Apache Cassandra:
Core Concepts, Skills, and Tools

Installing, configuring, and running Cassandra locally
Exercise Workbook

Joe Chu
Leo Schuman
October 20, 2014

Exercise 1: Start up the lab environment

In this exercise, you will:

- Start the lab virtual machine
- Survey the system configuration

Steps

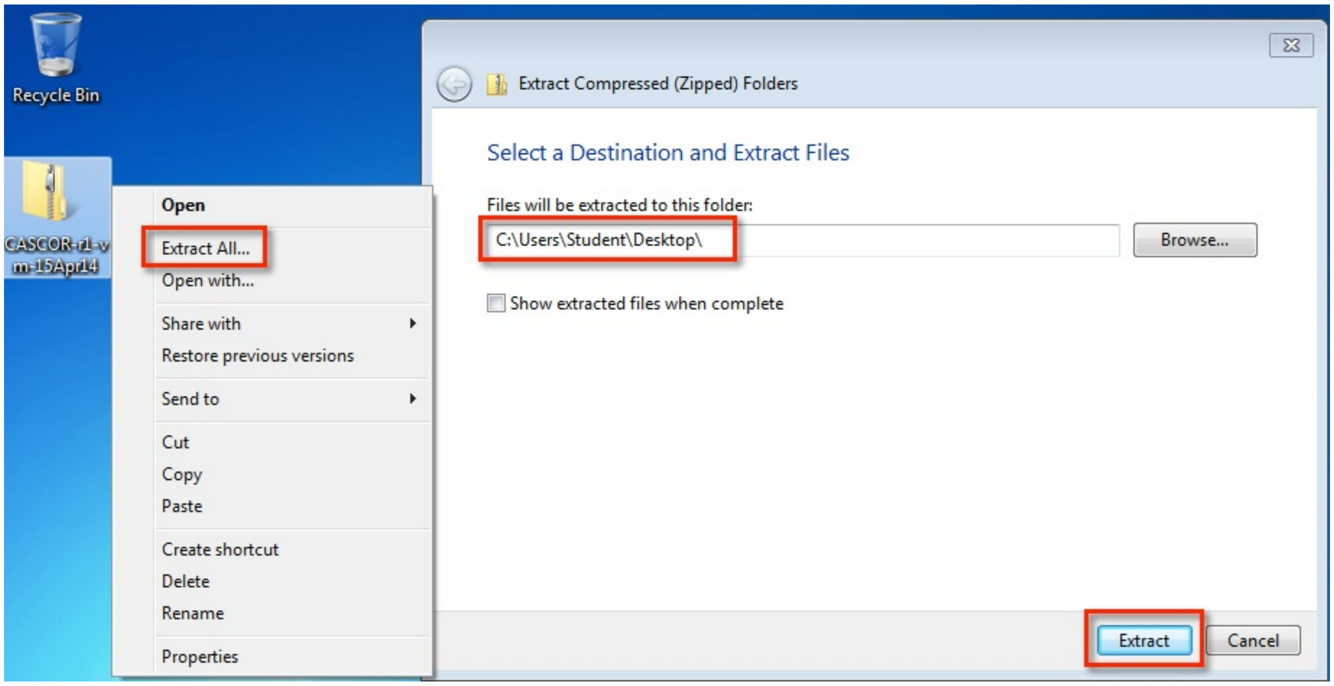
Start the lab virtual machine

1. Verify that your system has *VMware Player*, *Oracle VirtualBox*, or another virtual machine software installed.

2. On your computer, copy the file *CASCOR-r2-vm-[timestamp].zip* to the desktop.

If a computer is provided, the virtual machine may already be installed. Your instructor should provide additional instructions on what to do, and what steps you are able to skip.

3. From the desktop, decompress the file *CASCOR-r2-vm-*[timestamp]*.zip*.



Select a Destination and Extract Files

Files will be extracted to this folder:

C:\Users\Student\Desktop\

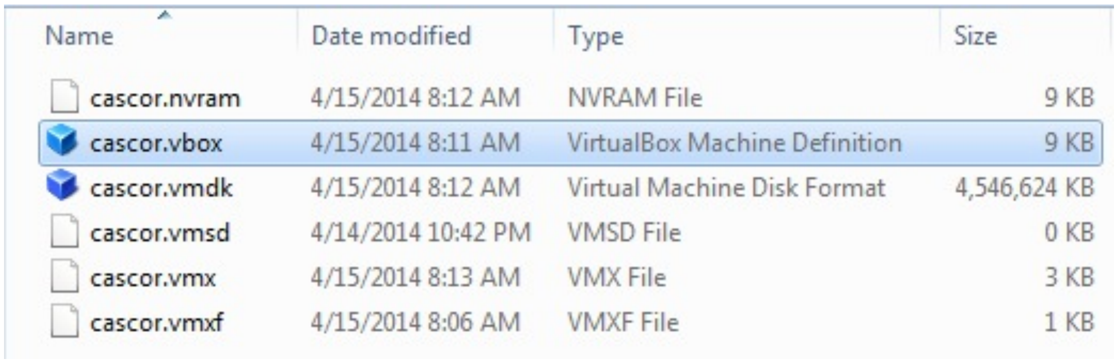
Show extracted files when complete

A folder called CASCOR-r2-vm-[timestamp] *should have been created.*

Copyright 2014 DataStax

4. Double-click *CASCOR-r2-vm-[timestamp]* to open the folder.

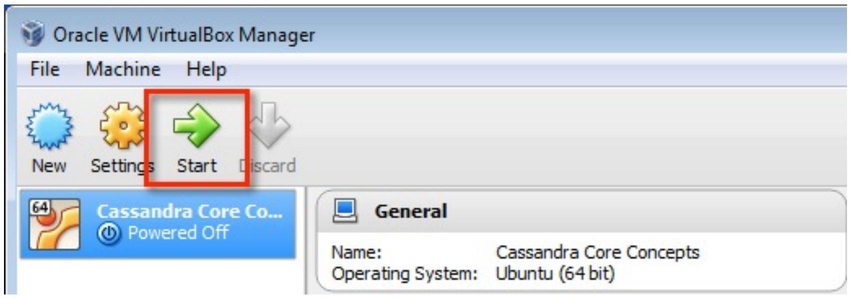
5. In the *CASCOR-r2-vm-[timestamp]* folder, double-click the appropriate file: *cascor.vmx* for *VMware Player* or *cascor.vbox* for *Oracle VirtualBox*.



Name	Date modified	Type	Size
cascor.nvram	4/15/2014 8:12 AM	NVRAM File	9 KB
cascor.vbox	4/15/2014 8:11 AM	VirtualBox Machine Definition	9 KB
cascor.vmdk	4/15/2014 8:12 AM	Virtual Machine Disk Format	4,546,624 KB
cascor.vmsd	4/14/2014 10:42 PM	VMSD File	0 KB
cascor.vmx	4/15/2014 8:13 AM	VMX File	3 KB
cascor.vmxf	4/15/2014 8:06 AM	VMXF File	1 KB

If your operating system hides file type extensions, you may need to unhide them to find the correct file. Double-clicking the file will start the appropriate virtual machine software.

6. From the VM software, make sure that the *Cassandra Core Concepts* virtual machine is selected and press the *Start* button.

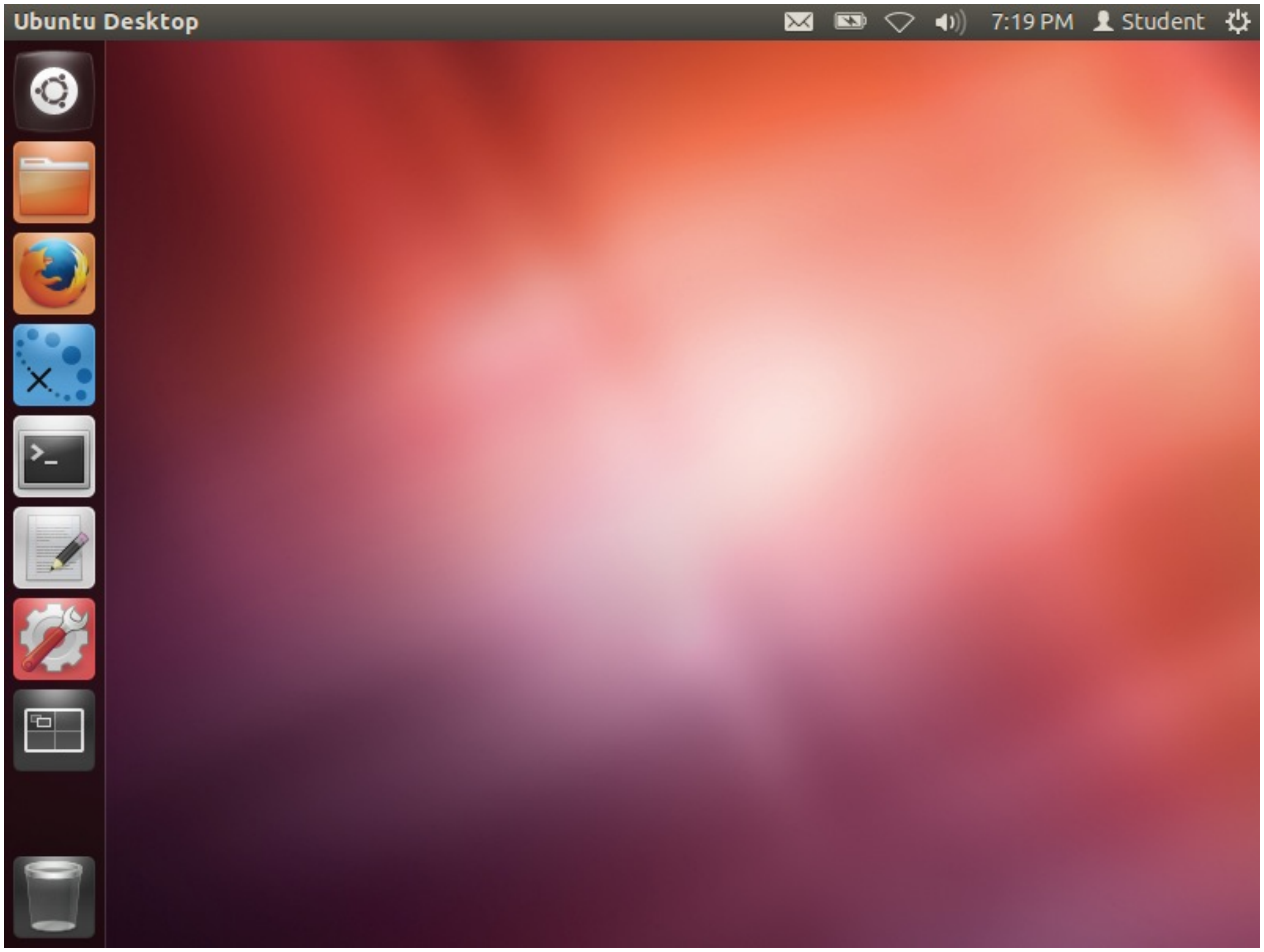


Name: Cassandra Core Concepts
Operating System: Ubuntu (64 bit)

Copyright 2014 DataStax

7. Verify the virtual machine has launched.

The VM has successfully started once you see the Ubuntu desktop.



Survey the system configuration

8. On the left hand side of the Ubuntu desktop, find the launcher for *Firefox*.

This is the default web browser provided in Ubuntu. This may be used if downloading Apache Cassandra, or interacting with DataStax OpsCenter and Apache Solr.

Copyright 2014 DataStax

9. Find the launcher for *DataStax DevCenter*.

This is a development IDE that will be used later for running CQL queries in Cassandra.

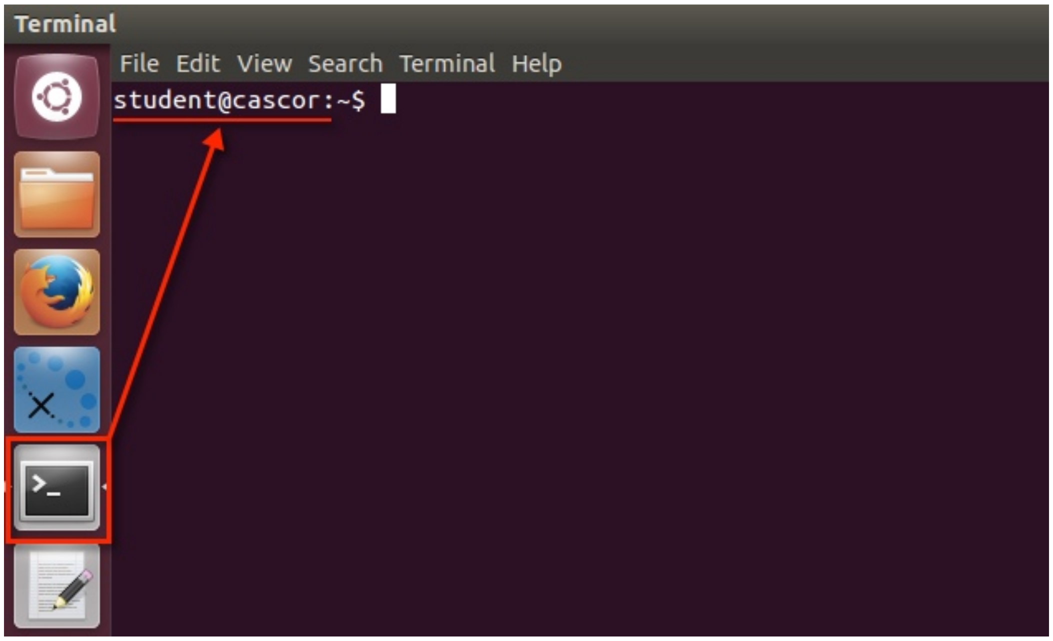
10. Find the launcher for *Terminal*.

The bulk of the exercises and demos and in this course will be done using Terminal windows.

11. Find the launcher for *gedit*.

This graphical text editor can be used for file editing if you are not familiar with any of the command line editors.

12. On the left side of the Ubuntu desktop, open a Terminal window.



```
student@cascor:~$
```

Copyright 2014 DataStax

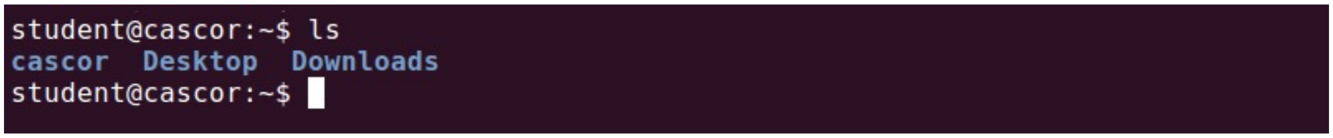
13. In the Terminal window, determine the current starting directory.

```
pwd
```

The current directory, or path, is /home/student. *This is the home directory for the student user account that you are logged in as.*

14. List the contents of the current directory.

```
ls
```



```
student@cascor:~$ ls
cascor  Desktop  Downloads
student@cascor:~$ 
```

There may be several directories here. The important one is the cascor *directory, which contains the files and scripts necessary to do the demos and exercises for this course.*

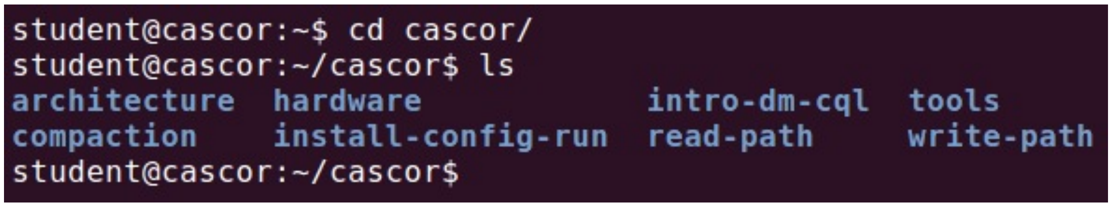
15. Navigate to the *cascor* directory.

If you have entered three or more characters of a file or path name, you may press Tab to auto-complete the rest of the file or path name.

```
cd cascor/
```

16. In the *cascor* directory, list the file and directory contents.

```
ls
```



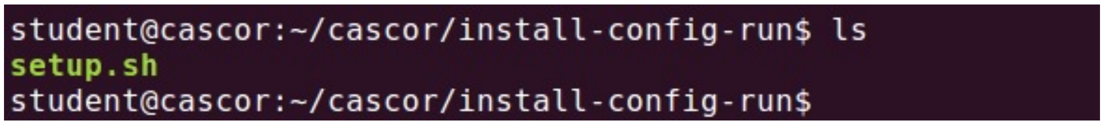
```
student@cascor:~$ cd cascor/
student@cascor:~/cascor$ ls
architecture  hardware            intro-dm-cql  tools
compaction    install-config-run  read-path     write-path
student@cascor:~/cascor$
```

In the cascor *directory there are additional directories, one for each course module that has a corresponding workbook.*

Copyright 2014 DataStax

17. Navigate to the *install-config-run* directory and view the file and directory contents.

```
cd install-config-run
ls
```



```
student@cascor:~/cascor/install-config-run$ ls
setup.sh
student@cascor:~/cascor/install-config-run$
```

Each of the module directories may have additional directories for exercises and demos. If there are any demos or exercises that do need additional files, a directory is created for it. However, not every demo or exercise has its own directory.

Additional instructions to use those files can be found in the workbook with the corresponding demo or exercise.

18. In the *install-config-run* directory, run the script *setup.sh*.

```
./setup.sh
```

If you ever want to reset your virtual machine to a state where you can start from the beginning of a workbook, run the setup.sh *script in the corresponding directory. Running the script in this step should not make any changes if this VM was not used previously.*

END OF EXERCISE

Copyright 2014 DataStax

Exercise 2: Install Cassandra

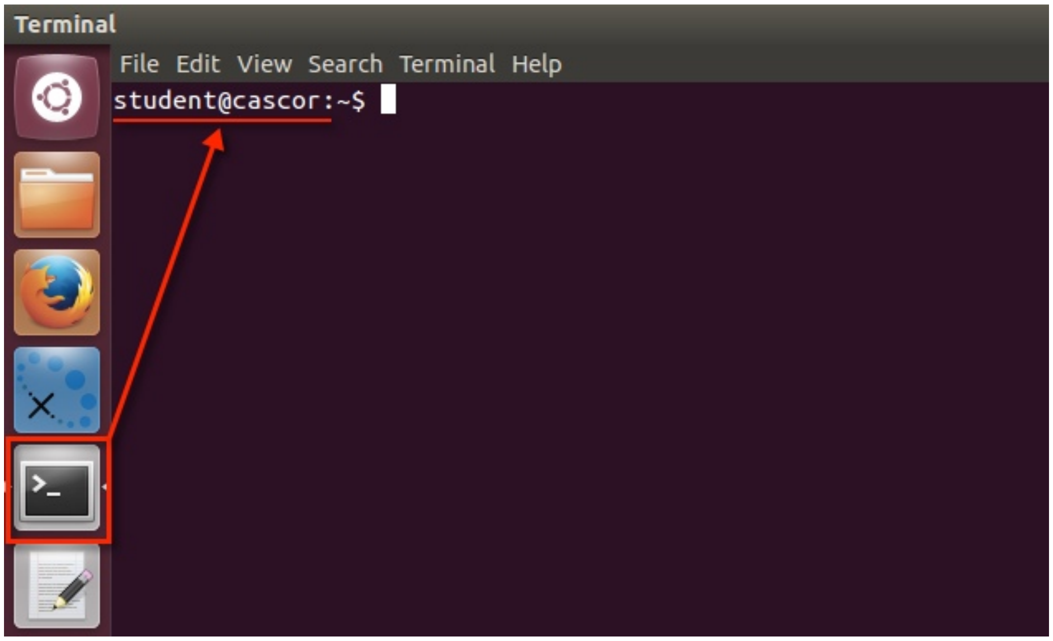
In this exercise, you will:

- Verify the prerequisites for Cassandra
- Install Cassandra using the binary tarball
- Explore the Cassandra directories

Steps

Verify the prerequisites for Cassandra

1. On the left sidebar in the virtual machine, click on the Terminal launcher.



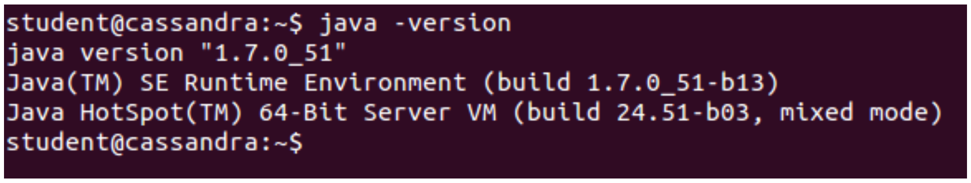
```
student@cascor:~$
```

If you are not familiar with the Linux terminal or command lines, first check out the Appendix at the end of this workbook for a quick primer.

Copyright 2014 DataStax

2. In your Terminal window, use the *java -version* command to check the currently installed Java version.

```
java -version
```

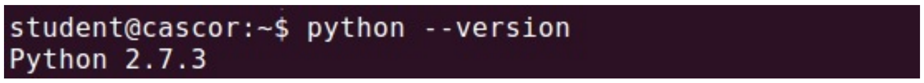


```
student@cassandra:~$ java -version
java version "1.7.0_51"
Java(TM) SE Runtime Environment (build 1.7.0_51-b13)
Java HotSpot(TM) 64-Bit Server VM (build 24.51-b03, mixed mode)
student@cassandra:~$
```

The minimum version of Java supported by Cassandra 2.0 is 1.7.0_25. If this step shows a lower version, the Java JRE will need to be upgraded to at least 1.7.0_25.

3. Use the *python --version* command to check the current installed Python version.

```
python --version
```



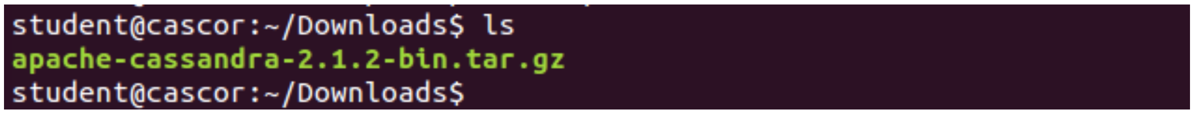
```
student@cascor:~$ python --version
Python 2.7.3
```

There is a dependency for Python because some Cassandra tools are written as Python scripts. Python 2.6 and 2.7 are supported, but newer versions are not currently compatible with Cassandra.

Install Cassandra using the binary tarball

4. In the Terminal window, navigate to the directory where the Cassandra binary tarball is stored on the virtual machine.

```
cd /home/student/Downloads
```



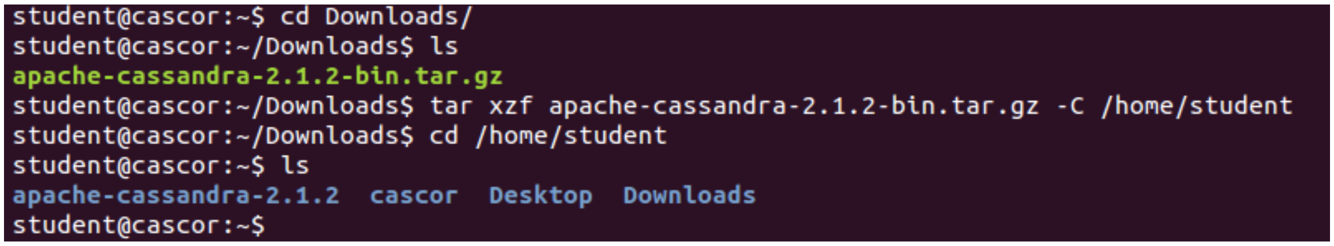
```
student@cascor:~/Downloads$ ls
apache-cassandra-2.1.2-bin.tar.gz
student@cascor:~/Downloads$
```

The Apache Cassandra version found in the VM is usually the most recent and may defer from the version in the screenshot and workbook instructions. Ask your instructor if you are unsure of which version to use.

Copyright 2014 DataStax

5. Use the *tar xzf* command, as shown, to decompress the binary tarball file to the user account home directory: */home/student*

```
tar xzf apache-cassandra-2.1.2-bin.tar.gz -C /home/student
```

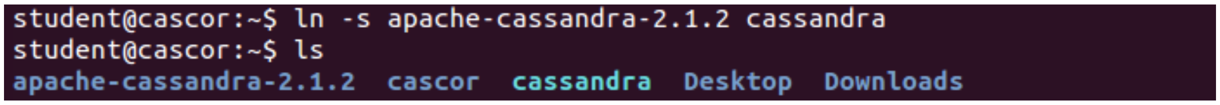


```
student@cascor:~$ cd Downloads/
student@cascor:~/Downloads$ ls
apache-cassandra-2.1.2-bin.tar.gz
student@cascor:~/Downloads$ tar xzf apache-cassandra-2.1.2-bin.tar.gz -C /home/student
student@cascor:~/Downloads$ cd /home/student
student@cascor:~$ ls
apache-cassandra-2.1.2  cascor  Desktop  Downloads
student@cascor:~$
```

The binary tarball file name and name of the decompressed directory may be different depending on which version of Cassandra you downloaded.

6. Use the *ln* command to create a symbolic link called *cassandra* to the *apache-cassandra-2.1.2* directory.

```
ln -s apache-cassandra-2.1.2 cassandra
```



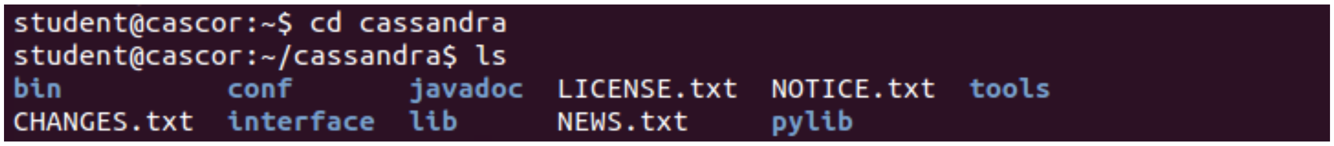
```
student@cascor:~$ ln -s apache-cassandra-2.1.2 cassandra
student@cascor:~$ ls
apache-cassandra-2.1.2  cascor  cassandra  Desktop  Downloads
```

This is done to minimize confusion over minor Cassandra releases. All Cassandra 2.1.x releases should exhibit all behavior taught in this course.

Explore the Cassandra directories

7. In the Terminal window, navigate to the decompressed Cassandra directory, and use the *ls* command to list and review the directory contents.

```
cd /home/student/cassandra
ls
```

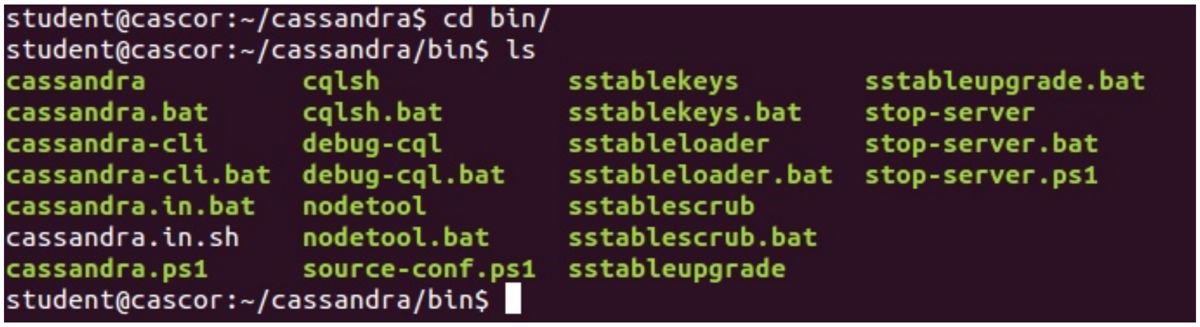


```
student@cascor:~$ cd cassandra
student@cascor:~/cassandra$ ls
bin          conf       javadoc  LICENSE.txt  NOTICE.txt  tools
CHANGES.txt  interface  lib      NEWS.txt     pylib
```

Copyright 2014 DataStax

8. Navigate to the *bin* sub-directory, then use the *ls* command to list and review the directory contents.

```
cd bin/
ls
```

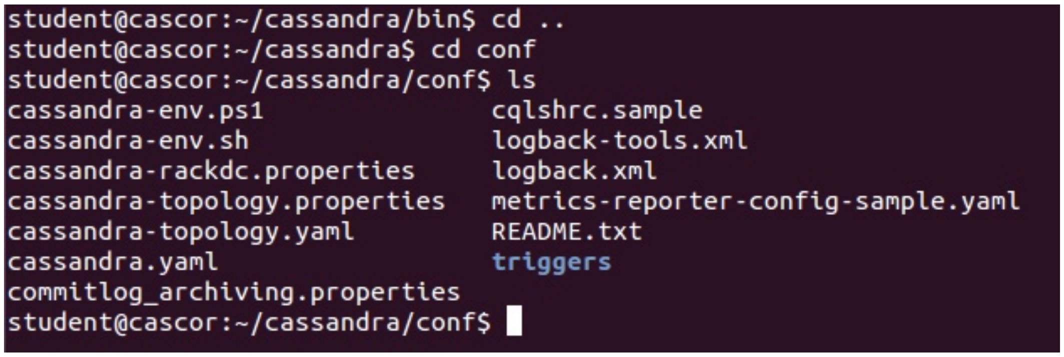


```
student@cascor:~/cassandra$ cd bin/
student@cascor:~/cassandra/bin$ ls
cassandra          cqlsh           sstablekeys        sstableupgrade.bat
cassandra.bat      cqlsh.bat       sstablekeys.bat    stop-server
cassandra-cli      debug-cql       sstableloader      stop-server.bat
cassandra-cli.bat  debug-cql.bat   sstableloader.bat  stop-server.ps1
cassandra.in.bat   nodetool        sstablescrub
cassandra.in.sh    nodetool.bat    sstablescrub.bat
cassandra.ps1      source-conf.ps1 sstableupgrade
student@cascor:~/cassandra/bin$ 
```

This directory contains the Cassandra binaries and scripts. The binaries that will be used most often in the exercises are cassandra, cqlsh, and nodetool.

9. Navigate to the *conf* directory, then list and review the directory contents.

```
cd ..
cd conf/
ls
```



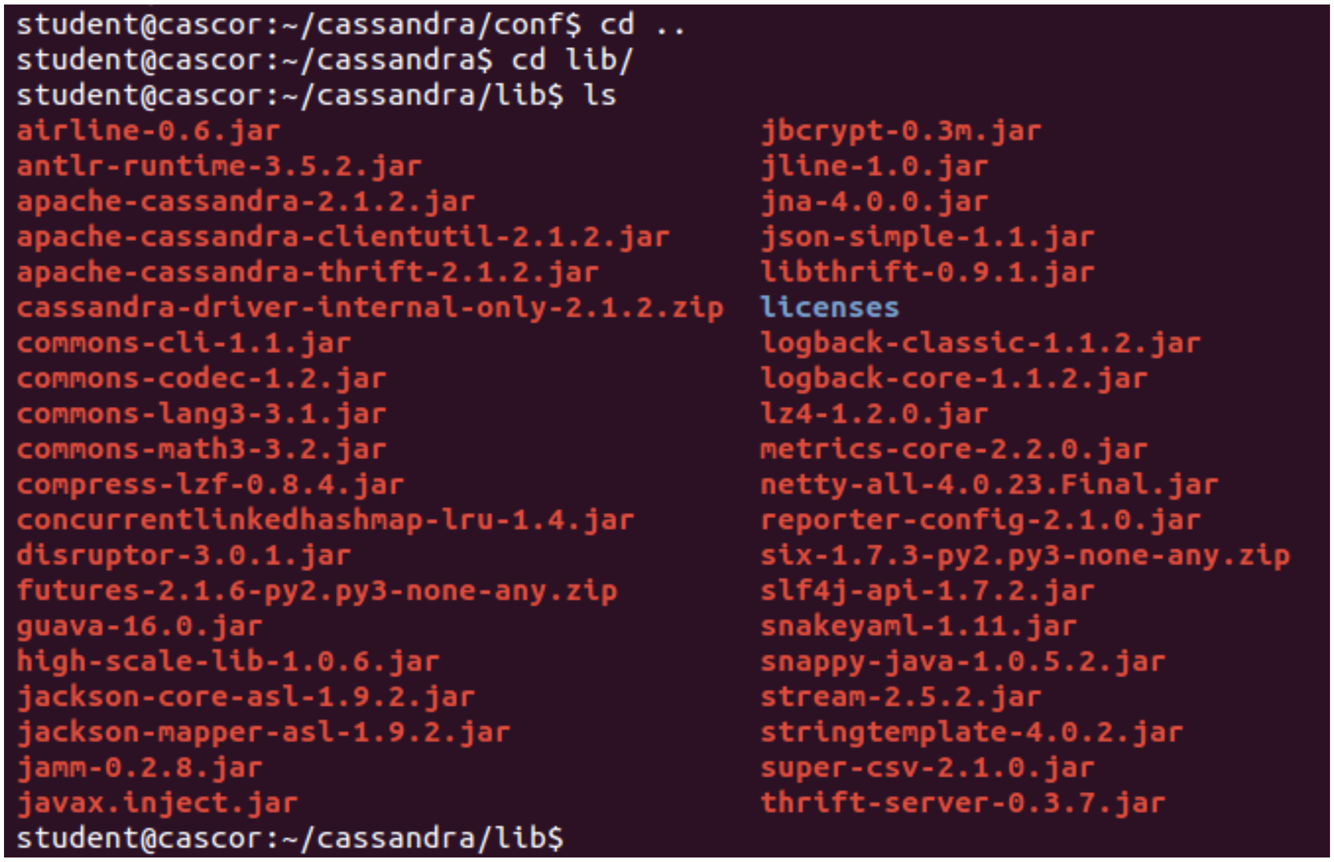
```
student@cascor:~/cassandra/bin$ cd ..
student@cascor:~/cassandra$ cd conf
student@cascor:~/cassandra/conf$ ls
cassandra-env.ps1              cqlshrc.sample
cassandra-env.sh               logback-tools.xml
cassandra-rackdc.properties    logback.xml
cassandra-topology.properties  metrics-reporter-config-sample.yaml
cassandra-topology.yaml        README.txt
cassandra.yaml                 triggers
commitlog_archiving.properties
student@cascor:~/cassandra/conf$ 
```

This directory contains the different configuration files for Cassandra. We will be looking at some of these files in more detail in the next task.

Copyright 2014 DataStax

10. Navigate to the *lib* directory, then list and review the directory contents.

```
cd ..
cd lib/
ls
```



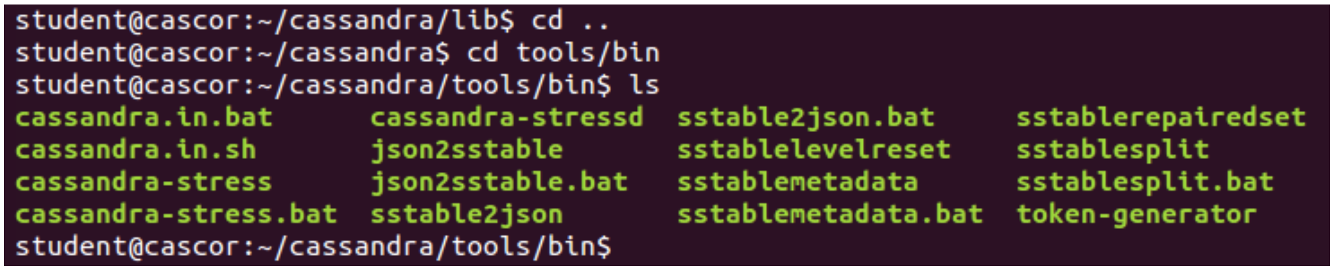
```
student@cascor:~/cassandra/conf$ cd ..
student@cascor:~/cassandra$ cd lib/
student@cascor:~/cassandra/lib$ ls
airline-0.6.jar                              jbcrypt-0.3m.jar
antlr-runtime-3.5.2.jar                      jline-1.0.jar
apache-cassandra-2.1.2.jar                   jna-4.0.0.jar
apache-cassandra-clientutil-2.1.2.jar        json-simple-1.1.jar
apache-cassandra-thrift-2.1.2.jar            libthrift-0.9.1.jar
cassandra-driver-internal-only-2.1.2.zip     licenses
commons-cli-1.1.jar                          logback-classic-1.1.2.jar
commons-codec-1.2.jar                        logback-core-1.1.2.jar
commons-lang3-3.1.jar                        lz4-1.2.0.jar
commons-math3-3.2.jar                        metrics-core-2.2.0.jar
compress-lzf-0.8.4.jar                       netty-all-4.0.23.Final.jar
concurrentlinkedhashmap-lru-1.4.jar          reporter-config-2.1.0.jar
disruptor-3.0.1.jar                          six-1.7.3-py2.py3-none-any.zip
futures-2.1.6-py2.py3-none-any.zip           slf4j-api-1.7.2.jar
guava-16.0.jar                               snakeyaml-1.11.jar
high-scale-lib-1.0.6.jar                     snappy-java-1.0.5.2.jar
jackson-core-asl-1.9.2.jar                   stream-2.5.2.jar
jackson-mapper-asl-1.9.2.jar                 stringtemplate-4.0.2.jar
jamm-0.2.8.jar                               super-csv-2.1.0.jar
javax.inject.jar                             thrift-server-0.3.7.jar
student@cascor:~/cassandra/lib$
```

This directory contains the Java dependencies that need to be included in the JVM classpath. The JNA library is also pre-packaged with the installation and can be found here as jna-.jar.*

Copyright 2014 DataStax

11. Navigate to the *tools/bin* directory, then list and review the directory contents.

```
cd ..
cd tools/bin
ls
```



```
student@cascor:~/cassandra/lib$ cd ..
student@cascor:~/cassandra$ cd tools/bin
student@cascor:~/cassandra/tools/bin$ ls
cassandra.in.bat      cassandra-stressd  sstable2json.bat     sstablerepairedset
cassandra.in.sh       json2sstable       sstablelevelreset    sstablesplit
cassandra-stress      json2sstable.bat   sstablemetadata      sstablesplit.bat
cassandra-stress.bat  sstable2json       sstablemetadata.bat  token-generator
student@cascor:~/cassandra/tools/bin$
```

These are the binaries and scripts for additional Cassandra tools. One tool that you will be using often in later exercises is cassandra-stress.

END OF EXERCISE

Copyright 2014 DataStax

Exercise 3: Configure a Cassandra instance

In this exercise, you will:

- Configure the *cassandra.yaml* file
- Configure the *cassandra-env.sh* file
- Configure the *logback.xml* file

Steps

Configure the *cassandra.yaml* file

1. In the Terminal window, navigate to the Cassandra *conf* directory.

```
cd /home/student/cassandra/conf
```

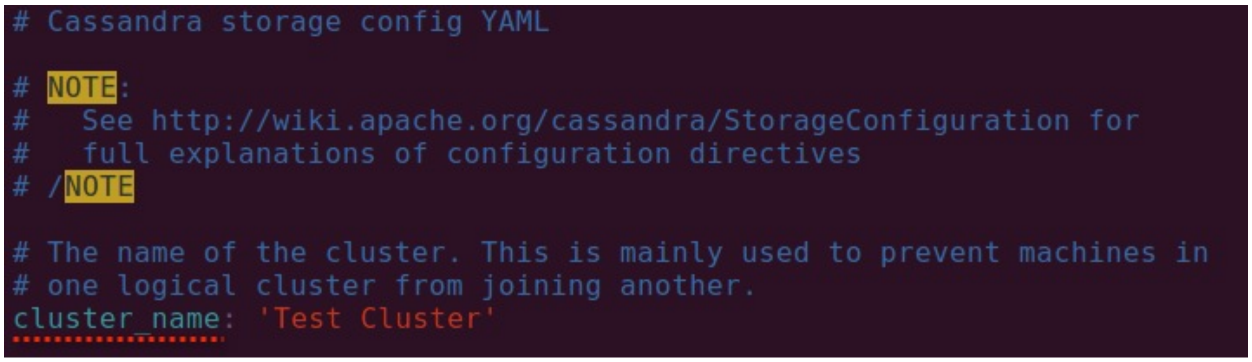
2. Use a text editor to open the *cassandra.yaml* file.

```
gedit cassandra.yaml
```

If you have a preferred text editor, you can use that instead of gedit. The lab environment comes with emacs, vim, and nano installed.

3. In the *cassandra.yaml* file, look for the *cluster_name* setting and change the default value to your name.

```
cluster_name: 'Your Name'
```



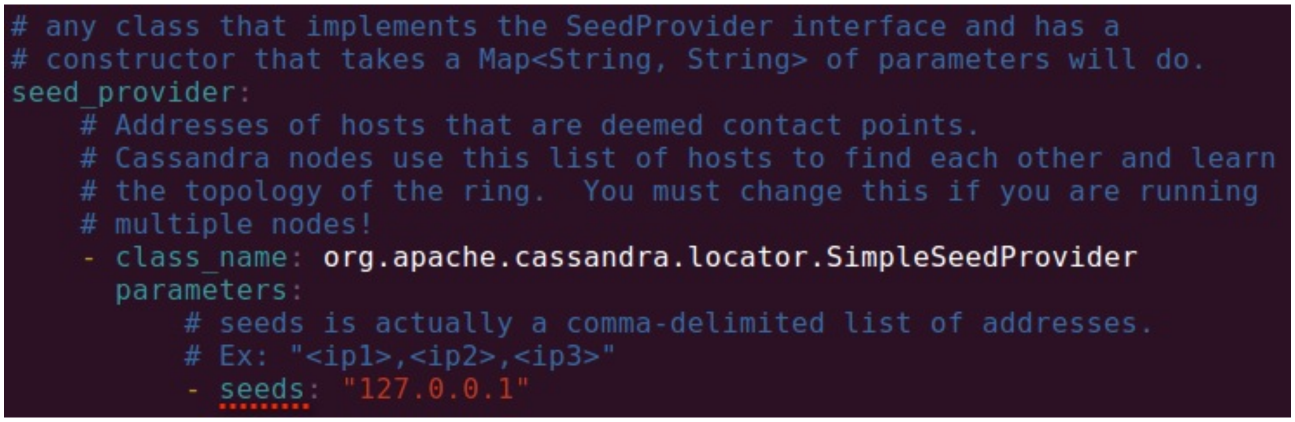
```
# Cassandra storage config YAML

# NOTE:
#   See http://wiki.apache.org/cassandra/StorageConfiguration for
#   full explanations of configuration directives
# /NOTE

# The name of the cluster. This is mainly used to prevent machines in
# one logical cluster from joining another.
cluster_name: 'Test Cluster'
```

Copyright 2014 DataStax

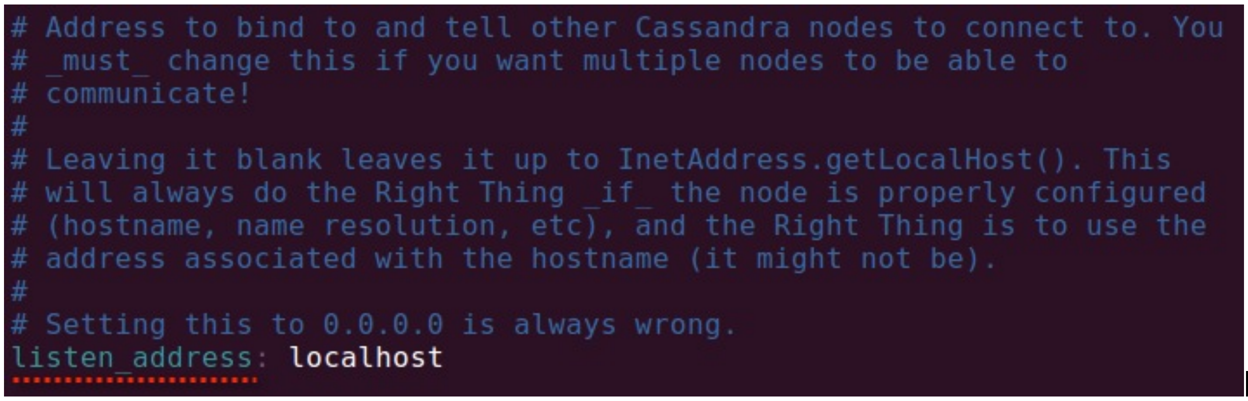
4. Find the *seeds* setting and make a note of the current value. No changes need to be made.



```
# any class that implements the SeedProvider interface and has a
# constructor that takes a Map<String, String> of parameters will do.
seed_provider:
    # Addresses of hosts that are deemed contact points.
    # Cassandra nodes use this list of hosts to find each other and learn
    # the topology of the ring.  You must change this if you are running
    # multiple nodes!
    - class_name: org.apache.cassandra.locator.SimpleSeedProvider
      parameters:
          # seeds is actually a comma-delimited list of addresses.
          # Ex: "<ip1>,<ip2>,<ip3>"
          - seeds: "127.0.0.1"
```

5. Change the *listen_address* setting to 127.0.0.1.

```
listen_address: 127.0.0.1
```



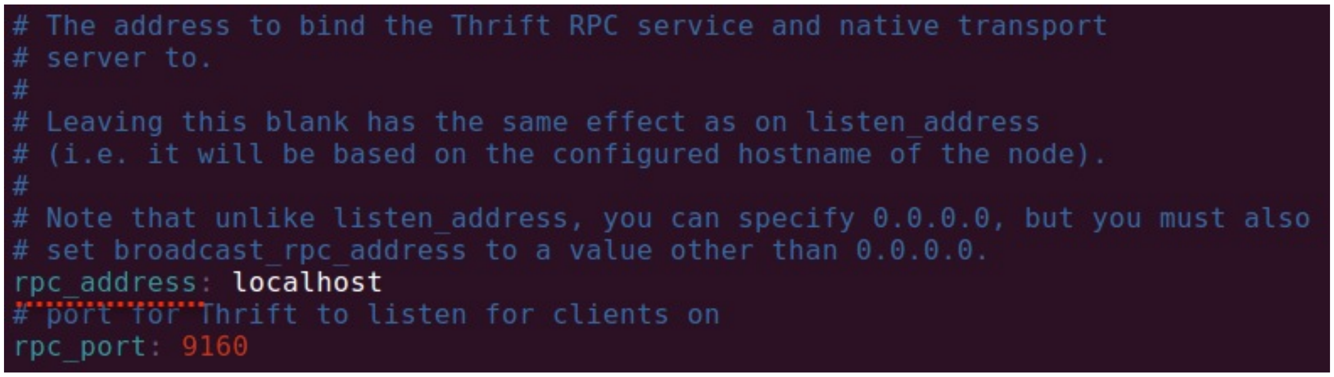
```
# Address to bind to and tell other Cassandra nodes to connect to. You
# _must_ change this if you want multiple nodes to be able to
# communicate!
#
# Leaving it blank leaves it up to InetAddress.getLocalHost(). This
# will always do the Right Thing _if_ the node is properly configured
# (hostname, name resolution, etc), and the Right Thing is to use the
# address associated with the hostname (it might not be).
#
# Setting this to 0.0.0.0 is always wrong.
listen_address: localhost
```

In

Copyright 2014 DataStax

6. Change the rpc_address setting to 127.0.0.1.

```
rpc_address: 127.0.0.1
```



```
# The address to bind the Thrift RPC service and native transport
# server to.
#
# Leaving this blank has the same effect as on listen_address
# (i.e. it will be based on the configured hostname of the node).
#
# Note that unlike listen_address, you can specify 0.0.0.0, but you must also
# set broadcast_rpc_address to a value other than 0.0.0.0.
rpc_address: localhost
# port for Thrift to listen for clients on
rpc_port: 9160
```

Steps 3-6 are the settings that may need to be edited in the cassandra.yaml, *at a minimum, to add a node into an existing cluster.*

7. Save the file and close the text editor.

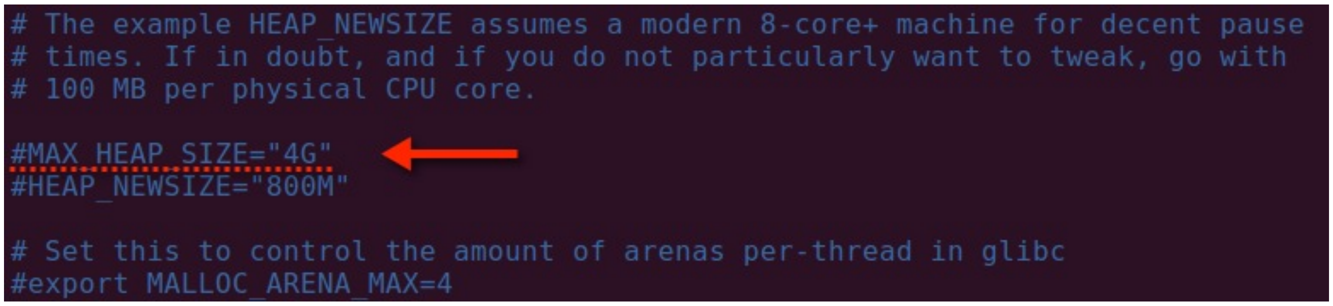
Configure the *cassandra-env.sh* file

8. In the Terminal window, open the *cassandra-env.sh* file in a text editor.

```
gedit cassandra-env.sh
```

9. In the *cassandra-env.sh* file, remove the # from *MAX_HEAP_SIZE* setting and change the value to "1024M".

```
MAX_HEAP_SIZE="1024M"
```



```
# The example HEAP_NEWSIZE assumes a modern 8-core+ machine for decent pause
# times. If in doubt, and if you do not particularly want to tweak, go with
# 100 MB per physical CPU core.

#MAX_HEAP_SIZE="4G"
#HEAP_NEWSIZE="800M"

# Set this to control the amount of arenas per-thread in glibc
#export MALLOC_ARENA_MAX=4
```

To quickly find the correct setting, run a search for the pattern #MAX_HEAP_SIZE.

Copyright 2014 DataStax

10. Remove the # from the *HEAP_NEWSIZE setting* and change the value to "100M".

```
HEAP_NEWSIZE="100M"
```

The values you are setting are the same values Cassandra will automatically use, if the virtual machine is still allocated with 4GB of RAM. Knowing how to change these settings manually will be important with later garbage collection and performance tuning.

11. Save the file and close the text editor.

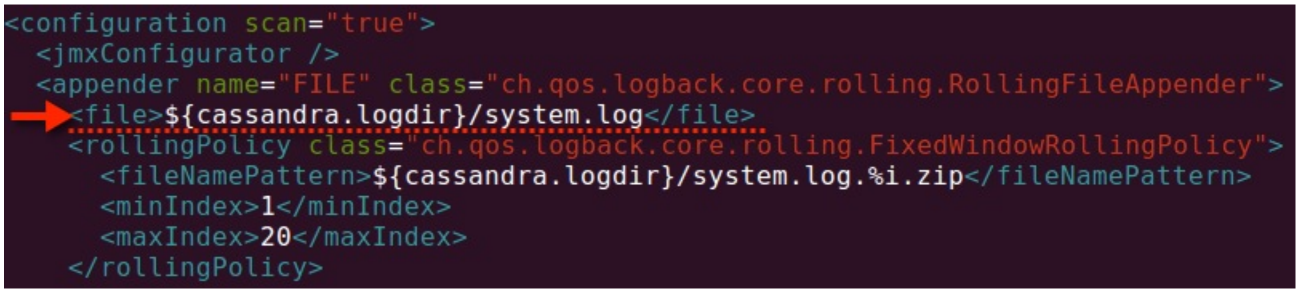
Configure the *logback.xml* file

12. In the Terminal window, use a text editor to open the *logback.xml* file.

```
gedit logback.xml
```

13. In the *logback.xml* file, change the log location to point to /home/student/cassandra/logs/system.log.

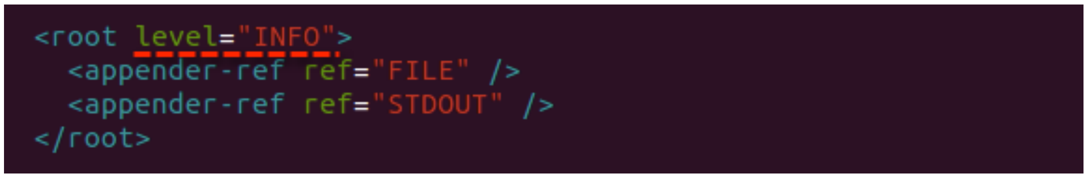
```
<file>/home/student/cassandra/logs/system.log</file>
```



```
<configuration scan="true">
  <jmxConfigurator />
  <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>${cassandra.logdir}/system.log</file>
    <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
      <fileNamePattern>${cassandra.logdir}/system.log.%i.zip</fileNamePattern>
      <minIndex>1</minIndex>
      <maxIndex>20</maxIndex>
    </rollingPolicy>
```

Copyright 2014 DataStax

14. Make a note of the current log level for the root appender. Nothing needs to be changed here.



```
<root level="INFO">
  <appender-ref ref="FILE" />
  <appender-ref ref="STDOUT" />
</root>
```

You normally would not need to change the logging level from INFO, but for learning purposes it may be useful to review the additional messages logged at the DEBUG level.

15. Save the file and close the text editor.

END OF EXERCISE

Copyright 2014 DataStax

Exercise 4: Run Cassandra and examine its logs

In this exercise, you will:

- Start up the Cassandra instance
- Stop Cassandra
- Browse the Cassandra system log

Steps

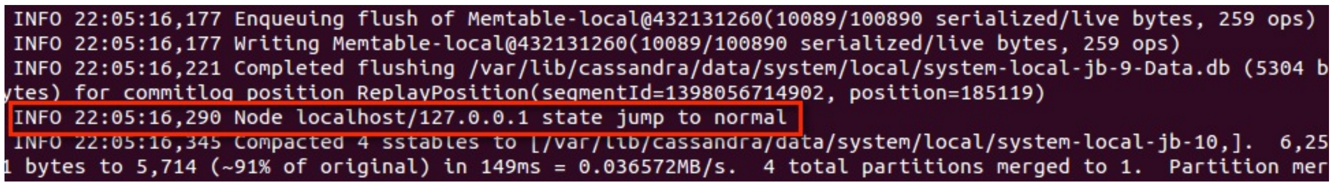
Start up the Cassandra instance

1. In your Terminal window, navigate to the Cassandra *bin* directory.

```
cd /home/student/cassandra/bin
```

2. In the *bin* directory, run the *cassandra* binary.

```
./cassandra
```



```
 INFO 22:05:16,177 Enqueuing flush of Memtable-local@432131260(10089/100890 serialized/live bytes, 259 ops)
 INFO 22:05:16,177 Writing Memtable-local@432131260(10089/100890 serialized/live bytes, 259 ops)
 INFO 22:05:16,221 Completed flushing /var/lib/cassandra/data/system/local/system-local-jb-9-Data.db (5304 b
ytes) for commitlog position ReplayPosition(segmentId=1398056714902, position=185119)
 INFO 22:05:16,290 Node localhost/127.0.0.1 state jump to normal
 INFO 22:05:16,345 Compacted 4 sstables to [/var/lib/cassandra/data/system/local/system-local-jb-10,].  6,25
1 bytes to 5,714 (~91% of original) in 149ms = 0.036572MB/s.  4 total partitions merged to 1.  Partition mer
```

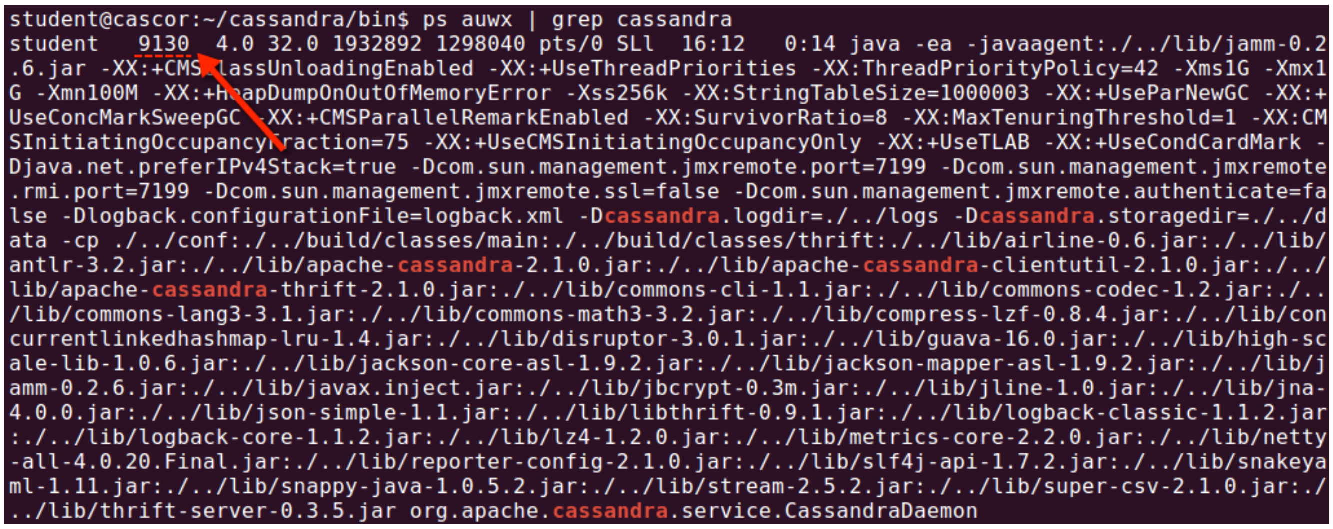
There will be a stream of text that will be displayed when Cassandra starts. You will know when Cassandra is up when you see the above message, but it can be easy to miss. After it looks like the text being displayed has paused, Cassandra should have started up and you can press <Enter> to get a prompt and continue on.

Copyright 2014 DataStax

Stop Cassandra

3. In your terminal window, display a list of processes running Cassandra.

```
ps auwx | grep cassandra
```



```
student@cascor:~/cassandra/bin$ ps auwx | grep cassandra
student   9130  4.0 32.0 1932892 1298040 pts/0 SLl  16:12   0:14 java -ea -javaagent:./../lib/jamm-0.2
.6.jar -XX:+CMSClassUnloadingEnabled -XX:+UseThreadPriorities -XX:ThreadPriorityPolicy=42 -Xms1G -Xmx1
G -Xmn100M -XX:+HeapDumpOnOutOfMemoryError -Xss256k -XX:StringTableSize=1000003 -XX:+UseParNewGC -XX:+
UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -XX:SurvivorRatio=8 -XX:MaxTenuringThreshold=1 -XX:CM
SInitiatingOccupancyFraction=75 -XX:+UseCMSInitiatingOccupancyOnly -XX:+UseTLAB -XX:+UseCondCardMark -
Djava.net.preferIPv4Stack=true -Dcom.sun.management.jmxremote.port=7199 -Dcom.sun.management.jmxremote
.rmi.port=7199 -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.authenticate=fa
lse -Dlogback.configurationFile=logback.xml -Dcassandra.logdir=./../logs -Dcassandra.storagedir=./../d
ata -cp ./../conf:./../build/classes/main:./../build/classes/thrift:./../lib/airline-0.6.jar:./../lib/
antlr-3.2.jar:./../lib/apache-cassandra-2.1.0.jar:./../lib/apache-cassandra-clientutil-2.1.0.jar:./../
lib/apache-cassandra-thrift-2.1.0.jar:./../lib/commons-cli-1.1.jar:./../lib/commons-codec-1.2.jar:./..
/lib/commons-lang3-3.1.jar:./../lib/commons-math3-3.2.jar:./../lib/compress-lzf-0.8.4.jar:./../lib/con
currentlinkedhashmap-lru-1.4.jar:./../lib/disruptor-3.0.1.jar:./../lib/guava-16.0.jar:./../lib/high-sc
ale-lib-1.0.6.jar:./../lib/jackson-core-asl-1.9.2.jar:./../lib/jackson-mapper-asl-1.9.2.jar:./../lib/j
amm-0.2.6.jar:./../lib/javax.inject.jar:./../lib/jbcrypt-0.3m.jar:./../lib/jline-1.0.jar:./../lib/jna-
4.0.0.jar:./../lib/json-simple-1.1.jar:./../lib/libthrift-0.9.1.jar:./../lib/logback-classic-1.1.2.jar
:./../lib/logback-core-1.1.2.jar:./../lib/lz4-1.2.0.jar:./../lib/metrics-core-2.2.0.jar:./../lib/netty
-all-4.0.20.Final.jar:./../lib/reporter-config-2.1.0.jar:./../lib/slf4j-api-1.7.2.jar:./../lib/snakeya
ml-1.11.jar:./../lib/snappy-java-1.0.5.2.jar:./../lib/stream-2.5.2.jar:./../lib/super-csv-2.1.0.jar:./
../lib/thrift-server-0.3.5.jar org.apache.cassandra.service.CassandraDaemon
```

The information to retrieve here is the Process ID (PID) for the Cassandra Java process, which is displayed on the second column. In this example, the PID is 9130.

Another great way to find the Cassandra process is to run the command jps *and look for the process ID for CassandraDaemon.*

4. Use the *kill* command with the PID you discovered, to shut down Cassandra.

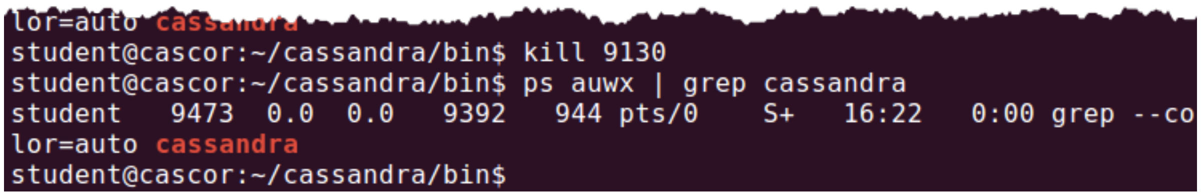
```
kill [PID]
```

The kill *command sends a signal to the Cassandra process to shut down using its normal shut down process. Note that this is different from a* kill -9*, which would force Cassandra to immediately stop without going through its normal shut down process.*

Copyright 2014 DataStax

5. Verify that Cassandra has shut down by running the *ps* command again.

```
ps auwx | grep cassandra
```



```
lor=auto cassandra
student@cascor:~/cassandra/bin$ kill 9130
student@cascor:~/cassandra/bin$ ps auwx | grep cassandra
student   9473  0.0  0.0   9392   944 pts/0    S+   16:22   0:00 grep --co
lor=auto cassandra
student@cascor:~/cassandra/bin$
```

The ps *command may still show other processes with the word "*cassandra*", but the actual* cassandra *process is distinctive with its large body of text. If you do not see that, then* cassandra *should have stopped.*

View the Cassandra system log

6. In your terminal window, navigate to the system log directory.

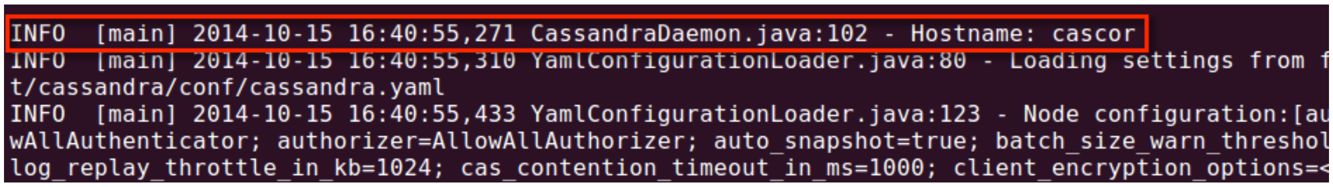
```
cd /home/student/cassandra/logs
```

7. Use a text editor to open the *system.log* file.

```
gedit system.log
```

You will find that the contents of the system log are the same as the output displayed when starting Cassandra.

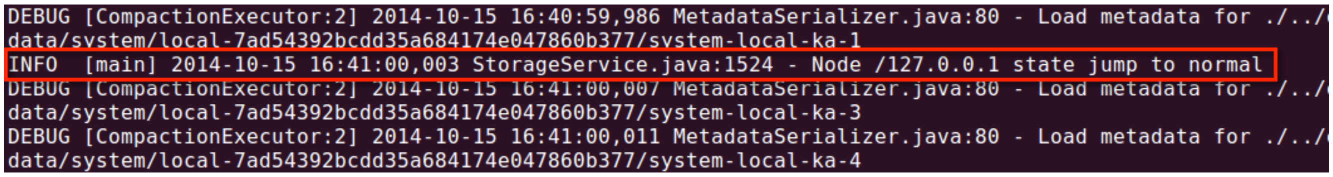
8. In the *system.log* file, find the entry that indicates when Cassandra was started.



```
INFO  [main] 2014-10-15 16:40:55,271 CassandraDaemon.java:102 - Hostname: cascor
INFO  [main] 2014-10-15 16:40:55,310 YamlConfigurationLoader.java:80 - Loading settings from f
t/cassandra/conf/cassandra.yaml
INFO  [main] 2014-10-15 16:40:55,433 YamlConfigurationLoader.java:123 - Node configuration:[au
wAllAuthenticator; authorizer=AllowAllAuthorizer; auto_snapshot=true; batch_size_warn_threshol
log_replay_throttle_in_kb=1024; cas_contention_timeout_in_ms=1000; client_encryption_options=<
```

Copyright 2014 DataStax

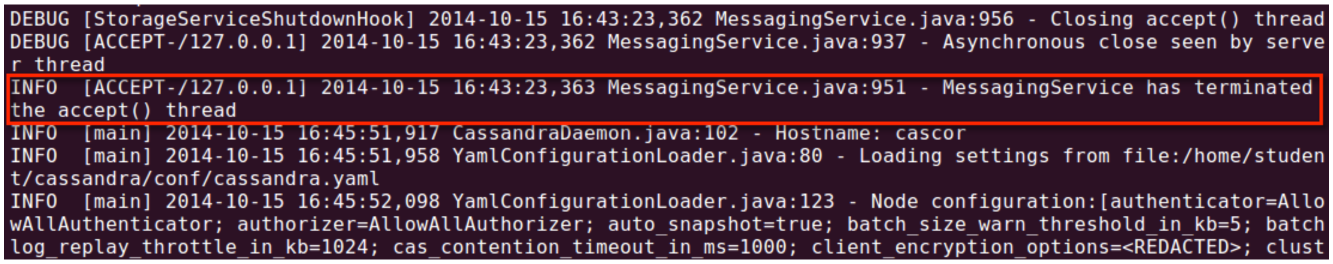
9. Find when the Cassandra node came up.



```
DEBUG [CompactionExecutor:2] 2014-10-15 16:40:59,986 MetadataSerializer.java:80 - Load metadata for ./../
data/system/local-7ad54392bcdd35a684174e047860b377/system-local-ka-1
INFO  [main] 2014-10-15 16:41:00,003 StorageService.java:1524 - Node /127.0.0.1 state jump to normal
DEBUG [CompactionExecutor:2] 2014-10-15 16:41:00,007 MetadataSerializer.java:80 - Load metadata for ./../
data/system/local-7ad54392bcdd35a684174e047860b377/system-local-ka-3
DEBUG [CompactionExecutor:2] 2014-10-15 16:41:00,011 MetadataSerializer.java:80 - Load metadata for ./../
data/system/local-7ad54392bcdd35a684174e047860b377/system-local-ka-4
```

Once an instance is up, it may still take a while before it is ready to accept connections from clients. Look for messages after the node has started to see if the instance is listening for client connections. After starting Cassandra you may want to wait at least 30 seconds before attempting to connect to the cluster.

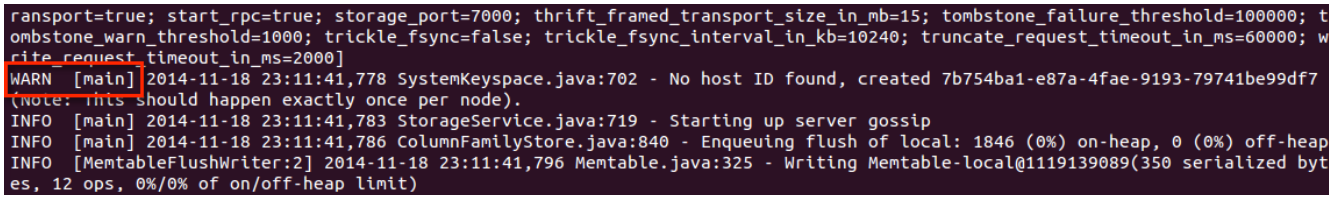
10. From the log, determine when Cassandra was shut down.



```
DEBUG [StorageServiceShutdownHook] 2014-10-15 16:43:23,362 MessagingService.java:956 - Closing accept() thread
DEBUG [ACCEPT-/127.0.0.1] 2014-10-15 16:43:23,362 MessagingService.java:937 - Asynchronous close seen by serve
r thread
INFO  [ACCEPT-/127.0.0.1] 2014-10-15 16:43:23,363 MessagingService.java:951 - MessagingService has terminated
the accept() thread
INFO  [main] 2014-10-15 16:45:51,917 CassandraDaemon.java:102 - Hostname: cascor
INFO  [main] 2014-10-15 16:45:51,958 YamlConfigurationLoader.java:80 - Loading settings from file:/home/studen
t/cassandra/conf/cassandra.yaml
INFO  [main] 2014-10-15 16:45:52,098 YamlConfigurationLoader.java:123 - Node configuration:[authenticator=Allo
wAllAuthenticator; authorizer=AllowAllAuthorizer; auto_snapshot=true; batch_size_warn_threshold_in_kb=5; batch
log_replay_throttle_in_kb=1024; cas_contention_timeout_in_ms=1000; client_encryption_options=<REDACTED>; clust
```

11. Search for messages that have a logging level of WARN.

```
WARN [main] ...
ERROR [main] ...
```



```
ransport=true; start_rpc=true; storage_port=7000; thrift_framed_transport_size_in_mb=15; tombstone_failure_threshold=100000; t
ombstone_warn_threshold=1000; trickle_fsync=false; trickle_fsync_interval_in_kb=10240; truncate_request_timeout_in_ms=60000; w
rite_request_timeout_in_ms=2000]
WARN  [main] 2014-11-18 23:11:41,778 SystemKeyspace.java:702 - No host ID found, created 7b754ba1-e87a-4fae-9193-79741be99df7
(Note: This should happen exactly once per node).
INFO  [main] 2014-11-18 23:11:41,783 StorageService.java:719 - Starting up server gossip
INFO  [main] 2014-11-18 23:11:41,786 ColumnFamilyStore.java:840 - Enqueuing flush of local: 1846 (0%) on-heap, 0 (0%) off-heap
INFO  [MemtableFlushWriter:2] 2014-11-18 23:11:41,796 Memtable.java:325 - Writing Memtable-local@1119139089(350 serialized byt
es, 12 ops, 0%/0% of on/off-heap limit)
```

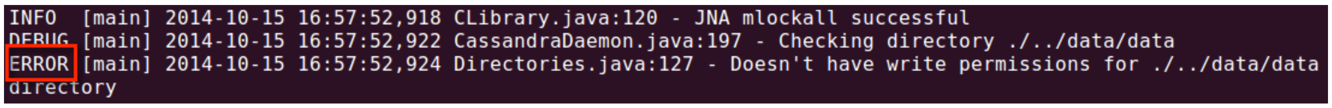
The system log should consist of mainly INFO messages. Potential problems saved in the system log would be categorized as WARN, ERROR, or would display an exception.

If there were no problems with configuring, starting, and stopping Cassandra, you may only find one WARN message about a host ID not being found. At this point in the log, the installation has started for the first time, and no host ID for this instance has been created yet.

Copyright 2014 DataStax

12. Search for messages that have a logging level of ERROR.

```
ERROR [main] ...
```



```
INFO  [main] 2014-10-15 16:57:52,918 CLibrary.java:120 - JNA mlockall successful
DEBUG [main] 2014-10-15 16:57:52,922 CassandraDaemon.java:197 - Checking directory ./../data/data
ERROR [main] 2014-10-15 16:57:52,924 Directories.java:127 - Doesn't have write permissions for ./../data/data
directory
```

If there were no problems with configuring, starting, and stopping Cassandra, you may only find a few ERROR messages indicating that the data directories, commitlog directory, and saved_cache directory doesn't exist. This is only an error if the directories and files files should already exist if the Cassandra instance has been run previously or if files are being transferred from another instance. Otherwise Cassandra will create these directories automatically, if it has write permission to do so.

END OF EXERCISE

Copyright 2014 DataStax

Additional Challenges (optional)

Configure and start a two node Cassandra cluster

- Make another copy of the /home/student/cassandra directory to use as the second node.

It will probably be easier to uncompress the binary tarball again to use as the second instance. Otherwise, if you are making a copy of the existing Cassandra instance, make sure to delete the data/data and data/commitlog directory from the copy.

- On the second node, configure the cassandra.yaml file and edit the appropriate settings using an IP address of 127.0.0.2 for the second node.

This should include the settings for seeds, listen_address, and rpc_address.

- On the second node, make a change to the cassandra-env.sh file to use a JMX_PORT of 7200.

- Start up the first node, if is not already online, and then the second node. If configured correctly, the second node should have started up without any errors.

- Use the command */home/student/cassandra/bin/nodetool status* to verify that two nodes are in the cluster.

Copyright 2014 DataStax

Appendix A: The Linux Terminal

The Linux Filesystem

The Linux filesystem can be complex, even for those that have experience with other command lines. Some important points to know include:

- The name of files and directories in Linux are all case-sensitive. Using the wrong capitalization will cause the command line to not be able to find that file or directory.
- A forward slash (/) is used to denote directory paths.
- The * character is used as a wildcard, and can represent all the possible combinations as part of a file name.
- The ~ character, when used as part of a path, represents the user's home directory.

Interacting with the Linux Filesystem

To navigate to different directories in the Linux filesystem, the primary command to use is *cd, which is short for* "change directory".

Description	Syntax	Example
To navigate to an absolute path	cd <absolute path>	cd /usr/bin
To go to the home directory	cd ~, or cd /home/<user>	cd ~
To go into a sub-directory	cd <directory name>	cd Downloads
To go into multiple sub-directories	cd <name>/<name>	cd cass/tools/bin
To go up one directory	cd ..	cd ..
To go up two directories	cd ../..	cd ../..

List files and sub-directories in the current directory	ls
List files and sub-directories with additional details	ls -l
List files and sub-directories starting with the letter a	ls a*
Display the path of the current directory	pwd

Copy a file	cp <source path> <target path>	cp cassandra.yaml cassandra.yaml.bak
Move a file	mv <source path> <target path>	mv original_file renamed_file

Copyright 2014 DataStax

Executing a File

Running a file in the Linux command line may not always work the way you expect it to. In general, there are three ways a file can be run:

Global Path – Typing in the name of a file may cause the file to run, if the command line can locate it. The command line will check certain locations it knows about in a PATH variable, and run the file from the first location that has a file with that name. Examples of files that can be run this way include *cd*, *ls*, and *ccm*.

Absolute path – It is possible that the command line finds and executes a file in a different location than what you wanted. To avoid this, you can specify the entire path where the file can be found, such as */home/student/cassandra/bin/cqlsh.*

Current directory – To run a file in the current directory, a ./ needs to be added in front of the file name so that the command line knows to find it in the current directory. One example of running a file this way would be *./nodetool.*

Copyright 2014 DataStax