**Power BI Project Report – Amazon Sales Analysis**

**Name: Prerana Gajanan Sonar  
Project Title: Amazon Sales Report Dashboard  
Tools Used: Power BI, Microsoft Excel  
Duration: [Insert Duration – e.g., 2 weeks]  
Domain: Data Analytics / Business Intelligence**

**1. Introduction**

This project focuses on analyzing Amazon sales data using Power BI. The purpose of this project is to visualize key performance metrics such as profit, sales, and order details across different categories, regions, and timelines. The final dashboard provides an interactive view of important KPIs and insights.

**2. Objectives:**

* **To visualize and analyze Amazon sales data.**
* **To track and monitor KPIs such as total sales, quantity sold, profit, and customer performance.**
* **To identify top-performing products and regions.**
* **To support data-driven decision-making through interactive dashboards.**

**3.Dataset Overview**

The dataset used in this analysis contains data related to Amazon sales transactions. It consists of a single consolidated table with the following columns:

* **Order ID**
* **Order Date**
* **Ship Date**
* **Customer Name**
* **Email ID**
* **Geography (State, City)**
* **Category**
* **Product Name**
* **Sales**
* **Quantity**
* **Profit**
* **Fiscal Year (FY)**
* **Month, Quarter, Timeline**

The dataset was loaded from a MySQL database into Power BI and transformed for visual analysis.

**3. Database Connectivity**

To perform the analysis, we establish a connection to the MySQL database containing the Amazon sales data. We use Python libraries such as pandas and mysql.connector to fetch and manipulate the data directly from the database.

**Code Snippet: Connecting to MySQL and Fetching Data**

import mysql.connector

import pandas as pd

# Connect to the MySQL database

connection = mysql.connector.connect(

user='root',

password='',

host='localhost',

database='ecommerce'

)

# Create a cursor object

cursor = connection.cursor()

# Query data from the 'customer' table

cursor.execute('SELECT \* FROM customer')

customer\_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in cursor.description])

# Query data from the 'product' table

cursor.execute('SELECT \* FROM product')

product\_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in cursor.description])

# Query data from the 'order\_details' table

cursor.execute('SELECT \* FROM order\_details')

order\_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in cursor.description])

# Print first 5 rows

print(customer\_data.head())

print(product\_data.head())

print(order\_data.head())

**Description**

customer\_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in cursor.description])

**Explanation:**

* **cursor.fetchall()**: This method retrieves all rows from the result set as a list of tuples. Each tuple represents a record from the database.
* **columns=[desc[0] for desc in cursor.description]**:
  + cursor.description returns metadata about the columns in the result set, such as name, type, etc.
  + The list comprehension extracts the **first element** (i.e., the column name) from each tuple in cursor.description, forming a list of column names.
* **pd.DataFrame(...)**:
  + Combines the retrieved data and column names to create a **structured Pandas DataFrame**.
  + This makes it easier to analyze and visualize the data using Pandas tools.

💡 **In Summary**: This line efficiently converts SQL result sets into DataFrames, allowing for further processing and visualization.

**4. Output Tables**

**Customer Table (customer\_data)**

|  |  |  |
| --- | --- | --- |
| **Customer Name** | **Email ID** | **Geography** |
| Daniel Ward | danielward@gmail.com | Los Angeles, CA |
| Jane Austin | janeaustin@gmail.com | Los Angeles, CA |

**Product Table (product\_data)**

|  |  |
| --- | --- |
| Product Name | Category |
| Avery 1" Heavy-Duty Binders | Binders |
| HP LaserJet Printer | Technology |

**Order Details Table (order\_data)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Order ID | Order Date | Sales | Quantity | Profit | FY | Quarter | Month |
| CA-2018-1001 | 12-Jun-19 | 17.68 | 2 | 4.52 | FY 18-19 | Q4 | Jun |
| CA-2019-1120 | 22-Aug-19 | 42.30 | 3 | 6.35 | FY 19-20 | Q1 | Aug |

⚠️ Note: These are sample rows. Full dataset contains 3,030 rows.

**5. Dashboard Overview**

The Power BI dashboard provides visual insights into the following key metrics:

* **KPIs:**
  + Total Orders: 12K
  + Total Sales: ₹725.46K
  + Total Profit: ₹108.42K
  + Unique Customers: 686
* **Visual Components:**
  + Sales vs Profit Trend Line Chart
  + Profit Year Over Year (Bar Chart)
  + Top Products by Sales (Horizontal Bar)
  + Category Distribution (Donut Chart)
  + Sales by Geography (Map or Region-based Visual)

These visuals help in identifying trends, high-performing categories, and customer distribution.

**7. Conclusion**

This Amazon Sales Dashboard enables business stakeholders to monitor performance, identify profitable products, and make informed decisions. The integration of MySQL and Power BI creates a powerful reporting ecosystem combining backend data storage and frontend visualization.

8. **Screenshots:**







