
🚀 Capstone Exercise – *E-Commerce Analytics with PySpark*

Step 1: Create DataFrames

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("ECommerce-Capstone").getOrCreate()

# Customers
customers_data = [
    (1, "Rahul Sharma", "Bangalore", 28),
    (2, "Priya Singh", "Delhi", 32),
    (3, "Aman Kumar", "Hyderabad", 25),
    (4, "Sneha Reddy", "Chennai", 35),
    (5, "Arjun Mehta", "Mumbai", 30),
    (6, "Divya Nair", "Delhi", 29)
]
customers_cols = ["customer_id", "name", "city", "age"]
customers_df = spark.createDataFrame(customers_data, customers_cols)

# Products
products_data = [
    (101, "Laptop", "Electronics", 55000),
    (102, "Mobile", "Electronics", 25000),
    (103, "Headphones", "Electronics", 3000),
    (104, "Chair", "Furniture", 5000),
    (105, "Book", "Stationery", 700),
    (106, "Shoes", "Fashion", 2500)
]
products_cols = ["product_id", "product_name", "category", "price"]
products_df = spark.createDataFrame(products_data, products_cols)

# Orders
orders_data = [
    (1001, 1, 101, 1),
    (1002, 2, 102, 2),
    (1003, 1, 103, 3),
    (1004, 3, 104, 1),
    (1005, 5, 105, 5),
    (1006, 6, 106, 2),
    (1007, 7, 101, 1)  # Order with non-existent customer
]
orders_cols = ["order_id", "customer_id", "product_id", "quantity"]
orders_df = spark.createDataFrame(orders_data, orders_cols)

customers_df.show()
products_df.show()
orders_df.show()
```

Step 2: Exercises (Capstone Tasks)

Basic Operations

1. Select all customer names and their cities.
2. List all distinct product categories.
3. Filter customers older than 30.

Aggregations

4. Find the total number of orders placed per customer.
5. Find the average age of customers per city.
6. Calculate the total revenue generated from each product.

Joins

7. Join customers with orders to list which customer bought what.
8. Join orders with products to get order details with product name and price.
9. Find all customers who have **never placed an order**.
10. Find all products that have **never been ordered**.

Sorting & Grouping

11. Show the top 3 most expensive products purchased.
12. Group orders by category and calculate total revenue per category.
13. List customers sorted by total money spent (highest first).

SQL Queries

14. Register all three DataFrames as temp views (`customers`, `products`, `orders`).
15. Write a query to find the **top 2 cities by total revenue**.
16. Write a query to find customers who spent more than ₹50,000 in total.
17. Write a query to find which **product category contributes the most revenue**.
