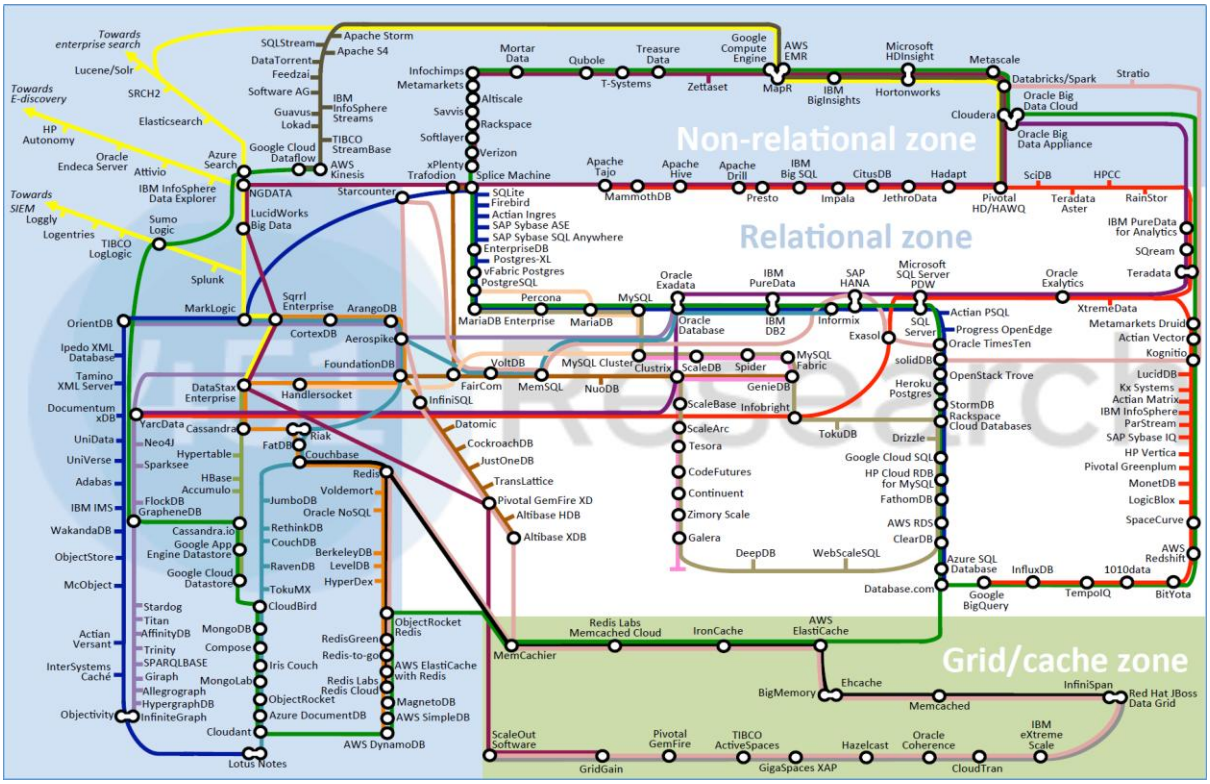
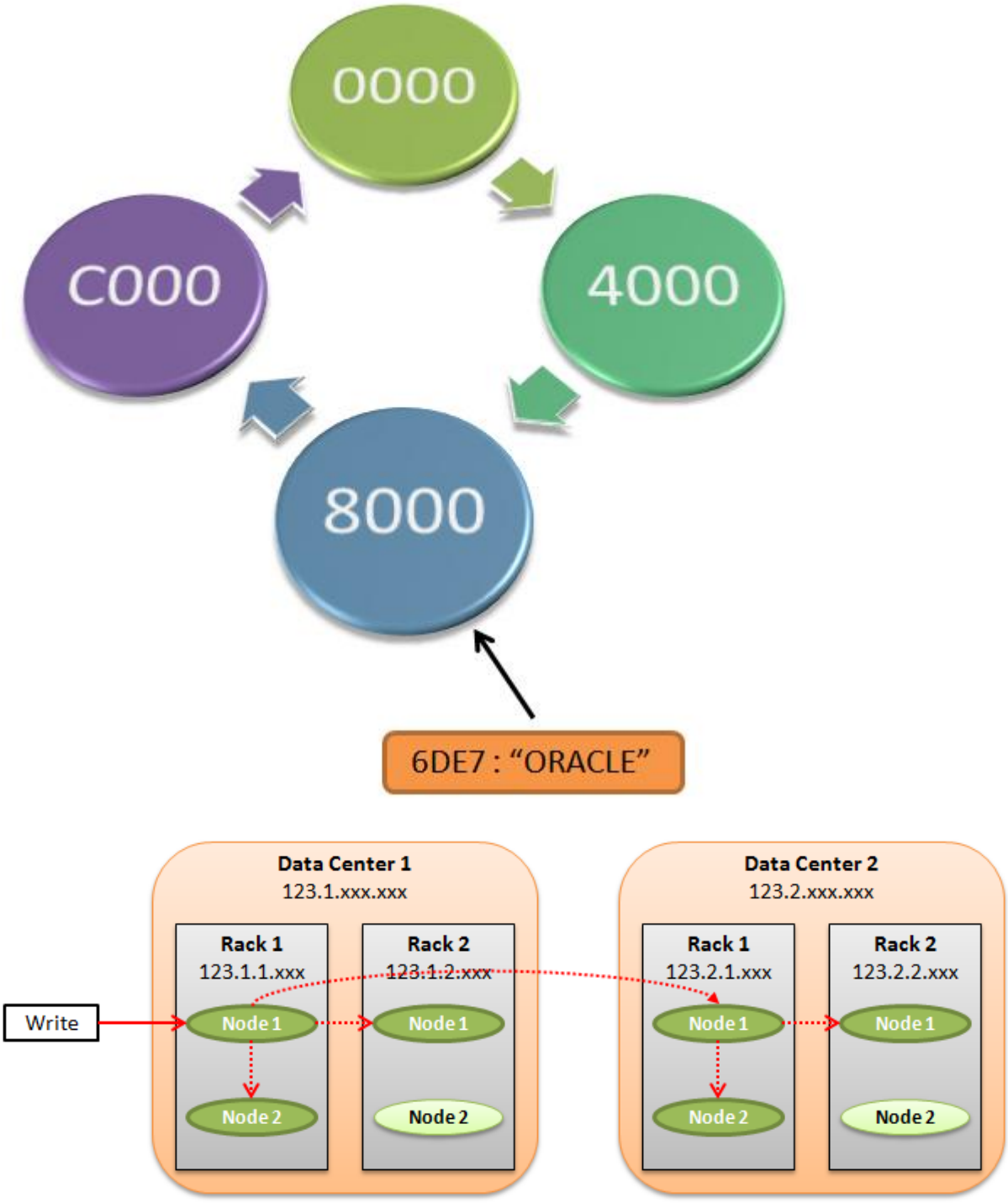
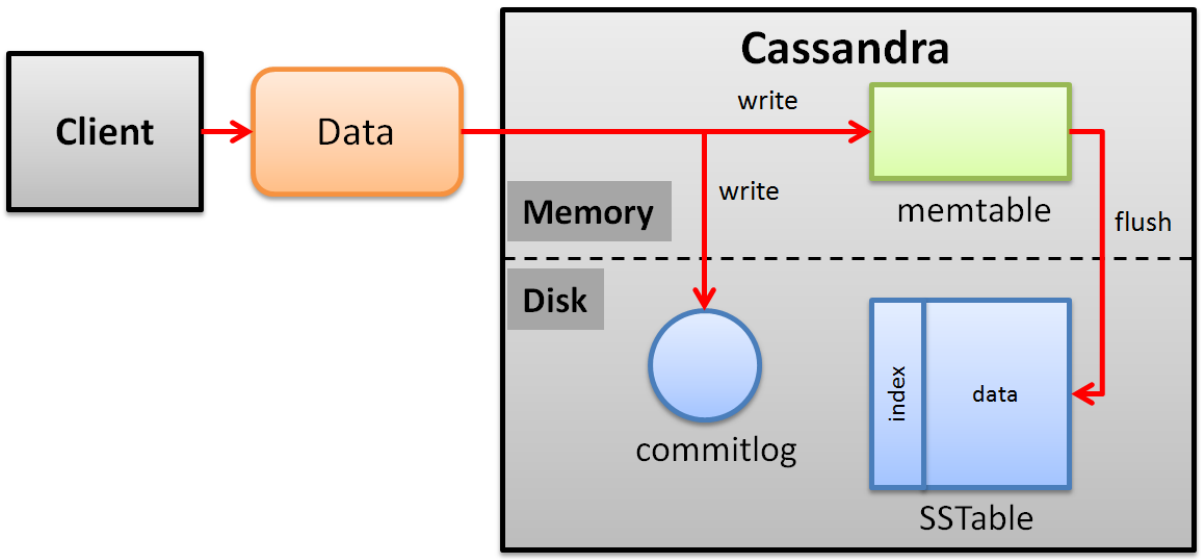
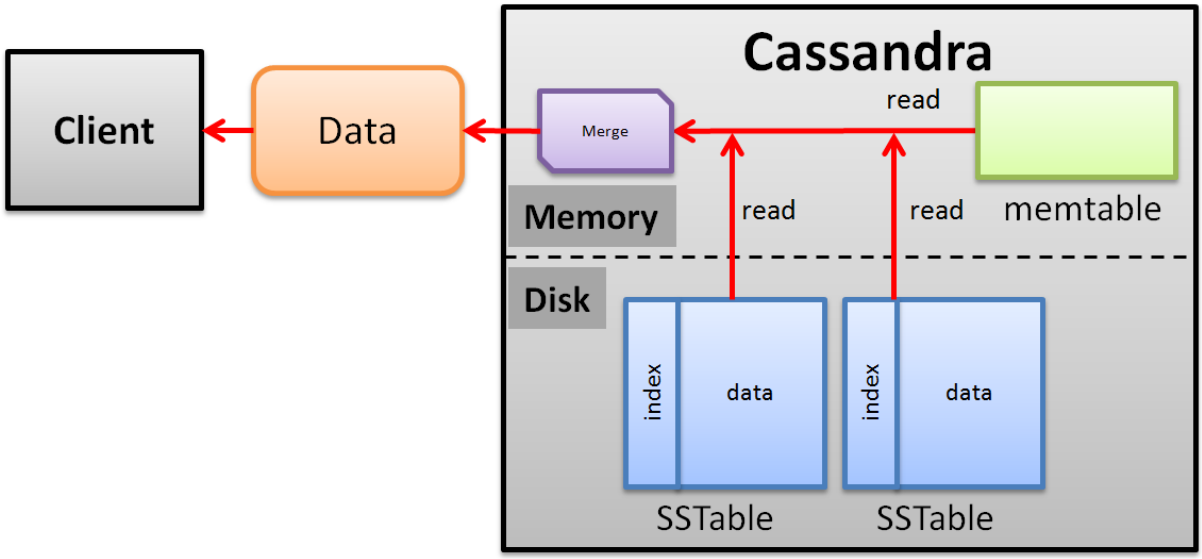
Chapter 1, Bird's Eye View of Cassandra

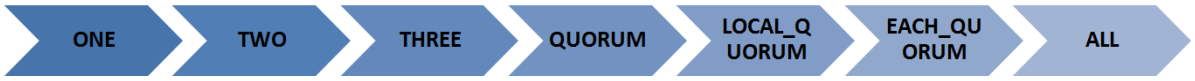


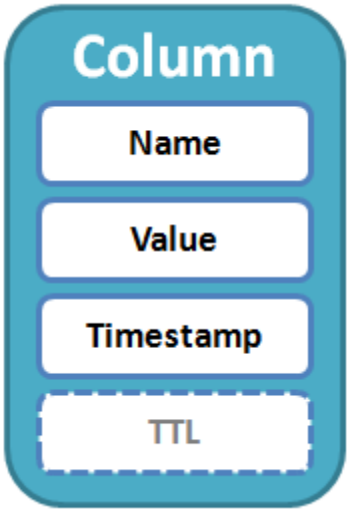

0000
4000
8000
C000
6DE7 : “ORACLE”
Data Center 1
123.1.xxx.xxx
Rack 1
123.1.1.xxx
Rack 2
123.1.2.xxx
Write
Node 1
Node 2
Node 1
Node 2
Data Center 2
123.2.xxx.xxx
Rack 1
123.2.1.xxx
Rack 2
123.2.2.xxx
Node 1
Node 2
Node 1
Node 2

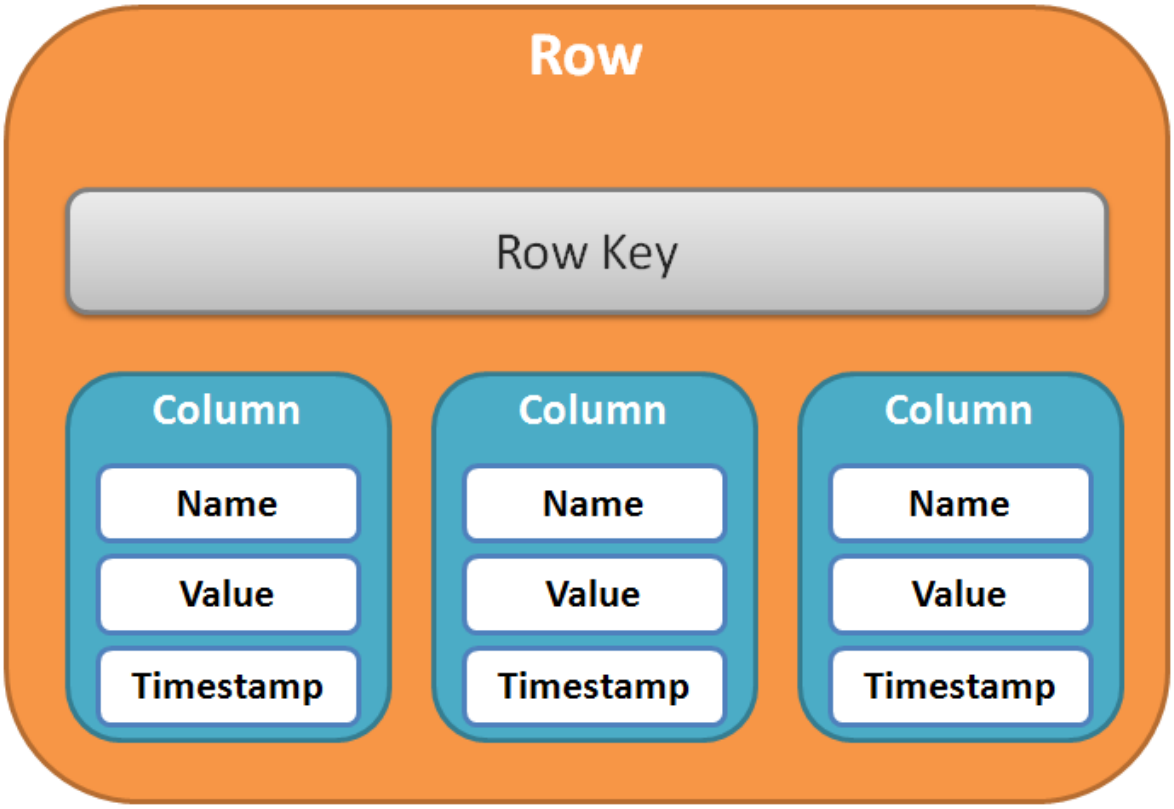

Client
Data
Cassandra
write
write
Memory
memtable
flush
Disk
commitlog
index
data
SSTable

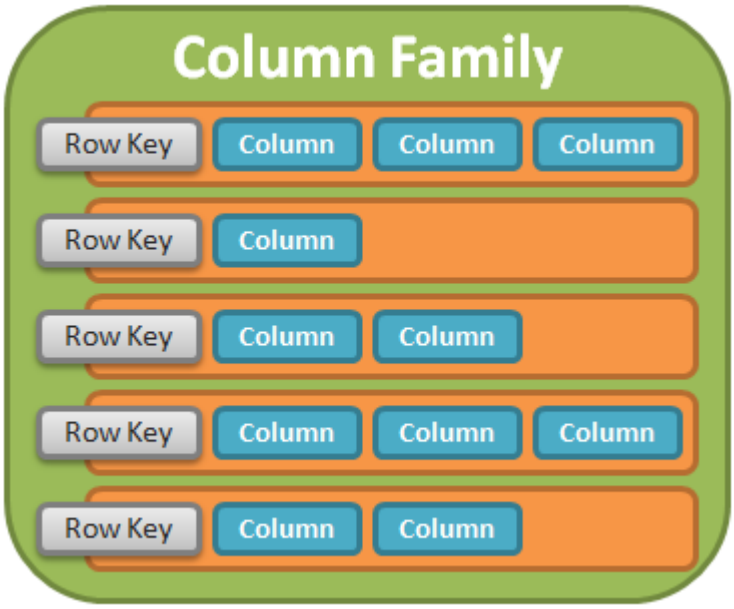

ANY
ONE
TWO
THREE
QUORUM
LOCAL_QUORUM
EACH_QUORUM
ALL

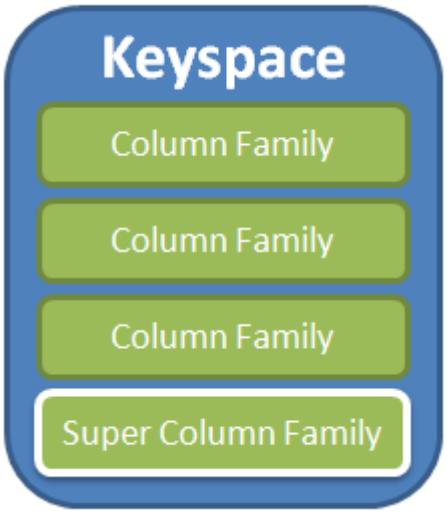

Client
Data
Cassandra
Merge
read
Memory
read
read
memtable
Disk
index
data
index
data
SSTable
SSTable

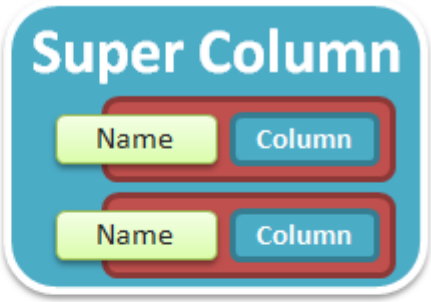

ONE
TWO
THREE
QUORUM
LOCAL_QUORUM
EACH_QUORUM
ALL

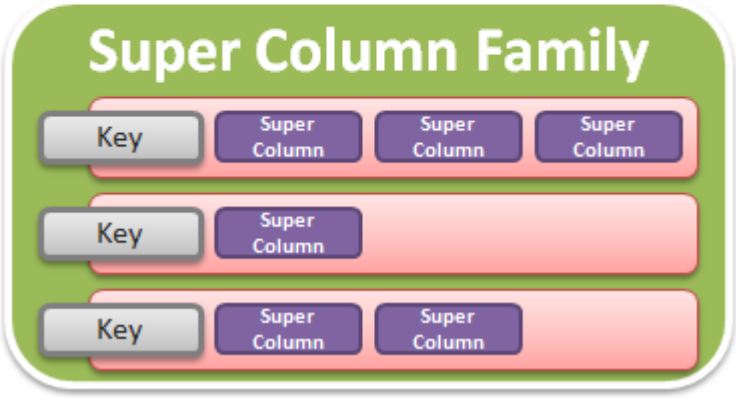

Column
Name
Value
Timestamp
TTL


Row
Row Key
Column
Name
Value
Timestamp
Column
Name
Value
Timestamp
Column
Name
Value
Timestamp


Column Family
Row Key
Column
Column
Column
Row Key
Column
Row Key
Column
Column
Row Key
Column
Column
Column
Row Key
Column
Column


Keyspace
Column Family
Column Family
Column Family
Super Column Family





stock_symbol

symbol	description	exchange
AAPL	Apple Inc.	NASDAQ
FB	Facebook, Inc.	NASDAQ

stock_ticker

symbol	tick_date	open	high	low	close	volume
AAPL	2014-04-24	568.21	570.00	560.73	567.77	27092600
FB	2014-04-24	63.60	63.65	59.77	60.87	138520000
AAPL	2014-04-25	564.53	571.99	563.96	571.94	13922800
FB	2014-04-25	59.97	60.01	57.57	57.71	92288700

stock_symbol		
RowKey	description	exchange
AAPL	Apple Inc.	NASDAQ
FB	Facebook, Inc.	NASDAQ

RowKey: AAPL
=> (name=, value=, timestamp=...)
=> (name=description, value=4170706c6520496e632e, timestamp=...)
=> (name=exchange, value=4e4153444151, timestamp=...)

RowKey: FB
=> (name=, value=, timestamp=...)
=> (name=description, value=46616365626f6f6b2c20496e632e, timestamp=...)
=> (name=exchange, value=4e4153444151, timestamp=...)

stock_ticker_by_exchange						
RowKey	AAPL:2014-04-24:	AAPL:2014-04-24:close	AAPL:2014-04-24:description	AAPL:2014-04-25:	AAPL:2014-04-25:close	AAPL:2014-04-25:description
NASDAQ	0	568.21	Apple Inc.	0	571.94	Apple Inc.
	FB:2014-04-24:	FB:2014-04-24:close	FB:2014-04-24:description	FB:2014-04-25:	FB:2014-04-25:close	FB:2014-04-25:description
	0	60.87	Facebook, Inc.	0	57.71	Facebook, Inc.

RowKey: NASDAQ
=> (name=AAPL:2014-04-24:, value=, timestamp=...)
=> (name=AAPL:2014-04-24:close, value=0000000200ddc9, timestamp=...)
=> (name=AAPL:2014-04-24:description, value=4170706c6520496e632e, timestamp=...)
=> (name=AAPL:2014-04-25:, value=, timestamp=...)
=> (name=AAPL:2014-04-25:close, value=0000000200df6a, timestamp=...)
=> (name=AAPL:2014-04-25:description, value=4170706c6520496e632e, timestamp=...)
=> (name=FB:2014-04-24:, value=, timestamp=...)
=> (name=FB:2014-04-24:close, value=0000000217c7, timestamp=...)
=> (name=FB:2014-04-24:description, value=46616365626f6f6b2c20496e632e,
timestamp=...)
=> (name=FB:2014-04-25:, value=, timestamp=...)
=> (name=FB:2014-04-25:close, value=00000002168b, timestamp=...)
=> (name=FB:2014-04-25:description, value=46616365626f6f6b2c20496e632e,
timestamp=...)

stock_ticker_by_exchange_date						
RowKey	AAPL:	AAPL:close	AAPL:description	FB:	FB:close	FB:description
NASDAQ:2014-04-24	0	567.77	Apple Inc.	0	60.87	Facebook, Inc.

RowKey: NASDAQ:2014-04-24
=> (name=AAPL:, value=, timestamp=...)
=> (name=AAPL:close, value=0000000200ddc9, timestamp=...)
=> (name=AAPL:description, value=4170706c6520496e632e, timestamp=...)
=> (name=FB:, value=, timestamp=...)
=> (name=FB:close, value=0000000217c7, timestamp=...)
=> (name=FB:description, value=46616365626f6f6b2c20496e632e, timestamp=...)

Chapter 3, CQL Data Types

```
kan@ubuntu: ~
kan@ubuntu:~$ cqlsh
Connected to Test Cluster at localhost:9160.
[cqlsh 4.1.1 | Cassandra 2.0.9 | CQL spec 3.1.1 | Thrift protocol 19.39.0]
Use HELP for help.
cqlsh>
```

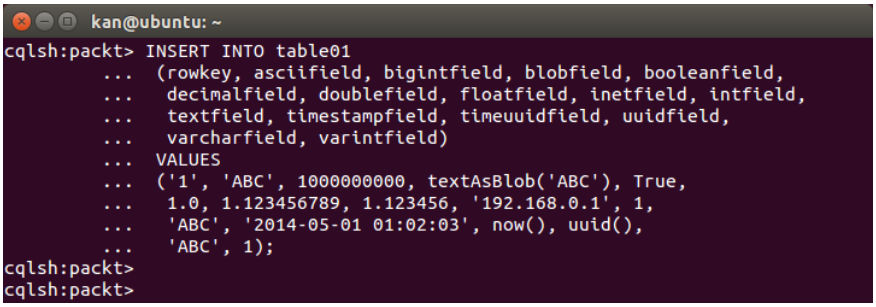
```
kan@ubuntu: ~
kan@ubuntu:~$ cqlsh
Connected to Test Cluster at localhost:9160.
[cqlsh 4.1.1 | Cassandra 2.0.9 | CQL spec 3.1.1 | Thrift protocol 19.39.0]
Use HELP for help.
cqlsh> CREATE KEYSPACE packt
   ...   WITH REPLICATION=
   ...   {'class':'SimpleStrategy', 'replication_factor':1};
cqlsh>
```

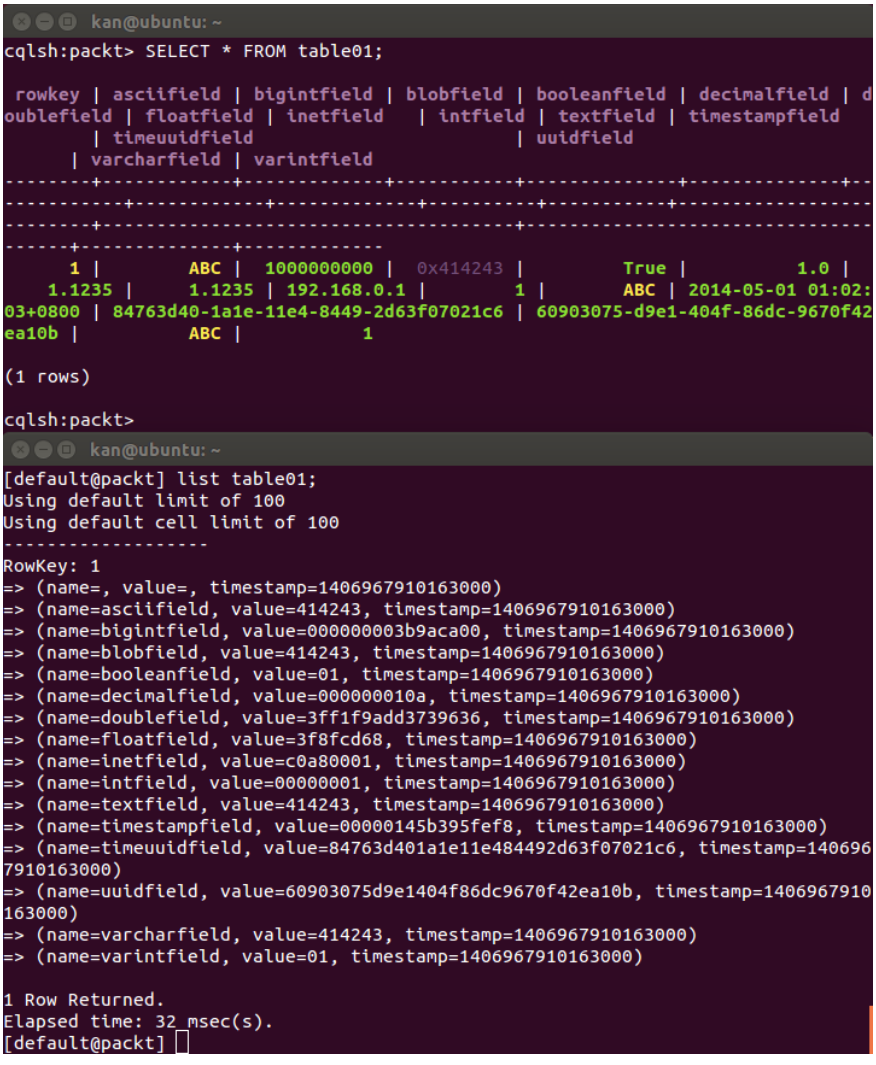
cassandra/CQL3Type

GitHub, Inc. [US] https://github.com/apache/cassandra/blob/trunk/src/java

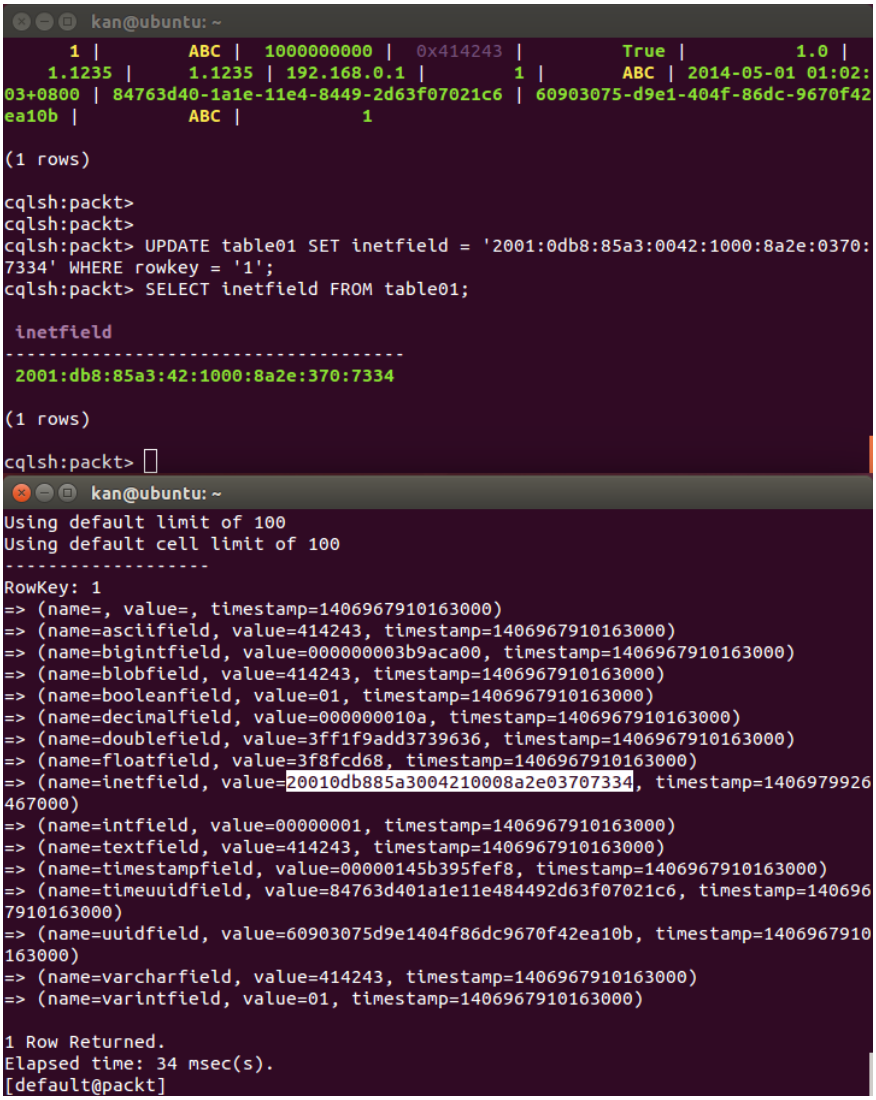
```
30  public interface CQL3Type
31  {
32      public boolean isCollection();
33      public AbstractType<?> getType();
34
35      public enum Native implements CQL3Type
36      {
37          ASCII     (AsciiType.instance),
38          BIGINT    (LongType.instance),
39          BLOB      (BytesType.instance),
40          BOOLEAN   (BooleanType.instance),
41          COUNTER   (CounterColumnType.instance),
42          DECIMAL   (DecimalType.instance),
43          DOUBLE    (DoubleType.instance),
44          FLOAT     (FloatType.instance),
45          INET      (InetAddressType.instance),
46          INT       (Int32Type.instance),
47          TEXT      (UTF8Type.instance),
48          TIMESTAMP(TimestampType.instance),
49          UUID      (UUIDType.instance),
50          VARCHAR   (UTF8Type.instance),
51          VARINT    (IntegerType.instance),
52          TIMEUUID (TimeUUIDType.instance);
53
```

```
kan@ubuntu: ~
kan@ubuntu:~$ cqlsh
Connected to Test Cluster at localhost:9160.
[cqlsh 4.1.1 | Cassandra 2.0.9 | CQL spec 3.1.1 | Thrift protocol 19.39.0]
Use HELP for help.
cqlsh> USE packt;
cqlsh:packt>
cqlsh:packt> CREATE TABLE table01 (
         ...  rowkey ascii,
         ...  asciifield ascii,
         ...  bigintfield bigint,
         ...  blobfield blob,
         ...  booleanfield boolean,
         ...  decimalfield decimal,
         ...  doublefield double,
         ...  floatfield float,
         ...  inetfield inet,
         ...  intfield int,
         ...  textfield text,
         ...  timestampfield timestamp,
         ...  timeuuidfield timeuuid,
         ...  uuidfield uuid,
         ...  varcharfield varchar,
         ...  varintfield varint,
         ...  PRIMARY KEY (rowkey)
         ... );
cqlsh:packt>
```


kan@ubuntu: ~
cqlsh:packt> INSERT INTO table01
... (rowkey, asciifield, bigintfield, blobfield, booleanfield,
... decimalfield, doublefield, floatfield, inetfield, intfield,
... textfield, timestampfield, timeuuidfield, uuidfield,
... varcharfield, varintfield)
... VALUES
... ('1', 'ABC', 1000000000, textAsBlob('ABC'), True,
... 1.0, 1.123456789, 1.123456, '192.168.0.1', 1,
... 'ABC', '2014-05-01 01:02:03', now(), uuid(),
... 'ABC', 1);
cqlsh:packt>
cqlsh:packt>


kan@ubuntu: ~
cqlsh:packt> SELECT * FROM table01;
rowkey | asciifield | bigintfield | blobfield | booleanfield | decimalfield | doublefield | floatfield | inetfield | intfield | textfield | timestampfield | timeuuidfield | uuidfield | varcharfield | varintfield
1 | ABC | 1000000000 | 0x414243 | True | 1.0 | 1.1235 | 1.1235 | 192.168.0.1 | 1 | ABC | 2014-05-01 01:02:03+0800 | 84763d40-1a1e-11e4-8449-2d63f07021c6 | 60903075-d9e1-404f-86dc-9670f42ea10b | ABC | 1
(1 rows)
cqlsh:packt>

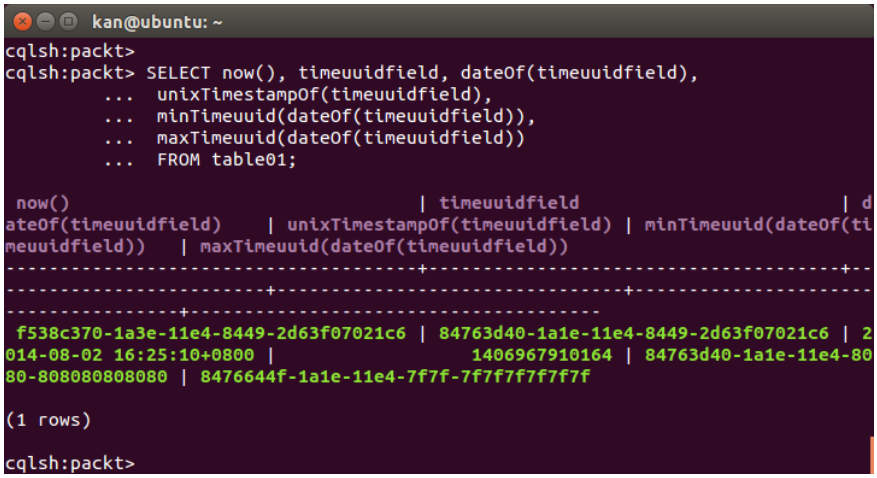
kan@ubuntu: ~
[default@packt] list table01;
Using default limit of 100
Using default cell limit of 100
RowKey: 1
=> (name=, value=, timestamp=1406967910163000)
=> (name=asciifield, value=414243, timestamp=1406967910163000)
=> (name=bigintfield, value=000000003b9aca00, timestamp=1406967910163000)
=> (name=blobfield, value=414243, timestamp=1406967910163000)
=> (name=booleanfield, value=01, timestamp=1406967910163000)
=> (name=decimalfield, value=000000010a, timestamp=1406967910163000)
=> (name=doublefield, value=3ff1f9add3739636, timestamp=1406967910163000)
=> (name=floatfield, value=3f8fcd68, timestamp=1406967910163000)
=> (name=inetfield, value=c0a80001, timestamp=1406967910163000)
=> (name=intfield, value=00000001, timestamp=1406967910163000)
=> (name=textfield, value=414243, timestamp=1406967910163000)
=> (name=timestampfield, value=00000145b395fef8, timestamp=1406967910163000)
=> (name=timeuuidfield, value=84763d401a1e11e484492d63f07021c6, timestamp=1406967910163000)
=> (name=uuidfield, value=60903075d9e1404f86dc9670f42ea10b, timestamp=1406967910163000)
=> (name=varcharfield, value=414243, timestamp=1406967910163000)
=> (name=varintfield, value=01, timestamp=1406967910163000)
1 Row Returned.
Elapsed time: 32 msec(s).
[default@packt]

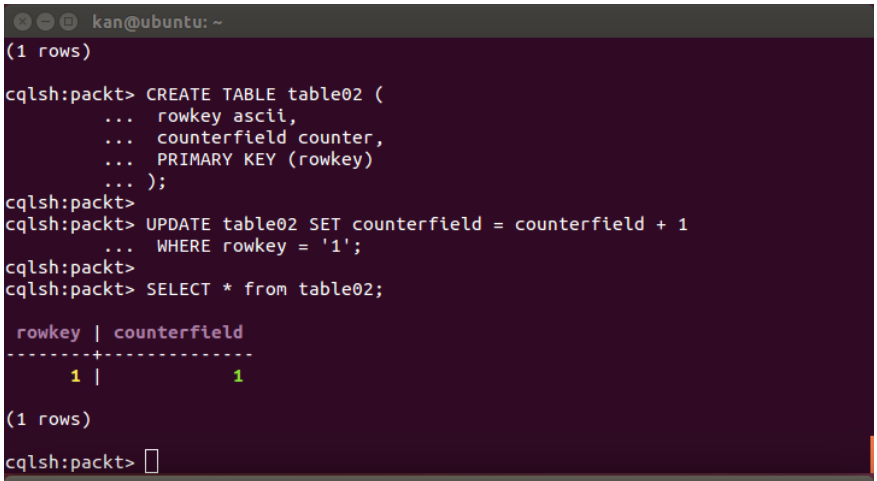

kan@ubuntu: ~
1 | ABC | 1000000000 | 0x414243 | True | 1.0 |
1.1235 | 1.1235 | 192.168.0.1 | 1 | ABC | 2014-05-01 01:02:
03+0800 | 84763d40-1a1e-11e4-8449-2d63f07021c6 | 60903075-d9e1-404f-86dc-9670f42
ea10b | ABC | 1
(1 rows)
cqlsh:packt>
cqlsh:packt>
cqlsh:packt> UPDATE table01 SET inetfield = '2001:0db8:85a3:0042:1000:8a2e:0370:
7334' WHERE rowkey = '1';
cqlsh:packt> SELECT inetfield FROM table01;
inetfield
2001:db8:85a3:42:1000:8a2e:370:7334
(1 rows)
cqlsh:packt>
kan@ubuntu: ~
Using default limit of 100
Using default cell limit of 100
RowKey: 1
=> (name=, value=, timestamp=1406967910163000)
=> (name=asciifield, value=414243, timestamp=1406967910163000)
=> (name=bigintfield, value=000000003b9aca00, timestamp=1406967910163000)
=> (name=blobfield, value=414243, timestamp=1406967910163000)
=> (name=booleanfield, value=01, timestamp=1406967910163000)
=> (name=decimalfield, value=000000010a, timestamp=1406967910163000)
=> (name=doublefield, value=3ff1f9add3739636, timestamp=1406967910163000)
=> (name=floatfield, value=3f8fcd68, timestamp=1406967910163000)
=> (name=inetfield, value=20010db885a3004210008a2e03707334, timestamp=1406979926
467000)
=> (name=intfield, value=00000001, timestamp=1406967910163000)
=> (name=textfield, value=414243, timestamp=1406967910163000)
=> (name=timestampfield, value=00000145b395fef8, timestamp=1406967910163000)
=> (name=timeuuidfield, value=84763d401a1e11e484492d63f07021c6, timestamp=140696
7910163000)
=> (name=uuidfield, value=60903075d9e1404f86dc9670f42ea10b, timestamp=1406967910
163000)
=> (name=varcharfield, value=414243, timestamp=1406967910163000)
=> (name=varintfield, value=01, timestamp=1406967910163000)
1 Row Returned.
Elapsed time: 34 msec(s).
[default@packt]

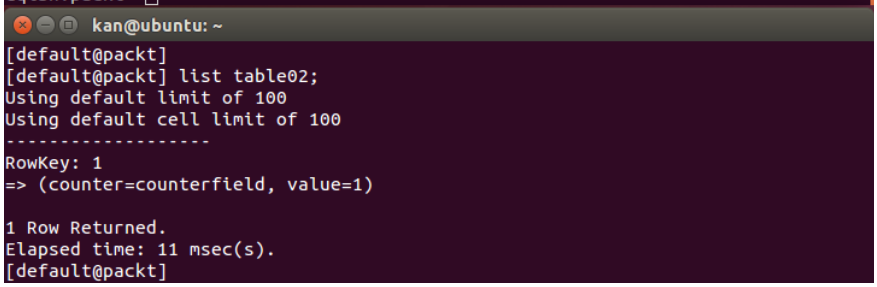
```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> UPDATE table01 SET textfield='资源提供ABC'
         ...  WHERE rowkey='1';
cqlsh:packt>
cqlsh:packt> SELECT textfield FROM table01;

 textfield
------------
 资源提供ABC

(1 rows)

cqlsh:packt>
```

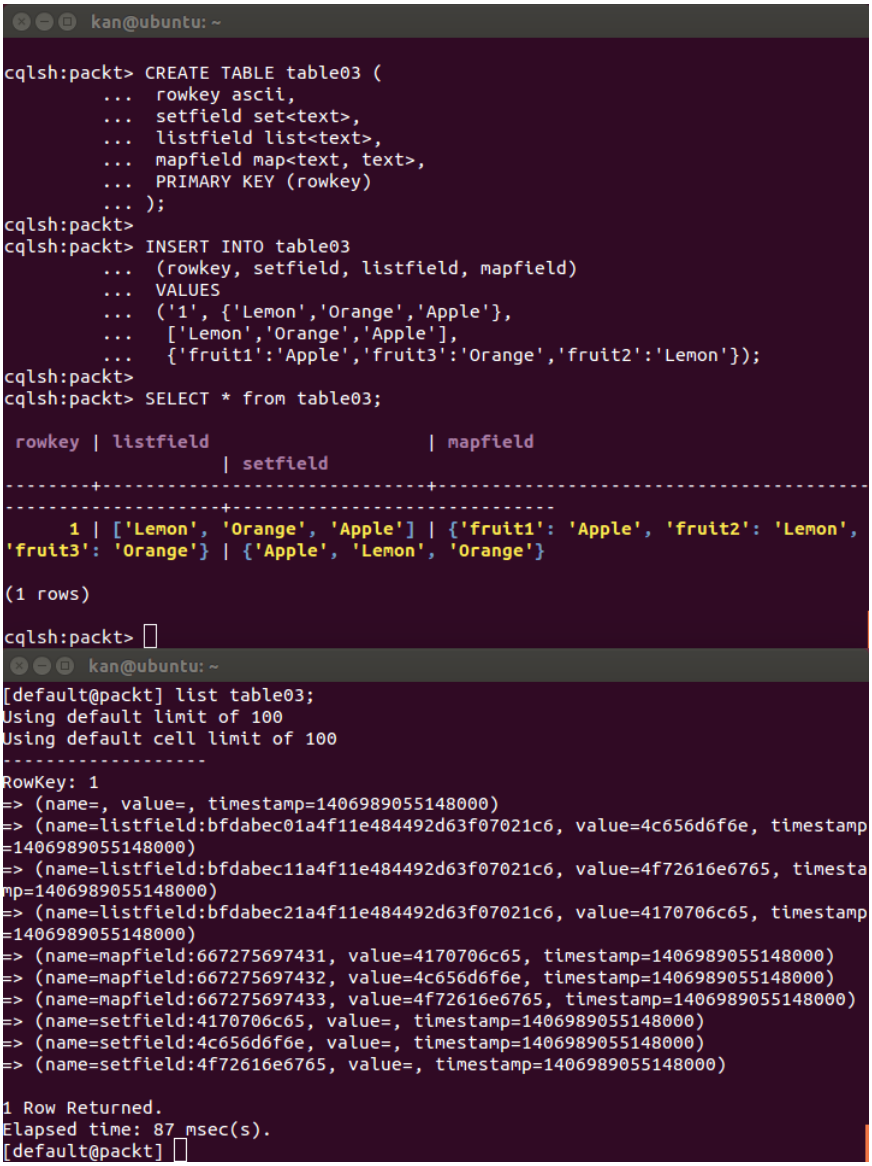
```
kan@ubuntu: ~
=> (name=bigintfield, value=000000003b9aca00, timestamp=1406967910163000)
=> (name=blobfield, value=414243, timestamp=1406967910163000)
=> (name=booleanfield, value=01, timestamp=1406967910163000)
=> (name=decimalfield, value=000000010a, timestamp=1406967910163000)
=> (name=doublefield, value=3ff1f9add3739636, timestamp=1406967910163000)
=> (name=floatfield, value=3f8fcd68, timestamp=1406967910163000)
=> (name=inetfield, value=20010db885a3004210008a2e03707334, timestamp=1406979926
467000)
=> (name=intfield, value=00000001, timestamp=1406967910163000)
=> (name=textfield, value=e8b584e6ba90e68f90e4be9b414243, timestamp=140699983297
9000)
=> (name=timestampfield, value=00000145b395fef8, timestamp=1406967910163000)
=> (name=timeuuidfield, value=84763d401a1e11e484492d63f07021c6, timestamp=140696
7910163000)
=> (name=uuidfield, value=60903075d9e1404f86dc9670f42ea10b, timestamp=1406967910
163000)
=> (name=varcharfield, value=414243, timestamp=1406967910163000)
=> (name=varintfield, value=01, timestamp=1406967910163000)

1 Row Returned.
Elapsed time: 128 msec(s).
[default@packt]
```

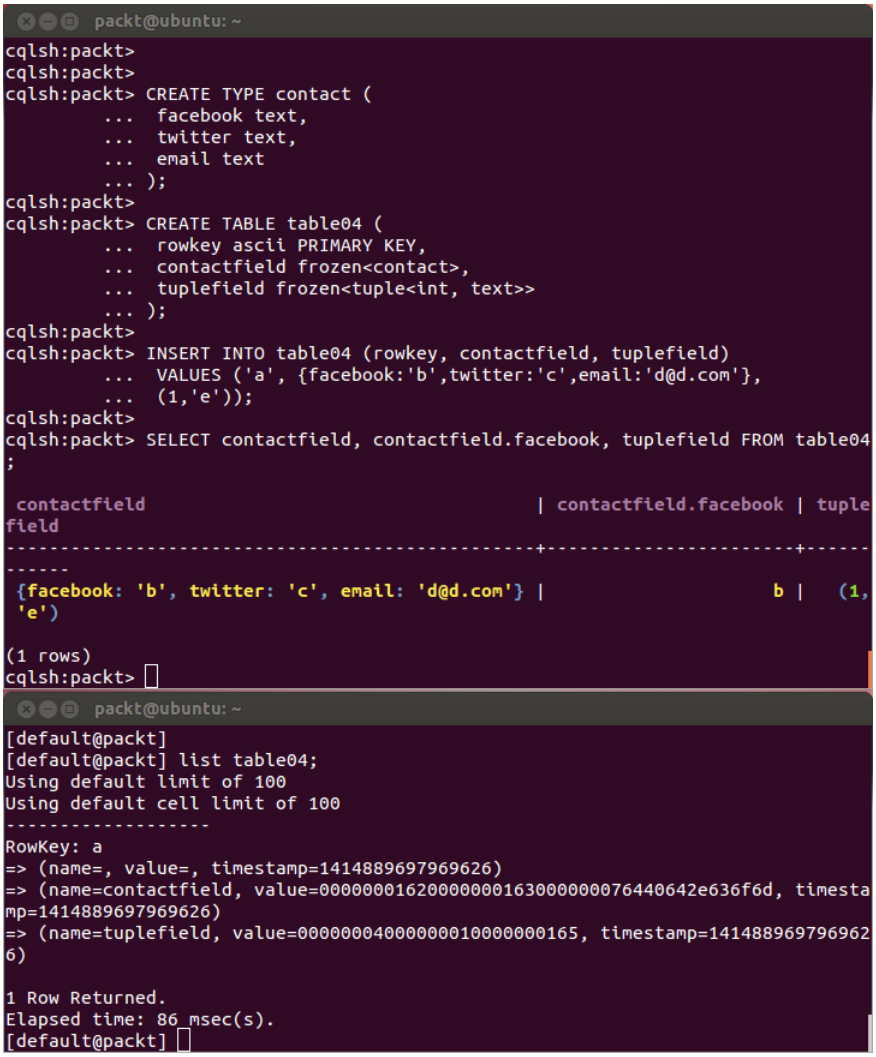
```
packt@ubuntu: ~
Using default limit of 100
Using default cell limit of 100
-------------------
RowKey: 1
=> (name=, value=, timestamp=1416811483812312)
=> (name=asciifield, value=414243, timestamp=1416811483812312)
=> (name=bigintfield, value=000000003b9aca00, timestamp=1416811483812312)
=> (name=blobfield, value=414243, timestamp=1416811483812312)
=> (name=booleanfield, value=01, timestamp=1416811483812312)
=> (name=decimalfield, value=000000010a, timestamp=1416811483812312)
=> (name=doublefield, value=3ff1f9add3739636, timestamp=1416811483812312)
=> (name=floatfield, value=3f8fcd68, timestamp=1416811483812312)
=> (name=inetfield, value=c0a80001, timestamp=1416811483812312)
=> (name=intfield, value=00000001, timestamp=1416811483812312)
=> (name=textfield, value=414243, timestamp=1416811483812312)
=> (name=timestampfield, value=00000145b395fef8, timestamp=1416811483812312)
=> (name=timeuuidfield, value=5f96213073a511e4a62fa92bc9056ee6, timestamp=141681
1483812312)
=> (name=uuidfield, value=62a866ba149b4eac8c9ef5400f52dfec, timestamp=1416811483
812312)
=> (name=varcharfield, value=414243, timestamp=1416811483812312)
=> (name=varintfield, value=01, timestamp=1416811483812312)

1 Row Returned.
Elapsed time: 90 msec(s).
[default@packt]
```


kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> SELECT now(), timeuuidfield, dateOf(timeuuidfield),
... unixTimestampOf(timeuuidfield),
... minTimeuuid(dateOf(timeuuidfield)),
... maxTimeuuid(dateOf(timeuuidfield))
... FROM table01;
now() | timeuuidfield | dateOf(timeuuidfield) | unixTimestampOf(timeuuidfield) | minTimeuuid(dateOf(timeuuidfield)) | maxTimeuuid(dateOf(timeuuidfield))
f538c370-1a3e-11e4-8449-2d63f07021c6 | 84763d40-1a1e-11e4-8449-2d63f07021c6 | 2014-08-02 16:25:10+0800 | 1406967910164 | 84763d40-1a1e-11e4-8080-808080808080 | 8476644f-1a1e-11e4-7f7f-7f7f7f7f7f7f
(1 rows)
cqlsh:packt>

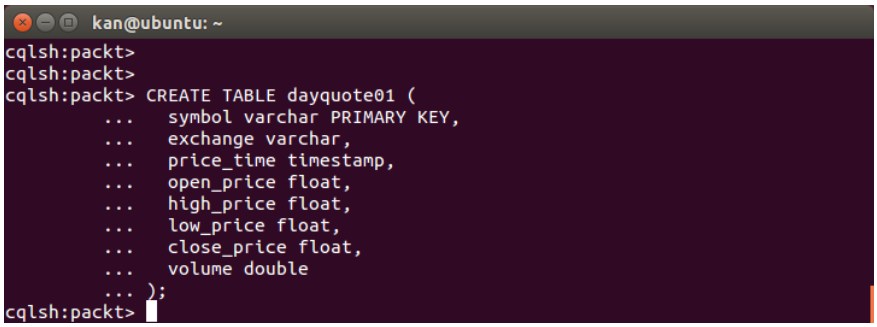

kan@ubuntu: ~
(1 rows)
cqlsh:packt> CREATE TABLE table02 (
... rowkey ascii,
... counterfield counter,
... PRIMARY KEY (rowkey)
...);
cqlsh:packt>
cqlsh:packt> UPDATE table02 SET counterfield = counterfield + 1
... WHERE rowkey = '1';
cqlsh:packt>
cqlsh:packt> SELECT * from table02;
rowkey | counterfield
1 | 1
(1 rows)
cqlsh:packt>

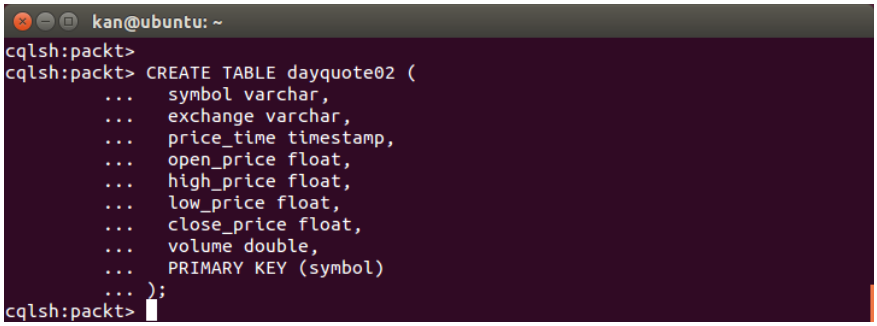

kan@ubuntu: ~
[default@packt]
[default@packt] list table02;
Using default limit of 100
Using default cell limit of 100
RowKey: 1
=> (counter=counterfield, value=1)
1 Row Returned.
Elapsed time: 11 msec(s).
[default@packt]

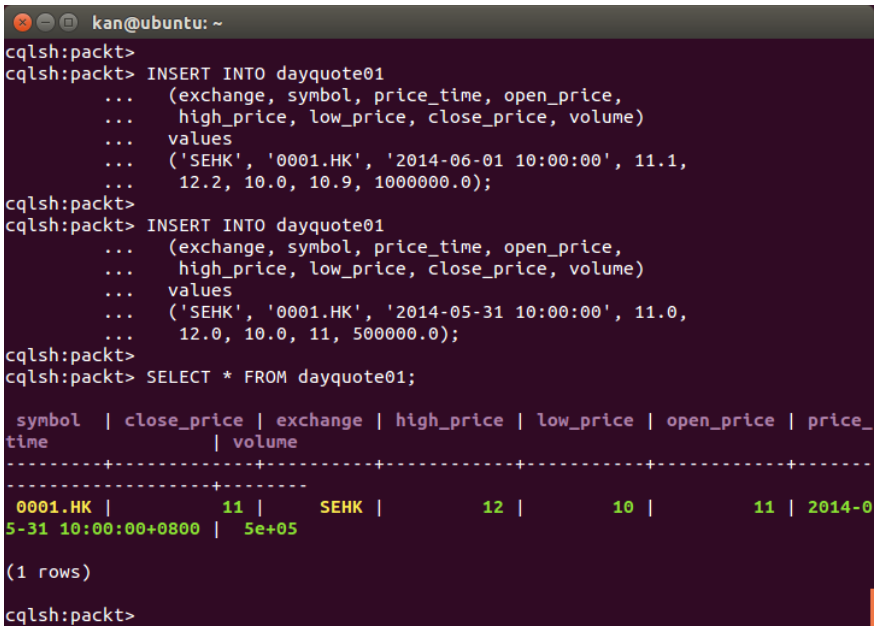

kan@ubuntu: ~
cqlsh:packt> CREATE TABLE table03 (
... rowkey ascii,
... setfield set<text>,
... listfield list<text>,
... mapfield map<text, text>,
... PRIMARY KEY (rowkey)
...);
cqlsh:packt>
cqlsh:packt> INSERT INTO table03
... (rowkey, setfield, listfield, mapfield)
... VALUES
... ('1', {'Lemon','Orange','Apple'},
... ['Lemon','Orange','Apple'],
... {'fruit1':'Apple','fruit3':'Orange','fruit2':'Lemon'});
cqlsh:packt>
cqlsh:packt> SELECT * from table03;
rowkey | listfield | mapfield | setfield
1 | ['Lemon', 'Orange', 'Apple'] | {'fruit1': 'Apple', 'fruit2': 'Lemon', 'fruit3': 'Orange'} | {'Apple', 'Lemon', 'Orange'}
(1 rows)
cqlsh:packt>
kan@ubuntu: ~
[default@packt] list table03;
Using default limit of 100
Using default cell limit of 100
RowKey: 1
=> (name=, value=, timestamp=1406989055148000)
=> (name=listfield:bfdabec01a4f11e484492d63f07021c6, value=4c656d6f6e, timestamp=1406989055148000)
=> (name=listfield:bfdabec11a4f11e484492d63f07021c6, value=4f72616e6765, timestamp=1406989055148000)
=> (name=listfield:bfdabec21a4f11e484492d63f07021c6, value=4170706c65, timestamp=1406989055148000)
=> (name=mapfield:667275697431, value=4170706c65, timestamp=1406989055148000)
=> (name=mapfield:667275697432, value=4c656d6f6e, timestamp=1406989055148000)
=> (name=mapfield:667275697433, value=4f72616e6765, timestamp=1406989055148000)
=> (name=setfield:4170706c65, value=, timestamp=1406989055148000)
=> (name=setfield:4c656d6f6e, value=, timestamp=1406989055148000)
=> (name=setfield:4f72616e6765, value=, timestamp=1406989055148000)
1 Row Returned.
Elapsed time: 87 msec(s).
[default@packt]


packt@ubuntu: ~
cqlsh:packt>
cqlsh:packt>
cqlsh:packt> CREATE TYPE contact (
... facebook text,
... twitter text,
... email text
...);
cqlsh:packt>
cqlsh:packt> CREATE TABLE table04 (
... rowkey ascii PRIMARY KEY,
... contactfield frozen<contact>,
... tuplefield frozen<tuple<int, text>>
...);
cqlsh:packt>
cqlsh:packt> INSERT INTO table04 (rowkey, contactfield, tuplefield)
... VALUES ('a', {facebook:'b',twitter:'c',email:'d@d.com'},
... (1,'e'));
cqlsh:packt>
cqlsh:packt> SELECT contactfield, contactfield.facebook, tuplefield FROM table04
;
contactfield | contactfield.facebook | tuplefield
{facebook: 'b', twitter: 'c', email: 'd@d.com'} | b | (1, 'e')
(1 rows)
cqlsh:packt>
packt@ubuntu: ~
[default@packt]
[default@packt] list table04;
Using default limit of 100
Using default cell limit of 100
RowKey: a
=> (name=, value=, timestamp=1414889697969626)
=> (name=contactfield, value=0000000162000000016300000076440642e636f6d, timestamp=1414889697969626)
=> (name=tuplefield, value=00000004000000010000000165, timestamp=1414889697969626)
1 Row Returned.
Elapsed time: 86 msec(s).
[default@packt]

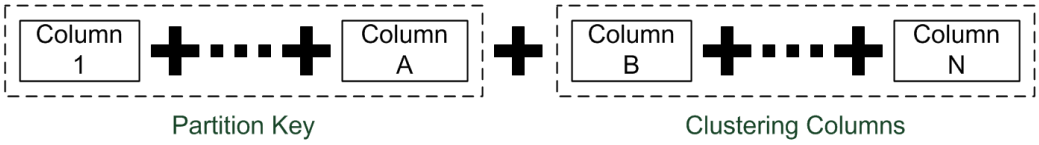
Chapter 4, Indexes







```
kan@ubuntu: ~
[default@packt]
[default@packt] list dayquote01;
Using default limit of 100
Using default cell limit of 100
-------------------
RowKey: 0001.HK
=> (name=, value=, timestamp=1407005232181000)
=> (name=close_price, value=41300000, timestamp=1407005232181000)
=> (name=exchange, value=5345484b, timestamp=1407005232181000)
=> (name=high_price, value=41400000, timestamp=1407005232181000)
=> (name=low_price, value=41200000, timestamp=1407005232181000)
=> (name=open_price, value=41300000, timestamp=1407005232181000)
=> (name=price_time, value=0000014650014900, timestamp=1407005232181000)
=> (name=volume, value=411e848000000000, timestamp=1407005232181000)

1 Row Returned.
Elapsed time: 23 msec(s).
[default@packt]
```




```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> CREATE TABLE dayquote03 (
         ...   symbol varchar,
         ...   exchange varchar,
         ...   price_time timestamp,
         ...   open_price float,
         ...   high_price float,
         ...   low_price float,
         ...   close_price float,
         ...   volume double,
         ...   PRIMARY KEY (symbol, price_time)
         ... );
cqlsh:packt>
```

```
kan@ubuntu: ~

cqlsh:packt> INSERT INTO dayquote03
         ...   (exchange, symbol, price_time, open_price,
         ...    high_price, low_price, close_price, volume)
         ...   values
         ...   ('SEHK', '0001.HK', '2014-06-01 10:00:00', 11.1,
         ...    12.2, 10.0, 10.9, 1000000.0);
cqlsh:packt>
cqlsh:packt> INSERT INTO dayquote03
         ...   (exchange, symbol, price_time, open_price,
         ...    high_price, low_price, close_price, volume)
         ...   values
         ...   ('SEHK', '0001.HK', '2014-05-31 10:00:00', 11.0,
         ...    12.0, 10.0, 11, 500000.0);
cqlsh:packt>
cqlsh:packt> SELECT * FROM dayquote03;

 symbol  | price_time               | close_price | exchange | high_price | low_
price | open_price | volume
---------+--------------------------+-------------+----------+------------+-----
------+------------+--------
 0001.HK | 2014-05-31 10:00:00+0800 |          11 |     SEHK |         12 |
   10 |         11 |  5e+05
 0001.HK | 2014-06-01 10:00:00+0800 |        10.9 |     SEHK |       12.2 |
   10 |       11.1 |  1e+06

(2 rows)

cqlsh:packt>
```

```
kan@ubuntu: ~
RowKey: 0001.HK
=> (name=2014-05-31 10\:00+0800:, value=, timestamp=1407018947768000)
=> (name=2014-05-31 10\:00+0800:close_price, value=41300000, timestamp=140701894
7768000)
=> (name=2014-05-31 10\:00+0800:exchange, value=5345484b, timestamp=140701894776
8000)
=> (name=2014-05-31 10\:00+0800:high_price, value=41400000, timestamp=1407018947
768000)
=> (name=2014-05-31 10\:00+0800:low_price, value=41200000, timestamp=14070189477
68000)
=> (name=2014-05-31 10\:00+0800:open_price, value=41300000, timestamp=1407018947
768000)
=> (name=2014-05-31 10\:00+0800:volume, value=411e848000000000, timestamp=140701
8947768000)
=> (name=2014-06-01 10\:00+0800:, value=, timestamp=1407018947757000)
=> (name=2014-06-01 10\:00+0800:close_price, value=412e6666, timestamp=140701894
7757000)
=> (name=2014-06-01 10\:00+0800:exchange, value=5345484b, timestamp=140701894775
7000)
=> (name=2014-06-01 10\:00+0800:high_price, value=41433333, timestamp=1407018947
757000)
=> (name=2014-06-01 10\:00+0800:low_price, value=41200000, timestamp=14070189477
57000)
=> (name=2014-06-01 10\:00+0800:open_price, value=4131999a, timestamp=1407018947
757000)
=> (name=2014-06-01 10\:00+0800:volume, value=412e848000000000, timestamp=140701
8947757000)

1 Row Returned.
Elapsed time: 106 msec(s).
[default@packt]
```

```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> CREATE TABLE dayquote04 (
         ...   symbol varchar,
         ...   exchange varchar,
         ...   price_time timestamp,
         ...   open_price float,
         ...   high_price float,
         ...   low_price float,
         ...   close_price float,
         ...   volume double,
         ...   PRIMARY KEY ((exchange, symbol), price_time)
         ... );
cqlsh:packt>
cqlsh:packt> INSERT INTO dayquote04
         ...   (exchange, symbol, price_time, open_price,
         ...    high_price, low_price, close_price, volume)
         ...   values
         ...   ('SEHK', '0001.HK', '2014-06-01 10:00:00', 11.1,
         ...    12.2, 10.0, 10.9, 1000000.0);
cqlsh:packt>
cqlsh:packt> INSERT INTO dayquote04
         ...   (exchange, symbol, price_time, open_price,
         ...    high_price, low_price, close_price, volume)
         ...   values
         ...   ('SEHK', '0002.HK', '2014-06-01 10:05:00', 11.0,
         ...    12.0, 10.0, 11, 500000.0);
cqlsh:packt>
cqlsh:packt>
```

```
kan@ubuntu: ~
Using default cell limit of 100
-------------------
RowKey: SEHK:0001.HK
=> (name=2014-06-01 10\:00+0800:, value=, timestamp=1407020296998000)
=> (name=2014-06-01 10\:00+0800:close_price, value=412e6666, timestamp=140702029
6998000)
=> (name=2014-06-01 10\:00+0800:high_price, value=41433333, timestamp=1407020296
998000)
=> (name=2014-06-01 10\:00+0800:low_price, value=41200000, timestamp=14070202969
98000)
=> (name=2014-06-01 10\:00+0800:open_price, value=4131999a, timestamp=1407020296
998000)
=> (name=2014-06-01 10\:00+0800:volume, value=412e848000000000, timestamp=140702
0296998000)
-------------------
RowKey: SEHK:0002.HK
=> (name=2014-06-01 10\:05+0800:, value=, timestamp=1407020298637000)
=> (name=2014-06-01 10\:05+0800:close_price, value=41300000, timestamp=140702029
8637000)
=> (name=2014-06-01 10\:05+0800:high_price, value=41400000, timestamp=1407020298
637000)
=> (name=2014-06-01 10\:05+0800:low_price, value=41200000, timestamp=14070202986
37000)
=> (name=2014-06-01 10\:05+0800:open_price, value=41300000, timestamp=1407020298
637000)
=> (name=2014-06-01 10\:05+0800:volume, value=411e848000000000, timestamp=140702
0298637000)

2 Rows Returned.
Elapsed time: 84 msec(s).
[default@packt]
```

```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> CREATE TABLE dayquote05 (
         ...   symbol varchar,
         ...   exchange varchar,
         ...   price_time timestamp,
         ...   quote_date varchar,
         ...   open_price float,
         ...   high_price float,
         ...   low_price float,
         ...   close_price float,
         ...   volume double,
         ...   PRIMARY KEY ((exchange, quote_date), symbol, price_time)
         ... );
cqlsh:packt>
cqlsh:packt>
```

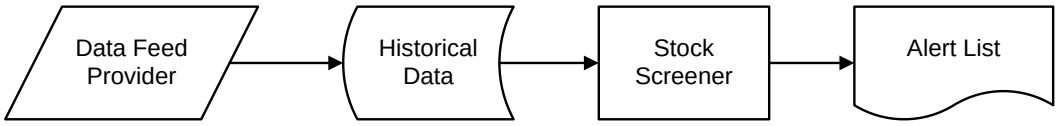
```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> SELECT symbol FROM dayquote04
         ...  WHERE TOKEN(exchange,symbol) < TOKEN('SEHK','0002.HK');

 symbol
---------
 0001.HK

(1 rows)

cqlsh:packt>
```

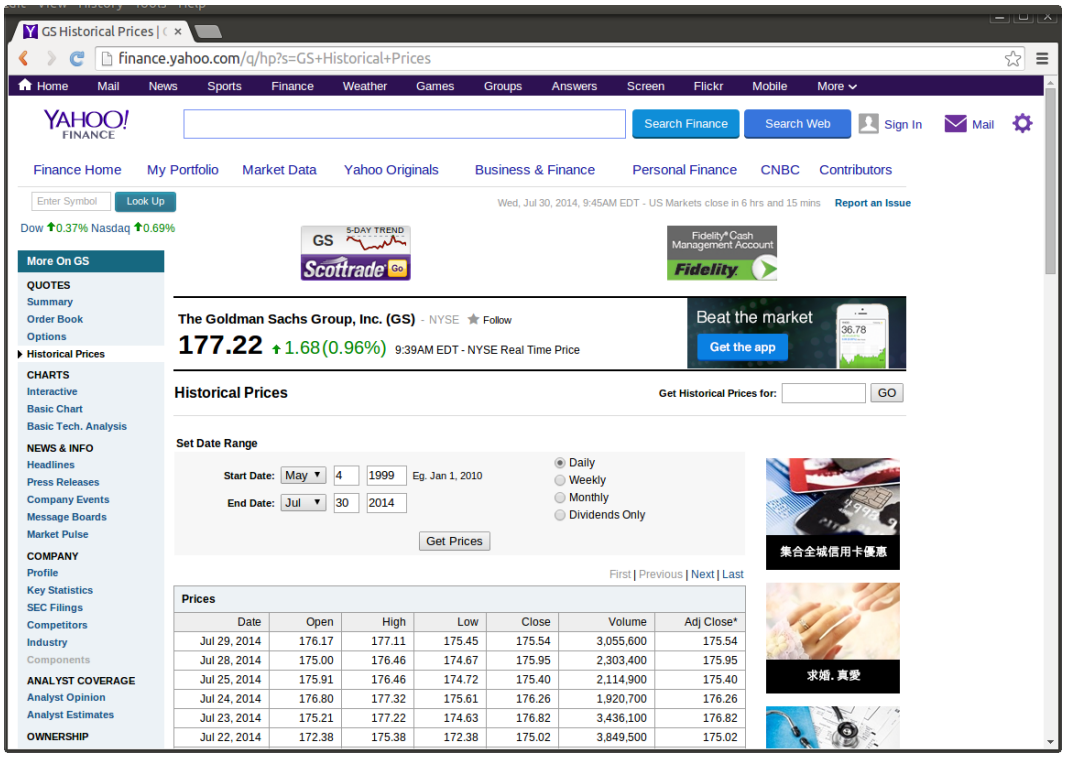
```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> CREATE TABLE dayquote06 (
         ...   symbol varchar,
         ...   exchange varchar,
         ...   sector varchar,
         ...   price_time timestamp,
         ...   quote_date varchar,
         ...   open_price float,
         ...   high_price float,
         ...   low_price float,
         ...   close_price float,
         ...   volume double,
         ...   PRIMARY KEY ((exchange, quote_date), symbol, price_time)
         ... );
cqlsh:packt>
cqlsh:packt> INSERT INTO dayquote06
         ...   (exchange, symbol, sector, price_time, open_price,
         ...    high_price, low_price, close_price, volume, quote_date)
         ...   values
         ...   ('SEHK', '0001.HK', 'Properties', '2014-06-01 10:00:00', 11.1,
         ...    12.2, 10.0, 10.9, 1000000.0, '20140601');
cqlsh:packt>
cqlsh:packt> INSERT INTO dayquote06
         ...   (exchange, symbol, sector, price_time, open_price,
         ...    high_price, low_price, close_price, volume, quote_date)
         ...   values
         ...   ('SEHK', '0002.HK', 'Utilities', '2014-06-01 10:05:00', 11.0,
         ...    12.0, 10.0, 11, 500000.0, '20140601');
cqlsh:packt>
```

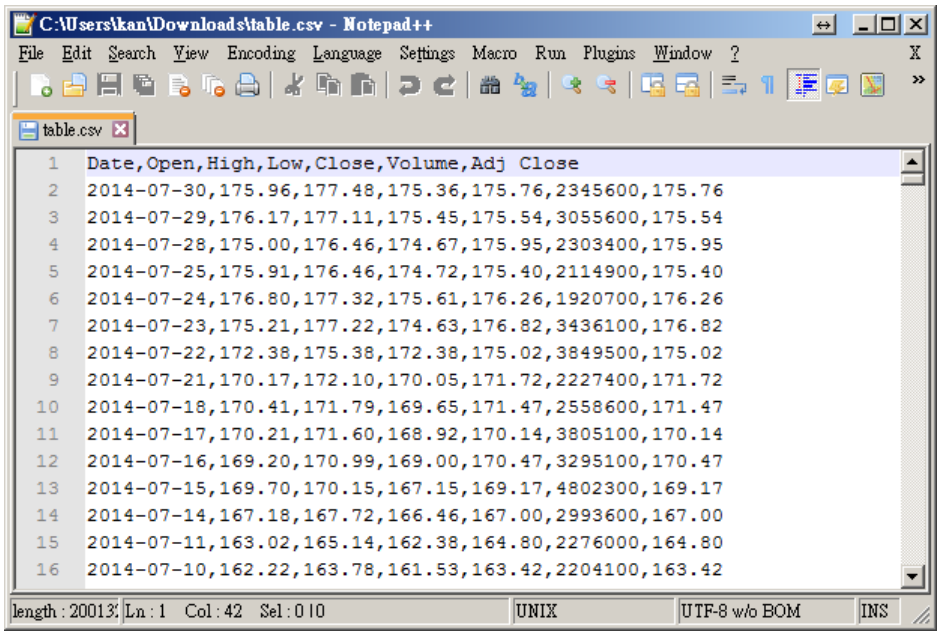
```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> SELECT symbol FROM dayquote06 WHERE sector = 'Properties';
Bad Request: No indexed columns present in by-columns clause with Equal operator
cqlsh:packt>
```

```
kan@ubuntu: ~
cqlsh:packt>
cqlsh:packt> CREATE INDEX dayquote06_sector_idx ON dayquote06 (sector);
cqlsh:packt>
cqlsh:packt> SELECT symbol FROM dayquote06 WHERE sector = 'Properties';

 symbol
---------
 0001.HK

(1 rows)

cqlsh:packt>
```

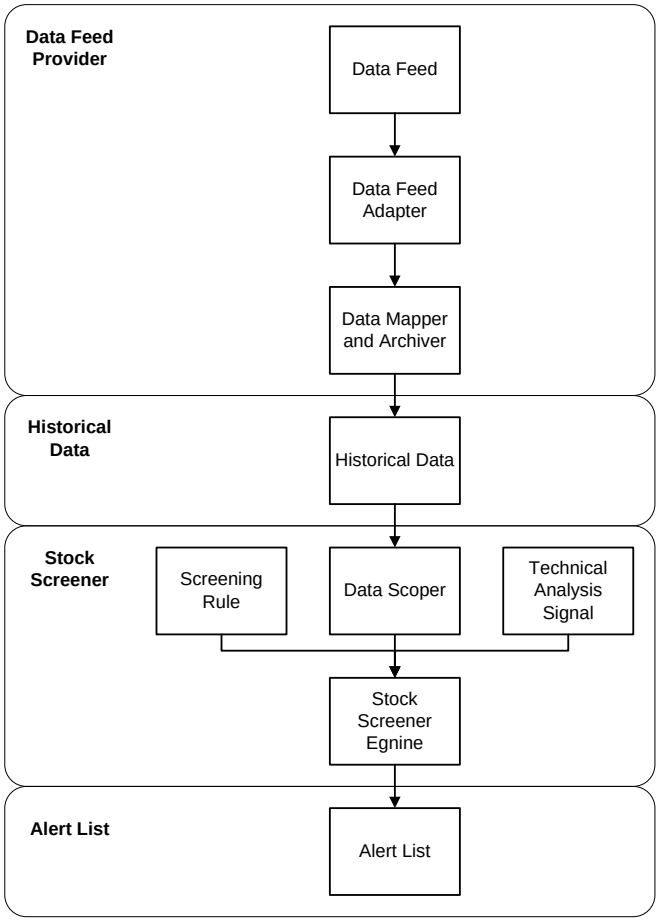
Chapter 5, First-Cut Design and Implementation

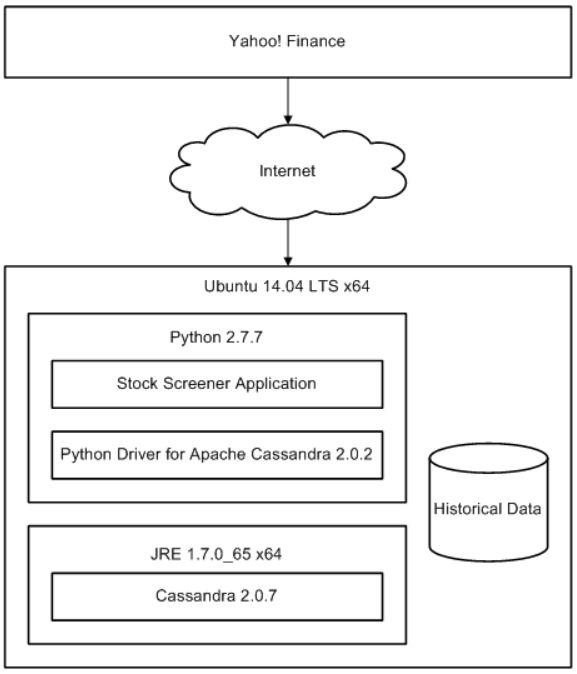


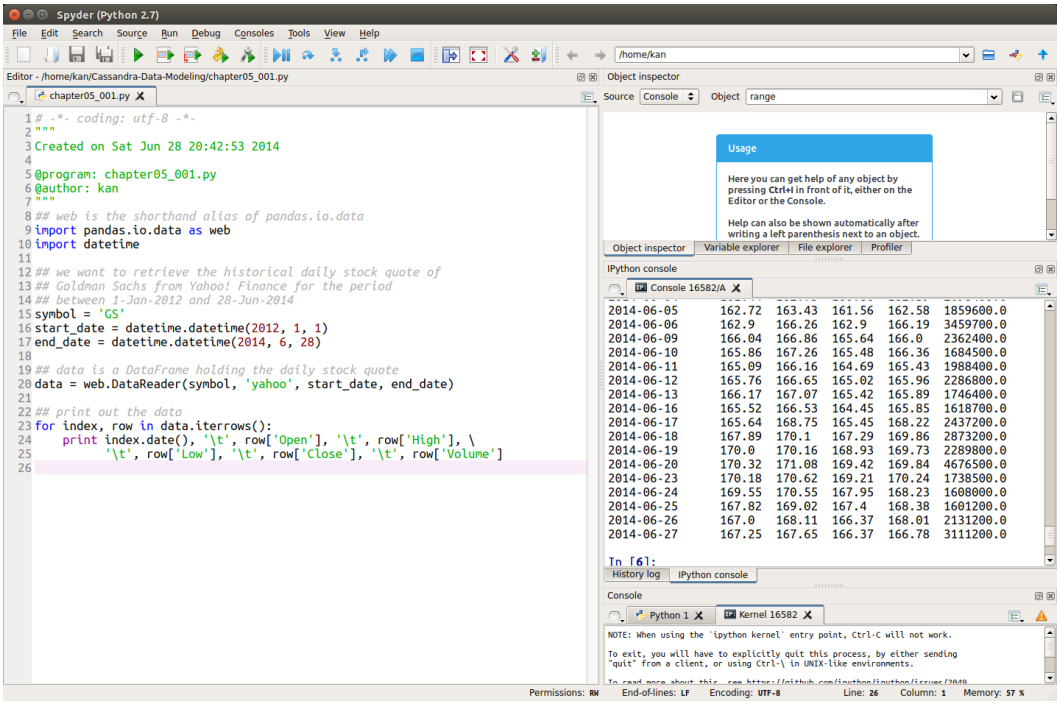


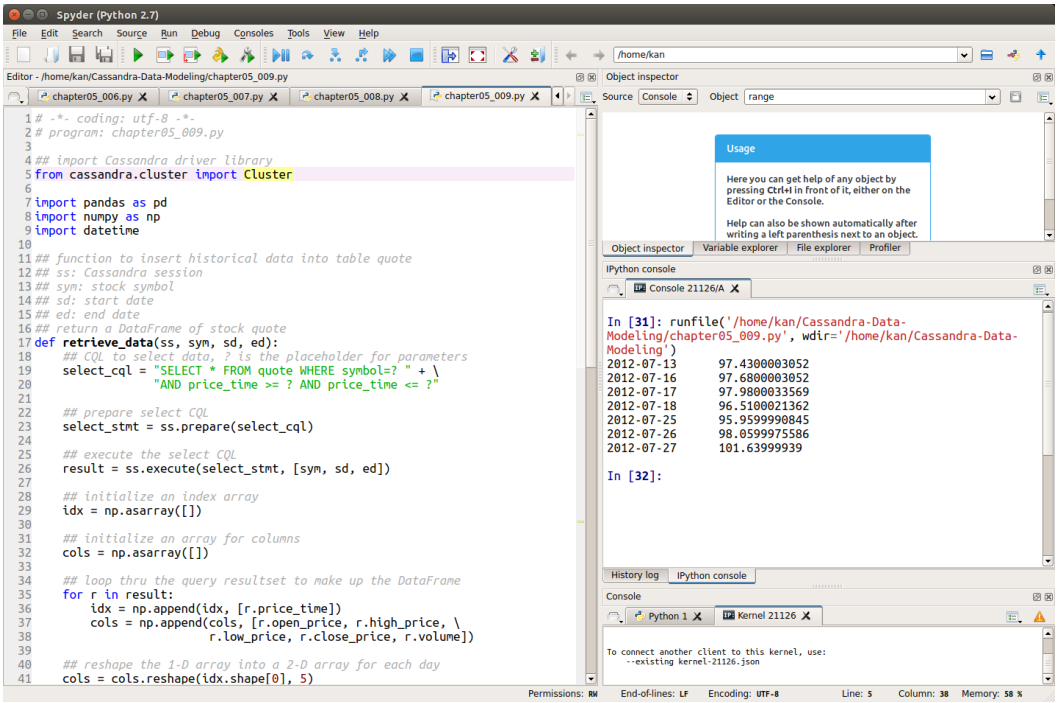


RowKey	2014-07-30 00:00:	2014-07-30 00:00:close_price	2014-07-30 00:00:high_price	2014-07-30 00:00:low_price	2014-07-30 00:00:open_price	2014-07-30 00:00:volume	2014-07-31 00:00:	...
GS		175.76	177.48	175.36	175.96	2345600		

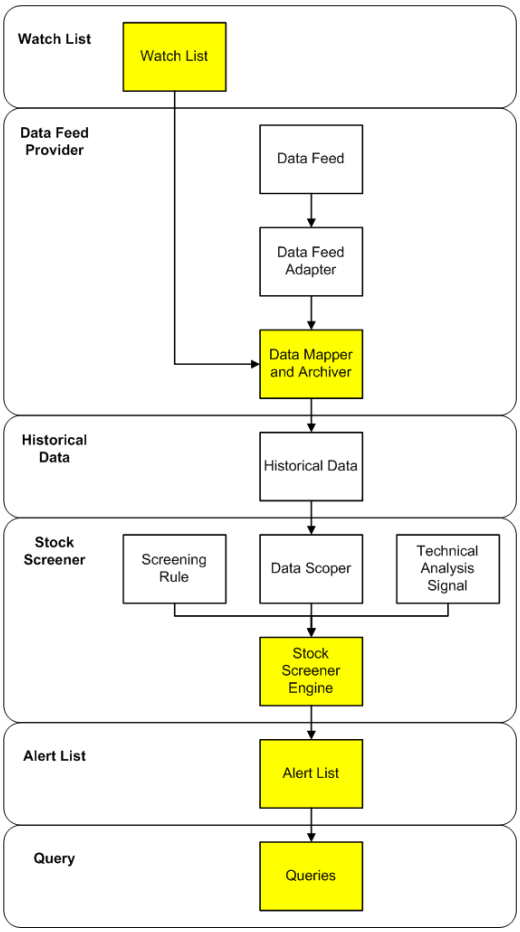


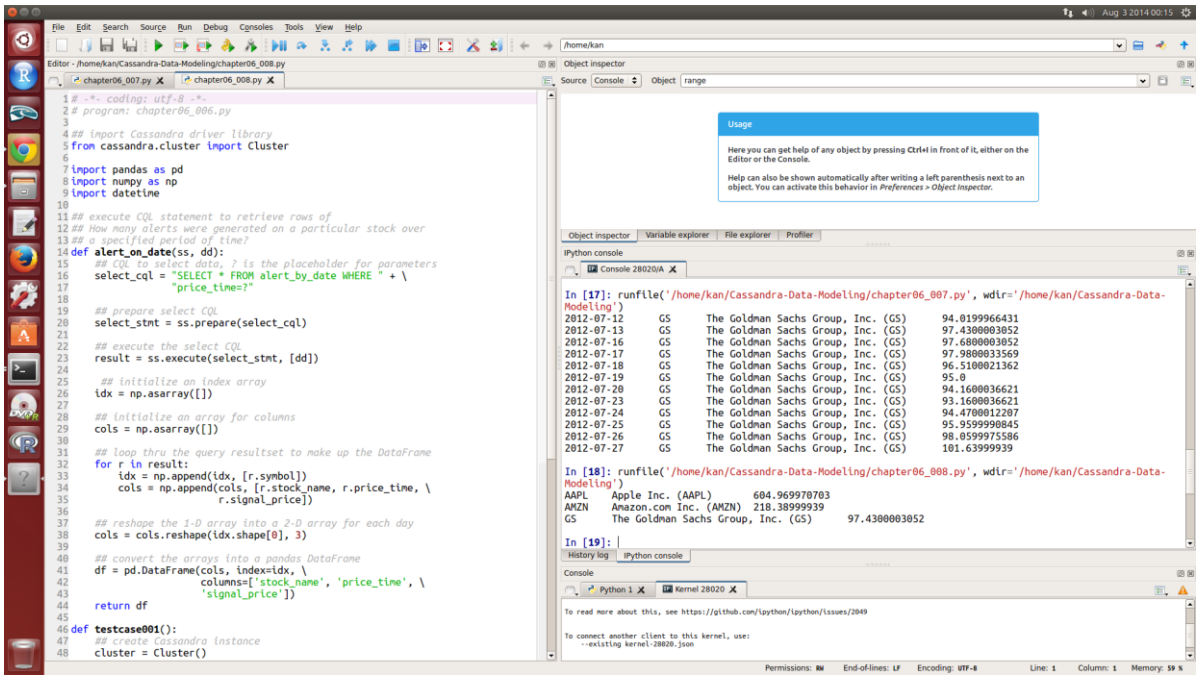



Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Tools View Help
/home/kan
Editor - /home/kan/Cassandra-Data-Modeling/chapter05_001.py
chapter05_001.py
Object inspector
Source Console Object range
Usage
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.
Help can also be shown automatically after writing a left parenthesis next to an object.
Object inspector Variable explorer File explorer Profiler
IPython console
Console 16582/A
2014-06-05 162.72 163.43 161.56 162.58 1859600.0
2014-06-06 162.9 166.26 162.9 166.19 3459700.0
2014-06-09 166.04 166.86 165.64 166.0 2362400.0
2014-06-10 165.86 167.26 165.48 166.36 1684500.0
2014-06-11 165.09 166.16 164.69 165.43 1988400.0
2014-06-12 165.76 166.65 165.02 165.96 2286800.0
2014-06-13 166.17 167.07 165.42 165.89 1746400.0
2014-06-16 165.52 166.53 164.45 165.85 1618700.0
2014-06-17 165.64 168.75 165.45 168.22 2437200.0
2014-06-18 167.89 170.1 167.29 169.86 2873200.0
2014-06-19 170.0 170.16 168.93 169.73 2289800.0
2014-06-20 170.32 171.08 169.42 169.84 4676500.0
2014-06-23 170.18 170.62 169.21 170.24 1738500.0
2014-06-24 169.55 170.55 167.95 168.23 1608000.0
2014-06-25 167.82 169.02 167.4 168.38 1601200.0
2014-06-26 167.0 168.11 166.37 168.01 2131200.0
2014-06-27 167.25 167.65 166.37 166.78 3111200.0
In [6]:
History log IPython console
Console
Python 1 Kernel 16582
NOTE: When using the `ipython kernel` entry point, Ctrl-C will not work.
To exit, you will have to explicitly quit this process, by either sending "quit" from a client, or using Ctrl-\ in UNIX-like environments.
Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 26 Column: 1 Memory: 57 %

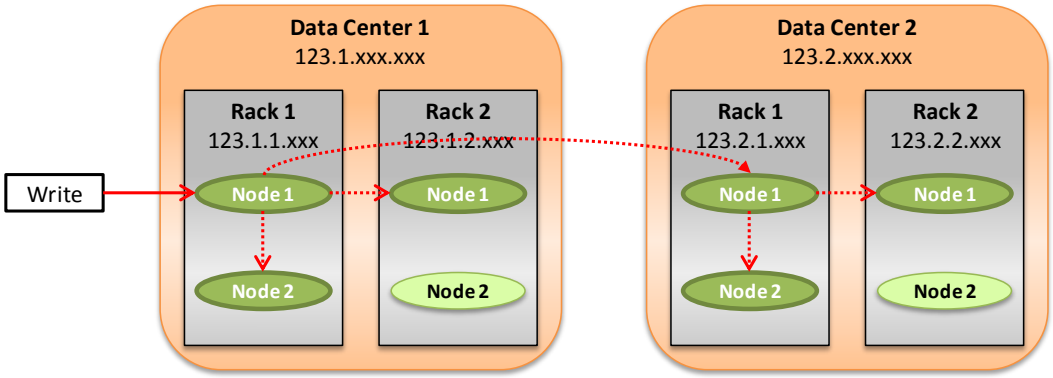

Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Tools View Help
/home/kan
Editor - /home/kan/Cassandra-Data-Modeling/chapter05_009.py
chapter05_006.py chapter05_007.py chapter05_008.py chapter05_009.py
Object inspector
Source Console Object range
Usage
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.
Help can also be shown automatically after writing a left parenthesis next to an object.
Object inspector Variable explorer File explorer Profiler
IPython console
Console 21126/A
In [31]: runfile('/home/kan/Cassandra-Data-Modeling/chapter05_009.py', wdir='/home/kan/Cassandra-Data-Modeling')
2012-07-13 97.4300003052
2012-07-16 97.6800003052
2012-07-17 97.9800033569
2012-07-18 96.5100021362
2012-07-25 95.9599990845
2012-07-26 98.0599975586
2012-07-27 101.63999939
In [32]:
History log IPython console
Console
Python 1 Kernel 21126
To connect another client to this kernel, use:
--existing kernel-21126.json
Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 5 Column: 38 Memory: 58 %

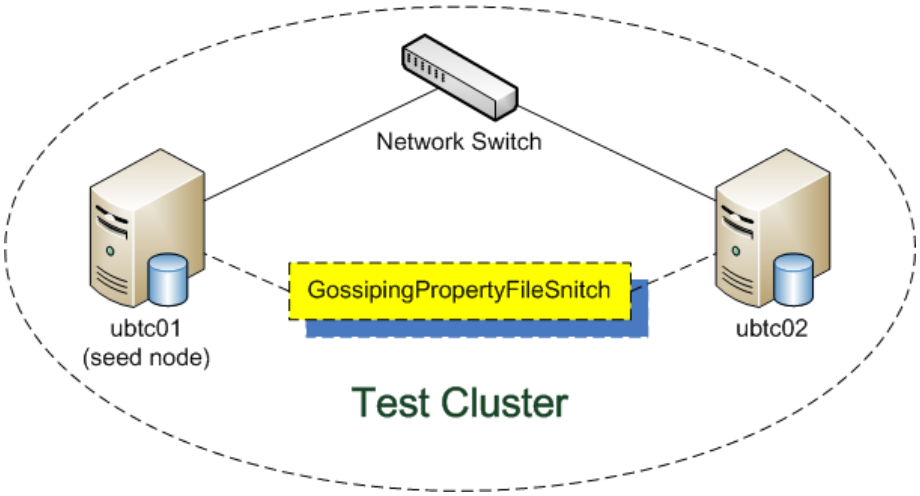
Chapter 6, Enhancing a Version

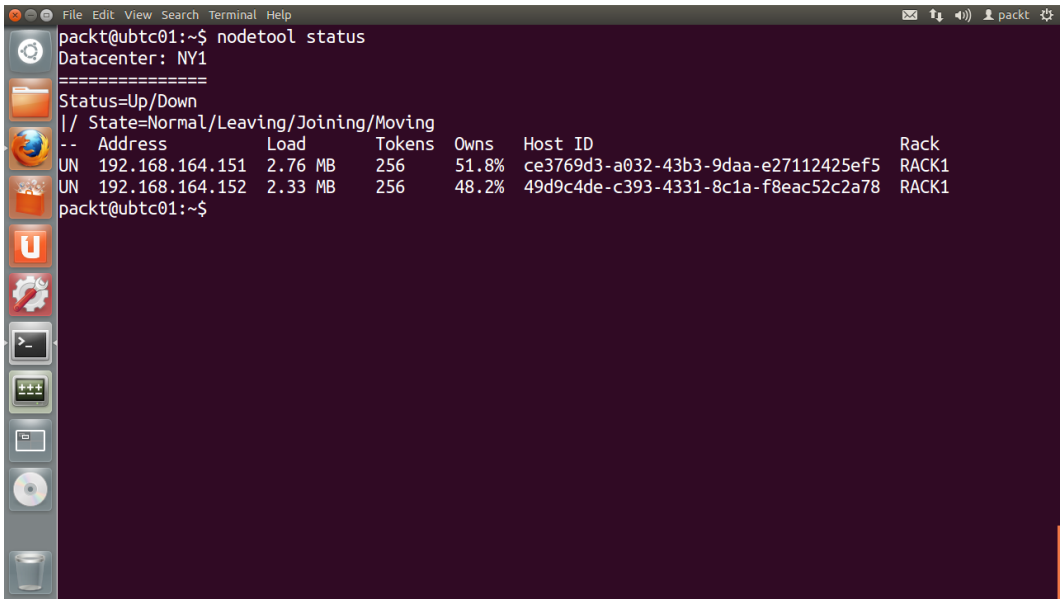


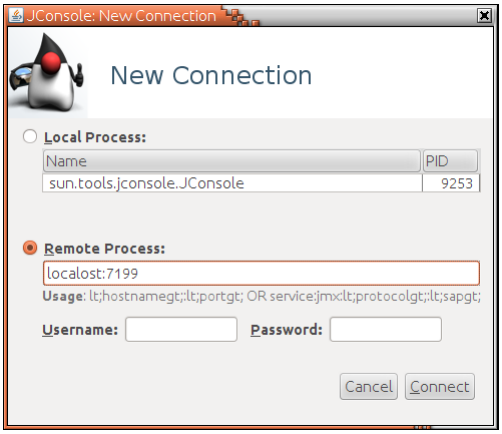


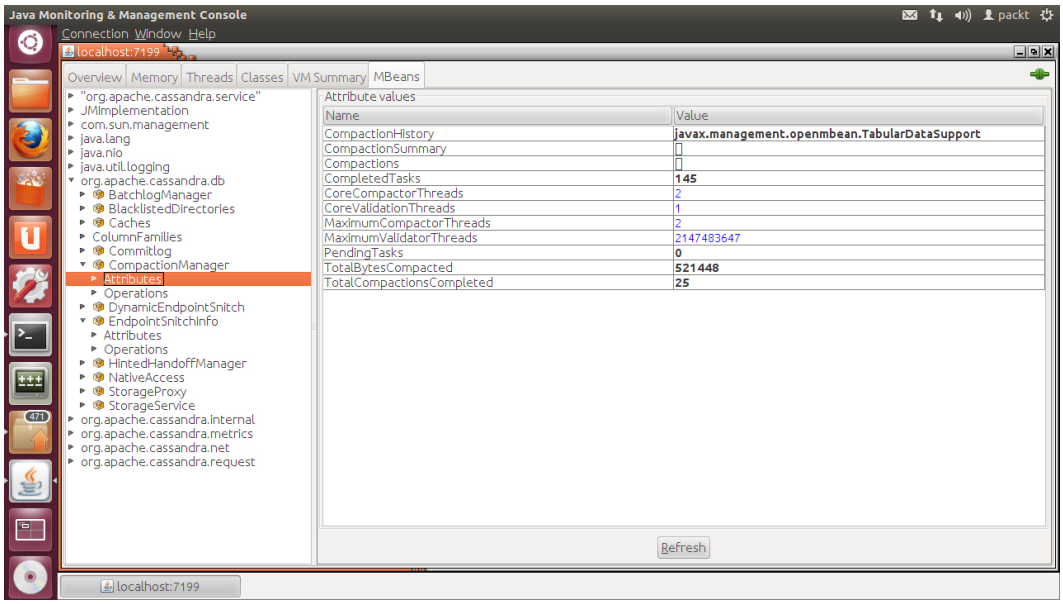
Chapter 7, Deployment and Monitoring

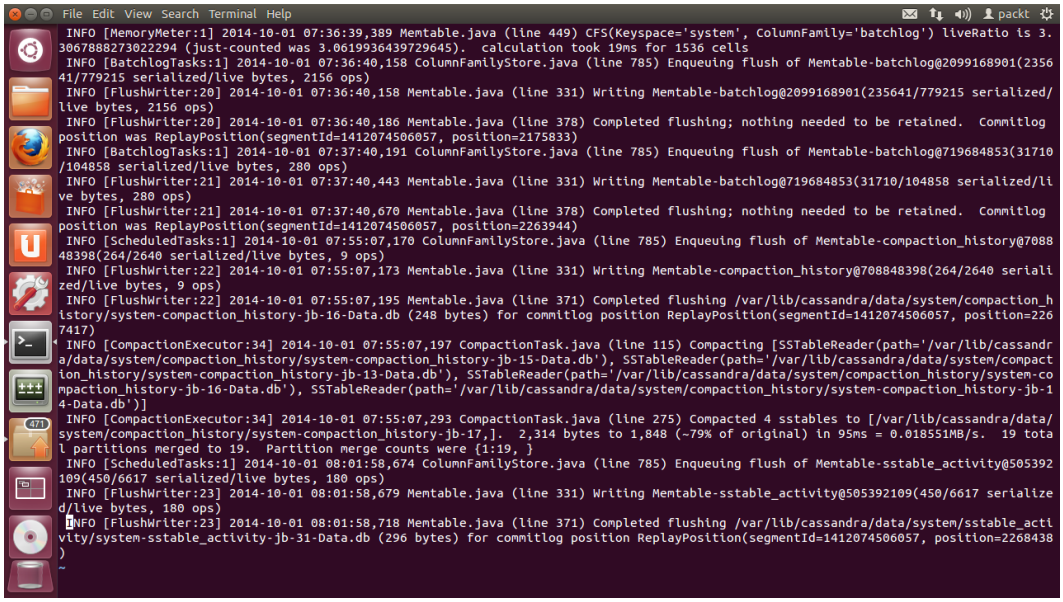


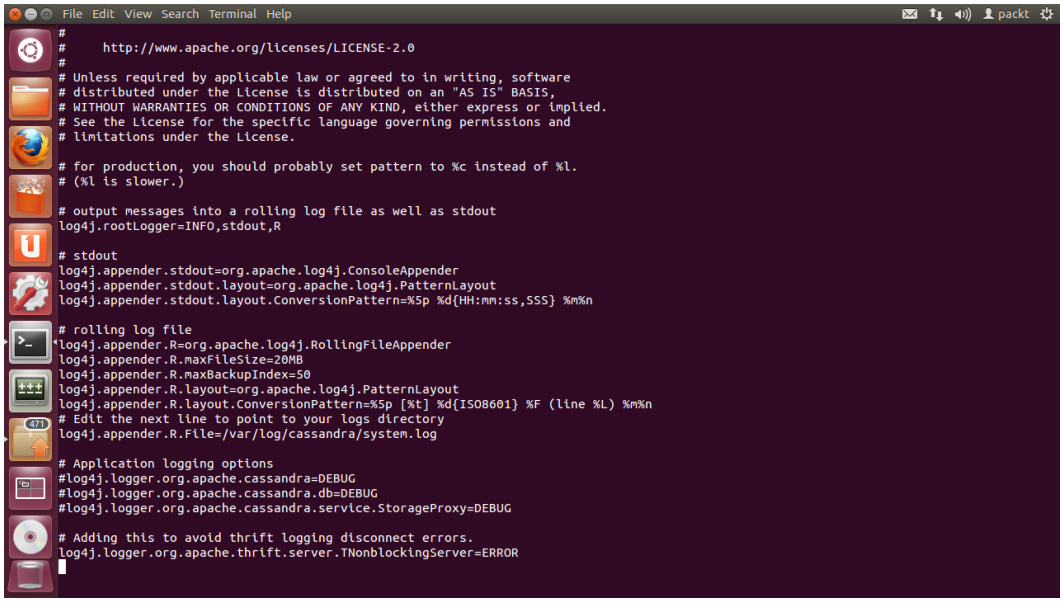





JConsole: New Connection
New Connection
Local Process:
Name
PID
sun.tools.jconsole.JConsole
9253
Remote Process:
localost:7199
Usage: lt;hostnamegt;:lt;portgt; OR service:jmx:lt;protocolgt;:lt;sapgt;
Username:
Password:
Cancel
Connect


Java Monitoring & Management Console
Connection Window Help
localhost:7199
Overview
Memory
Threads
Classes
VM Summary
MBeans
"org.apache.cassandra.service"
JMImplementation
com.sun.management
java.lang
java.nio
java.util.logging
org.apache.cassandra.db
BatchlogManager
BlacklistedDirectories
Caches
ColumnFamilies
Commitlog
CompactionManager
Attributes
Operations
DynamicEndpointSnitch
EndpointSnitchInfo
Attributes
Operations
HintedHandoffManager
NativeAccess
StorageProxy
StorageService
org.apache.cassandra.internal
org.apache.cassandra.metrics
org.apache.cassandra.net
org.apache.cassandra.request
Attribute values
Name
Value
CompactionHistory
javax.management.openmbean.TabularDataSupport
CompactionSummary
Compactions
CompletedTasks
145
CoreCompactorThreads
2
CoreValidationThreads
1
MaximumCompactorThreads
2
MaximumValidatorThreads
2147483647
PendingTasks
0
TotalBytesCompacted
521448
TotalCompactionsCompleted
25
Refresh
localhost:7199


File Edit View Search Terminal Help
INFO [MemoryMeter:1] 2014-10-01 07:36:39,389 Memtable.java (line 449) CFS(Keyspace='system', ColumnFamily='batchlog') liveRatio is 3.3067888273022294 (just-counted was 3.0619936439729645). calculation took 19ms for 1536 cells
INFO [BatchlogTasks:1] 2014-10-01 07:36:40,158 ColumnFamilyStore.java (line 785) Enqueuing flush of Memtable-batchlog@2099168901(235641/779215 serialized/live bytes, 2156 ops)
INFO [FlushWriter:20] 2014-10-01 07:36:40,158 Memtable.java (line 331) Writing Memtable-batchlog@2099168901(235641/779215 serialized/live bytes, 2156 ops)
INFO [FlushWriter:20] 2014-10-01 07:36:40,186 Memtable.java (line 378) Completed flushing; nothing needed to be retained. Commitlog position was ReplayPosition(segmentId=1412074506057, position=2175833)
INFO [BatchlogTasks:1] 2014-10-01 07:37:40,191 ColumnFamilyStore.java (line 785) Enqueuing flush of Memtable-batchlog@719684853(31710/104858 serialized/live bytes, 280 ops)
INFO [FlushWriter:21] 2014-10-01 07:37:40,443 Memtable.java (line 331) Writing Memtable-batchlog@719684853(31710/104858 serialized/live bytes, 280 ops)
INFO [FlushWriter:21] 2014-10-01 07:37:40,670 Memtable.java (line 378) Completed flushing; nothing needed to be retained. Commitlog position was ReplayPosition(segmentId=1412074506057, position=2263944)
INFO [ScheduledTasks:1] 2014-10-01 07:55:07,170 ColumnFamilyStore.java (line 785) Enqueuing flush of Memtable-compaction_history@708848398(264/2640 serialized/live bytes, 9 ops)
INFO [FlushWriter:22] 2014-10-01 07:55:07,173 Memtable.java (line 331) Writing Memtable-compaction_history@708848398(264/2640 serialized/live bytes, 9 ops)
INFO [FlushWriter:22] 2014-10-01 07:55:07,195 Memtable.java (line 371) Completed flushing /var/lib/cassandra/data/system/compaction_history/system-compaction_history-jb-16-Data.db (248 bytes) for commitlog position ReplayPosition(segmentId=1412074506057, position=2267417)
INFO [CompactionExecutor:34] 2014-10-01 07:55:07,197 CompactionTask.java (line 115) Compacting [SSTableReader(path='/var/lib/cassandra/data/system/compaction_history/system-compaction_history-jb-15-Data.db'), SSTableReader(path='/var/lib/cassandra/data/system/compaction_history/system-compaction_history-jb-13-Data.db'), SSTableReader(path='/var/lib/cassandra/data/system/compaction_history/system-compaction_history-jb-16-Data.db'), SSTableReader(path='/var/lib/cassandra/data/system/compaction_history/system-compaction_history-jb-14-Data.db')]
INFO [CompactionExecutor:34] 2014-10-01 07:55:07,293 CompactionTask.java (line 275) Compacted 4 sstables to [/var/lib/cassandra/data/system/compaction_history/system-compaction_history-jb-17,]. 2,314 bytes to 1,848 (~79% of original) in 95ms = 0.018551MB/s. 19 total partitions merged to 19. Partition merge counts were {1:19, }
INFO [ScheduledTasks:1] 2014-10-01 08:01:58,674 ColumnFamilyStore.java (line 785) Enqueuing flush of Memtable-sstable_activity@505392109(450/6617 serialized/live bytes, 180 ops)
INFO [FlushWriter:23] 2014-10-01 08:01:58,679 Memtable.java (line 331) Writing Memtable-sstable_activity@505392109(450/6617 serialized/live bytes, 180 ops)
INFO [FlushWriter:23] 2014-10-01 08:01:58,718 Memtable.java (line 371) Completed flushing /var/lib/cassandra/data/system/sstable_activity/system-sstable_activity-jb-31-Data.db (296 bytes) for commitlog position ReplayPosition(segmentId=1412074506057, position=2268438)


File Edit View Search Terminal Help
packt
#
http://www.apache.org/licenses/LICENSE-2.0
#
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

for production, you should probably set pattern to %c instead of %l.
(%l is slower.)

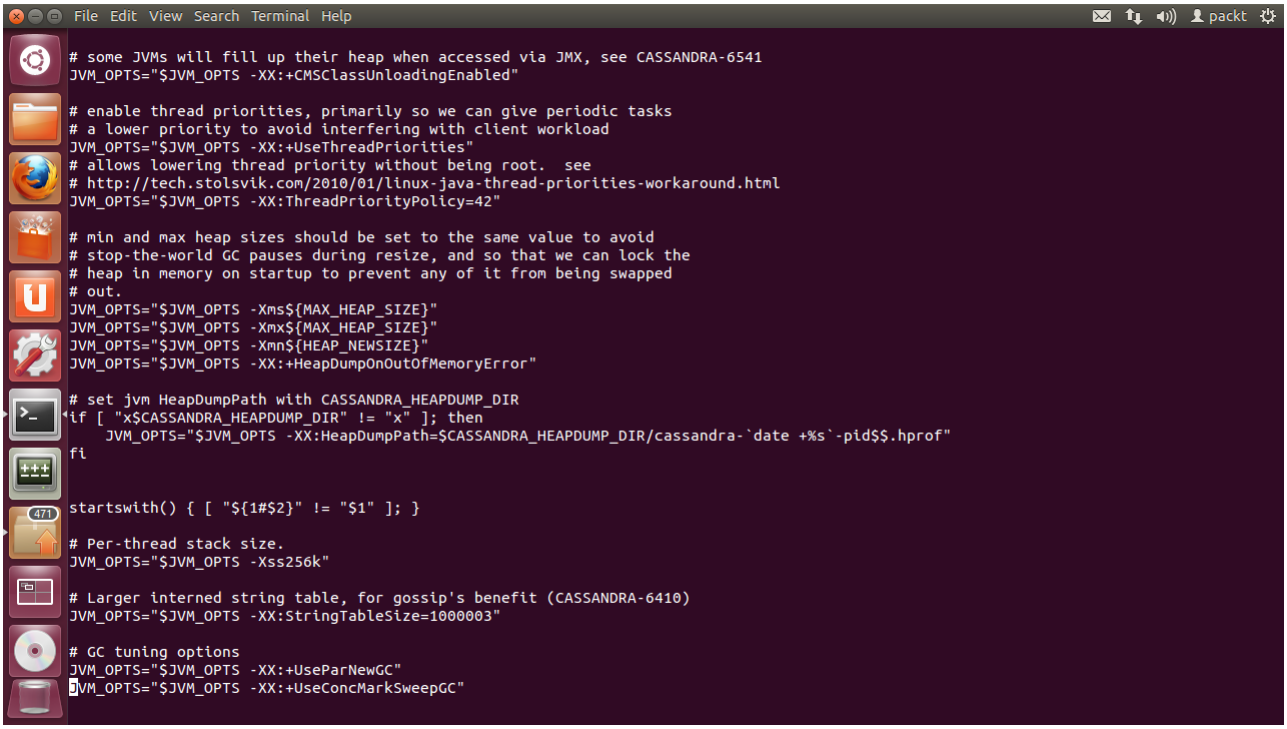
# output messages into a rolling log file as well as stdout
log4j.rootLogger=INFO,stdout,R

# stdout
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%5p %d{HH:mm:ss,SSS} %m%n

# rolling log file
log4j.appender.R=org.apache.log4j.RollingFileAppender
log4j.appender.R.maxFileSize=20MB
log4j.appender.R.maxBackupIndex=50
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R.layout.ConversionPattern=%5p [%t] %d{ISO8601} %F (line %L) %m%n
# Edit the next line to point to your logs directory
log4j.appender.R.File=/var/log/cassandra/system.log

Application logging options
#log4j.logger.org.apache.cassandra=DEBUG
#log4j.logger.org.apache.cassandra.db=DEBUG
#log4j.logger.org.apache.cassandra.service.StorageProxy=DEBUG

# Adding this to avoid thrift logging disconnect errors.
log4j.logger.org.apache.thrift.server.TNonblockingServer=ERROR


File Edit View Search Terminal Help
packt
# some JVMs will fill up their heap when accessed via JMX, see CASSANDRA-6541
JVM_OPTS="$JVM_OPTS -XX:+CMSClassUnloadingEnabled"

# enable thread priorities, primarily so we can give periodic tasks
# a lower priority to avoid interfering with client workload
JVM_OPTS="$JVM_OPTS -XX:+UseThreadPriorities"
# allows lowering thread priority without being root. see
# http://tech.stolsvik.com/2010/01/linux-java-thread-priorities-workaround.html
JVM_OPTS="$JVM_OPTS -XX:ThreadPriorityPolicy=42"

# min and max heap sizes should be set to the same value to avoid
# stop-the-world GC pauses during resize, and so that we can lock the
# heap in memory on startup to prevent any of it from being swapped
# out.
JVM_OPTS="$JVM_OPTS -Xms${MAX_HEAP_SIZE}"
JVM_OPTS="$JVM_OPTS -Xmx${MAX_HEAP_SIZE}"
JVM_OPTS="$JVM_OPTS -Xmn${HEAP_NEWSIZE}"
JVM_OPTS="$JVM_OPTS -XX:+HeapDumpOnOutOfMemoryError"

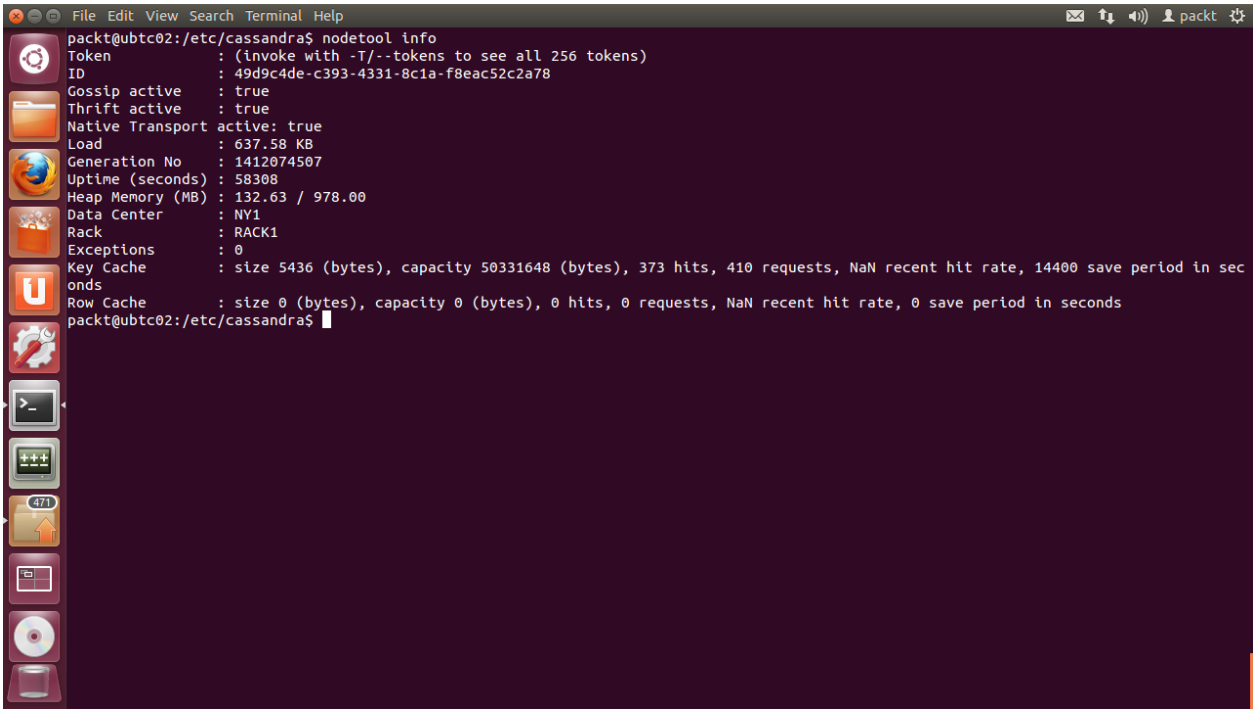
# set jvm HeapDumpPath with CASSANDRA_HEAPDUMP_DIR
if ["x$CASSANDRA_HEAPDUMP_DIR" != "x"]; then
    JVM_OPTS="$JVM_OPTS -XX:HeapDumpPath=$CASSANDRA_HEAPDUMP_DIR/cassandra-`date +%s`-pid$$.hprof"
fi

startswith() { ["${1#$2}" != "$1"]; }

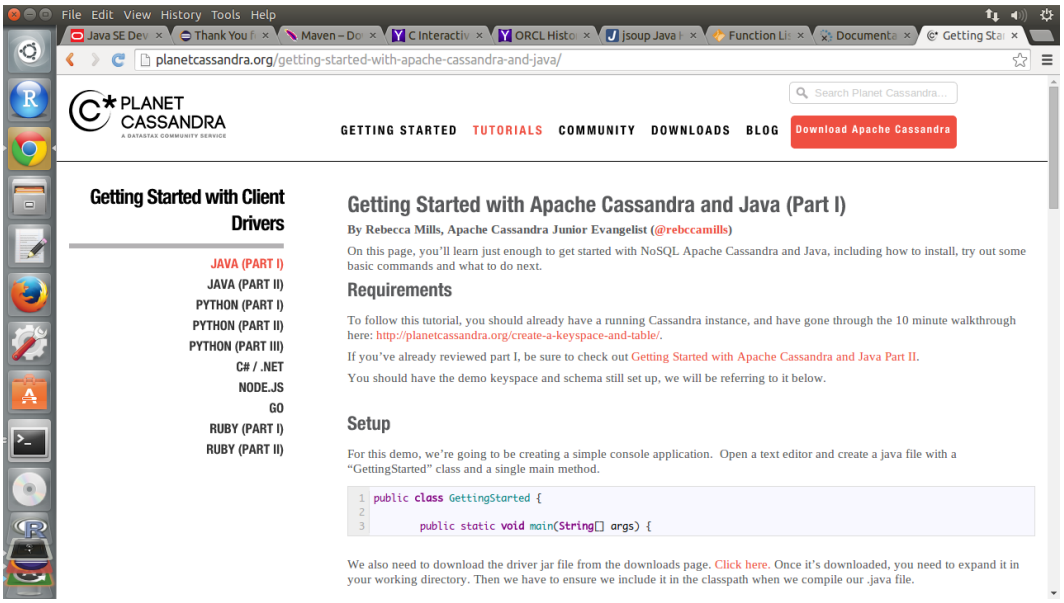
# Per-thread stack size.
JVM_OPTS="$JVM_OPTS -Xss256k"

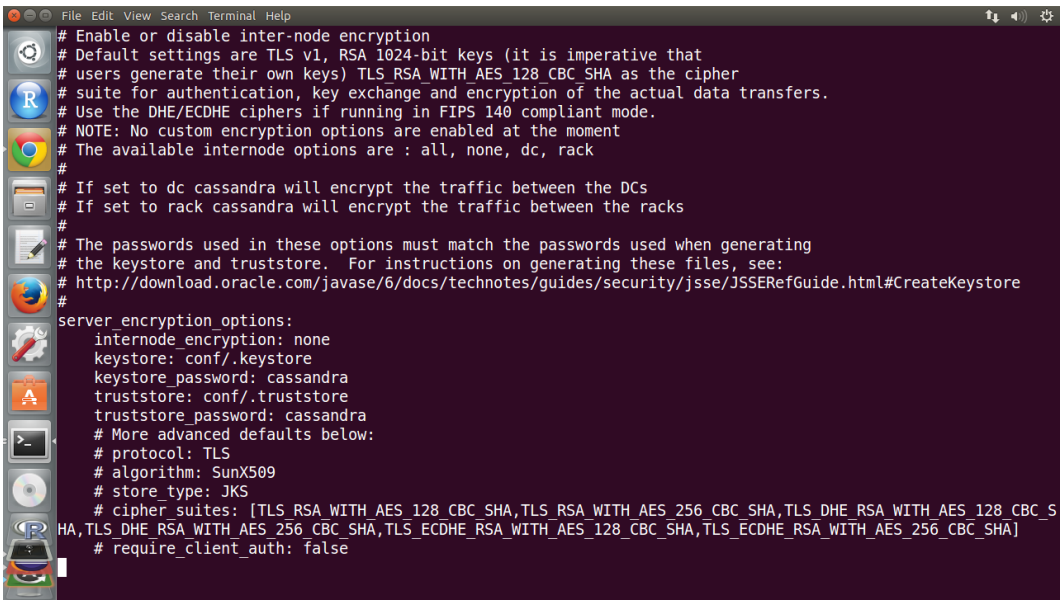
# Larger interned string table, for gossip's benefit (CASSANDRA-6410)
JVM_OPTS="$JVM_OPTS -XX:StringTableSize=1000003"

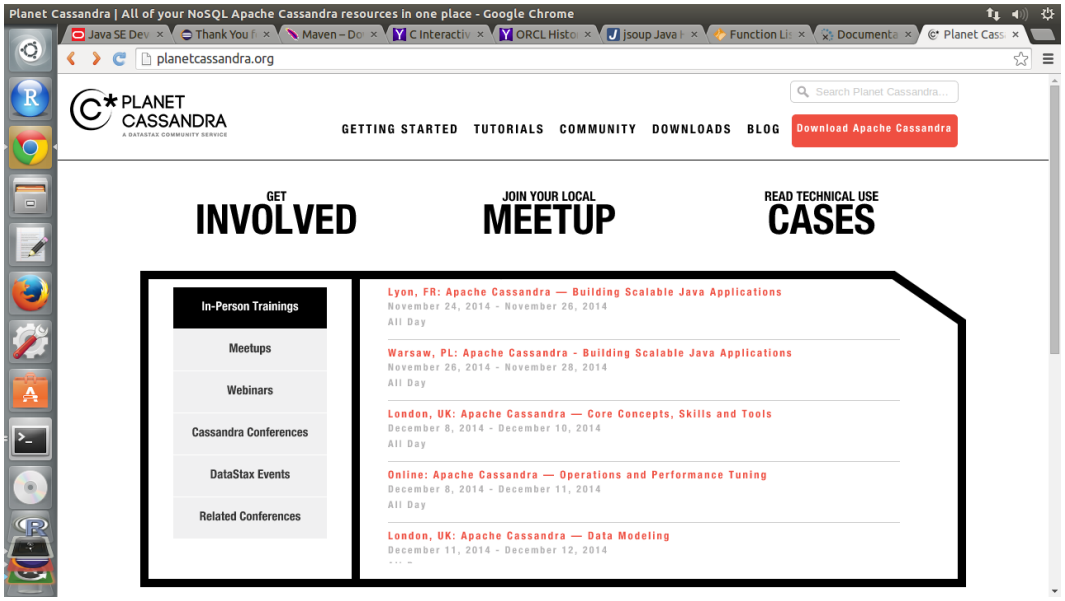
# GC tuning options
JVM_OPTS="$JVM_OPTS -XX:+UseParNewGC"
JVM_OPTS="$JVM_OPTS -XX:+UseConcMarkSweepGC"

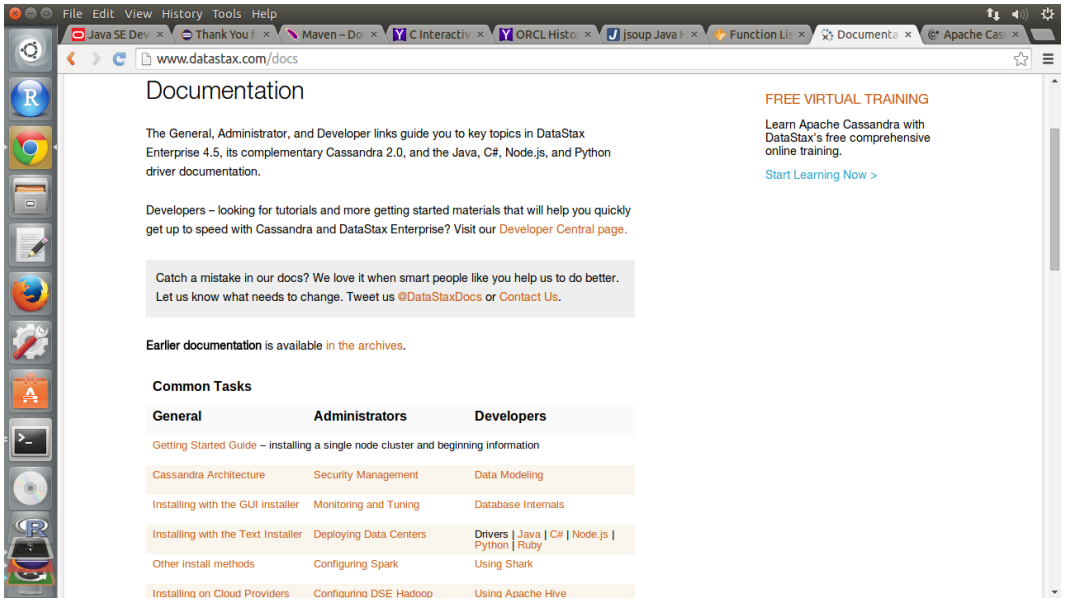

File Edit View Search Terminal Help
packt
packt@ubtc02:/etc/cassandra$ nodetool info
Token : (invoke with -T/--tokens to see all 256 tokens)
ID : 49d9c4de-c393-4331-8c1a-f8eac52c2a78
Gossip active : true
Thrift active : true
Native Transport active: true
Load : 637.58 KB
Generation No : 1412074507
Uptime (seconds) : 58308
Heap Memory (MB) : 132.63 / 978.00
Data Center : NY1
Rack : RACK1
Exceptions : 0
Key Cache : size 5436 (bytes), capacity 50331648 (bytes), 373 hits, 410 requests, NaN recent hit rate, 14400 save period in seconds
Row Cache : size 0 (bytes), capacity 0 (bytes), 0 hits, 0 requests, NaN recent hit rate, 0 save period in seconds
packt@ubtc02:/etc/cassandra$

Chapter 8, Final Thoughts






Planet Cassandra | All of your NoSQL Apache Cassandra resources in one place - Google Chrome
planetcassandra.org
PLANET CASSANDRA
A DATASTAX COMMUNITY SERVICE
Search Planet Cassandra...
GETTING STARTED
TUTORIALS
COMMUNITY
DOWNLOADS
BLOG
Download Apache Cassandra
GET INVOLVED
JOIN YOUR LOCAL MEETUP
READ TECHNICAL USE CASES
In-Person Trainings
Meetups
Webinars
Cassandra Conferences
DataStax Events
Related Conferences
Lyon, FR: Apache Cassandra — Building Scalable Java Applications
November 24, 2014 - November 26, 2014
All Day
Warsaw, PL: Apache Cassandra - Building Scalable Java Applications
November 26, 2014 - November 28, 2014
All Day
London, UK: Apache Cassandra — Core Concepts, Skills and Tools
December 8, 2014 - December 10, 2014
All Day
Online: Apache Cassandra — Operations and Performance Tuning
December 8, 2014 - December 11, 2014
All Day
London, UK: Apache Cassandra — Data Modeling
December 11, 2014 - December 12, 2014


File Edit View History Tools Help
www.datastax.com/docs
Documentation
The General, Administrator, and Developer links guide you to key topics in DataStax Enterprise 4.5, its complementary Cassandra 2.0, and the Java, C#, Node.js, and Python driver documentation.
Developers – looking for tutorials and more getting started materials that will help you quickly get up to speed with Cassandra and DataStax Enterprise? Visit our Developer Central page.
Catch a mistake in our docs? We love it when smart people like you help us to do better. Let us know what needs to change. Tweet us @DataStaxDocs or Contact Us.
Earlier documentation is available in the archives.
Common Tasks
General
Administrators
Developers
Getting Started Guide – installing a single node cluster and beginning information
Cassandra Architecture
Security Management
Data Modeling
Installing with the GUI installer
Monitoring and Tuning
Database Internals
Installing with the Text Installer
Deploying Data Centers
Drivers | Java | C# | Node.js | Python | Ruby
Other install methods
Configuring Spark
Using Shark
Installing on Cloud Providers
Configuring DSE Hadoop
Using Apache Hive
FREE VIRTUAL TRAINING
Learn Apache Cassandra with DataStax's free comprehensive online training.
Start Learning Now >