```java
package misc;

import java.io.File;
import java.io.FileNotFoundException;
import java.util.HashMap;
import java.util.Scanner;
import java.util.Set;

public class RecordProcessor {
    private static String [] firstNames;
    private static String [] lastNames;
    private static int [] employeeAge;
    private static String [] employeeType;
    private static double [] employeePay;

    public static String processFile(String f) {
        StringBuffer output = new StringBuffer();

        Scanner fileIn = null;
        try {
            fileIn = new Scanner(new File(f));
        } catch (FileNotFoundException e) {
            System.err.println(e.getMessage());
            return null;
        }

        int numOfRecords = 0;
        while(fileIn.hasNextLine()) {
            String l = fileIn.nextLine();
            if(l.length() > 0)
                numOfRecords++;
        }

        firstNames = new String[numOfRecords];
        lastNames = new String[numOfRecords];
        employeeAge = new int[numOfRecords];
        employeeType = new String[numOfRecords];
        employeePay = new double[numOfRecords];

        fileIn.close();
        try {
            fileIn = new Scanner(new File(f));
        } catch (FileNotFoundException e) {
            System.err.println(e.getMessage());
            return null;
        }

        numOfRecords = 0;
        while(fileIn.hasNextLine()) {
            String fileLine = fileIn.nextLine();
            if(fileLine.length() > 0) {

                String [] fileLineData = fileLine.split(",");

                int loopCount = 0;
                for(;loopCount < lastNames.length; loopCount++) {
                    if(lastNames[loopCount] == null)
                        break;

                    if(lastNames[loopCount].compareTo(fileLineData[1]) > 0) {
                        for(int i = numOfRecords; i > loopCount; i--) {
                            firstNames[i] = firstNames[i - 1];
                            lastNames[i] = lastNames[i - 1];
                            employeeAge[i] = employeeAge[i - 1];
                            employeeType[i] = employeeType[i - 1];
                            employeePay[i] = employeePay[i - 1];
                        }
                        break;
                    }
                }

                firstNames[loopCount] = fileLineData[0];
                lastNames[loopCount] = fileLineData[1];
                employeeType[loopCount] = fileLineData[3];

                try {
                    employeeAge[loopCount] = Integer.parseInt(fileLineData[2]);
                    employeePay[loopCount] = Double.parseDouble(fileLineData[4]);
                } catch(Exception e) {
                    System.err.println(e.getMessage());
                    fileIn.close();
                    return null;
                }

                numOfRecords++;
            }
        }

        if(numOfRecords == 0) {
            System.err.println("No records found in data file");
            fileIn.close();
            return null;
        }

        output.append(String.format("# of people imported: %d\n", firstNames.length));
        output.append(String.format("\n%-30s %s  %-12s %12s\n", "Person Name", "Age"
                                                                , "Emp. Type", "Pay"));
        for(int i = 0; i < 30; i++) output.append(String.format("-"));
        output.append(String.format(" ---  "));
        for(int i = 0; i < 12; i++) output.append(String.format("-"));
        output.append(String.format(" "));
        for(int i = 0; i < 12; i++) output.append(String.format("-"));
        output.append(String.format("\n"));

        for(int i = 0; i < firstNames.length; i++) {
            output.append(String.format("%-30s %-3d  %-12s $%12.2f\n", firstNames[i] + " "
                                    + lastNames[i], employeeAge[i], employeeType[i], employeePay[i]));
        }

        int sumOfAges = 0;
        float averageAge = 0f;
        int numComissionEmployee = 0, numHourlyEmployee = 0, numSalaryEmployee = 0;
        double sumOfCommissionPay = 0, sumOfHourlyPay = 0, sumOfSalaryPay = 0;
        double averageCommission = 0, averageHourly = 0, averageSalary = 0;
        for(int i = 0; i < firstNames.length; i++) {
            sumOfAges += employeeAge[i];
            if(employeeType[i].equals("Commission")) {
                sumOfCommissionPay += employeePay[i];
                numComissionEmployee++;
            } else if(employeeType[i].equals("Hourly")) {
                sumOfHourlyPay += employeePay[i];
                numHourlyEmployee++;
            } else if(employeeType[i].equals("Salary")) {
                sumOfSalaryPay += employeePay[i];
                numSalaryEmployee++;
            }
        }
        averageAge = (float) sumOfAges / firstNames.length;
        output.append(String.format("\nAverage age:         %12.1f\n", averageAge));
        averageCommission = sumOfCommissionPay / numComissionEmployee;
        output.append(String.format("Average commission:  $%12.2f\n", averageCommission));
        averageHourly = sumOfHourlyPay / numHourlyEmployee;
        output.append(String.format("Average hourly wage: $%12.2f\n", averageHourly));
        averageSalary = sumOfSalaryPay / numSalaryEmployee;
        output.append(String.format("Average salary:      $%12.2f\n", averageSalary));

        HashMap<String, Integer> firstNameMap = new HashMap<String, Integer>();
        int numOfDuplicateFirstNames = 0;
        for(int i = 0; i < firstNames.length; i++) {
            if(firstNameMap.containsKey(firstNames[i])) {
                firstNameMap.put(firstNames[i], firstNameMap.get(firstNames[i]) + 1);
                numOfDuplicateFirstNames++;
            } else {
                firstNameMap.put(firstNames[i], 1);
            }
        }

        output.append(String.format("\nFirst names with more than one person sharing it:\n"));
        if(numOfDuplicateFirstNames > 0) {
            Set<String> set = firstNameMap.keySet();
            for(String name : set) {
                if(firstNameMap.get(name) > 1) {
                    output.append(String.format("%s, # people with this name: %d\n", name
                                                            , firstNameMap.get(name)));
                }
            }
        } else {
            output.append(String.format("All first names are unique"));
        }

        HashMap<String, Integer> lastNameMap = new HashMap<String, Integer>();
        int numOfDuplicateLastNames = 0;
        for(int i = 0; i < lastNames.length; i++) {
            if(lastNameMap.containsKey(lastNames[i])) {
                lastNameMap.put(lastNames[i], lastNameMap.get(lastNames[i]) + 1);
                numOfDuplicateLastNames++;
            } else {
                lastNameMap.put(lastNames[i], 1);
            }
        }

        output.append(String.format("\nLast names with more than one person sharing it:\n"));
        if(numOfDuplicateLastNames > 0) {
            Set<String> set = lastNameMap.keySet();
            for(String name : set) {
                if(lastNameMap.get(name) > 1) {
                    output.append(String.format("%s, # people with this name: %d\n", name
                                                                , lastNameMap.get(name)));
                }
            }
        } else {
            output.append(String.format("All last names are unique"));
        }

        //close the file
        fileIn.close();

        return output.toString();
    }

}
```