MySQL with Python3

July 19, 2019

1 Set up:

- Download MySQL and Python3
- Set MySQL connector by executing following three commands and make sure that they have no error,
  - pip install mysql-connector
  - pip install mysql-connector-python
  - pip install mysql-connector-python-rf
- If possible, download MYSQLworkbench and set up password(better remember that)
- For .py file

```
import mysql.connector
mydb = mysql.connector.connect(host="localhost", user="root", passwd="xxx")
```

- If there is no error when running it, then the database is created :)
- For future working specific database, you need to add **database="dbname"** in the connect statement.
- for working in that database you need to add mycursor = mydb.cursor()

2 Insertion, Seletion

- Create table:
  - Since we have indicate that database="mydatabase", therefore every table that we created is in mydatabase
  - mycursor.execute("CREATE TABLE tablename (colname datatype, ...”))
  - datatype includes INT, VARCHAR, etc
- Insertion format:
  - sql = INSERT INTO *tablename* ($col1, col2$) VALUES ($\%s, \%s$)
  - val = ("xxxx","xxxx")
  - mycursor.execute(sql,val)
- If you want to enter more than one rows, you could use the following codes:
  - val = [("xxxx","xxxx"),("yyyy","yyyy"),$\cdots$]
  - mycursor.executemany(sql,val)
- You MUST add **mydb.commit()** to invoke new change
  - Be aware that **mydb** is not your database name,but it is the one that connect to your database.
- For working in MYSQLworkbench:
  - If you are confronted with "No database selected$\cdots$", you need to type **USE db_name** to resolve this error
  - Then you can run **SELECT* FROM db_name**, you should see the values you just entered
- Do general selection:
  - mycursor.execute("**SELECT* FROM tablename**"): it selects every col to display
  - mycursor.execute("**SELECT col1, col2, $\cdots$ FROM tablename**"): it selects some cols to display
  - mycursor.**fetchall()**: it displays all rows with cols that you select above (one element with selected cols per line)
  - mycursor.**fetchsome()**: it displays the first row with all cols in one line
  - mycursor.**fetchone()**: it displays the first row with all its cols in separate lines
- Select according to certain col(s)

  - mycursor.execute(**SELECT* FROM tablename WHERE col1='xxx'**), you can do all operations, e.g. $>, <, =, etc$
  - mycursor.execute(**SELECT* FROM tablename WHERE col1 LIKE '**$\%s$ **xx** $\%s$): apply **LIKE** gives you more choices for selection. The one above is to select the substring of col1 is 'xx', also to apply prefix you can do $\%xx$ and for suffix apply $xx\%$
  - Always allow to use $\%s$ as a placeholder to inject the information
    * sql = ("SELECT * FROM tablename WHERE col = $\%s$")
    * adr = ("content",)
    * mycursor.execute(sql, adr)
    * myresult = mycursor.fetchall()

- Select with certain limit number or which row to start
  - For only display certain amount of queries, we apply keyword "limit"
  - mycursor.execute("SELECT* FROM tablename LIMIT #"): it only displays # queries starting from the first one
  - mycursor.execute("SELECT* FROM tablename LIMIT #2 OFFSET #1): it will display #2 queries starting from $\#1^{th}$ query

3 Ordering, Deletion and Updating

- **Ordering:**
  - In my SQL, there are two ways to order the data, one is descending(DESC) and one is increasing(ASC).
  - mycursor.execute(**SELECT* FROM tablename ORDER BY column1 ASC/DESC, column2 ASC/DESC**)
  - Or we can just order by certain column, like "name" – just leave the column name without specifying any order.

- **Deletion:**
  - Deletion is really similar to insertion
  - mycursor.execute(**"DELETE FROM tablename WHERE column1='xxx'"**)
  - Then all the rows with specified value will be removed

- **Dropping:**
  - Deleting the table using "DROP TALBE"
  - mycursor.execute(**"DROP TABLE tablename"**)

  - Drop if exists
  - mycursor.execute(**"DROP TABLE IF EXISTS tablename"**)

- **Updating:**
  - Updating is really similar to insertion
  - mycursor.execute("UPDATE tablename SET colname='xx' WHERE colname='yy'")
  - It replaces the col with value yy to xx
  - Must add **mydb.commit()** to invoke the change