Kijit Desai
NJIT ID: 31246712

Implementation Project 1

Project Description/Objectives: Project Implements Apriori Algorithm on 5 different datasets. The algorithm takes user input for the minimum support and confidence for generating Rules.

Software used:
IDE: Netbeans.
DBMS: MySql Workbench.
Language: JAVA.

Note.: the following code is able to implement the Algorithm just partially. It can generate rules and frequent itemsets till 2-itemset only. The remaining itemset calculation is showing error and I was unable to debug. Yet, I have showed the rules and itemsets that I am able to generate.

Source Code:

```
package dbconnection;


import java.sql.*;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.Set;
import javax.sql.*;

/**
 *
 * @author kijit
 */
public class DBConnection {

    /**
     * @param args the command line arguments
     *
     */



    public static void main(String[] args) {
        // TODO code application logic here


    ArrayList list1=new ArrayList();
    ArrayList list2=new ArrayList() ;
    int[] a = new int[100];
    int[] b = new int[100];

   String query = "Select * FROM PurchasedItem";

        String userName = "root";
        String password = "";
        String url = "jdbc:mysql://localhost/dbtest";
        Connection con;



    try {

            Class.forName ("com.mysql.jdbc.Driver");
            con  = DriverManager.getConnection (url, userName, password);


        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery(query);
        int i=0;

            while (rs.next()) {
              String dbtime = rs.getString(1);
              String dbtime2 = rs.getString(2);
              list1.add(dbtime);
              list2.add(dbtime2);
              a[i]= rs.getInt("Itemid");
              b[i]= rs.getInt("Pid");
              //System.out.println(a[i]);
              i++ ;

             // System.out.println(dbtime);
            } //end while

    con.close();
    }catch(Exception e) {
        e.printStackTrace();
    }
        //System.out.println(list1);
        //System.out.println(list2);

ExecuteApriori3.ExecApriori3(list1,list2);

        //ExecuteApriori2 test= new ExecuteApriori2();
        //test.ExecApriori2(a,b,list1);

    }
}
```

Kijit Desai
NJIT ID: 31246712

```
/* Aprioiri Execution function*/
/* also calculates candidate itemsets and find frequent itemsets from Class
frequent*/
package dbconnection;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.StringTokenizer;
import java.util.Scanner;

/**
 *
 * @author kijit
 */
public class ExecuteApriori3 {


  public static void ExecApriori3(ArrayList i1,ArrayList p1)
  {
    //System.out.println(i1);
    //System.out.println(p1);

    Object items[]=i1.toArray();
    Object pid[]=i1.toArray();

    Scanner scan= new Scanner(System.in);

    int count[]={0,0,0,0,0};
    int temp=0;

    int itemsetnumber=0;

    ArrayList candidates= new ArrayList() ;
    ArrayList candid = new ArrayList();

    System.out.println("Enter Number of transactions");
    double transacno=scan.nextDouble();
    System.out.println("Enter minimum support %");
    double supmin= scan.nextDouble();
    supmin=supmin/100;
    supmin=supmin*transacno;
    //System.out.println(supmin);
    //int minsup=2 ;




    for(int i=0;i<items.length;i++)
    {
```

```
   //temp += (Integer.parseInt( items[i]));
   //supcount[temp] ++;
   //System.out.println(items[i]);
}
   itemsetnumber++ ;
   ArrayList tempcand = new ArrayList();
   String str1,str2;
   StringTokenizer st1,st2 ;
   if(itemsetnumber==1)
   {

      for(int i=1;i<=6;i++)
      {
         tempcand.add(Integer.toString(i));
         //System.out.println(tempcand);


      }
      Object tempcand2[]=tempcand.toArray();
      for (int i=0;i<tempcand.size();i++)
      {
         double countsup=0;
         temp=0;
         for(int k=0;k<items.length;k++)
         {
            //System.out.println(items[k]);
            //System.out.println(tempcand.get(i));

            if(tempcand.get(i).equals(i1.get(k)))
            {

               countsup++;

            }


         }
         if(countsup>=supmin)
         {
         candid.add(tempcand.get(i));
         }

         //boolean add = candid.add(tempcand.get(i));
      }
      tempcand.clear();

      System.out.println(" 1-temset = "+candid);
```

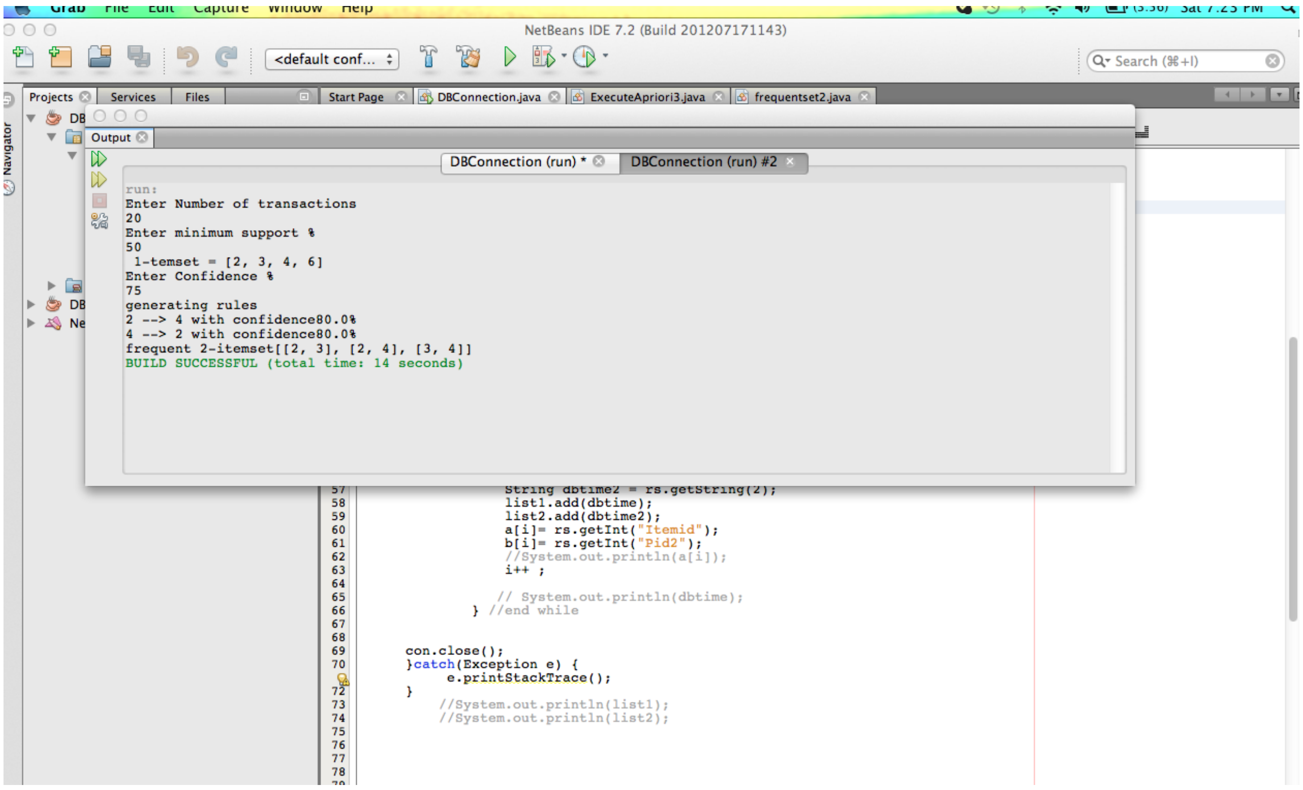
```
  itemsetnumber++;

}
   while(candid.size()!=0)
  {



ArrayList temp2= new ArrayList();

if(itemsetnumber==2)
{

   for(int i=0;i<candid.size();i++)
   {

      for (int j=i+1;j<candid.size();j++)
      {
         ArrayList child= new ArrayList();
         child.add(candid.get(i));
         child.add(candid.get(j));
         temp2.add(child);

      }
   }


}
 //System.out.println(temp2);

frequentset2 itemset2= new frequentset2();
itemset2.frequentset2(i1, p1, temp2,supmin);


/*ArrayList temp3= new ArrayList();

for(int i=0;i<temp2.size();i++)
   {

      for (int j=i+1;j<temp2.size();j++)
      {
         ArrayList child= new ArrayList();
         child.add(temp2.get(i));
         child.add(temp2.get(j));
         temp3.add(child);

      }
   }
```

```
        */

        //System.out.println(temp3);
        candidates.clear();
        candidates=new ArrayList<String>(tempcand);
        tempcand.clear();


        itemsetnumber++;





      //System.out.println(temp);
    }


    /*public static int[] convertIntegers(ArrayList<Integer> integers)
{
    int[] ret = new int[integers.size()];
    for (int i=0; i < ret.length; i++)
    {
        ret[i] = integers.get(i).intValue();
    }
    return ret;
}*/



}


/* Frequent sets calculation class*/
package dbconnection;

import java.util.ArrayList;
import java.util.Scanner;
import java.util.StringTokenizer;

/**
 *
 * @author kijit
 */
public class frequentset2 {
   public void frequentset2(ArrayList i1,ArrayList pid,ArrayList candid,double
supmin)
   {
      Scanner scan= new Scanner(System.in);
      System.out.println("Enter Confidence %");
      double confidence= scan.nextDouble() ;
      confidence=confidence/100;
      //confidence=confidence*20;
      //System.out.println(confidence);
      ArrayList freq = new ArrayList();

      Object candid2[]= i1.toArray();
      Object new2[]= candid.toArray();
      System.out.println("generating rules");
      for(int i=0;i<candid.size();i++)
      {
         ArrayList purchase1 = new ArrayList();
         ArrayList purchase2 = new ArrayList();
         ArrayList comp = new ArrayList();
         comp.add(((ArrayList)candid.get(i)).get(0));
         comp.add(((ArrayList)candid.get(i)).get(1));
         //System.out.println(comp);
         for(int j=0;j<2;j++)
         {
            for(int k=0;k<i1.size();k++)
            {
               if(comp.get(j).equals(i1.get(k)))
               {
                  if(j==0)
                  {
                     purchase1.add(pid.get(k));

                  }
                  else{
                     purchase2.add(pid.get(k));
                  }
               }
            }




         }
           // System.out.println(purchase1);
            //System.out.println(purchase2);

         double count=0;
         for(int a=0;a<purchase1.size();a++)
```

```
        {
          for(int b=0;b<purchase2.size();b++)
          {
            if(purchase1.get(a).equals(purchase2.get(b)))
            {
              count++;
              //System.out.println(count);

            }
          }
        }

        if(count>=supmin)
        {
          double r1=0;double r2=0;
          r1=count/(purchase1.size());
          //System.out.println(r1);
          r2=count/(purchase2.size());
          //System.out.println(r2);
          ArrayList tempo= new ArrayList();
            tempo.add(comp.get(0));
            tempo.add(comp.get(1));
            freq.add(tempo);
          if(r1>=confidence)
          {
            System.out.println(comp.get(0)+" --> "+comp.get(1)+" with
confidence"+r1*100+"%");


          }
          if(r2>=confidence)
          {
            System.out.println(comp.get(1)+" --> "+comp.get(0)+" with
confidence"+r2*100+"%");

          }
        }

      }
      System.out.println("frequent 2-itemset"+freq);

    }



}
```

Kijit Desai
NJIT ID: 31246712

Screnshot of Database 1 execution:

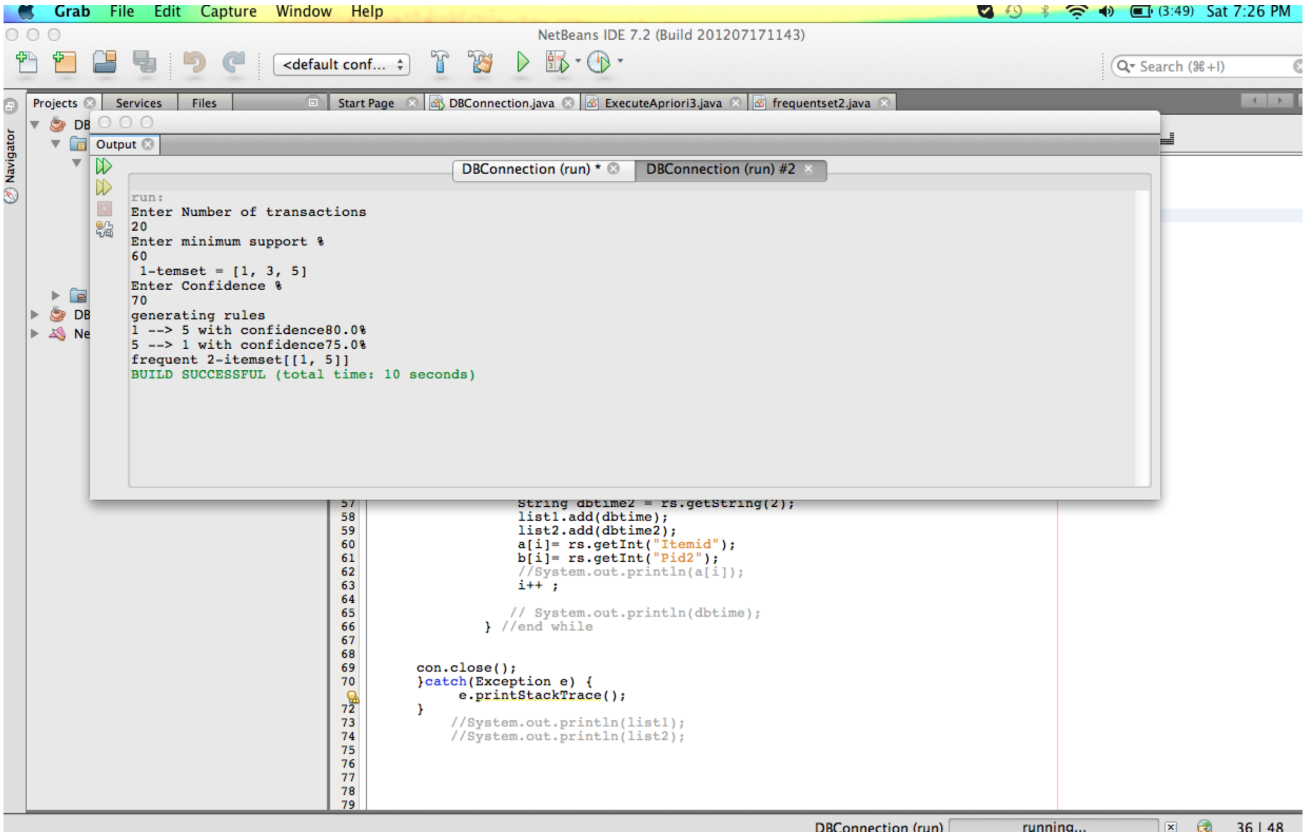


```
run:
Enter Number of transactions
20
Enter minimum support %
50
 1-temset = [2, 3, 4, 6]
Enter Confidence %
75
generating rules
2 --> 4 with confidence80.0%
4 --> 2 with confidence80.0%
frequent 2-itemset[[2, 3], [2, 4], [3, 4]]
BUILD SUCCESSFUL (total time: 14 seconds)
```

```
                String dbtime2 = rs.getString(2);
                list1.add(dbtime);
                list2.add(dbtime2);
                a[i]= rs.getInt("Itemid");
                b[i]= rs.getInt("Pid2");
                //System.out.println(a[i]);
                i++ ;

               // System.out.println(dbtime);
            } //end while


    con.close();
    }catch(Exception e) {
         e.printStackTrace();
    }
        //System.out.println(list1);
        //System.out.println(list2);
```

Kijit Desai
NJIT ID: 31246712

Screenshot of Database 2 Execution:

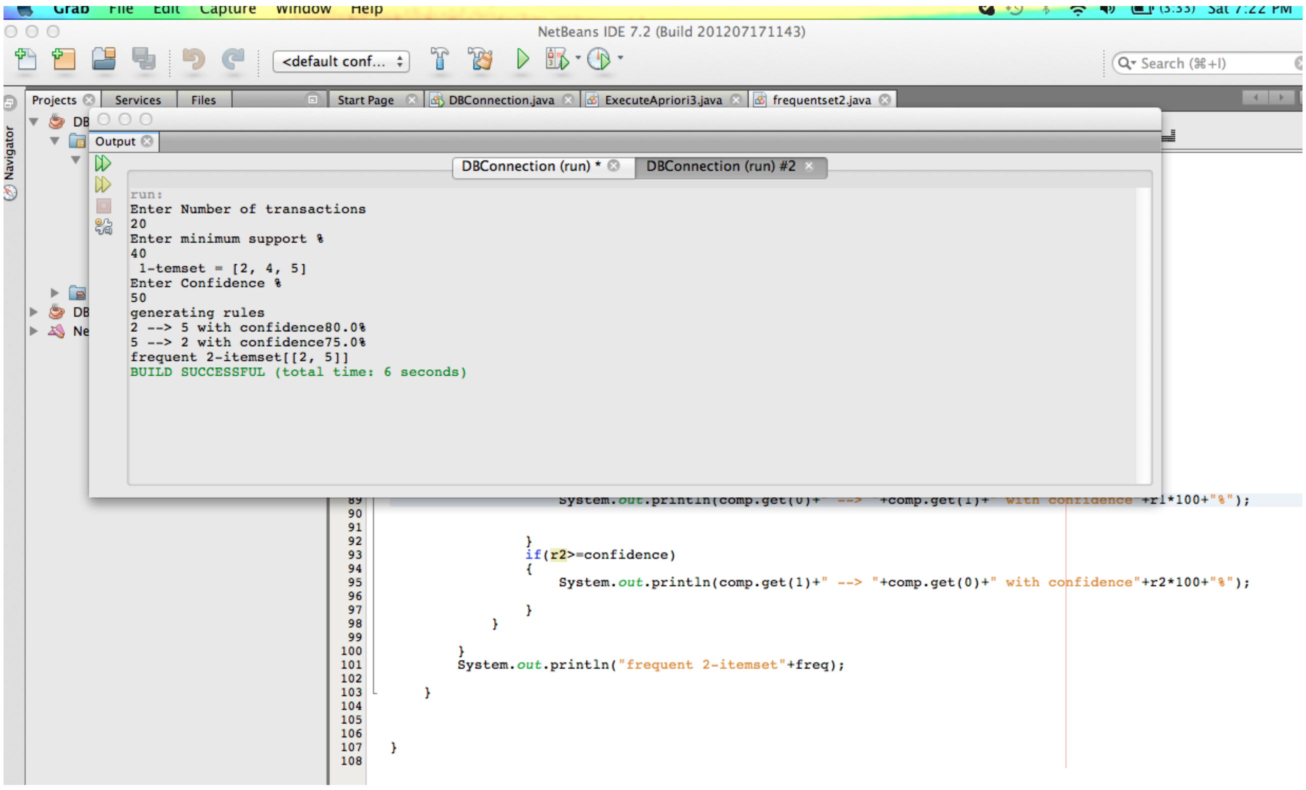


```
run:
Enter Number of transactions
20
Enter minimum support %
60
 1-temset = [1, 3, 5]
Enter Confidence %
70
generating rules
1 --> 5 with confidence80.0%
5 --> 1 with confidence75.0%
frequent 2-itemset[[1, 5]]
BUILD SUCCESSFUL (total time: 10 seconds)
```

```
                String dbtime2 = rs.getString(2);
                list1.add(dbtime);
                list2.add(dbtime2);
                a[i]= rs.getInt("Itemid");
                b[i]= rs.getInt("Pid2");
                //System.out.println(a[i]);
                i++ ;

               // System.out.println(dbtime);
            } //end while


        con.close();
        }catch(Exception e) {
            e.printStackTrace();
        }
            //System.out.println(list1);
            //System.out.println(list2);
```

Kijit Desai
NJIT ID: 31246712

Screenshot of Database 3 Execution:

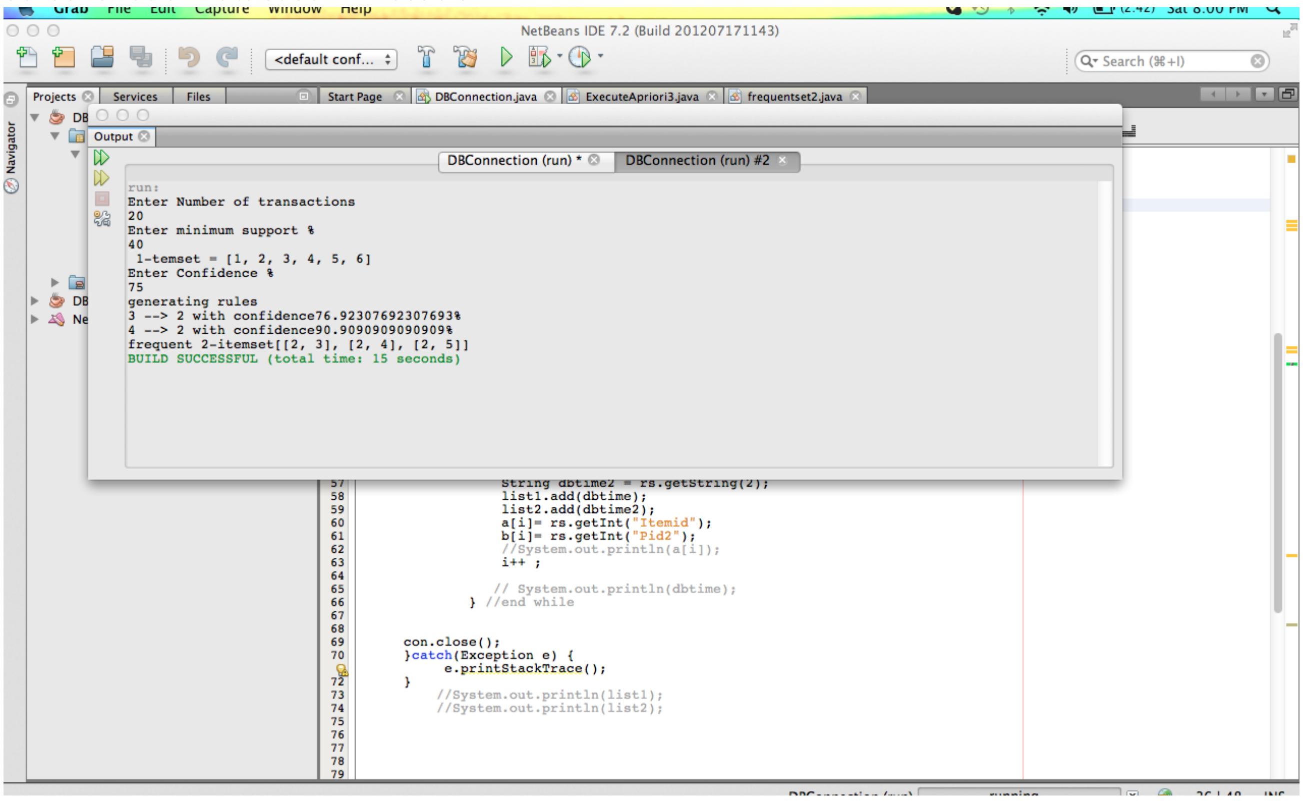


```
run:
Enter Number of transactions
20
Enter minimum support %
40
 1-temset = [2, 4, 5]
Enter Confidence %
50
generating rules
2 --> 5 with confidence80.0%
5 --> 2 with confidence75.0%
frequent 2-itemset[[2, 5]]
BUILD SUCCESSFUL (total time: 6 seconds)
```

```
                    System.out.println(comp.get(0)+" --> "+comp.get(1)+" with confidence"+r1*100+"%");


                }
                if(r2>=confidence)
                {
                    System.out.println(comp.get(1)+" --> "+comp.get(0)+" with confidence"+r2*100+"%");

                }
            }

        }
        System.out.println("frequent 2-itemset"+freq);

    }



}
```

Kijit Desai
NJIT ID: 31246712

Screenshot of Database 4 Execution:

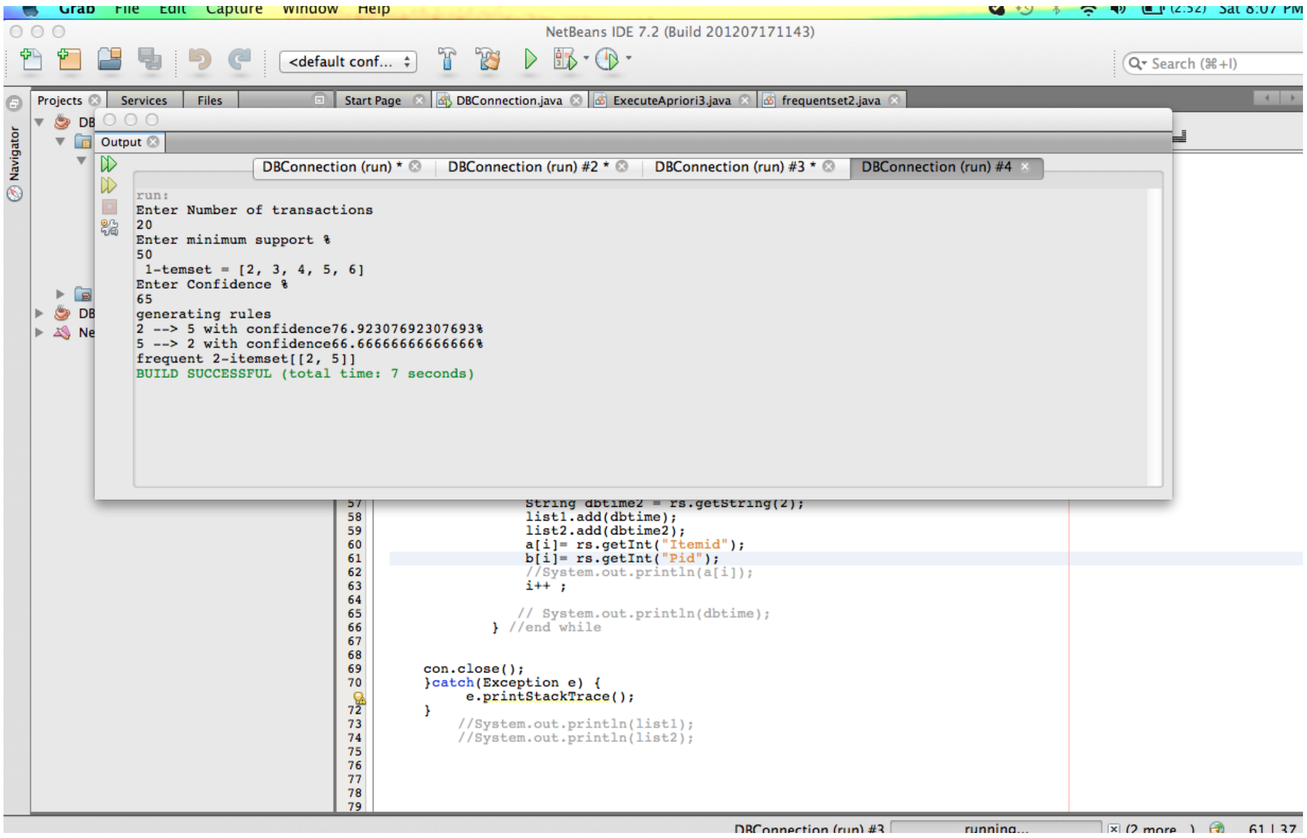


```
run:
Enter Number of transactions
20
Enter minimum support %
40
 1-temset = [1, 2, 3, 4, 5, 6]
Enter Confidence %
75
generating rules
3 --> 2 with confidence76.92307692307693%
4 --> 2 with confidence90.9090909090909%
frequent 2-itemset[[2, 3], [2, 4], [2, 5]]
BUILD SUCCESSFUL (total time: 15 seconds)
```

```
                 String dbtime2 = rs.getString(2);
                 list1.add(dbtime);
                 list2.add(dbtime2);
                 a[i]= rs.getInt("Itemid");
                 b[i]= rs.getInt("Pid2");
                 //System.out.println(a[i]);
                 i++ ;

                // System.out.println(dbtime);
            } //end while


      con.close();
      }catch(Exception e) {
           e.printStackTrace();
      }
          //System.out.println(list1);
          //System.out.println(list2);
```

Kijit Desai
NJIT ID: 31246712

Screenshot of Database 5 Execution:



```
run:
Enter Number of transactions
20
Enter minimum support %
50
 1-temset = [2, 3, 4, 5, 6]
Enter Confidence %
65
generating rules
2 --> 5 with confidence76.92307692307693%
5 --> 2 with confidence66.66666666666666%
frequent 2-itemset[[2, 5]]
BUILD SUCCESSFUL (total time: 7 seconds)
```

```
                String dbtime2 = rs.getString(2);
                list1.add(dbtime);
                list2.add(dbtime2);
                a[i]= rs.getInt("Itemid");
                b[i]= rs.getInt("Pid");
                //System.out.println(a[i]);
                i++ ;

               // System.out.println(dbtime);
            } //end while


    con.close();
    }catch(Exception e) {
         e.printStackTrace();
    }
        //System.out.println(list1);
        //System.out.println(list2);
```