BIG DATA ANALYTICS

Final Report

Apache Hbase

Lubaina Navaid
10966

Big Data Analytics

Final Project: Report

About The Dataset

The dataset contains e-commerce data for the month of October 2019, from a large, multi-category online store. Each row within the dataset represents a purchase event, with a many-to-many relationship between products and users. This means that each row has to do with a specific product that was purchased as a specific point in time, with the time of purchase being used as the row key. Multiple purchase events can be carried out by the same user.

This dataset was taken from Kaggle.

Use Cases of Big Data in E-commerce

Big data tools can be used in e-commerce for a much deeper analysis than would be possible using traditional methods. Data analytics tools for big data are used for

- Optimizing back-office processes
- Enhancing customer experience
- Streamlining supply chain
- Identifying and preventing fraud
- Improved position in market against competitors due to optimization in pricing
- Demand forecasting

HBase Commands

Setting Up Hbase

Downloading from git

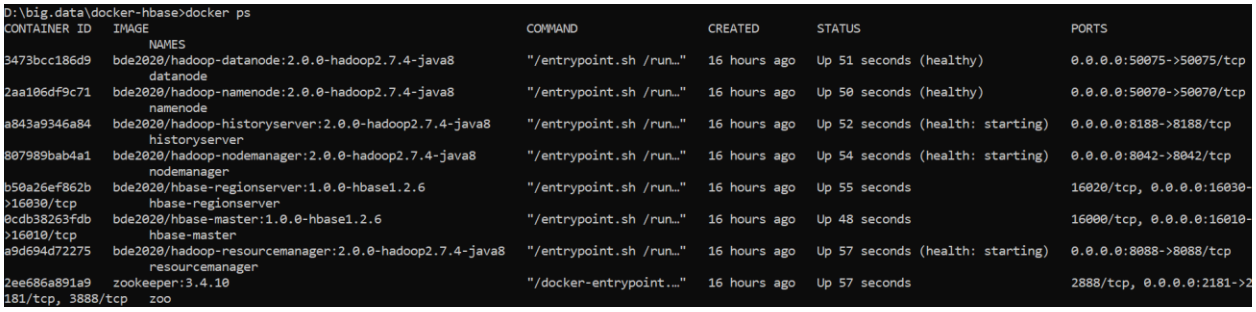
```
Git clone https://github.com/big-data-europe/docker-hbase
```

Using docker compose to set up container:

```
D:\big.data\docker-hbase>docker-compose -f docker-compose-distributed-local.yml up --d
```

Using docker ps to see running containers on network

```
D:\big.data\docker-hbase>docker ps
```



```
D:\big.data\docker-hbase>docker ps
CONTAINER ID   IMAGE                                                    COMMAND                  CREATED        STATUS                               PORTS
                         NAMES
3473bcc186d9   bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8          "/entrypoint.sh /run…"   16 hours ago   Up 51 seconds (healthy)              0.0.0.0:50075->50075/tcp
                         datanode
2aa106df9c71   bde2020/hadoop-namenode:2.0.0-hadoop2.7.4-java8          "/entrypoint.sh /run…"   16 hours ago   Up 50 seconds (healthy)              0.0.0.0:50070->50070/tcp
                         namenode
a843a9346a84   bde2020/hadoop-historyserver:2.0.0-hadoop2.7.4-java8     "/entrypoint.sh /run…"   16 hours ago   Up 52 seconds (health: starting)     0.0.0.0:8188->8188/tcp
                         historyserver
807989bab4a1   bde2020/hadoop-nodemanager:2.0.0-hadoop2.7.4-java8       "/entrypoint.sh /run…"   16 hours ago   Up 54 seconds (health: starting)     0.0.0.0:8042->8042/tcp
                         nodemanager
b50a26ef862b   bde2020/hbase-regionserver:1.0.0-hbase1.2.6              "/entrypoint.sh /run…"   16 hours ago   Up 55 seconds                        16020/tcp, 0.0.0.0:16030-
>16030/tcp               hbase-regionserver
0cdb38263fdb   bde2020/hbase-master:1.0.0-hbase1.2.6                    "/entrypoint.sh /run…"   16 hours ago   Up 48 seconds                        16000/tcp, 0.0.0.0:16010-
>16010/tcp               hbase-master
a9d694d72275   bde2020/hadoop-resourcemanager:2.0.0-hadoop2.7.4-java8   "/entrypoint.sh /run…"   16 hours ago   Up 57 seconds (health: starting)     0.0.0.0:8088->8088/tcp
                         resourcemanager
2ee686a891a9   zookeeper:3.4.10                                         "/docker-entrypoint.…"   16 hours ago   Up 57 seconds                        2888/tcp, 0.0.0.0:2181->2
181/tcp, 3888/tcp        zoo
```

Copying data into hbase master & namenode containers

D:\big.data\docker-hbase>docker cp 2019-Oct.csv 0cdb38263fdb:/hadoop-data

D:\big.data\docker-hbase>docker cp 2019-Oct.csv 2aa106df9c71:/hadoop-data

executing hbase master:

D:\big.data\docker-hbase>docker exec -it 0cdb38263fdb /bin/bash

Entering hbase shell:

root@hbase-master: hbase shell

Viewing tables:

hbase(main):005:0> list

Creating table:

hbase(main):006:0> create ‘ecommerce-data’,’event_type’,’event’,’product’,’category’,’brand_price’,’user’

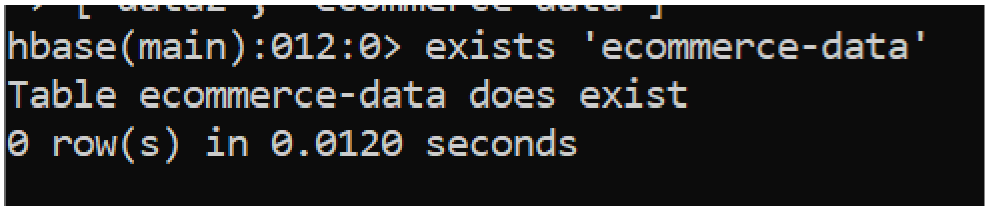
Importing data from container to hbase

root@hbase-master:/# hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=',' -Dimporttsv.columns=HBASE_ROW_KEY,event:event_type,product:product_id,category:category_id,category:category_code,brand_price:brand,brand_price:price,user:user_id,user:user_session ecommerce-data /hadoop-data/2019-Oct.csv

Basic Commands

Check if table exists

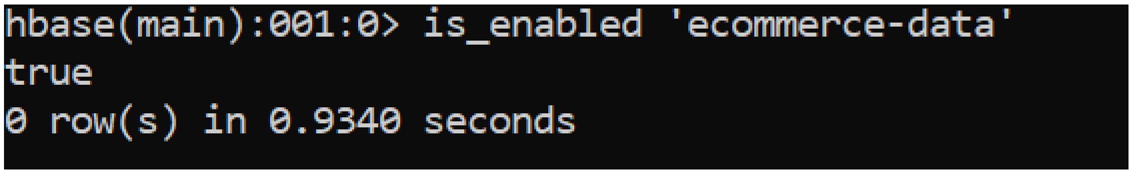
hbase(main):012:0> exists 'ecommerce-data'



```
hbase(main):012:0> exists 'ecommerce-data'
Table ecommerce-data does exist
0 row(s) in 0.0120 seconds
```

Check if table is enabled

hbase(main):001:0> Is_enabled ‘ecommerce-data’



```
hbase(main):001:0> is_enabled 'ecommerce-data'
true
0 row(s) in 0.9340 seconds
```

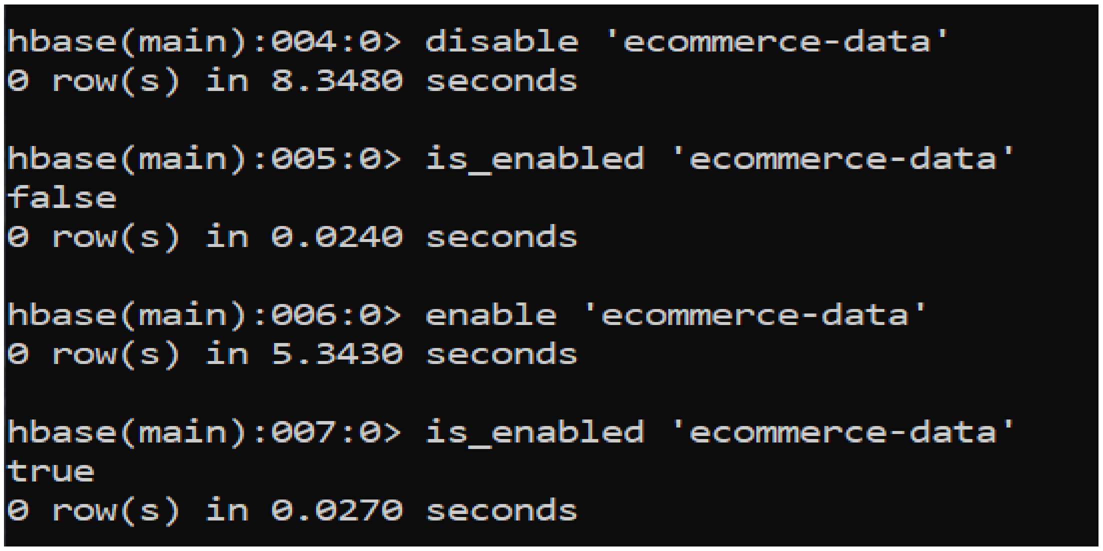
Disable and re-enable table

hbase(main):004:0> disable 'ecommerce-data'

hbase(main):005:0> is_enabled 'ecommerce-data'

hbase(main):006:0> enable 'ecommerce-data'

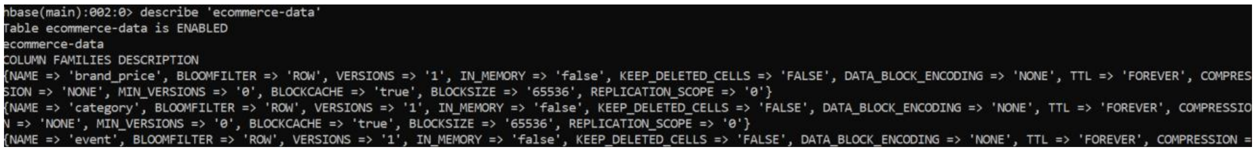
hbase(main):007:0> is_enabled 'ecommerce-data'



```
hbase(main):004:0> disable 'ecommerce-data'
0 row(s) in 8.3480 seconds

hbase(main):005:0> is_enabled 'ecommerce-data'
false
0 row(s) in 0.0240 seconds

hbase(main):006:0> enable 'ecommerce-data'
0 row(s) in 5.3430 seconds

hbase(main):007:0> is_enabled 'ecommerce-data'
true
0 row(s) in 0.0270 seconds
```

Describe table

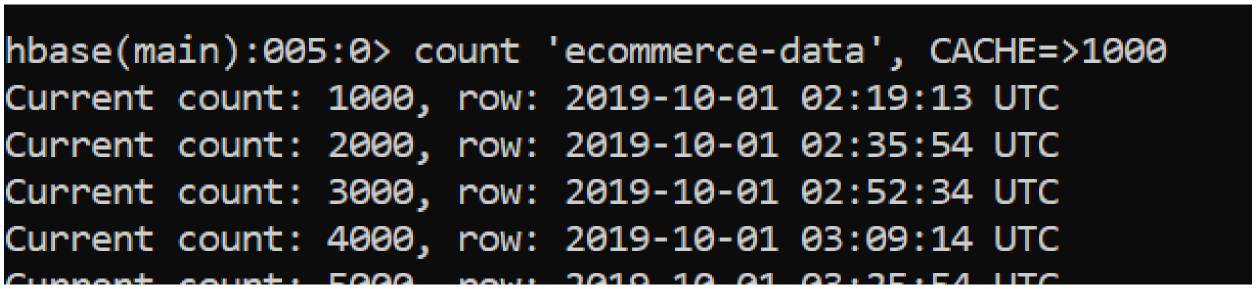
hbase(main):002:0>describe ‘ecommerce-data’



```
hbase(main):002:0> describe 'ecommerce-data'
Table ecommerce-data is ENABLED
ecommerce-data
COLUMN FAMILIES DESCRIPTION
{NAME => 'brand_price', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRES
SION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'category', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSIO
N => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'event', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION =
```

Counting values within table

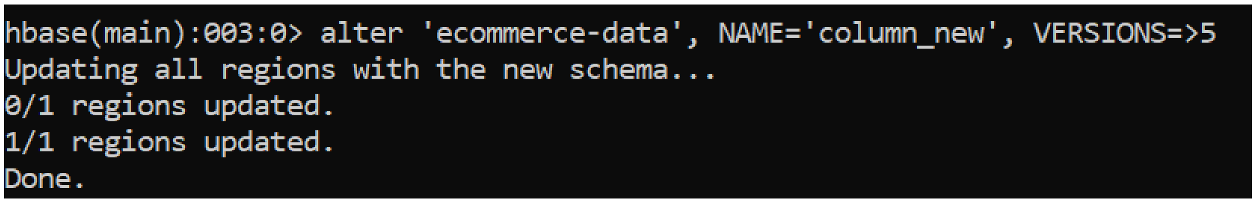
hbase(main):005:0>Count ‘ecommerce-data’



```
hbase(main):005:0> count 'ecommerce-data', CACHE=>1000
Current count: 1000, row: 2019-10-01 02:19:13 UTC
Current count: 2000, row: 2019-10-01 02:35:54 UTC
Current count: 3000, row: 2019-10-01 02:52:34 UTC
Current count: 4000, row: 2019-10-01 03:09:14 UTC
```

Altering table to add new column called ‘column_new’

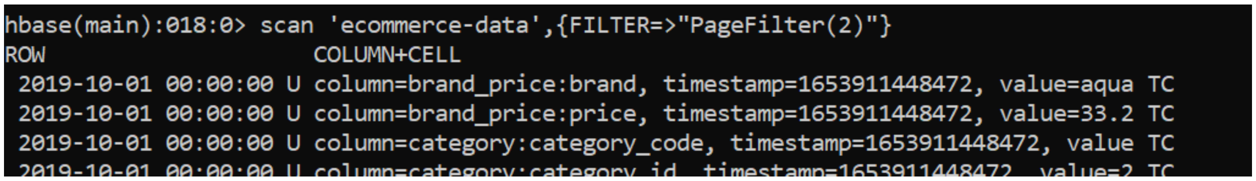
hbase(main):003:0>alter ‘ecommerce-data’, NAME=’column_new’, VERSIONS=>5



```
hbase(main):003:0> alter 'ecommerce-data', NAME='column_new', VERSIONS=>5
Updating all regions with the new schema...
0/1 regions updated.
1/1 regions updated.
Done.
```

Scanning table to check if imported correctly

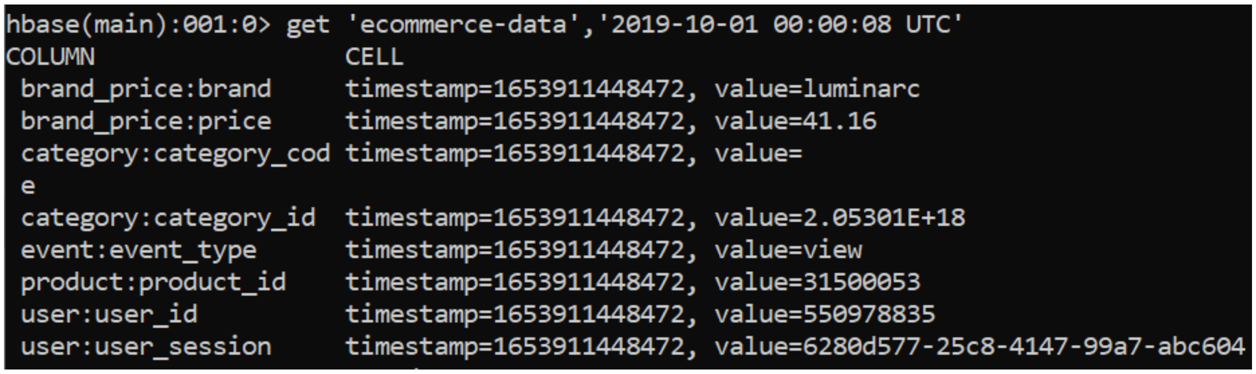
hbase(main):018:0> scan 'ecommerce-data',{FILTER=>"PageFilter(2)"}



```
hbase(main):018:0> scan 'ecommerce-data',{FILTER=>"PageFilter(2)"}
ROW                   COLUMN+CELL
 2019-10-01 00:00:00 U column=brand_price:brand, timestamp=1653911448472, value=aqua TC
 2019-10-01 00:00:00 U column=brand_price:price, timestamp=1653911448472, value=33.2 TC
 2019-10-01 00:00:00 U column=category:category_code, timestamp=1653911448472, value TC
```

Getting data from row number ‘2019-01-01 00:00:08 UTC’

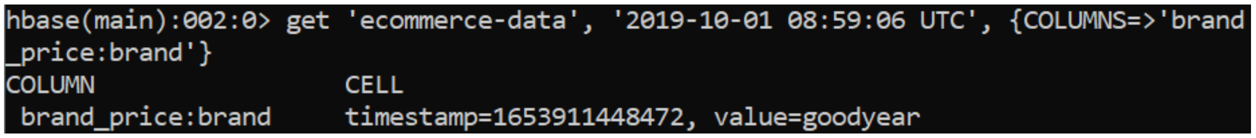
hbase(main):001:0> get 'ecommerce-data','2019-10-01 00:00:08 UTC'



```
hbase(main):001:0> get 'ecommerce-data','2019-10-01 00:00:08 UTC'
COLUMN                CELL
 brand_price:brand    timestamp=1653911448472, value=luminarc
 brand_price:price    timestamp=1653911448472, value=41.16
 category:category_cod timestamp=1653911448472, value=
 e
 category:category_id  timestamp=1653911448472, value=2.05301E+18
 event:event_type     timestamp=1653911448472, value=view
 product:product_id   timestamp=1653911448472, value=31500053
 user:user_id         timestamp=1653911448472, value=550978835
 user:user_session    timestamp=1653911448472, value=6280d577-25c8-4147-99a7-abc604
```

Getting data from column ‘brand’ in column family brand_price for ‘2019-10-01 08:59:06 UTC'

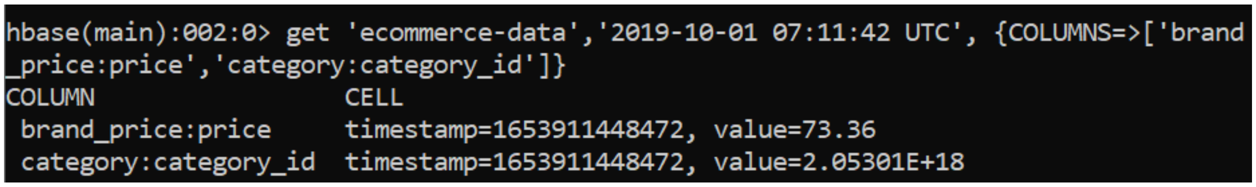
hbase(main):002:0> get 'ecommerce-data', '2019-10-01 08:59:06 UTC',
{COLUMNS=>'brand_price:brand'}



```
hbase(main):002:0> get 'ecommerce-data', '2019-10-01 08:59:06 UTC', {COLUMNS=>'brand
_price:brand'}
COLUMN                 CELL
 brand_price:brand     timestamp=1653911448472, value=goodyear
```

Getting data from columns ‘price’ and ‘category_id’ for '2019-10-01 07:11:42 UTC'

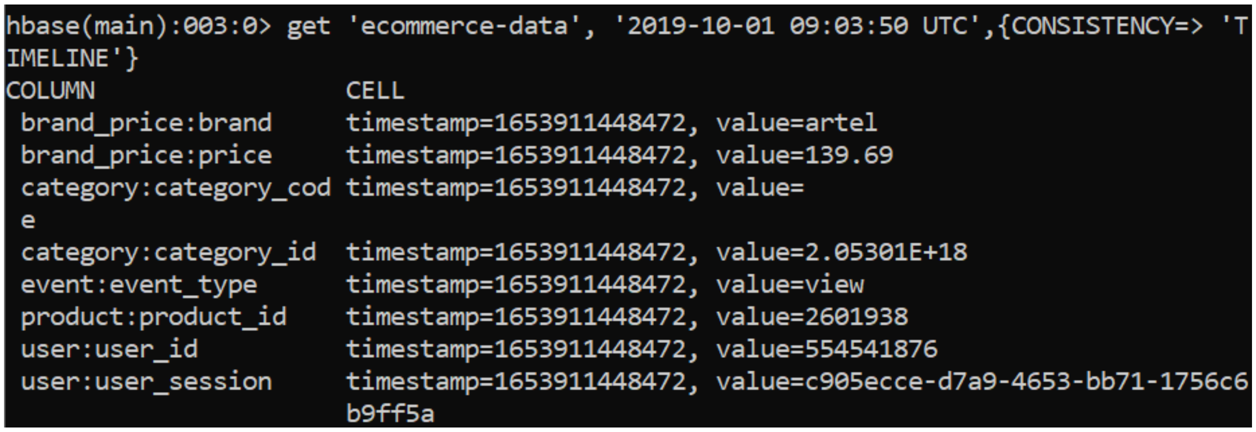
hbase(main):002:0> get 'ecommerce-data','2019-10-01 07:11:42 UTC', {COLUMNS=>['brand_price:price','category:category_id']}



```
hbase(main):002:0> get 'ecommerce-data','2019-10-01 07:11:42 UTC', {COLUMNS=>['brand
_price:price','category:category_id']}
COLUMN                 CELL
 brand_price:price     timestamp=1653911448472, value=73.36
 category:category_id  timestamp=1653911448472, value=2.05301E+18
```

Getting timeline and values for row '2019-10-01 09:03:50 UTC'

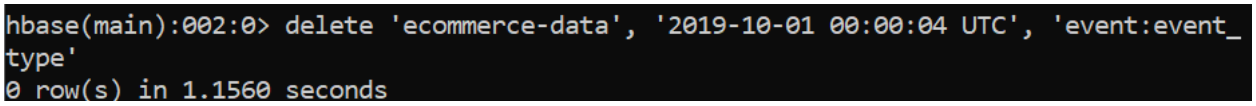
hbase(main):003:0> get 'ecommerce-data', '2019-10-01 09:03:50 UTC',{CONSISTENCY=> 'TIMELINE'}



```
hbase(main):003:0> get 'ecommerce-data', '2019-10-01 09:03:50 UTC',{CONSISTENCY=> 'T
IMELINE'}
COLUMN                 CELL
 brand_price:brand     timestamp=1653911448472, value=artel
 brand_price:price     timestamp=1653911448472, value=139.69
 category:category_cod timestamp=1653911448472, value=
 e
 category:category_id  timestamp=1653911448472, value=2.05301E+18
 event:event_type      timestamp=1653911448472, value=view
 product:product_id    timestamp=1653911448472, value=2601938
 user:user_id          timestamp=1653911448472, value=554541876
 user:user_session     timestamp=1653911448472, value=c905ecce-d7a9-4653-bb71-1756c6
                       b9ff5a
```

Deleting row ‘2019-10-01 00:00:04 UTC’ from table

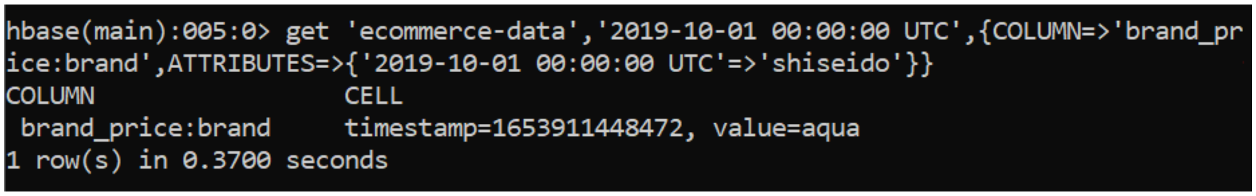
hbase(main):002:0> delete 'ecommerce-data', '2019-10-01 00:00:04 UTC', 'event:event_type'



```
hbase(main):002:0> delete 'ecommerce-data', '2019-10-01 00:00:04 UTC', 'event:event_
type'
0 row(s) in 1.1560 seconds
```

Get value in row ‘2019-10-01 00:00:00 UTC' where value ‘shiseido’ is included

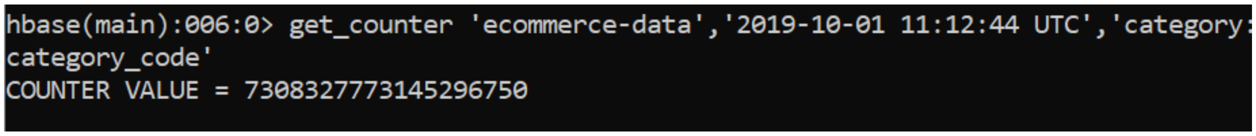
hbase(main):005:0> get 'ecommerce-data','2019-10-01 00:00:00 UTC',{COLUMN=>'brand_price:brand',ATTRIBUTES=>{'2019-10-01 00:00:00 UTC'=>'shiseido'}}



```
hbase(main):005:0> get 'ecommerce-data','2019-10-01 00:00:00 UTC',{COLUMN=>'brand_pr
ice:brand',ATTRIBUTES=>{'2019-10-01 00:00:00 UTC'=>'shiseido'}}
COLUMN                 CELL
 brand_price:brand     timestamp=1653911448472, value=aqua
1 row(s) in 0.3700 seconds
```

Get counter cell value for row ‘2019-10-01 11:12:44 UTC', column ‘category_code’ in column family ‘category’

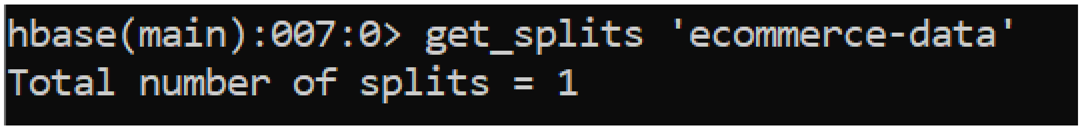
hbase(main):006:0> get_counter 'ecommerce-data','2019-10-01 11:12:44 UTC','category:category_code'



```
hbase(main):006:0> get_counter 'ecommerce-data','2019-10-01 11:12:44 UTC','category:
category_code'
COUNTER VALUE = 7308327773145296750
```

See total number of splits in data on hbase by region

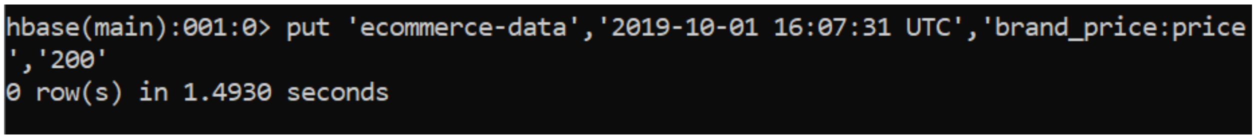
hbase(main):007:0> get_splits 'ecommerce-data'



```
hbase(main):007:0> get_splits 'ecommerce-data'
Total number of splits = 1
```

Put value ‘200’ in price column for row '2019-10-01 16:07:31 UTC'

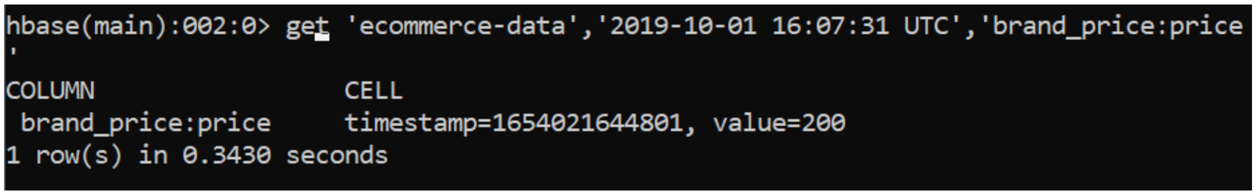
hbase(main):001:0> put 'ecommerce-data','2019-10-01 16:07:31 UTC','brand_price:price','200'



```
hbase(main):001:0> put 'ecommerce-data','2019-10-01 16:07:31 UTC','brand_price:price
','200'
0 row(s) in 1.4930 seconds
```

Verifying put command using get command

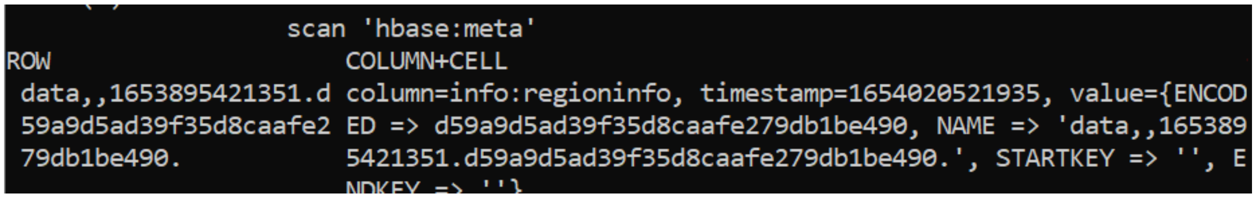
hbase(main):002:0> get 'ecommerce-data','2019-10-01 16:07:31 UTC','brand_price:price'



```
hbase(main):002:0> get 'ecommerce-data','2019-10-01 16:07:31 UTC','brand_price:price
'
COLUMN                 CELL
 brand_price:price     timestamp=1654021644801, value=200
1 row(s) in 0.3430 seconds
```

Perform meta scan on table

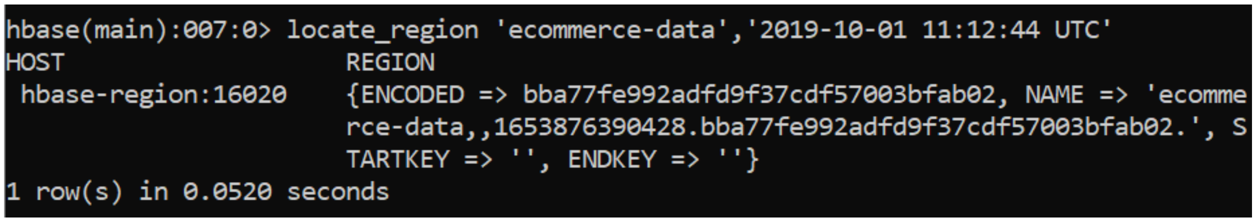
hbase(main):003:0> scan 'hbase:meta'



```
                   scan 'hbase:meta'
ROW                    COLUMN+CELL
 data,,1653895421351.d column=info:regioninfo, timestamp=1654020521935, value={ENCOD
 59a9d5ad39f35d8caafe2 ED => d59a9d5ad39f35d8caafe279db1be490, NAME => 'data,,165389
 79db1be490.           5421351.d59a9d5ad39f35d8caafe279db1be490.', STARTKEY => '', E
                       NDKEY => ''}
```

Locating region of given row

hbase(main):007:0> locate_region 'ecommerce-data','2019-10-01 11:12:44 UTC'

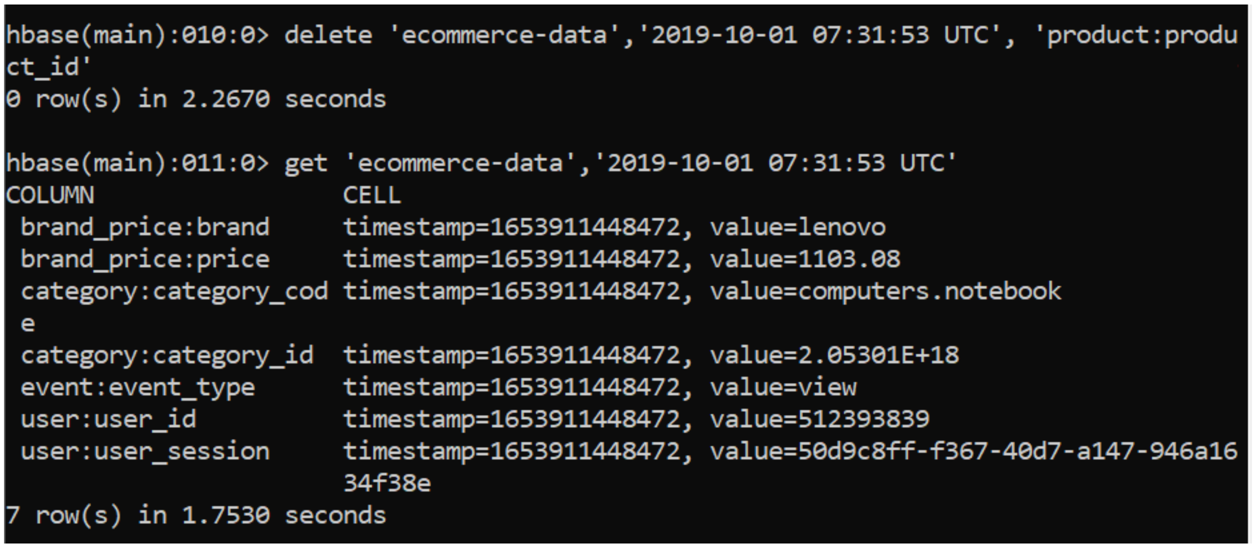


```
hbase(main):007:0> locate_region 'ecommerce-data','2019-10-01 11:12:44 UTC'
HOST                    REGION
 hbase-region:16020     {ENCODED => bba77fe992adfd9f37cdf57003bfab02, NAME => 'ecomme
                        rce-data,,1653876390428.bba77fe992adfd9f37cdf57003bfab02.', S
                        TARTKEY => '', ENDKEY => ''}
1 row(s) in 0.0520 seconds
```

Deleting cell value of column product_id in row '2019-10-01 07:31:53 UTC' and verifying deleting using get command

hbase(main):010:0> delete 'ecommerce-data','2019-10-01 07:31:53 UTC', 'product:product_id'

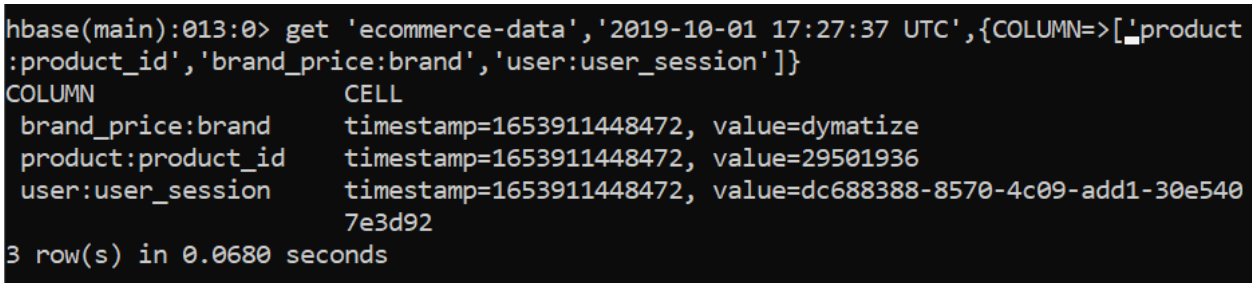
hbase(main):011:0> get 'ecommerce-data','2019-10-01 07:31:53 UTC'



```
hbase(main):010:0> delete 'ecommerce-data','2019-10-01 07:31:53 UTC', 'product:produ
ct_id'
0 row(s) in 2.2670 seconds

hbase(main):011:0> get 'ecommerce-data','2019-10-01 07:31:53 UTC'
COLUMN                  CELL
 brand_price:brand      timestamp=1653911448472, value=lenovo
 brand_price:price      timestamp=1653911448472, value=1103.08
 category:category_cod  timestamp=1653911448472, value=computers.notebook
 e
 category:category_id   timestamp=1653911448472, value=2.05301E+18
 event:event_type       timestamp=1653911448472, value=view
 user:user_id           timestamp=1653911448472, value=512393839
 user:user_session      timestamp=1653911448472, value=50d9c8ff-f367-40d7-a147-946a16
                        34f38e
7 row(s) in 1.7530 seconds
```

Get values for columns brand, user session and product_id for row '2019-10-01 17:27:37 UTC'

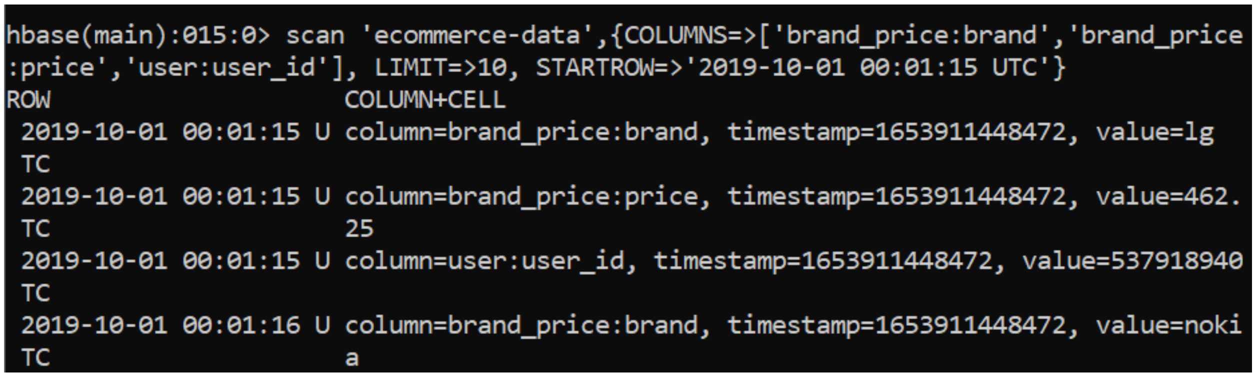
hbase(main):013:0> get 'ecommerce-data','2019-10-01 17:27:37 UTC',{COLUMN=>['product:product_id','brand_price:brand','user:user_session']}



```
hbase(main):013:0> get 'ecommerce-data','2019-10-01 17:27:37 UTC',{COLUMN=>['product
:product_id','brand_price:brand','user:user_session']}
COLUMN                  CELL
 brand_price:brand      timestamp=1653911448472, value=dymatize
 product:product_id     timestamp=1653911448472, value=29501936
 user:user_session      timestamp=1653911448472, value=dc688388-8570-4c09-add1-30e540
                        7e3d92
3 row(s) in 0.0680 seconds
```

Scan table for columns brand, price and user_id starting from row '2019-10-01 00:01:15 UTC' with limit 10

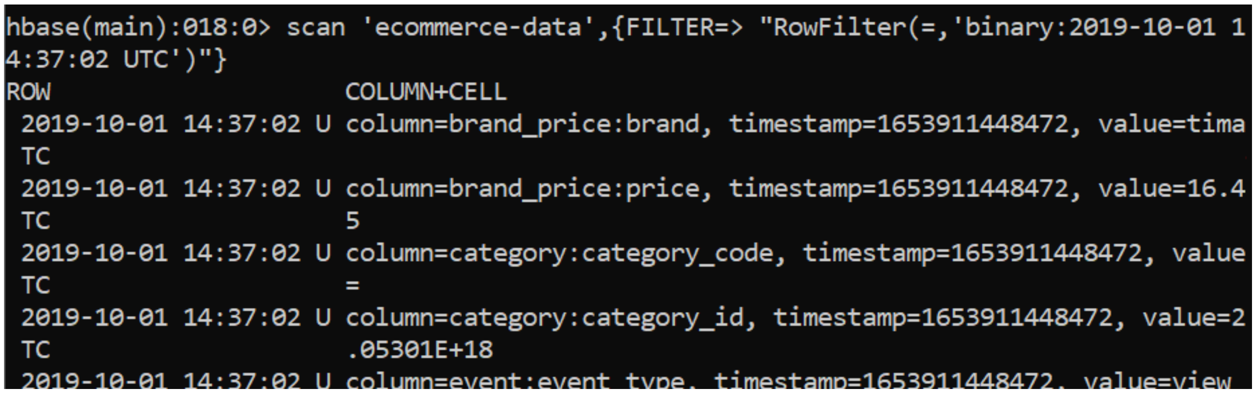
hbase(main):015:0> scan 'ecommerce-data',{COLUMNS=>['brand_price:brand','brand_price:price','user:user_id'], LIMIT=>10, STARTROW=>'2019-10-01 00:01:15 UTC'}



```
hbase(main):015:0> scan 'ecommerce-data',{COLUMNS=>['brand_price:brand','brand_price
:price','user:user_id'], LIMIT=>10, STARTROW=>'2019-10-01 00:01:15 UTC'}
ROW                      COLUMN+CELL
 2019-10-01 00:01:15 U column=brand_price:brand, timestamp=1653911448472, value=lg
 TC
 2019-10-01 00:01:15 U column=brand_price:price, timestamp=1653911448472, value=462.
 TC                      25
 2019-10-01 00:01:15 U column=user:user_id, timestamp=1653911448472, value=537918940
 TC
 2019-10-01 00:01:16 U column=brand_price:brand, timestamp=1653911448472, value=noki
 TC                      a
```

Scan table for row ‘2019-10-01 14:37:02 UTC' using RowFilter

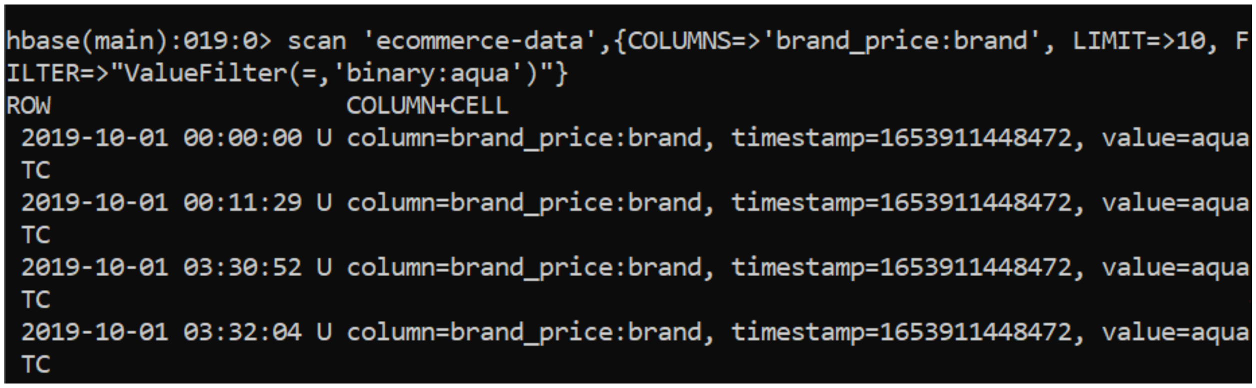
hbase(main):018:0> scan 'ecommerce-data',{FILTER=> "RowFilter(=,'binary:2019-10-01 14:37:02 UTC')"}



```
hbase(main):018:0> scan 'ecommerce-data',{FILTER=> "RowFilter(=,'binary:2019-10-01 1
4:37:02 UTC')"}
ROW                      COLUMN+CELL
 2019-10-01 14:37:02 U column=brand_price:brand, timestamp=1653911448472, value=tima
 TC
 2019-10-01 14:37:02 U column=brand_price:price, timestamp=1653911448472, value=16.4
 TC                      5
 2019-10-01 14:37:02 U column=category:category_code, timestamp=1653911448472, value
 TC                      =
 2019-10-01 14:37:02 U column=category:category_id, timestamp=1653911448472, value=2
 TC                      .05301E+18
 2019-10-01 14:37:02 U column=event:event_type, timestamp=1653911448472, value=view
```

Scan table for rows where brand=aqua with limit of 10 using value filter

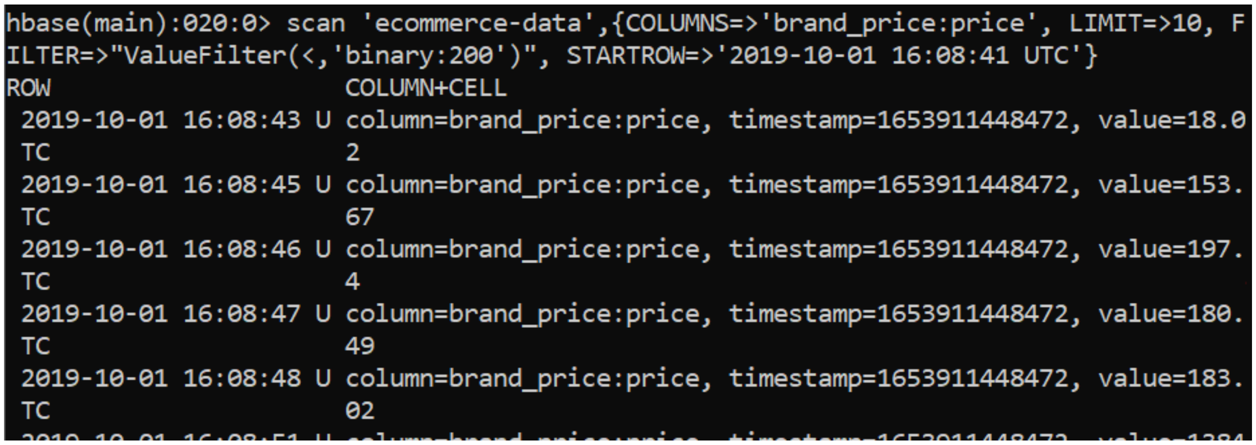
hbase(main):019:0> scan 'ecommerce-data',{COLUMNS=>'brand_price:brand', LIMIT=>10, FILTER=>"ValueFilter(=,'binary:aqua')"}



```
hbase(main):019:0> scan 'ecommerce-data',{COLUMNS=>'brand_price:brand', LIMIT=>10, F
ILTER=>"ValueFilter(=,'binary:aqua')"}
ROW                      COLUMN+CELL
 2019-10-01 00:00:00 U column=brand_price:brand, timestamp=1653911448472, value=aqua
 TC
 2019-10-01 00:11:29 U column=brand_price:brand, timestamp=1653911448472, value=aqua
 TC
 2019-10-01 03:30:52 U column=brand_price:brand, timestamp=1653911448472, value=aqua
 TC
 2019-10-01 03:32:04 U column=brand_price:brand, timestamp=1653911448472, value=aqua
 TC
```

Scan table for rows where price is less than 200 starting from row '2019-10-01 16:08:41 UTC', using ValueFilter

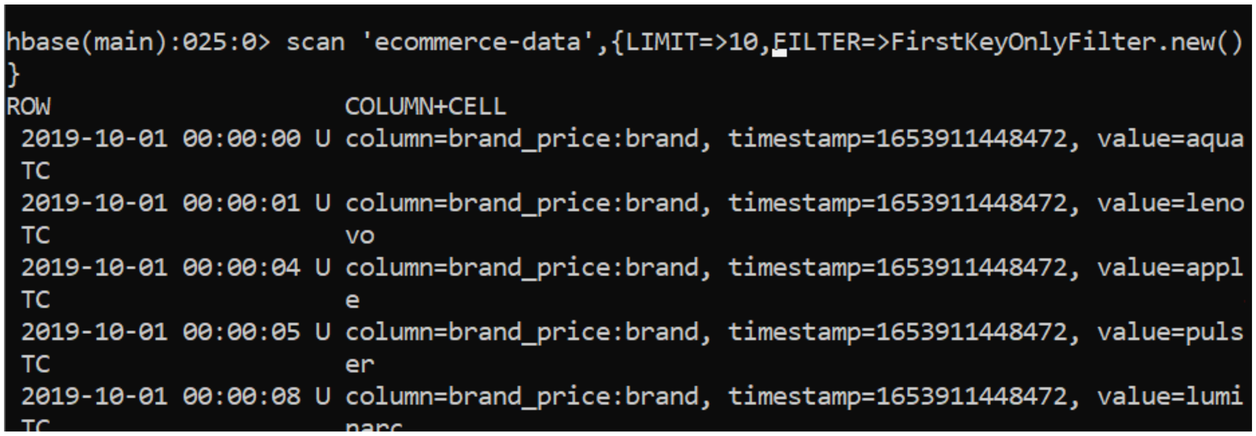
hbase(main):020:0> scan 'ecommerce-data',{COLUMNS=>'brand_price:price', LIMIT=>10, FILTER=>"ValueFilter(<,'binary:200')", STARTROW=>'2019-10-01 16:08:41 UTC'}



```
hbase(main):020:0> scan 'ecommerce-data',{COLUMNS=>'brand_price:price', LIMIT=>10, F
ILTER=>"ValueFilter(<,'binary:200')", STARTROW=>'2019-10-01 16:08:41 UTC'}
ROW                      COLUMN+CELL
 2019-10-01 16:08:43 U column=brand_price:price, timestamp=1653911448472, value=18.0
 TC                      2
 2019-10-01 16:08:45 U column=brand_price:price, timestamp=1653911448472, value=153.
 TC                      67
 2019-10-01 16:08:46 U column=brand_price:price, timestamp=1653911448472, value=197.
 TC                      4
 2019-10-01 16:08:47 U column=brand_price:price, timestamp=1653911448472, value=180.
 TC                      49
 2019-10-01 16:08:48 U column=brand_price:price, timestamp=1653911448472, value=183.
 TC                      02
```

Scan table for first key using FirstKeyOnlyFilter limiting rows to 10

hbase(main):025:0> scan 'ecommerce-data',{LIMIT=>10,FILTER=>FirstKeyOnlyFilter.new()}



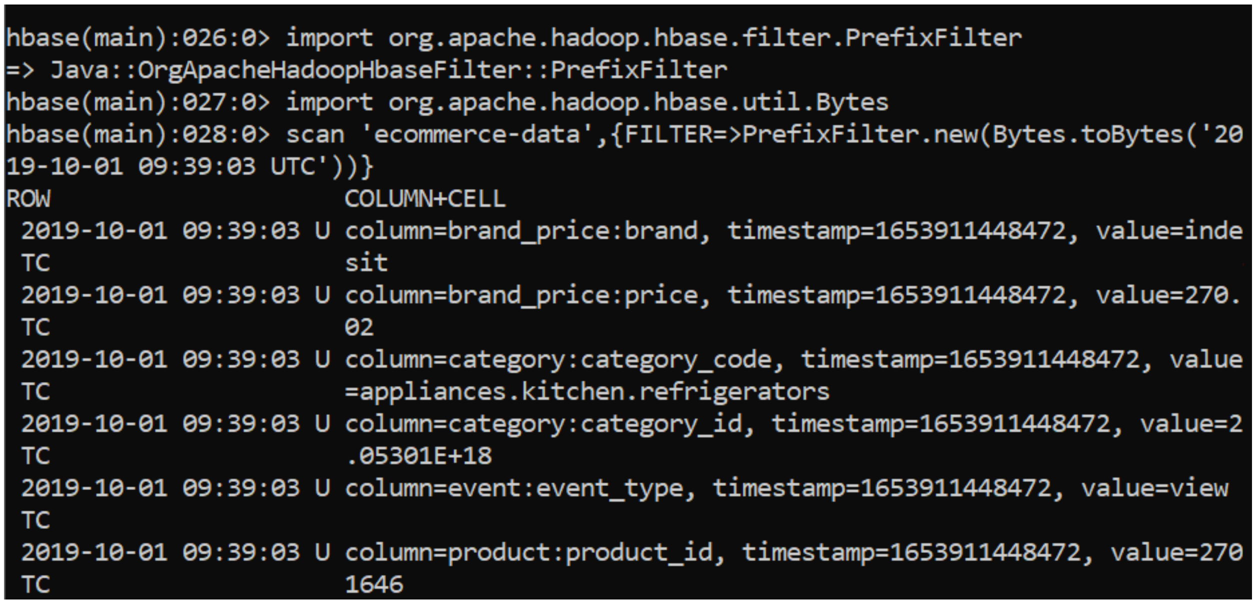
```
hbase(main):025:0> scan 'ecommerce-data',{LIMIT=>10,FILTER=>FirstKeyOnlyFilter.new()
}
ROW                      COLUMN+CELL
 2019-10-01 00:00:00 U column=brand_price:brand, timestamp=1653911448472, value=aqua
 TC
 2019-10-01 00:00:01 U column=brand_price:brand, timestamp=1653911448472, value=leno
 TC                      vo
 2019-10-01 00:00:04 U column=brand_price:brand, timestamp=1653911448472, value=appl
 TC                      e
 2019-10-01 00:00:05 U column=brand_price:brand, timestamp=1653911448472, value=puls
 TC                      er
 2019-10-01 00:00:08 U column=brand_price:brand, timestamp=1653911448472, value=lumi
```

Import PrefixFilter and Bytes utility, then scan table using prefix filter for row '2019-10-01 09:39:03 UTC'

hbase(main):026:0> import org.apache.hadoop.hbase.filter.PrefixFilter

hbase(main):027:0> import org.apache.hadoop.hbase.util.Bytes

hbase(main):028:0> scan 'ecommerce-data',{FILTER=>PrefixFilter.new(Bytes.toBytes('2019-10-01 09:39:03 UTC'))}

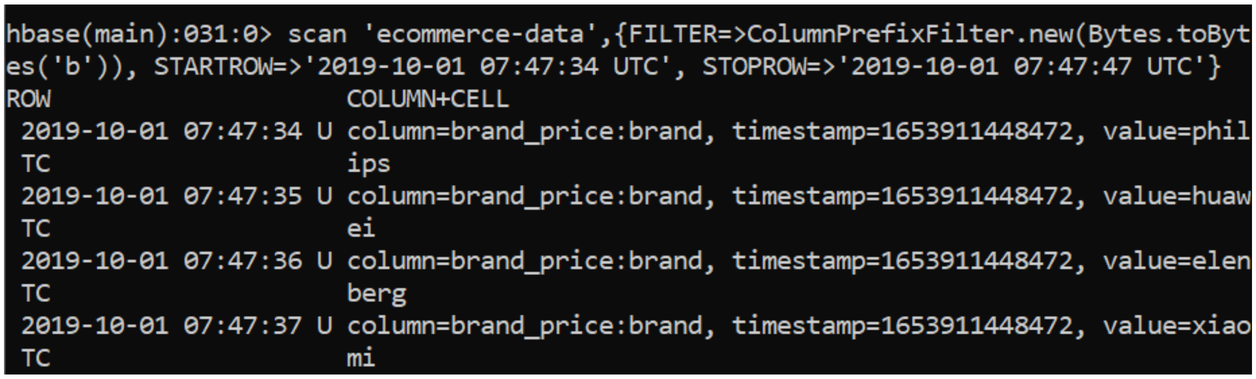


```
hbase(main):026:0> import org.apache.hadoop.hbase.filter.PrefixFilter
=> Java::OrgApacheHadoopHbaseFilter::PrefixFilter
hbase(main):027:0> import org.apache.hadoop.hbase.util.Bytes
hbase(main):028:0> scan 'ecommerce-data',{FILTER=>PrefixFilter.new(Bytes.toBytes('20
19-10-01 09:39:03 UTC'))}
ROW                      COLUMN+CELL
 2019-10-01 09:39:03 U column=brand_price:brand, timestamp=1653911448472, value=inde
 TC                    sit
 2019-10-01 09:39:03 U column=brand_price:price, timestamp=1653911448472, value=270.
 TC                    02
 2019-10-01 09:39:03 U column=category:category_code, timestamp=1653911448472, value
 TC                    =appliances.kitchen.refrigerators
 2019-10-01 09:39:03 U column=category:category_id, timestamp=1653911448472, value=2
 TC                    .05301E+18
 2019-10-01 09:39:03 U column=event:event_type, timestamp=1653911448472, value=view
 TC
 2019-10-01 09:39:03 U column=product:product_id, timestamp=1653911448472, value=270
 TC                    1646
```

Import ColumnPrefixFilter and scan table to get values of columns that start with ‘b’, from row '2019-10-01 07:47:34 UTC' to row '2019-10-01 07:47:47 UTC'

hbase(main):029:0> import org.apache.hadoop.hbase.filter.ColumnPrefixFilter

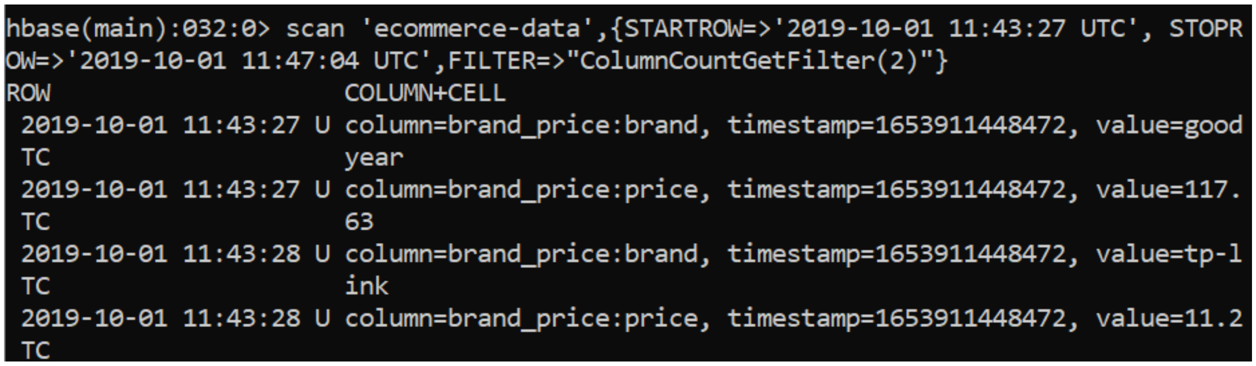
hbase(main):031:0> scan 'ecommerce-data',{FILTER=>ColumnPrefixFilter.new(Bytes.toBytes('b')), STARTROW=>'2019-10-01 07:47:34 UTC', STOPROW=>'2019-10-01 07:47:47 UTC'}



```
hbase(main):031:0> scan 'ecommerce-data',{FILTER=>ColumnPrefixFilter.new(Bytes.toByt
es('b')), STARTROW=>'2019-10-01 07:47:34 UTC', STOPROW=>'2019-10-01 07:47:47 UTC'}
ROW                      COLUMN+CELL
 2019-10-01 07:47:34 U column=brand_price:brand, timestamp=1653911448472, value=phil
 TC                    ips
 2019-10-01 07:47:35 U column=brand_price:brand, timestamp=1653911448472, value=huaw
 TC                    ei
 2019-10-01 07:47:36 U column=brand_price:brand, timestamp=1653911448472, value=elen
 TC                    berg
 2019-10-01 07:47:37 U column=brand_price:brand, timestamp=1653911448472, value=xiao
 TC                    mi
```

Scan table for first 2 columns using ColumnCountGetFilter from row 2019-10-01 11:43:27 UTC' to row '2019-10-01 11:47:04 UTC'

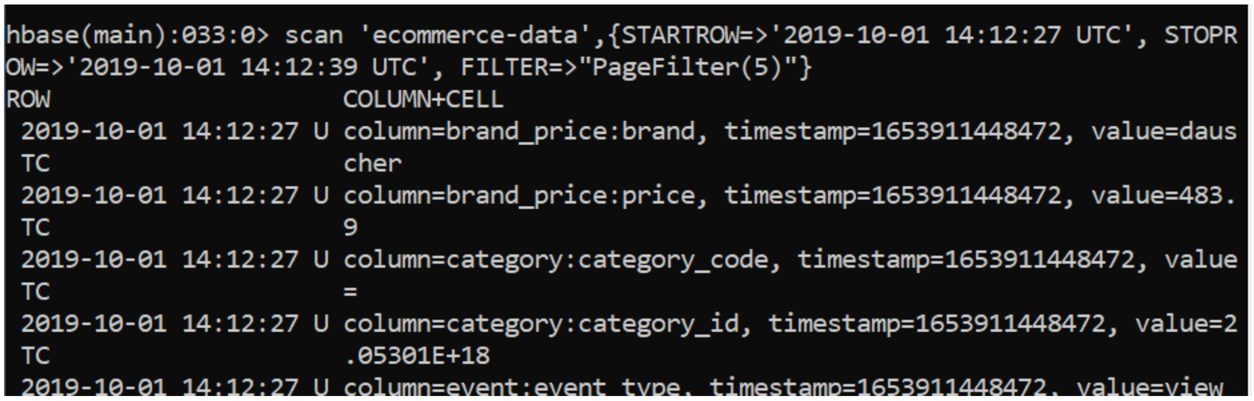
hbase(main):032:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 11:43:27 UTC', STOPROW=>'2019-10-01 11:47:04 UTC',FILTER=>"ColumnCountGetFilter(2)"}



```
hbase(main):032:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 11:43:27 UTC', STOPR
OW=>'2019-10-01 11:47:04 UTC',FILTER=>"ColumnCountGetFilter(2)"}
ROW                      COLUMN+CELL
 2019-10-01 11:43:27 U column=brand_price:brand, timestamp=1653911448472, value=good
 TC                      year
 2019-10-01 11:43:27 U column=brand_price:price, timestamp=1653911448472, value=117.
 TC                      63
 2019-10-01 11:43:28 U column=brand_price:brand, timestamp=1653911448472, value=tp-l
 TC                      ink
 2019-10-01 11:43:28 U column=brand_price:price, timestamp=1653911448472, value=11.2
 TC
```

Scan table for 5 rows using PageFilter starting from row '2019-10-01 14:12:27 UTC' to row '2019-10-01 14:12:39 UTC'

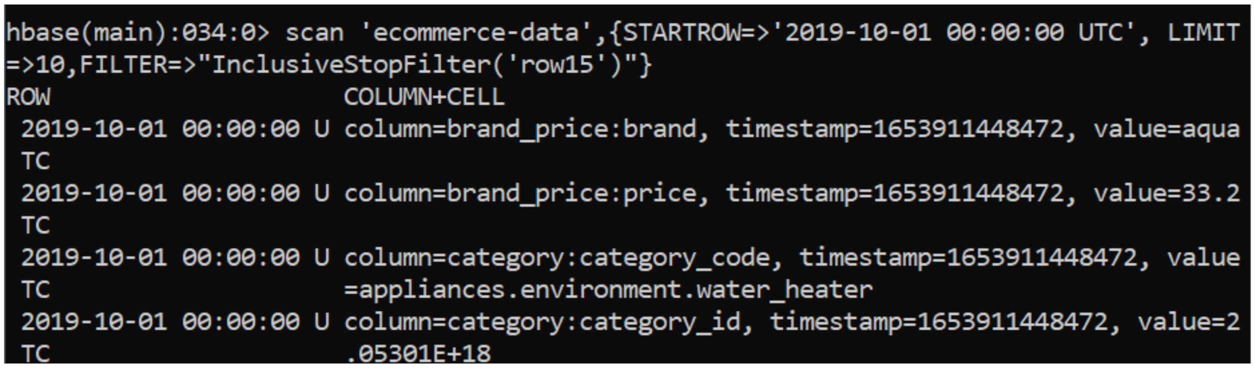
hbase(main):033:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 14:12:27 UTC', STOPROW=>'2019-10-01 14:12:39 UTC', FILTER=>"PageFilter(5)"}



```
hbase(main):033:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 14:12:27 UTC', STOPR
OW=>'2019-10-01 14:12:39 UTC', FILTER=>"PageFilter(5)"}
ROW                      COLUMN+CELL
 2019-10-01 14:12:27 U column=brand_price:brand, timestamp=1653911448472, value=daus
 TC                      cher
 2019-10-01 14:12:27 U column=brand_price:price, timestamp=1653911448472, value=483.
 TC                      9
 2019-10-01 14:12:27 U column=category:category_code, timestamp=1653911448472, value
 TC                      =
 2019-10-01 14:12:27 U column=category:category_id, timestamp=1653911448472, value=2
 TC                      .05301E+18
 2019-10-01 14:12:27 U column=event:event_type, timestamp=1653911448472, value=view
```

Scan table for rows starting from '2019-10-01 00:00:00 UTC' with limit 10 using InclusiveStopFilter

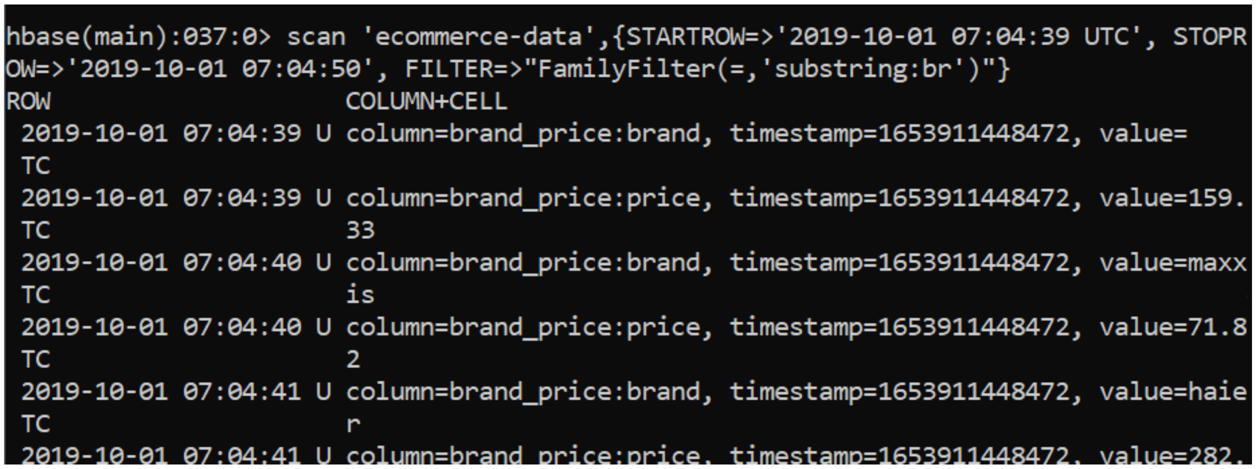
hbase(main):034:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 00:00:00 UTC', LIMIT=>10,FILTER=>"InclusiveStopFilter('row15')"}



```
hbase(main):034:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 00:00:00 UTC', LIMIT
=>10,FILTER=>"InclusiveStopFilter('row15')"}
ROW                      COLUMN+CELL
 2019-10-01 00:00:00 U column=brand_price:brand, timestamp=1653911448472, value=aqua
 TC
 2019-10-01 00:00:00 U column=brand_price:price, timestamp=1653911448472, value=33.2
 TC
 2019-10-01 00:00:00 U column=category:category_code, timestamp=1653911448472, value
 TC                      =appliances.environment.water_heater
 2019-10-01 00:00:00 U column=category:category_id, timestamp=1653911448472, value=2
 TC                      .05301E+18
```

Scan table to find values from column family which includes substring ‘br’ using FamilyFilter

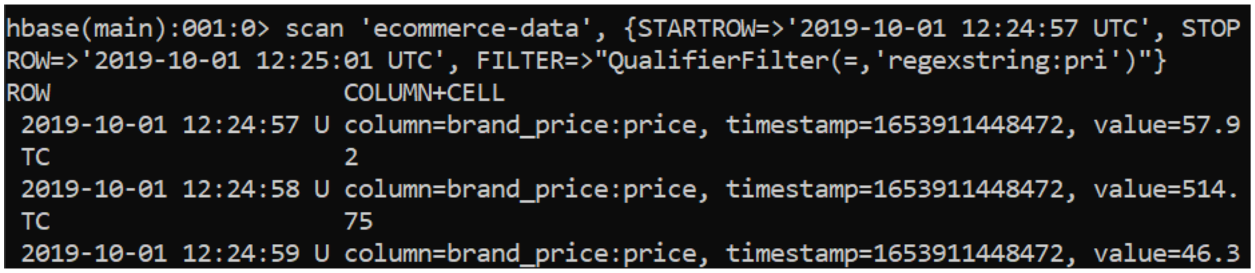
hbase(main):037:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 07:04:39 UTC', STOPROW=>'2019-10-01 07:04:50', FILTER=>"FamilyFilter(=,'substring:br')"}



```
hbase(main):037:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 07:04:39 UTC', STOPR
OW=>'2019-10-01 07:04:50', FILTER=>"FamilyFilter(=,'substring:br')"}
ROW                      COLUMN+CELL
 2019-10-01 07:04:39 U column=brand_price:brand, timestamp=1653911448472, value=
 TC
 2019-10-01 07:04:39 U column=brand_price:price, timestamp=1653911448472, value=159.
 TC                      33
 2019-10-01 07:04:40 U column=brand_price:brand, timestamp=1653911448472, value=maxx
 TC                      is
 2019-10-01 07:04:40 U column=brand_price:price, timestamp=1653911448472, value=71.8
 TC                      2
 2019-10-01 07:04:41 U column=brand_price:brand, timestamp=1653911448472, value=haie
 TC                      r
 2019-10-01 07:04:41 U column=brand_price:price, timestamp=1653911448472, value=282.
```

Scan for key values within column that contains the string 'pri' using QualifierFilter starting from row '2019-10-01 12:24:57 UTC'

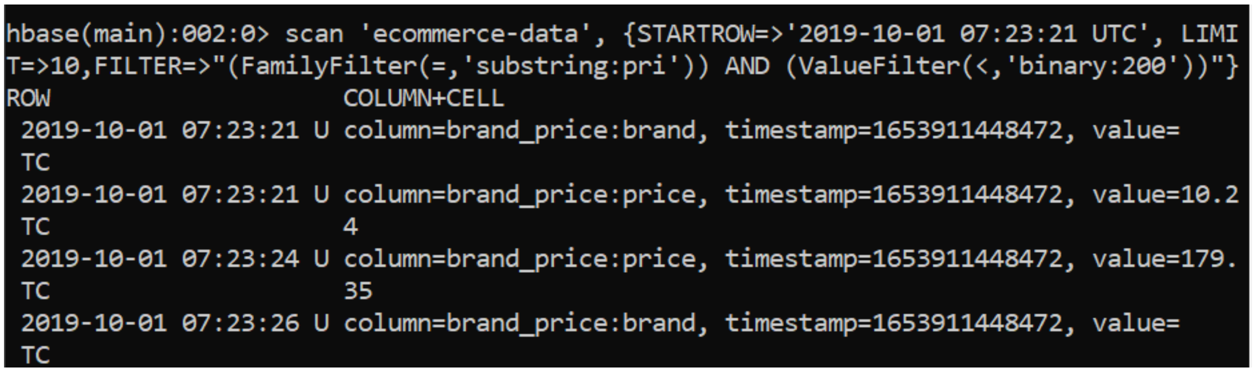
hbase(main):001:0> scan 'ecommerce-data', {STARTROW=>'2019-10-01 12:24:57 UTC', STOPROW=>'2019-10-01 12:25:01 UTC', FILTER=>"QualifierFilter(=,'regexstring:pri')"}



```
hbase(main):001:0> scan 'ecommerce-data', {STARTROW=>'2019-10-01 12:24:57 UTC', STOP
ROW=>'2019-10-01 12:25:01 UTC', FILTER=>"QualifierFilter(=,'regexstring:pri')"}
ROW                      COLUMN+CELL
 2019-10-01 12:24:57 U column=brand_price:price, timestamp=1653911448472, value=57.9
 TC                      2
 2019-10-01 12:24:58 U column=brand_price:price, timestamp=1653911448472, value=514.
 TC                      75
 2019-10-01 12:24:59 U column=brand_price:price, timestamp=1653911448472, value=46.3
```

Scan table for key values within column that includes substring 'pri', and value less than 200 using FamilyFilter, ValueFilter and AND operator.

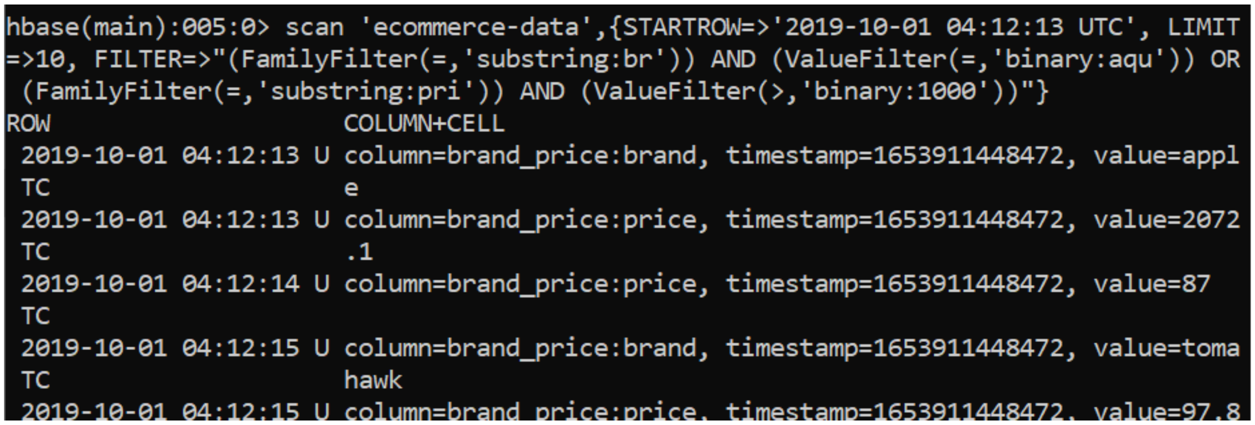
hbase(main):002:0> scan 'ecommerce-data', {STARTROW=>'2019-10-01 07:23:21 UTC', LIMIT=>10,FILTER=>"(FamilyFilter(=,'substring:pri')) AND (ValueFilter(<,'binary:200'))"}



```
hbase(main):002:0> scan 'ecommerce-data', {STARTROW=>'2019-10-01 07:23:21 UTC', LIMI
T=>10,FILTER=>"(FamilyFilter(=,'substring:pri')) AND (ValueFilter(<,'binary:200'))"}
ROW                      COLUMN+CELL
 2019-10-01 07:23:21 U column=brand_price:brand, timestamp=1653911448472, value=
 TC
 2019-10-01 07:23:21 U column=brand_price:price, timestamp=1653911448472, value=10.2
 TC                      4
 2019-10-01 07:23:24 U column=brand_price:price, timestamp=1653911448472, value=179.
 TC                      35
 2019-10-01 07:23:26 U column=brand_price:brand, timestamp=1653911448472, value=
 TC
```

Scan table for key values where column includes substring ‘br’ and value ‘aq’, OR where column includes substring ‘pri’ and value greater than 1000.

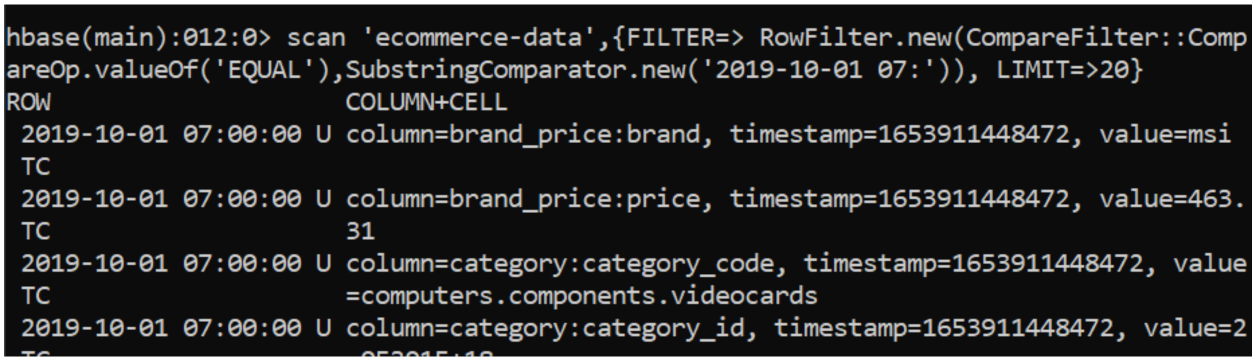
hbase(main):005:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 04:12:13 UTC', LIMIT=>10, FILTER=>"(FamilyFilter(=,'substring:br')) AND (ValueFilter(=,'binary:aqu')) OR (FamilyFilter(=,'substring:pri')) AND (ValueFilter(>,'binary:1000'))"}



```
hbase(main):005:0> scan 'ecommerce-data',{STARTROW=>'2019-10-01 04:12:13 UTC', LIMIT
=>10, FILTER=>"(FamilyFilter(=,'substring:br')) AND (ValueFilter(=,'binary:aqu')) OR
 (FamilyFilter(=,'substring:pri')) AND (ValueFilter(>,'binary:1000'))"}
ROW                      COLUMN+CELL
 2019-10-01 04:12:13 U column=brand_price:brand, timestamp=1653911448472, value=appl
 TC                      e
 2019-10-01 04:12:13 U column=brand_price:price, timestamp=1653911448472, value=2072
 TC                      .1
 2019-10-01 04:12:14 U column=brand_price:price, timestamp=1653911448472, value=87
 TC
 2019-10-01 04:12:15 U column=brand_price:brand, timestamp=1653911448472, value=toma
 TC                      hawk
 2019-10-01 04:12:15 U column=brand_price:price, timestamp=1653911448472, value=97.8
```

Scan table for rows using CompareFilter, RowFilter and SubstringComparator to find rowkeys which include the substring ‘2019-10-01 07:’ with a limit of 20 rows

hbase(main):012:0> scan 'ecommerce-data',{FILTER=> RowFilter.new(CompareFilter::CompareOp.valueOf('EQUAL'),SubstringComparator.new('2019-10-01 07:')), LIMIT=>20}

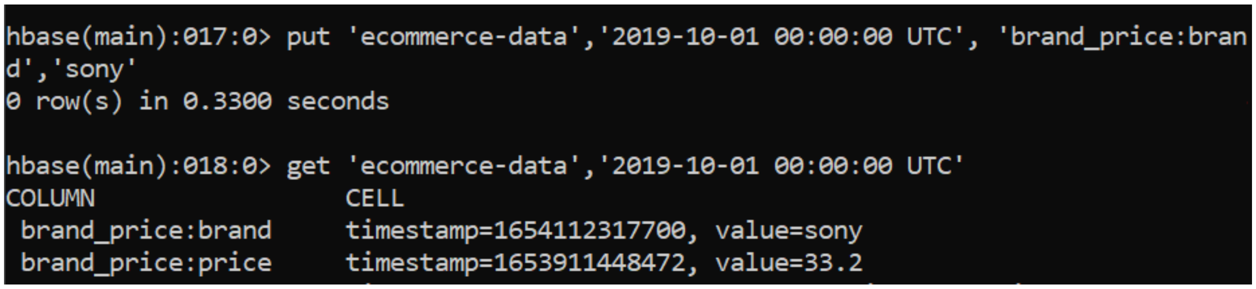


```
hbase(main):012:0> scan 'ecommerce-data',{FILTER=> RowFilter.new(CompareFilter::Comp
areOp.valueOf('EQUAL'),SubstringComparator.new('2019-10-01 07:')), LIMIT=>20}
ROW                      COLUMN+CELL
 2019-10-01 07:00:00 U column=brand_price:brand, timestamp=1653911448472, value=msi
 TC
 2019-10-01 07:00:00 U column=brand_price:price, timestamp=1653911448472, value=463.
 TC                      31
 2019-10-01 07:00:00 U column=category:category_code, timestamp=1653911448472, value
 TC                      =computers.components.videocards
 2019-10-01 07:00:00 U column=category:category_id, timestamp=1653911448472, value=2
```

Put value ‘sony’ into brand column for row '2019-10-01 00:00:00 UTC'. Verifying put command using get command

hbase(main):017:0> put 'ecommerce-data','2019-10-01 00:00:00 UTC', 'brand_price:brand','sony'

hbase(main):018:0> get 'ecommerce-data','2019-10-01 00:00:00 UTC'



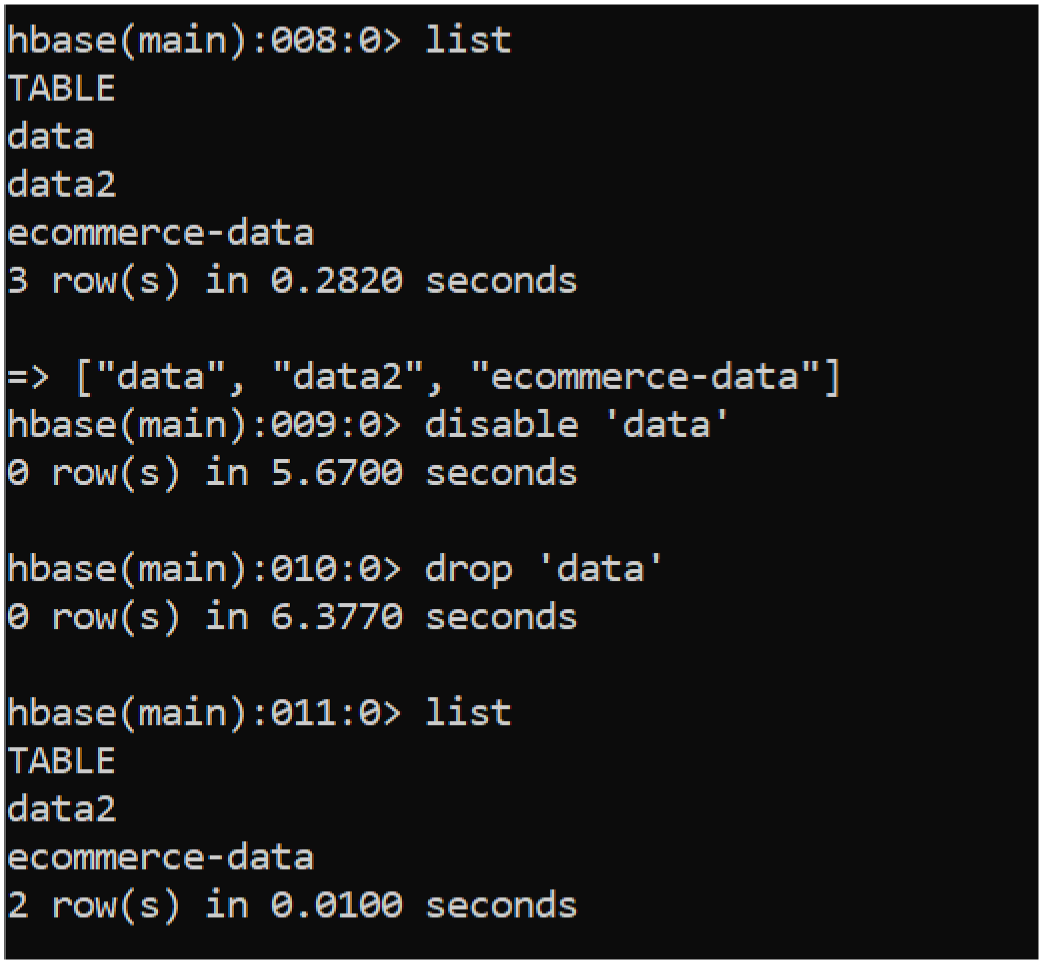
```
hbase(main):017:0> put 'ecommerce-data','2019-10-01 00:00:00 UTC', 'brand_price:bran
d','sony'
0 row(s) in 0.3300 seconds

hbase(main):018:0> get 'ecommerce-data','2019-10-01 00:00:00 UTC'
COLUMN                 CELL
 brand_price:brand     timestamp=1654112317700, value=sony
 brand_price:price     timestamp=1653911448472, value=33.2
```

Disable and drop table 'data'

hbase(main):008:0> list

hbase(main):009:0> disable 'data'

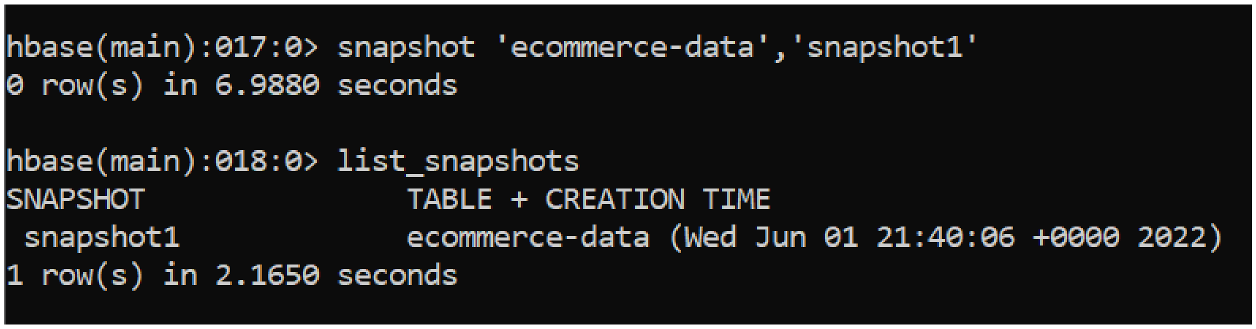
hbase(main):010:0> drop 'data'



```
hbase(main):008:0> list
TABLE
data
data2
ecommerce-data
3 row(s) in 0.2820 seconds

=> ["data", "data2", "ecommerce-data"]
hbase(main):009:0> disable 'data'
0 row(s) in 5.6700 seconds

hbase(main):010:0> drop 'data'
0 row(s) in 6.3770 seconds

hbase(main):011:0> list
TABLE
data2
ecommerce-data
2 row(s) in 0.0100 seconds
```

Snapshotting table and checking list

hbase(main):017:0> snapshot 'ecommerce-data','snapshot1' hbase(main):018:0> list_snapshots

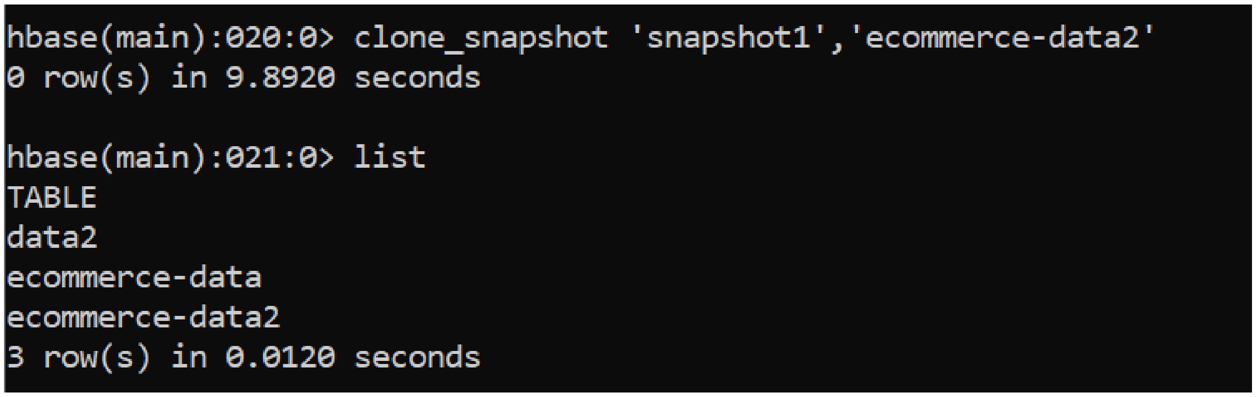


```
hbase(main):017:0> snapshot 'ecommerce-data','snapshot1'
0 row(s) in 6.9880 seconds

hbase(main):018:0> list_snapshots
SNAPSHOT                TABLE + CREATION TIME
 snapshot1              ecommerce-data (Wed Jun 01 21:40:06 +0000 2022)
1 row(s) in 2.1650 seconds
```

Using snapshot to clone table

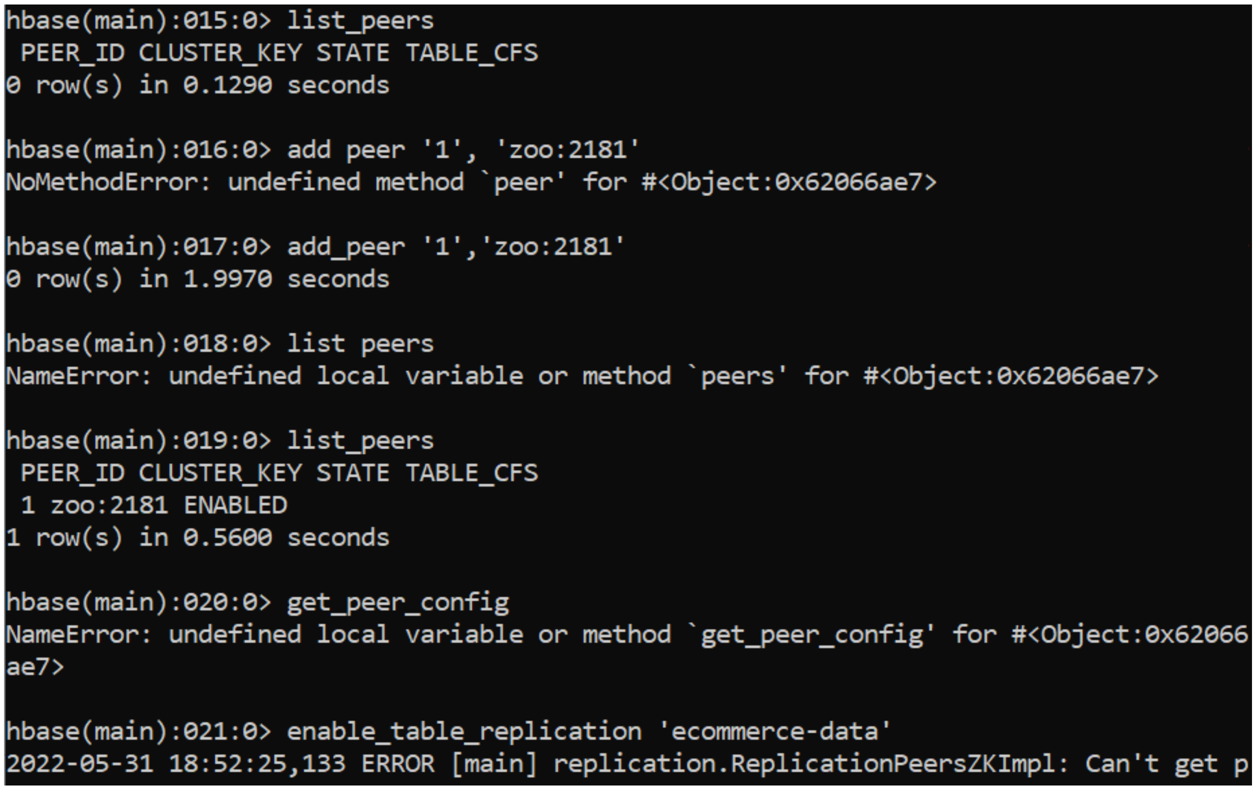
hbase(main):020:0> clone_snapshot 'snapshot1','ecommerce-data2'

hbase(main):021:0> list



```
hbase(main):020:0> clone_snapshot 'snapshot1','ecommerce-data2'
0 row(s) in 9.8920 seconds

hbase(main):021:0> list
TABLE
data2
ecommerce-data
ecommerce-data2
3 row(s) in 0.0120 seconds
```

Attempted peers and replication



```
hbase(main):015:0> list_peers
 PEER_ID CLUSTER_KEY STATE TABLE_CFS
0 row(s) in 0.1290 seconds

hbase(main):016:0> add peer '1', 'zoo:2181'
NoMethodError: undefined method `peer' for #<Object:0x62066ae7>

hbase(main):017:0> add_peer '1','zoo:2181'
0 row(s) in 1.9970 seconds

hbase(main):018:0> list peers
NameError: undefined local variable or method `peers' for #<Object:0x62066ae7>

hbase(main):019:0> list_peers
 PEER_ID CLUSTER_KEY STATE TABLE_CFS
 1 zoo:2181 ENABLED
1 row(s) in 0.5600 seconds

hbase(main):020:0> get_peer_config
NameError: undefined local variable or method `get_peer_config' for #<Object:0x62066
ae7>

hbase(main):021:0> enable_table_replication 'ecommerce-data'
2022-05-31 18:52:25,133 ERROR [main] replication.ReplicationPeersZKImpl: Can't get p
```