Senior Data Engineer Code Challenge

Data-Engineer Code Challenge: Snowflake ETL with Prefect & MinIO (Focused)

Objective

Build a small ETL pipeline that demonstrates **Snowflake expertise**, including:

- Handling CSV data
- Detecting column types and casting when creating tables
- Handling one **JSON / semi-structured column** with VARIANT
- Incremental updates into a parent table
- Creating **filtered subsets** (tables or secure views)

All workflows run locally via **Docker Compose** and MinIO (no AWS account needed and faster local testing).

Scope

Must-haves:

1. **Docker Compose**
    - Prefect + MinIO running locally.
    - Candidate can upload and download CSV files from MinIO.
2. **Load CSV → Snowflake (Parent Table)**
    - Detect column types (int, float, string, date, VARIANT for JSON).
    - Create the parent table with the correct types.
    - Load data using **staging table + MERGE**.
    - Re-running the flow with updated CSV must **reflect inserts/updates**.

    - No need to handle deletes or complex transformations.

3. **Subset Tables / Views**
    - Apply **one or two filters** (e.g., country = 'DE', event_type = 'signup').
    - Flatten JSON/VARIANT column as required.
    - Create subset tables or secure views.
4. **Documentation**
    - Short README explaining:
        - How to run the project and prepare necessary configiguartions (eg. on Snowflake)
        - How to configure MinIO and Snowflake credentials
        - How to run Prefect flows

Optional / Extra Credit

- Use **secure views** instead of tables for subsets.
- Include **data validation** (row counts, sample values).

Links for help

- Snowflake: Free trial sign-up — https://signup.snowflake.com/ (signup.snowflake.com)
- MinIO quick-start guide — https://docs.min.io/quickstart/overview.html (or see Docker setup guide) (charts.min.io)

Sample CSV structure

```
id,name,country,event_type,event_date,event_metadata
1,John,DE,signup,2025-01-01,"{\"user_id\": 123, \"session_duration\": 45}"
2,Maria,FR,purchase,2025-01-02,"{\"user_id\": 456, \"amount\": 120.5}"
```

Filters Example:

```yaml
filters:
  - name: germany_events
    where: "country = 'DE'"
    flatten: ["event_metadata.user_id", "event_metadata.session_duration"]
  - name: recent_signups
    where: "event_type = 'signup'"
```