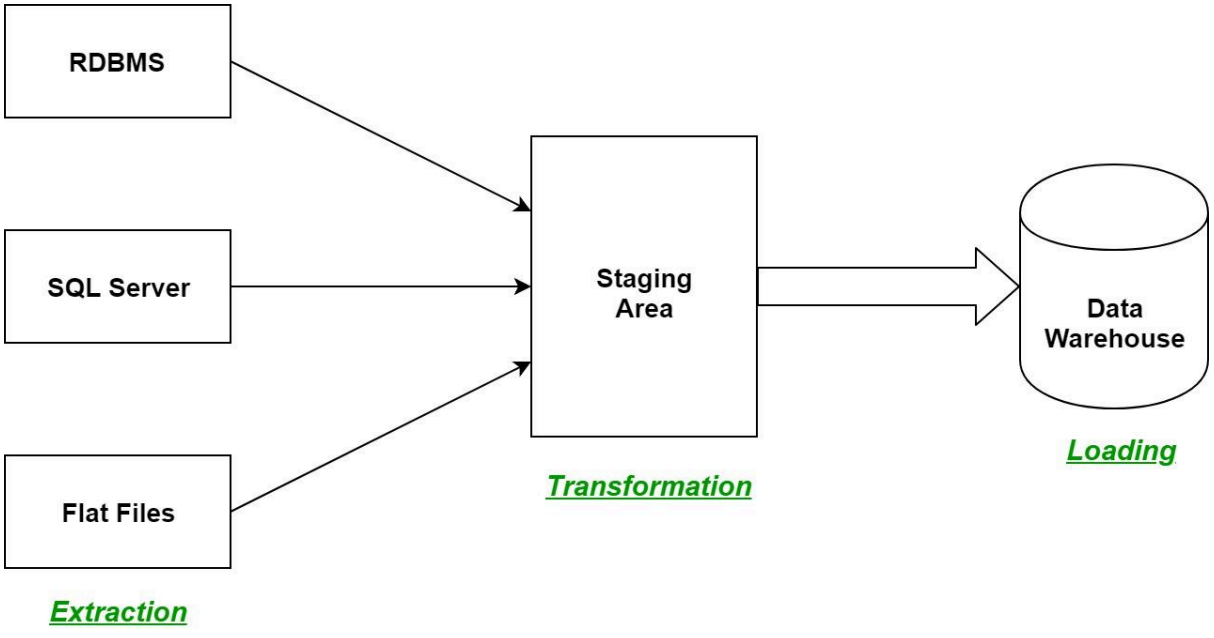
ETL Pipeline Using Multiple Data

Course: Big Data Analytics (BDA)

(CT-592)

Program: Masters in Data Science



Submitted by:

Name: Mehwish
Roll no: DS-W-013/2024-25

Submitted to Dr. Muhammad Umer Farooq

Data Extraction

In this project, data is extracted from **five different sources**, ensuring a comprehensive view of mobile sales, customer feedback, and market trends. The data sources include:

1. **Spreadsheet (Excel) – Mobile Sales Data**
    - The primary sales data is stored in an Excel spreadsheet (`mobile_sales.xlsx`).
    - It contains details such as **phone models, units sold, revenue (USD), and transaction dates**.
2. **JSON File – Customer Reviews**
    - A JSON file (`customer_reviews.json`) provides **customer ratings and feedback** on different phone models.
    - This helps in understanding customer satisfaction and sentiment.
3. **CSV File – Marketplace Sales Data**
    - A CSV file (`marketplace_sales.csv`) contains sales information from **various online marketplaces**.
    - It provides insights into **product demand and pricing trends**.
4. **MongoDB – Inventory Data**
    - Inventory data is stored in a **MongoDB database** (`mobile_store.inventory`).
    - This dataset includes **stock levels and last restocked dates** for different phone models.
5. **REST API – Competitor Pricing**
    - A **FastAPI-based REST API** (`https://ec06-34-85-135-100.ngrok-free.app/competitorPricing`) is used to fetch **competitor pricing data**.
    - This allows for price comparisons with competitors.

Data Transformation

Handling Missing Values

Missing values can cause errors in data analysis and affect model performance. The following techniques were used to handle them:

- **For numerical data** (e.g., sales revenue, units sold, stock levels):
    - Missing values were replaced with the **median** of the respective column to maintain consistency and prevent skewing the dataset.
- **For categorical data** (e.g., model names, review comments, competitor names):
    - Missing values were replaced with `"Unknown"` to maintain the dataset structure without introducing misleading values.

Removing Duplicates

Duplicate records can lead to incorrect data aggregation and analysis. To ensure data integrity:

- **All datasets** were scanned for duplicate rows based on key attributes such as phone model, date, and price.
- Identical records were removed to prevent **overcounting in sales and pricing analysis**.

Data Normalization

Normalization ensures that data is in a consistent format across all sources. The following transformations were applied:

a. Date Standardization

- Dates were converted to the **ISO 8601 UTC format** (`YYYY-MM-DDTHH:MM:SSZ`).
- This ensures uniformity across different sources where some might have **MM/DD/YYYY** or **DD-MM-YYYY** formats.
- Standardized timestamps help in accurate time-based analysis and trend detection.

b. Standardizing Product Names

- Phone model names from different sources were not always consistent.
- Some used uppercase letters (`IPHONE 14`), some used lowercase (`iphone 14`), while others had mixed formatting (`iPhone 14`).

- All product names were converted to **title case** (e.g., `Iphone 14`) for consistency in merging datasets.

Data Aggregation

To extract meaningful insights, data was aggregated at the **phone model level**:

- **Total Sales Per Model**
    - The sum of **units sold and revenue** was calculated for each phone model to understand its market demand.
- **Average Rating Per Model**
    - Customer reviews were grouped by model, and the **mean rating** was calculated to determine overall customer satisfaction.
- **Sales vs. Stock Levels**
    - Inventory data was merged with sales data to compute **remaining stock** and analyze restocking needs.

Feature Engineering

Additional features were derived to enhance data analysis:

- **Revenue Per Unit Calculation**
    - The revenue per unit sold was computed to assess pricing efficiency and profitability.
- **Customer Sentiment Classification**
    - Customer ratings were categorized into:
        - **Positive** (rating ≥ 4)
        - **Neutral** (rating = 3)
        - **Negative** (rating ≤ 2)
    - This helps in **understanding customer satisfaction trends**.

Standardizing Measurement Units

Since sales revenue from different sources might be in **different currencies**, all revenue figures were converted to **USD** using predefined exchange rates:

- **EUR → USD (1.1)**
- **GBP → USD (1.3)**
- **INR → USD (0.012)**
- **USD → USD (1)**

This ensures a **uniform monetary unit** for financial analysis.

Data Validation

To ensure data quality, records with **incorrect or unrealistic values** were flagged and removed:

- **Negative Sales Figures**
    - Transactions where the number of units sold was **less than or equal to zero** were removed, as they indicate incorrect entries.
- **Zero or Negative Prices**
    - Any product with a **competitor price ≤ 0** was filtered out since it is not a valid price.

Final Output

After data preprocessing, the cleaned and structured dataset was ready for analysis and storage. The refined dataset was:
✅ **Free of missing values and duplicates**
✅ **Standardized in terms of formats (dates, product names, and currency)**
✅ **Validated for inconsistencies**
✅ **Enhanced with new features (e.g., revenue per unit, sentiment analysis)**

Data Consolidation and Storage

- The final **consolidated dataset** is saved in:
    - **CSV file** (`consolidated_data.csv`) for further analysis.

    - **MongoDB collection** (`mobile_store.consolidated_data`) for structured storage and querying.

Use of Final DataFrame for Trend & Pattern Analysis

After cleaning, normalizing, and aggregating the data, the final **consolidated DataFrame** can now be used to derive meaningful insights and trends:

Sales & Revenue Trends 📊

- Track which **phone models** generate the highest revenue.
- Identify **seasonal trends** in mobile sales (e.g., increased sales during holiday seasons).
- Compare sales across different **marketplaces and competitor pricing** to determine price competitiveness.

Customer Sentiment Analysis 📝

- Use **average rating** and **sentiment classification** to determine how customers perceive each phone model.
- Identify potential correlations between **pricing and customer satisfaction** (e.g., Are lower-priced phones receiving more negative reviews due to quality issues?).

Inventory & Restocking Optimization 📦

- By **merging inventory with sales data**, businesses can track which models are selling fast and which have excess stock.
- **Stock levels vs. demand** analysis can help in planning restocking strategies to avoid shortages or overstocking.

Competitive Market Analysis 💰

- **Competitor pricing vs. sales trends** can highlight whether pricing influences purchasing decisions.
- **Price sensitivity analysis**: Identify if changes in a competitor's price affect sales volume.

Revenue Per Unit & Profitability Analysis 💵

- Calculate the **revenue per unit** to assess how efficiently each model contributes to total revenue.

- Compare models with **high sales but low revenue per unit** vs. **low sales but high revenue per unit** to optimize pricing strategies.

Automation Overview

This ETL pipeline is automated to run daily using the `schedule` module in Python. The `scheduler.py` script triggers the main ETL process (`etl_pipeline.py`) every 24 hours. It collects and processes data from five sources (CSV, JSON, API, MongoDB, and Google Sheets), performs cleaning, transformation, and then loads the consolidated data into a MongoDB collection. This automation ensures the database stays updated with fresh, clean data without manual intervention.

Final Thoughts:

By transforming raw, unstructured data into a **clean, structured, and unified** dataset, the pipeline enables deeper analysis and decision-making. Whether tracking sales trends, monitoring customer sentiment, or optimizing inventory, this DataFrame serves as a **valuable asset for business intelligence in the mobile sales industry**.