Steps to Copy a CSV File into an SQL Table on macOS

1. Navigate to Your Directory Containing the CSV File

Use the terminal to navigate to the directory where your CSV file is located:

```
192:~ Faiem$ ls
```

This will list the contents of your home directory.

Next, move to the correct directory:

```
192:~ Faiem$ cd '/Users/Faiem/Desktop/DataScience/Power BI Sulabh Sir/Credit Card financial Dashboard'
```

You should now be inside the folder containing the CSV file (in this case, `credit_card.csv`).

2. Start MySQL with Local File Option

Run MySQL with the `--local-infile=1` option to allow the loading of local files into MySQL:

```
192:Credit Card financial Dashboard Faiem$ mysql --local-infile=1 --show-warnings -h 127.0.0.1 -u root -p
```

Enter your MySQL root password when prompted.

3. Enable the Local Infile Option

Before loading the CSV data, you need to enable the `local_infile` setting in MySQL to allow reading from local files:

```
mysql> SET GLOBAL local_infile = 1;
    Query OK, 0 rows affected (0.00 sec)
```

This command tells MySQL to allow loading files from local disk using the `LOAD DATA LOCAL INFILE` statement.

4. Select the Database

Select the database where you want to load the data (e.g., `ccdb`):

```
mysql> use ccdb;
    Database changed
```

Ensure you're working in the correct database before importing the data.

5. Load Data from the CSV File

Use the `LOAD DATA LOCAL INFILE` command to import the CSV data into the SQL table. Here's the syntax to load the CSV file into the `cc_detail` table:

```
mysql> LOAD DATA LOCAL INFILE 'credit_card.csv'
    INTO TABLE cc_detail
    FIELDS TERMINATED BY ','
    ENCLOSED BY '"'
    LINES TERMINATED BY '\r\n'
    IGNORE 1 LINES;
```

- `FIELDS TERMINATED BY ','` : Specifies that fields in the CSV file are separated by commas.
- `ENCLOSED BY '"'` : Indicates that fields are enclosed by double quotes (optional but common for CSV files).
- `LINES TERMINATED BY '\r\n'` : Defines the line terminator. For Windows CSV files, this is usually `\r\n` . For macOS/Linux, it is often `\n` . Adjust this based on your file format.
- `IGNORE 1 LINES` : Skips the header row of the CSV file.

6. **Verify the Import**

After the command runs, you should see the following confirmation:

```
Query OK, 10108 rows affected, 10108 warnings (0.16 sec)
    Records: 10108  Deleted: 0  Skipped: 0  Warnings: 10108
```

This output confirms that 10,108 rows were successfully imported into the `cc_detail` table.

7. **Check the Data**

To verify that the data has been successfully loaded, you can run a `SELECT` query:

```
mysql> SELECT * FROM cc_detail LIMIT 10;
```

This will show the first 10 rows of the `cc_detail` table to confirm the import.

Key Points to Remember:

1. **Enable `local_infile`** : You must enable the `local_infile` option in MySQL both at the global and session levels to load local files.
2. **Correct Path to CSV**: Ensure the file path to the CSV is correct. If using relative paths, make sure you are in the right directory.
3. **File Format**: Ensure that your CSV file has the correct formatting:
   - Fields are separated by commas.
   - Data is optionally enclosed by double quotes.
   - Line breaks are appropriate for your operating system (macOS typically uses `\n`, while Windows uses `\r\n`).
4. **Permission Issues**: The MySQL server needs permission to read the file, especially if it is located outside the standard MySQL directories. If the file is located in a system directory, you might need to adjust file permissions.

By following these steps, you can successfully load CSV data into a MySQL table on your macOS system.

```
In [ ]:
```