Data Processing in Shell

Downloading Data on the Command Line

`curl` (Get data from urls)

- `curl -O https://websitename.com/datafilename.txt` (Get data and save with url name)
- `curl -o renamedatafile.txt https://websitename.com/datafilename.txt` (Get data with diferent file name)
- `curl -O https://websitename.com/datafilename*.txt` (Get all datafiles)
- `curl -O https://websitename.com/datafilename[0-100].txt` (Get all datafiles from 0 to 100)
- `curl -O https://websitename.com/datafilename[0-100:10].txt` (Get all datafiles from 0 to 100 10 by 10)
- `curl -L https://websitename.com/datafilename[0-100:10].txt` (Get datafile if redirect occours)

`wget` (Multi purpose curl)

- `wget -b -c https://websitename.com/datafilename.txt` (Get data -b run in background, -c lets download partiaçly if error)
- `wget -i allurls.txt` (Get data from all urls inside allurls.txt file)
- `wget --wait=2.5 -i urllist.txt` (Get data from urls inside urlList and wait 2.5sec between urls)

Data Cleaning and Munging on the Command Line

This is part of csvkit (pip install csvkit)

`in2csv` (convert files into csv)

- `in2csv somefile.xlsx > somefile.csv` (Create csv file from xlsx first sheet)
- `in2csv somefile.xlsx` (Print first sheet of xlsx)
- `in2csv -n somefile.xlsx` (Print all sheet names inside file)

`in2csv somefile.xlsx --sheet "sheet_name1" > somefile.csv` (Create csv file from sheet_name1 inside from xlsx)

`csvlook` (Preview data in csv)

`csvlook somefile.csv`

`csvstat` (Print descriptive sumarry statistics ‘similar to describePandas’)

`csvstat somefile.csv`

`csvcut` (Filter data by columns)

`csvcut -n somefile.csv` (Print all columns names inside csv)

`csvcut -c 1 somefile.csv` (Print the first columns inside csv)

`csvcut -c "ID" somefile.csv` (Print the “ID” column inside csv)

`csvcut -c "ID","Name" somefile.csv` (Print the “ID” column and “Name” inside csv)

`csvgrep` (Filter data by row value)

csvgrep -c “trackID” -m 5RCP875 somefile.csv (Print csv where trackID == 5RCP875)

`csvstack` (Stack csv files ‘concat’)

`csvstack somefile1.csv somefile2.csv > somefile1and2.csv` (Stack this two csv files into one)

`csvstack -g "From1","From2" somefile1.csv somefile2.csv > somefile1and2.csv` (Stack this two csv files into one but create a source column ‘group’)

`csvstack -g "From1","From2" -n "source" somefile1.csv somefile2.csv > somefile1and2.csv` (Stack two csv files into one and create a source column named ‘source’)

Chaining Commands

`;` More than one command in the same line

`&&` And operator (If right dont suceced dont do left)

`>` Redirect (Store ouput from right to left)

`|` Pipe operator (Right is input for left)

Database Operations on the Command Line

`sql2csv` (Convert dabasequery to csv)

- `sql2csv --db “sqlite:///sampledatabase.db” --query “SELECT * FROM table_name” > samplefile.csv` (- - db =database conection, query)

`csvsql` (Manipulate csv file with sql queries and upload data to csv)

- `csvsql --query “SELECT * FROM table_name LIMIT 1”  samplefile.csv` (Select first row from samplefile.csv)
- `csvsql --query “SELECT * FROM file_a as a INNER JOIN file_b as b ON a.id = b.id ” file_a.csv file_b.csv` (Inner join example with csv files
- `csvsql --query $variable samplefile.csv` (Using variable on csvsql)
- `csvsql --db “sqlite:///sampledatabase.db”  --insert samplefile.csv` (Insert csv as table into DB)

Data Pipeline on the Command Line

`python`

- `python samplefile.py` (Run python file)
- `python --version` (Check Python version)
- `echo “print('Hello world')” > samplefile.py` (Create pythonfile inside terminal)

`pip`

- `pip install package` (Install package)
- `pip install package==0.19.2` (Install specific package version)
- `pip install --upgrade package` (Upgrade Package)
- `pip install -r requirements.txt` (Install a list of packages inside file)
- `pip list` (Show all packages installed)

cron

- `crontab -l` (Central file to keep track of cronjobs)
- `echo “* * * * * python create_model.py” | crontab` (Add a comand to crontab. Edit with vim also avaliable)

```
.---------------- minute (0 - 59)
|  .------------- hour (0 - 23)
|  |  .---------- day of month (1 - 31)
|  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
|  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed ...
|  |  |  |  |
*  *  *  *  * command-to-be-executed
```

`15 * * * * python model.py` (Every hour, on the 15 minute of an hour? (e.g. 12:15 PM, 1:15 AM, 2:15 AM, etc))

`* * * * * python model.py` (Runs every minute)