Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 85 (2016) 463 – 474

Procedia Computer Science

International Conference on Computational Modeling and Security (CMS 2016)

Managing Mysql Cluster Data using Cloudera Impala

Sahithi Tummalapalli[a], Venkata rao Machavarapu[b]

[a]*Department of C.S.E,VFSTR University,Guntur,Andhra Pradesh*
[b]*Department of C.S.E, Chirala Engineering College,Chirala,Andhra Pradesh*

Abstract

MySQL Cluster is a widely used clustered database used to store and manipulate data which has a shared-nothing clustering for the MySql database management system providing high availability and high throughput with low latency. The problem with MySQL Cluster is that as the data grows larger, the time required to process the data increases and additional resources may be needed.With Hadoop and Impala,data processing time can be faster than MySql cluster and probably faster than Hive and Pig.This paper provides preliminary results.Evaluation results indicates that Impala achieves acceptable perfomance for some data analysis and processing tasks even compared with Hive and Pig and MySql cluster.

Keywords: Hadoop, Impala, Hive, Pig; MySql cluster; Big data, Metastore.

1.Introduction

Hadoop is a framework which provides open source libraries for distributed computing using simple single map-reduce interface and its own distributed filesystem.It facilitates scalability and takes cares of detecting and handling failures. Hadoop has components which take care of all complexities for us and by using a simple map reduce framework we are able to harness the power of distributed computing without having to worry about complexities like fault tolerance, data loss. Hadoop can be used for storing large data and for processing data such as data mining, report generation, file analysis, web indexing, and bioinformatic research.

MySQL Cluster is a technology providing shared nothing clustering and auto-sharding for the MySQL Database management system.MySQL Cluster has no single point of failure.It is designed to provide high availability and high throughput with low latency, while allowing for near linear scalability. MySQL Cluster is implemented as a fully distributed multi-master database ensuring updates made by any application or SQL node are instantly available to all of the other nodes accessing the cluster, and each data node can accept write operations. MySQL Cluster scales horizontally on commodity hardware with auto-sharding to serve read and write intensive workloads,

Corresponding author. Tel.: 91-9949413955.
*E-mail address:*mvrao239@gmail.com

Peer-review under responsibility of the Organizing Committee of CMS 2016
doi:10.1016/j.procs.2016.05.193

accessed via SQL and NoSQL interfaces. It supports in-memory and disk-based data, automatic data partitioning with load balancing and the ability to add nodes to a running cluster with zero downtime allows linear database scalability to handle the most unpredictable workloads. It consists of multiple nodes that are distributed across machines to make sure the system can work, even in case a node having a problem such as network failure[1].

Apache Hive and Apache Pig are open source programs for analyzing large data sets in a high-level language.Pig is a high level dataflow system along with a simple query algebra that lets the user declare data transformation to files or groups of files. Hive is data warehouse software that facilitates queries and manages a large data set in distributed storage. Hive allows users to extend the system with their own types and functions. The query language is very similar to SQL and therefore can be easily understood by anyone familiar with SQL. Hive and Pig run on top of Hadoop.[2][3]

Cloudera Impala is an Apache-licensed,real time query engine for data stored in HDFS.Impala is well suited to use cases where real time queries and speed are essential.But while many developers will be familiar with Hive and Pig,Impala uses its own daemons that are spread across the cluster for queries.Furthermore,Impala does not leverage MapReduce,allowing Impala to return result in real time.[4]

When it comes to querying large data sets on MySQL Cluster, it can take seconds. As the data grows larger, the time required to process the data increases too. Hadoop with Hive and Pig can process queries much faster than MySql cluster.But Hadoop with Cloudera Impala processes the queries fastest as it has its own daemons spread across the cluster for queries.

This paper presents the processing time of Impala,Hive, Pig, and MySQL Cluster on a simple data model with simple queries while the data is growing. Section 3 discusses a proposed method. Section 4 shows the results and explanations. And the last section, section 5 provides a conclusion.

2.Impala

Impala is the industry's first native real-time SQL query engine for Apache Hadoop, it is the newest component of CDH. Impala completely changes the way organizations can benefit from Hadoop. **Cloudera Impala** is Cloudera's open source massively parallel processing (MPP) SQL query engine for data stored in a computer cluster running Apache Hadoop. Using Impala,Data processing workload acceleration, with data pipelines will last seconds instead of minutes or hours, to meet tighter service-level agreement (SLA) specifications. It has a Interactive business intelligence with popular tools. This opens up real-time access to big data to every analyst in the organization, without requiring any special training, significantly lowering the adoption risk of a big data project and accelerating return on investment (ROI). It reduces overall cost of data management, Instead of replicating large amounts of data to a relational database to get interactive SQL performance, Cloudera customers can obtain the same experience without added cost or complexity.Impala is meant to be good at what hive is bad at i.e fast response queries and it is also meant to be good at what hive is good at.

Impala brings scalable parallel database technology to Hadoop, enabling users to issue low-latency SQL queries to data stored in HDFS and Apache HBase without requiring data movement or transformation. Impala is integrated with Hadoop to use the same file and data formats, metadata, security and resource management frameworks used by MapReduce, Apache Hive, Apache Pig and other Hadoop software. Impala is used by analysts and data scientists to perform analytics on data stored in Hadoop via SQL or business intelligence tools. The result is that large-scale data processing and interactive queries can be done on the same system using the same data and metadata – removing the need to migrate data sets into specialized systems and/or proprietary formats simply to perform analysis.

Before Impala, if your relational database was at capacity, you may have had no choice but to expand that system to maintain your expectations of performance. If you were using Hadoop to affordably analyze any amount or kind of data, but wanted interactive performance, you had to move that data into a fast relational database. You then had to accept the cost and effort of duplicate storage and data synchronization; accept the rigidity of requiring fixed schemas; accept that when you moved and transformed data you would inevitably leave something behind; accept that your analysis options would be limited in that target database.With Impala, you now have a choice. As a native component of the Hadoop ecosystem, Impala combines all of the benefits of other Hadoop frameworks, including flexibility, scalability, and cost-effectiveness, with the performance, usability, and SQL functionality necessary for an enterprise-grade analytic database.Impala was specifically targeted for integration with standard business

intelligence environments, and to that end supports most relevant industry standards: clients can connect via ODBC or JDBC; authentication is accomplished with Kerberos or LDAP; authorization follows the standard SQL roles and privileges.In order to query HDFS-resident data, the user creates tables via the familiar CREATE TABLE statement, which, in addition to providing the logical schema of the data, also indicates the physical layout, such as file format(s) and placement within the HDFS directory structure. Those tables can then be queried with standard SQL syntax.

WORKING OF IMPALA WITH HIVE:Impala makes use of many familiar components within the Hadoop ecosystem. Impala can interchange data with other Hadoop components, as both a consumer and a producer, so it can fit in flexible ways into your ETL and ELT pipelines.

A major Impala goal is to make SQL-on-Hadoop operations fast and efficient enough to appeal to new categories of users and open up Hadoop to new types of use cases. Where practical, it makes use of existing Apache Hive infrastructure that many Hadoop users already have in place to perform long-running, batch-oriented SQL queries.In particular,Impala keeps its table definitions in a traditional MySQL or PostgreSQL database known as the **metastore**, the same database where Hive keeps this type of data. Thus, Impala can access tables defined or loaded by Hive, as long as all columns use Impala-supported data types, file formats, and compression codecs as shown in **Fig.1**.The initial focus on query features and performance means that Impala can read more types of data with the `SELECT` statement than it can write with the `INSERT` statement. To query data using the Avro, RCFile, or SequenceFile file formats, you load the data using Hive.

The Impala query optimizer can also make use of table statistics and column statistics. Originally, you gathered this information with the `ANALYZE TABLE` statement in Hive; in Impala and higher, use the Impala `COMPUTE STATS` statement instead. `COMPUTE STATS` requires less setup, is more reliable and faster, and does not require switching back and forth between impala-shell and the Hive shell.

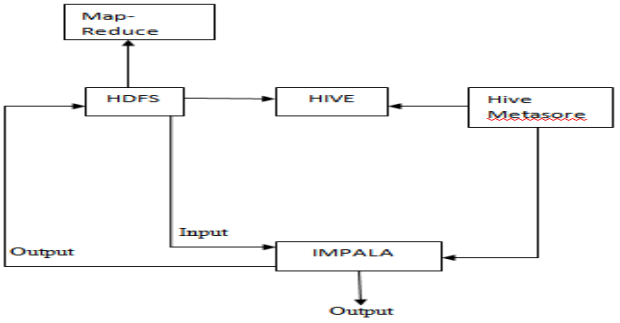

Fig.1.Relationship of Impala with hive

3.Proposed System

In this paper,we have three datasets with the same data model.The first dataset labeled as D1 having 1 year of airline data stored in it and the next two datasets labeled as D2,D3 containing 2 and 3 years of airline data respectively.Each year of airline data has approximately 70-80 lakh rows of data records.

There are different perfomance factors that will be determine the result namely:

1.data set file size; 2.query statements; 3.Data replication factor; 4.HDFS block size;

5.query average time;

A.Hadoop Environment

In the Hadoop environment,there is one hadoop name node,four hadoop data nodes,one Sqoop,one Hive,one pig and one Impala as shown in **Fig.2**.Sqoop helps in getting the data from MYSQL Server and imports it directly to Hadoop Distributed File System(HDFS).Sqoop is designed for efficiently transferring bulk data between hadoop and

relational databases such as MYSQL.Data replication factor in the HDFS configuration is set to 3.

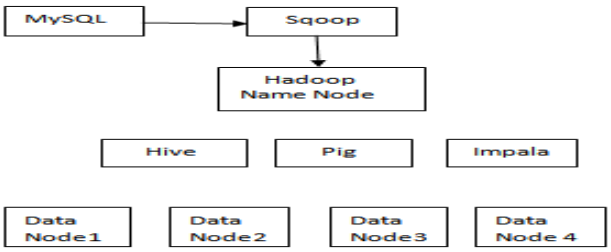

Fig.2.Hadoop environment

B.MySQL Cluster Environment

MySQL cluster has one management node,four data nodes and one MySQL server as the application node shown in Fig.3.Each node is deployed one machine.The replica factor in MySQL cluster is set to 2 ,i.e it will create two node groups. The diagram illustrates a MySQL Cluster with four data nodes, arranged in two node groups of two nodes each; nodes 1 and 2 belong to node group 0, and nodes 3 and 4 belong to node group 1. Note that only data (**ndbd**) nodes are shown here; although a working cluster requires an **ndb_mgm** process for cluster management and at least one SQL node to access the data stored by the cluster.The replica number is set to 2 because this is the minimum requirement to make MySQl cluster to prevent a single point of failure[5].

C.Data sets

The data tester uses a data set from ASA sections on:statistical computing statistical Graphics Data expo '09[11].The data originally came from RITA and is described with every details.These files have derivable variables removed, are packaged in yearly chunks and have been more heavily compressed than the originals.Each file describes airline data of a year using 23 attributes and contains approximately 75-80 lakh rows of data records.

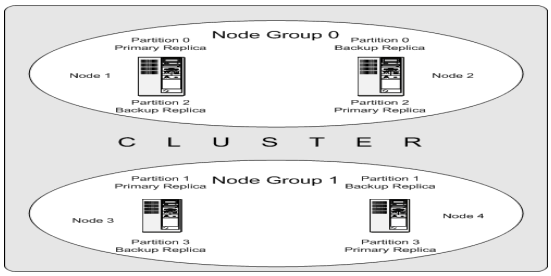

Fig.3.MySQL Cluster environment

D.Query Statments

There are ten queries executed on MySQL and Hive,Pig and Impala on Hadoop.Each query is run three times to get the average time of execution.The list of query statements that are executed are shown in Table I.

TABLE I. Query Statements

Query	Description
Q1	Report the average arrival delay for each year on monthly basis
Q2	Average flights cancelled monthly
Q3	Report state wise average arrival delay
Q4	Report max arrival delay for each year
Q5	Find the airport which has frequent arrival Delays
Q6	Find which Origin and destination combination having more Delays
Q7	List top ten arrival delays of the year
Q8	Find the average CRS Elapsed Time for each Airport
Q9	Find flight number that recorded frequent Arrival delays
Q10	Find the distance range that is having more arrival delays

4.Experimental Results

Experiment is done on Oracle VM VirtualBox.Each system is run on top of Red Hat Operating system with 2 processors core and 4GB RAM.

A.MYSQL Cluster Results

Queries listed in Table I are run on MySQL Cluster on datasets D1,D2,D3 and the results are shown in Table II and the mean result of MySQL cluster processing is shown in Fig 4.

TABLE II.D1,D2 and D3 query time taken on MySQL Cluster

I- Data Set D1					II-Data Set D2				
D1	Attempt 1	Attempt 2	Attempt 3	Mean	D2	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	23.84	19.02	16.59	19.81	Q1	46.66	42.30	34.35	41.10
Q2	4.86	5.87	6.76	5.83	Q2	29.58	29.12	31.96	30.22
Q3	23.27	19.06	13.11	18.48	Q3	43.68	37.04	37.17	39.29
Q4	12.73	7.03	6.68	8.81	Q4	39.07	36.07	34.96	36.70
Q5	15.81	9.51	6.69	10.67	Q5	37.84	38.82	36.57	37.74
Q6	14.19	10.60	7.96	10.92	Q6	40.69	39.85	40.57	40.37
Q7	10.80	8.16	8.21	9.06	Q7	37.48	35.01	33.41	35.30
Q8	10.50	7.40	7.50	8.46	Q8	43.45	42.14	38.79	41.46

Q9	9.09	5.95	5.64	6.89	Q9	29.80	29.46	26.82	28.69
Q10	4.89	4.15	4.63	4.56	Q10	30.01	28.57	26.87	28.48

III-Data Set D3

D3	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	131.28	121.33	109.85	124.15
Q2	81.18	81.93	78.85	80.65
Q3	113.81	110.85	112.95	112.54
Q4	106.09	104.46	109.70	106.75
Q5	158.70	126.17	123.66	128.84
Q6	111.09	102.92	112.18	108.73
Q7	113.61	109.41	104.76	109.26
Q8	123.65	121.44	115.73	120.27
Q9	78.58	77.74	80.91	79.08
Q10	84.02	78.81	75.51	79.45

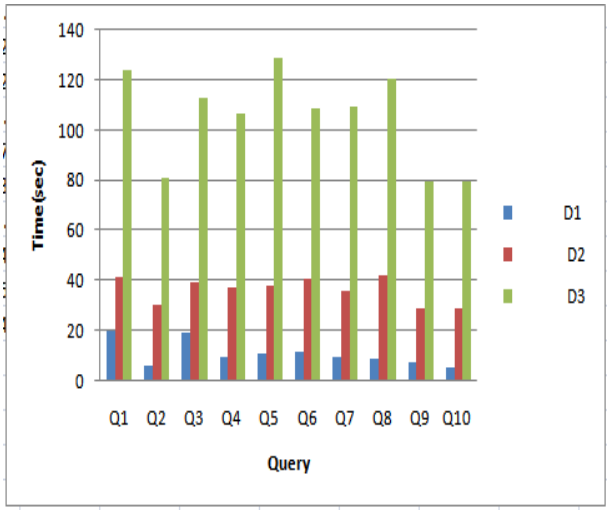

Fig 4.Mean query MySQL Cluster from D1,D2 and D3

As the data grow larger,query processing time in MySQL Cluster increases respectively.MySQL Cluster is distributed and shared data that has a set of computers,each running one or more processes and these nodes are all connected by network.However,it causes a cost on network access when accessing data between the MySQL server and tables distributed across data nodes.To execute the query, data must be retrieved from all data nodes and it may result in a delay[6].There are query performance issues due to sequential access to the storage engine and Unique

hash indexes created with `USING HASH` cannot be used for accessing a table if `NULL` is given as part of the key.

B. Hive Result

The following result is a Hive query on datasets D1,D2 and D3 as shown in TableIII and the mean result is shown in Fig 5.Hive on Hadoop makes data processing straight forward and scalable. Hive is a powerful tool to perform queries on large data sets ,well designed tables and queries can greatly improve query speed and reduce processing cost.

TABLE III.D1,D2 and D3 query time taken on Hive

I-Data Set D1

D1	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	71.66	68.21	67.98	69.28
Q2	45.39	45.72	46.89	46
Q3	48.99	49.09	46.78	48.28
Q4	48.92	48.16	45.64	47.57
Q5	34.06	33.71	33.52	33.76
Q6	22.74	22.63	24.60	23.32
Q7	21.26	21.39	21.09	21.24
Q8	37.93	38.80	37.85	38.19
Q9	26.39	28.63	28.56	27.86
Q10	31.75	32.50	31.47	31.90

II-Data Set D2

D2	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	88.12	88.88	86.54	87.84
Q2	54.31	50.44	52.51	52.42
Q3	56.66	59.10	53.40	56.38
Q4	68.89	64.14	63.98	65.67
Q5	74.56	73.03	74.01	73.86
Q6	42.41	40.91	39.82	41.04
Q7	43.684	41.12	36.49	40.43
Q8	41.04	45.84	45.46	44.11
Q9	56.25	56.49	56.17	56.30
Q10	45.64	45.07	43.94	44.88

III-Data Set D3

D3	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	111.16	110.50	112.17	111.27
Q2	116.46	115.37	117.76	116.53
Q3	109.29	108.53	108.20	108.67
Q4	108.37	109.34	107.26	108.32
Q5	123.23	124.93	121.31	123.15
Q6	110.23	106.63	105.58	107.48
Q7	102.93	98.24	95.86	99.01
Q8	52.48	51.24	48.17	50.63
Q9	69.36	70.67	81.03	73.68
Q10	70.27	74.65	74.03	72.98

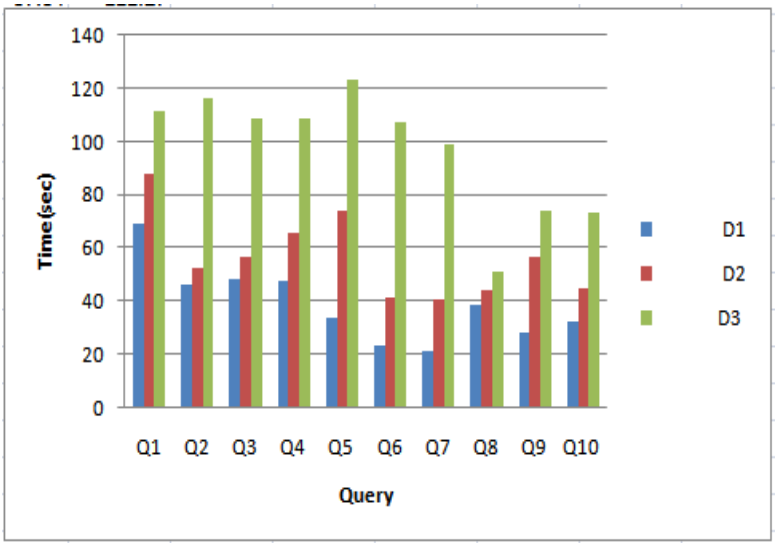

Fig 5.Mean query Hive from D1,D2 and D3

Hive data is stored in HDFS in a plain text file with CSV, as imported by Sqoop.Hive uses indexing mechanism for reading the file faster.If the query have any aggregation,join or sorting function ,hive will immediately start a Map Reduce job[7]. Hadoop can execute map reduce jobs in parallel and several queries executed on Hive make automatically use of this parallelism. However, single, complex Hive queries commonly are translated to a number of map reduce jobs that are executed by default sequentially. Often though some of a query's map reduce stages are not interdependent and could be executed in parallel. They then can take advantage of spare capacity on a cluster and improve cluster utilization while at the same time reduce the overall query executions time.

C. Pig Result

The following result is a Pig query on D1,D2 and D3 data sets as shown in Table IV and the mean result is shown in Fig 6.Pig is a high level procedural language for querying large semi structured data sets so pig did not work well with these data sets which are highly structured.Pig executes a step-by-step approach as defined by the programmer but that doesnot work well with queries that have few aggregations,joins and sorting functions.Due to the step-by-step approach,Pig consumes more time for this data sets[8].

TABLE IV.D1,D2 and D3 query time taken on Pig

I-Data Set D1					II-Data Set D2				
D1	Attempt 1	Attempt 2	Attempt 3	Mean	D2	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	88	86	86	86.6	Q1	110	108	105	107.6
Q2	94	91	90	91.66	Q2	149	151	150	150
Q3	87	85	85	85.66	Q3	114	112	109	111.66
Q4	28	28	27	27.66	Q4	45	45	43	44.33
Q5	24	22	21	22.33	Q5	42	41	42	41.66
Q6	24	23	23	23.33	Q6	41	41	41	41
Q7	94	92	92	92.66	Q7	118	116	114	116

Q8	26	26	26	26		Q8	51	54	52	52.33
Q9	27	28	26	26		Q9	57	55	57	56.33
Q10	28	26	26	26.6		Q10	62	62	61	61.66

III-Data Set D3

D3	Attem pt1	Attem pt2	Attem pt3	Mean
Q1	218	219	224	220.33
Q2	223	223	221	222.33
Q3	232	238	237	235.66
Q4	119	121	123	121
Q5	94	94	92	93.33
Q6	88	89	89	88.66
Q7	296	303	300	299.66
Q8	110	114	118	114
Q9	114	119	121	118
Q10	121	123	127	123.66

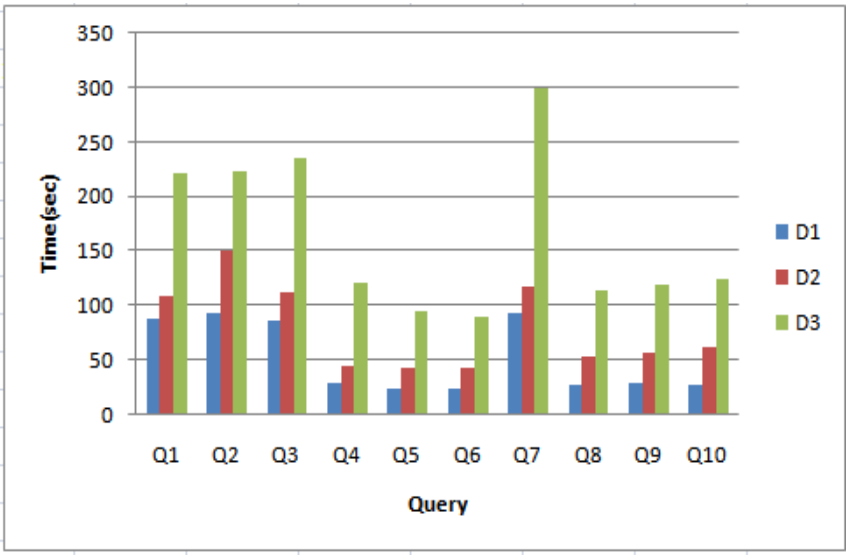

Fig 6.Mean query Pig from D1,D2 and D3

D. Impala Result

The following result is a Impala query on datasets D1,D2 and D3 as shown in Table IV and the mean result is shown in Fig 6.Impala is a native SQL query engine that runs on Hadoop clusters, providing easy query access to raw HDFS data and to HBase databases. Impala is API-compatible with Hive and its ODBC driver. Hive converts/compiles SQL queries into Java-based MapReduce code, which then runs in batch mode, just like other Hadoop tasks.Hive actually adds a step to MapReduce, while Impala replaces MapReduce[9].

Impala provides faster response as it uses MPP(massively parallel processing) unlike Hive which uses MapReduce.Massively parallel processing is a type of computing that uses many separate CPUs running in parallel to execute a single program where each CPU has it's own dedicated memory. The very fact that Impala, being MPP based,without involving the overheads of a MapReduce jobs i.*e* job creation & setup,slot assignment, split creation, map generation etc, which makes it incredibly fast.Being highly memory intensive(MPP), it is not a good fit for tasks that require heavy data operations like joins etc, as you just can't fit everything into the memory but it is idle for real time, ad-hoc queries over a subset of the data[10].

TABLE IV.D1,D2 and D3 query time taken on Impal

I-Data Set D1

D1	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	17.60	16.18	15.07	16.28
Q2	15.06	14.69	12.43	14.06
Q3	14.16	9.38	7.89	10.47
Q4	12.63	8.50	8.19	9.77
Q5	13.57	9.43	8.30	10.43
Q6	10.68	8.02	10.47	9.72
Q7	11.94	8.05	8.30	9.43
Q8	10.73	7.99	8.29	9.00
Q9	11.51	8.53	8.93	9.66
Q10	12.00	8.49	8.24	9.58

II-Data Set D2

D1	Attempt 1	Attempt 2	Attempt 3	Mean
Q1	32.70	31.34	32.14	32.06
Q2	29.75	31.30	28.68	29.91
Q3	31.75	30.28	31.23	31.08
Q4	32.26	30.16	30.35	30.90
Q5	33.01	30.16	29.78	30.98
Q6	32.76	32.43	30.87	32.02
Q7	29.60	30.69	29.75	30.01
Q8	31.03	30.41	30.07	30.50
Q9	29.77	30.56	29.67	30.00
Q10	32.33	33.18	30.20	31.90

III- Data Set D3

D3	Attempt1	Attempt2	Attempt3	Mean
Q1	51.45	47.96	46.54	48.65
Q2	46.84	46.51	46.85	46.73
Q3	49.13	44.40	45.47	46.33
Q4	50.33	49.02	46.86	48.73
Q5	46.84	45.32	45.38	45.85
Q6	49.97	43.51	44.97	46.15

Q7	50.60	45.23	43.86	46.56
Q8	48.28	47.54	47.13	47.65
Q9	48.90	47.43	45.94	47.42
Q10	45.27	44.27	45.01	44.85

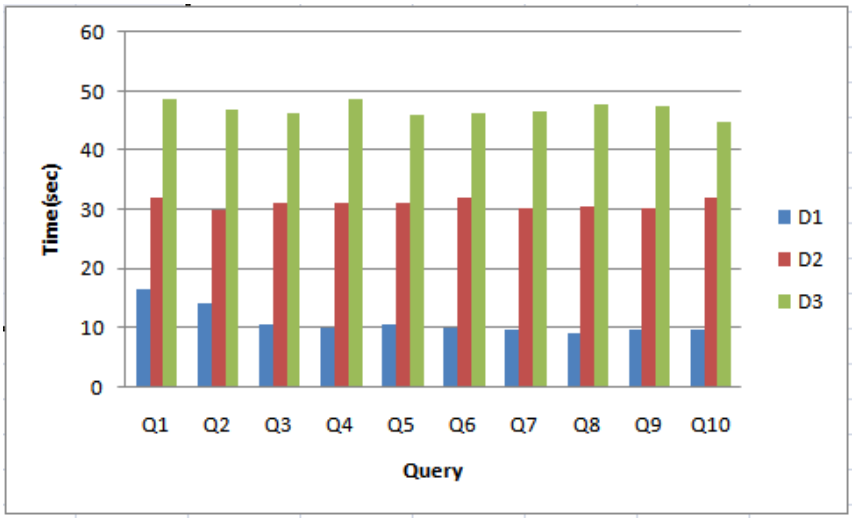

Fig 7.Mean query Impala from D1,D2 and D3

In this data set,Impala overcomes Hive processing time.Fig 7 shows the processing time query in general between MySQL Cluster,Hive,Pig and Impala. MySQL Cluster is faster than Hive at some point but as the data grow larger MySQL needs more time for processing the data whereas Hive can process the data effectively with in less time But Impala is far more effecient than Hive and requires far more less time than Hive for processing the data.On the other hand,Pig is not suitable for this data model and it can perform well when the queries are more complex.

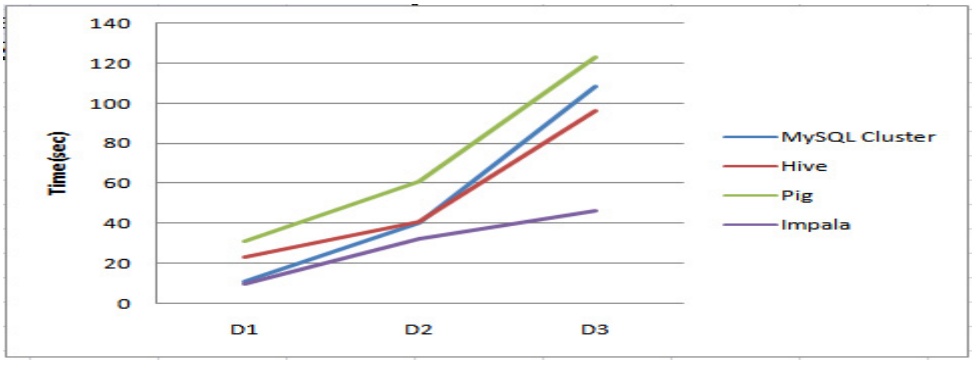

Fig 8.Processing time query in general between MySQL Cluster,Hive,Pig and Impala

5. Conclusions And Future Works

Impala is capable of handling vast amount of data and is more efficient than Hive.Pig is not suitable for this data set and is more suitable for complex queries.Impala is intended to handle real time adhoc queries to handle data exploration and is well-suited to executing SQL queries for interactive exploratory analytics on large data sets.Perfomance of Impala scales with the number of hosts.

However, this is tested on a low-cost hardware. Perfomance may change when better hardware is used for certain software.Performance varies if the number of data nodes increases.
This can be the next future work,by comparing each software perfomance in a better hardware environment and by increasing the number of hosts.

References:

[1]Oracle,MySQL Cluster Evaluation guide.A MySQL Technical White Paper,2013.
[2]Taoying Liu, Jing Liu, Hong Liu, Wei Li.*A Performance Evaluation of Hive for Scientific DataManagement.*
[3] Alan F. Gates, Olga Natkovich, Shubham Chopra, Pradeep Kamath,Shravan M. Narayanamurthy, Christopher Olston, Benjamin Reed,Santhosh Srinivasan, and Utkarsh Srivastava. *Building a high-leveldataflow system on top of Map-Reduce: the Pigexperience.* Proceedings of the VLDB Endowment 2, no. 2 (2009):1414-1425, 2009.
[4]Adrian Bridgwater. *Cloudera Impala: Processing Petabytes at The Speed Of Thought.*
[5]Ammar Fuad, Alva Erwin, Heru Purnomo Ipung. Processing Performance on Apache Pig, Apache Hive and MySQL Cluster.
[6] Andrew Brust . *"Cloudera's Impala brings Hadoop to SQL and BI"*.
[7] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins. *Pig latin: a not-so-foreign language for data processing*. Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 1099-1110. ACM, 2008.
[8] Alan F. Gates, Jianyong Dai, and Thejas Nair. *Apache Pig's Optimizer.* IEEE Data Engineering Bulletin 36, no. 1, 2013.
[9] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. *Hive: a warehousing solution over a map-reduce framework.* Proceedings of the VLDB Endowment 2, no. 2 (2009):1626-1629.
[10] Ronstrom, Mikael, and Lars Thalmann. *MySQL cluster architecture overview.* MySQL Technical White Paper, 2004.
[11] http://stat-computing.org/dataexpo/2009/