IN402 - Modeling and Predictive Analysis

Unit 9 Assignment Part 1 / Module 6 Competency Assessment Part 1

Using a NoSQL Database to Ingest and Retrieve Data

Apache HBase Overview

Understanding how to work with a NoSQL database is a crucial step before modeling the use of unstructured data. The objective of this assignment is for you to become familiar with a NoSQL database (Apache HBase).

A few decades ago, the Internet was not available, which is also when there was far less data generation. In addition, stored data were primarily structured in nature. Structured data means data having a definite structure and maintaining a standard order. This data was stored in relational database management systems (RDBMS).

With the evolution of the Internet, huge volumes of semi-structured data started getting generated. Semi-structured data consists of a combination of both structured and unstructured data, which includes emails, JSON, XML, and .csv files to name a few. Tremendous amounts of semi-structured data are continually being created across the globe. As a result, storing and processing this data became a major challenge. RDBMS systems cannot effectively store and process such large volumes of data. Apache HBase was created to help address this major challenge.

Apache HBase is a distributed column-oriented non-relational database written in Java and built on top of the Hadoop File System (HDFS). It is an open-source project and is horizontally scalable. It is modeled after Google's Bigtable, which is designed to provide quick random access to huge amounts of semi-structured data. It leverages the fault tolerance provided by HDFS. It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in HDFS.

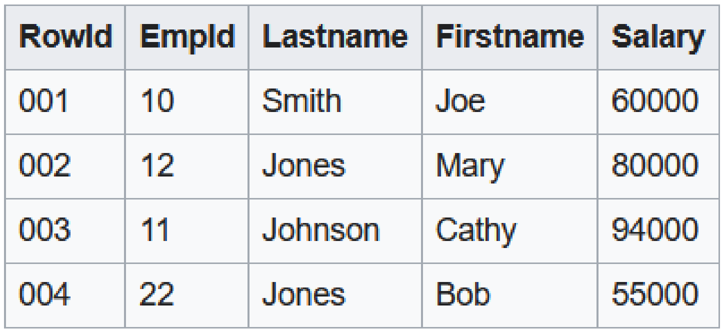
One can store the data in HDFS either directly or through Apache HBase. Data consumer reads/accesses the data in HDFS randomly using Apache HBase. Apache HBase sits on top of HDFS and provides read and write access.

Apache HBase is a column-oriented database and the tables in it are sorted by row. The table schema defines only column families, which are the key value pairs. A table can have multiple column families, and each column family can have any number of columns. Subsequent column values are stored contiguously on the disk. Each cell value of the table has a timestamp.

A column-oriented DBMS (or columnar database management system) is a database management system (DBMS) that stores data tables by column rather than by row. Both columnar and row databases can use traditional database query languages like SQL to load data and perform queries. Both row and columnar databases can become the backbone in a system to serve data for common extract, transform, load (ETL) and data visualization tools.

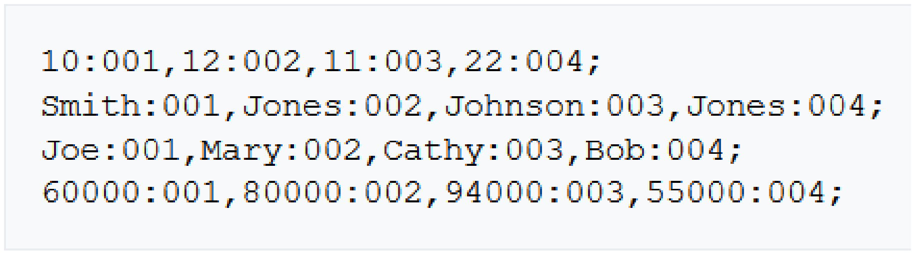
However, by storing data in columns rather than rows, the database can more precisely access the data it needs to answer a query rather than scanning and discarding unwanted data in rows. Query performance is increased for certain workloads.

Here is a relational database example:



RowId	EmpId	Lastname	Firstname	Salary
001	10	Smith	Joe	60000
002	12	Jones	Mary	80000
003	11	Johnson	Cathy	94000
004	22	Jones	Bob	55000

A column-oriented database serializes all the values of a column together, then the values of the next column, and so on. For the table presented above, the data would be stored in the following fashion in a columnar database:

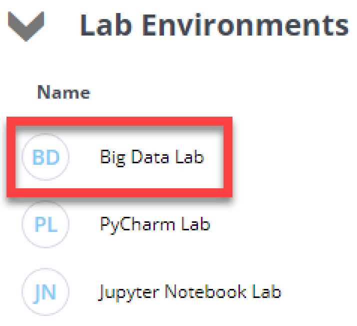


```
10:001,12:002,11:003,22:004;
Smith:001,Jones:002,Johnson:003,Jones:004;
Joe:001,Mary:002,Cathy:003,Bob:004;
60000:001,80000:002,94000:003,55000:004;
```

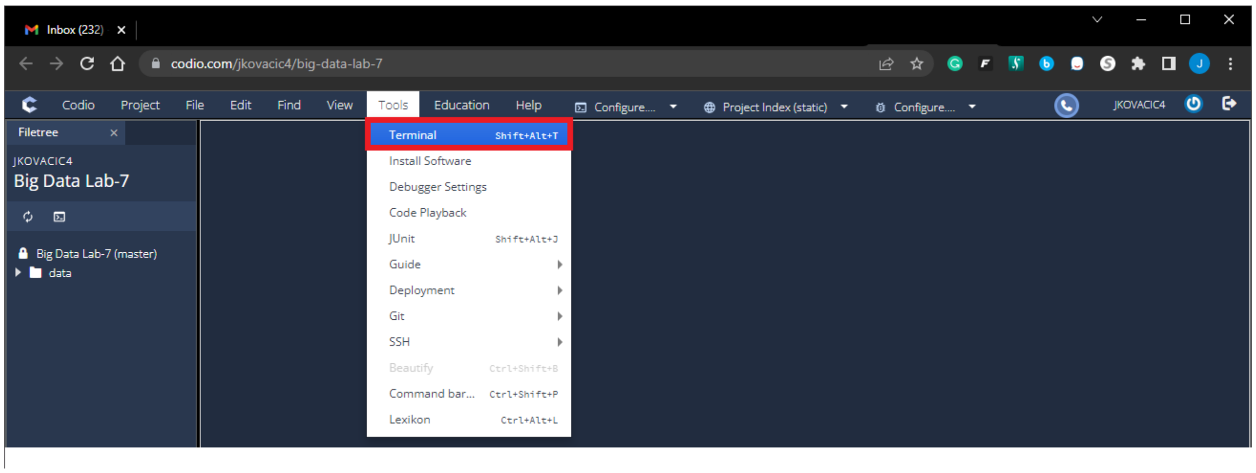
Apache HBase is horizontally scalable, which is the ability to increase capacity by connecting multiple hardware or software entities so that they work as a single logical unit. When servers are clustered, the original server is being scaled out horizontally. If a cluster requires more resources to improve performance and provide high availability (HA), an administrator can scale out by adding more servers to the cluster. An important advantage of horizontal scalability is that it can provide administrators with the ability to increase capacity on the fly. Another advantage is that in theory, horizontal scalability is only limited by how many entities can be connected successfully

Assignment Procedure

Access your IN402 Codio lab environment. Click on the **Big Data Lab** link in the list of IN402 options.

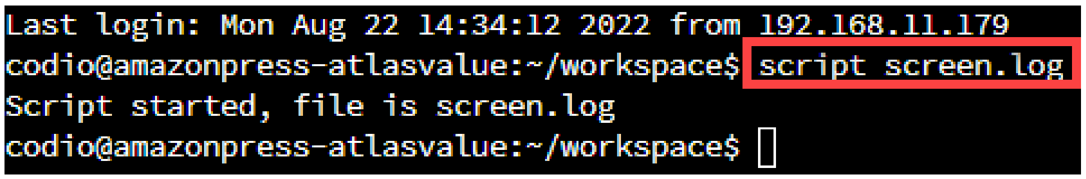


Open a Linux terminal session, which you can do by clicking the "**Tools → Terminal**" from the top menu in your Codio Big Data Lab environment.



Once the Terminal window is open, execute the following command at the Linux prompt to start logging your session in the **screen.log** file:

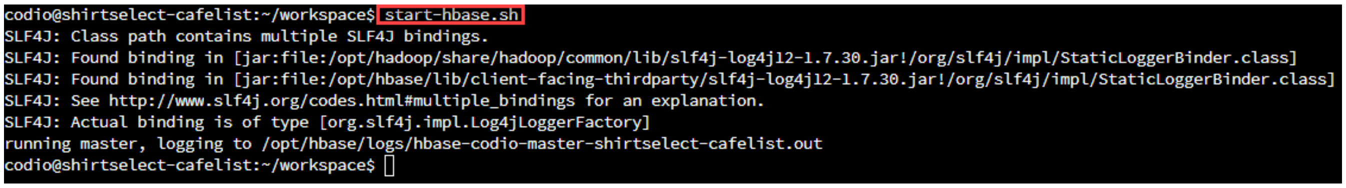
```
script screen.log
```



```
Last login: Mon Aug 22 14:34:12 2022 from 192.168.11.179
codio@amazonpress-atlasvalue:~/workspace$ script screen.log
Script started, file is screen.log
codio@amazonpress-atlasvalue:~/workspace$
```

Start the Apache HBase database service by entering the following command at the Linux prompt:

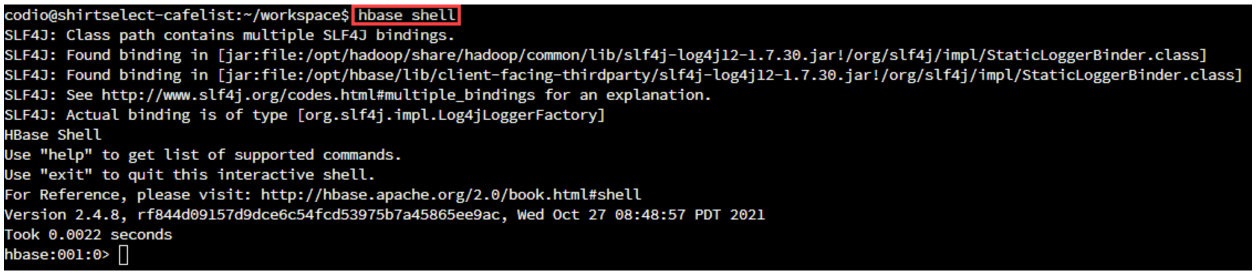
```
start-hbase.sh
```



```
codio@shirtselect-cafelist:~/workspace$ start-hbase.sh
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/hbase/lib/client-facing-thirdparty/slf4j-log4j12-1.7.30.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]
running master, logging to /opt/hbase/logs/hbase-codio-master-shirtselect-cafelist.out
codio@shirtselect-cafelist:~/workspace$
```

Start the Apache HBase client shell by entering the following command at the Linux prompt:

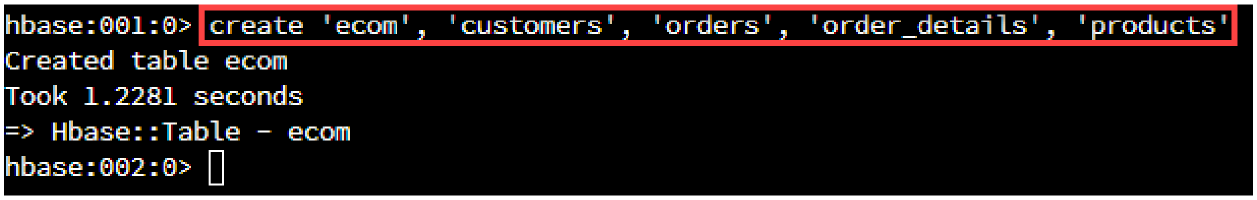
```
hbase shell
```



```
codio@shirtselect-cafelist:~/workspace$ hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/hbase/lib/client-facing-thirdparty/slf4j-log4j12-1.7.30.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
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
For Reference, please visit: http://hbase.apache.org/2.0/book.html#shell
Version 2.4.8, rf844d09157d9dce6c54fcd53975b7a45865ee9ac, Wed Oct 27 08:48:57 PDT 2021
Took 0.0022 seconds
hbase:001:0>
```

Create a table with column families using the following command at the Apache HBase prompt:

```
create 'ecom','customers','orders','order_details','products'
```



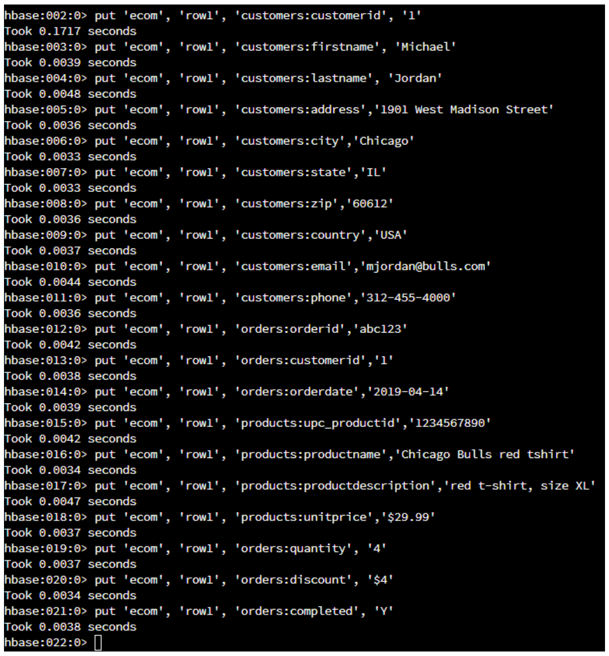
```
hbase:001:0> create 'ecom', 'customers', 'orders', 'order_details', 'products'
Created table ecom
Took 1.2281 seconds
=> Hbase::Table - ecom
hbase:002:0>
```

The syntax for the above executed command is as follows:

```
create '<table name>', '<column-family1>', '<column-family2>' etc..
```

Insert the data for Michael Jordan into the **ecom** table via the following commands at the HBase prompt:

```
put 'ecom','row1','customers:customerid','1'
put 'ecom','row1','customers:firstname','Michael'
put 'ecom','row1','customers:lastname','Jordan'
put 'ecom','row1','customers:address','1901 West Madison Street'
put 'ecom','row1','customers:city','Chicago'
put 'ecom','row1','customers:state','IL'
put 'ecom','row1','customers:zip','60612'
put 'ecom','row1','customers:country','USA'
put 'ecom','row1','customers:email','mjordan@bulls.com'
put 'ecom','row1','customers:phone','312-455-4000'
put 'ecom','row1','orders:orderid','abc123'
put 'ecom','row1','orders:customerid','1'
put 'ecom','row1','orders:orderdate','2019-04-14'
put 'ecom','row1','products:upc_productid','1234567890'
put 'ecom','row1','products:productname','Chicago Bulls red tshirt'
put 'ecom','row1','products:productdescription','red t-shirt, size XL'
put 'ecom','row1','products:unitprice','$29.99'
put 'ecom','row1','orders:quantity','4'
put 'ecom','row1','orders:discount','$4'
put 'ecom','row1','orders:completed','Y'
```



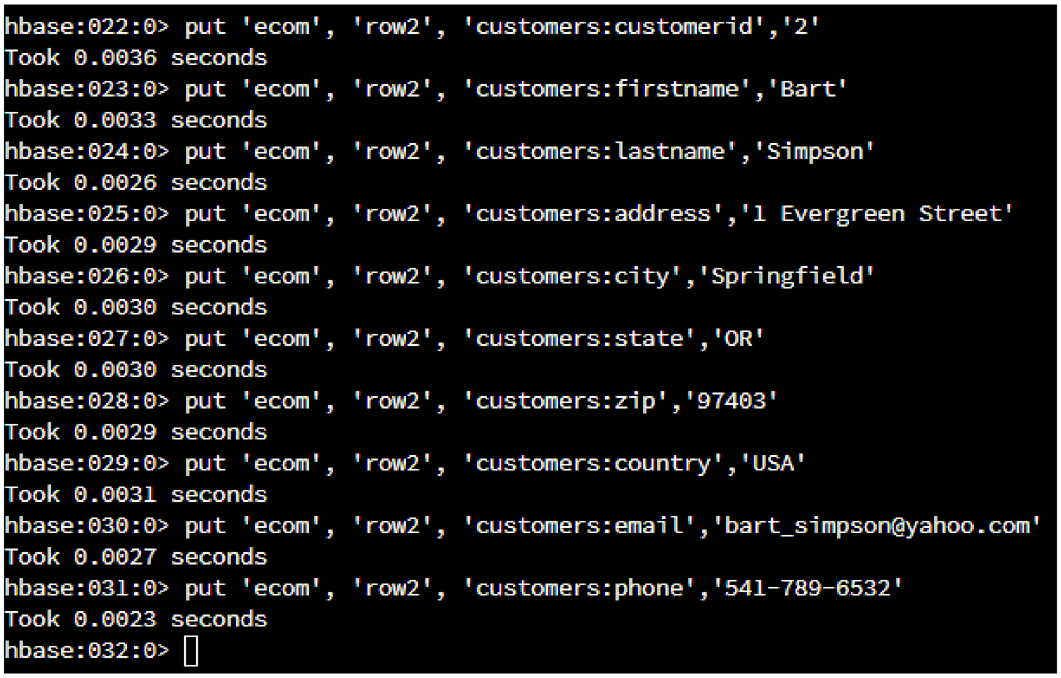
```
hbase:002:0> put 'ecom', 'row1', 'customers:customerid', '1'
Took 0.1717 seconds
hbase:003:0> put 'ecom', 'row1', 'customers:firstname', 'Michael'
Took 0.0039 seconds
hbase:004:0> put 'ecom', 'row1', 'customers:lastname', 'Jordan'
Took 0.0048 seconds
hbase:005:0> put 'ecom', 'row1', 'customers:address','1901 West Madison Street'
Took 0.0036 seconds
hbase:006:0> put 'ecom', 'row1', 'customers:city','Chicago'
Took 0.0033 seconds
hbase:007:0> put 'ecom', 'row1', 'customers:state','IL'
Took 0.0033 seconds
hbase:008:0> put 'ecom', 'row1', 'customers:zip','60612'
Took 0.0036 seconds
hbase:009:0> put 'ecom', 'row1', 'customers:country','USA'
Took 0.0037 seconds
hbase:010:0> put 'ecom', 'row1', 'customers:email','mjordan@bulls.com'
Took 0.0044 seconds
hbase:011:0> put 'ecom', 'row1', 'customers:phone','312-455-4000'
Took 0.0036 seconds
hbase:012:0> put 'ecom', 'row1', 'orders:orderid','abc123'
Took 0.0042 seconds
hbase:013:0> put 'ecom', 'row1', 'orders:customerid','1'
Took 0.0038 seconds
hbase:014:0> put 'ecom', 'row1', 'orders:orderdate','2019-04-14'
Took 0.0039 seconds
hbase:015:0> put 'ecom', 'row1', 'products:upc_productid','1234567890'
Took 0.0042 seconds
hbase:016:0> put 'ecom', 'row1', 'products:productname','Chicago Bulls red tshirt'
Took 0.0034 seconds
hbase:017:0> put 'ecom', 'row1', 'products:productdescription','red t-shirt, size XL'
Took 0.0047 seconds
hbase:018:0> put 'ecom', 'row1', 'products:unitprice','$29.99'
Took 0.0037 seconds
hbase:019:0> put 'ecom', 'row1', 'orders:quantity', '4'
Took 0.0037 seconds
hbase:020:0> put 'ecom', 'row1', 'orders:discount', '$4'
Took 0.0034 seconds
hbase:021:0> put 'ecom', 'row1', 'orders:completed', 'Y'
Took 0.0038 seconds
hbase:022:0> 
```

The syntax for the Apache HBase **put** command is as follows:

```
put '<HBase_table_name>', 'row_key', '<colfamily:colname>', '<value>'
```

Insert the data for Bart Simpson into the **ecom** table via the following commands at the HBase prompt:

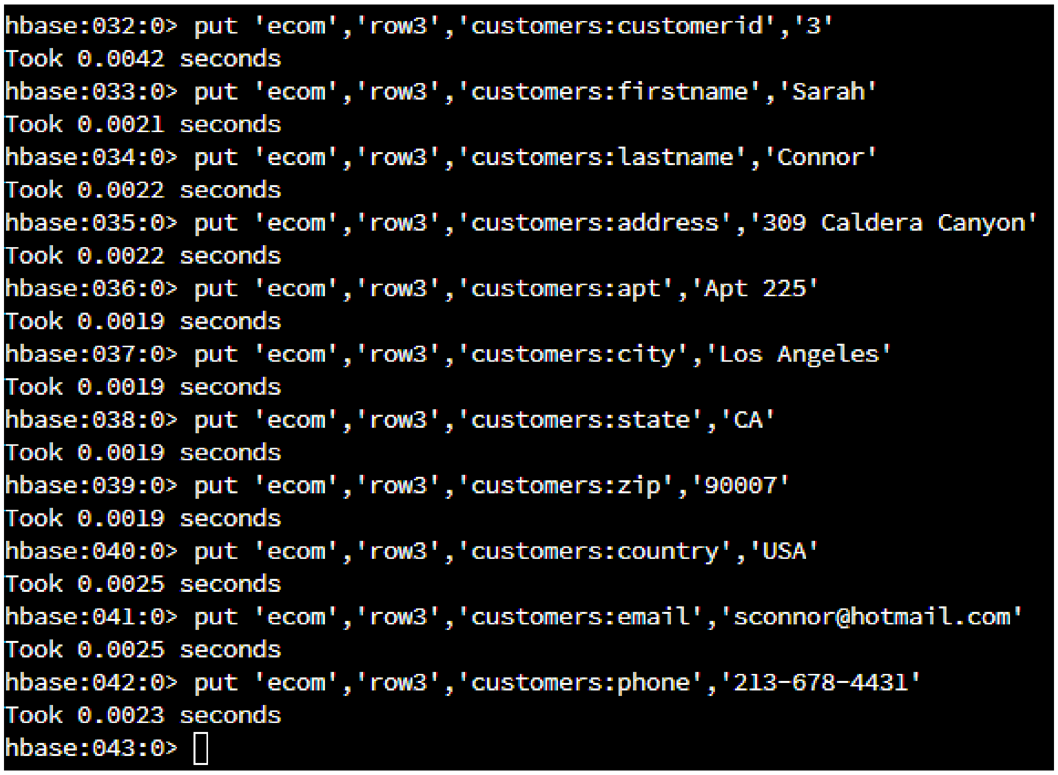
```
put 'ecom','row2','customers:customerid','2'
put 'ecom','row2','customers:firstname','Bart'
put 'ecom','row2','customers:lastname','Simpson'
put 'ecom','row2','customers:address','1 Evergreen Street'
put 'ecom','row2','customers:city','Springfield'
put 'ecom','row2','customers:state','OR'
put 'ecom','row2','customers:zip','97403'
put 'ecom','row2','customers:country','USA'
put 'ecom','row2','customers:email','bart_simpson@yahoo.com'
put 'ecom','row2','customers:phone','541-789-6532'
```



```
hbase:022:0> put 'ecom', 'row2', 'customers:customerid','2'
Took 0.0036 seconds
hbase:023:0> put 'ecom', 'row2', 'customers:firstname','Bart'
Took 0.0033 seconds
hbase:024:0> put 'ecom', 'row2', 'customers:lastname','Simpson'
Took 0.0026 seconds
hbase:025:0> put 'ecom', 'row2', 'customers:address','1 Evergreen Street'
Took 0.0029 seconds
hbase:026:0> put 'ecom', 'row2', 'customers:city','Springfield'
Took 0.0030 seconds
hbase:027:0> put 'ecom', 'row2', 'customers:state','OR'
Took 0.0030 seconds
hbase:028:0> put 'ecom', 'row2', 'customers:zip','97403'
Took 0.0029 seconds
hbase:029:0> put 'ecom', 'row2', 'customers:country','USA'
Took 0.0031 seconds
hbase:030:0> put 'ecom', 'row2', 'customers:email','bart_simpson@yahoo.com'
Took 0.0027 seconds
hbase:031:0> put 'ecom', 'row2', 'customers:phone','541-789-6532'
Took 0.0023 seconds
hbase:032:0>
```

Insert the data for Sarah Connor into the **ecom** table via the following commands at the HBase prompt:

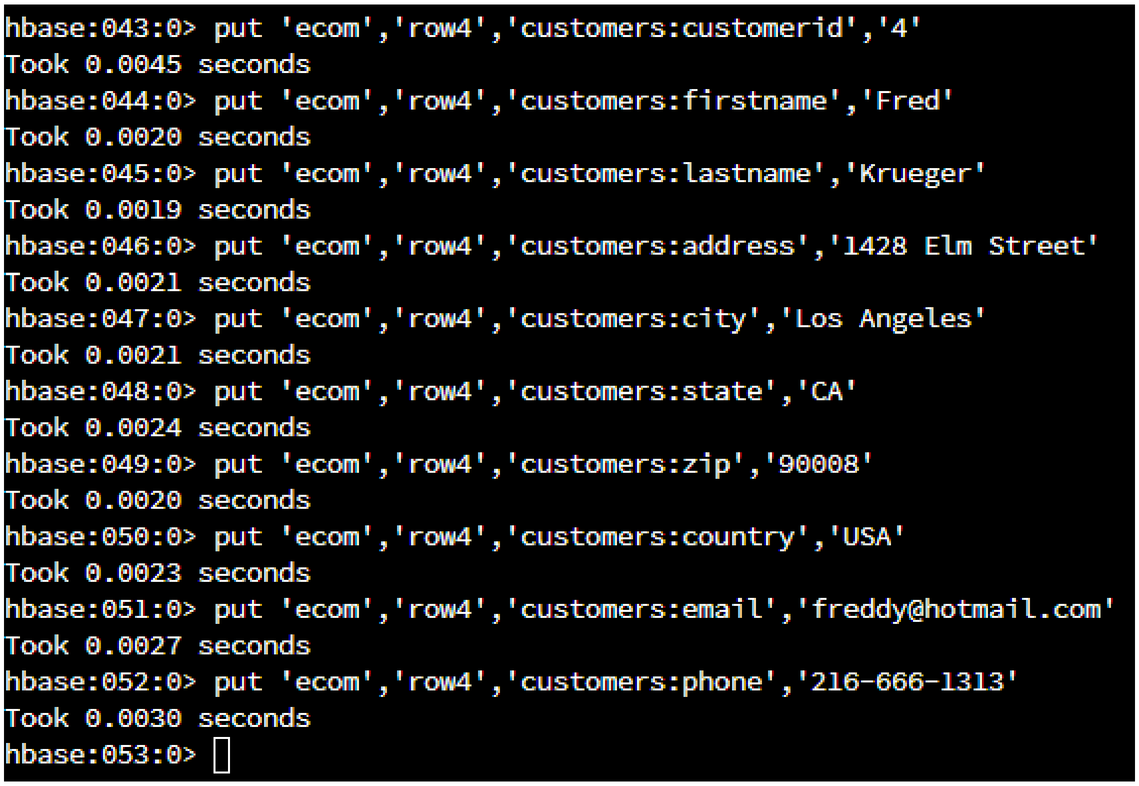
```
put 'ecom','row3','customers:customerid','3'
put 'ecom','row3','customers:firstname','Sarah'
put 'ecom','row3','customers:lastname','Connor'
put 'ecom','row3','customers:address','309 Caldera Canyon'
put 'ecom','row3','customers:apt','Apt 225'
put 'ecom','row3','customers:city','Los Angeles'
put 'ecom','row3','customers:state','CA'
put 'ecom','row3','customers:zip','90007'
put 'ecom','row3','customers:country','USA'
put 'ecom','row3','customers:email','sconnor@hotmail.com'
put 'ecom','row3','customers:phone','213-678-4431'
```



```
hbase:032:0> put 'ecom','row3','customers:customerid','3'
Took 0.0042 seconds
hbase:033:0> put 'ecom','row3','customers:firstname','Sarah'
Took 0.0021 seconds
hbase:034:0> put 'ecom','row3','customers:lastname','Connor'
Took 0.0022 seconds
hbase:035:0> put 'ecom','row3','customers:address','309 Caldera Canyon'
Took 0.0022 seconds
hbase:036:0> put 'ecom','row3','customers:apt','Apt 225'
Took 0.0019 seconds
hbase:037:0> put 'ecom','row3','customers:city','Los Angeles'
Took 0.0019 seconds
hbase:038:0> put 'ecom','row3','customers:state','CA'
Took 0.0019 seconds
hbase:039:0> put 'ecom','row3','customers:zip','90007'
Took 0.0019 seconds
hbase:040:0> put 'ecom','row3','customers:country','USA'
Took 0.0025 seconds
hbase:041:0> put 'ecom','row3','customers:email','sconnor@hotmail.com'
Took 0.0025 seconds
hbase:042:0> put 'ecom','row3','customers:phone','213-678-4431'
Took 0.0023 seconds
hbase:043:0>
```

Insert the data for Fred Krueger into the **ecom** table via the following commands at the HBase prompt:

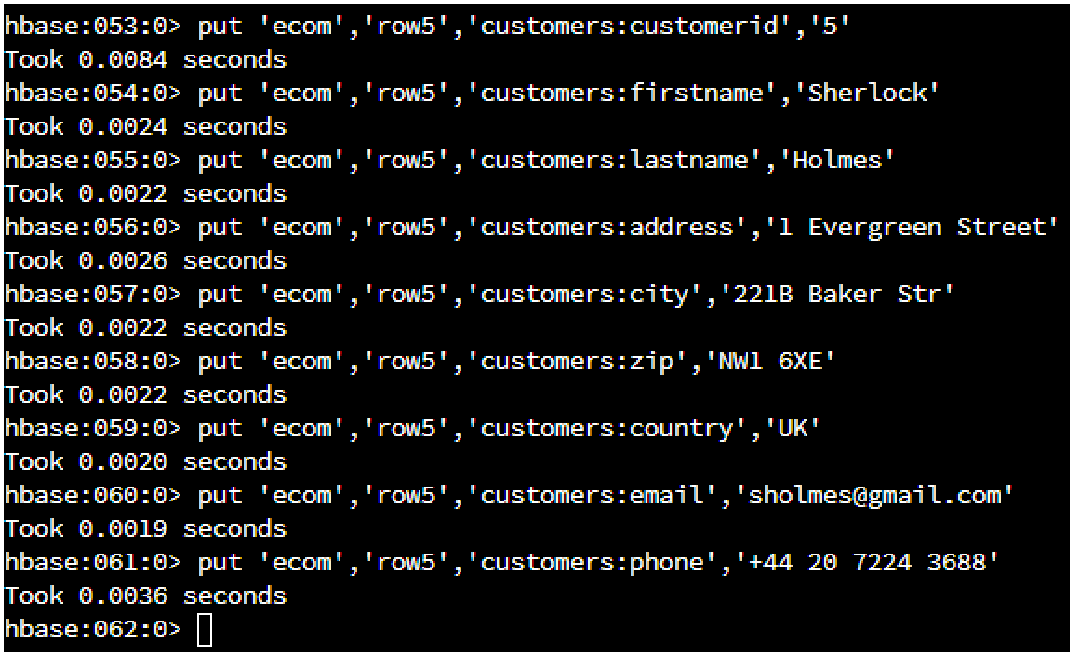
```
put 'ecom','row4','customers:customerid','4'
put 'ecom','row4','customers:firstname','Fred'
put 'ecom','row4','customers:lastname','Krueger'
put 'ecom','row4','customers:address','1428 Elm Street'
put 'ecom','row4','customers:city','Los Angeles'
put 'ecom','row4','customers:state','CA'
put 'ecom','row4','customers:zip','90008'
put 'ecom','row4','customers:country','USA'
put 'ecom','row4','customers:email','freddy@hotmail.com'
put 'ecom','row4','customers:phone','216-666-1313'
```



```
hbase:043:0> put 'ecom','row4','customers:customerid','4'
Took 0.0045 seconds
hbase:044:0> put 'ecom','row4','customers:firstname','Fred'
Took 0.0020 seconds
hbase:045:0> put 'ecom','row4','customers:lastname','Krueger'
Took 0.0019 seconds
hbase:046:0> put 'ecom','row4','customers:address','1428 Elm Street'
Took 0.0021 seconds
hbase:047:0> put 'ecom','row4','customers:city','Los Angeles'
Took 0.0021 seconds
hbase:048:0> put 'ecom','row4','customers:state','CA'
Took 0.0024 seconds
hbase:049:0> put 'ecom','row4','customers:zip','90008'
Took 0.0020 seconds
hbase:050:0> put 'ecom','row4','customers:country','USA'
Took 0.0023 seconds
hbase:051:0> put 'ecom','row4','customers:email','freddy@hotmail.com'
Took 0.0027 seconds
hbase:052:0> put 'ecom','row4','customers:phone','216-666-1313'
Took 0.0030 seconds
hbase:053:0>
```

Insert the data for Sherlock Holmes into the **ecom** table via the following commands at the HBase prompt:

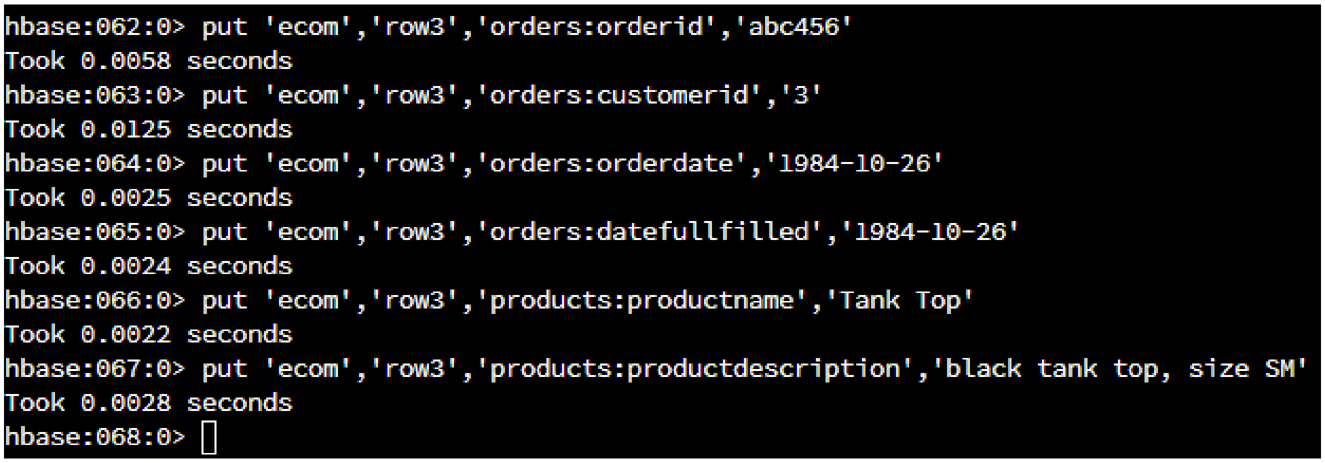
```
put 'ecom','row5','customers:customerid','5'
put 'ecom','row5','customers:firstname','Sherlock'
put 'ecom','row5','customers:lastname','Holmes'
put 'ecom','row5','customers:address','1 Evergreen Street'
put 'ecom','row5','customers:city','221B Baker Str'
put 'ecom','row5','customers:zip','NW1 6XE'
put 'ecom','row5','customers:country','UK'
put 'ecom','row5','customers:email','sholmes@gmail.com'
put 'ecom','row5','customers:phone','+44 20 7224 3688'
```



```
hbase:053:0> put 'ecom','row5','customers:customerid','5'
Took 0.0084 seconds
hbase:054:0> put 'ecom','row5','customers:firstname','Sherlock'
Took 0.0024 seconds
hbase:055:0> put 'ecom','row5','customers:lastname','Holmes'
Took 0.0022 seconds
hbase:056:0> put 'ecom','row5','customers:address','1 Evergreen Street'
Took 0.0026 seconds
hbase:057:0> put 'ecom','row5','customers:city','221B Baker Str'
Took 0.0022 seconds
hbase:058:0> put 'ecom','row5','customers:zip','NW1 6XE'
Took 0.0022 seconds
hbase:059:0> put 'ecom','row5','customers:country','UK'
Took 0.0020 seconds
hbase:060:0> put 'ecom','row5','customers:email','sholmes@gmail.com'
Took 0.0019 seconds
hbase:061:0> put 'ecom','row5','customers:phone','+44 20 7224 3688'
Took 0.0036 seconds
hbase:062:0> 
```

Sarah Connor ordered a small-sized black tank top on October 26, 1984. Add these details in the **orders**, **order_details**, and **products** column families for Sarah Connor using the following commands:

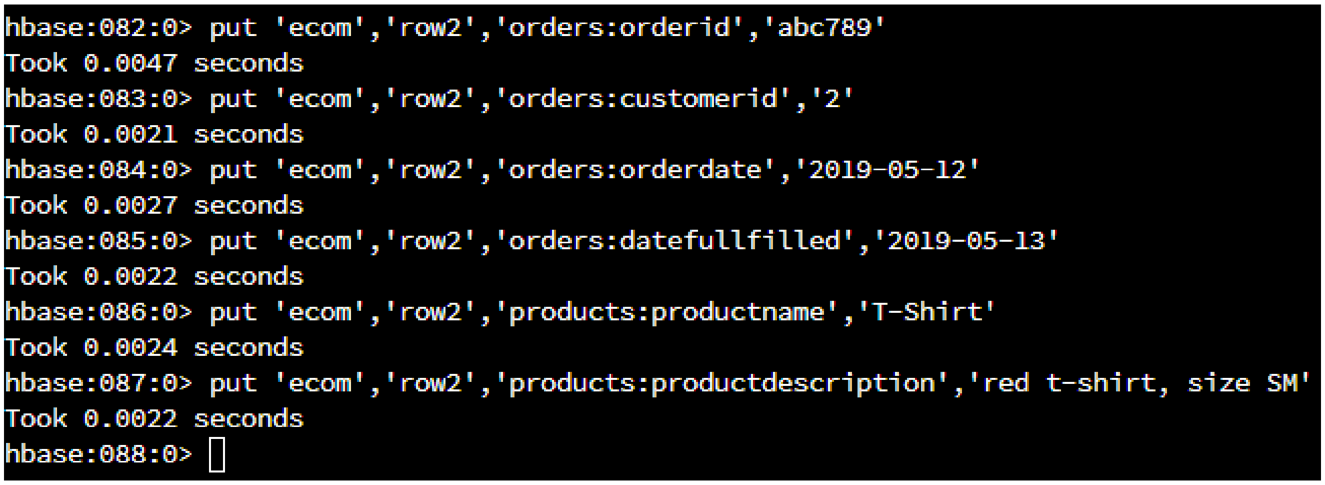
```
put 'ecom','row3','orders:orderid','abc456'
put 'ecom','row3','orders:customerid','3'
put 'ecom','row3','orders:orderdate','1984-10-26'
put 'ecom','row3','orders:datefullfilled','1984-10-26'
put 'ecom','row3','products:productname','Tank Top'
put 'ecom','row3','products:productdescription','black tank top, size SM'
```



```
hbase:062:0> put 'ecom','row3','orders:orderid','abc456'
Took 0.0058 seconds
hbase:063:0> put 'ecom','row3','orders:customerid','3'
Took 0.0125 seconds
hbase:064:0> put 'ecom','row3','orders:orderdate','1984-10-26'
Took 0.0025 seconds
hbase:065:0> put 'ecom','row3','orders:datefullfilled','1984-10-26'
Took 0.0024 seconds
hbase:066:0> put 'ecom','row3','products:productname','Tank Top'
Took 0.0022 seconds
hbase:067:0> put 'ecom','row3','products:productdescription','black tank top, size SM'
Took 0.0028 seconds
hbase:068:0> 
```

Bart Simpson ordered a small-sized red t-shirt on May 12, 2019. Add these details in the **orders**, **order_details**, and **products** column families for Bart Simpson using the following commands:

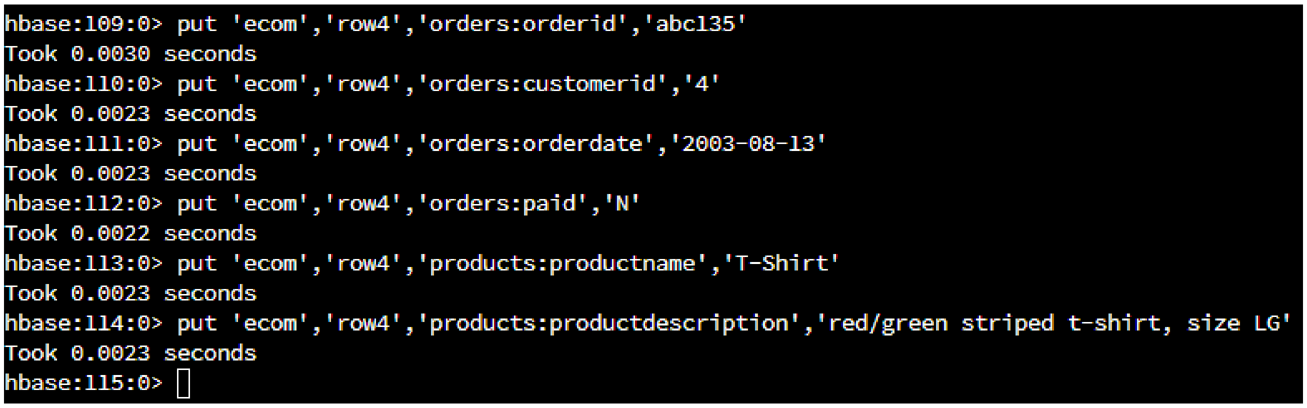
```
put 'ecom','row2','orders:orderid','abc789'
put 'ecom','row2','orders:customerid','2'
put 'ecom','row2','orders:orderdate','2019-05-12'
put 'ecom','row2','orders:datefullfilled','2019-05-13'
put 'ecom','row2','products:productname','T-Shirt'
put 'ecom','row2','products:productdescription','red t-shirt, size SM'
```



```
hbase:082:0> put 'ecom','row2','orders:orderid','abc789'
Took 0.0047 seconds
hbase:083:0> put 'ecom','row2','orders:customerid','2'
Took 0.0021 seconds
hbase:084:0> put 'ecom','row2','orders:orderdate','2019-05-12'
Took 0.0027 seconds
hbase:085:0> put 'ecom','row2','orders:datefullfilled','2019-05-13'
Took 0.0022 seconds
hbase:086:0> put 'ecom','row2','products:productname','T-Shirt'
Took 0.0024 seconds
hbase:087:0> put 'ecom','row2','products:productdescription','red t-shirt, size SM'
Took 0.0022 seconds
hbase:088:0>
```

Fred Krueger ordered a large-sized red and green striped t-shirt on August 13, 2003. Add these details in the **orders**, **order_details**, and **products** column families for Fred Krueger using the following commands:

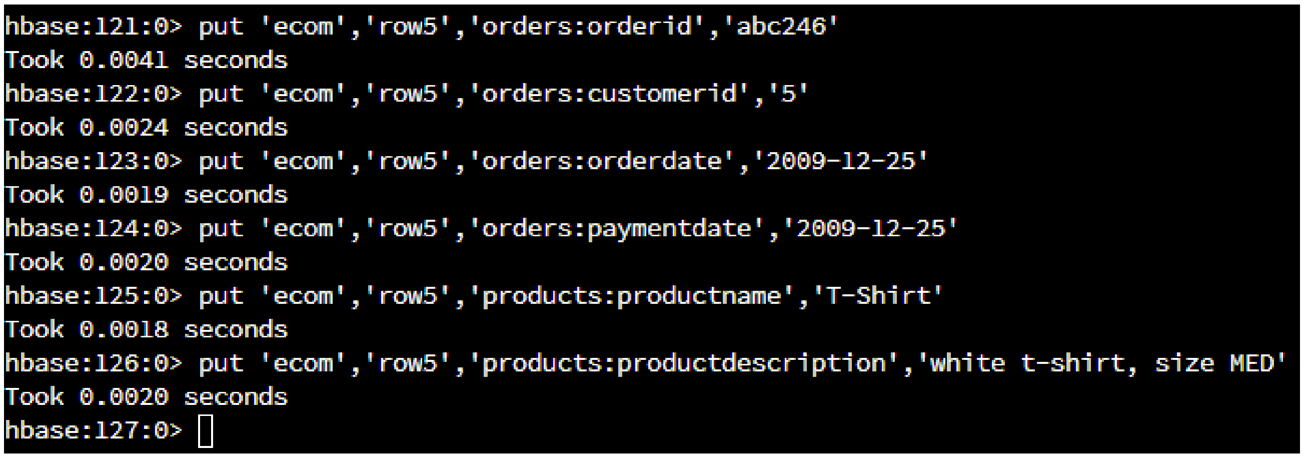
```
put 'ecom','row4','orders:orderid','abc135'
put 'ecom','row4','orders:customerid','4'
put 'ecom','row4','orders:orderdate','2003-08-13'
put 'ecom','row4','orders:paid','N'
put 'ecom','row4','products:productname','T-Shirt'
put 'ecom','row4','products:productdescription','red/green striped t-shirt, size LG'
```



```
hbase:109:0> put 'ecom','row4','orders:orderid','abc135'
Took 0.0030 seconds
hbase:110:0> put 'ecom','row4','orders:customerid','4'
Took 0.0023 seconds
hbase:111:0> put 'ecom','row4','orders:orderdate','2003-08-13'
Took 0.0023 seconds
hbase:112:0> put 'ecom','row4','orders:paid','N'
Took 0.0022 seconds
hbase:113:0> put 'ecom','row4','products:productname','T-Shirt'
Took 0.0023 seconds
hbase:114:0> put 'ecom','row4','products:productdescription','red/green striped t-shirt, size LG'
Took 0.0023 seconds
hbase:115:0>
```

Sherlock Holmes ordered a medium-sized white t-shirt on December 25, 2009. Add these details in the **orders**, **order_details**, and **products** column families for Sherlock Holmes using the following commands:

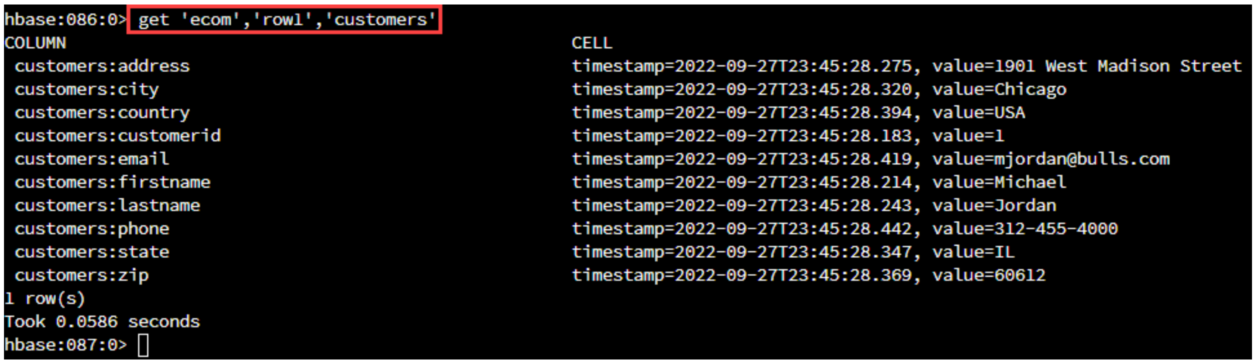
```
put 'ecom','row5','orders:orderid','abc246'
put 'ecom','row5','orders:customerid','5'
put 'ecom','row5','orders:orderdate','2009-12-25'
put 'ecom','row5','orders:paymentdate','2009-12-25'
put 'ecom','row5','products:productname','T-Shirt'
put 'ecom','row5','products:productdescription','white t-shirt, size MED'
```



```
hbase:121:0> put 'ecom','row5','orders:orderid','abc246'
Took 0.0041 seconds
hbase:122:0> put 'ecom','row5','orders:customerid','5'
Took 0.0024 seconds
hbase:123:0> put 'ecom','row5','orders:orderdate','2009-12-25'
Took 0.0019 seconds
hbase:124:0> put 'ecom','row5','orders:paymentdate','2009-12-25'
Took 0.0020 seconds
hbase:125:0> put 'ecom','row5','products:productname','T-Shirt'
Took 0.0018 seconds
hbase:126:0> put 'ecom','row5','products:productdescription','white t-shirt, size MED'
Took 0.0020 seconds
hbase:127:0> 
```

Retrieve the **customers** family column data for the first row using the following command:

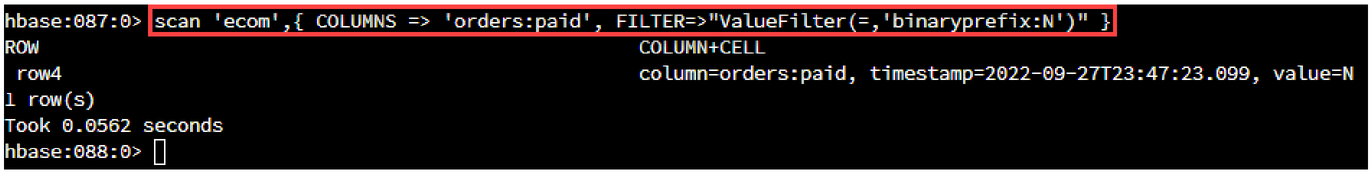
```
get 'ecom','row1','customers'
```



```
hbase:086:0> get 'ecom','row1','customers'
COLUMN                                              CELL
 customers:address                                  timestamp=2022-09-27T23:45:28.275, value=1901 West Madison Street
 customers:city                                     timestamp=2022-09-27T23:45:28.320, value=Chicago
 customers:country                                  timestamp=2022-09-27T23:45:28.394, value=USA
 customers:customerid                               timestamp=2022-09-27T23:45:28.183, value=1
 customers:email                                    timestamp=2022-09-27T23:45:28.419, value=mjordan@bulls.com
 customers:firstname                                timestamp=2022-09-27T23:45:28.214, value=Michael
 customers:lastname                                 timestamp=2022-09-27T23:45:28.243, value=Jordan
 customers:phone                                    timestamp=2022-09-27T23:45:28.442, value=312-455-4000
 customers:state                                    timestamp=2022-09-27T23:45:28.347, value=IL
 customers:zip                                      timestamp=2022-09-27T23:45:28.369, value=60612
1 row(s)
Took 0.0586 seconds
hbase:087:0>
```

Show the rows for orders that have not been paid yet using the following command:

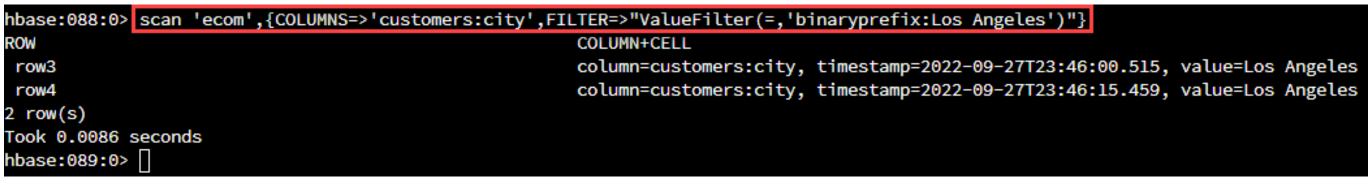
```
scan 'ecom',{ COLUMNS => 'orders:paid', FILTER=>"ValueFilter(=,'binaryprefix:N')" }
```



```
hbase:087:0> scan 'ecom',{ COLUMNS => 'orders:paid', FILTER=>"ValueFilter(=,'binaryprefix:N')" }
ROW                                                 COLUMN+CELL
 row4                                               column=orders:paid, timestamp=2022-09-27T23:47:23.099, value=N
1 row(s)
Took 0.0562 seconds
hbase:088:0>
```

Retrieve the rows with orders made from Los Angeles, California using the following command:

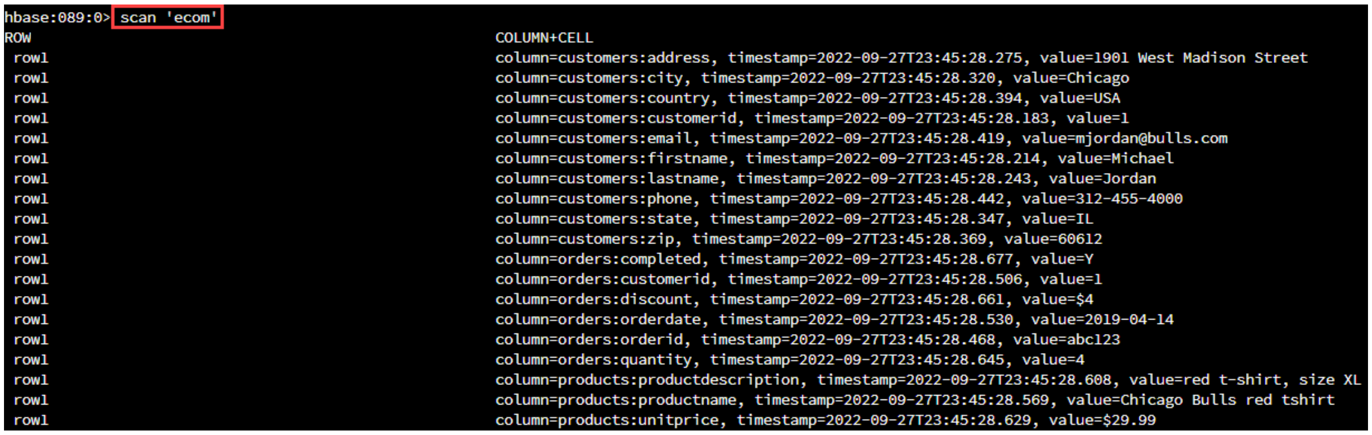
```
scan 'ecom',{COLUMNS=>'customers:city',FILTER=>"ValueFilter(=,'binaryprefix:Los Angeles')"}
```



```
hbase:088:0> scan 'ecom',{COLUMNS=>'customers:city',FILTER=>"ValueFilter(=,'binaryprefix:Los Angeles')"}
ROW                                                 COLUMN+CELL
 row3                                               column=customers:city, timestamp=2022-09-27T23:46:00.515, value=Los Angeles
 row4                                               column=customers:city, timestamp=2022-09-27T23:46:15.459, value=Los Angeles
2 row(s)
Took 0.0086 seconds
hbase:089:0>
```

Verify the **ecom** table records using the following command:

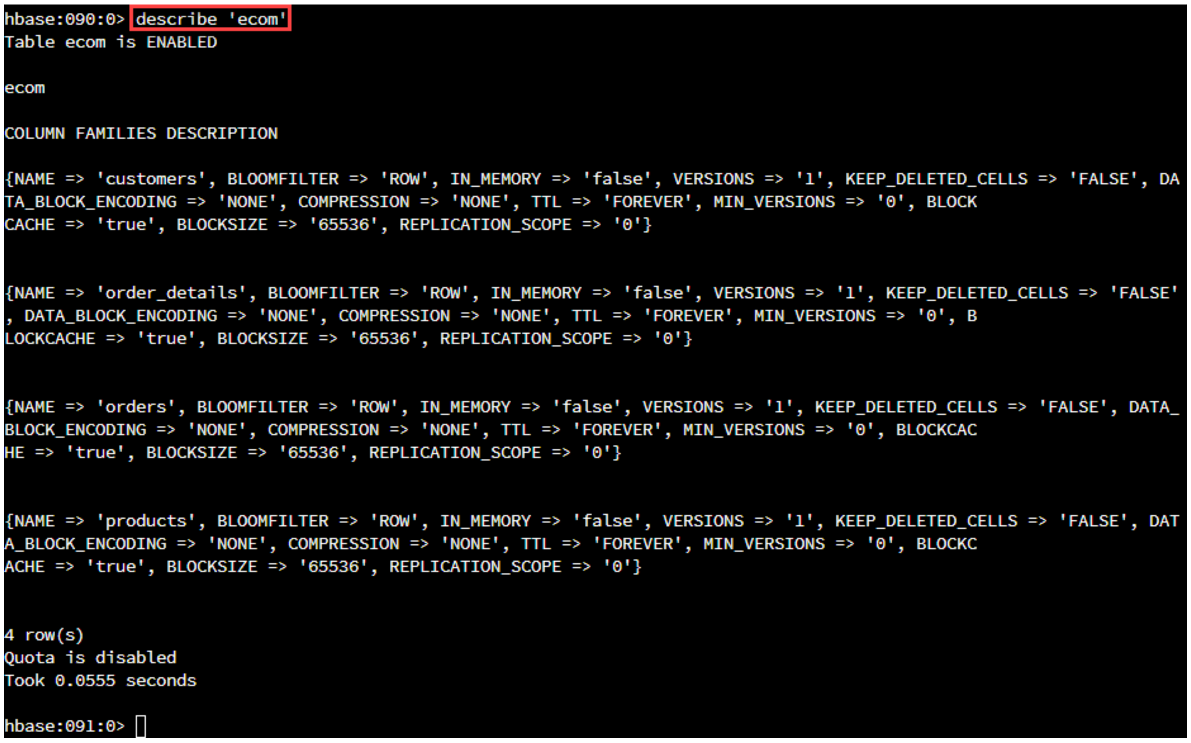
```
scan 'ecom'
```



```
hbase:089:0> scan 'ecom'
ROW                                                     COLUMN+CELL
 row1                                                   column=customers:address, timestamp=2022-09-27T23:45:28.275, value=1901 West Madison Street
 row1                                                   column=customers:city, timestamp=2022-09-27T23:45:28.320, value=Chicago
 row1                                                   column=customers:country, timestamp=2022-09-27T23:45:28.394, value=USA
 row1                                                   column=customers:customerid, timestamp=2022-09-27T23:45:28.183, value=1
 row1                                                   column=customers:email, timestamp=2022-09-27T23:45:28.419, value=mjordan@bulls.com
 row1                                                   column=customers:firstname, timestamp=2022-09-27T23:45:28.214, value=Michael
 row1                                                   column=customers:lastname, timestamp=2022-09-27T23:45:28.243, value=Jordan
 row1                                                   column=customers:phone, timestamp=2022-09-27T23:45:28.442, value=312-455-4000
 row1                                                   column=customers:state, timestamp=2022-09-27T23:45:28.347, value=IL
 row1                                                   column=customers:zip, timestamp=2022-09-27T23:45:28.369, value=60612
 row1                                                   column=orders:completed, timestamp=2022-09-27T23:45:28.677, value=Y
 row1                                                   column=orders:customerid, timestamp=2022-09-27T23:45:28.506, value=1
 row1                                                   column=orders:discount, timestamp=2022-09-27T23:45:28.661, value=$4
 row1                                                   column=orders:orderdate, timestamp=2022-09-27T23:45:28.530, value=2019-04-14
 row1                                                   column=orders:orderid, timestamp=2022-09-27T23:45:28.468, value=abc123
 row1                                                   column=orders:quantity, timestamp=2022-09-27T23:45:28.645, value=4
 row1                                                   column=products:productdescription, timestamp=2022-09-27T23:45:28.608, value=red t-shirt, size XL
 row1                                                   column=products:productname, timestamp=2022-09-27T23:45:28.569, value=Chicago Bulls red tshirt
 row1                                                   column=products:unitprice, timestamp=2022-09-27T23:45:28.629, value=$29.99
```

View the **ecom** table metadata using the following command:

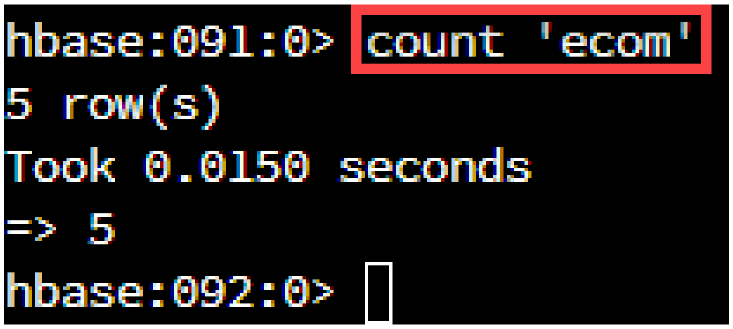
```
describe 'ecom'
```



```
hbase:090:0> describe 'ecom'
Table ecom is ENABLED

ecom

COLUMN FAMILIES DESCRIPTION

{NAME => 'customers', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE', DA
TA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCK
CACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}


{NAME => 'order_details', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE'
, DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', B
LOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}


{NAME => 'orders', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE', DATA_
BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCAC
HE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}


{NAME => 'products', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE', DAT
A_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKC
ACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}


4 row(s)
Quota is disabled
Took 0.0555 seconds

hbase:091:0>
```

Get the count of the records/rows in the **ecom** table using the following command:

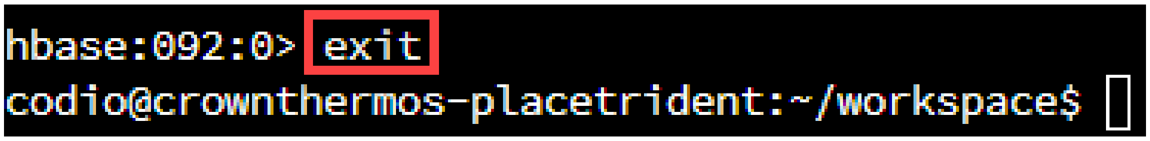
```
count 'ecom'
```



```
hbase:091:0> count 'ecom'
5 row(s)
Took 0.0150 seconds
=> 5
hbase:092:0>
```

Enter the **`exit`** command at the Apache HBase prompt in the Terminal window and then hit the **ENTER** key to exit the Apache HBase client application.

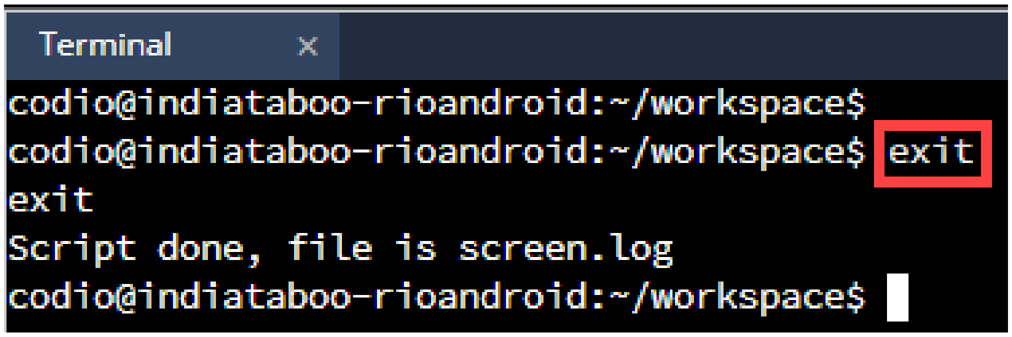
```
exit
```



```
hbase:092:0> exit
codio@crownthermos-placetrident:~/workspace$
```

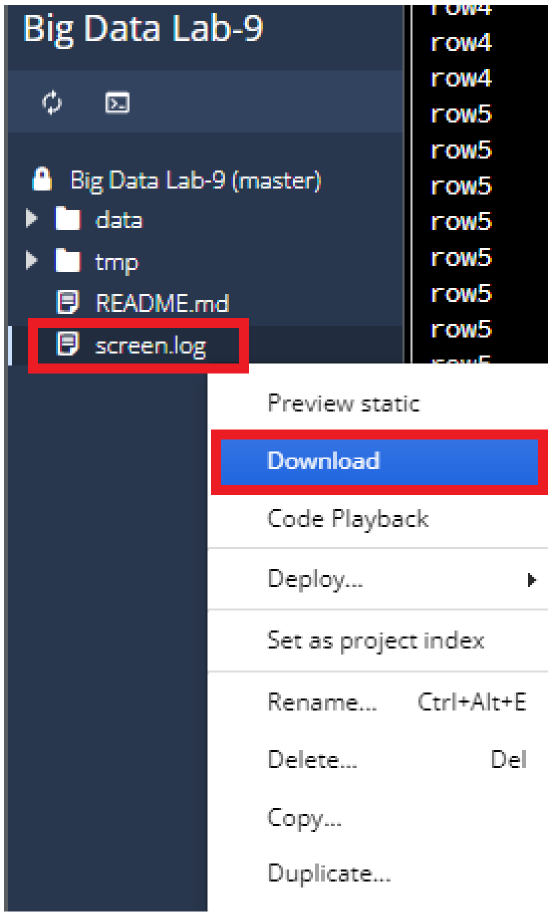
Enter the **`exit`** command at the Linux Terminal prompt and then hit the **ENTER** key to stop writing to the **screen.log** file.

```
exit
```

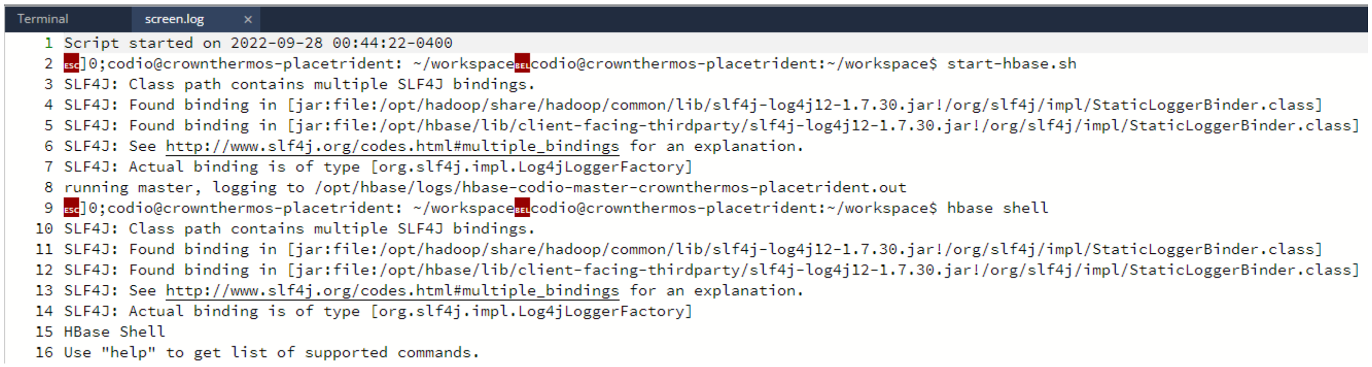


```
codio@indiataboo-rioandroid:~/workspace$
codio@indiataboo-rioandroid:~/workspace$ exit
exit
Script done, file is screen.log
codio@indiataboo-rioandroid:~/workspace$
```

In the Codio file tree, right-click on the **screen.log** file and download it to an accessible location on your personal computer system. You will need to provide this file in your assignment submittal. This file can be viewed in a text editor like Microsoft Notepad.



Here is a snippet example of the **screen.log** contents.



```
Script started on 2022-09-28 00:44:22-0400
␛]0;codio@crownthermos-placetrident: ~/workspace␇codio@crownthermos-placetrident:~/workspace$ start-hbase.sh
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/hbase/lib/client-facing-thirdparty/slf4j-log4j12-1.7.30.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]
running master, logging to /opt/hbase/logs/hbase-codio-master-crownthermos-placetrident.out
␛]0;codio@crownthermos-placetrident: ~/workspace␇codio@crownthermos-placetrident:~/workspace$ hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/hbase/lib/client-facing-thirdparty/slf4j-log4j12-1.7.30.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
Use "help" to get list of supported commands.
```