ETL Report

Group 6
8/10/22

The primary focus of the data that was collected is about predicting sales information for retailers across the United States. Predicting retail sales would allow for a trickle down effect for numerous markets to more accurately predict inventory levels and plan accordingly across the timespan. We collected four separate datasets for this endeavor. The primary dataset was retrieved from the US census Bureau (Bureau, *All Sectors: Products by Industry for the U.S.: 2017* 2017) involving statistics that fall under the 44-45 retail sales code. This dataset sets the basis for how the retail industry is broken down by NAPCS and subcategorized by NAICS codes. The second dataset retrieved is an excel workbook file (Bureau, *Monthly Retail Trade Report* 2022) with numerous sheets detailing retail sales data broken down by NAICS code. With sales data spanning from the 1990's and broad range retail sales data we then chose to retrieve broad concept tables that relate to United states sales. The first data set is a macroeconomics (Dchaen, *Macroeconomics us* 2022) dataset also broken down by year and month with over 120 columns outlining macroeconomic principles across Census Bureau classifications. Our last dataset retrieved is structured similarly to the macroeconomics dataset in that it is indexed by year and month. This last data set holds consumer price index information (Avigan, *Consumer price index (CPI)* 2020). This will hereafter be referred to as CPI. With the data in place largely broken down by month we used a database composed of all the compiled data to predict future retail sales by different categorizations. Before anything could be composed all datasets needed to undergo ETL processes to be in relatable usable form.

The ET for the excel workbook files was done using python code and pandas in Visual Studio Code, the L was done using pyspark within Azure Databricks.

Excel workbook ETL (Bureau, *Monthly Retail Trade Report* 2022)

- Import pandas
- Use pandas and the read_excel method to create a dictionary of dataframes that are representative of the sheets in the excel workbook file. Ignore the top 4 rows of extraneous data.
- Ignore the 2022 data since it is incomplete and formatted differently.
- Drop ‘Total’ column since it is a calculated value
- Drop row with adjusted [0]
- Rename ‘Unnamed: 0’ and ‘Unnamed: 1’ as NAPCSCode and NAPCSName respectively
- Locate all occurrences of 722 in the NAPCSCode column, write that index value to a list, select the second occurrence, and keep everything above. (not inclusive)
- Create the adjusted sales dataframe by finding ‘ADJUSTED(2)’ index in the NAPCSName column, writing it to a list and selecting everything after +7 rows to remove calculated totals.
- Drop sum rows of index 75 and 77 from the adjusted dataframe. (adjusted data frame should be complete)
- Use the same index selector mentioned two steps previous in creating the adjusted dataframe selecting everything after -6 rows removing unwanted values.
- Select everything after the first 7 rows in the non adjusted dataframe.
- Drop index rows 9 and 14 to remove calculated values.(non adjusted dataframe should be complete.
- Write the two dataframes to new respective dictionaries specifying the naming convention desired for the keys.
- Use .melt to turn the columns, which are month and year combinations, into one column of months and years
- Split the month and year into individual columns
- Make sure all column data types are correct
- Join the adjusted sales dataframe with the unadjusted sales dataframe on month and year
- Write the superframe to csv (“Salestable_final.csv) to be used later

This data will be loaded in at the end of the document

Time-Series ETL

- Using the created dictionaries of dataframes transpose each data frame with the transpose method.
- Use the NAPCSname as the new column header
- Drop the NAPCSname and NAPCSCode rows
- The new index made up of the date was split by month and year. The string representation of the month was converted to a numeric integer representation and both year and month were made into new columns while keeping the index as the combination of the two.
- These new data frames were then written to a new dictionary.
- Combine all data frames from their respective dictionaries in a single data frame in order by date.
- A new index was generated for each row

The other three dataset ETL processes were done using python and pyspark in the Microsoft Azure Gen10 databrick. Multiple files were manufactured to process the data sets and load the numerous information into the database. The ETL process is broken down by file in the databrick.

Producer ETL in producer

- Read the census dataset into a dataframe using pyspark.
- Replace all string values in columns with Null values.
    - For the sales column replace D and A with Null
    - For the number of establishments column replace D and S with Null
- Remove the (s) from the end of numbers in the sales column, remove the commas from the "Sales, value of shipments, or revenue of NAPCS collection code ($1,000) (NAPCSDOL)" and "Number of establishments (ESTAB)".
    - This will create new columns that hold the original data but transformed. Name these “NEWSales” and “NEWEstab” respectively.
- Drop the old columns.
    - "Sales, value of shipments, or revenue of NAPCS collection code ($1,000) (NAPCSDOL)" and "Number of establishments (ESTAB)"
- Change the Sales and Establishment columns to ints.
- Rename the new columns to the correct names.
    - 'NEWSales', 'Sales, value of shipments, or revenue of NAPCS collection code ($1,000) (NAPCSDOL)'
    - 'NEWEstab', 'Number of establishments (ESTAB)'

Census ETL (Consumer)

- Read the census dataset into a dataframe using pyspark
- Create census data frame excluding extraneous columns (census_df).
    - Chosen columns: (‘2017 NAPCS collection code (NAPCS2017)', '2017 NAICS code (NAICS2017)', 'Number of establishments (ESTAB)','Sales, value of shipments, or revenue of NAPCS collection code ($1,000) (NAPCSDOL)’)
- Create a specific data frame to house NAPCS name and code (napcs_df).
    - '2017 NAPCS collection code (NAPCS2017)', 'Meaning of NAPCS collection code (NAPCS2017_LABEL)'
    - Use .distinct() to prevent duplicates
- Create a specific data frame to house NAICS name and code (naics_df).
    - '2017 NAICS code (NAICS2017)', 'Meaning of NAICS code (NAICS2017_LABEL)'
    - Use .distinct() to prevent duplicates
- Write the dataframes to individual csvs within a blob container within Azure Datalake.

The loading is all done within a single databrick

Creating a Database Connection

- To connect to the SQL database you need to define:

    - Database (the name of the database)
    - Tables (each individual table you want to read or write to needs to be defined)
    - User and password (these are used to give you permission to access/alter the database)
    - Server (where the database is located)

Census ETL (Final Steps)

- Read in the csvs to dataframes
    - census_df, naics_id_df, napcs_id_df
- Write the napcs and naics data to their respective tables (NAPCSTables and NAICSTable) within the SQL database
- Read in the data from those tables within the database
    - This provides the primary key that was generated when each table was populated
- Join the census_df and naics_id_df on "2017 NAICS code (NAICS2017)"
- Drop the "2017 NAICS code (NAICS2017)" column
    - At this point the census data should have: 'NAICS ID', '2017 NAPCS collection code (NAPCS2017)', 'Number of establishments (ESTAB)', and 'Sales, value of shipments, or revenue of NAPCS collection code ($1,000) (NAPCSDOL)'
- Join that dataframe with napcs_id_df on "2017 NAPCS collection code (NAPCS2017)"
- Drop the "2017 NAPCS collection code (NAPCS2017)" column
    - The final census columns should now be: 'NAICSID', 'NAPCSID', 'Number of establishments (ESTAB)', and 'Sales, value of shipments, or revenue of NAPCS collection code ($1,000) (NAPCSDOL)'
    - Label this "complete_census_df"
- Populate the census table (NAICS_NAPCS) by writing to jdbc using "complete_census_df"

CPI ETL

- Load in the CPI table .csv file as a dataframe
- Split the "DATE" column into "Year" and "Month" columns
- Rename "USACPIALLMINMEI" column to "CPI"
- Drop date column that was used to populate month and year column
- Drop the leading 0 from the month column

Macro ETL

- Load in the CPI table .csv file as a dataframe
    - Only select these columns: 'observation_date', 'RPI', 'RETAIL', 'USTRADE', 'USWTRADE'
- Rename column "observation_date" to "Date"
- Split the "DATE" column into "Year" and "Month" columns
    - Ignore the day information from the "DATE" column, it is unnecessary
- Drop the "DATE" column

Merge MACRO and CPI

- Merge the Macro dataframe and the CPI dataframe on "Month" and "Year" (merged_and_ordered_df)
- Change the type of "Month" and "Year" to int

- Order the dataframe in ascending order by both “Year” and “Month”

DateTable

- Create a dataframe (time_df) that only contains the “Year” and “Month” from the previous dataframe
- Populate the DateTable by writing to jdbc using time_df
- Read in the DateTable and label the data frame (dateid_df)

MacroTable

- Join the “merged_and_ordered_df” dataframe with “dateid_df” on “Year” and “Month”
- Use .na.drop(subset = ‘DateID’) to drop all rows with null DateIDs
- Drop the “Year” and “Month” columns
- Drop all columns with null values for everything but “DateID”
    - Call this dataframe “macro_table_df”
- Populate the MacroTable by writing to jdbc using “macro_table_df”

Excel workbook ETL (Final Steps)

- Load in the “Salestable_final.csv” into a dataframe called “sales_df
- Create a dataframe titled append_NAICS_df that takes the NAICScode and NAICSname from “sales_df”
    - Use .distinct() to ensure there are no duplicates
    - Rename the columns '2017 NAICS code (NAICS2017)' and 'Meaning of NAICS code (NAICS2017_LABEL)' respectively
    - Append the NAICSTable by writing to jdbc and using “append_NAICS_df”
- Drop the ‘_c0’ column from “sales_df”, it is unnecessary
- Inner join sales_df with dateid_df on “Year and “Month”
    - Label this “cleaned_sales_df”
    - This provides the DateID
    - Drop “Year” and “Month”
- Read in the updated NAICSTable as a data frame
- Create a new data frame that joins cleaned_sales_df with the NAICSTable
    - This provides the NAICSID
- Drop all NAICS related columns from cleaned_sales_df except for the NAICSID
- Replace all string values within the “Adjusted Sales” and “Unadjusted Sales” columns with null values
- Populate the SalesTable by writing to jdbc

The steps provided took an ample amount of time across the entirety of our team. By dividing the datasets into subtasks and splitting the work it was able to be completed in a timely manner. Through these processes we were able to produce a working database that is normalized with no repeated data and easily traversed structure.

Sources:

Bureau, U. S. C. (2017). *All Sectors: Products by Industry for the U.S.: 2017*. Explore census data. Retrieved August 3, 2022, from https://data.census.gov/cedsci/table?q=ECNNAPCSPRD2017.EC1700NAPCSPRDIND&n=N0600.44&tid=ECNNAPCSPRD2017.EC1700NAPCSPRDIND&hidePreview=true

Avigan, A. (2020, February 16). *Consumer price index (CPI)*. Kaggle. Retrieved August 4, 2022, from https://www.kaggle.com/datasets/aavigan/consumer-price-index-usa-all-items

Dchaen. (2022, May 11). *Macroeconomics us*. Kaggle. Retrieved August 4, 2022, from https://www.kaggle.com/datasets/denychaen/us-macro?select=US_MACRO110522.csv

Bureau, U. S. C. (2022, July 15). *Monthly Retail Trade Report*. United States Census Bureau. Retrieved August 10, 2022, from https://www.census.gov/retail/index.html