NIIT

SAN-JAN DATA ANALYST AND TECH

BIG DATA FOR BETTER SOLUTION

Version *<2.0>*

<24/11/2016>

Author: Janakiraman

Table of Contents

1. Project definition 3
   1.1 What is BIG DTA 3
   1.2 Background 3
   1.3 Business Case 3
   1.4 Project objectives 3
   1.5 Project Scope 4
   1.6 Desired output for project 4
   1.7 Project budget 4
   1.8 Tools and Techniques 4

2. E-Commerce 7
   2.1 What is E-Commerce 7
   2.2 Databases for Transaction details 7
   2.3 Databases for Customer details 7
   2.4 List of use cses 8

3. Project through Mapreduce 9

4. Project through Hive 13

5. Project through Pig 15

6. Software and Hardware Requirement 19

7. Conclusion 19

Project Definition

1.1 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 analysed for insights that lead to better decisions and strategic business moves. Big data is changing the way people within organizations work together. It is creating a culture in which business and IT leaders must join forces to realize value from all data. Insights from big data can enable all employees to make better decisions—deepening customer engagement, optimizing operations, preventing threats and fraud, and capitalizing on new sources of revenue. But escalating demand for insights requires a fundamentally new approach to architecture, tools and practices.

1.2 Background

San-Jan Data Analyst and Tech is a MNC (Multi National Company) for data analysis and to provide technology solution to the major industries. We uses various scenarios in Big Data to provide desired outcomes.

We are provide accurate data solution to the industries who`s generating huge amount data, using various tools and techniques under Big Data technology.

1.3 Business Case

Huge amount of data being generated by everything around us at all times. Every digital process and social media exchange produces it. Industries struggling with handle this amount of data. So we made it as a business to give accurate data solution.

1.4 Project objectives

Here our current project fully based on E-Commerce, there are tons of transaction process, log files and feedback files created. All Industries struggling to handle these huge volume of data generated by the customers. This project specifically for provide clean data solution about E-Commerce.

1.5 Project scope

Our E-Commerce project scope to provide clear outcomes as per the client`s requirement.

1.6 Desired output for project

Our project outputs comes with clear scenarios like different use cases, conditions and filtration that has applied on each phases. So it should be clear vision about what client expected.

1.7 Project budget

We offer less charge depend with volume of data, and for this particular E-Commerce project our offer is very flexible charge for clients.

1.8 Tools and Techniques

Various complex tools and mind crashing techniques we applied for this project are…

1.8.1 Hadoop Framework

1.8.2 Hive

1.8.3 Pig

1.8.1 Hadoop Framework

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

The project includes these modules:

Hadoop Common: The common utilities that support the other Hadoop modules.

Hadoop Distributed File System (HDFS™): A distributed file system that provides high-throughput access to application data.

Hadoop YARN: A framework for job scheduling and cluster resource management.

Hadoop MapReduce: A YARN-based system for parallel processing of large data sets.

Benefits

Some of the reasons organizations use Hadoop is its' ability to store, manage and analyse vast amounts of structured and unstructured data quickly, reliably, flexibly and at low-cost.

Scalability and Performance – distributed processing of data local to each node in a cluster enables Hadoop to store, manage, process and analyse data at petabyte scale.

Reliability – large computing clusters are prone to failure of individual nodes in the cluster. Hadoop is fundamentally resilient – when a node fails processing is re-directed to the remaining nodes in the cluster and data is automatically re-replicated in preparation for future node failures.

Flexibility – unlike traditional relational database management systems, you don't have to create structured schemas before storing data. You can store data in any format, including semi-structured or unstructured formats, and then parse and apply schema to the data when read.

Low Cost – unlike proprietary software, Hadoop is open source and runs on low-cost commodity hardware.

1.8.2 Hive

The Apache Hive data warehouse software facilitates reading, writing, and managing large datasets residing in distributed storage using SQL. Structure can be projected onto data already in storage. A command line tool and JDBC driver are provided to connect users to Hive.

Benefits

Time-It take very less time to write Hive Query compared to Map Reduce code. For example, the word count problem which takes around 50 lines of code can be written in 5 lines in Hive. So, you save time.

Easy-It is very easy to write query involving joins (if there are few joins) in Hive.

Maintenance-It has very low maintenance and is very simple to learn & use (low learning curve).

1.8.3 Pig

Apache Pig is a platform for analysing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

Benefits

Ease of programming. It is trivial to achieve parallel execution of simple, "embarrassingly parallel" data analysis tasks

Optimization opportunities. The way in which tasks are encoded permits the system to optimize their execution automatically, allowing the user to focus on semantics rather than efficiency.

Extensibility. Users can create their own functions to do special-purpose processing.

E-Commerce

2.1 What is E-Commerce?

E-commerce (electronic commerce or EC) is the buying and selling of goods and services, or the transmitting of funds or data, over an electronic network, primarily the internet. These business transactions occur either as business-to-business, business-to-consumer, consumer-to-consumer or consumer-to-business.

2.2 Database for Transaction details

Here is the transaction database for example.

Tid	Date	Cust id	Amount	Category	Product	City	State	Payment mode
00000001	06-26-2015	4007024	040.33	Exercise & Fitness	Cardio Machine Accessories	Clarksville	Tennessee	credit
00000002	06-26-2015	4007023	238	Gymnastics	Gymnastics Rings	Des Moines	Iowa	credit
00000005	06-26-2015	4007028	1265	Outdoor Recreation	Camping & Backpacking & Hiking	Chicago	Illinois	credit

2.3 Database for Customer details

Here is the customer database for example.

Custid	Fname	Lname	Age	Profession
4000001	Sunitha	Devi	22	Teacher
4000002	Arul	Selvan	25	mentor
4000003	Vinod	Kumar	33	manager

2.4 List of use cases

- Find all the transaction where amt>160.
- Count all the transaction where amount is between 175-200.
- Calculate the total sum and total count of all the transaction for each user id.
- Calculate the average transaction value for each user id.
- Calculate total sales amt for each Month.
- Divide the file into 12 files, each file containing each month of data. For eg. File 1 should contain data of January txn, file 2 should contain data of February txn.
- Find the profession of user who has spent the maximum amount.
- Find the name of top 3 spenders.
- Find the user who has spent the max amount in July month

Project through MapReduce

Use case 1:

To get all the details from the transaction amount detail that is greater than with a specific amount which the user wants. This use case work with user interaction means it will work with the input given by the user and validation also successfully done here.

Taken input: 150

Expected output:

```
00049986        156.38
00049991        191.29
00049994        177.22
00049996        163.81
00049998        180.41
00049999        168.49
hduser@ubuntu64server:~$
```

Use case 2:

To count all the transaction where amount is between 150 and 500. This use case also work with user interaction means it will work with the input given by the user and validation also successfully done here.

Taken input: 150 and 500

```
hduser@ubuntu64server:~$ hadoop fs -cat /johncount/p*^C
hduser@ubuntu64server:~$ hadoop jar amtbw.jar /johnin/txns-large.dat /johnamtbw
Enter the lower limit
150
Enter the upper limit
170
16/11/21 13:16:55 INFO client.RMProxy: Connecting to ResourceManager at /192.168.56.123:8032
16/11/21 13:16:57 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed.
```

Expected output:

```
hduser@ubuntu64server:~$ hadoop fs -ls /johnamtbw
Found 2 items
-rw-r--r--   1 hduser supergroup          0 2016-11-21 13:17 /johnamtbw/_SUCCESS
-rw-r--r--   1 hduser supergroup          5 2016-11-21 13:17 /johnamtbw/part-r-00000
hduser@ubuntu64server:~$ hadoop fs -cat /johnamtbw/p*
5068
hduser@ubuntu64server:~$
```

Use case 3:

Calculate the total sum and total count of all the transaction for each user id. This use case also work with user interaction means it will work with the input given by the user and validation also successfully done here.

Taken input:

```
hduser@ubuntu64server:~$ hadoop jar sum.jar /johnin/txns-large.dat /johnsum
Enter the User Id
4004613
16/11/21 13:24:42 INFO client.RMProxy: Connecting to ResourceManager at /192.168.56.123:8032
16/11/21 13:24:44 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed.
```

Expected output:

```
hduser@ubuntu64server:~$ hadoop fs -ls /johnsum
Found 2 items
-rw-r--r--   1 hduser supergroup          0 2016-11-21 13:25 /johnsum/_SUCCESS
-rw-r--r--   1 hduser supergroup         63 2016-11-21 13:25 /johnsum/part-r-00000
hduser@ubuntu64server:~$ hadoop fs -cat /johnamtbw/p*^C
hduser@ubuntu64server:~$ hadoop fs -cat /johnsum/p*
4004613 Sum : 800.05   Count : 9   Average : 88.89444444444445
hduser@ubuntu64server:~$
```

Use case 4:

Calculate total sales amount for each Month. This use case also work with user interaction means it will work with the input given by the user and validation also successfully done here.

Taken input:

```
hduser@ubuntu64server:~$ hadoop jar ttlsl.jar /johnin/txns-large.dat /johnttlsl
Enter the Months
05
16/11/21 13:37:24 INFO client.RMProxy: Connecting to ResourceManager at /192.168.56.123:8032
16/11/21 13:37:26 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed.
```

Expected output:

```
hduser@ubuntu64server:~$ hadoop fs -ls /johnttlsl/p*
-rw-r--r--   1 hduser supergroup         22 2016-11-21 13:38 /johnttlsl/part-r-00000
hduser@ubuntu64server:~$ hadoop fs -cat /johnttlsl/p*
05      432627.58000000013
hduser@ubuntu64server:~$
```

Use case 5:

Divide the file into 12 files, each file containing each month of data. For e.g. file 1 should contain data of January transaction, file 2 should contain data of February transaction. This use case also work with user interaction means it will work with the input given by the user and validation also successfully done here.

Taken input:

```
hduser@ubuntu64server:~$ hadoop fs -ls /johnmonthofdata/p*
-rw-r--r--   1 hduser supergroup     432933 2016-11-21 13:51 /johnmonthofdata/part-r-00000
-rw-r--r--   1 hduser supergroup     389153 2016-11-21 13:51 /johnmonthofdata/part-r-00001
-rw-r--r--   1 hduser supergroup     442575 2016-11-21 13:51 /johnmonthofdata/part-r-00002
-rw-r--r--   1 hduser supergroup     422696 2016-11-21 13:51 /johnmonthofdata/part-r-00003
-rw-r--r--   1 hduser supergroup     426463 2016-11-21 13:51 /johnmonthofdata/part-r-00004
-rw-r--r--   1 hduser supergroup     422470 2016-11-21 13:51 /johnmonthofdata/part-r-00005
-rw-r--r--   1 hduser supergroup     430830 2016-11-21 13:52 /johnmonthofdata/part-r-00006
-rw-r--r--   1 hduser supergroup     429555 2016-11-21 13:52 /johnmonthofdata/part-r-00007
-rw-r--r--   1 hduser supergroup     422035 2016-11-21 13:52 /johnmonthofdata/part-r-00008
-rw-r--r--   1 hduser supergroup     427267 2016-11-21 13:52 /johnmonthofdata/part-r-00009
-rw-r--r--   1 hduser supergroup     409774 2016-11-21 13:52 /johnmonthofdata/part-r-00010
-rw-r--r--   1 hduser supergroup     424714 2016-11-21 13:52 /johnmonthofdata/part-r-00011
hduser@ubuntu64server:~$
```

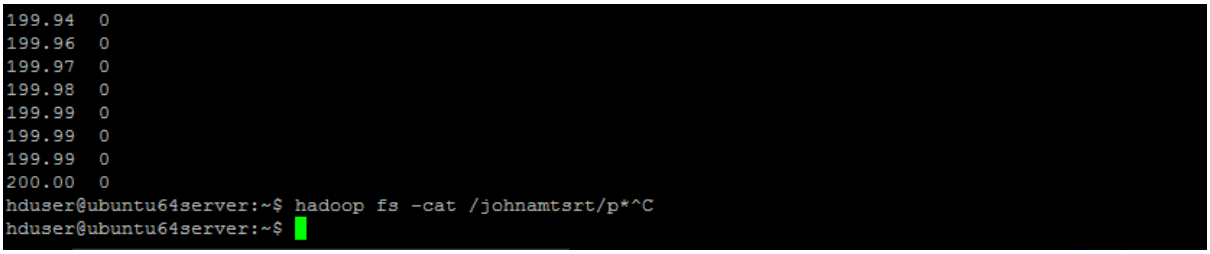
Expected output:

```
6	00004827	06-26-2015	4008117	092.66	Gymnastics	Pommel Horses	El Paso	Texas	credit
6	00043676	06-02-2015	4007212	034.62	Combat Sports	Martial Arts	Boston	Massachusetts	cash
6	00040361	06-16-2015	4008509	157.99	Outdoor Play Equipment	Playhouses	Tampa	Florida	credit
6	00033323	06-29-2015	4003604	155.52	Exercise & Fitness	Cardio Machines	Columbia	South Carolina	credit
6	00008611	06-10-2015	4003102	018.02	Water Sports	Life Jackets	Long Beach	California	credit
6	00038036	06-22-2015	4001694	108.44	Team Sports	Cricket	Milwaukee	Wisconsin	credit
6	00026380	06-11-2015	4006188	187.19	Gymnastics	Vaulting Horses	Midland	Texas	credit
6	00017388	06-23-2015	4007484	036.71	Exercise & Fitness	Jump Ropes	Stamford	Connecticut	cash
6	00021249	06-05-2015	4008624	141.27	Outdoor Recreation	Riding Scooters	Midland	Texas	credit
6	00029186	06-01-2015	4009240	091.88	Outdoor Play Equipment	Water Tables	Charlotte	North Carolina	credit
6	00045234	06-08-2015	4006223	159.87	Exercise & Fitness	Free Weight Bars	Omaha	Nebraska	credit
6	00043675	06-01-2015	4005590	011.22	Jumping	Trampolines	Portland	Oregon	cash
6	00016906	06-04-2015	4005417	061.17	Winter Sports	Downhill Skiing	Washington	District of Columbia	credit
6	00029182	06-23-2015	4005664	144.61	Gymnastics	Springboards	New York	New York	credit
6	00029181	06-13-2015	4000964	184.74	Games	Board Games	San Antonio	Texas	credit
6	00024472	06-11-2015	4007815	142.86	Outdoor Recreation	Fishing	Brownsville	Texas	credit
6	00018115	06-17-2015	4000792	198.25	Jumping	Trampolines	Pasadena	Texas	credit
6	00045236	06-03-2015	4000447	125.59	Exercise & Fitness	Weightlifting Machine Accessories	Philadelphia	Pennsylvania	credit
6	00033325	06-30-2015	4005382	028.95	Racquet Sports	Racquetball	Kansas City	Kansas	credit
6	00000598	06-28-2015	4007452	062.34	Team Sports	Cricket	Denton	Texas	credit
6	00007593	06-05-2015	4006795	078.51	Team Sports	Cheerleading	Oklahoma City	Oklahoma	credit
6	00001743	06-13-2015	4006354	173.72	Gymnastics	Gymnastics Rings	Scottsdale	Arizona	credit
6	00029173	06-06-2015	4007332	037.89	Outdoor Recreation	Camping & Backpacking & Hiking	Oakland	California	cash
6	00029172	06-02-2015	4007044	110.54	Outdoor Recreation	Riding Scooters	Indianapolis	Indiana	credit
6	00037182	06-14-2015	4003597	097.44	Gymnastics	Vaulting Horses	Detroit	Michigan	credit
6	00024474	06-23-2015	4008215	180.41	Water Sports	Water Polo	Memphis	Tennessee	credit
```

Use case 6:

Sorting all the transaction amount in ascending order.

Expected output:



Project through Hive

Use case 1:

To get all the details from the transaction amount detail that is greater than with a specific amount which the user wants.

Execution Query:

```
select * from Ecom1 where amt>160;
```

Use case 2:

Count all the transaction where amount is between 175 to 200.

Execution Query:

```
select * from Ecom1 where amt between 175 and 200
```

Use case 3:

Calculate the total sum and total count and average of all the transaction for each user id.

Execution Query:

```
select uid,sum(amt),count(amt),avg(amt) from Ecom1 group by uid
```

Use case 4:

Find the name of user who has spend the maximum amount.

Execution Query:

select a.uid,SUM(a.amt) as Res,b.fname from ecom1 a join cust b on a.uid=b.uid group by a.uid,b.fname order by Res DESC limit 1;

Use case 5:

Find the name and product category for each user id and print in ascending order by product category.

Execution Query:

Select cust.fname, transaction.cat from cust inner join transaction on cust.id=transaction.id order by transaction .cat;

Project through Pig

Use case 1:

To get all the details from the transaction amount detail that is greater than with a specific amount which the user wants.

Execution Script:

```
a= load '/user/cloudera/Transactional.dat' using PigStorage(',');

b = foreach a generate $2,$3;

c = filter b by $1>160;

dump c;

store c into '/user/cloudera/task1';
```

Use case 2:

Count all the transaction where amount is between 175 to 200.

Execution Script:

```
A = load '/user/cloudera/txns-large.dat' using PigStorage (',') as (tid, d, uid, amt : double , cat, prod,city,state,pt);

B = foreach  A generate tid, amt;

C = filter B by ($1>170 and $1<200);

D = foreach C generate 1 as one;

E = group D by one;

F = foreach E generate COUNT(D.one);

dump F;
```

Use case 3:

Calculate the total sum and total count of all the transaction for each user id.

Execution Script:

```
step1 = load '/user/cloudera/txns-large.dat' using PigStorage(',');
step2 = foreach step1 generate $2 as uid,$3 as amt;
step3 = group step2 by uid;
step4 = foreach step3 generate group, SUM(step2.amt);
dump step4;
```

Use case 4:

Calculate the average transaction value for each user id.

Execution Script:

```
step1 = load '/user/cloudera/txns-large.dat' using PigStorage(',');
step2 = foreach step1 generate $2 as uid,$3 as amt;
step3 = group step2 by uid;
step4 = foreach step3 generate group,
SUM(step2.amt),COUNT(step2.amt),AVG(step2.amt);
dump step4;
```

Use case 5:

Calculate total sales amt for each Month.

Execution Script:

```
a = load '/user/cloudera/txns-large.dat' using PigStorage(',') as
(tid,tdate:chararray,uid,amt:double,cat,acc,city,state,pay);
b = foreach a generate SUBSTRING(tdate,0,2) as mon, amt;
c = group b by mon;
d = foreach c generate group, SUM(b.amt) as sum;
dump d;
```

Use case 6:

Divide the file into 12 files, each file containing each month of data. For eg. file1 should contain data of january txn, file 2 should contain data of feb txn.

Execution Script:

```
a = load '/user/cloudera/txns-large.dat' using PigStorage(',') as
(tid,tdate:chararray,uid,amt:double,cat,acc,city,state,pay);;

b = foreach a generate SUBSTRING(tdate,0,2) as
month,tid,tdate,uid,amt,cat,acc,city,state,pay;

c = filter b by month=='01';

dump c;

d = filter b by month=='02';

dump d;

e = filter b by month=='03';

dump e;

f = filter b by month=='04';

dump f;

g = filter b by month=='05';

dump g;

h = filter b by month=='06';

dump h;

i = filter b by month=='07';

dump i;

j = filter b by month=='08';

dump j;

k = filter b by month=='09';

dump k;

l = filter b by month=='10';

dump l;

m = filter b by month=='11';

dump m;

n = filter b by month=='12'; dump n;
```

Use case 7:

Find the name of top 3 spenders.

Execution Script:

```
a = load '/user/cloudera/txns-large.dat' using PigStorage(',') as
(tid,tdate,uid:int,amt:double,cat,acc,city,state,pay);

b = load '/user/cloudera/custs-large.dat' using PigStorage(',') as

(uid:int,fname:chararray,lname,age,prof);

c = join a by uid,b by uid;

d = foreach c generate $2 as uid, $3 as amt,$10 as fname;

e = group d by (uid,fname);

f = foreach e generate group, SUM(d.amt) as Total;

g = order f by Total DESC;

h = limit g 3;

dump h;
```

Use case 8:

Find the profession of user who has spend the maximum amount.

Execution Script:

```
a = load '/user/cloudera/txns-large.dat' using PigStorage(',') as
(tid,tdate,uid:int,amt:double,cat,acc,city,state,pay);

b = load '/user/cloudera/custs-large.dat' using PigStorage(',') as

(uid:int,fname:chararray,lname,age,prof:chararray);

c = join a by uid,b by uid;

d = foreach c generate $2 as ID,$3 as tamt,$13 as Prof;

f = group d by ($0,$2);

h = foreach f generate group,SUM(d.tamt) as Res;

i = order h by Res DESC;

j = limit i 1;  dump j;
```

Software and Hardware requirement

- **Operating System** : Windows 7,8,10 and Mac.
- **Supporting software's:** Ubuntu,putty, Oracle VM VirtualBox,WinSCP.
- **RAM** : Minimum 4GB.

Conclusion

With these different scenarios we can give different dimensional solution in accurate with a huge dataset. This type of data solution will help Industries to maintain their customer and transaction details as well. We are very trustable MNC for data solution and we can proudly say, we are the best data analyser and tech solution provider for other MNC`s and start-ups companies who needs data solution.