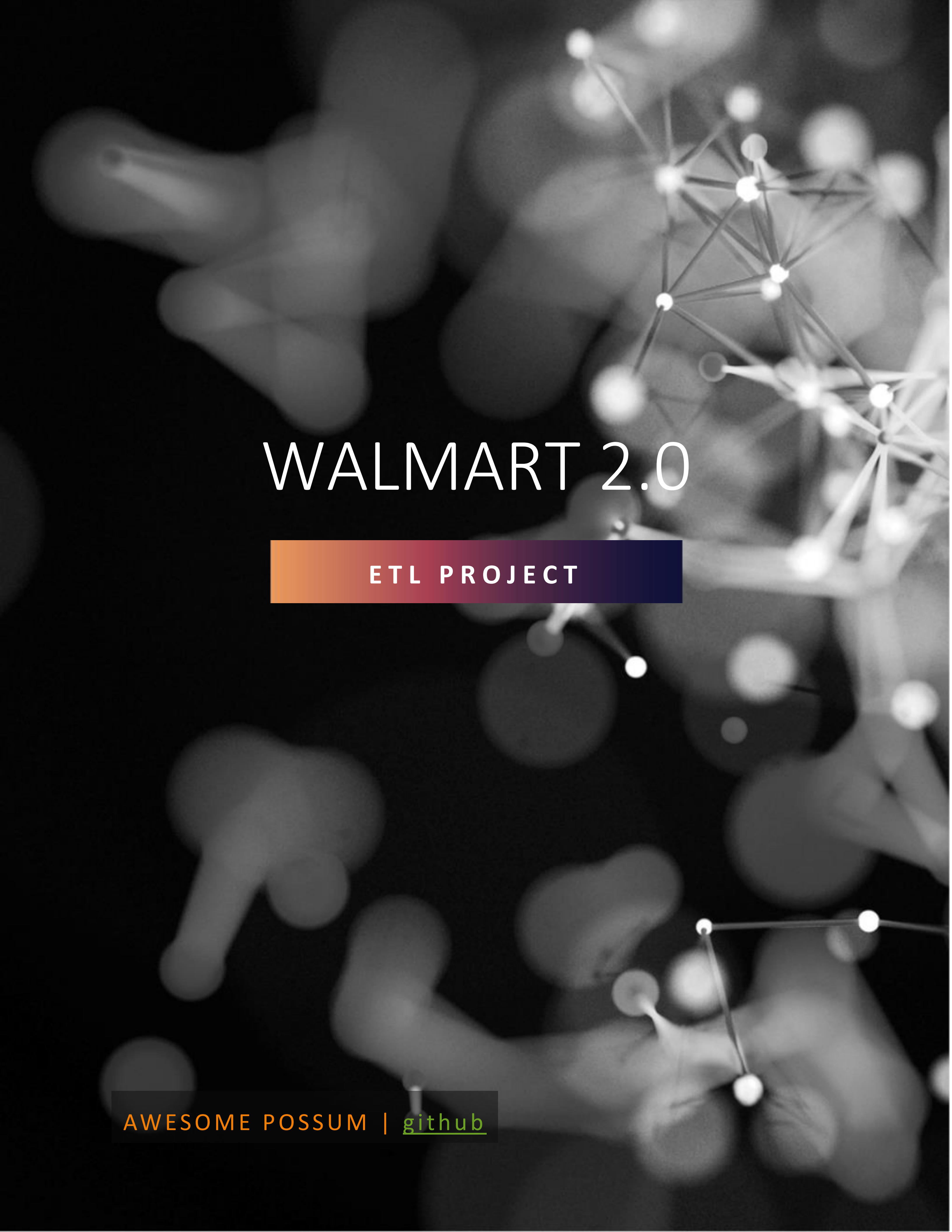
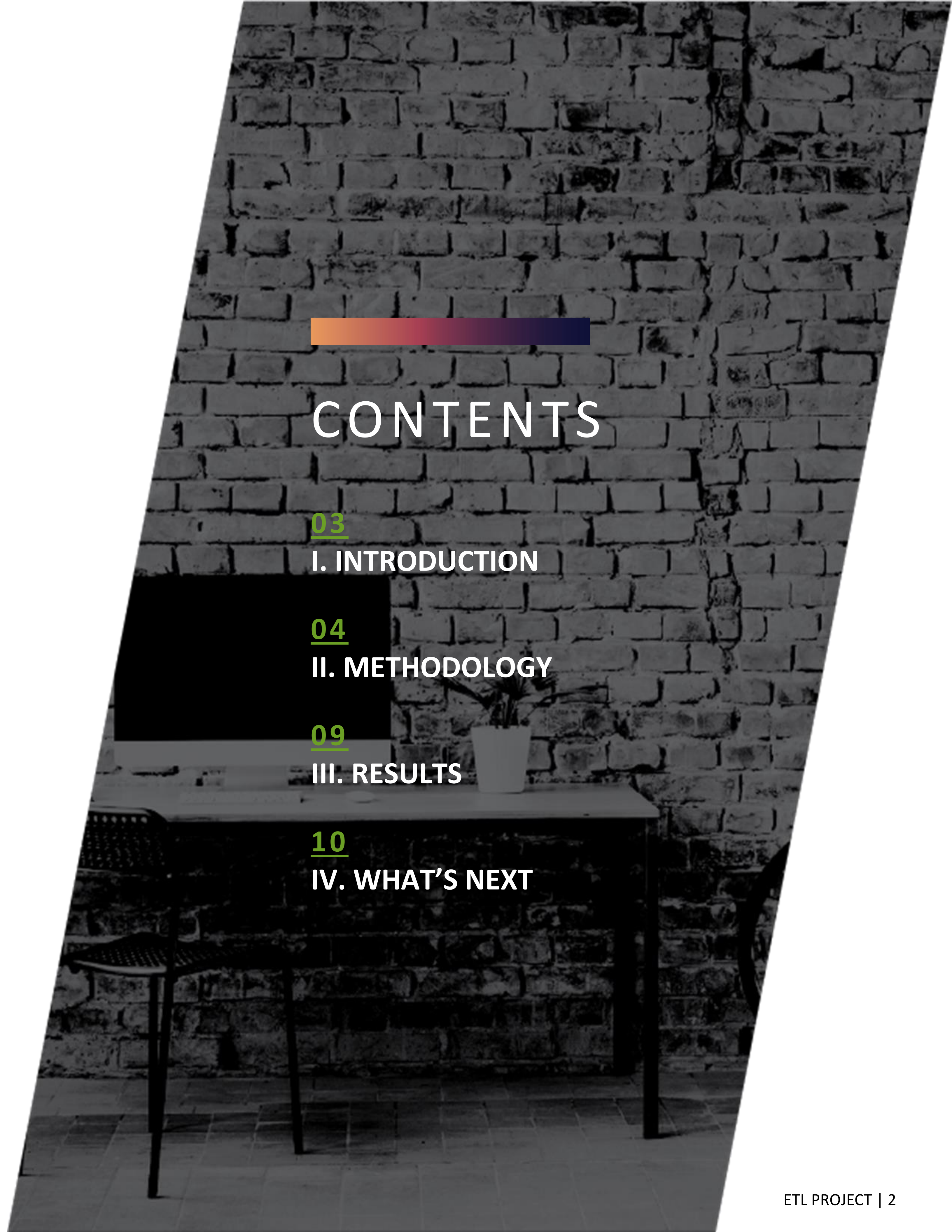


WALMART 2.0

ETL PROJECT

AWESOME POSSUM | github



CONTENTS

GROUP MEMBERS

- Hanieh Babaee
- Ronald Clarke
- May Lacdao
- Rebecca Pham
- Elizabeth Salas-Martinez

INTRODUCTION

MOTIVATION

In business, gaining a competitive edge entails an understanding of the growth of a business and the factors that affect it. Current technological advancements allow businesses to make calculated decisions through data analytics. Retail giants, like Walmart and Amazon, amass huge amounts of data, which, when processed with the right set of tools could provide powerful actionable insights.

For this ETL project, our group decided to expand on our first project and continue working and investigating deeper into Walmart datasets.

DATA SOURCES

Data sets were retrieved from the following sources:

- Kaggle
- data.world
- calendarific
- Reddit
- pypi
- Institute for Local Self-Reliance

METHODOLOGY

EXTRACTION

The following contains the datasets collected and the means in orders to extract the files and data:

CSV

- Datasets:
    1. Sales Zip File (5 Years)
    2. Online Product Database
    3. Walmart Sales 2010-2012
    4. Walmart Daily Stock Data
    5. Prices, Sales and E-commerce Product Details
    6. Growth Profitability and Financial Ratios
    7. Walmart Market Share
- Read through jupyter notebook, using the pandas module

JSON

- Dataset: Store Locations
- Read through jupyter notebook, using pandas module

API

- Dataset: Holiday
- Request get was loaded in order to extract holiday dates
- A for loop was created to get the dates from 2010-2012 in the United States for the 5 holidays that were selected and store in a data frame using the pandas module

Python Package

- Dataset: US python package
- Module was pip installed in jupyter notebook
- The required data were retrieved into dictionary format using the syntax provided by the package developers, which were then exported to csv formats

METHODOLOGY

TRANSFORMATION

Sales Zip File

For both sales and price, the day columns were transposed so the sales and prices changes can be viewed by row. The dataset did not specify when the data was collected so it was not possible to determine the exact start or end date of the data. We later discovered that Walmart was closed only on Christmas Day. It was then determined that there were days with 0 sales for all items and they were 365 days apart, one range being 366 days apart likely due to a leap year. This helped determined what day and month the sales were during. The only thing that could not be determined was the year but knowing the day and month can assist in identifying trends and how sales is impacted from price changes. After determining the date, day and month were included into the dataset. Columns such as state_id and cat_id (category) were also included in replacement of their strings to help reduce storage space. Since the product ID was a string, to reduce storage a product ID table was created with integer id values. Various columns were also deleted as there was no additional information regarding it and its values. After cleaning it resulted in the below table:

- SALES TABLE
- UNIQUE PRODUCT ID TABLE
- STATE ID TABLE
- CAT ID TABLE

For the price file, a similar process was done to the sales dataset. Item, category, and state were replaced with their integer ID. The day and month were also included in the dataset. The dataset for the price changes was large and to gain insightful information it was decided that instead of showing price changes every day for every product a “Markup” and “Markdown” column was created. These columns would count the numbers of markdown and markup every 6 months over the 5 years. It was decided to work initially with 6 months to simplify the data for the future examination of every month may be considered instead to have a more detailed look at price changes. This was calculated by sorting the dataset by the product name and day. If the next row price increased it would be counted as a Markup, if the next row was lowered then it would be counted as a Markdown, and if there was no change it would continue onto the next row without counting anything. The function would sum these counts and then reset the sum once it reached 6 months and this process would iterate. From this function it resulted in the following table: PRICE CHANGE TABLE (6 MONTH/5 YEARS).

Online Product Database

Various columns were deleted such as URL link, UPC and timestamp of data collection. The columns were deleted due to not being able to gain insightful information from them. The main column that was transformed was the categories column as it was formatted to include it’s multiple sub-categories. It was decided to focus on categories and not the sub-categories as the other datasets did not have the column so the online product database alone would not be enough to get important findings from sub-categories. After splitting the first category from the column, the categories column was overwritten with only the first category. Once that was completed the string was then replaced with a category integer ID. The cleaned table is the following: ONLINE PRODUCT TABLE.

Walmart Sales 2010-2012

The extracted zip file contained 3 datasets including features, stores, and training were inner merge was implemented on stores to eliminate NA's. Various columns were deleted such as the price markdown columns due to majority of the values being NA's and due to not being able to determine what the values meant and how it related to weekly sales. The data was also broken down by store and category level but due to the category column being a category integer ID and not being able to determine the category name of the integer ID, it was decided to group the weekly sales by store to remove category.

API Holiday

The holiday table required no transformation or cleaning as the data frame was created during the extraction process of getting the holiday dates from the holiday API. This table was used to merge on the Walmart sales 2010-2012 dataset to identify when holidays occurs and how it impacted the weekly sales.

Walmart Daily Stock Data

The data was filtered on the beginning and end data of the 2010-2012 Walmart dataset in order to merge the 2 data frame for future analysis. Since the stock data was collected daily, a week column was added to identify the days that follow under that week in order to relate this dataset to the Walmart sales file. This was done using the datetime and timedelta module and function.

e-Commerce Product Details

Initially, the Walmart e-commerce category data (which was a composite field of up to 6 levels of category descriptions) was separated and normalized in category level tables. Additionally, a table that 'mapped' the categories was created. To further reduce the size of this data, a single e-commerce table comprised only of the first category level and sale price data were retained. This category level is sufficient for mapping to the existing data.

Walmart Json Data

A function was created to see the data types and total NAN values and their percentage for each column. The function showed that the Walmart.json has around 15% NAN values for openDate and less than a percent null values for timezone and since these two columns will not be included in the analysis, they were left intact to gain other information from those rows. It was then filtered by storeType to include only Walmart Supercenter and Walmart and relevant columns were selected from it with the names of the Walmart store database. The column STATE and ID of state_id data were renamed to state and state_id and then it was merged with store on state.

Walmart Market Share

First the data types were determined. The '%' sign were removed from the values in the 'MARKET_SHARE' column, then datatype was converted from object to integer.

The values (AP style state abbreviations) in 'STATE' column were replaced with the FIPS state codes, using the 'id_code' dictionary created prior.
The 'STATE' column renamed to 'STATE_ID' and set it as index for the final dataframe.

Walmart, Amazon, Target and Costco Financial Database

The first step was to transpose the table, switching index with columns. Since first column was set to index and the dates were positioned as the column names on the first row, when applying df.T to each dataset the dates then became the indexes and the categories became the columns. So, the index was then reset using df.reset_index(inplace=True) on each dataframe. This resulted in a new column with the dates, however, they were under a column named "index" so those column names needed to be changed to "Date". Each dataset also included some values for TTM (Trailing Twelve Months) which muddied the desired clarity so they were dropped using df[:-1]. The Target dataset also had figures for 2020, so those were dropped with df[:-2]. To take a look at which categories had the most complete data, a df.dropna() was performed on each. Looking at the resulting dataframes, the categories that remained in all 4 datasets were considered for further transformation and analysis. The categories decided on were:

- Revenue in Millions of USD
- Net Income in Millions of USD
- Earnings Per Share in USD

The next issue was that the dates were presented as a year-month pair. Some of the datasets had different months, but the figures remained consistent. In order to have consistent data for a date on which to merge, the months were removed using a dt.strftime(%Y) function. Once the dates were consistent, the tables were ready for a merge along the Date column. Merging two tables at a time, columns were renamed to distinguish Walmart, Amazon, Target, and Costco's category columns, respectively. All four tables were merged into one. The Target dataframe unfortunately did not have any values for 2010, so an outer join had to be performed to merge that into the main merged table. The last step was separating the merged table into the three desired tables. This was accomplished by creating new dataframes for each category and selecting the relevant columns to group together with each including Date as the first column. From there, the five dataframes had their indexes set as "Date" to provide a key for the rest of the data so they can be related to each other and any other data linked by a year value for date.

METHODOLOGY

LOADING

After finalizing the tables, an engine was created to load the tables to ElephantSQL, using a connection string. To hide the username and password for the connection, the string was saved into a separate file named ‘config’. In addition, a .gitignore file was created in the root folder to ignore the config file in the checkpoints. For most of the tables, a df.to_sql() was performed with distinguishing names, a connection to the engine, parameter if_exists='append' and index=True.

For the large tables such as sales, prices range, price changes, and product ecommerce, datasets were loaded in SQL using psycopg2 and StringIO. Due to the performance of sqlalchemy being slow with huge datasets, this led to a search for a more efficient way to load the tables. By using StringIO, the data frames could be stored in csv in memory and read in chunks into PostgreSQL using COPY STDIN. Based on time estimates, this was at least 20 times faster for the long tables (originally 80+ million rows).

After all the tables were loaded into the database, a primary key was added to various tables that had an integer IDs as a column. With the final tables loaded, a final check using the table names function was run to ensure that the tables were successfully added to the database. Lastly, the engine was closed to avoid any errors or conflicts.

RESULTS

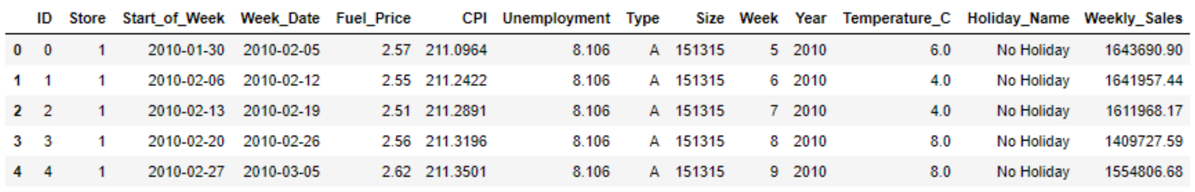
DATABASE

The final database is hosted in ElephantSQL housing all the extracted and transformed tables.

TABLES

The final tables are:

1. Category ID: Category Description, Category ID
2. State ID: State ID, State
3. Sales (5 Year)
4. Walmart Market Share: State ID, City, Population, Market Share
5. Financial Information Comparison Tables
    a. Revenue
    b. Earnings per share
    c. Net income
6. Walmart Store Listing
7. Holiday: Holiday Name, Holiday Date
8. Walmart Sales Data 2010-2012
9. Price Changes
10. Online Product Database
11. Daily Stock Prices and Volume



	ID	Store	Start_of_Week	Week_Date	Fuel_Price	CPI	Unemployment	Type	Size	Week	Year	Temperature_C	Holiday_Name	Weekly_Sales
0	0	1	2010-01-30	2010-02-05	2.57	211.0964	8.106	A	151315	5	2010	6.0	No Holiday	1643690.90
1	1	1	2010-02-06	2010-02-12	2.55	211.2422	8.106	A	151315	6	2010	4.0	No Holiday	1641957.44
2	2	1	2010-02-13	2010-02-19	2.51	211.2891	8.106	A	151315	7	2010	4.0	No Holiday	1611968.17
3	3	1	2010-02-20	2010-02-26	2.56	211.3196	8.106	A	151315	8	2010	8.0	No Holiday	1409727.59
4	4	1	2010-02-27	2010-03-05	2.62	211.3501	8.106	A	151315	9	2010	8.0	No Holiday	1554806.68

Figure 1. Walmart Sales Data from 2010-2012

WHAT’S NEXT

CHALLENGES

Datasets and Data sources

- Walmart Sales 2010-2012: This dataset only included location details for Walmart stores. It also included information about other store types unrelated to Walmart.

- Sales Zip File: This dataset contained several unknowns and unnamed variables. The format of the data required extensive manipulation to create columns by day for sales data, making the data extraction difficult. Despite the cleaning process, the dataset was still over 500MB and over 18 million rows. In order to solve this a random sample of 500 products was selected.

- Financial Data re. Walmart, Amazon, Target and COSTCO: A more thorough month-by-month or week-by-week breakdown of the stats would be more helpful in providing more granular tracking of each corporation's trajectory. This would also afford more opportunities to aggregate specific data for more focused ranges of time.

- Market Share Data by Region: Pre-processing was required to obtain the Walmart Market Share data. The data is originally in pdf and jpg format and needed OCR technology, to retrieve information from jpg and save it in .csv format.

- Online Product Database: The e-commerce dataset has 30,000 rows and appears to be a sample from a larger file. We decided to only use the first level of the e-commerce category, since the price/sales data are categorized at a similar level.

ETL Process

- Storage and Database creation: Too many tables were generated, leading to storage issues. Table had to be reviewed in order to simplify the necessary tables loaded into the database
- Github: Intermediary files were added to .gitignore to free up space

WHAT’S NEXT

FUTURE GOALS

- Analyze the data and determine any actionable insights or trends from the table created
- Be able to use a cloud database with larger storage capacity in order to load and perform analysis on the whole dataset instead of a smaller random sample
- Relate the e-commerce sales to other datasets using the ‘category’ integer ID, revisit adding foreign keys to the dataset in order to link the various integer ID’s implemented
- Webscrape the Walmart website or online shopping platform especially with respect to categories for merchandise being sold and relate it to the sales dataset that also contains category data
- Examine sales and price changes for each month instead of examining only the bi-annual price changes

DATA SOURCES

Data 1
Link: https://www.kaggle.com/naresh31/walmart-recruiting-store-sales-forecasting
Data Source: Kaggle
Data Type: csv
Description: Store sales from 2010-2012 containing various socio economic variables, dates & holidays. (CONT. PROJECT 1)
Key Columns: Date (Day), Store #, Dept #, Sales, CPI, Fuel Price, Temperature, Is_Holiday

Data 2
Link: https://www.kaggle.com/aayushkandpal/walmart-inc-stock-data-19722020-latest
Data Source: Kaggle
Data Type: csv
Description: Stock analysis for Walmart from 1972 through 2020
Key Columns: Date (Day), Open Price, Close Price

Data 3
Link: https://www.kaggle.com/ulrich07/walmartadd?select=sales_aug.csv
Data Source: Kaggle
Data Type: csv
Description: Sales in different categories
Key Columns: id, item_id, state_id, sales1

Data 4
Link:https://gist.githubusercontent.com/anonymous/83803696b0e3430a52f1/raw/29f2b252981659dfa6ad51922c8155e66ac261b2/walmart.json
Data Source: reddit.com
Data Type: .json
Description: Store names, timezone, and location
Key Columns: ID, storeType, timeZone, openDate, name, postalCode, address1, city, state, country, latitude, longitude, phoneNumber

Data 5
Link: https://www.kaggle.com/britneyia/fromcsv
Data Source: Kaggle
Data Type: csv
Description: Financial Data from Walmart, Amazon, Target and Costco from last 10 years.

Data 6
Link: https://ilsr.org/walmarts-monopolization-of-local-grocery-markets/#_edn14
Data Source: Institute for Local Self-Reliance
Data Type: csv
Description: Market share percentage of metro or micro region
Key Columns: City, State, Population, Market Share %

Data7
Link: https://pypi.org/project/us/
Data Source: pypi
Data Type: python package
Description: US and state metadata