Penn State University
Great Valley Campus

Engineering Division

Data Specification for
Data Warehouse – Part 1

Version 1.0

Date started - 09/15/2023

Modified from template at www.dpw.state.pa.us

Table of Contents

Data Warehouse using PostgreSQL
by Eric Brown

INTRODUCTION

A data warehouse is a centralized repository for storing and managing large volumes of structured data from various sources within an organization. Thus, making it an integral part of an information system. Its primary purpose is to support business intelligence (BI) and data analytics activities by providing a structured and optimized environment for data storage, retrieval, and analysis. Data warehouses are typically made of several data sources, some being remote. Though, data warehouses attempt to minimize their remote query selections.

While data warehouses bear many benefits, one challenge they face occurs during the design phase of their development. Which involves determining the correct number of virtual tables/views to initially create because of data warehouses' finite storage capacity. Which is also why it is not normally possible to store all data inside of them locally. And ultimately why views included in data warehouses are chosen in a manner that can answer queries of interest in a relatively fast time frame.

PURPOSE

Overall, the scope of this term project is to compare two different types of information systems. One being a data warehouse and one being a big data application framework called Hadoop. In this project we will focus on the creation and analysis of the data warehouse system using PostgreSQL. Then in the second part of this project we will focus on the usage of Apache's Hadoop. To see how the usage of clustering across multiple computers makes the analyzation process of large datasets in parallel more quickly amongst big data infrastructures.

PROJECT SUMMARY

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

A. Objectives

[Provide a concise description of the objectives of the document.]

Is to use the data from 2020 election contributions from the Propublica website to answer questions about the trends in election contributions amongst political parties.

B. Scope

[Briefly describe the scope of the requirements specification.]

The data requirements include the following:

- contributions must be for at least two candidates from each party
- contributions must be from at least four states of your choice
- contributions must be from three time periods which are to be loaded separately. These time periods can be weekly, monthly, or quarterly.

C. References

[Identify sources of information used to develop this document, such as IEEE or project documentation.]

- Project 1 Documentation
- Propublica
- PostgreSQL Documentation
- DBeaver Documentation
- KNIME Documentation
- Project 2 Documentation
- Hadoop Documentation
- Apache Pig Documentation
- Apache Hive Documentation

D. Outstanding Issues

[Provide a succinct list of issues or problems which are known to be outstanding with this revision.]

REQUIREMENTS DEFINITION

A. Goals

[[Provide a clear list of the expectations of the new application(s), both in terms of what must be improved and what must be retained from the current processes. All detailed requirements should address one or more of these goals]].

To provide a relational database management system (RDBMS) where the historical election data is stored in a star schema. And the necessary method to

perform the ETL (Extract, Transform, Load) or CRUD (create, retrieve, update, delete) process on this system will be conducted through standard SQL commands. In conjunction to the use of Knime as a tool to make the ETL process more efficient and less cumbersome. Since Knime has a wide variety of connectors and data transformation nodes to automate the ETL process.

Another goal/requirement for this project is to use the freeware open-source relational database management tool called PostgreSQL. This software will be used because it is free to use, and its freeware nature makes it consistent with Hadoop. Which is a tool that is planned to be used in the next phase of the project. Also, PostgreSQL is highly extensible in terms of custom data types, functions, and operators. While also being ACID compliant in terms of data integrity and reliability. And compatible across various types of operating systems. Whether it be used on Windows, macOS, or Linux, PostgreSQL runs in the same manner. While providing the same security measures to protect data from unauthorized access across all operating systems.

A third goal of this project is to use PostgreSQL adherence to SQL standards to generate analysis on the historical data given in this project. This analysis would be conducted through different SQL commands and Knime's Report Writer tool ad-on to answer various prescribed business questions about the elections of 2020.

The last goal for this project is to use PostgreSQL as a freeware relational database management tool to stay consistent with the freeware nature of Hadoop. Mainly because the Hadoop software is to be used in the next portion of this project. Which involves using data warehouse systems to analyze big data.

B. Usability Requirements

[[Specify the requirements associated with ease of use, query capabilities, report layouts, online help and other interfaces to users and/or supervisors]].

Query Capabilities:

- PostgreSQL supports SQL (Structured Query Language) and provides a wide range of SQL features for querying and manipulating data. For example, users can perform complex queries on multiple tables and perform subqueries, joins, and aggregations on multiple tables.
- PostgreSQL supports the creation and execution of stored procedures and user-defined functions using various procedural languages such as PL/pgSQL, PL/Python, and PL/Java.
- PostgreSQL also allows for Window functions. Which is important because window functions do not cause rows to become grouped into a

single output row — the rows retain their separate identities. Behind the scenes, the window functions can access more than just the current row of the query result. Which allows for advanced data analysis. As well as reporting by partitioning and ordering data within result sets.

- PostgreSQL also supports various indexing techniques, including B-tree, Hash, GIN, GiST, and SP-GiST, which can enhance a query's performance.

Report Layouts:

- By itself PostgreSQL does not provide report layout capabilities. However, one could create custom reports by designing SQL queries to extract and format data in a manner that would be shown on a report.
- Reporting tools like JasperReports, Tableau, or Power BI can be integrated with PostgreSQL to design and generate reports with advanced layout features.
- One can use tools like LaTeX, in conjunction to PostgreSQL to generate formatted documents from database queries to create complex report layouts.
- In conjunction with PostgreSQL, Knime will be used mainly for constructing the design for the report segment of this project.
- Report Designer is an extension of the KNIME Analytics Platform. It is used to create custom reports and documents based on the data preprocessed through Knime.
- The Report Designer uses concepts of Business Intelligence and Reporting Tools (BIRT) to generate the overall structure of the report through two segments. The Master Page, which contains information for headers, footers, and titles common to every page in the report. And the Layout, which details how tables, charts, images, text, and other items are arranged.
- One could also bind report elements to one's KNIME data. Meaning the content of one's report could be dynamically updated as one's data or analysis changes. This is particularly useful when you have regularly updated data and want to automate the reporting process.

Online Help:

- PostgreSQL has extensive online documentation available on its official website. This documentation covers installation, configuration, SQL reference, and more.
- One tool that could be used as help is PostgreSQL's pg_dump utility. It could be used to generate a SQL script containing the schema and data of a given database, which can serve as documentation to one's database.
- The KNIME Analytics Platform has many online resources for troubleshooting, learning, and community engagement. Here are some of the common online support resources for KNIME: KNIME Forum, KNIME Documentation, and KNIME Hub. As well as KNIME Blog and KNIME events/webinars.

Interfaces to Users:

- PostgreSQL provides a command-line tool called psql that allows users to interact with the database using SQL commands through the command line.
- There are several GUI tools available, such as pgAdmin and DBeaver that provide a user-friendly interface for interacting with a PostgreSQL database.
- Developers can integrate PostgreSQL into applications using programming languages like Python, Java, and PHP by using appropriate libraries and drivers.
- Web applications can be built using frameworks like Django, Ruby on Rails, or Node.js, with PostgreSQL as the backend database.
- Users can connect to PostgreSQL databases through ODBC (Open Database Connectivity) or JDBC (Java Database Connectivity) drivers, enabling database interaction from various applications.
- PostgreSQL has both web and mobile accessibility through REST API
- KNIME has various user interfaces for users to interact with the platform as well. These interfaces include KNIME's desktop interface, webportal, executor, and REST API.

Additional Requirement to ease computer memory:

1.) On a Windows computer, if one sets docker with WSL, one needs to set some reasonable resource constraints on what WSL2 can actually use.

2.) Create a .wslconfig file (no extension) with the following:

- memory=4GB

IMPORTANT: no more than half of the ram in your computer, minimum 2.5 GB

- processors=2

IMPORTANT: no more than half the virtual processors you have

3.) To confirm these optimization changes, one should Open Windows PowerShell with admin rights, restart WSL2 by typing, *Restart-Service LxssManager*

C. System Security Requirements

[[Provide details of the security classification of the data handled by the system, special handling required for the data, and the types and levels of protection and control required for user access to the data]].

A Docker image runs a PostgreSQL database server by:
(1) pulling the base of a Linux distribution such as Ubuntu
(2) modifying it to include the PostgreSQL database and all other dependencies
(3) making a new image out of that can be pushed to the Docker Hub

Once Docker creates the image, it can be used to create a container. Then the container is created in the user's file system and a read-write layer is added to the image. Also, the creation of a docker file can allow a network interface to be created to allow the local host to be connected through a user's port. This way an IP address is attached to the container and PostgreSQL database can run through the command line. In our project the local host will be through port 49168 on the user's local machine.

In terms of Third-Party Extensions:
Security measures will be in place based on the initial connection to DBeaver through the following authorization.
Host: localhost
Port: *49168*
Database/Schema: student
User name: student
Password: student

D. Business Questions

[[Identify the business questions that should be answered by the software product. Include any prioritization of these requirements]].

1.) Which campaigning party had the highest contributions in terms of dollars between the four states of California, New Jersey, Florida, and Arizona between the month of January and November 2019

2.) Which campaigning party had the lowest contributions in terms of dollars amongst the four states of California, New Jersey, Florida, and Arizona between the month of January and November 2019

3.) Which state had the highest contributions to the top two campaigning parties between the month of January and March 2019

4.) Contributions made in California, New Jersey, Florida, and Arizona aggregated monthly between July and December in the year of 2019

5.) Which company made the most contributions in terms of dollars to the top campaigning party in the month of March 2019

6.) What was the overall demographic makeup of the contributors of each of the top campaigning parties in the month of November 2019

E. Data Requirements

[[Identify the data elements and logical data groupings that will be required to answer the business questions above. Include any archiving requirements and the projected level of effort. This section may be supported by a data model and a data dictionary which should be included in an appendix]].

Typically, there are four steps in developing a data warehouse:

1. ***Select the business process:*** *This step is completed by converting the database schema into a set of dimensional models for each discrete business processes.*

   The main question for this project would be to provide information regarding contribution totals for each of the campaigning parties during different time periods of the 2020 election.

2. ***Declare the grain:*** *This step identifies exactly what each record in the fact table represents. For example, we may be interested in daily sales figures.*

   For this project, we will choose weekly aggregation totals. This implies that we need to aggregate all our data sharing the same dimensions in the same month and same week.

3. ***Identify the dimensions:*** *This step identifies those aspects of the facts that are interesting for the analytical process.*

filing_id = Normally, we would use this as a degenerate dimension but it is not aggregated sequentially, so it is irrelevant data

tran_id = We can use this as a degenerate dimension

linenumber = This a dimension that is irrelevant, since it has information that has no importance to our business objectives/findings

flag_orgind = This a dimension that is irrelevant, since it has information that has no importance to our business objectives/findings

date = This is a good dimension that define the grain of our data

cycle = Could be used a dimension to define time period, but it is a static data point

employer = This is a good dimension if we want to identify which employer are active in giving campaign contributions

occupation = This is a good dimension if we want to identify which occupation are active in giving campaign contributions

committee_name = This is an important dimension because it identifies which candidates a given contributor supports in the 2020 elections

org_name = This is an irrelevant dimension since it has no data list inside of it

address_one = This is a dimension that I will consider irrelevant in our case because we want to look only at trends and not at predicting individual behavior

address_two = This is a dimension that I will consider irrelevant in our case because we want to look only at trends and not at predicting individual behavior

last_name = This is a dimension that I will consider irrelevant in our case because we want to look only at trends and not at predicting individual behavior

first_name = This is a dimension that I will consider irrelevant in our case because we want to look only at trends and not at predicting individual behavior

middle_name = This is a dimension that I will consider irrelevant in our case because we want to look only at trends and not at predicting individual behavior

prefix = This is a dimension that could provide information regarding which gender contributed the most during the 2020 elections. But I would still consider it irrelevant for our business questions established previously. Also, there are many null values in this dimension

suffix = This is a dimension that could provide information regarding which profession contributed the most during the 2020 elections. But I would still

consider it irrelevant for our business questions established previously. Mainly because most of the values in this dimension are null values

state = This is a good dimension if we want to identify which part of the country supports a particular type of candidate

zip = This is a good dimension if we want to delve deeper into identifying which part of the country supports a particular type of candidate

city = This is a good dimension if we want to delve deeper into identifying which part of the country supports a particular type of candidate

amount = This is not a dimension but rather a fact

aggregate_amount = This is not a dimension but rather a fact. But I would probably not want to use this aggregate total because it aggregates the amount field in a different manner than what we need for our business questions

memo_code = This is a dimension that is irrelevant, since it has information that has no importance to our business objectives/findings

memo_text = This is a dimension that is irrelevant, since it has information that has no importance to our business objectives/findings

back_ref_tran_id = This is a dimension that has similar information to another dimension in our database, tran_id. So, I would consider it not necessary to include as another dimension in our data warehouse

back_ref_sched_name = This a dimension that is irrelevant, since it has information that has no importance to our business objectives/findings

prigen = This a dimension that is irrelevant, since it has information that has no importance to our business objectives/findings

fecid = This a dimension that is irrelevant, since it has information that has no importance to our business objectives/findings

In conclusion we should create a star schema with eight dimensions:

- Tran_id
- Date
- Employer
- Occupation
- Committee_name
- State
- Zip
- City

For each of these dimensions we will create a table containing an id and a description field.

4. ***Identify the facts:*** *This step identifies what actionable data we need to store.*

   Amount = is the main fact that we are trying to create our analysis around; total monetary contributions that were made to each political party

F. Design Constraints

[[Document any design constraints that should be taken into consideration during the system design phase]].

There must be at least three tables loaded into the PostgreSQL database. Create a star schema with eight dimensions:

- Tran_id
- Date
- Employer
- Occupation
- Committee_name
- State
- Zip
- City

CONSIDERATIONS

(May include as separate document)

DOCUMENT CHANGE LOG

Change Date	Version	CR #	Change Description	Author and Organization
09/24/23	1.0		Initial creation	Eric Brown
10/05/2023	2.0		Add more table details about data warehouse	Eric Brown
10/11/2023	3.0		Add schema details about data warehouse	Eric Brown
10/20/2023	4.0		Add information about ETL process	Eric Brown

10/14/2023	5.0		Add information regarding the report design and discoveries learned	Eric Brown
10/15/2023	6.0		Final edits for Project 1 submission	Eric Brown
10/22/2023	7.0		First creation of Part 2 of Project – Hadoop Implementation	Eric Brown
10/29/2023	8.0		Add information about Hadoop table architecture	Eric Brown
11/14/2023	9.0		Revised Part 1 of Project from comments made by professor	Eric Brown
11/23/2023	10.0		Add information about Hadoop data preparation	Eric Brown
11/29/2023	11.0		Add more information about ETL process for part 2 of project	Eric Brown
12/2/2023	12.0		Add report information generated by Hive table in KNIME	Eric Brown
12/4/2023	13.0		Update information regarding the reports generated for part 2 of project	Eric Brown
12/6/2023	14.0		Final edits for Project 2 submission	Eric Brown
12/8/2023	15.0		Additional final edits for Project 2 submission	Eric Brown

2. ARCHITECTURE DESIGN

2.1 Relational Data Warehouse

Data Dictionary

A Data Dictionary is a document that describes the basic organization of a database. Typically a data dictionary will contain a list of variables in the database as well as the assigned variable names and a description of each type of variable. A data dictionary should also include the values accepted for each variable and any helpful comment such as important exclusions and skip patterns. The primary use of a data dictionary is for data analysis.

My relational database was made up of four tables that had different contribution information inside of them. The four tables were based on the first quarter contribution results amongst four states. These states were Arizona, New Jersey, California, and Florida. All four tables were organized in the same way and listed below was generally how they were structured.

Contribution Table – Data Dictionary				
Variable	***Variable name***	***Variable type***	***Values***	***notes***
Filling Id	filing_id	String	SA17A, SA18, SB28A	Can not be null
Line number	linenumber	String	SA17A, SA18, SB28A	Can not be null
Organization type	flag_orgind	String	IND	
Organization Name	org_name	String	ADP, 1144 Summit Ave, LLC, Atlantic Agri Imports, LLC	Has mostly null values
Contributor Last name	fname	String	Doe	Can not be null
Contributor First name	lname	String	John	Can not be null
Contributor middle name	middle_name	String	Bruce	
Professional abbreviations	prefix	String	Dr., Mrs, Mr, Judge	

Contribution Table – Data Dictionary				
Professional & Personal abbreviations	suffix	String	III, Jr., JR., M.D.	
Address	address_one	String	12 Larson Rd	
Second Part of Address	address_two	String	Unit 45, Suite 100	
City of Contributor	city	String	Phoenix, Los Angeles, Newark, Orlando	
State of Contributor	state	String	California, New Jersey, Florida, Arizona	must have a California, New Jersey, Arizona, or Florida state String value
Zipcode of Contributor	zip	String	85327	
Employer of Contributor	employer	String	Stagecoach Digital, City University Of New York- John Jay, etc..	
Occupation of Contributor	occupation	String	Engineer, Mechanic, Software programmer, etc…	
Dollar amount of contribution	amount	Numeric	2800, 1000, 26	
Date of contribution	date	Date/Time	3/6/2019, 3/5/2019, 3/31/2019, etc…	

Contribution Table – Data Dictionary				
Aggregated Amount contributed by contributor	aggregate_amount	Numeric	2800, 262.55000	
Memo code	memo_code	String	Null, X	Mostly null values
Memo text	memo text	String	* Earmarked Contribution through ACTBLUE on 01/13/2019, SEE REDESIGNATION	
Transaction Id of contribution	tran_id	Integer	819053, 815124, 815077	
Additional transaction Info	back_ref_tran_id	String	1688351E, 169662E, 1691257E	
Campaign Contribution Year Id	prigen	String	P2020, G2020, Blank	
Campaign Year	cycle	Date	2020	Must be for year 2020
Id of Committee	fecid	String	C00698258, C00694455	
Committee Name	committee_name	String	Kamala Harris For The People, Bernie 2020, Friends Of John Delaney	

Tables schemas

Provide a description of the physical schema of the data warehouse. Use the steps in the lesson and explain.

Name of the table	State		
Description	This table describes information regarding each state of the contributors.		
Attribute	Description	Type	*Examples of values*
id	Id of each unique state	Integer	Between 1 and 12000
name	Name of state	String	California, New Jersey, Arizona, Florida
Primary Key	id		
Foreign Keys	NA		

Name of the table	City		
Description	This table describes information regarding each city of the contributors.		
Attribute	Description	Type	*Examples of values*
id	Id of each unique city	Integer	Between 1 and 12000
city	Name of the city	String	Los Angeles, Key Largo, Tempe, Montclair
Primary Key	id		
Foreign Keys	NA		

Name of the table	Zipcode		
Description	This table describes information regarding each zipcode of the contributors.		
Attribute	Description	Type	*Examples of values*
id	Id of each unique zipcode	Integer	Between 1 and 12000
zip	Zipcode of the city	Integer	33095, 7950, 85266, 90274
Primary Key	id		
Foreign Keys	NA		

Name of the table	Employer		
Description	This table describes information regarding each employer of the contributors.		
Attribute	Description	Type	*Examples of values*
id	Id of each unique employer	Integer	Between 1 and 12000
employer	Employer name of each contributor	String	Town Of Prescott, Self Employed, Whole foods, Not Employed
Primary Key	id		
Foreign Keys	NA		

Name of the table	Occupation		
Description	This table describes information regarding each occupation of the contributors.		
Attribute	Description	Type	*Examples of values*

id	Id of each unique occupation	Integer	Between 1 and 12000
employer_title	Occupation title of each contributor	String	Sales, Planner, Attorney, Investor
Primary Key	id		
Foreign Keys	NA		

Name of the table	Date		
Description	This table displays all the contribution dates made by contributors.		
Attribute	Description	Type	*Examples of values*
id	Id of each unique employer	Integer	Between 1 and 12000
date	Contribution transaction date	Date	9/30/2019, 10/1/2019, 11/4/2019
year	year portion of the date value		2019
quarter	quarter value of the year in which the date was in		1 to 4
month	month portion of the date value		1 to 12 1 = January 12 = December
weekofyear	week value of the date		ranging from 1 to 52
dayofweek	day value of the date value		ranging from 1 to 7 1 being equal to Monday & 2 being equal to Tuesday, etc..

Primary Key	id
Foreign Keys	NA

Name of the table	Transactions		
Description	This table displays all the contribution transactions made by contributors.		
Attribute	Description	Type	*Examples of values*
id	Id of each unique contribution transaction	Integer	Between 1 and 12000
tran_id	Contribution transaction reference	Integer	1764024, 1964608, 454961
Primary Key	id		
Foreign Keys	NA		

Name of the table	Committee_Name		
Description	This table describes information regarding each committee in which the contributors made contributions.		
Attribute	Description	Type	*Examples of values*
id	Id of each unique committee name	Integer	Between 1 and 12000
committee_name	Names of the politicians that contributions were made to in the 2020 elections	String	Kamala Harris For The People, Bernie 2020, Friends Of John Delaney
Primary Key	id		
Foreign Keys	NA		

Name of the Fact table	AmountFacts			
Description	This table will store the fact variable - contribution amount in the data warehouse			
Attribute	Description	Type	*Examples of values*	*Notes*
stateid	State id	Integer	10	From the dimension table
cityid	City id	Integer	150	From the dimension table
zipcodeid	Zipcode id	Integer	500	From the dimension table
employerid	Employer id	Integer	1000	From the dimension table
occupationid	Occupation id	Integer	25	From the dimension table
dateid	Date id	Integer	45	From the dimension table
transactionid	Transactions id	Integer	1200	From the dimension table
committeeid	Committee id	Integer	60	From the dimension table
amount	Contribution amount by each contributor	Numeric(6,2)	-2838.81, 800.90, 600	Already available
Primary Key	Combination of stateid, cityid, zipcodeid, employerid, occupationid, dateid, transactionid, committeeid			

Foreign Keys	stateid REFERENCES id in State table cityid REFERENCES id in City table zipcodeid REFERENCES id in Zipcode table employerid REFERENCES id in Employer table occupationid REFERENCES id in Occupation table dateid REFERENCES id in Date table transactionid REFERENCES id in Transactions table committeid REFERENCES id in Committee_Name table

Please try use at least two types of input formats (SQl and text)

2.2 Hadoop Implementation

To be completed with project number 2

My Hadoop implementation of the 2020 presidential elections database was structured slightly differently than my PostgreSQL implementation. However, the data dictionary of the election contributions was still the same. The only structure that changed was the overall schema of the data warehouse. The change that occurred was because of two reasons: Hive had slightly different data types than PostgreSQL and Hive typically did not validate primary and foreign key constraints. Only since Hive's recent 2.1.0 update release had it included support for primary and foreign key constraints. But it is still best practice to perform integrity constraints through queries that are executed in Hive.
As a result, I created only one fact table that stored all the 2020 election contribution data for the four states of Florida, New Jersey, California, and Arizona. Though each state's election contribution data was loaded at separate times into the election fact in Hive. This way the data was not concatenated at one moment in time.

Tables schemas

Provide a description of the physical schema of the data warehouse. Use the steps in the lesson and explain.

Name of the table	Elections		
Description	This table describes information regarding each state's election contributions in 2020.		
Attribute	Description	Type	*Examples of values*

amount	Contribution amount by each contributor	Int	-2838.81, 800.90, 600
city	Name of the city	String	Los Angeles, Key Largo, Tempe, Montclair
zip	Zipcode of the city	Int	33095, 7950, 85266, 90274
employer	Employer name of each contributor	String	Town Of Prescott, Self Employed, Whole foods, Not Employed
employer_title	Occupation title of each contributor	String	Sales, Planner, Attorney, Investor
transdate	Contribution transaction date	Date	9/30/2019, 10/1/2019, 11/4/2019
year	year portion of the date value	Int	2019
month	month portion of the date value	Int	1 to 12 1 = January 12 = December
weekofyear	week value of the date	Int	ranging from 1 to 52
tran_id	Contribution transaction reference	Int	1764024, 1964608, 454961
committee_name	Names of the politicians that contributions were made to in the 2020 elections	String	Kamala Harris For The People, Bernie 2020, Friends Of John Delaney

2.3 Reflective analysis of using a data warehouse vs Hadoop.

In project 1, please enter your reflective findings. Final version to be completed with project number 2.

My reflective findings during the design stage of this project included some discoveries of data types which PostgreSQL allowed in its data management system. These values included Boolean types, Character Types (char, varchar, and text), Numeric types (such as integer and floating-point numbers), Temporal types (such as date, time, timestamp), and serial types. I found this finding interesting because I had the assumption that all data management systems used similar data types. Overall, this hypothesis was true to an extent, but there were still some features that I was not aware of that PostgreSQL provided.

One of these features included the data type serial. Before completing this project, I was used to creating auto incrementing columns in tables through AUTO_INCREMENT or using the Identity modifier to increment a column. However, in PostgreSQL the serial data type was used to automatically increment a column value, so it performed like a primary key. This feature was possible because of regex expressions that used an auto generated sequence to increase a column's value sequentially.

Another concept and/or feature of data warehouse tables I discovered was indexing. I learned that indexing played a crucial role in enhancing the performance and efficiency of database operations. Especially when one was designing tables in a data warehouse to handle data from large datasets. In this project I mainly used indexes on columns that served as primary keys or had unique constraints. This way PostgreSQL's data management system could have fast retrieval times for specific rows in these columns.

In terms of my use with Hadoop, I learned that it does not create traditional databases. However, Hadoop is a powerhouse in terms of distributed storage and processing for large-scale, unstructured, or semi-structured data. Since Hadoop's primary components of Hadoop Distributed File System (HDFS) and MapReduce, enable the storage and parallel processing of vast amounts of data. Also, in comparison to PostgreSQL, Hadoop doesn't enforce a schema on the data; it allows storing and processing diverse data types, making it suitable for big data analytics.

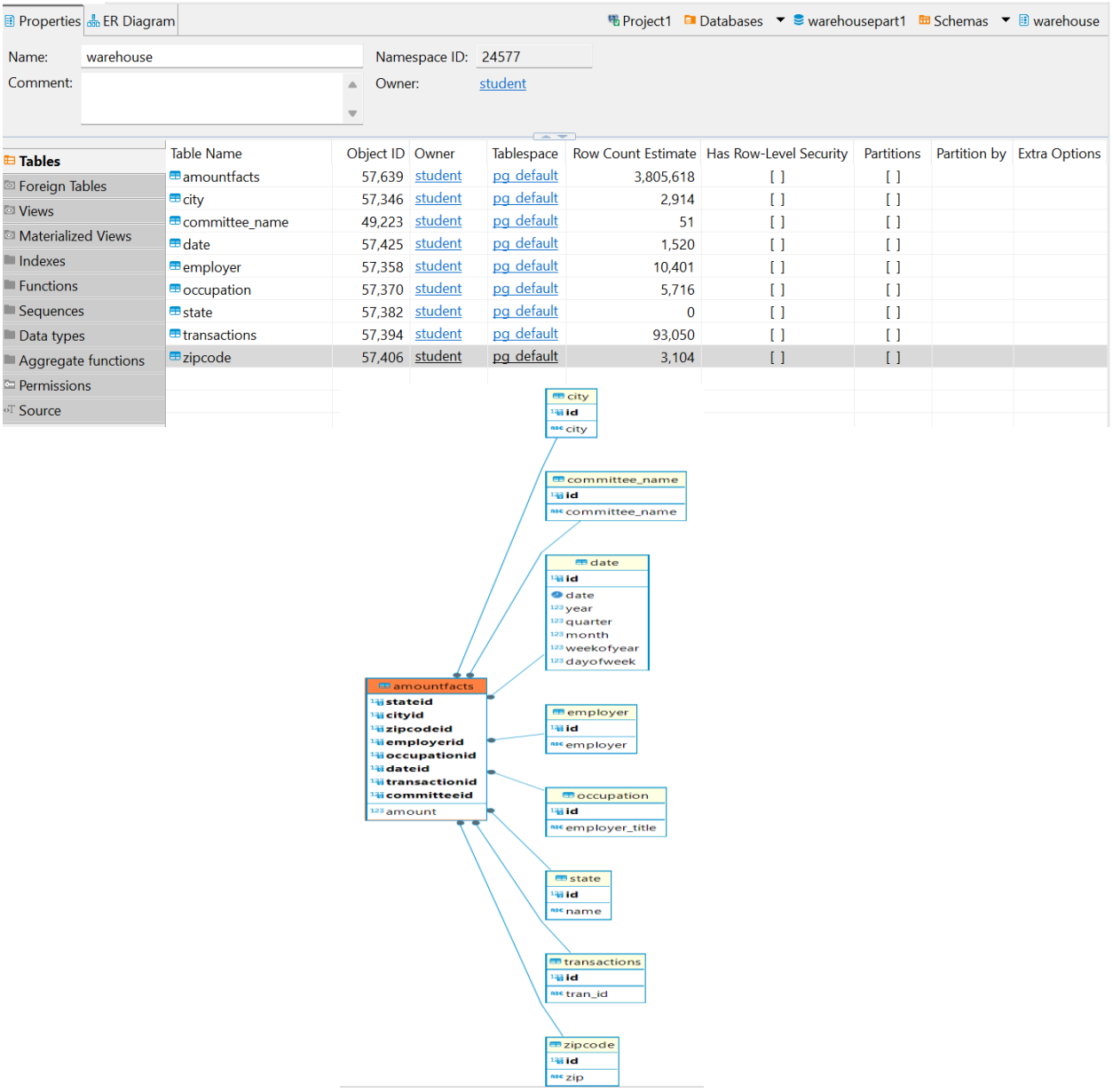
Also, in terms of scalability, I learned that one could add more nodes to a Hadoop cluster to properly scale its storage and processing capabilities. Which was an aspect lacking in PostgreSQL because adding more resources to a single server could become cumbersome.

In terms of Hadoop's ecosystem, I learned that tools, such as Apache Spark, Hive, and Pig, provide higher-level abstractions for various data processing tasks. Such as

Hadoop's MapReduce process used to access, transform, and load big data stored in Hadoop File System (HDFS).

Lastly in terms of Hadoop's ecosystem, I learned that tools, such as Apache Spark, Hive, and Pig, provide higher-level abstractions for various data processing tasks. Such as Hadoop's MapReduce process used to access, transform, and load big data stored in the Hadoop File System (HDFS).

Overall Schema of PostgreSQL Datawarehouse:



Overall Schema of Hadoop Datawarehouse:

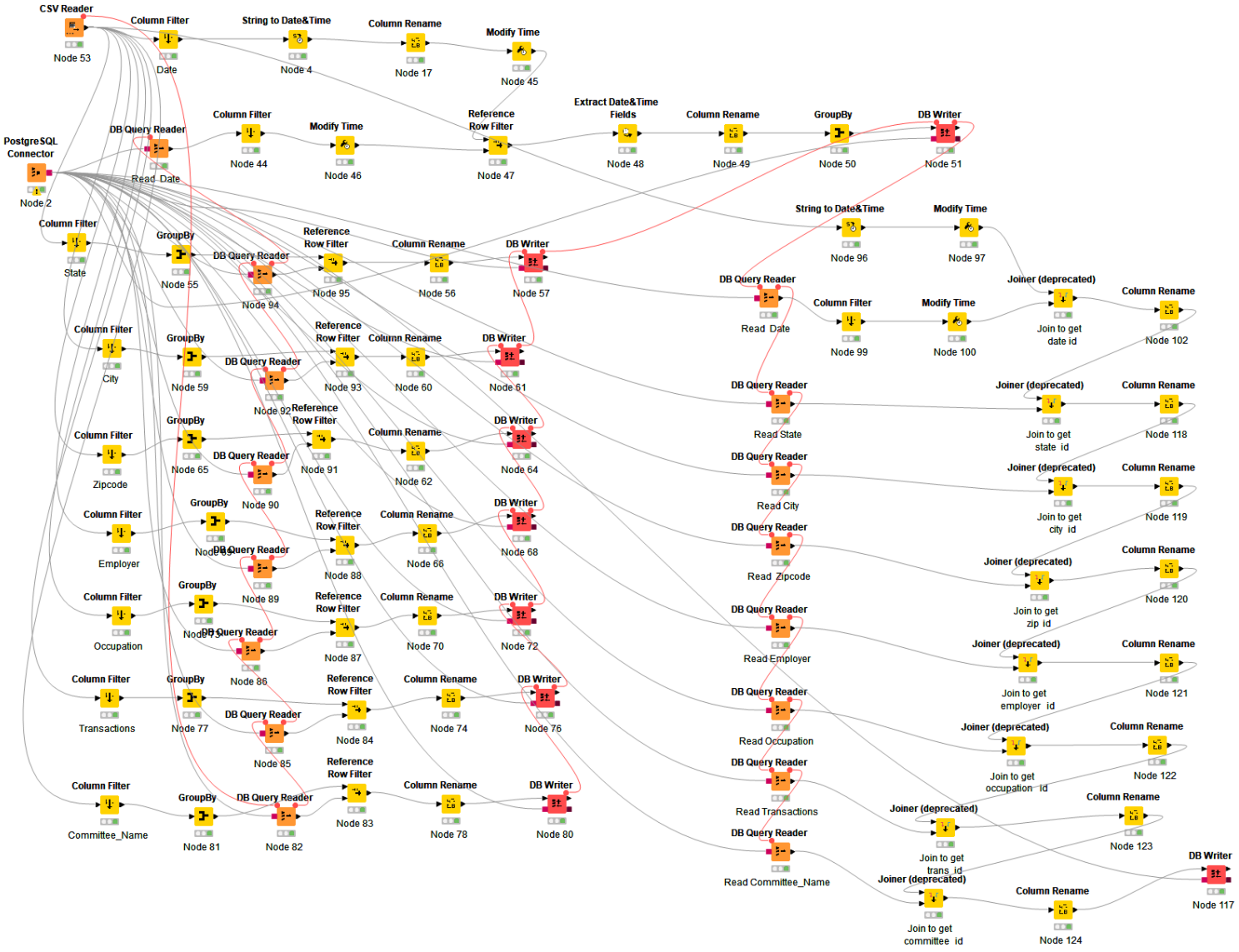
```
CREATE TABLE election (
   amount INT
   state_name VARCHAR(40),
   city VARCHAR(40),
   zip SMALLINT,
   employer STRING,
   employer_title VARCHAR(50),
    tran_id SMALLINT,
   committee_name STRING,
   transdate DATE,
   year SMALLINT,
   month SMALLINT,
   weekofyear SMALLINT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;
```

```
0: jdbc:hive2://localhost:10000> describe election;
+-----------------+-------------+----------+
|    col_name     |  data_type  | comment  |
+-----------------+-------------+----------+
| amount          | int         |          |
| state_name      | varchar(40) |          |
| city            | varchar(40) |          |
| zip             | int         |          |
| employer        | string      |          |
| employer_title  | varchar(50) |          |
| tran_id         | int         |          |
| committee_name  | string      |          |
| transdate       | date        |          |
| year            | smallint    |          |
| month           | smallint    |          |
| weekofyear      | smallint    |          |
+-----------------+-------------+----------+
```

3. Data Preparation

3.1 Relational Data Warehouse Implementation ETL considerations

ETL Process Flow with description

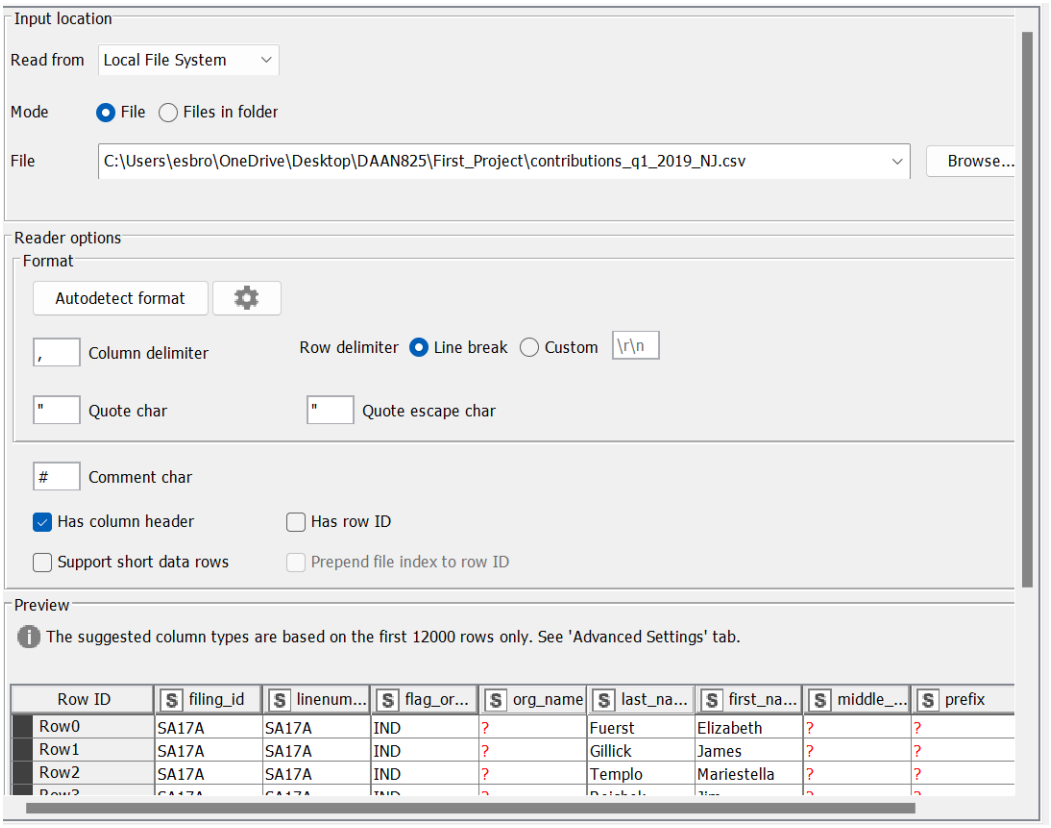


As one could see this ETL process included many steps to draw data from the initial source (4 election contribution data files). Then transfers the data into the data

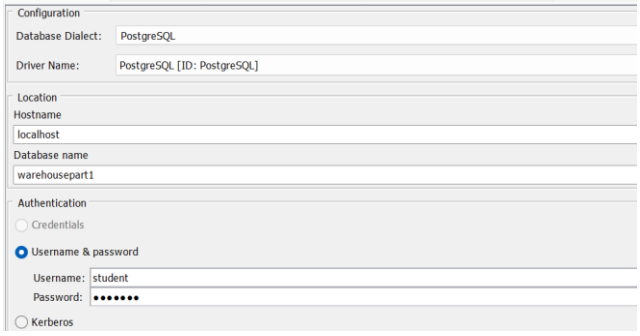
warehouse under warehousepart1 project created from the student database under the schema warehouse in PostgreSQL.

Step 1: Extract the data from the input files

In the first step I used a CSV reader to access the data from each of the four comma delimited files stored on my computer of the four state's election contributions data.



Then I used a PostgreSQL connector to access and connect to the data in the data warehouse in PostgreSQL using the following credentials:



Step 2: Transform and Load data in the Dimension Tables

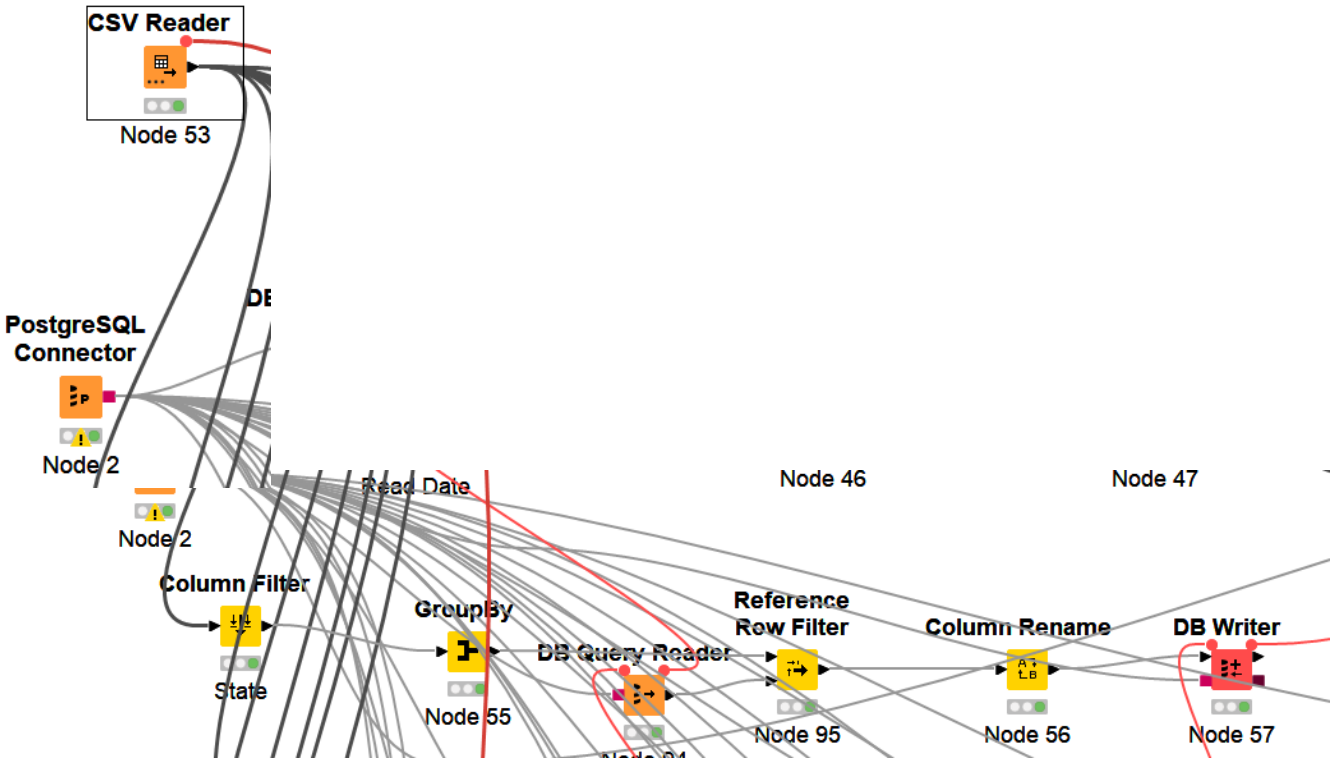
Then I had to look at the dimensional data, for example State. The following was needed for each dimension:

- Remove all other columns except for the dimension
- Keep only unique values for the dimension
- Save the dimensions in the PostgreSQL table

Note that you may get some errors if:

- You save the data file to a different path
- Your Database runs under a different IP address

In Knime the corresponding operators were found in the Manipulation section. Though, you could still search for them by name. As an example, below is an ETL flow I created for loading data in the “State” table.



I used a similar approach for the seven other dimensions that included city, zipcode, transactions, committee_name, employer, and occupation to load the data into the data warehouse. For the date dimension I needed to follow a different approach since the dates were transient and every ETL will add new dates.

Once the ETL completed I viewed the top couple of rows to make sure the data was compiled correctly by KNIME. Displayed below are portions of data displayed in each dimension.

Input Data with Write Status - 4:57 - DB Writer

File Edit Hilite Navigation View

Table "default" - Rows: 1 Spec - Column: 1 Properties Flow Va

Row ID	S name
Row0	NJ

Table "default" - Rows: 767 Spec - Column: 1 Properties Flow Variables

Row ID	S city
Row0	ABSECON
Row1	ALLENDALE
Row2	ALLENHURST
Row3	ANDOVER
Row4	ASBURY PARK
Row5	ATCO
Row6	ATLANTIC HIGHLANDS
Row7	AUDUBON
Row8	AVENEL
Row9	AVON BY THE SEA
Row10	Aberdeen
Row11	Absecon
Row12	Allendale
Row13	Allentown

Row ID	I zip
Row0	0
Row1	7001
Row2	7002
Row3	7003
Row4	7004
Row5	7005
Row6	7006
Row7	7007
Row8	7008
Row9	7009
Row10	7010

Row ID	S employer_title
Row0	?
Row1	638 STEAMFITTER
Row2	ACCOUNT ANALYST
Row3	ACCOUNT EXECUTIVE
Row4	ACCOUNTANT
Row5	ACCOUNTING
Row6	ACTUARY
Row7	ADMIN ASSISTANT
Row8	ADMINISTRATION
Row9	ADMINISTRATIVE
Row10	ADMINISTRATIVE ASSISTANT
Row11	ADMINISTRATOR

Row ID	S employer
Row0	?
Row1	1919Investment Counsel
Row2	1975
Row3	21St Century Fox / Truex
Row4	221 Direct
Row5	24 EAST 73 LLC
Row6	39 New York Ave, LLC
Row7	7 Toy Drive
Row8	79 Hudson St LLC
Row9	A La Carte Premier Servers LLC
Row10	A.P.G. SECURITY, L.L.C.
Row11	A1A CLAIMS SVCS

Row ID	S tran_id
Row0	1000010
Row1	1000015
Row2	1000017
Row3	1000032
Row4	1000037
Row5	1000038
Row6	1000042
Row7	1000050
Row8	1000051

Table "default" - Rows: 16 | Spec - Column: 1 | Properties | Flow Variables

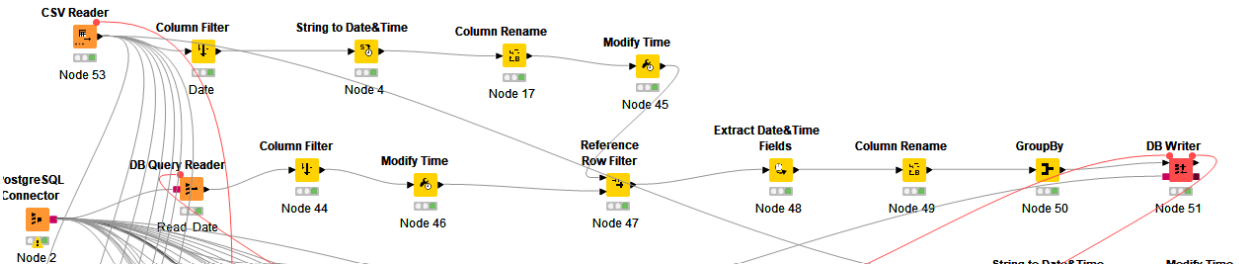
Row ID	committee_name
Row0	Amy For America
Row1	Bernie 2020
Row2	Beto For America
Row3	Cory 2020
Row4	Donald J. Trump For President, Inc.
Row5	Friends Of Andrew Yang
Row6	Friends Of John Delaney
Row7	Gillibrand 2020
Row8	Hickenlooper 2020
Row9	Inslee For America
Row10	Julian For The Future
Row11	Kamala Harris For The People
Row12	Marianne Williamson For President
Row13	Pete For America, Inc.
Row14	Tulsi Now
Row15	Warren For President, Inc.

Step 3: Load Data in the Date Dimension Table

To load data in the facts table, I needed to:

- Load the date dimension into dimension table
- Change all the dimension names with dimensions ids
- Remove data that was not needed

For this, I needed to first read in the data file and the dimension tables. The dimension tables needed to be joined with the data from the data table so I could find each dimension's id variable. The final workflow I created is shown below.



The most interesting operator here was the "Reference Row Filter" node that performed a "minus" operation on data, meaning it kept only rows that existed in the first table but not in the second table. That was necessary because I needed to add only dimensions that were not in existence in the dimension table.

Another addition was the red line between the Excel Reader and the PostgreSQL Connector nodes on the left. This connection, displayed through the red line, delayed

the execution of the PostgreSQL connection till the successful completion of the Excel Reader node. Since I needed to get a new connection every time, I changed the input file. For reference to enable this feature one has to right click on each node and select "Show Flow Variable Ports" and then connect each respective port.

Step 4: Load Data in the Facts Table

Once I added the dates, I loaded the data in the facts table. The flow at the very beginning of the ETL section of this project shows the flow for loading the election data in the warehouse schema data warehouse.

Again, the most interesting node is the “Joiner” node that I used to join the data with the dimension tables, on description/name to the id of the dimension. A second important operator was the String to date, that converted the string in the file to a date using a mask.

3.2 Hadoop Implementation

To be completed with project number 2

Step 1: Extract & Load in Data

First, I had to copy the election result data files from my host computer to the docker container and then to the Hadoop data filing system (HDFS). I accomplished this task by creating an input folder called elect2020/input through the following commands:

```
hadoop fs -mkdir /user/root/elect2020
hadoop fs -mkdir /user/root/elect2020/input
hadoop fs -ls /user/root/elect2020 #to see if the folder was created properly
```

Next, I had to copy the four election csv files from my host computer to the container and then to the HDFS. To accomplish this task, I used the following commands, while working inside my computer’s command prompt system and not my Docker Hadoop container.

1.) Navigated to where I stored the election data for my Hadoop project:

```
cd C:\Temp\Project2
```

2. Zipped all four csv files into a zip file called elect2020 to then copy all files into my Hadoop container.

```
docker cp '.\elect2020.zip' 2b83lfl5fa94:/tmp/elect2020.zip
```

3. Accessed the Hadoop container using:

```
docker exec –it 2b83lfl5fa94 bash
```

4. In my container, I moved to the folder where the archive was placed:

```
cd /tmp
```

5. Then Unzipped the file:

```
unzip election2020.zip
```

6. Next I checked if the files were created

```
ls /tmp/Data/*.csv
```

Copy Files from Hadoop Container to HDFS

Then, I needed to copy the four election files from the "/tmp" folder to HDFS in the "elect2020/input" folder. For each file I had to issue the following commands; where the "-put" option copied the files from my local storage (/tmp) to Hadoop in the "elect2020/input" folder.

```
hadoop fs -put /tmp/Data/contributions_q1_2019_AZ.csv /user/root/elect2020/input
hadoop fs -put /tmp/Data/contributions_q1_2019_CA.csv /user/root/elect2020/input
hadoop fs -put /tmp/Data/contributions_q1_2019_NJ.csv /user/root/elect2020/input
hadoop fs -put /tmp/Data/contributions_q1_2019_FL.csv /user/root/elect2020/input
```

To check if the files were saved in HDFS I issued the following command:

```
hadoop fs -ls elect2020/input
```

ETL - Transform using Apache Pig

To transform the elections data to be loaded into my single elections table, I used Apache Pig. I decided to use Apache Pig over Hadoop MapReduce because I found its syntax to be easier to understand and debug. Mainly being that Apache Pig uses a high-level scripting language called Pig Latin which is built on top of Hadoop to abstract the complexities of writing MapReduce programs. As a result, I was able to write more concise code compared to if I used Java code to write MapReduce functions for my transformations. The scope of transformation of data will be the following:

- Remove empty lines
- Remove any meta-data lines
- Create user defined variables from original data
- Keep only required columns
- Save result to HDFS

Step 1: Locate were you stored the files for the Hadoop project in the HDFS

```
root@2b831f15fa94:/# hadoop fs -ls
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Found 4 items
drwxr-xr-x   - root supergroup          0 2023-11-06 02:03 poems
drwxr-xr-x   - root supergroup          0 2023-12-02 21:38 projectPart2
drwxr-xr-x   - root supergroup          0 2023-11-19 22:20 week13
drwxr-xr-x   - root supergroup          0 2023-11-06 01:03 wk11UsePig
```

```
Deleted /user/root/projectPart2/output
root@2b831f15fa94:/# hadoop fs -ls /user/root/projectPart2/input
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Found 4 items
-rw-r--r--   1 root supergroup    1646484 2023-11-08 17:54 /user/root/projectPart2/input/electAZ2019.csv
-rw-r--r--   1 root supergroup   11734504 2023-11-08 17:54 /user/root/projectPart2/input/electCA2019.csv
-rw-r--r--   1 root supergroup    5110511 2023-11-08 17:53 /user/root/projectPart2/input/electFL2019.csv
-rw-r--r--   1 root supergroup    2455751 2023-11-08 17:54 /user/root/projectPart2/input/electNJ2019.csv
```

I stored my files in my root directory in Hadoop's file system under the folder projectPart2. Inside of my projectPart2 folder I created an input folder to store all four of my election donation data files for each of the four states. This way Hadoop would be able to find my input files when compiling its map reduce commands in the backend from my pig script commands. Since, as stated Pig scripts are translated into a series of MapReduce jobs that run on a Hadoop cluster.

Step 2: Load pig and read the election files into memory

```
DEFINE CSVLoader org.apache.pig.piggybank.storage.CSVLoader();

elect = LOAD '/user/root/projectPart2/input/*.csv' USING CSVLoader(',') AS (filing_id:chararray,
linenumber:chararray, flag_orgind:chararray, org_name:chararray, last_name:chararray,
first_name:chararray, middle_name:chararray, prefix:chararray, suffix:chararray,
address_one:chararray, address_two:chararray, city:chararray, state:chararray, zip:chararray,
employer:chararray, occupation:chararray, amount:chararray, date:chararray,
aggregate_amount:chararray, memo_code:chararray, memo_text:chararray, tran_id:chararray,
back_ref_tran_id:chararray, back_ref_sched_name:chararray, prigen:chararray, cycle:chararray,
fecid:chararray, committee_name:chararray);
```

Step 3: Check to see the structure of the data using a dump command

```
ranked_elect = rank elect;

elect_first_rows = Filter ranked_elect by ($0 < 10);

dump elect_first_rows;
```

The following code above allowed me to display only the first 9 rows of data from my election dataset

```
(1,amount,state,city,zip,employer,occupation,tran_id,committee_name,date,,,)
(2,500.0,CA,Los Angeles,90064,Efni,Executive,813700,Hickenlooper 2020,2019-03-04,2019,3,10)
(3,500.0,CA,San Francisco,94114,Centricom Llc,Real Estate Developer,814900,Hickenlooper 2020,2019-03-04,2019,3,10)
(4,2800.0,CA,Tiburon,94920,Winston & Strawn LLP,Attorney,813630,Hickenlooper 2020,2019-03-04,2019,3,10)
(5,2800.0,CA,San Francisco,94118,Bay Grove,Investor,817230,Hickenlooper 2020,2019-03-18,2019,3,12)
(6,100.0,CA,Menlo Park,94025,Not Employed,Not Employed,818740,Hickenlooper 2020,2019-03-22,2019,3,12)
(7,1000.0,CA,San Francisco,94123,RETIRED,RETIRED,817940,Hickenlooper 2020,2019-03-15,2019,3,11)
(8,500.0,CA,Berkeley,94704,Not Employed,Not Employed,816040,Hickenlooper 2020,2019-03-06,2019,3,10)
(9,2800.0,CA,San Francisco,94104,Retired,Retired,815280,Hickenlooper 2020,2019-03-06,2019,3,10)
```

From my findings, I was able to see that there was a header row of data in my dataset. And there was a good amount of data that was missing in columns. So, I needed to fix this problem. As well as get rid of columns that I did not need to include in my election table.

Step 4: Get rid of unnecessary columns from original dataset. And create user defined columns for the election table

In my Apache Hive schema, I only included the following fields: amount, state_name, city, zip, employer, employer_title, tran_id, committee_name, transdate, year, month, and weekofyear. The syntax for this is shown below.

```
elect1 = foreach elect generate amount, state as state_name, city, zip, employer,
occupation as employer_title, tran_id, committee_name, date as transdate,
SUBSTRING(date, 0, 4) AS year_str, SUBSTRING(date, 5, 7) AS month_str,
SUBSTRING(date, 8, 10) AS day_str;

elect2= FOREACH elect1 GENERATE amount, state_name, city, zip, employer,
employer_title, tran_id, committee_name, transdate, year_str, month_str, day_str,
CONCAT(CONCAT(CONCAT(year_str, '/'), month_str), CONCAT('/', day_str)) AS
new_date_str;

elect3= FOREACH elect2 GENERATE amount, state_name, city, zip, employer,
employer_title, tran_id, committee_name, transdate, new_date_str,
ToDate(new_date_str, 'yyyy/MM/dd') as dt_obj;

elect4= FOREACH elect3 GENERATE amount, state_name, city, zip, employer,
employer_title, tran_id, committee_name, transdate, dt_obj, GetYear(dt_obj) as year,
GetMonth(dt_obj) as month, GetWeek(dt_obj) as weekofyear;

election= FOREACH elect4 GENERATE amount, state_name, city, zip, employer,
employer_title, tran_id, committee_name, transdate, year, month, weekofyear;
```

In the code above, I kept my nine required fields. And then created my three expression fields based on my original date variable in the election dataset. However, in order to get the correct corresponding year, month, and week number I needed to do the following manipulations shown above. Since, I needed to use the ToDate function to convert the date column into a DateTime object. To then extract the year, month and week using respective Get functions.

Sadly, the ToDate function needed a date variable that had data in yyyy/mm/dd format before it could parse the object. This is why I had to create a string variable that built the original date field in the proper format using multiple concat statements. To eventually pass into the ToDate object. At the end I would get rid of this date object field and string variables used to create the date object. When I rearranged my fields to match my Hive election table.

Output from my pig latin script shown above:

```
(1,amount,state,city,zip,employer,occupation,tran_id,committee_name,date,,,)
(2,500.0,CA,Los Angeles,90064,Efni,Executive,813700,Hickenlooper 2020,2019-03-04,2019,3,10)
(3,500.0,CA,San Francisco,94114,Centricom Llc,Real Estate Developer,814900,Hickenlooper 2020,2019-03-04,2019,3,10)
(4,2800.0,CA,Tiburon,94920,Winston & Strawn LLP,Attorney,813630,Hickenlooper 2020,2019-03-04,2019,3,10)
(5,2800.0,CA,San Francisco,94118,Bay Grove,Investor,817230,Hickenlooper 2020,2019-03-18,2019,3,12)
(6,100.0,CA,Menlo Park,94025,Not Employed,Not Employed,818740,Hickenlooper 2020,2019-03-22,2019,3,12)
(7,1000.0,CA,San Francisco,94123,RETIRED,RETIRED,817940,Hickenlooper 2020,2019-03-15,2019,3,11)
(8,500.0,CA,Berkeley,94704,Not Employed,Not Employed,816040,Hickenlooper 2020,2019-03-06,2019,3,10)
(9,2800.0,CA,San Francisco,94104,Retired,Retired,815280,Hickenlooper 2020,2019-03-06,2019,3,10)
```

Step 5: Delete missing values and inspect data

Next, I needed to delete the lines that had empty values in the fields of interest. While also getting rid of the header line. The syntax is shown below.

```
election = FILTER election BY amount != '';

election = FILTER election BY state_name != '';

election = FILTER election BY city != '';

election = FILTER election BY city != 'city';  -- deletes header

election = FILTER election BY zip != '';

election = FILTER election BY employer != '';

election = FILTER election BY employer_title != '';

election = FILTER election BY tran_id != '';

election = FILTER election BY committee_name != '';

election = FILTER election BY transdate != '';

election = foreach election generate amount, state_name, city, zip, employer, employer_title, tran_id,
committee_name, transdate, year, month, weekofyear;

ranked_elect = rank elect;
elect_first_rows = Filter ranked_elect by ($0 < 10);
dump elect_first_rows; -- inspect first 9 rows
```

Output from my filter statements written above:

```
(1,500.0,CA,Los Angeles,90064,Efni,Executive,813700,Hickenlooper 2020,2019-03-04,2019,3,10)
(2,500.0,CA,San Francisco,94114,Centricom Llc,Real Estate Developer,814900,Hickenlooper 2020,2019-03-04,2019,3,10)
(3,2800.0,CA,Tiburon,94920,Winston & Strawn LLP,Attorney,813630,Hickenlooper 2020,2019-03-04,2019,3,10)
(4,2800.0,CA,San Francisco,94118,Bay Grove,Investor,817230,Hickenlooper 2020,2019-03-18,2019,3,12)
(5,100.0,CA,Menlo Park,94025,Not Employed,Not Employed,818740,Hickenlooper 2020,2019-03-22,2019,3,12)
(6,1000.0,CA,San Francisco,94123,RETIRED,RETIRED,817940,Hickenlooper 2020,2019-03-15,2019,3,11)
(7,500.0,CA,Berkeley,94704,Not Employed,Not Employed,816040,Hickenlooper 2020,2019-03-06,2019,3,10)
(8,2800.0,CA,San Francisco,94104,Retired,Retired,815280,Hickenlooper 2020,2019-03-06,2019,3,10)
(9,1000.0,CA,Santa Barbara,93109,Not Employed,Not Employed,814290,Hickenlooper 2020,2019-03-04,2019,3,10)
```

As shown above, the header lines at the top were removed successfully. And each line contained the same number of non-empty columns. However, some string values contained words that were capitalized, and some words were partially capitalized. I thought I needed to fix this issue because I did not want my reports to show duplicate values for fields based on differences in capitalization of words. So, I decided to capitalize all string variables in my dataset.

Step 6: Clean up dataset and revise column order to match Hive table

To capitalize each string variable, I used upper functions to capitalize each letter of my string variables. Then I removed all irrelevant columns and reordered them to match the order of my Apache Hive table. The syntax is shown below.

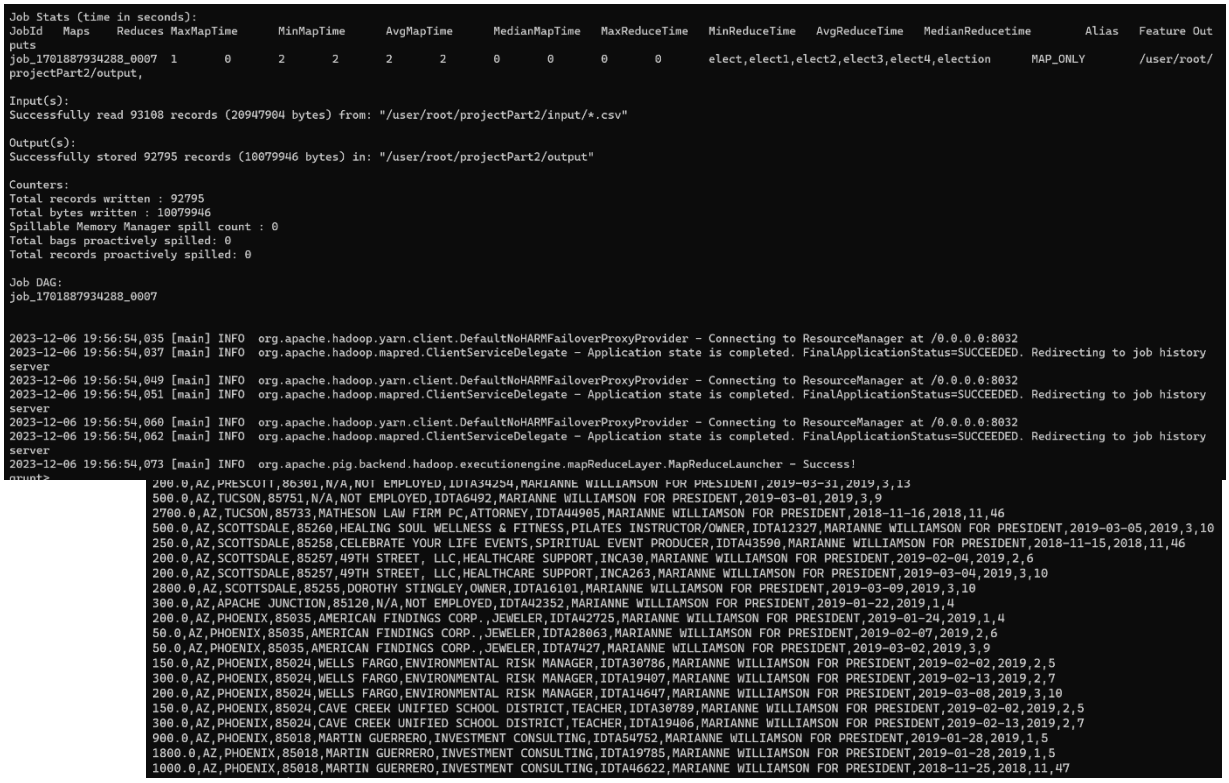
election = FOREACH election GENERATE amount, UPPER(state_name) as state_name, UPPER(city) as city, zip, UPPER(employer) as employer, UPPER(employer_title) as employer_title, tran_id, UPPER(committee_name) as committee_name, transdate, year, month, weekofyear;

ETL Load

Before I could load my data into my Hive table, I had to write the data to the HDFS, using the following command:

```
STORE election INTO '/user/root/projectPart2/output' USING PigStorage (',');
```

Then I inspected the data to make sure everything was copied properly using the follow command: `hadoop fs -cat /user/root/ projectPart2/output/part-m-00000`



As shown above the pig commands successfully compiled through MapReduce. Once the data was loaded in HDFS, I moved it to the Hadoop container and then uploaded it into my hive table. I used the following commands to load my data into my Hive table.

```
hadoop fs -get /user/root/projectPart2/output/part-m-00000 /tmp/projPart2.csv
load data local inpath '/tmp/projPart2.csv' overwrite into table project2.election;
```

```
0: jdbc:hive2://localhost:10000> select count(*) from project2.election;
+--------+
|  _c0   |
+--------+
| 92795  |
+--------+
1 row selected (13.799 seconds)
0: jdbc:hive2://localhost:10000>
```

As a result, 92,795 rows of data were loaded into my election Hive table.

3.3 Reflective analysis of data preparation in relational data warehouse vs Hadoop.

In project 1, please enter your reflective findings. Final version to be completed with project number 2.

Using KNIME I was able to learn how to automate the ETL process through different nodes and functions to save time, reduce manual errors, and ensure that my data integration tasks were performed consistently and efficiently. I enjoyed that KNIME used a graphical interface to make the data extraction, transforming, and loading phases easier to understand, build, and test compared to other software applications I used to conduct this process.

Since the ETL process should be automated in most cases. And should be able to run at fixed times with limited human interaction. I believe I was able to accomplish this objective, because I only had to perform two steps to run the total ETL workflow in KNIME. These two steps included:

1: load each of the four respective new files into the csv node one at a time
2: execute the last DB Writer node on the right

In terms of Hadoop, I was able to learn that Hadoop has many tools such as Pig and Hive to perform various data preparation tasks amongst various types of data. I discovered that Hadoop’s ecosystem was effective because of its Distributed File

System (HDFS) that provides high-output access to application data. While distributing data across multiple nodes in a Hadoop cluster.

From my data preparation process I was able to learn how tools such as Apache Pig and Spark were used for data cleaning and transformation. While still allowing one the capability to write scripts or programs to process and transform raw data into a more suitable format for analysis. Tasks could include cleaning missing values, transforming data types, filtering unwanted records, and more.

In terms of querying data, I learned that Apache Hive allowed one the capability to define a schema and query data using SQL-like language called HiveQL. While not a full-fledged relational database language, HiveQL still provided many SQL-like constructs, making the learning curve easier to overcome with my SQL background.

4. Reporting System

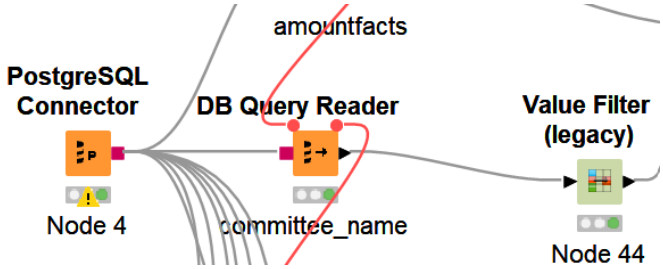
Provide screenshot and sample results with discussion on potential knowledge that was elicited.

4.1 Relational Data Warehouse Implementation

Retrieving and aggregating the data from the data warehouse

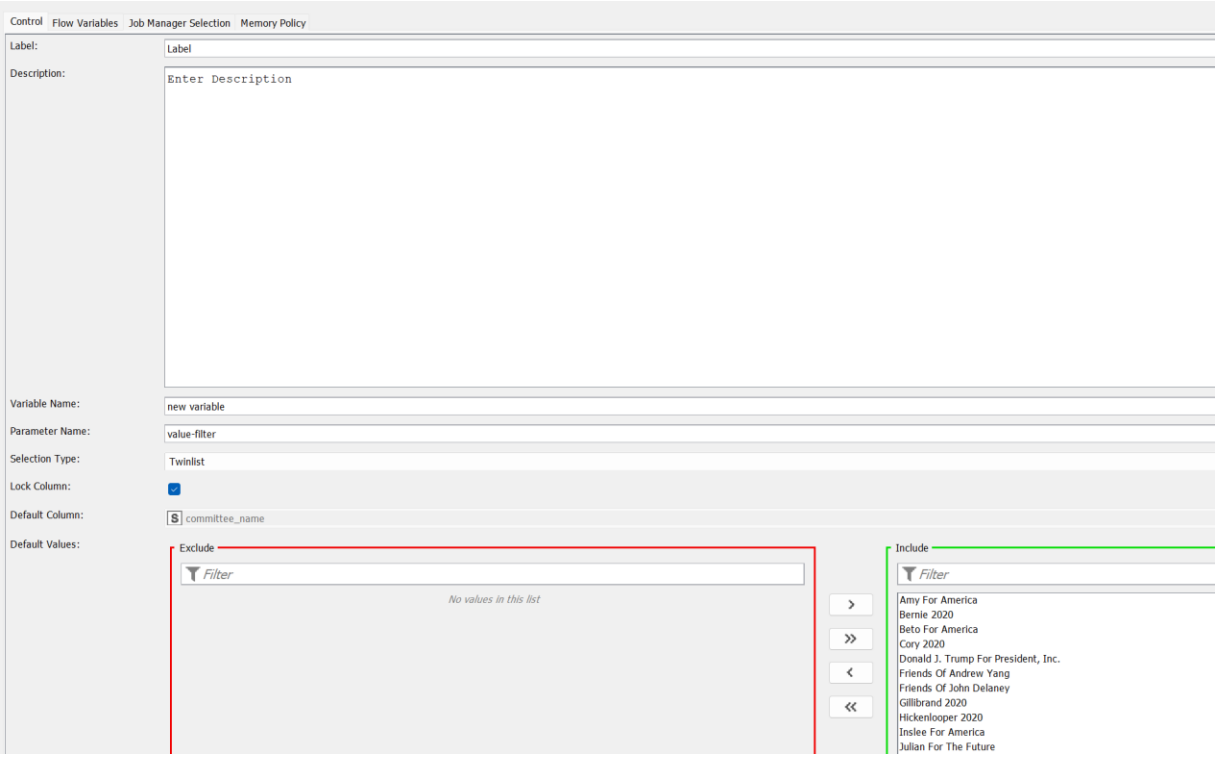
Our goal was to evaluate election contributions by state, political party, occupation, employer, month of the year, or day of year.

The first step was to create an interactive Knime workflow that was to aggregate the data existing in the data warehouse. For this I used the QuickForms “Value filter” that allowed the selection of desired values from my database. For example, the flow with the three nodes below, established a connection to the PostgreSQL server, read the data from the committee_name table, and then allowed me to interactively select political party values.



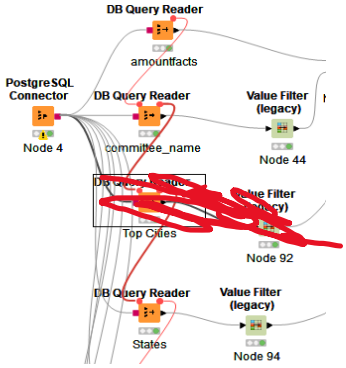
The first node returns the id and committee name of the political party in the database table.

The second node allows me to define the column that was used for selection. As one could see in the screenshot below, I chose the “priority” column and have locked it. I also chose to use all the values by default, see the “Include” panel.



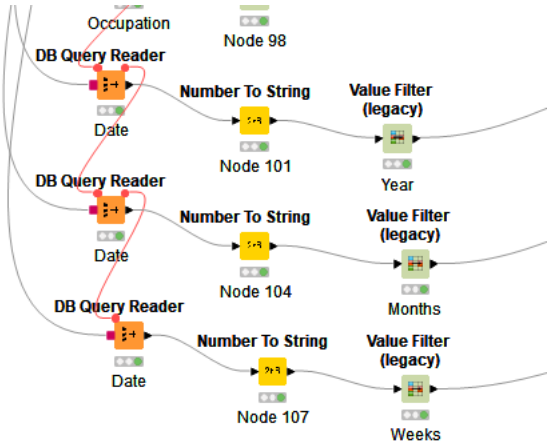
The selection of the priority values was done by right clicking on the “Value Filter” node and then clicking “Execute and Open View”. Note that if the node was green one would need to first reset the node to have the “Execute and Open View” available. The new window will give you the option to select the desired values. The new selected values would be applied when one clicked the button “Apply” and chose the “temporarily” option.

I repeated this process for two dimensions state and amountfacts tables, as shown in the figure below. The first node was to read from the facts table and that was not used yet. The following code was used to retrieve data from the database:



state Table: SELECT * FROM "state"
amountfacts Table: SELECT * FROM "amountfacts"

The data in the “Date” table needs to be processed some more due to the fact that the “Value Filter” node accepts only string columns/values while all the values in this table are numeric. To accomplish this we convert the Day of Month, day of week, and year to string. And then we feed them into the “value Filter” nodes.



For the other three dimensions city, employer, and occupation I used a slightly different retrieval method to get the proper data from these dimensions in the data warehouse. For example, for the “city” dimension I really only wanted to select city values that appeared in the table the most often. This is why I used a count function to count the number of occurrences of each city value in the city table. Then I only displayed values that occurred three times or more in the table. Which was basically outputting cities that contributed the most frequently during the 2020 elections in the first quarter.

```
Select Upper(c.city) as city, count, id
from
(select count(Upper(city)) as count, Upper(city) as city
from warehouse."city"
group by Upper(city)
having count(Upper(city))>=3
order by count(Upper(city)) desc) as countCity join warehouse."city" as c on
Upper(countCity.city)=Upper(c.city)
order by count desc;
```

I used this same concept for extracting data from the “occupation” table in the data warehouse, as well. I used the count function to count the number of occurrences of each job title listed in the occupation table. Then I only displayed values that occurred seven times or more in the table. Which was basically outputting job titles that contributed the most frequently during the 2020 elections in the first quarter.

SQL Statement

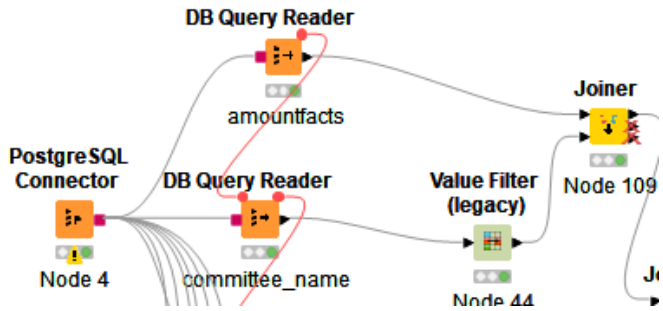
```
1  Select Upper(o.employer_title) as job_title, count, id
2  from
3  (select count(Upper(employer_title)) as count, Upper(employer_title) as Job_Title
4  from warehouse."occupation"
5  group by Upper(employer_title)
6  having count(Upper(employer_title))>=7
7  order by count(Upper(employer_title)) desc) as countOcc join warehouse."occupation" as o on
8  Upper(countOcc.Job_title)=Upper(o.employer_title)
9  order by count desc;
```

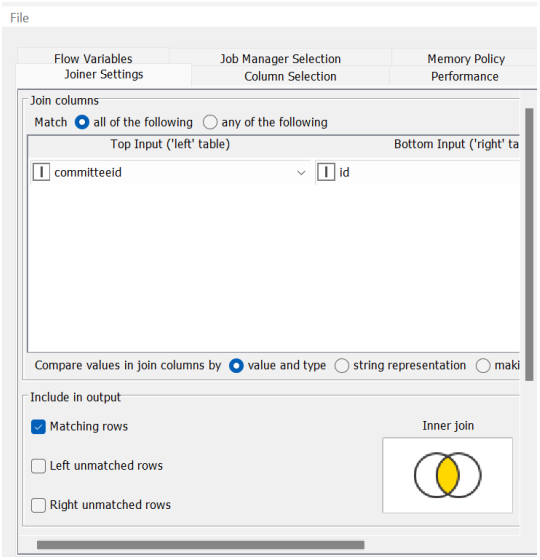
Lastly for the employer table I did not need to select certain values from this table. Since I wanted to get a full analysis of which employers contributed the most in the 2020 elections. This did not mean the employers who contributed the most frequently.

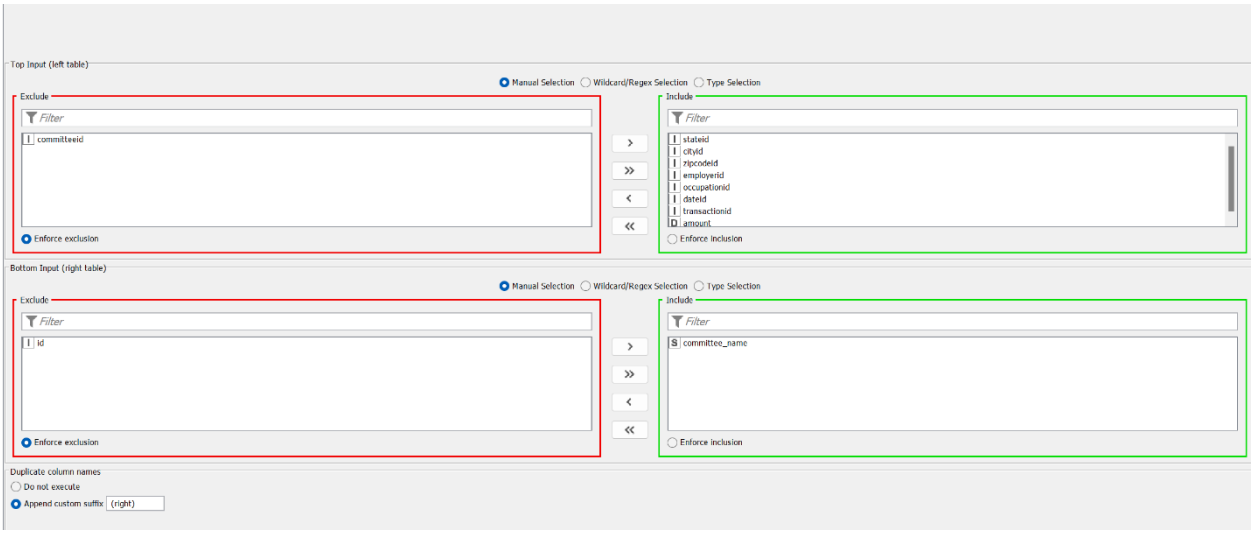
employer Table: SELECT * FROM "employer"

Next, we will join the values in each dimension table with the fact table. For example, the flow below, joins the data in the facts table with the selected political parties. Joining is done on the pair “committeeid” from the facts table and the “id” field from the

committee_name dimension table. I also decided to discard the id fields related to committee_name from the data set.







Here I joined the fact table with all the dimensions that I had in a similar fashion. The result is shown in the figure below, together with a partial view of the resulting data.

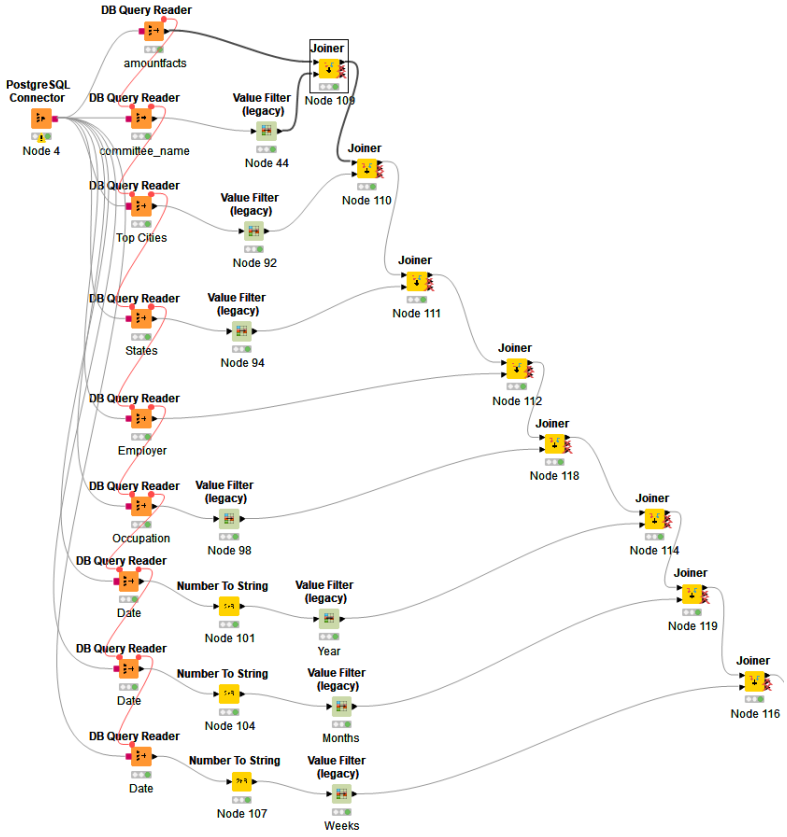
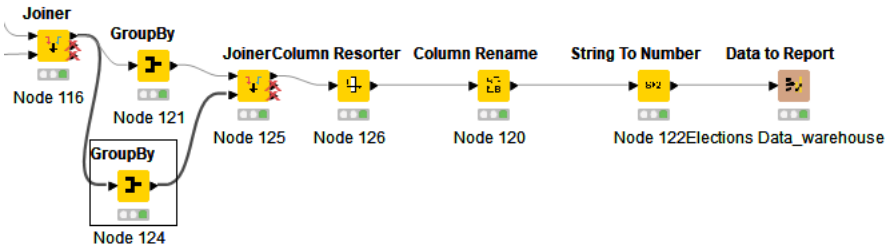
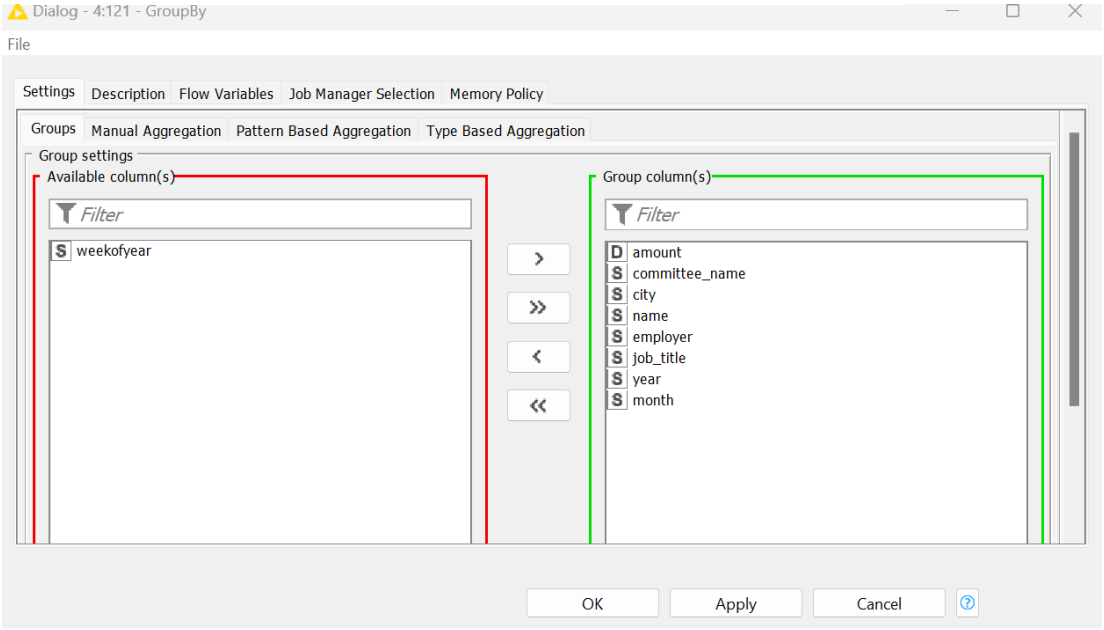


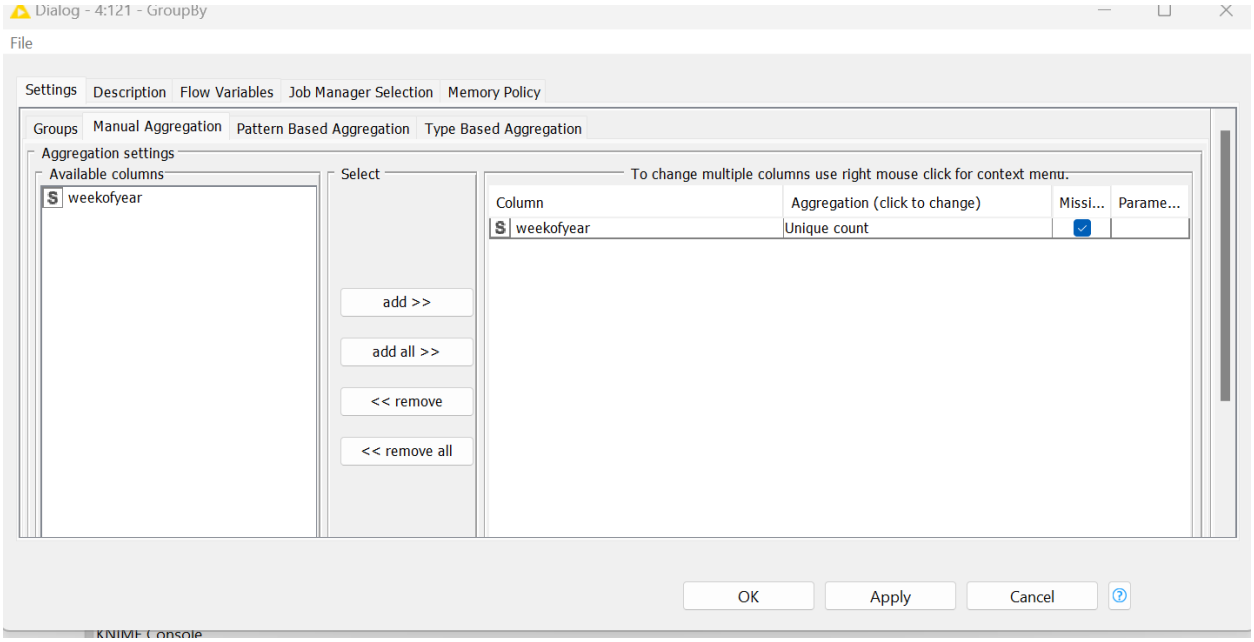
Table "default" - Rows: 536196 Spec - Columns: 9 Properties Flow Variables

Row ID	amount	commit...	city	name	employer	job_title	year	month	weekof...
Row290_Row...	50	Bernie 2020	GLENDALE	AZ	Not Employed	NOT EMPLOY...	2019	3	13
Row321_Row...	27	Bernie 2020	FLORENCE	AZ	Not Employed	NOT EMPLOY...	2019	3	13
Row431_Row...	100	Bernie 2020	FLORENCE	AZ	Not Employed	NOT EMPLOY...	2019	3	12
Row601_Row...	75	Donald J. Tr...	GLENDALE	AZ	RETIRED	RETIRED	2019	2	7
Row602_Row...	75	Donald J. Tr...	GLENDALE	AZ	RETIRED	RETIRED	2019	1	4
Row603_Row...	75	Donald J. Tr...	GLENDALE	AZ	RETIRED	RETIRED	2019	1	1
Row604_Row...	26.25	Donald J. Tr...	GLENDALE	AZ	RETIRED	RETIRED	2018	12	50
Row605_Row...	149.32	Donald J. Tr...	GLENDALE	AZ	RETIRED	RETIRED	2016	12	51
Row764_Row...	37.5	Donald J. Tr...	GLENDALE	AZ	UPS	DRIVER	2019	2	5
Row910_Row...	37.5	Donald J. Tr...	GLENDALE	AZ	HOSPICE OF...	REGISTERED ...	2019	3	10
Row911_Row...	37.5	Donald J. Tr...	GLENDALE	AZ	HOSPICE OF...	REGISTERED ...	2019	2	6
Row912_Row...	37.5	Donald J. Tr...	GLENDALE	AZ	HOSPICE OF...	REGISTERED ...	2019	1	1
Row913_Row...	37.5	Donald J. Tr...	GLENDALE	AZ	HOSPICE OF...	REGISTERED ...	2018	12	49
Row914_Row...	37.5	Donald J. Tr...	GLENDALE	AZ	HOSPICE OF...	REGISTERED ...	2018	11	45
Row915_Row...	26.25	Donald J. Tr...	GLENDALE	AZ	HOSPICE OF...	REGISTERED ...	2018	10	40

Creating a Counter Field







Once all the joins were completed, I decided to create a counter field that would be used to have a better understanding of what the amount column truly meant in terms of election donations. Since I thought it was possible for my audience to get confused about what the variable “amount” meant in my analysis.

So, I decided to add a count field that would be based on the number of election donations that happen during a certain week of the year for a particular election party based on the grouping variables listed above.

Consequently, by adding a count to my fact table I felt as if I provided valuable insights into the volume or frequency of election donations that happen throughout the year of 2019 based on each political party. This way users would understand patterns, trends, and the overall impact of election donation activities.

Displayed below is a screenshot of the counter variable data combined with my original fact table variable data.

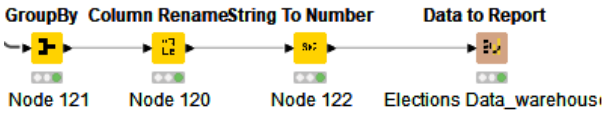
Renamed/Retyped table - 4:120 - Column Rename

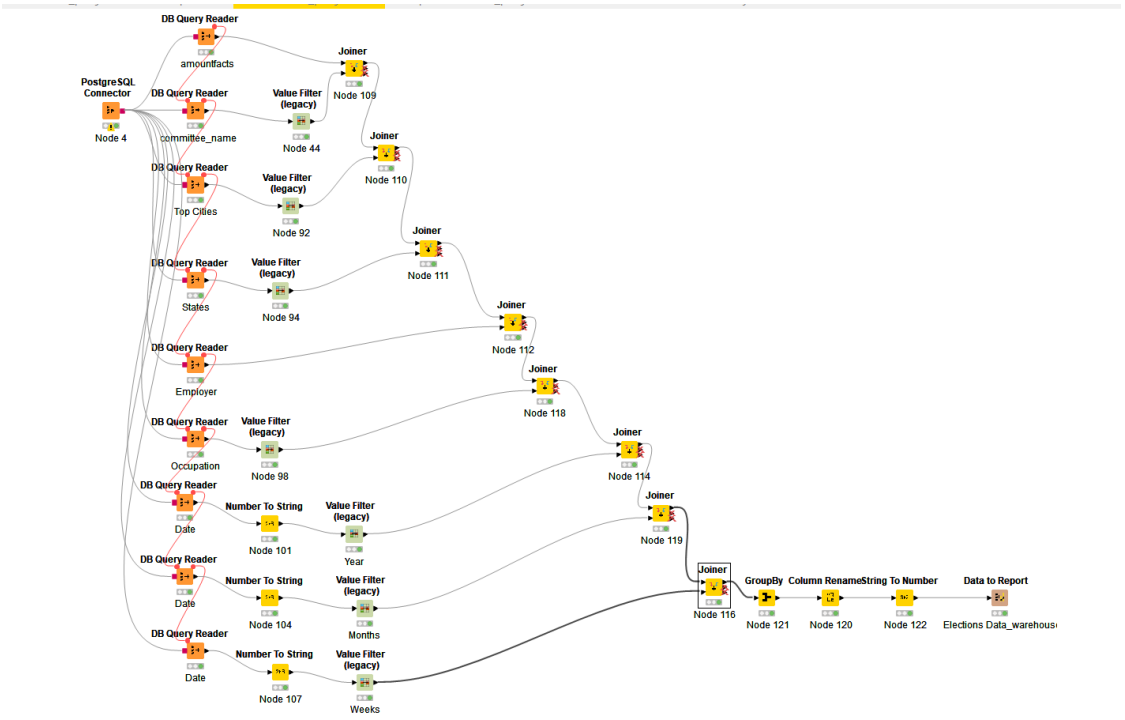
File Edit Hilite Navigation View

Table "default" - Rows: 3643 Spec - Columns: 10 Properties Flow Variables

Row ID	amount	commit...	city	state	employer	job_title	year	month	weekofyear	Number_electDonation_weekly_...
Row0_Row0	-2,600	Cory 2020	BEVERLY HIL...	CA	Not Employed	HOMEMAKER	2019	2	6	1
Row1_Row1	-1,000	Cory 2020	JACKSON	NJ	Gibbons PC	ADMINISTR...	2019	2	8	1
Row2_Row2	-1,000	Cory 2020	MONTCLAIR	NJ	Gibbons PC	ATTORNEY	2019	2	8	1
Row3_Row3	-900	Cory 2020	PRINCETON	NJ	IFM Resourc...	OWNER	2019	2	8	1
Row4_Row4	-372.28	Cory 2020	NEWARK	NJ	Mercury Pub...	MANAGING ...	2019	3	14	1
Row5_Row5	-250	Donald J. Tr...	VERO BEACH	FL	RETIRED	RETIRED	2018	2	9	1
Row6_Row6	-200	Donald J. Tr...	DELRAY BEA...	FL	THE GEO G...	PARALEGAL	2019	1	4	1
Row7_Row7	-150	Donald J. Tr...	TITUSVILLE	FL	RETIRED	RETIRED	2019	1	2	1
Row8_Row8	-150	Donald J. Tr...	TITUSVILLE	FL	RETIRED	RETIRED	2019	2	6	1
Row9_Row9	-150	Donald J. Tr...	TITUSVILLE	FL	RETIRED	RETIRED	2019	3	10	1
Row10_Row10	-100	Cory 2020	MONTCLAIR	NJ	Audible, Inc.	CEO	2019	3	12	1
Row11_Row11	-100	Cory 2020	MONTCLAIR	NJ	Not Employed	HOMEMAKER	2019	2	7	1
Row12_Row12	-100	Donald J. Tr...	DELRAY BEA...	FL	THE GEO G...	PARALEGAL	2019	1	3	1
Row13_Row13	-100	Donald J. Tr...	DELRAY BEA...	FL	THE GEO G...	PARALEGAL	2019	2	9	1
Row14_Row14	-100	Donald J. Tr...	HOLLYWOOD	FL	JASON SAN...	SURGEON	2019	3	11	1
Row15_Row15	-100	Donald J. Tr...	VERO BEACH	FL	TAX SAVING...	BUSINESS O...	2019	1	4	1
Row16_Row16	-100	Donald J. Tr...	VERO BEACH	FL	TAX SAVING...	BUSINESS O...	2019	2	8	1
Row17_Row17	-100	Donald J. Tr...	VERO BEACH	FL	TAX SAVING...	BUSINESS O...	2019	3	12	1
Row18_Row18	-75	Donald J. Tr...	HOLLYWOOD	FL	JASON SAN...	SURGEON	2019	1	3	1
Row19_Row19	-50	Donald J. Tr...	DELRAY BEA...	FL	THE GEO G...	PARALEGAL	2019	2	7	1
Row20_Row20	-31.5	Donald J. Tr...	VENICE	FL	RETIRED	RETIRED	2018	11	45	1
Row21_Row21	-25	Donald J. Tr...	TITUSVILLE	FL	RETIRED	RETIRED	2019	1	1	1
Row22_Row22	-25	Donald J. Tr...	TITUSVILLE	FL	RETIRED	RETIRED	2019	2	6	1
Row23_Row23	-25	Donald J. Tr...	TITUSVILLE	FL	RETIRED	RETIRED	2019	3	10	1
Row24_Row24	-18.75	Donald J. Tr...	VENICE	FL	RETIRED	RETIRED	2018	11	44	1
Row25_Row25	0.75	Donald J. Tr...	MIAMI	FL	RETIRED	RETIRED	2018	10	43	1
Row26_Row26	0.75	Donald J. Tr...	MIAMI	FL	RETIRED	RETIRED	2018	11	48	1
Row27_Row27	0.75	Donald J. Tr...	MIAMI	FL	RETIRED	RETIRED	2018	12	52	1
Row28_Row28	0.75	Donald J. Tr...	MIAMI	FL	RETIRED	RETIRED	2019	1	4	1
Row29_Row29	0.75	Donald J. Tr...	MIAMI	FL	RETIRED	RETIRED	2019	2	9	1
Row30_Row30	0.75	Donald J. Tr...	MIAMI	FL	RETIRED	RETIRED	2019	3	13	1
Row31_Row31	0.75	Donald J. Tr...	OAKLAND	CA	RETIRED	RETIRED	2019	2	6	1
Row32_Row32	1	Bernie 2020	LODI	CA	Not Employed	NOT EMPLO...	2019	3	14	1
Row33_Row33	1	Bernie 2020	OAKLAND	CA	Self	REALTOR	2019	3	14	1
Row34_Row34	1	Bernie 2020	PRINCETON	NJ	Princeton Un...	PROFESSOR	2019	3	14	1
Row35_Row35	1	Bernie 2020	SOMERSET	NJ	None	NOT EMPLO...	2019	3	14	1
Row36_Row36	1	Bernie 2020	STOCKTON	CA	ITT Techn	TEACHER	2019	3	14	1
Row37_Row37	2	Bernie 2020	PRINCETON	NJ	Princeton Un...	PROFESSOR	2019	3	14	1

The last three nodes aggregate the data by year, month of year, week of year, state, city, occupation, employer, and committee_name, and change the name of columns to more meaningful values.



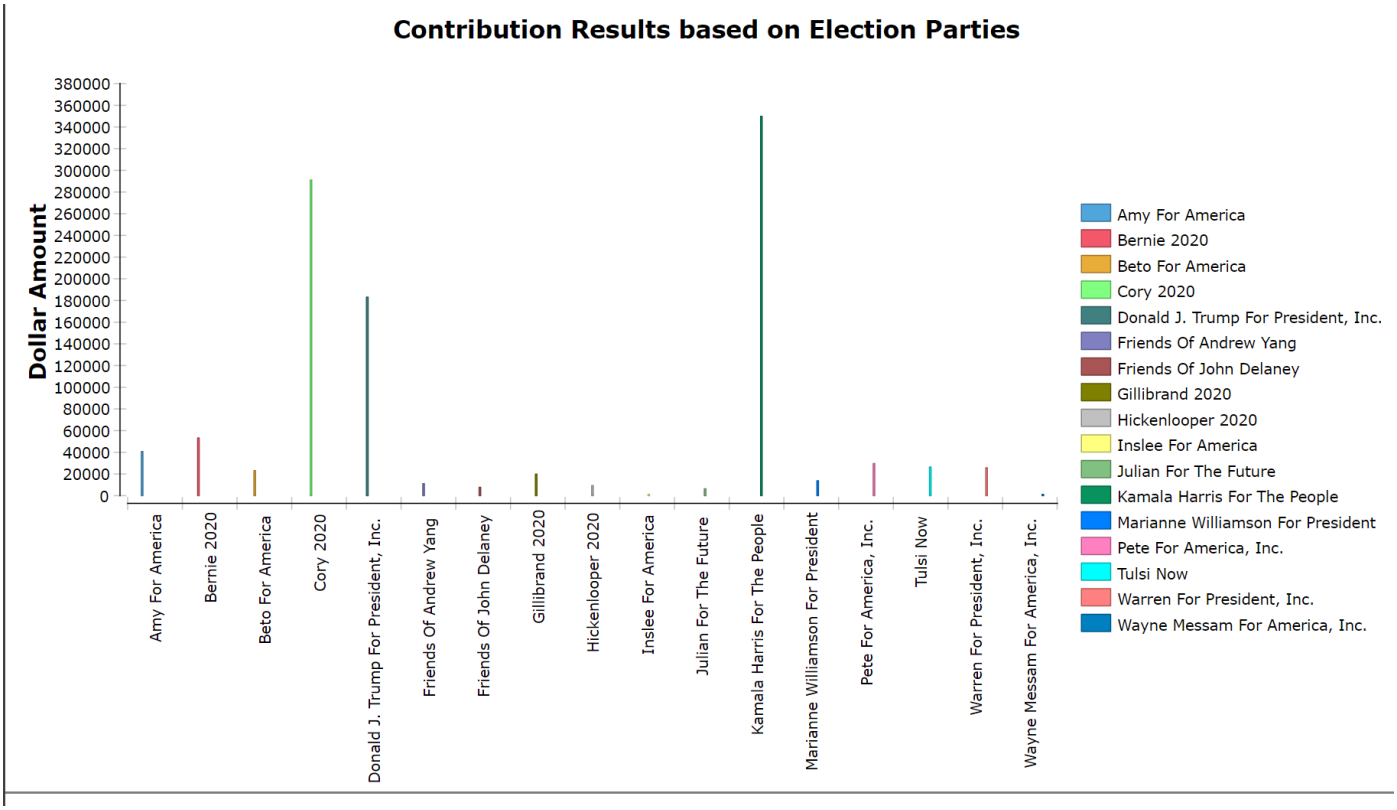


Creating a Report

Once I added the report node and connected it to the data, I customized it by clicking on the “Open Report” button on the menu.

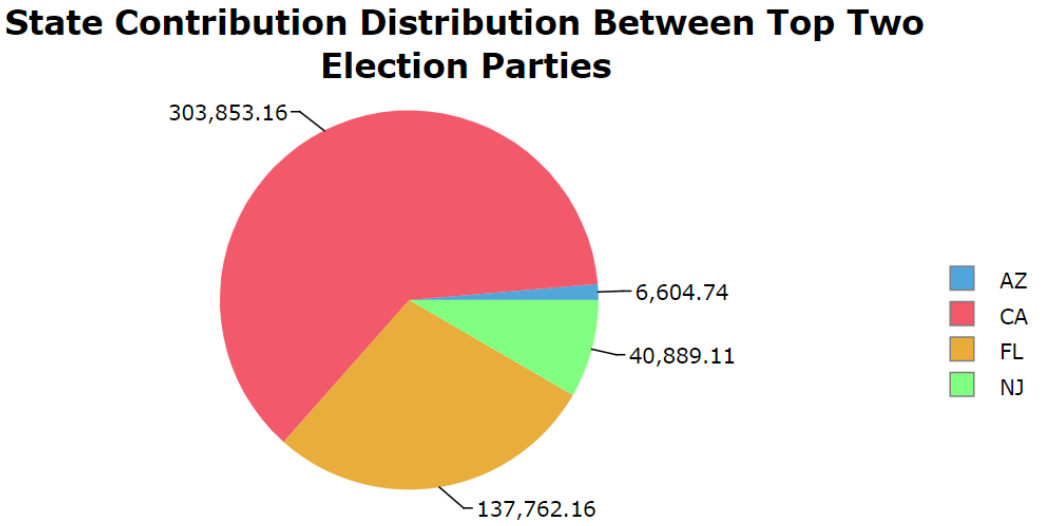
Once the data was established to conduct the report building process, I began to create the necessary charts and tables needed to answer each of the six business questions. The first chart I began to create was used to answer the first business question that I established. Which was *“Which campaigning party had the highest contributions in terms of dollars between the four states of California, New Jersey, Florida, and Arizona between the months of January and November 2019?”*

This chart, I felt needed to be a bar chart since we wanted to get a visual understanding of what the top campaigning/political parties were based on the four states. And bar charts should be used when you are showing segments of information. Vertical bar charts are useful to compare different categorical or discrete variables (*Statistics of Canada, 2023)*.



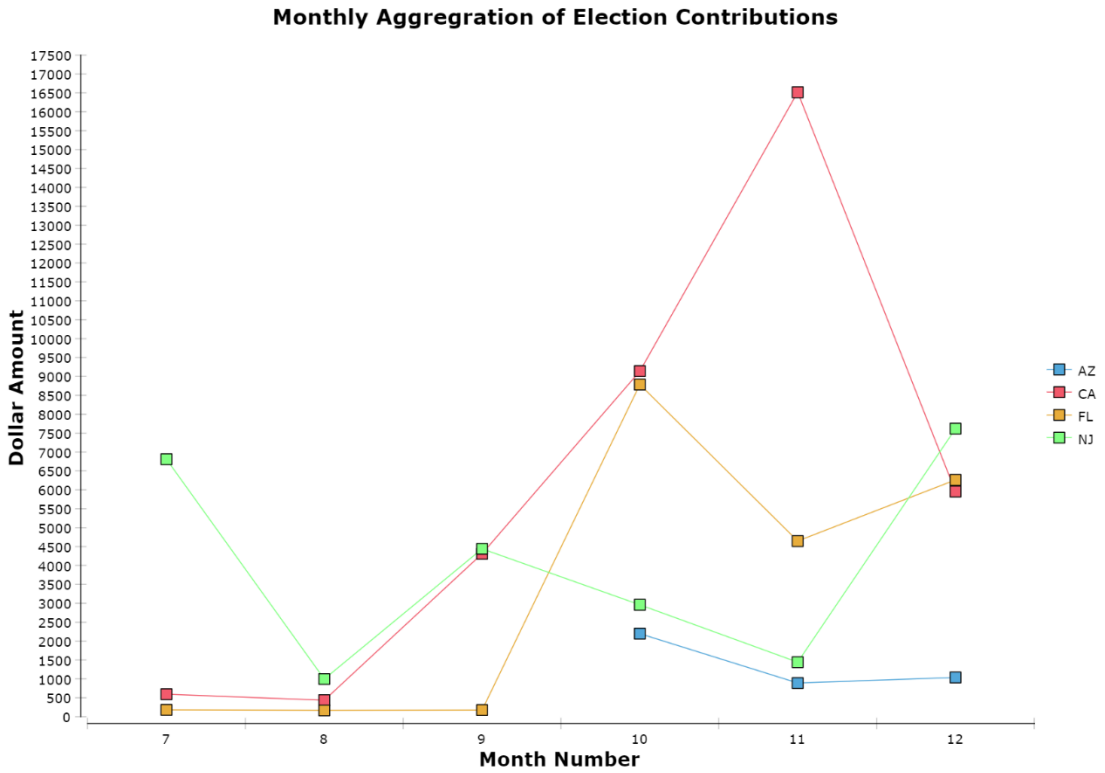
The second chart I began to create was used to answer the third business question that I established. Which was *"Which state had the highest contributions to the top two campaigning parties between the month of January and March 2019?"*

This chart, I felt, needed to be a pie chart since we wanted to get a visual understanding of which state contributed the most to the top political parties. And since the number of categorical variables that needed to be analyzed were less than five. I felt a pie chart could show this data point as a percentage of a whole. Instead of individual categorical aggregational values, which a bar chart would show.



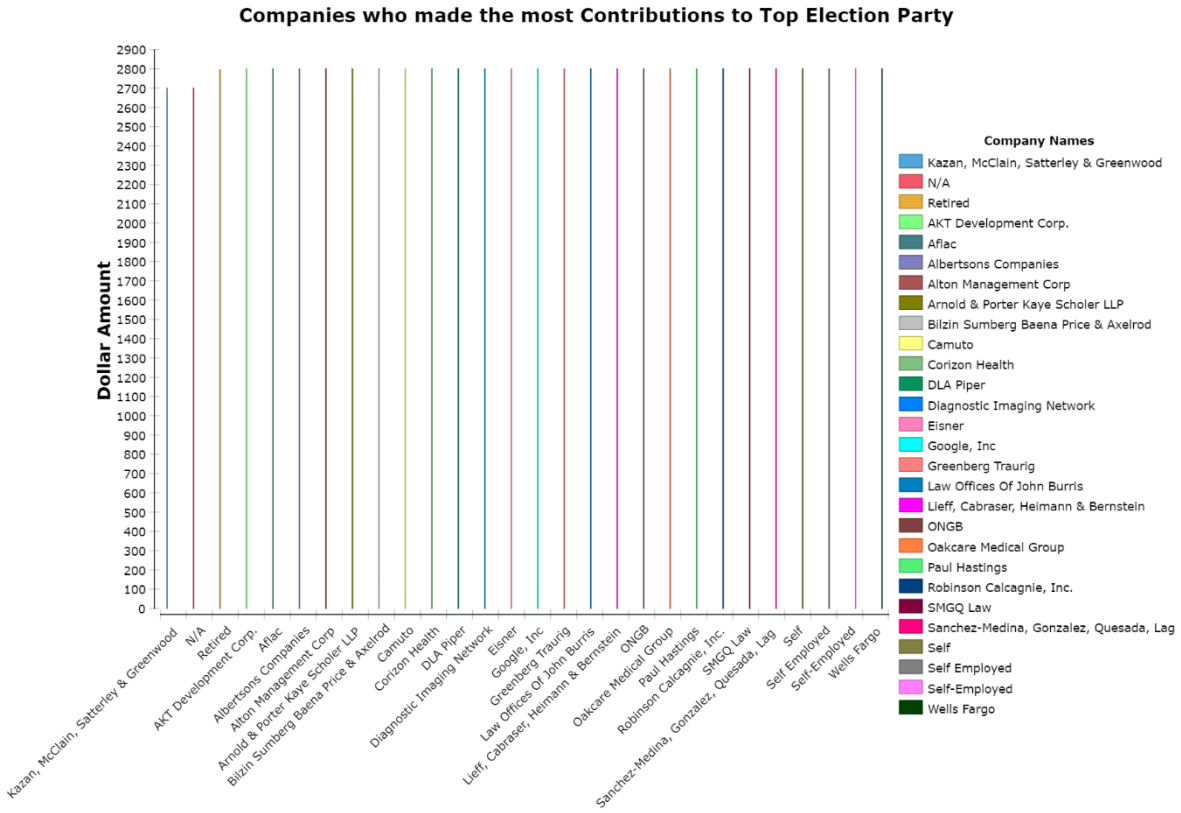
The third chart I began to create was used to answer the fourth business question that I established. Which was "*Display contributions made in California, New Jersey, Florida, and Arizona aggregated monthly between July and December in the year of 2019.*"

This chart, I felt, needed to be a line graph since we wanted to get a visual understanding of how contributions by individuals changed over time between July and December of the elections in 2019. And line graphs are best used when showing how something changes over time (KidsZone, 2023).



The fourth chart I began to create was used to answer the fifth business question that I established. Which was *"Which company made the most contributions in terms of dollars to the top campaigning party in the month of March 2019?"*

To answer this question, I felt I could choose from numerous different chart types. Though from further analysis I felt a bar chart would best depict the visual needed to answer this question. Since there was a great number of companies used to create the dataset for this report. And I felt a pie chart would not show the percentage of contributions from a given company compared to other companies in the dataset best. Because there would be too many companies to analyze and compare against with the use of slices in a pie chart. Also, a line graph would not suffice because we were not analyzing a change in values over a particular time period. I was only analyzing data during a particular month in a year.



The fifth and final chart I began to create was used to answer the sixth business question that I established. Which was *"What was the overall demographic makeup of the contributors of each of the top campaigning parties in the month of November 2019?"*

For this question I had to think twice about what I actually wanted to show to the audience of the report. And I came to the conclusion that not a graph or plot would best describe the demographic makeup of the contributors of the top campaigning parties. There would be too much information needed to be displayed to describe a given supporter of a given political party. So, I decided to use a table to show all the different types of political party supporters for each of the top campaigning parties. The information I decided to show included the different job professions each political party had supporting their campaign in the month of November in the 2020 elections.

Political Party & Month	Occupation	Dollar Amount
Donald J. Trump For President, Inc.		
10		
	RETIRED	0.75
	RETIRED	2.25
	BANKER	2.25
	RETIRED	2.25
	RETIRED	2.25
	RETIRED	2.25
	RETIRED	2.25
	RETIRED	2.25
	RETIRED	3
	RETIRED	3.75
	INFORMATION REQUESTED	3.75
	RETIRED	3.75
	RETIRED	3.75
	RETIRED	3.75
	RETIRED	3.75
	RETIRED	3.75
	RETIRED	3.75
	RETIRED	3.75
	RETIRED	4
	RETIRED	7.5
	RETIRED	7.5
	RETIRED	7.5
	RETIRED	7.5
	RETIRED	7.5
	RETIRED	11.25
	RETIRED	11.25
	RETIRED	11.25
	RETIRED	11.25
	NOT EMPLOYED	11.25
	RETIRED	15
	RETIRED	15
	RETIRED	15
	RETIRED	15
	RETIRED	15
	RETIRED	15
	RETIRED	15.15
	RETIRED	18.75
	RETIRED	18.75
	PHYSICIAN	18.75
	PHYSICIAN	18.75
10		
	RETIRED	18.75
	TRUCK DRIVER	18.75
	RETIRED	18.75
	RETIRED	18.75
	RETIRED	18.75
	BUSINESS OWNER	18.75
	RETIRED	18.75
	RETIRED	18.75
	RETIRED	18.75
	RETIRED	18.75
	RETIRED	18.75
	RETIRED	18.75
	RETIRED	22.5
	STUDENT	23.1
	REAL ESTATE	26.25
	REAL ESTATE	26.25
	RETIRED	26.25
	RETIRED	26.25
	RETIRED	26.25
	RETIRED	26.25
	REGISTERED NURSE	26.25
	RETIRED	26.25
	RETIRED	26.25
	RETIRED	26.25
	RETIRED	26.25
	RETIRED	26.25
	CONSTRUCTION	26.25
	RETIRED	26.25
	BUSINESS OWNER	26.25
	PHYSICIAN	26.25
	PHYSICIAN	26.25
	RETIRED	26.25
	RETIRED	26.25
	TRUCK DRIVER	26.25
	RETIRED	26.25
	RETIRED	26.25
	RETIRED	26.25
	REALTOR	26.25
	RETIRED	26.25
	REALTOR	26.25

REALTOR	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
RETIRED	26.25
ENTREPRENEUR	27
STUDENT	28.72
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
SALES	31.5
RETIRED	31.5
RETIRED	31.5
CEO	31.5
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
TECHNICIAN	31.5
RETIRED	31.5
RETIRED	31.5
RETIRED	31.5
REAL ESTATE BROKER	33.75
RETIRED	33.75
RETIRED	33.75
RETIRED	33.75
RETIRED	33.75
RETIRED	33.75
REAL ESTATE BROKER	37.5
REAL ESTATE BROKER	37.5
RETIRED	37.5
SELF-EMPLOYED	37.5
MANAGER	37.5
RETIRED	37.5
RETIRED	37.5
RETIRED	37.5
SELF-EMPLOYED	37.5
RETIRED	37.5
RETIRED	37.5
REGISTERED NURSE	37.5
RETIRED	37.5
RETIRED	37.5
SELF-EMPLOYED	37.5
RETIRED	37.5
RETIRED	37.5
RETIRED	37.5
RETIRED	37.5
RETIRED	37.5
SALES	37.5
CONSTRUCTION	37.5
RETIRED	37.5
LAWYER	37.5
RETIRED	37.5
RETIRED	37.5
RETIRED	37.5
VICE PRESIDENT	37.5
RETIRED	37.5
BUSINESS OWNER	37.5
RETIRED	37.5
RETIRED	37.5
RETIRED	37.5
RETIRED	37.5
ENTREPRENEUR	37.5
HOMEMAKER	37.5
CEO	37.5
RETIRED	37.5
RETIRED	37.5
ENTREPRENEUR	40.5
CEO	44.71
RETIRED	48.46
RETIRED	49.57
RETIRED	52.5
RETIRED	52.5
BUSINESS OWNER	56.25
RETIRED	56.25
PHYSICIAN	56.25
REGISTERED NURSE	56.25
ENGINEER	62.63
RETIRED	67.5
MANAGING DIRECTOR	67.5
ENTREPRENEUR	67.5
ENGINEER	67.5
RETIRED	75
RETIRED	75
DESIGNER	75
RETIRED	75
RETIRED	75
RETIRED	75
RETIRED	75
RETIRED	75
RETIRED	75
EXECUTIVE	75
HOMEMAKER	75
RETIRED	75
SALES	75
RETIRED	75
RETIRED	75
RETIRED	75
RETIRED	75
INFORMATION REQUESTED	75
REGISTERED NURSE	75
RETIRED	75
CONSTRUCTION	75
CONSTRUCTION	75
OWNER	75
REAL ESTATE	75
CEO	75
RETIRED	75
RETIRED	75
REAL ESTATE	75

Political Party & Month	Occupation	Dollar Amount
Donald J. Trump For President, Inc.		
10		
	CEO	75
	RETIRED	75
	RETIRED	75
	TRUCK DRIVER	75
	ATTORNEY	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	PHYSICIAN	75
	HOMEMAKER	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75
	RETIRED	75.67
	SELF-EMPLOYED	78.75
	RETIRED	78.75
	RETIRED	78.75
	CONSTRUCTION	80.87
	RETIRED	82.5
	STUDENT	83.01
	RETIRED	86.25
	REAL ESTATE	87.55
	INFORMATION REQUESTED	90
	STUDENT	90.67
	RETIRED	101.25
	ENTREPRENEUR	101.25
	RETIRED	112.5
	MANAGING DIRECTOR	112.5
	RETIRED	150
	HOMEMAKER	150
	RETIRED	150
	RETIRED	150
	RETIRED	150
10		
	ENGINEER	153.75
	RETIRED	172.5
	REAL ESTATE BROKER	187.5
	SELF-EMPLOYED	187.5
	RETIRED	187.5
	RETIRED	187.5
	RETIRED	187.5
	CEO	187.5
	RETIRED	225
	PHYSICIAN	225
	RETIRED	300
	SELF-EMPLOYED	300
	ENGINEER	316.12
	REAL ESTATE BROKER	375
	RETIRED	375
	SELF-EMPLOYED	375
	SELF-EMPLOYED	375
	SELF-EMPLOYED	375
	RETIRED	375
	RETIRED	375
	RETIRED	375
	SURGEON	375
	RETIRED	375
	PHYSICIAN	750
	RETIRED	750
Total Contribution Amount		**18506**

Political Party & Month	Occupation	Dollar Amount
Kamala Harris For The People		
10		
	RETIRED	2
	RETIRED	2.5
	RETIRED	3
	RETIRED	3.33
	RETIRED	3.33
	RETIRED	4
	RETIRED	4
	RETIRED	5
	RETIRED	5
	RETIRED	5
	ATTORNEY	5
	RETIRED	6
	RETIRED	7
	RETIRED	7.5
	RETIRED	8
	RETIRED	8
	RETIRED	10
	RETIRED	10
	RETIRED	10
	RETIRED	11
	RETIRED	12
	RETIRED	12
	RETIRED	12.5
	DESIGNER	12.5
	RETIRED	15
	RETIRED	15
	RETIRED	15
	ENGINEER	16.66
	RETIRED	25
	ATTORNEY	25
	SALES	33.33
	SOFTWARE ENGINEER	33.33
	RETIRED	50
	ATTORNEY	50
	RETIRED	50
	DESIGNER	50
	RETIRED	50
	RETIRED	100
	RETIRED	250
Total Contribution Amount		**946**

Political Party & Month	Occupation	Dollar Amount
Cory 2020		
10		
	NOT EMPLOYED	2.5
	NOT EMPLOYED	2.5
	NOT EMPLOYED	4
	NOT EMPLOYED	5
	NOT EMPLOYED	5
	NOT EMPLOYED	5
	NOT EMPLOYED	5
	ATTORNEY	5
	NOT EMPLOYED	5
	NOT EMPLOYED	5
	NOT EMPLOYED	5
	NOT EMPLOYED	10
	NOT EMPLOYED	10
	MANAGER	10
	ATTORNEY	10
	NOT EMPLOYED	10
	NOT EMPLOYED	10
	NOT EMPLOYED	12.5
	NOT EMPLOYED	15
	NOT EMPLOYED	15
	NOT EMPLOYED	19
	NOT EMPLOYED	19
	ATTORNEY	19
	NOT EMPLOYED	19
	NOT EMPLOYED	25
	PHYSICIAN	25
	BANKER	25
	PHYSICIAN	25
	NOT EMPLOYED	25
	VICE PRESIDENT	25
	NOT EMPLOYED	25
	NOT EMPLOYED	25
	NOT EMPLOYED	25
	NOT EMPLOYED	25
	NOT EMPLOYED	25
	DESIGNER	25
	NOT EMPLOYED	25
	NOT EMPLOYED	25
	NOT EMPLOYED	50
	PROFESSOR	50

Lastly, to answer the second business question of “Which campaigning party had the lowest contributions in terms of dollars amongst the four states of California, New Jersey, Florida, and Arizona between the months of January and November 2019?”, I decided that I did not need to create a sixth and final chart. Mainly because my first bar chart of campaign contributions showed this finding.

Business Questions Answered – Using PostgreSQL’s Reports

When analyzing and answering the business question stated earlier in this project assignment, I was able to discover some interesting facts. One point I was able to discover was the campaigning party who had the highest contributions in terms of dollars between the four states was Kamala Harris. Second was Cory Booker and third was Donald J. Trump. Kamala Harris had raised around $350,000, Cory Booker raised around $284,000, and Donald J. Trump raised around $171,000 from the four states. The lowest campaigning party amongst the four states was a tie between Jay Inslee and Wayne Messam. Both raised only $1,250 dollars amongst the four states from the months of January to November 2019.

Then the state who had the highest contribution value amongst the top two political candidates of Donald J. Trump and Kamala Harris was California in the months from January to April. With a total of $303,853.16 raised in the state. Florida came in second, New Jersey third, and Arizona last.

Next, in terms of monthly aggregations of contributions from July to December amongst each of the four states, I was able to see there was a spike of contributions from August to November. Contributions went from around $1,000 to $16,500 during this time period. However, after the November contributions, donation totals amongst the four states came crashing down to around $6,000 on average for the remainder of the year.

Then in terms of companies who contributed the most to Kamala Harris campaign in the month of March were people who were retired, or worked for companies like Google, Wells Fargo, Eisner, Aflac, Bilzin Sumberg Baena Price & Axelrod, Camuto, Corizon Health, DLA Piper, and Diagnostic Imaging Network. All these companies and/or people on average contributed around $2,800 to her campaign.

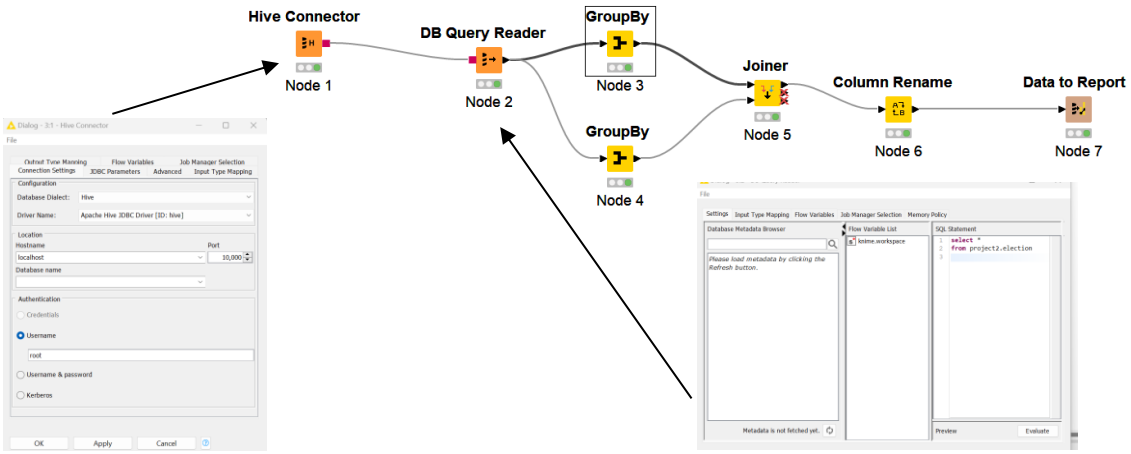
Lastly in terms of demographic makeup of the top campaigning candidates. Of which people who contributed the most were attorneys, accountants, artists, principals, real estate agents, executive directors, dentists, CEOs, business owners, bankers, people who worked in marketing, physicians, architects, surgeons, or students.

4.2 Hadoop Implementation

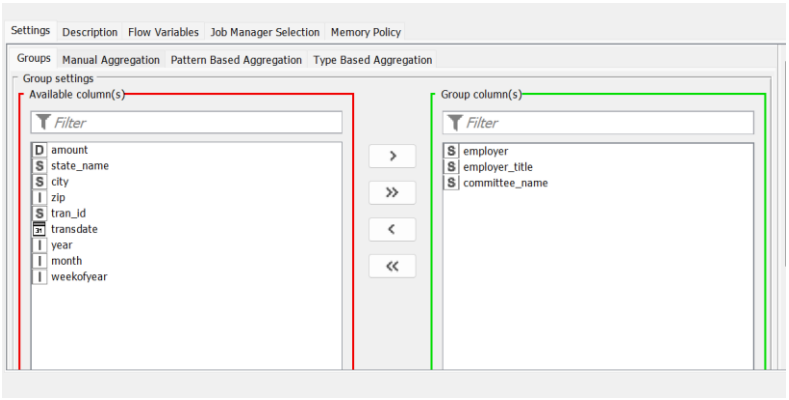
To be completed with project number 2

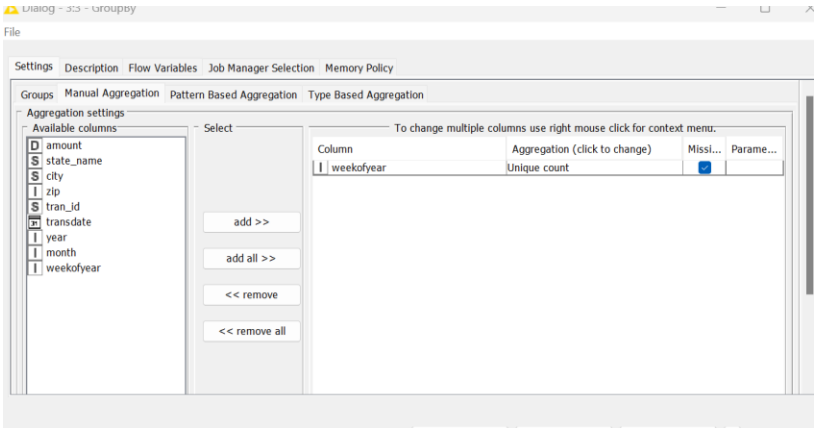
Once the data was loaded in a Hive table I connected to it from Knime using a Hive connector as shown in Figure 13.13.

- In this workflow, the IP address is the address that your container is running on and the port is “10000”.
- Once connected to the Hive database we can read the data in the “week13.salefacts” table using a Database reader node.



Once the data was read into KNIME's workflow, I then proceeded to create a counter field to aggregate the data based on weekly election donations made by each company to their respective political party. I used the following groupBy configurations to accomplish this task listed below:





Lastly, I visualized the data in the table by right-clicking on the Column Rename node and selecting "Renamed Table." Before I created my charts and graphs using the Data to Report node.

number...	amount	state_n...	city	zip	employer	employ...	tran_id	commit...	transdate	year	month	weekofyear
1	-2,800	CA	ATHERTON	94027	SELF-EMPLOY...	EDUCATION	1082798	CORY 2020	2019-03-20	2019	3	12
3	-2,800	CA	MOUNTAIN ...	94040	ALPHABET INC.	TECHNICAL ...	1071584	CORY 2020	2019-02-14	2019	2	7
2	-2,800	CA	SAN FRANCI...	94103	PINTEREST	DESIGNER	513071	PETE FOR A...	2019-03-18	2019	3	12
1	-2,800	CA	SAN FRANCI...	94121	WORKDAY	CEO	821931	HICKENLOO...	2019-03-31	2019	3	13
3	-2,800	NJ	MONMOUTH...	7750	PITTA & GIBLI...	ATTORNEY	1078123	CORY 2020	2019-03-26	2019	3	13
1	-2,800	NJ	RUMSON	7760	SELF-EMPLOY...	MUSICIAN	1077793	CORY 2020	2019-03-26	2019	3	13
1	-2,800	NJ	SKILLMAN	8558	PRINCETON V...	CO-FOUNDER	1071712	CORY 2020	2019-03-07	2019	3	10
1	-2,700	CA	BEVERLY HI...	90210	SELF	PYCHOLOGIST	SA17A.80570	DONALD J. ...	2019-01-23	2019	1	4
1	-2,700	CA	ENCINO	91436	SELF-EMPLOY...	FILM PROD...	1082807	CORY 2020	2019-03-15	2019	3	11
11	-2,700	CA	MILL VALLEY	94941	RPR PUBLIC R...	OWNER	1067696	CORY 2020	2019-02-26	2019	2	9
1	-2,700	CA	MILL VALLEY	94941	SPO PARTNERS	CHAIRMAN	1067694	CORY 2020	2019-02-26	2019	2	9
1	-2,700	FL	PALM BEACH	33480	DIVERSITYINC	CEO	1071719	CORY 2020	2019-02-19	2019	2	8
2	-2,700	FL	SAINT PETE...	33704	ASHLEY FURNI...	OWNER	SA18.242103	DONALD J. ...	2018-09-19	2018	9	38
1	-2,700	FL	SAINT PETE...	33704	ASHLEY FURNI...	OWNER	SA18.243243	DONALD J. ...	2018-09-19	2018	9	38
1	-2,700	FL	SAINT PETE...	33704	HOMEMAKER	HOMEMAKER	SA18.242101	DONALD J. ...	2018-09-19	2018	9	38
1	-2,700	FL	SAINT PETE...	33704	HOMEMAKER	HOMEMAKER	SA18.243242	DONALD J. ...	2018-09-19	2018	9	38
1	-2,700	FL	WELLBORN	32094	RETIRED	RETIRED	SA17A.21542	DONALD J. ...	2019-01-20	2019	1	3
2	-2,700	NJ	MORRISTO...	7960	NOT EMPLOYED	VOLUNTEER	1089641	CORY 2020	2019-03-29	2019	3	13
1	-2,600	CA	BEVERLY HI...	90210	NOT EMPLOYED	HOMEMAKER	1066413	CORY 2020	2019-02-07	2019	2	6
1	-2,600	CA	LOS ANGELES	90069	OPRAH WINFR...	MEDIA EXE...	1071717	CORY 2020	2019-02-07	2019	2	6

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

In project 1, please enter your reflective findings. Final version to be be completed with project number 2.

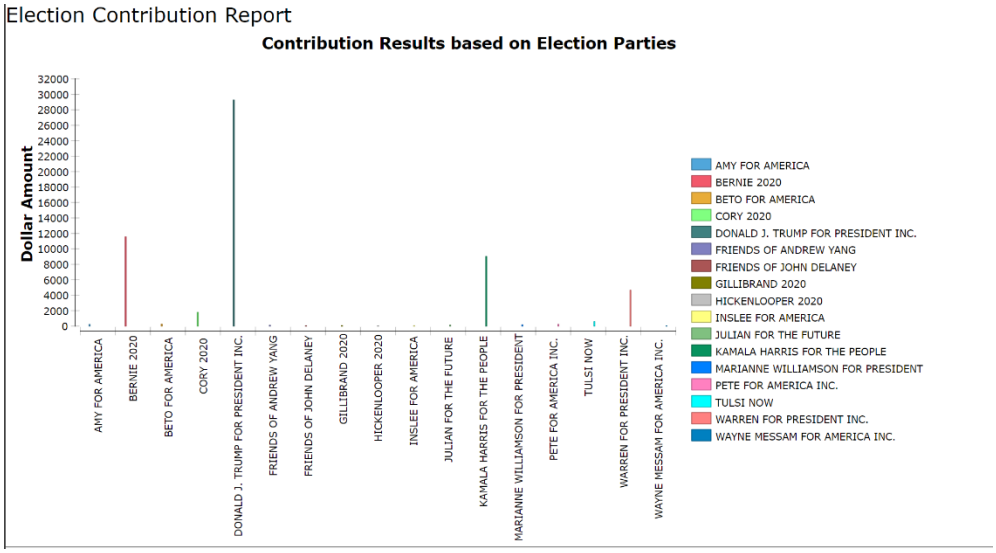
When comparing my Hadoop's final reports to my PostgreSQL reports, I did notice some differences with regards to top political parties. The counter variable which I created established some grouping differences in terms of weekly donations made by each employer of each political party. I believe that when I used the counter variable I created for my relational database, it did not completely aggregate the data of my weekly donations. Instead, it only aggregated data from my weekly donations and all other categorical variables listed in the fact table from the dimensional tables.

However, when I created the counter variable in KNIME in conjunction with my HIVE table I properly aggregated and grouped the data by weekly election donations made by each employer and employee to their respective political party. I believe this because the average weekly donations made by each person were not as high as listed in the reports generated by my PostgreSQL tables. Though the frequency or quantity of donations made by each person was higher, in my Hadoop's reports.

Another concept I noticed was that I did not need to use an abundance of QuickForms "Value filter" nodes in KNIME to select the desired values from my Hadoop data file system. Since, all the necessary values were already stored and accessible in my only Hive table. I found this feature of Hadoop and big data storage systems in general to be extremely useful and efficiently effective. By being able to use the concept of denormalization to store only relative data that pertains to predetermined business questions.

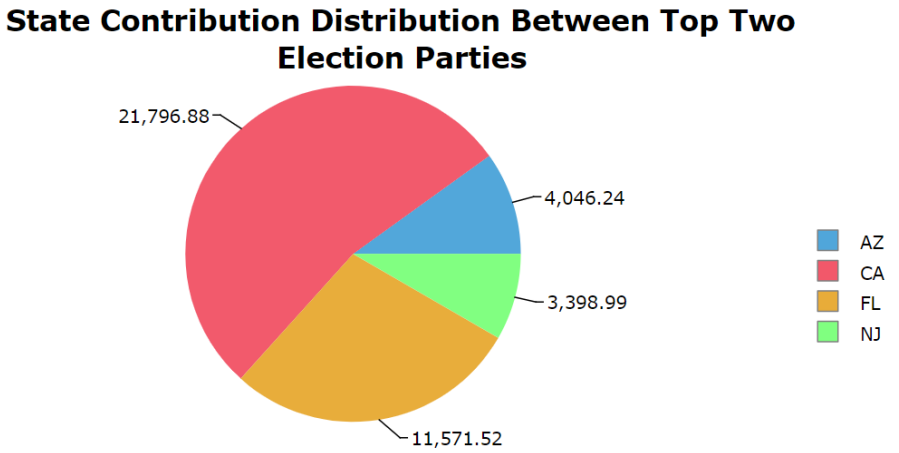
Listed below are the charts I created from my Hadoop Hive table in KNIME:

Question 1 - Which campaigning party had the highest contributions in terms of dollars between the four states of California, New Jersey, Florida, and Arizona between the month of January and November 2019?



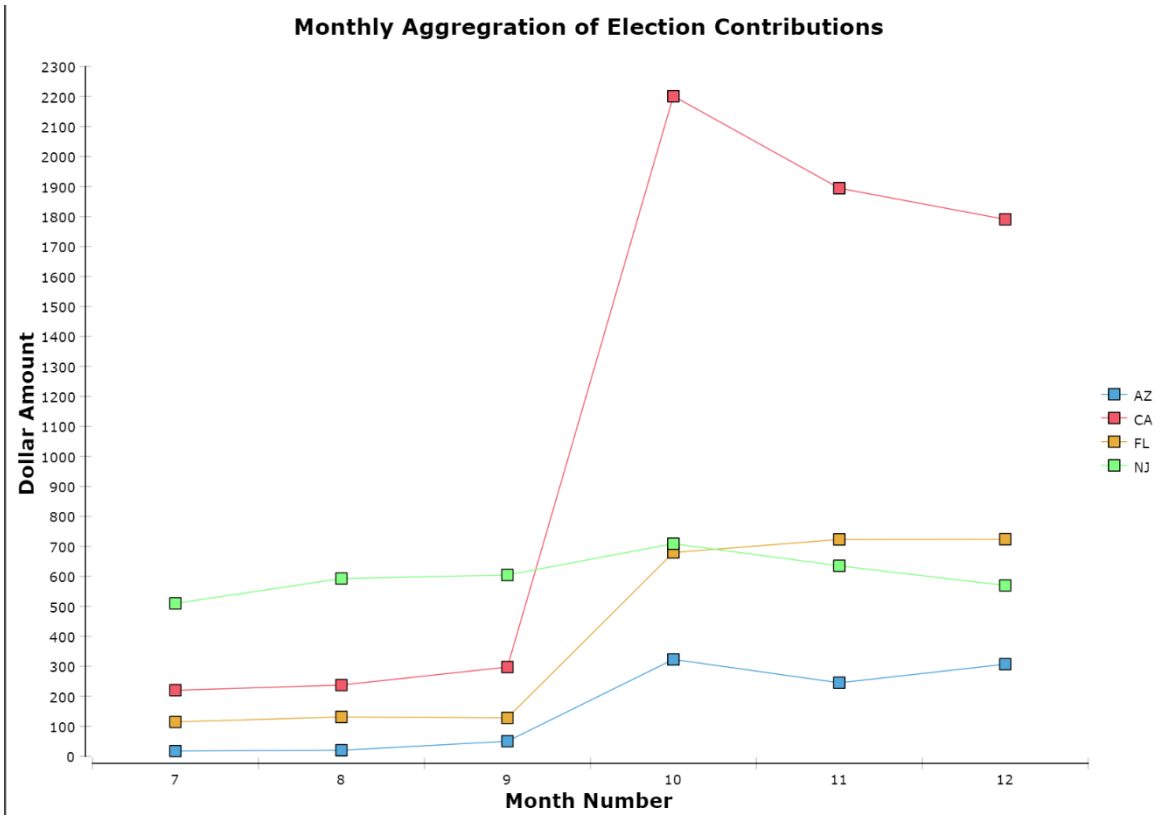
As shown above, my Hive table aggregated by weekly donations established Donald Trump to be the candidate with the highest total election donations. With Bernie Sanders and Kamala Harris coming in second and third respectively. However, my PostgreSQL tables produced a report that showed Donald Trump coming in third and Kamala Harris and Cory Booker coming in first and second respectively.

Question 3 - Which state had the highest contributions to the top two campaigning parties between the month of January and March 2019?



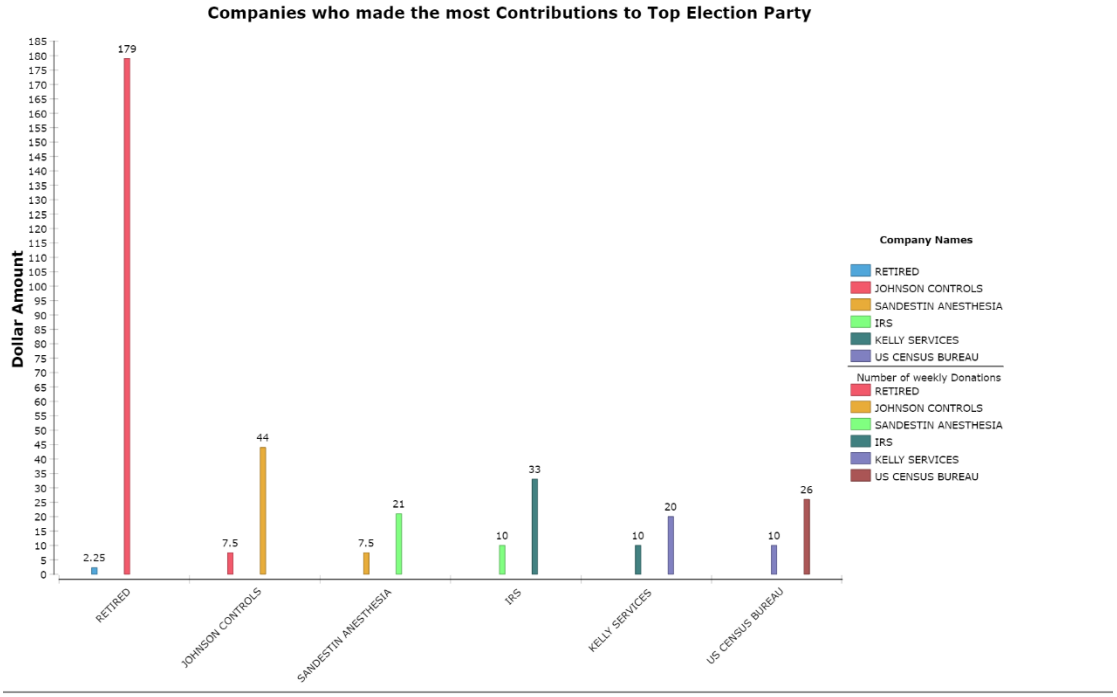
As shown above, my Hive table generated a pie chart that showed California to be the state that contributed the most to both Donald Trump and Bernie Sanders. Who both were established as the top two candidates receiving the most donations between the four states of Arizona, California, Florida, and New Jersey from my previous bar chart shown above. Again, this chart displayed slightly different values because my PostgreSQL pie chart established Kamala Harris and Donald Trump to be the top two candidates who received the most election donations between the four states of Arizona, California, Florida, and New Jersey. Though in both charts California was the state which contributed the most to the top two political candidates.

Question 4 - Contributions made in California, New Jersey, Florida, and Arizona aggregated monthly between July and December in the year 2019?



As shown in this line graph the total dollar amount of weekly donations was relatively low in the months between July and September. However, there was a big increase in weekly donations from September to October. Then the donations leveled off from October to December 2019. This trend was slightly different from my PostgreSQL graph because I was able to see an increase in election contributions from August to November. Which was a longer period of positive trend in donations between July and December 2019. However, the increase in donations calculated from my Hadoop Hive table showed a greater increase of donations in one month of $1900.

Question 5 - Which company made the most contributions in terms of dollars to the top campaigning party in the month of March 2019?



In this chart my Hive table really showed the difference between aggregations of my two different data storage systems. In my PostgreSQL table, the chart it generated showed high donation values that ranged from 0 to $2800. Which in general terms does not make much sense because normally one would not make high consecutive donations in a week. But for some reason my PostgreSQL table generated this display of multiple companies making $1,000 donations weekly to Kamala Harris campaign in the month of march. However, my Hive table showed that people made less than $1000 donations in the month of March to Donald Trump's campaign. However, the quantity/number of donations made weekly was higher and shown in my chart. For example, people who were retired contributed the most weekly donations to Donald Trump with a total number of 179 donations in the month of March. Johnson Controls came in second with the number of weekly donations totaling 44.

Question 6 - What was the overall demographic makeup of the contributors of each of the top campaigning parties in the month of November 2019?

In terms of demographic makeup of the top campaigning candidates. Donald Trump had the most diversity in terms of people who contributed to his campaign. Similar to the chart generated by my PostgreSQL tables. Then Kamala Harris came in second, in terms of diversity and Bernie Sanders came in last. Mainly because he did not have

anybody contributing to his campaign in the month of November 2019. Donald Trump had numerous people who were retired that contributed to his campaign. Other people who contributed to his campaign a lot weekly in the month of November were bus drivers, attorneys, account managers, R.Ns, and entrepreneurs. Another thing I noticed with his campaign was that people donated negative amounts of money. Which I assume to be because they gave him a different form of money instead of cash.

Meanwhile people who contributed to Kamala Harris campaign were people who were retired as well. And people who were investment managers, technologists, professors, architects, producers, counselors, campaign managers, physicians, and people who worked in hospitals.

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amou
DONALD J. TRUMP FOR PRESIDENT INC.			
11			
	RETIRED	5	-450
	PHYSICIAN	2	-187.5
	RETIRED	6	-187.5
	RETIRED	1	-187.5
	RETIRED	1	-179.74
	ENTREPRENEUR	7	-75.62
	RETIRED	1	-75
	PRESIDENT	1	-75
	RETIRED	6	-75
	RETIRED	2	-75
	RETIRED	3	-75
	BUSINESS OWNER	2	-75
	NEUROSURGEON	1	-75
	PHYSICIAN	1	-75
	TRUCKING	2	-75
	ENTREPRENEUR	1	-64.26
	RETIRED	2	-52.21
	RETIRED	2	-37.5
	DIRECTOR	5	-37.5
	RETIRED	1	-37.5
	RETIRED	5	-37.5
	RETIRED	1	-37.5
	FARMER	1	-37.5
	PHYSICIAN	1	-37.5
	RETIRED	1	-37.5
	RETIRED	1	-37.5
	NEUROSURGEON	1	-31.5
	RETIRED	2	-31.5
	RETIRED	1	-26.25
	RETIRED	1	-26.25
	RETIRED	1	-26.25
	RETIRED	1	-18.75
	RETIRED	4	-7.5
	CALL CENTER	1	0.75
	RETIRED	1	0.75
	RETIRED	2	0.75
	CASHIER/CLERK	2	0.75
	RETIRED	1	0.75
	INTERIOR DESIGNER	1	0.75
	NATIONAL DIRECTOR OF CLIENT SERVICES	1	0.75

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amount
DONALD J. TRUMP FOR PRESIDENT INC.			
11			
	RETIRED	2	2.25
	RETIRED	1	2.25
	RETIRED	6	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	EXEC ASST	7	2.25
	RETIRED	5	2.25
	RETIRED	2	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	NURSE PRACTITIONER	1	2.25
	RETIRED	6	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	ENGINEER	1	2.25
	RETIRED	1	2.25
	RETIRED	2	2.25
	RETIRED	1	2.25
	WASTEWATER TREATMENT PLANT OPERATOR	1	2.25
	ENGINEER	2	2.25
	RETIRED	5	2.25
	RETIRED	1	2.25
	EDUCATION SPECIALIST	1	2.25
	RETIRED	6	2.25
	HEALTH CARE	1	2.25
	RETIRED	6	2.25
	PHYSICIAN	7	2.25
	PHYSICIAN	9	2.25
	HOMEMAKER	8	2.25
	RETIRED	3	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	PRIVATE INVESTOR	1	2.25
	TAX PLANNER	1	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	RETIRED	2	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amount
DONALD J. TRUMP FOR PRESIDENT INC.			
11			
	RETIRED	1	0.75
	RETIRED	2	0.75
	RETIRED	1	0.75
	RETIRED	1	0.75
	RETIRED	4	0.75
	CONSULTANT	2	0.75
	RETIRED	1	0.75
	RETIRED	1	0.75
	ENTREPRENEUR	2	0.75
	ENTREPRENEUR	14	0.75
	RETIRED	1	0.75
	RETIRED	1	0.75
	RETIRED	2	0.75
	CONTRACTOR	1	1.47
	SCALEHOUSE ATTENDANT	1	1.5
	SCALEHOUSE ATTENDANT	1	1.5
	RETIRED	1	1.5
	WEALTH MGR	2	1.5
	MEDICAL TANSCRIBER	1	1.5
	LAWYER	1	1.5
	INSTRUCTOR	12	1.5
	SALES IT / SECURITY	2	1.87
	DRIVER	1	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	PROGRAMMER	2	2.25
	RETIRED	1	2.25
	MIXER DRIVER	1	2.25
	CLERK/GLUTEN FREE BAKER	1	2.25
	STAFF	2	2.25
	RETIRED	1	2.25
	RETIRED	3	2.25
	RETIRED	1	2.25
	RETIRED	2	2.25
	BUSINESS OWNER	1	2.25
	RETIRED	1	2.25
	RETIRED	3	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amount
DONALD J. TRUMP FOR PRESIDENT INC.			
11			
	SCIENTIST	1	2.25
	RETIRED	3	2.25
	RETIRED	3	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	PRIEST	1	2.25
	RETIRED	5	2.25
	ENTREPRENEUR	3	2.25
	RETIRED	1	2.25
	RETIRED	3	2.25
	SALES	1	2.25
	PROFESSIONAL EXPERT	1	2.25
	RETIRED	1	2.25
	HOMEMAKER	3	2.25
	TEACHER	1	2.25
	RETIRED	4	2.25
	RETIRED	1	2.25
	RETIRED	4	2.25
	RETIRED	1	2.25
	WATER TREATMENT OPERATOR	2	2.25
	RETIRED	4	2.25
	RETIRED	3	2.25
	RESIDENTIAL CARE	3	2.25
	DIRECTOR ENERGY MANAGEMENT SERVICES	1	2.25
	DESIGNER	2	2.25
	RETIRED	3	2.25
	RETIRED	1	2.25
	ANALYST	1	2.25
	RETIRED	2	2.25
	AGENT	1	2.25
	REAL EATATE	3	2.25
	SMALL BUSINESS OWNER	6	2.25
	SALES MANAGER	1	2.25
	SALES MANAGER	1	2.25
	RETIRED	1	2.25
	TECH	2	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25
	RETIRED	1	2.25

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amount
DONALD J. TRUMP FOR PRESIDENT INC.			
11			
	DISHWASHER	1	3.75
	DISHWASHER	6	3.75
	DISHWASHER	1	3.75
	SPORTS COACH	1	3.75
	ACCOUNT MANAGER	19	3.75
	RETIRED	7	3.75
	FASHION	1	3.75
	FISH AND WILDLIFE TECH	3	3.75
	RETIRED	1	3.75
	RETIRED	2	3.75
	MINISTER	1	3.75
	ACCOUNTING SPECIALIST	1	3.75
	ACCOUNTING SPECIALIST	1	3.75
	RETIRED	11	3.75
	RETIRED	1	3.75
	RETIRED	1	3.75
	RETIRED	1	3.75
	RETIRED	3	3.75
	RETIRED	2	3.75
	RETIRED	2	3.75
	RETIRED	1	3.75
	RETIRED	2	3.75
	RETIRED	1	3.75
	FIELD SERVICE ELECTRICIAN	2	3.75
	PROFESSIONAL EXPERT	11	3.75
	RETIRED	1	3.75
	RETIRED	2	3.75
	RETIRED	1	3.75
	RETIRED	1	3.75
	RETIRED	1	3.75
	RETIRED	1	3.75
	RETIRED	1	3.75
	AUTO MECHANIC	1	3.75
	RETIRED	1	3.75
	BUS OPERATOR	1	3.75
	R.N.	19	3.75
	RETIRED	1	3.75
	SMALL BUSINESS OWNER	1	3.75
	RETIRED	1	3.75
	AUTHOR	1	3.75

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amount
DONALD J. TRUMP FOR PRESIDENT INC.			
11			
	LIBRARIAN	1	7.5
	LIBRARIAN	1	7.5
	LIBRARIAN	1	7.5
	RETIRED	1	7.5
	RETIRED	2	7.5
	REALTOR	1	7.5
	RETIRED	4	7.5
	RETIRED	1	7.5
	RETIRED	1	7.5
	RETIRED	1	7.5
	RETIRED	5	7.5
	PERSONAL CARE ASSISTANT	1	7.5
	RETIRED	1	7.5
	PAINTER	1	7.5
	PAINTER	1	7.5
	PAINTER	1	7.5
	DRIVER	1	9
	RETIRED	1	9
	RETIRED	1	9
	SCHOOL BUS	7	9
	REALTOR ASSOCIATE	1	9
	RETIRED	1	9.75
Total Contribution Amount			**-728.3000000000009**

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar
KAMALA HARRIS FOR THE PEOPLE			
11			
	ZUMBA INSTRUCTOR	1	5
	ARTIST	1	5
	ARTIST	1	5
	ARTIST	2	5
	CAMPAIGN MANAGER	2	5
	CAMPAIGN MANAGER	7	5
	RESEARCH SYSTEMS ANALYST	4	5
	RETIRED	3	5
	ENGINEER	2	5
	RETIRED	4	5
	ATTORNEY	1	5
	ATTORNEY	1	5
	MUSICIAN	1	5
	RETIRED	4	5
	MUSIC/FILM PRODUCER	1	5
	RETIRED	1	5
	RETIRED	1	5
	RETIRED	1	6
	RETIRED	1	6
	RETIRED	1	6
	RETIRED	1	6
	ARCHITECT	1	6
	NOT-EMPLOYED	1	6
	COSTUME DESIGNER	1	6
	RETIRED	4	6
	RETIRED	2	6
	RETIRED	1	6
	RETIRED	3	6
	RETIRED	1	6
	RETIRED	1	6
	ATTORNEY	1	6
	ONLINE STORE OWNER/OPERATOR	1	6
	RETIRED	1	7
	NOT-EMPLOYED	2	7.5
	VICE PRESIDENT	3	7.5
	OWNER & WINEMAKER	1	7.5
	RETIRED	1	7.5
	REALTOR	5	7.5
	RETIRED	1	7.5
	RETIRED	1	7.5

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amount
KAMALA HARRIS FOR THE PEOPLE			
11			
	INVESTMENT MANAGEMENT	2	1
	LEAD HEALTH NAVIGATOR	3	1.66
	RETIRED	3	2.5
	CEO	2	2.5
	RETIRED	1	2.5
	RETIRED	3	2.5
	CRNA	1	2.5
	FILMMAKER	4	2.5
	MANAGEMENT CONSULTING	2	2.5
	RETIRED	1	2.5
	RETIRED	9	3
	RETIRED	1	3
	ARCHITECT	1	3
	RETIRED	1	3
	SCREENWRITER	1	3
	RETIRED	5	3
	RETIRED	2	3
	NOT-EMPLOYED	6	4
	ARCHITECT	1	4
	RETIRED	2	5
	RETIRED	2	5
	CONSULTANT	1	5
	RETIRED	1	5
	RETIRED	7	5
	RETIRED	1	5
	RETIRED	1	5
	TECH SUPPORT	1	5
	TECH SUPPORT	7	5
	MENTAL HEALTH THERAPIST	1	5
	MENTAL HEALTH THERAPIST	1	5
	RETIRED	1	5
	RETIRED	1	5
	NOT-EMPLOYED	2	5
	NOT-EMPLOYED	1	5
	NOT-EMPLOYED	1	5
	NOT-EMPLOYED	1	5
	RETIRED	1	5
	MARKETING MANAGER	1	5
	MUSICIAN	1	5
	PROFESSOR	3	5

Political Party & Month	Occupation	number_elecDonation_weekly	Dollar Amount
KAMALA HARRIS FOR THE PEOPLE			
11			
	PRODUCER/CONSULTANT	1	7.5
	RETIRED	2	7.5
	GRAPHICS TECH	5	7.5
	RETIRED	3	8
	RETIRED	9	8
	RETIRED	6	10
	RETIRED	2	10
	RETIRED	1	10
	RETIRED	13	10
	HOSPITALITY	1	10
	RETIRED	1	10
	RETIRED	2	10
	ART DIRECTOR	2	10
	WRITER	1	10
	PHYSICIAN	1	10
	RETIRED	1	10
	RETIRED	1	10
	RETIRED	1	10
	ANTHROPOLOGIST	1	10
	RETIRED	1	10
	FILMMAKER	5	10
	PHYSICIAN	8	10
Total Contribution Amount			**799.66**

Conclusions

Overall conclusions of the project. In project 2, add a reflective analysis of the advantages and disadvantages of the two implementations.

In conclusion, Hadoop and PostgreSQL serve distinct purposes in the data management landscape. Hadoop excels in handling massive volumes of diverse data through its scalable and distributed architecture, while PostgreSQL is a better choice for applications demanding transactional consistency, complex queries, and structured data. The choice between Hadoop and PostgreSQL depends on specific use cases, data requirements, and performance considerations. I truly believe organizations can effectively integrate both types of systems within their data ecosystems to leverage optimal results.

To reiterate the advantages of PostgreSQL are that it is ACID Compliance. Meaning PostgreSQL ensures data integrity with Atomicity, Consistency, Isolation, and Durability. PostgreSQL has full access to expressive query commands and user-defined functions. However, disadvantages are mainly related to scalability limitations. Since vertical scaling may have practical limits for handling extremely large datasets.

The advantages of Hadoop are that it is easily scalable in terms of its ability to handle large datasets. Another advantage of Hadoop relates to its flexibility at supporting diverse data types. Since Hadoop's filing system (HDFS) can parse through structured, semi-structured, and unstructured data. And Hadoop's parallelized computation allows for fault tolerance by replicating data across nodes. Which is a major plus in terms of data security and uninterrupted functionality. However, disadvantages of Hadoop fall under the category of latency and complexity. Latency meaning it less suitable for real-time or low-latency processing applications. And complexity meaning developing and debugging MapReduce jobs can be challenging for some users.

Works Cited

Government of Canada, Statistics Canada. "5 Data Visualization 5.2 Bar Chart." *5.2 Bar Chart*, 2 Sept. 2021, www150.statcan.gc.ca/n1/edu/power-pouvoir/ch9/bargraph-diagrammeabarres/5214818-eng.htm.

"Line Graph." *Kids Graphing Page - Line Graph - NCES Kids' Zone*, 14 Oct. 2023, nces.ed.gov/nceskids/graphing/classic/line.asp#:~:text=Line%20graphs%20can%20be%20used,for%20what%20is%20being%20measured.

DBeaver documentation (2023) *DBeaver*. Available at: https://dbeaver.com/docs/dbeaver/ (Accessed: 10 October 2023).

KNIME documentation (2023) *KNIME*. Available at: https://docs.knime.com/ (Accessed: 2 October 2023).

"The Collaborative Data Science Platform | Mode." *Mode.com*, 2 Oct. 2023, mode.com/. Accessed 2 Oct. 2023.

"Hive Foreign Keys?" *Stack Overflow*, 14 Mar. 2012, stackoverflow.com/questions/9696369/hive-foreign-keys.

GeeksforGeeks. "Hadoop Pros and Cons." *GeeksforGeeks*, 29 June 2020, www.geeksforgeeks.org/hadoop-pros-and-cons.

Team, DataFlair, et al. "13 Big Limitations of Hadoop &Amp; Solution to Hadoop Drawbacks." *DataFlair*, 7 Mar. 2019, data-flair.training/blogs/13-limitations-of-hadoop.

Vskills, Team. "Pig Latin Commands." *Tutorial*, 19 Feb. 2021, www.vskills.in/certification/tutorial/pig-latin-commands.

Pedamkar, Priya. "Pig Commands." *EDUCBA*, 9 June 2023, www.educba.com/pig-commands.

Pig Latin Basics. pig.apache.org/docs/latest/basic.html.

"Hadoop Vs PostgreSQL | TrustRadius." *TrustRadius*, www.trustradius.com/compare-products/apache-hadoop-vs-postgresql#:~:text=Hadoop%20is%20popular%20for%20its,functionality%20available%20across%20commoditized%20hardware.&text=PostgreSQL%20(alternately%20Postgres)%20is%20a,%2C%20feature%20robustness%2C%20and%20performance.

GeeksforGeeks. “Difference Between PostgreSQL and HBase.” *GeeksforGeeks*, 9 June 2022, www.geeksforgeeks.org/difference-between-postgresql-and-hbase.

Talend. “MapReduce 101: What It Is and How to Get Started.” *Talend - a Leader in Data Integration & Data Integrity*, www.talend.com/resources/what-is-mapreduce/#:~:text=MapReduce%20is%20a%20programming%20model,functioning%20of%20the%20Hadoop%20framework.

Great Learning. “Top 20 Essential Docker Commands You Should Know in 2024.” *Great Learning Blog: Free Resources What Matters to Shape Your Career!*, 8 Nov. 2023, www.mygreatlearning.com/blog/top-essential-docker-commands.

“Top 10 Hadoop HDFS Commands With Examples and Usage.” *DataFlair*, 25 Aug. 2021, data-flair.training/blogs/top-hadoop-hdfs-commands-tutorial.

“HDFS Commands.” *GeeksforGeeks*, 4 Apr. 2019, www.geeksforgeeks.org/hdfs-commands.

Singhal, Shashank. “HDFS Commands Cheat Sheet - Geek Culture - Medium.” Medium, 2 Apr. 2022, medium.com/geekculture/hdfs-commands-cheat-sheet-1cd7bf22e795.

“What Is Fault Tolerance? Definition and FAQs | Avi Networks.” *Avi Networks*, 25 May 2023, avinetworks.com/glossary/fault-tolerance/#:~:text=Cloud%20fault%20tolerance%20simply%20means,in%20the%20same%20data%20centers.