WHAT IS BIG DATA ?

Big data is a term that describes the large volume of data – both structured and unstructured – that inundates a business on a day-to-day basis. But it’s not the amount of data that’s important. It’s what organizations do with the data that matters. Big data can be analyzed for insights that lead to better decisions and strategic business moves.

What is history of Big Data?

While the term “big data” is relatively new, the act of gathering and storing large amounts of information for eventual analysis is ages old. The concept gained momentum in the early 2000s when industry analyst Doug Laney articulated the now-mainstream definition of big data as the three Vs:

**Volume:** Organizations collect data from a variety of sources, including business transactions, social media and information from sensor or machine-to-machine data. In the past, storing it would’ve been a problem – but new technologies (such as Hadoop) have eased the burden.

**Velocity.** Data streams in at an unprecedented speed and must be dealt with in a timely manner. RFID tags, sensors and smart metering are driving the need to deal with torrents of data in near-real time.

**Variety.** Data comes in all types of formats – from structured, numeric data in traditional databases to unstructured text documents, email, video, audio, stock ticker data and financial transactions

At SAS, we consider two additional dimensions when it comes to big data:

**Variability.** In addition to the increasing velocities and varieties of data, data flows can be highly inconsistent with periodic peaks. Is something trending in social media? Daily, seasonal and event-triggered peak data loads can be challenging to manage. Even more so with unstructured data.

**Complexity.** Today's data comes from multiple sources, which makes it difficult to link, match, cleanse and transform data across systems. However, it’s necessary to connect and correlate relationships, hierarchies and multiple data linkages or your data can quickly spiral out of control.

**Why Is Big Data Important?**

The importance of big data doesn’t revolve around how much data you have, but what you do with it. You can take data from any source and analyze it to find answers that enable 1) cost reductions, 2) time reductions, 3) new product development and optimized offerings, and 4) smart decision making. When you combine big data with high-powered analytics, you can accomplish business-related tasks such as:

* Determining root causes of failures, issues and defects in near-real time.
* Generating coupons at the point of sale based on the customer’s buying habits.
* Recalculating entire risk portfolios in minutes.
* Detecting fraudulent behavior before it affects your organization.

**Big data analytics technologies and tools**

Unstructured and semi-structured data types typically don't fit well in traditional [data warehouses](https://searchsqlserver.techtarget.com/definition/data-warehouse) that are based on [relational databases](https://searchsqlserver.techtarget.com/definition/relational-database) oriented to structured data sets. Furthermore, data warehouses may not be able to handle the processing demands posed by sets of big data that need to be updated frequently -- or even continually, as in the case of real-time data on stock trading, the online activities of website visitors or the performance of mobile applications.

As a result, many organizations that collect, process and analyze big data turn to [NoSQL](https://searchdatamanagement.techtarget.com/definition/NoSQL-Not-Only-SQL) databases as well as Hadoop and its companion tools, including:

* [YARN](https://searchdatamanagement.techtarget.com/definition/Apache-Hadoop-YARN-Yet-Another-Resource-Negotiator): a cluster management technology and one of the key features in second-generation Hadoop.
* [MapReduce](https://searchcloudcomputing.techtarget.com/definition/MapReduce): a software framework that allows developers to write programs that process massive amounts of unstructured data in parallel across a distributed cluster of processors or stand-alone computers.
* [Spark](https://searchbusinessanalytics.techtarget.com/definition/Apache-Spark): an open-source parallel processing framework that enables users to run large-scale data analytics applications across clustered systems.
* [HBase](https://searchdatamanagement.techtarget.com/definition/Apache-HBase): a column-oriented key/value data store built to run on top of the Hadoop Distributed File System (HDFS).
* [Hive](https://searchdatamanagement.techtarget.com/definition/Apache-Hive): an open-source data warehouse system for querying and analyzing large datasets stored in Hadoop files.
* [Kafka](https://whatis.techtarget.com/definition/Apache-Kafka): a distributed publish-subscribe messaging system designed to replace traditional message brokers.
* [Pig](https://searchdatamanagement.techtarget.com/definition/Apache-Pig): an open-source technology that offers a high-level mechanism for the parallel programming of MapReduce jobs to be executed on Hadoop clusters.
* Sqoop:is a tool designed to transfer data between Hadoop and relational database servers. It is used to import data from relational databases such as MySQL, Oracle to Hadoop HDFS, and export from Hadoop file system to relational databases

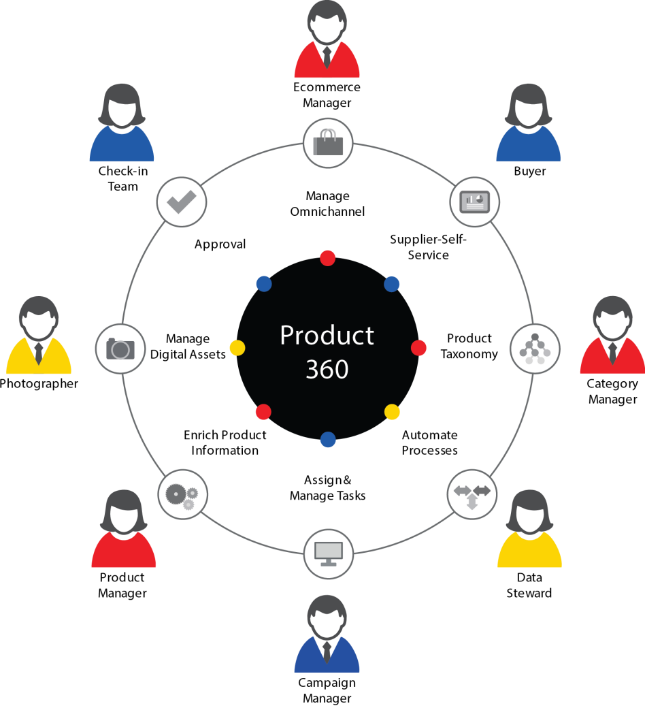
**What is customer 360 ?**

The term a “360-degree view of the customer” has been used in the industry for several years. But what exactly does it mean?

A 360-degree view of the customer is a single, end-to-end picture of the customer’s journey and experience with a company, and how they felt at steps along the journey.

It is a strategic approach enabling businesses to offer the best customer experience across all channels, by allowing for a unified view of all customer touchpoints with all departments involved in customer relationships.

This unified view is created by aggregating the ‘hard’ and ‘soft’ data the company captures about its customers and their interactions, across multiple channels and from different data sources.



Our task is to do some operation on customer 360 data sets

We have given data sets ,we have to move those data sets in to HDFS system from LFS system. Then,

We have to create a table customer360\_amit table in hbase and columns

We have to transfer data to hbase table using pig from hdfs

Then using hive we have to show those data .

### we have to start BIG INSIGHTS using command

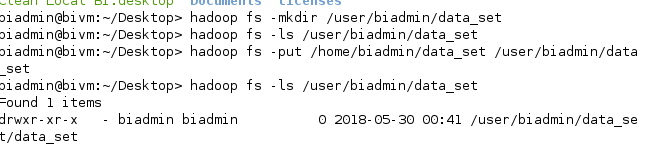
1. cd $BIGINSIGHTS\_HOME/bin
2. ./start-all.sh

we can also start by clicking on icon on desktop big insights start.

After starting big insights we have to move our data sets from LFS to HDFS

hadoop fs -put #path #toDest

hadoop fs -put /home/biadmin/data\_set /user/biadmin



check using command hadoop fs –ls #path of the file

to ensure we sussesfully put data there .

afterwards

create table in hbase system and respective column

fisrst start hbase shell

cd $HBASE\_HOME/bin

./hbase shell

After starting hbase create table using commands

create 'custmer360\_amit',{NAME=>'demographics'},{NAME=>'savings'},{NAME=>'loan'},{NAME=>'credit'},{NAME=>'deposit'},{NAME=>'credittrxsummary'}



Using list command we ensure that tabel created sussesfuly or not

list 'custmer360.\*'

describe command provides a decription of the specified table or view. For a list of tables in the current schema

describe 'custmer360\_amit'

Transfer data to hbase table using pig from hdfs

Start pig shell

cd $PIG\_HOME/bin

./pig

Load data of demographics.csv from hdfs to hbase

demographics = LOAD '/user/biadmin/data\_set/demographics.csv' USING PigStorage(',') as (customerid:chararray,registrationdate:chararray, age:int,gender:chararray,occupation:chararray,income:chararray);

after loading generate diffrent diffrent column

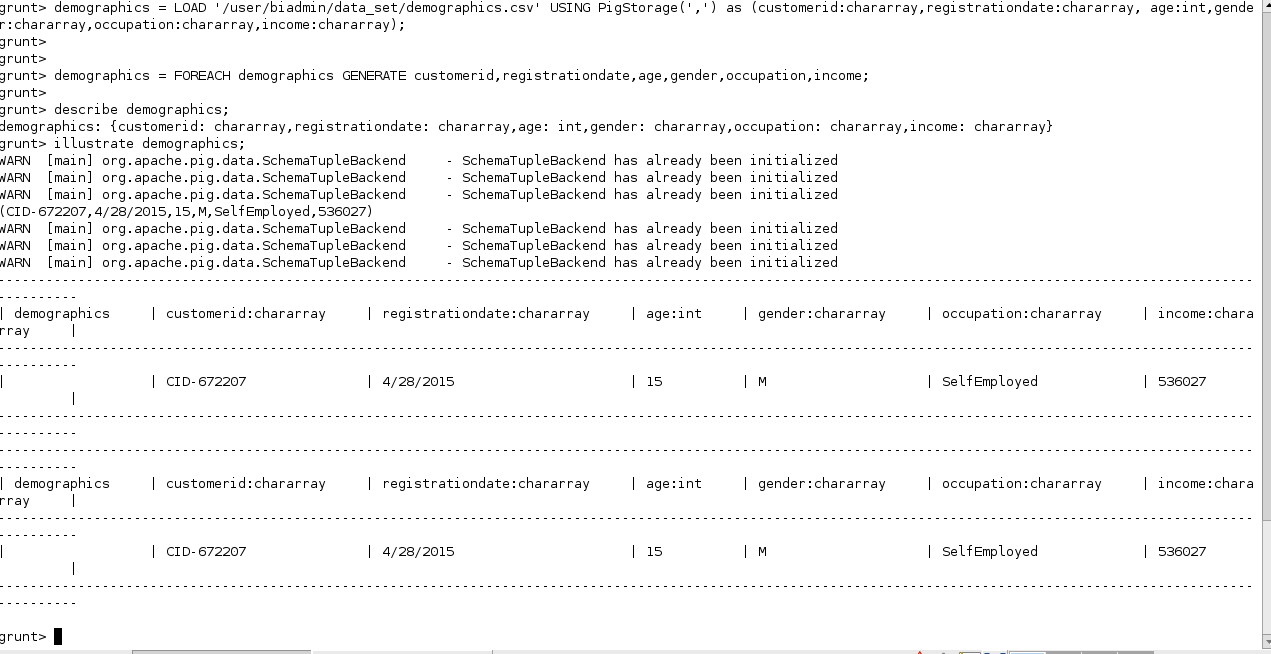
using command

demographics = FOREACH demographics GENERATE customerid,registrationdate,age,gender,occupation,income;

then describe using,

describe demographics;

and ilustrate , illustrate demographics;



And finnaly store data to hbase

STORE demographics INTO 'hbase://custmer360\_amit' USING org.apache.pig.backend.hadoop.hbase.HBaseStorage(

'demographics:registrationdate

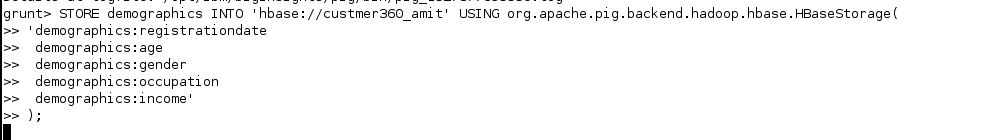
demographics:age

demographics:gender

demographics:occupation

demographics:income'

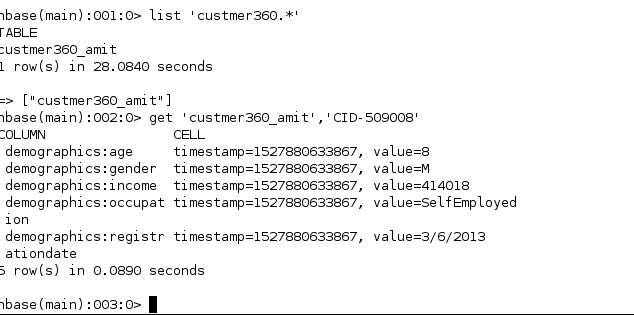
);



After storing in habse we have to check data are stored in hbase or not

By this command

get 'custmer360\_amit','CID-509008'



cid number can vary from diffrent machine

by same procedure we perform same operation on all sets of data

CREDITCARD

creditcard = LOAD '/user/biadmin/data\_set/creditcard.csv' USING PigStorage(',') as

(customerid:chararray, cardnumber:chararray, type:chararray, limit:chararray);

describe creditcard;

illustrate creditcard;

STORE creditcard INTO 'hbase://custmer360\_amit' USING org.apache.pig.backend.hadoop.hbase.HBaseStorage(

'credit:cardnumber

credit:type

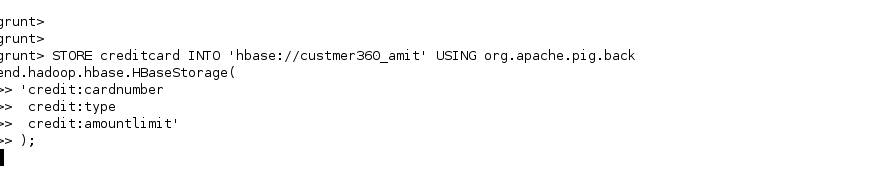
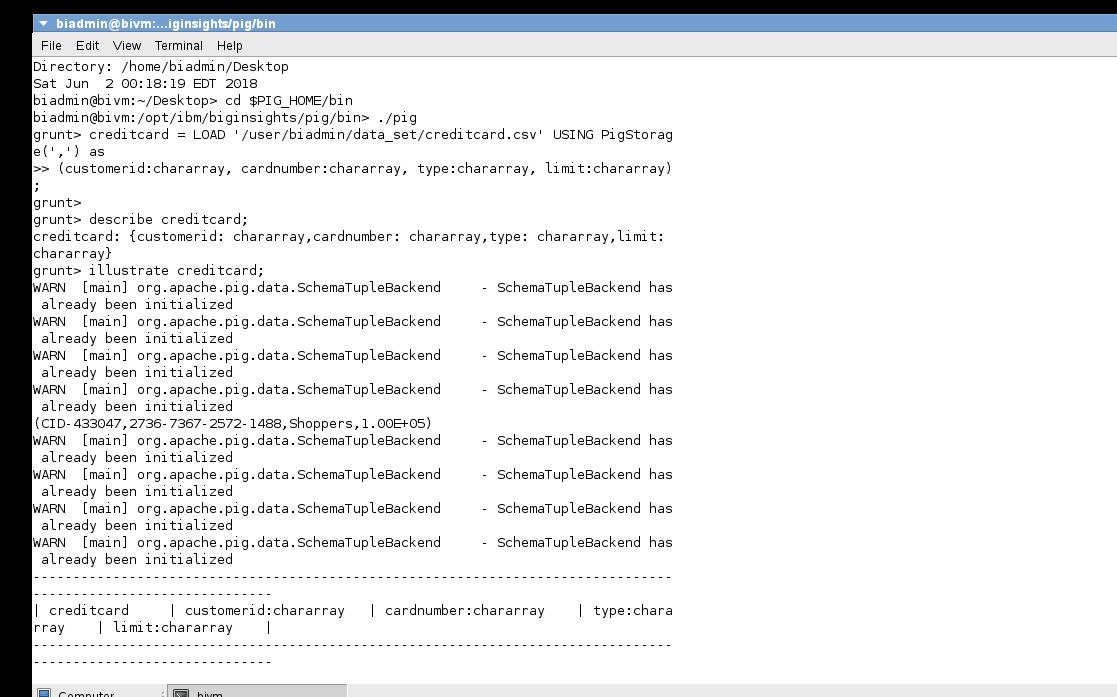
credit:amountlimit'

);

HBASE\_\_\_\_

get 'custmer360\_amit','CID-433047'

scan 'custmer360\_amit',{LIMIT=>10}



deposit account

depositaccount = LOAD '/user/biadmin/data\_set/depositaccount.csv' USING PigStorage(',') as (customerid:chararray, type:chararray,tenure:chararray);

describe depositaccount;

illustrate depositaccount;

STORE depositaccount INTO 'hbase://custmer360\_amit' USING org.apache.pig.backend.hadoop.hbase.HBaseStorage(

'deposit:type

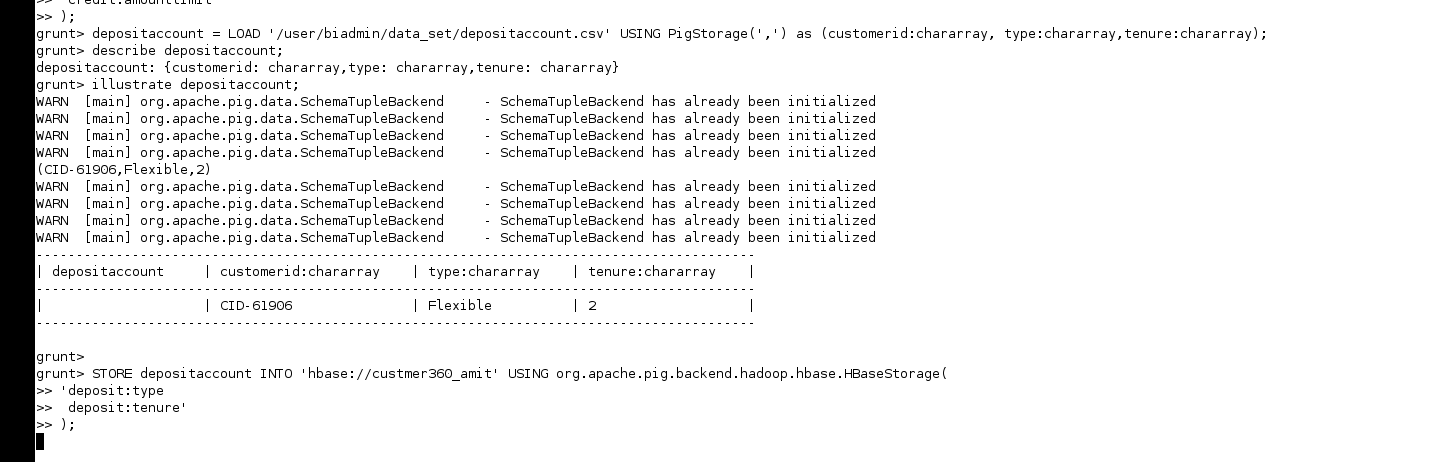
deposit:tenure'

);

HBASE\_\_\_\_

get 'custmer360\_amit','CID-61906'

scan 'custmer360\_amit',{LIMIT=>10}



loanaccount

loanaccount = LOAD '/user/biadmin/data\_set/loanaccount.csv' USING PigStorage(',') as (customerid:chararray, loanid:chararray,type:chararray, santionedamount:chararray,tenure:chararray);

describe loanaccount;

illustrate loanaccount;

STORE loanaccount INTO 'hbase://custmer360\_amit' USING org.apache.pig.backend.hadoop.hbase.HBaseStorage(

'loan:loanid

loan:type

loan:santionedamount

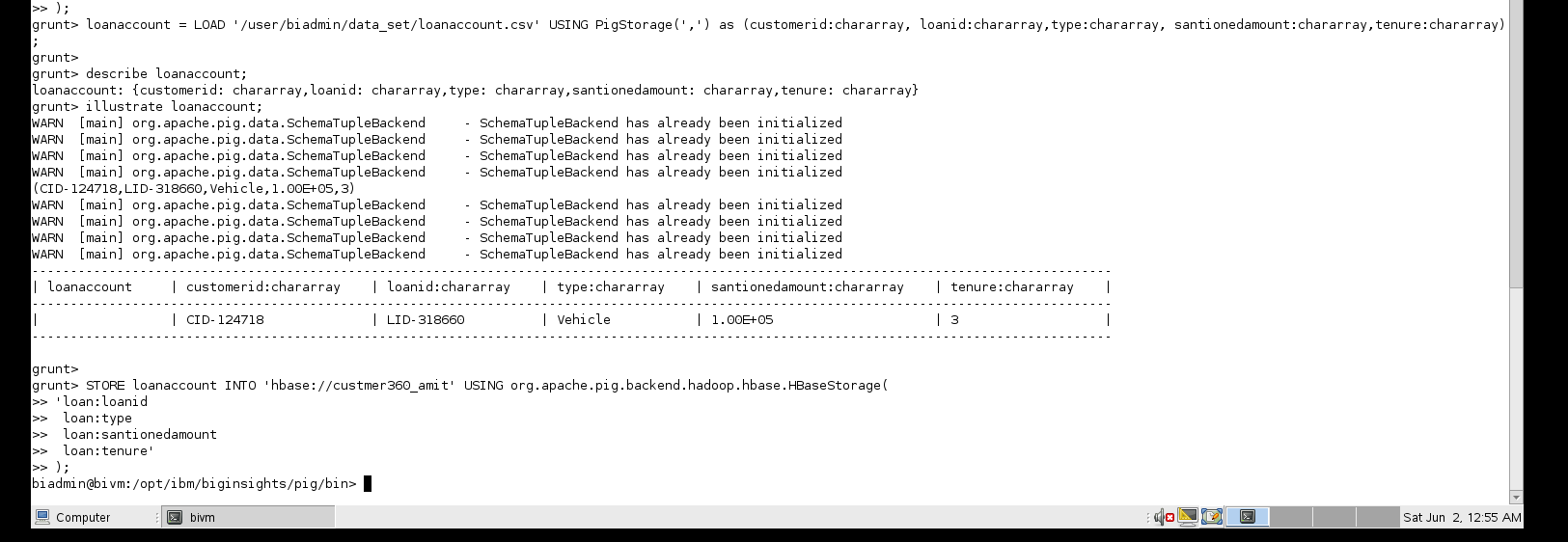
loan:tenure'

);

HBASE\_\_\_\_

get 'custmer360\_amit','CID-124718'

scan 'custmer360\_amit',{LIMIT=>10}



SAVING ACCOUNT

savingsaccount = LOAD '/user/biadmin/data\_set/savingsaccount.csv' USING PigStorage(',') as (customerid:chararray, savingsid:chararray, avgbalance:chararray);

describe savingsaccount;

illustrate savingsaccount;

STORE savingsaccount INTO 'hbase://custmer360\_amit' USING org.apache.pig.backend.hadoop.hbase.HBaseStorage(

'savings:savingsid

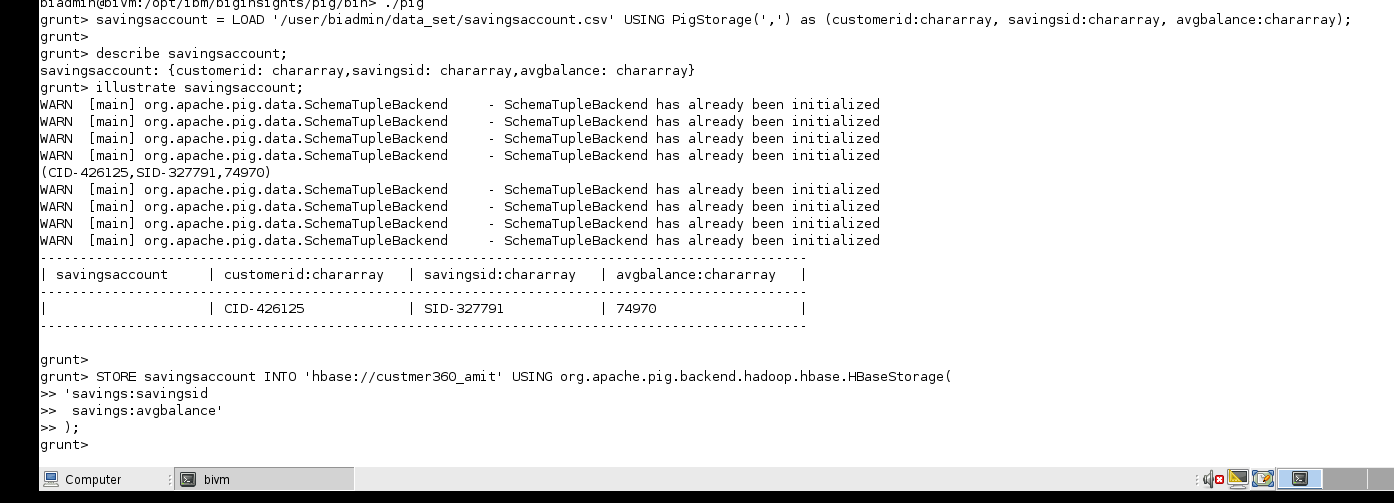
savings:avgbalance'

);

HBASE\_\_\_\_

get 'custmer360\_amit','CID-426125'

scan 'custmer360\_amit',{LIMIT=>10}



**creditcardtrx**

creditcardtrx = LOAD '/user/biadmin/data\_set/creditcardtrx.csv' USING PigStorage(',') as (customerid:chararray, trxamount:int, trxdate:chararray, trxtype:chararray);

describe creditcardtrx;

illustrate creditcardtrx;

grouped = GROUP creditcardtrx by (customerid, trxtype);

describe grouped;

illustrate grouped;

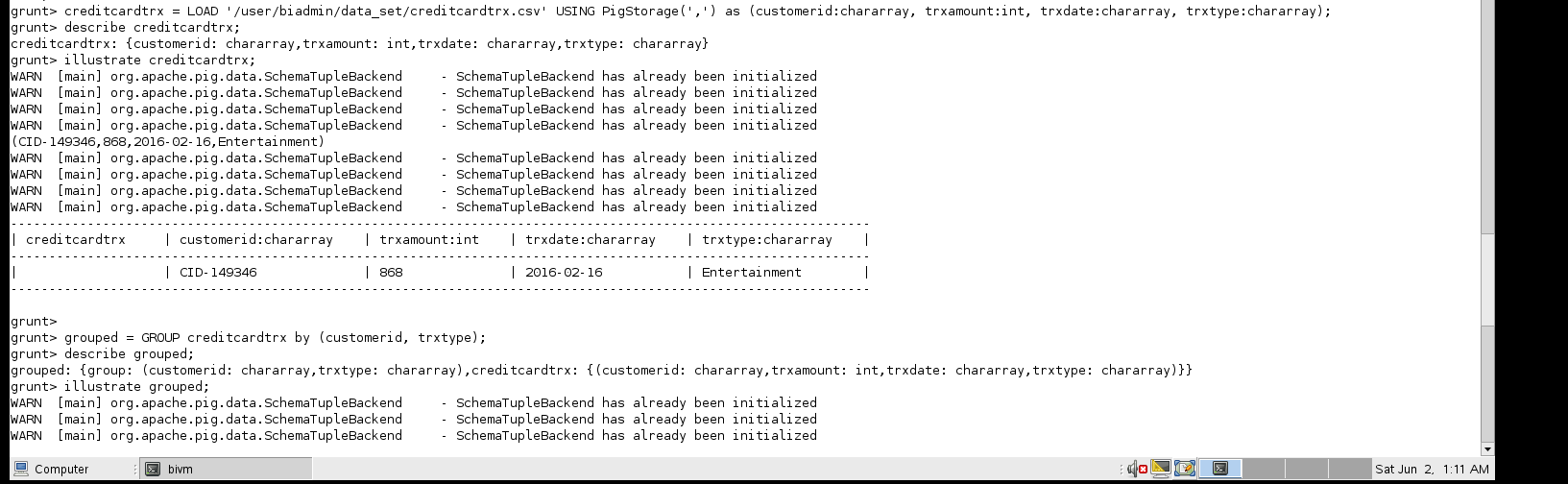
#Sum

total = FOREACH grouped GENERATE group.customerid as customerid, group.trxtype as trxtype, SUM(creditcardtrx.trxamount) as salevalue;

total = LIMIT total 100;

describe total;

illustrate total;



#Distinct transaction types

distincttrntype = FOREACH total GENERATE trxtype;

distincttrntype = DISTINCT distincttrntype;

dump distincttrntype;

///////////////////////////////////////////

//////////////

#Filter

food = FILTER total BY trxtype == ')Food';

food = FOREACH food GENERATE customerid, salevalue;

STORE food INTO 'custmer360\_amit' USING org.apache.pig.backend.hadoop.hbase.HBaseStorage( 'credittrxsummary:food' );

HIVE..

Using hive we have to perfrom some analysis on data

Login hive shell by

cd $HIVE\_HOME/bin

./hive shell

Create data base in hive

create database custmer360\_amit;

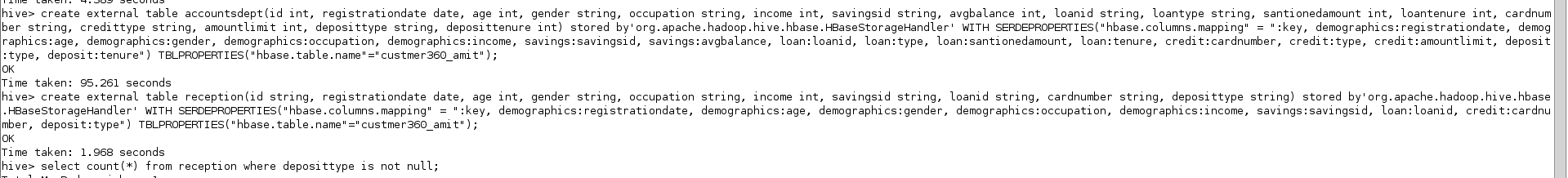
use database;

use custmer360\_amit;

create table

create external table accountsdept(id int, registrationdate date, age int, gender string, occupation string, income int, savingsid string, avgbalance int, loanid string, loantype string, santionedamount int, loantenure int, cardnumber string, credittype string, amountlimit int, deposittype string, deposittenure int) stored by'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES("hbase.columns.mapping" = ":key, demographics:registrationdate, demographics:age, demographics:gender, demographics:occupation, demographics:income, savings:savingsid, savings:avgbalance, loan:loanid, loan:type, loan:santionedamount, loan:tenure, credit:cardnumber, credit:type, credit:amountlimit, deposit:type, deposit:tenure") TBLPROPERTIES("hbase.table.name"="custmer360\_amit");

create external table reception(id string, registrationdate date, age int, gender string, occupation string, income int, savingsid string, loanid string, cardnumber string, deposittype string) stored by'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES("hbase.columns.mapping" = ":key, demographics:registrationdate, demographics:age, demographics:gender, demographics:occupation, demographics:income, savings:savingsid, loan:loanid, credit:cardnumber, deposit:type") TBLPROPERTIES("hbase.table.name"="custmer360\_amit");



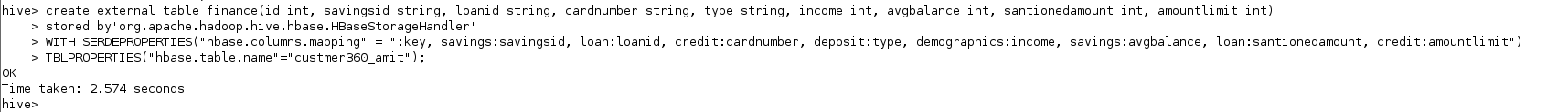
select count(\*) from reception where deposittype is not null;

create external table finance(id int, savingsid string, loanid string, cardnumber string, type string, income int, avgbalance int, santionedamount int, amountlimit int)

stored by'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

WITH SERDEPROPERTIES("hbase.columns.mapping" = ":key, savings:savingsid, loan:loanid, credit:cardnumber, deposit:type, demographics:income, savings:avgbalance, loan:santionedamount, credit:amountlimit")

TBLPROPERTIES("hbase.table.name"="custmer360\_amit");



create external table if not exists loandept(id int, savingsid string, avgbalance int, loanid string, type string, santionedamount int, tenure int)

stored by'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

WITH SERDEPROPERTIES("hbase.columns.mapping" = ":key, savings:savingsid, savings:avgbalance, loan:loanid, loan:type, loan:santionedamount, loan:tenure")

TBLPROPERTIES("hbase.table.name"="custmer360\_amit");

DESCRIBE EXTENTEDED accountsdept;

SELECT \* FROM accountsdept LIMIT 5;

EXPLAIN SELECT \* FROM accountsdept;

Select \* from loandept where type = ‘Vehicle’;

select count(\*) from loandept where avgbalance is not null;

