🧠 BUSINESS CONTEXT

You are a **Data Engineer** working for a multi-city retail chain.

The company wants to:

- Analyze sales performance
- Optimize data storage
- Improve query performance
- Prepare data for analytics & ML teams
- Ensure scalability for future streaming systems

Your job is to **design, clean, optimize, store, and analyze data using PySpark**.

📂 DATASETS PROVIDED (RAW & DIRTY)

You will be given **three datasets**, intentionally messy.

DATASET 1 — SALES TRANSACTIONS (CSV)

```
sales_data = [
    ("TXN001","Delhi ","Laptop","Electronics","45000","2024-01-05","Compl
    ("TXN002","Mumbai","Mobile ","electronics","32000","05/01/2024","Comp
    ("TXN003","Bangalore","Tablet"," Electronics ","30000","2024/01/06","
    ("TXN004","Delhi","Laptop","Electronics","","2024-01-07","Cancelled")
    ("TXN005","Chennai","Mobile","Electronics","invalid","2024-01-08","C
    ("TXN006","Mumbai","Tablet","Electronics",None,"2024-01-08","Complet
    ("TXN007","Delhi","Laptop","electronics","45000","09-01-2024","Compl
    ("TXN008","Bangalore","Mobile","Electronics","28000","2024-01-09","C
    ("TXN009","Mumbai","Laptop","Electronics","55000","2024-01-10","Compl
    ("TXN009","Mumbai","Laptop","Electronics","55000","2024-01-10","Compl
]
```

DATASET 2 — CUSTOMER MASTER (JSON)

```
customer_data = [
    ("C001","Delhi","Premium"),
    ("C002","Mumbai","Standard"),
    ("C003","Bangalore","Premium"),
    ("C004","Chennai","Standard"),
    ("C005","Mumbai","Premium")
]
```

DATASET 3 — CITY CLASSIFICATION (LOOKUP)

```
city_lookup = [
    ("Delhi","Tier-1"),
    ("Mumbai","Tier-1"),
    ("Bangalore","Tier-1"),
    ("Chennai","Tier-2")
]
```

🎯 CAPSTONE OBJECTIVES (WHAT THEY MUST BUILD)

PHASE 1 — DATA INGESTION & SCHEMA MANAGEMENT

Topics covered:

- StructType / StructField
- Data types
- Corrupt data handling

Tasks

1. Create schemas explicitly for all datasets
2. Load raw data into DataFrames
3. Handle incorrect data types gracefully
4. Identify corrupt and invalid records

PHASE 2 — DATA CLEANING & TRANSFORMATION

Topics covered:

- Column operations
- Filter, select, withColumn
- String normalization
- Date handling

Tasks

5. Trim and normalize string columns
6. Convert category to uppercase
7. Convert amount to integer
8. Handle invalid and null amounts
9. Parse multiple date formats into DateType
10. Remove duplicate transactions
11. Keep only Completed transactions

PHASE 3 — DATA ENRICHMENT & JOINS

Topics covered:

- Joins
- Broadcast joins
- Explain plan

Tasks

12. Join sales data with city lookup
13. Use broadcast join where appropriate
14. Explain join strategy used
15. Enrich sales data with city tier

PHASE 4 — ANALYTICS & WINDOW FUNCTIONS

Topics covered:

- Aggregations
- Window functions
- Ranking
- Over clause

Tasks

16. Revenue per city
17. Revenue per product
18. Rank cities by total revenue
19. Rank products within each city
20. Identify top-performing city per day

PHASE 5 — CACHING, PARTITIONS & OPTIMIZATION

Topics covered:

- cache / persist
- repartition / coalesce
- shuffles
- explain()

Tasks

21. Identify reusable DataFrames
22. Apply caching appropriately
23. Compare performance with and without cache
24. Repartition data by city
25. Explain why partitioning helps

PHASE 6 — FILE FORMAT STRATEGY

Topics covered:

- Parquet
- ORC
- Avro (conceptual)

Tasks

26. Write cleaned data to Parquet
27. Write aggregated data to ORC
28. Compare file structure and size
29. Explain why Avro is not used here
30. Design a future streaming ingestion using Avro

PHASE 7 — DEBUGGING & ERROR HANDLING (Practice by tampering data)

Topics covered:

- AnalysisException
- NoneType errors
- Debug workflow

Tasks

31. Identify common mistakes (intentional bugs)
32. Debug schema mismatch errors
33. Debug NoneType DataFrame errors
34. Use explain() to identify inefficiencies

PHASE 8 — FINAL VALIDATION & DELIVERABLES

Tasks

35. Validate record counts
36. Ensure no nulls in critical fields
37. Confirm schema correctness
38. Document optimization decisions

📦 EXPECTED OUTPUTS (HIGH LEVEL)

- Clean, analytics-ready dataset
- Optimized storage format
- Correct joins and aggregations
- Clear performance reasoning
- Industry-aligned design choices
