AWS AIRFLOW SETUP

The key steps in building a data pipeline are:

1. Selecting the cloud platform, which in this case is AWS.
2. Choosing the data storage solution, which is AWS S3 for storing the raw data.
3. Implementing the database for storing the processed data, which is AWS RDS MySQL DB.
4. Utilizing data orchestration tools, specifically Apache Airflow deployed on an AWS EC2 instance, to manage the data workflow.
5. Performing the data processing tasks using AWS EMR and Spark.

By combining these components, an effective and automated data pipeline can be established to ingest, transform, and load data for further analysis and business intelligence.

AWS services provisioning

- I have created an ec2 ubuntu machine t2 small (*ec2-3-15-34-218.us-east-2.compute.amazonaws.com*) in AWS cloud .
- I have installed Airflow in the Ubuntu machine
- I have made use EMR(emr-6.8.0,Hadoop 3.2.1, Spark 3.3.0)
- I have provisioned RDS instance mysql database (*income-mysql-db.cjuyq4k026ji.us-east-2.rds.amazonaws.com*)
- I have uploaded the sql client for jdbc connectivity (*s3://income-sales-data/airflow/scripts/spark-mssql-connector_2.12-1.3.0-BETA.jar*)
- The input csv files are available in S3 bucket (*s3://income-sales-data*)
- Created a Key Pair for EMR
- Created an AWS CLI role with admin access.
- Run the command aws emr create-default-roles to get the default roles with IAM policies attached to facilitate communication between AWS services.

IDE

- I made use VS code editor to SSH into the remote machine.
- I connected the database using the database connector in VS code

Airflow Installation steps

- sudo apt update sudo apt install python3-pip
- sudo apt install python3.10-venv
- python3 -m venv python3 -m venv income_eccomerce_venv
- source income_eccomerce_venv/bin/activate
- sudo pip install apache-airflow
- pip install apache-airflow-providers-amazon
- pip install pyopenssl==24.0.0
- airflow standalone

Local Setup (for quick testing)

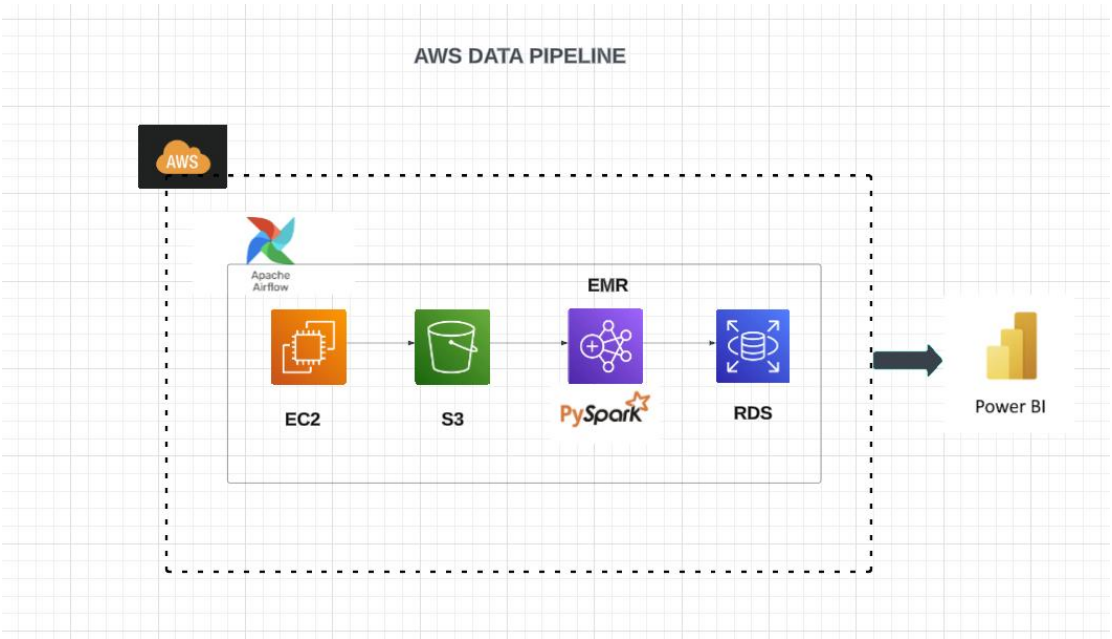
- Installed Hadoop 3.2.1, Spark 3.3.0
- Pyspark version 3.1.1
- Anaconda Navigator (Jupyter Notebook)for testing in local

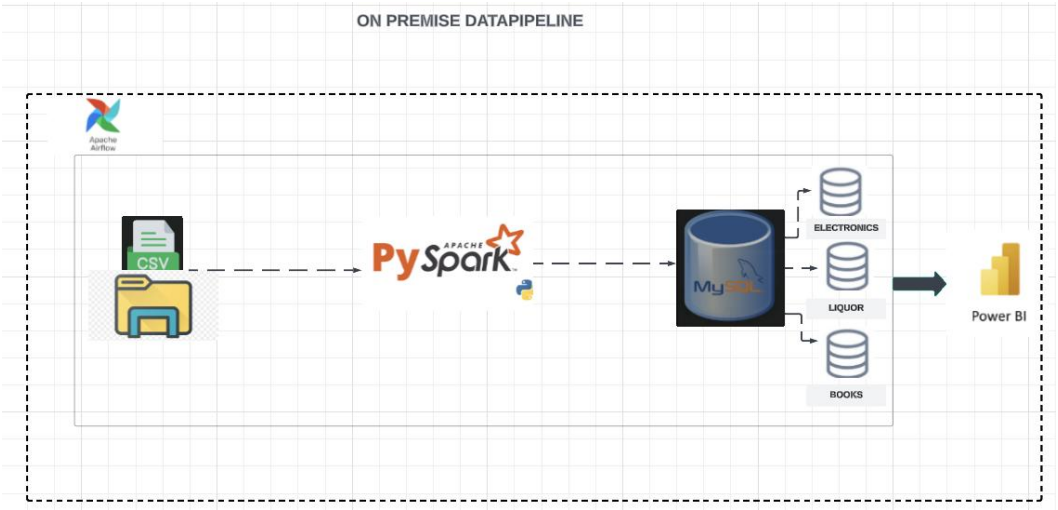
Architecture Diagram

The key elements of the architecture are:

- The data orchestration is handled by Apache Airflow, which is deployed on an Amazon EC2 instance.
- The Airflow orchestrator spins up Amazon EMR clusters to perform the data processing tasks.
- The processed data is then loaded into an Amazon RDS MySQL database instance.
- The data marts for Electronics, Liquor, and Books are connected to a Power BI dashboard.
- The Power BI dashboard is used to create visualizations and provide useful insights based on the data in the data marts.
- I have implemented 2 pipelines. 1) Cloud 2) In Device Local Laptop
- I did this to do quick testing in local. Hence I will push to the cloud if the logic is working in the local

AWS DATAPIPLINE ARCHITECTURE

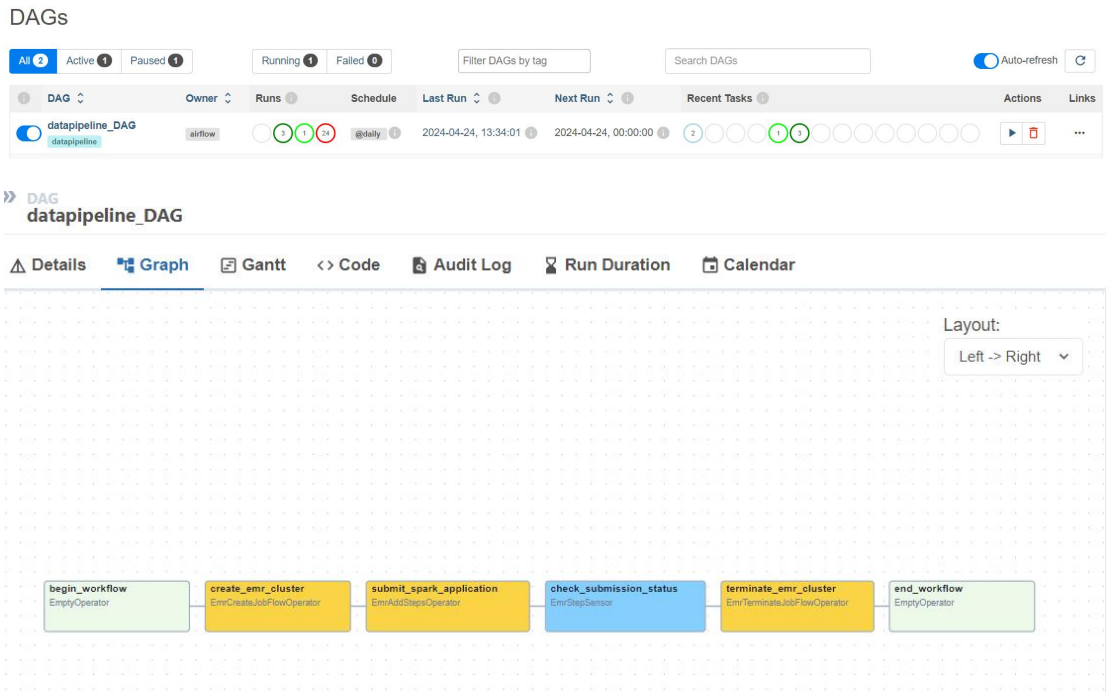




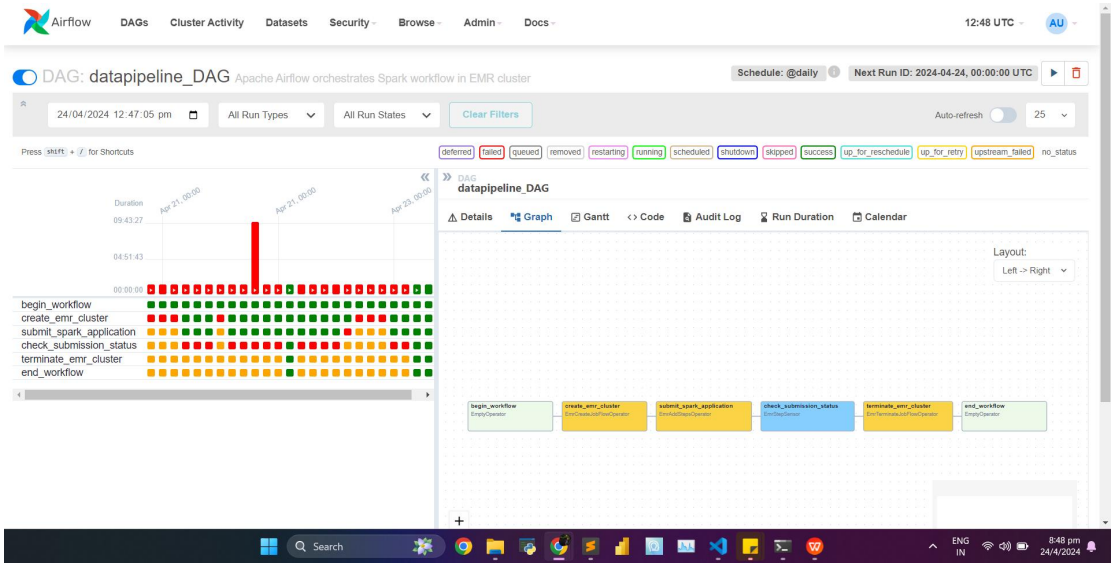
AIRFLOW DAG ARCHITECTURE

- The dag triggers reads the data from S3, spins the EMR cluster where data processing and transformations are done.
- The schedule frequency is configurable using cron expressions, I have currently configured it to run daily .

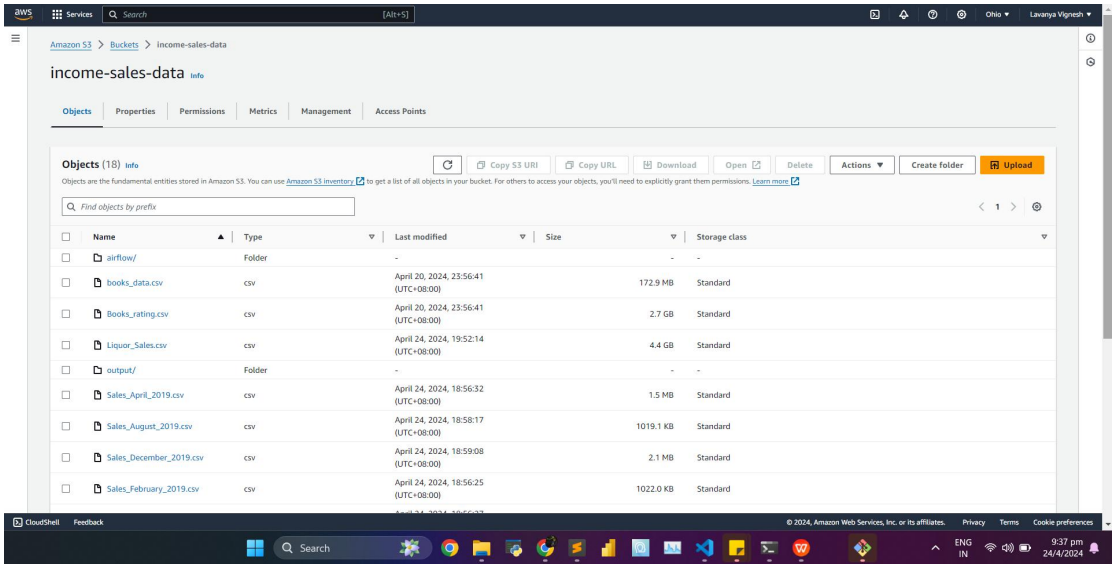
DATAPIPLINE DAG STEPS



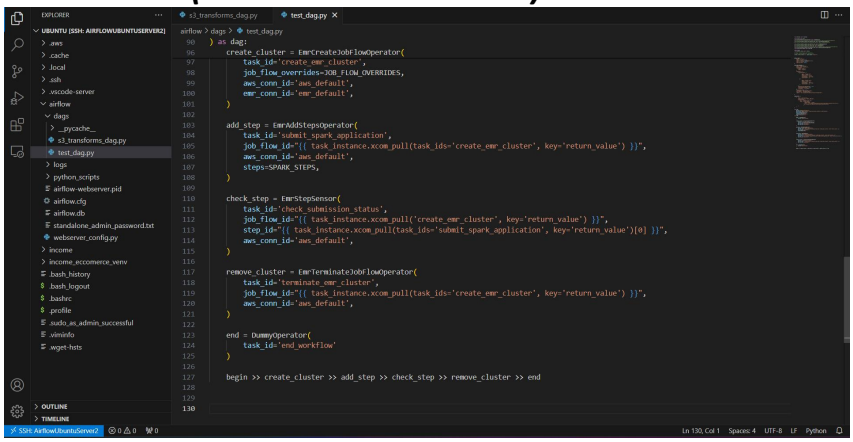
AIRFLOW GRAPH VIEW



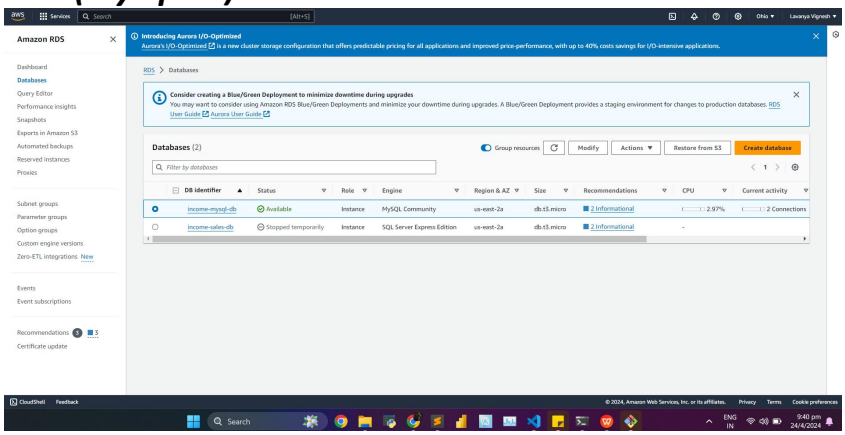
S3 SNAPSHOT

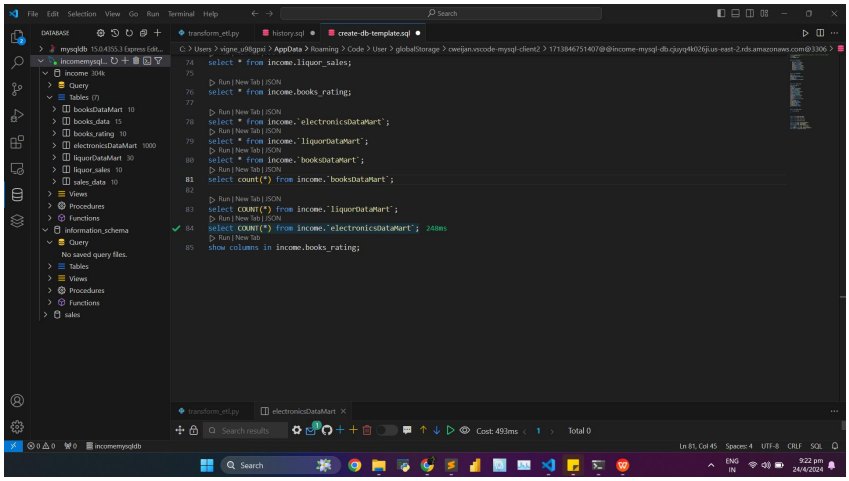


Remote SSH(EC2 UBUNTU SERVER)



RDS (mysql db)





GIT HUB LINK

https://github.com/Lavanya-Mohan/Income_Case_study

BONUS

Cloud Integration: Architect the solution using cloud services (AWS/ GCP / Azure)

- The data pipeline is successfully implemented on AWS cloud

to enhance scalability, performance, and reliability. Data Enrichment for Analytics : Deriving information from existing fields :

- I have derived columns from date fields, store location,category, address etc

Classifying the electronic products into categories like laptops, mobile phones etc. can help business to get a drill down view of insights.

- Implemented

Derive state, country information from the address to enable analysis at region level

- Done

o Categories columns imputation using simple NLP techniques for books data

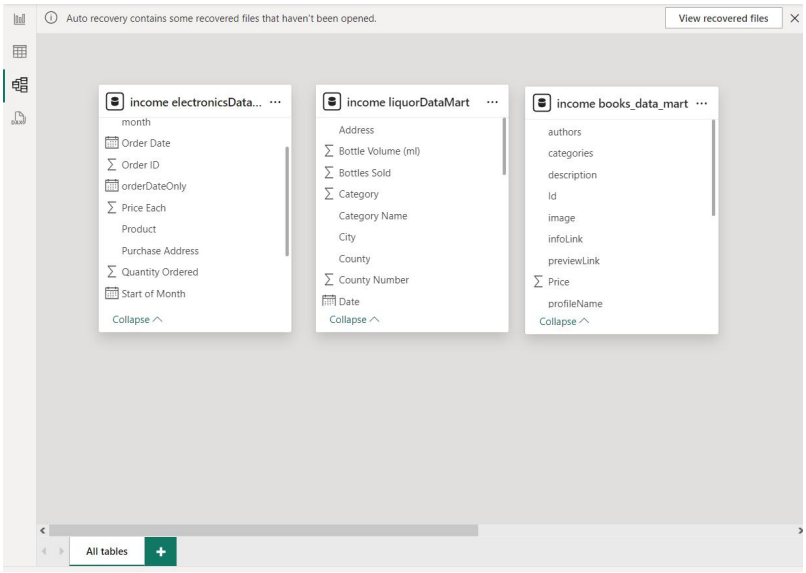
I have imputed the columns using simple NLP technique

1) The columns ['image', 'previewLink', 'infoLink',] had irrelavant information like numbers and string. Cleaned the columns to contain only the URL's
2) Imputed ['publishedDate','ratingsCount','review/score','review/time', 'Price']columns to contain only relevant data.The columns contained irrelevant junk strings.
3) Imputed the columns ['Title', 'description','authors','publisher','categories','User_id',] to contain only string information
4) Imputed ['review/helpfulness',] to be in the format of 1/5, 4/5 and removed the junk.

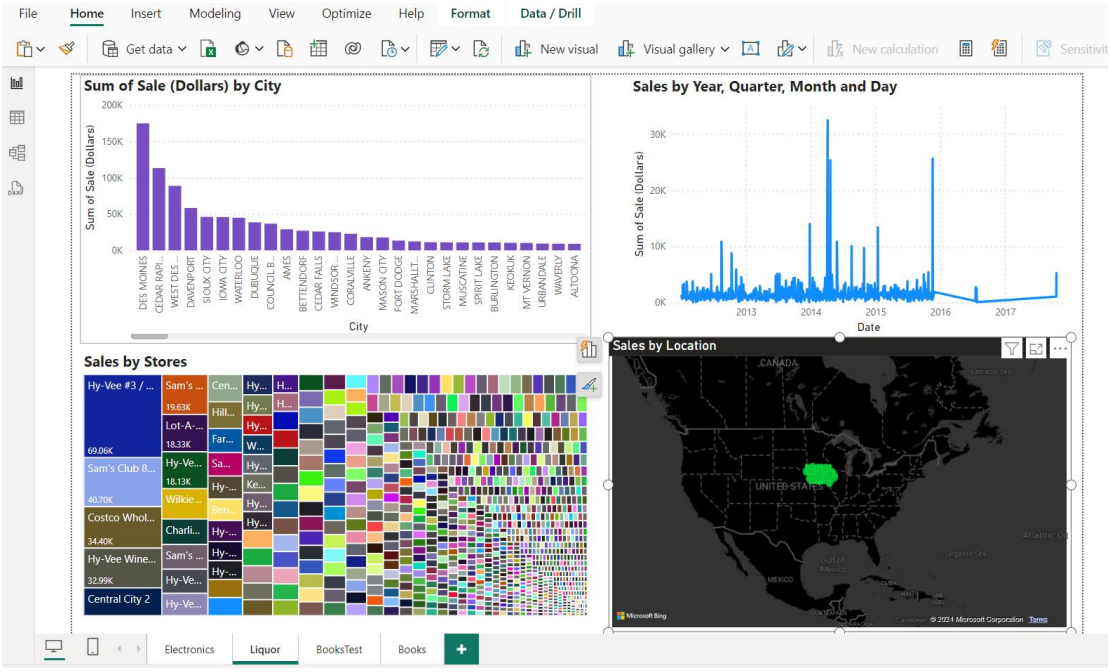
POWER BI DASHBOARD

I am using desktop version of Power BI

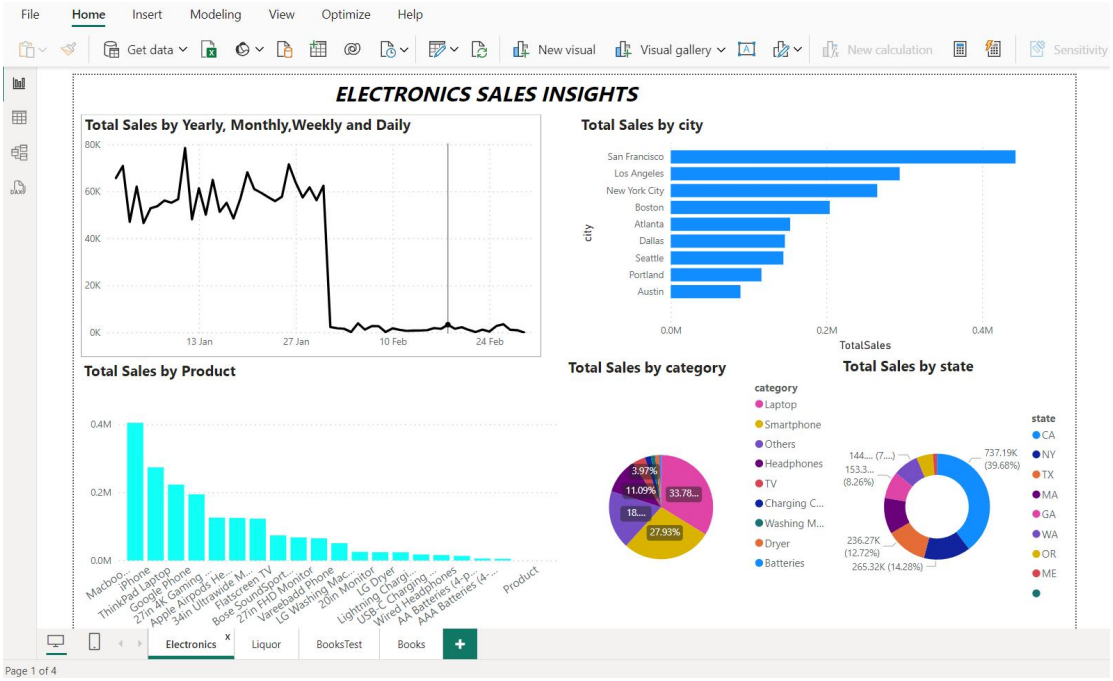
DATA MART -DATA MODEL



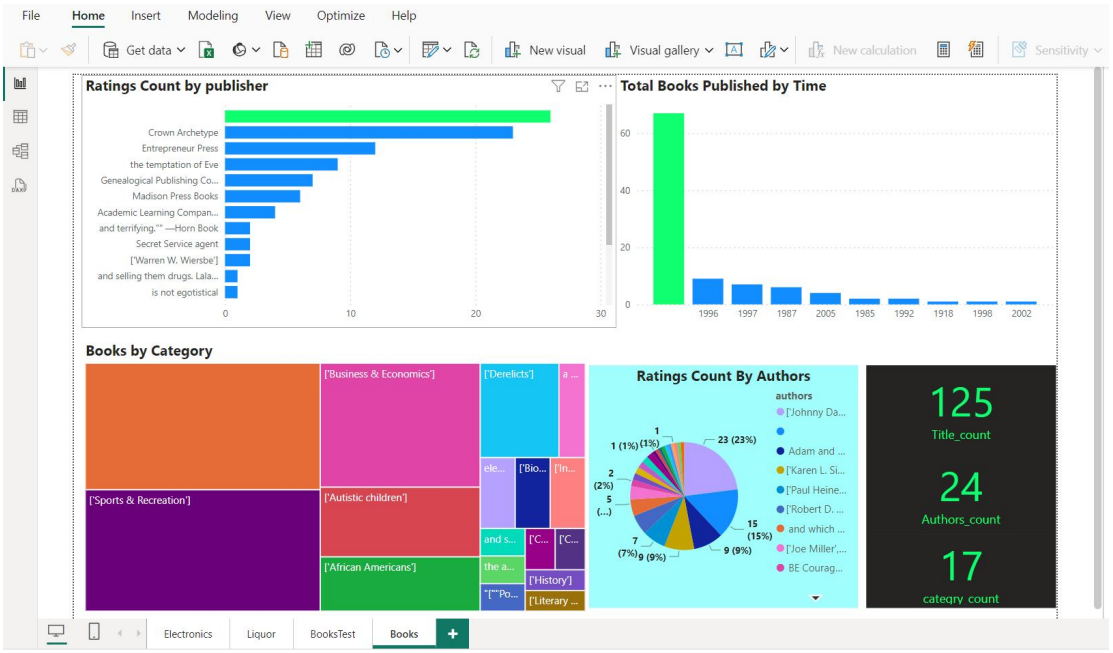
LIQUOR SALES INSIGHTS



ELECTRONICS SALES INSIGHTS



BOOKS SALES INSIGHTS



CHALLENGES FACED

- While we restart the EC2 machine, we cannot SSH through the private IP,we can only SSH with public DNS name.
- It took time so much time to analyse all the free tier frameworks within AWS.There is a tradeoff with data storage as well.So I decided to pay for some services to make everything in cloud
- There was some version compatability issues while installing spark in local.The pyspark,spark, jdk and hadook versions has to compatible.

- I Initially experimented with MS SQL data base in RDS, but for the latest MS SQL SSERVER there was no proper jdbc connection support, so I used MY SQL database.

Future Works

- Given the time constraint, where I need to implement full scale end to end architecture in cloud. I ended up spending lot of time in choosing the right framework/services under free tier account. But given time and oppurtunity I can do absolute best.
- Dashboard can be improved further with different views
- Data imputation quality can be improved by adopting classifier technique. We can leverage Deep Learning Embedding to choose the closest category for the null values. Similarly we can perform relevant ML/DL techniques to impute NULL values smartly.
- Data pipeline can be more optimized with incremental load