Домашнее задание

1. Подключить к Cassandra

```
[student1_13@bigdataanalytics-head-0 ~]$ /cassandra/bin/cqlsh 10.0.0.18
-bash: /cassandra/bin/cqlsh: No such file or directory
[student1_13@bigdataanalytics-head-0 ~]$ ssh 10.0.0.18
The authenticity of host '10.0.0.18 (10.0.0.18)' can't be established.
ECDSA key fingerprint is SHA256:FlX3eC5ZoVY/Ep1XEU1GV6VgbCtLwwMtBS2PDcYYOLM.
ECDSA key fingerprint is MD5:cd:54:0e:25:bb:b8:9d:04:92:57:89:af:db:12:cd:ed.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '10.0.0.18' (ECDSA) to the list of known hosts.
[student1_13@bigdataanalytics-worker-2 ~]$ /cassandra/bin/cqlsh 10.0.0.18
Connection error: ('Unable to connect to any servers', {'10.0.0.18': error(111,
"Tried connecting to [('10.0.0.18', 9042)]. Last error: Connection refused")})
```

2. Создать таблицы

3. Вставить записи

4. Изучить особенности работы where

- ***Cassandra***
- *Подключаемся к Cassandra на worker-2: /cassandra/bin/cqlsh 10.0.0.18*
- *Создаем пространство ключей: CREATE KEYSPACE lesson7
  WITH REPLICATION = {
  'class' : 'SimpleStrategy', 'replication_factor' : 1 }*
- *Создаем таблицу и вставляем значения: CREATE TABLE animals
  (id int,
  name text,
  size text,
  primary key (id));
  insert into animals (id, name, size) values (3, 'Deer', 'Big');*
- *Проверяем как работает фильтрация: select * from animals
  where id = 3 and name = '12321';*
- *Сравниваем удаление и вставку пустого значения: delete id from animals where
  id = 1;*
- *insert into animals (id, name, size) values (3, null, null);*

5. Подключиться к HBase

```
[student1_13@bigdataanalytics-worker-2 ~]$ hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
```

```
SLF4J: Found binding in [jar:file:/usr/hdp/3.1.4.0-315/phoenix/
phoenix-5.0.0.3.1.4.0-315-server.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/hdp/3.1.4.0-315/hadoop/lib/slf4j-
log4j12-1.7.25.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 2.0.2.3.1.4.0-315, r, Fri Aug 23 05:15:48 UTC 2019
Took 0.0016 seconds
hbase(main):001:0>
```

6. Создать таблицы и вставить значения

```
hbase(main):001:0> create_namespace 'lesson7'

ERROR: KeeperErrorCode = NoNode for /hbase-unsecure/master

Create namespace; pass namespace name,
and optionally a dictionary of namespace configuration.
Examples:

  hbase> create_namespace 'ns1'
  hbase> create_namespace 'ns1', {'PROPERTY_NAME'=>'PROPERTY_VALUE'}

Took 8.3662 seconds
hbase(main):002:0> create 'lesson7:animals', 'name', 'size'

ERROR: KeeperErrorCode = NoNode for /hbase-unsecure/master

Creates a table. Pass a table name, and a set of column family
specifications (at least one), and, optionally, table configuration.
Column specification can be a simple string (name), or a dictionary
(dictionaries are described below in main help output), necessarily
including NAME attribute.
Examples:

Create a table with namespace=ns1 and table qualifier=t1
  hbase> create 'ns1:t1', {NAME => 'f1', VERSIONS => 5}

Create a table with namespace=default and table qualifier=t1
  hbase> create 't1', {NAME => 'f1'}, {NAME => 'f2'}, {NAME => 'f3'}
  hbase> # The above in shorthand would be the following:
  hbase> create 't1', 'f1', 'f2', 'f3'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 1, TTL => 2592000, BLOCKCACHE
=> true}
  hbase> create 't1', {NAME => 'f1', CONFIGURATION =>
{'hbase.hstore.blockingStoreFiles' => '10'}}
  hbase> create 't1', {NAME => 'f1', IS_MOB => true, MOB_THRESHOLD => 1000000,
MOB_COMPACT_PARTITION_POLICY => 'weekly'}

Table configuration options can be put at the end.
Examples:

  hbase> create 'ns1:t1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS_FILE => 'splits.txt', OWNER => 'johndoe'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 5}, METADATA => { 'mykey' =>
'myvalue' }
  hbase> # Optionally pre-split the table into NUMREGIONS, using
  hbase> # SPLITALGO ("HexStringSplit", "UniformSplit" or classname)
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit'}
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit',
REGION_REPLICATION => 2, CONFIGURATION =>
{'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}}
  hbase> create 't1', {NAME => 'f1', DFS_REPLICATION => 1}

You can also keep around a reference to the created table:

  hbase> t1 = create 't1', 'f1'

Which gives you a reference to the table named 't1', on which you can then
call methods.

Took 8.1717 seconds

hbase(main):004:0> put 'lesson7:animals', '3', 'name', 'Deer'

^[[A

ERROR: Connection refused

Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates.  To put a cell value into table 'ns1:t1' or 't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

  hbase> put 'ns1:t1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value', ts1
  hbase> put 't1', 'r1', 'c1', 'value', {ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|SECRET'}

The same commands also can be run on a table reference. Suppose you had a
reference
t to table 't1', the corresponding command would be:

  hbase> t.put 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}

Took 139.7062 seconds
hbase(main):005:0>
hbase(main):006:0* put 'lesson7:animals', '3', 'name', 'Deer'

ERROR: Connection refused

Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates.  To put a cell value into table 'ns1:t1' or 't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

  hbase> put 'ns1:t1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value', ts1
  hbase> put 't1', 'r1', 'c1', 'value', {ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|SECRET'}
```

```
The same commands also can be run on a table reference. Suppose you had a
reference
t to table 't1', the corresponding command would be:

  hbase> t.put 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}

Took 138.7584 seconds
hbase(main):007:0> show namespaces
NameError: undefined local variable or method `namespaces' for main:Object

hbase(main):008:0> show namespace
NameError: undefined local variable or method `namespace' for main:Object

hbase(main):009:0> create 'lesson7:animals', {NAME => 'name', VERSIONS => 3}

ERROR: KeeperErrorCode = NoNode for /hbase-unsecure/master

Creates a table. Pass a table name, and a set of column family
specifications (at least one), and, optionally, table configuration.
Column specification can be a simple string (name), or a dictionary
(dictionaries are described below in main help output), necessarily
including NAME attribute.
Examples:

Create a table with namespace=ns1 and table qualifier=t1
  hbase> create 'ns1:t1', {NAME => 'f1', VERSIONS => 5}

Create a table with namespace=default and table qualifier=t1
  hbase> create 't1', {NAME => 'f1'}, {NAME => 'f2'}, {NAME => 'f3'}
  hbase> # The above in shorthand would be the following:
  hbase> create 't1', 'f1', 'f2', 'f3'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 1, TTL => 2592000, BLOCKCACHE
=> true}
  hbase> create 't1', {NAME => 'f1', CONFIGURATION =>
{'hbase.hstore.blockingStoreFiles' => '10'}}
  hbase> create 't1', {NAME => 'f1', IS_MOB => true, MOB_THRESHOLD => 1000000,
MOB_COMPACT_PARTITION_POLICY => 'weekly'}

Table configuration options can be put at the end.
Examples:

  hbase> create 'ns1:t1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS_FILE => 'splits.txt', OWNER => 'johndoe'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 5}, METADATA => { 'mykey' =>
'myvalue' }
  hbase> # Optionally pre-split the table into NUMREGIONS, using
  hbase> # SPLITALGO ("HexStringSplit", "UniformSplit" or classname)
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit'}
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit',
REGION_REPLICATION => 2, CONFIGURATION =>
{'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}}
  hbase> create 't1', {NAME => 'f1', DFS_REPLICATION => 1}

You can also keep around a reference to the created table:

  hbase> t1 = create 't1', 'f1'

Which gives you a reference to the table named 't1', on which you can then
call methods.
```

```
Took 8.1909 seconds


hbase(main):001:0> create_namespace 'homework7'

ERROR: KeeperErrorCode = NoNode for /hbase-unsecure/master

Create namespace; pass namespace name,
and optionally a dictionary of namespace configuration.
Examples:

  hbase> create_namespace 'ns1'
  hbase> create_namespace 'ns1', {'PROPERTY_NAME'=>'PROPERTY_VALUE'}

Took 15.3151 seconds
hbase(main):002:0> create_namespace 'hw7'

ERROR: KeeperErrorCode = NoNode for /hbase-unsecure/master

Create namespace; pass namespace name,
and optionally a dictionary of namespace configuration.
Examples:

  hbase> create_namespace 'ns1'
  hbase> create_namespace 'ns1', {'PROPERTY_NAME'=>'PROPERTY_VALUE'}

Took 18.0665 seconds
```

7. Изучить особенности хранения данных

- ***HBase***
- *Подключаемся к HBase hbase shell*
  create_namespace 'lesson7'
  create 'lesson7:animals', 'name', 'size'

- *Вставляем значения:*
  put 'lesson7:animals', '3', 'name', 'Deer' put 'lesson7:animals', '3', 'size', 'Big'
  put 'lesson7:animals', '5', 'name', 'Snake' put 'lesson7:animals', '3', 'name', 'Doe'

- *Удаляем значение:*
  delete 'lesson7:animals', '5'

- *Делаем запрос к созданной таблице: get 'lesson7:animals', '5'*