**ETL for ARCOS buyers of pain pills’ Components and their transactions in USA**

**The goal is to demonstrate the details of top 100 buyers of one of the main pain pills components and subsequently, the amounts and transaction dates of top 2 purchases per each top 100 buyers**

**Data set source URL:**

<https://www.kaggle.com/paultimothymooney/pain-pills-in-the-usa>

Data size : 78GB

**Extracting the first 12 GB of the data set to upload it to Oracle cloud:**

head -n 25500000 arcos\_all\_washpost.tsv > arcos\_all\_washpost1.tsv

scp arcos\_all\_washpost1.tsv fmamagh2@129.150.69.91:/dev/shm/

**Downloading Pain pills dataset from Oracle cloud to Hadoop file system**

**Connect to Oracle Cloud:**

ssh fmamagh2@129.150.69.91

**Create a folder in hdfs to store the dataset:**

hdfs dfs –mkdir Group5

hdfs dfs –get /dev/shm/arcos\_all\_washpost1.tsv Group5/

hdfs dfs -ls Group5

Found 1 items

-rw-r--rw- 2 fmamagh2 hdfs 11432128252 2020-12-01 04:25 Group5/arcos\_all\_washpost1.tsv

**Remove pain pills dataset from oracle cloud storage:**

rm /dev/shm/arcos\_all\_washpost1.tsv

ls dev/shm/

**Creating Hive Table to Query Pain Pills Data**

**Give the permission to beeline for edit files/data:**

hdfs dfs -chmod -R o+w Group5/

**Connect to beeline/Hive:**

beeline

!connect jdbc:hive2://bigdai-nov-bdcsce-1:2181,bigdai-nov-bdcsce-2:2181,bigdai-nov-bdcsce-3:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2?tez.queue.name=interactive  bdcsce\_admin

**Change the database to your username:**

use fmamagh2;

**Create main table which will include everything from the data:**

|  |
| --- |
| CREATE TABLE if not exists pain\_pills ( |
| REPORTER\_DEA\_NO STRING , |
| REPORTER\_BUS\_ACT STRING , |
| REPORTER\_NAME STRING , |
| REPORTER\_ADDL\_CO\_INFO STRING , |
| REPORTER\_ADDRESS1 STRING , |
| REPORTER\_ADDRESS2 STRING , |
| REPORTER\_CITY STRING , |
| REPORTER\_STATE STRING , |
| REPORTER\_ZIP STRING , |
| REPORTER\_COUNTY STRING , |
| BUYER\_DEA\_NO STRING , |
| BUYER\_BUS\_ACT STRING , |
| BUYER\_NAME STRING , |
| BUYER\_ADDL\_CO\_INFO STRING , |
| BUYER\_ADDRESS1 STRING , |
| BUYER\_ADDRESS2 STRING , |
| BUYER\_CITY STRING , |
| BUYER\_STATE STRING , |
| BUYER\_ZIP STRING , |
| BUYER\_COUNTY STRING , |
| TRANSACTION\_CODE STRING , |
| DRUG\_CODE INT , |
| NDC\_NO BIGINT , |
| DRUG\_NAME STRING , |
| QUANTITY INT , |
| UNIT INT , |
| ACTION\_INDICATOR INT , |
| ORDER\_FORM\_NO INT , |
| CORRECTION\_NO INT , |
| STRENGTH INT , |
| TRANSACTION\_DATE STRING , |
| CALC\_BASE\_WT\_IN\_GM FLOAT , |
| DOSAGE\_UNIT INT , |
| TRANSACTION\_ID INT , |
| Product\_Name STRING , |
| Ingredient\_Name STRING , |
| Measure STRING , |
| MME\_Conversion\_Factor INT , |
| Combined\_Labeler\_Name STRING , |
| Revised\_Company\_Name STRING , |
| Reporter\_family STRING , |
| dos\_str INT) |
| ROW FORMAT DELIMITED |
| FIELDS TERMINATED BY '\t' |
| STORED AS TEXTFILE |
| LOCATION "/user/fmamagh2/Group5" |
| TBLPROPERTIES ('skip.header.line.count'='1'); |

Note: The yellow highlights are fields that will be selected for further queries in Pig.

**Use describe and select from function to check the table is created correct:**

describe pain\_pills;

select \* from pain\_pills limit 3;

**Connecting to pig interactive mode using Grunt Shell and utilizing HCatalog to share data between Hive and Pig**

**Open a new terminal and ssh to the Hadoop cluster**

**Connect to pig and load data from hive table:**

data = LOAD 'fmamagh2.pain\_pills' using org.apache.hive.hcatalog.pig.HCatLoader();

DESCRIBE data;

**Extract the required columns for ETL processing:**

Buyers = foreach data generate buyer\_dea\_no, buyer\_bus\_act, buyer\_name, buyer\_address1, buyer\_address2, buyer\_city, buyer\_state, buyer\_zip, buyer\_county, drug\_code, drug\_name, quantity, transaction\_date;

describe Buyers;

Buyers: {buyer\_dea\_no: chararray,buyer\_bus\_act: chararray,buyer\_name: chararray,buyer\_address1: chararray,buyer\_address2: chararray,buyer\_city: chararray,buyer\_state: chararray,buyer\_zip: chararray,buyer\_county: chararray,drug\_code: int,drug\_name: chararray,quantity: int,transaction\_date: chararray}

**Categorize the drug names:**

c = foreach Buyers generate drug\_code, drug\_name;

Drugname = DISTINCT c;

dump Drugname;

(9143,OXYCODONE)

(,DRUG\_NAME)

(9193,HYDROCODONE)

**Split the Buyers based on their drug category purchase:**

SPLIT Buyers INTO B\_OXY IF drug\_name == 'OXYCODONE', B\_HYDRO IF drug\_name == 'HYDROCODONE' ;

(Optional) **Shorten the names of the buyers to the first 10 characters in the names:**

boN = foreach B\_OXY generate buyer\_dea\_no, SUBSTRING(buyer\_name, 0, 11) AS buyer\_name, drug\_name;

boName = DISTINCT boN;

**Extracting the top 100 Buyers of Oxycodone**

bo1 = foreach B\_OXY generate buyer\_dea\_no, drug\_name, quantity;

bo2 = group bo1 by buyer\_dea\_no;

describe bo2;

bo\_total = foreach bo2 generate group, SUM(bo1.quantity) As total;

bo\_sort = ORDER bo\_total BY total desc;

bo100 = limit bo\_sort 100;

-- The reason for this join is to add shortened buyer\_name

oxy\_total1 = JOIN bo100 BY group, boName BY buyer\_dea\_no;

oxy\_total2 = foreach oxy\_total1 generate boName::buyer\_dea\_no AS buyer\_dea\_no , boName::buyer\_name AS buyer\_name , boName::drug\_name AS drug\_name , bo100::total AS total;

oxy\_total100 = ORDER oxy\_total2 BY total DESC;

**Store the results into HDSF:**

STORE oxy\_total100 INTO 'Buyers/oxytotal';

**List of top 2 purchases per each buyer in top 100**

Buyer\_oxy1 = group B\_OXY BY buyer\_dea\_no;

Buyer\_oxy2 = foreach Buyer\_oxy1{

sorted = ORDER B\_OXY BY quantity DESC;

high\_qty = limit sorted 2;

generate group, FLATTEN(high\_qty);};

Buyer\_oxy3 = JOIN oxy\_total100 BY buyer\_dea\_no , Buyer\_oxy2 BY group;

Buyers\_Detail1 = foreach Buyer\_oxy3 generate oxy\_total100::buyer\_dea\_no AS buyer\_dea\_no, Buyer\_oxy2::high\_qty::buyer\_bus\_act AS buyer\_bus\_act, oxy\_total100::buyer\_name AS buyer\_name, Buyer\_oxy2::high\_qty::buyer\_name AS buyer\_name\_dtl, Buyer\_oxy2::high\_qty::buyer\_address1 AS buyer\_address1,Buyer\_oxy2::high\_qty::buyer\_address2 As buyer\_address2, Buyer\_oxy2::high\_qty::buyer\_city AS buyer\_city,Buyer\_oxy2::high\_qty::buyer\_state AS buyer\_state,Buyer\_oxy2::high\_qty::buyer\_zip As buyer\_zip ,Buyer\_oxy2::high\_qty::buyer\_county AS buyer\_county, Buyer\_oxy2::high\_qty::drug\_name AS drug\_name ,Buyer\_oxy2::high\_qty::quantity AS quantity, Buyer\_oxy2::high\_qty::transaction\_date AS Date;

Describe Buyers\_Detail1;

**Prepare the date format for 3D map in excel**

Buyers\_Detail = foreach Buyers\_Detail1 generate buyer\_dea\_no, buyer\_bus\_act, buyer\_name, buyer\_name\_dtl, buyer\_address1, buyer\_address2, buyer\_city, buyer\_state, buyer\_zip, buyer\_county, drug\_name, quantity, CONCAT(SUBSTRING(Date,0,2),'/',SUBSTRING(Date,2,4),'/',SUBSTRING(Date,4,9)) AS Date;

describe Buyers\_Detail;

**Store the results into HDSF:**

STORE Buyers\_Detail INTO 'Buyers/oxydetail';

**(Optional) change the name of the stored files to prevent overwriting the file in get process**

hdfs dfs -mv Buyers/oxydetail/part-r-00000 Buyers/oxydetail/oxydetail

hdfs dfs -mv Buyers/oxytotal/part-r-00000 Buyers/oxytotal/oxytotal

**Download the data into your PC:**

hdfs dfs -get Buyers/oxytotal/oxytotal

hdfs dfs -get Buyers/oxydetail/oxydetail

scp fmamagh2@129.150.69.91:/home/fmamagh2/oxytotal oxytotal.tsv

scp [fmamagh2@129.150.69.91:/home/fmamagh2/oxydetail oxydetail.tsv](mailto:fmamagh2@129.150.69.91:/home/fmamagh2/oxydetail%20oxydetail.tsv)

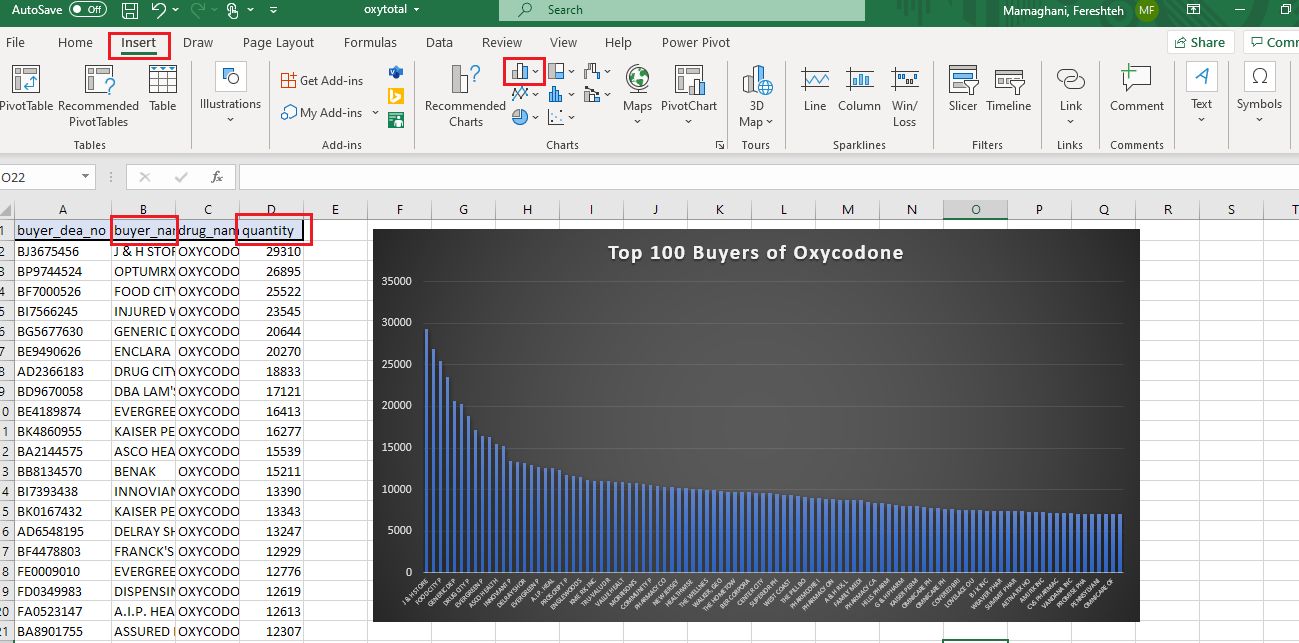
**Loading Data into and Visualizing using Power Map in Excel**

Open the oxydetail and oxytotal files in excel and separate the columns as Tab delimited

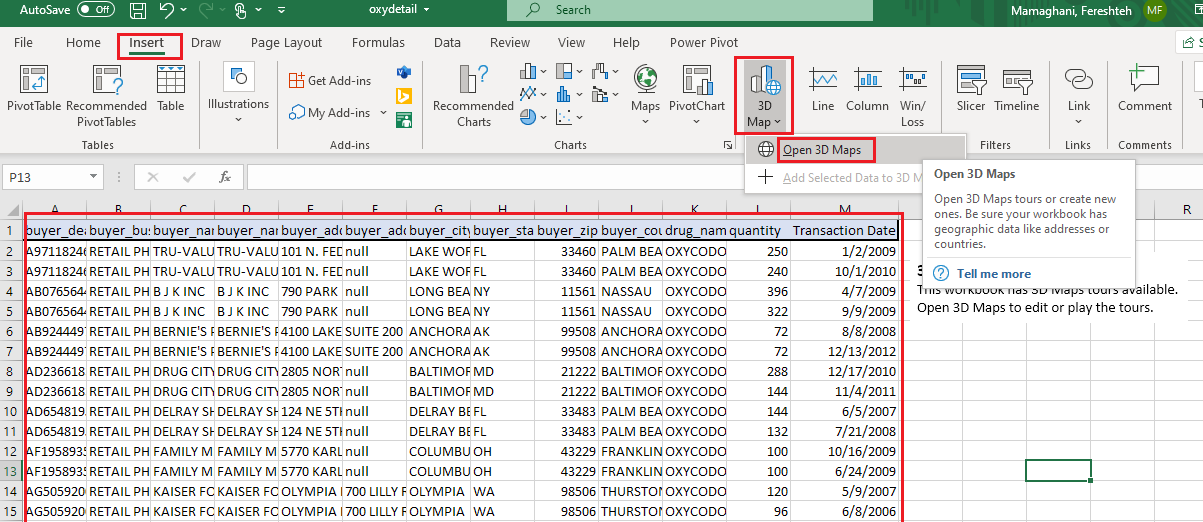
Add the header names to columns of each file oxy detail and oxytotal based on Buyers\_Detail and oxy\_total100 headers, repectively.

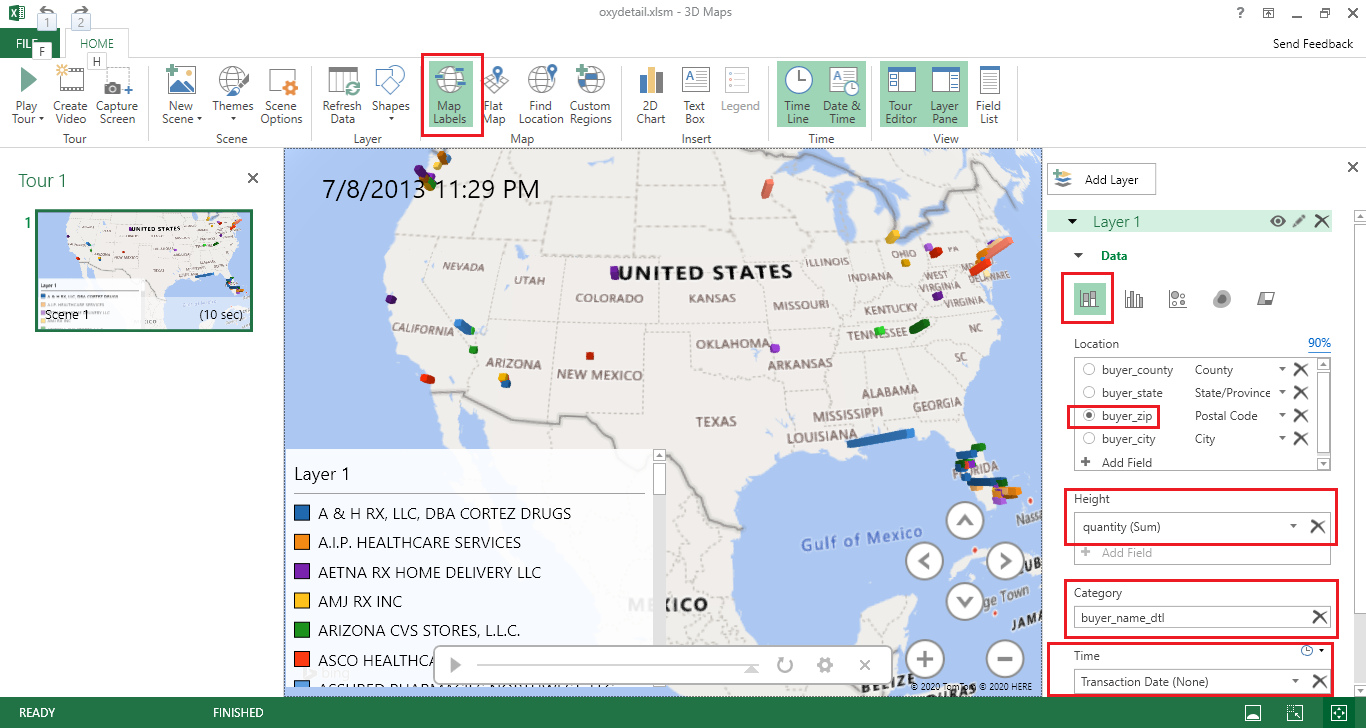
Use the Pig Latins describe Buyers\_Detail; and describe oxy\_total100; for the header names.

**2-D Bar chart for top 100 Buyers:**

****

**3-D MAP for Buyers details and the amount of their 2 transactions:**

****

****