Assignment one:

DATA CLEANING AND WRANGLING

TEAM 1

MANASI DALVI

VISHAL SATAM

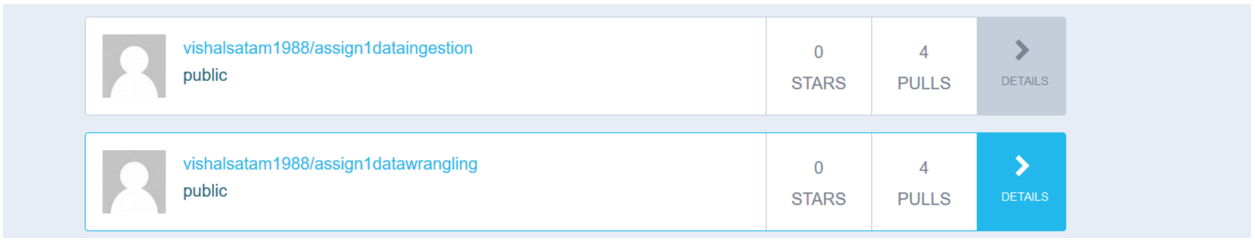
DOCKER & Execution Instructions

The required files have been packaged into 2 Docker images and are present on Docker Hub with the image names as vishalsatam1988/assign1dataingestion and vishalsatam1988/assign1datawrangling. The manual execution steps are given below.

Note : The docker container runs in the **UTC timezone**.

If you encounter any Memory Error issues. Please increase the RAM of your docker virtual machine to minimum 4 GB.

Docker-hub image

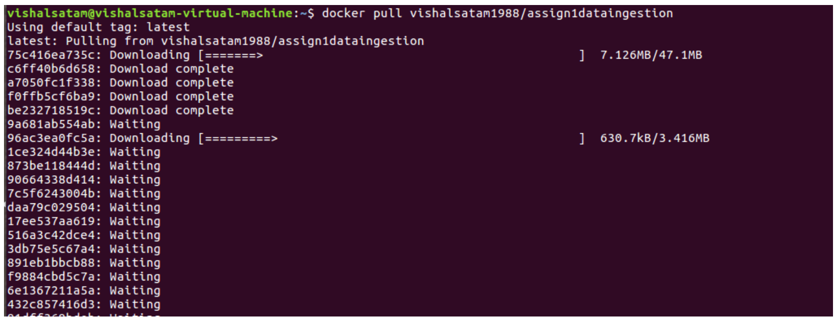

vishalsatam1988/assign1dataingestion public	0 STARS	4 PULLS	DETAILS
vishalsatam1988/assign1datawrangling public	0 STARS	4 PULLS	DETAILS

Data Ingestion

Step 1 : Pull the image

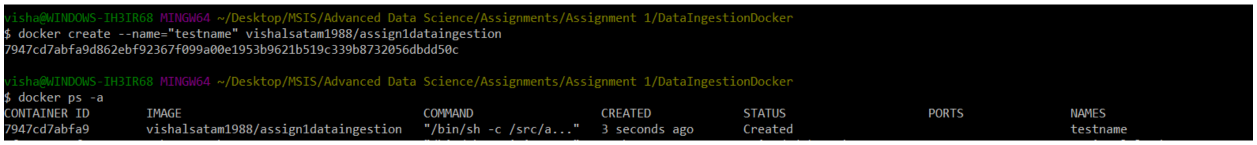
The docker image is present on the docker hub and is available to pull using the following command

docker pull vishalsatam1988/assign1dataingestion


```
vishalsatam@vishalsatam-virtual-machine:~$ docker pull vishalsatam1988/assign1dataingestion
Using default tag: latest
latest: Pulling from vishalsatam1988/assign1dataingestion
75c416ea735c: Downloading [=======>                                           ]  7.126MB/47.1MB
c6ff40b6d658: Download complete
a7050fc1f338: Download complete
f0ffb5cf6ba9: Download complete
be232718519c: Download complete
9a681ab554ab: Waiting
96ac3ea0fc5a: Downloading [=========>                                         ]  630.7kB/3.416MB
1ce324d44b3e: Waiting
873be118444d: Waiting
90664338d414: Waiting
7c5f6243004b: Waiting
daa79c029504: Waiting
17ee537aa619: Waiting
516a3c42dce4: Waiting
3db75e5c67a4: Waiting
891eb1bbcb88: Waiting
f9884cbd5c7a: Waiting
6e1367211a5a: Waiting
432c857416d3: Waiting
```

Step 2 : Create the container using

docker create --name="testname" vishalsatam1988/assign1dataingestion

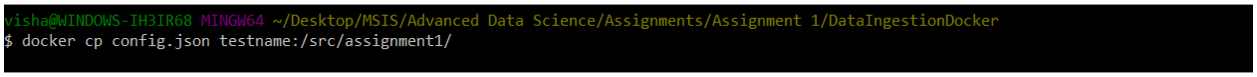


```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker create --name="testname" vishalsatam1988/assign1dataingestion
7947cd7abfa9d862ebf92367f099a00e1953b9621b519c339b8732056dbdd50c

visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker ps -a
CONTAINER ID        IMAGE                                  COMMAND                  CREATED             STATUS                    PORTS               NAMES
7947cd7abfa9        vishalsatam1988/assign1dataingestion   "/bin/sh -c /src/a..."   3 seconds ago       Created                                       testname
```

Step 3 : copy your config.json and the configIntial.json file to update the links and your AWS credentials. Please do not change the name of these files.

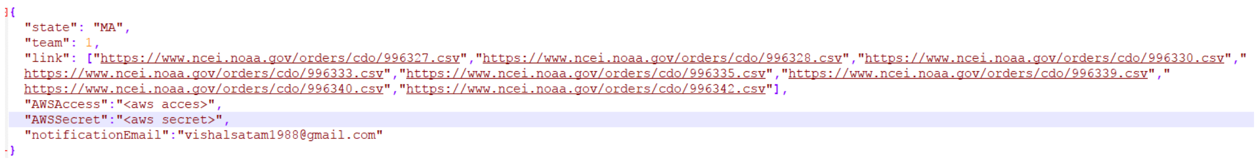
docker cp <local file path> <containername>:/src/assignment1/

docker cp config.json testname:/src/assignment1/



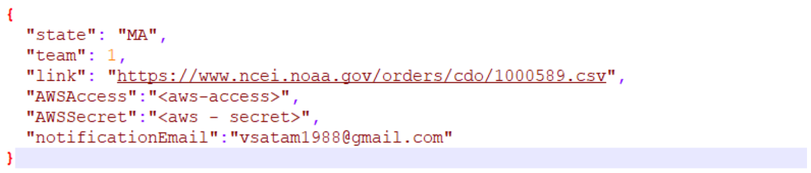
```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker cp config.json testname:/src/assignment1/
```

Sample configInitial.json



```
{
  "state": "MA",
  "team": 1,
  "link": ["https://www.ncei.noaa.gov/orders/cdo/996327.csv","https://www.ncei.noaa.gov/orders/cdo/996328.csv","https://www.ncei.noaa.gov/orders/cdo/996330.csv","
  https://www.ncei.noaa.gov/orders/cdo/996333.csv","https://www.ncei.noaa.gov/orders/cdo/996335.csv","https://www.ncei.noaa.gov/orders/cdo/996339.csv","
  https://www.ncei.noaa.gov/orders/cdo/996340.csv","https://www.ncei.noaa.gov/orders/cdo/996342.csv"],
  "AWSAccess":"<aws acces>",
  "AWSSecret":"<aws secret>",
  "notificationEmail":"vishalsatam1988@gmail.com"
}
```

Sample config.json

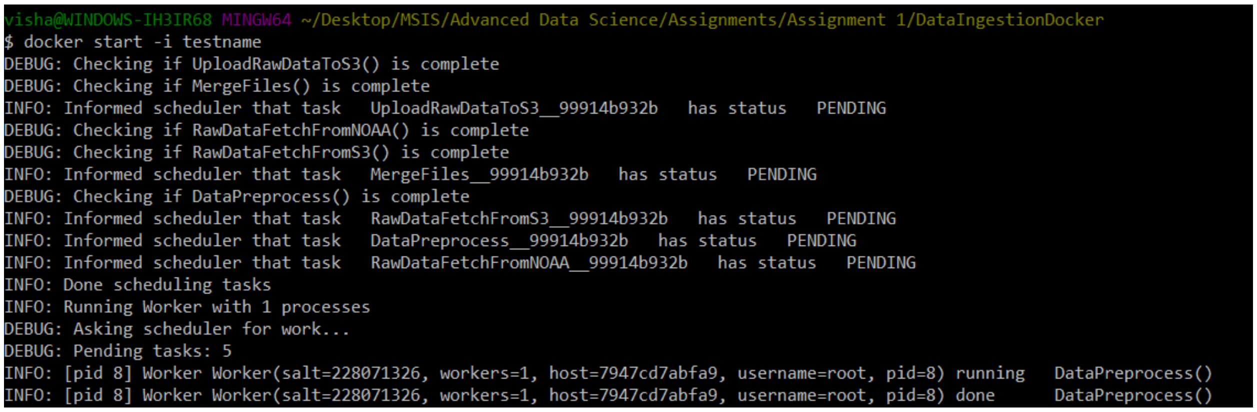


```
{
  "state": "MA",
  "team": 1,
  "link": "https://www.ncei.noaa.gov/orders/cdo/1000589.csv",
  "AWSAccess":"<aws-access>",
  "AWSSecret":"<aws - secret>",
  "notificationEmail":"vsatam1988@gmail.com"
}
```

Step 4 : Start the container

docker start <containername>

docker start -i testname

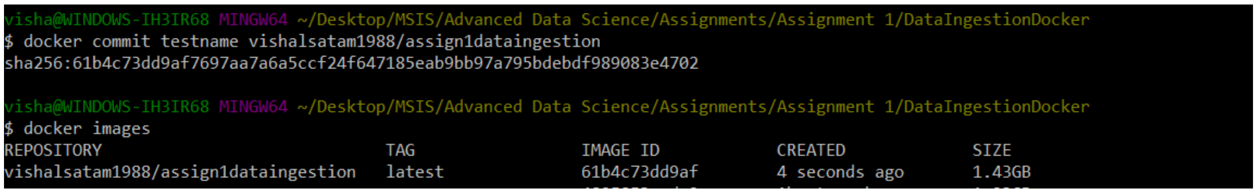


```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker start -i testname
DEBUG: Checking if UploadRawDataToS3() is complete
DEBUG: Checking if MergeFiles() is complete
INFO: Informed scheduler that task   UploadRawDataToS3__99914b932b   has status   PENDING
DEBUG: Checking if RawDataFetchFromNOAA() is complete
DEBUG: Checking if RawDataFetchFromS3() is complete
INFO: Informed scheduler that task   MergeFiles__99914b932b   has status   PENDING
DEBUG: Checking if DataPreprocess() is complete
INFO: Informed scheduler that task   RawDataFetchFromS3__99914b932b   has status   PENDING
INFO: Informed scheduler that task   DataPreprocess__99914b932b   has status   PENDING
INFO: Informed scheduler that task   RawDataFetchFromNOAA__99914b932b   has status   PENDING
INFO: Done scheduling tasks
INFO: Running Worker with 1 processes
DEBUG: Asking scheduler for work...
DEBUG: Pending tasks: 5
INFO: [pid 8] Worker Worker(salt=228071326, workers=1, host=7947cd7abfa9, username=root, pid=8) running   DataPreprocess()
INFO: [pid 8] Worker Worker(salt=228071326, workers=1, host=7947cd7abfa9, username=root, pid=8) done      DataPreprocess()
```

Step 5 : Commit the container to persist the changes

docker commit <containername> <new image name>

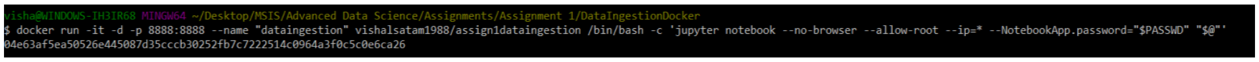
docker commit testname vishalsatam1988/assign1dataingestion



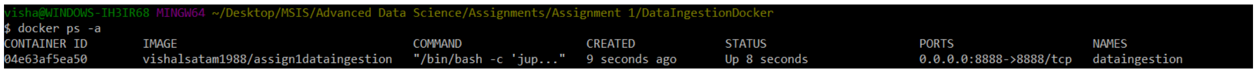
```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker commit testname vishalsatam1988/assign1dataingestion
sha256:61b4c73dd9af7697aa7a6a5ccf24f647185eab9bb97a795bdebdf989083e4702

visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker images
REPOSITORY                             TAG                 IMAGE ID            CREATED             SIZE
vishalsatam1988/assign1dataingestion   latest              61b4c73dd9af        4 seconds ago       1.43GB
```

Step 6 : Run the jupyter notebook on the committed image in detached mode

docker run -it -d --name “dataingestion” -p 8888:8888 vishalsatam1988/assign1dataingestion /bin/bash -c 'jupyter notebook --no-browser --allow-root --ip=* --NotebookApp.password="$PASSWD" "$@"'



```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker run -it -d -p 8888:8888 --name "dataingestion" vishalsatam1988/assign1dataingestion /bin/bash -c 'jupyter notebook --no-browser --allow-root --ip=* --NotebookApp.password="$PASSWD" "$@"'
04e63af5ea50526e445087d35cccb30252fb7c7222514c0964a3f0c5c0e6ca26
```

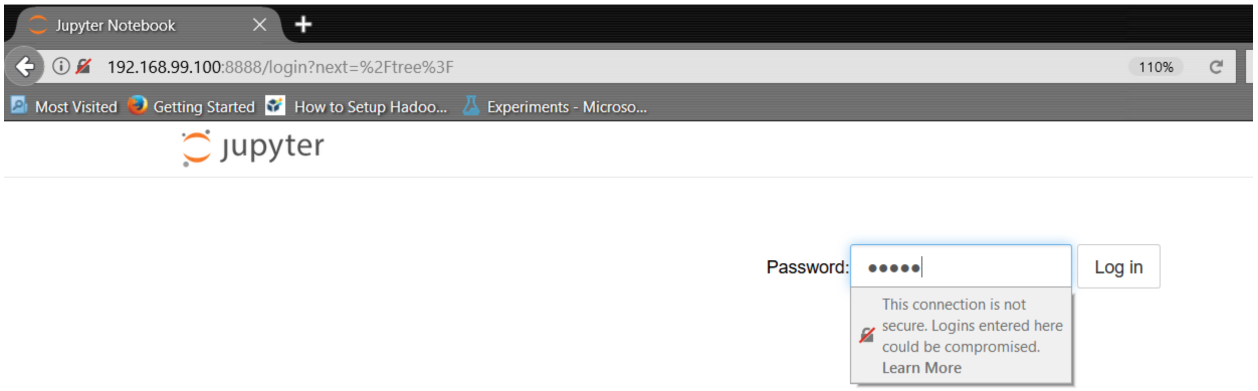


```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker ps -a
CONTAINER ID        IMAGE                                  COMMAND                  CREATED             STATUS              PORTS                    NAMES
04e63af5ea50        vishalsatam1988/assign1dataingestion   "/bin/bash -c 'jup..."   9 seconds ago       Up 8 seconds        0.0.0.0:8888->8888/tcp   dataingestion
```

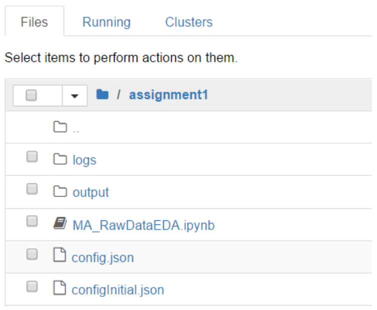
Step 7 : Connect to this running container via browser by entering

http://<docker machine ip address>:8888

Password for the jupyter notebook is keras

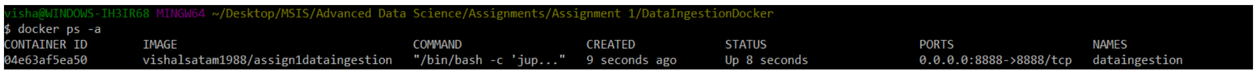

Jupyter Notebook
192.168.99.100:8888/login?next=%2Ftree%3F
110%
Most Visited
Getting Started
How to Setup Hadoo...
Experiments - Microso...
jupyter
Password:
Log in
This connection is not secure. Logins entered here could be compromised. Learn More

Navigate to the assignment1 folder and open the MA_RawDataEDA ipython notebook to see the analysis

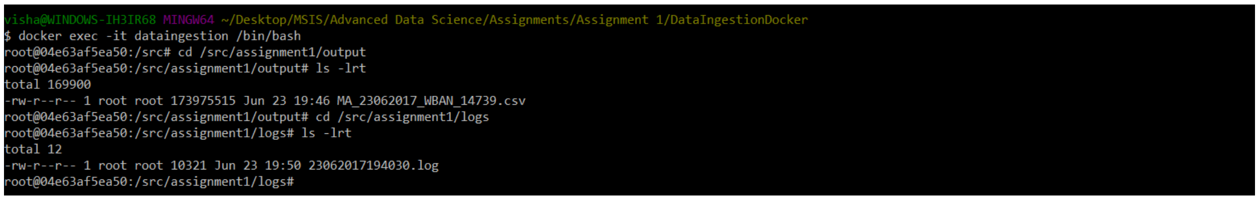


Step 8 : Execute the bin/bash command to enter the running container to check output and logs

docker exec -it dataingestion /bin/bash



```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker ps -a
CONTAINER ID        IMAGE                                COMMAND                  CREATED             STATUS                        PORTS                    NAMES
04e63af5ea50        vishalsatam1988/assign1dataingestion   "/bin/bash -c 'jup..."   9 seconds ago       Up 8 seconds                  0.0.0.0:8888->8888/tcp   dataingestion
```



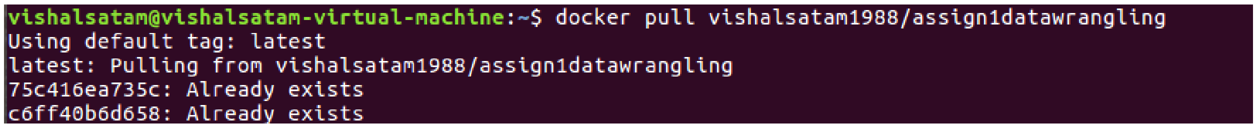
```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataIngestionDocker
$ docker exec -it dataingestion /bin/bash
root@04e63af5ea50:/src# cd /src/assignment1/output
root@04e63af5ea50:/src/assignment1/output# ls -lrt
total 169900
-rw-r--r-- 1 root root 173975515 Jun 23 19:46 MA_23062017_WBAN_14739.csv
root@04e63af5ea50:/src/assignment1/output# cd /src/assignment1/logs
root@04e63af5ea50:/src/assignment1/logs# ls -lrt
total 12
-rw-r--r-- 1 root root 10321 Jun 23 19:50 23062017194030.log
root@04e63af5ea50:/src/assignment1/logs#
```

Data Wrangling

Step 1 : Pull the image

The docker image is present on the docker hub and is available to pull using the following command

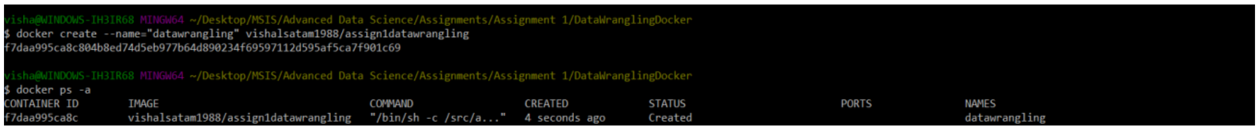
docker pull vishalsatam1988/assign1datawrangling



```
vishalsatam@vishalsatam-virtual-machine:~$ docker pull vishalsatam1988/assign1datawrangling
Using default tag: latest
latest: Pulling from vishalsatam1988/assign1datawrangling
75c416ea735c: Already exists
c6ff40b6d658: Already exists
```

Step 2 : Create the container using the below command

docker create --name="datawrangling" vishalsatam1988/assign1datawrangling

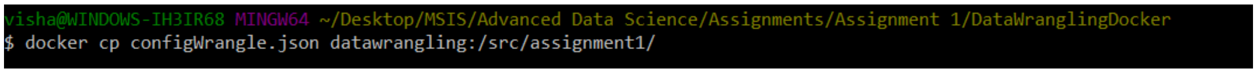


```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker create --name="datawrangling" vishalsatam1988/assign1datawrangling
f7daa995ca8c804b8ed74d5eb977b64d890234f69597112d595af5ca7f901c69

visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker ps -a
CONTAINER ID        IMAGE                                  COMMAND                  CREATED             STATUS                          PORTS               NAMES
f7daa995ca8c        vishalsatam1988/assign1datawrangling   "/bin/sh -c /src/a..."   4 seconds ago       Created                                             datawrangling
```

Step 3 : copy your configWrangle.json file to update the link. Please do not change the name of the config file.

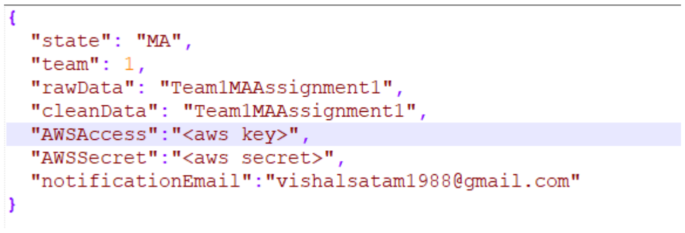
docker cp <local file path> <containername>:/src/assignment1/

docker cp configWrangle.json datawrangling:/src/assignment1/



```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker cp configWrangle.json datawrangling:/src/assignment1/
```

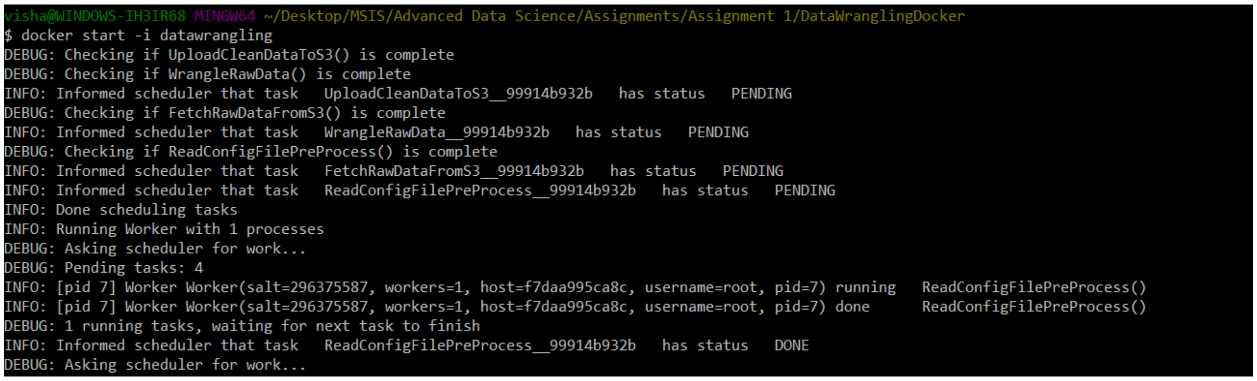
Sample configWrangle.json file


```
{
  "state": "MA",
  "team": 1,
  "rawData": "Team1MAAssignment1",
  "cleanData": "Team1MAAssignment1",
  "AWSAccess":"<aws key>",
  "AWSSecret":"<aws secret>",
  "notificationEmail":"vishalsatam1988@gmail.com"
}
```

Step 4 : Start the container

docker start <containername>

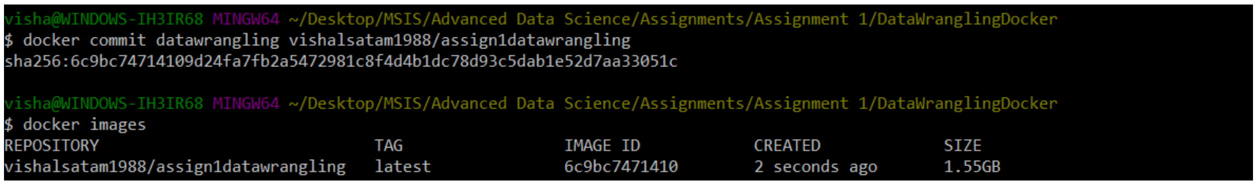
docker start -i datawrangling


```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker start -i datawrangling
DEBUG: Checking if UploadCleanDataToS3() is complete
DEBUG: Checking if WrangleRawData() is complete
INFO: Informed scheduler that task   UploadCleanDataToS3__99914b932b   has status   PENDING
DEBUG: Checking if FetchRawDataFromS3() is complete
INFO: Informed scheduler that task   WrangleRawData__99914b932b   has status   PENDING
DEBUG: Checking if ReadConfigFilePreProcess() is complete
INFO: Informed scheduler that task   FetchRawDataFromS3__99914b932b   has status   PENDING
INFO: Informed scheduler that task   ReadConfigFilePreProcess__99914b932b   has status   PENDING
INFO: Done scheduling tasks
INFO: Running Worker with 1 processes
DEBUG: Asking scheduler for work...
DEBUG: Pending tasks: 4
INFO: [pid 7] Worker Worker(salt=296375587, workers=1, host=f7daa995ca8c, username=root, pid=7) running   ReadConfigFilePreProcess()
INFO: [pid 7] Worker Worker(salt=296375587, workers=1, host=f7daa995ca8c, username=root, pid=7) done      ReadConfigFilePreProcess()
DEBUG: 1 running tasks, waiting for next task to finish
INFO: Informed scheduler that task   ReadConfigFilePreProcess__99914b932b   has status   DONE
DEBUG: Asking scheduler for work...
```

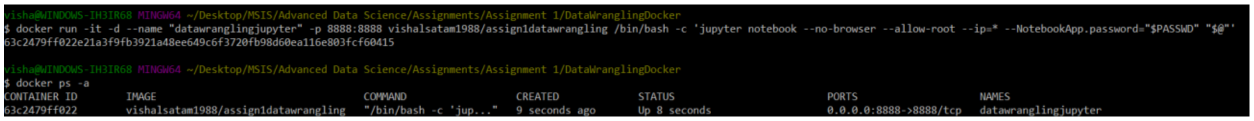
Step 5 : Commit the container to persist the changes

docker commit <containername> <new image name>

docker commit datawrangling vishalsatam1988/assign1datawrangling


```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker commit datawrangling vishalsatam1988/assign1datawrangling
sha256:6c9bc74714109d24fa7fb2a5472981c8f4d4b1dc78d93c5dab1e52d7aa33051c

visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker images
REPOSITORY                            TAG          IMAGE ID       CREATED           SIZE
vishalsatam1988/assign1datawrangling  latest       6c9bc7471410   2 seconds ago     1.55GB
```

Step 6 : Run the jupyter notebook on the committed image in detached mode docker run -it -d --name “datawranglingjupyter” -p 8888:8888 vishalsatam1988/assign1datawrangling /bin/bash -c 'jupyter notebook --no-browser --allow-root --ip=* --NotebookApp.password="$PASSWD" "$@"'

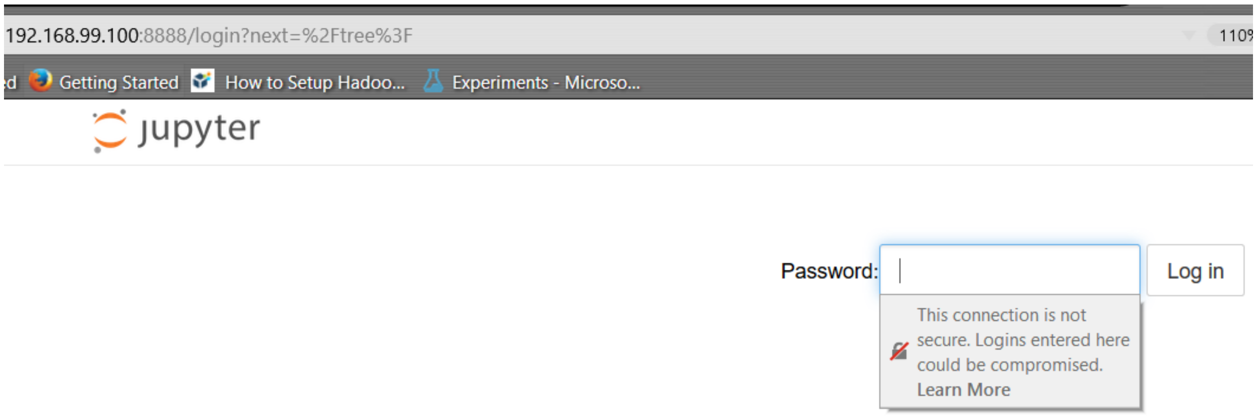


```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker run -it -d --name "datawranglingjupyter" -p 8888:8888 vishalsatam1988/assign1datawrangling /bin/bash -c 'jupyter notebook --no-browser --allow-root --ip=* --NotebookApp.password="$PASSWD" "$@"'
63c2479ff022e21a3f9fb3921a48ee649c6f3720fb98d60ea116e803fcf60415

visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker ps -a
CONTAINER ID        IMAGE                                 COMMAND                  CREATED             STATUS                          PORTS                    NAMES
63c2479ff022        vishalsatam1988/assign1datawrangling  "/bin/bash -c 'jup..."   9 seconds ago       Up 8 seconds                    0.0.0.0:8888->8888/tcp   datawranglingjupyter
```

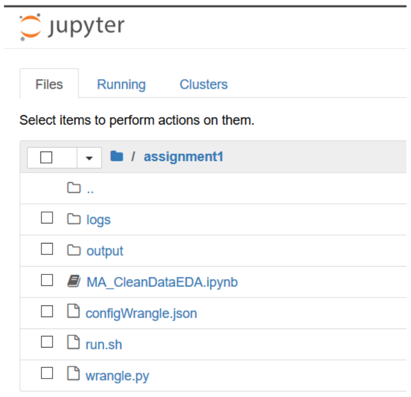
Step 7 : Connect to this running container via browser by entering

http://<docker machine ip address>:8888

Password for the jupyter notebook is keras

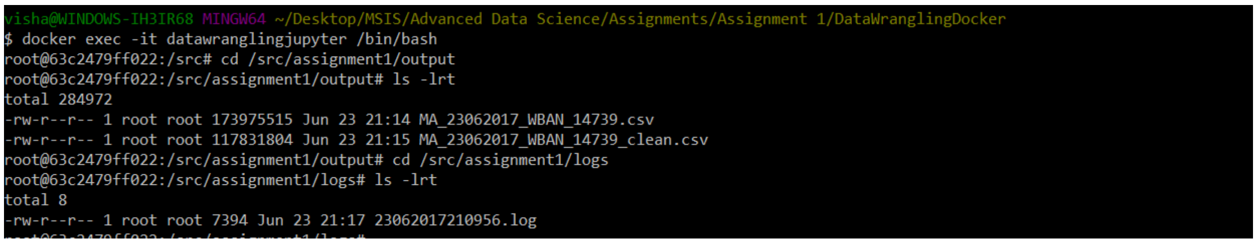

192.168.99.100:8888/login?next=%2Ftree%3F
Getting Started
How to Setup Hadoo...
Experiments - Microso...
jupyter
Password:
Log in
This connection is not secure. Logins entered here could be compromised. Learn More

Navigate to the assignment1 folder and open the MA_CleanDataEDA ipython notebook to see the analysis



Step 8 : Execute the bin/bash command to enter the running container to check output and logs

docker exec -it datawranglingjupyter /bin/bash



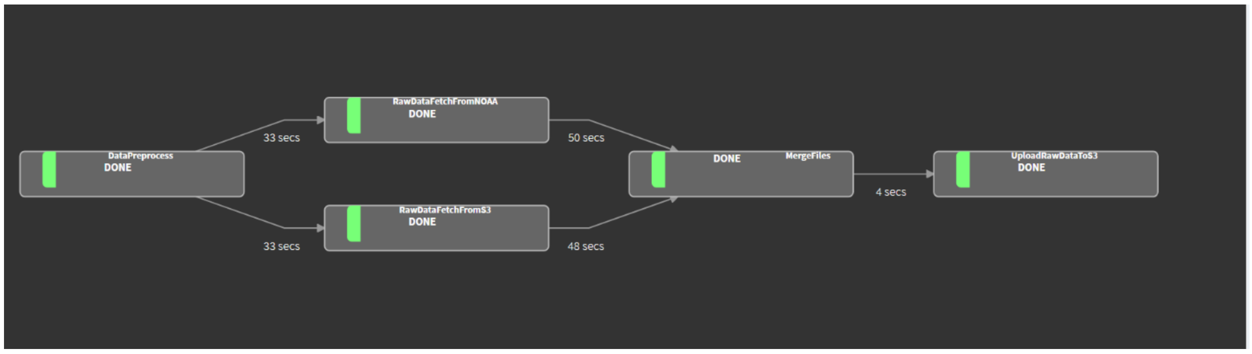
```
visha@WINDOWS-IH3IR68 MINGW64 ~/Desktop/MSIS/Advanced Data Science/Assignments/Assignment 1/DataWranglingDocker
$ docker exec -it datawranglingjupyter /bin/bash
root@63c2479ff022:/src# cd /src/assignment1/output
root@63c2479ff022:/src/assignment1/output# ls -lrt
total 284972
-rw-r--r-- 1 root root 173975515 Jun 23 21:14 MA_23062017_WBAN_14739.csv
-rw-r--r-- 1 root root 117831804 Jun 23 21:15 MA_23062017_WBAN_14739_clean.csv
root@63c2479ff022:/src/assignment1/output# cd /src/assignment1/logs
root@63c2479ff022:/src/assignment1/logs# ls -lrt
total 8
-rw-r--r-- 1 root root 7394 Jun 23 21:17 23062017210956.log
root@63c2479ff022:/src/assignment1/logs#
```

Pipeline

The steps for data ingestion and cleaning have been organized into a data processing pipeline. The pipelining framework that we have utilized for this step is Lugi. We have divided the steps into sub tasks to achieve the process of data ingestion and wrangling.

Data Ingestion

The following diagram shows the pipeline that we have built for data ingestion.



Step 1 : DataPreprocess

During this step, we check the config.json and configInitial.json file to ensure the validity of these files.

Step2 : Fetching Data

This step is divided into 2 tasks.

RawDataFetchFromNOAA – This task fetches the raw data from the NOAA website using the links defined in the config.json / configInitial.json file based on whether the S3 bucket is empty or not.

RawDataFetchFromS3 – This task connects to S3 and checks if there is any data present on the bucket. If there is no data on the bucket, then we don't fetch anything. If there is data on the bucket, then the task downloads the latest file present according to the timestamp present in the file name.

The latest file is fetched just inn case we haven't run the job on a previous date.

Step3 : Merge Files

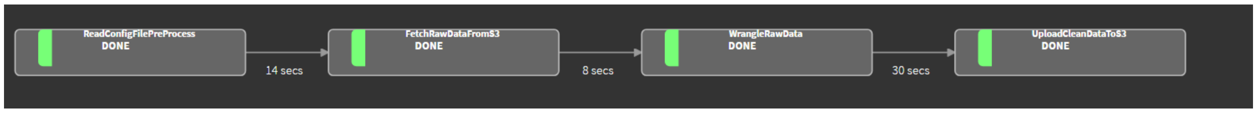
This task merges the data fetched from NOAA as well as the data fetched from S3 and outputs the final file on the local folder.

Step4 : Upload Raw Data To S3

This step collects the file returned from the merging step and uploads the data to S3

Data Wrangling

We have created a pipeline for this part of the data processing as illustrated in the below diagram



The pipeline has been divided into 4 steps.

Step 1 : DataPreprocess

This task checks the configWrangle.json file to ensure that it is valid and there are no errors in parsing this file.

Step2 : Fetch Raw Data From S3

During this task, we check for a couple of conditions.

1. If the bucket is empty, then the task logs an error and fails.
2. If the bucket already contains clean data which has todays date in the timestamp, then the task logs an error saying that todays file has already been processed. The task downloads todays clean file in case it is missing from the local directory.
3. If the bucket does not contain todays raw data then, the task fails by logging an error.
4. If the bucket contains todays raw data file which was put here by the data ingestion pipeline, then this task proceeds to download the data.

Step 3 : Wrangling Raw Data

This task performs the cleaning and wrangling on the raw data as explained in the Wrangling section of this document and outputs the final clean file to the local directory.

Step4 : Upload Clean Data To S3

This task collects the file from the earlier wrangling task and uploads the final Clean data file to the S3 bucket.

Automation

The docker containers have been automated to run using the run.sh scripts. These scripts internally invoke the luigi data processing pipelines. The main strategy for automation of these scripts is by setting up a cron job as described in the next section.

Batch Job (Cron)

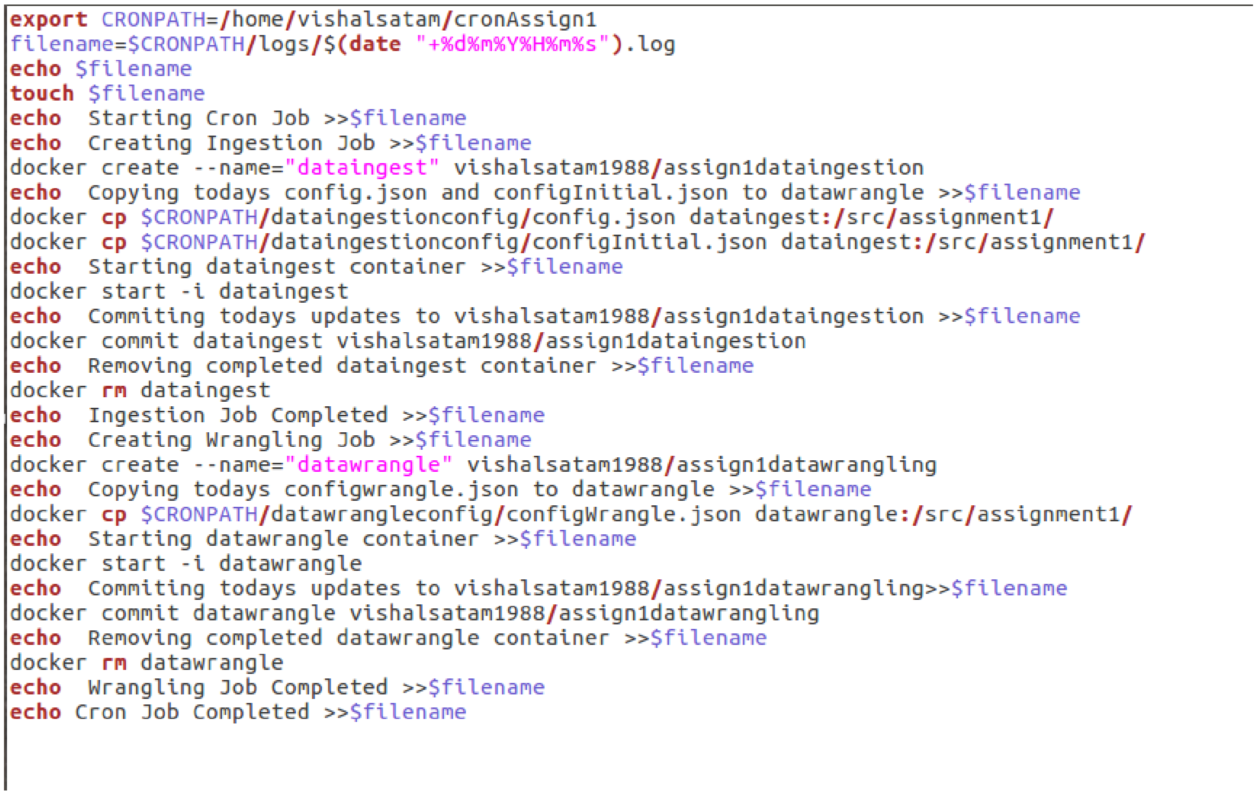
The Batch job has been designed in such a way that we will have to follow a protocol to put the config files into a folder on the local file system. The config files would mainly be updated for the daily link from the NOAA website before 8:00 PM. If one doesn’t follow this protocol, then we will end up running the job on the older links thereby wasting system resources.

The design for creating the batch job is such that we can update the config file with the new NOAA link every day in our local directory (~/dataingestionconfig and ~/datawrangleconfig) and the cron will read from this and execute the 2 docker images. Since, this is a daily batch job, we also stop and remove the running containers. This way, we preserve memory. The changes are incrementally commited to the docker images locally so that we can run our jupyter notebooks for analyzing the data later.

A cron job has been setup using the unix crontab which runs a shell script cronRun.sh to perform the following operations

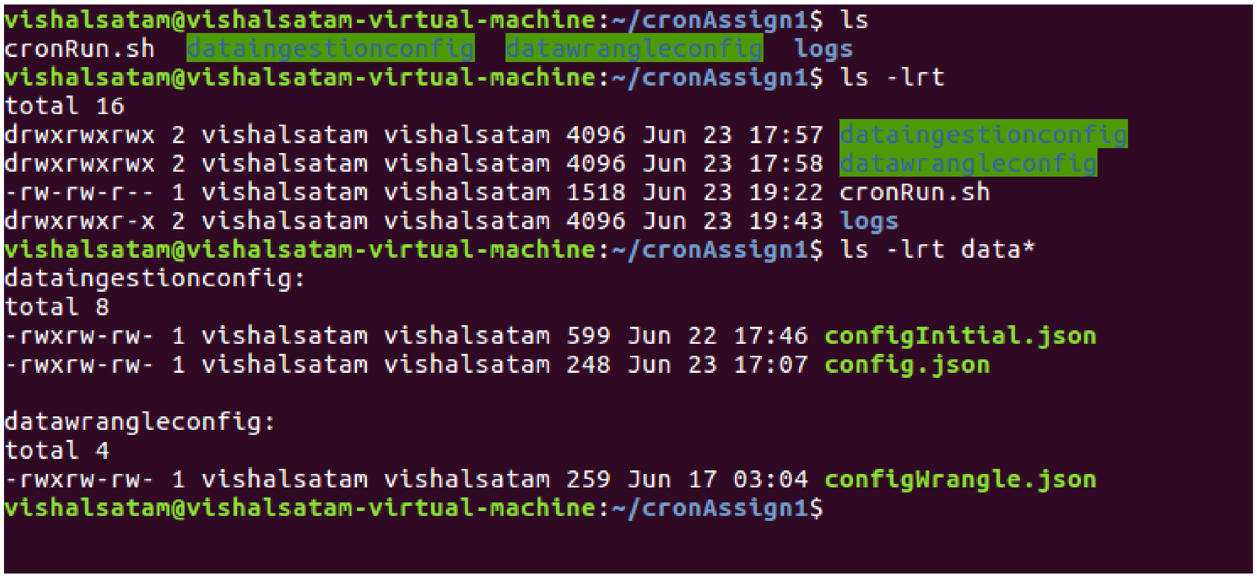
1. Create a new container for dataingestion and datawrangling every day at 8:00 pm
2. copy the new config files containing the updated links
3. start the created containers
4. commit the changes
5. remove the running containers
6. The shell script also logs the output to a log file to indicate to the user about the execution summary of the cron job.

Shell script that executes the 2 docker images which is scheduled to run at 8:00 PM everyday



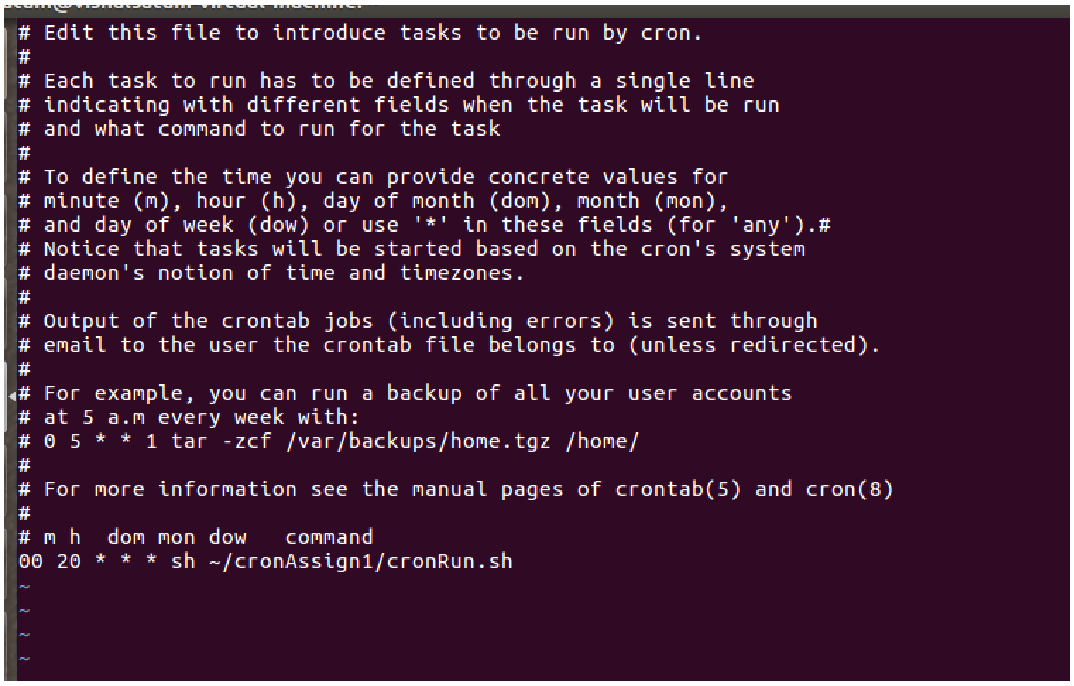
```
export CRONPATH=/home/vishalsatam/cronAssign1
filename=$CRONPATH/logs/$(date "+%d%m%Y%H%m%s").log
echo $filename
touch $filename
echo  Starting Cron Job >>$filename
echo  Creating Ingestion Job >>$filename
docker create --name="dataingest" vishalsatam1988/assign1dataingestion
echo  Copying todays config.json and configInitial.json to datawrangle >>$filename
docker cp $CRONPATH/dataingestionconfig/config.json dataingest:/src/assignment1/
docker cp $CRONPATH/dataingestionconfig/configInitial.json dataingest:/src/assignment1/
echo  Starting dataingest container >>$filename
docker start -i dataingest
echo  Commiting todays updates to vishalsatam1988/assign1dataingestion >>$filename
docker commit dataingest vishalsatam1988/assign1dataingestion
echo  Removing completed dataingest container >>$filename
docker rm dataingest
echo  Ingestion Job Completed >>$filename
echo  Creating Wrangling Job >>$filename
docker create --name="datawrangle" vishalsatam1988/assign1datawrangling
echo  Copying todays configwrangle.json to datawrangle >>$filename
docker cp $CRONPATH/datawrangleconfig/configWrangle.json datawrangle:/src/assignment1/
echo  Starting datawrangle container >>$filename
docker start -i datawrangle
echo  Commiting todays updates to vishalsatam1988/assign1datawrangling>>$filename
docker commit datawrangle vishalsatam1988/assign1datawrangling
echo  Removing completed datawrangle container >>$filename
docker rm datawrangle
echo  Wrangling Job Completed >>$filename
echo Cron Job Completed >>$filename
```

Folder Structure



```
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$ ls
cronRun.sh  dataingestionconfig  datawrangleconfig  logs
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$ ls -lrt
total 16
drwxrwxrwx 2 vishalsatam vishalsatam 4096 Jun 23 17:57 dataingestionconfig
drwxrwxrwx 2 vishalsatam vishalsatam 4096 Jun 23 17:58 datawrangleconfig
-rw-rw-r-- 1 vishalsatam vishalsatam 1518 Jun 23 19:22 cronRun.sh
drwxrwxr-x 2 vishalsatam vishalsatam 4096 Jun 23 19:43 logs
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$ ls -lrt data*
dataingestionconfig:
total 8
-rwxrw-rw- 1 vishalsatam vishalsatam 599 Jun 22 17:46 configInitial.json
-rwxrw-rw- 1 vishalsatam vishalsatam 248 Jun 23 17:07 config.json

datawrangleconfig:
total 4
-rwxrw-rw- 1 vishalsatam vishalsatam 259 Jun 17 03:04 configWrangle.json
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$
```

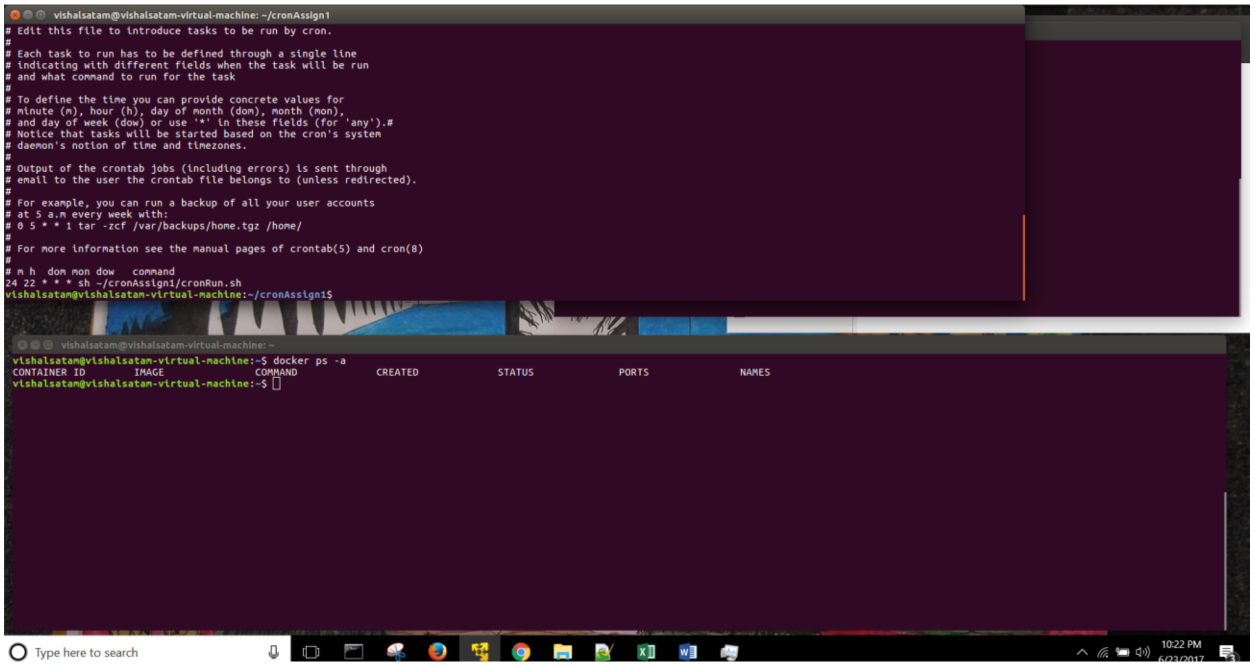
Crontab is created as follows



```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
00 20 * * * sh ~/cronAssign1/cronRun.sh
~
~
~
~
```

CRON Demo

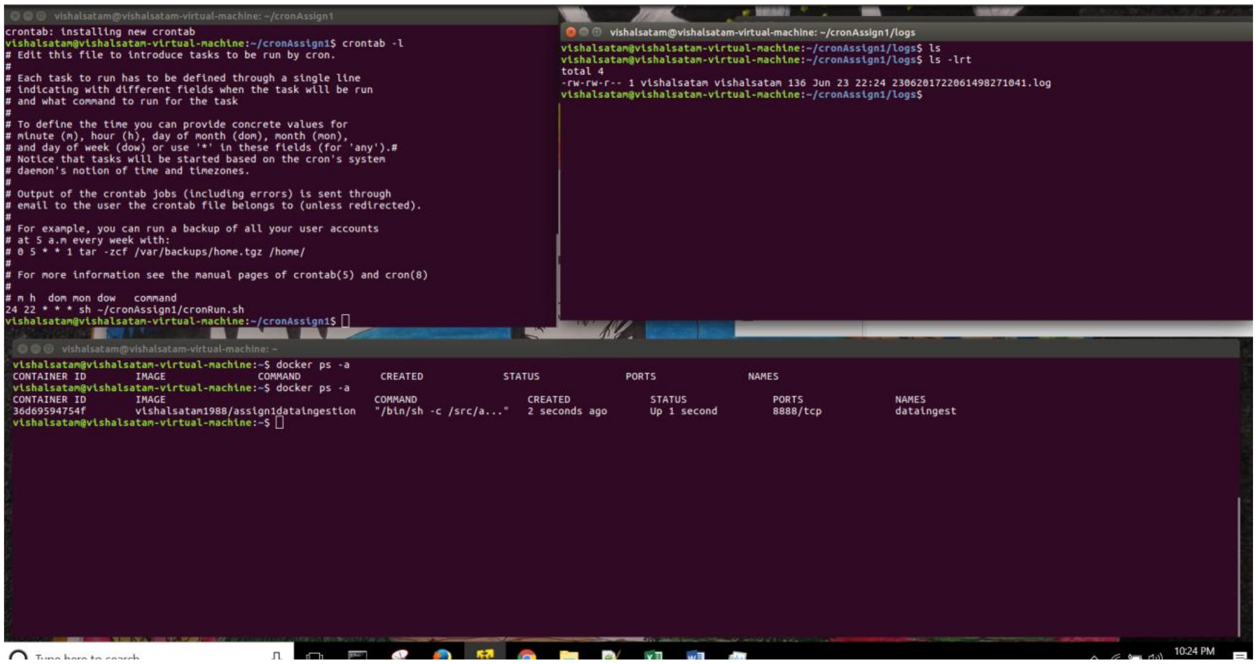
Scheduling the job at 22:24 daily for demo purposes. We should see a log file appear and the container execution starting



```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
24 22 * * * sh ~/cronAssign1/cronRun.sh
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$
```

```
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
vishalsatam@vishalsatam-virtual-machine:~$
```

As we can see, the cron job has started execution.

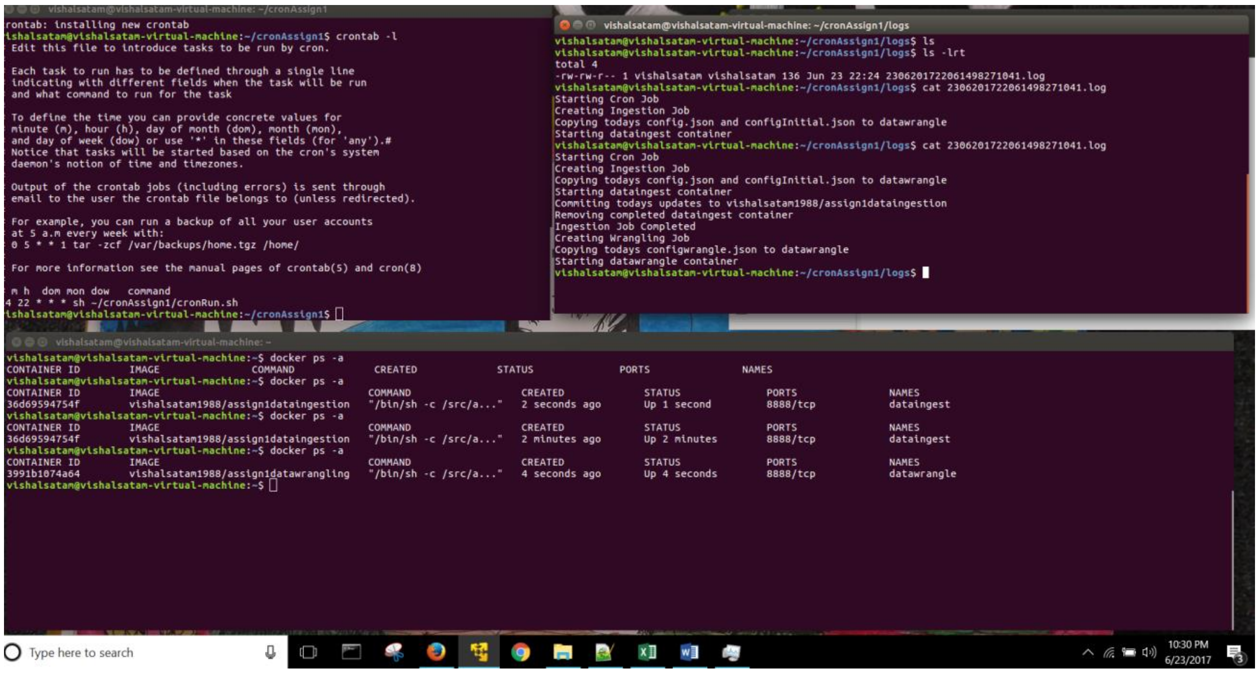


```
crontab: installing new crontab
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
24 22 * * * sh ~/cronAssign1/cronRun.sh
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$
```

```
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$ ls
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$ ls -lrt
total 4
-rw-rw-r-- 1 vishalsatam vishalsatam 136 Jun 23 22:24 2306201722061498271041.log
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$
```

```
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE                                  COMMAND                  CREATED             STATUS              PORTS               NAMES
36d69594754f        vishalsatam1988/assign1dataingestion   "/bin/sh -c /src/a..."   2 seconds ago       Up 1 second         8888/tcp            dataingest
vishalsatam@vishalsatam-virtual-machine:~$
```

Now we can see that the cron job has commited changes to the data ingestion container and has fired up the data wrangling docker container. Since, this is a daily batch job, we also stop and remove the running containers. This way, we preserve memory.

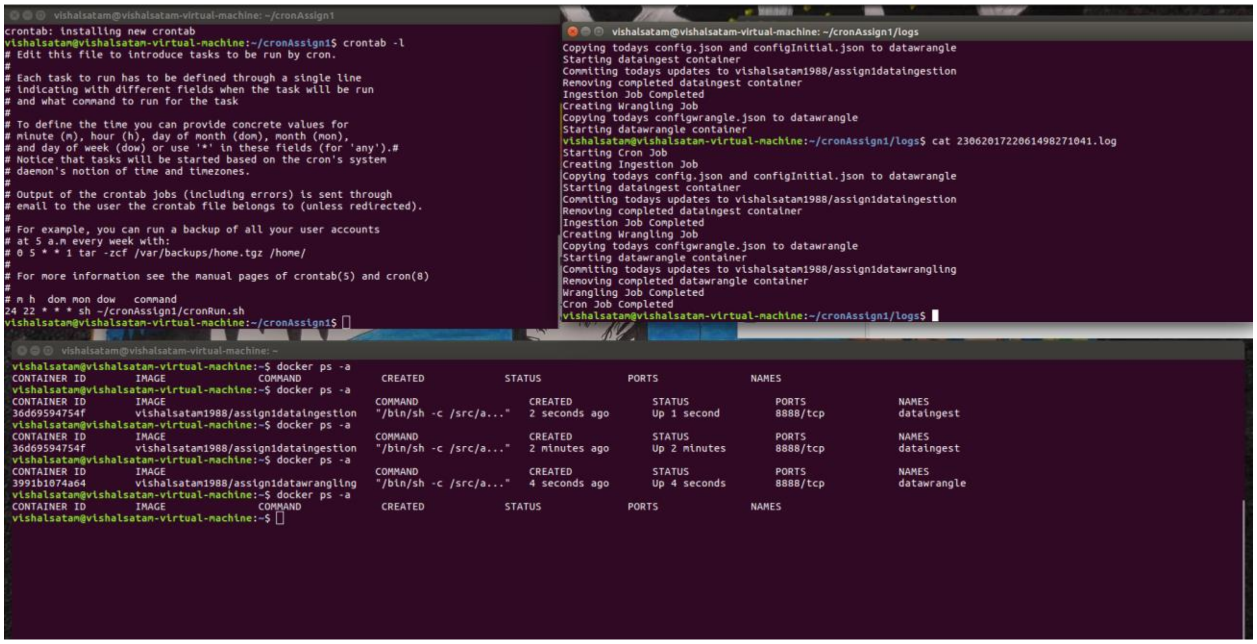


```
rontab: installing new crontab
ishalsatam@vishalsatam-virtual-machine:~/cronAssign1$ crontab -l
 Edit this file to introduce tasks to be run by cron.

 Each task to run has to be defined through a single line
 indicating with different fields when the task will be run
 and what command to run for the task

 To define the time you can provide concrete values for
 minute (m), hour (h), day of month (dom), month (mon),
 and day of week (dow) or use '*' in these fields (for 'any').#
 Notice that tasks will be started based on the cron's system
 daemon's notion of time and timezones.

 Output of the crontab jobs (including errors) is sent through
 email to the user the crontab file belongs to (unless redirected).

 For example, you can run a backup of all your user accounts
 at 5 a.m every week with:
 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/

 For more information see the manual pages of crontab(5) and cron(8)

 m h  dom mon dow   command
4 22 * * * sh ~/cronAssign1/cronRun.sh
ishalsatam@vishalsatam-virtual-machine:~/cronAssign1$
```

```
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$ ls
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$ ls -lrt
total 4
-rw-rw-r-- 1 vishalsatam vishalsatam 136 Jun 23 22:24 2306201722061498271041.log
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$ cat 2306201722061498271041.log
Starting Cron Job
Creating Ingestion Job
Copying todays config.json and configInitial.json to datawrangle
Starting dataingest container
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$ cat 2306201722061498271041.log
Starting Cron Job
Creating Ingestion Job
Copying todays config.json and configInitial.json to datawrangle
Starting dataingest container
Commiting todays updates to vishalsatam1988/assign1dataingestion
Removing completed dataingest container
Ingestion Job Completed
Creating Wrangling Job
Copying todays configwrangle.json to datawrangle
Starting datawrangle container
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$
```

```
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE                                  COMMAND                  CREATED             STATUS              PORTS               NAMES
36d69594754f        vishalsatam1988/assign1dataingestion   "/bin/sh -c /src/a..."   2 seconds ago       Up 1 second         8888/tcp            dataingest
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE                                  COMMAND                  CREATED             STATUS              PORTS               NAMES
36d69594754f        vishalsatam1988/assign1dataingestion   "/bin/sh -c /src/a..."   2 minutes ago       Up 2 minutes        8888/tcp            dataingest
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE                                  COMMAND                  CREATED             STATUS              PORTS               NAMES
3991b1074a64        vishalsatam1988/assign1datawrangling   "/bin/sh -c /src/a..."   4 seconds ago       Up 4 seconds        8888/tcp            datawrangle
vishalsatam@vishalsatam-virtual-machine:~$
```

We can see from the logs that the cron job has completed execution. The 2 docker images were fired up sequentially and the data processing pipeline has been successfully executed.



```
crontab: installing new crontab
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
24 22 * * * sh ~/cronAssign1/cronRun.sh
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1$
```

```
Copying todays config.json and configInitial.json to datawrangle
Starting dataingest container
Commiting todays updates to vishalsatam1988/assign1dataingestion
Removing completed dataingest container
Ingestion Job Completed
Creating Wrangling Job
Copying todays configwrangle.json to datawrangle
Starting datawrangle container
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$ cat 2306201722061498271041.log
Starting Cron Job
Creating Ingestion Job
Copying todays config.json and configInitial.json to datawrangle
Starting dataingest container
Commiting todays updates to vishalsatam1988/assign1dataingestion
Removing completed dataingest container
Ingestion Job Completed
Creating Wrangling Job
Copying todays configwrangle.json to datawrangle
Starting datawrangle container
Commiting todays updates to vishalsatam1988/assign1datawrangling
Removing completed datawrangle container
Wrangling Job Completed
Cron Job Completed
vishalsatam@vishalsatam-virtual-machine:~/cronAssign1/logs$
```

```
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE                                 COMMAND                  CREATED             STATUS              PORTS               NAMES
36d69594754f        vishalsatam1988/assign1dataingestion  "/bin/sh -c /src/a..."   2 seconds ago       Up 1 second         8888/tcp            dataingest
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE                                 COMMAND                  CREATED             STATUS              PORTS               NAMES
36d69594754f        vishalsatam1988/assign1dataingestion  "/bin/sh -c /src/a..."   2 minutes ago       Up 2 minutes        8888/tcp            dataingest
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE                                 COMMAND                  CREATED             STATUS              PORTS               NAMES
3991b1074a64        vishalsatam1988/assign1datawrangling  "/bin/sh -c /src/a..."   4 seconds ago       Up 4 seconds        8888/tcp            datawrangle
vishalsatam@vishalsatam-virtual-machine:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
vishalsatam@vishalsatam-virtual-machine:~$
```

EDA on RAW DATA:

Dataset Description: Local Climatological Data (LCD) summaries provide a synopsis of climatic values for a single weather station over a specific month. This dataset is for BOSTON MA station.

Total row count : 724623   Columns: 90

STATION_NAME: BOSTON MA US

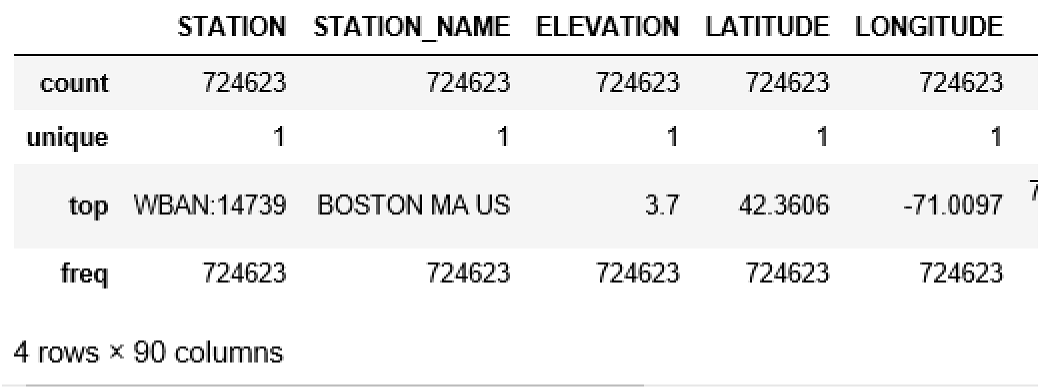
STATION: WBAN:14739

ELEVATION: 3.7

LATITUDE: 42.3606

LONGITUDE: -71.0097

Cleaning strategy: This data is constant and should not vary. For this dataset there are no missing values for the above columns. In case of missing data replace by median.



	STATION	STATION_NAME	ELEVATION	LATITUDE	LONGITUDE
count	724623	724623	724623	724623	724623
unique	1	1	1	1	1
top	WBAN:14739	BOSTON MA US	3.7	42.3606	-71.0097
freq	724623	724623	724623	724623	724623

4 rows × 90 columns

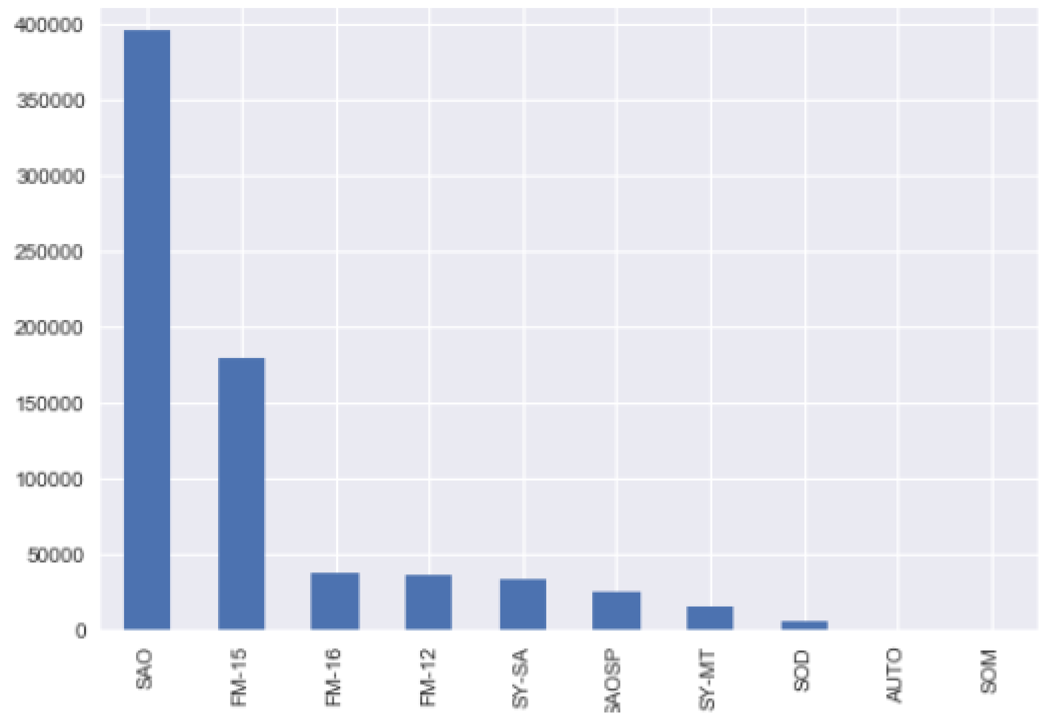
DATE:

This column consists of dates and timestamp starting from 11/21/1943 1:00:00 AM to 6/16/2017 11:59:00 PM., for readings of the weather report. It is the Index column and contains all unique values.

Characteristics: Date ranges are inconsistent due to the type of reporting used for the weather data. Data from 1943 to 1993 consists of 17 reports per day while the later data consists of 24 reports each for one hour.

Cleaning: check for nulls. If a value is null discard the row as it invalidates the data reported. For missing time series create day-wise bins for analysis.

REPORTTPYE:



SAO = Airways report FM-15 = METAR Aviation routine weather report

SAOSP = Airways special report FM-16 = SPECI Aviation selected special weather report

SY-MT = Synoptic and METAR merged report SY-SA = Synoptic and airways merged report

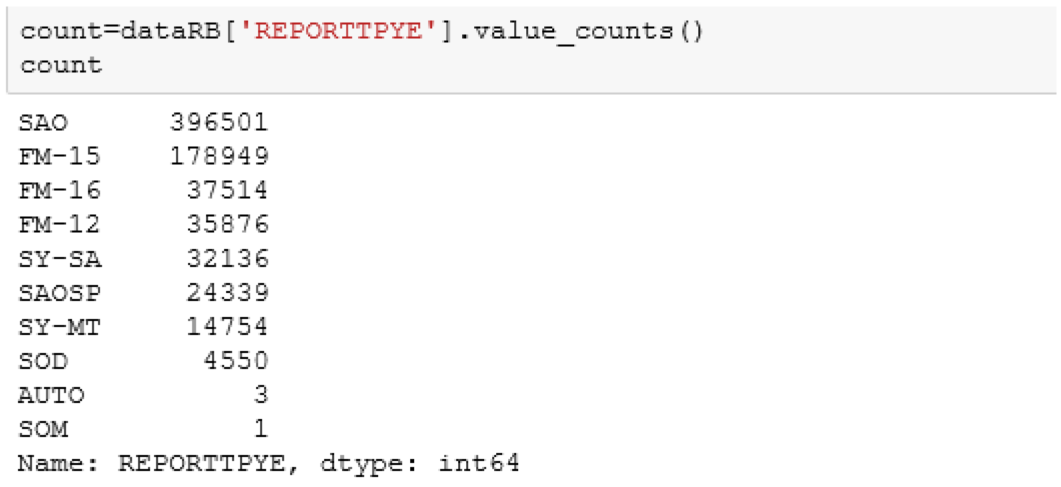
FM-12 = SYNOP Report of surface observation form a fixed land station

We have 10 types of report types as seen in in the above, with maximum types as SAO and FM-15..

SAO and SAOSP are taken till the year 1993 and later are replaced with FM-15 and FM-16. The report types SOD : Summary of Data and SOM : Summary of Month are introduced after 1993 with the FM-15 reports and are missing in the earlier data.

Cleaning:

NO null values found. FM-15 and SAO can be replaced in case of null values as they are taken on periodic basis in a span of 60 minutes. FM-12 and SY-MT can be replaced as they are also taken every three hours. FM-16 and SAOSP are special repots taken to indicate sudden weather changes. If the timestamp is not periodic it can be replaced by SAOSP for data prior to 1993 and FM-16 for later.



```
count=dataRB['REPORTTPYE'].value_counts()
count
```

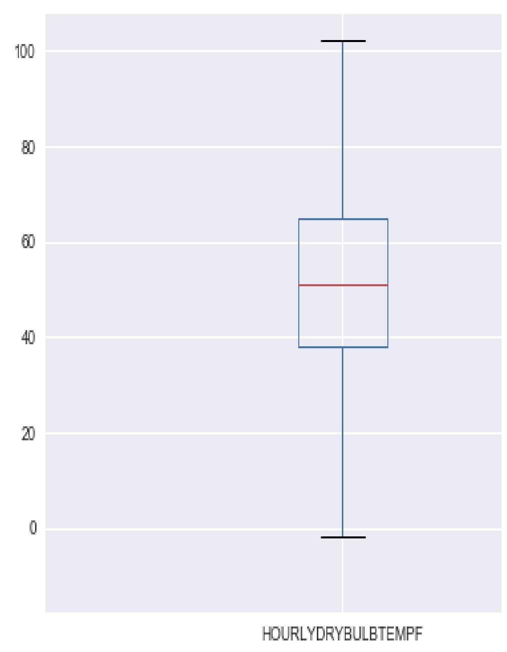
```
SAO      396501
FM-15    178949
FM-16     37514
FM-12     35876
SY-SA     32136
SAOSP     24339
SY-MT     14754
SOD        4550
AUTO          3
SOM           1
Name: REPORTTPYE, dtype: int64
```

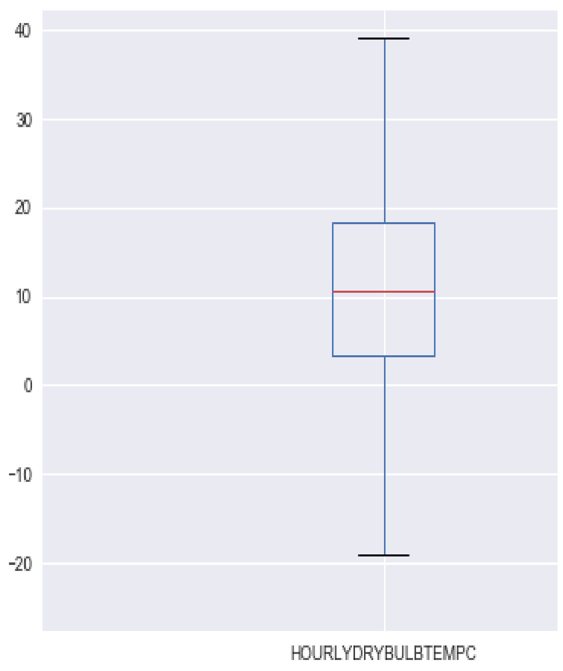
HOURLYDRYBULBTEMPF & HOURLYDRYBULBTEMPC

This column represents the standard air temperature and is given in whole Fahrenheit and tenths of degree Celsius.

Cleaning: The data varies between -12 f to 101 F with mild outliers suggesting extreme weather. Missing values can be calculated by taking mean of values of the previous and next value. Can use interpolation.

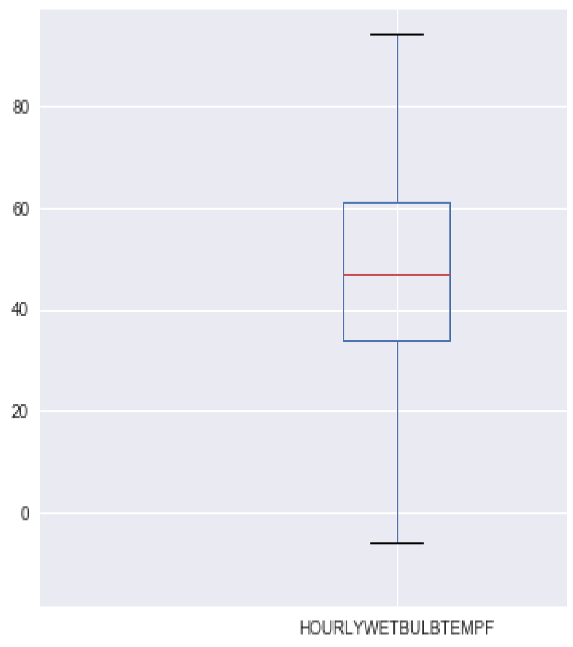
Data missing for either columns can be filled with Fahrenheit to Celsius conversions and vice versa.

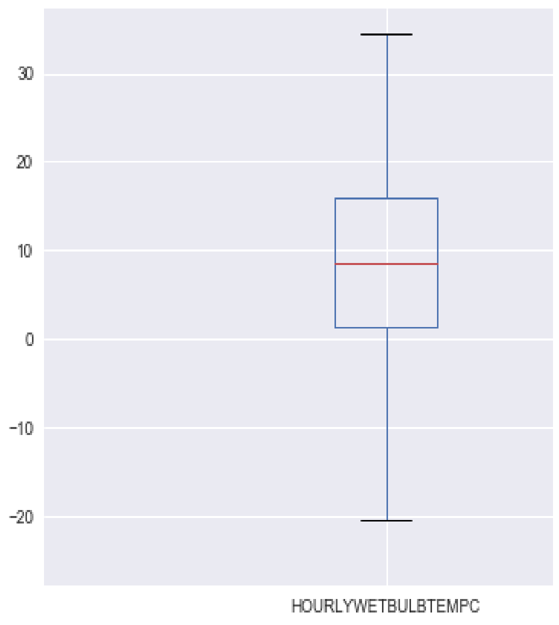




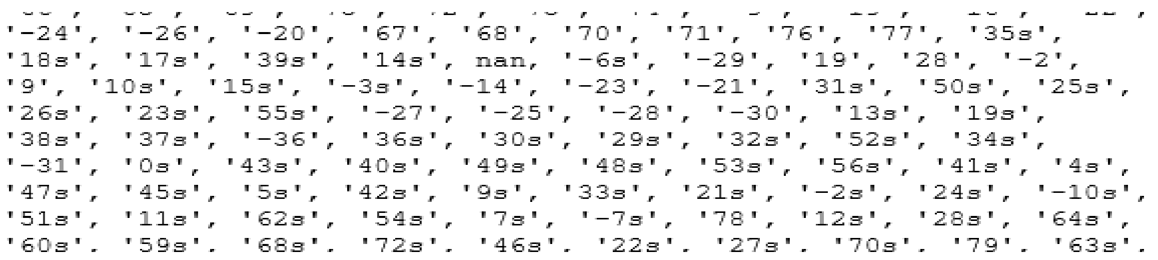
HOURLYWETBULBTEMPF and HOURLYWETBULBTEMPC

The wet-bulb temperature is the lowest temperature that can be reached under current ambient conditions by the evaporation of water only and is largely determined by both actual air temperature (dry-bulb temperature) and the amount of moisture in the air (humidity).





Cleaning: The data has many null values and has missing values for first two years. It also has character values as shown below. s = suspect value and v = variable value which can be replace by numbers.



```
'-24', '-26', '-20', '67', '68', '70', '71', '76', '77', '35s',
'18s', '17s', '39s', '14s', nan, '-6s', '-29', '19', '28', '-2',
'9', '10s', '15s', '-3s', '-14', '-23', '-21', '31s', '50s', '25s',
'26s', '23s', '55s', '-27', '-25', '-28', '-30', '13s', '19s',
'38s', '37s', '-36', '36s', '30s', '29s', '32s', '52s', '34s',
'-31', '0s', '43s', '40s', '49s', '48s', '53s', '56s', '41s', '4s',
'47s', '45s', '5s', '42s', '9s', '33s', '21s', '-2s', '24s', '-10s',
'51s', '11s', '62s', '54s', '7s', '-7s', '78', '12s', '28s', '64s',
'60s'. '59s'. '68s'. '72s'. '46s'. '22s'. '27s'. '70s'. '79'. '63s'.
```

For missing data, Interpolation or mean value substitution won't work. Wetbulb temperature is highly correlated with DryBulb temperature and the Dewpoint temperature and an approximate value can be calculated for conditions where drybulb temperature is equal to or below 65 F. The formula is as follows:

X = DryBulbTemp – DewPointTemp

Y = X/3

Approximate WetBulbTemp = DryBulb Temp – Y

Once major chunks of missing data are filled, interpolation can be tried.

Data missing for either columns can be filled with Fahrenheit to Celsius conversions and vice versa.

[For more info : Wetbulbcalculation]

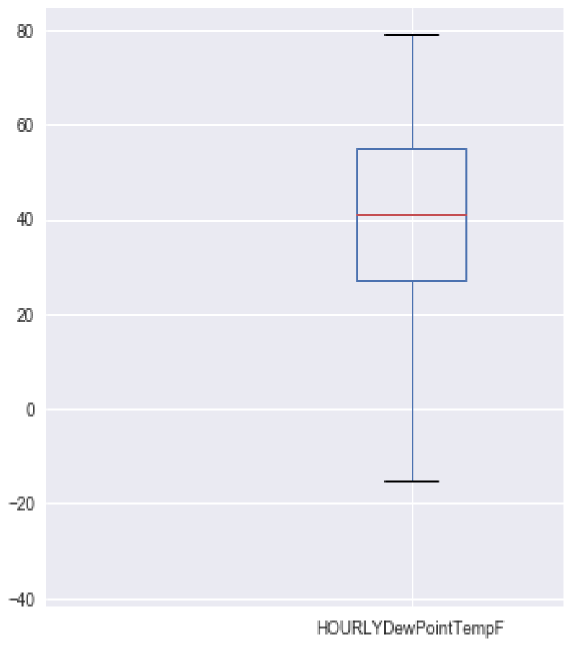
HOURLYDewPointTempF & HOURLYDewPointTempC

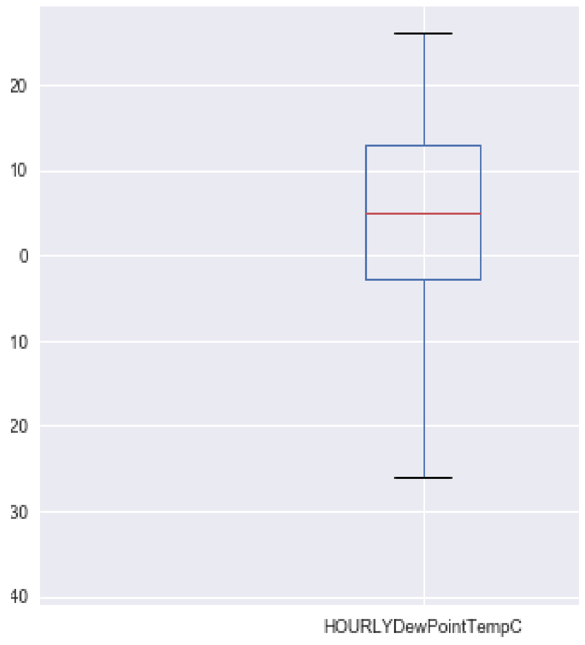
The temperature to which a given parcel of air must be cooled at constant pressure and water vapor content in order for saturation to occur. It is given in whole Fahrenheit and tenths of degree Celsius.

Cleaning:

The data has many null values and has missing values for first two years. It also has character values as

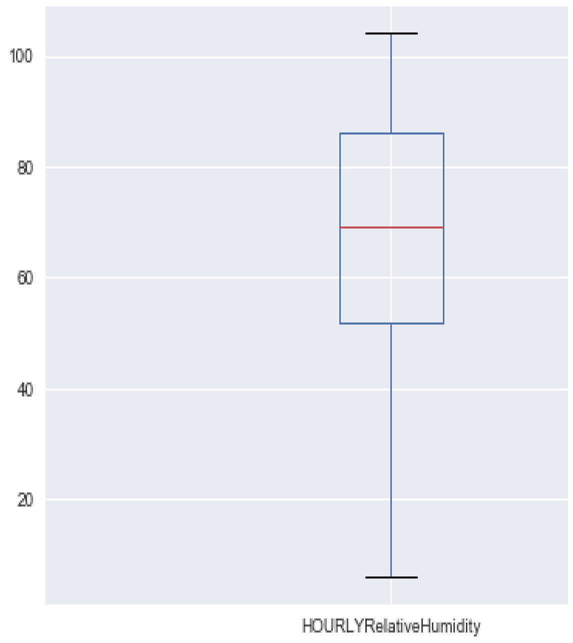
s = suspect value and v = variable value which can be replace by numbers. Data missing for either columns can be filled with Fahrenheit to Celsius conversions and vice versa.





HourlyRelativeHumidity:

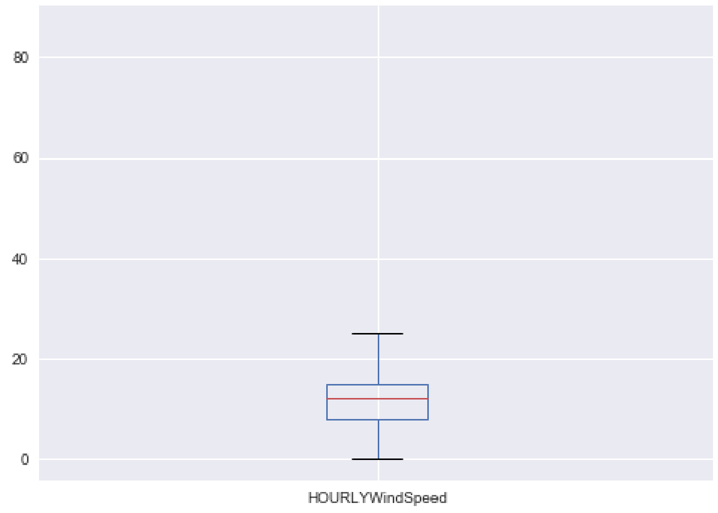
Relative humidity (RH) is the ratio of the partial pressure of water vapor to the equilibrium vapor pressure of water at a given temperature. Relative humidity depends on temperature and the pressure of the system of interest. The data is varying between 10 to 100 Hg, but with no outlier value, The extreme values can be considered as extreme temperature.



Cleaning: Data contains character values of suspect and variable type , nan values and adjusted values denoted by ‘*’ can be interpolated and filled in case previous and next value is present.

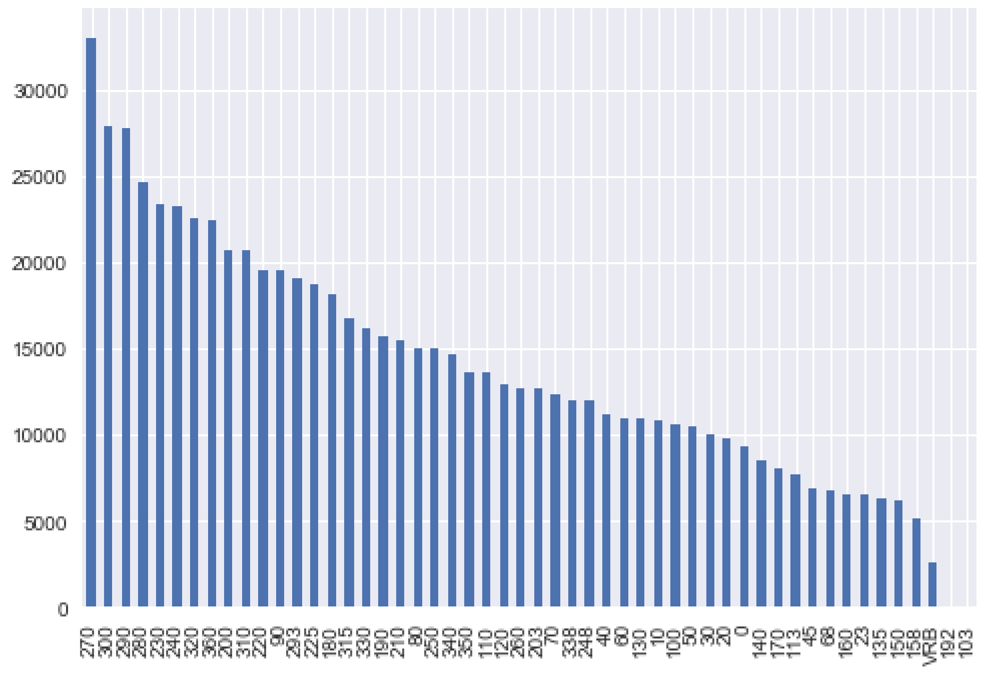
HOURLYWindSpeed:

Speed of the wind at the time of observation given in miles per hour (mph), most common is 9mph. Data has character values, nan values. Values cannot be negative for speed, so a check for negative values.



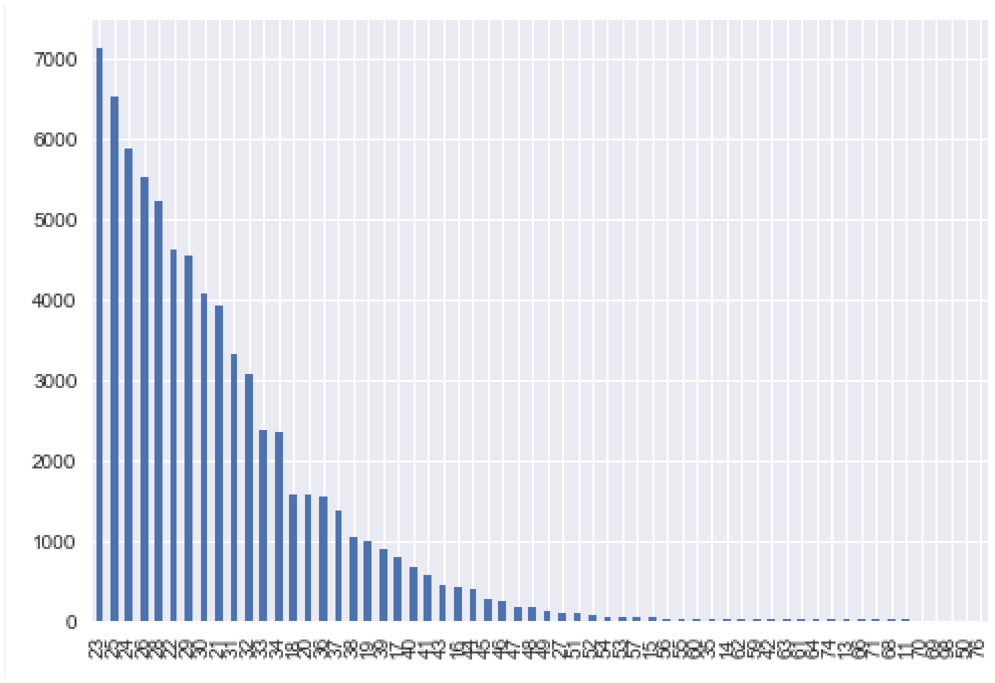
HOURLYWindDirection:

Wind direction from true north using compass directions (360 = true north, 180 = south, 270 = east) A direction of “000” is given for calm winds. Values cannot be negative for speed, so a check for negative values. Also 0 is not missing value but denotes ‘calm winds’.



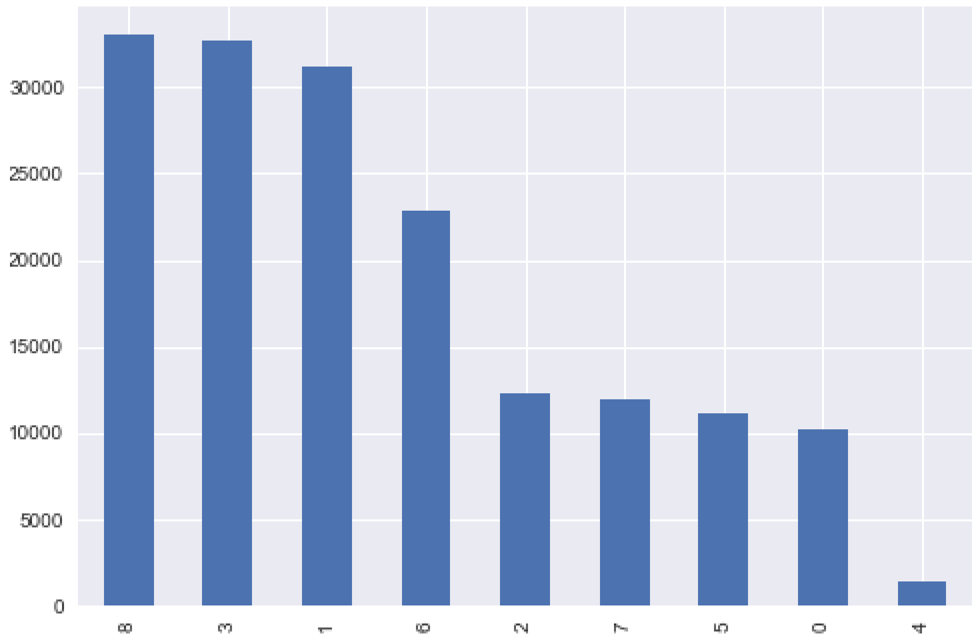
HOURLYWindGustSpeed:

Wind gusts occurring during time of observation. Given in miles per hour (mph).



HOURLYPressureTendency:

Pressure tendency (In general a 0 through 3 here indicates an increase in pressure over previous 3 hours and a 5 through 8 indicates a decrease over the previous 3 hours and 4 indicates no change during the previous 3 hours).



HOURLYPrecip:

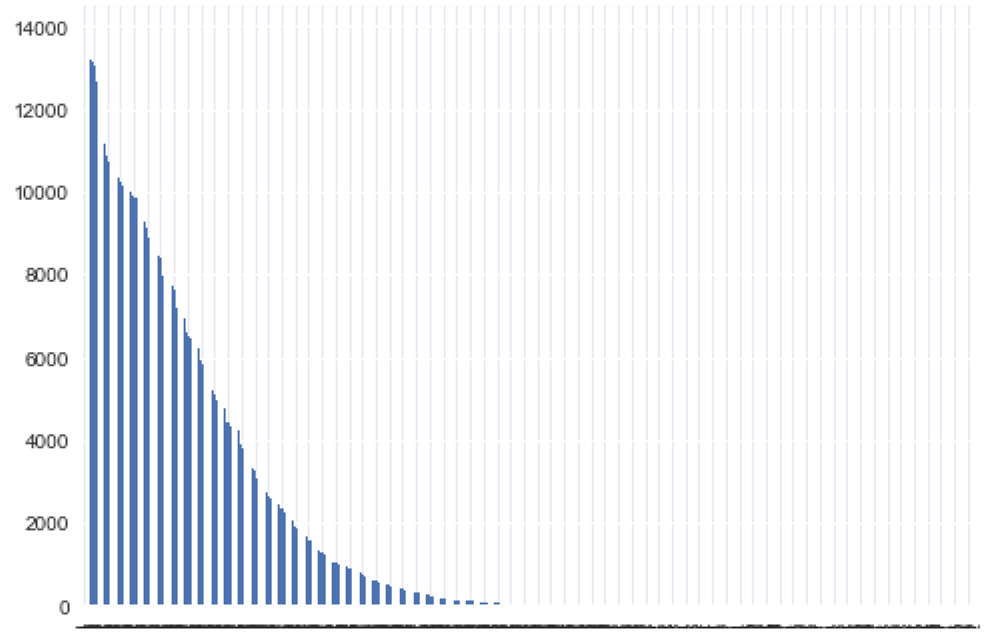
The liquid precipitation observation values given in inches to hundredths and are placed in the cell corresponding to the intersection of the day and hour. All quantities represent what amount of precipitation fell for the hour ending at the time indicated on the table. Trace amounts of precipitation are indicated with a "T."

Blank/null = no precipitation was observed/reported for the hour ending at that date/time. M = missing

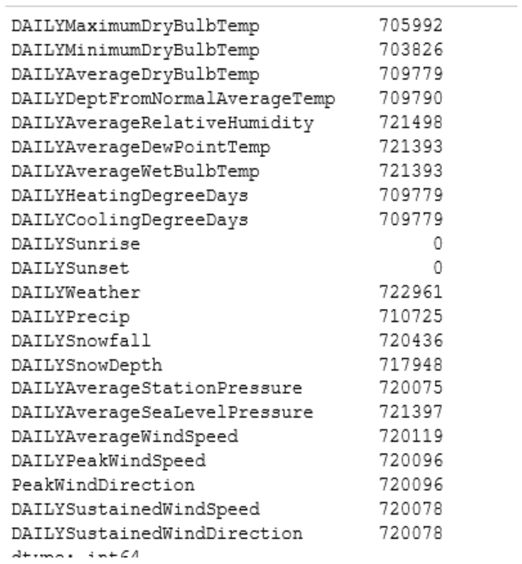
Cleaning: Clean character values, replace T or trace amounts by 0.005 to keep the column numeric.

HOURLYSeaLevelPressure:

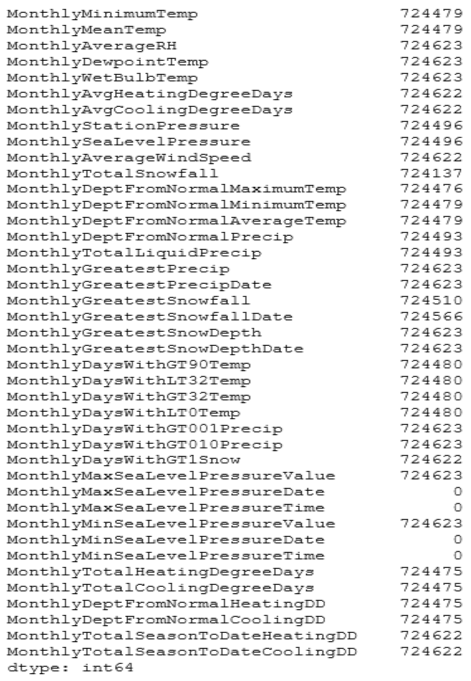
Sea level pressure given in inches of Mercury (in Hg). Clean out the character values.



For Daily and Monthly summary:



```
DAILYMaximumDryBulbTemp              705992
DAILYMinimumDryBulbTemp              703826
DAILYAverageDryBulbTemp              709779
DAILYDeptFromNormalAverageTemp       709790
DAILYAverageRelativeHumidity         721498
DAILYAverageDewPointTemp             721393
DAILYAverageWetBulbTemp              721393
DAILYHeatingDegreeDays               709779
DAILYCoolingDegreeDays               709779
DAILYSunrise                              0
DAILYSunset                               0
DAILYWeather                         722961
DAILYPrecip                          710725
DAILYSnowfall                        720436
DAILYSnowDepth                       717948
DAILYAverageStationPressure          720075
DAILYAverageSeaLevelPressure         721397
DAILYAverageWindSpeed                720119
DAILYPeakWindSpeed                   720096
PeakWindDirection                    720096
DAILYSustainedWindSpeed              720078
DAILYSustainedWindDirection          720078
```



```
MonthlyMinimumTemp                      724479
MonthlyMeanTemp                         724479
MonthlyAverageRH                        724623
MonthlyDewpointTemp                     724623
MonthlyWetBulbTemp                      724623
MonthlyAvgHeatingDegreeDays             724622
MonthlyAvgCoolingDegreeDays             724622
MonthlyStationPressure                  724496
MonthlySeaLevelPressure                 724496
MonthlyAverageWindSpeed                 724622
MonthlyTotalSnowfall                    724137
MonthlyDeptFromNormalMaximumTemp        724476
MonthlyDeptFromNormalMinimumTemp        724479
MonthlyDeptFromNormalAverageTemp        724479
MonthlyDeptFromNormalPrecip             724493
MonthlyTotalLiquidPrecip                724493
MonthlyGreatestPrecip                   724623
MonthlyGreatestPrecipDate               724623
MonthlyGreatestSnowfall                 724510
MonthlyGreatestSnowfallDate             724566
MonthlyGreatestSnowDepth                724623
MonthlyGreatestSnowDepthDate            724623
MonthlyDaysWithGT90Temp                 724480
MonthlyDaysWithLT32Temp                 724480
MonthlyDaysWithGT32Temp                 724480
MonthlyDaysWithLT0Temp                  724480
MonthlyDaysWithGT001Precip              724623
MonthlyDaysWithGT010Precip              724623
MonthlyDaysWithGT1Snow                  724622
MonthlyMaxSeaLevelPressureValue         724623
MonthlyMaxSeaLevelPressureDate               0
MonthlyMaxSeaLevelPressureTime               0
MonthlyMinSeaLevelPressureValue         724623
MonthlyMinSeaLevelPressureDate               0
MonthlyMinSeaLevelPressureTime               0
MonthlyTotalHeatingDegreeDays           724475
MonthlyTotalCoolingDegreeDays           724475
MonthlyDeptFromNormalHeatingDD          724475
MonthlyDeptFromNormalCoolingDD          724475
MonthlyTotalSeasonToDateHeatingDD       724622
MonthlyTotalSeasonToDateCoolingDD       724622
dtype: int64
```

Maximum values are null except for the values for the day and respective months. Can be eliminated to maintain consistency of the Hourly data OR can be retained by eliminating the SOD and SOM rows. and shifting the summary values one row up,ie, on the last timestamp of the day or last day of the month.

DATA CLEANING SUMMARY:

For preliminary cleaning, character values such as 0.9v, 9Vs, 89s are replaced by their numeric types as keeping v and s that is, variable and suspect values don't add much value to the data. The VRB found in WindDirection column is replaced by -1. The Trace amounts or T found in HourlyPrecipitaion can be replaced by 0.005 as it is less than 0.005 [mentioned in the document.]

For all temperature[HOURLYDRYBULBTEMPF, HOURLYWETBULBTEMPF,DEWPOINTTEMP] columns with Fahrenheit and Celsius values, we replaced the missing values based on their available counterpart. Calculated the Wetblbtemp missing values by the formula given above.

Removed the SOD and SOM columns and to make the Hourly data consistent for liner interpolation of values having previous and next value available.

ISSUES FACED:

Our initial data was from Bedford Station which had missing data for ten years, 1993 – 2003. Switched to BOSTON station.