Task Assessment: ETL Pipeline with Excel Input, Python, MySQL, and PostgreSQL

Objective:

You are required to build a complete **ETL (Extract, Transform, Load)** pipeline using **Python**, based on an Excel file containing raw customer data. The purpose of this task is to assess your ability to read data, clean and organize it, and store it into relational databases, along with generating metadata.

Task Details:

1. Extract:

- Load the provided Excel file (`messy_customers_data.xlsx`) using Python (e.g., `pandas`).
- Ensure the data contains the following fields: `name`, `gender`, `email`, `signup_date`, `address`, `country`, `department`, `designation`.

---

2. Transform (Clean the Data):

Perform the following data cleaning steps:

- Remove **null values** and **duplicate rows**.
- Standardize the **date format** in `signup_date` to `YYYY-MM-DD`.
- Strip **leading/trailing whitespace** from all string fields.
- Convert relevant text fields to **lowercase** (e.g., email, address, designation).
- Ensure consistency in `country` values (should be only `usa`, `uk`, `india` in lowercase).

Split the cleaned data by **country**, and save them as CSV files inside a dynamically created folder structure:

CopyEdit

```
final_data/
├── country_usa.csv
├── country_uk.csv
└── country_india.csv
```

---

3. Load:

- **Create a MySQL database** (e.g., `customer_us_db`) and create a table for US customer data. Load all cleaned USA customer records into it.
- **Create a PostgreSQL database** (e.g., `customer_global_db`) and create a table for UK and India customers. Load all cleaned UK and India customer records into it.

---

4. Metadata Generation:

- Create **metadata** for the processed data. This metadata should include:
    - Number of records processed.
    - Number of records per country.
    - Timestamp of processing.
    - Column data types and sample values.
- You may choose to:
    - Store this metadata in a **MySQL table**, or
    - Export it as a structured **JSON** file (`metadata.json`).

Requirements:

- Use **pandas** for data processing.

- Use **mysql-connector-python** or **SQLAlchemy** for MySQL operations.
- Use **psycopg2** or **SQLAlchemy** for PostgreSQL operations.
- Organize your code into **functions or classes** for modularity.
- Handle exceptions and include **basic logging** if possible.
- Dynamically create directories (`final_data/`) if not present.

Deliverables:

1. Cleaned CSV files inside the `final_data/` directory.
2. Python script(s) used for ETL.
3. MySQL and PostgreSQL database schemas.
4. Metadata (either as JSON or inside MySQL).
5. README file (optional, but appreciated) describing how to run the script and prerequisites.