

---

## BUSINESS CONTEXT

You are a **Data Engineer** working for a **multi-category online marketplace**.

The business team wants:

- Clean, analytics-ready data
  - City and category performance insights
  - Optimized Spark execution
  - Data stored efficiently for BI tools
- 

## DATASET

### RAW ORDERS DATA

```
orders_data = [  
    ("ORD001", "C001", "Delhi ", "Electronics", "Laptop", "45000", "2024-01-05"  
    ("ORD002", "C002", "Mumbai", "Electronics", "Mobile ", "32000", "05/01/2024"  
    ("ORD003", "C003", "Bangalore", "Electronics", "Tablet", "30000", "2024/01/  
    ("ORD004", "C004", "Delhi", "Electronics", "Laptop", "", "2024-01-07", "Canc  
    ("ORD005", "C005", "Chennai", "Electronics", "Mobile", "invalid", "2024-01-  
    ("ORD006", "C006", "Mumbai", "Home", "Mixer", None, "2024-01-08", "Completec  
    ("ORD007", "C001", "Delhi", "Electronics", "Laptop", "47000", "09-01-2024",  
    ("ORD008", "C007", "Bangalore", "Home", "Vacuum", "28000", "2024-01-09", "Cc  
    ("ORD009", "C002", "Mumbai", "Electronics", "Laptop", "55000", "2024-01-10"  
    ("ORD010", "C008", "Delhi", "Home", "AirPurifier", "38000", "2024-01-10", "C  
    ("ORD011", "C009", "Mumbai", "Home", "Vacuum", "29000", "2024-01-11", "Compl  
    ("ORD012", "C010", "Bangalore", "Electronics", "Mobile", "33000", "2024-01-  
    ("ORD013", "C003", "Bangalore", "Home", "Mixer", "21000", "2024-01-12", "Con  
    ("ORD014", "C004", "Delhi", "Electronics", "Tablet", "26000", "2024-01-12",  
    ("ORD015", "C005", "Chennai", "Electronics", "Laptop", "62000", "2024-01-13"  
    ("ORD016", "C006", "Mumbai", "Home", "AirPurifier", "40000", "2024-01-13", "  
    ("ORD017", "C007", "Bangalore", "Electronics", "Laptop", "51000", "2024-01-  
    ("ORD018", "C008", "Delhi", "Home", "Vacuum", "31000", "2024-01-14", "Comple  
    ("ORD019", "C009", "Mumbai", "Electronics", "Tablet", "29000", "2024-01-15"  
    ("ORD020", "C010", "Bangalore", "Electronics", "Laptop", "54000", "2024-01-  
    ("ORD020", "C010", "Bangalore", "Electronics", "Laptop", "54000", "2024-01-
```

]

# Columns

```
order_id  
customer_id  
city  
category  
product  
amount  
order_date  
status
```

---

## EXERCISE OBJECTIVES

---

## PHASE 1 – SCHEMA & INGESTION

Topics:

StructType, StructField, data types

### Tasks

1. Define an explicit schema
  2. Create a DataFrame using the schema
  3. Print and verify schema
- 

## PHASE 2 – DATA CLEANING

Topics:

Column operations, replace, cast, filter

### Tasks

4. Trim all string columns
5. Standardize `city`, `category`, and `product` values
6. Convert `amount` to IntegerType
7. Handle invalid, empty, and null `amount` values

8. Convert `order_date` into `DateType` (handle multiple formats)
  9. Remove duplicate `order_id` records
  10. Keep only `Completed` orders
- 

## PHASE 3 – DATA VALIDATION

Topics:

Debugging, validation checks

### Tasks

11. Count records before and after cleaning
  12. Verify no nulls in `order_id`, `amount`, and `order_date`
  13. Confirm correct data types
- 

## PHASE 4 – ANALYTICS & AGGREGATIONS

Topics:

`GroupBy`, aggregate functions

### Tasks

14. Total revenue per city
  15. Total revenue per category
  16. Total revenue per product
  17. Average order value per city
  18. Identify top 3 products by revenue
- 

## PHASE 5 – WINDOW FUNCTIONS

Topics:

`Window`, `rank`, `dense_rank`

### Tasks

19. Rank cities by total revenue

- 20. Rank products within each category by revenue
  - 21. Identify the top product per category
- 

## PHASE 6 – PERFORMANCE AWARENESS

Topics:

Cache, explain, partitions

### Tasks

- 22. Cache the cleaned DataFrame
  - 23. Run multiple aggregations and observe behavior
  - 24. Use `explain(True)` to inspect shuffle and execution plan
  - 25. Repartition data by `city` and explain why
- 

## PHASE 7 – FILE FORMAT OUTPUT

Topics:

Parquet, ORC

### Tasks

- 26. Write cleaned order-level data to **Parquet**
  - 27. Write aggregated analytics to **ORC**
  - 28. Read both back and validate schema
- 

## PHASE 8 – DEBUGGING CHECK

Topics:

Common PySpark mistakes

### Tasks

- 29. Explain why this line is incorrect:

```
df = df.filter(df.amount > 30000).show()
```

30. Write the corrected version

---