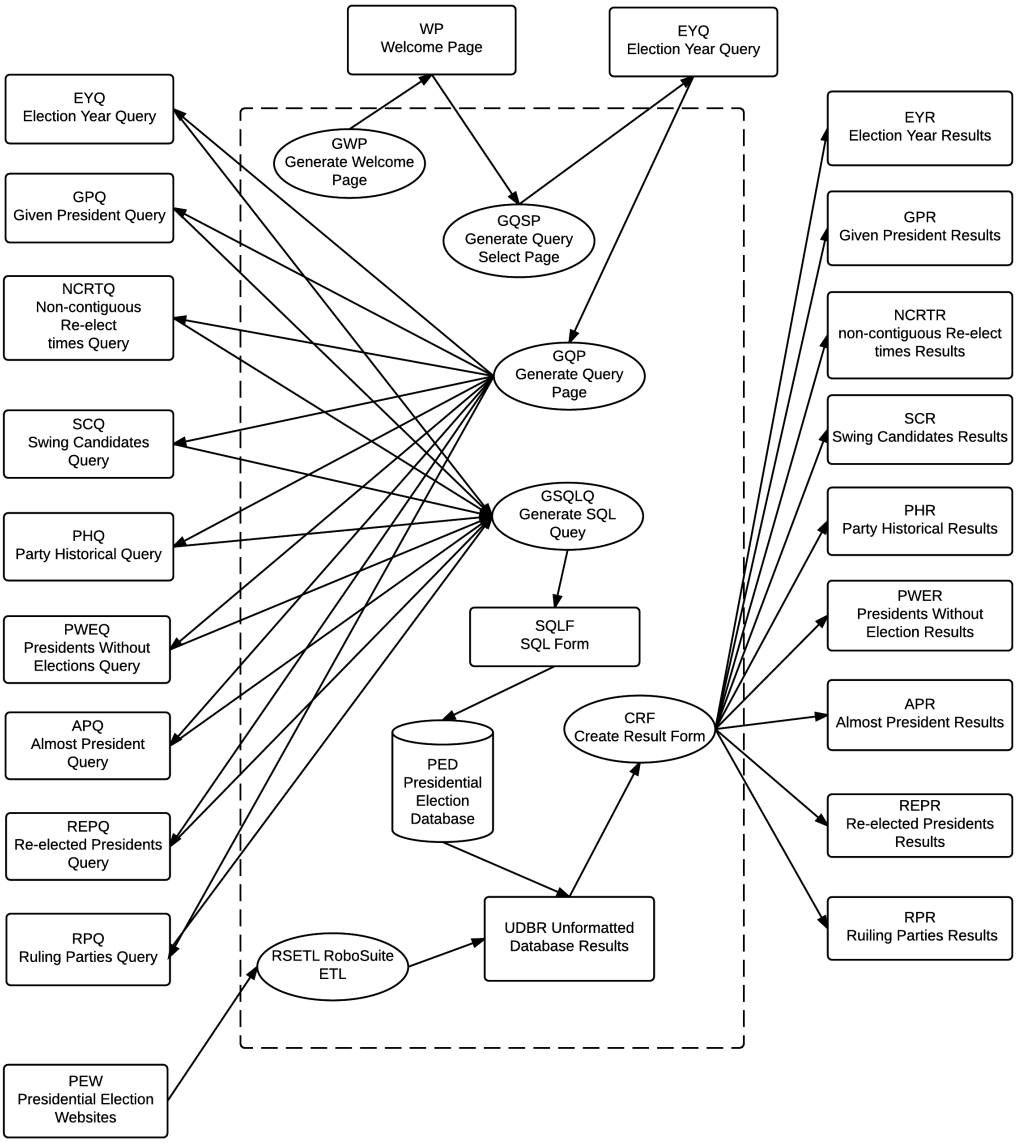
2.1.4 Web server Procedures

The PED internal procedures include the engine that powers the website. This involves providing various queries and results of those queries available to the user. This is accomplished by several scripts and code for pages located on a UNIX web server. When a user navigates to the PED website, the initial web page is generated and served to the user. The remaining scripts and procedures will be described by following a typical use case scenario. The user will proceed to enter the website. A page with a list of nine queries is presented to the user. The user will then select one from among the queries to be performed. The selection will be sent to the web server where another procedure will generate a page with various options relevant to that query. The user will fill out the options desired and submit the query. The web server will receive the query request; generate the appropriate SQL commands which are then sent to the PED. The database will then produce a results table and send it to the server from which the query was sent. Another process on the server will format the results into a web page and serve it up to the user.


INTERNET
WEB SERVER
USER
WP
Welcome
Page
SQP
Select Query
Page
QP
Query
Page
RP
Results Page
GWP
Generate
Welcome Page
GQSP
Generate Query
Select Page
GQP
Generate Query
Page
GSQLQ
Generate SQL
Query
GRP
Generate
Results Page
SQLF
SQL Form
UDBR
Unformatted
Database
Results
PED

2.2 Documentation

2.2.1 Top-Level Flow Diagram



2.2.2 Tasks, Subtasks, and Task Forms

Table 1: Pages Research Task.

TASK NAME	Web Pages Research
PERFORMER	PED designers
PURPOSE	To research the internet for web sites that contain data for the American presidential election from 1789 to 2016.
DESCRIPTION	Research the internet.
ENABLING COND	To populate the PED.
FREQUENCY	As often as necessary
DURATION	Varies
INPUT	Web queries
OUTPUT	Index of queried results
DOCUMENT USE	Web based search engines
OPS PERFORMED	Researching and bookmarking web sites and/or pages with American presidential election data
SUBTASKS	None
ERROR COND	None

Table 2: ETL Task.

TASK NAME	Extract, Transform, and Load Task
PERFORMER	Developers
PURPOSE	To extract data, transform or reformat it and load it into the PED
DESCRIPTION	Write web crawler in Python to extract specific data from a web page, and load it into a predefined data relation or table.
ENABLING COND	The creation of the OlympicsDB and any addition of data or updates to the OlympicsDB
FREQUENCY	Once for the creation of the OlympicsDB and during any updates.
DURATION	Varies
INPUT	A selected web page
OUTPUT	Data into a relation in the PED
DOCUMENT USE	HTML documents
OPS PERFORMED	Data extraction, data transformation, and data loading
SUBTASKS	Web pages Research
ERROR COND	None

Table 3: Generate Welcome Page Task.

TASK NAME	Generate Welcome Page.
PERFORMER	Apache web server
PURPOSE	To generate the welcome page.
DESCRIPTION	The Apache/Tomcat web server will generate the welcome page when a user wants to access the information on the PED.
ENABLING COND	User accessing the web interface.
FREQUENCY	As often as a user accesses the web address
DURATION	Very short
INPUT	None
OUTPUT	Welcome Page
DOCUMENT USE	WIFWF: Web Interface Welcome Form
OPS PERFORMED	Generation of welcome page, send it to the user and wait for user action
SUBTASKS	None
ERROR COND	If A/TServer == busy, then Process=TimeOut.

Table 4: Generate Query Select Page Task.

TASK NAME	Generate Query Select Page.
PERFORMER	Apache web server
PURPOSE	To generate the query select page.
DESCRIPTION	The web server will generate the query-select page when requested by the user (from the Welcome page). It will provide options for users to select or input their queries.
ENABLING COND	User clicking on the welcome image on the Welcome page.
FREQUENCY	As often as the user clicks on the welcome image on the Welcome Page or back from a Query Result Page.
DURATION	Very short
INPUT	Request from user to web server.
OUTPUT	Query Select Page
DOCUMENT USE	WISF: Web Interface Select Form
OPS PERFORMED	Generate the Query Select page, send it to the user and wait for user input.
SUBTASKS	None
ERROR COND	If A/TServer == busy, then Process=TimeOut.

Table 5: Generate SQL Query Task

TASK NAME	Generate SQL Query
PERFORMER	Apache web server
PURPOSE	To create SQL query based on users input.
DESCRIPTION	Every time when a user submits a web query form, corresponding SQL commands will be generated to task the presidential elections database.
ENABLING COND	Submitting a web query form.
FREQUENCY	As often as a user submits a web query form.
DURATION	Short
INPUT	Web query form
OUTPUT	SQL form
DOCUMENT USE	EYQ: Election Year Query; GPQ: Given Person Query; NCRTQ: Non-contiguous Re-elect times Query; SCQ: Swing Candidates Query; PHQ: Party Historical Query; GSQ: Given State Query; APQ: Almost President Query; REPQ: Re-elected President Query; RPQ: Ruling Parties Query (We implement them in Query Result Page Tasks, instead of separate documents. Code for Query are in Chapter 5.)
OPS PERFORMED	If Q == EYQ or Q == GPQ or Q == NCRTQ or Q == SCQ or Q == PHQ or Q == GSQ or Q == APQ or Q == REPQ or Q == RPQ
SUBTASKS	None
ERROR COND	If server is busy, then Process == TimeOut

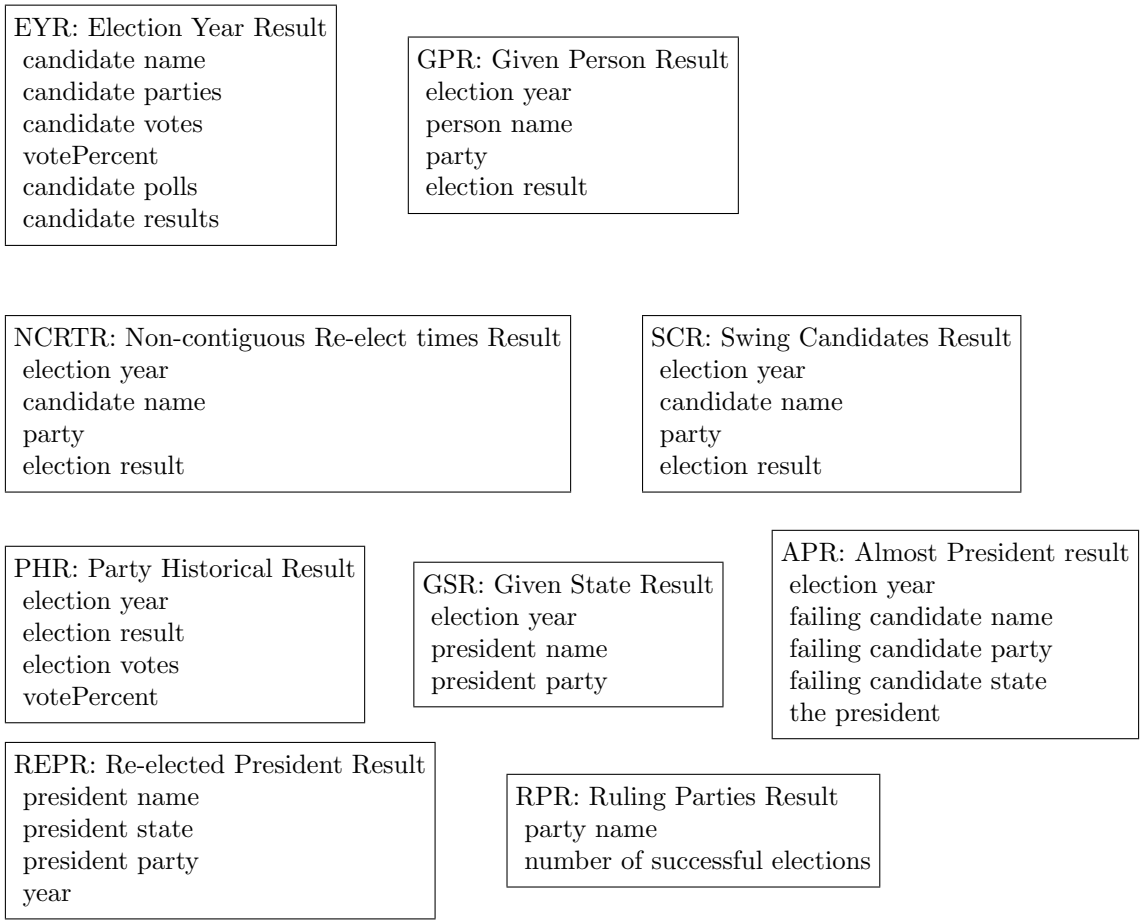
Table 6: Create Query Result Form Task

TASK NAME	Create Query Result Form
PERFORMER	Server side script
PURPOSE	To provide a formatted result from the Presidential Elections DB.
DESCRIPTION	To transform the result of a success query in PED into a format that can be interpreted by a web browser.
ENABLING COND	Database completing operations.
FREQUENCY	As often as a user submits a web query form and the query is performed successfully.
DURATION	Depends on the complexity of the query result.
INPUT	Data in Presidential Elections DB.
OUTPUT	EYR: Election Year Result; GPR: Given Person Result; NCRTR: Non-contiguous Re-elect times Result; SCR: Swing Candidates Result; PHR: Party Historical Result; GSR: Given State Result; APR: Almost President Result; REPR: Re-elected President Result; RPR: Ruling Parties Result
DOCUMENT USE	None
OPS PERFORMED	Transform the result of a success query in PED into a format that can be interpreted by a web browser.
SUBTASKS	None
ERROR COND	If the output is unknown, then produce error message and stop.

Table 7: Generate Results Page Task

TASK NAME	Generate Results Page
PERFORMER	Apache web server
PURPOSE	To generate results page.
DESCRIPTION	Every time when the query on Presidential Elections DB is performed, query results will be generated. The server will generate the results page and display it to users.
ENABLING COND	Getting the query result form.
FREQUENCY	As often as a user submits a web query form and the query is performed successfully.
DURATION	Short
INPUT	Query result
OUTPUT	Results page
DOCUMENT USE	WIRF: None
OPS PERFORMED	Generate a Results Page to be displayed to the user from the Result form.
SUBTASKS	None
ERROR COND	If server is busy, then Process == TimeOut

2.2.3 Document Forms

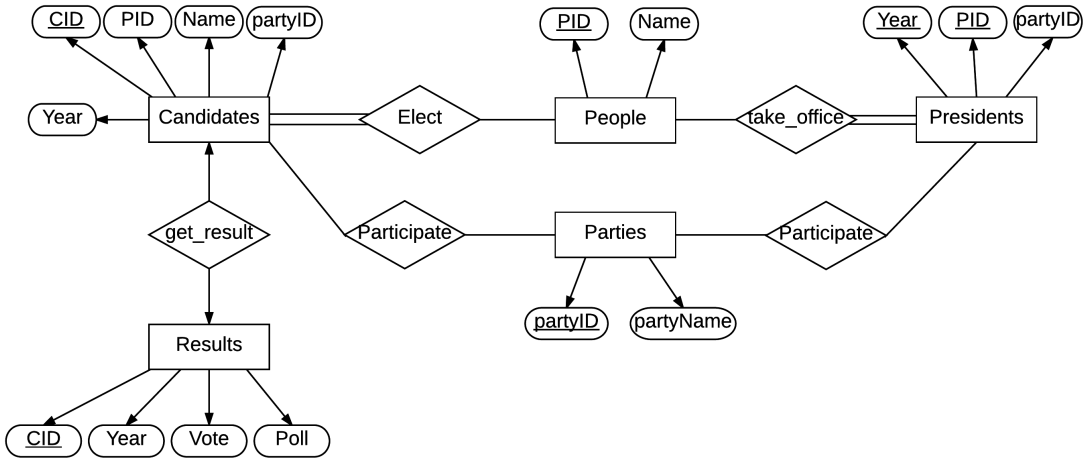


3 CONCEPTUAL MODELING

3.1 Conceptual Schema

The conceptual schema is the higher level representation of the PED project as conceived by the designers. This includes the different identified entities and relationships based on the document forms specified in the Requirements Document for this project. These entities and relationships include the internal processes by which the data will be extracted, transformed, and loaded (ETL) into the PED, as well as the process of user queries.

3.1.1 ER Model Graphical Schema



3.2 Functional Dependencies

The functional dependencies identified are:
For Candidate entity:

- CID → PID,Year, partyID

For People entity:

- PID → name

For Presidents entity:

- Year, PID → partyID

For Results entity:

- CID → poll
- CID → vote
- CID → votePercent

4 LOGICAL MODELING

4.1 Logical Schema

The logical schema is the next level in the representation of the PED comprised of the relation schemas derived from the ER diagram in the conceptual schema.
And our relation schemas are as follows:

4.1.1 Relational Model

Candidates: CID, Year, CID, partyID
People: PID, name
Presidents: PID,Year, partyID
Parties: partyID, partyName
Results: CID, polls, vote, votePercent

4.1.2 Normalization

Relations need to be in either Boyce-Codd normal form (BCNF) or in Third normal form (3NF) in order to obtain lossless and sometimes dependency preserving relations. In order to normalize these relations, we need to use the functional dependencies derived in the previous section, and check if the relations are BCNF or 3NF, and if they are not, then the relations need to be decomposed into BCNF or 3NF relations.

5 Task Emulation

5.1 Task Design Specification

5.1.1 Extract, Transform and Load Task Design

```
\\Google query to find American Presidents Elections sites
if website has complete and reliable data to be used by the ElectionsDB
    Bookmark
else
    skip
```

Write web crawlers in Python

```
for each website found in Google
    req = urllib2.Request(url)
    response = urllib2.urlopen(req)
    the_page = response.read()
    soup = BeautifulSoup(the_page, 'lxml')
    find values look for
    extract information into a specific format
    write out tables
```

5.1.2 Generate Welcome Page Task Design

```
{HTML for webpage layout}
{HTML for welcome image with a link}
if click_image == true
    link to Query Select Page
Else
    no action
```

5.1.3 Generate Query Select Page Task Design

```
{HTML for webpage layout}
{HTML for navigation bar}
{HTML for header with welcome image and introduction}
{HTML to display queries}
{HTML for provide users with input or options for each query}
if click_option == true
    link to SQL Query
    query PresidentsElectionsDB
    get results
    link to Query Result Page
else
    no action
```

```
if type_valid_input == true
    link to SQL Query
    query PresidentsElectionsDB
    get results
    link to Query Result Page
else
    return 'The input is not valid.'
```

5.1.4 Generate SQL Query Task Design

```
If query == Election_Year_Query
   SELECT c.year as Year, p1.name as Name, p2.partyName as Party,
   r.vote as Vote, votePercent, poll, Result
   FROM Candidates c , People p1, Parties p2, Results r
   WHERE c.year =  year_input
   AND c.PID = p1.PID
   AND c.CID = r.CID
   AND c.partyID = p2.partyID
   order by vote desc
Else if query == Given_Person_Query
    SELECT Year, p.name as Name, p2.partyName as Party, r.result as Result
    FROM People p, Candidates c, Results r, Parties p2
    WHERE c.PID = p.PID
    AND c.CID = r.CID
    AND c.partyID = p2.partyID
    AND p.name like name_input
Else if query  == Non-contiguous_Re-elect_Times_Query
    SELECT a.year as Year, b.name as Name, d.partyName as Party, r1.result as Result
        FROM Candidates a, People b, Parties d, Results r1,
        (SELECT p.name
                FROM Candidates c, People p, Results r,
                (SELECT p2.year as year1, p1.year as year2, p1.PID
                FROM Presidents p1, Presidents p2
                WHERE p1.PID = p2.PID
                AND p1.year-p2.year > 4) as temp
                WHERE c.PID = temp.PID
        AND c.CID = r.CID
                AND c. year > year1
                AND c.year < year2
                AND result = 'fail'
                AND c.PID = p.PID) as target
                WHERE b.name = target .name
                AND a.PID = b.PID
        AND a.partyID = d.partyID
        AND a.CID = r1.CID
Else if query == Swing_Party_Query
    SELECT c.year as Year, p.name as Name, partyName as Party, Result
    FROM Candidates c, People p, Parties p2, Results r,
    (SELECT distinct c1.PID
    FROM Candidates c1, Candidates c2
    WHERE c1.PID = c2.PID
    and c1.partyID != c2.partyID) as temp
    WHERE c.PID = temp.PID
    and c.PID = p.PID
    AND c.partyID = p2.partyID
    AND c.CID = r.CID
    ORDER BY p.name
Else if query == Party_Historical_Query
    SELECT c.year as Year, p.Name as Name, y.partyName as Party, Vote,
    votePercent, Result
    FROM Parties y, Candidates c, People p, Results r
```

```
    WHERE y.partyName LIKE party_input
    AND c.partyID = y.partyID
    AND c.CID = r.CID
    AND c.PID = p.PID
    ORDER BY c.year
Else if query == Presidents_Without_Elections_Query
    SELECT distinct pp.name AS PresidentName, p.year AS Year,
    py.partyName  AS Party
    FROM test.Candidates c, test.Presidents p, test.People pp, test.Parties py
    WHERE p.year NOT IN (
    SELECT  distinct c1.year
    FROM test.Candidates c1
    ) AND p.PID = pp.PID AND p.partyID = py.partyID;
Else if query == Almost_President _Query
    SELECT distinct c.year as Year, pp1.name AS Failed_Candidate, py.partyName
    AS Candidate_Party,r.poll AS
Candidate_Poll,r.vote AS Candidate_Vote, pp2.name AS President, py2.partyName
AS President_Party,
r1.poll AS President_Poll,r1.vote AS President_Vote
    FROM test.candidates c,test.People pp1,test.People pp2, test.Results r,
    test.Results r1, test.Presidents p,
  test.Parties py,test.Parties py2,test.candidates c1,
    (SELECT MAX(r.poll) AS maxpoll,c.year
    FROM test.candidates c, test.Results r
    WHERE c.CID = r.CID and c.year > 1932
    GROUP BY c.year) AS temp
    WHERE (c.CID = r.CID AND temp.year = c.year
    AND temp.maxpoll = r.poll AND r.result = 'fail'
    AND c.partyID = py.partyID AND c.PID = pp1.PID)
    AND c.year = p.year AND p.PID = pp2.PID
    AND p.partyID = py2.partyID AND c1.year = c.year
    AND c1.PID = p.PID AND r1.CID = c1.CID ;
Else if query == Re-election_Presidents_Query
   SELECT name AS President, count AS Reappointment_Times
    FROM People p,
    (select p1.PID, count(p1.PID)*2-(count(p1.PID)-1) as count
    FROM Presidents p1, Presidents p2
    WHERE p1.PID = p2.PID
    AND p2.year-p1.year <= 4
    AND p2.year > p1.year
    GROUP BY p1.PID) as temp
    WHERE p.PID = temp.PID;
Else if query == Ruling_Parties_Query
   SELECT  temp.partyName AS Party, COUNT(temp.PID) AS PresidentNum FROM
  (SELECT DISTINCT p.PID, PT.partyName
  FROM test.Presidents p, test.Parties PT
  WHERE p.partyID = PT.partyID AND PT.partyName != 'None') AS temp
  GROUP BY temp.partyName
  ORDER BY COUNT(temp.PID) DESC
```

5.1.5 Generate Result Page Task Design

```
{HTML for webpage layout}
Generate table
    create table for results
    populate with results from Presidential Elections Database
    display the result able
If click_back == true
    link to Query Select Page
```

6 SOURCE PROGRAM LISTING

See the attachments.

7 SQL Queries

All the SQL queries created are embedded in the PHP code presented above.

8 USER MANUAL

8.1 How to Navigate Through the Demo Webpage

Once at the demo website for the history of presidents elections database, click on the image in the middle of the welcome page to enter. You will be redirected to the SelectPage where you will be able to select your query. To see what each query does, mouse over its name and a description will appear. If you are interested in the query, click on it and you will be redirected to that query. This is where you can choose your options for your query. There is a navigation bar on the top side of the scree to help you go forward and backward between home page, SelectPage and QueryPage.

8.1.1 Election Year Query

The Election Year Query allows you to input a election year. If the input is a valid election year, it returns information on the year, such as candidates, their party affiliation, percentage of votes received, winner, polls prior to the election etc. If the input is not valid, it displays "It is not a valid input!"

8.1.2 Given Person Query

The Given Person Query allows you to input a candidate or a president name to see related information. If it is a valid name, it displays information including the years he ran, the results, etc. otherwise, it will display "It is not a valid input!" For user's convenience, you can simply try the first name or last name, information of all names beginning or ending with it will be returned.

8.1.3 Query for Re-elected on non-contiguous times

This query returns information about the presidents that were re-elected after losing one or more elections in between. Click on the "Show me the result!" button, information including election year, name, their party affiliation and election result will be returned.

8.1.4 Swing Candidates

This query will find any candidates who ran on one party one time and on another party another time. For such candidates, return the information like, names, the years they run, their party affiliation, results etc. You can simply click on the button to see the result.

8.1.5 Party Historical Query

The Party Historical Query allows you to input a party name to return the results of a party throughout the history of the elections. Along with it, some statistics about the performance of the party wins/loses, electoral votes each time. For candidates had no party affiliation, please enter "none". If it is an valid party name, it will display corresponding information. If it is not a valid party name or no one in this party ever participated in any election, it will display "It is not a valid input!".

8.1.6 Presidents without Election Query

The query finds presidents who took office not through election. It is because that the previous president died or resigned in office. Click on the button, it will display the years, their names and their party affiliation.

8.1.7 Almost President Query Query

This query finds the candidates who almost became a president. It means that these candidates had higher polls prior the election and finally lost to the winner. Click on the button, you will see the years, their names, their party affiliation and the polls and votes contrasted to the winner.

8.1.8 Re-elected President Query

This query will find presidents who were in office on contiguous times. There are two cases: 1. They took office contiguously winning election. 2. They succeeded and then won an election. Click the button to see the result.

8.1.9 Ruling Parties Query

The Ruling Parties Query will display parties with the number of presidents in this party in descending order so that we can simply see which are the ruling parties. Click the button to see the result.

8.1.10 How to update database

1. Update People table
Find out if there are new candidates that never shown in the current People table. If so, use the follow query to update the People table:
INSERT INTO People VALUES(PID, PersonName)

2. Update Parties table
Find out if there are new party among your date that never show in the current Parties table. If so, use the query to update the Parties table:
INSERT INTO Parties VALUES(partyID, partyName)

3. Update President table
To add new presidents to the President table;
(1)first find out the PID of the president using the People table;
(2)second find out the partyID of the president using the Parties table;
(3)then update the Presidents table with the following query:
INSERT INTO Presidents VALUES(PID, year, partyID)

4. Update Candidates table
(1) Find out the PID of candidate using People table;
(2) Find out the partyID of the candidate using Parties table;
(3) Create a new CID for this candidate;
(4) Update the Candidates table with the following query:
INSERT INTO Candidate VALUES(CID,PID, year, partyID)

5. Update results table
(1) Find the candidate CID that the result correspond to using Candidates table;
(2) Find out the corresponding vote, votePercent, poll data for this result;
(3) Update the Results table using the following query:
INSERT INTO Results VALUES(CID, vote, votePercent, poll)

9 TESTING EFFORTS

9.1 Web Interface

First, we need to make sure that the functions and navigation of our website can work well. Second, all queries with input box were tested with various different inputs to make sure that they work and in the case that no results are found for that query for the particular options chosen by the user, then a message is displayed letting the user know that.

9.2 Data Beans

The data beans are used to retrieve the information from the database once the user has given an input or has created a query. The input validation is also done here. We tested to make sure that the connection to the database has not failed.

9.3 SQL Queries

The SQL queries were created using specific examples (i.e. when the input is a year, then using a specific year, etc) and they were tested with different inputs to assure the correctness of the results.

10 SYSTEM LIMITATIONS

The major limitation in developing this project was the lack of time and knowledge about the different technologies that were needed for improvement of the project.

11 POSSIBILITIES FOR IMPROVEMENTS

- Learning the methods to add images to the database and display them on the webpages.
- Improving user experience with our webpage.
- Optimizing the SQL queries.

12 WEB SITE RESOURCES

Our most source of data was the Wikipedia.

https://en.wikipedia.org/wiki/United_States_presidential_election
https://en.wikipedia.org/wiki/List_of_Presidents_of_the_United_States
https://en.wikipedia.org/wiki/List_of_political_parties_in_the_United_States
https://en.wikipedia.org/wiki/Historical_polling_for_U.S._Presidential_elections