In [1]:
```
#Neha Nemade
#Roll_no:22150
```

In [3]:
```
'''1. Numeric Data type
      a) Int
      b)Float
      c)Complex '''
# Int data type

x = 10
y = 20
z = x + y
print("Data type of the variable x is :-",type(x))
print("Data type of the variable y is :-",type(y))
print("Data type of the variable y is :-",type(y))
print("The value of z is:-",z)
```

```
Data type of the variable x is :- <class 'int'>
Data type of the variable y is :- <class 'int'>
Data type of the variable y is :- <class 'int'>
The value of z is:- 30
```

In [4]:
```
# float data type
k = 52.66
print("\nThe data type of variable k is:- ",type(k))
```

```
The data type of variable k is:-  <class 'float'>
```

In [5]:
```
# complex data type
m = complex(x,y)
print("\nThe complex value of m is :-",m)
print("The type of the variable m is:-",type(m))
```

```
The complex value of m is :- (10+20j)
The type of the variable m is:- <class 'complex'>
```

In [6]:
```
# int to float and vice versa
print("\nThe integer part of k is :-",int(k))
print("The float value of x is :-",float(y))
```

```
The integer part of k is :- 52
The float value of x is :- 20.0
```

In [7]:
```
'''2.Boolean datatype'''
# Examples of boolean data type
p = True
print("\nThe type of variable p is:-",type(p))
k =  1>8
print("The result of k is:-",k)
f = 45<96
print("The result of f is:-",f)
```

```
The type of variable p is:- <class 'bool'>
The result of k is:- False
The result of f is:- True
```

In [8]:
```
# We can get the integer notation of the boolean as follows :-
print(int(k))
print(int(f))
print()
'''3.Sequence data types
    a)String
    b)List
    c)Tuple'''
```

```
0
1
```

Out[8]:
```
'3.Sequence data types \n    a)String \n    b)List\n    c)Tuple'
```

In [6]:
```
# String data type
a = "Neha"
print(a)
s = "My name is Neha Nemade"
print(s)
print("The data type of the variable a is:-",type(a))
print("The data type of the variable s is:-",type(s))
print()
# Various methods in strings
k = "Neha"
```

```
Neha
My name is Neha Nemade
The data type of the variable a is:- <class 'str'>
The data type of the variable s is:- <class 'str'>
```

In [7]:
```
#1. Length of string
print("The length of the string is:-",len(k))
#2. Upper case
print("The upper case of k is",k.upper())
#2. Lower case
print("The lower case of k is",k.lower())
#3. Capitalize
k.capitalize()
s = "My name is Neha Nemade"
```

```
The length of the string is:- 4
The upper case of k is NEHA
The lower case of k is neha
```

In [8]:
```
#4. title() converts first letter of sentence to upper case
print(s.title())
print()

a = "Python"
b = "Strings"
c = a+b
print("The string after concatenation is :-",c)
#If we want space we can do as
d = a+" "+b
print("The string after concatenation is :-",d)
```

```
My Name Is Neha Nemade

The string after concatenation is :- PythonStrings
The string after concatenation is :- Python Strings
```

In [9]:
```
# string indexing
print("The first five letters of the string s are :-",s[0:5])
print()

#List data type
list = [5,9,55,'Neha',2,'python']
print(list)
print("The data type of list is ",type(list))
# slicing
print(list[0:4])
# list is mutable ie we can modify the elements
list[2]='k'
print(list)
print(type(list))
```

```
The first five letters of the string s are :- My na

[5, 9, 55, 'Neha', 2, 'python']
The data type of list is  <class 'list'>
[5, 9, 55, 'Neha']
[5, 9, 'k', 'Neha', 2, 'python']
<class 'list'>
```

In [14]:
```
#tuple data type
k = ('python',10,85,66)
print(k)
print(type(k))
```

```
('python', 10, 85, 66)
<class 'tuple'>
```

In [ ]:
```

```

In [1]:
```
#Neha Nemade
#Roll_no:22150
```

In [1]:
```
import os
import pandas as pd
import numpy as np
os.chdir("C:\Pandas")
car=pd.read_csv('Toyota.csv',index_col=0)
```

In [2]:
```
car.index
```

Out[2]:
```
Int64Index([   0,    1,    2,    3,    4,    5,    6,    7,    8,    9,
            ...
            1426, 1427, 1428, 1429, 1430, 1431, 1432, 1433, 1434, 1435],
           dtype='int64', length=1436)
```

In [3]:
```
car.columns
```

Out[3]:
```
Index(['Price', 'Age', 'KM', 'FuelType', 'HP', 'MetColor', 'Automatic', 'CC',
       'Doors', 'Weight'],
      dtype='object')
```

In [4]:
```
car.size
```

Out[4]:
```
14360
```

In [5]:
```
car.shape
```

Out[5]:
```
(1436, 10)
```

In [6]:
```
#memory use by each column in bytes

car.memory_usage()
```

Out[6]:
```
Index        11488
Price        11488
Age          11488
KM           11488
FuelType     11488
HP           11488
MetColor     11488
Automatic    11488
CC           11488
Doors        11488
Weight       11488
dtype: int64
```

In [7]:
```
car.ndim

```

Out[7]:
```
2
```

In [8]:
```
#indexing
#head-return first n rows
car.head(5)
```

Out[8]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	13500	23.0	46986	Diesel	90	1.0	0	2000	three	1165
1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	13950	24.0	41711	Diesel	90	NaN	0	2000	3	1165
3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170

In [9]:
```
#tail - -returns last n rows
car.tail(5)
```

Out[9]:

	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
1431	7500	NaN	20544	Petrol	86	1.0	0	1300	3	1025
1432	10845	72.0	??	Petrol	86	0.0	0	1300	3	1015
1433	8500	NaN	17016	Petrol	86	0.0	0	1300	3	1015
1434	7250	70.0	??	NaN	86	1.0	0	1300	3	1015
1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

In [10]:
```
#label based scalar lookup:
#selecting data based on label using 'at'
#getting data of row 4 coloum with label 'KM'

car.at[4,'KM']
```

Out[10]:
```
'38500'
```

In [11]:
```
#integer based lookup using iat
car.iat[4,9]
```

Out[11]:
```
1170
```

In [ ]:
```
#PANDAS 2
```

In [12]:
```
'''
1) Data Types
 Numeric(int64,float64)
 Character(category,object)
 2)checking Data types each column
 3)count of unique data types
 4)selecting data based on data type
 5)Concise summary of dataframe
 6)Checking Format of each column
 7)Getting Unique elements of each column
 '''
```

Out[12]:
```
'\n1) Data Types\n Numeric(int64,float64)\n Character(category,object)\n 2)chec
king Data types each column\n 3)count of unique data types\n 4)selecting data b
ased on data type\n 5)Concise summary of dataframe\n 6)Checking Format of each
column\n 7)Getting Unique elements of each column\n '
```

In [14]:
```
import os
import numpy as np
os.chdir("C:\Pandas")
car=pd.read_csv('Toyota.csv',index_col=0)
```

In [15]:
```
#checking data types each column
car.dtypes
```

Out[15]:
```
Price          int64
Age          float64
KM            object
FuelType      object
HP            object
MetColor     float64
Automatic      int64
CC             int64
Doors         object
Weight         int64
dtype: object
```

In [17]:
```
#count of unique data types

car.dtypes.value_counts()
```

Out[17]:
```
object     4
int64      4
float64    2
dtype: int64
```

In [18]:
```
#selecting data based on data types

car.select_dtypes(exclude=[object])
```

Out[18]:

	Price	Age	MetColor	Automatic	CC	Weight
0	13500	23.0	1.0	0	2000	1165
1	13750	23.0	1.0	0	2000	1165
2	13950	24.0	NaN	0	2000	1165
3	14950	26.0	0.0	0	2000	1165
4	13750	30.0	0.0	0	2000	1170
...	...	...	...	...	...	...
1431	7500	NaN	1.0	0	1300	1025
1432	10845	72.0	0.0	0	1300	1015
1433	8500	NaN	0.0	0	1300	1015
1434	7250	70.0	1.0	0	1300	1015
1435	6950	76.0	0.0	0	1600	1114

1436 rows × 6 columns

In [19]:
```
#concise summary of data frame and checking format of each column

car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Price       1436 non-null   int64  
 1   Age         1336 non-null   float64
 2   KM          1436 non-null   object 
 3   FuelType    1336 non-null   object 
 4   HP          1436 non-null   object 
 5   MetColor    1286 non-null   float64
 6   Automatic   1436 non-null   int64  
 7   CC          1436 non-null   int64  
 8   Doors       1436 non-null   object 
 9   Weight      1436 non-null   int64  
dtypes: float64(2), int64(4), object(4)
memory usage: 123.4+ KB
```

In [20]:
```
#Getting unique elements of each column

print(np.unique(car['KM']))
```

```
['1' '10000' '100123' ... '99865' '99971' '??']
```

In [21]:
```
print(np.unique(car['HP']))
```

```
['107' '110' '116' '192' '69' '71' '72' '73' '86' '90' '97' '98' '????']
```

In [22]:
```
print(np.unique(car['MetColor']))
```

```
[ 0.  1. nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan nan
 nan nan nan nan nan nan nan nan]
```

In [23]:
```
print(np.unique(car['Automatic']))
```

```
[0 1]
```

In [24]:
```
print(np.unique(car['Doors']))
```

```
['2' '3' '4' '5' 'five' 'four' 'three']
```

In [25]:
```
#Pandas 3
```

In [26]:
```
'''
1)importing data
2)concise summary of dataframe
3)converting variable's data types
4)category vs object data types
5)cleaning column 'Doors'
6)Getting count of missing values
'''
```

Out[26]:
```
"\n1)importing data\n2)concise summary of dataframe\n3)converting variable's da
ta types\n4)category vs object data types\n5)cleaning column 'Doors'\n6)Getting
count of missing values\n"
```

In [27]:
```
# importing data
import os
import numpy as np
os.chdir("C:\Pandas")
car=pd.read_csv('Toyota.csv',index_col=0)
car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Price       1436 non-null   int64  
 1   Age         1336 non-null   float64
 2   KM          1436 non-null   object 
 3   FuelType    1336 non-null   object 
 4   HP          1436 non-null   object 
 5   MetColor    1286 non-null   float64
 6   Automatic   1436 non-null   int64  
 7   CC          1436 non-null   int64  
 8   Doors       1436 non-null   object 
 9   Weight      1436 non-null   int64  
dtypes: float64(2), int64(4), object(4)
memory usage: 123.4+ KB
```

In [28]:
```
#concise summary of dataframe:Now , importing the data considering data frames
car=pd.read_csv('Toyota.csv',index_col=0,na_values=["??","???"])
car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Price       1436 non-null   int64  
 1   Age         1336 non-null   float64
 2   KM          1421 non-null   float64
 3   FuelType    1336 non-null   object 
 4   HP          1436 non-null   object 
 5   MetColor    1286 non-null   float64
 6   Automatic   1436 non-null   int64  
 7   CC          1436 non-null   int64  
 8   Doors       1436 non-null   object 
 9   Weight      1436 non-null   int64  
dtypes: float64(3), int64(4), object(3)
memory usage: 123.4+ KB
```

In [30]:
```
#converting data types
car['MetColor']=car['MetColor'].astype('object')
car['Automatic']=car['Automatic'].astype('object')
car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Price      1436 non-null   int64
 1   Age        1336 non-null   float64
 2   KM         1421 non-null   float64
 3   FuelType   1336 non-null   object
 4   HP         1436 non-null   object
 5   MetColor   1286 non-null   object
 6   Automatic  1436 non-null   object
 7   CC         1436 non-null   int64
 8   Doors      1436 non-null   object
 9   Weight     1436 non-null   int64
dtypes: float64(2), int64(3), object(5)
memory usage: 123.4+ KB
```

In [31]:
```
#category vs data object type

print('Memory whwn data is of object Type')
car['FuelType'].nbytes
```

```
Memory whwn data is of object Type
```

Out[31]:
```
11488
```

In [32]:
```
print('Memory whwn data is of category Type')
car['FuelType'].astype('category').nbytes
```

```
Memory whwn data is of category Type
```

Out[32]:
```
1460
```

In [33]:
```
#cleaning column doors
#useof replace()

print(np.unique(car['Doors']))
car['Doors'].replace('three',3,inplace=True)
car['Doors'].replace('four',4,inplace=True)
car['Doors'].replace('five',5,inplace=True)
```

```
['2' '3' '4' '5' 'five' 'four' 'three']
```

In [34]:
```
car['Doors']=car['Doors'].astype('int64')
print(np.unique(car['Doors']))
```

```
[2 3 4 5]
```

In [35]:
```
#Getting count of missing values

car.isnull().sum()
```

Out[35]:
```
Price          0
Age          100
KM            15
FuelType     100
HP             0
MetColor     150
Automatic      0
CC             0
Doors          0
Weight         0
dtype: int64
```

In [ ]:
```

```

In [ ]:
```
#Neha Nemade
#Roll_no:22150
```

In [1]:
```
my_list=[1,2,3,4,5,6]
print(my_list)
```

```
[1, 2, 3, 4, 5, 6]
```

In [2]:
```
import numpy as np
```

In [3]:
```

array=np.array(my_list,dtype=int)
print(array)
```

```
[1 2 3 4 5 6]
```

In [4]:
```
print(type(array))
print(len(array))
print(array.ndim)
print(array.shape)
```

```
<class 'numpy.ndarray'>
6
1
(6,)
```

In [5]:
```
array2=array.reshape(3,2)
print(array2)
array2.shape
```

```
[[1 2]
 [3 4]
 [5 6]]
```

Out[5]:
```
(3, 2)
```

In [7]:
```
array3=array.reshape(3,-1)
print(array3)
print(array3.ndim)
```

```
[[1 2]
 [3 4]
 [5 6]]
2
```

In [11]:
```
##Intializing numpy arrays from nested python lists

my_list2=[1,2,3,4,5]
my_list3=[2,3,4,5,6]
my_list4=[9,7,6,8,9]

mul_arr=np.array([my_list2,my_list3,my_list4])
print(mul_arr)
print(mul_arr.shape)
```

```
[[1 2 3 4 5]
 [2 3 4 5 6]
 [9 7 6 8 9]]
(3, 5)
```

In [12]:
```
mul_arr.reshape(1,15)
```

Out[12]:
```
array([[1, 2, 3, 4, 5, 2, 3, 4, 5, 6, 9, 7, 6, 8, 9]])
```

In [13]:
```
#NUMPY Attributes

a=np.array([[1,2,3],[4,5,6]])
print(a.shape)
```

```
(2, 3)
```

In [14]:
```
#reshaping the ndarray
a.shape=(3,2)
print(a)
```

```
[[1 2]
 [3 4]
 [5 6]]
```

In [15]:
```
#reshape function to resize an array
b=a.reshape(3,2)
print(b)
```

```
[[1 2]
 [3 4]
 [5 6]]
```

In [22]:
```
r=range(24)
print(r)
```

```
range(0, 24)
```

In [23]:
```
#an array of evenly spaced numbers
a=np.arange(24)
print(a)
print(a.ndim)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23]
1
```

In [24]:
```
#reshaping the array 'a'
b=a.reshape(6,4,1)
print(b)
```

```
[[[ 0]
  [ 1]
  [ 2]
  [ 3]]

 [[ 4]
  [ 5]
  [ 6]
  [ 7]]

 [[ 8]
  [ 9]
  [10]
  [11]]

 [[12]
  [13]
  [14]
  [15]]

 [[16]
  [17]
  [18]
  [19]]

 [[20]
  [21]
  [22]
  [23]]]
```

In [26]:
```
#dtype of array is int8(1 byte)
x=np.array([1,2,3,4,5],dtype=np.int8)
print(x.itemsize)
```

```
1
```

In [27]:
```
#dtype of array is now float32(4 bytes)
x=np.array([1,2,3,4,5],dtype=np.float32)
print(x.itemsize)
```

```
4
```

In [29]:
```
x=np.array([[1,2],[3,4]],dtype=np.float64)
y=np.array([[5,6],[7,8]],dtype=np.float64)
print(x)
print(y)
```

```
[[1. 2.]
 [3. 4.]]
[[5. 6.]
 [7. 8.]]
```

In [30]:
```
print(x+y)
print(np.add(x,y))
```

```
[[ 6.  8.]
 [10. 12.]]
[[ 6.  8.]
 [10. 12.]]
```

In [32]:
```
print(x-y)
print(np.subtract(x,y))
```

```
[[-4. -4.]
 [-4. -4.]]
[[-4. -4.]
 [-4. -4.]]
```

In [33]:
```
print(x*y)
print(np.multiply(x,y))
print(x.dot(y))
```

```
[[ 5. 12.]
 [21. 32.]]
[[ 5. 12.]
 [21. 32.]]
[[19. 22.]
 [43. 50.]]
```

In [34]:
```
print(x.dot(y))
print(np.dot(x,y))
```

```
[[19. 22.]
 [43. 50.]]
[[19. 22.]
 [43. 50.]]
```

In [35]:
```
print(x/y)
print(np.divide(x,y))
```

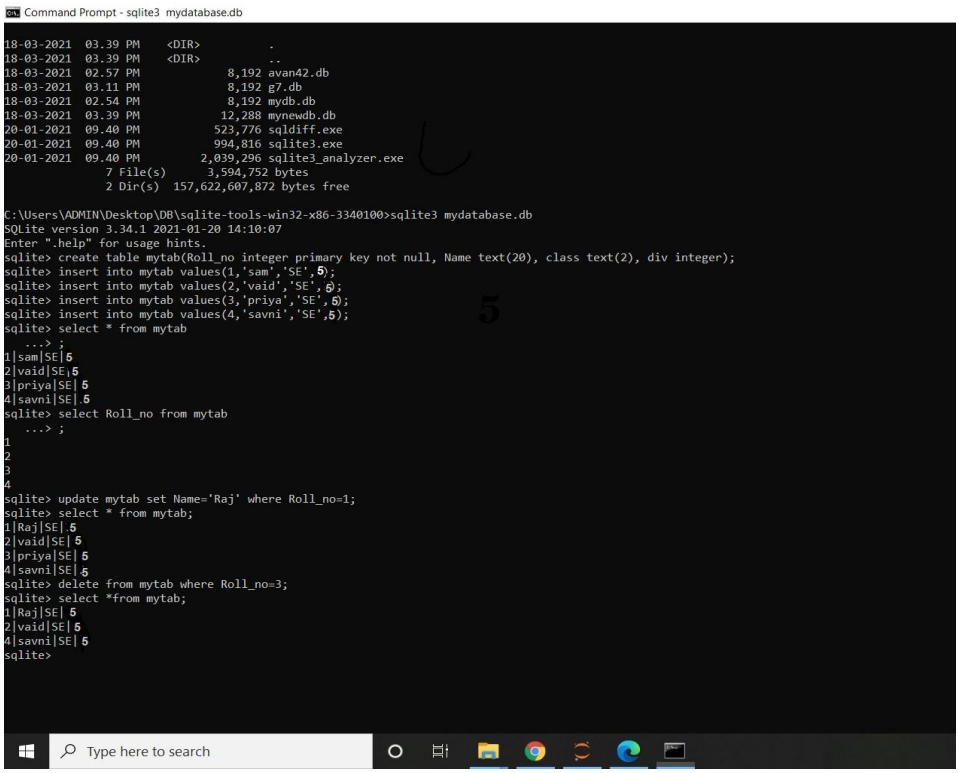
```
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
```

In [36]:
```
print(np.sum(x))
print(np.sum(x,axis=0))
print(np.sum(x,axis=1))
```

```
10.0
[4. 6.]
[3. 7.]
```

In [ ]:
```

```



In [ ]:
```
#ROLL NO:22150
#Name: Neha Nemade
```

In [1]:
```
import sqlite3 as sq
```

In [2]:
```
db=sq.connect('Employee.db')
```

In [3]:
```
cur=db.cursor()
```

In [4]:
```
cur.execute('''CREATE TABLE employee7(Emp_Id text(20) primary key , Name text(20
```

Out[4]:
```
<sqlite3.Cursor at 0x149eabf08f0>
```

In [5]:
```
def new_entry(cur,db):
    Emp_ID=input('Enter the Emp_ID of employee: ')
    Name=input('Enter the Name: ')
    DoB=input('DoB of employee: ')
    Qualification=input('Enter Qualification of the employee : ')
    Adhar=int(input('Enter the Adhar no. of employee: '))
    PAN=input('Enter the Pan  no. of employee: ')
    try:
        cur.execute('''insert into employee values(?,?,?,?,?,?)''',(Emp_ID, Name
        db.commit()
    except:
        print('Enter unique Emp_ID')
```

In [6]:
```
def display(cur):
    cur.execute('''select * from employee ''')
    if cur:
        for i in cur:
            print(i)
    else:
        print('Db is empty')
```

In [7]:
```
def update(cur,db):
    Emp_ID=input('Enter Emp_ID which u want to update: ')
    cur.execute('''SELECT * from employee where Emp_ID is "{}"'''.format(Emp_ID)
    if cur:
        Name=input('Enter the Name: ')
        DoB=input('DoB of employee: ')
        Qualification=input('Enter Qualification of the employee : ')
        Adhar=int(input('Enter the Adhar no. of employee: '))
        PAN=input('Enter the Pan  no. of employee: ')
        cur.execute('''UPDATE employee
                       SET Name ="{}",DoB='{}',Qualification='{}',Adhar={},PAN=
                       WHERE Emp_ID="{}";'''.format(Name,DoB,Qualification,Adha
    else:
        print('Emp_ID do not exit')
```

In [8]:
```
def delete(cur,db):
    Emp_ID=input('Enter Emp_ID which u want to delete: ')
    cur.execute('''SELECT * from employee where Emp_ID is "{}"'''.format(Emp_ID)
    if cur:
        cur.execute('''DELETE from employee
                       WHERE Emp_ID="{}";'''.format(Emp_ID))
    else:
        print('Emp_ID do not exit')
```

In [9]:
```
print("\t----------------------------")
print('\t1.Create New Data Entry\n\t2.Display All Data\n\t3.Update Data\n\t4.Del
print("\t----------------------------")

while(True):
    c=int(input('Enter your choice'))
    if(c==1):
        new_entry(cur,db)
    elif(c==2):
        display(cur)
    elif(c==3):
        update(cur,db)
    elif(c==4):
        delete(cur,db)
    else:
        print('You have quit !!!')
        break;
```

```
        --------------------
        1.Create New Data Entry
        2.Display All Data
        3.Update Data
        4.Delete Data Entry
        5.Exit
        --------------------
Enter your choice1
Enter the Emp_ID of employee: Neha
Enter the Name: Neha
DoB of employee: 05/06/01
Enter Qualification of the employee : se
Enter the Adhar no. of employee: 852140000
Enter the Pan  no. of employee: gfh2222
Enter your choice2
('Neha', 'Neha1', '05/06/01', 'se', 852140000, 'gfh2222')Enter
your choice3
Enter Emp_ID which u want to update: Neha
Enter the Name: Prasad
DoB of employee: 21/5/2002
Enter Qualification of the employee : 12
Enter the Adhar no. of employee: 782100000
Enter the Pan  no. of employee: bgf34
Enter your choice2
('Neha', 'Prasad', '21/5/2002', '12', 782100000, 'bgf34')
Enter your choice6
You have quit !!!
```

In [ ]:

In [ ]:
```
#ROLL NO:22150
#NAME: Neha Kamalakar Nemade
```

In [1]:
```
import os
```

In [2]:
```
os.chdir('C:/Users/ADMIN/Desktop/My DataBase')
```

In [3]:
```
import sqlite3 as sq

db = sq.connect('mynewdb.db')
cur = db.cursor()
cur.execute('''create table mynewtab14(Roll_no integer primary key, Name text, M
db.close()
```

In [4]:
```
db = sq.connect('mynewdb.db')
cur = db.cursor()
cur.execute('''insert into mynewtab14
values(1,'Siddhant',99);''')db.commit()
db.close()
```

In [5]:
```
db = sq.connect('mynewdb.db')
cur = db.cursor()
cur.execute('''insert into mynewtab14
values(2,'Gayatri',89);''')db.commit()
db.close()
```

In [6]:
```
db = sq.connect('mynewdb.db')
cur = db.cursor()
cur.execute('''insert into mynewtab14 values(3,'Neha',80);''')
db.commit()
db.close()
```

In [7]:
```
db = sq.connect('mynewdb.db')
cur = db.cursor()
cur.execute('''select * from mynewtab11;''')
x = cur.fetchall()
print(x)
db.close()
```

```
[(1, 'Siddhant', 99), (2, 'Gayatri', 89), (3, 'Neha', 80)]
```

In [8]:
```
mydb = sq.connect('myschool.db')
mycur = mydb.cursor()
myid = int(input("Enter ID : "))
myname = input("Enter Name : ")
mymarks = float(input("Enter marks : "))

mydb.commit()
mydb.close()
```

```
Enter ID : 45
Enter Name : Neha
Enter marks : 45
```

In [9]:
```
def insertdata(myid, myname, mymarks):
    db = sq.connect('myschool.db')
    cur = db.cursor()
    cur.execute('''create table mynewtab15(Roll_no integer primary key, Name tex
    q = '''insert into mynewtab15 values(4,"savani",90);'''
    data = (myid,myname,mymarks)
    #cur.execute(q,data)
    db.commit()
    db.close()

insertdata(10,'Gayatri',45)
```

In [10]:
```
def update(myatt,myval,myid):
    db=sq.connect('mydb.db')
    cur=db.cursor()

    if(myatt=='Name'):
        q='''update mynewtab15 set Name='savani' where ID= 4'''
    elif(myatt== 'Division'):
         q='''update mynewtab15 set Division="shruti" '''

    data=(myval,myid)
    #cur.execute(q,data)
    print('updated successfully ..\n')
    db.close()
    display()

update('Name','Thomas',7)    #Updatename
update('Division','c',7)     #UpdateDivision
```

```
updated successfully ..

updated successfully ..
```

In [ ]:
```

```