“VoltDB A database for fast real time processing” Final Project

Overview:

The goal is to investigate VoltDB which is in-memory, high velocity relational database, how we can use this with fast real time processing. We have learned various technologies, tools, strategies to solve problems in class. Using the technologies, we learned and integrate this with VoltDB is something I’d like to demonstrate.

Workflow concept:

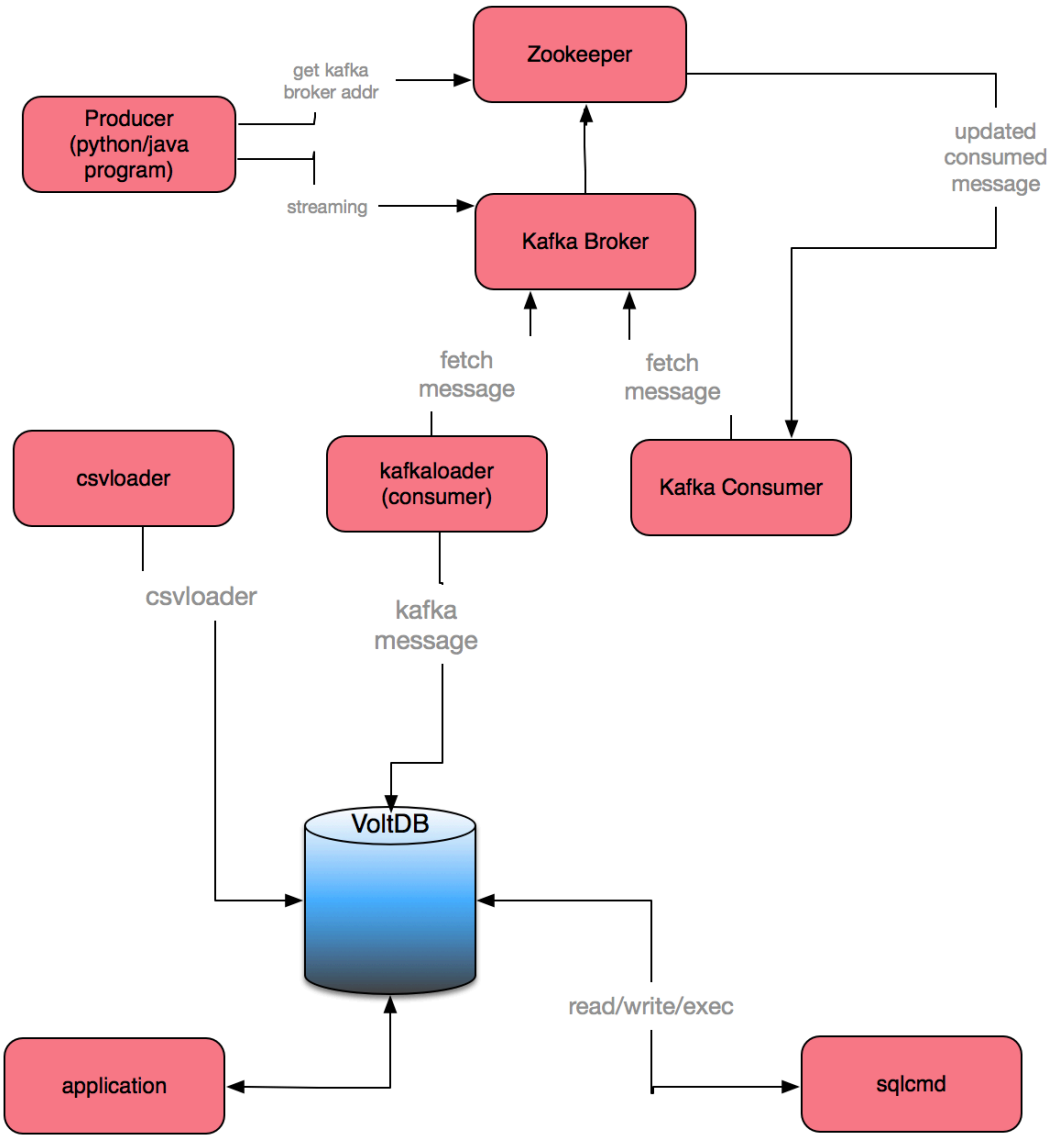
Using Kafka, Zookeeper to stream the data and populate kafka messages to VoltDB, then using VoltDB stored procedure to retrieve the data thru application and run regression on the data.

VoltDB provides:

- **In-memory high velocity relational database to maximize throughput**
- **For new generation of big data applications**
- **On demand scaling**
- **ACID transactions (Atomicity, Consistency, Isolation, Durability)**
- **Partition and replication**
- **Stored procedure interface as well as SQL operations**
- **Lockless database**

VoltDB drawback:

- **VoltDB is in-memory database, it must fit into available memory**
- **Durability for VoltDB is provide through replication across network to another server or writing periodic snapshots to disk.**
- **If you have terabytes of data in database, it won’t scale, although memory is cheaper, but not that cheap**
- **Replication is synchronous within the cluster, but asynchronous between clusters. There is a possibility of lost commited transactions if there is a power outage**
- **Limited use-cases**
- **Running complex queries**



My machine: Mac OSX El Capitan

Installation Requirements:

(1) Kafka: I have downloaded and installed
`kafka_2.11-0.9.0.1.tar`

After Kafka installation, I followed Zookeer & Kafka setup instructions from lecture 5 lab by Marina Popova.

(2) VoltDB Installation:
UISMHLPT4013032:~ tul560$ brew install voltdb

Updating Homebrew...
==>
Downloading https://homebrew.bintray.com/bottles/voltdb-6.9.el_capitan.bottle.tar.gz
100.0%
==>
Pouring voltdb-6.9.el_capitan.bottle.tar.gz
🍺 /usr/local/Cellar/voltdb/6.9: 381 files, 52.5MB
UISMHLPT4013032:~ tul560$

(4)Add to .bash_profile

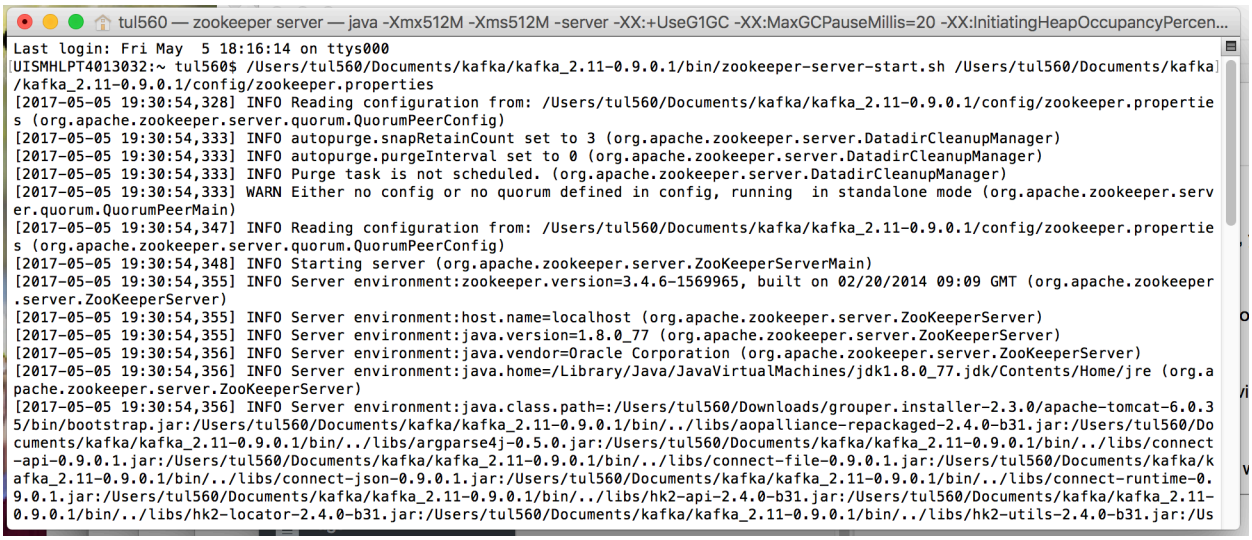
- #.bash_profile modification on Mac
- #Add and change this path to point to your kafka lib
- #for voltdb kafka loaderexport
- ZKLIB=/Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/libs

Inventories:
(1)Directory: /Users/tul560/myvoltdb/finalproject
hads_table.sql (to test with csvloader)
hads2.sql (to test with kafkaloader)
thads2013n.csv
kafka_python_producer.py
voltdbclient.py

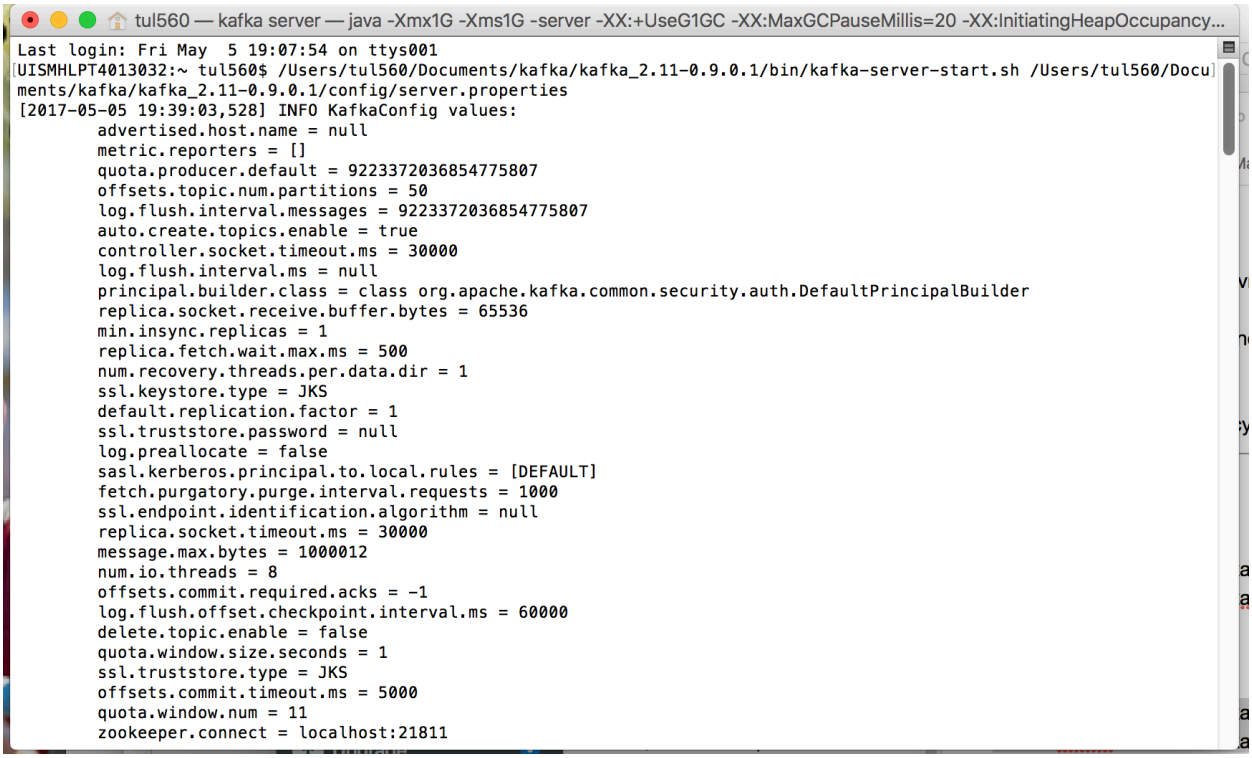
(2)Directory: /Users/tul560/Documents/PlayProjects/p2_env1
voltdbclient.py
finalproj.ipynb

(3)
#.bash_profile modification
#Add and change this path to point to your kafka lib
#for voltdb kafka loader
export ZKLIB=/Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/libs

Step(1)
Start the Zookeeper:
/Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/bin/zookeeper-server-start.sh /Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/config/zookeeper.properties

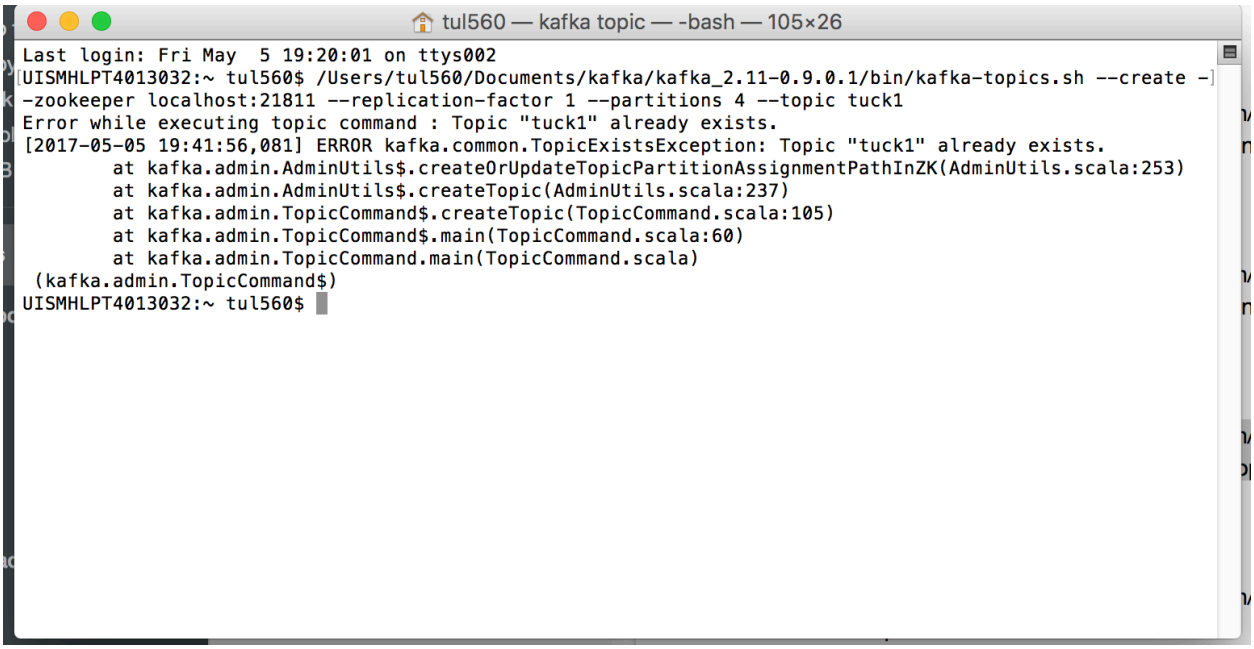


Step(2)
Start the kafka server:
/Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/bin/kafka-server-start.sh /Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/config/server.properties



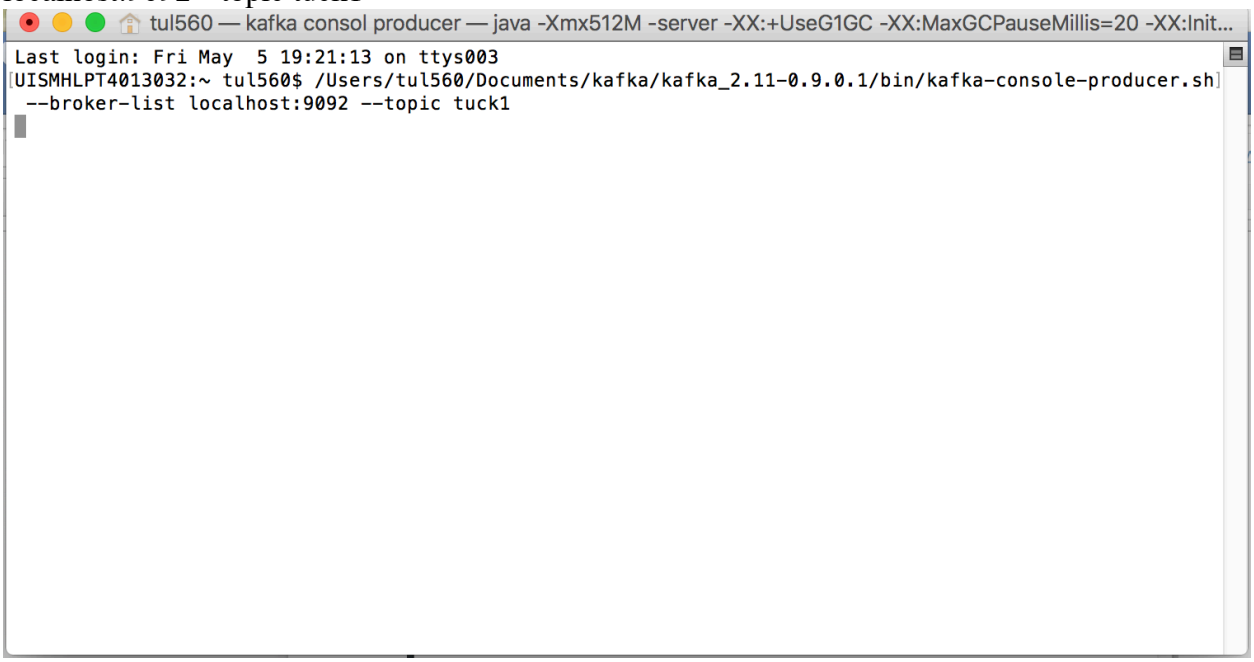
Step(3)
kafka Topic:

/Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/bin/kafka-topics.sh --create --zookeeper localhost:21811 --replication-factor 1 --partitions 4 --topic tuck1

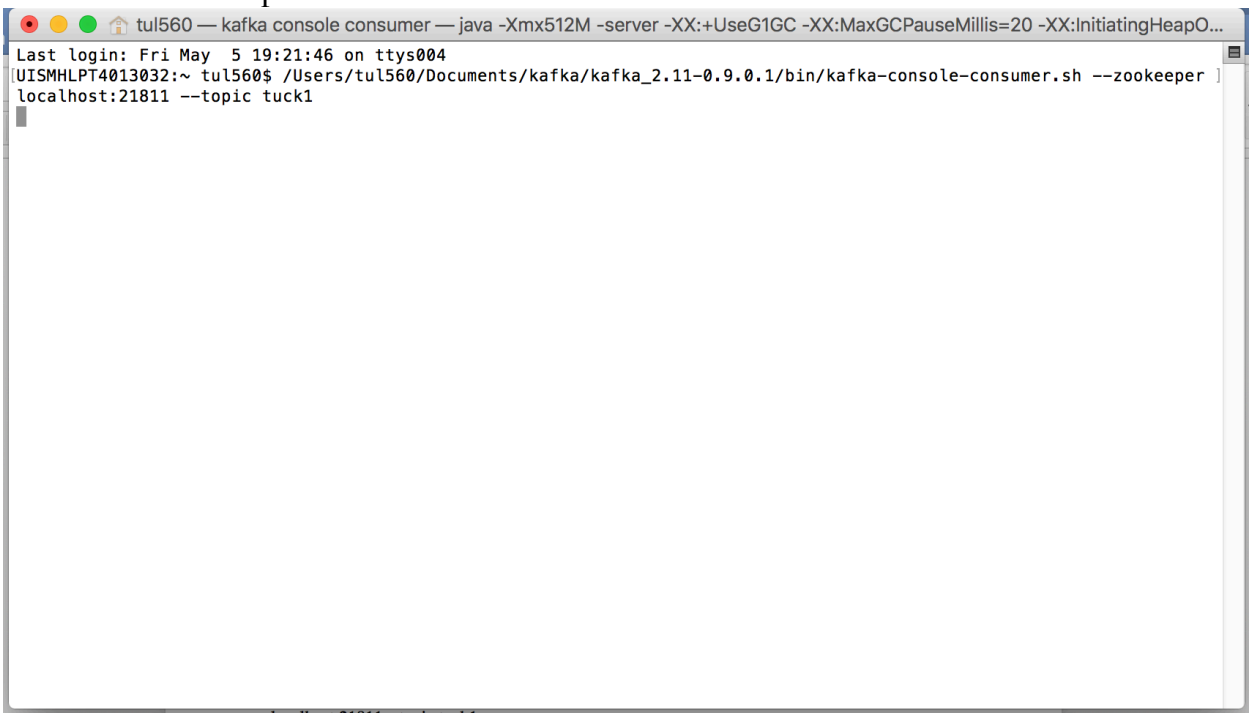


Here I already have created topic “tuck1” before. Therefore, the exception is thrown.

Step(4)
Start console producer:
/Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic tuck1



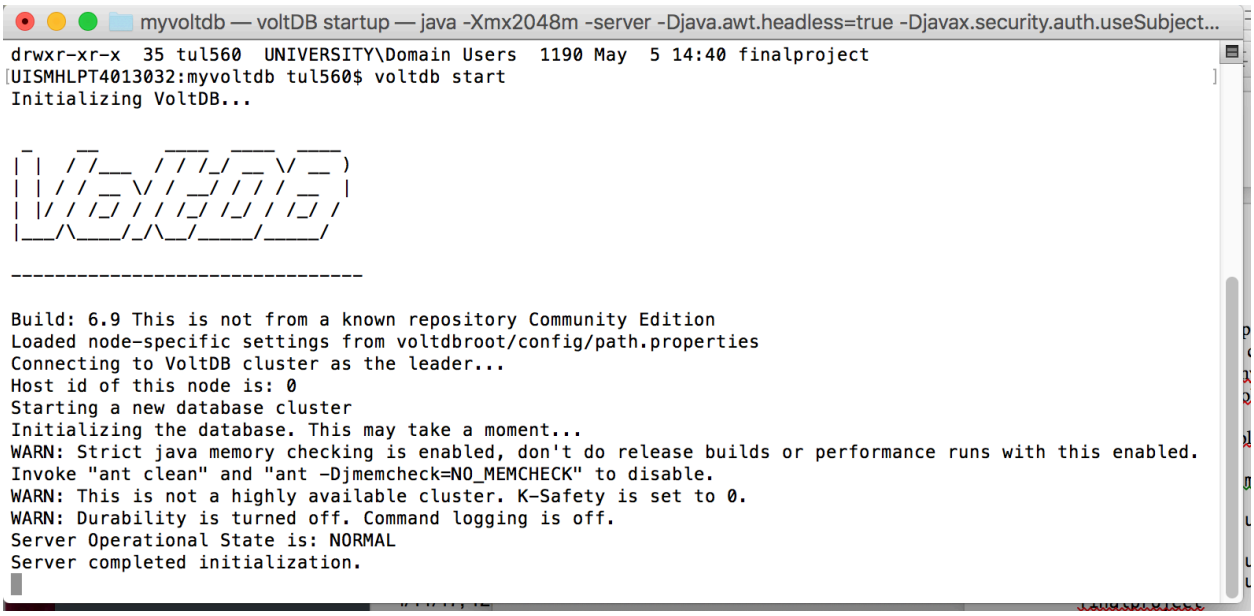
Step(5)
Start console customer:
/Users/tul560/Documents/kafka/kafka_2.11-0.9.0.1/bin/kafka-console-consumer.sh --zookeeper localhost:21811 --topic tuck1



step(6) VoltDB startup
For the first time, you can create a new folder you like to run voltdb from.
mkdir /Users/tul560/myvoltdb
cd /Users/tul560/myvoltdb/
$voltdb init
This will create the voltdbroot and log folders.

```
UISMHLPT4013032:myvoltdb tul560$ ls -ltr
total 0
drwxr-xr-x  16 tul560  UNIVERSITY\Domain Users   544 May  1 19:54
voltdbroot
drwxr-xr-x   6 tul560  UNIVERSITY\Domain Users   204 May  4 08:45 log
drwxr-xr-x  35 tul560  UNIVERSITY\Domain Users  1190 May  5 14:40
finalproject
UISMHLPT4013032:myvoltdb tul560$
```

$voltdb start
You can run this in background with –B, but I want to see this on screen.



step(7) sqlcmd
Here I want VoltDB sqlcmd client to install DDL, stored procedure and run queries.
cd /Users/tul560/myvoltdb/finalproject
$sqlcmd
> file /Users/tul560/myvoltdb/finalproject/hads_table.sql
> file /Users/tul560/myvoltdb/finalproject/hads2.sql

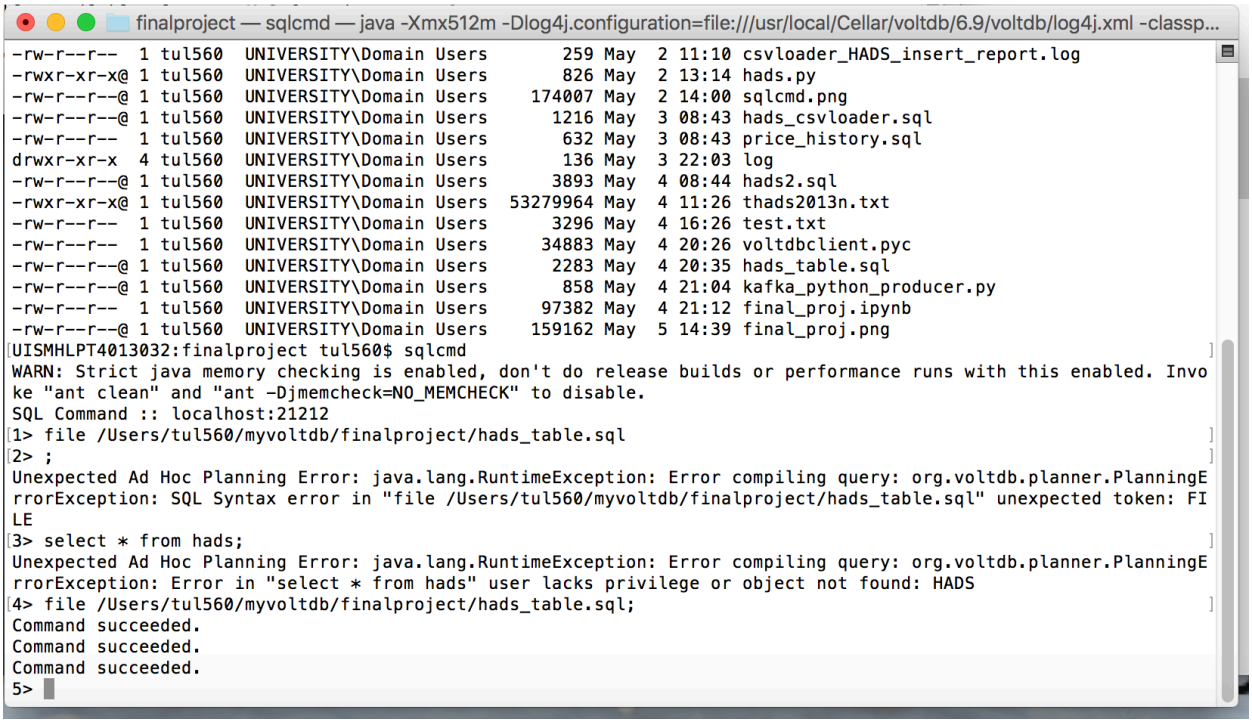
Installation for hads_table.sql

Hads_table.sql has a table and 2 stored procedures. Table HADS is based on data file format of 2013 the *Housing Affordability Data System (HADS).*

https://www.huduser.gov/portal/datasets/hads/hads.html

HADS Data derived from AHS National Data

Year	ASCII version	SAS version
2013	*.zip (11.3 MB)	*.zip (18.8 MB)



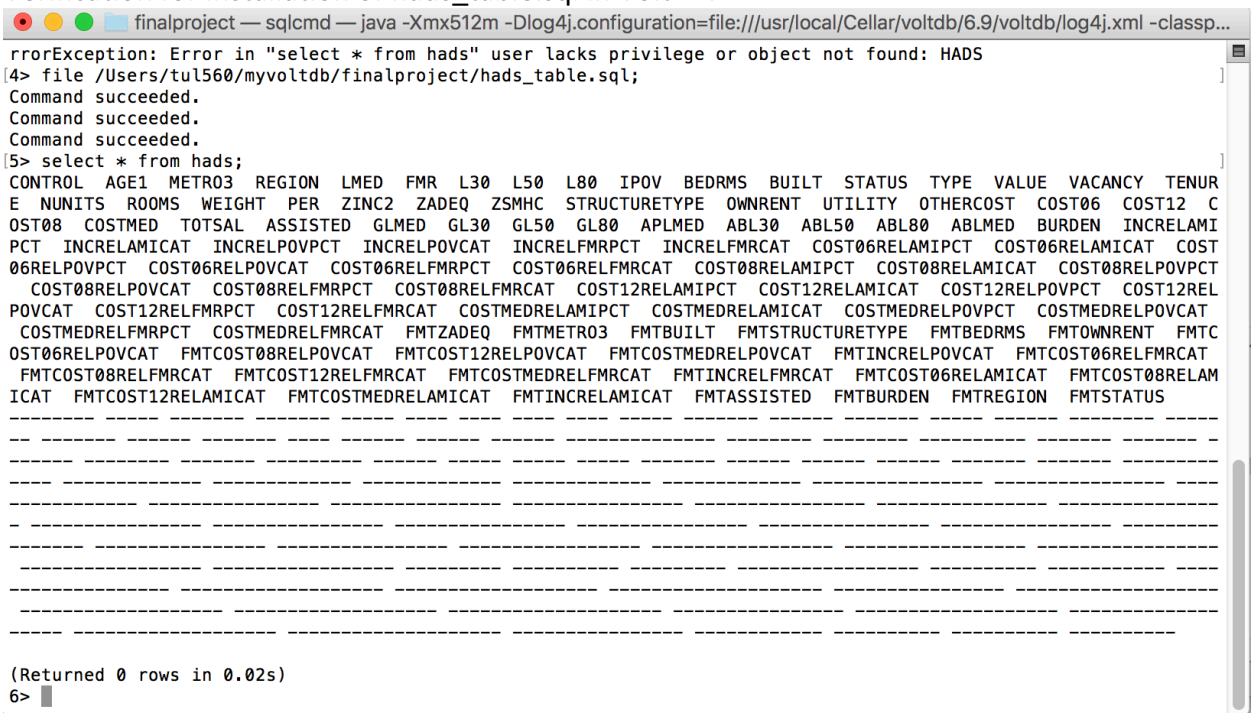
hads_table.sql

```
CREATE TABLE HADS (CONTROL VARCHAR(20),
AGE1 INTEGER,
METRO3 VARCHAR(5),
REGION VARCHAR(5),
LMED INTEGER,
FMR INTEGER,
L30 INTEGER,
L50 INTEGER,
L80 INTEGER,
IPOV INTEGER,
BEDRMS INTEGER,
BUILT INTEGER,
STATUS VARCHAR(5),
TYPE VARCHAR(5),
VALUE INTEGER,
VACANCY INTEGER,
TENURE VARCHAR(5),
NUNITS INTEGER,
ROOMS INTEGER,
WEIGHT FLOAT,
PER INTEGER,
ZINC2 INTEGER,
ZADEQ VARCHAR(5),
ZSMHC INTEGER,
STRUCTURETYPE INTEGER,
OWNRENT VARCHAR(5),
UTILITY FLOAT,
OTHERCOST FLOAT,
COST06 FLOAT,
COST12 FLOAT,
COST08 FLOAT,
COSTMED FLOAT,
TOTSAL INTEGER,
ASSISTED INTEGER,
GLMED INTEGER,
GL30 INTEGER,
GL50 INTEGER,
GL80 INTEGER,
APLMED FLOAT,
ABL30 FLOAT,
ABL50 FLOAT,
ABL80 FLOAT,
ABLMED FLOAT,
BURDEN FLOAT,
INCRELAMIPCT FLOAT,
INCRELAMICAT INTEGER,
INCRELPOVPCT FLOAT,
INCRELPOVCAT INTEGER,
INCRELFMRPCT FLOAT,
INCRELFMRCAT INTEGER,
COST06RELAMIPCT FLOAT,
COST06RELAMICAT INTEGER,
COST06RELPOVPCT FLOAT,
COST06RELPOVCAT INTEGER,
COST06RELFMRPCT FLOAT,
COST06RELFMRCAT INTEGER,
COST08RELAMIPCT FLOAT,
COST08RELAMICAT INTEGER,
```

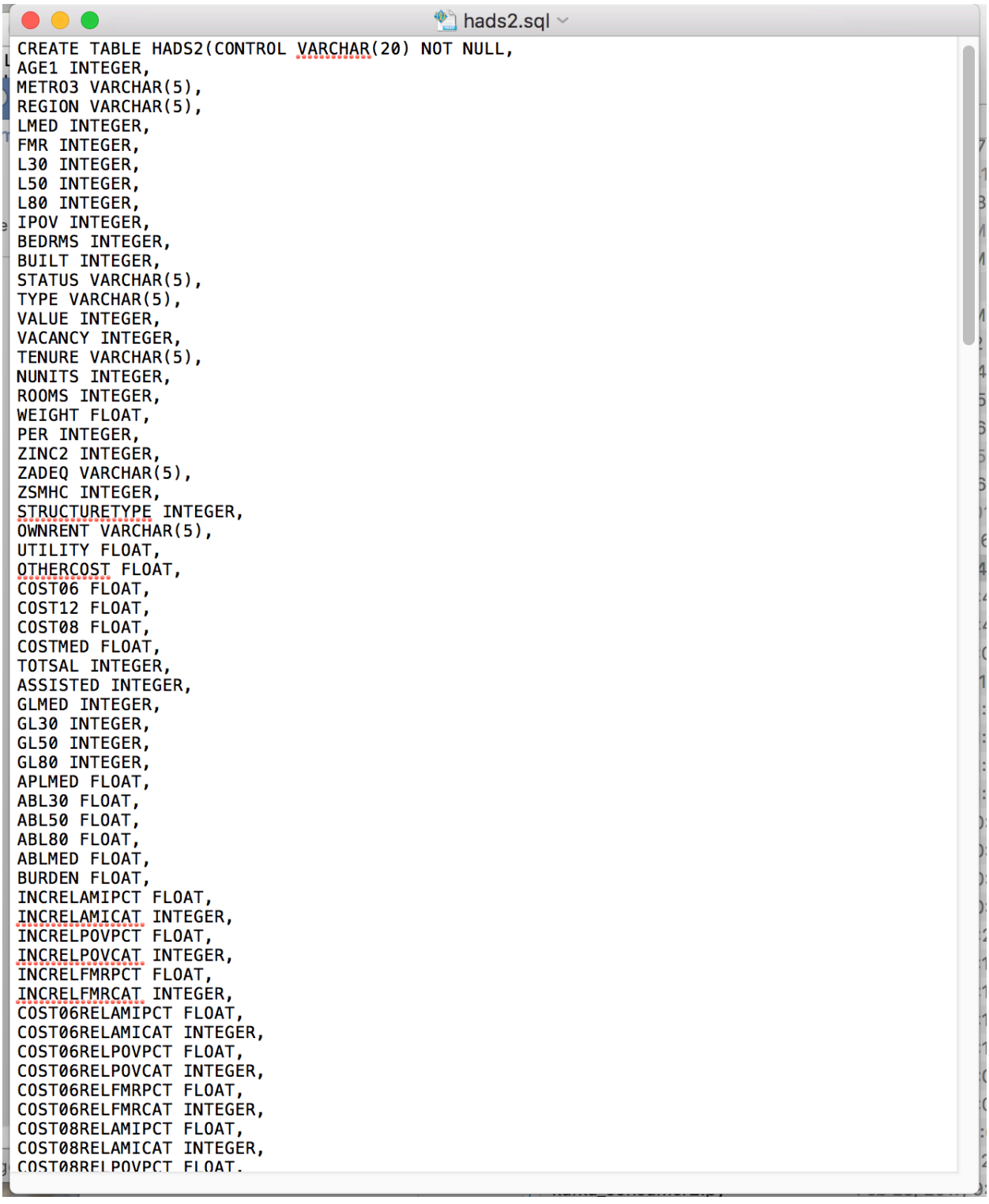
hads_table.sql

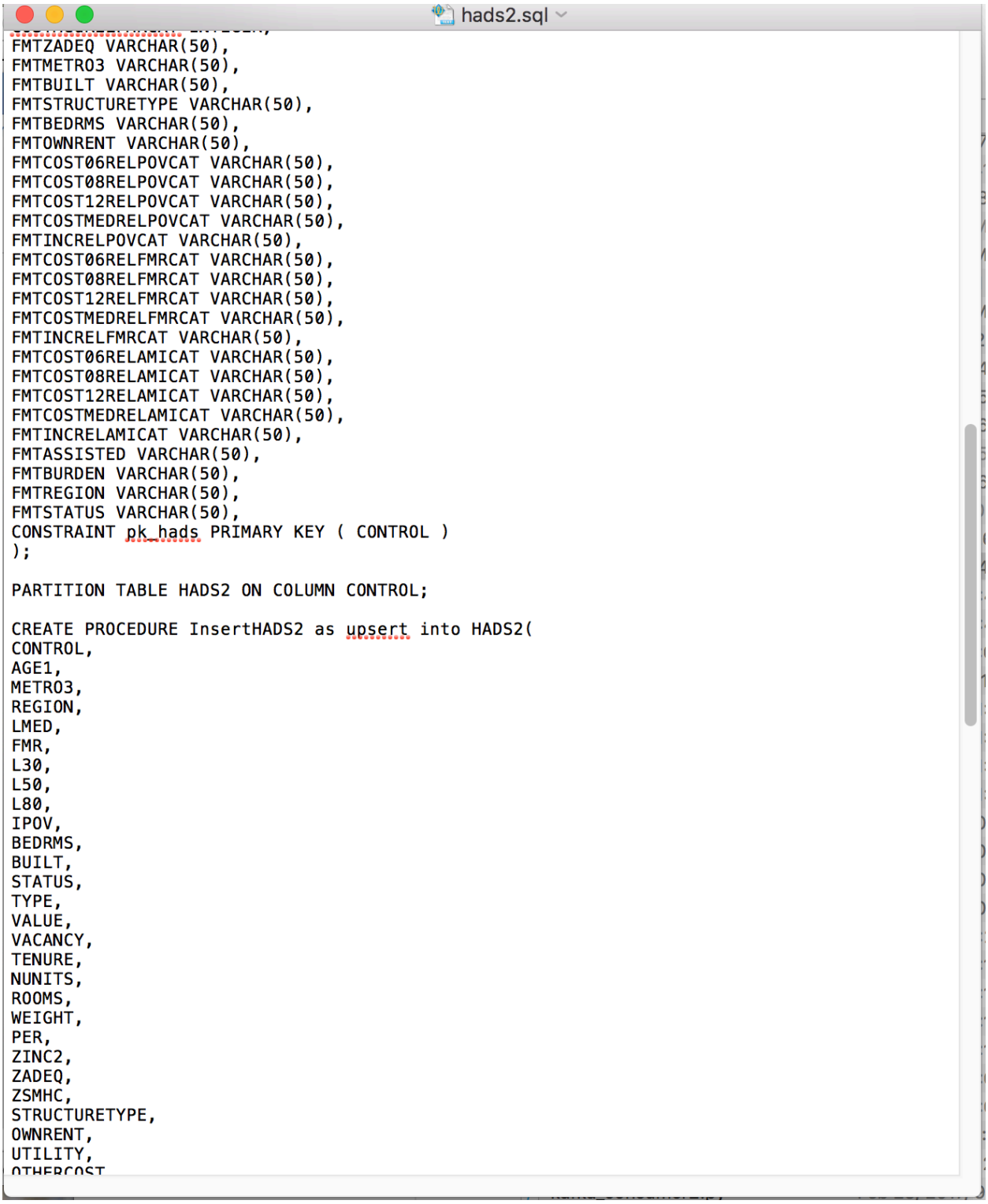
```
COST06RELAMICAT INTEGER,
COST06RELPOVPCT FLOAT,
COST06RELPOVCAT INTEGER,
COST06RELFMRPCT FLOAT,
COST06RELFMRCAT INTEGER,
COST08RELAMIPCT FLOAT,
COST08RELAMICAT INTEGER,
COST08RELPOVPCT FLOAT,
COST08RELPOVCAT INTEGER,
COST08RELFMRPCT FLOAT,
COST08RELFMRCAT INTEGER,
COST12RELAMIPCT FLOAT,
COST12RELAMICAT INTEGER,
COST12RELPOVPCT FLOAT,
COST12RELPOVCAT INTEGER,
COST12RELFMRPCT FLOAT,
COST12RELFMRCAT INTEGER,
COSTMedRELAMIPCT FLOAT,
COSTMedRELAMICAT INTEGER,
COSTMedRELPOVPCT FLOAT,
COSTMedRELPOVCAT INTEGER,
COSTMedRELFMRPCT FLOAT,
COSTMedRELFMRCAT INTEGER,
FMTZADEQ VARCHAR(50),
FMTMETRO3 VARCHAR(50),
FMTBUILT VARCHAR(50),
FMTSTRUCTURETYPE VARCHAR(50),
FMTBEDRMS VARCHAR(50),
FMTOWNRENT VARCHAR(50),
FMTCOST06RELPOVCAT VARCHAR(50),
FMTCOST08RELPOVCAT VARCHAR(50),
FMTCOST12RELPOVCAT VARCHAR(50),
FMTCOSTMEDRELPOVCAT VARCHAR(50),
FMTINCRELPOVCAT VARCHAR(50),
FMTCOST06RELFMRCAT VARCHAR(50),
FMTCOST08RELFMRCAT VARCHAR(50),
FMTCOST12RELFMRCAT VARCHAR(50),
FMTCOSTMEDRELFMRCAT VARCHAR(50),
FMTINCRELFMRCAT VARCHAR(50),
FMTCOST06RELAMICAT VARCHAR(50),
FMTCOST08RELAMICAT VARCHAR(50),
FMTCOST12RELAMICAT VARCHAR(50),
FMTCOSTMEDRELAMICAT VARCHAR(50),
FMTINCRELAMICAT VARCHAR(50),
FMTASSISTED VARCHAR(50),
FMTBURDEN VARCHAR(50),
FMTREGION VARCHAR(50),
FMTSTATUS VARCHAR(50)
);

CREATE PROCEDURE CountHADS AS
            SELECT COUNT(*) FROM HADS;

CREATE PROCEDURE GetHADSAge AS
            SELECT FMR, ABL30 FROM HADS WHERE AGE1 = ?;
```

After the installation, verity the table to make sure it was created.
Verification for installation of hads_table.sql in VoltDB.



Installation of hads2.sql. This is the same as above HADS table, but I want to demo different method of loading this table.

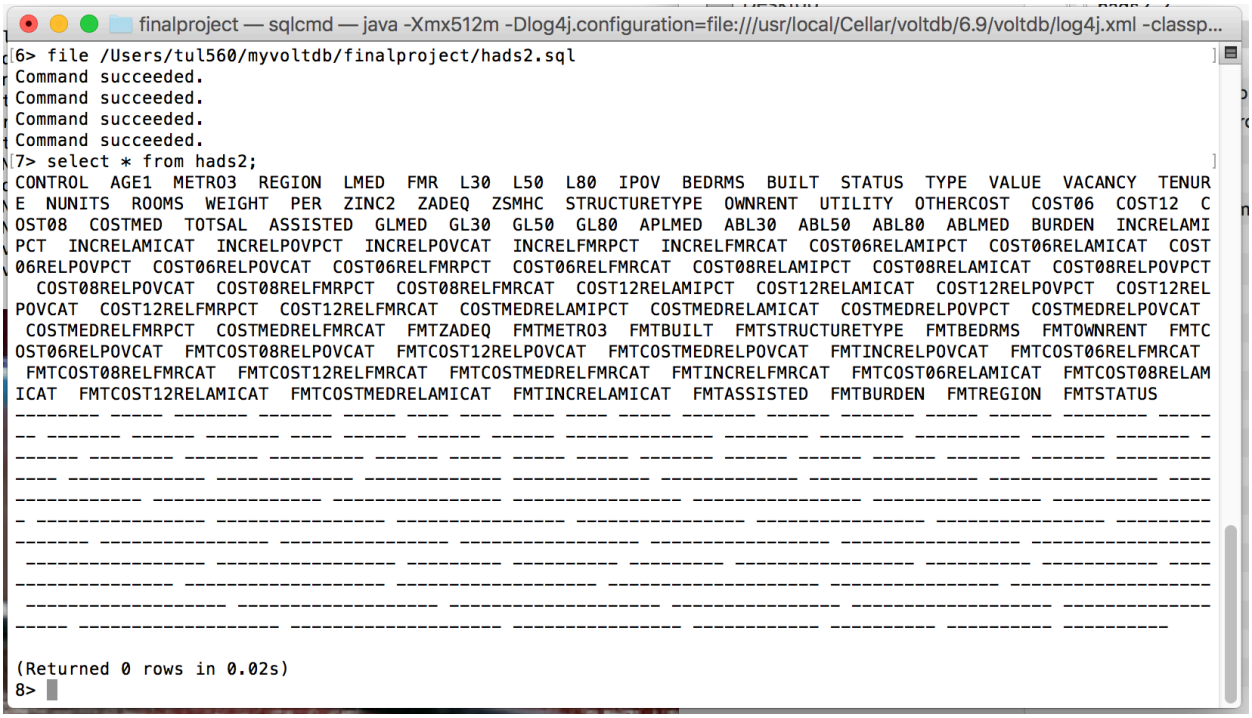

hads2.sql
CREATE TABLE HADS2(CONTROL VARCHAR(20) NOT NULL,
AGE1 INTEGER,
METRO3 VARCHAR(5),
REGION VARCHAR(5),
LMED INTEGER,
FMR INTEGER,
L30 INTEGER,
L50 INTEGER,
L80 INTEGER,
IPOV INTEGER,
BEDRMS INTEGER,
BUILT INTEGER,
STATUS VARCHAR(5),
TYPE VARCHAR(5),
VALUE INTEGER,
VACANCY INTEGER,
TENURE VARCHAR(5),
NUNITS INTEGER,
ROOMS INTEGER,
WEIGHT FLOAT,
PER INTEGER,
ZINC2 INTEGER,
ZADEQ VARCHAR(5),
ZSMHC INTEGER,
STRUCTURETYPE INTEGER,
OWNRENT VARCHAR(5),
UTILITY FLOAT,
OTHERCOST FLOAT,
COST06 FLOAT,
COST12 FLOAT,
COST08 FLOAT,
COSTMED FLOAT,
TOTSAL INTEGER,
ASSISTED INTEGER,
GLMED INTEGER,
GL30 INTEGER,
GL50 INTEGER,
GL80 INTEGER,
APLMED FLOAT,
ABL30 FLOAT,
ABL50 FLOAT,
ABL80 FLOAT,
ABLMED FLOAT,
BURDEN FLOAT,
INCRELAMIPCT FLOAT,
INCRELAMICAT INTEGER,
INCRELPOVPCT FLOAT,
INCRELPOVCAT INTEGER,
INCRELFMRPCT FLOAT,
INCRELFMRCAT INTEGER,
COST06RELAMIPCT FLOAT,
COST06RELAMICAT INTEGER,
COST06RELPOVPCT FLOAT,
COST06RELPOVCAT INTEGER,
COST06RELFMRPCT FLOAT,
COST06RELFMRCAT INTEGER,
COST08RELAMIPCT FLOAT,
COST08RELAMICAT INTEGER,
COST08RELPOVPCT FLOAT,


hads2.sql
FMTZADEQ VARCHAR(50),
FMTMETRO3 VARCHAR(50),
FMTBUILT VARCHAR(50),
FMTSTRUCTURETYPE VARCHAR(50),
FMTBEDRMS VARCHAR(50),
FMTOWNRENT VARCHAR(50),
FMTCOST06RELPOVCAT VARCHAR(50),
FMTCOST08RELPOVCAT VARCHAR(50),
FMTCOST12RELPOVCAT VARCHAR(50),
FMTCOSTMEDRELPOVCAT VARCHAR(50),
FMTINCRELPOVCAT VARCHAR(50),
FMTCOST06RELFMRCAT VARCHAR(50),
FMTCOST08RELFMRCAT VARCHAR(50),
FMTCOST12RELFMRCAT VARCHAR(50),
FMTCOSTMEDRELFMRCAT VARCHAR(50),
FMTINCRELFMRCAT VARCHAR(50),
FMTCOST06RELAMICAT VARCHAR(50),
FMTCOST08RELAMICAT VARCHAR(50),
FMTCOST12RELAMICAT VARCHAR(50),
FMTCOSTMEDRELAMICAT VARCHAR(50),
FMTINCRELAMICAT VARCHAR(50),
FMTASSISTED VARCHAR(50),
FMTBURDEN VARCHAR(50),
FMTREGION VARCHAR(50),
FMTSTATUS VARCHAR(50),
CONSTRAINT pk_hads PRIMARY KEY (CONTROL)
);
PARTITION TABLE HADS2 ON COLUMN CONTROL;
CREATE PROCEDURE InsertHADS2 as upsert into HADS2(
CONTROL,
AGE1,
METRO3,
REGION,
LMED,
FMR,
L30,
L50,
L80,
IPOV,
BEDRMS,
BUILT,
STATUS,
TYPE,
VALUE,
VACANCY,
TENURE,
NUNITS,
ROOMS,
WEIGHT,
PER,
ZINC2,
ZADEQ,
ZSMHC,
STRUCTURETYPE,
OWNRENT,
UTILITY,

hads2.sql

```
COST06RELFMRPCT,
COST06RELFMRCAT,
COST08RELAMIPCT,
COST08RELAMICAT,
COST08RELPOVPCT,
COST08RELPOVCAT,
COST08RELFMRPCT,
COST08RELFMRCAT,
COST12RELAMIPCT,
COST12RELAMICAT,
COST12RELPOVPCT,
COST12RELPOVCAT,
COST12RELFMRPCT,
COST12RELFMRCAT,
COSTMedRELAMIPCT,
COSTMedRELAMICAT,
COSTMedRELPOVPCT,
COSTMedRELPOVCAT,
COSTMedRELFMRPCT,
COSTMedRELFMRCAT,
FMTZADEQ,
FMTMETRO3,
FMTBUILT,
FMTSTRUCTURETYPE,
FMTBEDRMS,
FMTOWNRENT,
FMTCOST06RELPOVCAT,
FMTCOST08RELPOVCAT,
FMTCOST12RELPOVCAT,
FMTCOSTMEDRELPOVCAT,
FMTINCRELPOVCAT,
FMTCOST06RELFMRCAT,
FMTCOST08RELFMRCAT,
FMTCOST12RELFMRCAT,
FMTCOSTMEDRELFMRCAT,
FMTINCRELFMRCAT,
FMTCOST06RELAMICAT,
FMTCOST08RELAMICAT,
FMTCOST12RELAMICAT,
FMTCOSTMEDRELAMICAT,
FMTINCRELAMICAT,
FMTASSISTED,
FMTBURDEN,
FMTREGION,
FMTSTATUS
) VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?
);
PARTITION PROCEDURE InsertHADS2 ON TABLE HADS2 COLUMN CONTROL;
```

Verification of hads2 table.



You can execute system stored procedures to get info.

finalproject — sqlcmd — java -Xmx512m -Dlog4j.configuration=file:///usr/local/Cellar/voltdb/6.9/voltdb/log4j.xml -class...

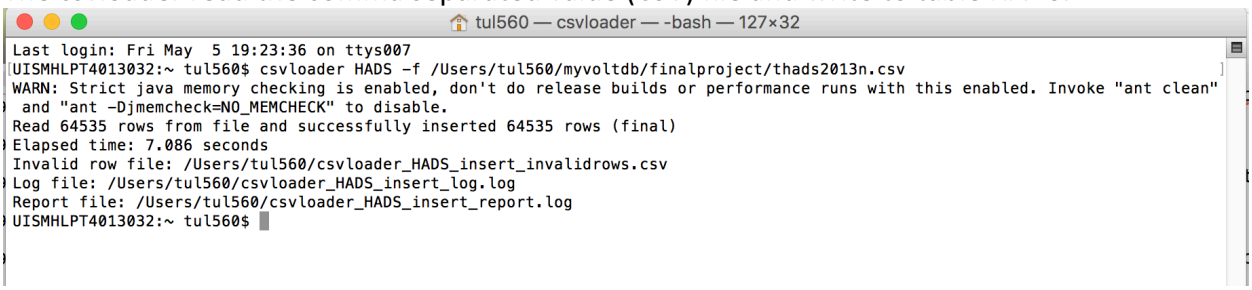
```
------------- ---------------- ---------------- ----------------- ----------------- ----------------- ----------
------- ----------------- ----------------- --------- ---------- --------- ----------------- ---------- --------
--- ------------------- ------------------- ------------------- -------------------- --------------- ----------
--------- ------------------- ------------------- -------------------- --------------- ----------------- ----
--------------- ------------------- -------------------- ---------------- ------------ ---------- ---------- ---
-------

(Returned 0 rows in 0.02s)
8> exec @SystemInformation overview;
HOST_ID  KEY                     VALUE
-------- ----------------------- -------------------------------
       0 IPADDRESS               192.168.1.157
       0 HOSTNAME                UISMHLPT4013032
       0 CLIENTINTERFACE
       0 CLIENTPORT              21212
       0 ADMININTERFACE
       0 ADMINPORT               21211
       0 HTTPINTERFACE
       0 HTTPPORT                8080
       0 INTERNALINTERFACE
       0 INTERNALPORT            3021
       0 ZKINTERFACE             127.0.0.1
       0 ZKPORT                  7181
       0 DRINTERFACE
       0 DRPORT                  0
       0 PUBLICINTERFACE
       0 BUILDSTRING             6.9_This is not from a known repository
       0 VERSION                 6.9
       0 CATALOG                 NULL
       0 DEPLOYMENT              /Users/tul560/myvoltdb/voltdbroot/config/deployment.xml
       0 CLUSTERSTATE            RUNNING
       0 INITIALIZED             true
       0 REPLICATIONROLE         NONE
       0 LASTCATALOGUPDATETXNID  3619631923331071
       0 CATALOGCRC              2805389849
       0 IV2ENABLED              true
       0 STARTTIME               1494028916356
       0 UPTIME                  0 days 00:46:39.697
       0 LOG4JPORT               0
       0 PLACEMENTGROUP          0
       0 PARTITIONGROUP          0

(Returned 30 rows in 0.02s)
9>
```

```
9> exec @SystemCatalog procedures;
PROCEDURE_CAT  PROCEDURE_SCHEM  PROCEDURE_NAME  RESERVED1  RESERVED2  RESERVED3  REMARKS                         PROCEDURE_TYPE  SPECIFIC_NAME
-------------- ---------------- --------------- ---------- ---------- ---------- ------------------------------- --------------- --------------
NULL           NULL             CountHADS       NULL       NULL       NULL       {"readOnly":true,"singlePartition":false}             0 CountHADS

NULL           NULL             GetHADSAge      NULL       NULL       NULL       {"readOnly":true,"singlePartition":false}             0 GetHADSAge

NULL           NULL             HADS.insert     NULL       NULL       NULL       {"readOnly":false,"singlePartition":false}             0 HADS.insert

NULL           NULL             HADS2.delete    NULL       NULL       NULL       {"partitionParameter":0,"partitionParameterType":9,"readOnly":false,"s
inglePartition":true}               0 HADS2.delete
NULL           NULL             HADS2.insert    NULL       NULL       NULL       {"partitionParameter":0,"partitionParameterType":9,"readOnly":false,"s
inglePartition":true}               0 HADS2.insert
NULL           NULL             HADS2.select    NULL       NULL       NULL       {"partitionParameter":0,"partitionParameterType":9,"readOnly":true,"si
nglePartition":true}               0 HADS2.select
NULL           NULL             HADS2.update    NULL       NULL       NULL       {"partitionParameter":99,"partitionParameterType":9,"readOnly":false,"
singlePartition":true}               0 HADS2.update
NULL           NULL             HADS2.upsert    NULL       NULL       NULL       {"partitionParameter":0,"partitionParameterType":9,"readOnly":false,"s
inglePartition":true}               0 HADS2.upsert
NULL           NULL             InsertHADS2     NULL       NULL       NULL       {"partitionParameter":0,"partitionParameterType":9,"readOnly":false,"s
inglePartition":true}               0 InsertHADS2

(Returned 9 rows in 0.01s)
10>
```

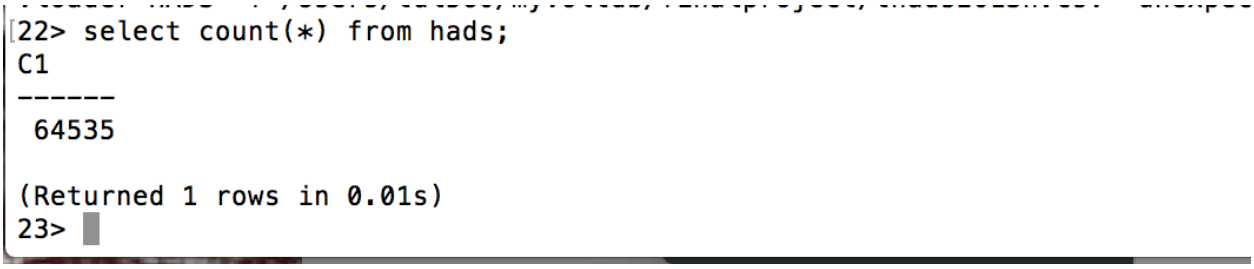
```
17> exec @Statistics TABLE, 0;
TIMESTAMP      HOST_ID  HOSTNAME         SITE_ID  PARTITION_ID  TABLE_NAME  TABLE_TYPE       TUPLE_COUNT  TUPLE_ALLOCATED_MEMORY  TUPLE_DATA_MEMORY  ST
RING_DATA_MEMORY  TUPLE_LIMIT  PERCENT_FULL
--------------- -------- --------------- -------- ------------ ----------- ---------------- ------------ ----------------------- ------------------ --
----------------- ------------ -------------
 1494031995801        0 UISMHLPT4013032         0            0 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         0            0 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         1            1 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         1            1 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         2            2 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         2            2 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         3            3 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         3            3 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         4            4 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         4            4 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         5            5 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031995801        0 UISMHLPT4013032         5            5 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
 1494031996022        0 UISMHLPT4013032         6            6 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031996022        0 UISMHLPT4013032         6            6 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
 1494031996022        0 UISMHLPT4013032         7            7 HADS        PersistentTable             0                    2048                  0
                0         NULL             0
 1494031996022        0 UISMHLPT4013032         7            7 HADS2       PersistentTable             0                    2048                  0
                0         NULL             0
```

Step(8) csvload
$csvloader HADS -f /Users/tul560/myvoltdb/finalproject/thads2013n.csv

We are loading the csv file as a onetime load to hads table that we created.
The csvloader read the comma separated value (CSV) file and write to table HADS.



Verify successful onetime data loading with csvloader using sqlcmd.

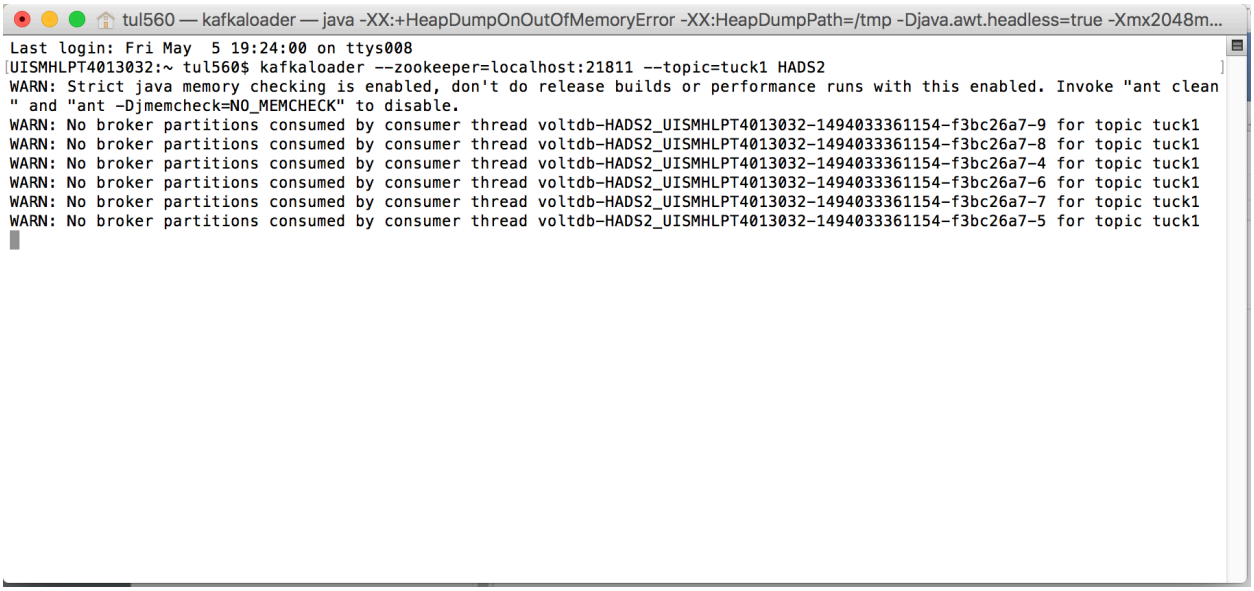


Run the row count stored procedure to verify.

```
[24> exec CountHADS;
C1
------
 64535

(Returned 1 rows in 0.01s)
25>
```

step(9) kafka voltdb loader
kafkaloader --zookeeper=localhost:21811 --topic=tuck1 HADS2

Now, we will simulate kafka streaming and load the data into hads2. We need to bring it up voltdb kafkaloader app. This kafkaloader will interact with zookeeper, kafka topic “tuck1” and VoltDB table “HADS2”.



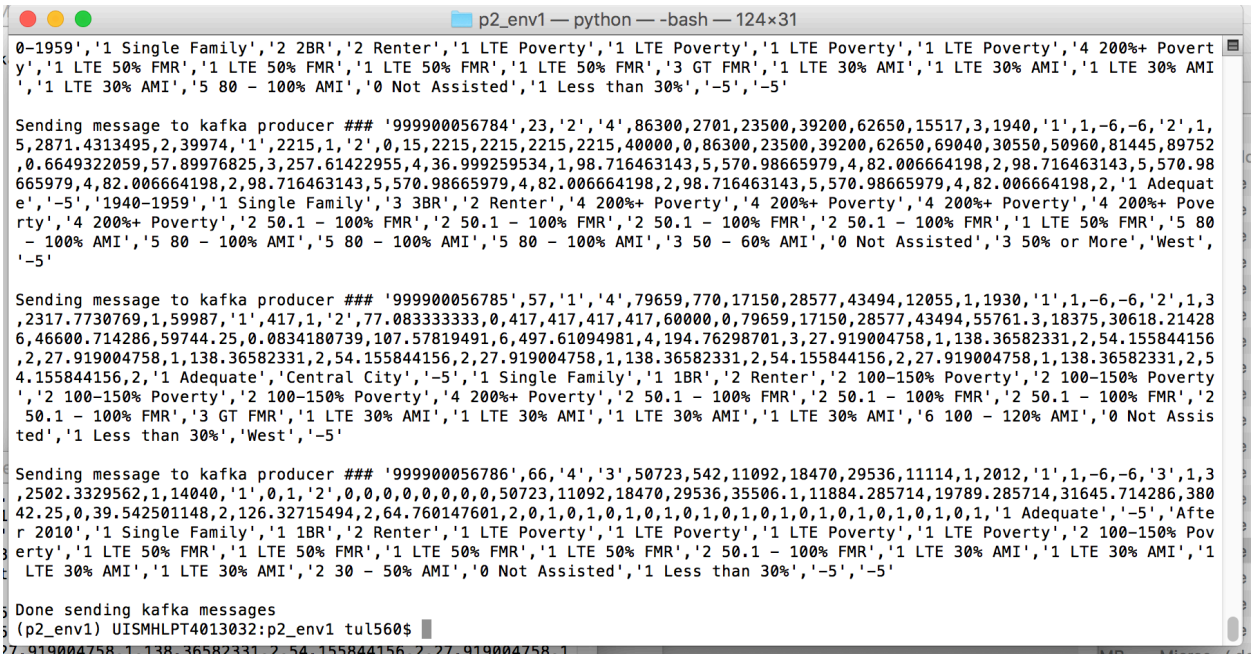
step(10) Run Simulation code
I have already setup and followed Marina class lab lecture on setting up python virtual environment for class projects. So this is an existing python virtual python 2.7 I have.

```
UISMHLPT4013032:p2_env1 tul560$ . bin/activate
(p2_env1) UISMHLPT4013032:p2_env1 tul560$ pwd
/Users/tul560/Documents/PlayProjects/p2_env1
(p2_env1) UISMHLPT4013032:p2_env1 tul560$
```

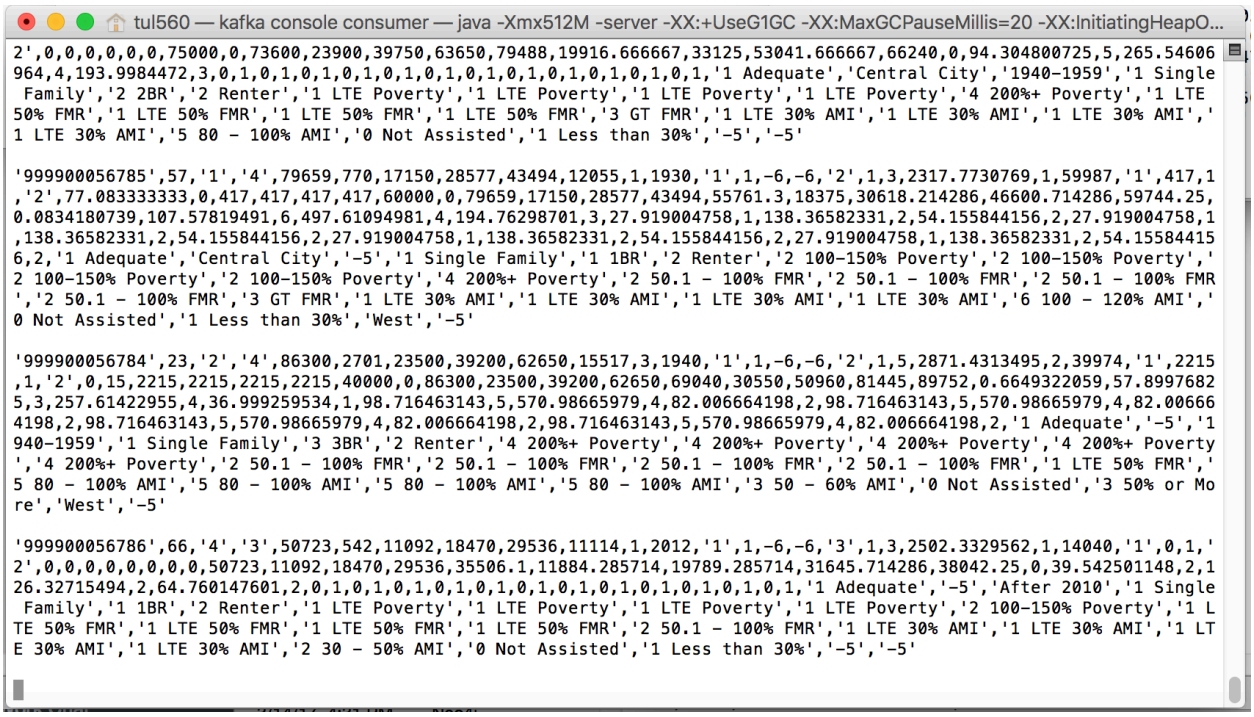
$python /Users/tul560/myvoltdb/finalproject/kafka_python_producer.py
This code will read the csv file and send messages to kafka producer.

```
-rw-r--r--@ 1 tul560  UNIVERSITY\Domain Users       858 May  5 21:18 kafka_python_producer.py
UISMHLPT4013032:finalproject tul560$ more kafka_python_producer.py
from kafka import KafkaProducer
import time

#we are going to send the messages to kafka producer, just mimicking that we can stream live data.
#when Kafka producer, we can have multiple comsumers can substribe the messages and route to many place.
#In the class lecture, we demostrated already that consumer can send to HDFS.
#Here we want to demostrate that we can send messages to VoltDB and performance analysis very fast.
#In theory, analyzed data can be stored in VoltDB and send it for HDFS. With not much of time left, we will not built this pi
ece.

f = open('/Users/tul560/myvoltdb/finalproject/thads2013n.csv', 'r')

producer = KafkaProducer(bootstrap_servers='localhost:9092')
topic = 'tuck1'

for line in f:
    print 'Sending message to kafka producer ### ' + line
    producer.send(topic, line )

f.closed
print 'Done sending kafka messages'
UISMHLPT4013032:finalproject tul560$
```

After running simulation code.



You can see kafka messages come to consumer console as well.



Now, we can check whether or not Kafkaloader is working by running select statement in VoltDB. You can see that the data has been loaded successfully.

```
[23> select count (*) from hads2;
C1
------
 64535

(Returned 1 rows in 0.01s)
24> exec
```

Step(11) jupyter notebook

I have already setup and followed Marina class lab lecture on setting up python virtual environment for class projects. So this is an existing python virtual python 2.7 I have.

```
UISMHLPT4013032:p2_env1 tul560$ . bin/activate
(p2_env1) UISMHLPT4013032:p2_env1 tul560$ pwd
/Users/tul560/Documents/PlayProjects/p2_env1
(p2_env1) UISMHLPT4013032:p2_env1 tul560$
```

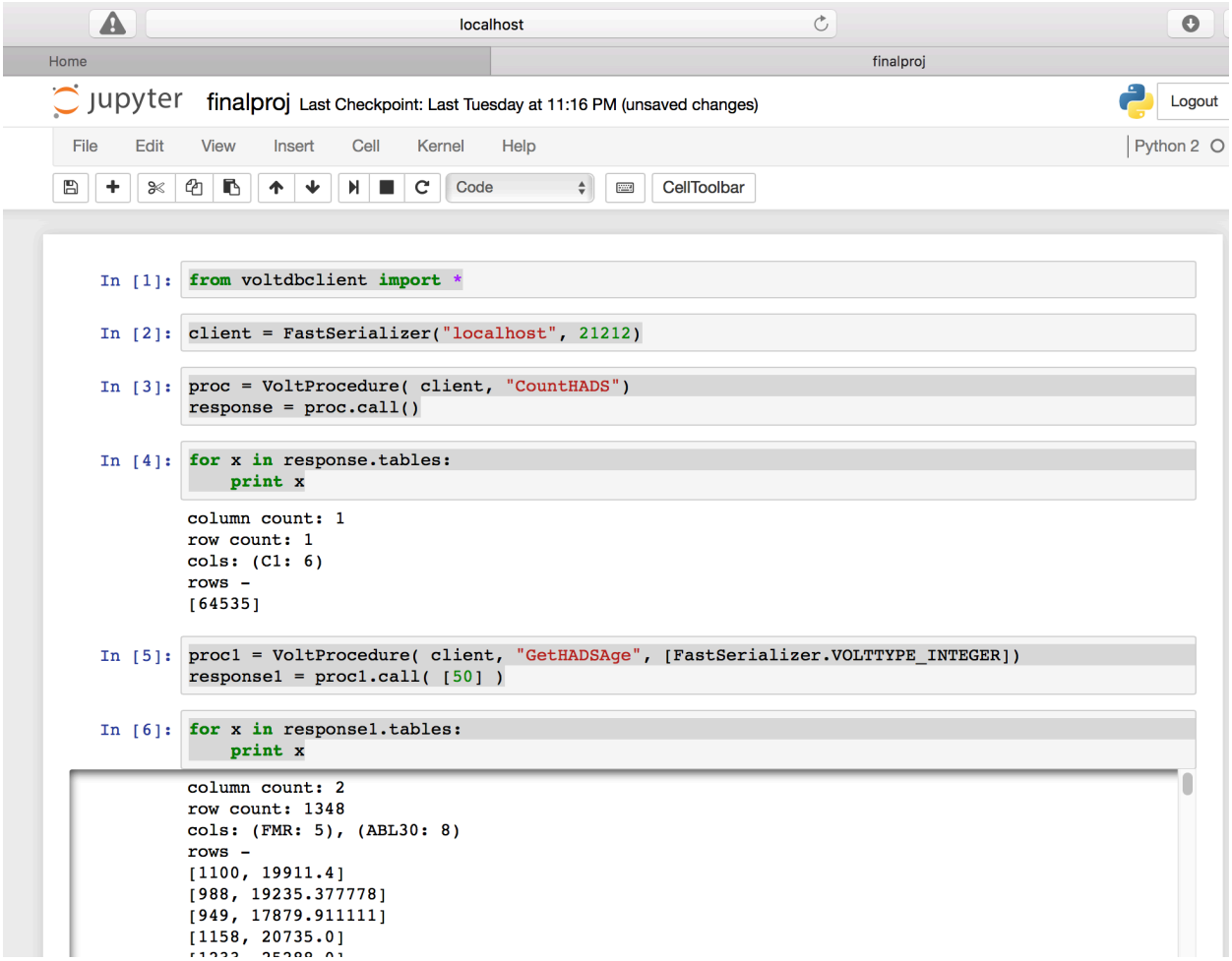
$cd `/Users/tul560/Documents/PlayProjects/p2_env1`

$jupyter notebook

Run the `/Users/tul560/Documents/PlayProjects/p2_env1/`final_proj.ipynb from jupyter notebook.

Now we want to run regression using VoltDB data.
Note that we need to place voltdbclient.py which is kind of like driver program to connect to VoltDB.

I run the finalproj.ipynb code which shows that I can connect to VoltDB database and execute the stored procedure I have. And retrieve the data and run the regression which show that low income adjusted for #bedroom vs. Fair Market Rent.


localhost
Home
finalproj
jupyter
finalproj Last Checkpoint: Last Tuesday at 11:16 PM (unsaved changes)
Logout
File Edit View Insert Cell Kernel Help
Python 2
Code
CellToolbar
In [1]:
from voltdbclient import *
In [2]:
client = FastSerializer("localhost", 21212)
In [3]:
proc = VoltProcedure(client, "CountHADS")
response = proc.call()
In [4]:
for x in response.tables:
print x
column count: 1
row count: 1
cols: (C1: 6)
rows -
[64535]
In [5]:
proc1 = VoltProcedure(client, "GetHADSAge", [FastSerializer.VOLTTYPE_INTEGER])
response1 = proc1.call([50])
In [6]:
for x in response1.tables:
print x
column count: 2
row count: 1348
cols: (FMR: 5), (ABL30: 8)
rows -
[1100, 19911.4]
[988, 19235.377778]
[949, 17879.911111]
[1158, 20735.0]

```
[1436, 25685.714286]
[819, 17497.125]
```

In [7]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [8]:
```
%matplotlib inline
```

In [9]:
```
table = response1.tables[0]
for row in table.tuples:
    d = row[0]
    e = row[1]
    print d, e
```

```
1100 19911.4
988 19235.377778
949 17879.911111
1158 20735.0
1233 25288.0
1129 23113.125
971 19760.0
956 20217.375
873 15808.0
1760 24783.407407
1528 21547.0
804 15525.0
2367 40675.555556
1577 19908.571429
1282 19347.9
646 16706.25
907 16471.0
978 17382.857143
1004 19068.4
```

jupyter finalproj Last Checkpoint: a few seconds ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Help Python 2

Code CellToolbar

In [10]:
```
d = []
table = response1.tables[0]
for row in table.tuples:
    d.append({"A":row[0],"B":row[1]})

df = pd.DataFrame(d)
```

In [11]:
```
print df.shape
```

```
(1348, 2)
```

In [14]:
```
from pylab import *
x = df['B']
y = df['A']
m,b = polyfit(x, y, 1)
plot(x, y, 'yo', x, m*x+b, '--k')
title('Regression of Income and Rent') ;
xlabel('Low Income Adjusted for #bedrooms')
ylabel('Fair Market Rent')
show()
savefig('finalProj.png', fmt='png', dpi=100)
```

After everything is done, you can shutdown the VoltDB.
$voltadmin shutdown

Since I am running community version, I cannot save.

```
UISMHLPT4013032:p2_env1 tul560$ voltadmin save
ERROR: "@SnapshotSave" procedure call failed.
ERROR:   Status: -2
ERROR:   Information: @SnapshotSave is available in the Enterprise
Edition of VoltDB only.
ERROR:
UISMHLPT4013032:p2_env1 tul560$
```

References:

Zookeeper/Kafka diagram
https://www.slideshare.net/rahuldausa/introduction-to-kafka-and-zookeeper

VoltDB:
https://docs.voltdb.com/UsingVoltDB/exportimport.php
https://docs.voltdb.com/UsingVoltDB/ChapExport.php
https://www.voltdb.com/blog/2016/10/11/connecting-voltdb-amazon-kinesis-streams/

Dataset:
https://www.huduser.gov/portal/datasets/hads/hads.html