

Student Name: _____

Roll No: _____

Section: _____

Lab Series No. 14.

Lab 14 –Introduction to Database SQLITE3 with Python.

Lab Objectives:

1. Introduction to Database
2. Connection
3. Create SQLite3 Cursor
4. Create table inside database
5. Insert data in Table
6. Alternate method of insert data via ‘?’
7. Update Table
8. Select specific columns from Table
9. Select all records from the Table
10. Close Connection

1. Introduction to Database

A database is an organized collection of data, generally stored and accessed electronically from a computer system. Where databases are more complex they are often developed using formal design and modeling techniques. SQLite is a lightweight database that can provide a relational database management system with zero-configuration because there is no need to configure or setup anything to use it.

To use SQLite3 in Python, first of all, you will have to import the sqlite3 module and then create a connection object which will connect us to the database and will let us execute the SQL statements.

2. Connection

A connection object is created using the connect() function:

```
import sqlite3
con = sqlite3.connect('mydatabase.db')
```












Student Name: _____	Roll No: _____	Section: _____	
 mydatabase	15/01/2020 4:55 AM	SQLite database	2 KB
 step1	14/01/2020 10:19 PM	JetBrains PyCharm ...	1 KB
 step1b	14/01/2020 10:24 PM	JetBrains PyCharm ...	1 KB
 step2	14/01/2020 10:22 PM	JetBrains PyCharm ...	1 KB
 step3	14/01/2020 10:29 PM	JetBrains PyCharm ...	1 KB
 step4	15/01/2020 4:55 AM	JetBrains PyCharm ...	1 KB
 step5	15/01/2020 4:42 AM	JetBrains PyCharm ...	1 KB
 step6	15/01/2020 4:40 AM	JetBrains PyCharm ...	1 KB
 step7	15/01/2020 5:00 AM	JetBrains PyCharm ...	1 KB
 step8	15/01/2020 4:57 AM	JetBrains PyCharm ...	1 KB
 Week13 Database Programming SQLITE	15/01/2020 8:53 AM	Microsoft PowerPo...	4,151 KB

Figure 14.1 Create mydatabase.db using python code.

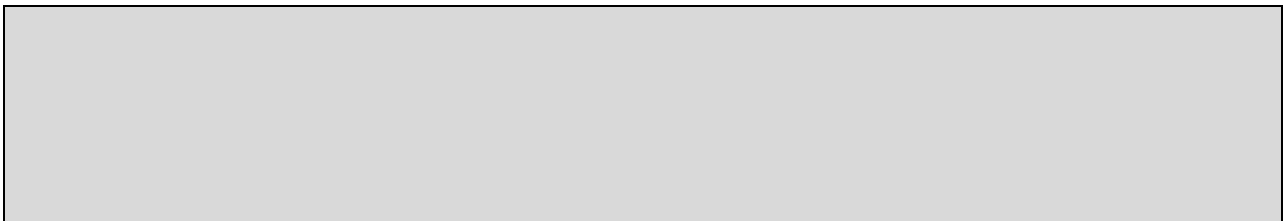
Program 1: Write a Python program to create a database with your own name.

Code:

```
import sqlite3
con = sqlite3.connect('mydatabase.db')
```

Note: instead of mydatabase.db use your name.db without any space.

Output:



3. Create SQLite3 Cursor

To execute SQLite statements in Python, you need a cursor object. You can create it using the cursor() method.

The SQLite3 cursor is a method of the connection object. To execute the SQLite3 statements, a connection is established at first and then an object of the cursor is created using the connection object as follows:

```
con = sqlite3.connect('mydatabase.db')
cursorObj = con.cursor()
```

Now we can use the cursor object to call the execute() method to execute any SQL queries.

Student Name: _____

Roll No: _____

Section: _____

Program 2: Create cursor for SQLite3 to access your database.**Code:**

```
import sqlite3

con = sqlite3.connect('mydatabase.db')
cursorObj = con.cursor()
```

Output:

4. Create Table inside Database

To create a table in SQLite3, you can use the Create Table query in the execute() method. Consider the following steps:

1. The connection object is created
2. Cursor object is created using the connection object
3. Using cursor object, execute method is called with create table query as the parameter

Let's create employees with the following attributes:

employees (id, name, salary, department, position, hireDate)

Program 3: Create a table name it as employees with employee id as primary key, name, salary, hiredate and department.

Code:

```
import sqlite3
from sqlite3 import Error

def sql_connection():
    try:
        con = sqlite3.connect('mydatabase.db')
        return con
    except Error:
```

Student Name: _____

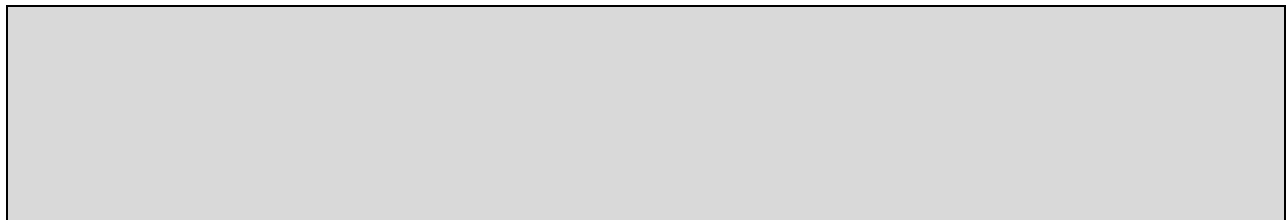
Roll No: _____

Section: _____

```
print(Error)

def sql_table(con):
    cursorObj = con.cursor()
    cursorObj.execute("CREATE TABLE employees(id integer PRIMARY KEY,
name text, salary real, department text, position text, hireDate
text)")

    con.commit()
con = sql_connection()
sql_table(con)
```

Output:

In the above code, we have defined two methods, the first one establishes a connection and the second method creates a cursor object to execute the create table statement.

The commit() method saves all the changes we make. In the end, both methods are called.

To check if our table is created, you can use the DB browser for sqlite to view your table. Open mydatabase.db file with the program and you should see your table:

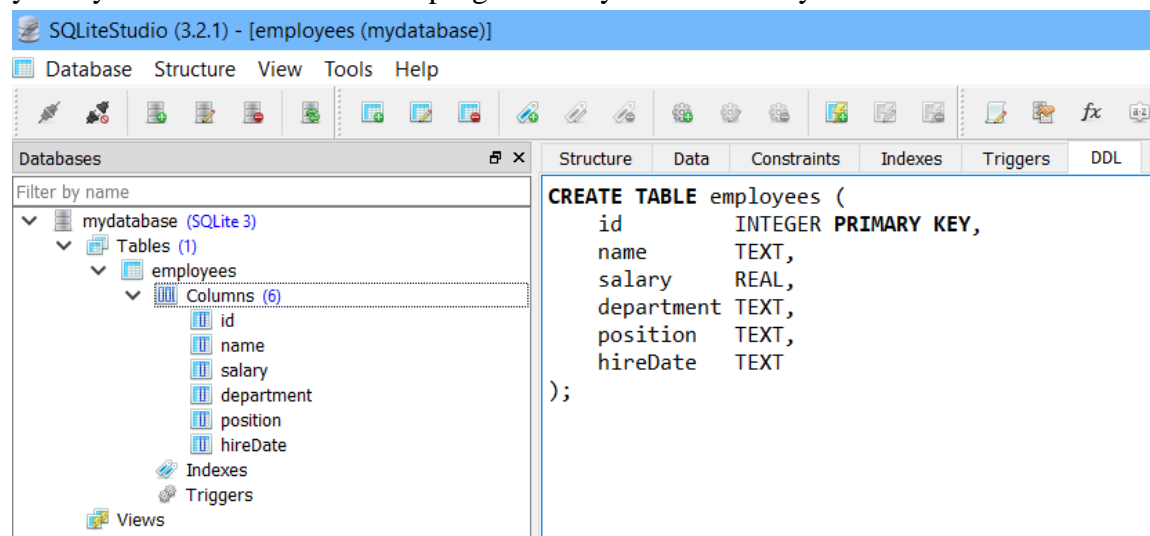


Figure 14.2 Structure of employee table.

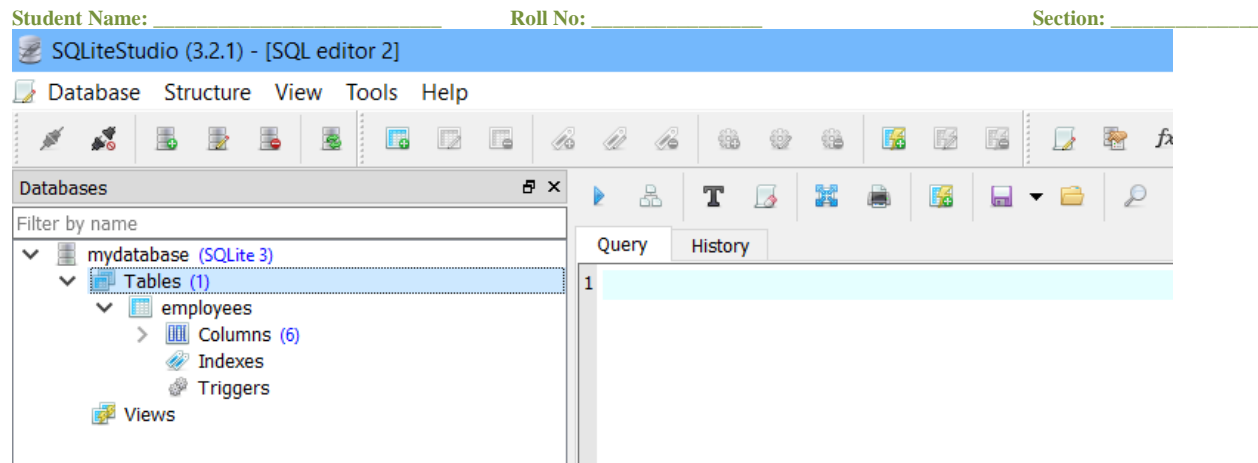


Figure 14.3 In Database mydatabase.db, create employee table.

5. Insert Data inside the Table

To insert data in a table, we use the INSERT INTO statement. Consider the following line of code:

```
cursorObj.execute("INSERT INTO employees VALUES(5, 'Wasim', 200000,
'CS', 'Asst.Prof.', '1995-05-21')")
```

Program 4: Write a program which will insert at least 10 records inside your employee data. Having different departments, joining date and salaries. For reference few lines of code is given. Keep in mind that each employee has new and unique employee Id as it's a primary key and primary key cannot be duplicated and cannot left blank in database.

Code:

```
# Insert 3 records in a table

import sqlite3
con = sqlite3.connect('mydatabase.db')

cursorObj = con.cursor()

cursorObj.execute("INSERT INTO employees VALUES(3, 'Parkash', 200000,
'CS', 'Assoc.Prof.', '2002-03-10')")
cursorObj.execute("INSERT INTO employees VALUES(6, 'Fauzan', 120000,
'CS', 'Asst.Prof.', '2007-09-01')")
cursorObj.execute("INSERT INTO employees VALUES(5, 'Wasim', 200000,
'CS', 'Asst.Prof.', '1995-05-21')")
con.commit()
```

Student Name: _____

Roll No: _____

Section: _____

Output:

To check if the data is inserted, click on Browse Data in the DB Browser:

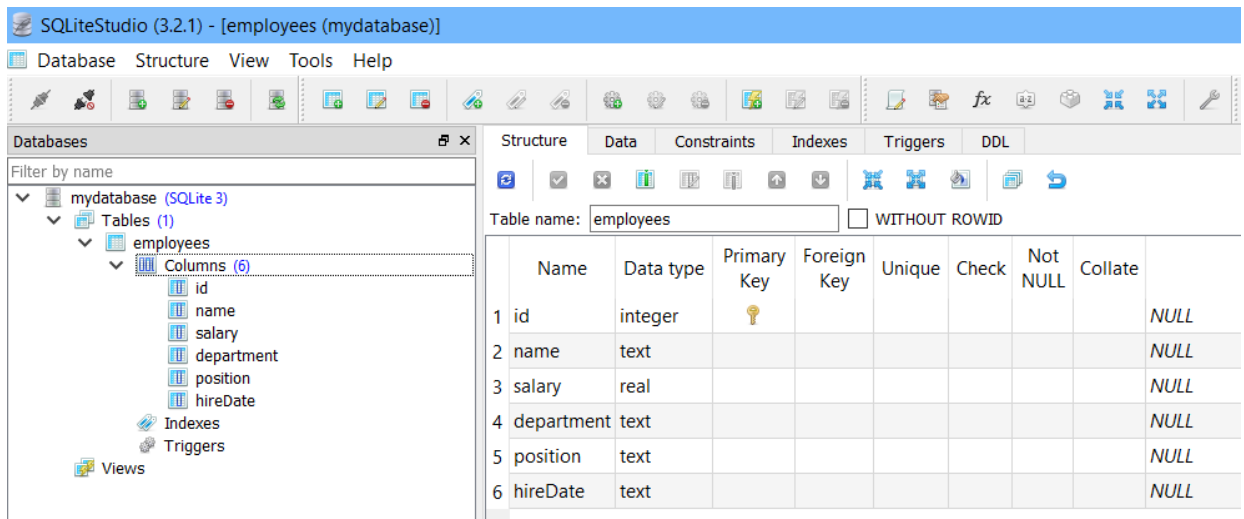


Figure 14.4 Structure of employee table with Primary Key.

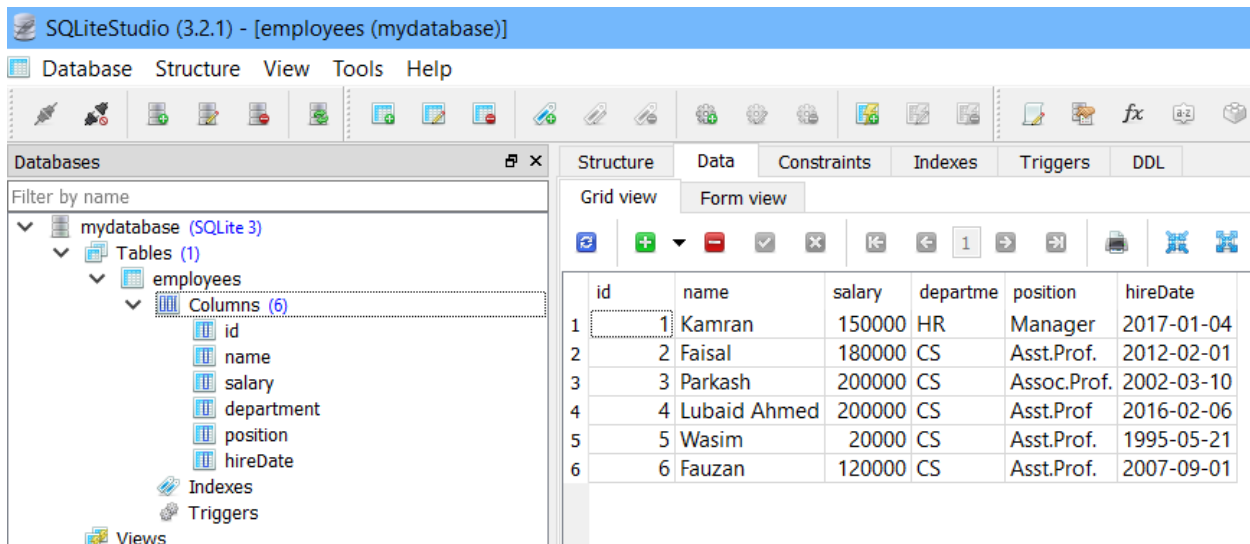


Figure 14.5 In Employee table, few data records have been inserted

Student Name: _____

Roll No: _____

Section: _____

Alternate Method of Data Insertion via ‘?’

We can also pass values/arguments to an INSERT statement in the *execute()* method. You can use the question mark (?) as a placeholder for each value. The syntax of the INSERT will be like the following:

```
cursorObj.execute('INSERT INTO employees(id, name, salary, department, position, hireDate) VALUES(?, ?, ?, ?, ?, ?)', entities)
```

Program 5: Write a program which will insert the data by using ? in placeholder.

Code:

```
import sqlite3
con = sqlite3.connect('mydatabase.db')

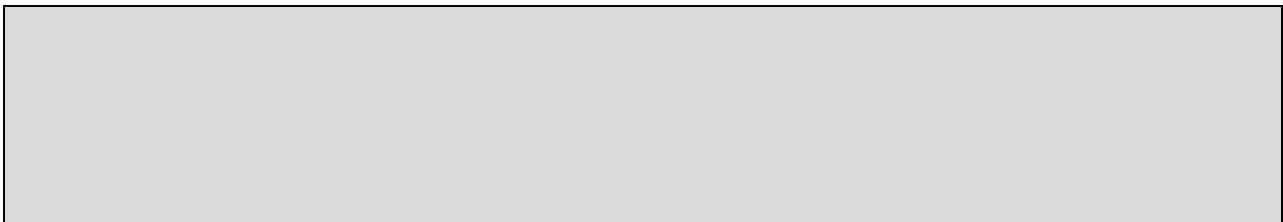
def sql_insert(con, entities):
    cursorObj = con.cursor()

    cursorObj.execute('INSERT INTO employees(id, name, salary, department, position, hireDate) VALUES(?, ?, ?, ?, ?, ?)', entities)

    con.commit()

entities = (4, 'Lubaid', 200000, 'CS', 'Asst.Prof', '2016-02-06')
sql_insert(con, entities)
```

Output:



6. Update Table

To update the table simply create a connection, then create a cursor object using the connection and finally use the UPDATE statement in the *execute()* method.

Student Name: _____ Roll No: _____ Section: _____

Suppose that we want to update the name of the employee whose id equals 4. For updating, we will use the UPDATE statement and for the employee whose id equals 4. We will use the WHERE clause as a condition to select this employee.

Consider the following code:

```
cursorObj.execute('UPDATE employees SET name = "Lubaid Ahmed" where id = 4')
```

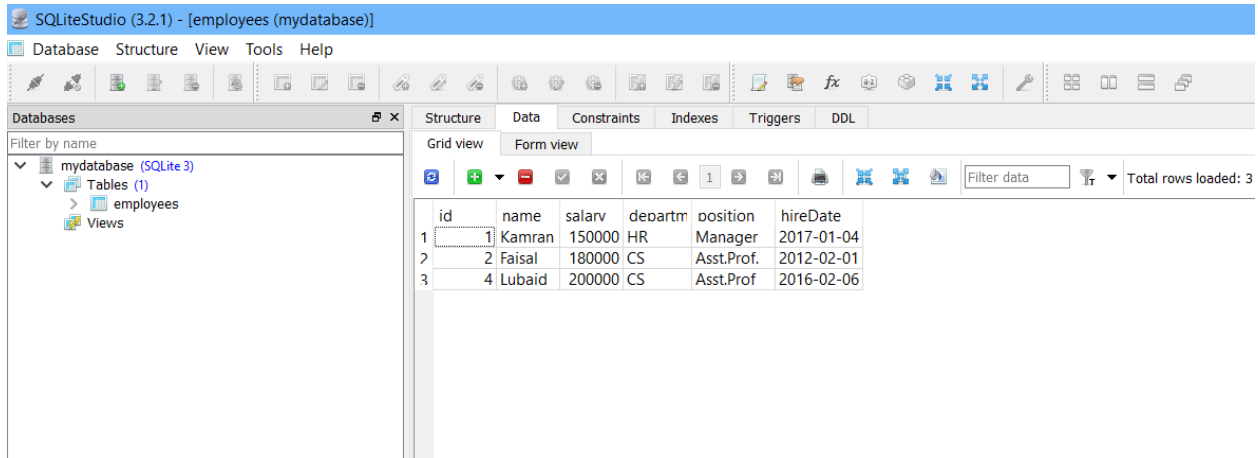


Figure 14.6 Before update the Employee table, record of Dr. Lubaid.

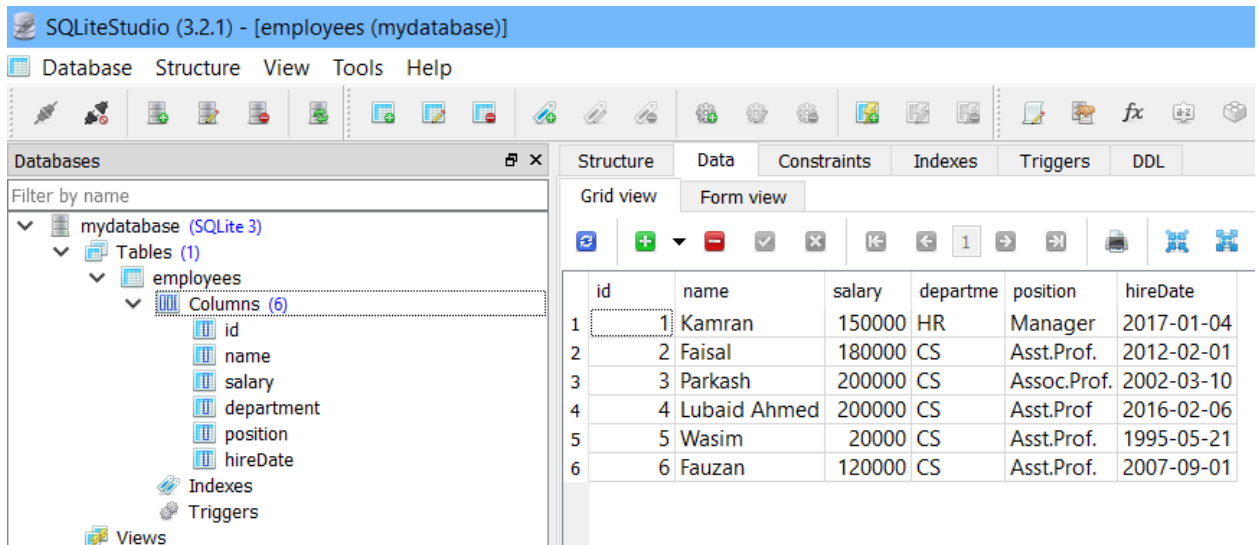


Figure 14.7 After updating the Employee table, record of Dr. Lubaid.

Student Name: _____

Roll No: _____

Section: _____

Program 6: Write a program which will update the record of Dr. Lubaid Ahmed by filling full name instead of just 'Lubaid'. The id of the record is 4.

Code:

```
import sqlite3

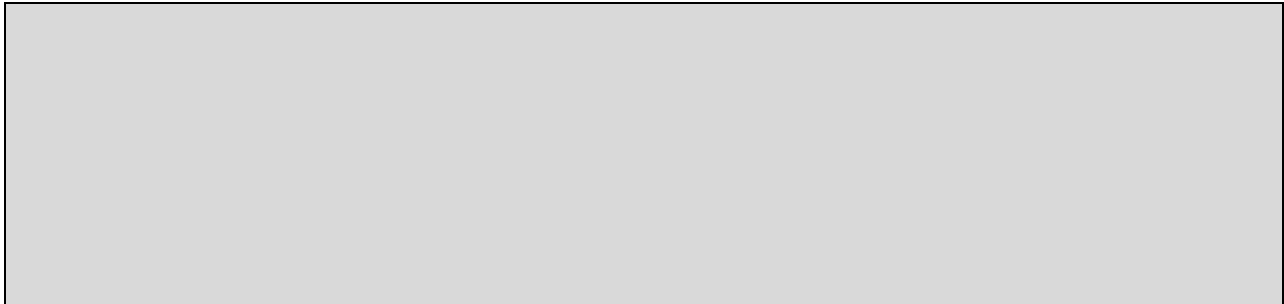
con = sqlite3.connect('mydatabase.db')

def sql_update(con):

    cursorObj = con.cursor()

    cursorObj.execute('UPDATE employees SET name = "Lubaid Ahmed"
where id = 4')

    con.commit()
sql_update(con)
```

Output:

7. Select specific columns from the Table

The select statement is used to select data from a particular table. If you want to select all the columns of the data from a table, you can use the asterisk (*). The syntax for this will be as follows:

```
select * from table_name
```

In SQLite3, the SELECT statement is executed in the execute method of the cursor object. For example, select all the columns of the employees' table, run the following code:

```
cursorObj.execute('SELECT * FROM employees ')
```

If you want to select a few columns from a table then specify the columns like the following:

```
select column1, column2 from tables_name
```

Student Name: _____

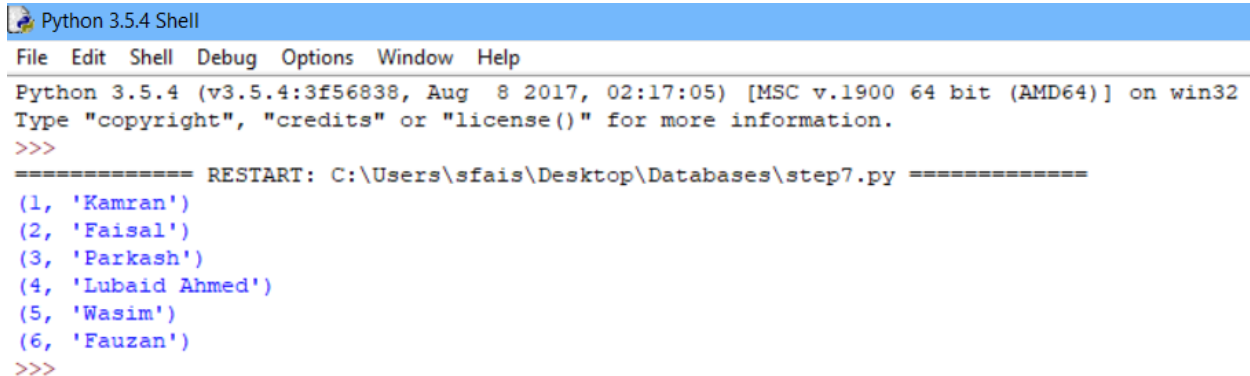
Roll No: _____

Section: _____

For example:

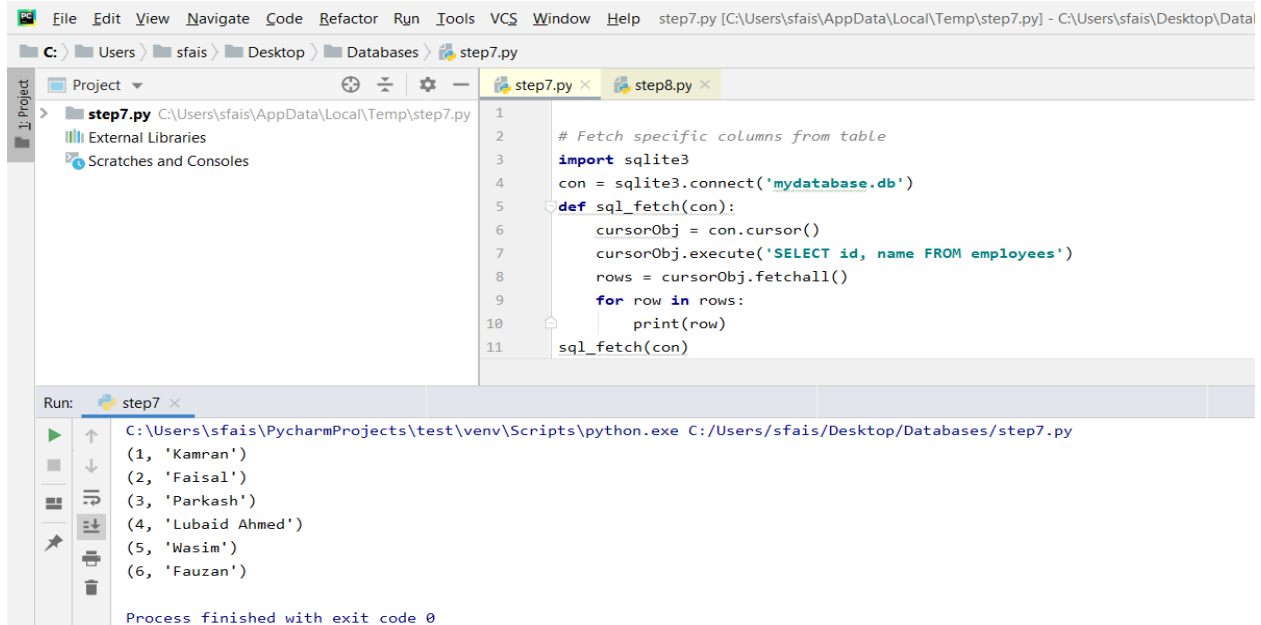
```
cursorObj.execute('SELECT id, name FROM employees')
```

The select statement selects the required data from the database table and if you want to fetch the selected data, the fetchall() method of the cursor object is used.



```
Python 3.5.4 Shell
File Edit Shell Debug Options Window Help
Python 3.5.4 (v3.5.4:3f56838, Aug 8 2017, 02:17:05) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\sfaish\Desktop\Databases\step7.py =====
(1, 'Kamran')
(2, 'Faisal')
(3, 'Parkash')
(4, 'Lubaid Ahmed')
(5, 'Wasim')
(6, 'Fauzan')
>>>
```

Figure 14.8 Fetch specific data from Employee Table on IDLE Environment.



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help step7.py [C:\Users\sfaish\AppData\Local\Temp\step7.py] - C:\Users\sfaish\Desktop\Data
C:\Users\sfaish\Desktop\Databases\step7.py
Project
step7.py C:\Users\sfaish\AppData\Local\Temp\step7.py
External Libraries
Scratches and Consoles
1 # Fetch specific columns from table
2 import sqlite3
3 con = sqlite3.connect('mydatabase.db')
4 def sql_fetch(con):
5     cursorObj = con.cursor()
6     cursorObj.execute('SELECT id, name FROM employees')
7     rows = cursorObj.fetchall()
8     for row in rows:
9         print(row)
10
11 sql_fetch(con)
Run: step7
C:\Users\sfaish\PycharmProjects\test\venv\Scripts\python.exe C:/Users/sfaish/Desktop/Databases/step7.py
(1, 'Kamran')
(2, 'Faisal')
(3, 'Parkash')
(4, 'Lubaid Ahmed')
(5, 'Wasim')
(6, 'Fauzan')
Process finished with exit code 0
```

Figure 14.9 Fetch specific data from Employee Table on Pycharm Environment.

Student Name: _____

Roll No: _____

Section: _____

Program 7: Write a program which will fetch all records but limited or selected columns such as id and name.

Code:

```
# Fetch specific columns from table
import sqlite3
con = sqlite3.connect('mydatabase.db')
def sql_fetch(con):
    cursorObj = con.cursor()
    cursorObj.execute('SELECT id, name FROM employees')
    rows = cursorObj.fetchall()
    for row in rows:
        print(row)
sql_fetch(con)
```

Output:

8. Select all records from the Table

To fetch the data from a database we will execute the SELECT statement and then will use the fetchall() method of the cursor object to store the values into a variable. After that, we will loop through the variable and print all values.

Program 8: Write a program which will fetch all records from the table.

Code:

```
# Fetch all columns from table
import sqlite3

con = sqlite3.connect('mydatabase.db')

def sql_fetch(con):

    cursorObj = con.cursor()
```

Student Name: _____

Roll No: _____

Section: _____

```
cursorObj.execute('SELECT * FROM employees')

rows = cursorObj.fetchall()

for row in rows:

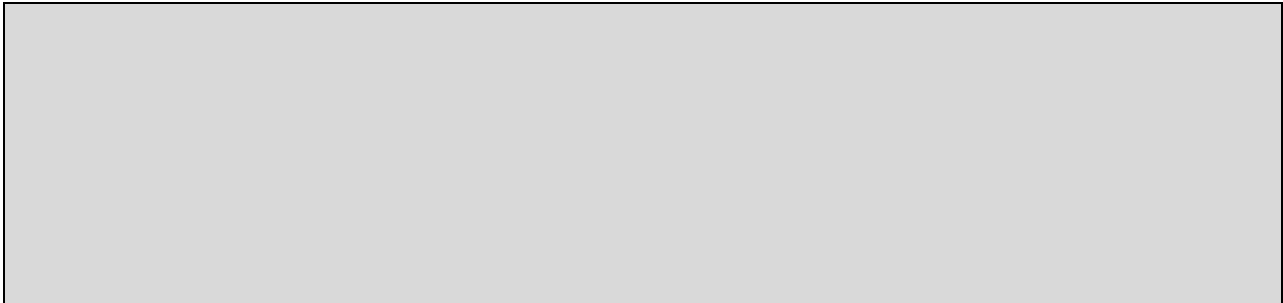
    print(row)

sql_fetch(con)

===== RESTART: C:\Users\sfaish\Desktop\Databases\step8.py =====
(1, 'Kamran', 150000.0, 'HR', 'Manager', '2017-01-04')
(2, 'Faisal', 180000.0, 'CS', 'Asst.Prof.', '2012-02-01')
(3, 'Parkash', 200000.0, 'CS', 'Assoc.Prof.', '2002-03-10')
(4, 'Lubaid Ahmed', 200000.0, 'CS', 'Asst.Prof.', '2016-02-06')
(5, 'Wasim', 20000.0, 'CS', 'Asst.Prof.', '1995-05-21')
(6, 'Fauzan', 120000.0, 'CS', 'Asst.Prof.', '2007-09-01')
>>> |
```

Figure 14.10 Fetch all data from Employee Table on IDLE Environment.

Output:



9. Close Connection

Once you are done with your database, it is a good practice to close the connection. The connection can be closed by using the close() method.

To close a connection, use the connection object and call the close() method as follows:

```
con = sqlite3.connect('mydatabase.db')

#program statements

con.close()
```

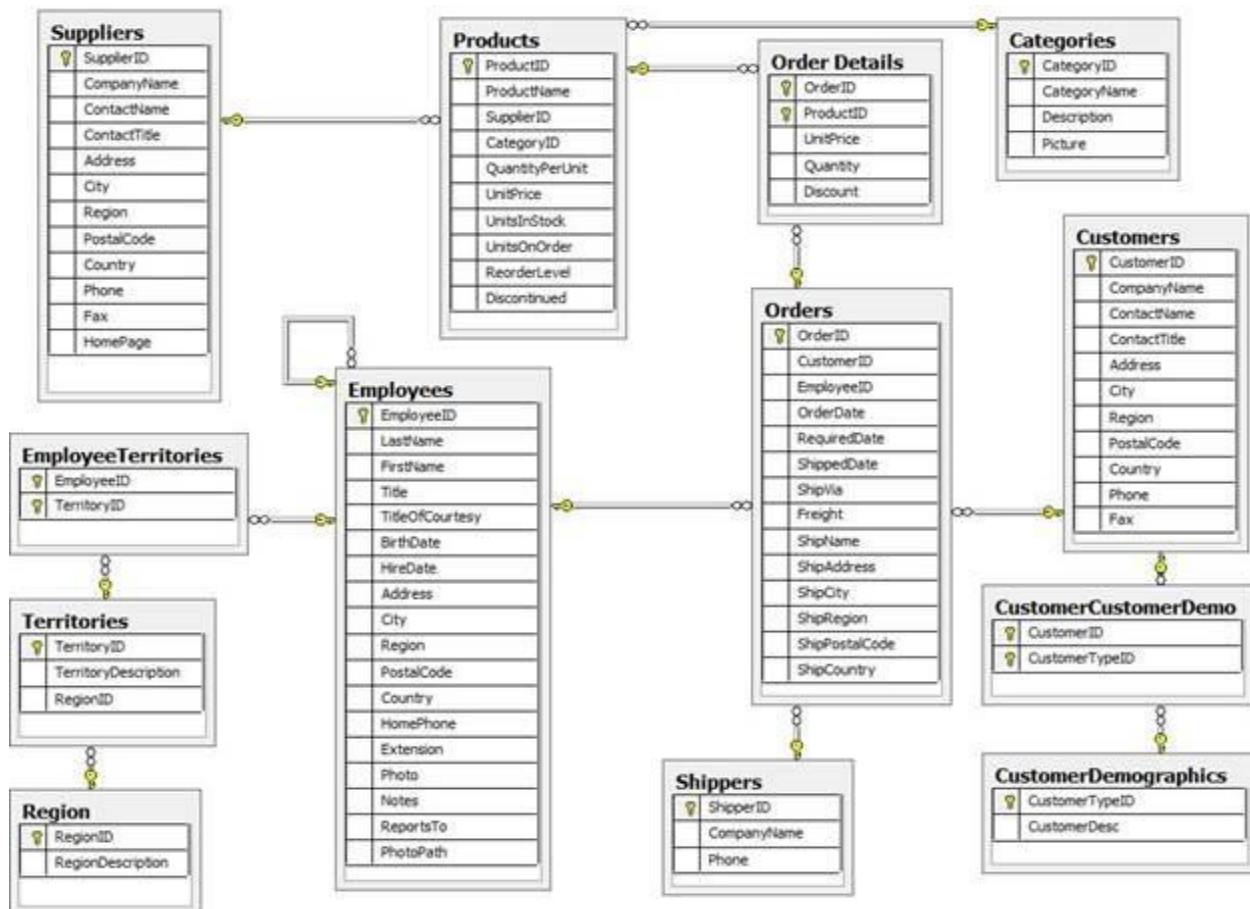
Student Name: _____

Roll No: _____

Section: _____

Programming Exercise

1. Create a database of your classmates. Having studentid as Primary key. Use maximum number of fields to store maximum data about them.
 - a. After creating data insert records of 10 friends.
 - b. Update the cell number of the friend
 - c. Delete the record of the friend who left the UIT.
 - d. Use select command to list all the number of friends you have right now with all the data inside your friend_database.
 - e. Write a query which friend is living near to you.
 - f. Write a query which friend has same age as you have.
 - g. Find all the friends having same hobbies
 - h. List all the friends who have birthday in same month.
2. Now use Northwind.db database to solve the following queries.

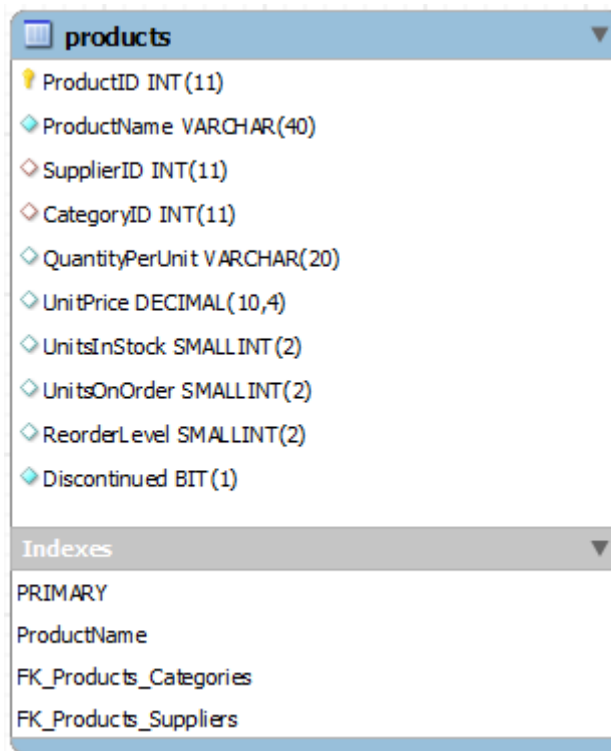


Student Name: _____

Roll No: _____

Section: _____

Consider Products Table:



products
ProductID INT(11)
ProductName VARCHAR(40)
SupplierID INT(11)
CategoryID INT(11)
QuantityPerUnit VARCHAR(20)
UnitPrice DECIMAL(10,4)
UnitsInStock SMALLINT(2)
UnitsOnOrder SMALLINT(2)
ReorderLevel SMALLINT(2)
Discontinued BIT(1)

Indexes
PRIMARY
ProductName
FK_Products_Categories
FK_Products_Suppliers

- Select * from products. Find how many rows have been returned?
- Find how many rows have been returned if categories table have been selected? What are the columns in categories table?
- Select only ProductName and QuantityPerUnit from Products table. Find number of records that has been extracted.
- If “Select * from Categories where CategoryName like “G%” is executed what type of data has been extracted?
- Analyze the query what will be outcome if “Select ProductID, ProductName from Products Where Discontinued =”1” Order By ProductName”.
- What will be the result if query e is executed but Discontinued =”0”?
- If “Select ProductName, UnitPrice from Products Order By UnitPrice Desc”, what results will you receive? Comment on the result extracted.
- Use command “Select ProductId, ProductName, UnitPrice from Products Where (((UnitPrice)<20) AND ((Discontinued)=False)) Order By UnitPrice Desc”. What results will you extract? Analyze the result.
- Find the average unit price from Products?
- Analyze the command “Select Distinct ProductName, UnitPrice from Products Where UnitPrice > (Select avg(UnitPrice) from Products) Order by UnitPrice.

Student Name: _____ Roll No: _____ Section: _____

3. Design a test_database with Book table having following data. Now create a GUI and show each query result in a Label.

Book Table {Book id, book title, book author, publisher, year of publish, price, pages, softcopy availability format, etc} where Book id is Primary Key.

4. From book chapter 12 “Databases and SQL”. Perform the solved examples and submit it as assignment. Where all the DML and DDL commands are used. Perform multiple operations using SELECT, WHERE, ORDERBY, GROUP BY, single and multiple tables known as JOINING of Tables.