WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

Theory :

Assumptions:

Assume n processors, P0, P1, …, Pn-1 and n disks D0, D1, …, Dn-1.

Disk Di is associated with Processor Pi.

Relation R is partitioned into R0, R1, …, Rn-1 using Round-robin technique or Hash Partitioning technique or Range Partitioning technique (if range partitioned on some other attribute other than sorting attribute)

Objective:

Our objective is to sort a relation (table) Ri that resides on n disks on an attribute A in parallel.

Steps:

Step 1: Partition the relations Ri on the sorting attribute A at every processor using a range vector v. Send the partitioned records which fall in the ith range to Processor Pi where they are temporarily stored in Di.

Step 2: Sort each partition locally at each processor Pi. And, send the sorted results for merging with all the other sorted results which is a trivial process.

Program Code :

```
import mysql.connector
import threading


def connector():
```

WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
    conn =
mysql.connector.connect(host='localhost',user='root',password='root',database=
'student')
    cursor = conn.cursor()
    return conn, cursor


def part1():
    conn, cursor = connector()
    sql = "INSERT INTO stud2 partition(u1)  (SELECT * FROM stud_temp
partition(k1) order by marks)"
    cursor.execute(sql)
    cursor.execute("SELECT * FROM stud2 partition(u1)")
    result = cursor.fetchall()
    print("\n")
    print("After sorting by marks in partition 1 ")
    for row in result:
        print(row)
    conn.commit()
    conn.close()


def part2():
    conn, cursor = connector()
    sql = "INSERT INTO stud2 partition(u2)  (SELECT * FROM stud_temp
partition(k2) order by marks)"
    cursor.execute(sql)
    cursor.execute("SELECT * FROM stud2 partition(u2)")
    result = cursor.fetchall()
    print("\n")
    print("After sorting by marks in partition 2")
    for row in result:
        print(row)
    conn.commit()
    conn.close()

def part3():
    conn, cursor = connector()
    sql = "INSERT INTO stud2 partition(u3)  (SELECT * FROM stud_temp
partition(k3) order by marks)"
    cursor.execute(sql)
```

WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
    cursor.execute("SELECT * FROM stud2 partition(u3)")
    result = cursor.fetchall()
    print("\n")
    print("After sorting by marks in partition 3")
    for row in result:
        print(row)
    conn.commit()
    conn.close()
def part4():
    conn, cursor = connector()
    sql = "INSERT INTO stud2 partition(u4)  (SELECT * FROM stud_temp
partition(k4) order by marks)"
    cursor.execute(sql)
    cursor.execute("SELECT * FROM stud2 partition(u4)")
    result = cursor.fetchall()
    print("\n")
    print("After sorting by marks in partition 4")
    for row in result:
        print(row)
    conn.commit()
    conn.close()



#establishing the connection
conn, cursor = connector()


#creating stud2 table
cursor.execute("DROP TABLE IF EXISTS stud2")
sql ='''CREATE TABLE stud2(
   rollno int ,
   name varchar(25),
   marks int
)'''
cursor.execute(sql)
sql = "alter table stud2 partition by range (marks) (PARTITION u1 VALUES LESS
THAN (20), partition u2 values less than (30) , partition u3 values less than
(60), partition u4 values less than (100))"
cursor.execute(sql)
```

WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
cursor.execute("DROP TABLE IF EXISTS student")

sql ="""CREATE TABLE STUDENT(
   rollno int ,
   name varchar(25),
   marks int
)"""
cursor.execute(sql)
#creating employee_temp table
cursor.execute("DROP TABLE IF EXISTS stud_temp")
sql ="""CREATE TABLE stud_temp(
    rollno int ,
    name varchar(25),
    marks int
    )"""
cursor.execute(sql)


sql = "INSERT INTO student (rollno, name , marks) VALUES (%s, %s , %s)"
val = [
  (0,   'student0', 78),
(1, 'student1', 54),
(2, 'student2', 35),
(3, 'student3', 98),
(4, 'student4', 65),
(5, 'student5', 85),
(6, 'student6', 42),
(7, 'student7', 36),
(8, 'student8', 14),
(9, 'student9', 54),
(10, 'student10',   21),
(11,'student11',    78),
(12,    'student12',    36),
(13,    'student13', 92),
(14,    'student14',    15),
(15,    'student15',    54),
(16,    'student16',    56),
(17,    'student17',    65),
(18,    'student18',    35),
(19,    'student19',    53),
(20,    'student20',    24),
```

WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
]
cursor.executemany(sql, val)


cursor.execute("SELECT * FROM student")

# fetch all the matching rows
result = cursor.fetchall()
print("Before Partitioning")

# loop through the rows
for row in result:
    print(row)
print("\n")
print("After partition using rollno")
sql = "alter table STUDENT partition by range columns (rollno) (PARTITION p1
VALUES LESS THAN (5), partition p2 values less than (10) , partition p3 values
less than (21))"
cursor.execute(sql)
print("\n")
print(("Rows in partition 1 : "))
cursor.execute("SELECT * FROM student partition(p1)")
result = cursor.fetchall()
for row in result:
    print(row)
print("\n")
print(("Rows in partition 2 : "))
cursor.execute("SELECT * FROM student partition(p2)")
result = cursor.fetchall()
for row in result:
    print(row)
print("\n")
print(("Rows in partition 3 : "))
cursor.execute("SELECT * FROM student partition(p3)")
result = cursor.fetchall()
for row in result:
    print(row)
print("\n")
sql = "INSERT INTO stud_temp  (SELECT * FROM student)"
cursor.execute(sql)
print("After partitioning by marks")
```

WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
sql = "alter table stud_temp partition by range (marks) (PARTITION k1 VALUES
LESS THAN (20), partition k2 values less than (30) , partition k3 values less
than (60), partition k4 values less than (80) , partition k5 values less than
(100))"
cursor.execute(sql)
print(("Rows in partition 1 : "))
cursor.execute("SELECT * FROM stud_temp partition(k1)")
result = cursor.fetchall()
for row in result:
    print(row)
print("\n")
print(("Rows in partition 2 : "))
cursor.execute("SELECT * FROM stud_temp partition(k2)")
result = cursor.fetchall()
for row in result:
    print(row)
print("\n")
print(("Rows in partition 3 : "))
cursor.execute("SELECT * FROM stud_temp partition(k3)")
result = cursor.fetchall()
for row in result:
    print(row)
print("\n")
print(("Rows in partition 4 : "))
cursor.execute("SELECT * FROM stud_temp partition(k4)")
result = cursor.fetchall()
for row in result:
    print(row)
print("\n")
conn.close()
t1 = threading.Thread(target=part1)
t2 = threading.Thread(target=part2)
t3 = threading.Thread(target=part3)
t4 = threading.Thread(target=part4)
# starting thread 1
t1.start()
# starting thread 2
t2.start()
# starting thread 3
t3.start()
t4.start()
```

WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I
Advanced Database System

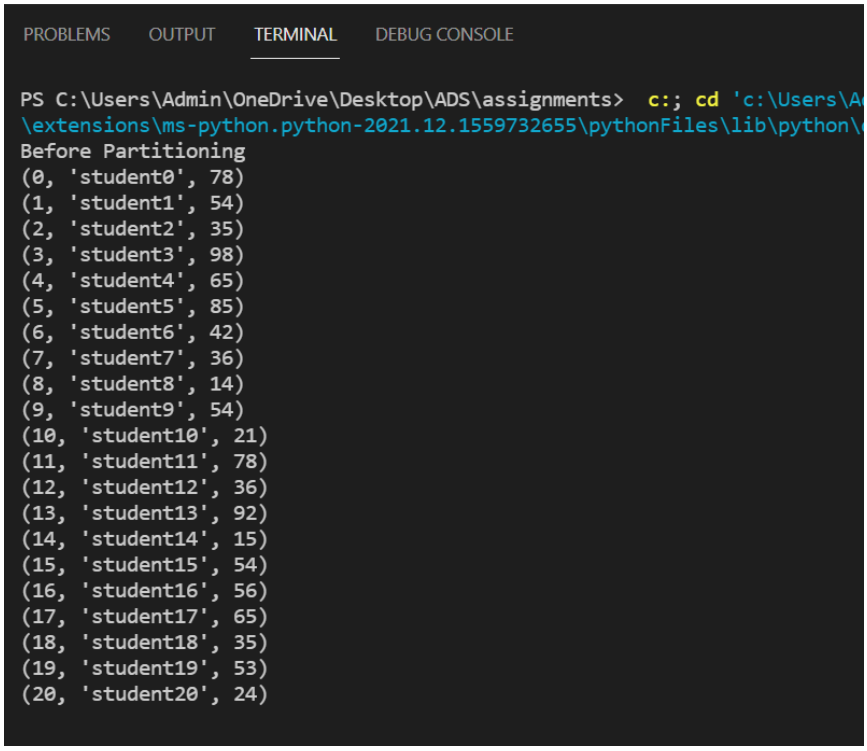
Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
t1.join()
t2.join()
t3.join()
t4.join()
print("\n")
print("After merging")
conn,cursor =connector()
cursor.execute("SELECT * FROM stud2")
result = cursor.fetchall()
for row in result:
    print(row)
conn.close()
```

Screenshots:



WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
After partition using rollno


Rows in partition 1 :
(0, 'student0', 78)
(1, 'student1', 54)
(2, 'student2', 35)
(3, 'student3', 98)
(4, 'student4', 65)


Rows in partition 2 :
(5, 'student5', 85)
(6, 'student6', 42)
(7, 'student7', 36)
(8, 'student8', 14)
(9, 'student9', 54)


Rows in partition 3 :
(10, 'student10', 21)
(11, 'student11', 78)
(12, 'student12', 36)
(13, 'student13', 92)
(14, 'student14', 15)
(15, 'student15', 54)
(16, 'student16', 56)
(17, 'student17', 65)
(18, 'student18', 35)
(19, 'student19', 53)
(20, 'student20', 24)
```

```
After partitioning by marks
Rows in partition 1 :
(8, 'student8', 14)
(14, 'student14', 15)


Rows in partition 2 :
(10, 'student10', 21)
(20, 'student20', 24)


Rows in partition 3 :
(1, 'student1', 54)
(2, 'student2', 35)
(6, 'student6', 42)
(7, 'student7', 36)
(9, 'student9', 54)
(12, 'student12', 36)
(15, 'student15', 54)
(16, 'student16', 56)
(18, 'student18', 35)
(19, 'student19', 53)


Rows in partition 4 :
(0, 'student0', 78)
(4, 'student4', 65)
(11, 'student11', 78)
(17, 'student17', 65)
```

WALCHAND INSTITUTE OF TECHNOLOGY, SOLAPUR
INFORMATION TECHNOLOGY
2021-22 SEMESTER –I

Advanced Database System

Name: Alaikya S Yemul Roll No: 62

ASSIGNMENT NO: 6

Title: Implementation of intraquery parallelism using multithreading

```
After sorting by marks in partition 2

(10, 'student10', 21)
After sorting by marks in partition 1

(8, 'student8', 14)
(14, 'student14', 15)
(20, 'student20', 24)


After sorting by marks in partition 3
(2, 'student2', 35)
(18, 'student18', 35)


After sorting by marks in partition 4
(4, 'student4', 65)
(17, 'student17', 65)
(0, 'student0', 78)(7, 'student7', 36)

(11, 'student11', 78)(12, 'student12', 36)

(6, 'student6', 42)
(19, 'student19', 53)
(1, 'student1', 54)
(9, 'student9', 54)
(15, 'student15', 54)
(16, 'student16', 56)
```

```
After merging
(8, 'student8', 14)
(14, 'student14', 15)
(10, 'student10', 21)
(20, 'student20', 24)
(2, 'student2', 35)
(18, 'student18', 35)
(7, 'student7', 36)
(12, 'student12', 36)
(6, 'student6', 42)
(19, 'student19', 53)
(1, 'student1', 54)
(9, 'student9', 54)
(15, 'student15', 54)
(16, 'student16', 56)
(4, 'student4', 65)
(17, 'student17', 65)
(0, 'student0', 78)
(11, 'student11', 78)
```