**Theory:**

## Interoperation Parallelism:

It is about executing different operations of a query in parallel. A single query may involve multiple operations at once. We may exploit parallelism to achieve better performance of such queries. Consider the example query given below;

SELECT AVG(Salary) FROM Employee GROUP BY Dept\_Id;

It involves two operations. First one is an Aggregation and the second is grouping. For executing this query,

We need to group all the employee records based on the attribute Dept\_Id first.

Then, for every group we can apply the AVG aggregate function to get the final result.

We can use Interoperation parallelism concept to parallelize these two operations.

[Note: Intra-operation is about executing single operation of a query using multiple processors in parallel]

The following are the variants using which we would achieve Interoperation Parallelism;

1. Pipelined Parallelism

2. Independent Parallelism

**Program:**

import numpy as np

import mysql.connector

import mysql.connector as connector

#connection

db = mysql.connector.connect(

         host="localhost",

         user="root",

         passwd="root",

        )

mycursor = db.cursor()

def create\_db():

    global db

    mycursor.execute("CREATE DATABASE db8")

    db.commit()

    db = mysql.connector.connect(

     host="localhost",

     user="root",

     passwd="root",

     database="db8"

    )

import mysql.connector as connector

def connection():

    config = {

        "user": "root",

        "password": "root",

        "host": "localhost",

        "port": 3306,

        "database": "db8"

    }

    try:

        c = connector.connect(\*\*config)

        return c

    except:

        print ("connection error")

        exit(1)

def connect():

    global db,mycursor

    db = mysql.connector.connect(

     host="localhost",

     user="root",

     passwd="root",

     database="db8"

    )

    mycursor=db.cursor()

def reset\_db():

    mycursor.execute('DROP DATABASE db8;')

    create\_db()

    create\_attribute()

    add\_values()

def create\_attribute():

    create\_Students = 'CREATE TABLE `db8`.`students` ( `Roll\_no` INT(10) NOT NULL   , `Name` VARCHAR(30) NOT NULL , `Address` VARCHAR(20) NULL , `Marks` INT(10) NULL , `Donations` INT(10) NULL , PRIMARY KEY (`Roll\_no`))ENGINE = InnoDB ;'

    create\_Class = 'CREATE TABLE `db8`.`class` ( `Division` INT(10) NOT NULL   , `Min\_marks` INT(10) NOT NULL , `Max\_marks` INT(10) NOT NULL,`Donations` INT(10) NULL , PRIMARY KEY (`Division`))ENGINE = InnoDB ;'

    create\_customer = 'CREATE TABLE `db8`.`customer` ( `C\_id` INT(10) NOT NULL   , `Name` VARCHAR(30) NOT NULL , `Address` VARCHAR(20) NULL , `Products` INT(10) NULL , `Price` INT(10) NULL , PRIMARY KEY (`C\_id`))ENGINE = InnoDB ;'

    create\_orders = 'CREATE TABLE `db8`.`orders` ( `O\_id` INT(10) NOT NULL   , `Min\_orders` INT(10) NOT NULL , `Max\_orders` INT(10) NOT NULL,`C\_id` INT(10) NULL , PRIMARY KEY (`O\_id`))ENGINE = InnoDB ;'

    sql = [create\_Students,create\_Class,create\_customer,create\_orders]

    mycursor = db.cursor()

    for i in sql:

        mycursor.execute(i)

    db.commit()

def add\_values():

    mycursor = db.cursor()

    sql = "INSERT INTO students (Roll\_no,Name,Address,Marks,Donations) VALUES (%s, %s, %s ,%s,%s);"

    val = [('1','Aupmanyu','Nagpur','95','845'),('2','Rahul','Banglore','98','2123'),('3','Samay','Pune','87','454'),('4','Urooj','Delhi','81','980'),('5','Prashasthi','Pune','86','727'),('6','Zakir','Delhi','98','1006'),('7','Munawar','Junagad','94','3023'),('8','Sumukhi','Mumbai','88','122')]

    mycursor.executemany(sql, val)

    print(mycursor.rowcount, "values were inserted.")

    sql = "INSERT INTO class (Division,Min\_marks,Max\_marks,Donations) VALUES (%s, %s, %s,%s );"

    val = [('1','96','100','1000'),('2','91','95','750'),('3','86','90','500'),('4','81','85','250')]

    mycursor.executemany(sql, val)

    print(mycursor.rowcount, "values were inserted.")

    db.commit()

    mycursor = db.cursor()

    sql = "INSERT INTO customer (C\_id,Name,Address,Products,Price) VALUES (%s, %s, %s ,%s,%s);"

    val = [('1','Aupmanyu','Nagpur','34','845'),('2','Rahul','Banglore','13','2123'),('3','Samay','Pune','65','454'),('4','Urooj','Delhi','563','980'),('5','Prashasthi','Pune','23','727'),('6','Zakir','Delhi','54','1006'),('7','Munawar','Junagad','6','3023'),('8','Sumukhi','Mumbai','34','122')]

    mycursor.executemany(sql, val)

    print(mycursor.rowcount, "values were inserted.")

    sql = "INSERT INTO orders (O\_id,Min\_orders,Max\_orders,C\_id) VALUES (%s, %s, %s,%s );"

    val = [('1','1','25','4'),('2','25','50','2'),('3','50','100','5'),('4','100','500','6'),('5','500','1000','3')]

    mycursor.executemany(sql, val)

    print(mycursor.rowcount, "values were inserted.")

    db.commit()

try:

    connect()

except:

    create\_db()

    create\_attribute()

connect()

reset\_db()

import threading

connect()

#reset\_db()

from tkinter import \*

top = Tk()

top.geometry("400x70")

def fun():#show database

    connect()

    global db,mycursor

    db = mysql.connector.connect(

     host="localhost",

     user="root",

     passwd="root",

     database="db8"

    )

    mycur=db.cursor()

    mycur.execute('SHOW TABLES')

    tables=[]

    for i in mycur:

        tables.append(i[0])

    for table\_names in tables:

        top2=Tk()

        top2.title(table\_names)

        connect()

        mycursor.execute("SELECT \* FROM "+table\_names+' LIMIT 0,25')

        i=0

        for book in mycursor:

            for j in range(len(book)):

                e = Entry(top2, width=20, fg='blue')

                e.grid(row=i, column=j)

                e.insert(END, book[j])

            i=i+1

    top2.mainloop()

b = Button(top,text = "Show Database",command=fun)

which\_join = 0

def join():

    global which\_join

    which\_join+=1

    def main\_join1(st\_table,cl\_table):

        db = mysql.connector.connect(

         host="localhost",

         user="root",

         passwd="root",

         database="db8"

        )

        result = []

        mycursor=db.cursor()

        mycursor.execute('CREATE TABLE sNc (`Roll\_no` INT(10) NOT NULL   , `Name` VARCHAR(30) NOT NULL , `Address` VARCHAR(20) NULL , `Marks` INT(10) NULL , `Donation` INT(10) NULL ,`Division` INT(10) NOT NULL   , `Min\_marks` INT(10) NOT NULL , `Max\_marks` INT(10) NOT NULL,`Donations` INT(10) NULL);')

        mycursor.execute('INSERT INTO sNc SELECT \* FROM students JOIN class ON students.Marks BETWEEN class.Min\_marks AND class.Max\_marks;')

        db.commit()

    def main\_join2(st\_table,cl\_table):

        db = mysql.connector.connect(

         host="localhost",

         user="root",

         passwd="root",

         database="db8"

        )

        result = []

        mycursor=db.cursor()

        mycursor.execute('CREATE TABLE cNo (`C\_id1` INT(10) NOT NULL   , `Name` VARCHAR(30) NOT NULL , `Address` VARCHAR(20) NULL , `Products` INT(10) NULL , `Price` INT(10) NULL , `O\_id` INT(10) NOT NULL   , `Min\_orders` INT(10) NOT NULL , `Max\_orders` INT(10) NOT NULL,`C\_id` INT(10) NULL );')

        mycursor.execute('INSERT INTO cNo SELECT \* FROM customer JOIN orders ON customer.C\_id = orders.C\_id')

        db.commit()

    top2=1

    def main\_join3():

        global top2

        db = mysql.connector.connect(

         host="localhost",

         user="root",

         passwd="root",

         database="db8"

        )

        result = []

        mycursor=db.cursor()

        mycursor.execute('SELECT \* FROM cNo JOIN sNc ON cNo.Name = snc.Name')

        temp=mycursor.fetchall()

        for i in temp:

            result.append(i)

        top2 = Tk()

        i=0

        for stud in result:

            for j in range(len(stud)):

                e = Entry(top2, width=20, fg='blue')

                e.grid(row=i, column=j)

                e.insert(END, stud[j])

            i=i+1

    tables = [['class','students'],['customer','orders']]

    def f(tb1,tb2):

        t1 = threading.Thread(target=main\_join1, args=(tb1[0],tb1[1]))

        t1.start()

        t2 = threading.Thread(target=main\_join2, args=(tb2[0],tb2[1]))

        t2.start()

        t1.join()

        t2.join()

    f(tables[1],tables[0])

    main\_join3()

    try:

        top2.mainloop()

    except:

        pass

j = Button(top,text = "Join",command=join)

b.pack()

j.pack()

top.mainloop()

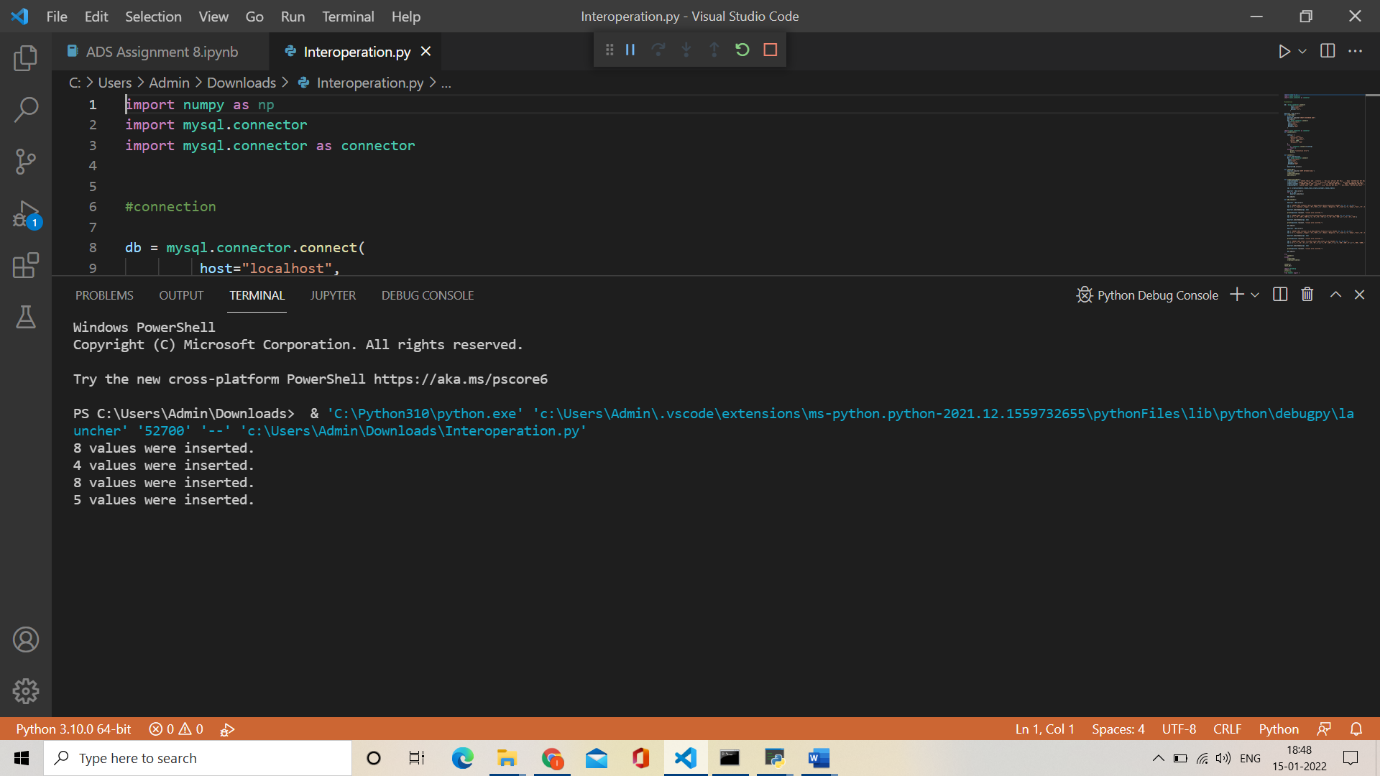
#db.commit()

db.close()

db.is\_connected()

db.close()

**Screenshots:**

****

