```java
package Electric_Grid_MST;
import java.util.*;
class Connection{
    String node1, node2;
    int cost;
    public Connection(String a, String b, int c){
            node1 = a;
            node2 = b;
            cost = c;
    } }

public class MST {
    static class DisjointSet {
        Set<String> set;
        Map<String, String> map;
        int cnt = 0;
        public DisjointSet() {
            set = new HashSet<>();
            map = new HashMap<>();
        }

        public void makeSet(String s) {
            if (set.contains(s)) return;
            cnt++;
            set.add(s);
            map.put(s, s);
        }

        public String findRoot (String s1) {
            if (!set.contains(s1)) return null;
            String s = s1;
            while (!s1.equals(map.get(s1))) {
                s1 = map.get(s1);
            }
            map.put(s, s1);
            return s1;
        }

        public void union (String s1, String s2) {
            if (!set.contains(s1) || !set.contains(s2)) return;
            if (s1.equals(s2)) return;
            cnt--;
            map.put(s1, s2);
        }
    }


    public static List<Connection> getMST(List<Connection> connections) {
        Collections.sort(connections, new Comparator<Connection>() {
            @Override
            public int compare (Connection c1, Connection c2) {
                return c1.cost - c2.cost;
            }
        });
        DisjointSet ds = new DisjointSet();
        List<Connection> res = new ArrayList<>();
        for (Connection c : connections) {
            ds.makeSet(c.node1);
            ds.makeSet(c.node2);
        }

        for (Connection c : connections) {
            String s1 = ds.findRoot(c.node1);
            String s2 = ds.findRoot(c.node2);
            if (!s1.equals(s2)) {
                ds.union(c.node1, c.node2);
                res.add(c);
                if (ds.cnt == 1) break;
            }
        }

        if (ds.cnt == 1) {
            Collections.sort(res, new Comparator<Connection>() {
                @Override
                public int compare(Connection c1, Connection c2) {
                    if (c1.node1.equals(c2.node1)) {
                        return c1.node2.compareTo(c2.node2);
                    } else {
                        return c1.node1.compareTo(c2.node1);
                    }
                }
            });
            return res;
        }
        return new ArrayList<>();
    }
}
```