transformdata

October 20, 2023

[108]:
```python
import re

def parse_file(file_path):
    with open(file_path, 'r') as file:
        content = file.read()

    entities = re.split(r'--- (\w+) ---', content)[1:]

    parsed_data = []
    for i in range(0, len(entities), 2):
        entity_type = entities[i].strip()
        entity_data = entities[i + 1].strip().split('\n')

        entity_dict = {"type": entity_type}
        for line in entity_data:
            key, value = re.match(r'"(.+)"=(.+)', line).groups()
            entity_dict[key] = value

        parsed_data.append(entity_dict)

    return parsed_data

file_path = 'generated_entities.txt'
parsed_data = parse_file(file_path)

# Now 'parsed_data' contains a list of dictionaries, each representing an entity
count = 0
for entity in parsed_data:
    print(entity)
    count += 1
    if count == 10:
        break
```

```
{'type': 'Enemy', 'id': '302', 'enemy_name': '"Shadow Stalker"', 'enemy_type':
'"Humanoid"', 'hitpoints': '384', 'warcry': '"xgiisl"'}
{'type': 'GuildName', 'id': '378', 'name': '"Wizards of the Coast"',
'description': '"Mage Training"', 'leader': '"Aethor"', 'members': '483',
'founded_year': '359'}
```

```
{'type': 'Enemy', 'id': '40', 'enemy_name': '"Harpy Scream"', 'enemy_type':
'"Dragon"', 'hitpoints': '4', 'warcry': '"joo"'}
{'type': 'questions', 'id': '121', 'content': '"One does not simply walk into
the End Zone."', 'choice_options': '659', 'emotion': '309'}
{'type': 'GuildName', 'id': '73', 'name': '"Elixir Masters"', 'description':
'"PvP"', 'leader': '"Oromis"', 'members': '127', 'founded_year': '470'}
{'type': 'Character', 'id': '172', 'first_name': '"Thaelis"', 'firstname':
'"Darkbane"', 'race': '"Elf"', 'class': '"Mage"', 'guild': '"Knights of Valor"',
'last_login': '"2021-07-05T02:18:37"'}
{'type': 'NPC', 'id': '101', 'npc_type': '"Quest-givers"', 'first_name':
'"Raelis"', 'last_name': '"Shadowgale"', 'location': '626.99'}
{'type': 'Team', 'id': '61', 'team_name': '"Crimson_Brigade"', 'kingdom':
'"Elphora"', 'n_members': '500'}
{'type': 'GuildName', 'id': '263', 'name': '"Moonlit Mystics"', 'description':
'"Knowledge"', 'leader': '"Nymriel"', 'members': '570', 'founded_year': '525'}
{'type': 'questions', 'id': '193', 'content': '"To be or not to be, that is the
quest."', 'choice_options': '640', 'emotion': '508'}
```

[101]:
```
import mysql.connector
from mysql.connector import Error
from datetime import datetime

# Establish a connection to MySQL
try:
    connection = mysql.connector.connect(
        host='localhost',
        database='Aetheria',
        user='root',
        password='12345678'
    )

    if connection.is_connected():
        print('Connected to MySQL database')

        cursor = connection.cursor()

        # Loop through parsed_data and insert records into the database
        for entity in parsed_data:
            # if entity['type'] == 'Enemy':
            #     query = """INSERT INTO Enemy (ID ,Name, Type, Hitpoints,␣
 ↪Warcry)
            #                 VALUES (%s,%s, %s, %s, %s)"""
            #     values = (int(entity['id']),entity['enemy_name'],␣
 ↪entity['enemy_type'], int(entity['hitpoints']), entity['warcry'])
            #     cursor.execute(query, values)
```

```
            # if entity['type'] == 'GuildName':
            #     print(entity)
            #     query = """INSERT INTO Guild (ID ,GuildName, LeaderName,␣
 ↪Description, NumOfMembers, FoundedYear)
            #                 VALUES (%s,%s, %s, %s, %s,%s)"""
            #     values = (int(entity['id']),entity['name'],␣
 ↪entity['leader'],entity['description'],␣
 ↪int(entity['members']),int(entity['founded_year']))
            #     cursor.execute(query, values)

#              if  entity['type'] == 'Character':

#                 query = """INSERT INTO `Character` (ID , Name, Race , Class)
#                           VALUES (%s,%s, %s, %s)"""
#                 values = (int(entity['id']), entity['first_name'].
 ↪replace('"', "'") ,entity['race'].replace('"', "'"), entity['class'].
 ↪replace('"', "'"))
#                 cursor.execute(query, values)

#             if entity['type'] == 'questions':
#                 query = """INSERT INTO Question (ID, Content, ChoiceOptions,␣
 ↪Emotion)
#                            VALUES (%s, %s, %s, %s)"""
#                 values = (int(entity['id']), entity['content'],␣
 ↪entity['choice_options'], entity['emotion'])
#                 cursor.execute(query, values)

#             ##Insert data into NPC table
#             if entity['type'] == 'NPC':
#                 query = """INSERT INTO NPC (ID, Type, FirstName, LastName,␣
 ↪Location)
#                            VALUES (%s, %s, %s, %s, %s)"""
#                 values = (int(entity['id']), entity['npc_type'],␣
 ↪entity['first_name'], entity['last_name'], entity['location'])
#                 cursor.execute(query, values)

#             # Insert data into Team table
#             if entity['type'] == 'Team':
#                 query = """INSERT INTO Team (ID, TeamName, Kingdom, TeamSize)
#                            VALUES (%s, %s, %s, %s)"""
#                 values = (int(entity['id']), entity['team_name'],␣
 ↪entity['kingdom'], int(entity['n_members']))
#                 cursor.execute(query, values)

#             if entity['type'] == 'Event':
#                 query = """INSERT INTO Event (ID, Name, Time)
```

```
#                               VALUES (%s, %s, %s)"""
#                   values = (int(entity['id']), entity['event_name'],␣
 ↪int(entity['event_time']))
#                   cursor.execute(query, values)

#               if entity['type'] == 'Vendors':
#                   query = """INSERT INTO NPC (ID, Type, FirstName, LastName,␣
 ↪Location)
#                               VALUES (%s, %s, %s, %s, %s)"""
#                   values = (int(entity['id']), entity['npc_type'],␣
 ↪entity['first_name'], entity['last_name'], float(entity['location']))
#                   cursor.execute(query, values)

#               if entity['type'] == 'Item':
#                   query = """INSERT INTO Item (ID, ItemName, ItemType)
#                               VALUES (%s, %s, %s)"""
#                   values = (int(entity['id']), entity['item_name'],␣
 ↪entity['item_type'])
#                   cursor.execute(query, values)

        # Commit the changes
        connection.commit()

except Error as e:
    print(f"Error: {e}")

finally:
    # Close the database connection
    if connection.is_connected():
        cursor.close()
        connection.close()
        print('MySQL connection closed')
```

```
Connected to MySQL database
MySQL connection closed
```

[5]:
```
import re
from datetime import datetime
import mysql.connector

# Establish a connection to the MySQL database
try:
    db_connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="12345678",
        database="Aetheria"
```

```
    )

    # Create a cursor object to execute SQL queries
    with db_connection.cursor() as cursor:
        # Open and read the file
        with open('generated_events.txt', 'r') as file:
            lines = file.read().split('=====\n')

        # Iterate through each event block in the file
        for block in lines:
            # Use regular expressions to extract information from each block
            match = re.match(r'\[Event Type\]: (.+)\n\[Timestamp\]: (.
 ↪+)\n\[Entity1\]: (.+)\n\[Entity2\]: (.+)\n\[Value\]: (.+)', block.strip())

            if match:
                # Extracting matched groups
                event_type = match.group(1)
                timestamp = datetime.fromisoformat(match.group(2))
                entity1 = match.group(3)  # Removed eval
                entity2 = match.group(4)  # Removed eval
                value = match.group(5)

                # SQL query to insert data into the 'Relation' table
                insert_query = "INSERT INTO Relation (EventType, Timestamp,␣
 ↪Entity1, Entity2, Value) VALUES (%s, %s, %s, %s, %s)"

                # Data to be inserted into the table
                data = (event_type, timestamp, entity1, entity2, value)

                # Execute the query
                cursor.execute(insert_query, data)

        # Commit the changes to the database
        db_connection.commit()

except mysql.connector.Error as err:
    print(f"Error: {err}")

finally:
    # Close the database connection outside the try-except block
    db_connection.close()
```

[ ]: