UNIVERSITY OF MALAYA

WQD7007 Case Study Report

24/06/2022

GROUP 2

Dr. ALI FEIZOLLAH

ZHANG ZITENG

S2149768

Table of Contents

Question 1:

a. Why big data?

With the rapid development of the data analysis industry, some personal data generated by people in the past are collected, processed and analysed, and this process can be called small data. Small data systems don't just mean small amounts of data. The goal of small data is often the customer's own provision of temporary data to design a specific small objective that is sufficient for a single organization to accomplish through a computer. The objective can be solved simultaneously by processing and analysing highly consequential data. The cost of small data is low, and the recoverability of data damage is strong. However, these data are usually kept for no more than 7 years after the project is completed.

Due to the ever-increasing volume of data, mainstream software tools for small data have been unable to achieve the goal in a suitable time limit. So, the concept of big data was put forward, and currently it has gradually replaced small data. Compared with smaller data, big data is larger on Volume, faster on Velocity, more diverse in Variety, more accurate in Veracity, more continuous in Variability, clearer in visualization, and more valuable of the data products. The task of big data is often to formulate a flexible goal in mind by importing unstructured data sources that may come from any area covered by the Internet. Then follow the steps of data extraction, review, simplification, standardization, transformation, visualization, interpretation and reanalysis to reach the goal. Data sources for big data can be data in many different formats and when the big data project is done, the data will also be stored for longer periods of time. However, big data is relatively expensive and difficult to replicate.

Feature	Small Data	Big Data
Technology	Traditional	Modern
Collection	Obtained in an organized manner than is inserted into the database	Done by using pipelines having queues like AWS Kinesis or Google Pub / Sub to balance high-speed data
Volume	Hundreds of Gigabytes	More than Terabytes
Analysis Areas	Data marts	Clusters and Data marts
Quality	Contains less noise	Not guaranteed
Processing	Batch-oriented	Batch and stream
Database	SQL	NoSQL
Velocity	A regulated and constant flow of data, data aggregation is slow	Data arrives at extremely high speeds, large volumes of data aggregation in a short time
Structure	Structured data in tabular format with fixed schema (Relational)	Numerous varieties of data set including tabular data, text, audio, images, video, logs, JSON etc. (Non-Relational)
Scalability	Vertically scaled	Horizontally scaling architectures
Query Language	only Sequel	Python, R, Java, Sequel
Hardware	A single server is sufficient	Requires more than one server
Value	Business Intelligence, analysis and reporting	Complex data mining techniques for pattern finding, recommendation, prediction etc.
Optimization	Data can be optimized manually (human powered)	Requires machine learning techniques for data optimization
Storage	Storage within enterprises, local servers etc.	Usually requires distributed storage systems on cloud or in external file systems
People	Data Analysts, Database Administrators and Data Engineers	Data Scientists, Data Analysts, Database Administrators and Data Engineers
Security	Security practices for Small Data include user privileges, data encryption, hashing, etc.	Securing Big Data systems are much more complicated. Best security practices include data encryption, cluster network isolation, strong access control protocols etc.
Nomenclature	Database, Data Warehouse, Data Mart	Data Lake
Infrastructure	Predictable resource allocation, mostly vertically scalable hardware.	More agile infrastructure with horizontally scalable hardware

Figure 1 Big data VS Small data

Through the figure 1 online comparison table from “Greeksforgreeks”, it shows the advantages of big data over small data more clearly. So, the given resources consider big data not small data.

b. Customer Churn datasets in Kaggle

Big data can help the telecom industry realize different values according to different data. Such as fraud prevention, customer value prediction, new product development, price optimization, network optimization, etc. There are various types of data about telecommunications. Here we will use the customer churn dataset from Kaggle as the research case for this article.

Maintaining long-term customer relationships and avoiding customer churn is very important in every industry. The telecommunications industry is certainly no exception. Big data can analyse customer behaviour and alert companies to act accordingly. Comprehensive analysis of customer transaction data and real-time communication data can help to gain insight into customers' true satisfaction with the company's services and prevent customer churn.

The dataset I choose including 17 columns and 3150 rows.

Call Failure	Complains	Subscription Length	Charge Amount	Seconds of Use	Frequency of use	Frequency of SMS	Distinct Called Numbers	Age Group	Tariff Plan	Status	Age	Customer Value	FN	FP	Churn
8	0	38	0	4370	71	5	17	3	1	1	30	197.64	177.876	69.764	0
0	0	39	0	318	5	7	4	2	1	2	25	46.035	41.4315	60	0
10	0	37	0	2453	60	359	24	3	1	1	30	1536.52	1382.868	203.652	0
10	0	38	0	4198	66	1	35	1	1	1	15	240.02	216.018	74.002	0
3	0	38	0	2393	58	2	33	1	1	1	15	145.805	131.2245	64.5805	0
11	0	38	1	3775	82	32	28	3	1	1	30	282.28	254.052	78.228	0
4	0	38	0	2360	39	285	18	3	1	1	30	1235.96	1112.364	173.596	0
13	0	37	2	9115	121	144	43	3	1	1	30	945.44	850.896	144.544	0
7	0	38	0	13773	169	0	44	3	1	1	30	557.68	501.912	105.768	0
7	0	38	1	4515	83	2	25	3	1	1	30	191.92	172.728	69.192	0
6	0	38	0	5918	95	7	12	3	1	1	30	268.52	241.668	76.852	0
9	0	38	0	2238	54	8	17	3	1	2	30	123.68	111.312	62.368	0
25	0	38	3	15140	225	54	32	3	1	1	30	830.6	747.54	133.06	0
4	0	38	1	3095	27	483	8	3	1	1	30	2056.88	1851.192	255.688	0
9	0	37	0	15485	182	150	30	2	1	1	25	1380.015	1242.014	188.0015	0
3	0	37	1	6500	86	186	26	3	1	1	30	1007.44	906.696	150.744	0
0	0	37	0	875	14	0	11	2	1	2	25	40.005	36.0045	60	1
2	0	38	0	710	14	13	8	3	1	2	30	80.96	72.864	60	0
0	0	37	0	0	0	0	0	2	1	2	25	0	0	60	1
3	0	37	0	7508	127	384	43	2	1	1	25	2071.575	1864.418	257.1575	0
7	0	37	1	11465	154	11	47	4	1	1	45	317.975	286.1775	81.7975	0
8	0	37	1	6718	75	108	37	2	1	1	25	791.685	712.5165	129.1685	0
23	1	33	0	955	47	16	17	2	1	2	25	117.09	105.381	61.709	1
21	1	36	8	10435	93	1	42	5	2	1	55	159.42	143.478	65.942	0
13	1	36	1	5818	98	26	24	2	1	1	25	383.22	344.898	88.322	1
1	0	34	0	2840	22	0	14	3	1	1	30	114.48	103.032	61.448	0
9	0	35	0	2990	41	9	16	3	1	2	30	157.24	141.516	65.724	1
9	1	36	0	2268	44	34	31	3	1	2	30	228.48	205.632	72.848	1
0	0	36	0	133	2	0	2	3	1	2	30	5.4	4.86	60	1
1	0	36	0	1668	25	0	6	3	1	1	30	67.72	60.948	60	0

Question 2

I choose Hbase as the most suitable method to store telecommunication big data resources. Firstly, Hbase is a column-oriented schema-less semi-structured and structured data store. This means that in the face of real-time data generated by telecom customers, even if it has not been structured yet, Hbase can still adapt and process it well. Second, Hbase is built for low-latency operations and can handle parallel access to billions of data. If telecom adopts Hbase, even though telecom may generate hundreds of millions of data streams every day, Hbase can still easily process it in parallel without incurring high cost. In addition, the operation commands of Hbase are very simple and convenient, such as sell, java, etc., which can be well compatible. The most important thing is that Hbase can be integrated with hadoop. Through the distributed storage of HDFS, work efficiency can be improved, fault tolerance and data recovery in case of failure can be improved. Therefore, Hbase is very suitable for telecom data processing.

For sure, Hbase also has some disadvantages. First, it is impossible to completely replace the traditional model with Hbase. For example, HBase may not be able to perform SQL-like functions, nor does it support SQL-structured data. This issue does not have a significant impact in telecommunications. Second, it can cause unpredictable latency when Hbase is integrated with MapReduce. Also, in a shared cluster environment, when there is a single point of failure. The entire cluster may be affected and cannot continue to work. Data analysis work in telecommunications is interruptible. Even if the system is paralyzed for a short time due to a single point of failure, it will not bring significant losses to the company. From a configuration point of view, Hbase requires high-performance memory-side configuration machines and hardware. It is undeniable that Hbase has some shortcomings. However, these shortcomings don't have a major impact on telcos.

Question 3

The dataset from question 1 I selected totally include 16 columns. I divided them by the following four family groups:

info: Distinct Called Numbers, Age Group, Status, Age

sub: Subscription Length, Charge Amount, Tariff Plan

exp: Call Failure, Complains, Seconds of Use, Frequency of use, Frequency of SMS

pred: Customer Value, FN, FP, Churn

To store dataset and access in Hbase, first need to upload the csv dataset to hdfs. So, I used mkdir command to create a hbase folder in hdfs first and then use put command to upload the dataset from local to hdfs. As you can view in first screen shoot.

```
student@student-VirtualBox:/home/WQD7007$ hadoop/bin/hadoop fs -put /home/student/Downloads/tele_churn.csv /hbase
```

Next is to create a table in hbase. So, I used hbase shell to open hbase and create a table called 'telecom' with four families, which are 'info', 'sub', 'exp', and 'pred'.

```
hbase(main):011:0> create_namespace 'tel'
0 row(s) in 0.0810 seconds

hbase(main):012:0> create 'tel:telcom','info','sub','exp','pred'
0 row(s) in 2.4780 seconds

=> Hbase::Table - tel:telcom
hbase(main):013:0>
```

After I created the table, as you can view in the following diagram, I used 'describe' command can clearly check the table attributes.

```
hbase(main):015:0> describe 'tel:telcom'
Table tel:telcom is ENABLED
tel:telcom
COLUMN FAMILIES DESCRIPTION
{NAME => 'exp', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLO
CKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'info', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BL
OCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'pred', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BL
OCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'sub', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLO
CKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
4 row(s) in 0.0690 seconds
```

Then I used ImportTsv lib to import the data in hdfs to the table we created. Columns are followed by ',' and different columns I set as different family. As shown in the following diagram.

```
student@student-VirtualBox:/home/WQD7007$ hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=',' -Dimporttsv.columns=HBASE_ROW_KEY,exp:Call_Failure,exp:Complains,sub:Subscription_Leng
th,sub:Charge_Amount,exp:Seconds_of_Use,exp:Frequency_of_use,exp:Frequency_of_SMS,info:Distinct_Called_Numbers,info:Age_Group,sub:Tariff_Plan,info:Status,info:Age,pred:Customer_Value,pred:FN,pred:FP,pred:
Churn telecom /hbase/tele_churn.csv
```

After the data upload to Hbase table, the data store process in Hbase has been done. It will show the following information. Such as map input records, map output records, bad line etc. If there are some errors or bad lines shown after transferring, there might have some issue with the dataset or the transfer command.

```
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=153216
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=173435
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=2
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=0
        Job Counters
                Launched map tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=5618
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=5618
                Total vcore-seconds taken by all map tasks=5618
                Total megabyte-seconds taken by all map tasks=1438208
        Map-Reduce Framework
                Map input records=3150
                Map output records=3150
                Input split bytes=107
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=167
                CPU time spent (ms)=4700
                Physical memory (bytes) snapshot=217067520
                Virtual memory (bytes) snapshot=1941065728
                Total committed heap usage (bytes)=141557760
        ImportTsv
                Bad Lines=0
        File Input Format Counters
                Bytes Read=173328
        File Output Format Counters
                Bytes Written=0
```

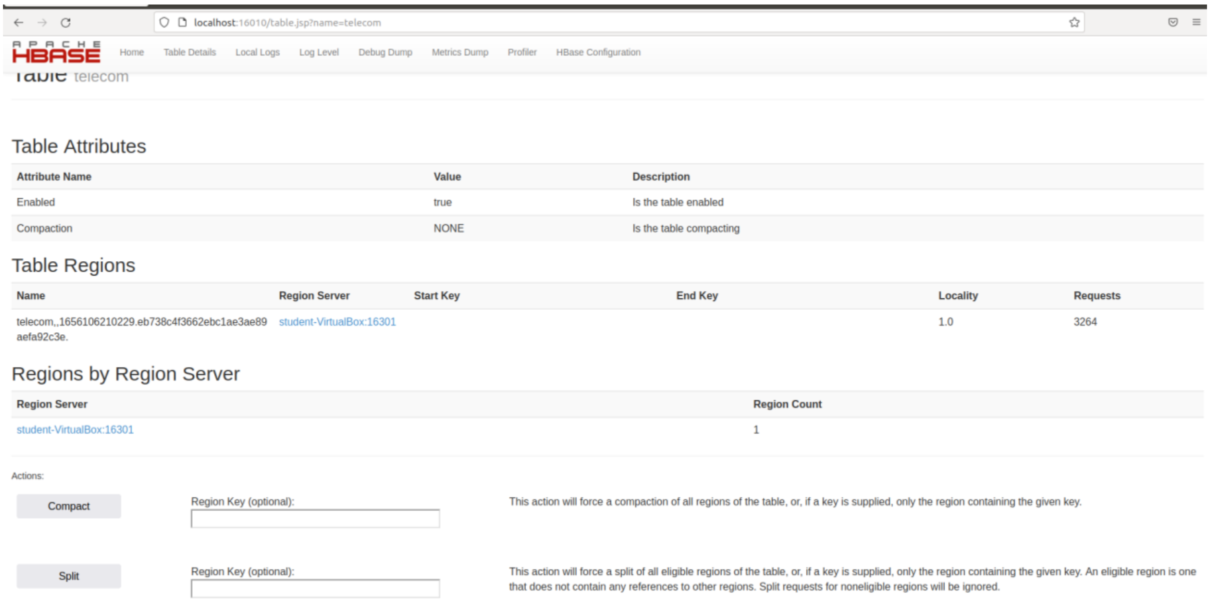
We can go back to hbase and check if the table are still empty. As shown in following diagram, it shows there are 37 rows in the table now. It shows 37 rows because most of the rows are hidden.

```
student@student-VirtualBox:/home/WQD7007$ hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/WQD7007/hbase/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/WQD7007/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.12, r91d5ec4c4dcd10ceec984c6e663ea82acf353995, Sat Apr  6 15:27:28 CDT 2019

hbase(main):001:0> count 'telecom'
37 row(s) in 0.3820 seconds

=> 37
```

To manage hbase, there are many ways. Such as the following example I used scan ‘telecom’ to view the data information for each rows.

```
hbase(main):002:0> scan 'telecom'
ROW                                                   COLUMN+CELL
 0                                                    column=exp:Call_Failure, timestamp=1656106704923, value=0
 0                                                    column=exp:Complains, timestamp=1656106704923, value=27
 0                                                    column=exp:Frequency_of_SMS, timestamp=1656106704923, value=5
 0                                                    column=exp:Frequency_of_use, timestamp=1656106704923, value=3
 0                                                    column=exp:Seconds_of_Use, timestamp=1656106704923, value=5
 0                                                    column=info:Age, timestamp=1656106704923, value=18.125
 0                                                    column=info:Age_Group, timestamp=1656106704923, value=1
 0                                                    column=info:Distinct_Called_Numbers, timestamp=1656106704923, value=4
 0                                                    column=info:Status, timestamp=1656106704923, value=45
 0                                                    column=pred:Customer_Value, timestamp=1656106704923, value=16.3125
 0                                                    column=pred:FN, timestamp=1656106704923, value=60
 0                                                    column=pred:FP, timestamp=1656106704923, value=0
 0                                                    column=sub:Charge_Amount, timestamp=1656106704923, value=420
 0                                                    column=sub:Subscription_Length, timestamp=1656106704923, value=0
 0                                                    column=sub:Tariff_Plan, timestamp=1656106704923, value=2
 1                                                    column=exp:Call_Failure, timestamp=1656106704923, value=0
 1                                                    column=exp:Complains, timestamp=1656106704923, value=10
 1                                                    column=exp:Frequency_of_SMS, timestamp=1656106704923, value=12
 1                                                    column=exp:Frequency_of_use, timestamp=1656106704923, value=197
 1                                                    column=exp:Seconds_of_Use, timestamp=1656106704923, value=35
 1                                                    column=info:Age, timestamp=1656106704923, value=948.8
 1                                                    column=info:Age_Group, timestamp=1656106704923, value=1
 1                                                    column=info:Distinct_Called_Numbers, timestamp=1656106704923, value=3
 1                                                    column=info:Status, timestamp=1656106704923, value=30
 1                                                    column=pred:Customer_Value, timestamp=1656106704923, value=853.92
 1                                                    column=pred:FN, timestamp=1656106704923, value=144.88
 1                                                    column=pred:FP, timestamp=1656106704923, value=0
 1                                                    column=sub:Charge_Amount, timestamp=1656106704923, value=3985
 1                                                    column=sub:Subscription_Length, timestamp=1656106704923, value=1
 1                                                    column=sub:Tariff_Plan, timestamp=1656106704923, value=1
 10                                                   column=exp:Call_Failure, timestamp=1656106704923, value=0
 10                                                   column=exp:Complains, timestamp=1656106704923, value=28
 10                                                   column=exp:Frequency_of_SMS, timestamp=1656106704923, value=9
 10                                                   column=exp:Frequency_of_use, timestamp=1656106704923, value=0
 10                                                   column=exp:Seconds_of_Use, timestamp=1656106704923, value=20
 10                                                   column=info:Age, timestamp=1656106704923, value=12.6
 10                                                   column=info:Age_Group, timestamp=1656106704923, value=1
 10                                                   column=info:Distinct_Called_Numbers, timestamp=1656106704923, value=3
 10                                                   column=info:Status, timestamp=1656106704923, value=30
 10                                                   column=pred:Customer_Value, timestamp=1656106704923, value=11.34
 10                                                   column=pred:FN, timestamp=1656106704923, value=60
 10                                                   column=pred:FP, timestamp=1656106704923, value=0
 10                                                   column=sub:Charge_Amount, timestamp=1656106704923, value=295
 10                                                   column=sub:Subscription_Length, timestamp=1656106704923, value=0
 10                                                   column=sub:Tariff_Plan, timestamp=1656106704923, value=2
```

Also, there are hbase web table management UI. I can use ‘localhost:16010’ to visit the web UI of hbase to manage easier. As you can view the following diagram:



For sure, hbase can scan with conditions. Such as the following example, first 2 rows of FN in pred family.

```
hbase(main):012:0> scan 'telecom', {COLUMN => 'pred:FN',LIMIT=>2}
ROW                                                  COLUMN+CELL
 0                                                   column=pred:FN, timestamp=1656106704923, value=60
 1                                                   column=pred:FN, timestamp=1656106704923, value=144.88
2 row(s) in 0.0210 seconds
```

This example shows a filter usage of Hbase.

```
hbase(main):028:0> scan 'telecom', FILTER=>"RowFilter(=,'substring:11')"
ROW                                                  COLUMN+CELL
 11                                                  column=exp:Call_Failure, timestamp=1656106704923, value=0
 11                                                  column=exp:Complains, timestamp=1656106704923, value=38
 11                                                  column=exp:Frequency_of_SMS, timestamp=1656106704923, value=25
 11                                                  column=exp:Frequency_of_use, timestamp=1656106704923, value=182
 11                                                  column=exp:Seconds_of_Use, timestamp=1656106704923, value=59
 11                                                  column=info:Age, timestamp=1656106704923, value=960.48
 11                                                  column=info:Age_Group, timestamp=1656106704923, value=1
 11                                                  column=info:Distinct_Called_Numbers, timestamp=1656106704923, value=2
 11                                                  column=info:Status, timestamp=1656106704923, value=25
 11                                                  column=pred:Customer_Value, timestamp=1656106704923, value=864.432
 11                                                  column=pred:FN, timestamp=1656106704923, value=146.048
 11                                                  column=pred:FP, timestamp=1656106704923, value=0
 11                                                  column=sub:Charge_Amount, timestamp=1656106704923, value=3085
 11                                                  column=sub:Subscription_Length, timestamp=1656106704923, value=1
 11                                                  column=sub:Tariff_Plan, timestamp=1656106704923, value=1
1 row(s) in 0.0330 seconds
```

There are many kinds of filter that can be used in hbase. As shown in the following table:

PrefixFilter	scan 'telecom',FILTER=>"PrefixFilter('0001')"
KeyOnlyFilter	scan 'telecom',FILTER=>"KeyOnlyFilter()"
RowFilter	scan 'telecom',FILTER=>"RowFilter(=,'substring:0001')"

	scan 'telecom',FILTER=>"RowFilter(>,'binary:0001')"
FirstKeyOnlyFilter	scan 'telecom',FILTER=>"FirstKeyOnlyFilter()"
InclusiveStopFilter	scan 'telecom',{STARTROW=>'0001',FILTER=>"InclusiveStopFilter('binary:0002')"}
FamilyFilter	scan 'telecom',FILTER=>"FamilyFilter(=,'substring:FN')"
QualifierFilter	scan 'telecom',FILTER=>"QualifierFilter(=,'substring:FP')"
ColumnPreFilter	scan 'telecom',FILTER=>"ColumnPreFilter('Customer_value')"
MultipleColumnPrefixFilter	scan 'telecom',FILTER=>"MultipleColumnPrefixFilter('FN','FP')"
ColumnRangeFilter	scan 'telecom',FILTER=>"ColumnRangeFilter('FN',true,'FP',false)"
ValueFilter	scan 'telecom',FILTER=>"ValueFilter(=,'substring:60s')" get 'telecom','0004',FILTER=>"ValueFilter(=,'substring:60s')"
SingleColumnValueFilter	scan 'telecom',FILTER=>"SingleColumnValueFilter('FN','Customer_value',=,'substring:60s')"
SingleColumnValueExcludeFilter	scan 'telecom',FILTER=>"SingleColumnValueFilter('FN','Customer_value',=,'substring:60s')"
ColumnCountGetFilter	get 'telecom','0001',FILTER=>"ColumnCountGetFilter(3)"
TimestampsFilter	scan 'telecom',FILTER=>"TimestampsFilter(1,4)"
InclusiveStopFilter	scan 'telecom',{STARTROW=>'0001',ENDROW=>'0004',FILTER=>"InclusiveStopFilter('binary:0003')"}
PageFilter	scan 'telecom',{STARTROW=>'0001',ENDROW=>'0004',FILTER=>"PageFilter(3)"}

ColumnPaginationFilter	scan 'telecom',{STARTROW=>'0001',ENDROW=>'0004',FILTER=>"ColumnPaginationFilter(2,1)"}

Other than filter, there are some more command such as the following example:

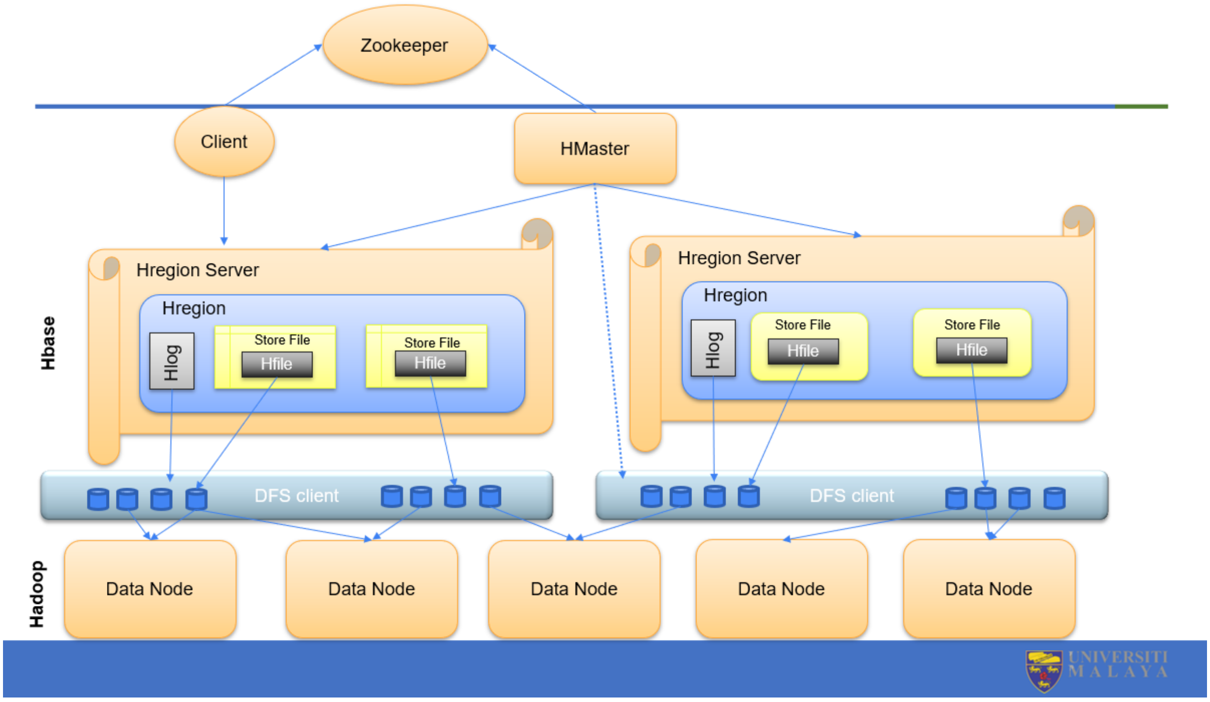
put	put 'telecom',‘row’,‘pred:Customer_Value’,‘0’
delete	delete 'telecom', ‘1’,‘Customer_Value’, ‘1656106704923’
deleteall	deleteall 'telecom', ‘1’
get	get 'telecom',’1’
truncate	truncate 'telecom'
grant	grant student w ['telecom' [pred [<Customer_Value; qualifier>]]
user_permission	user_permission 'telecom'
revoke	revoke student

Due to the page limit, not all the command performed progress is demonstrated in this case study. I have listed some other command of Hbase as you can view in the above two tables.

Also, the outcome of these commands is meaningful based on different requests. Such as grant, it can allocate specific rights of the table to the particular user. It can help telecommunication to protect user privacy, prevent data leakage, tampering by unrelated persons and other security issues. Also, the different kinds of filter can help telecommunication data scientist quickly filter out the user data they need from hundreds of millions of pieces of data. The delete command can help telecommunications achieve the indicated data deletion.

Thus, in my opinion, Hbase is a better choice compare to MongoDB and Relational database in telecommunication sector. In comparison, relational databases require structured data for analysis, and MongoDB's single master node limits the speed of data writing to the database, making it unsuitable for real-time data analysis. And MongoDB does not enable security technologies such as authentication by default. So add a network setting that needs to be configured by the administrator, otherwise it will affect the network connection between MongoDB and the database.

Question 4



There are many database/data warehouse management tools for big data. After the previous discussion, we believe that Hbase is more suitable for the telecom database architecture mainly because of the fast data transmission speed and the huge amount of data that the table can hold. Therefore, I drew the above picture based on my own understanding, which can more clearly depict the data processing pipeline structure of Hbase.

Hbase is a distributed column-oriented open-source database built on HDFS. As shown, the client can maintain some cache to speed up access to Hbase. And Zookeeper is used to ensure that there is only one master in the cluster, and at the same time, the online and offline status of the shared region server is monitored to the master in real time. The tasks of the Hmaster include region allocation, load balancing adjustment of region servers, screening of invalid region servers and redistribution, processing garbage collection in HDFS, and processing schema update requests. The HRegion server maintains the regions assigned to it by the Hmaster to process input/output requests to these regions. The Hregion server is responsible for segmenting regions that become too large during operation. Hmaster does not need to participate in the process of client access to data on Hbase, which refers to the metadata information of why and Hregion, so the load of Hmaster is very low.

Reference

A. (2021a, February 24). *Detailed explanation of HBase basic principle*. Develop Paper. https://developpaper.com/detailed-explanation-of-hbase-basic-principle/

Botelho, B., & Vaughan, J. (2020, August 28). *MongoDB*. SearchDataManagement. https://www.techtarget.com/searchdatamanagement/definition/MongoDB

Customer Churn. (n.d.). Kaggle. Retrieved June 26, 2022, from https://www.kaggle.com/datasets/royjafari/customer-churn

GeeksforGeeks. (2021, September 29). *Difference Between Small Data and Big Data*. Retrieved June 26, 2022, from https://www.geeksforgeeks.org/difference-between-small-data-and-big-data/

HBase Advantages and Disadvantages. (2021, July 14). Programsbuzz. Retrieved June 26, 2022, from https://www.programsbuzz.com/article/hbase-advantages-and-disadvantages

N. (2021, April 8). *Operations using HBase Shell*. Intellipaat Blog. https://intellipaat.com/blog/tutorial/hbase-tutorial/hbase-commands/

Team, T. (2021, June 16). *Role of Big Data in Telecom Industry with Case Studies*. TechVidvan. Retrieved June 26, 2022, from https://techvidvan.com/tutorials/big-data-in-telecommunication/

Taylor, D. (2022, May 21). *HBase Advantages, Disadvantages & Performance Bottleneck*. Guru99. https://www.guru99.com/hbase-limitations-advantage-problems.html