Data Analysis & Report

Name: Jingyi Lin
NUID: 001888193

1. Data Resource Introduction

2. Technology Stack

3. Analysis & Implementation

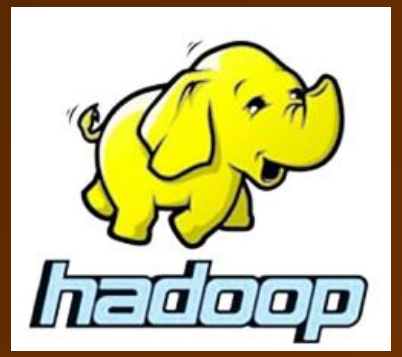
1. Data Resource Introduction

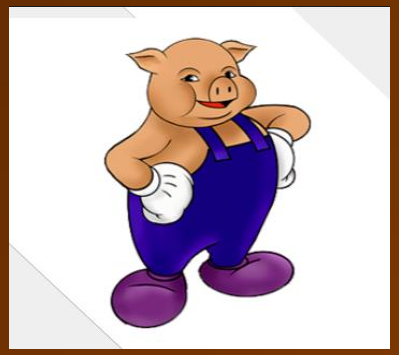
Los Angeles Crime & Arrest Data

- This is a dataset hosted by the **city of Los Angeles**. The organization has an **open data platform** found in Kaggle and they update their information according the amount of data that is brought in. All of the data sources available through the city of Los Angeles organization page.

- **Arrest-data-from-2010-to-present**
- **Crime-data-from-2010-to-present**

2. Technology Stack


hadoop



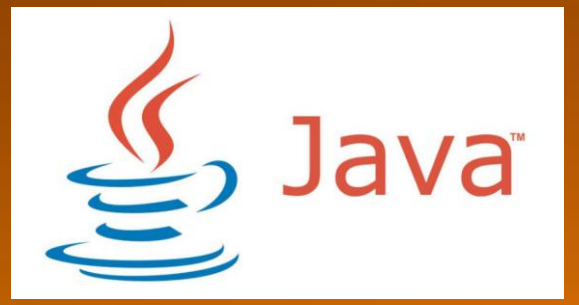

HIVE


mahout


MapReduce


hadoop
HDFS


ubuntu


Java

Data Storage	Data Analysis	Data Testing
Use HDFS to Save Data	Use Map-Reduce framework with Java Code to do the data preprocessing work	Use Hive to write SQL queries to check the result
Use Hive to store Structured data for simple query	Use pig script with Pig Latin to do the main analysis	
	Use Mahout to do the simple KMeans Clustering	

3. Analysis & Implementation

Data Preprocessing

- Original Dataset
- Arrest

Attribute Name
Report ID
Arrest Date
Time
Area ID
Area Name
Reporting District
Age
Sex Code
Descent Code

Attribute Name
Charge Group Code
Charge Group Description
Arrest Type Code
Charge
Charge Description
Address
Cross Street
Location

Crime

Attribute Name
DR Number
Date Reported
Date Occurred
Time Occurred
Area ID
Area Name
Reporting District
Crime Code
Crime Code Description

Attribute Name
MO Codes
Victim Age
Victim Sex
Victim Descent
Premise Code
Premise Description
Weapon Used Code
Weapon Description
Status Code

Attribute Name
Status Description
Crime Code 1
Crime Code 2
Crime Code 3
Crime Code 4
Address
Cross Street
Location
Status Description

Data Preprocessing – Useless Column

Useless column in Arrest:

- Time: 24 hour military time: It's useless for our Arrest analysis purpose
- Cross Street: Cross Street of rounded Address: most of values in this attribute is are null.

Useless column in Crime:

- MO codes: Activities associated with the suspect in commission of the crime: It's useless for our Arrest analysis purpose.
- Cross Street: most of values in this attribute are null.
- Crime code 1: duplicate with Crime code
- Crime code 3: null
- Crime code 4: null

Data Preprocessing – Duplicate Column

Arrest:

- Area ID/AreaName
- Charge Group Code/Charge Group Description
- Charge/Charge Description

Crime:

- Area ID/AreaName
- Crime Code/Crime Code Description
- Premise Code/Premise Description
- Weapon Used Code/Weapon Description
- Status Code/Status Description

MapReduce Filtering Patterns: Distinct Pattern

1. Extract Duplicated data

- Notes: Change delimiter from "," to ";" is necessary because some value contains ",".

AreaID	AreaName
1	Central
10	West Valley
11	Northeast
12	77th Street
13	Newton
…	…

```
public static class DistinctAreaArrestMapper extends Mapper<Object,Text,Text,NullWritable>{
    private Text area = new Text();
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException{
        if(value.toString().contains("Report ID")) {
            return;
        }
        String[] list = value.toString().split(";");
        area.set(list[3]+"\t"+list[4]);
        context.write(area, NullWritable.get());
    }
}
public static class DistinctAreaArrestReducer extends Reducer<Text,NullWritable,Text,NullWritable>{
    public void reduce(Text key, Iterable<NullWritable> values, Context context) throws IOException, InterruptedException{
        context.write(key,NullWritable.get());
    }
}
```

ChargeCode	ChargeDescription
103.102LAMC	CAFE ENTERTAINMENT VIOL
103.106BLAM	CONDUCT DANCE W/O PERMIT
103.107.1BL	ESCORT WITHOUT PERMIT
103.107BLAM	RUN ESCORT SERVICE W/O PERMIT
103.112ALAM	BUSINESS REGS
…	…

```
public static class DistinctChargeMapper extends Mapper<Object,Text,Text,NullWritable>{
    private Text area = new Text();
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException{
        if(value.toString().contains("Report ID")) {
            return;
        }
        String[] list = value.toString().split(";");
        area.set(list[12]+"\t"+list[13]);
        context.write(area, NullWritable.get());
    }
}
public static class DistinctChargeReducer extends Reducer<Text,NullWritable,Text,NullWritable>{
    public void reduce(Text key, Iterable<NullWritable> values, Context context) throws IOException, InterruptedException{
        context.write(key,NullWritable.get());
    }
}
```

MapReduce Filtering Patterns: Distinct Pattern

1. Extract Duplicated data

ChargeGroupCode	ChargeGroupDescription
1	Homicide
10	Fraud/Embezzlement
11	Receive Stolen Property
12	Weapon (carry/poss)
13	Prostitution/Allied
…	…

```
public static class DistinctChargeGroupMapper extends Mapper<Object,Text,Text,NullWritable>{
    private Text area = new Text();
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException{
        if(value.toString().contains("Report ID")) {
            return;
        }
        String[] list = value.toString().split(";");
        area.set(list[9]+"\t"+list[10]);
        context.write(area, NullWritable.get());
    }
}
public static class DistinctChargeGroupReducer extends Reducer<Text,NullWritable,Text,NullWritable>{
    public void reduce(Text key, Iterable<NullWritable> values, Context context) throws IOException, InterruptedException{
        context.write(key,NullWritable.get());
    }
}
```

MapReduce Filtering Patterns: Distinct Pattern

1. Extract Duplicated data

AreaID	AreaName
1	Central
10	West Valley
11	Northeast
12	77th Street
13	Newton
…	…

PremiseCode	PremiseDescription
101	STREET
102	SIDEWALK
103	ALLEY
104	DRIVEWAY
105	PEDESTRIAN OVERCROSSING
…	…

StatusCode	StatusDescription
13	UNK
19	UNK
AA	Adult Arrest
AO	Adult Other
CC	UNK
…	…

CrimeCode	CrimeCodeDescription
110	CRIMINAL HOMICIDE
113	MANSLAUGHTER, NEGLIGENT
121	RAPE, FORCIBLE
122	RAPE, ATTEMPTED
210	ROBBERY
…	…

WeapoUsedCode	WeaponDescription
101	REVOLVER
102	HAND GUN
103	RIFLE
104	SHOTGUN
105	SAWED OFF RIFLE/SHOTGUN
…	…

MapReduce Filtering Patterns: Distinct Pattern

2. Clean Main Data and Add Year Month features

- Arrest

Attribute Name
Report ID
Arrest Date
Year
Month
Area ID
Reporting District
Age
Sex Code
Descent Code
Charge Group Code
Arrest Type Code
Charge
Location

```
public static class CleanArrestMapper extends Mapper<Object,Text,Text,NullWritable>{
    private Text area = new Text();
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException{
        if(value.toString().contains("Report ID")) {
            return;
        }
        String[] list = value.toString().split(";");
        String y = list[1].substring(0,4);
        String m = list[1].substring(5,7);
        area.set(list[0]+";"+list[1]+";"+y+";"+m+";"+list[3]+";"+list[5]+";"+list[6]+";"+list[7]+";"
            +list[8]+";"+list[9]+";"+list[11]+";"+list[12]+";"+list[14]+";"+list[16]);
        context.write(area, NullWritable.get());
    }
}
public static class CleanArrestReducer extends Reducer<Text,NullWritable,Text,NullWritable>{
    public void reduce(Text key, Iterable<NullWritable> values, Context context)
    throws IOException, InterruptedException{
        context.write(key,NullWritable.get());
    }
}
```

```
public static class CleanCrimeMapper extends Mapper<Object,Text,Text,NullWritable>{
    private Text area = new Text();
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException{
        if(value.toString().contains("DR Number")) {
            return;
        }
        String[] list = value.toString().split(";");
        String y = list[1].substring(0,4);
        String m = list[1].substring(5,7);
        area.set(list[0]+";"+list[1]+";"+y+";"+m+";"+list[2]+";"+list[3]+";"+list[4]+";"
            +list[6]+";"+list[7]+";"+list[10]+";"+list[11]+";"+list[12]+";"+list[13]+";"+list[15]+";"
            +list[17]+";"+list[20]+";"+list[23]+";"+list[25]);
        context.write(area, NullWritable.get());
    }
}
public static class CleanCrimeReducer extends Reducer<Text,NullWritable,Text,NullWritable>{
    public void reduce(Text key, Iterable<NullWritable> values, Context context)
    throws IOException, InterruptedException{
        context.write(key,NullWritable.get());
    }
}
```

Crime

Attribute Name
DR Number
Date Reported
Year
Month
Date Occurred
Time Occurred
Area ID
Reporting District
Crime Code

Attribute Name
Victim Age
Victim Sex
Victim Descent
Premise Code
Weapon Used Code
Status Code
Crime Code 2
Address
Location

MapReduce Summarization Patterns: Counter Pattern

Count arrest number by Year.

2010	162459
2011	157696
2012	163438
2013	152852
2014	139737
2015	126696
2016	118656
2017	104567
2018	102339
2019	21607

```java
public static class CounterArrestMapper extends Mapper<Object, Text, NullWritable, NullWritable>{
    public static final String YEAR_COUNTER_GROUP = "Year";
    public static final String UNKNOWN_COUNTER="Unknown";
    public static final String NULL_OR_EMPTY_COUNTER = "Null or Empty";
    private String[] YEAR = new String[] {
            "2010","2011","2012","2013","2014","2015","2016","2017","2018","2019"
    };
    private HashSet<String> YEARSet = new HashSet<String>(Arrays.asList(YEAR));
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException{
        String[] line = value.toString().split(";");
        String year = line[1].substring(0,4);
        if (year != null && !year.isEmpty()) {
            if (YEARSet.contains(year)) {
                context.getCounter(YEAR_COUNTER_GROUP, year).increment(1);
            }else {
                context.getCounter(YEAR_COUNTER_GROUP,UNKNOWN_COUNTER).increment(1);
            }
        }else {
            context.getCounter(YEAR_COUNTER_GROUP,NULL_OR_EMPTY_COUNTER).increment(1);
        }
    }
}
```

MapReduce Summarization Patterns: Counter Pattern

Count crime reported number by Year.

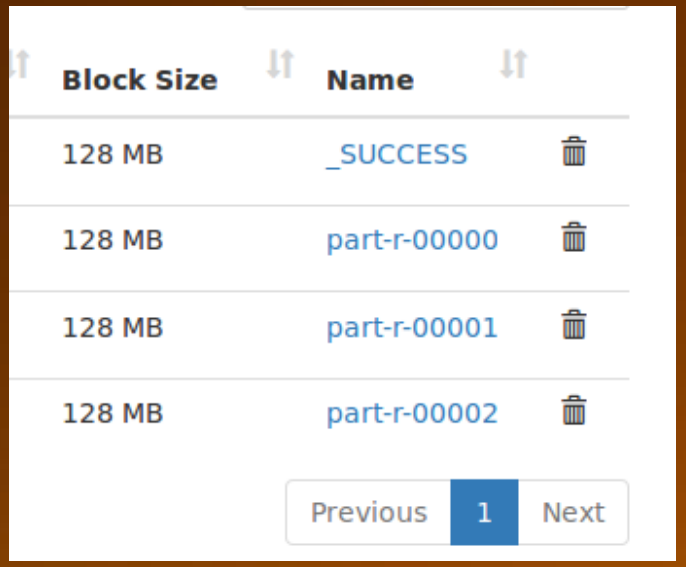
2010	200507
2011	197763
2012	200011
2013	192032
2014	194883
2015	214930
2016	225864
2017	231561
2018	230467
2019	54320

```java
public static class CounterCrimeMapper extends Mapper<Object, Text, NullWritable, NullWritable>{
    public static final String YEAR_COUNTER_GROUP = "Year";
    public static final String UNKNOWN_COUNTER="Unknown";
    public static final String NULL_OR_EMPTY_COUNTER = "Null or Empty";
    private String[] YEAR = new String[] {
            "2010","2011","2012","2013","2014","2015","2016","2017","2018","2019"
    };
    private HashSet<String> YEARSet = new HashSet<String>(Arrays.asList(YEAR));
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException{
        String[] line = value.toString().split(";");
        String year = line[1].substring(0,4);
        if (year != null && !year.isEmpty()) {
            if (YEARSet.contains(year)) {
                context.getCounter(YEAR_COUNTER_GROUP, year).increment(1);
            }else {
                context.getCounter(YEAR_COUNTER_GROUP,UNKNOWN_COUNTER).increment(1);
            }
        }else {
            context.getCounter(YEAR_COUNTER_GROUP,NULL_OR_EMPTY_COUNTER).increment(1);
        }
    }
}
```

MapReduce Organization Patterns: Partitioning Pattern

Spilt two main datasets by YEAR

Arrest

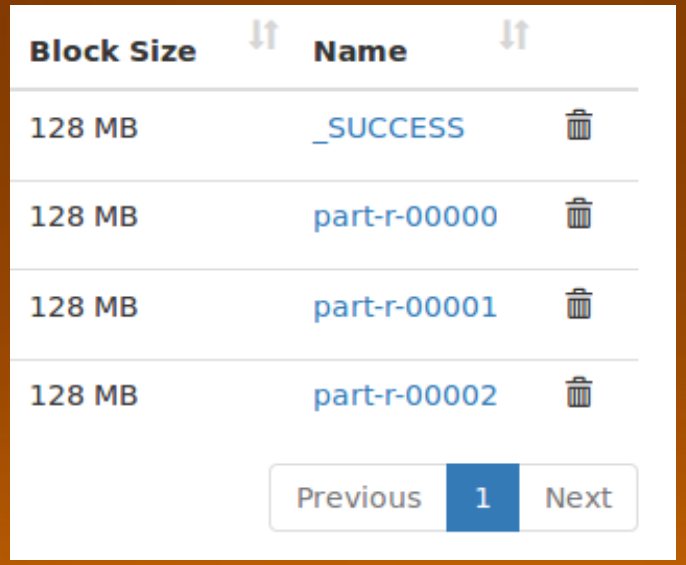


Block Size	Name
128 MB	_SUCCESS
128 MB	part-r-00000
128 MB	part-r-00001
128 MB	part-r-00002

Previous 1 Next

2010-2012
2013-2015
2016-2019

Crime



Block Size	Name
128 MB	_SUCCESS
128 MB	part-r-00000
128 MB	part-r-00001
128 MB	part-r-00002

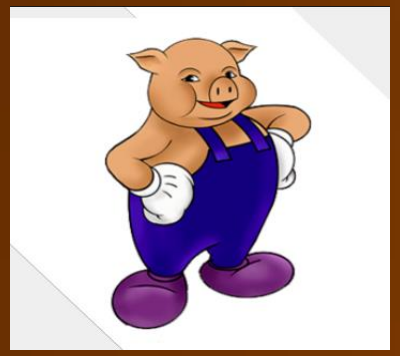
Previous 1 Next

```java
public static class PartiArrestMapper extends Mapper<Object, Text, Text, NullWritable> {
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
        context.write(value,NullWritable.get());
    }
}
public static class PartiPartitioner extends Partitioner<Text, NullWritable> {

    @Override
    public int getPartition(Text key, NullWritable value, int numPartitions) {
        String[] line = key.toString().split(";");
        String y = line[1].substring(0,4);
        int year = Integer.parseInt(y.toString());
        int partition = 0;
        if(numPartitions == 0) {
            partition = 0;
        }
        if(year<2013) {
            partition = 0;
        }
        else if(2013<=year && year<= 2015) {
            partition = 1 % numPartitions;
        }
        else {
            partition = 2 % numPartitions;
        }
        return partition;
    }

}
public static class PartiArrestReducer extends Reducer<Text, Text, Text, NullWritable> {
    public void reduce(Text key, Iterable<Text> values, Context context)
            throws IOException, InterruptedException {
        context.write(key, NullWritable.get());
    }
}
```

Other

- Also use MapReduce distinct pattern to prepare data for Mahout.
- Details will be introduced later.

Use Pig to Analyze:



- (Take 2016-2019 dataset as example to explain the process and implementation, 2013-2015 dataset will do the same analysis and implementation)

Load data & define data type with column name

Arrest

```
otable = LOAD '/FinalProject/PartiArrest/part-r-00002' USING PigStorage(';') AS (ReportID:long,Ar
restDate:chararray,Year:chararray,Month:chararray,AreaID:long,ReportingDistrict:long,Age:int,SexC
ode:chararray,DescentCode:chararray,ChargeGroupCode:chararray,ArrestTypeCode:chararray,Charge:cha
rarray,Address:chararray,Location:chararray);
```

Crime

```
otable = LOAD '/FinalProject/PartiCrime/part-r-00002' USING PigStorage(';') AS (DRNumber:long,Da
teReported:chararray,Year:chararray,Month:chararray,DateOccurred:chararray,TimeOccurred:long,
AreaID:long,ReportingDistrict:long,CrimeCode:long,VictimAge:int,VictimSex:chararray,
VictimDescent:chararray,PremiseCode:long,WeaponUsed:long,StatusCode:chararray,CrimeCode2:long,Ad
dress:chararray,Location:chararray);
```

Use Pig to Analyze:
Arrest dataset analysis

- Sorted Area by the incidence of Arrest: left outer join

1	40820	Central
6	29901	Hollywood
14	25732	Pacific
9	20634	Van Nuys
12	20337	77th Street
3	20242	Southwest
2	19234	Rampart
13	17517	Newton
19	15947	Mission
15	15705	N Hollywood
20	12996	Olympic
18	12977	Southeast
5	12560	Harbor
4	11601	Hollenbeck
10	11429	West Valley
11	10954	Northeast
21	10865	Topanga
17	10545	Devonshire
16	10359	Foothill
8	8796	West LA
7	8018	Wilshire

```
/*Count Arrest number in each areas 2016-2019.*/
groupArea = GROUP otable BY (AreaID);
count = FOREACH groupArea GENERATE group, COUNT(otable) AS sum;

/*left join with dataset that has areas' detail*/
areadetail = LOAD '/FinalProject/DistinctAreaCrime/part-r-00000' AS (
AreaID:long,AreaName:chararray);
joindata1 = JOIN count BY $0 LEFT OUTER, areadetail BY $0;
joindata = FOREACH joindata1 GENERATE $0,$1,$3;

/*sorted by number*/
sorted = ORDER joindata BY sum DESC;

STORE sorted INTO '/FinalProject/PigOut/SortedCrimeArea';
```

Conclusion: Central is the area where Arrest happens most and Hollywood, Pacific and Wan Nuys follow by.

Use Pig to Analyze:
Arrest dataset analysis

- Sorted Arrest type by the incidence of Arrest.
- D - Dependent F - Felony I - Infraction M - Misdemeanor O – Other

```
M    193825
F    114769
I    29361
O    7554
D    1660
```

2016-2019

```
groupType = GROUP otable BY (ArrestTypeCode);
typeCount = FOREACH groupType GENERATE group, COUNT(otable) AS sum;

sortedType = ORDER typeCount BY sum DESC;
STORE sortedType INTO '/FinalProject/PigOut/SortedArrestType';
```

```
M    255304
F    130181
I    17513
O    13384
D    2903
```

2013-2015

Conclusion: Most of people are arrested for misdemeanor. We can also conclude that the numbers of each kind of arrest are declining.

Use Pig to Analyze:
Arrest dataset analysis

- Proportion of 2 genders being arrested.
- (use COUNT,SUM,ROUND_TO,CONCAT)

F	73935	21.3%
M	273234	78.7%

2016-2019

F	88109	21.01%
M	331176	78.99%

2013-2015

```
groupGender = Group otable BY (SexCode);
genderCount = FOREACH groupGender GENERATE group, COUNT(otable) AS sum;
temp = GROUP genderCount ALL;
gendersum = FOREACH temp GENERATE SUM(genderCount.sum) AS total;
temp2 = FOREACH genderCount GENERATE $0, $1,ROUND_TO((sum/(double)gendersum.total)*100,2) AS
perc;
result = FOREACH temp2 GENERATE $0, $1,CONCAT((chararray)perc,'%');

STORE result INTO '/FinalProject/PigOut/ArrestGenderProportion1315';
```

Conclusion: Males are the main group of detainees. The total number of people being arrest is declining.

Use Pig to Analyze:
Arrest dataset analysis

- Month ratio of Arrest. (use COUNT,SUM,ROUND_TO,CONCAT)

01	35198	10.14%
02	33545	9.66%
03	35525	10.23%
04	28303	8.15%
05	29755	8.57%
06	27621	7.96%
07	28660	8.26%
08	30054	8.66%
09	27527	7.93%
10	25540	7.36%
11	22100	6.37%
12	23341	6.72%

```
/*month ratio of Arrest*/
groupMonth = Group otable BY (Month);
monthCount = FOREACH groupMonth GENERATE group, COUNT(otable) AS sum;
monthtemp = GROUP monthCount ALL;
monthsum = FOREACH monthtemp GENERATE SUM(monthCount.sum) AS total;
monthtemp2 = FOREACH monthCount GENERATE $0,$1,ROUND_TO((sum/(double)monthsum.total)*100,2) AS
perc;

result = FOREACH monthtemp2 GENERATE $0,$1,CONCAT((chararray)perc,'%');

STORE result INTO '/FinalProject/PigOut/ArrestMonthProportion';
```

Conclusion: Arrest happens most on January, February and March. Cold winter has the least number of arrest.

Use Pig to Analyze:
Arrest dataset analysis

- Proportion of different age being arrested.

- The result is from 0 years old to 92 years old. More analysis of this topic will be discussed in Hive Part.

```
groupAge = GROUP otable BY (Age);
ageCount = FOREACH groupAge GENERATE group, COUNT(otable) AS sum;
agetemp = GROUP ageCount ALL;
agesum = FOREACH agetemp GENERATE SUM(ageCount.sum) AS total;
agetemp2 = FOREACH ageCount GENERATE $0,$1,ROUND_TO((sum/(double)agesum.total)*100,4) AS perc;

result = FOREACH agetemp2 GENERATE $0,$1,CONCAT((chararray)perc,'%');

STORE result INTO '/FinalProject/PigOut/ArrestAgeProportion';
```

Use Pig to Analyze:
Crime dataset analysis

- Sorted area by the incidence of Crime reported in 2016-2019: left outer join.

12	49447	77th Street
3	46675	Southwest
15	39633	N Hollywood
14	39249	Pacific
1	39063	Central
18	38235	Southeast
13	36342	Newton
6	36092	Hollywood
20	35491	Olympic
21	35020	Topanga
19	34673	Mission
9	34337	Van Nuys
11	33850	Northeast
17	32980	Devonshire
7	32120	Wilshire
10	31233	West Valley
2	30933	Rampart
8	30859	West LA
5	30426	Harbor
4	29290	Hollenbeck
16	26264	Foothill

```
groupArea = GROUP otable BY (AreaID);
count = FOREACH groupArea GENERATE group, COUNT(otable) AS sum;

/*left join with dataset that has areas' detail*/
areadetail = LOAD '/FinalProject/DistinctAreaCrime/part-r-00000' AS (
AreaID:long,AreaName:chararray);
joindata1 = JOIN count BY $0 LEFT OUTER, areadetail BY $0;
joindata = FOREACH joindata1 GENERATE $0,$1,$3;

/*sorted by number*/
sorted = ORDER joindata BY sum DESC;

STORE sorted INTO '/FinalProject/PigOut/SortedCrimeArea';
```

Conclusion: 77th Street is the area where Crime happens most and Southwest, N Hollywood and Pacific follow by.

Use Pig to Analyze:
Crime dataset analysis

- Which kind of crime occurs most frequently each year top 10?

2016	510	18354	VEHICLE - STOLEN
2016	624	17944	BATTERY - SIMPLE ASSAULT
2016	330	16778	BURGLARY FROM VEHICLE
2016	440	14816	THEFT PLAIN - PETTY ($950 & UNDER)
2016	310	14558	BURGLARY
2016	354	14040	THEFT OF IDENTITY
2016	740	12812	VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)
2016	626	12405	INTIMATE PARTNER - SIMPLE ASSAULT
2016	230	10801	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
2016	420	10647	THEFT FROM MOTOR VEHICLE - PETTY ($950 & UNDER)
2017	624	19075	BATTERY - SIMPLE ASSAULT
2017	510	18758	VEHICLE - STOLEN
2017	330	18082	BURGLARY FROM VEHICLE
2017	310	15279	BURGLARY
2017	440	14772	THEFT PLAIN - PETTY ($950 & UNDER)
2017	354	13055	THEFT OF IDENTITY
2017	740	12974	VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)
2017	626	12602	INTIMATE PARTNER - SIMPLE ASSAULT
2017	230	10978	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
2017	420	10646	THEFT FROM MOTOR VEHICLE - PETTY ($950 & UNDER)
2018	624	19448	BATTERY - SIMPLE ASSAULT
2018	330	18067	BURGLARY FROM VEHICLE
2018	510	17134	VEHICLE - STOLEN
2018	440	15422	THEFT PLAIN - PETTY ($950 & UNDER)
2018	310	14817	BURGLARY
2018	740	12850	VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)
2018	626	12482	INTIMATE PARTNER - SIMPLE ASSAULT
2018	354	11562	THEFT OF IDENTITY
2018	230	10787	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
2018	420	10718	THEFT FROM MOTOR VEHICLE - PETTY ($950 & UNDER)
2019	624	4404	BATTERY - SIMPLE ASSAULT
2019	330	4179	BURGLARY FROM VEHICLE
2019	510	4025	VEHICLE - STOLEN
2019	440	3871	THEFT PLAIN - PETTY ($950 & UNDER)
2019	310	3491	BURGLARY
2019	740	3132	VANDALISM - FELONY ($400 & OVER, ALL CHURCH VANDALISMS)
2019	626	2831	INTIMATE PARTNER - SIMPLE ASSAULT
2019	354	2755	THEFT OF IDENTITY
2019	420	2583	THEFT FROM MOTOR VEHICLE - PETTY ($950 & UNDER)
2019	230	2413	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT

Use JOIN, Secondary Sorting and Limit.

```
crimeCodeDetail = LOAD '/FinalProject/DistinctCrime/part-r-00000' AS (
CrimeCode:long,Describe:chararray);
groupCrimeCode = GROUP otable BY (Year,CrimeCode);
crimeCount = FOREACH groupCrimeCode GENERATE group.Year,group.CrimeCode, COUNT(otable) AS sum;

joindata1 = JOIN crimeCount BY $1 LEFT OUTER, crimeCodeDetail BY $0;
joinresult = FOREACH joindata1 GENERATE $0,$1,$2,$4;

groupCountCrime = GROUP joinresult BY $0;
resultCrimeCode = FOREACH groupCountCrime {
    sorted = ORDER joinresult BY $2 DESC;
    lim = LIMIT sorted 10;
    GENERATE FLATTEN(lim);
}

STORE resultCrimeCode INTO '/FinalProject/PigOut/SortedCrimeCode';
```

Conclusion: VEHCLE – STOLEN, BATTERY – SIMPLE ASSULT and BURGLARY FROM VEHICLE are the top three occurring most frequently crime in 2016,2017,2018 and the first 4 month of 2019.

Use Pig to Analyze:
Crime dataset analysis

- Month ratio of Crime. (use COUNT,SUM,ROUND_TO,CONCAT)

```
01   76029    10.24%
02   68834    9.27%
03   75753    10.21%
04   57273    7.72%
05   59072    7.96%
06   57644    7.77%
07   59322    7.99%
08   59694    8.04%
09   56484    7.61%
10   59894    8.07%
11   55904    7.53%
12   56309    7.59%
```

```
groupMonth = Group otable BY (Month);
monthCount = FOREACH groupMonth GENERATE group, COUNT(otable) AS sum;
monthtemp = GROUP monthCount ALL;
monthsum = FOREACH monthtemp GENERATE SUM(monthCount.sum) AS total;
monthtemp2 = FOREACH monthCount GENERATE $0,$1,ROUND_TO((sum/(double)monthsum.total)*100,2) AS
perc;

result = FOREACH monthtemp2 GENERATE $0,$1,CONCAT((chararray)perc,'%');

STORE result INTO '/FinalProject/PigOut/CrimeMonthProportion';
```

Conclusion: Crime happens most on January, February and March. Cold winter has the least number of Crime.

Use Pig to Analyze:
Crime dataset analysis

- Proportion of different Victim age.

- The result is from -9 years old to 118 years old. More analysis of this topic will be discussed in Hive Part.

```
groupVicAge = Group otable BY (VictimAge);
ageCount = FOREACH groupVicAge GENERATE group, COUNT(otable) AS sum;
agetemp = GROUP ageCount ALL;
agesum = FOREACH agetemp GENERATE SUM(ageCount.sum) AS total;
agetemp2 = FOREACH ageCount GENERATE $0,$1,ROUND_TO((sum/(double)agesum.total)*100,4) AS perc;

result = FOREACH agetemp2 GENERATE $0,$1,CONCAT((chararray)perc,'%');

STORE result INTO '/FinalProject/PigOut/CrimeVicAgeProportion';
```

Use Pig to Analyze:
Crime dataset analysis

- portion of different Vitim gender.

```
F    302367   40.74%
H    26   0.0%
M    334783   45.11%
N    17   0.0%
X    32794    4.42%
     72225    9.73%
```

2016-2019

```
groupGender = Group otable BY (VictimSex);
genderCount = FOREACH groupGender GENERATE group, COUNT(otable) AS sum;
temp = GROUP genderCount ALL;
gendersum = FOREACH temp GENERATE SUM(genderCount.sum) AS total;
temp2 = FOREACH genderCount GENERATE $0, $1,ROUND_TO((sum/(double)
gendersum.total)*100,2) AS perc;
result = FOREACH temp2 GENERATE $0, $1,CONCAT((chararray)perc,'%');

STORE result INTO '/FinalProject/PigOut/CrimeVicGenderProportion';
```

```
F    258454   42.94%
H    26   0.0%
M    281482   46.77%
X    8398     1.4%
     53485    8.89%
```

2013-2015

Conclusion: Males and Females have almost the same possibility to become the victim of crime. The victim number of Crime is declining.

Use Pig to Analyze:

Crime dataset analysis

- Proportion of different Victim descent.

Descent Code: A - Other Asian B - Black C - Chinese D - Cambodian F - Filipino G - Guamanian H - Hispanic/Latin/Mexican I - American Indian/Alaskan Native J - Japanese K - Korean L - Laotian O - Other P - Pacific Islander S - Samoan U - Hawaiian V - Vietnamese W - White X - Unknown Z - Asian Indian

```
H    249863   33.6646%
W    170098   22.9177%
B    112407   15.1449%
O    72520    9.7708%
     72241    9.7332%
X    40965    5.5193%
A    19231    2.591%
K    2851     0.3841%
F    786 0.1059%
C    416 0.056%
I    364 0.049%
J    125 0.0168%
P    115 0.0155%
V    71   0.0096%
U    48   0.0065%
Z    47   0.0063%
G    36   0.0049%
S    13   0.0018%
D    8    0.0011%
L    5    7.0E-4%
-    2    3.0E-4%
```

```
groupDescent = Group otable BY (VictimDescent);
descentCount = FOREACH groupDescent GENERATE group, COUNT(otable) AS sum;
temp = GROUP descentCount ALL;
descentsum = FOREACH temp GENERATE SUM(descentCount.sum) AS total;
temp2 = FOREACH descentCount GENERATE $0, $1,ROUND_TO((sum/(double)
descentsum.total)*100,2) AS perc;
result = FOREACH temp2 GENERATE $0, $1,CONCAT((chararray)perc,'%');

STORE result INTO '/FinalProject/PigOut/CrimeVicDescentProportion';
```

Conclusion: Victim Descent Top 3:
Hispanic/Latin/Mexican
White
Black

Use Pig to Analyze:
Crime dataset analysis

- Proportion of different Weapons used in crime.

```
    490599  66.0996%
400 148937  20.0666%    STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
500 22293   3.0036% UNKNOWN WEAPON/OTHER WEAPON
511 19326   2.6038% VERBAL THREAT
102 12393   1.6697% HAND GUN
```

```
weapondetail = LOAD '/FinalProject/DistinctWeaponUsed/part-r-00000' AS (
WeaponUsed:long,WeaponDescribed:chararray);
groupWeapon = Group otable BY (WeaponUsed);
WeaponCount = FOREACH groupWeapon GENERATE group, COUNT(otable) AS sum;
Weapontemp = GROUP WeaponCount ALL;
Weaponsum = FOREACH Weapontemp GENERATE SUM(WeaponCount.sum) AS total;
Weapontemp2 = FOREACH WeaponCount GENERATE $0,$1,ROUND_TO((sum/(double)
Weaponsum.total)*100,4) AS perc;
result = FOREACH Weapontemp2 GENERATE $0,$1,CONCAT((chararray)perc,'%');

joindata1 = JOIN result BY $0 LEFT OUTER, weapondetail BY $0;
joindata = FOREACH joindata1 GENERATE $0,$1,$2,$4;

sorted = ORDER joindata BY $1 DESC;
STORE sorted INTO '/FinalProject/PigOut/CrimeWeaponProportion';
```

Conclusion: Most of crimes reported do not have weapons. STRONG-ARM is the weapon being used most in crime.

Use Pig to Analyze:

Crime dataset analysis

- Proportion of different Status of crime.

```
IC  575939  77.5976%    Invest Cont
AO  85633   11.5375%    Adult Other
AA  73770   9.9392% Adult Arrest
JA  5048    0.6801% Juv Arrest
JO  1815    0.2445% Juv Other
CC  5   7.0E-4% UNK
    1   1.0E-4%
19  1   1.0E-4% UNK
```

2016-2019

```
IC  449765  74.731% Invest Cont
AO  76912   12.7794%    Adult Other
AA  67459   11.2087%    Adult Arrest
JA  5864    0.9743% Juv Arrest
JO  1829    0.3039% Juv Other
CC  14  0.0023% UNK
    1   2.0E-4%
13  1   2.0E-4% UNK
```

2013-2015

```
statusdetail = LOAD '/FinalProject/DistinctStatus/part-r-00000' AS (
StatusCode:chararray,StatusDescribed:chararray);
groupStatus = Group otable BY (StatusCode);
StatusCount = FOREACH groupStatus GENERATE group, COUNT(otable) AS sum;
Statustemp = GROUP StatusCount ALL;
Statussum = FOREACH Statustemp GENERATE SUM(StatusCount.sum) AS total;
Statustemp2 = FOREACH StatusCount GENERATE $0,$1,ROUND_TO((sum/(double)
Statussum.total)*100,4) AS perc;
result = FOREACH Statustemp2 GENERATE $0,$1,CONCAT((chararray)perc,'%');

joindata1 = JOIN result BY $0 LEFT OUTER, statusdetail BY $0;
joindata = FOREACH joindata1 GENERATE $0,$1,$2,$4;

sorted = ORDER joindata BY $1 DESC;
STORE sorted INTO '/FinalProject/PigOut/CrimeStatusProportion';
```

Conclusion: Most Crime being reported even several years ago are under Investigation.

Use Hive to Easy Query:

- 1. Create Table
- 2. Load Data
- 3. Test query.

Use Hive to Easy Query:

- Create Table

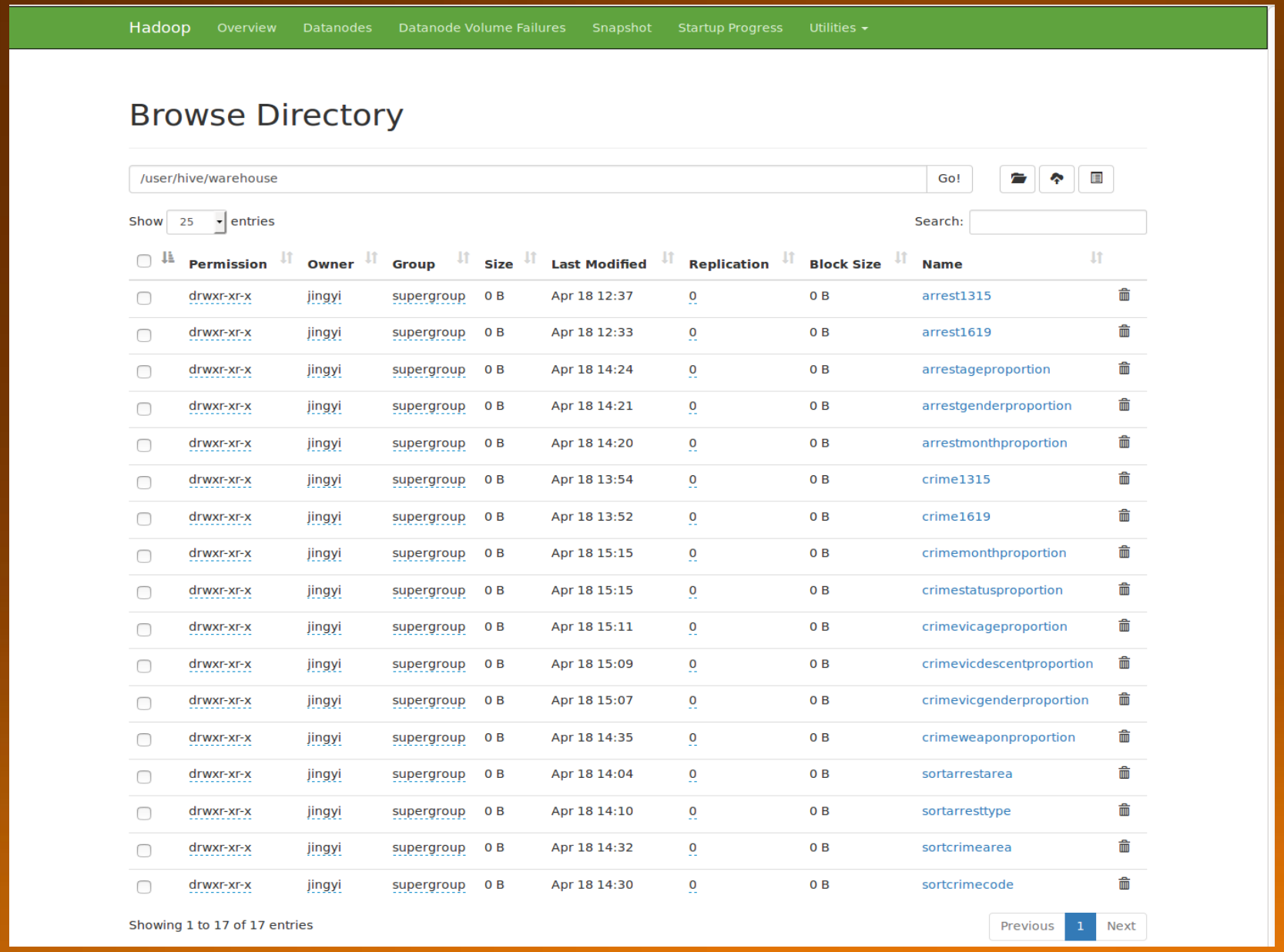
```
CREATE TABLE arrest1619 (ReportID INT,ArrestDate STRING,Year STRING,Month STRING,
AreaID INT,ReportingDistrict INT,Age INT,SexCode STRING,DescentCode STRING,
ChargeGroupCode STRING,ArrestTypeCode STRING,Charge STRING,Address STRING,Location STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ";"
STORED AS TEXTFILE;
```

- Load Data

```
LOAD DATA INPATH "/FinalProject/PartiArrest/part-r-00002" INTO TABLE arrest1619;
```

Use Hive to Easy Query:

```
hive> show tables;
OK
arrest1315
arrest1619
arrestageproportion
arrestgenderproportion
arrestmonthproportion
crime1315
crime1619
crimemonthproportion
crimestatusproportion
crimevicageproportion
crimevicdescentproportion
crimevicgenderproportion
crimeweaponproportion
sortarrestarea
sortarresttype
sortcrimearea
sortcrimecode
Time taken: 0.07 seconds, Fetched: 17 row(s)
```

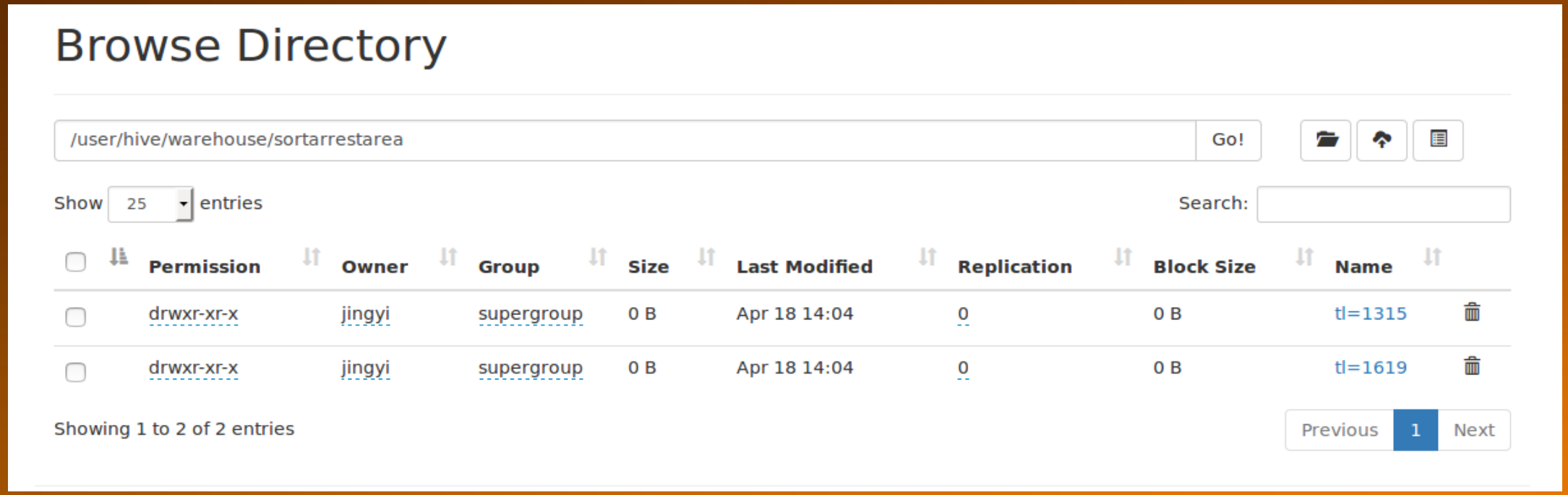


Use Hive to Easy Query:

- Create Table and definite partition.

```
CREATE TABLE sortArrestArea (AreaID INT,AreaCount INT,Name STRING)
PARTITIONED BY (tl STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY "\t";
```

```
LOAD DATA INPATH "/FinalProject/PigOut/SortedArrestArea/part-r-00000" INTO TABLE sortArrestArea PARTITION (tl="1619");
LOAD DATA INPATH "/FinalProject/PigOut/SortedArrestArea1315/part-r-00000" INTO TABLE sortArrestArea PARTITION (tl="1315");
```



Use Hive to Easy Query:

- Query:

```
SELECT * FROM CrimeVicAgeProportion WHERE Age <= 14;
```

0	98432	16.355%	1315
2	405	0.0673%	1315
3	555	0.0922%	1315
4	587	0.0975%	1315
5	701	0.1165%	1315
6	709	0.1178%	1315
7	709	0.1178%	1315
8	731	0.1215%	1315
9	774	0.1286%	1315
10	946	0.1572%	1315
11	1295	0.2152%	1315
12	2362	0.3925%	1315
13	3179	0.5282%	1315
14	4059	0.6744%	1315
-4	3	5.0E-4%	1315
-3	20	0.0033%	1315
-2	37	0.0061%	1315
-1	85	0.0141%	1315

0	139238	18.7599%	1619
2	391	0.0527%	1619
3	462	0.0622%	1619
4	598	0.0806%	1619
5	722	0.0973%	1619
6	710	0.0957%	1619
7	804	0.1083%	1619
8	810	0.1091%	1619
9	967	0.1303%	1619
10	1038	0.1399%	1619
11	1454	0.1959%	1619
12	2050	0.2762%	1619
13	2602	0.3506%	1619
14	3237	0.4361%	1619
-9	1	1.0E-4%	1619
-8	1	1.0E-4%	1619
-7	4	5.0E-4%	1619
-6	10	0.0013%	1619
-5	17	0.0023%	1619
-4	18	0.0024%	1619
-3	31	0.0042%	1619
-2	61	0.0082%	1619
-1	108	0.0146%	1619

Time taken: 6.885 seconds, Fetched: 41 row(s)

Conclusion: Children under 14 years old are easy to be the victims of crime. More than 15 % victims are infants.

Use Hive to Easy Query:

- Query:

```
SELECT * FROM ArrestAgeProportion WHERE Age <= 14 AND tl="1619";
```

```
0       155     0.0446% 1619
1       93      0.0268% 1619
2       114     0.0328% 1619
3       111     0.032%  1619
4       95      0.0274% 1619
5       104     0.03%   1619
6       79      0.0228% 1619
7       101     0.0291% 1619
8       74      0.0213% 1619
9       95      0.0274% 1619
10      109     0.0314% 1619
11      152     0.0438% 1619
12      427     0.123%  1619
13      925     0.2664% 1619
14      1606    0.4626% 1619
Time taken: 0.48 seconds, Fetched: 15 row(s)
```

There are also lots of children under 14 were arrested during the past several years.

Use Hive to Easy Query:

- Query:

```
SELECT Age, SexCode,DescentCode,ArrestTypeCode FROM arrest1619 WHERE Age == 0;
```

0	F	H	O
0	M	H	D
0	M	B	D
0	M	B	D
0	M	H	D
0	F	H	D
0	F	H	D
0	F	O	O
0	F	O	O
0	F	H	D
0	M	B	D
0	M	H	D
0	F	O	D
0	M	H	O

Most of the infants being arrested because of their parents are being arrested. (ArrestTypeCode D : dependent)

Use Mahout to Clustering:

- Clustering analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups(clusters).

- K-means is a simple clustering algorithms

Use Mahout to Clustering:

- Data: Weapons being used in crimes.

```
jingyi@ubuntu:~/Desktop/Finalproject/test/MahoutData$ split -l 5 /home/jingyi/De
sktop/Finalproject/test/part-r-00000
```

```
jingyi@ubuntu:~$ hadoop fs -copyFromLocal -f /home/jingyi/Desktop/Finalproject/t
est/MahoutData/* /Mahout/
```

```
-rw-r--r--   1 jingyi supergroup        111 2019-04-21 12:35 /Mahout/xaa
-rw-r--r--   1 jingyi supergroup         68 2019-04-21 12:35 /Mahout/xab
-rw-r--r--   1 jingyi supergroup         60 2019-04-21 12:35 /Mahout/xac
-rw-r--r--   1 jingyi supergroup         74 2019-04-21 12:35 /Mahout/xad
-rw-r--r--   1 jingyi supergroup         74 2019-04-21 12:35 /Mahout/xae
-rw-r--r--   1 jingyi supergroup         82 2019-04-21 12:35 /Mahout/xaf
-rw-r--r--   1 jingyi supergroup        143 2019-04-21 12:35 /Mahout/xag
-rw-r--r--   1 jingyi supergroup        151 2019-04-21 12:35 /Mahout/xah
-rw-r--r--   1 jingyi supergroup         86 2019-04-21 12:35 /Mahout/xai
-rw-r--r--   1 jingyi supergroup         64 2019-04-21 12:35 /Mahout/xaj
-rw-r--r--   1 jingyi supergroup         62 2019-04-21 12:35 /Mahout/xak
-rw-r--r--   1 jingyi supergroup         83 2019-04-21 12:35 /Mahout/xal
-rw-r--r--   1 jingyi supergroup         73 2019-04-21 12:35 /Mahout/xam
-rw-r--r--   1 jingyi supergroup         90 2019-04-21 12:35 /Mahout/xan
-rw-r--r--   1 jingyi supergroup         79 2019-04-21 12:35 /Mahout/xao
-rw-r--r--   1 jingyi supergroup        114 2019-04-21 12:35 /Mahout/xap
```

Use Mahout to Clustering:

- Then convert data into sequence file

```
jingyi@ubuntu:~$ mahout seqdirectory -i /Mahout/ -o /Mahout/KmeansSeqFile -ow
```

- Next convert sequence file to TF-IDF vector.

```
jingyi@ubuntu:~$ mahout seq2sparse -i /Mahout/KmeansSeqFile -o /Mahout/KmeansVector -ow
```

- Kmean clustering

```
jingyi@ubuntu:~$ mahout kmeans -i /Mahout/KmeansVector/tfidf-vectors/part-r-0000
0 -c /Mahout/kmeanscentroids -cl -o /Mahout/kmeansclusters -k 4 -ow -x 50 -dm or
g.apache.mahout.common.distance.CosineDistanceMeasure
```

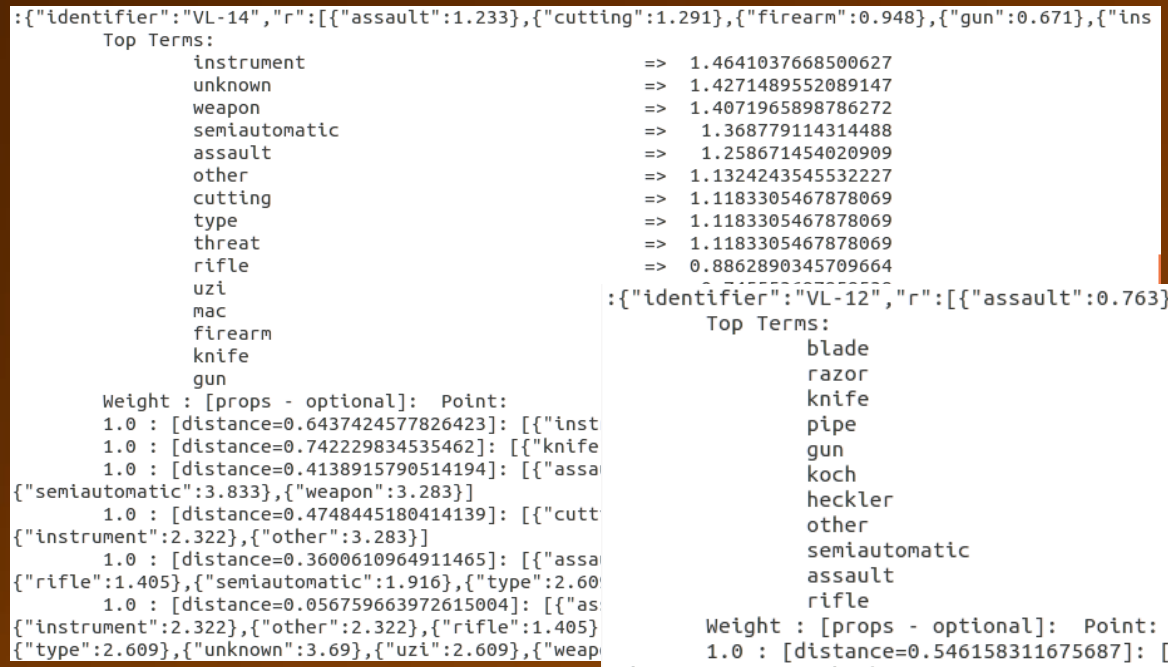
Use Mahout to Clustering:

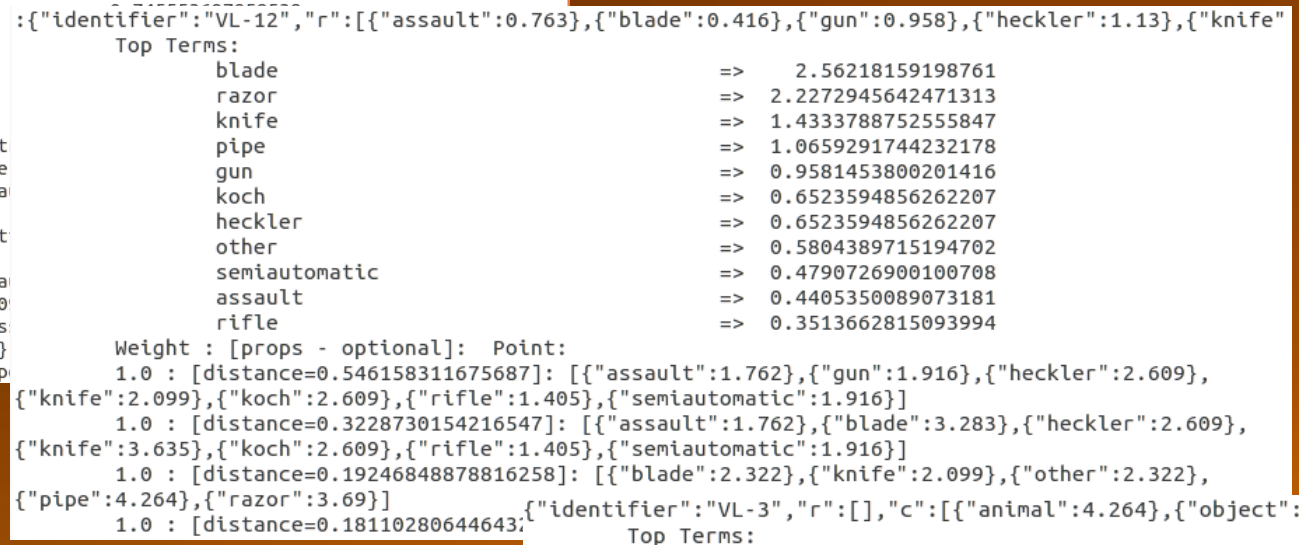
- Result:
- Dump the clusters created into a text file (local file).

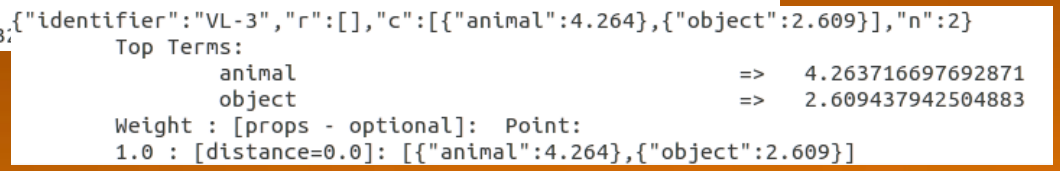
```
jingyi@ubuntu:~$ mahout clusterdump -d /Mahout/KmeansVector/dictionary.file-0 -dt sequencefile -i /Mahou
t/kmeansclusters/clusters-1-final -n 20 -b 100 -o /Mahout/dumpfile.txt -p /Mahout/kmeansclusters/cluster
edPoints/
```


```
:{"identifier":"VL-0","r":[{"assault":0.881},{"automatic":1.094},{"firearm":1.049},{"gun":1.138},{"he
        Top Terms:
                automatic                               =>   1.547837257385254
                pistol                                  =>   1.547837257385254
                revolver                                =>   1.547837257385254
                gun                                     =>  1.5421117146809895
                rifle                                   =>  1.4054651260375977
                weapon                                  =>  1.0944862365722656
                firearm                                 =>  1.0493061542510986
                assault                                 =>  0.8810700178146362
                uzi                                     =>  0.8698126475016276
                semi                                    =>  0.8698126475016276
                shotgun                                 =>  0.8698126475016276
                object                                  =>  0.4349063237508138
                koch                                    =>  0.4349063237508138
                heckler                                 =>  0.4349063237508138
                knife                                   =>  0.3497687180836995
                semiautomatic                           =>  0.3193817933400472
        Weight : [props - optional]:  Point:
        1.0 : [distance=0.11207220839051213]: [{"assault":1.762},{"automatic":2.322},{"firearm":2.099},{"gun":2.71},{"pistol":2.322},{"revolver":2.322},
{"rifle":1.405},{"uzi":2.609},{"weapon":3.283}]
        1.0 : [distance=0.5085755666187366]: [{"firearm":2.099},{"object":2.609},{"revolver":2.322},{"rifle":1.405}]
        1.0 : [distance=0.365423405896272]: [{"automatic":2.322},{"pistol":2.322},{"rifle":1.405},{"semi":2.609},{"shotgun":2.609}]
        1.0 : [distance=0.19089655326791843]: [{"automatic":2.322},{"gun":1.916},{"pistol":2.322},{"revolver":2.322},{"rifle":1.405},{"semi":2.609},
{"shotgun":2.609}]
```

Use Mahout to Clustering:







Thank you