CIRCLE 97
TASK ONE.

In this project, we are working with two JSON folders and are expected to derive two CSV files as the output. The files are kept in two folders, "cards" and "users," containing JSON folders of numerous lengths.
The project must read the folders and load the multiple JSON files into Python. Also, the JSON file type will then be converted into a Python dictionary.
The Python dictionaries from the two folders will be saved as cards.csv and users.csv using the CSV.writer module, as seen in the screenshots of the Python script below.

events.py > ...

```python
# import libraries needed on this project
import os
import json
import csv



def load_json_files_from_folder(folder_path):
    json_data = []

    for file_name in os.listdir(folder_path):
        if file_name.endswith(".json"):
            file_path = os.path.join(folder_path, file_name)
            with open(file_path, "r") as f:
                json_data.append(json.load(f))

    return json_data

# Get a list of all JSON files in the folder.
folder_path_users = ("C:/Users/Olayinka Akerekan/Desktop/altschool/events/events/users/")
folder_path_cards = ("C:/Users/Olayinka Akerekan/Desktop/altschool/events/events/cards/")

# Load the json data into variables for users and cards
users = load_json_files_from_folder(folder_path_users)
cards = load_json_files_from_folder(folder_path_cards)

# Process user data
user_lists = []
for user in users:
    user_dict = {
        "type": user["metadata"]["type"],
        "event_at": user["metadata"]["event_at"],
        "event_id": user["metadata"]["event_id"],
        "id": user["payload"]["id"],
        "name": user["payload"]["name"],
        "address": user["payload"]["address"],
        "job": user["payload"]["job"],
        "score": user["payload"]["score"]
    }
```

```python
# Process card data
card_lists = []
for card in cards:
    if card["payload"].get("user_id") is not None:
        card_dict = {
            "id": card["payload"]["id"],
            "user_id": card["payload"]["user_id"],
            "created_by_name": card["payload"]["created_by_name"],
            "updated_at": card["payload"]["updated_at"],
            "created_at": card["payload"]["created_at"],
            "active": card["payload"]["active"],
            "type": card["metadata"]["type"],
            "event_at": card["metadata"]["event_at"],
            "event_id": card["metadata"]["event_id"]
        }
        card_lists.append(card_dict)

# Write user data to CSV
with open('users.csv', 'w', newline='') as csv_file:
    fieldnames = user_lists[0].keys()
    writer = csv.DictWriter(csv_file, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(user_lists)

# Write card data to CSV
with open('cards.csv', 'w', newline='') as csv_file:
    fieldnames = card_lists[0].keys()
    writer = csv.DictWriter(csv_file, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(card_lists)
```

This shows how the project will flow.
JSON files in folder -----> open the folder ----> load it into python ---> convert the JSON datatype into a python dictionary --> save as a list of dictionaries ----> use CSV.Writer to write into a CSV file ----> save as "cards.csv" and "users.csv"

Using drawsql.app draw the schema of the new tables formed and save
Import Data obtained into tables using DBeaver

The tables below give a detailed description of each column in the users and cards tables.

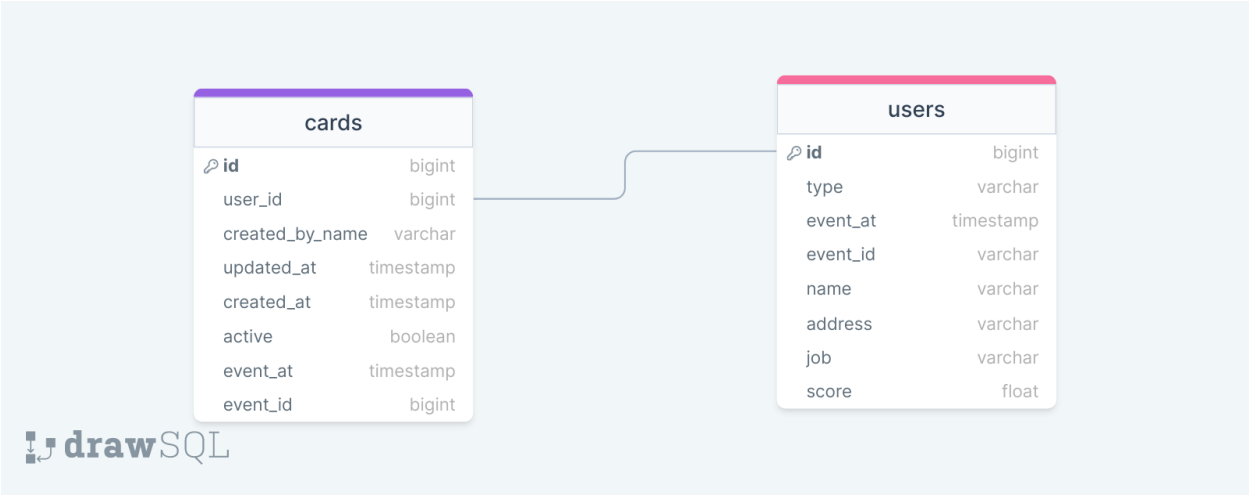
CARDS TABLE:

Column	Data type	Description
id	bigint	An identifier for card event
User_id	bigint	Identifier linking the card to a user
Created_by_name	varchar	Name of the user who created the card
Updated_at	timestamp	The date and time card were updated
Created_at	timestamp	A date and time card were created
active	boolean	Shows active card
Event_at	timestamp	Date and time of card event
Event_id	varchar	Key identifier for card event

USERS TABLE

Column	Data type	Description
id	bigint	Unique identifier for the user
type	varchar	Type of users
Event_at	timestamp	Date and time of user event
Event_id	timestamp	Unique identifier for the user event
name	varchar	Name of user
address	varchar	Address of user
job	varchar	Job of user
score	float	The score for each user

The Drawsql Schema for the users and card events.



We identified common columns that link the two tables to establish the relationship between the ‘users’ and ‘cards’ tables. The ‘user_id’ column in the ‘cards’ table is established as the foreign key, referencing the ‘id’ column of the ‘users’ table as the primary key.