Penn State University

Great Valley Campus

Engineering Division

Data Specification for

Campaign Finance Data

[Siva Yogitha Mokkapati- sqm6303@psu.edu]

Version 1.0

[2-2-2020]

**Table of Contents**

[Introduction 3](#_Toc457330103)

[Purpose 3](#_Toc457330104)

[Project Summary 3](#_Toc457330105)

[Requirements Definition 4](#_Toc457330106)

[Considerations 4](#_Toc457330107)

[Document Change Log 4](#_Toc457330108)

[Architecture Design 5](#_Toc457330109)

[Entity Relationship Diagram 5](#_Toc457330110)

[Data Dictionary 5](#_Toc457330111)

[Tables schemas 5](#_Toc457330112)

[Examples of values 5](#_Toc457330113)

[ETL Process 6](#_Toc457330114)

[ETL considerations 6](#_Toc457330115)

[ETL Process Flow with description 6](#_Toc457330116)

[Reporting System 7](#_Toc457330117)

[Conclusions 7](#_Toc457330118)

# Introduction

Requirements definition and decomposition are the important first steps in any systems development project. This document provides the recommended structure in which the project scope and requirements for data warehouse and knowledge management development projects are to be defined and documented.

# Purpose

The purpose of this document is to provide a template for a project specification document for data warehouse and knowledge management development projects.

# Project Summary

This paragraph is used to introduce the following subsections, which can be used for an executive level overview.

1. **Objectives**

To design a data warehouse and Hadoop implementation for the 2016 election contributions made by lobbyists and to make predictions by analyzing the data.

1. **Scope**

Constructing a data warehouse using star schema, implementing Hadoop for storing the data and analyzing the data to make insights about it.

1. **References**

The data is taken from the finance data of federal election commission – United States (fec.gov)

# Requirements Definition

1. **Goals**

* Building the data warehouse by designing dimension tables and fact tables in Dbeaver.
* Extract, transfer and load the data to the postgres database (warehouse) using KNIME.
* Implementing Hadoop for data extraction and storing the data in HIVE.
* Analyze the data from both HIVE and postgre using KNIME to answer the business questions and comparing the results.

1. **Business Questions**

* What are the trends in election contributions, and can we make any prediction based on the data available?
* What are the types of individuals that contribute to elections and how these populations evolved over time?
* Which month has the highest contribution amount (transaction amount)?

1. **Data Requirements**

We are considering US elections 2016 data of six consecutive weeks. The data is a subset of the entire dataset taken as per our need.

1. **Design Constraints**

* There should not be any loss of information while extracting and loading the data into the data warehouse and to HIVE from hadoop.
* The datatypes should be compatible for all the attributes.

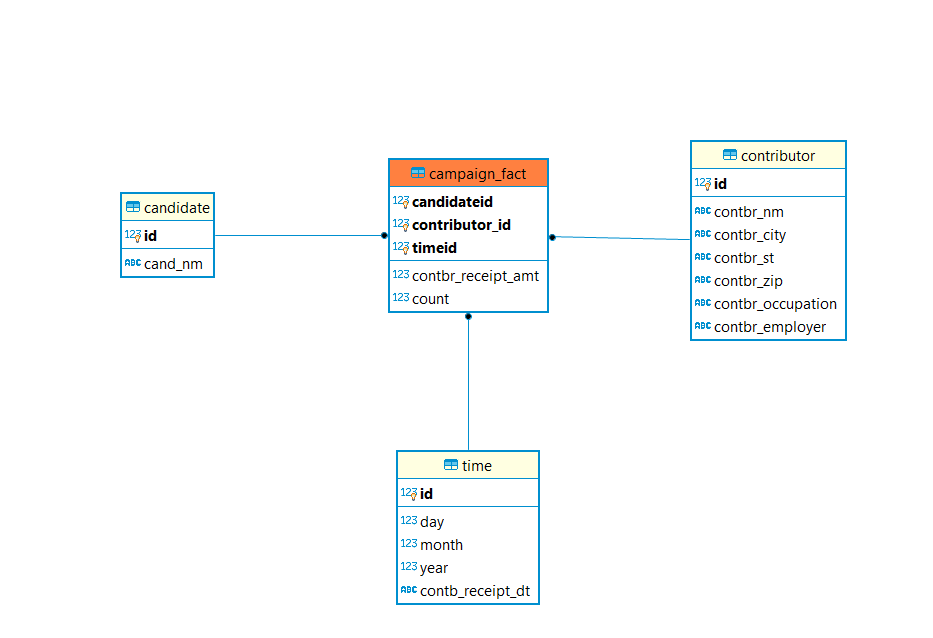
# Document Change Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Change Date** | **Version** | **CR #** | **Change Description** | **Author and Organization** |
| 02/02/20 | 1.0 |  | Initial creation. | Siva Yogitha Mokkapti |
| 03/01/2020 | 2.0 |  | Revision | Siva Yogitha Mokkapati |

# 2. Architecture Design

## 2.1 Relational Data Warehouse

### Design and schema



Star schema is implemented with fact table and dimension tables. The fact table is the center of the star schema.

Three-dimension tables are created they are time, candidate, contributor.

* Time dimension contains the day, month, year and date of the when the transaction happened.
* Contributor dimension has the contributor details like name, city, street, zip, occupation and employer.
* The candidate dimension has candidate id and candidate name.

Fact table contains the ids of the other three-dimension tables, and they become the primary key of the table. Along with these two other columns contrb\_receipt\_amount which the contribution amount and count which is the number of transactions made by the same contributor to the same candidate on the same day.

### Tables schemas

|  |  |  |  |
| --- | --- | --- | --- |
| ***Name of the table*** | Candidate |  |  |
| **Description** | This table gives information about the candidate | | |
| **Attribute** | **Description** | **Type** | Examples of values |
| **Id** | Id of the table | Serial | Between 1 and 999999999 |
| **Cand\_nm** | Name of the candidate | String | John |
| **Primary Key** | Id | | |
| **Candidate Keys** (if any) | Cand\_id | | |
| **Foreign Keys** | Id | | |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Name of the table*** | Contributor |  |  |
| **Description** | This table gives information about the contributor | | |
| **Attribute** | **Description** | **Type** | Examples of values |
| **Id** | Id of the contributor | Integer | Between 1 and 999999999 |
| **Contbr\_nm** | Name of the contributor | String | John |
| **Contbr\_Occupation** | Contributor occupation | String | HomeMaker |
| **Contbr\_city** | Contributor city | string | Braintree |
| **Contbr\_st** | contributor state | String | MA |
| **Contbr\_zip** | Contributor zip | string | 12184 |
| **Contbr\_employer** | employer of contributor. | string | Self employed |
| **Primary Key** | Id | | |
| **Candidate Keys** (if any) |  | | |
| **Foreign Keys** | Id | | |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Name of the table*** | time |  |  |
| **Description** | This table describes the date details of the transaction | | |
| **Attribute** | **Description** | **Type** | Examples of values |
| **Id** | Id of the table | Serial | Between 1 and 999999999 |
| **Day** | Day of the date | Integer | Between 1 and 7 |
| **Month** | Month of the date | Integer | Between 1 and12 |
| **year** | Year of the date | String | 2012 |
| **Contbr\_receipt\_amt** | date | string | 1 |
| **Primary Key** | Id | | |
| **Candidate Keys** (if any) |  | | |
| **Foreign Keys** | Id | | |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Name of the table*** | Campaign\_fact |  |  |
| **Description** | This table describes the details of the fact table(campaign\_fact) | | |
| **Attribute** | **Description** | **Type** | Examples of values |
| **Candidateid** | Id of the candidate | Integer | Between 1 and 999999999 |
| **Contributorid** | Id of the contributor | Integer | Between 1 and 999999999 |
| **Timeid** | Id of the date | Integer | Between 1 and 999999999 |
| **Contrb\_receipt\_amt** | Contribution amount of the transaction | Integer | Between 1 and 999999999 |
| **count** | number of transactions made by the same contributor to the same candidate on the same day | Integer | Between 1 and 9999 |
| **Primary Key** | (**Candidateid, Contributorid, dateid**) | | |
| **Candidate Keys** (if any) |  | | |
| **Foreign Keys** | (**Candidateid, Contributorid, dateid**) | | |

## 2.2 Hadoop Implementation

The input files (6 consecutive files are taken from the last) are downloaded from the canvas to the local system. The files are then moved to the appropriate container using **tar -cv \*.txt | docker exec -i #container id tar x -C /tmp.**

The files are then moved to the input directory in Hadoop using put command.

**docker cp Poem1.txt d1fbc1ca5149:/tmp/Poem1.txt.**

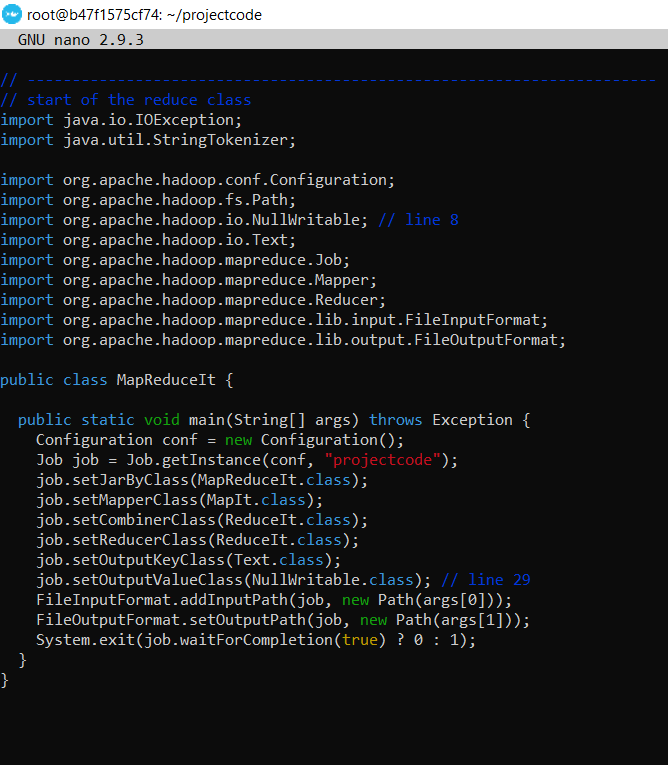
Then MapIt, ReduceIt and MapReduceIt are written to get the appropriate input required for reporting.

The purpose of MapReduce is to:

* Selects the files that should be used for input for the MapReduce job.
* Identify the data splits necessary for the completion of the job, and
* Passes the data splits to the mapper functions

MapReduceIt:

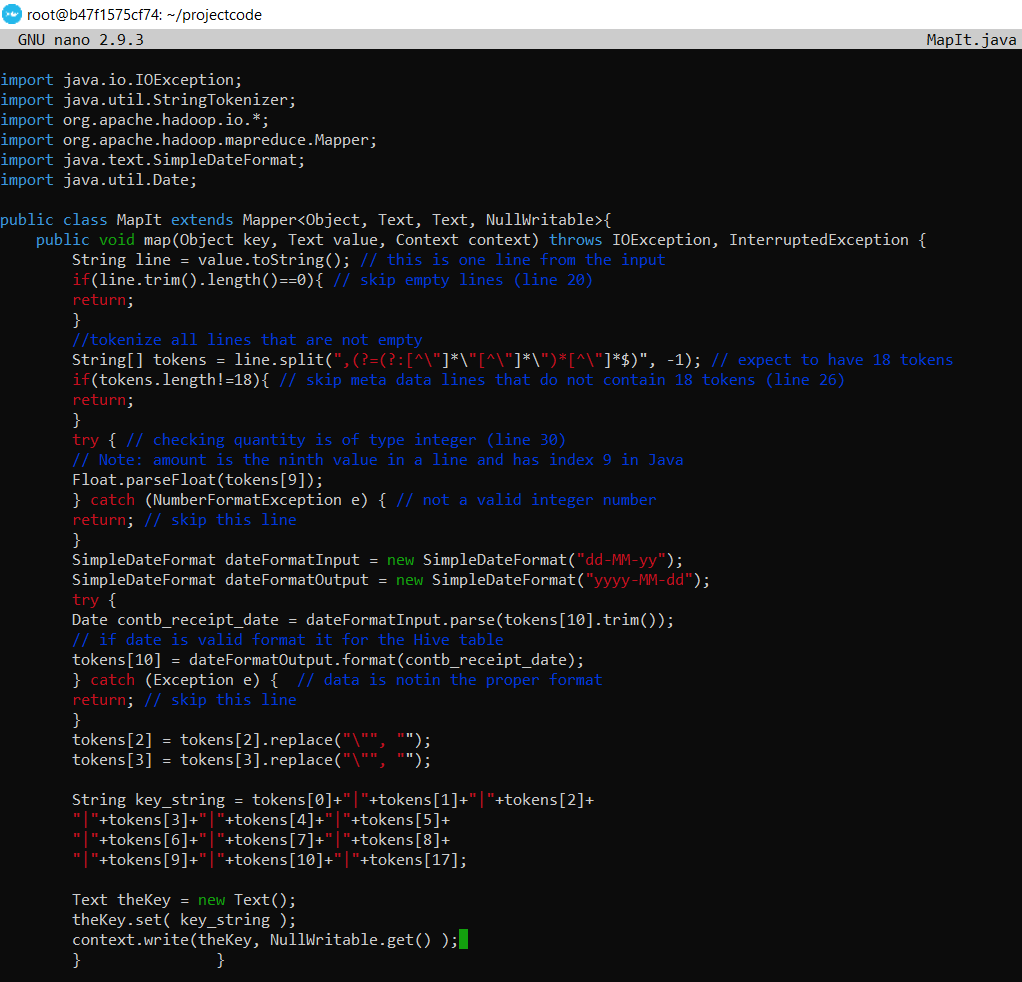
The path for the Map and Reduce java files is given in the MapReduceIt.java file. The Mapper, Combiner and reducer classes are allocated to Map and reduce files as per required. The data splits required are identified.



MapIt.java:

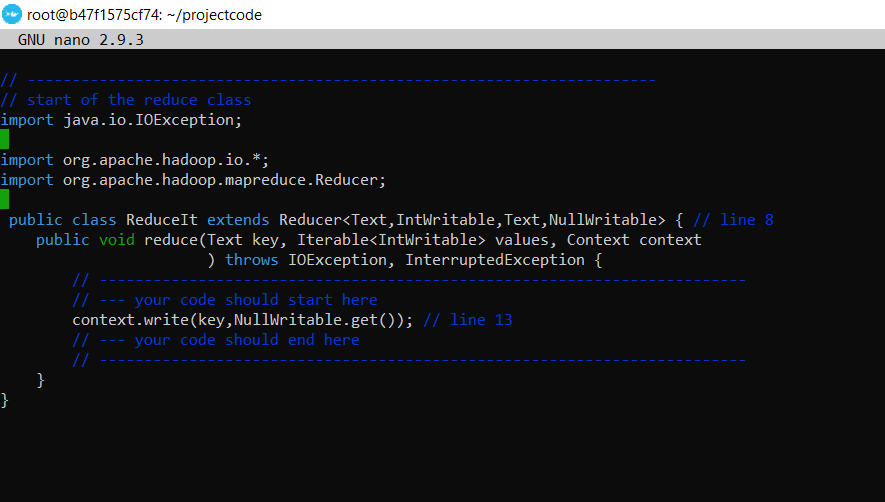
In MapIt, each row of every file is read line by line. For each row, empty lines are skipped and then row is split to columns using a regular expression. After splitting we have 18 tokens which are the columns of the dataset. The date columns format is changed to YYYY-MM-dd. The transaction receipt amount is changed to float as the default tokens are considered string.

Remaining tokens are left as such. The key is the combined string of all the required tokens and value is null. The key is created using a pipeline as a separator between tokens (columns).



ReduceIt.java

 Reducer is responsible for reducing the values associated with intermediate keys.



The instances of OutputFormat provided by Hadoop write to files in HDFS. This can be done by first compiling the java classes and adding them to a jar file named mri.jar. Below are the commands for doing that.

**hadoop com.sun.tools.javac.Main \*.java**

**jar cf mri.jar \*.class**

Next the files are run using**:**

**bin/hadoop jar /root/projectcode/mri.jar MapReduceIt   /user/root/project/input (input file path) /user/root/project/output (output file path)**

After this the result is in output file.

The tokens considered are:

|  |  |  |  |
| --- | --- | --- | --- |
| ***Name of the table*** | Election\_campaign |  |  |
| **Description** | This table describes the details of the of the final output attributes | | |
| **Attribute** | **Description** | **Type** | Examples of values |
| **Cand\_nm** | Name of the candidate | VARCHAR | CLINTON |
| **contbr\_nm** | Name of the Contributor | VARCHAR | Yogitha |
| **contbr\_city** | Name of the Contributor City | VARCHAR | Malvern |
| **contbr\_st** | Name of the Contributor State | VARCHAR | Pennsylvania |
| **contbr\_zip** | Contributor ZIP code | VARCHAR | 324234 |
| **contbr\_employer** | Contributor Employer Name | VARCHAR | pavan |
| **contbr\_occupation** | Contributor Occupation | VARCHAR | Retired |
| **contb\_receipt\_amt** | Contribution Amount | FLOAT | 567 |
| **Contb\_receipt\_dt** | Date of the Transaction | VARCHAR | 12 October 2016 |
| **Cmte\_id** | Committee ID | VARCHAR | C1214 |
| **Cand\_id** | Candidate ID | VARCHAR | C13 |
| **Election\_tp** | Election type | VARCHAR | type |

## 2.3 Reflective analysis of using a data warehouse vs Hadoop.

* In data warehouse all the facts are in the fact table. Frequency changing data is stored in fact tables and slowly changing data is stored in dimension tables. In Hadoop all types of data are stored in a single file whether it is slowly changing or fast changing
* In data warehouse, not all the attributes are loaded. Almost all the unstructured data is removed. In contrast in Hadoop almost all the data is dumped because of its ability to process all kinds of data.
* Hadoop is fast compared to postgres database.
* Data loaded into data warehouse should be clean while in Hadoop any king of data can be loaded.

3. Data Preparation

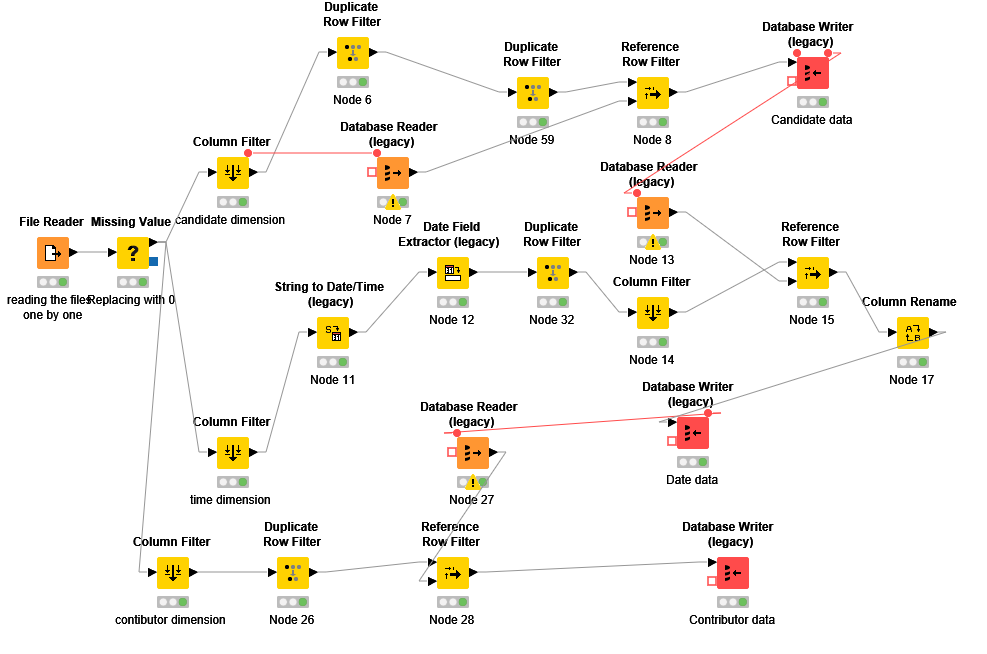
## 3.1 Relational Data Warehouse Implementation

## ETL considerations

* Loading the data using KNIME without losing information.
* Data types and column names should be same in both Data warehouse and database.
* Should not violate any foreign key constraints while loading the fact table.

## ETL Process Flow

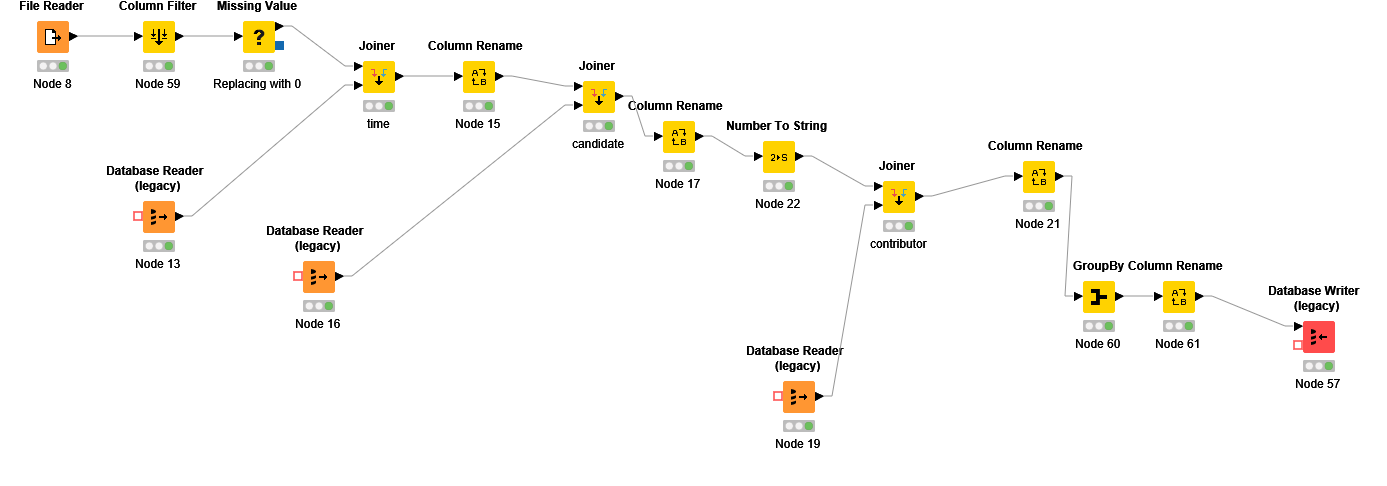
Dimension tables:



**Procedure for loading data into dimensional table:**

* The last 6 csv are loaded into knime **one after another** (loading one file at a time) using file reader.
* The missing values are replaced for all columns with 0. I did this instead of removing them because I faced a problem while loading data into fact table as combination of primary keys did not exist for some rows. Hence, I did the missing value treatment for both dimensional and fact tables in the same way, that is replacing with 0.
* Filtered the columns for the first dimension (candiate) table as necessary and removed the duplicates.
* Reference row filter is used between the values in the database and the rows that are read from csv file so that same data is not loaded twice. (This usually happens when we re-run the workflow for the same csv file).
* Then the data is pushed into the database using database writer.
* Database writer is used to store the preprocessed data into the postgres tables.
* The same procedure from column filter is used for all the tables expect for time as a string to date time and time date extractor is used in the middle to get the month day and year fields.
* String to date time is used to convert a string into date format.
* Date field extractor is used to extract day, month and year from date.
* Column rename is used to rename a column to match column name in database.
* Workflow is established while loading tables into the database such that data is loaded into one table upon completion of other.

Fact table:



**Procedure for loading data into fact table:**

* The last 6 csv are loaded into knime **one after another** (loading one file at a time) using file reader.
* The missing values are replaced for all columns with 0. (Reason- As explained above)
* A joiner is used between table in the database and the csv file read using the common fields that are unique. This is done to get the ids from the dimension table which is one of the primary keys and foreign keys of the fact table.
* Column names are renamed based on requirement.
* This procedure is repeated for all the dimension table
* Group by is used to group all the ids of the dimension tables and based on the grouped columns an aggregation is performed (sum, count) to on the contribution.
* Data is finally pushed to the fact table.

## 3.2 Hadoop Implementation

Hive is used to store the large amount of data generated using hadoop.

**Hive Implementation:**

The data is stored in a text file which is an output of the Hadoop MapReduce. This file is taken as an input to hive.

Hive is a high-level component and sits on the top of the Hadoop. Hive converts the queries to MapReduce jobs and submits them to the cluster. Every time a query is submitted to Hive the Meta store is also updated.

Database named project is created and a table name election\_campaign is created using the following syntax:

**CREATE TABLE project2.elections(**

**CMTE\_ID VARCHAR(9),**

**CAND\_ID VARCHAR(9),**

**CAND\_NM VARCHAR(200),**

**CONTBR\_NM VARCHAR(200),**

**CONTBR\_CITY VARCHAR(30),**

**CONTBR\_ST VARCHAR(2),**

**CONTBR\_ZIP VARCHAR(9),**

**CONTBR\_EMPLOYER VARCHAR(38),**

**CONTBR\_OCCUPATION VARCHAR(38),**

**CONTB\_RECEIPT\_AMT FLOAT,**

**CONTB\_RECEIPT\_DT VARCHAR(15),**

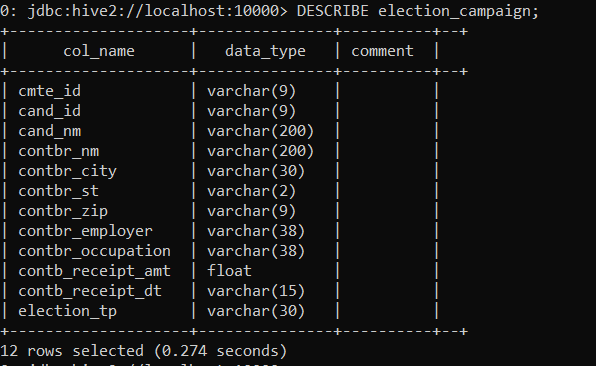
**ELECTION\_TP VARCHAR(30),**

**)ROW FORMAT**

**DELIMITED FIELDS TERMINATED BY '|'**

**STORED AS TEXTFILE;**

We can see that description of the using the Describe election\_campaign; command and the output is below:

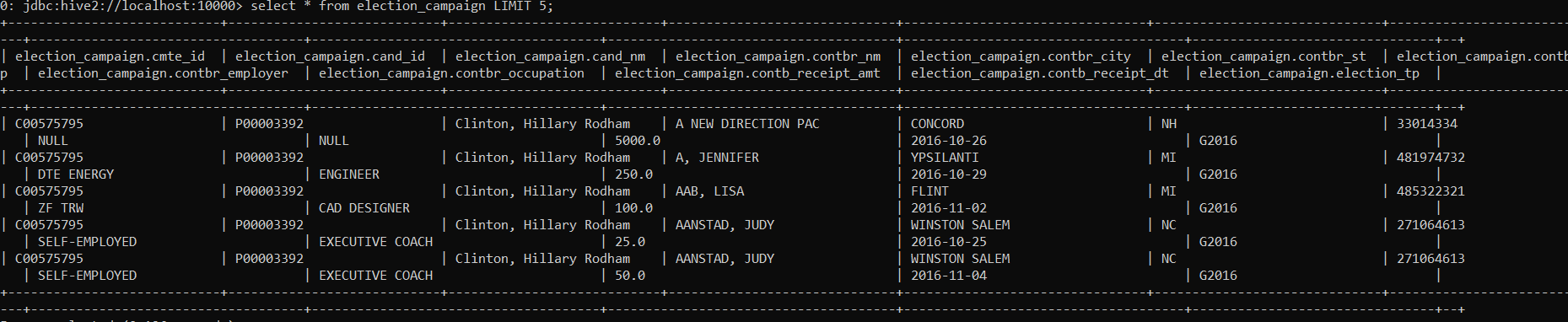


The output file from Hadoop is loaded into the hive table using:

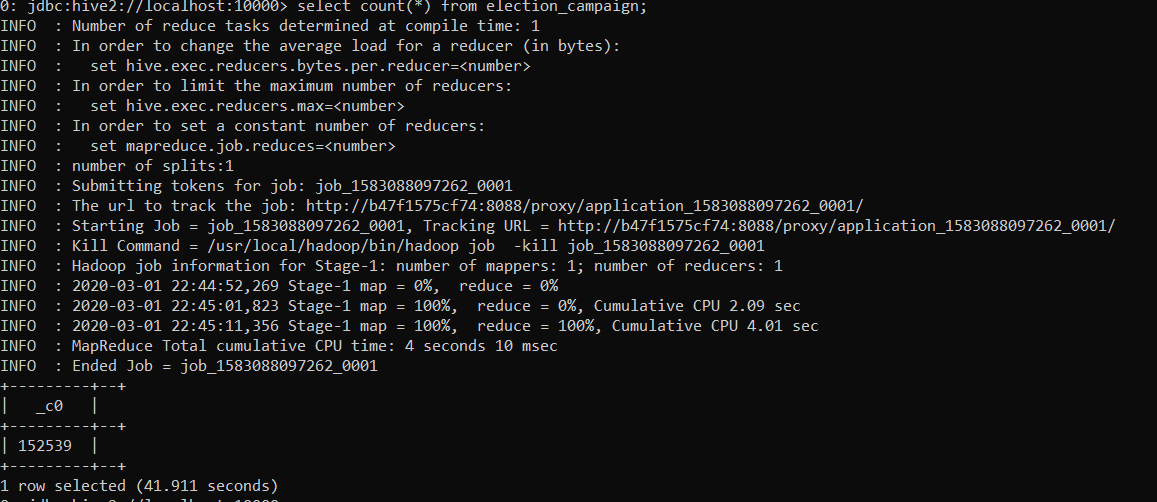
**load data local inpath '/tmp/output.txt' overwrite into table project.ELECTION\_CAMPAIGN;**

The first 5 rows can be seen using:

**select \* from election\_campaign LIMIT 5**; and the result is as below:



The total number of rows in the table can be seen as **select count(\*) from election\_camaign**; which is 152539 rows.



The empty values are converted to null values using the following command:

**alter table project.ELECTION\_CAMPAIGN set tblproperties ('serialization.null.format'='');**

## 3.3 Reflective analysis of data preparation in relational data

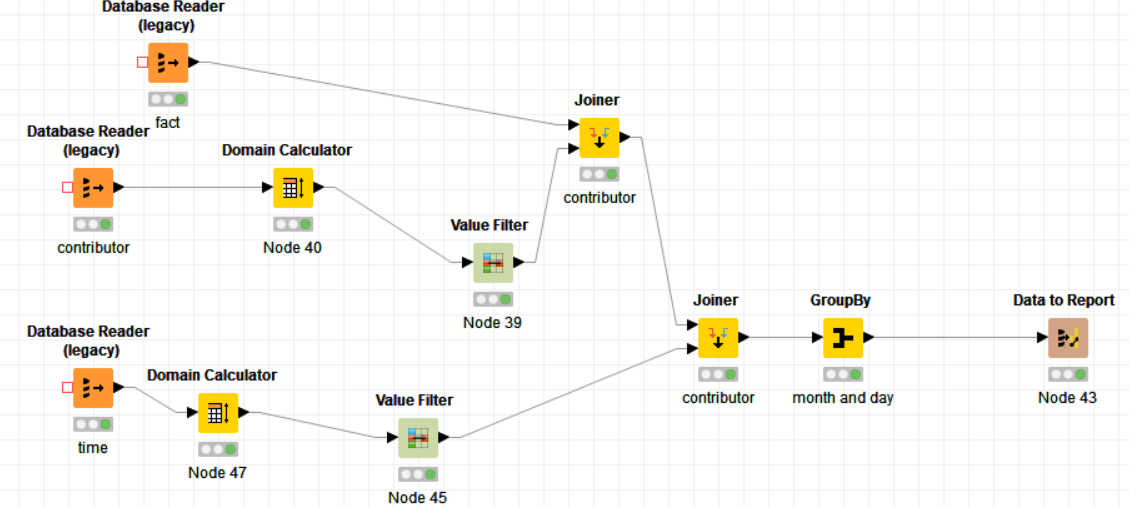
## warehouse vs Hadoop.

* In Data Warehouse the data is not directly loaded as a file. Instead, it is broken down into dimension and fact tables.
* In Hadoop all the data is loaded into a single output file using mapper and reducer class.
* Data manipulation is easy with hive rather than postgres.
* In data warehouse, once the data is prepared and loaded into database, we will not have to look at the initial data file. We only do report from the data we loaded into the database. Hence, any errors made while loading will lead to wrong interpretations. In Hadoop data is never discarded. The original data is always used.

4. Reporting System

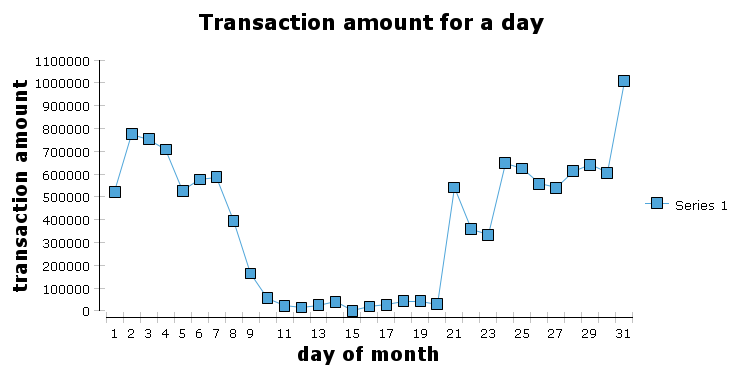
## 4.1 Relational Data Warehouse Implementation

**Question 1:** What are the trends in election contributions, and can we make any prediction based on the data available?



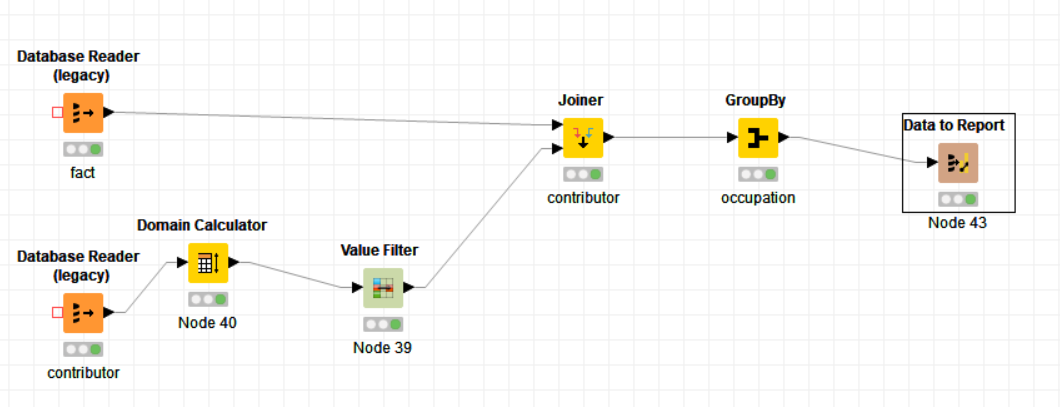
Derived the ids and transaction amount from the fact table and merged it with state in the contributor table and again joined this with day, month and year in the timetable. This is then grouped by day, month and year and a report is made. Value filter for data is used to include only month 10.

Below is the bar graph of the report to see the day wise trend of month 10 (October):

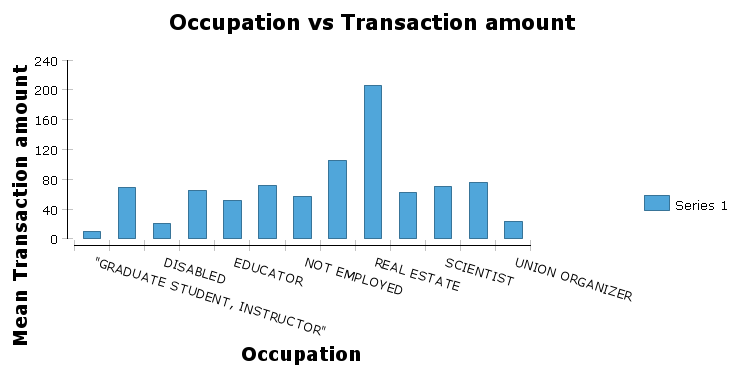


Interpretation: We can see that the transaction amount increased from October 20. The reason for this can be that- Elections were held on November 8. Usually transaction will be more before the elections. This could be the reason for the increase. The future trend could be that it is increasing even more till November 7 and decreases after that.

**Question2:** What are the types of individuals that contribute to elections and how these populations evolved over time?

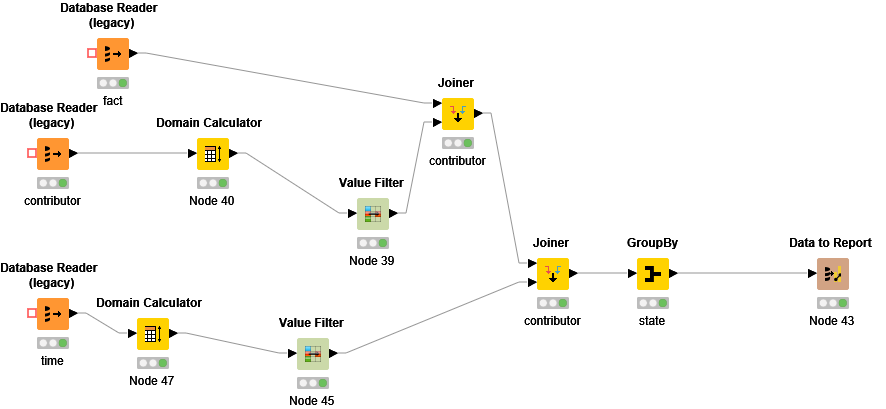


Derived the ids and transaction amount from the fact table and merged it with occupation in the contributor table. This is then grouped by occupation and a report is made. Value filter is used to filter only few occupations. This is because there are too many occupations and plotting all of them would create chaos.

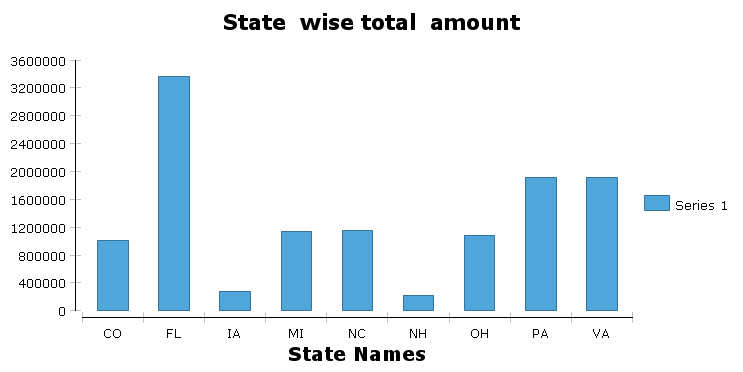


Interpretation: We can see than the mean of transaction amount is the highest from real estate people. Also, from the data to report table we can see that the number of people contributing is almost the same. We can interpret that the same people are contributing every time.

**Question 3:** Which month has the highest contribution amount (transaction amount)?

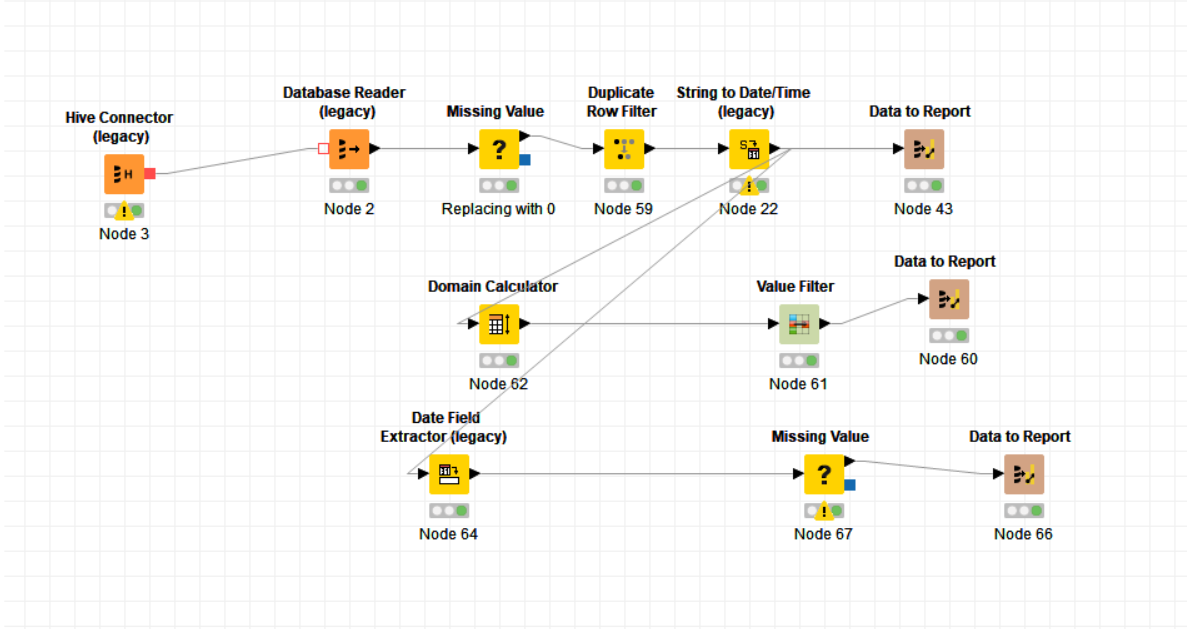


Derived the ids and transaction amount from the fact table and merged it with occupation in the contributor table using id and again joined this with month and year in the timetable. This is then grouped by the required columns and a report is made. Even though value filter is used, no filtering was done there. All the columns were included in value filter.



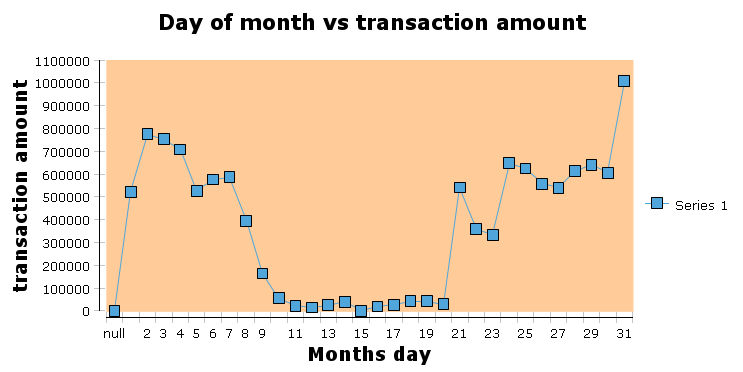
The transaction amount is highest in Florida followed by Pennsylvania and Virginia. It is least in New Hampshire.

## 4.2 Hadoop Implementation



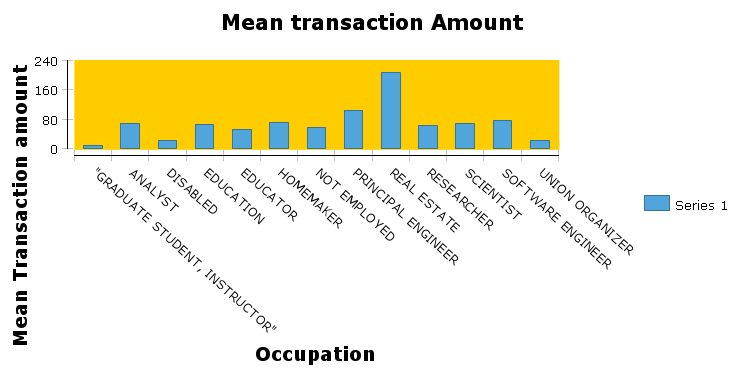
* Data from the HIVE table is loaded to KNIME. This is done using HIVE connector and database reader.
* HIVE connector is used to connect to the database and database reader is used to read data from the table.
* Missing values are replaced with 0 like done for data warehousing.
* Duplicate rows are removed using duplicate row filter.
* String to data and time legacy is used to covert the date to from varchar format to data and time format. This is because in hive table data is stored as string instead of date.
* After this data cubes are created, and plots are plotted.

**Question 1:** What are the trends in election contributions, and can we make any prediction based on the data available?



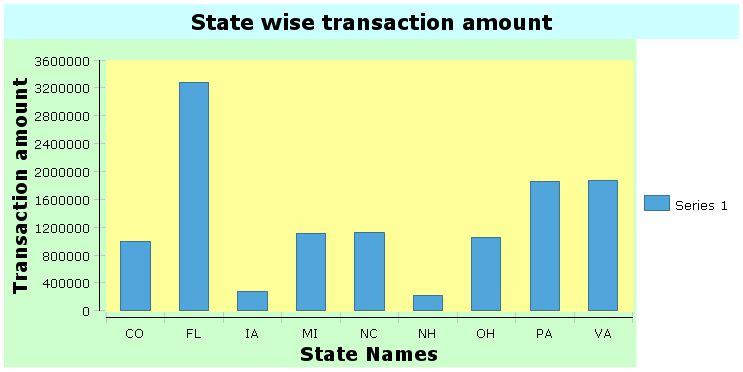
Interpretation: We can see that the transaction amount increased from October 20. The reason for this can be that- Elections were held on November 8. The future trend could be that it is increasing even more till November 7 and decreases after that.

**Question2:** What are the types of individuals that contribute to elections and how these populations evolved over time?



We can see than the mean of transaction amount is the highest from real estate people. We can interpret that the same people are contributing every time.

**Question 3:** Which month has the highest contribution amount (transaction amount)?



The transaction amount is highest in Florida followed by Pennsylvania and Virginia. It is least in New Hampshire.

## 4.3 Reflective analysis of result in relational data warehouse vs Hadoop.

* The reports generated by HIVE and postgres are similar
* Data cubes are used for reports for data loaded from HIVE instead of group by used in postgres.
* Workflow for generating reports from HIVE data is easier compared to postgres.

Conclusions

* There is a specific trend with respect to transaction amount in the month of October. The amount and count of transactions is more as it is nearing November 8. We can predict that it might reduce after November 8th (Election day).
* The Highest amount of contributions were made by real estate occupation people. This is true as real estate people are usually rich, and we can predict that the trend with be the same with respect to occupation.
* Florida is the state from which most election contribution was received. It was followed by Virginia for the months October and November.

**Project 1: Data warehousing:**

Advantages:

Data warehouse converts data from different sources into a consistent format. As the data from across is standardized, everyone will produce results that are consistent. Anyone can query the data themselves with little to no IT support, saving more time and money.

Disadvantages:

Preprocessing and Reporting the data stored in data warehouse is usually time consuming. In data warehouse, we only do report from the data we loaded into the database. Hence, any errors made while loading will lead to wrong interpretations.

**Project 2: Hadoop**

Advantages:

Hadoop is a scalable storage platform as it can store and distribute large datasets across many of inexpensive servers that operate in parallel. As a result, data retrieval is fast in Hadoop. Hadoop can store all kinds of data while data warehouse cannot.

Disadvantages:

Not everyone can handle a Hadoop file system. It requires IT support. Hadoop is not compatible for reading small files.

.