

# E\_COMMERCE ANALYSIS PROJECT

*Presented by: Team 6*

*23/5/2025*

**EYouth** ✨  
Data Analytics  
Bootcamp

# Table Of Content

1. Project Overview
2. Main Workflow
3. Database Overview
4. Cloud
5. SQL Server & SSMS Requirements
6. Dashboards Analysis
7. Machine Learning
8. Recommendations

# PROJECT OVERVIEW

**Our project focuses on analyzing a comprehensive e-commerce database that simulates real-world online retail operations. The database includes detailed and interconnected tables that cover essential components**

1

Deploy and secure the Azure SQL database

**Main  
Workflow**

# Azure Cloud for Data Analytics

## Benefits :

- Transform data into actionable insights with Azure's cloud platform.
- Enjoy scalable, secure, and cost-effective solutions built for modern analytics.
- Empower smarter, data-driven decisions to accelerate your business.

The screenshot shows the 'Create SQL Database' wizard in the Microsoft Azure portal. The 'Review + create' tab is active, showing the following details:

- Product details:** SQL database by Microsoft. Links for Terms of use and Privacy policy are provided.
- Estimated cost:** Storage cost 0.00 USD / month + Compute cost 0.000000 USD / vCore second.
- Terms:** A legal disclaimer stating that by clicking 'Create', the user agrees to the legal terms and privacy statement(s) associated with the Marketplace offering(s).
- Basics:** Subscription: Azure for Students; Resource group: E-YOUTH; Region: UAE North; Database name: ecommerce\_db.
- Cost summary:**
  - General Purpose (GP\_S\_Gen5\_2): Cost per GB (in USD) 0.00, Max storage selected (in GB) x 41.6.
  - First 32 GB storage free.
  - First 100,000 vCore seconds free.
  - Overage billing: Disabled.
  - ESTIMATED STORAGE COST / MONTH: 0.00 USD.
  - COMPUTE COST / VCORE SECOND: 0.000000 USD.

At the bottom, there is a 'Create' button, a '< Previous' button, and a link to 'Download a template for automation'.



1

Deploy and secure the Azure SQL database

2

Use SSMS to connect to the Azure SQL database and run basic to advanced SQL queries

# Main Workflow


# SQL Server & SSMS Requirements

## Basics Queries

1

**CUSTOMER CHURN  
PREDICTION MODEL**

```
SELECT SUM(total_amount) AS total_revenue  
FROM Orders
```




	total_revenue
1	44943966.02

2

**List the top 5 best-selling products by quantity sold.**

Product Name	Total Quantity Sold
1. Caramelized onion and mushroom appetizer	201
2. House-owned potato and olive	203
3. Vegetable terrine with herb	262
4. Up-sized interactive time-table	245
5. House-owned independent day-warehouse	242



```
SELECT TOP 5  
p.name AS Product_Name,  
SUM(o.quantity) AS Total_Quantity_Sold  
FROM products p  
JOIN order_details od ON p.id = od.product_id  
GROUP BY p.name  
ORDER BY Total_Quantity_Sold DESC
```

3

**Identify customers with the highest number of orders.**

```
SELECT  
c.first_name + ' ' + c.last_name AS Customer_Name,  
COUNT(o.id) AS Total_Orders  
FROM customers c  
JOIN orders o ON c.id = o.customer_id  
WHERE o.status != 'cancelled'  
GROUP BY c.first_name, c.last_name  
ORDER BY Total_Orders DESC
```



Customer Name	Total Orders
1. John Doe	10
2. Jane Smith	10
3. Bob Brown	10
4. Alice Green	10
5. Charlie White	10
6. David Black	10
7. Emily Gold	10
8. Frank Silver	10
9. Grace Bronze	10
10. Henry Copper	10
11. Irene Iron	10
12. Jack Steel	10
13. Karen Nickel	10
14. Leo Tin	10
15. Mia Lead	10
16. Noah Zinc	10
17. Olivia Cadmium	10
18. Peter Platinum	10
19. Quinn Silver	10
20. Ryan Gold	10
21. Sarah Copper	10
22. Thomas Iron	10
23. Ursula Steel	10
24. Victor Nickel	10
25. Wendy Tin	10

# SQL Server & SSMS Requirements

## Basics Queries

4

# Generate an alert for products with stock quantities below 20 units

5

**Determine the percentage of orders that used a discount.**

6

## Calculate the average rating for each product.

```

10:17
11: 44: 40 product_id,
12: 100 to 1000000, unit,
13:      priority,
14:      1000
15: with ('rows_per_iter' = 1000) 'look-back-start'
16: ('look-back-end' = 1000000)
17: with 2000000, 1000000
18: (look-back-end - 1)

```



	product_id	product_name	stock_quantity	store_id
1	1	Springer-Verlag Study Service	13	100-Stock Store
2	1	Springer-Verlag Study Service	17	100-Stock Store
3	1	Springer-Verlag Study Service	17	100-Stock Store
4	1	Springer-Verlag Study Service	20	100-Stock Store
5	1	Springer-Verlag Study Service	36	100-Stock Store
6	1	Springer-Verlag Study Service	36	100-Stock Store
7	1	Springer-Verlag Study Service	36	100-Stock Store
8	1	Springer-Verlag Study Service	36	100-Stock Store
9	1	Springer-Verlag Study Service	36	100-Stock Store
10	1	Springer-Verlag Study Service	36	100-Stock Store
11	1	Springer-Verlag Study Service	36	100-Stock Store
12	1	Springer-Verlag Study Service	36	100-Stock Store
13	1	Springer-Verlag Study Service	36	100-Stock Store
14	1	Springer-Verlag Study Service	36	100-Stock Store
15	1	Springer-Verlag Study Service	36	100-Stock Store
16	1	Springer-Verlag Study Service	36	100-Stock Store
17	1	Springer-Verlag Study Service	36	100-Stock Store
18	1	Springer-Verlag Study Service	36	100-Stock Store
19	1	Springer-Verlag Study Service	36	100-Stock Store
20	1	Springer-Verlag Study Service	36	100-Stock Store
21	1	Springer-Verlag Study Service	36	100-Stock Store
22	1	Springer-Verlag Study Service	36	100-Stock Store
23	1	Springer-Verlag Study Service	36	100-Stock Store
24	1	Springer-Verlag Study Service	36	100-Stock Store
25	1	Springer-Verlag Study Service	36	100-Stock Store
26	1	Springer-Verlag Study Service	36	100-Stock Store
27	1	Springer-Verlag Study Service	36	100-Stock Store
28	1	Springer-Verlag Study Service	36	100-Stock Store
29	1	Springer-Verlag Study Service	36	100-Stock Store
30	1	Springer-Verlag Study Service	36	100-Stock Store

```

[50:107]
      SELECT COUNT(*) AS cnt, SUM (
        [50:113] COUNT(*) FROM orders WHERE status = 'Cancelled') AS Cancelled, Usage_Percentage
FROM orders o
[50:119] ORDER BY cnt DESC, SUM (
[50:125] Usage_Percentage) DESC, product_id
[50:131] WHERE o.order_date BETWEEN d.start_date AND d.end_date
[50:137] AND o.status = 'Cancelled';

```



	Discount_Usage_Percentage
1	3.88

[illegible]



# SQL Server & SSMS Requirements

## Advanced Queries

1

Compute the 30-day customer retention rate after their first purchase

```
WITH customerfirst AS
(
    SELECT customer_id, MIN(order_date) AS firstorder
    FROM orders
    GROUP BY customer_id
),
Next30Days AS
(
    SELECT o.customer_id
    FROM orders o
    JOIN customerfirst cfp ON cfp.customer_id = o.customer_id
    WHERE o.order_date > cfp.firstorder
    AND o.order_date <= DATEADD(DAY, 30, cfp.firstorder)
)
SELECT
    CAST(COUNT(DISTINCT Next30Days.customer_id) * 100.0 /
    (SELECT COUNT(*) FROM customerfirst) AS DECIMAL(5,2)) AS RetentionRate
FROM Next30Days;
```



RetentionRate	
1	92.77

2

Recommend products frequently bought together with items in customer wishlists

```
SELECT
    w.customer_id,
    w.product_id AS wishlist_product,
    od.product_id AS bought_together_product,
    COUNT(*) AS frequency
FROM wishlists w
JOIN orders o ON w.customer_id = o.customer_id
JOIN order_details od ON o.id = od.order_id
WHERE od.product_id <> w.product_id
GROUP BY w.customer_id, w.product_id, od.product_id
ORDER BY frequency DESC;
```



customer_id	wishlist_product	bought_together_product	frequency
100	127	307	6
229	248	402	6
202	201	136	6
229	548	402	6
193	320	311	6
229	548	476	6
229	548	456	6
229	248	476	6
229	248	456	6
202	201	107	6
86	131	296	5
220	548	141	4
229	548	533	4
151	476	465	4
202	201	108	4
242	336	280	4
100	127	112	4
202	201	191	4
203	135	570	4
151	476	179	4
202	201	254	4
202	201	377	4
76	472	237	4
202	201	483	4
100	127	106	4

3

Track inventory turnover trends using a 30-day moving average

```
--Query to find the stock product in daily
with daily_sell as
(
    select o.order_date, od.product_id, od.quantity as quantity_sold,
    from order_details od
    join orders o on od.order_id = o.id
    where o.order_date <= GETDATE() - 30
    group by o.order_date, od.product_id
),
--Query to find the stock product in daily
with stock as
(
    select o.order_date, o.product_id, o.quantity as o_quantity,
    from orders o
    join products p on o.product_id = p.id
),
--Inventory turnover = quantity sold in day / current stock
avg_turnover as
(
    select order_date, product_id, quantity_sold, stock_quantity,
    (quantity_sold * 100.0 / stock_quantity) as inventory_turnover,
    ROW_NUMBER() OVER (PARTITION BY product_id ORDER BY order_date) as rn
)
select
    order_date,
    product_id,
    quantity_sold,
    stock_quantity,
    inventory_turnover,
    avg_turnover
from avg_turnover
order by
    product_id, order_date;
```



order_date	product_id	quantity_sold	stock_quantity	inventory_turnover	avg_turnover
2020-04-20 17:45:54.000	1	3	14	0.214285714285714	0.214285714285714
2020-04-21 09:40:40.000	5	3	28	0.107142857142857	0.107142857142857
2020-04-22 18:53:00.000	5	3	25	0.120000000000000	0.107142857142857
2020-04-23 18:51:00.000	7	3	26	0.115384615384615	0.107142857142857
2020-04-24 20:26:29.000	6	1	12	0.083333333333333	0.107142857142857
2020-04-25 18:43:21.000	14	6	25	0.24	0.24
2020-04-26 22:54:01.000	14	1	25	0.04	0.14
2020-04-27 22:54:01.000	14	1	25	0.04	0.14
2020-04-28 15:27:40.000	18	1	14	0.071428571428571	0.093750000000000
2020-04-29 02:04:01.000	17	3	10	0.3	0.1
2020-04-30 20:28:00.000	17	4	10	0.4	0.100000000000000
2020-04-30 18:54:22.000	18	4	28	0.142857142857143	0.142857142857143
2020-04-31 19:29:34.000	18	4	28	0.142857142857143	0.142857142857143
2020-04-30 18:54:22.000	21	5	28	0.178571428571429	0.178571428571429
2020-04-28 20:58:47.000	21	4	28	0.142857142857143	0.178571428571429
2020-04-23 18:43:21.000	22	2	17	0.117647058823529	0.117647058823529

# SQL Server & SSMS Requirements

## Advanced Queries

4

Identify customers who have purchased every product in a specific category

```
DECLARE @category_id INT = 13;

SELECT
    c.id AS customer_id,
    CONCAT(c.first_name, ' ', c.last_name) AS customer_name
FROM customers c
JOIN orders o ON c.id = o.customer_id
JOIN order_details od ON o.id = od.order_id
JOIN products p ON od.product_id = p.id
WHERE p.category_id = @category_id
GROUP BY c.id, c.first_name, c.last_name
HAVING COUNT(DISTINCT p.id) = (
    SELECT COUNT(*) FROM products WHERE category_id = @category_id
);
```



customer_id	customer_name
-------------	---------------

5

Find pairs of products commonly bought together in the same order.

```
SELECT
    p1.name AS Product_1,
    p2.name AS Product_2,
    COUNT(*) AS Products_Together
FROM
    order_details o1, o2
WHERE
    o1.order_id = o2.order_id
    AND o1.product_id < o2.product_id
    AND o1.quantity > 0
    AND o2.quantity > 0
GROUP BY
    p1.name, p2.name
ORDER BY
    Products_Together DESC;
```



Prod_Product	Second_Product	Products_Together
1. Stainless steel drink bottles	10. Re-engineered industrial aluminium	5
2. Distributed sound testing challenge	10. Re-engineered engineering standardisation	4
3. Right hand rotating benchmark	10. Re-engineered engineering standardisation	4
4. Vision-oriented Segmentation data warehouse	10. Quality-focused new centre display interface	4
5. Stand alone secondary research	10. Free mobile sample encoding	4
6. Processed mobile based interface	10. Profit focused bandwidth reduction protocol	4
7. Enterprise wide sample storage	10. Self encoding national warehouse	4
8. Total background network flow	10. Self encoding national warehouse	4
9. Secondary composite capacity	10. Self encoding system security core	4
10. Free sample policy implementation	10. Stand alone zero defect challenge	4
11. Fundamental incremental structure	10. Strategic horizontal model	4
12. Team-oriented experimental solution	10. Strategic interactive framework	4
13. Fully configurable value added Internet solution	10. Strategic horizontal model	4

6

Calculate the time taken to deliver orders in days.

```
SELECT
    O.customer_id AS Order_ID,
    O.order_date,
    sh.shipping_date,
    DATEDIFF(DAY, O.order_date, sh.shipping_date) AS Delivers
FROM
    orders O
LEFT JOIN
    shipping sh ON O.id = sh.order_id
WHERE
    O.status = 'delivered';
```



Order_ID	order_date	shipping_date	Delivers
1	2025-01-11 04:00:18.000	2025-01-13 04:00:18.000	1
2	2025-01-03 14:00:18.000	2025-01-04 14:00:18.000	1
3	2025-01-24 08:10:42.000	2025-01-25 08:10:42.000	1
4	2025-01-20 16:32:46.000	2025-01-21 16:32:46.000	1
5	2025-01-01 01:18:01.000	2025-01-02 01:18:01.000	1
6	2025-01-10 02:30:21.000	2025-01-11 02:30:21.000	1
7	2025-01-02 21:18:27.000	2025-01-03 21:18:27.000	1
8	2025-01-20 07:02:44.000	2025-01-21 07:02:44.000	1
9	2025-01-20 20:20:08.000	2025-01-21 20:20:08.000	1
10	2025-01-04 11:20:24.000	2025-01-05 11:20:24.000	1
11	2025-01-13 16:37:24.000	2025-01-14 16:37:24.000	1
12	2025-01-10 02:27:06.000	2025-01-11 02:27:06.000	1

1

Deploy and secure the Azure SQL database

2

Use SSMS to connect to the Azure SQL database and run basic to advanced SQL queries

3

Connect Power BI to Azure SQL using to build interactive dashboards

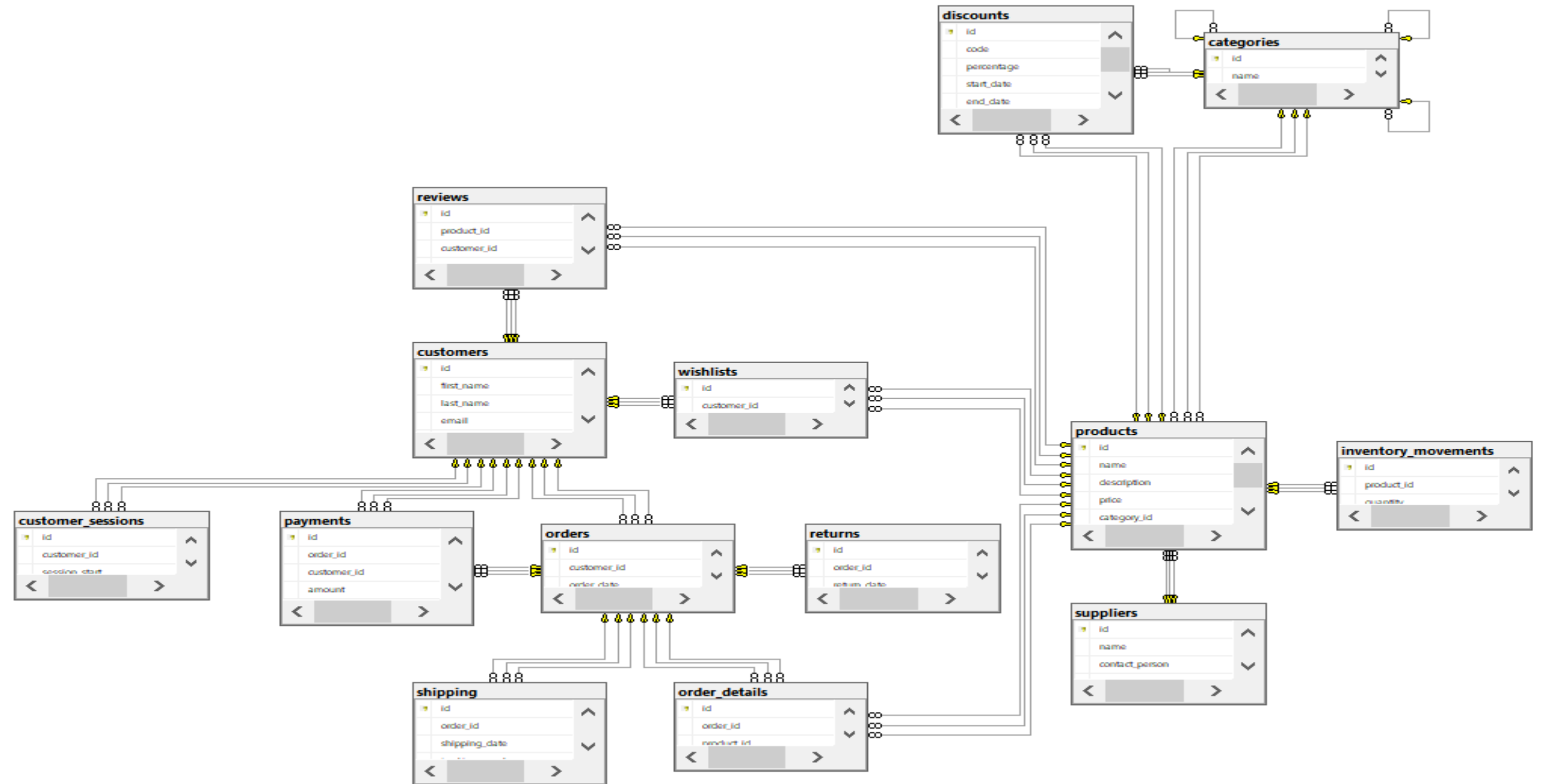
# Main Workflow

# Data Source

## Snowflake schema

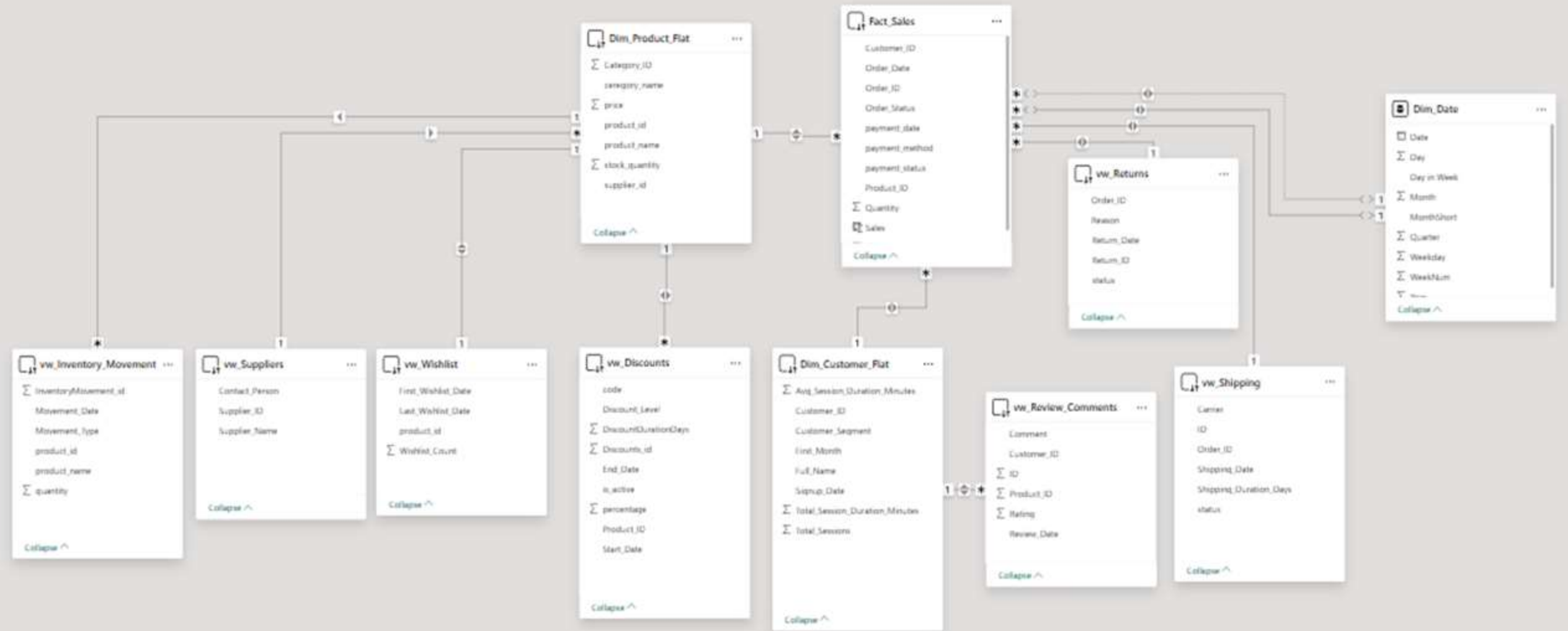
### Problem :

We encountered performance issues as it was heavy on the model and not optimal for reporting



# Data Source

**As a Solution :**  
the model was restructured  
into a **Star Schema** in  
Power BI, using SQL **Views**  
as the base for each  
dimension and fact table





# DASHBOARDS ANALYSIS



## SALES DASHBOARD

Provide an overview of sales trends by many Important Indicators.



## CUSTOMERS DASHBOARD

Track customer behavior and purchasing behavior to understand target audiences, analyze strengths and weaknesses to increase customer numbers and loyalty.



## PRODUCT DASHBOARD

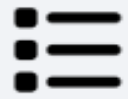
Analyze product performance with a focus on sales volume.



## RETURNS & CANCELLED DASHBOARD

Analyze all Returns Reasons, focus on Category, payment method any reason Leads to increased returns.

# Sales Analysis



## E-Commerce Analysis

Filter



Total Orders  
**10200**

PM 9670  
MOM **5.5%▲**



Potential Revenue  
**\$44.9M**

PM \$42.67M  
MOM **5.3%▲**



Orders Cancelled  
**2012**

PM 1918  
MOM **4.9%▲**



Lost Revenue  
**\$9.5M**

PM \$9M  
MOM **4.6%▲**



Net Revenue  
**\$35.4M**

PM \$33.53M  
MOM **5.5%▲**

### % For Discount Used

Unused 56.67%

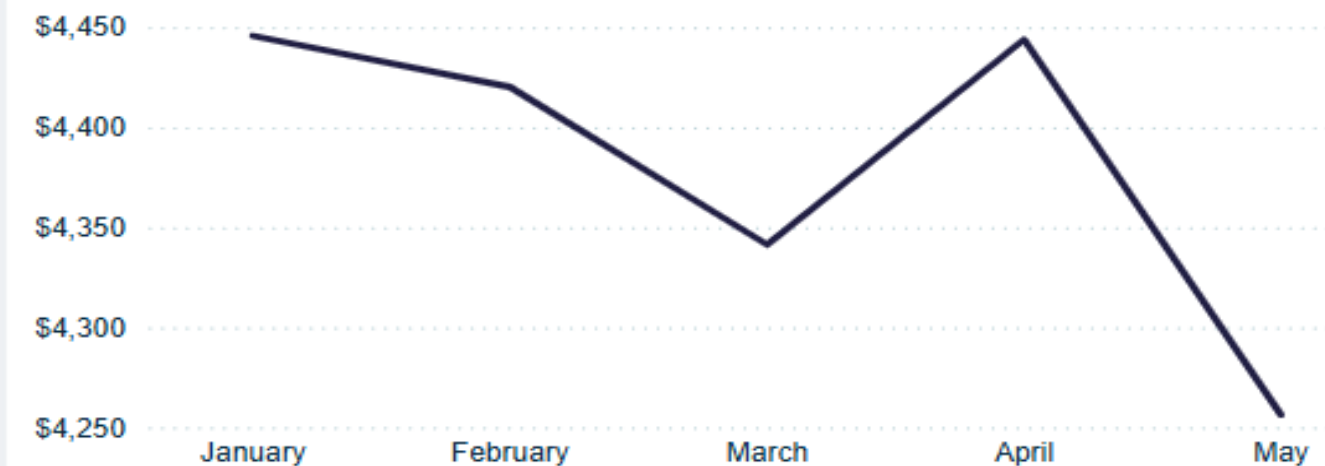
Used 43.33%

### % Total Orders By WeekDay and WeekEnd

Weekday 72.09%

Weekend 27.91%

### Avg Order Value By Month



### KPIs by Date

Net Revenue

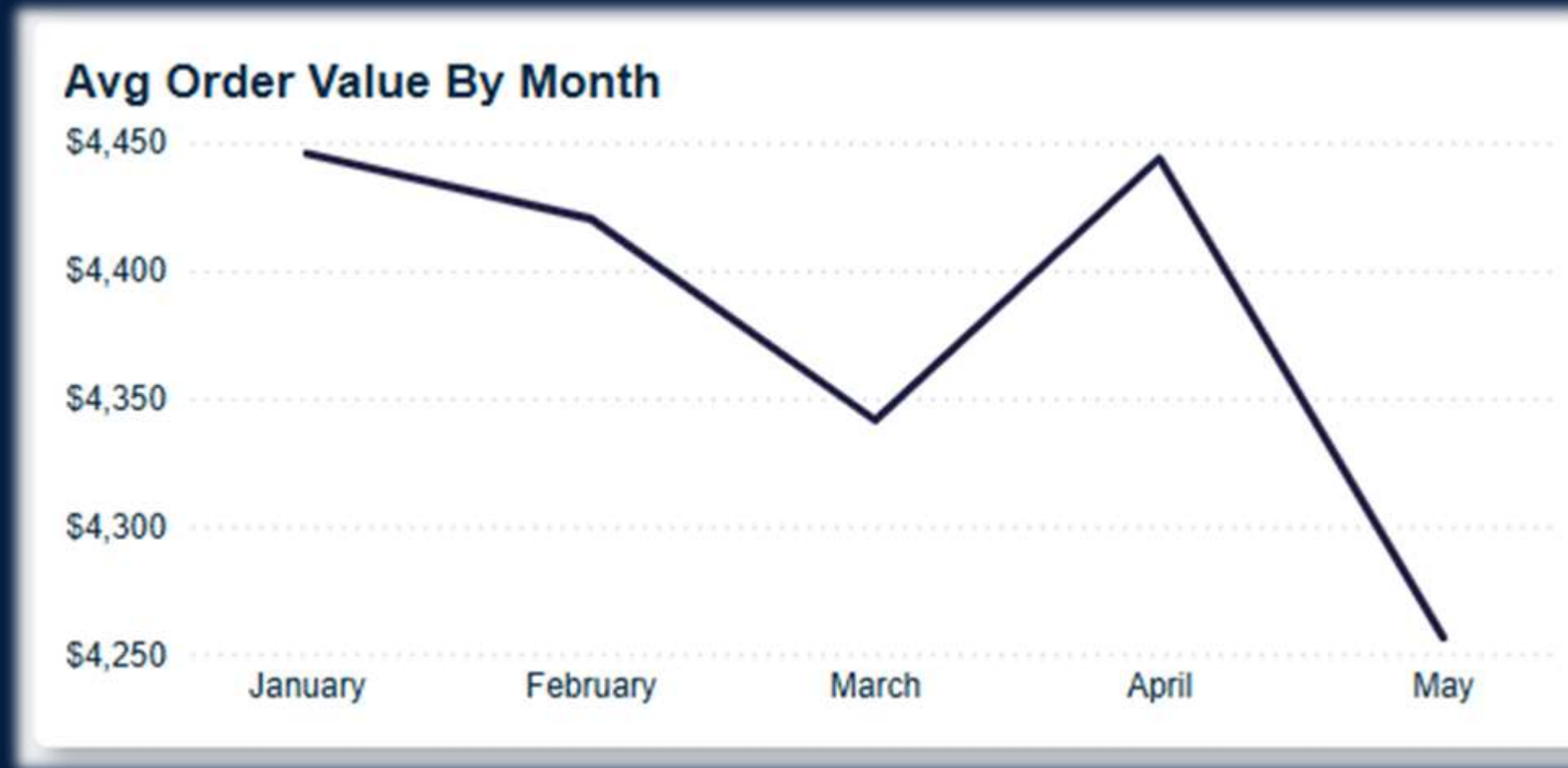
Total Orders

Total Customers

Total Quantity

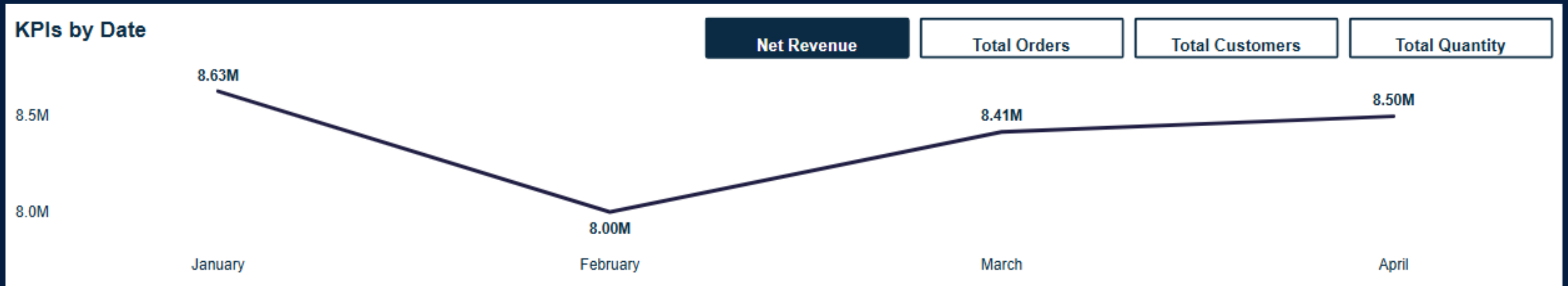


# Sales Analysis



- The Average Order Value was low in March, at \$4,300 compared to \$4,400 in the other months
- The best strategy here would be to offer discounts and promotions for orders above \$4,500 to encourage higher spending
- **Important note:** Data for May is incomplete, covering only the first 7 days , So we ignore

# Sales Analysis



The highest revenue was recorded in January , reaching approximately 8.63M , while February had the lowest at 8M

Overall , the revenue figures are relatively close throughout the period

# Customer Analysis



## Customer 360

Filter



Total Customers  
**250**



Avg Revenue Per Customer  
**\$142K**



Churn rate %  
**14.00%**



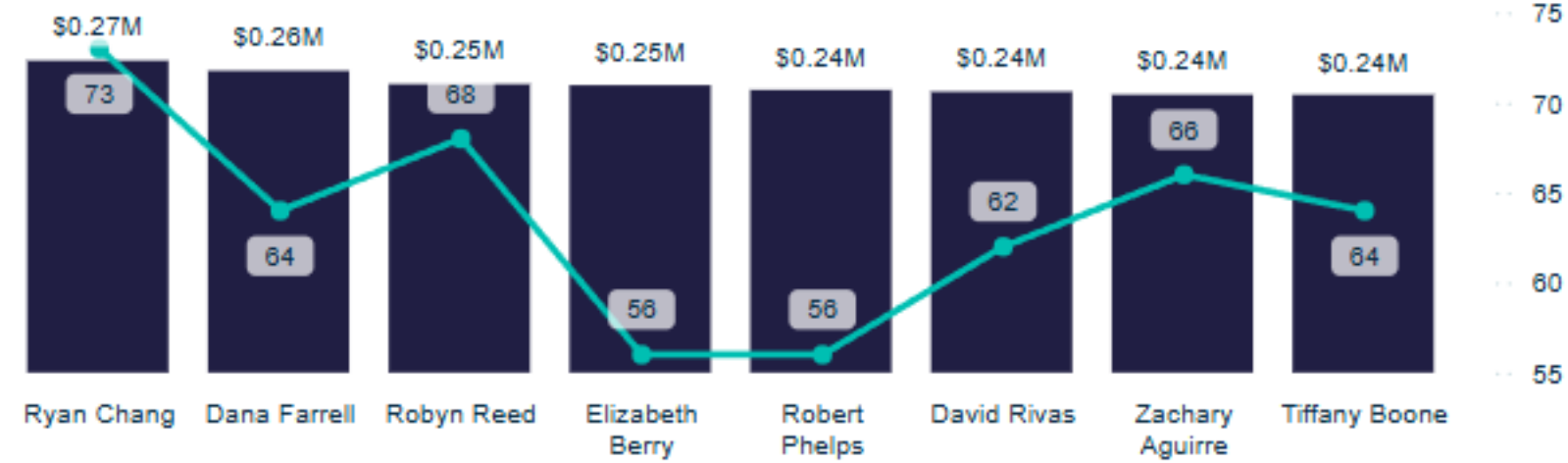
Retention Rate  
**86.00%**



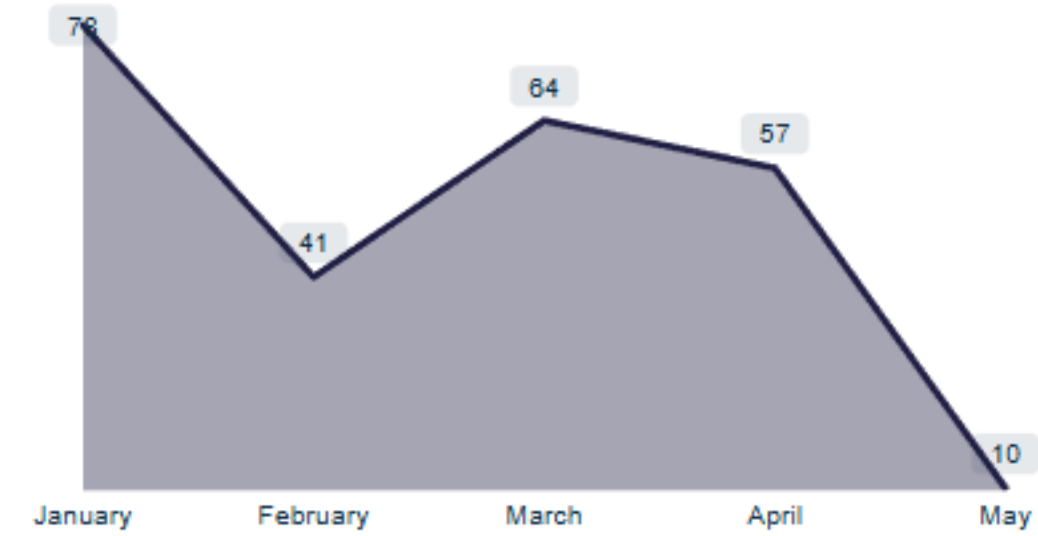
Avg Session Duration (Mins)  
**61.90 Min**

### Top 8 Customer By Net Revenue and Total Orders

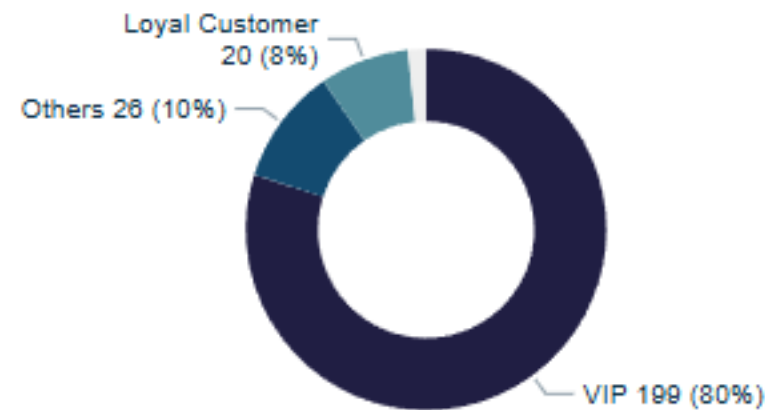
● Net Revenue ● Total Orders



### Number Of Sign Up By Month



### Customers By Segment



### Customers Table

Customers	Total Sessions	Monetary	Frequency	AOV	Total Quantity	RFM	Segment
Aaron Jones	44	\$161,823	44	\$4,378	421	455	VIP
Adrian Johnson	35	\$188,770	49	\$5,008	495	455	VIP
Alexis Knox	47	\$168,766	42	\$4,693	387	455	VIP
Amber Coleman	31	\$160,748	55	\$4,509	536	455	VIP
Amber French	13	\$221,467	56	\$5,135	556	455	VIP
Amy Hill	30	\$210,044	52	\$4,721	506	455	VIP
Andrea Macdonald	30	\$148,737	57	\$3,212	392	455	VIP



# Customer Analysis

Top 8 Customer By Net Revenue and Total Orders

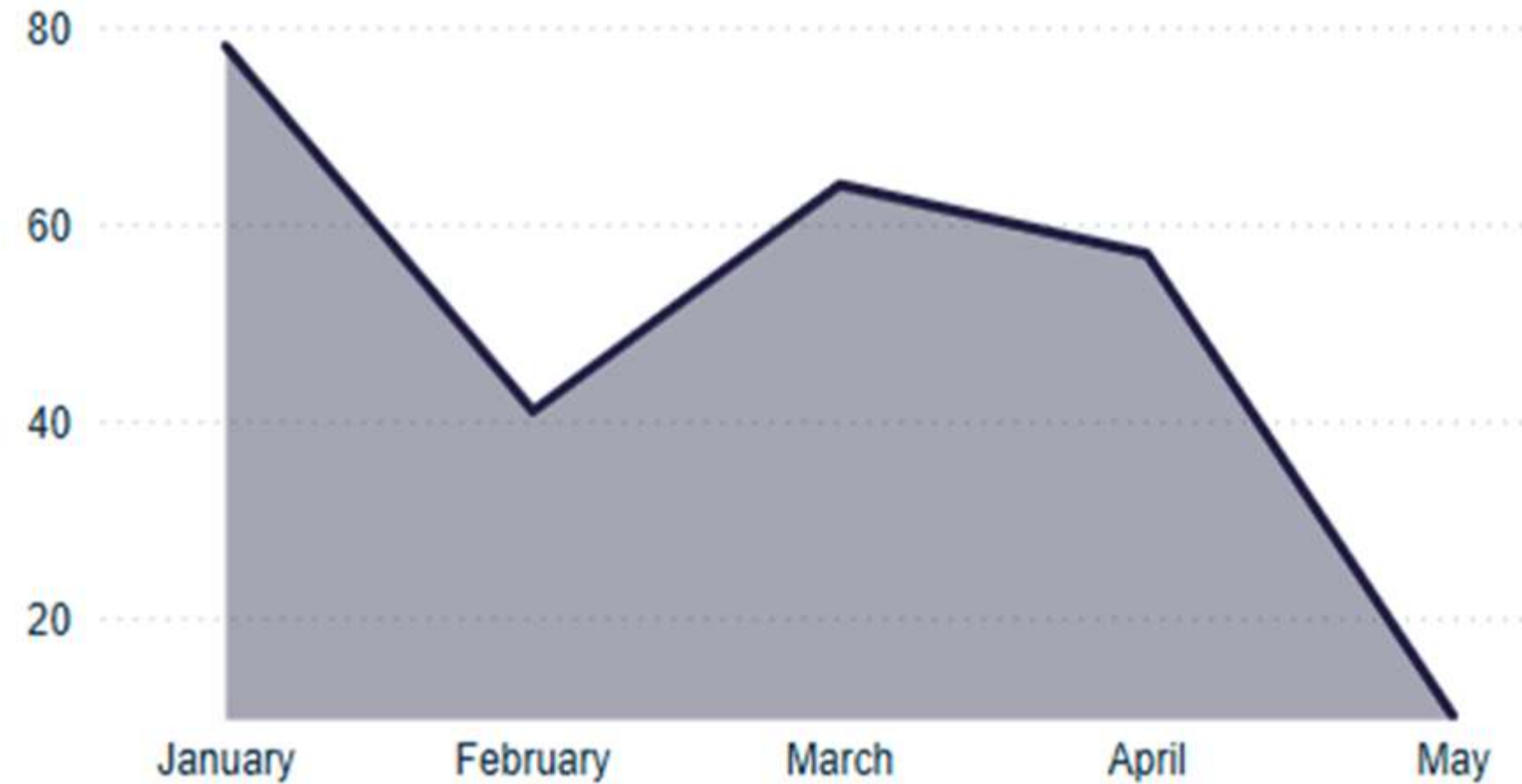


- The number of orders doesn't always reflect the profit.
- Some clients place fewer orders but bring in the highest profit.
- If we increase sales to those clients, our profit will grow significantly.

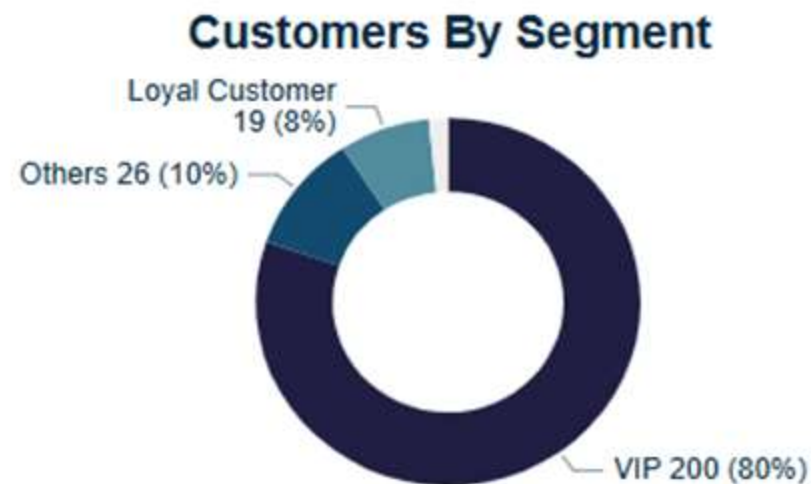
# Customer Analysis

- There was a drop in the number of new customers in February and March compared to January.

Number Of Sign Up By Month



# Customer Analysis

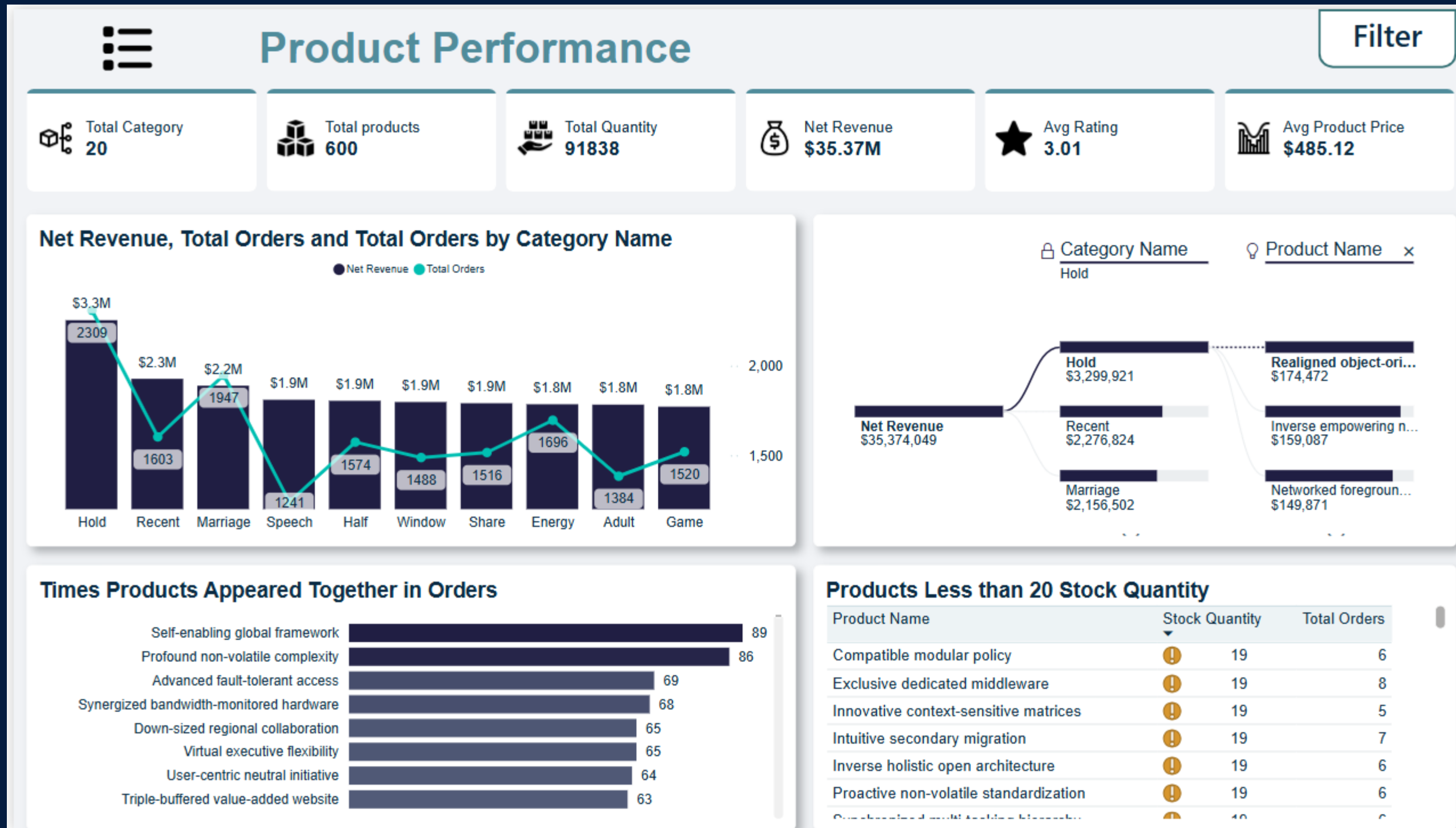


**Customers Table**

Customers	Total Sessions	Monetary	Frequency	AOV	Total Quantity	RFM	Segment
Aaron Jones	44	\$161,823	44	\$4,378	421	455	VIP
Adrian Johnson	35	\$188,770	49	\$5,008	495	455	VIP
Alexis Knox	47	\$168,766	42	\$4,693	387	455	VIP
Amber Coleman	31	\$160,748	55	\$4,509	536	455	VIP
Amber French	13	\$221,467	56	\$5,135	556	455	VIP
Amy Hill	30	\$210,044	52	\$4,721	506	455	VIP
Andrea Macdonald	30	\$148,737	57	\$3,212	392	455	VIP

- 80% of our Customers are VIP , which is a positive indicator.
- The table shows full details including:
- Number of orders
- Average order value
- Customer classification

# PRODUCT ANALYSIS



# Product Analysis

- Although the Speech category has lower sales compared to other categories, it ranks among the top 5 in revenue.
- Increasing sales in this category can lead to a significant boost in total revenue

Net Revenue, Total Orders and Total Orders by Category Name



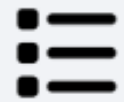


# Product Analysis

Products Less than 20 Stock Quantity		
Product Name	Stock Quantity	Total Orders
Compatible modular policy	19	6
Exclusive dedicated middleware	19	8
Innovative context-sensitive matrices	19	5
Intuitive secondary migration	19	7
Inverse holistic open architecture	19	6
Proactive non-volatile standardization	19	6
Comprehensive modular architecture	19	6

- The table shows products with less than 20 units in stock.
- It also includes the number of times each product was sold.
- This helps identify which items should be prioritized for restocking.

# RETURNS & CANCELLED ANALYSIS



## Order Issues

Filter



Total Orders  
**10200**



Potential Revenue  
**\$44.9M**



Orders Cancelled  
**2012**



Lost Revenue  
**\$9.5M**



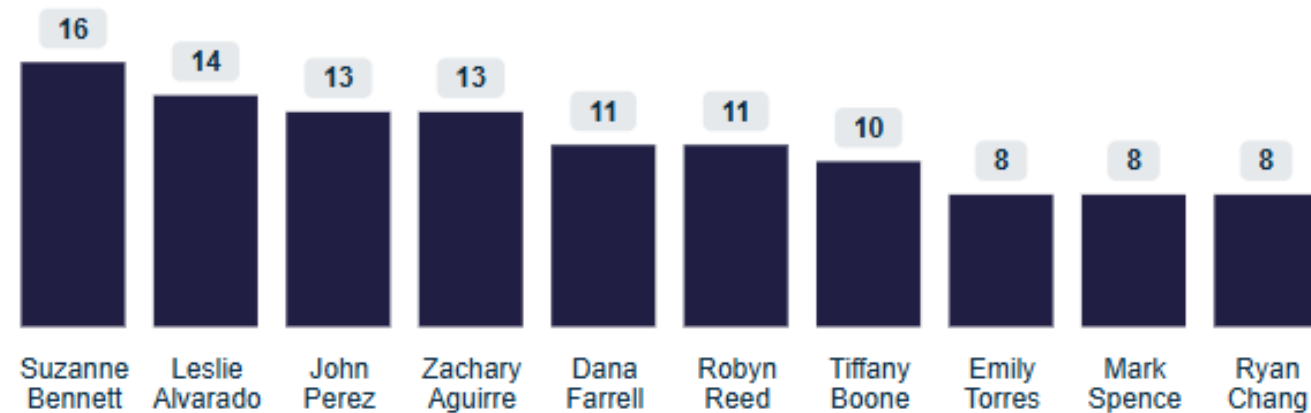
Net Revenue  
**\$35.4M**



Returned Orders  
**51**

Return rate %  
**0.50%**

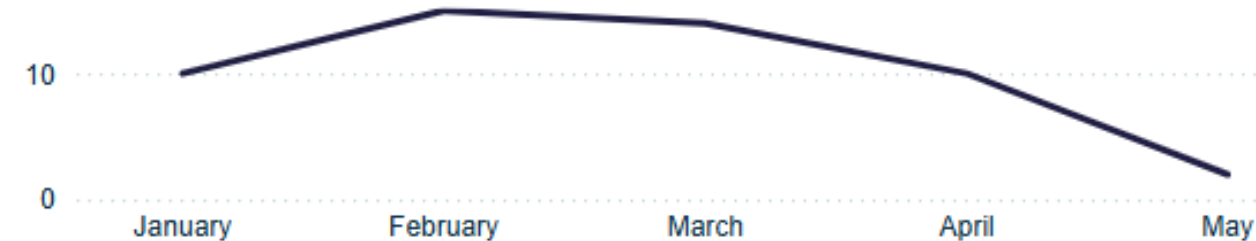
### Top 10 Cancelled Orders by Customers



### Products Have more 20 Cancelled Orders



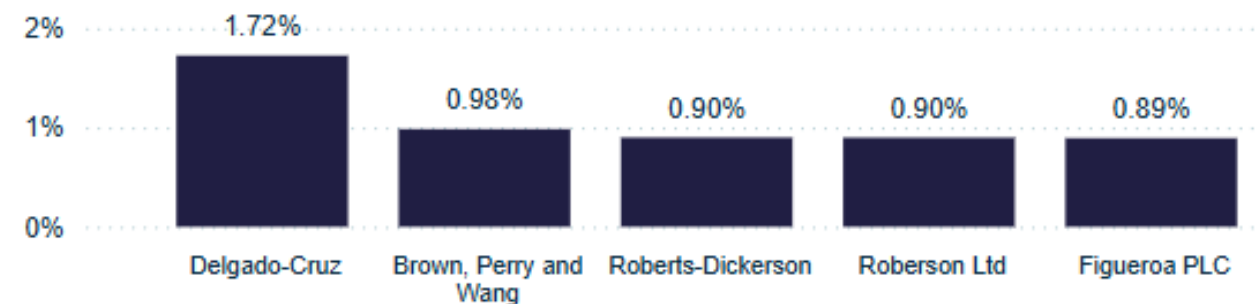
### Return Order By Month



### Returned Orders by Reason



### Top 5 Return Rate By Supplier Name



# Recommendations

These are some recommendations we can make based on the dashboard analysis:

## Inventory & Product Management

- **Restock high-demand, low-stock items** promptly to prevent stockouts and lost sales opportunities.

## Revenue & Orders

- **Prioritize promotional efforts** on top-performing product categories such as **Hold**, **Recent**, and **Speech** to maximize revenue impact.

## Customer Retention

- **Strengthen loyalty programs** to retain the **top 80% of VIP customers**, offering exclusive perks and personalized experiences.

## Returns & Cancellations

- **Reduce cancellation rates** by enhancing product descriptions with clearer visuals, sizing guides, and usage expectations.

## Sales Trends & Discounts

- **Leverage discount strategies more effectively**—with current usage at **43%**, explore targeted discounts and limited-time offers to drive conversions without impacting margins.

1

Deploy and secure the Azure SQL database

2

Use SSMS to connect to the Azure SQL database and run basic to advanced SQL queries

3

Connect Power BI to Azure SQL using to build interactive dashboards

4

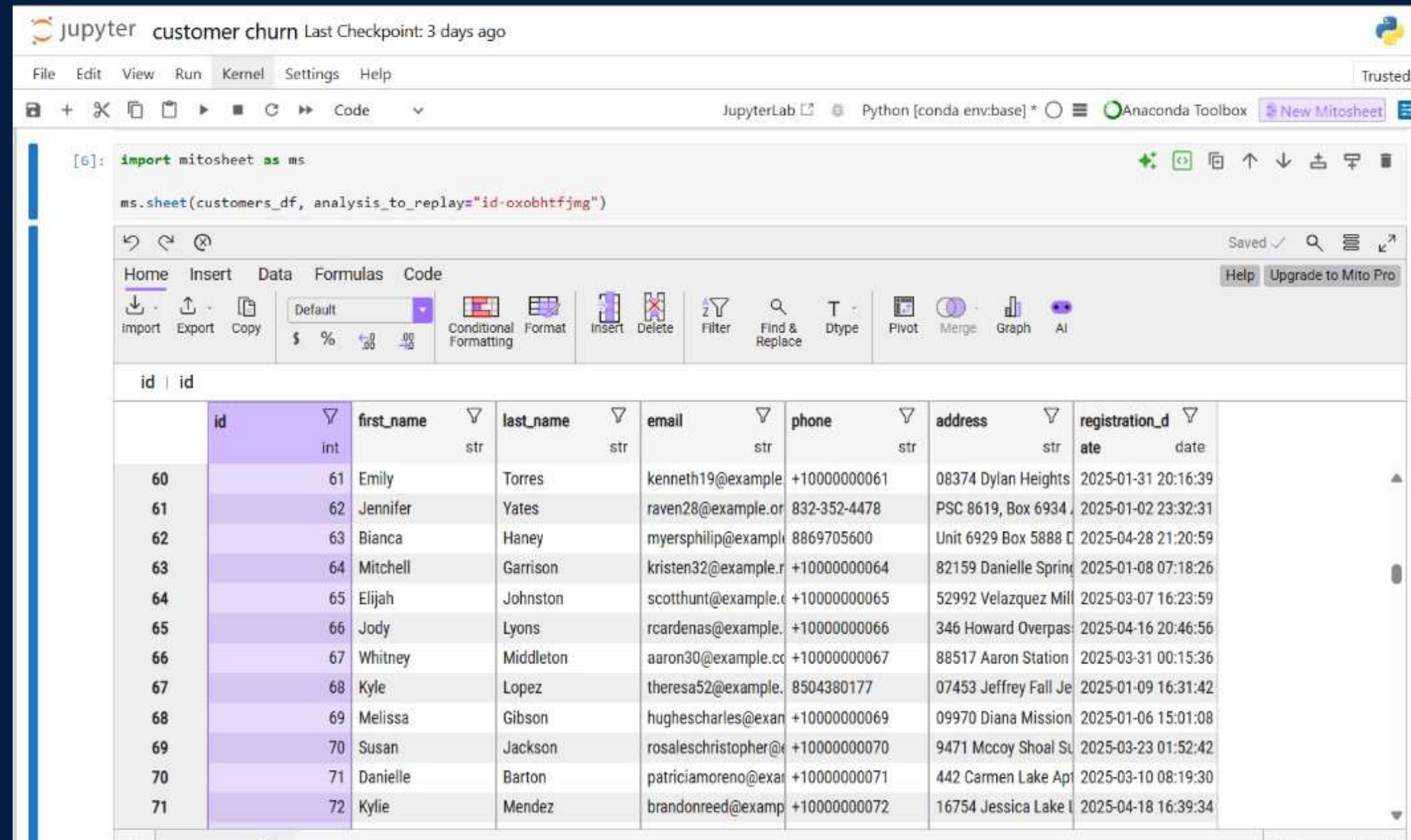
Apply machine learning to predict key business metrics

# Main Workflow



# MACHINE LEARNING

- We used Mito for spreadsheet-style cleaning, along with Python libraries such as Pandas and NumPy to handle missing values, convert data types, and standardize formats. These steps ensured the dataset was clean and ready for modeling.
- we used two of machine learning models that predict a group of different things that will affect our decision-making, which we will detail in the upcoming slides



The screenshot displays a JupyterLab environment with a Mito spreadsheet editor. The code cell at the top shows the following code:

```
[6]: import mitosheet as ms
ms.sheet(customers_df, analysis_to_replay="id-oxobhtfjmg")
```

Below the code, the Mito interface shows a table with the following columns and data:

	id	first_name	last_name	email	phone	address	registration_d
60	61	Emily	Torres	kenneth19@example	+10000000061	08374 Dylan Heights	2025-01-31 20:16:39
61	62	Jennifer	Yates	raven28@example.or	832-352-4478	PSC 8619, Box 6934	2025-01-02 23:32:31
62	63	Bianca	Haney	myersphilip@exampl	8869705600	Unit 6929 Box 5888	2025-04-28 21:20:59
63	64	Mitchell	Garrison	kristen32@example.r	+10000000064	82159 Danielle Sprin	2025-01-08 07:18:26
64	65	Elijah	Johnston	scotthunt@example.e	+10000000065	52992 Velazquez Mill	2025-03-07 16:23:59
65	66	Jody	Lyons	rcardenas@example.	+10000000066	346 Howard Overpas	2025-04-16 20:46:56
66	67	Whitney	Middleton	aaron30@example.co	+10000000067	88517 Aaron Station	2025-03-31 00:15:36
67	68	Kyle	Lopez	theresa52@example.	8504380177	07453 Jeffrey Fall Je	2025-01-09 16:31:42
68	69	Melissa	Gibson	hughescharles@exan	+10000000069	09970 Diana Mission	2025-01-06 15:01:08
69	70	Susan	Jackson	rosaleschristopher@	+10000000070	9471 Mccoy Shoal Su	2025-03-23 01:52:42
70	71	Danielle	Barton	patriciamoreno@exa	+10000000071	442 Carmen Lake Ap	2025-03-10 08:19:30
71	72	Kylie	Mendez	brandonreed@examp	+10000000072	16754 Jessica Lake	2025-04-18 16:39:34



# ML MODELS

## ▶ **Customer Churn Prediction**

A classification model that predicts if a customer will return for another purchase

## ▶ **Revenue Prediction**

A regression model that predicts payment amount based on order details

# Customer Churn Prediction Using Azure Machine Learning

Powered by Azure ML Designer

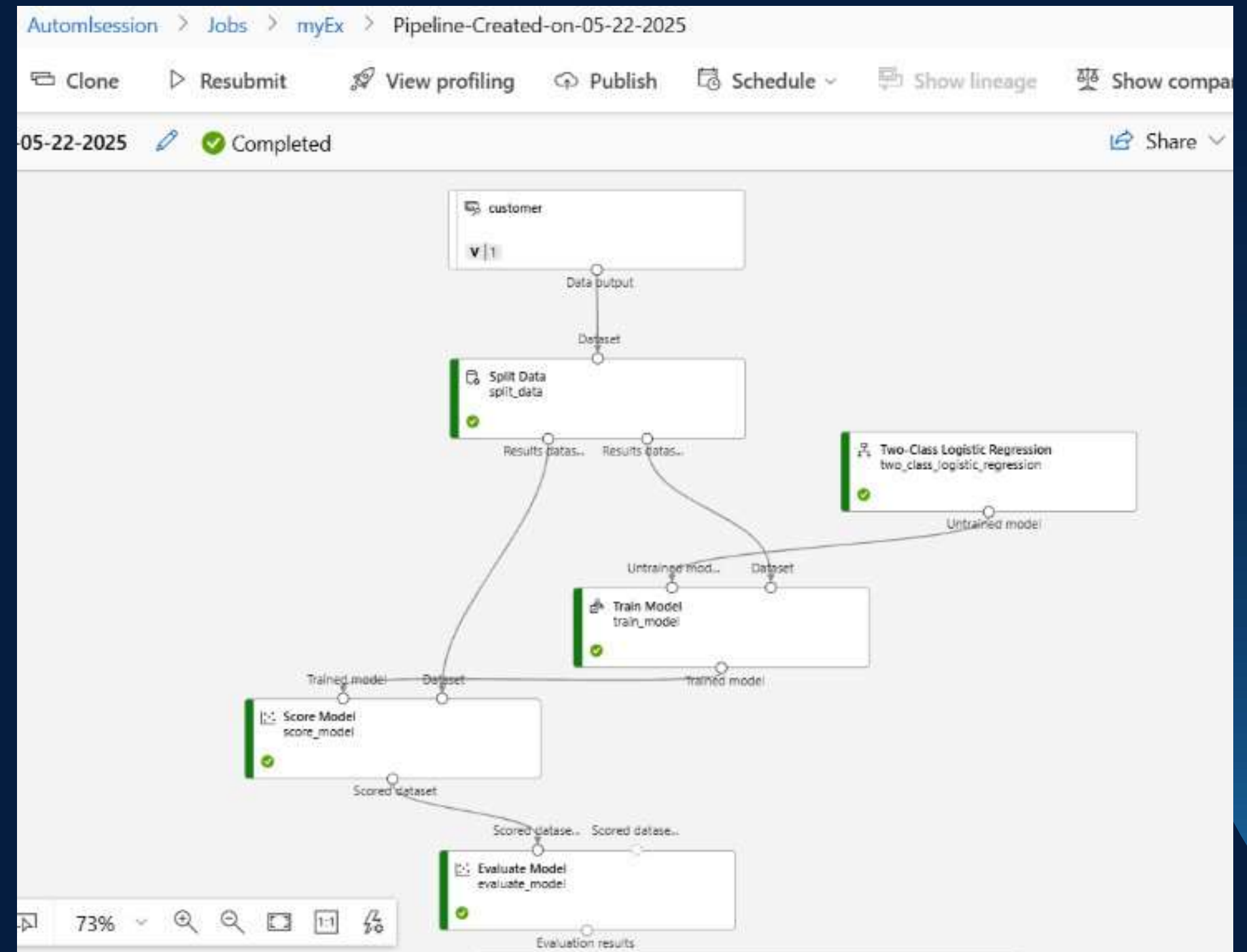
## Business Problem

### Customer Churn: A Critical Challenge

- Losing customers affects revenue and growth.
- Early prediction helps you retain valuable customers through targeted actions.

- **Goal:**

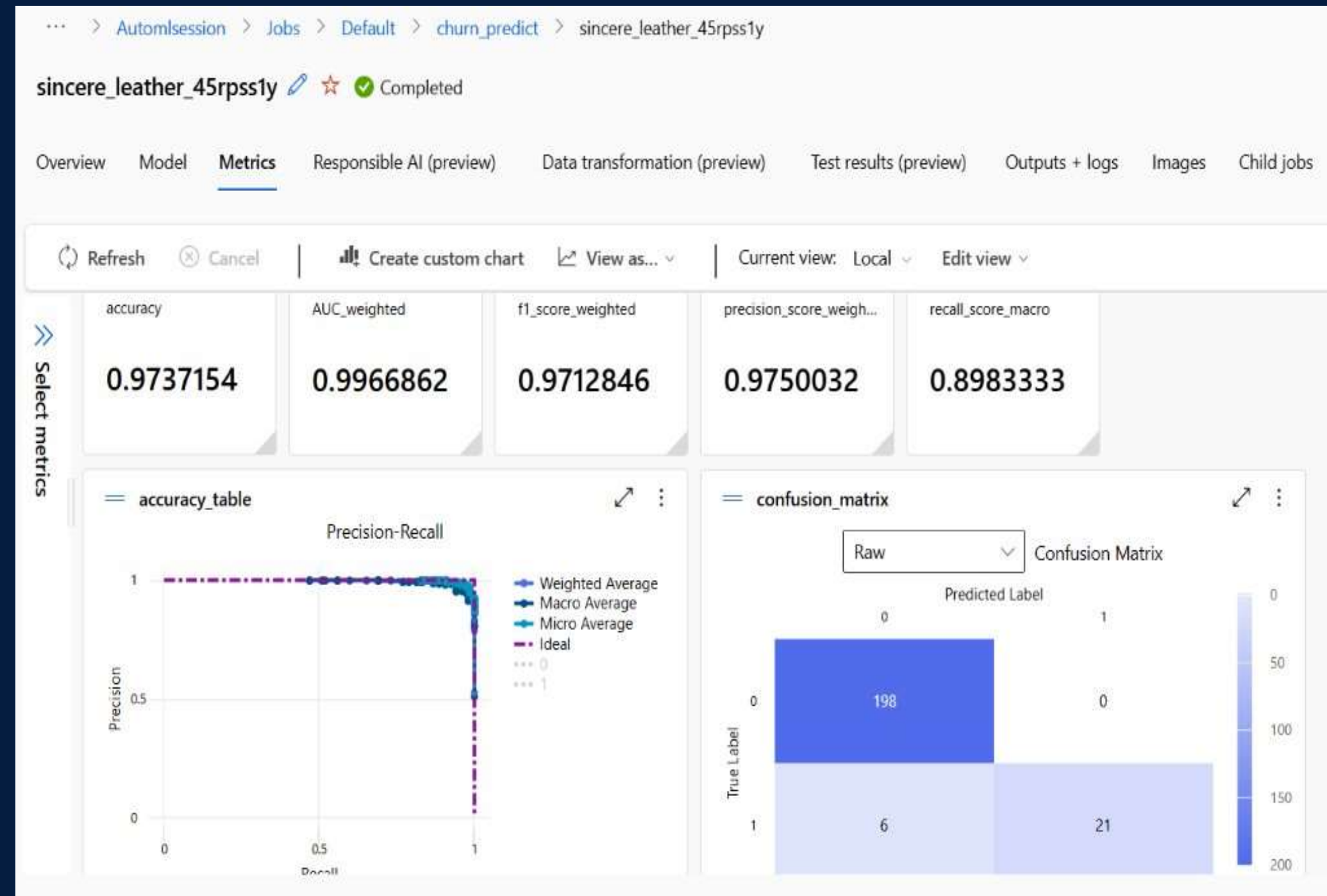
Predict customers who are likely to leave so you can act before they do.



# Customer Churn Prediction Using Azure Machine Learning

Powered by Automated ML

- **Data Source:**  
Customer transaction and behavior data
- **Key Features:**  
Days since last order, Total Spend, num of sessions, Session duration, return rate
- **Labeled Dataset:**  
Customers marked as **churned**



# Revenue Prediction Using Azure Machine Learning

## • "Why Predict Revenue?"

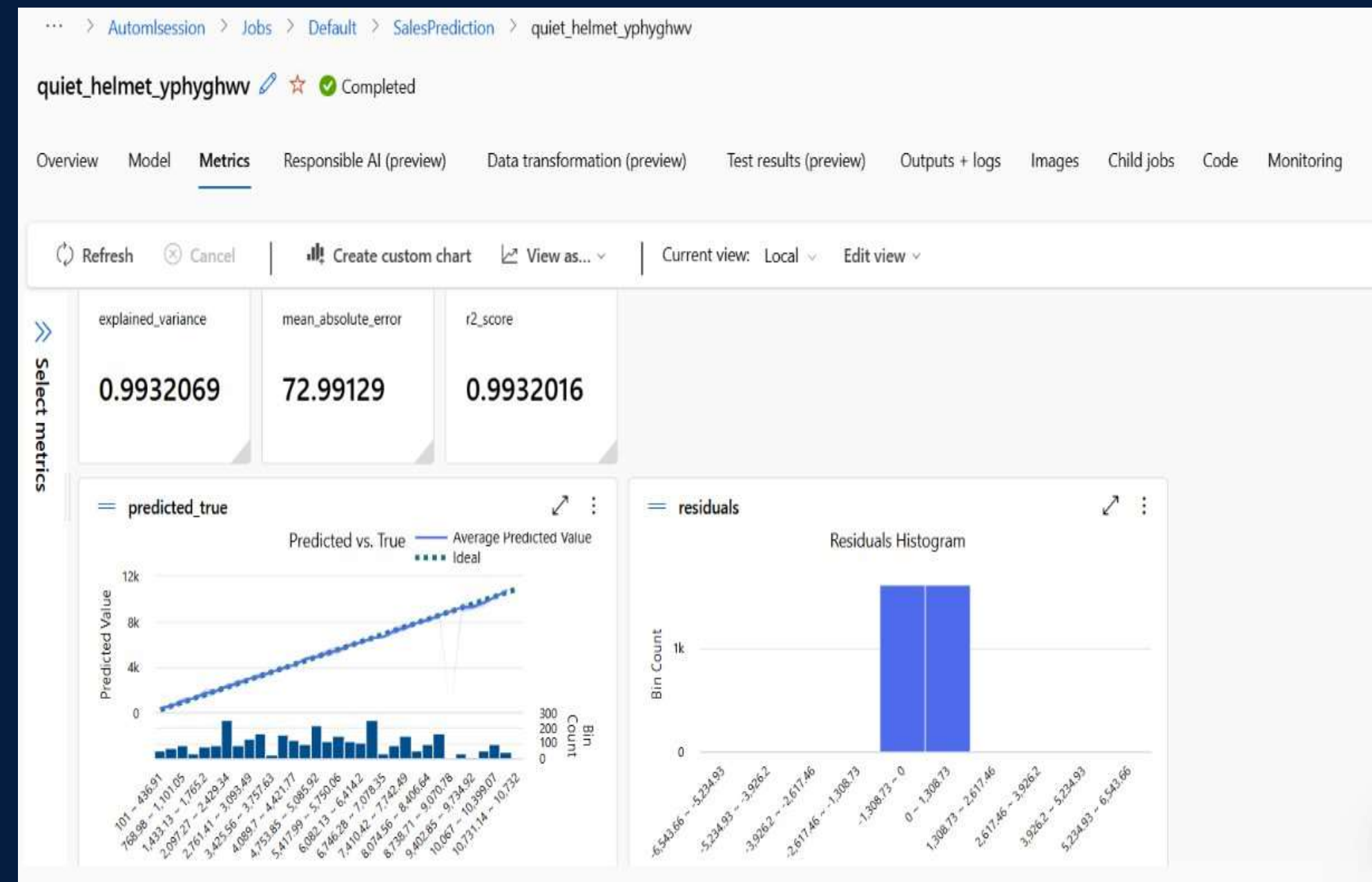
### Key Challenges:

- Fluctuating demand, pricing inefficiencies, missed upsell opportunities.

### Solution Benefits:

- Forecast sales trends by product, customer segment, and time.
- Optimize discounts/pricing dynamically.
- Identify high-value purchase windows (e.g., weekends, evenings).

Powered by Automated ML



# Any Questions?

We would like to hear your thoughts or answer any questions you may have.





# OUR TEAM

Mohamed Hesham

Aya Abd El-mageed

Yousef Mohamed

Ayman Abdrabo

Tarek Abd elhamid

Omar Wahby

Ahmed Abdo

Aya Yasser

Rama Alawad

Habiba Abd Elrazik

Abdelrahman Elsayed

Samy Halim

*Thank  
You*

