**Command Line**

* pwd
* Root directory - /
* absolute paths - starts with / —
* relative paths - doesn’t start with / — and search for the directory in current directory
* whoami — check username
* cd ~ — directly takes us to current user’s home directory
* mkdir <dirname>
* mkdir -v <dirname> — modifying behaviour of commands - here we have used ‘v’ to turn on verbose
* -help after the command to get help for any command (2 dashes are not without space)
* ls -l
* rmdir
* touch test.txt
* echo “some string”
* echo “some string” > test.txt —> redirects the stdout to the file ‘test’
* nano <filename> - command line text editor
* file permissions - (owner, group, everyone) - (read, write, execute)
* Here are the combinations and their corresponding digits:
* --- : No permissions; corresponds to 0
* --x : Execute only permission; corresponds to 1
* -w- : Write only permissions; corresponds to 2
* -wx : Write and execute permissions; corresponds to 3
* r-- : Read only permissions; corresponds to 4
* r-x : Read and execute permissions; corresponds to 5
* rw- : Read and write permissions; corresponds to 6
* rwx : Read, write, and execute permissions; corresponds to 7
* stat <filenam> - details about the file
* chmod <octal code> filename
* mv <filename> <destination folder>
* cp <filename> <filename>
* rm <filenam>
* unzip <filenam>
* wc <filename> - count of lines in a file

**BASH**

/home/dq$ FOOD="Shrimp gumbo"

/home/dq$ $FOOD

bash: Shrimp: command not found

/home/dq$ echo $FOOD

Shrimp gumbo

/home/dq$ export FOOD="Chicken and waffles"

/home/dq$ python

Python 3.4.3 (default, Nov 17 2016, 01:08:31)

[GCC 4.8.4] on linux

Type "help", "copyright", "credits" or "license" for more information.

>>> import os

>>> print(os.environ["FOOD"])

Chicken and waffles

>>> exit()

/home/dq$ echo $FOOD

Chicken and waffles

/home/dq$ /usr/bin/python

Python 2.7.6 (default, Oct 26 2016, 20:30:19)

/home/dq$ echo $PATH

**— combining flags**

ls -a -l ——————> ls -al

longer flags with double dash (- -) without space

———

Create a virtual environment

virtualenv -p /usr/bin/python3 python3

virtualenv  python111

——

activate a virtual environment

source python3/bin/activate

deactivate a virtual environment

deactivate

—

pip freeze (gives all installed packages)

**CSVKit**

* csvstack: for stacking rows from multiple CSV files.
* csvlook: renders CSV in pretty table format.
* csvcut: for selecting specific columns from a CSV file.
* csvstat: for calculating descriptive statistics for some or all columns.
* csvgrep: for filtering tabular data using specific criteria.

**Examples**:

1. csvstack -n origin -g 1,2,3 file1.csv file2.csv file3.csv > final.csv
2. head -n 10 Combined\_hud.csv | csvlook
3. csvcut -n Combined\_hud.csv ——> displays all column names with an indentifier
4. csvcut -c 1 Combined\_hud.csv | head -n 10 ——> will output first column (only 10 values, since the output is pipelined with head)
5. csvstat --mean Combined\_hud.csv —> with no flag all stats are calculated
6. csvcut -c 2 Combined\_hud.csv | csvstat --max —> mean of just the second column
7. csvgrep -c <colindex> -m <pattern> <filename> —> a regex can be specified with -r flag instead on -m flag —> (-i flag for getting all rows where pattern doesn’t match)

**PIPELINING**

* Override - /home/dq$ echo "99 bottles of beer on the wall..." > beer.txt
* Append - /home/dq$ echo "Take one down, pass it around, 98 bottles of beer on the

 wall..." >> beer.txt

* Redirecting from a file -
  1. sort < <filename>
* GREP
  1. grep <string in quotes> <file name of names>
  2. Wildcard -
     1. ? — matches with any character
     2. \* — matches with any character any number of times (including 0)
     3. | — used for pipelining (output of one command fed as an input for next command
* cat — prints contents of a file
* tail -n <int> — prints last lines of a file
* head -n <int> — print starting n lines of a file
* && — runs the first then second command
* \ — used for escaping chars in bash