**PATH ANALAYSIS USING APACHE PIG**

**/\* CREATE A TEXT FILES IN YOUR LOCAL MACHINES AND**

**PROVIDE SOME VALUES TO IT \*/**

$ nano 201904sales\_reciepts.txt

$nano customer.txt

$nano product.txt

$nano staff.txt

$nano generations.txt

$nano sales\_outlet.txt

$nano sales\_targets.txt

**-- UPLOADING TEXT FILES ON HDFS IN PigDirectory.**

$hdfs dfs -put 201904sales\_reciepts.txt / PigDirectory

$hdfs dfs -put customer.txt / PigDirectory

$hdfs dfs -put product.txt / PigDirectory

$hdfs dfs -put staff.txt / PigDirectory

$hdfs dfs -put generations.txt / PigDirectory

$hdfs dfs -put sales\_outlet.txt / PigDirectory

$hdfs dfs -put sales\_targets.txt / PigDirectory

**--OPEN PIG MapReduce RUN MODE**

$ pig

**/\* WHEN TERMINAL OPENS, LOAD THE**

**FILES THAT CONTAINS THE DATA \*/**

grunt> A=LOAD ‘/PigDirectory/201904sales\_reciepts.txt’ USING PigStorage(‘,’) AS (transaction\_id:int , transaction\_date: Datetime, transaction\_time: Datetime, sales\_outlet\_id: int , staff\_id:int, customer\_id:int, instore\_yn:chararray, order:int, line\_item\_id:int,product\_id:int,quantity:int,line\_item\_amount:float,unit\_price:float,promo\_item\_yn:chararray);

grunt> B=LOAD ‘/PigDirectory/customer.txt’ USING PigStorage(‘,’) AS (customer\_id:int, home\_store:int, customer\_first-name:chararray, customer\_email:chararray, loyalty\_card\_number:long, birthdate:Datetime, gender:chararray, birth\_year:int);

grunt> C=LOAD ‘/PigDirectory/product.txt’ USING PigStorage (‘,’) AS (product\_id:int,product\_group:chararray,product\_category:chararray,product\_type:chararray,product:chararray,product\_description:chararray,unit\_of\_measure:float,current\_wholesale\_price:float, current\_retail\_price:float,tax\_exempt\_yn:chararray,promoyn:chararray,new\_product\_yn:chararray);

grunt> D=LOAD ‘/PigDirectory/staff.txt’ USING PigStorage (‘,’) AS (staff\_id:int, first\_name:chararray,last\_name:chararray,position:chararray,start\_date:Datetime,location:chararray);

grunt> E=LOAD ‘/PigDirectory/generations.txt’ USING PigStorage (‘,’) AS (birth\_year:int, generation:chararray);

grunt> F=LOAD ‘/PigDirectory/sales\_outlet.txt’ USING PigStorage (‘,’) AS (sales\_outlet\_id : int, sales\_outlet\_type:chararray,store\_square\_feet:int,store\_address:chararray,store\_city:chararray,store\_telephone:long,store\_postal\_code:int,store\_longitude:double,store\_latitude:double,manger:int,Neighorhood:chararray);

grunt> G=LOAD ‘/PigDirectory/sales\_targets.txt’ USING PigStorage (‘,’) AS (sales\_outlet\_id:int, year\_month:int,beans\_goal:int,beverage\_goal:int,food\_goal:int,merchandise\_goal:int,total\_goal:int);

**--TO CHECK THE CORRESPONDING SCHEMA**

grunt> DESCRIBE customer.txt;

grunt> DESCRIBE customer.txt;

B: {customer\_id:int, home\_store:int, customer\_first-name:chararray, customer\_email:chararray, loyalty\_card\_number:long, birthdate:Datetime, gender:chararray, birth\_year:int}

grunt>

**/\* TO RETRIEVE birth\_year AND customer\_id OF A PERSONS WHO PURCHASED**

**PRODUCT WITH ID 112 AND BIRTH\_YEAR IS 1969 \*/**

grunt> outer\_join\_for\_ 201904sales\_receipts \_customer = JOIN A BY customer\_id FULL OUTER, B BY customer\_id;

/\* HERE A IS 201904sales\_reciepts.txt AND B IS customer.txt

grunt> DUMP outer\_join\_for\_ 201904sales\_receipts \_customer;

grunt> filter\_data= FILTER by outer\_join\_for\_ 201904sales\_receipts \_customer BY ( birth\_year = 1969 and product\_id=112);

grunt>perform\_group\_operation= GROUP filter\_data BY customer\_id;

grunt> spe\_oper= FOREACH perform\_group\_operation GENERATE (customer\_id,customer\_email,gender,transaction\_date,quantity);

grunt> DUMP spe\_oper; -- IT OUTPUT THE ABOVE MENTIONED

**-- TO RETRIEVE SUM(total\_goals) FROM SALES\_TARGETS.TXT ON 04-19-2021**

grunt> grouped\_data=GROUP G BY year\_month;

grunt>Res=FOREACH grouped\_data GENERATE G.year\_month , SUM(G.total\_goal), AVG(G.total\_goal);

grunt>DUMP(Res);

**-- TO RETRIEVE “Baby Boommers” generation customer ID, Email**

grunt>outer\_join\_for\_generations\_customer = JOIN B BY birth\_year FULL OUTER, E BY birth\_year;

--B IS customer.txt AND E IS generation.txt

grunt>SPLIT outer\_join\_for\_generations\_customer INTO splitted\_outer\_join\_for\_generations\_customer if generation=”Baby Boomers”;

grunt>Result=FOREACH splitted\_outer\_join\_for\_generations\_customer GENERATE outer\_join\_for\_generations\_customer.customer\_id , outer\_join\_for\_generations\_customer.customer\_email;

grunt>DUMP Result;

**/\* TO RETRIEVE THE sales\_outlet\_type, store\_city and Neighbourhood**

**for Max and Min count of Manager field FROM SALES\_OUTLET.txt \*/**

grunt>grouped\_sales\_outlet\_data = GROUP F BY sales\_outlet\_type;

grunt>Result=FOREACH MAX(F.manager) GENERATE group,store\_city,Neighbourhood;

grunt>DUMP(Result);

grunt> Re=FOREACH MIN(F.manager) GENERATE group,store\_city,Neighbourhood;

grunt>DUMP(Re);

/**\* TO RETRIEVE AVG(CURRENT\_WHOLE\_SALE\_PRICE)**

**FOR ALL PRODUCT\_CATEGORIES \*/**

grunt> grouped\_data= GROUP C BY product\_category;

grunt>RES= FOREACH grouped\_data GENERATE C.product\_category, AVG(C. current\_wholesale\_price),C.product;

grunt>DUMP Res;

**/\* TO RETRIEVE store\_city ,store\_square\_feet AND**

**store\_telephone FOR MAX store\_square\_feet \*/**

grunt>ordered\_data=ORDER F BY store\_square\_feet DESC;

grunt>grouped\_data=GROUP ordered\_data BY sales\_outlet\_type;

grunt>for\_all\_data=FOREACH grouped\_data GENERATE F.store\_city, F.store\_telephone,F.store\_square\_feet;

grunt>Res=LIMIT for\_all\_data BY 1;

grunt>DUMP(Res);

**-- TO RETRIEVE ID AND EMAIL FOR A CUSTOMER NAMED “scott” FROM cutomer.txt**

grunt>data=FOREACH B GENERATE (customer\_id, customer\_email), STARTSWITH(customer\_First-name,”Scott”);

grunt>DUMP data;

**-- EDITING ALL CUSTOMER\_NAMES TO STARTS WITH UPPER CASE LETTER**

grunt> res=FOREACH B GENERATE (customer\_id,customer\_name) , UCFIRST(customer\_name);

grunt>DUMP res;

-- **TO EXIT FROM GRUNT SHELL**

Ctrl+d