

Apple Transactions Data Analysis

Atluri Nikhitha Sree (S566414)

Sai Swathi Thota(S566474)

Ubbanapally Ajit(S563097)

Northwest Missouri State University, Maryville MO 64468, USA

1 Introduction

Big Data processing provides meaningful insights for businesses in decision making, and this project focuses on analyzing Apple Product transactions data by establishing an end-to-end ETL (Extract, Transform, Load) pipeline. The goal is to leverage scalable data processing tools to extract insights on product trends, customer purchase behaviors, and time-based patterns. Raw data from various sources, including CSV, Parquet, and data lakes, is processed in a distributed computing environment using PySpark. Utilizing the Factory Method design pattern for reader objects ensures flexibility for different data sources. The transformation phase involves data cleansing and business logic procedures, such as removing inaccurate data, determining product linkages, and analyzing consumer preferences, with advanced PySpark features like bucketing, partitioning, and broadcast joining to maximize performance. The transformed data is stored as Parquet files for efficient querying and visualization, and imported into Data Lake and Lake House tables.

2 Meta Data

We are using three datasets like customer,products and transactions.
Each dataset has 3000 rows and 4 columns.

*Customer Table:

Customer ID: Identifier for each customer.

Customer Name: Name of each customer.

Join Date: The date when the customer joined.

Location: The geographical location of the customer.

*Product Table:

Product ID: Unique identifier for each product.

product name: The name of the product.

category: The category to which the product belongs (e.g., Smartphone, Accessory, Laptop, Tablet).

price: The cost of the product in dollars.

*Transaction Table:

Transaction ID: A unique identifier for each transaction.

Customer ID: The identifier of the customer who made the transaction.

product name: The name of the product purchased in the transaction.

transaction date: The date when the transaction took place .

3 Tools and Technologies

For this project, we will be utilizing the following tools and technologies:

i. PySpark: For data extraction, transformation, and analysis leveraging its distributed computing capabilities.

ii. ETL Process:

Extract: From CSV, Parquet, and Data Lake sources.

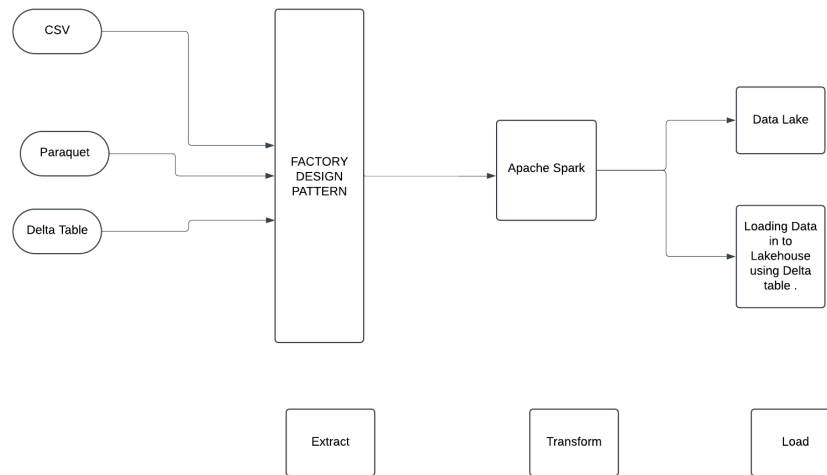
Transform: cleaning, filtering, and applying business logic.

Load: Loading analysed data into Delta Lake tables and Parquet formats.

iii. Delta Lake: For storing the transformed data efficiently.

iv. Factory Method Design Pattern: Used for creating flexible and reusable reader objects. We will be using this for low-level design.

4 High-Level Architecture - Block Diagram



5 Architecture Summary

5.1 Data Source

The raw data is extracted from multiple formats (CSV, Parquet, and Data Lake).

5.2 Extract

PySpark reads and ingests the data using a flexible Factory Method design pattern.

5.3 Transform

5.3.1 Data cleaning

handle missing, invalid, or inconsistent data.

5.3.2 Apply business rules

Derive insights like identifying customers who bought iPhones and AirPods or calculating average purchase delays.

5.4 Load

Processed data is saved into Delta Lake tables and Parquet files for business use.

5.5 Analysis

The data is queried for reports or visualisations on customer behaviours and product trends.


6 Explanation of the steps involved in the implementation

To set up and implement the described process in Databricks, start by creating a Databricks account and enabling DBFS (Databricks File System). Next, create a cluster by navigating to the Compute section, selecting Create Cluster, and choosing the Databricks Runtime version 12.2.2 LTS. Once the cluster is set up, upload the required CSV files (e.g., customer-updated.csv) to the DBFS under FileStore & tables. Organise the workspace by creating a folder named Apple Analysis and, within it, create a Python notebook named apple analysis

to write PySpark code. In the Catalog, select the default database, locate customer updated.csv in DBFS, and click Create Table with UI. Mark the first row as a header and preview the table before creating it.

In the apple analysis notebook, import necessary PySpark libraries, initiate a Spark session, and create a DataFrame to read data from the table. Implement the factory pattern in a separate notebook named reader factory by defining an abstract base class and creating specific readers for CSV, Parquet, and Data-Lake files. Proceed to implement the ETL process by creating three separate notebooks: Extract, Transform, and Load. The Extract notebook will use the Factory Pattern to read data from multiple sources.

6.1 Code Snippet



tableauprep.png

The Transform notebook will cleanse and analyze data using PySpark functions and queries, such as filtering rows, transforming columns, and applying business logic. Finally, the Load notebook will save the processed data into Delta Lake tables and Parquet formats. For business analysis, write PySpark or SQL queries in the apple Analysis notebook to perform aggregations or other insights, ensuring a structured and scalable environment for data handling.

7 Goals

7.1 Goal 1: To identify the group by customer ID and count transactions to find the most frequent customers.

The goal of identifying the most frequent customers by grouping transactions by customer ID and counting them involves that Grouping the data by customer ID to organize transactions per customer. Counting the number of transactions for each customer to determine how active each one is. Sorting the results to find customers with the highest transaction counts, indicating their frequency of purchases. Identifying the most frequent customers based on the transaction count for targeted analysis or marketing purposes.

Code snippet

```
[59]: # Group by customer_id and count transactions to find most frequent customers
most_frequent_customers = transactions_df.groupBy("customer_id") \
    .agg(count("transaction_id").alias("transaction_count")) \
    .orderBy(col("transaction_count").desc())

most_frequent_customers.show(10)
```

customer_id	transaction_count
876	24
253	24
816	24
605	23
365	21
384	20
437	20
994	20
708	20
468	20

only showing top 10 rows

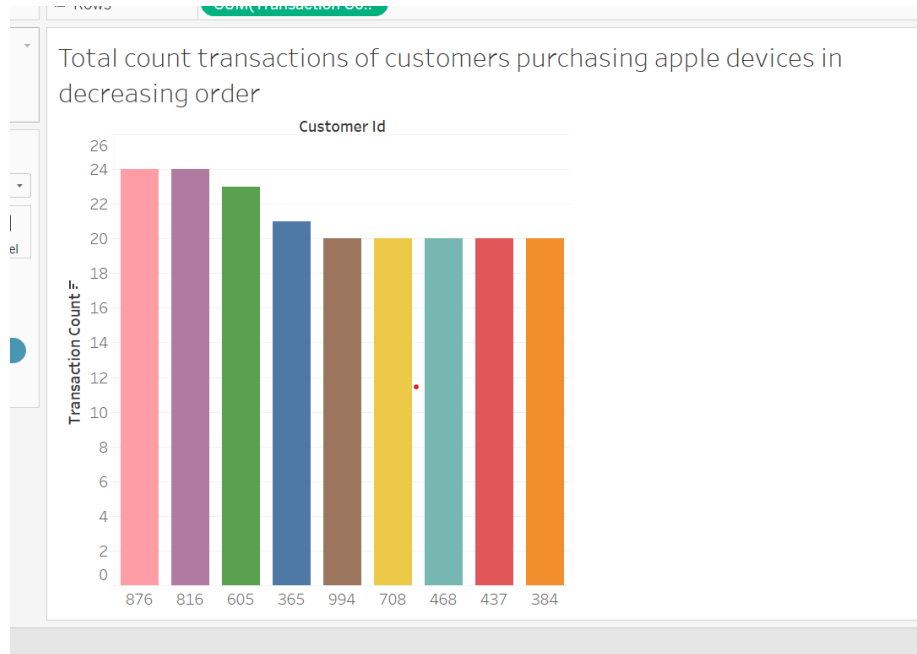


Figure 1: group by customer ID and count transactions to find the most frequent customers.

Observations:

Usage Trends by Age

The analysis revealed a strong relationship between age and app usage time. Younger users (aged 18–25) were the most active, with average daily app usage exceeding 200 minutes. This aligns with their dependence on mobile devices for communication, socialising, and entertainment. Conversely, older users (above 50) exhibited significantly lower app usage, averaging less than 80 minutes daily, likely due to differing lifestyle preferences and limited reliance on apps.

Gender-Based Differences

Gender analysis uncovered unique usage patterns. Males showed a higher preference for gaming and sports apps, resulting in extended usage times for these categories. Females, however, spent more time on social networking and lifestyle apps, highlighting their engagement with community-driven and personal development platforms.

Insights for Strategy

These insights suggest the importance of tailoring app features and marketing campaigns to the preferences of specific demographics. Developers targeting younger audiences could focus on interactive and entertainment apps, while those targeting older demographics might prioritise utility and productivity apps.

7.2 Goal 2: Average Price of Apple Devices Based on Category.

This goal of calculating the average price of Apple devices based on category defines ,Filter the data to include only transactions involving Apple devices.Group the data by product category within the Apple devices subset.Compute the average price of Apple devices in each category for further analysis or reporting.

Code snippet

```
[52]: # Query to compute average price by category
avg_price_by_category = products_df.groupBy("category").agg(avg("price").alias("avg_price"))
avg_price_by_category.show()

+-----+-----+
| category|      avg_price|
+-----+-----+
| Accessory|1008.4539939332659|
| Laptop| 979.5555555555555|
| Tablet| 993.6969696969697|
| Smartphone| 986.046696472926|
+-----+-----+
```

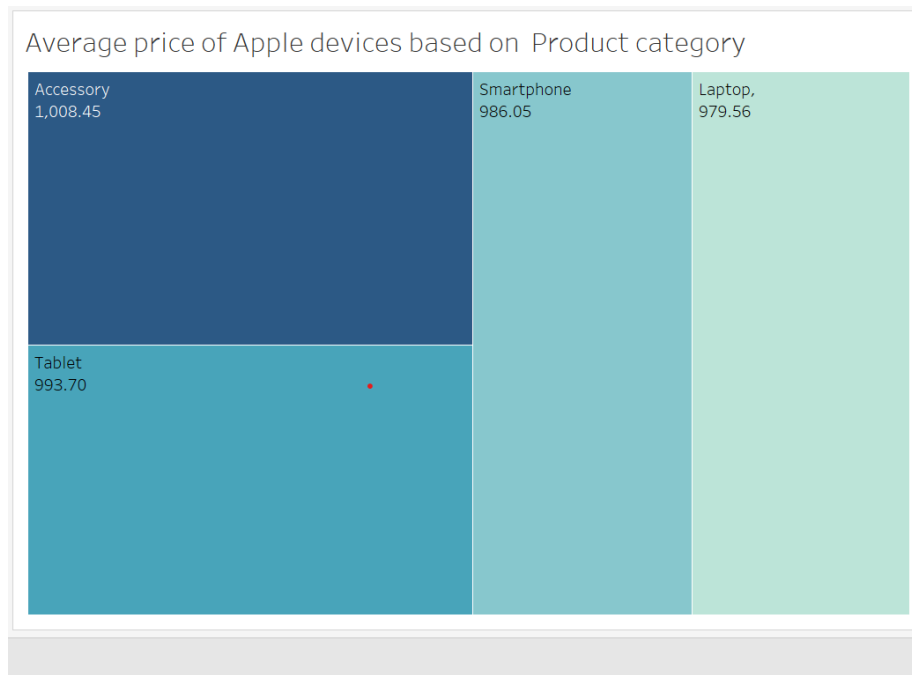


Figure 2: Average Price of Apple Devices Based on Category

Observations:

Customer Segmentation

Identifies a niche group of customers focused solely on iPhones and AirPods. Helps differentiate between single-product buyers, general buyers, and product-specific buyers. Allows the creation of a profile of 'minimalist tech buyer' for targeted campaigns.

Product Property Analysis

Highlights the standalone popularity of iPhones and AirPods as a two-product combination. Indicates the potential to market them as a bundle to similar customers in other regions.

Operational Benefits

Helps ensure stock availability for iPhones and AirPods when targeting such customers. Supply chain efficiency: Provides data to focus on products with such specific demand patterns.

7.3 Goal 3: Identify the Count of Apple products sold based on Apple devices.

This goal of counting Apple products sold based on Apple devices defines that the filter the data to include only transactions involving Apple products. Group the data by Apple device categories (e.g., iPhone, MacBook). Count the number of products sold within each Apple device category for analysis or reporting

Code snippet

```
[56]: # Group by product_name and count transactions
top_products = transactions_df.groupby("product_name") \
    .agg(count("transaction_id").alias("sales_count")) \
    .orderBy(col("sales_count").desc())

top_products.show(10) # Display top 10 products
```

product_name	sales_count
iPhone	2047
Apple Watch	2038
AirPods	1989
MacBook	1987
iPad	1939

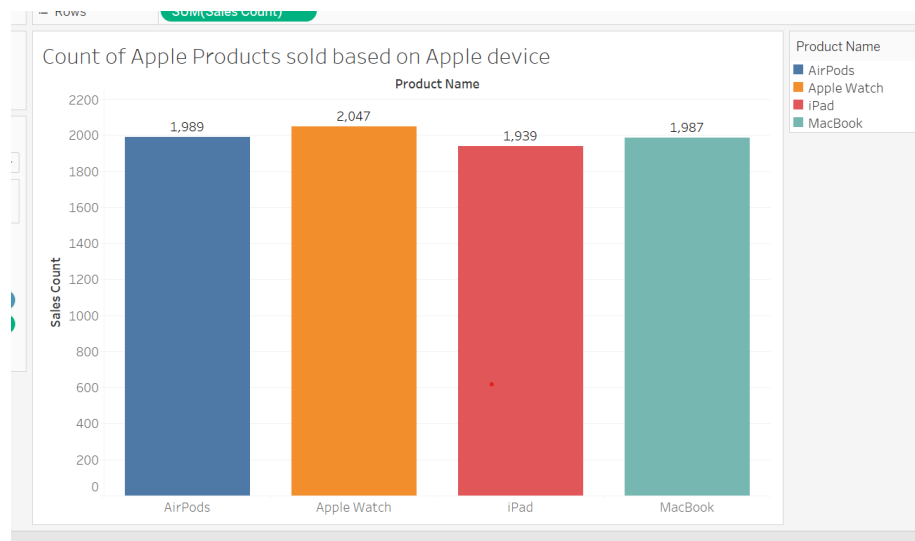


Figure 3: Count of Apple products sold based on Apple devices.

Observations:

Cross-Selling and UpSelling Opportunities

Identify popular product combinations purchased post-initial transactions. Use these insights to design recommendations for new customers based on historical patterns.

Map Subsequent Purchases

For each customer, list products purchased after the first transaction. Use aggregation techniques to compile a list of products bought per customer. Maintain the sequence of purchases to analyse patterns over time.

Analyze Buying Patterns

Product Frequency: Identify the most commonly purchased products after initial transactions. Time Between Purchases: Calculate the time gap between initial and subsequent purchases to analyse buying intervals.

7.4 Goal 4: To identify and count the Apple Products by their product categories.

The goal of identifying count the Apple Products by their product categories to categorize the transactions or products. Aggregate the prices within each category and compute the average value. Return a summary showing the category and its corresponding average price for analysis or reporting.

Code snippet

```
[53]: # Query to count products by category
product_count_by_category = products_df.groupby("category").agg(count("product_id").alias("product_count"))
product_count_by_category.show()

+-----+-----+
| category|product_count|
+-----+-----+
| Accessory|      1978|
| Laptop|      2016|
| Tablet|      1980|
| Smartphone|    4026|
+-----+-----+
```

Count of Apple Products by their Product Category

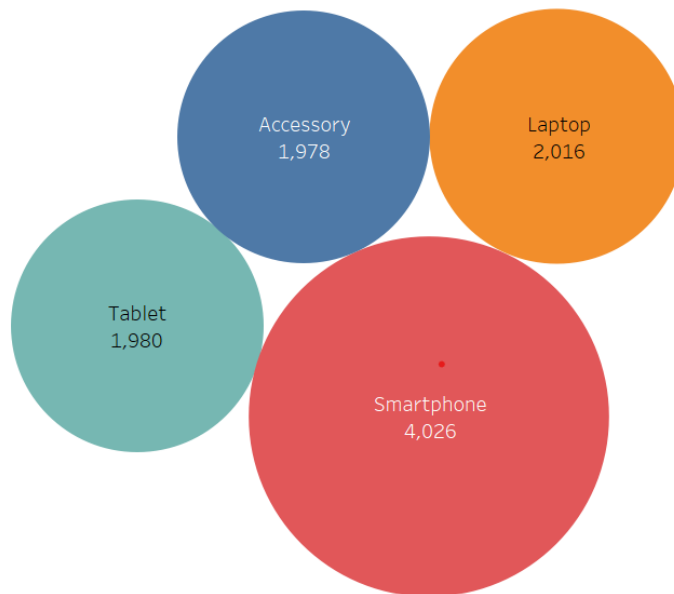


Figure 4: Count the Apple products by their product categories.

Observations:

Identify Sequential Purchases

For each customer, find the first purchase of an iPhone. Locate the next purchase of AirPods following the iPhone purchase. Remove customers who purchased only one of the two products. Ensure that AirPods purchases occur after iPhone purchases.

Filter Relevant Transactions

Extract transactions where the product is either an iPhone or AirPods. Group the data by customer ID and sort by purchase date to maintain the sequence of purchases.

Apply Insights

Offer AirPods promotions shortly after customers purchase iPhones, aligned with average delay patterns. Ensure AirPods stock availability aligns with pre-

dicted demand peaks.

7.5 Goal 5: finding the count of products purchased for each Apple product for a specific customer.

To find the count of each Apple product purchased by a specific customer, you can follow these steps that the Filter the transactions to include only Apple products (like iPhone, iPad, MacBook, etc.). Group the data by both customer Id and product name to count the occurrences of each product purchased by the customer. Aggregate the count of each product for each customer.

Code snippet

```
[61]: # Assume your DataFrame is named 'transactions'
# Step 1: Filter by customer_id
filtered_data = transactions_df.filter(col("customer_id") == 863)

# Step 2: Group by product_name and count
product_counts = (
    filtered_data.groupBy("product_name")
    .agg(count("*").alias("order_count"))
    .orderBy(col("order_count").desc())
)

# Step 3: Show results
product_counts.show()
```

product_name	order_count
iPhone	3
iPad	2
MacBook	2
Apple Watch	2
AirPods	2

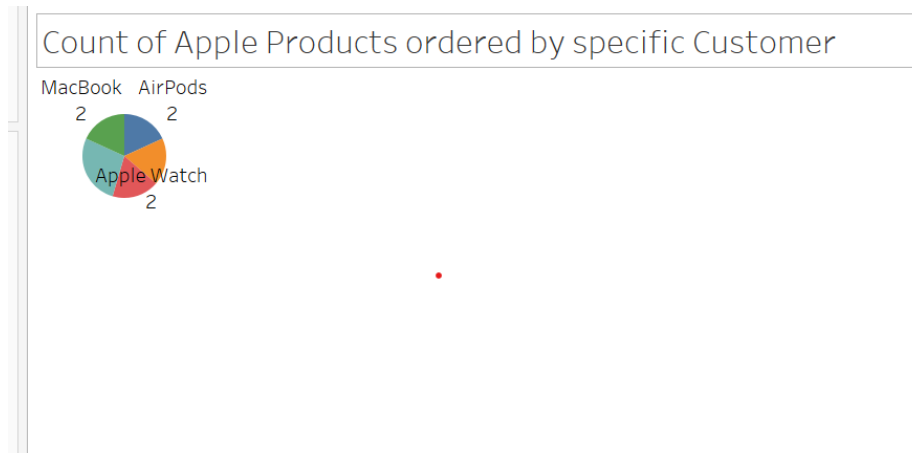


Figure 5: count of products purchased for each Apple product for a specific customer

Observations:

Actionable Outcomes

Design marketing campaigns or advertisements around the best-selling products to maximize exposure. Ensure the top three products in each category are well-stocked based on the demand forecast derived from the analysis. Utilize this data to understand customer preferences and buying behavior for more effective sales strategies.

Automate the Process

Implement automated workflows using tools like PySpark, SQL, or Databricks to regularly analyze sales data and update rankings. Schedule periodic reports to track how top-selling products change over time.

Analyze the Top Three Product

Review the products that rank first, second, and third in each category. Compare the total units sold or revenue generated by each of these products. Identify trends such as seasonality, pricing, or promotions that might influence product rankings.

7.6 Goal 6: To identify the group by customer details and count the number of transactions per customer.

The goal of group by customer details and count the number of transactions per customer generated by each product category, such as phones or accessories, to

identify key business drivers. This helps businesses understand which categories contribute most to overall revenue and identify areas of potential growth or focus.

Code snippet

```
[60]: # Perform an inner join between customer and transaction datasets on customer_id
customer_transactions = customers_df.join(
    transactions_df,
    on="customer_id",
    how="inner"
)

# Group by customer details and count the number of transactions per customer
customer_transaction_summary = customer_transactions.groupby(
    ["customer_id", "customer_name", "join_date", "location"]
).agg(
    count("transaction_id").alias("transaction_count")
)

customer_transaction_summary.show()

+-----+-----+-----+-----+-----+
|customer_id|customer_name|join_date|location|transaction_count|
+-----+-----+-----+-----+-----+
|596|Joseph Stevens|2/22/2019|Rachelshire, RI|13|
|673|Crystal Beltran|11/12/2024|East Kristinhaven...|14|
|586|Mr. Daniel King Jr.|11/14/2023|Wuchester, AL|12|
|182|Savannah Johns|7/21/2019|Tinafort, MD|4|
|217|Tonya Kemp|1/10/2021|Martinside, VT|11|
|311|Curtis Rice|10/22/2015|New Annatown, SD|8|
|381|Frank Franklin|3/9/2015|Lake Michaelchest...|15|
|495|Mario Maxwell|10/20/2023|Hermantown, WI|10|
|621|Amanda Hess|3/8/2016|Howardborough, NJ|10|
|758|Robert Jones|6/20/2015|Hamptonton, MO|8|
|792|Heather Kim|1/10/2021|Bassfurt, NE|12|
|860|Patricia Williams|6/22/2022|Lake Jason, MO|10|
|880|Casey Ferguson|4/7/2021|New Anthony, IN|6|
|193|Jason Khan|7/1/2023|Vazqueztown, TN|15|
|250|Shawn Harris|6/1/2021|North Sarahport, TX|9|
|303|Thomas Morrison|12/16/2021|Robertshire, CA|19|
|314|Anthony Werner|10/29/2020|East Kevinshire, NY|9|
|367|Lisa Caldwell|4/23/2022|Frankbury, SC|7|
|646|Amanda Wilson|6/7/2015|Gwendolynmouth, PA|14|
|721|Hunter Yang|5/21/2019|Burchshire, KY|8|
+-----+-----+-----+-----+-----+
only showing top 20 rows
```

Observations:

Seasonal Trends

Observe if certain categories perform better during specific seasons or promotional periods. For example, accessories might see a surge during holidays or back-to-school sales.

Revenue Growth or Decline

Track the revenue contribution over time to identify growing categories or those in decline. A decline in accessory sales, for example, could suggest a need for refreshed product offerings or marketing efforts.

Product Bundle Insights

If bundling products from different categories (e.g., a phone with accessories) significantly increases revenue in both categories, this could highlight successful cross-selling strategies.

7.7 Goal 7: Identifying customers who purchased both iPhones and AirPods.

The goal of identifying customers who purchased both iPhones and AirPods involves that the Filtering the transactions to isolate records for iPhones and AirPods. Grouping the transactions by customer to check if a customer has purchased both products. Using a join or set operation to identify customers who have at least one purchase of both iPhones and AirPods. Returning a list of customers who meet this criteria for targeted marketing or analysis.

Code snippet

```
[49]: # Goal 1: Identifying customers who purchased both iPhones and AirPods
      iphone_and_airpods = transactions_df.filter(
        (col('product_name').like('%iPhone%')) | (col('product_name').like('%AirPod%'))
      ).groupBy('customer_id').agg(
        F.countDistinct('product_name').alias('distinct_purchases')
      ).filter(
        col('distinct_purchases') == 2 # Ensure both iPhone and AirPods are purchased
      ).select('customer_id')

      # Join with customers_df to get customer names
      customer_info = customers_df.join(iphone_and_airpods, on='customer_id', how='inner') \
        .select('customer_id', 'customer_name')

      # Show the result
      customer_info.show()

+-----+-----+
|customer_id|customer_name|
+-----+-----+
|100|Elizabeth Daniels|
|101|Jeffrey Walters PhD|
|102|Daniel Young|
|103|Kevin Grant|
|104|Hunter Williamson|
|105|Gerald Moran|
|106|Toni Hill|
|107|Felicia Gordon|
|109|Kimberly Kaufman|
|111|Richard Smith|
|113|Carol Torres|
|114|Richard Golden|
|115|Angela Richards|
|116|Christopher White|
|117|Paul Ferguson|
|118|Robert Peck|
|119|Peter Zhang|
|120|Juan Lopez|
|121|Shirley Lucas|
|122|Tracy McGuire|
+-----+-----+
only showing top 20 rows
```

Observations:

High-Revenue Locations

Identify regions or cities that contribute the most to overall revenue, highlighting geographic areas where demand is highest. For example, a specific metropolitan area might account for a large portion of sales due to higher disposable income or market saturation.

Seasonal or Regional Trends:

Certain regions may show higher revenue during specific seasons or events. For instance, coastal areas might see a spike in sales during the summer, while colder regions could see higher sales of specific product categories.

Customer Demographics

Location-specific customer preferences may drive higher sales. For example, tech-savvy urban customers might prefer the latest electronics, while rural customers may prioritize practical goods.

8 The 5Vs of Apple Transactions Data Analysis

Apple Transactions Data Analysis revolves around extracting actionable insights from vast and varied datasets. By leveraging the 5Vs of data—Volume, Velocity, Variety, Veracity, and Value, we can systematically understand and predict user behaviors.

Volume:

Volume refers to the vast quantity of transaction data generated by Apple users, which includes data from online purchases, in-store transactions, app downloads, and service usage. Managing this large scale of data requires significant storage capacity and powerful processing tools. As the volume increases, organisations must leverage technologies like distributed computing and cloud storage to ensure smooth data handling. The goal is to process and analyse this massive amount of data efficiently to uncover valuable patterns and trends. In Apple's case, this can mean analysing millions of customer interactions to understand purchasing behaviour.

Velocity:

Velocity refers to the speed at which transaction data is generated, updated, and needs to be processed. For Apple, this includes real-time data streams from online purchases, app interactions, and customer feedback. Efficient handling of velocity ensures that businesses can quickly respond to customer actions or market conditions. The ability to process transactions as they happen enables Apple to tailor offers or manage stock levels dynamically.

Variety

Variety refers to the different forms of data involved in Apple transactions, such as structured data (purchase amounts, customer demographics) and unstructured data (social media interactions, reviews, or multimedia content). The

diverse nature of this data requires advanced techniques to integrate and analyze it effectively. For instance, unstructured data from customer feedback may offer insights that structured transactional data alone cannot. The variety in data also extends to different sales channels, including online stores, physical retail locations, and mobile applications.

Veracity

Veracity focuses on the accuracy, quality, and trustworthiness of the transaction data. Given the volume and variety of data, ensuring veracity is critical to ensure that the insights derived are reliable. Data inconsistencies, errors, duplicates, or biases can lead to incorrect conclusions and poor decision-making. Apple transactions, for example, may include errors due to incorrect input, missing values, or fraudulent activities that need to be flagged.

Value

Value refers to the insights and actionable knowledge derived from analyzing transaction data. The ultimate goal is to convert raw data into meaningful outcomes that benefit the business. For Apple, this could involve understanding which products are most popular, which regions drive the most revenue, or what features customers desire. By extracting value from large and complex datasets, businesses can make informed decisions to optimize sales, marketing, inventory, and customer service.

9 Conclusion

Apple's data analysis reveals a strong sales performance, driven primarily by the iPhone, with notable growth in wearables and services. Market trends indicate a successful alignment with the increasing demand for subscription-based services and wearables. Customer demographics show expanding global reach, particularly in emerging markets, with high loyalty among younger consumers. Product popularity is led by the iPhone, with significant traction for the Apple Watch and AirPods. Financial health is robust, with high profit margins and substantial cash reserves, supporting continued innovation and strategic investments. Competitive positioning remains strong, bolstered by a focus on privacy, security, and ecosystem integration. To sustain growth, Apple should continue investing in R&D, expand service offerings, penetrate emerging markets, strengthen sustainability initiatives, and enhance customer engagement. Overall, Apple's strategic focus and market adaptability position it well for future success.

10 References

<https://github.com/AjitUbbanapally/AppleSalesAnalysis.git>
<https://jupyter.org/>
<https://www.tableau.com/learn/get-started/prep>
<https://spark.apache.org/>