SL- VI
Expt. 4

Aim: Implement any one Partitioning technique in Parallel Databases

Steps:
First install and configure Cassandra from,
https://sl6it.wordpress.com/2015/12/10/4-installation-of-nosql-database-cassandra/

start cassandra daemon
~/cassandra/bin/cassandra -f

The cassandra daemon should start in the foreground
(don't press ctrl + c; as it'll terminate the daemon)

Open eclipse->new java project->project name **Partitioning**

Create new class under your project -> **Partitioning**-> add following code

```
import java.util.Scanner;
import java.util.Vector;
import com.datastax.driver.core.Cluster;
import com.datastax.driver.core.ResultSet;
import com.datastax.driver.core.Row;
import com.datastax.driver.core.Session;

public class Partitioning {
   static Cluster cluster;
        static Session session;
        static ResultSet results;
        static Row rows;
        public static void main(String[] args)
        {
                //start the cassandra server in one terminal
                try
                {       cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
                        session = cluster.connect();

                        session.execute("CREATE KEYSPACE IF NOT EXISTS keyspace1 WITH
replication " + "= {'class':'SimpleStrategy','replication_factor':1}; ");
                        session.execute("USE keyspace1");

                        //create users table
                  session.execute("CREATE TABLE IF NOT EXISTS users (lastname text, age int,
city text, email text PRIMARY KEY, firstname text);");

                  // Insert records into the users table
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Jones', 35, 'Austin', 'bob@example.com', 'Bob')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Swift', 25, 'Paris', 'mark@example.com', 'Mark')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
```

```
VALUES ('Castle', 35, 'New-York', 'kate@example.com', 'Kate')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Jordon', 45, 'Mumbai', 'mike@example.com', 'Mike')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Hanks', 48, 'Seoul', 'tom@example.com', 'Tom')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Depp', 55, 'HongKong', 'johnny@example.com', 'Johnny')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Nicholson', 65, 'Rio de Janeiro', 'jack@example.com', 'Jack')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Freeman', 60, 'St Petersburg', 'morgan@example.com', 'Morgan')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Lewis', 50, 'Alexandria', 'daniel@example.com', 'Daniel')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Brando', 85, 'Ahmedabad', 'marlon@example.com', 'Marlon')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('DeNiro', 80, 'Toronto', 'robert@example.com', 'Robert')");
                        session.execute("INSERT INTO users (lastname, age, city, email, firstname)
VALUES ('Bale', 50, 'Sydney', 'chris@example.com', 'Christian')");

                        //do partitioning

                    results = session.execute("SELECT * FROM users ");
                        int i=0,disk_no=0, total_disks=4;
                        System.out.println("\nRound Robin Partitioning");
                    for (Row row : results)
                        {
                     /* Implement Round-Robin partitioning
Let, 'total_disks' = 4
disk_no = row_number % total_disks
*/
                       disk_no=i++%total_disks;

                            System.out.format("Disk: %d \t %s %d %s %s %s \n",
disk_no,row.getString("lastname"),
row.getInt("age"),row.getString("city"),row.getString("email"),row.getString("firstname"));
                        }

                    int age;
                    results = session.execute("SELECT * FROM users ");
                    System.out.println("\nHash Partitioning");
                    for (Row row : results)
                    {
                        /* Implement Hash partitioning
Let, 'total_disks' = n = 4
suppose 'age' is the partitioning attribute
disk_no = (Partitioning attribute value) % total_disks
*/

                        age=row.getInt("age");
                        disk_no=age%total_disks;
                        System.out.format("Disk: %d \t %s %d %s %s %s \n",
```

```
disk_no,row.getString("lastname"),
row.getInt("age"),row.getString("city"),row.getString("email"),row.getString("firstname"));

                                }

                        int v,n;
                        results = session.execute("SELECT * FROM users ");

                      /* Implement Range Partitioning
suppose 'age' is the partitioning attribute
Let, 'total_disks' = n = 4
Let, partitioning vector : [vo, v1, ..., vn-2] = [v0, v1, v2] = [30,50,70]
Let, v be the partitioning attribute value of a tuple
Let, 0 <= i < n-2
Rule:
Tuples such that vi <= v < vi+1 go to disk i + 1. Tuples with v < v0 go to disk 0 and tuples
with v >= vn-2 go to disk n-1.
*/
                         n= total_disks;
                              int[] p_vector= new int[n-1];
                              int flag;
                              Scanner sc=new Scanner(System.in);
                              System.out.println("\nRange Partitioning");
                              System.out.println("\nPlease enter Partitioning vector values:-");
                              for(i=0;i<=n-2;i++)
                              {
                                System.out.println("Enter v"+i+" : ");
                                    p_vector[i]=Integer.parseInt(sc.next());
                              }

                             System.out.println("\nRange Partitioning - output \n");
                        for (Row row : results)
                        {

                              v=row.getInt("age");
                              flag=0;

                              for(i=0;i<n-2;i++)
                              {
                                    if(p_vector[i]<= v && v < p_vector[i+1])
                                    {
                                          disk_no=i+1;
                                          flag=1;
                                          break;
                                    }
                              }

                              if(flag==0)
                              {
                                    if( v < p_vector[0])
                                    {
                                          disk_no=0;      }
```

```
                                    else if (v>= p_vector[n-2])
                                    {
                                            disk_no=n-1;
                                    }
                                }

                               System.out.format("Disk: %d \t %s %d %s %s %s \n",
disk_no,row.getString("lastname"),
row.getInt("age"),row.getString("city"),row.getString("email"),row.getString("firstname"));

                               }

                        sc.close();

                        }
                        catch(Exception e)
                        {
                          System.out.println("Error: "+e.getMessage());
                        }

                    cluster.close();
                }

}
```

save the file

It will display some errors, so we are going to import some jar files in the project.

Please download the cassandra-java driver from
http://downloads.datastax.com/java-driver/cassandra-java-driver-2.0.2.tar.gz

Extract it
Paste all jar files (total 12)from the extracted folder into your eclipse project
(10 jar files are present under lib folder)

One by one, right click on pasted jarfile (in eclipse)-> Build path -> add to build path
Add all pasted jar files to build path

Run the project

Reference:

AviSilberschatz , Henry F. Korth , S. Sudarshan, “Database System Concepts, Sixth Edition”, ISBN-13: 978-93-3290-138-4, MCGraw Hill

--

Prof. Sachin T. Kolhe
(IT DEPT, SRES COE Kopargaon)