**Tailwind Traders Sales**

**Business Task**

Tailwind Traders wants to generate comprehensive reports and dashboards that provide actionable insights into global sales and profits. The company requires accurate data processing, transformations, and visualizations to support strategic decision-making, including time-based profit summaries, currency conversions, and mobile-accessible dashboards.

**Step 1: Prepare Sales Excel data.**

1. Opened the Excel workbook Tailwind Traders Sales.xlsx.
2. Inserted a Gross Revenue column and calculated revenue per product using Gross Product Price \* Quantity Purchased, formula =E2\*G2.
3. Created a Total Tax column and calculated tax per product using Tax Per Product \* Quantity Purchased, formula =F2\*G2.
4. Added a Net Revenue column and computed net earnings using Gross Revenue - Total Tax, formula =H2-I2.
5. Applied formulas to all rows by dragging down.
6. Reviewed the first 10 records to identify highest and lowest values for Net Revenue, Quantity Purchased, and Total Tax, and observed trends with related columns like Sales Rep.

**Step 2: Configure data sources.**

1. Loaded Sales, Purchases, and Countries data into Power BI.
2. Set correct data types for all columns (e.g., Whole Number, Fixed Decimal Number, Text, Date).
3. Checked column quality, distribution, and statistics for key fields like OrderID, Quantity Purchased, Warranty, and ReturnStatus.
4. Filtered out refunded purchases to maintain data integrity.
5. Imported historical currency exchange data using a Python script and integrated it into the report.

import pandas as pd

from io import StringIO

data = """Exchange ID;ExchangeRate;Exchange Currency

1;1;USD

2;0,75;GBP

3;0,85;EUR

4;3,67;AED

5;1,3;AUD"""

df = pd.read\_csv(StringIO(data), sep=';')

# Return the transformed dataframe

df

1. Saved the Power BI project as Tailwind Traders Report.pbix.

**Step 3: Design and develop the data model.**

1. Created relationships between Countries and Exchange Data tables (1:1, bi-directional).
2. Created relationships between Sales and Countries tables (\*:1, bi-directional).
3. Created relationships between Purchases and Sales tables (1:1, bi-directional).
4. Configured a Calendar table using DAX with Year, Month, Quarter, Weekday, and Day columns.

CalendarTable =

ADDCOLUMNS(

CALENDAR(DATE(2020, 1, 1), DATE(2023, 12, 31)),

"Year", YEAR([Date]),

"Month Number", MONTH([Date]),

"Month", FORMAT([Date], "MMMM"),

"Quarter", QUARTER([Date]),

"Weekday", WEEKDAY([Date]),

"Day", DAY([Date])

)

1. Created a relationship between Calendar and Purchases tables (\*:1).
2. Created a Sales in USD calculated table with Gross Revenue USD, Net Revenue USD, and Total Tax USD using exchange rates.

Sales in USD =

ADDCOLUMNS(

    Sales,

    "Country Name", RELATED(Countries[Country]),

    "Exchange Rate", RELATED('Exchange Data'[Exchange Rate]),

    "Exchange Currency", RELATED('Exchange Data'[Exchange Currency]),

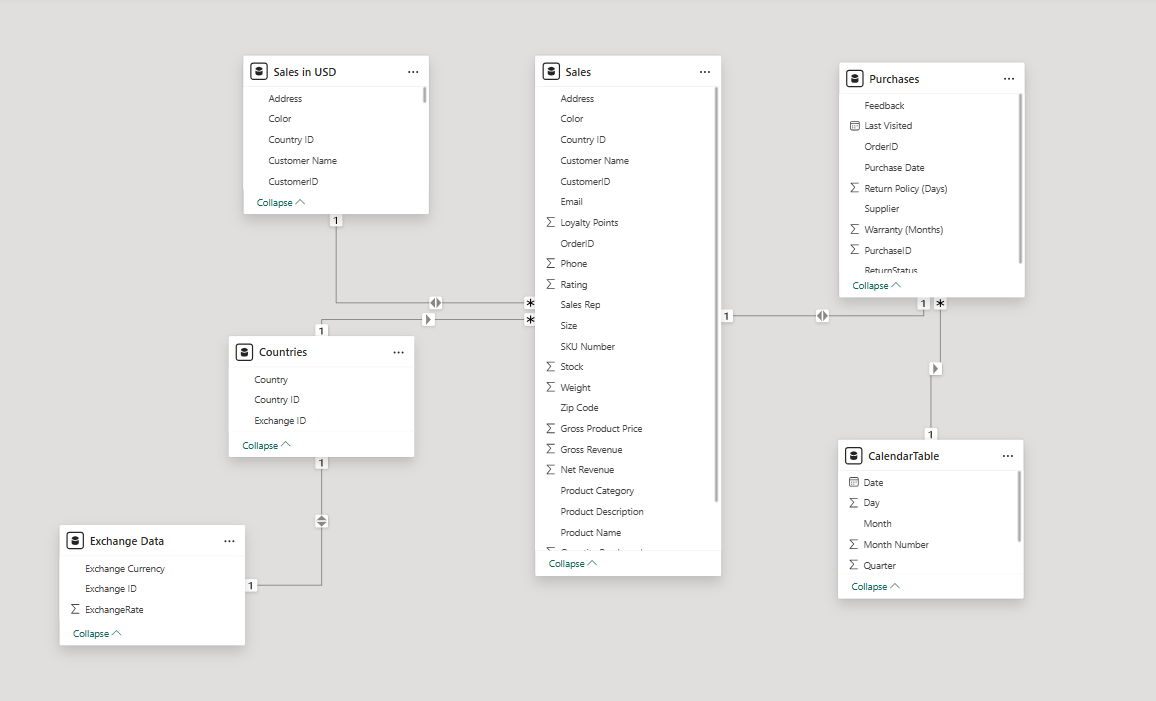
    "Gross Revenue USD", [Gross Revenue] \* RELATED('Exchange Data'[Exchange Rate]),

    "Net Revenue USD", [Net Revenue] \* RELATED('Exchange Data'[Exchange Rate]),

    "Total Tax USD", [Total Tax] \* RELATED('Exchange Data'[Exchange Rate])

)

1. Created a relationship between Sales in USD and Sales tables (1:1).



**Step 4: Configure aggregations using DAX.**

1. Created a Yearly Profit Margin measure in the Sales in USD table by dividing total profit by net revenue.

Yearly Profit Margin =

DIVIDE(

    SUM('Sales in USD'[Net Revenue USD]),

    SUM('Sales in USD'[Gross Revenue USD])

)

1. Created a Quarterly Profit measure using DATESQTD to aggregate profits by quarter.

Quarterly Profit Margin =

CALCULATE(

[Yearly Profit Margin],

DATESQTD('CalendarTable'[Date]))

1. Created a Year-to-Date Profit measure using TOTALYTD to calculate running totals from the start of the year.

YTD Profit Margin =

DIVIDE(

    TOTALYTD(SUM('Sales in USD'[Net Revenue USD]), 'CalendarTable'[Date]),

    TOTALYTD(SUM('Sales in USD'[Gross Revenue USD]), 'CalendarTable'[Date])

)

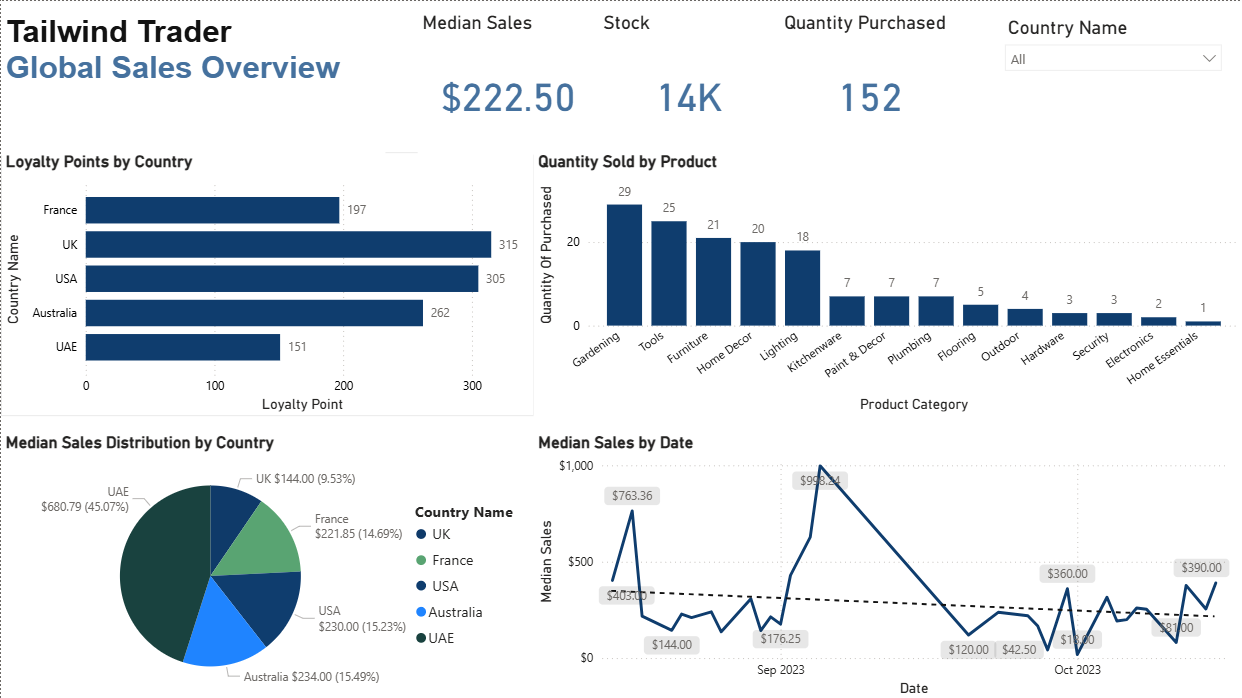
1. Created a Median Sales measure using the MEDIAN function on Gross Revenue USD.

Median Sales = MEDIAN('Sales in USD'[Gross Revenue USD])

1. Used the Performance Analyzer to test report performance: added Card visuals for Yearly Profit Margin, Median Sales, Quarterly Profit, and YTD Profit; recorded load times and ensured DAX queries were under 200ms.

**Step 5: Create a Sales report.**

1. Opened Tailwind Traders Report.pbix and renamed the report to Sales Overview.
2. Created a Clustered Bar Chart for Loyalty Points by Country with data labels enabled and identified the country with the highest loyalty points.
3. Built a Clustered Column Chart for Quantity Sold by Product, positioned beside the bar chart, with data labels enabled.
4. Designed a Pie Chart for Median Sales Distribution by Country, sorted ascending, and displayed with data labels.
5. Created a Line Chart for Median Sales Over Time using Date and Median Sales, enabled data labels, and added a trend line.
6. Added Card visuals for Stock, Quantity Purchased, and Median Sales, positioned strategically above relevant charts for clarity.
7. Inserted a Slicer by Country Name to filter the report dynamically.
8. Saved the updated Sales Overview report.



**Step 6: Create a Profit report.**

1. Opened the Sales Overview report and created a new page named Profit Overview.
2. Built a Clustered Bar Chart for Net Revenue by Product, enabled data labels, sorted descending, and identified the top product by Net Revenue.
3. Created a Donut Chart for Yearly Profit Margin by Country with detailed labels (category and percent of total).
4. Designed an Area Chart for Yearly Profit Margin Over Time using Calendar Date and Profit Margin, enabled data labels for clarity.
5. Added Card visuals for YTD Profit and Net Revenue USD, positioned above the Net Revenue by Product chart.
6. Configured a KPI visual for Gross Revenue USD with Date as the trend axis, placed beside the Net Revenue card.
7. Inserted a Slicer based on Calendar Date to filter profit insights dynamically.
8. Saved and published the report to Power BI Service.

