CASE STUDY 5

Title: Online Food Delivery Operations & Delay Optimization using PySpark

A food delivery company wants to improve delivery time, reduce cancellations, and understand which areas and restaurants are operationally weak.
They export their daily operational data as a CSV file:

```
delivery_data.csv
```

Columns:

```
order_id
customer_id
restaurant_id
restaurant_city
delivery_city
order_time
pickup_time
delivery_time
delivery_status
delivery_partner_id
order_amount
payment_mode
rating
```

Where:

- delivery_status = PLACED, PICKED, DELIVERED, CANCELLED
- rating is given only if delivered
- pickup_time and delivery_time can be missing
- order_time formats are inconsistent
- order_amount is string and may contain commas
- Some rows have negative or unrealistic times
- Some deliveries are marked DELIVERED but have no delivery_time

This is classic operational chaos data.

Your job is to build a **Delivery Performance Intelligence Pipeline** using PySpark.

PHASE 1 – Ingestion

1. Read delivery_data.csv as all StringType.

2. Print schema and record count.
3. Show sample rows.
4. Identify obvious data issues.

PHASE 2 – Cleaning

1. Trim all string columns.

2. Clean order_amount:

    - Remove commas
    - Convert to IntegerType
    - Handle invalid values safely

3. Parse timestamps into:

```
order_time_clean
pickup_time_clean
delivery_time_clean
```

Support multiple formats:

- yyyy-MM-dd HH:mm:ss
- dd/MM/yyyy HH:mm:ss
- yyyy/MM/dd HH:mm:ss

Keep original columns for audit.

4. Create flags:

```
order_time_valid
pickup_time_valid
delivery_time_valid
```

PHASE 3 – Derived Metrics

Create time-based metrics:

```
prep_time    = pickup_time - order_time
delivery_time_gap = delivery_time - pickup_time
total_fulfillment_time = delivery_time - order_time
```

All in minutes.

Rows with negative times must be flagged.

PHASE 4 – Data Validation

1. Count rows where:
    - pickup_time < order_time
    - delivery_time < pickup_time
2. Count rows where delivery_status = DELIVERED but delivery_time is null.
3. Count rows with invalid amounts.
4. Count CANCELLED orders.

Deliverable: Operational Data Quality Report.

PHASE 5 – Operational Analytics

1. Average prep_time per restaurant.
2. Average delivery_time_gap per delivery city.
3. Top 10 slowest restaurants.
4. Cities with highest average delivery time.
5. Cancellation percentage per city.

PHASE 6 – Window Functions

1. Rank restaurants by:
    - Average fulfillment time
    - Cancellation rate
2. Rank delivery partners by:
    - Fastest average delivery
    - Highest ratings
3. Identify top 3 and bottom 3 per city.

PHASE 7 – Customer Experience

1. Average rating per restaurant.
2. Relationship between:
    - Delivery time

- Rating

3. Identify if slower deliveries reduce ratings.

PHASE 8 – Performance Engineering

1. Cache the cleaned DataFrame.
2. Use explain(True) on:
   - Restaurant ranking query
   - City delay analysis query
3. Identify shuffle stages.
4. Repartition by delivery_city.
5. Compare plans before and after.

PHASE 9 – RDD

1. Convert delivered orders to RDD.
2. Compute:
   - Total revenue using reduce
   - Order count per city using map-reduce
3. Explain why DataFrames are better here.
