

# ***File I/O in Python***

## **Details, Source & Information**

The Following work contains 3 works,

- i) Reading CSV file and passing it as a record in Database.
- ii) Saving an existing database into .csv (Comma Separated Value) file.
- iii) Reading TXT file and passing it as a record in Database.

### **Modules & Software used:**

**Python 3.9 +** and higher, **MySQL Community Server**, CSV files, Text Files and **python-mysql-connector**.

### **Knowledge's and Prerequisites:**

- i) Basic Knowledge of Python and MySQL.
- ii) Usage of MySQL in Python, establishing connection with MySQL Server locally.
- iii) Know to Read and Write CSV files.
- iv) Know to Read and Write Text files.

### **Sources to learn afore mentioned Knowledge:**

Python:

<https://docs.python.org/3/tutorial/index.html>

[https://www.w3schools.com/python/python\\_intro.asp](https://www.w3schools.com/python/python_intro.asp)

MySQL with Python:

[https://www.w3schools.com/python/python\\_mysql\\_getstarted.asp](https://www.w3schools.com/python/python_mysql_getstarted.asp)

<https://www.mysqltutorial.org/python-mysql/>

Read and Write CSV File in Python:

<https://realpython.com/python-csv/>

<https://docs.python.org/3/library/csv.html>

Read and Write Text File in Python:

<https://www.geeksforgeeks.org/reading-writing-text-files-python/>

<https://realpython.com/read-write-files-python/>

## ***Part 1: Reading and Passing a CSV File into Database:***

A Comma Separated Value file abbreviated as CSV is a special type of file with other features and properties is a type of EXCEL sheet or file, usually used to share large chunks of data in file format.

Now, for this part we will have to create a sample CSV file first, for the sake of the project we already have a Sample CSV file containing Employee Data of an Organization – The file has 7 columns like,

**Employee ID | Name | Floor Number | Address | Contact Number | Designation | Salary (Yearly) \$**

Now, keep in mind it is recommended for beginners to keep the **Python file** where you are writing the code and the **Sample CSV data file** in the **same location** i.e same folder on your local system.

Next, up we will establish connection with the MySQL server with the following arguments,

```
2
3 db = mysql.connector.connect(
4     host = "localhost",
5     user = "root",
6     passwd = "root",
7     database = "mydb"
8 )
```

Now, we will set up a cursor for executing commands of MySQL in Python.

```
9
10 myCursor = db.cursor()
11
```

Now, we are ready to execute commands in MySQL.

**Note:** We have pre-assumed that you already have a database called “mydb” created in your system, if you will use another table you are free to do that.

Now, to keep records in the database we need a table and we will create that with the following command,

```
12 """
13 Creation of the Sample Table and passing it's fields as in Columns, and reading it to see if everything is correct
14 """
15
16 myCursor.execute("""create table PythonCsvData (EmployeeID VARCHAR(20) PRIMARY KEY, Name VARCHAR(30), FloorNumber INT(5),
17 | | | | | ContactNumber VARCHAR(50), Address VARCHAR(200), Designation VARCHAR(60), Salary INT(20))""")
18
```

Now that we have a table called ('pythoncsvdata',) we will insert all the records into this table in the database “mydb”.

## Approach:

We will have a straight forward approach for this, the CSV file which is stored in the same location where our .py file is stored will be opened by a special command in python called,

**with open ( “folder name\file name” ) as csv\_file:**

and the passing successive arguments and commands to read each row at a time and then appending them to an empty list in the form of tuple, i.e each row in the file will be stored in the list as a single tuple element, which means that the list will contain 41 tuple elements which are rows of the CSV file each. The code is inserted below:

```
24
25 with open("CSV File Operations with Python\PythonCSVData.csv") as csv_file:
26     csvfile = csv.reader(csv_file, delimiter = ",", )
27     all_value = []
28     for row in csvfile:
29         value = (row[0], row[1], row[2], row[3], row[4], row[5], row[6])
30         all_value.append(value)
31
```

Furthermore, we will now write a query and then pass all the records which are in the form of single tuple elements of a list into the table ('pythoncsvdata',) of the database "mydb". The command as follows:

```
33
34 query = """INSERT INTO PythonCsvData(EmployeeID, Name, FloorNumber, ContactNumber, Address, Designation, Salary)
35 VALUES (%s, %s, %s, %s, %s, %s, %s)"""
36 myCursor.executemany(query, all_value)
37 db.commit()
```

The following data entry transaction is committed using the command "db.commit()", it saves the data we have inserted into the SQL database. Make Sure you run this command every time you perform a **CRUD** (Create, Read, Update, Delete) operation in the Database.

## ***Part 2: Extracting Data from a local database in CSV file format:***

For this part, we pre-assume that you have a sample table in your local database ready to be extracted as a CSV file.

Here for our reference, we will use the Database “mydb” and the table (‘table1’), the Data is a file containing 6 columns,

**Name | Class | Section | ID | Percentages | Roll Number**

Next, up we will establish connection with the MySQL server with the following arguments,

```
2
3  db = mysql.connector.connect(
4      host = "localhost",
5      user = "root",
6      passwd = "root",
7      database = "mydb"
8  )
```

Now, we will set up a cursor for executing commands of MySQL in Python.

```
9
10  myCursor = db.cursor()
11
```

Now, we are ready to execute commands in MySQL.

**Note:** We have pre-assumed that you already have a database called “mydb” created in your system, if you will use another table you are free to do that.

## Approach:

First we will fetch the data from the table of the database using the fields name by executing the SQL query using myCursor and then we will store the result in a variable called "result".

```
13 query = "SELECT name, class, sec, id, percentages, roll_num FROM table1"
14
15 myCursor.execute(query)
16 result = myCursor.fetchall()
```

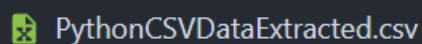
Here, we will first use the special command,

**with open("Specify a Location (optional)\File name you want to give") as file:**

and now we will write the file each time a row is fetched from the database and then insert into the CSV file and then we will close the cursor for the safety of not executing any more commands. The Code looks like,

```
18 with open("CSV File Operations With Python\PythonCSVDataExtracted.csv", "w") as file:
19     for row in result:
20         csv.writer(file).writerow(row)
21
22 myCursor.close()
```

After you do the following steps and do it correctly, you should receive a CSV file containing the above said data stored in you File Explorer or the Folder you are in. It looks like,

A screenshot of a file explorer window showing a single file named "PythonCSVDataExtracted.csv". The file icon is a green document with a small 'x' symbol.

**Note:** It has the same name we have given in the open command.

In MySQL table the data looks like, shown below:

```
mysql> use mydb;
Database changed
mysql> select * from table1;
```

name	class	sec	id	percentages	roll_num
Apurba Ghosh	11	A	1	90%	5
Himangshu De	11	C	2	78%	6
Raima Ray	11	B	3	62%	7
Priyanka Das	11	D	4	78%	2
Rupankar Das	11	C	5	72%	9
Shovan Das	12	F	7	71.5%	8
Shreyashi Halder	12	A	8	85%	2
Debabrata Patikar	12	B	9	98.5%	1
Sangram Sarkar	10	A	10	74.5%	7
Subhajit De	9	E	11	95%	10

```
10 rows in set (0.12 sec)
```

After having run all the commands and receiving the file in a CSV format the Data in the sheet looks like this,

	A	B	C	D	E	F	G
1	Apurba Ghosh	11	A	1	90%	5	
2							
3	Himangshu De	11	C	2	78%	6	
4							
5	Raima Ray	11	B	3	62%	7	
6							
7	Priyanka Das	11	D	4	78%	2	
8							
9	Rupankar Das	11	C	5	72%	9	
10							
11	Shovan Das	12	F	7	71.5%	8	
12							
13	Shreyashi Halder	12	A	8	85%	2	
14							
15	Debabrata Patika	12	B	9	98.5%	1	
16							
17	Sangram Sarkar	10	A	10	74.5%	7	

**Congratulations!**, you have successfully created a CSV format of the data stored in your Local MySQL server.

### ***Part 3: Reading and Passing a Text file into a Database:***

A **text file** (sometimes spelled **textfile**; an old alternative name is **flatfile**) is a kind of **computer file** that is structured as a sequence of **lines** of **electronic text**. A text file exists **stored as data** within a **computer file system**. In operating systems such as **CP/M** and **MS-DOS**, where the operating system does not keep track of the file size in bytes, the end of a text file is denoted by placing one or more special characters, known as an **end-of-file** marker, as padding after the last line in a text file. On modern operating systems such as **Microsoft Windows** and **Unix-like** systems, text files do not contain any special EOF character, because file systems on those operating systems keep track of the file size in bytes. There are for most text files a need to have **end-of-line delimiters**, which are done in a few different ways depending on operating system.

For this part, we pre-assume that you have a Sample TXT file stored in your local system.

For reference we will use a TXT file named PythonTextData.txt

Which contains data divided into 3 columns,

**SongName | ArtistName | ReleaseDate**



Next, up we will establish connection with the MySQL server with the following arguments,

```
2
3 db = mysql.connector.connect(
4     host = "localhost",
5     user = "root",
6     passwd = "root",
7     database = "mydb"
8 )
```

Now, we will set up a cursor for executing commands of MySQL in Python.

```
9
10 myCursor = db.cursor()
11
```

Now, we are ready to execute commands in MySQL.

**Note:** We have pre-assumed that you already have a database called “mydb” created in your system, if you will use another table you are free to do that.

Here we will use a table called (‘pythontextdata’,) and now we will create the table using the following command in our database and then we will check if the table is created or not.

```
16
17 myCursor.execute("""create table pythontextdata(SongName VARCHAR(60),
18 ArtistName VARCHAR(60) PRIMARY KEY, ReleaseDate VARCHAR(40))""")
19 myCursor.execute("show tables")
20 for x in myCursor:
21     print(x)
22
```

### ***Approach:***

Here, we will read the text file line by line and then try to store the data into an empty list and then use that to execute a query to pass the data into an SQL database.

**Note:** There are some limitations in this process, just like while in a text file we move to a new line each time an escape sequence character is generated in the system “\n” and to deal with that we will use a special string function called “strip” to remove that every time it’s generated, as the data which is read each time one line is received as a String datatype we can perform all string operation in the data.

Next up we will use a function called tuple to convert the data into a tuple which will help us to pass that as query into the database.

Although the tuple function has a problem i.e it treats each and every letter of the data as an element so avoid that problem we will use the map function wherein it will take two parameters i.e the datatype we want the data in and then the split parameter i.e in how many parts we want it to split.

**Note:** We need to split the data and best preferred way is by giving comm (’,’) as the split parameter. As it’s a simple text file so it does not contain rows and columns and cells. So it will treat all the data in a single line as one, so map each column with correct data we split it into parts so that the correct data goes into the correct column.

And then we append the data into the empty list.

Here, we will first create an empty list and then use the special command

**With open (“name of the file”, “r”) as text\_file:**

And now each time a line from the text file is read we will receive the data as a single string. So, we will first remove the “\n” escape sequence character and then convert it into a tuple and by using the tuple and map function while splitting it into parts according to columns and then pass it to the empty list as a

single element tuple, so the list here will contain 6 tuples each of which is tuple containing the data of the specified columns as a single element of that individual tuple.

Below is the code for the reference,

```
22
23     """
24     Reading and passing the text data into SQL Database mydb
25     """
26
27
28     values = []
29     with open("PythonTextData.txt", "r") as text_file:
30         for text in text_file:
31             value1 = text.strip("\n")
32             value1 = tuple(map(str, value1.split(', ')))
33             values.append(value1)
```

Finally, we will execute a SQL Query to move all the data stored as single tuple elements in the list to the table ('pythontextdata',) of the database "mydb" and then we will print the results of the table to see if the data is moved or not. The below code will do the work,

```
34     query = "INSERT INTO pythontextdata(SongName, ArtistName, ReleaseDate) VALUES (%s,%s,%s)"
35     myCursor.executemany(query, values)
36     db.commit()
37     myCursor.execute("select * from pythontextdata")
38     for x in myCursor:
39         print(x)
```

**Note:** All the above information are presented into parts which represents only the specific tasks mentioned here like passing of Data in CSV or TXT format into database or extracting it in the form of CSV file but things like creation of database or checking if table has data or seeing the fields of table and other SQL queries are not mentioned here. For that you need to have a basic knowledge of usage of MySQL with Python. Also, the sample dataset files are not attached here. If you require those

then consider going to the Resources section below and check the locations and for video tutorials checkout the links.

## **Resources:**

All the Code in detailed manner and the Sample dataset files are stored into my GitHub repository consider checking that:

[https://github.com/iamapurba2003/Higher-Secondary\\_CS/tree/main/File%20IO%20Operations%20With%20Python](https://github.com/iamapurba2003/Higher-Secondary_CS/tree/main/File%20IO%20Operations%20With%20Python)

**For the Video tutorials follow this YouTube Links:**

Reading and Passing the CSV Data into SQL database:

[https://www.youtube.com/watch?v=LNgg\\_YJ29OY](https://www.youtube.com/watch?v=LNgg_YJ29OY)

Exporting an existing database into CSV file format:

<https://www.youtube.com/watch?v=3hllGdiM2Ys>

**Note:** The following tutorials might be bit different, but the overall matter is the same. Good Luck !

© Apurba Ghosh