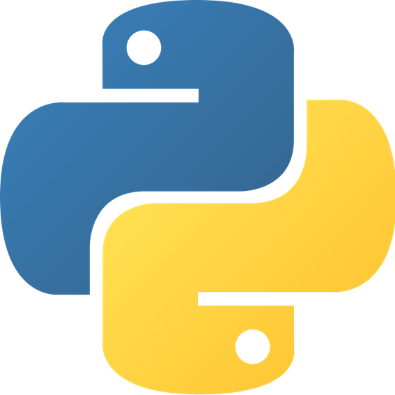
CS PRACTICAL FILE

BY ISHAAN RASTOGI 12 B



****

Name- Ishaan Rastogi  
Class- 12th B  
Roll No: 16

**#1: WAP to take data from the user & put in a list if the data is even**

l=[]

C='y'

while C in ('y','Y'):

i=int(input('Enter a no.:'))

r=i%2

if r==0:

l.append(i)

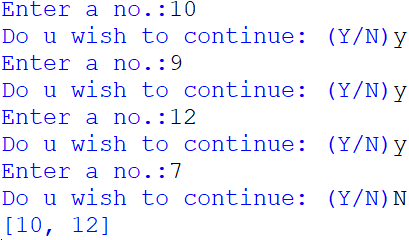
C=input('Do u wish to continue: (Y/N)')

else:

C=input('Do u wish to continue: (Y/N)')

print(l)

INPUT & OUTPUT:



**#2: WAP to find frequency of a no in a given list**

l=[]

n=int(input('Enter the no. of numbers going to be added:'))

for i in range (n):

a=int(input('Enter the no.:'))

l.append(a)

N=int(input('Enter the no. whose frequency needs to be checked:'))

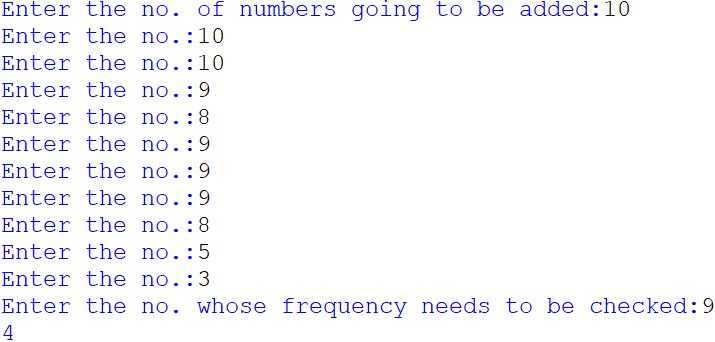
b=0

for i in l:

if i==N:

b+=1

print(b)  
INPUT & OUTPUT:



**#3: WAP to find the sum of items in a list**

list = [40,30,70,80,0,7,23]

sum = 0

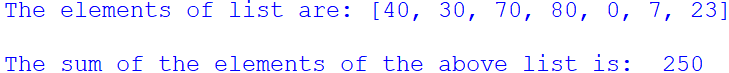
for i in list:

sum+=i

print('The elements of list are:',list)

print(' ')

print('The sum of the elements of the above list is: ',sum)  
INPUT & OUTPUT:



**#4: WAP to count the no. of words in a sentence**

str=input('Enter a sentence:')

x=str.split()

print('No. of words in the sentence are:',len(x))

INPUT & OUTPUT:

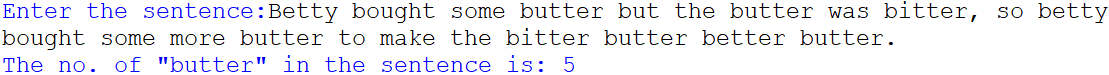


**#5: WAP to count the no. of 'butter' in a sentence**

str=input('Enter the sentence:')

print('The no. of "butter" in the sentence is:' ,str.count('butter'))

INPUT & OUTPUT:



**#6: WAP to sort a given list using the bubble sort method**

n=int(input('Enter the no. of items in a list:'))

l=[]

for i in range(n):

a=int(input('Enter a value:'))

l.append(a)

print('unsorted list:',l)

for i in range(len(l)-1):

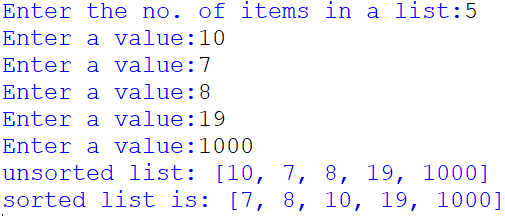
for j in range(len(l)-i-1):

if l[j]>l[j+1]:

l[j],l[j+1]=l[j+1],l[j]

print('sorted list is:',l)

INPUT & OUTPUT:



**#7: WAP to to make a calculator for basic calculations with operators (+ , – , \* , / )**

a=int(input("Enter 1st no: "))

b=int(input("Enter 2nd no: "))

op=input("Enter the operator (+,-,\*,/): ")

if(op=="+"):

c=a+b

print("Sum = ",c)

elif(op=="\*"):

c=a\*b

print("Product = ",c)

elif(op=="-"):

if(a>b):

c=a-b

else:

c=b-a

print("Difference = ",c)

elif(op=="/"):

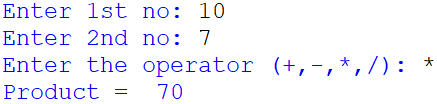
c=a/b

print("Division = ",c)

else:

print("Invalid operator")

INPUT & OUTPUT:



**#8: WAP to maintain book details like book code, book title and price using stacks data structures**

book=[]

def push():

bcode=input("Enter bcode: ")

btitle=input("Enter btitle: ")

price=input("Enter price: ")

bk=(bcode,btitle,price)

book.append(bk)

def pop():

if(book==[]):

print("Underflow / Book Stack in empty")

else:

bcode,btitle,price=book.pop()

print("popped element is: ")

print("bcode ",bcode," btitle ",btitle," price ",price)

def traverse():

if not (book==[]):

n=len(book)

for i in range(n-1,-1,-1):

print(book[i])

else:

print("Empty , No book to display")

while True:

print("1. Push")

print("2. Pop")

print("3. Traversal")

print("4. Exit")

ch=int(input("Enter your choice: "))

if(ch==1):

push()

elif(ch==2):

pop()

elif(ch==3):

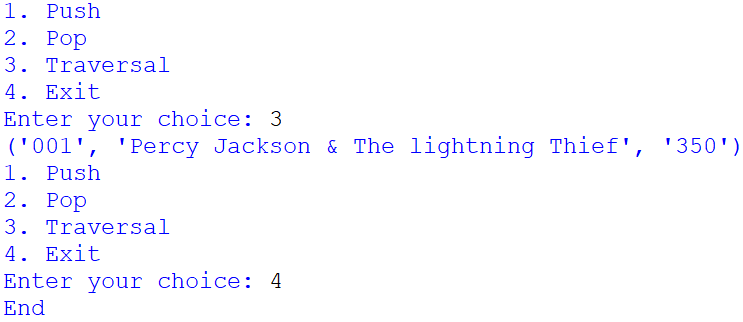
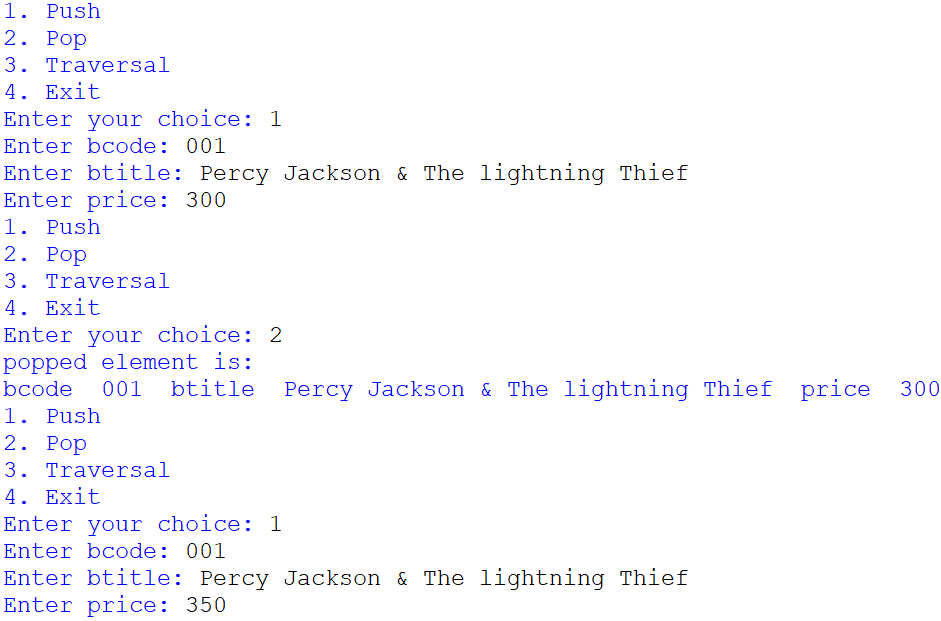
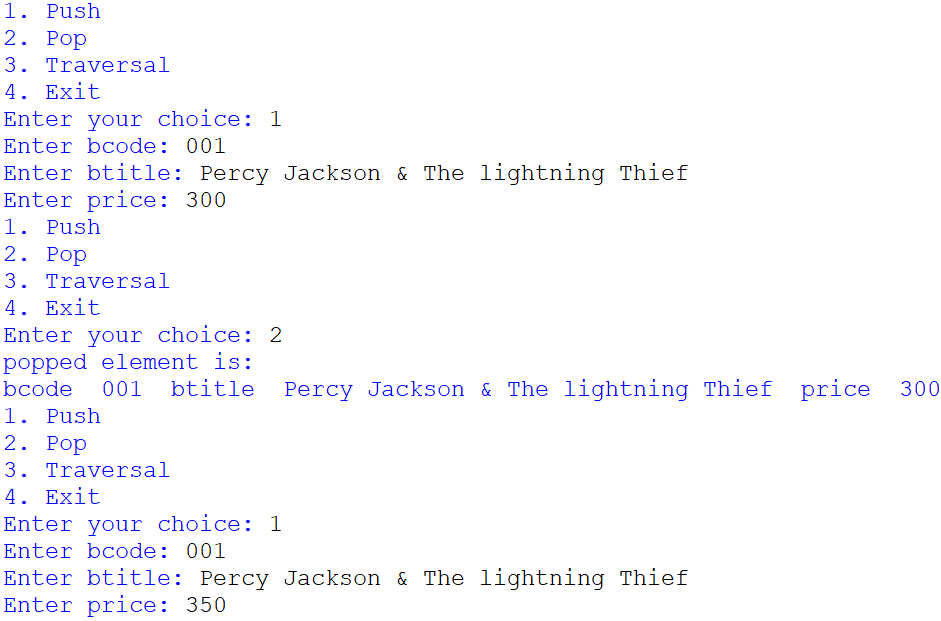
traverse()

elif(ch==4):

print("End")

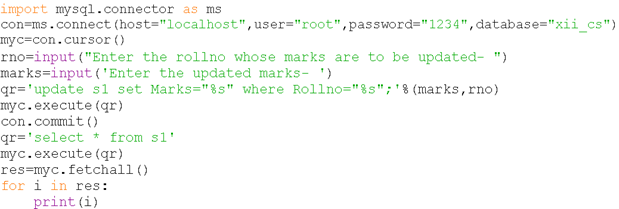
break

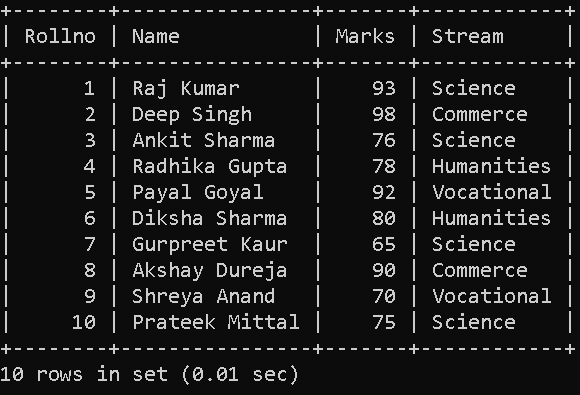
else:

print("Invalid choice")

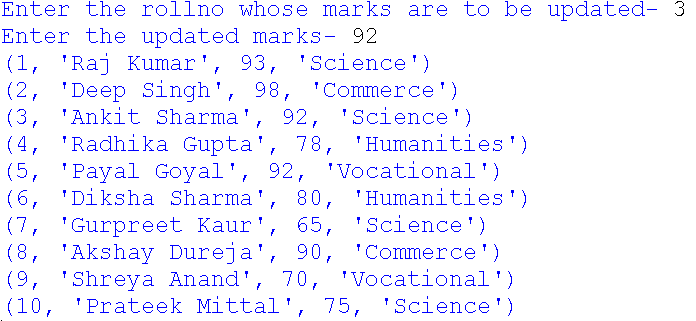
* Interface Python With SQL

1. Update values into MySQL table s1

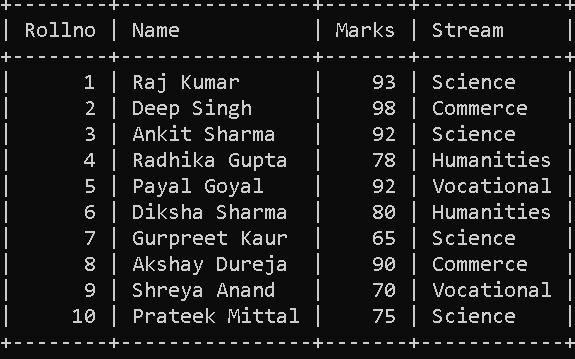
**SOURCE CODE**-  
**Table before updating-**



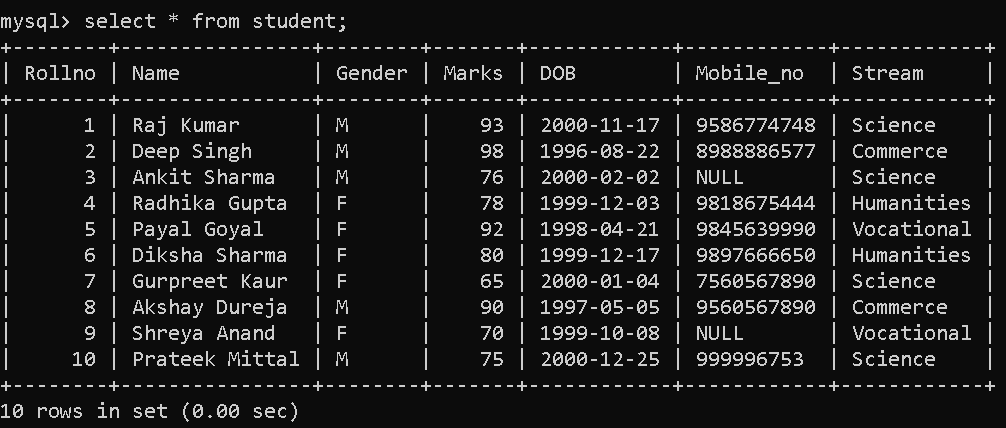
OUTPUT:

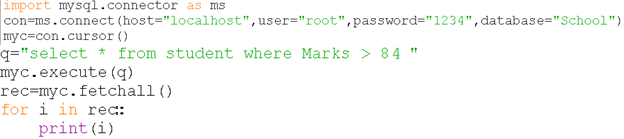


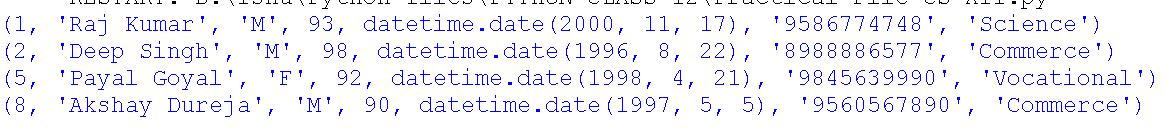
**Table after updating-**



1. Write a code to connect to a MySQL database named School & then fetch all those records from table Student where marks are more than 84.



**SOURCE CODE**- OUTPUT:

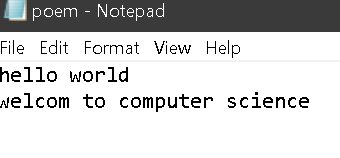


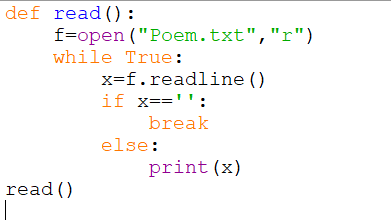
* Data FILE handling

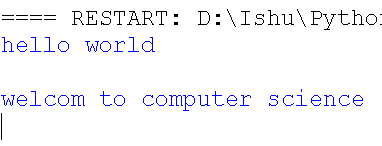
TEXT FILE

1. WAP to read the content from a text file ‘poem.txt’ line by line and display the same on the screen

**>>> poem.txt <<<**

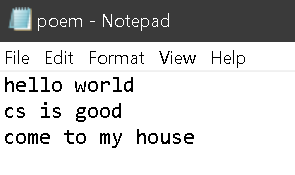
****

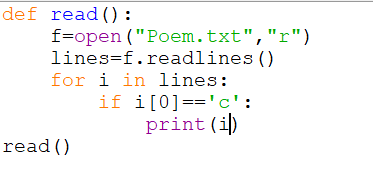
**SOURCE CODE**-  


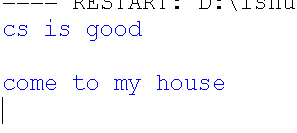
OUTPUT:  


1. WAP to read the content from a text file ‘poem.txt’ & display those lines which are starting with an alphabet C.

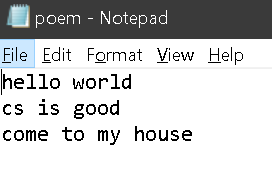
**>>> poem.txt <<<**

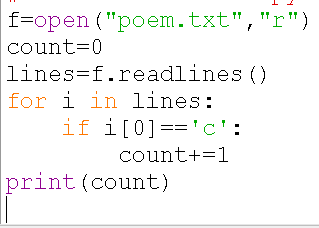


**SOURCE CODE**-  
****

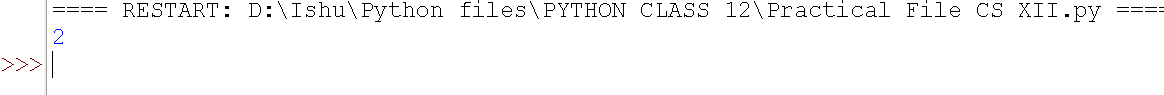
OUTPUT:  
****

1. Write a method in python to read the content from a text file ‘poem.txt’ and count those lines which are starting with an alphabet C.

**>>> poem.txt <<<  
**

**SOURCE CODE**-  
****

OUTPUT:

****

BINARY FILE

**#14: Create a binary file student.dat to hold students’ records like Roll no., name, address using the list. Write functions to write data, read them, and print on the screen.**

import pickle

rec=[]

def file\_create():

f=open("student.dat","wb")

rno = int(input("Enter Student Roll No:"))

sname = input("Enter Student Name:")

address = input("Enter Address:")

rec=[rno,sname,address]

pickle.dump(rec,f)

def read\_data():

f = open("student.dat","rb")

print("\*"\*85)

print("Data stored in File....")

rec=pickle.load(f)

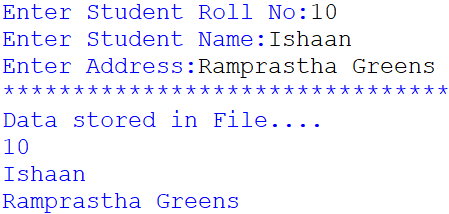
for i in rec:

print(i)

file\_create()

read\_data()

INPUT & OUTPUT:



**#15: WAP to store customer data into a binary file cust.dat using a dictionary & print them on screen after reading them. The customer data contains ID as key, and name, city as values.**

import pickle

def dict():

f = open("cust.dat","wb")

d = {'C001':['Ishaan','Ghaziabad'],'C002':['Modi','Vadnagar'],'C003':['Ram','Faridabad']}

pickle.dump(d,f)

f.close()

f = open("cust.dat","rb")

d = pickle.load(f)

print(d)

f.close()

dict()

INPUT & OUTPUT:



**#16:** **Write a function to write data into binary file marks.dat and display the records of students who scored more than 95 marks.**

import pickle  
def search\_95plus():

f = open("marks.dat","ab")

while True:

rn=int(input("Enter the rollno:"))  
sname=input("Enter the name:")  
marks=int(input("Enter the marks:"))  
rec=[]  
data=[rn,sname,marks]  
rec.append(data)  
pickle.dump(rec,f)  
ch=input("Wnat more records?Yes:")  
if ch.lower() not in 'yes':

break

f.close()  
f = open("marks.dat","rb"  
cnt=0  
try:

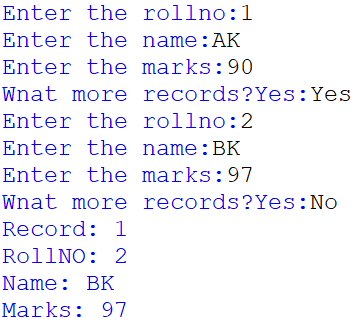
while True:  
 data = pickle.load(f)  
 for s in data:  
 if s[2]>95:

cnt+=1  
print("Record:",cnt)  
print("RollNO:",s[0])  
print("Name:",s[1])  
print("Marks:",s[2])

except Exception:

f.close()

search\_95plus()  
INPUT & OUTPUT:



CSV FILE

**#17: Read a CSV file top5.csv and print the contents in a proper format. The data for top5.csv file are as following:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.No. | Batsman | Team | Runs | Highest |
| 1 | K L Rahul | KXI | 670 | 132\* |
| 2 | S Dhawan | DC | 618 | 106\* |
| 3 | David Warner | SRH | 548 | 85\* |
| 4 | Shreyas Iyer | DC | 519 | 88\* |
| 5 | Ishan Kishan | MI | 516 | 99 |

from csv import reader

def top5():

f= open("top5.csv","r")

d = reader(f)

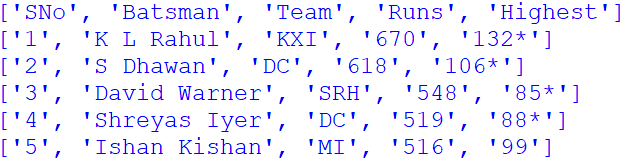
data=list(d)

for i in data:

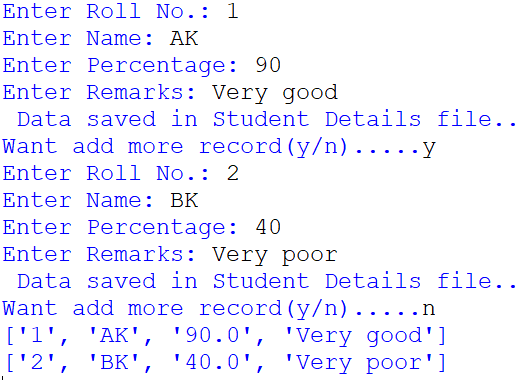
print(i)

f.close()

top5()

OUTPUT: 

**#18: WAP to perform read and write operation onto a student.csv file having fields as roll number, name, stream and percentage.**

import csv  
with open('Student\_Details.csv','w',newline='') as csvf:  
     writecsv=csv.writer(csvf,delimiter=',')  
     choice='y'  
     while choice.lower()=='y':  
          rl=int(input("Enter Roll No.: "))  
          n=input("Enter Name: ")  
          p=float(input("Enter Percentage: "))  
          r=input("Enter Remarks: ")  
          writecsv.writerow([rl,n,p,r])  
          print(" Data saved in Student Details file..")  
          choice=input("Want add more record(y/n).....")  
            
with open('Student\_Details.csv','r',newline='') as fileobject:  
     readcsv=csv.reader(fileobject)  
     for i in readcsv:  
          print(i)  
INPUT & OUTPUT: 

# #19: WAP to search the record of a particular student from CSV file on the basis of inputted name.

import csv

#input Roll number you want to search

number = input('Enter number to find: ')  
found=0  
#read csv, and split on "," the line

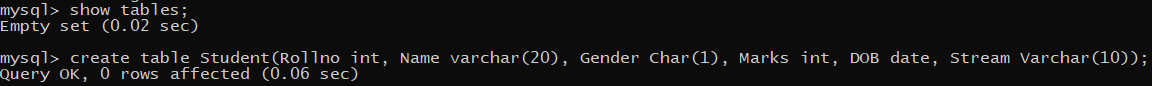
with open('Student\_Details.csv') as f:  
     csv\_file = csv.reader(f, delimiter=",")  
     #loop through csv list  
     for row in csv\_file:

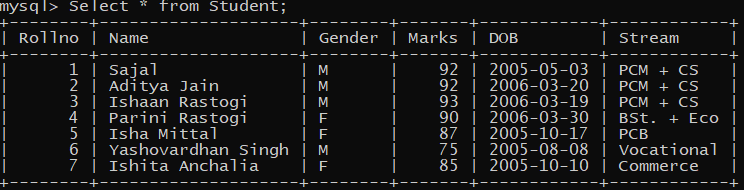
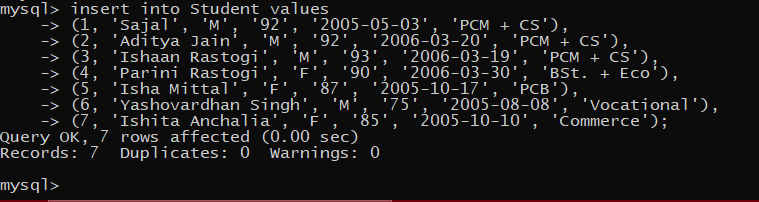
     #if current rows index value (here 0) is equal to input, print that row  
          if number ==row[0]:  
              print (row)  
              found=1  
          else:  
               found=0  
if found==1:  
     pass

else:  
     print("Record Not found")

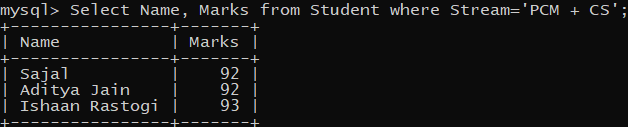
# INPUT & OUTPUT:

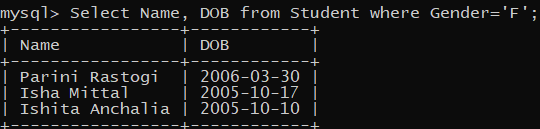
#Relational Database and SQL

Q.20  


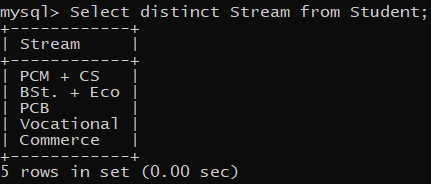


**Queries-**

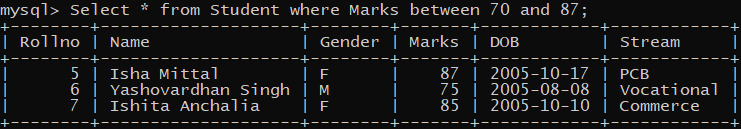
1. Display name & marks of PCM+CS students
2. Display name & DOB of female students



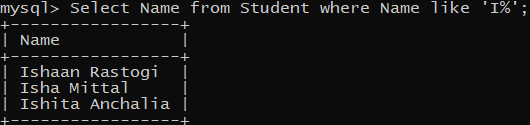
1. Display unique streams



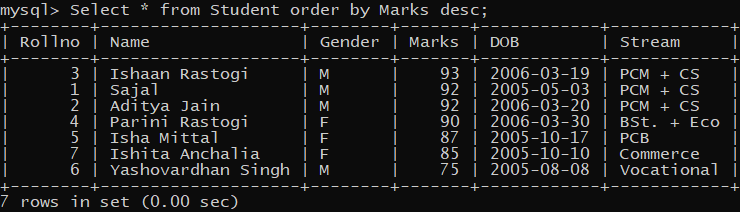
1. Display details of students whose marks are in the range 70-87

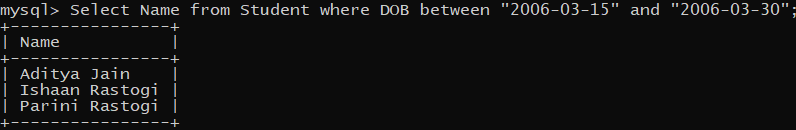


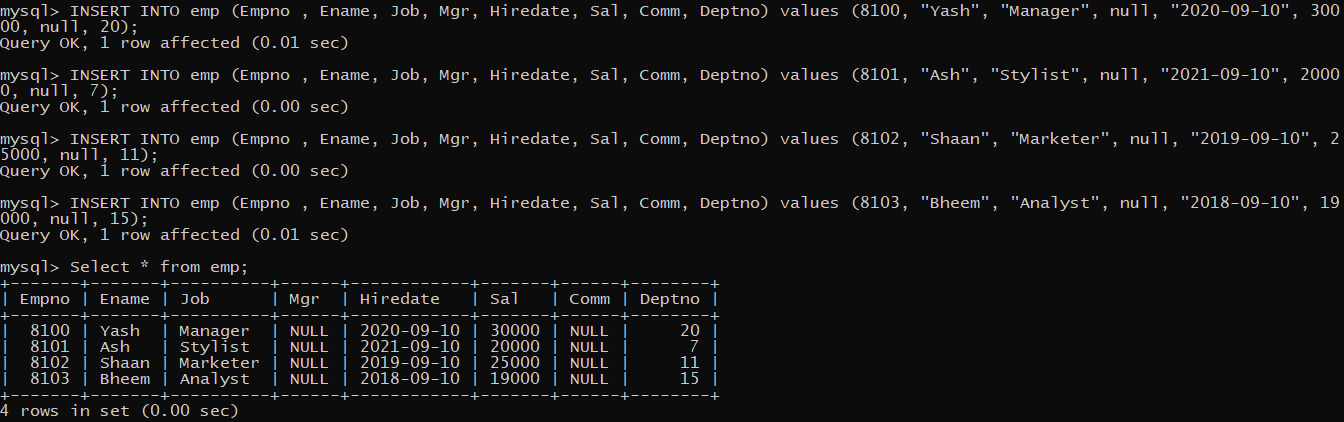
1. Display name of students whose name start with Letter I

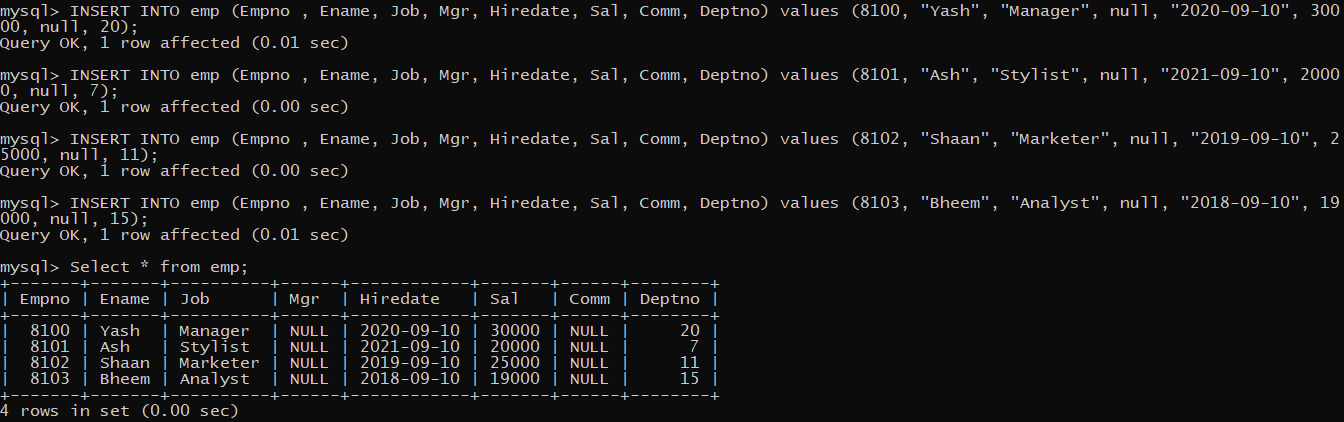


1. Display details of students in descending order of their DOB

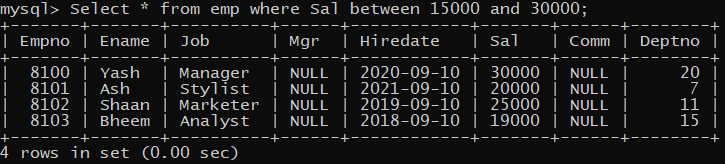
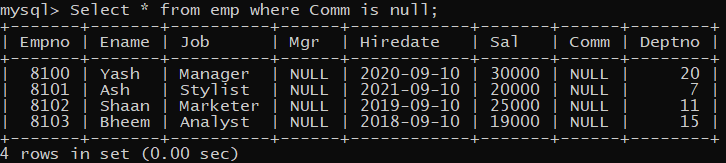
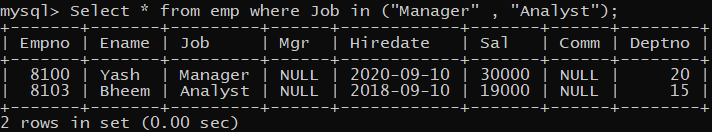
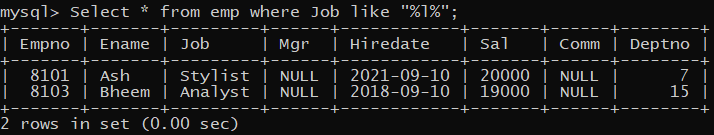
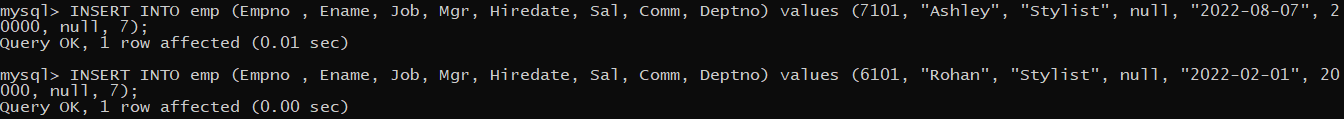
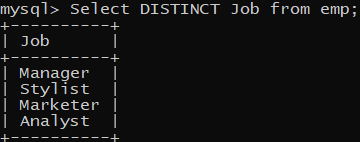
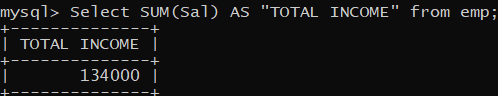
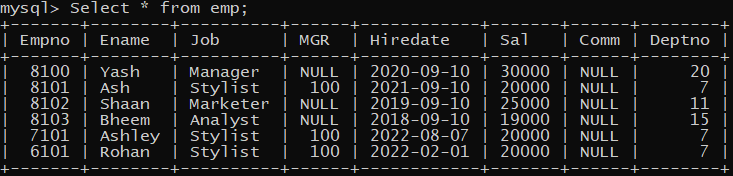


1. Display the Name of students whose DOB is between 2006-03-15 & 2006-03-30

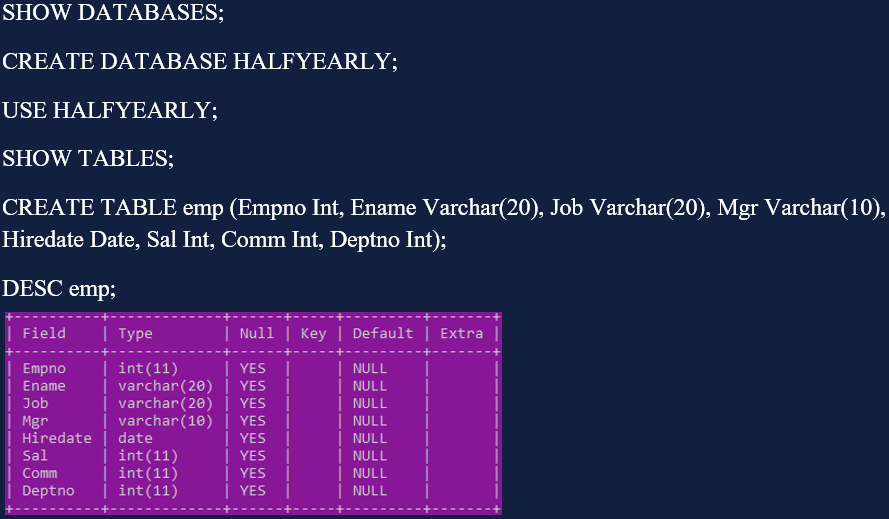
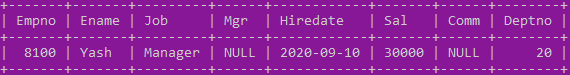
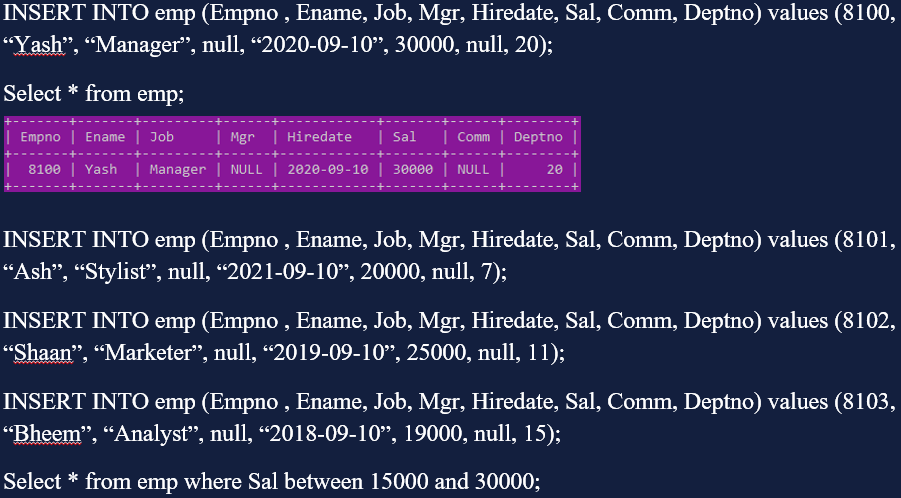
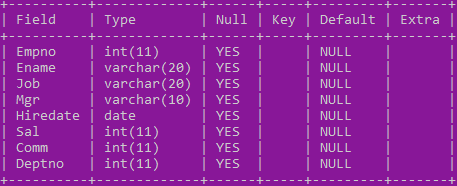
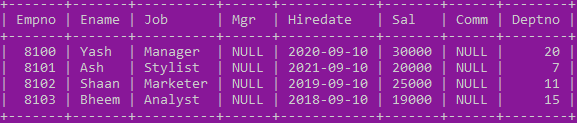
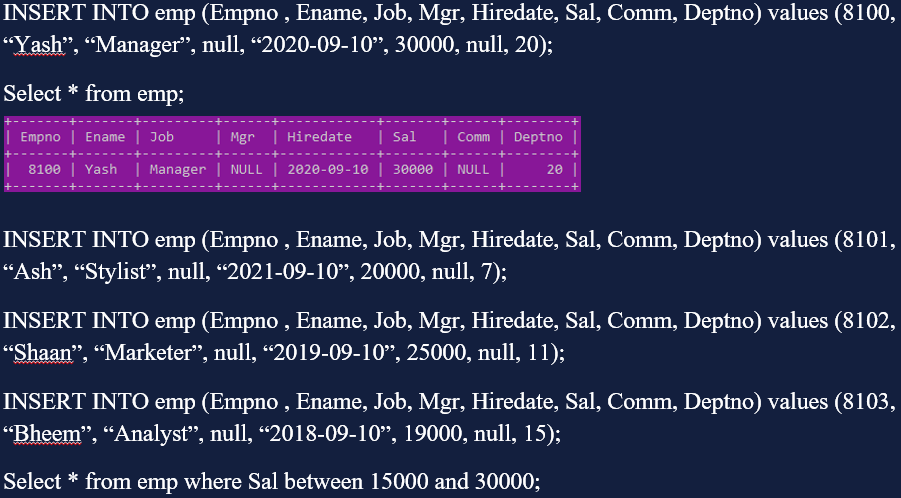
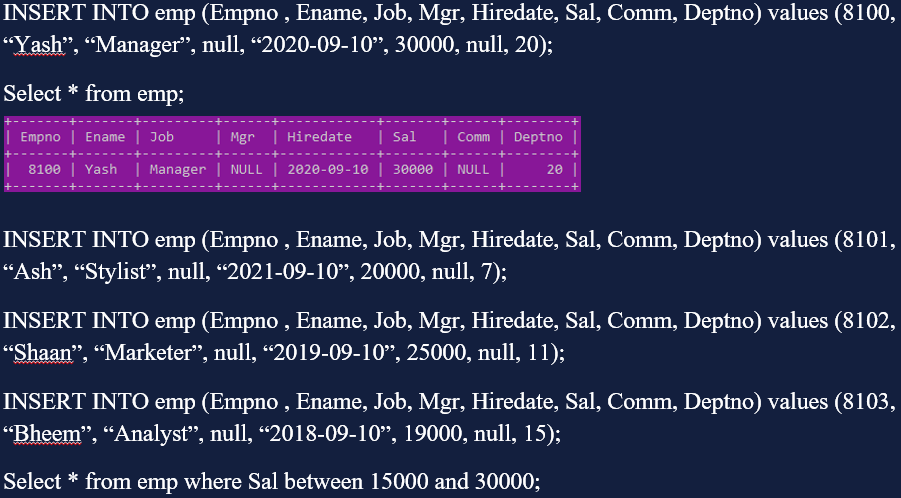
Q.21  


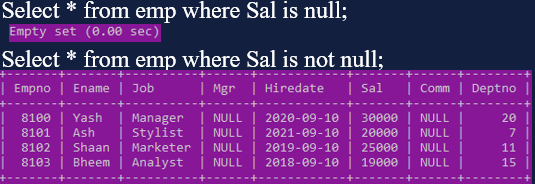


**Queries-**

1. Display details of Employee where Salary is between 15000 and 30000 
2. Display details of Employee where Commission is null 
3. Display details of Employee where Job is Manager and Analyst 
4. Display details of employee where Job consists the letter l 
5. Insert the records which were left behind 
6. Display Unique Jobs   
   
7. Display Salary of all employees as Total Income 
8. Alter Table ( Drop A Column, Modify a value and add a Column)    
9. 

Q.22

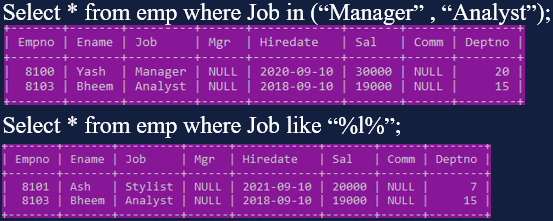
    
**Queries-**

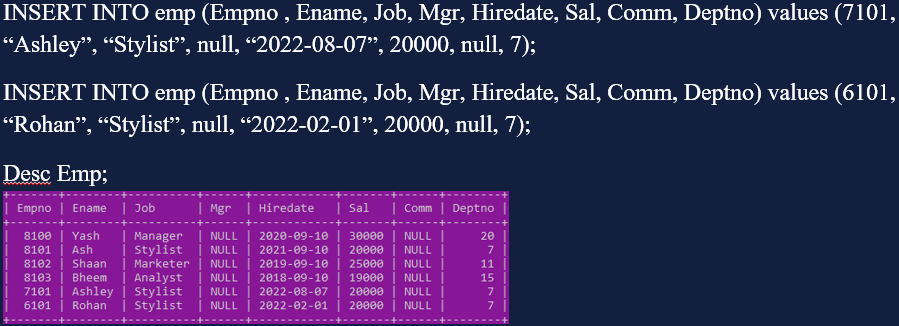
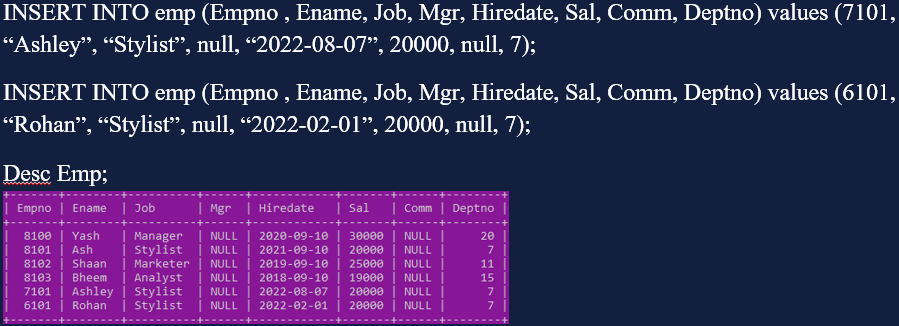
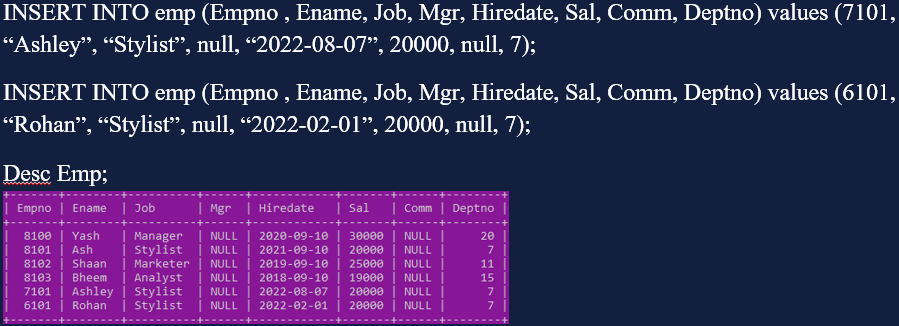
1. Display details of Emp where Sal is null and is not null respectively. 
2. Display details of Emp where

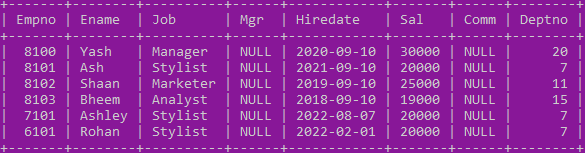
Job is Manager & Analyst

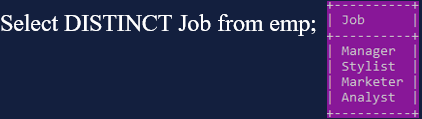
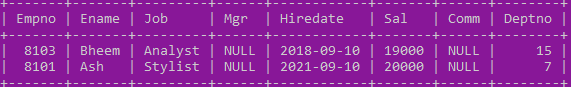
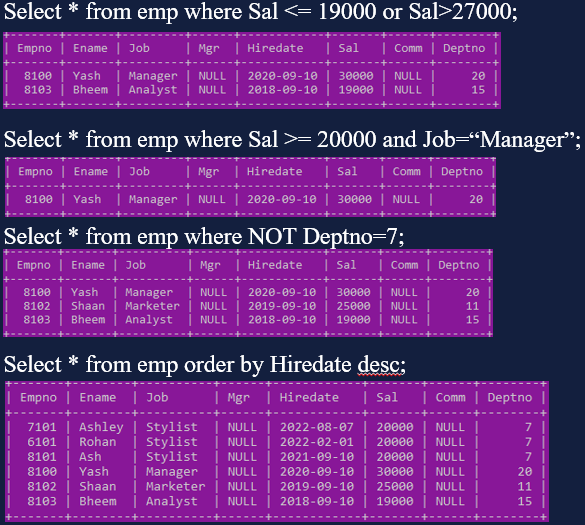
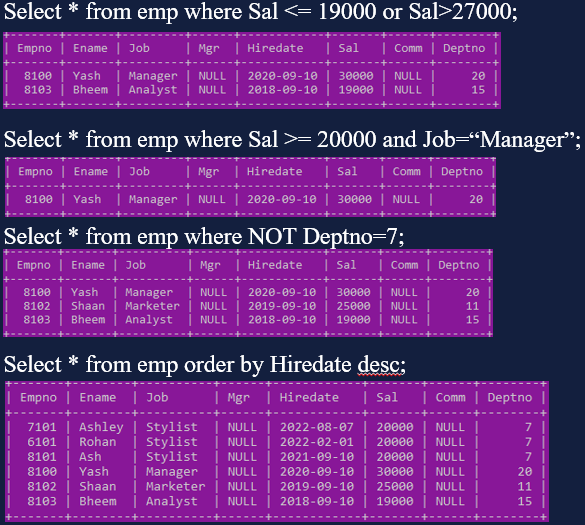
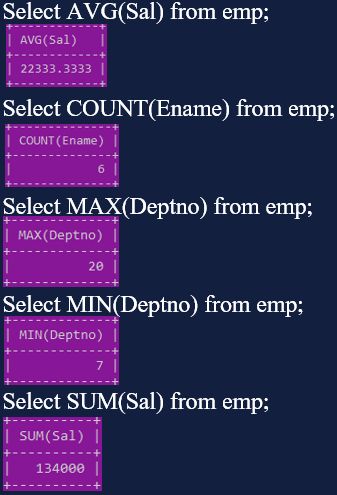
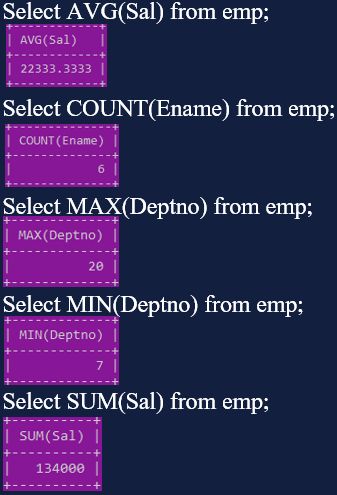
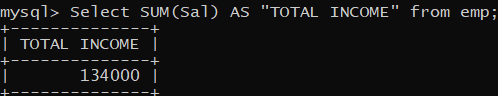
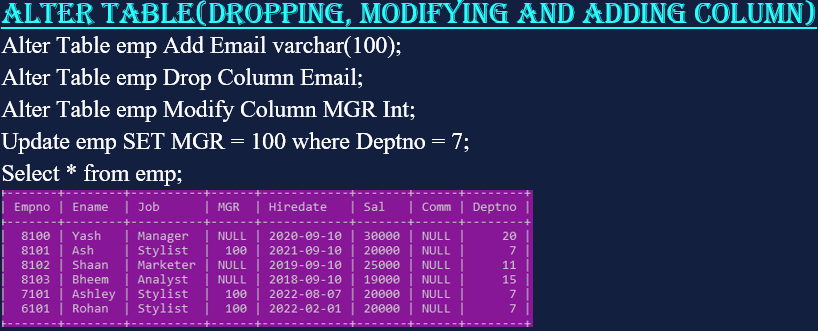
Job name has letter l in it

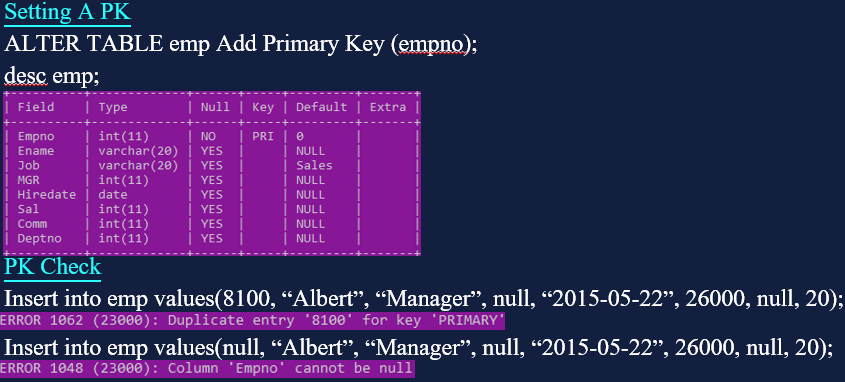
respectively.

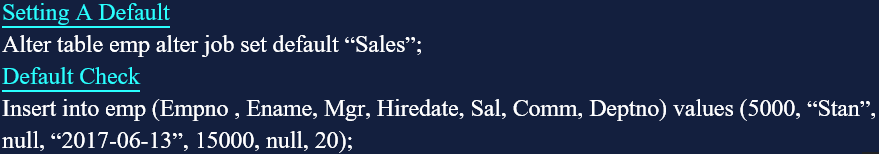
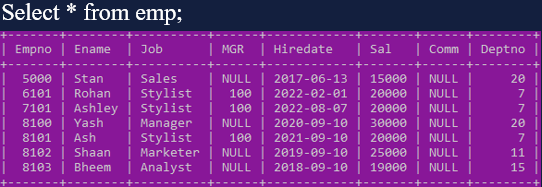


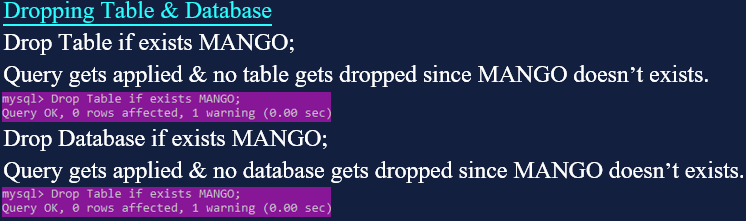


1. Display distinct jobs. 
2. Display details of Emp where Job is Stylist & Analyst grouped by salary. 
3. Try out queries based on operators 
4. Try out queries based on aggregate functions.  
5. Find out the total salary of all the employees and show it as TOTAL INCOME. 
6. Try out queries based on altering table. 
7. Set Empno as Primary Key and show that it has been completed successfully.



1. Set Job’s default value to Sales & show that it has been completed successfully.  

Q.23 Write Queries to drop certain table and database respectively.



#24: Stack Menu Driven Program

def isEmpty(s): #s is stack

if s==[]:

return True

else:

return False

def push(s,item):

s.append(item) #append means to add smthing at the end of txt files

top = len(s)-1

def Pop(s):

if isEmpty(s):

return "Underflow"

else:

item=s.pop()

if len(s) ==0:

top = None

else:

top = len(s)-1

return item

def Peek(s):

if isEmpty(s):

return "Underflow"

else:

top = len(s)-1

return s[top]

def Display(s):

global top

if len(s) ==0:

return "Stack Empty!"

top = None

else:

t=top

while t>=0:

print(s[t],end='')

t=t-1

#\_\_\_\_\_\_\_\_\_\_\_MENU\_\_\_\_\_\_\_\_\_\_\_

s=[] #initially stack is empty

top= None

while True:

print("Stack Operating")

print("1.Push")

print("2.Pop")

print("3.Peek")

print("4.Display")

print("5.Exit")

ch=int(input("Enter your choice:"))

if ch==1:

item = int(input("Enter item:"))

push(s, item)

elif ch==2:

item=Pop(s)

print(item)

elif ch==3:

Peek(s)

elif ch==4:

Display(s)

elif ch==5:

break

else:

print("Invalid choice!")

INPUT & OUTPUT:

