Project Title	**Retail Sales Analytics Using Excel, SQL, Power BI and Python**
Skills take away From This Project	• **Data Cleaning and Preprocessing using Excel** • **Data Storage, Querying, and Transformation using SQL** • **Data Analysis and Visualization using Python (Pandas, NumPy, Matplotlib/Seaborn)** • **Customer Segmentation using Machine Learning (KMeans/Clustering)** • **Business Intelligence and Reporting using Power BI** • **End-to-End Data Pipeline Integration**
Domain	**Retail, Sales & Inventory Analytics**

Problem Statement:

In a competitive retail market, businesses need to track sales performance, understand customer behavior, and identify growth opportunities. This project aims to build an end-to-end analytics solution that enables insights generation from raw transactional data using a multi-tool approach.

Business Use Cases:

1. Analyze sales performance across different stores and staff
2. Identify high-value customers and buying behavior patterns
3. Perform customer segmentation for targeted marketing
4. Monitor inventory levels and product performance
5. Create an interactive Power BI dashboard for management insights

Approach:

Phase 1: Excel – Data Cleaning & Preparation

1. **Standardize Column Headers**
   - Ensure all columns across sheets (like customers, orders, products) are consistently named (e.g., use snake_case or camelCase).

2. **Remove Duplicates**
   - Identify and remove any duplicate records (e.g., in customers or

order_items).

3. **Handle Missing Values**
    - Use filtering and formulas to find blanks/nulls.
    - Fill missing ZIP codes or phone numbers with placeholders or "NA".
4. **Data Type Conversion**
    - Convert order dates, shipped dates, and required dates into Excel Date format.
    - Ensure numeric fields (quantity, price) are set to number format.
5. **Data Validation**
    - Use dropdowns for fields like order_status (Pending, Shipped, Delivered) to ensure consistency.
6. **Create New Derived Columns**
    - Add a column in order_items to calculate total_price = (list_price * quantity) - discount.
7. **Merge Lookup Data**
    - Use VLOOKUP or XLOOKUP to merge product names into order_items using product_id.
8. **Create Basic Pivot Table**
    - Build a pivot table summarizing total sales by product category.
9. **Sort and Filter for Outliers**
    - Find outliers in pricing (e.g., products with very high or low list_price).
10. **Prepare Final CSVs**
    - Save cleaned sheets as CSVs named: customers.csv, orders.csv, etc., for SQL import.

Phase 2: SQL – Database Management and Querying

1. **Create Tables Based on ERD**

- Use CREATE TABLE statements to replicate the exact structure of the ER diagram (with constraints).

2. **Import CSVs into SQL**
   - Load cleaned Excel files using LOAD DATA or MySQL Workbench import feature.
3. **Inner Join for Order Details**
   - Join orders, order_items, and products to display detailed line items.
4. **Total Sales by Store**
   - Write a query to group sales (total_price) by each store_id.
5. **Top 5 Selling Products**
   - Use ORDER BY and LIMIT to get the top 5 most sold products by quantity.
6. **Customer Purchase Summary**
   - For each customer, return total orders placed, total items purchased, and total revenue.
7. **Segment Customers by Total Spend**
   - Write a query to classify customers into spending brackets (e.g., low, medium, high).
8. **Staff Performance Analysis**
   - Analyze total revenue generated by each staff member based on their handled orders.
9. **Stock Alert Query**
   - Write a query to list products where stock quantity < 10 in any store.
10. **Create Final Segmentation Table**
    - Create a table customer_segments that will be populated from Python ML results later.

Phase 3: Python and ML Tasks

Mandatory Tasks

1. **Load Data from SQL**
    - Use pandas.read_sql() to pull the orders, order_items, and customers tables into a DataFrame.
2. **Basic EDA (Exploratory Data Analysis)**
    - Use df.describe(), df.info(), and df.value_counts() to summarize the dataset.
    - Plot basic charts using matplotlib or seaborn.
3. **Calculate RFM Features for Customers**
    - Compute Recency, Frequency, and Monetary values for each customer.
    - Recency: Days since last order
    - Frequency: Number of orders
    - Monetary: Total value of all purchases
4. **Export Segmentation Results to SQL**
    - Save the rfm_data with the segment label to SQL as customer_segments table.

Phase 4: Power BI – Visualization & Dashboarding
Mandatory Tasks

1. **Connect Power BI to SQL**
    - Import key tables like orders, products, order_items, customers, stores, and staffs from the SQL database.
2. **Create Relationships Between Tables**
    - Use Model View to define foreign key relationships based on the ER diagram (e.g., link orders.customer_id to customers.customer_id).
3. **Sales Overview Report**
    - Create visuals for:
        - Total sales over time (line or area chart)

        - Monthly sales trend
        - Total orders placed

4. **Top Products by Sales**
    - Bar chart or table listing top-selling products by revenue or quantity.
5. **Customer Purchase Analysis**
    - Add a stacked bar chart to show purchase patterns by city or state using the customers and orders table.
6. **Sales by Store Map**
    - Use the map visual with store.state or store.zip_code to show sales distribution geographically.
7. **Low Stock Alert Dashboard**
    - Use conditional formatting and cards to display products with stock levels below a threshold (e.g., 10 units).
8. **Interactive Filters and Slicers**
    - Add slicers for:
        - Order Status
        - Product Category
        - Store
9. **Staff Performance Report**
    - Table or chart showing total sales/revenue per staff member based on handled orders.
10. **Consolidated Dashboard Page**
    - Final report page with KPIs:
    - Total Revenue
    - Active Customers

   - Avg Order Value
   - Total Orders

11. **Import customer_segments Table**
    - Load the segmentation result (exported from Python) from SQL into Power BI.

12. **Visualize Customer Segments**
    - Use Pie Chart / Donut / TreeMap to show the distribution of segments (e.g., Loyal, New, At-Risk).

13. **Segment-Level Revenue Breakdown**
    - Add a bar chart that shows total revenue per customer segment.

14. **Use Segments as Report Filters**
    - Enable filtering across dashboards by customer segment using slicers.

Results:

- Fully cleaned, relational dataset stored in SQL
- Insightful analytics using Python
- Segmentation-based insights using ML
- Interactive dashboard in Power BI
- Complete ETL pipeline from Excel to Power BI

Project Evaluation Metrics:

- Accuracy of cleaned and processed data
- SQL query proficiency (joins, aggregations, filters)
- Quality of Python EDA and ML models (if used)
- Insight depth and relevance
- Power BI dashboard interactivity and design
- Code structure, comments, and modularity

Technical Tags:

Excel, SQL, Python, Pandas, NumPy, Power BI, KMeans, EDA, ETL, Data Cleaning, Retail Analytics

Data Set: [Retail Sales Data]

Name: Retail Sales Data

Source: Provided schema diagram

Format: Relational Table Schema (to be exported to CSV/Excel and loaded to SQL)

Tables:

- customers
- orders
- order_items
- products
- brands
- categories
- stocks
- stores
- staffs

Data Set Explanation:

- **customers**: Buyer information including contact and location
- **orders:** Transaction log with staff and store references
- **order_items:** Detailed item info per order
- **products:** Item details with category and brand
- **brands/categories:** Metadata about items
- **stocks:** Inventory across stores
- **stores/staffs:** Organizational structure

Preprocessing Tasks in Excel:

- Validate foreign key relationships
- Standardize text formats

- Fill missing values (e.g., zip codes, dates)

Project Deliverables:

- Cleaned Excel files
- SQL database creation scripts and query scripts
- Python notebook or script with analysis and ML
- Power BI .pbix file with visuals
- Final report documenting steps, insights, and visuals
- Project Guidelines:
- Use proper naming conventions and comments in SQL/Python
- Keep backups of Excel and SQL stages
- Use GitHub or local versioning (if collaborative)
- Follow visualization best practices (label axes, use tooltips, limit clutter)
- Submit documentation in PDF or markdown format

Project Authorization: Vinsup Team

Project Created By	Project Approved by
M S Gaurav Kumar (Skill Mentor – DA Vinsup Skill Academy)	**Pooranam Annamalai** (Chief Business & Production Officer Vinsup Skill Academy)