▯ Low-Level Design (LLD) – Retail Sales Analysis

Difficulty Level: Easy | **Total Marks:** 20
Standards Followed: 4 Functions | 4 Visible Test Cases

▯ Summary of Corrections (Based on SME Feedback)

- ▯ Followed strict ordering of `groupby → aggregation → order → limit` in operations
- ▯ Corrected category and product logic to use accurate Pandas flow
- ▯ Function blocks now show all: Prototype, Input, Output, Implementation Flow
- ▯ Ensured output structure matches test case expectations
- ▯ Sample data and expected result formats clarified

▯ Concepts Tested

▯ Pandas CSV Handling
▯ Data Aggregation with `groupby()`
▯ Sorting using `.sort_values()`
▯ Top-N selection using `.head(n)`
▯ Tuple/List formatting from grouped results

▯ Problem Statement

You are given a retail transaction CSV file that tracks purchases by customers.
Your task is to perform various types of sales analysis using **Pandas**, including:

- Loading data
- Calculating total revenue
- Finding the best-selling product category
- Listing top 3 products sold

▯ Operations

▯ *1. Load Transactions*

▯ Loads the CSV file into a Pandas DataFrame.

▯ **Function Prototype:**

```
def load_transactions(file_path: str) -> pd.DataFrame:
```

▯ **Input:** `"sales.csv"`
▯ **Output:** DataFrame

▯ **Implementation Flow:**

- Use `pd.read_csv(file_path)`
- Return the full DataFrame

▯ *2. Total Purchase Value*

▯ Calculates total purchase value across all transactions.

▯ **Function Prototype:**

```
def total_purchase_value(df: pd.DataFrame) -> float:
```

▯ **Input:** DataFrame with `quantity` and `unit_price`
▯ **Output:** Float – sum of `quantity * unit_price`

▯ **Implementation Flow:**

- Add new column `revenue = quantity * unit_price`
- Use `.sum()` on `revenue` column
- Return total purchase value

▯ *3. Find Top Product Category*

▯ Find the product category with highest total sales value.

▯ **Function Prototype:**

```
def top_product_category(df: pd.DataFrame) -> tuple:
```

▯ **Input:** DataFrame
▯ **Output:** Tuple – `(category_name, total_revenue)`

▯ **Implementation Flow:**

- ▯ GroupBy `product_category`
- ▯ Compute `sum(quantity * unit_price)`
- ▯ Order by descending total revenue
- ▯ Limit to top 1
- ▯ Return as tuple

▯ *4. Get Top 3 Products by Units Sold*

▯ Find the top 3 best-selling products based on quantity.

▯ **Function Prototype:**

```
def top_n_products(df: pd.DataFrame) -> list:
```

▯ **Input:** DataFrame
▯ **Output:** List of tuples – `[(product_name, total_quantity), ...]`

▯ **Implementation Flow:**

- ▯ GroupBy `product_name`
- ▯ Aggregate `count or sum(quantity)`
- ▯ Sort descending
- ▯ Limit to top 3
- ▯ Return as list of tuples

□ Implementation Code

```
# □ Implementation Hints for Retail Sales Analysis
import pandas as pd

class RetailSalesAnalyzer:

    def load_transactions(self, file_path: str) -> pd.DataFrame:
        """
        Loads a CSV file and returns it as a Pandas DataFrame.
        □ Use: pd.read_csv()
        """
        pass  # TODO: Implement logic

    def total_purchase_value(self, df: pd.DataFrame) -> float:
        """
        Calculates total sales value from quantity * unit_price.
        □ Add a new column for revenue
        □ Return the sum of the revenue column
        """
        pass  # TODO: Implement logic

    def top_product_category(self, df: pd.DataFrame) -> tuple:
        """
        Finds the top-selling category by revenue.
        □ Use groupby on 'product_category'
        □ Multiply quantity and price to get total revenue
        □ Sort and return the highest category with revenue
        """
        pass  # TODO: Implement logic

    def top_n_products(self, df: pd.DataFrame) -> list:
        """
        Returns a list of top 3 products sold by quantity.
        □ Group by product_name
        □ Sum the quantities
        □ Sort and return top 3 as list of tuples
        """
        pass  # TODO: Implement logic
```

□ Test Cases & Marks Allocation

Test Case ID	Description	Associated Function	Marks
TC1	Load CSV into DataFrame	`load_transactions()`	□ 5
TC2	Calculate total purchase value	`total_purchase_value()`	□ 5
TC3	Find top product category	`top_product_category()`	□ 5
TC4	Get top 3 products by quantity sold	`top_n_products()`	□ 5

□ **Total Marks:** 20

□ Visible Test Cases (4)

□ *TC1: Load CSV File*

Input: `"sales.csv"`
Expected Output: Valid DataFrame (non-empty with expected columns)

TC2: Total Purchase Calculation

```
df = load_transactions("sales.csv")
total_purchase_value(df)
```

Expected Output: `128500.75`

TC3: Best Category

```
df = load_transactions("sales.csv")
top_product_category(df)
```

Expected Output: `("Grocery", 42000.0)`

TC4: Top 3 Products

```
df = load_transactions("sales.csv")
top_n_products(df)
```

Expected Output:

```
[("Rice", 430), ("Notebook", 420), ("Soap", 390)]
```