**“Customer Purchase Analytics for E-commerce Platform”**

**Project Report  
Submitted by**

**Jenifer B**

**BADM-WD-T-B3**

**Project Overview:**

The Customer Purchase Analytics project aims to provide an in-depth analysis of customer purchasing behaviour on an e-commerce platform. By leveraging various datasets, the project seeks to uncover patterns in customer purchases, identify high-value products, and evaluate customer retention strategies. The insights gained from this analysis will inform marketing initiatives, aid in customer

**Key Milestones and Activities:**

1. Data Cleaning and Preprocessing

- Conducted data validation to ensure the accuracy and consistency of the datasets.

- Handled missing values and outliers through various imputation techniques and filtering methods.

- Standardized data formats to ensure uniformity across various data points (e.g., date formats, currency conversion).

2. Aggregation and Summary Statistics

- Generated summary statistics to quantify key metrics such as total sales, average transaction value, and customer count.

- Aggregate purchases by categories to identify trends and anomalies.

3. Data Joins and Merging

- Merged multiple datasets including customer profiles, transaction records, and product inventory to view customer behaviours and performance comprehensively.

- Ensured the integrity of the data through effective join techniques (inner join, outer join, etc.).

4. Window Functions and Advanced SQL Queries

- Utilized window functions to conduct operations over specified ranges of data, providing deeper insights into customer behaviours over time.

- Developed complex SQL queries to extract nuanced insights and facilitate advanced trend analyses.

5. Percentile Calculations and Ranking

- Computed percentiles to evaluate the distribution of purchase values across different customer segments and product categories.

- Ranked products and customers based on key performance metrics (e.g., frequency of purchases, average spend) to identify top performers.

6. Date Analysis and Time Series Analysis

- Conducted time series analysis to examine purchasing trends over time, identifying seasonal patterns and peak purchasing periods.

- Evaluated customer behaviour changes during promotional events and holiday seasons.

7. Advanced Data Visualization in Power BI

- Developed interactive dashboards in Power BI to visualize key metrics, making it easier for stakeholders to derive actionable insights.

- Implemented visual elements such as charts, graphs, and maps to effectively convey findings.

8. Business Insights Generation for E-commerce Strategy

- Synthesized analysis results to generate actionable business insights.

- Provided recommendations for targeted marketing strategies, customer segmentation, and inventory management based on the findings.

**Data Analytics Tasks:**

**#A. Basic Aggregation:**

1. **Find the total number of purchases per product per year and list down the top-selling products every year**

Select \* from products\_dataset;

Select \* from purchase\_history\_dataset;

Select \* from purchase\_history\_dataset where extract(year from purchase\_date)=2023;

SELECT

YEAR(purchase\_date) AS purchase\_year,

product\_id,

SUM(quantity) AS total\_amount

FROM

purchase\_history\_dataset

GROUP BY

YEAR(purchase\_date),

product\_id

ORDER BY

purchase\_year DESC,

total\_amount DESC;

1. **Calculate the average quantity purchased per product and list down the top 5 products with high average quantity purchased**

SELECT

product\_id,

AVG(quantity) AS avg\_quantity

FROM

purchase\_history\_dataset

GROUP BY

product\_id

ORDER BY

avg\_quantity DESC

LIMIT 5;

**#B. Join Operations:**

1. **Join purchase history with products dataset to get the product name for each purchase.**

SELECT

purchase\_history\_dataset.purchase\_id,

purchase\_history\_dataset.product\_id,

purchase\_history\_dataset.purchase\_date,

purchase\_history\_dataset.quantity,

products\_dataset.product\_name

FROM

purchase\_history\_dataset

JOIN

products\_dataset

ON

purchase\_history\_dataset.product\_id = products\_dataset.product\_id;

1. **Join purchase history with customer profile dataset to include customer information for each purchase and list the top 5 customers with high purchases**

SELECT

customer\_profile\_dataset.customer\_id,

customer\_profile\_dataset.first\_name,

customer\_profile\_dataset.email,

SUM(purchase\_history\_dataset.quantity) AS total\_purchases

FROM

purchase\_history\_dataset

JOIN

customer\_profile\_dataset

ON

purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

GROUP BY

customer\_profile\_dataset.customer\_id, customer\_profile\_dataset.first\_name, customer\_profile\_dataset.email

ORDER BY

total\_purchases DESC

LIMIT 5;

**#C. Window Functions:**

1. **Find the cumulative sum of purchases for each product category over year.**

SELECT

YEAR(purchase\_history\_dataset.purchase\_date) AS purchase\_year,

products\_dataset.category,

SUM(purchase\_history\_dataset.quantity) AS total\_purchases,

SUM(SUM(purchase\_history\_dataset.quantity)) OVER (PARTITION BY products\_dataset.category ORDER BY YEAR(purchase\_history\_dataset.purchase\_date)) AS cumulative\_purchases

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON products\_dataset.product\_id = products\_dataset.product\_id

GROUP BY

purchase\_year, products\_dataset.category

ORDER BY

purchase\_year, products\_dataset.category;

**#D. Rank and Dense Rank:**

1. **Rank customers based on their total expenditure.**

SELECT

customer\_profile\_dataset.customer\_id,

customer\_profile\_dataset.first\_name,

customer\_profile\_dataset.email,

SUM(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) AS total\_expenditure,

RANK() OVER (ORDER BY SUM(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) DESC) AS expenditure\_rank

FROM

purchase\_history\_dataset

JOIN

customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

GROUP BY

customer\_profile\_dataset.customer\_id, customer\_profile\_dataset.first\_name, customer\_profile\_dataset.email

ORDER BY

expenditure\_rank;

1. **Identify the top 10 customers by purchase frequency using DENSE\_RANK().**

SELECT

customer\_profile\_dataset.customer\_id,

customer\_profile\_dataset.first\_name,

COUNT(purchase\_history\_dataset.purchase\_id) AS purchase\_frequency,

DENSE\_RANK() OVER (ORDER BY COUNT(purchase\_history\_dataset.purchase\_id) DESC) AS Rank\_no

FROM

purchase\_history\_dataset

JOIN

customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

GROUP BY

customer\_profile\_dataset.customer\_id, customer\_profile\_dataset.first\_name

ORDER BY

Rank\_no

LIMIT 10;

**#E. Percentiles:**

1. **Calculate the 25th, 50th (median), and 75th percentiles of total purchase amounts for each customer segment.**

WITH total\_purchases AS (

SELECT

customer\_profile\_dataset.customer\_id,

purchase\_history\_dataset.total\_amount

FROM

purchase\_history\_dataset

JOIN

customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

),

ranked\_purchases AS (

SELECT

customer\_id,

total\_amount,

NTILE(100) OVER (PARTITION BY customer\_id ORDER BY total\_amount) AS percentile\_rank

FROM

total\_purchases

)

SELECT

customer\_id,

MAX(CASE WHEN percentile\_rank <= 25 THEN total\_amount END) AS percentile\_25,

MAX(CASE WHEN percentile\_rank <= 50 THEN total\_amount END) AS percentile\_50,

MAX(CASE WHEN percentile\_rank <= 75 THEN total\_amount END) AS percentile\_75

FROM

ranked\_purchases

GROUP BY

customer\_id;

**#F. Median Calculation:**

**1.Compute the median purchase amount per product category.**

WITH total\_purchase\_amounts AS (

SELECT

purchase\_history\_dataset.purchase\_id,

purchase\_history\_dataset.product\_id,

purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount AS total\_purchase\_amount

FROM

purchase\_history\_dataset

),

ranked\_purchase\_amounts AS (

SELECT

t.total\_purchase\_amount,

products\_dataset.category,

ROW\_NUMBER() OVER (PARTITION BY products\_dataset.category ORDER BY t.total\_purchase\_amount) AS row\_num,

COUNT(\*) OVER (PARTITION BY products\_dataset.category) AS total\_rows

FROM

total\_purchase\_amounts t

JOIN

products\_dataset ON t.product\_id = products\_dataset.product\_id

)

SELECT

category,

AVG(total\_purchase\_amount) AS median\_purchase\_amount

FROM

ranked\_purchase\_amounts

WHERE

row\_num IN (FLOOR((total\_rows + 1) / 2), CEIL((total\_rows + 1) / 2))

GROUP BY

category;

**#G. Complex Aggregation:**

1. **Find the average, maximum, and minimum purchase value for each product type and customer age group.**

SELECT

products\_dataset.category,

CASE

WHEN customer\_profile\_dataset.date\_of\_birth < 18 THEN 'Under 18'

WHEN customer\_profile\_dataset.date\_of\_birth BETWEEN 18 AND 34 THEN '18-34'

WHEN customer\_profile\_dataset.date\_of\_birth BETWEEN 35 AND 54 THEN '35-54'

WHEN customer\_profile\_dataset.date\_of\_birth BETWEEN 55 AND 74 THEN '55-74'

ELSE '75+'

END AS age\_group,

AVG(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) AS avg\_purchase\_value,

MAX(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) AS max\_purchase\_value,

MIN(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) AS min\_purchase\_value

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

JOIN

customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

GROUP BY

products\_dataset.category,

CASE

WHEN customer\_profile\_dataset.date\_of\_birth < 18 THEN 'Under 18'

WHEN customer\_profile\_dataset.date\_of\_birth BETWEEN 18 AND 34 THEN '18-34'

WHEN customer\_profile\_dataset.date\_of\_birth BETWEEN 35 AND 54 THEN '35-54'

WHEN customer\_profile\_dataset.date\_of\_birth BETWEEN 55 AND 74 THEN '55-74'

ELSE '75+'

END

ORDER BY

products\_dataset.category, age\_group;

**#H. Grouping:**

1. **Group purchases by day of the week and find the average number of purchases made on each day.**

SELECT

DAYNAME(purchase\_day) AS day\_of\_week,

COUNT(\*) AS total\_days,

SUM(daily\_purchases) AS total\_purchases,

AVG(daily\_purchases) AS average\_purchases

FROM (

SELECT

DATE(purchase\_date) AS purchase\_day,

COUNT(\*) AS daily\_purchases

FROM purchase\_history\_dataset

GROUP BY purchase\_day

) AS daily\_stats

GROUP BY day\_of\_week

ORDER BY FIELD(day\_of\_week, 'Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday');

1. **Group customers by city and find the total number of purchases and total revenue generated.**

SELECT

customer\_profile\_dataset.city,

COUNT(purchase\_history\_dataset.purchase\_id) AS total\_purchases,

SUM(purchase\_history\_dataset.total\_amount) AS total\_revenue

FROM purchase\_history\_dataset

JOIN customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

GROUP BY customer\_profile\_dataset.city

ORDER BY total\_revenue DESC;

**#I. Case Statement:**

1. **Classify customers as “High-Spending,” “Medium-Spending,” or “Low-Spending” based on their total purchase amounts (using percentiles in multiples of 33).**

WITH customer\_totals AS (

SELECT

customer\_id,

SUM(total\_amount) AS total\_spent

FROM purchase\_history\_dataset

GROUP BY customer\_id

),

ranked\_customers AS (

SELECT

customer\_id,

total\_spent,

NTILE(100) OVER (ORDER BY total\_spent) AS percentile\_rank

FROM customer\_totals

)

SELECT

customer\_id,

total\_spent,

CASE

WHEN percentile\_rank <= 33 THEN 'Low-Spending'

WHEN percentile\_rank > 33 AND percentile\_rank <= 66 THEN 'Medium-Spending'

ELSE 'High-Spending'

END AS spending\_category

FROM ranked\_customers;

**#J. Join with Condition:**

1. **Join purchase history with customer profile dataset where the customer’s age is above 30, and display their purchase details.**

SELECT

purchase\_history\_dataset.purchase\_id,

purchase\_history\_dataset.purchase\_date,

purchase\_history\_dataset.total\_amount,

customer\_profile\_dataset.customer\_id,

customer\_profile\_dataset.first\_name,

customer\_profile\_dataset.date\_of\_birth,

customer\_profile\_dataset.city

FROM purchase\_history\_dataset

JOIN customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

WHERE customer\_profile\_dataset.date\_of\_birth > 30;

**#K. Top N Analysis:**

1. **Find the top 5 products contributing to the highest revenue.**

SELECT

products\_dataset.product\_id,

products\_dataset.product\_name,

SUM(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) AS total\_revenue

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

GROUP BY

products\_dataset.product\_id, products\_dataset.product\_name

ORDER BY

total\_revenue DESC

LIMIT 5;

1. **Identify the top 3 cities with the most number of unique customers.**

SELECT

customer\_profile\_dataset.city,

COUNT(DISTINCT customer\_profile\_dataset.customer\_id) AS unique\_customers

FROM

customer\_profile\_dataset

GROUP BY

customer\_profile\_dataset.city

ORDER BY

unique\_customers DESC

LIMIT 3;

**#L. Window Functions for Trend Analysis:**

1. **Create a 7-day moving average of total purchases per product and which customer had high 7 day MA**

WITH daily\_product\_sales AS (

SELECT

purchase\_history\_dataset.purchase\_date,

purchase\_history\_dataset.product\_id,

SUM(purchase\_history\_dataset.total\_amount) AS total\_purchase\_amount

FROM

purchase\_history\_dataset

GROUP BY

purchase\_history\_dataset.purchase\_date, purchase\_history\_dataset.product\_id

),

product\_sales\_with\_ma AS (

SELECT

d.purchase\_date,

d.product\_id,

d.total\_purchase\_amount,

AVG(d.total\_purchase\_amount) OVER (

PARTITION BY d.product\_id

ORDER BY d.purchase\_date

ROWS BETWEEN 6 PRECEDING AND CURRENT ROW

) AS seven\_day\_ma

FROM

daily\_product\_sales d

)

SELECT

p.product\_id,

p.purchase\_date,

p.seven\_day\_ma,

customer\_profile\_dataset.customer\_id,

customer\_profile\_dataset.first\_name,

customer\_profile\_dataset.last\_name

FROM

product\_sales\_with\_ma p

JOIN

purchase\_history\_dataset ON p.product\_id = purchase\_history\_dataset.product\_id AND p.purchase\_date = purchase\_history\_dataset.purchase\_date

JOIN

customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

ORDER BY

p.seven\_day\_ma DESC

LIMIT 1;

**#M. Nested Queries:**

1. **Find customers who have never purchased products from the top 5 most popular categories.**

WITH top\_categories AS (

SELECT

products\_dataset.category,

SUM(purchase\_history\_dataset.total\_amount) AS total\_revenue

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

GROUP BY

products\_dataset.category

ORDER BY

total\_revenue DESC

LIMIT 5

),

customers\_in\_top\_categories AS (

SELECT DISTINCT

customer\_profile\_dataset.customer\_id

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

JOIN

customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

WHERE

products\_dataset.category IN (SELECT category FROM top\_categories)

)

SELECT

customer\_profile\_dataset.customer\_id,

customer\_profile\_dataset.first\_name,

customer\_profile\_dataset.last\_name

FROM

customer\_profile\_dataset

LEFT JOIN

customers\_in\_top\_categories citc ON customer\_profile\_dataset.customer\_id = citc.customer\_id

WHERE

citc.customer\_id IS NULL;

1. **Identify products purchased only once in the entire dataset.**

SELECT

products\_dataset.product\_id,

products\_dataset.product\_name,

COUNT(purchase\_history\_dataset.purchase\_id) AS purchase\_count

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

GROUP BY

products\_dataset.product\_id, products\_dataset.product\_name

HAVING

purchase\_count = 1;

**#N. Date Analysis:**

1. **Identify the month with the highest total sales volume.**

SELECT

YEAR(purchase\_history\_dataset.purchase\_date) AS purchase\_year,

MONTH(purchase\_history\_dataset.purchase\_date) AS purchase\_month,

SUM(purchase\_history\_dataset.total\_amount) AS total\_sales\_volume

FROM

purchase\_history\_dataset

GROUP BY

purchase\_year, purchase\_month

ORDER BY

total\_sales\_volume DESC

LIMIT 1;

1. **Calculate the year-over-year growth of total sales.**

WITH yearly\_sales AS (

SELECT

YEAR(purchase\_history\_dataset.purchase\_date) AS purchase\_year,

SUM(purchase\_history\_dataset.total\_amount) AS total\_sales

FROM

purchase\_history\_dataset

GROUP BY

purchase\_year

)

SELECT

current\_year.purchase\_year AS year,

current\_year.total\_sales AS total\_sales\_current\_year,

previous\_year.total\_sales AS total\_sales\_previous\_year,

IFNULL(((current\_year.total\_sales - previous\_year.total\_sales) / previous\_year.total\_sales) \* 100, 0) AS yoy\_growth\_percentage

FROM

yearly\_sales current\_year

LEFT JOIN

yearly\_sales previous\_year ON current\_year.purchase\_year = previous\_year.purchase\_year + 1

ORDER BY

current\_year.purchase\_year DESC;

**#O. Join with Aggregation:**

1. **Join all three datasets to find the total revenue per product per customer.**

SELECT

customer\_profile\_dataset.customer\_id,

customer\_profile\_dataset.first\_name,

customer\_profile\_dataset.last\_name,

products\_dataset.product\_id,

products\_dataset.product\_name,

SUM(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) AS total\_revenue\_per\_product\_per\_customer

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

JOIN

customer\_profile\_dataset ON purchase\_history\_dataset.customer\_id = customer\_profile\_dataset.customer\_id

GROUP BY

customer\_profile\_dataset.customer\_id, customer\_profile\_dataset.first\_name, customer\_profile\_dataset.last\_name, products\_dataset.product\_id, products\_dataset.product\_name

ORDER BY

customer\_profile\_dataset.customer\_id, products\_dataset.product\_id;

**#P. Customer Retention:**

1. **Find the percentage of repeat customers in the dataset.**

WITH customer\_purchase\_count AS (

SELECT

customer\_id,

COUNT(DISTINCT purchase\_id) AS purchase\_count

FROM

purchase\_history\_dataset

GROUP BY

customer\_id

)

SELECT

(COUNT(CASE WHEN purchase\_count > 1 THEN 1 END) / COUNT(\*)) \* 100 AS repeat\_customers\_percentage

FROM

customer\_purchase\_count;

1. **Calculate the average number of days between purchases for repeat customers.**

WITH customer\_purchases AS (

SELECT

customer\_id,

purchase\_date,

ROW\_NUMBER() OVER (PARTITION BY customer\_id ORDER BY purchase\_date) AS purchase\_rank

FROM

purchase\_history\_dataset

),

purchase\_differences AS (

SELECT

cp1.customer\_id,

DATEDIFF(cp1.purchase\_date, cp2.purchase\_date) AS days\_between\_purchases

FROM

customer\_purchases cp1

JOIN

customer\_purchases cp2 ON cp1.customer\_id = cp2.customer\_id

WHERE

cp1.purchase\_rank > cp2.purchase\_rank

)

SELECT

AVG(days\_between\_purchases) AS avg\_days\_between\_purchases

FROM

purchase\_differences;

**#Q. Time Series Analysis:**

1. **Find the products with the highest growth in purchase frequency over time (average purchase YOY).**

WITH product\_yearly\_sales AS (

SELECT

products\_dataset.product\_id,

YEAR(purchase\_history\_dataset.purchase\_date) AS purchase\_year,

COUNT(purchase\_history\_dataset.purchase\_id) AS purchase\_count

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

GROUP BY

products\_dataset.product\_id, purchase\_year

),

product\_growth AS (

SELECT

current\_year.product\_id,

current\_year.purchase\_year,

current\_year.purchase\_count AS current\_year\_sales,

previous\_year.purchase\_count AS previous\_year\_sales,

IFNULL(((current\_year.purchase\_count - previous\_year.purchase\_count) / previous\_year.purchase\_count) \* 100, 0) AS yoy\_growth\_percentage

FROM

product\_yearly\_sales current\_year

LEFT JOIN

product\_yearly\_sales previous\_year

ON current\_year.product\_id = previous\_year.product\_id

AND current\_year.purchase\_year = previous\_year.purchase\_year + 1

)

SELECT

product\_id,

AVG(yoy\_growth\_percentage) AS avg\_yoy\_growth

FROM

product\_growth

GROUP BY

product\_id

ORDER BY

avg\_yoy\_growth DESC

LIMIT 5;

**#R. Subqueries:**

1. **Identify customers whose total expenditure is above the average expenditure of all customers.**

WITH customer\_expenditures AS (

SELECT

customer\_id,

SUM(total\_amount) AS total\_expenditure

FROM

purchase\_history\_dataset

GROUP BY

customer\_id

),

average\_expenditure AS (

SELECT

AVG(total\_expenditure) AS avg\_expenditure

FROM

customer\_expenditures

)

SELECT

ce.customer\_id,

ce.total\_expenditure

FROM

customer\_expenditures ce

JOIN

average\_expenditure ae ON ce.total\_expenditure > ae.avg\_expenditure;

1. **Find products with a purchase amount higher than the average purchase amount across all products.**

WITH product\_purchase\_amounts AS (

SELECT

products\_dataset.product\_id,

products\_dataset.product\_name,

SUM(purchase\_history\_dataset.quantity \* purchase\_history\_dataset.total\_amount) AS total\_purchase\_amount

FROM

purchase\_history\_dataset

JOIN

products\_dataset ON purchase\_history\_dataset.product\_id = products\_dataset.product\_id

GROUP BY

products\_dataset.product\_id, products\_dataset.product\_name

),

average\_purchase\_amount AS (

SELECT

AVG(total\_purchase\_amount) AS avg\_purchase\_amount

FROM

product\_purchase\_amounts

)

SELECT

p.product\_id,

p.product\_name,

p.total\_purchase\_amount

FROM

product\_purchase\_amounts p

JOIN

average\_purchase\_amount a ON p.total\_purchase\_amount > a.avg\_purchase\_amount;

**#S. Correlated Subqueries:**

1. **Find the purchase dates where a customer made a purchase larger than their average purchase amount.**

WITH customer\_average\_purchase AS (

SELECT

customer\_id,

AVG(total\_amount) AS avg\_purchase\_amount

FROM

purchase\_history\_dataset

GROUP BY

customer\_id

)

SELECT

purchase\_history\_dataset.customer\_id,

purchase\_history\_dataset.purchase\_date,

purchase\_history\_dataset.total\_amount

FROM

purchase\_history\_dataset

JOIN

customer\_average\_purchase cap ON purchase\_history\_dataset.customer\_id = cap.customer\_id

WHERE

purchase\_history\_dataset.total\_amount > cap.avg\_purchase\_amount;

1. **List all products that were purchased more frequently than the average frequency of products in their category.**

WITH product\_purchase\_frequency AS (

SELECT

p.product\_id,

p.product\_name,

p.category,

COUNT(pu.purchase\_id) AS purchase\_frequency

FROM

purchase\_history\_dataset pu

JOIN

products\_dataset p ON pu.product\_id = p.product\_id

GROUP BY

p.product\_id, p.product\_name, p.category

),

category\_average\_frequency AS (

SELECT

category,

AVG(purchase\_frequency) AS avg\_frequency

FROM

product\_purchase\_frequency

GROUP BY

category

)

SELECT

ppf.product\_id,

ppf.product\_name,

ppf.category,

ppf.purchase\_frequency

FROM

product\_purchase\_frequency ppf

JOIN

category\_average\_frequency caf ON ppf.category = caf.category

WHERE

ppf.purchase\_frequency > caf.avg\_frequency;

**#T. Date Functions:**

1. **Extract the day, month, and year from the purchase date, and group total purchases by month across all years**

SELECT

YEAR(purchase\_date) AS purchase\_year,

MONTH(purchase\_date) AS purchase\_month,

COUNT(\*) AS total\_purchases

FROM

purchase\_history\_dataset

GROUP BY

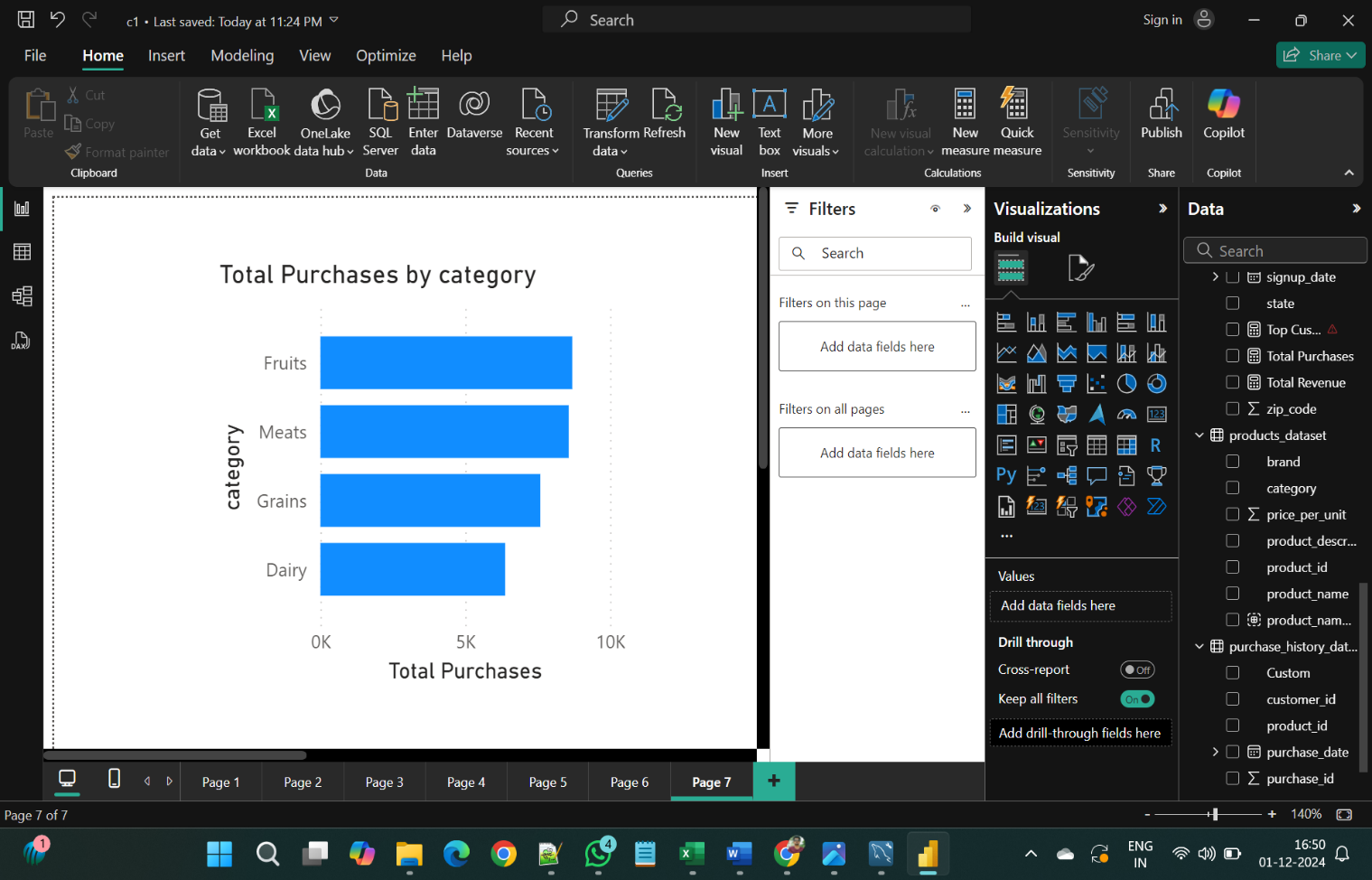
purchase\_year, purchase\_month

ORDER BY

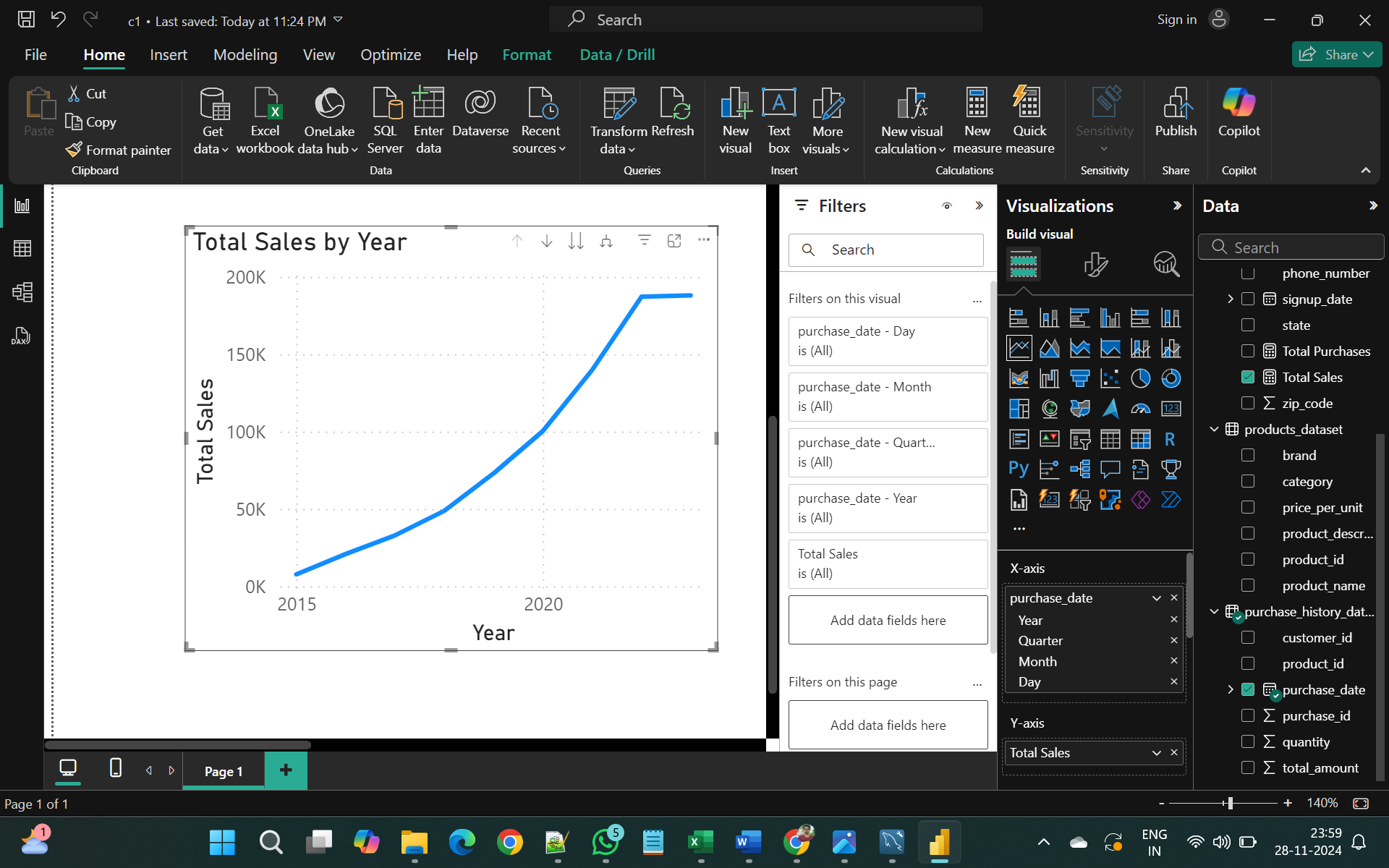
purchase\_year, purchase\_month;

**U. Visualization in Power BI:**

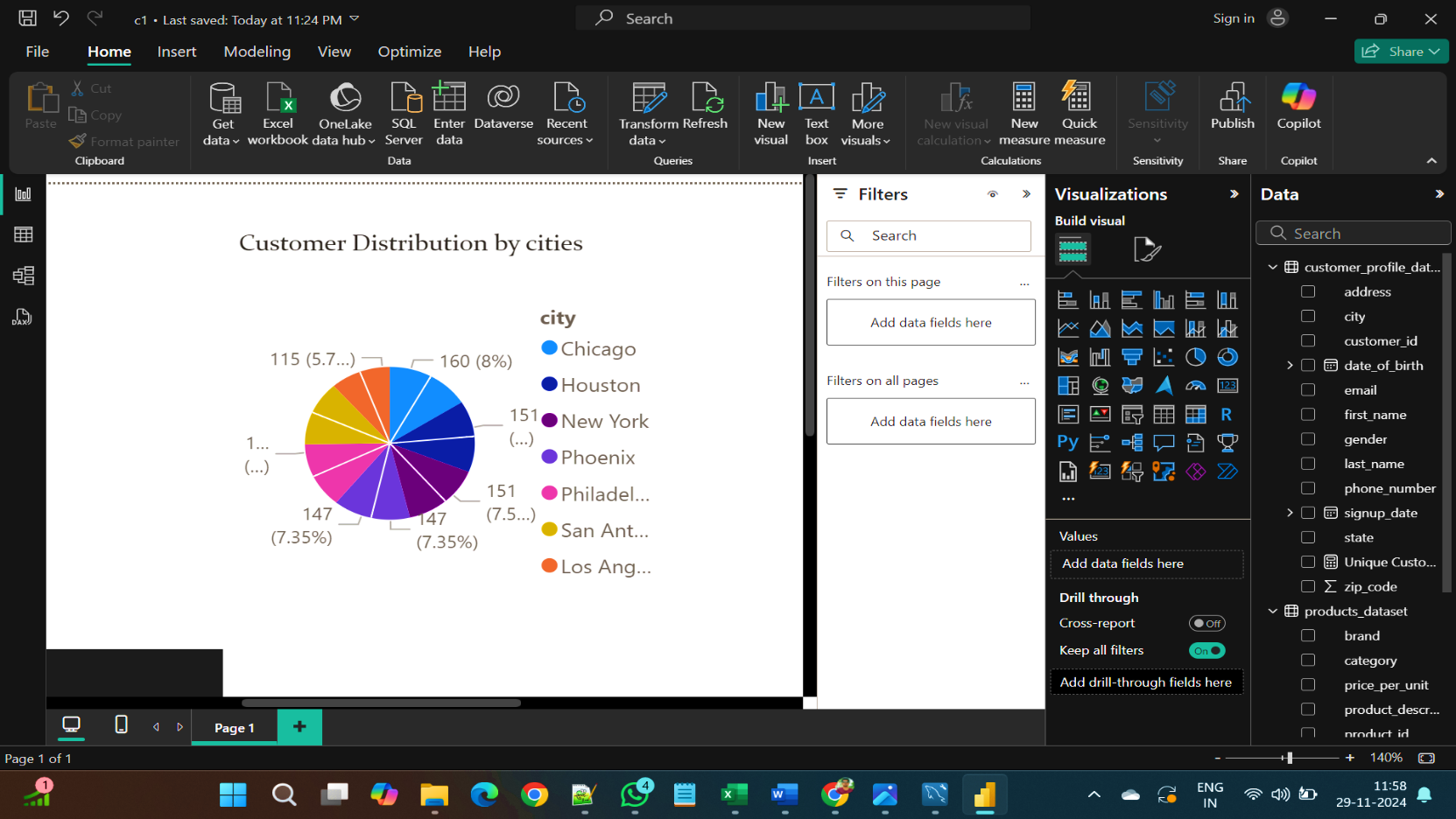
1. Create a bar chart showing the total purchases per product category.



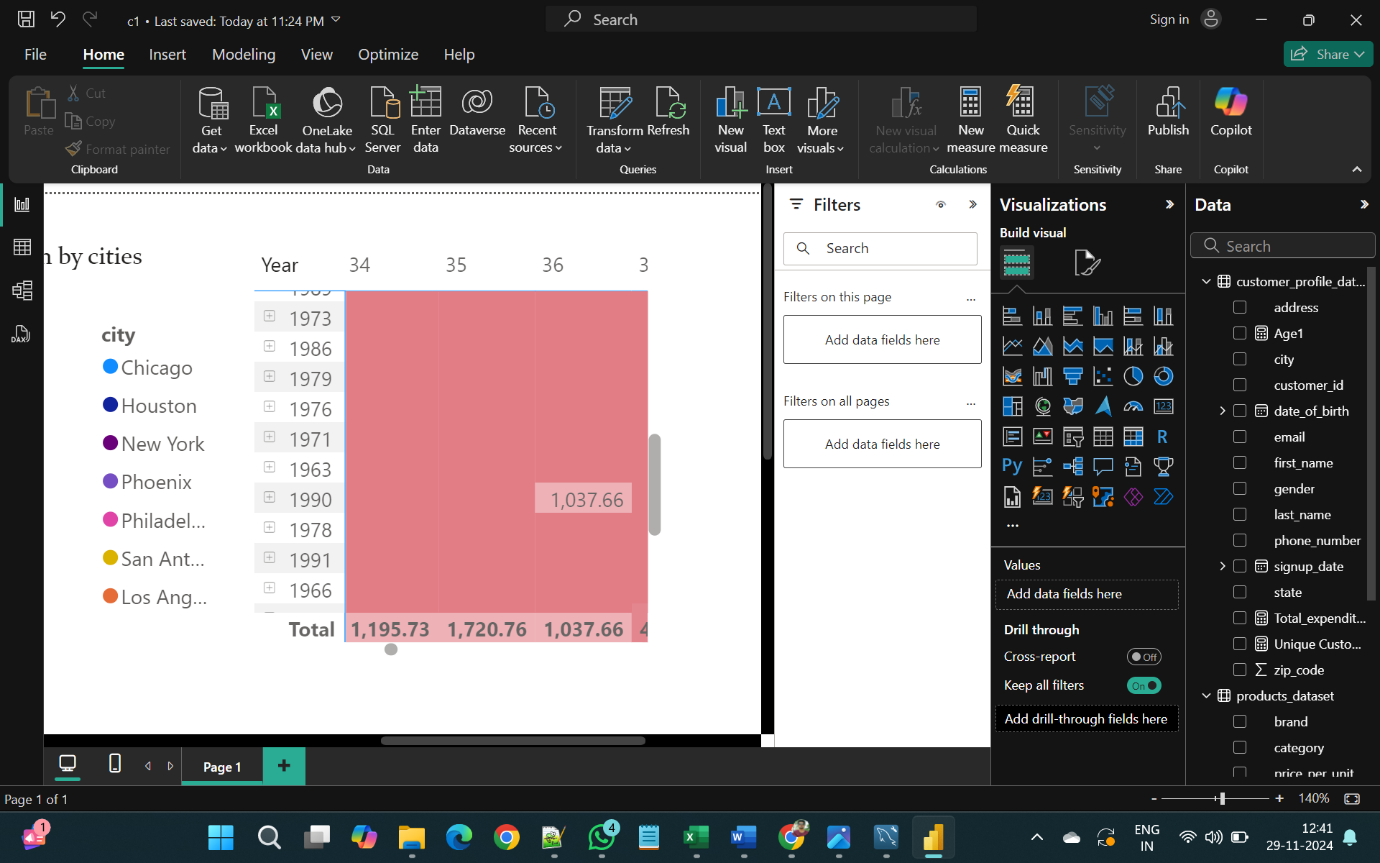
1. Build a line chart showing the trend of total sales over time.



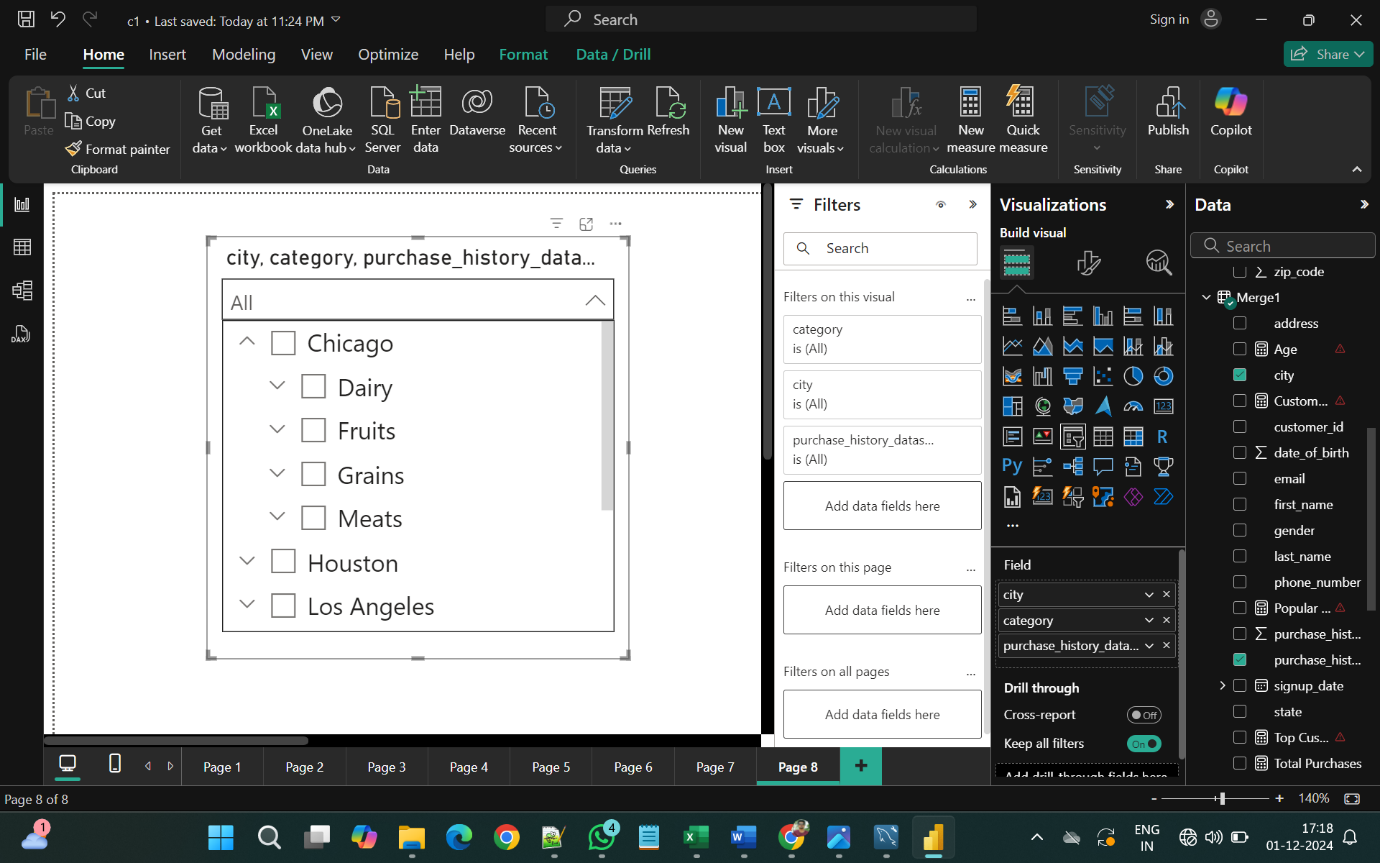
1. Design a pie chart representing the distribution of customers by city.



1. Implement a heat map displaying the correlation between customer age and total expenditure.

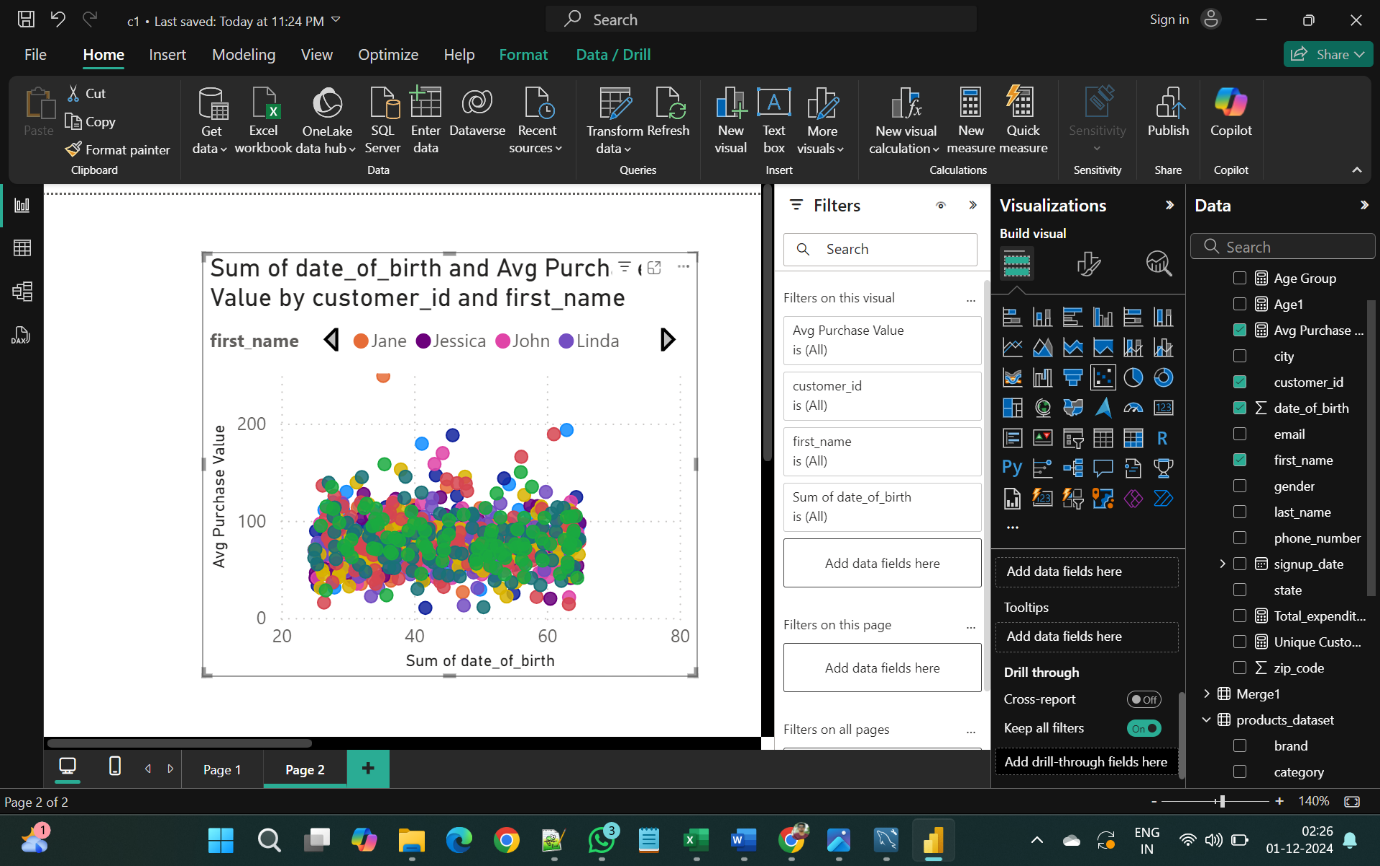


1. Use slicers to filter data by product category, customer age group, and city.



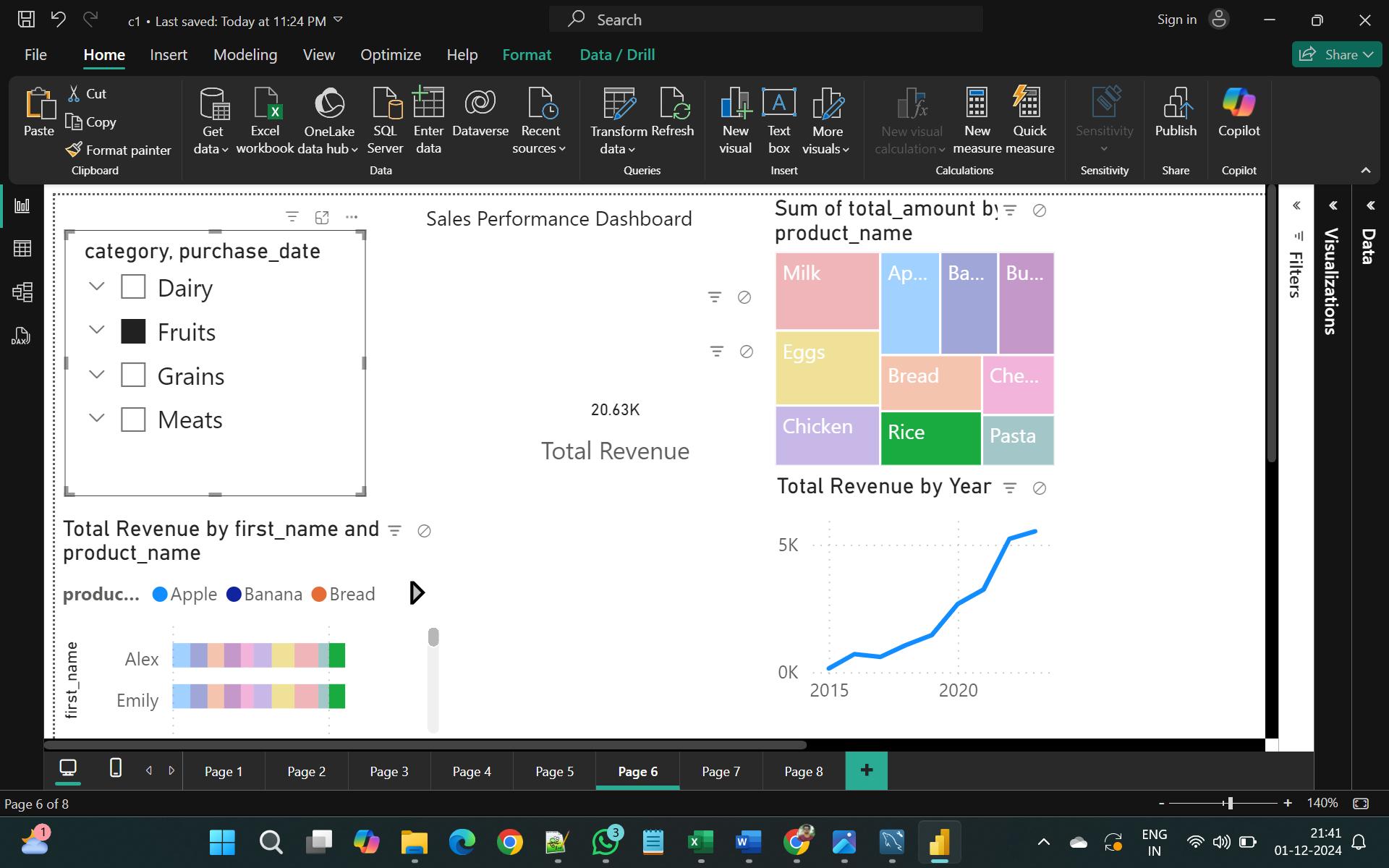
**V. Advanced Visuals:**

1. Implement a scatter plot to visualize the relationship between customer age and average purchase value.



**W. Power BI Dashboard:**

1. Combine multiple visualizations into a dashboard summarizing key performance indicators like total revenue, top customers, and popular products.



**Project Outcomes:**

The analysis yielded several significant insights that can guide the e-commerce platform's strategic decisions:

1. Gender Distribution of Purchases

- Analysis revealed a clear gender distribution in purchasing behaviour, enabling targeted marketing efforts to better engage specific demographics.

2. Average Purchase Value

- The report highlighted the average purchase value per product category and customer segment, helping identify potentially underperforming areas that might benefit from promotional activities.

3. Revenue Growth Analysis

- Identified top-performing products and cities, revealing key areas for expansion and marketing focus.

- Provided insights into revenue growth trends, indicating a positive trajectory alongside identifiable seasonal fluctuations.

4. Customer Retention Metrics

- Metrics on repeat purchases and frequency of customer transactions were analysed to assess customer loyalty and retention strategies.

- Insights into customer churn rates were also generated, suggesting areas for improvement in customer engagement practices.

5. Power BI Dashboard

- A comprehensive Power BI dashboard was created, summarizing key performance metrics and allowing stakeholders to visualize trends and insights in real time.

- The dashboard serves as an interactive tool for ongoing monitoring and decision-making.

**Conclusion:**

The Customer Purchase Analytics project successfully achieved its objectives, providing valuable insights into customer purchasing behaviours on the e-commerce platform. The findings have significant implications for marketing strategies, customer retention initiatives, and inventory management. Future work could involve implementing A/B testing for marketing campaigns and further refining customer segmentation strategies based on the analysis conducted.

**Improvement Steps:**

Based on the insights generated, the following next steps are recommended:

- Implement targeted marketing campaigns focusing on high-value customer segments and top-performing products.

- Develop a customer loyalty program aimed at enhancing retention based on frequency analysis results.

- Regularly update the Power BI dashboard to reflect real-time data for continuous monitoring of key performance indicators.