**Capstone Project**

**Problem Statement:** A small company, Axon, a retailer selling classic cars, is facing issues in managing and analyzing its sales data. The sales team is struggling to make sense of the data, and it does not have a centralized system to manage and analyze it. The management cannot get accurate and up-to-date sales reports, which is affecting the decision-making process.

**Aim:** The goal of the capstone project is to design and implement a BI solution using PowerBI and SQL that can help the company manage and analyze their sales data effectively. The action plan includes the following:

1. Import and integrate the data from MySQL database into PowerBI.
2. Clean and transform the data to make it ready for analysis.
3. Build interactive dashboards and reports using PowerBI that can help the sales team and management make sense of the data.
4. Use SQL to perform advanced analytics on the data and extract insights that can help the company improve its sales.
5. Enable the management to access the dashboards and reports in real-time and make data-driven decisions.

**About Dataset:** The dataset contains business data from 06/01/2003 until 31/05/2005. It consists of 8 tables and a brief description of this table are:

1. Customers: stores customer’s data.
2. Products: stores a list of scale model cars.
3. ProductLines: stores a list of product line categories.
4. Orders: stores sales orders placed by customers.
5. OrderDetails: stores sales order line items for each sales order.
6. Payments: stores payments made by customers based on their accounts.
7. Employees: stores all employee information as well as the organization structure such as who reports to whom.
8. Offices: stores sales office data

**Steps for Analysis**

1. Data Extraction: Created a new database in MySQL and used it as a data source to load data in PowerBI.
2. Data Transformation: Performed cleaning of data by using Power Query Editor. It included removing duplicates, rectifying datatype, removing null rows, and ensuring data consistency. The cleaned data was loaded to the PowerBI desktop.
3. Data Modelling: Managed the different table relationships and created two additional tables as Dim Date and Calculations for using Date Intelligence feature and calculating measures.

SQL Queries: -

1. revenue from each product-by-product code

select productcode,sum(quantityOrdered \* priceEach) as revenue\_generated\_per\_product

from orderdetails group by productcode ;

**Top-performing products by revenue:** The product with the highest revenue is S18\_3232, generating a revenue of $276,839.98, followed closely by S12\_1108 at $190,755.86. These products significantly outperform others, suggesting they are key drivers in the company's revenue.

**Product distribution**: A large portion of revenue is concentrated in the top 10 products, each contributing over $140,000, indicating that a few high-performing products dominate the sales.

2. revenue from each product by productname

select products.productcode,products.productname,sum(quantityOrdered \* priceEach) as revenue\_generated\_per\_product

from orderdetails

inner join products using(productcode)

group by productcode ;

**Top-selling product**: The product generating the highest revenue is the 1992 Ferrari 360 Spider red (S18\_3232), with $276,839.98 in revenue, highlighting it as a major contributor to overall sales in the dataset.

**Strong performance from classic vehicles**: Many of the top-performing products are classic cars and vintage vehicles, like the 2001 Ferrari Enzo (S12\_1108) and 1952 Alpine Renault 1300 (S10\_1949), showing a strong market demand for nostalgic or luxury items.

3. product never sold by Axon

select products.productCode, products.productname from

products left join orderdetails

using(productcode) where orderdetails.productCode is null;

**Product not sold**: The 1985 Toyota Supra (S18\_3233) is the only product listed in the dataset that has never been sold, as indicated by its absence from the orderdetails table.

4. product avgerage selling price, average discount, average profit and average profit percentage

select productname,buyprice as cost\_price,msrp as markup\_price,

sum(quantityordered \* priceeach) / sum(quantityOrdered) as avg\_selling\_price,

msrp - (sum(quantityordered \* priceeach) / sum(quantityOrdered)) as avg\_discount,

((sum(quantityordered \* priceeach) / sum(quantityOrdered)) - buyprice) as avg\_profit,

round(((sum(quantityordered \* priceeach) / sum(quantityOrdered)) - buyprice) / buyprice \*100) as avg\_profit\_percentage

from orderdetails

inner join products using(productcode)

group by productCode

order by avg\_profit\_percentage desc;

**Key Insights**

Products like vintage cars and classic vehicles exhibit exceptionally high-profit margins (above 100%).

These products often have a significant markup compared to their price due to rarity or luxury appeal.

Highest profit coming from 1937 Horch 930V Limousine, 1961 Chevrolet Impala, 1926 Ford Fire Engine.

5. quatity sold, quantity left product\_wise

select products.productname, quantityInStock, sum(quantityOrdered) quantity\_sold,

quantityInStock - sum(quantityOrdered) stock\_left

from products inner join orderdetails using(productcode)

group by productCode;

**Key Insights**

This captures the essential data for product quantities sold and stock left.

6. order status

SELECT

count(\*) as total\_orders,

SUM(CASE WHEN status = 'Shipped' THEN 1 ELSE 0 END) AS Shipped,

SUM(CASE WHEN status = 'Cancelled' THEN 1 ELSE 0 END) AS Cancelled,

SUM(CASE WHEN status = 'Resolved' THEN 1 ELSE 0 END) AS Resolved,

SUM(CASE WHEN status = 'On Hold' THEN 1 ELSE 0 END) AS On\_Hold,

SUM(CASE WHEN status = 'In Process' THEN 1 ELSE 0 END) AS In\_Process,

SUM(CASE WHEN status = 'Disputed' THEN 1 ELSE 0 END) AS Disputed

FROM orders;

**Key Insights**

The total number of orders in the system is 326. Among these, 303 orders have been Shipped, 6 orders are Cancelled, 4 orders are Resolved, 4 orders are On Hold, 6 orders are In Process, and 3 orders are marked as Disputed.

7. classify shipping speed

SELECT ordernumber,

orderdate,

DATEDIFF(shippeddate, orderdate) AS dispacted\_in,

CASE

WHEN DATEDIFF(shippeddate, orderdate) > 0 AND DATEDIFF(shippeddate, orderdate) <= 2 THEN "Quick"

WHEN DATEDIFF(shippeddate, orderdate) > 2 AND DATEDIFF(shippeddate, orderdate) <= 3 THEN "Medium"

ELSE "Slow"

END AS dispatch\_category

FROM orders

WHERE status = "Shipped";

**Key Insights**

The query retrieves the order number, order date, and calculates the number of days taken from order date to shipment date (dispatched\_in). It then classifies the shipping speed into three categories:

Quick: Dispatch in 1 to 2 days

Medium: Dispatch in 3 days

Slow: Dispatch in more than 3 days

8. total orders placed by customers having status "shipped" arranged highest to lowest

select orders.customernumber, customers.customername, sum(case when status = 'shipped' then 1 else 0 end) as order\_counts

from orders join customers using(customernumber)

group by customerNumber

order by 3 desc ;

**Key Insights**

The query retrieves the total number of orders with the status "Shipped" for each customer. The results are sorted in descending order based on the total count of shipped orders.

Top customers with the most orders (highest to lowest):

Euro+ Shopping Channel: 22 orders

Mini Gifts Distributors Ltd.: 16 orders

Australian Collectors, Co.: 5 orders

Reims Collectables: 5 orders

Dragon Souveniers, Ltd.: 5 orders

9. customers with total order placed, transaction worth having status "shipped" arranged highest to lowest

select orders.customernumber, customers.customername, sum(case when status = 'shipped' then 1 else 0 end) as order\_counts, round(sum(quantityordered \* priceeach)) worth

from orders

join customers using(customernumber)

join orderdetails using(ordernumber)

group by customerNumber

order by 3 desc, 4 desc ;

**Key Insights**

The query retrieves the total number of orders and the total transaction worth (calculated as quantityordered \* priceeach) for customers with a "shipped" status. The results are sorted in descending order by order counts and then by transaction worth.

Top customers by order counts and transaction worth:

Euro+ Shopping Channel: 213 orders, $820,690 in total worth

Mini Gifts Distributors Ltd.: 178 orders, $591,827 in total worth

Australian Collectors, Co.: 55 orders, $180,585 in total worth

AV Stores, Co.: 51 orders, $148,410 in total worth

Muscle Machine Inc.: 48 orders, $177,914 in total worth

10. Top 3 best-selling products by revenue

with cte as(

select

productcode ,sum(quantityordered \* priceeach) revenue\_generated, row\_number() over (order by sum(quantityordered \* priceeach) desc) as ranking

from orderdetails

group by productcode)

select productcode, productname, revenue\_generated

from cte

join products using(productcode)

where ranking<=3

order by ranking;

**Key Insights**

The query retrieves the top 3 best-selling products by revenue generated (calculated as quantityordered \* priceeach). The results are ordered by the highest revenue.

Top 3 best-selling products:

1992 Ferrari 360 Spider red (Product Code: S18\_3232) - $276,839.98 in revenue

2001 Ferrari Enzo (Product Code: S12\_1108) - $190,755.86 in revenue

1952 Alpine Renault 1300 (Product Code: S10\_1949) - $190,017.96 in revenue

11. top 2 Customers who placed the highest number of orders each year

with cte as (

select

year(orderdate) as ordered\_year, customernumber,

count(\*) as no\_of\_orders\_placed,

rank() over (partition by year(orderdate) order by count(\*) desc) as ranking

from orders

group by customernumber, year(orderdate)

order by 1, 3 desc)

select ordered\_year, customernumber, customername, no\_of\_orders\_placed

from cte join customers using(customernumber)

where ranking <= 2;

**Key Insights**

The query identifies the top 2 customers who placed the highest number of orders each year.

2003:

Euro+ Shopping Channel (Customer Number: 141) - 8 orders

Mini Gifts Distributors Ltd. (Customer Number: 124) - 4 orders

2004:

Euro+ Shopping Channel (Customer Number: 141) - 9 orders

Mini Gifts Distributors Ltd. (Customer Number: 124) - 6 orders

2005:

Euro+ Shopping Channel (Customer Number: 141) - 9 orders

Mini Gifts Distributors Ltd. (Customer Number: 124) - 7 orders

12. Products with increasing sales trends

with cte as(

select

year(orderdate) year\_of\_order,

productcode,

sum(quantityOrdered) present\_year\_sale,

lead(sum(quantityOrdered),1) over (partition by productCode order by year(orderdate)) next\_year\_sale

from orders join orderdetails using(ordernumber)

group by productcode, year(orderdate)),

cte2 as (

select \* , case when present\_year\_sale< next\_year\_sale then "Increasing" else "Decresing" end as trend

from cte)

select productcode, present\_year\_sale, next\_year\_sale from cte2 where trend = "increasing" order by productcode;

**Key Insights**

The query identifies the products where the sales (in terms of quantity ordered) are rising compared to the previous year.

These products show a trend of increasing sales from one year to the next:

S10\_1678 – 334 units in the present year, 530 units in the next year

S10\_1949 – 342 units in the present year, 445 units in the next year

S10\_2016 – 300 units in the present year, 469 units in the next year

S10\_4698 – 314 units in the present year, 462 units in the next year

S10\_4757 – 349 units in the present year, 463 units in the next year

and so on..

13. customers who have never placed an order

select customernumber, customername

from customers

where customernumber not in

(select distinct customernumber from orders);

**Key Insights**

It will returns those customers that have registered but have not yet placed any order in the system.

14. Orders that took more than the average shipping time

select ordernumber,

datediff(shippeddate,orderdate) as shipping\_Time

from orders where datediff(shippeddate,orderdate) >(select avg(datediff(shippeddate,orderdate)) from orders) ;

**Key Insights**

The Average shipping time is 3.6 days Approx so all orders taking more than this will be returned as the output of this query.

The number **65 days** for Order Number **10165** stands out as an extreme case, which may be worth further investigation.

15. Monthly revenue trend of the company

WITH MonthlyRevenue AS (

SELECT

YEAR(orderdate) AS year\_of\_order,

MONTH(orderdate) AS month\_of\_order,

SUM(quantityOrdered \* priceEach) AS revenue

FROM orders

JOIN orderdetails USING(ordernumber)

GROUP BY 1,2

)

SELECT

year\_of\_order,

month\_of\_order,

round(revenue) revenue,

round(SUM(revenue) OVER (PARTITION BY year\_of\_order ORDER BY month\_of\_order)) AS cumulative\_revenue

FROM MonthlyRevenue;

**Key Insights**

The cumulative revenue grows steadily over time, reflecting the company's increasing sales over the months. The trends in the monthly revenue reflect key performance periods, with peaks in the later months of the year (e.g., November and December).

16. Percentage contribution of each product to total sales

select productcode, productname,

sum(quantityOrdered\* priceEach) revenue\_generated,

round(sum(quantityOrdered\* priceEach)/(select round(sum(quantityOrdered\* priceEach)) total\_sales from orderdetails) \*100,3) percentage\_of\_total\_sales

from orderdetails join products using(productcode)

group by productcode;

**Key Insights**

The query output reveals that the product with the highest contribution to total sales is the 1992 Ferrari 360 Spider red (S18\_3232), with a significant percentage of 2.882% of total sales. On the other hand, the 1997 BMW R 1100 S (S24\_1578) has one of the smallest contributions at 0.346%. The data indicates that luxury cars, such as Ferrari and Porsche models, tend to generate a larger share of the total revenue.

17. Finding the top 5 customers contributing the most revenue each year

with cte as(

select

year(orderdate) transaction\_year,

customernumber,

sum(quantityOrdered \* priceEach) revenue,

row\_number() over (partition by year(orderdate) order by sum(quantityOrdered \* priceEach) desc ) ranking

from orders join orderdetails using(ordernumber)

group by year(orderdate), customernumber

order by 1 asc, 3 desc)

select customername, transaction\_year, revenue , ranking

from cte join customers using(customernumber)

where ranking <= 5;

**Key Insights**

The query output reveals the top 5 customers contributing the most revenue each year. For 2003, Euro+ Shopping Channel leads with 189,840.15 in revenue, followed by Mini Gifts Distributors Ltd. at 167,783.08. In 2004, Euro+ Shopping Channel again takes the top spot with 340,830.87, and in 2005, they are still the top contributor with 290,018.52 in revenue. The results show that Euro+ Shopping Channel is consistently one of the highest-revenue-generating customers across all three years.

18. Total Orders and Sales for each productLine

select productline, count(\*) total\_orders, sum(quantityOrdered \* priceEach) total\_sales

from products join orderdetails using(productcode)

group by productline;

**Key Insights**

The query output shows the total orders and sales for each product line. Classic Cars lead with the highest sales of 3,853,922.49 from 1010 orders. Motorcycles follow with 1,121,426.12 in sales from 359 orders. In contrast, Trains have the least sales of 188,532.92 from just 81 orders, highlighting a significant disparity between the most and least popular product lines in terms of sales.

19. Orders where the customer ordered more than their average order quantity

select ordernumber, customernumber, sum(quantityOrdered) quantity\_ordered

from orders o join orderdetails od using(ordernumber)

group by ordernumber, customernumber

having quantity\_ordered >

(select sum(quantityordered)/count(distinct ordernumber) avg\_quantity

from orders o1 join orderdetails od1 using(ordernumber)

where o1.customernumber = o.customernumber);

**Key Insights**

The query identifies orders where customers ordered more than their average order quantity. The data reveals that several customers placed orders with quantities exceeding their personal average. For example, Customer 103 in order 10123 ordered 156 items, significantly higher than their average order quantity. Similarly, Customer 114 in order 10120 ordered 525 items, surpassing their typical order quantity. This trend is seen across numerous customers, indicating a few larger-than-usual orders within the dataset.