Using MySQLdb in Python

1. Setting Up the Database

Before starting with the lab tasks, ensure that you have executed the SQL script provided earlier to create the `US_States` database.

2. Github Repository

GitHub repository: `first_name-MySQL-Python-Lab.git`
Directory: `python-MySQLdb`

Create the above github repository , replace first-name with your first name and also create the directory python-MySQLdb that will contain all of your scripts.

Lab Tasks:
Task 1: List All States from the Database

Write a script ***(file name: 0-select_states.py)*** that lists all states from the database `US_States`:

- Your script should take 3 arguments: MySQL username, MySQL password, and database name (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Results must be sorted in ascending order by `states.id`.
- Results must be displayed as they are in the example below.
- Your code should not be executed when imported.

Below is the code for the above script:

```
import sys
import MySQLdb

if __name__ == "__main__":
        username = sys.argv[1]
    password = sys.argv[2]
    dbname = sys.argv[3]
```

```
db = MySQLdb.connect(host="localhost", user=username, passwd=password, db=dbname)
cursor = db.cursor()
cursor.execute("SELECT * FROM states ORDER BY id ASC")
results = cursor.fetchall()
for row in results:
    print(row)

cursor.close()
db.close()
```

Task 2: Filter States by Starting Letter

Write a script **(filename: 1-filter_states.py)** that lists all states with a name starting with 'A' from the database `US_States`:
- Your script should take 3 arguments: MySQL username, MySQL password, and database name (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Results must be sorted in ascending order by `states.id`.
- Your code should not be executed when imported.

Task 3: Insert a New State

Write a script **(filename: 2-insert_state.py)** that inserts a new state into the `states` table in the database `US_States`:
- Your script should take 5 arguments: MySQL username, MySQL password, database name, state name, and state abbreviation (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- The new state should be added to the database with default values for other fields.
- Your code should not be executed when imported.

Task 4: Update a State's Population

Write a script ***(filename: 3-update_population.py)*** that updates the population of a state in the `states` table in the database `US_States`:
- Your script should take 5 arguments: MySQL username, MySQL password, database name, state id, and new population (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- The population of the state with the specified id should be updated.
- Your code should not be executed when imported.

Task 5: Delete a State

Write a script ***(filename:* 4-delete_state.py*)*** that deletes a state from the `states` table in the database `US_States`:

- Your script should take 4 arguments: MySQL username, MySQL password, database name, and state id (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- The state with the specified id should be deleted from the database.
- Your code should not be executed when imported.

Task 6: Search for a State by Name

Write a script ***(filename:* 5-search_state.py*)*** that searches for a state by name in the database `US_States`:
- Your script should take 4 arguments: MySQL username, MySQL password, database name, and state name (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- The search should be case-insensitive.
- Results must be sorted in ascending order by `states.id`.
- Your code should not be executed when imported.

Example output of your script:
```
(1, 'Alabama', 'AL', 'Montgomery', 4903185, 1819)
```

Task 7: List All State Capitals

Write a script **(filename: 6-list_capitals.py)**that lists all state capitals from the database `US_States`:

- Your script should take 3 arguments: MySQL username, MySQL password, and database name (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Results must be sorted in ascending order by `states.capital`.
- Your code should not be executed when imported.

Task 8: Find the Most Populous State

Write a script **(filename: 7-most_populous_state.py)** that finds the most populous state in the database `US_States`:

- Your script should take 3 arguments: MySQL username, MySQL password, and database name (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Your code should not be executed when imported.

Task 9: Calculate the Average Population

Write a script **(filename: 8-average_population.py)** that calculates the average population of states in the database `US_States`:

- Your script should take 3 arguments: MySQL username, MySQL password, and database name (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Your code should not be executed when imported.

Task 10: List States Admitted After a Certain Year

Write a script **(filename: 9-states_after_year.py)**that lists all states admitted to the Union between the years of 1750 and 1850 and from the database `US_States`:

- Your script should take 4 arguments: MySQL username, MySQL password, database name, and year (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Results must be sorted in ascending order by `states.year_admitted`.
- Your code should not be executed when imported.

Task 11: Count States by Population Range

Write a script **(filename: 10-count_population_range.py)** that counts the number of states within a given population between 1,000,000 and 5,000,000 people in the database `US_States`:
- Your script should take 5 arguments: MySQL username, MySQL password, database name, minimum population, and maximum population (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Your code should not be executed when imported.

Task 12: Join States with Capitals Table

Create a capitals table and write a script **(filename: 11-join_states_capitals.py`**) that joins the `states` and `capitals` tables to display state names along with their capitals from the database `US_States`:
- Create the `capitals` table with the following structure:

```
CREATE TABLE capitals (
id INT AUTO_INCREMENT PRIMARY KEY,
state_id INT,
capital_name VARCHAR(100),
FOREIGN KEY (state_id) REFERENCES states(id)
);
```

- Your script should take 3 arguments: MySQL username, MySQL password, and database name (no argument validation needed).
- You must use the module MySQLdb (import MySQLdb).
- Your script should connect to a MySQL server running on `localhost` at port `3306`.
- Results must be sorted in ascending order by `states.id`.
- Your code should not be executed when imported.