Junior Data Engineer Challenge

Part 1: ETL Pipeline — Real Estate Data Ingestion (Python Script)

This Jupyter Notebook processes real estate transaction data, cleans and transforms it, loads it into a PostgreSQL database, and performs analytical computations to generate insights.

The code includes the following main components:

✓ **Data Cleaning** – Handles null values, duplicates, and inconsistent formats
✓ **Data Transformation** – Standardizes column names, converts data types, and derives new metrics
✓ **PostgreSQL Integration** – Stores processed data in a relational database
✓ **Analytics & Visualization** – Computes key metrics and generates visual reports

Brief breakdown

1. Initial Setup

- **Spark Session Initialization**

  Configures PySpark with PostgreSQL JDBC driver for database connectivity.

  Sets custom port (4041) to avoid conflicts.

- **Reading data file**

  Reads CSV data with proper handling of multiline fields and escaped quotes.

2. Data Cleaning & Transformation (transform())

- **Standardizes Column Names**

  Converts to lowercase and replaces spaces with underscores (e.g., trans_value).

- **Handles Missing & Duplicate Data**

Drops null records in critical fields (e.g., transaction_number). Can add more fields to the subset as required.

Removes duplicate rows.

- **Date & Numeric Formatting**

  Converts instance_date from string to timestamp.

  Extracts separate date_col (yyyy-MM-dd) and time_col (HH:mm:ss).

  Ensures numeric fields (trans_value, procedure_area, actual_area) are rounded to 2 decimal places.

- **Derived Columns**

  Price per sqm: trans_value / procedure_area

  Price per room: Adjusts for "Studio" (counted as 1 room).

  Budget Tier: Classifies transactions as Low/Medium/High.

  Has parking (flag): Converts nulls in parking field to "NO" for clarity.

3. Loading to database table (load())

- **PostgreSQL Connection**

  Uses JDBC to write the cleaned dataframe to the dubai_real_estate_transactions table in UAE_Real_Estate database.

  Overwrites existing data (mode="overwrite"). Can change to “append” as needed

4. Analytics (analyze())

- **Average Price by Region**

  Groups by area_en and computes mean transaction value.

  Exports results to CSV and generates a bar chart (avg_price_per_region.png).

- **Transactions per Month**

  Aggregates transactions by year-month.

  Produces a line chart (transactions_per_month.png).

- **Highest and Lowest Priced Properties**

  Identifies extreme values in trans_value.

  Saves results to CSV for further review.

For usage instructions, see inline documentation in the notebook. It will tell you where to change file paths, database connection information and other parameters.

Analysis Results Plots