REGISTER /usr/hadoop/anonymizer.jar

***--Load the data from its location***

1. Data = LOAD 'hdfs://zobbi01:9000/input/dataset' using PigStorage(',') as (f0:chararray,f1:chararray,f2:chararray,f3:chararray);

***--Filter class by class value***

1. G0 = FILTER Data by (f3 matches 'yes');
2. G1 = FILTER Data by (f3 matches 'no');

***--Group each G***

1. G00= group G0 by (f0,f1,f2);
2. G000= foreach G00 generate COUNT (G0) as cnt0:long, G0.f0 as v0, G0.f1 as v1,G0.f2 as v2,G0.f3 as v3;
3. --Filter out the full-equivalent records
4. SG0= FILTER G000 by (cnt0 >= 20);
5. --Ungroup the records and store them in SG
6. SG00 = FOREACH SG0 GENERATE anonymizer.adjust(v0,v1,v2,v3);
7. STORE SG00 into 'hdfs://zobbi01:9000/output/1/SG0' using PigStorage(',');
8. --Filter out the non-equivalent records, ungroup and store them in SSG
9. SSG0= FILTER G000 by (cnt0 < 20);
10. SSG00 = FOREACH SSG0 GENERATE anonymizer.adjust(v0,v1,v2,v3,v4,v5,v6,v7,v8,v9,v10);
11. STORE SSG00 into 'hdfs://zobbi01:9000/input/1/SSG0' using PigStorage(',');
12. G101= group G1 by (f0,f1,f2);
13. G1001= foreach G101 generate COUNT (G1) as cnt1:long, G1.f0 as v0, G1.f1 as v1,G1.f2 as v2,G1.f3 as v3;
14. --Filter out the full-equivalent records
15. SG1= FILTER G1001 by (cnt1 >= 20);
16. --Ungroup the records and store them in SG
17. SG101 = FOREACH SG1 GENERATE anonymizer.adjust(v0,v1,v2,v3);
18. STORE SG101 into 'hdfs://zobbi01:9000/output/1/SG1' using PigStorage(',');
19. --Filter out the non-equivalent records, ungroup and store them in SSG
20. SSG01= FILTER G1001 by (cnt1 < 20);
21. SSG101 = FOREACH SSG01 GENERATE anonymizer.adjust(v0,v1,v2,v3);
22. STORE SSG101 into 'hdfs://zobbi01:9000/input/1/SSG1' using PigStorage(',');

***--GROUP with TWO Q-IDS***

1. S\_Data0 = LOAD 'hdfs://zobbi01:9000/input/1/SSG0' using PigStorage(',') as (f0:chararray,f1:chararray,f2:chararray,f3:chararray);
2. S\_G00= group S\_Data0 by (f1,f2);
3. S\_G000= foreach S\_G00 generate COUNT (S\_Data0) as count0:long, S\_Data0.f0 as v0, S\_Data0.f1 as v1,S\_Data0.f2 as v2,S\_Data0.f3 as v3;
4. S\_SG0= FILTER S\_G000 by (count0 >= 20);
5. S\_SG00 = FOREACH S\_SG0 GENERATE anonymizer.SSG\_P(v0,v1,v2,v3);
6. STORE S\_SG00 into 'hdfs://zobbi01:9000/output/1/SG16' using PigStorage(',');
7. S\_SSG0= FILTER S\_G000 by (count0 < 20);
8. S\_SSG00 = FOREACH S\_SSG0 GENERATE anonymizer.adjust(v0,v1,v2,v3);
9. STORE S\_SSG00 into 'hdfs://zobbi01:9000/input/1/SSG16' using PigStorage(',');
10. S\_Data1 = LOAD 'hdfs://zobbi01:9000/input/1/SSG1' using PigStorage(',') as (f0:chararray,f1:chararray,f2:chararray,f3:chararray);
11. S\_G01= group S\_Data1 by (f1,f2);
12. S\_G011= foreach S\_G01 generate COUNT (S\_Data1) as count1:long, S\_Data1.f0 as v0, S\_Data1.f1 as v1,S\_Data1.f2 as v2,S\_Data1.f3 as v3;
13. S\_SG1= FILTER S\_G011 by (count1 >= 20);
14. S\_SG01 = FOREACH S\_SG1 GENERATE anonymizer.SSG\_P1(v0,v1,v2,v3);
15. STORE S\_SG01 into 'hdfs://zobbi01:9000/output/1/SG17' using PigStorage(',');
16. S\_SSG1= FILTER S\_G011 by (count1 < 20);
17. S\_SSG01 = FOREACH S\_SSG1 GENERATE anonymizer.adjust(v0,v1,v2,v3);
18. STORE S\_SSG01 into 'hdfs://zobbi01:9000/input/1/SSG17' using PigStorage(',');

***--GROUP with ONE Q-IDS***

1. S\_Data16 = Load 'hdfs://zobbi01:9000/input/1/SSG16' using PigStorage(',') as (f0:chararray,f1:chararray,f2:chararray,f3:chararray);
2. S\_G016= group S\_Data16 by (f2);
3. S\_G01616= foreach S\_G016 generate COUNT (S\_Data16) as count16:long, S\_Data16.f0 as v0, S\_Data16.f1 as v1,S\_Data16.f2 as v2,S\_Data16.f3 as v3;
4. S\_SG16= FILTER S\_G01616 by (count16 >= 20);
5. S\_SG016 = FOREACH S\_SG16 GENERATE trial.NG\_interval(v0,v1,v2,v3);
6. STORE S\_SG016 into 'hdfs://zobbi01:9000/output/1/SG32' using PigStorage(',');
7. S\_SSG16= FILTER S\_G01616 by (count16 < 20);
8. S\_SSG016 = FOREACH S\_SSG16 GENERATE trial.suppress(v0,v1,v2,v3);
9. STORE S\_SSG016 into 'hdfs://zobbi01:9000/output/1/SG\_SP32' using PigStorage(',');
10. S\_Data17 = Load 'hdfs://zobbi01:9000/input/1/SSG17' using PigStorage(',') as (f0:chararray,f1:chararray,f2:chararray,f3:chararray);
11. S\_G017= group S\_Data17 by (f2);
12. S\_G01717= foreach S\_G017 generate COUNT (S\_Data17) as count17:long, S\_Data17.f0 as v0, S\_Data17.f1 as v1,S\_Data17.f2 as v2,S\_Data17.f3 as v3;
13. S\_SG17= FILTER S\_G01717 by (count17 >= 20);
14. S\_SG017 = FOREACH S\_SG17 GENERATE trial.NG\_interval(v0,v1,v2,v3);
15. STORE S\_SG017 into 'hdfs://zobbi01:9000/output/1/SG33' using PigStorage(',');
16. S\_SSG17= FILTER S\_G01717 by (count17 < 20);
17. S\_SSG017 = FOREACH S\_SSG17 GENERATE trial.suppress(v0,v1,v2,v3);
18. STORE S\_SSG017 into 'hdfs://zobbi01:9000/output/1/SG\_SP33' using PigStorage(',');