Python Script Documentation

Overview

This Python script is designed to manage user and department data, allowing the user to input new data, display it, search for specific users, and insert data into an SQL Server database. The script interacts with files to store and retrieve data, and it also validates user input to ensure data integrity.

Key Features:

- **File Management**: Read from and write to text files containing user and department information.
- **Data Validation**: Validate user input to maintain consistent and accurate data.
- **Data Search**: Search for specific users by their User ID.
- **SQL Server Integration**: Insert data into an SQL Server database using ODBC.
- **Dictionary Management**: Store data in nested dictionaries for easy access and manipulation.

Modules and Libraries Used

1. **datetime**: Provides classes for manipulating dates and times.
2. **os**: Provides functions for interacting with the operating system.
3. **pyodbc**: Provides a Python DB API 2.0 compliant interface for ODBC databases.

Functions

1. insert_data_from_file(fileName, insertFunction, conn, cursor)

- **Purpose**: Reads data from a file and inserts it into an SQL Server database using a provided insert function.
- **Parameters**:
    - fileName: The name of the file to read data from.
    - insertFunction: The function to be used for inserting data into the database.
    - conn: The database connection object.
    - cursor: The database cursor object.
- **Operation**:
    - Checks if the file exists.

- Reads all lines from the file.
- Calls the provided insert function to insert data into the database.
- Commits the transaction.

2. user_table_insert(lines, cursor)

- **Purpose**: Inserts user data into the Users table in the SQL Server database if the user ID is not already present.
- **Parameters**:
  - lines: List of strings representing lines from the file.
  - cursor: The database cursor object.
- **Operation**:
  - Iterates over the lines and splits them into fields.
  - Checks if the UserID is already in the Users table.
  - If not, inserts the data into the table.

3. department_table_insert(lines, cursor)

- **Purpose**: Inserts department data into the Departments table in the SQL Server database if the department ID is not already present.
- **Parameters**:
  - lines: List of strings representing lines from the file.
  - cursor: The database cursor object.
- **Operation**:
  - Iterates over the lines and splits them into fields.
  - Checks if the DepartmentID is already in the Departments table.
  - If not, inserts the data into the table.

4. insert_into_sql_server()

- **Purpose**: Establishes a connection to an SQL Server database and inserts data from files into the Users and Departments tables.
- **Operation**:
  - Establishes a connection to the SQL Server database using pyodbc.
  - Creates the Users and Departments tables if they do not already exist.

- Calls insert_data_from_file to insert data from the user info.txt and department info.txt files into the respective tables.

5. read_from_file(fileName, read_to, firstStatement, departments)

- **Purpose**: Reads data from a file and updates a dictionary with the data. If the file doesn't exist, it creates it and writes the first statement.
- **Parameters**:
  - fileName: The name of the file to read from.
  - read_to: The function to call to process the lines from the file.
  - firstStatement: The header to write to the file if it's empty.
  - departments: The dictionary to update with the data.
- **Operation**:
  - Checks if the file exists.
  - If it exists, reads the lines and processes them using read_to.
  - If the file is empty, writes firstStatement to it.

6. read_from_userInfo_file_to_dictionary(departments, lines)

- **Purpose**: Processes lines from the user info file and updates the departments dictionary with user data.
- **Parameters**:
  - departments: The dictionary to update with the data.
  - lines: List of strings representing lines from the file.
- **Operation**:
  - Iterates over the lines and splits them into fields.
  - Updates the departments dictionary with user data.

7. read_from_departmentInfo_to_dictionary(departments, lines)

- **Purpose**: Processes lines from the department info file and updates the departments dictionary with department data.
- **Parameters**:
  - departments: The dictionary to update with the data.
  - lines: List of strings representing lines from the file.
- **Operation**:

- Iterates over the lines and splits them into fields.
- Updates the departments dictionary with department data.

8. write_to_file(fileName, line)

- **Purpose**: Appends a line of text to a file.
- **Parameters**:
  - fileName: The name of the file to write to.
  - line: The line of text to append to the file.

9. get_valid_input(prompt, validation_func, error_message)

- **Purpose**: Repeatedly prompts the user for input until a valid input is provided based on a validation function.
- **Parameters**:
  - prompt: The prompt to display to the user.
  - validation_func: The function to validate the input.
  - error_message: The error message to display if the input is invalid.

10. Various Validation Functions

- **Purpose**: Validate specific types of input data (e.g., is_alpha, is_valid_gender, is_valid_option, etc.).
- **Operation**: Each validation function checks the input against specific criteria and returns True or False.

11. search_for_specific_user()

- **Purpose**: Prompts the user to enter a User ID and searches for the user in the departments dictionary.
- **Operation**:
  - Prompts the user to enter a User ID.
  - Searches the departments dictionary for the user.
  - If found, displays the user's information.

12. display()

- **Purpose**: Displays all user data from the user info.txt file.
- **Operation**:
  - Checks if the user info.txt file exists.

- If it exists and contains data, displays the data.
- If the file is empty or doesn't exist, displays an appropriate message.

13. userOption()

- **Purpose**: Presents a menu to the user and calls the appropriate function based on the user's selection.
- **Operation**:
  - Displays a menu of options to the user.
  - Calls the appropriate function based on the user's selection (e.g., userInfo, display, search_for_specific_user, etc.).

14. userInfo()

- **Purpose**: Prompts the user for new user information, validates the input, and stores the data in both the departments dictionary and the user info.txt file.
- **Operation**:
  - Prompts the user to enter various details (Department ID, User ID, First Name, Last Name, Gender, Year of Birth).
  - Validates each input using the validation functions.
  - Stores the data in the departments dictionary and appends it to the user info.txt file.

Workflow

1. **Initialization**:
   - The departments dictionary is initialized to store department and user data.
2. **User Options**:
   - The script presents a menu of options to the user (userOption function).
   - Based on the user's selection, the script calls the appropriate function to perform tasks like entering new data, displaying data, searching for users, or inserting data into SQL Server.
3. **File and Dictionary Management**:
   - The script reads from and writes to files (read_from_file, write_to_file).
   - Data is stored in a nested dictionary (departments), which holds department information and associated users.
4. **Database Operations**:

- The script connects to an SQL Server database and creates tables if they don't exist.
- It then inserts data from the user info.txt and department info.txt files into the Users and Departments tables in the database.

5. **User Interaction**:
    - The script interacts with the user through prompts and displays appropriate messages based on the user's actions (e.g., successful data entry, user not found).

How to Run the Script

1. Ensure that Python and the required modules (pyodbc) are installed on your system.
2. Place the script in a directory along with the user info.txt and department info.txt files.
3. Run the script using a Python interpreter (python script_name.py).
4. Follow the on-screen prompts to interact with the script.