PySpark + Spark SQL Task Sheet

Goal: Cover all major PySpark + Spark SQL topics **Data:** Customers and Orders (prepared below)

Step 1: Data Preparation

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import expr

spark =
SparkSession.builder.appName("PracticeProject").enableHiveSupport().getOrCreate()

# Customers Data
customers_data = [
    (101, 'Ali', 'ali@gmail.com', 'Mumbai', '2022-05-10'),
    (102, 'Neha', 'neha@yahoo.com', 'Delhi', '2023-01-15'),
    (103, 'Ravi', 'ravi@hotmail.com', 'Bangalore', '2021-11-01'),
    (104, 'Sneha', 'sneha@outlook.com', 'Hyderabad', '2020-07-22'),
    (105, 'Amit', 'amit@gmail.com', 'Chennai', '2023-03-10'),
]

orders_data = [
    (1, 101, 'Laptop', 'Electronics', 2, 50000.0, '2024-01-10'),
    (2, 101, 'Mouse', 'Electronics', 1, 1200.0, '2024-01-15'),
    (3, 102, 'Tablet', 'Electronics', 1, 20000.0, '2024-02-01'),
    (4, 103, 'Bookshelf', 'Furniture', 1, 3500.0, '2024-02-10'),
    (5, 104, 'Mixer', 'Appliances', 1, 5000.0, '2024-02-15'),
    (6, 105, 'Notebook', 'Stationery', 5, 500.0, '2024-03-01'),
    (7, 102, 'Phone', 'Electronics', 1, 30000.0, '2024-03-02'),
]

customers_df = spark.createDataFrame(customers_data, ["CustomerID", "Name", "Email",
"City", "SignupDate"])
orders_df = spark.createDataFrame(orders_data, ["OrderID", "CustomerID", "Product",
"Category", "Quantity", "Price", "OrderDate"])

# Write as tables
customers_df.write.mode("overwrite").saveAsTable("sales.customers")
orders_df.write.mode("overwrite").saveAsTable("sales.orders")
```

Tasks to Assign

SECTION A: PySpark DataFrame Tasks

1. **Add a column** `TotalAmount = Price * Quantity` to the `orders_df`.
2. **Filter** all orders with `TotalAmount > 10000`.
3. **Standardize** the `City` field in `customers_df` (e.g., lowercase).

4. **Extract year** from `OrderDate` and add a new column `OrderYear` .
5. **Fill null values** in any column of your choice with defaults.
6. **Use `when/otherwise`** to categorize orders:
   - `<5000` : "Low"
   - `5000-20000` : "Medium"
   - `>20000` : "High"

SECTION B: Spark SQL Tasks

7. Run a SQL query to list **all orders made by “Ali”**.
8. Get **total spending by each customer** using SQL.
9. Find out **which category made the highest total revenue**.
10. Create a **view** `customer_orders` showing `CustomerName, Product, TotalAmount` .
11. Query the view for **products ordered after Feb 2024**.

SECTION C: Advanced Practice

12. **Create a Global Temp View** from `customers_df` , then query it using:

```sql
SELECT * FROM global_temp.customers WHERE City = 'Mumbai';
```

13. **Save** the transformed `orders_df` (with TotalAmount) to a **Parquet** file.
14. **Read back** the Parquet file and count how many orders are in it.

SECTION D: UDF + Built-in Function Tasks

15. **Write a UDF** that masks emails like: `ali@gmail.com → a***@gmail.com` .
16. **Use `concat_ws()`** to create a full label like: `'Ali from Mumbai'` .
17. Use `regexp_replace()` to remove special characters from product names.
18. Use `to_date()` and `datediff()` to calculate customer age in days (from SignupDate to today).
