Master DataFrame Assignment – Retail Sales Superstore Dataset (All-In-One)

We'll use a **well-structured retail dataset** like the *Superstore Sales* dataset (mini version provided below).

Step 1: Sample Dataset (create CSV)

Save as `superstore.csv`:

```
OrderID,OrderDate,Customer,Segment,Region,Product,Category,SubCategory,Quantity,UnitPric

CA-1001,2023-01-15,Ravi,Consumer,South,Laptop,Technology,Computers,1,55000,0.10,5000
CA-1002,2023-02-
20,Priya,Corporate,North,Printer,Technology,Peripherals,2,12000,0.15,1800
CA-1003,2023-01-25,Amit,Consumer,East,Notebook,Office Supplies,Paper,3,200,0.05,150
CA-1004,2023-03-01,Anita,Home Office,West,Table,Furniture,Tables,1,18000,0.20,-1500
CA-1005,2023-02-05,Divya,Consumer,South,Phone,Technology,Phones,2,20000,0.00,3000
```

TASKS ACROSS Pandas, PySpark, and Dask

PART 1: Pandas DataFrame Operations

1. Load the CSV using `pandas`.
2. Print schema, head, shape, dtypes.
3. Select `Customer`, `Product`, `Profit` columns.
4. Filter orders where `Profit > 2000` and `Discount = 0`.
5. Sort by `Profit` descending.
6. GroupBy `Category` → Total Profit, Avg Discount.
7. Add a column `TotalPrice = Quantity * UnitPrice`.
8. Drop the `SubCategory` column.
9. Fill nulls in Discount with 0.10.
10. Apply a function to categorize orders:

```python
def classify(row):
    if row['Profit'] > 4000:
        return 'High'
    elif row['Profit'] > 0:
        return 'Medium'
    else:
        return 'Low'
```

PART 2: PySpark DataFrame Operations

1. Load the same CSV using PySpark.

2. Show schema and first 5 rows.

3. Select columns, Rename `Customer` → `Client`.

4. Filter `Segment = 'Consumer'` and `Profit < 1000`.

5. GroupBy `Region` and show average profit.
6. Use `withColumn` to create `TotalPrice = Quantity * UnitPrice`.
7. Use `when().otherwise()` to classify Profit as:
    - `'Profit' > 2000` → `'High'`
    - `'Profit' <= 0` → `'Loss'`
    - else `'Medium'`
8. Use `drop()` to remove `SubCategory`.
9. Handle nulls in Discount using `fillna(0.10)`.
10. Convert `OrderDate` to `date` type and extract `year`, `month`.

PART 3: Dask DataFrame Operations (Pandas Alternative)

1. Install Dask:

```
!pip install dask
```

2. Load the same `superstore.csv`:

```python
import dask.dataframe as dd
df = dd.read_csv('superstore.csv')
```

3. Do the following:

- Compute average discount by category.
- Filter orders with more than 1 quantity and high profit.
- Save filtered data to new CSV.

PART 4: JSON Handling (Complex Nested)

1. Create a nested JSON file:

```json
[
  {
    "OrderID": "CA-1001",
    "Customer": {"Name": "Ravi", "Segment": "Consumer"},
    "Details": {"Region": "South", "Profit": 5000}
  },
  {
    "OrderID": "CA-1002",
    "Customer": {"Name": "Priya", "Segment": "Corporate"},
    "Details": {"Region": "North", "Profit": 1800}
  }
]
```

2. Load it using PySpark:

```python
df_json = spark.read.json('orders.json', multiLine=True)
df_json.printSchema()
df_json.select("OrderID", "Customer.Name", "Details.Profit").show()
```