**Installing Python**

1. First go to the following and install the appropriate version of Python 2.7 for your system:

<https://www.python.org/download/releases/2.7.6/>

1. Now that python is installed we need to set up the environment variables on windows so that you can call python from command prompt (for windows 7):
   1. Go to control panel
   2. Go to system and security
   3. Go to system
   4. Go to advanced system settings
   5. Click on Environment Variables
   6. Go to System variables and find Path
   7. Click edit
   8. At the very end add this (do NOT delete anything else!):
      1. ; C:\Python27
2. Ok now open up a command terminal and type in ‘python’ something should come up. Congratulations you have successfully configured python on your windows machine.

**Installing 3rd Party Libraries**

Getting Pip:

1. Download the following file: <https://raw.github.com/pypa/pip/master/contrib/get-pip.py>
2. Save it as ‘get-pip.py’ in a folder that you can easily locate using command prompt
3. Open up command prompt and run in the folder where you saved the file from before:
   1. python get-pip.py
4. Now lets check that you have successfully got pip:
   1. Type in ‘python –m pip’ in your command terminal
   2. If you get a list of options then you have successfully gotten pip configured on your python installation

Getting PyCrypto:

1. http://www.voidspace.org.uk/python/modules.shtml#pycrypto
2. And select Pycrpto 2.6 for Python 2.7 64 bit
3. Go ahead and install this.

Getting Paramiko:

1. open up a command prompt window and run:
   1. ‘python –m pip install paramiko’

**Congratulations you can now use the hdfs\_load.py script! See the next page**

**Using hdfs\_load.py**

1. How do you run the program:
   1. Navigate to the ‘edmunds\_hdfs\_load’ folder that you downloaded from pypi
   2. To run the script you will type in command terminal:
      1. ‘python main.py <path to configuration file here>’
2. Configuration files explained
   1. Navigate to the ‘documentation’ folder where you found this document and you will see a file named ‘sample\_config.cfg’
   2. This file is where you specify the parameters for your job. Once made you will not have to worry about editing this file again. The idea will be to have a different configuration file for each group of files that you will need to move over. Hopefully the purpose of these files will become clearer in the next section.
3. **Important Assumptions!**
   1. **The name of the hive table created will be the name of the .csv file**
   2. **All types will be assumed to be strings**
   3. **A partition will be created based off the year/month/day of when you UPLOADED the data. Thus if you run the script more than once on a single day, it will overwrite any other data you uploaded that day. Uploads from different days will not be overwritten**

**The configuration file:**

#This is a sample configuration file, lines starting with '#' are comments. Lines not starting with a '#' are structured like so -> parameter = value, you will need to alter the values to match your needs. Please see the 'Hdfs\_load Documentation.docx' file for more information

[LocalPaths]

#This is the parent directory containing all of your .csv files on your local machine

**local\_dir: /Users/sshuster/Documents/Common\_Data\_Platform\_Challenge\_Team/transaction\_temp**

[RemotePaths]

#The server to connect to

**server: pl1rhd402.internal.edmunds.com**

**username=sshuster**

**password=**

#Do not change

base\_remote=/misc/%(username)s

[HDFSLocation]

#This is the folder on HDFS where your hive tables will reside. NOTE you will need to contact the DWH team to have a folder created for your team as otherwise you will not have permission to write to a folder

**hdfs\_base\_folder: /modeled-data**

[Hive]

#Set equal to True if you want to create the hive tables, otherwise False

**create\_tables: True**

#Set equal to True if you want to overwrite existing tables, otherwise False (ONLY SET TO TRUE IF YOU WANT TO DELETE ALL EXISTING DATA!!)

**overwrite\_existing\_hive: False**

#The Delimiter of your csv files

**delimiter = ^**

#The string denoting missing values (will be replaced with "")

**missing\_value = NA**

*General Overview:*

In the example file above you will notice that lines starting with a ‘#’ are comments and are just there to help explain the following settings. All other lines are in the form:

<parameter> = <value>

e.g.

missing\_value = NA

*What you will definitely need to change:*

1. First change the value of local\_dir to be that of the parent directory containing all of your .csv’s for that particular job. For example in the sample case above I have specified the absolute path of the directory where my .csv’s are located. In my case the folder ‘transaction\_temp’ contains 7 .csv’s corresponding to the transaction tables.
2. Then change the value of the username and password to whatever your LDAP credentials are
3. Finally change the values of delimiter and missing\_value to be whatever they are in the corresponding .csv files you are working with. NOTE that you should NOT put the values in quotations.

*What the other stuff is and when you will need to change it*

1. The server keyword under the [RemotePaths] section specifies which hadoop cluster you will be connecting to. There should not be any reasons to change this from its current value
2. The ‘hdfs\_base\_folder’ parameter specifies where your hive tables will live on the hadoop distributed file system. This is owned by the data warehouse group and we will let you know if this ever needs to be changed for any reason. For now, this value should be ‘/modeled-data’
3. The ‘create\_tables’ can be set to False after the first time you run the script. But there is no harm in always leaving it set to True
4. The ‘overwrite\_existing\_hive’ should only be changed if you would like to delete an existing table. This should be set to True if you have changed the columns or types of the data and thus everything loaded in prior is now obsolete.