L04: MapReduce Design Patterns: Joins

ANLY 502: Massive Data Fundamentals

Simson Garfinkel & Marck Vaisman

February 6, 2017

Outline for today's class

Questions

- Course, homework, lab, AWS, EMR, mrJob, etc.

A2 Homework Review

Joins in MapReduce

Lab # 1 (7:15pm - 7:45pm)

Total Order Sort

Lab # 2 (8:30pm - 9:00pm)

L04: Homework Review

ANLY 502: Massive Data Fundamentals

Simson Garfinkel & Marck Vaisman

January 30, 2017

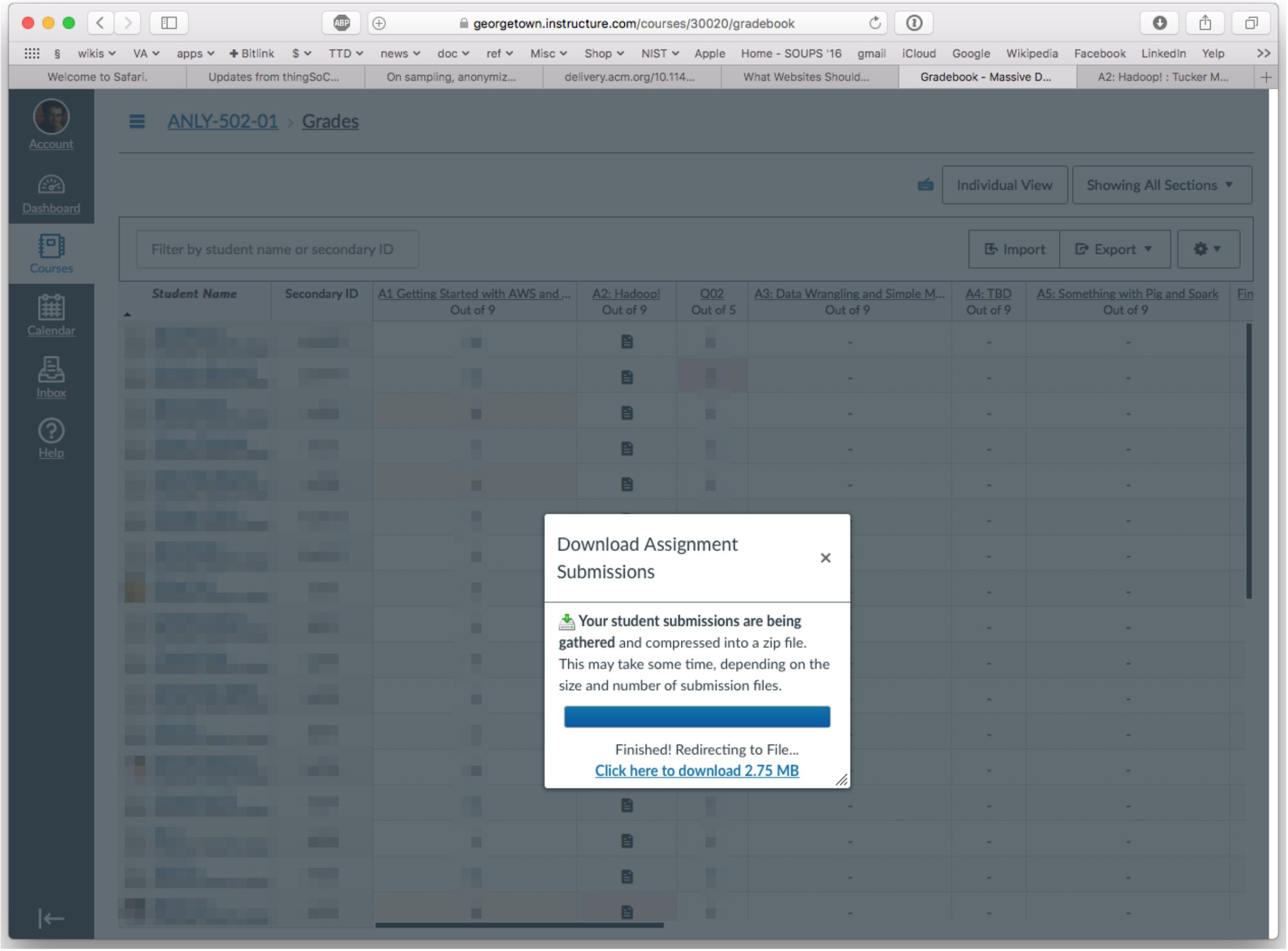
Homework A2 Redux

Key issues:

- Submission time — many students started this project set at the last moment.
- Line endings (Unix, Windows & Mac)
- "screen"
- git conflicts
- Late homework policy

Submission Time

We download all of the A2 homework as a ZIP file



The file!

```
[Dance ~/Downloads 13:32:12]$ ls -l submissions.zip
-rw-r--r--@ 1 simsong  staff  2883699 Feb  5 13:25 submissions.zip
[Dance ~/Downloads 13:32:14]$
```

The contents:

```
[Dance ~/Downloads/submissions 13:31:38]$ ls -l
total 2908
-rw-r--r--@ 1 simsong  staff    9892 Feb  5  2017 [redacted]7_924930_aa1603-2.zip
-rw-r--r--@ 1 simsong  staff   58561 Feb  5  2017 [redacted]_183212_925632_amb497-2.zip
-rw-r--r--@ 1 simsong  staff  322489 Feb  5  2017 [redacted]120_925438_zc104-2.zip
-rw-r--r--@ 1 simsong  staff   58720 Feb  5  2017 [redacted]_184051_925607_krc75-2.zip
-rw-r--r--@ 1 simsong  staff   95954 Feb  5  2017 [redacted]_186035_925354_yc704-2-1.zip
-rw-r--r--@ 1 simsong  staff   61744 Feb  5  2017 [redacted]14773_925475_mc2153-2.zip
-rw-r--r--@ 1 simsong  staff  101732 Feb  5  2017 [redacted]91_924091_jmc511-2.zip
-rw-r--r--@ 1 simsong  staff  214443 Feb  5  2017 [redacted]193_925565_zf38-2-1.zip
-rw-r--r--@ 1 simsong  staff   44270 Feb  5  2017 [redacted]_late_107472_925674_ljf30-2.zip
-rw-r--r--@ 1 simsong  staff   54478 Feb  5  2017 [redacted]7870_925454_tg550-2.zip
-rw-r--r--@ 1 simsong  staff   48325 Feb  5  2017 [redacted]_168091_925441_th762-2.zip
-rw-r--r--@ 1 simsong  staff    7032 Feb  5  2017 [redacted]_925618_jj820-2.zip
-rw-r--r--@ 1 simsong  staff   48438 Feb  5  2017 [redacted]h_71142_925653_hk754-2.zip
-rw-r--r--@ 1 simsong  staff    7467 Feb  5  2017 [redacted]87976_924341_ckl39-2.zip
-rw-r--r--@ 1 simsong  staff  154289 Feb  5  2017 [redacted]_925289_wl521-2-1.zip
-rw-r--r--@ 1 simsong  staff    6100 Feb  5  2017 [redacted]_181462_925662_zl118-2-1.zip
-rw-r--r--@ 1 simsong  staff  156490 Feb  5  2017 [redacted]25241_yl808-2.zip
-rw-r--r--@ 1 simsong  staff   82652 Feb  5  2017 [redacted]01_925511_jm3055-2-1.zip
-rw-r--r--@ 1 simsong  staff   45220 Feb  5  2017 [redacted]ate_171441_925673_tsm47-2-1.zip
-rw-r--r--@ 1 simsong  staff   61178 Feb  5  2017 [redacted]_18646_925643_an692-2.zip
-rw-r--r--@ 1 simsong  staff   16295 Feb  5  2017 [redacted]_late_162000_926317_sp1191-2.zip
-rw-r--r--@ 1 simsong  staff   69752 Feb  5  2017 [redacted]ate_102138_925660_ks1537-2-1.zip
-rw-r--r--@ 1 simsong  staff   64982 Feb  5  2017 [redacted]1648_925629_ds1534-2.zip
-rw-r--r--@ 1 simsong  staff   52043 Feb  5  2017 [redacted]h_59664_925433_ers92-2.zip
-rw-r--r--@ 1 simsong  staff   52835 Feb  5  2017 [redacted]8_925637_cs1692-2-5.zip
-rw-r--r--@ 1 simsong  staff   32165 Feb  5  2017 [redacted]te_90604_925659_js4479-2.zip
-rw-r--r--@ 1 simsong  staff   89554 Feb  5  2017 [redacted]_925642_ls1229-2.zip
-rw-r--r--@ 1 simsong  staff   69081 Feb  5  2017 [redacted]n_111008_925421_lw698-2.zip
-rw-r--r--@ 1 simsong  staff   31963 Feb  5  2017 [redacted]91_925469_yw485-2.zip
-rw-r--r--@ 1 simsong  staff   84939 Feb  5  2017 [redacted]e_150867_925664_rw848-2-3.zip
-rw-r--r--@ 1 simsong  staff   35851 Feb  5  2017 [redacted]_183844_925666_yw486-2.zip
-rw-r--r--@ 1 simsong  staff   71925 Feb  5  2017 [redacted]18_925622_jx94-2-1.zip
-rw-r--r--@ 1 simsong  staff    8170 Feb  5  2017 [redacted]2519_925654_cz211-2.zip
-rw-r--r--@ 1 simsong  staff  542790 Feb  5  2017 [redacted]2915_924927_dz181-2.zip
-rw-r--r--@ 1 simsong  staff   60206 Feb  5  2017 [redacted]ate_182815_926029_yz539-2.zip
[Dance ~/Downloads/submissions 13:31:39]$ cd ..
[Dance ~/Downloads 13:32:12]$ ls -l submissions.zip
```

Notice:

- Wildly different sizes
- 35 submissions!

The submission script guarantees that the ZIP file name conventions are correct.

Here is one student's submission (netid changed to sg1224):

```
$ unzip -l simsongarfinkel_182815_926029_sg1224-2.zip
Archive:  simsongarfinkel_late_182815_926029_sg1224-2.zip
  Length      Date    Time    Name
---------  ---------- -----   ----
       43  02-01-2017 14:20   sg1224-2/q5_combiner.py
    12574  02-04-2017 20:10   sg1224-2/q2_plot.pdf
     2285  02-04-2017 18:55   sg1224-2/q2_results.txt
      362  02-01-2017 14:51   sg1224-2/q2_mapper.py
     3816  02-03-2017 17:48   sg1224-2/q4_run.py
       43  02-01-2017 14:20   sg1224-2/q5_mapper.py
      555  02-03-2017 17:45   sg1224-2/q4_reducer.py
      612  02-03-2017 17:45   sg1224-2/q4_mapper.py
       43  02-01-2017 14:20   sg1224-2/q5_reducer.py
      204  02-04-2017 19:31   sg1224-2/q4_monthly.txt
      498  02-04-2017 18:58   sg1224-2/q3_results.txt
     1122  02-04-2017 19:26   sg1224-2/q1.txt
    48248  02-04-2017 20:10   sg1224-2/q2_plot.png
      555  02-01-2017 14:51   sg1224-2/q2_reducer.py
     3804  02-01-2017 14:20   sg1224-2/q2_python.py
      668  02-01-2017 14:20   sg1224-2/Makefile
---------                     -------
    75432                     16 files
```

Let's look at results file!:

```
[Dance ~/Downloads/submissions 20:32:09]$ cat XXXXX/q2_results.txt
#
# Your results go into this file.
# They will be written as JSON objects
{"input:": "s3://gu-anly502/A1/quazyilx1.txt", "nodes": 2, "seconds": 373.0450801849365,
"output": "output-2017-02-03T01-26-26", "node": "ip-172-31-4-219", "date": 1486085561.7805748}
{"input:": "s3://gu-anly502/A1/quazyilx2.txt", "output": "output-2017-02-03T01-49-40", "date":
1486087436.5008893, "node": "ip-172-31-4-219", "seconds": 853.7396121025085, "nodes": 2}
{"output": "output-2017-02-03T21-16-15", "input:": "s3://gu-anly502/A1/quazyilx3.txt", "nodes":
2, "date": 1486158450.2826307, "seconds": 1873.1529202461243, "node": "ip-172-31-35-156"}
{"seconds": 181.30789136886597, "nodes": 3, "date": 1486160269.5000417, "output":
"output-2017-02-03T22-14-46", "input:": "s3://gu-anly502/A1/quazyilx1.txt", "node":
"ip-172-31-35-156"}
{"date": 1486161229.3729312, "nodes": 3, "output": "output-2017-02-03T22-23-38", "seconds":
609.3414239883423, "node": "ip-172-31-35-156", "input:": "s3://gu-anly502/A1/quazyilx2.txt"}
{"output": "output-2017-02-03T22-35-15", "node": "ip-172-31-35-156", "input:": "s3://gu-anly502/
A1/quazyilx3.txt", "date": 1486162797.1595008, "seconds": 1479.5804240703583, "nodes": 3}
{"nodes": 4, "node": "ip-172-31-59-2", "output": "output-2017-02-03T23-54-56", "input:": "s3://
gu-anly502/A1/quazyilx1.txt", "date": 1486166241.9708796, "seconds": 143.95424008369446}
{"nodes": 4, "input:": "s3://gu-anly502/A1/quazyilx2.txt", "seconds": 523.5885500907898, "node":
"ip-172-31-59-2", "output": "output-2017-02-04T00-03-17", "date": 1486167122.6089153}
{"node": "ip-172-31-59-2", "input:": "s3://gu-anly502/A1/quazyilx3.txt", "date":
1486168136.8667717, "seconds": 911.0414657592773, "output": "output-2017-02-04T00-13-43",
"nodes": 4}
{"nodes": 6, "date": 1486169055.7561696, "seconds": 102.70592164993286, "input:": "s3://gu-
anly502/A1/quazyilx1.txt", "node": "ip-172-31-59-2", "output": "output-2017-02-04T00-42-31"}
{"nodes": 6, "date": 1486169548.9940424, "output": "output-2017-02-04T00-47-20", "input:":
"s3://gu-anly502/A1/quazyilx2.txt", "node": "ip-172-31-59-2", "seconds": 306.4689152240753}
{"date": 1486170250.0787191, "input:": "s3://gu-anly502/A1/quazyilx3.txt", "seconds":
612.0978214740753, "output": "output-2017-02-04T00-53-56", "node": "ip-172-31-59-2", "nodes": 6}
[Dance ~/Downloads/submissions 20:32:14]$
```

Let's look at a results file:

```
[Dance ~/Downloads/submissions 20:32:09]$ cat XXXXX/q2_results.txt
#
# Your results go into this file.
# They will be written as JSON objects
{"input:": "s3://gu-anly502/A1/quazyilx1.txt", "nodes": 2, "seconds": 373.0450801849365,
"output": "output-2017-02-03T01-26-26", "node": "ip-172-31-4-219", "date": 1486085561.7805748}
{"input:": "s3://gu-anly502/A1/quazyilx2.txt", "output": "output-2017-02-03T01-49-40", "date":
1486087436.5008893, "node": "ip-172-31-4-219", "seconds": 853.7396121025085, "nodes": 2}
{"output": "output-2017-02-03T21-16-15", "input:": "s3://gu-anly502/A1/quazyilx3.txt", "nodes":
2, "date": 1486158450.2826307, "seconds": 1873.1529202461243, "node": "ip-172-31-35-156"}
{"seconds": 181.30789136886597, "nodes": 3, "date": 1486160269.5000417, "output":
"output-2017-02-03T22-14-46", "input:": "s3://gu-anly502/A1/quazyilx1.txt", "node":
"ip-172-31-35-156"}
{"date": 1486161229.3729312, "nodes": 3, "output": "output-2017-02-03T22-23-38", "seconds":
609.3414239883423, "node": "ip-172-31-35-156", "input:": "s3://gu-anly502/A1/quazyilx2.txt"}
{"output": "output-2017-02-03T22-35-15", "node": "ip-172-31-35-156", "input:": "s3://gu-anly502/
A1/quazyilx3.txt", "date": 1486162797.1595008, "seconds": 1479.5804240703583, "nodes": 3}
{"nodes": 4, "node": "ip-172-31-59-2", "output": "output-2017-02-03T23-54-56", "input:": "s3://
gu-anly502/A1/quazyilx1.txt", "date": 1486166241.9708796, "seconds": 143.95424008369446}
{"nodes": 4, "input:": "s3://gu-anly502/A1/quazyilx2.txt", "seconds": 523.5885500907898, "node":
"ip-172-31-59-2", "output": "output-2017-02-04T00-03-17", "date": 1486167122.6089153}
{"node": "ip-172-31-59-2", "input:": "s3://gu-anly502/A1/quazyilx3.txt", "date":
1486168136.8667717, "seconds": 911.0414657592773, "output": "output-2017-02-04T00-13-43",
"nodes": 4}
{"nodes": 6, "date": 1486169055.7561696, "seconds": 102.70592164993286, "input:": "s3://gu-
anly502/A1/quazyilx1.txt", "node": "ip-172-31-59-2", "output": "output-2017-02-04T00-42-31"}
{"nodes": 6, "date": 1486169548.9940424, "output": "output-2017-02-04T00-47-20", "input:":
"s3://gu-anly502/A1/quazyilx2.txt", "node": "ip-172-31-59-2", "seconds": 306.4689152240753}
{"date": 1486170250.0787191, "input:": "s3://gu-anly502/A1/quazyilx3.txt", "seconds":
612.0978214740753, "output": "output-2017-02-04T00-53-56", "node": "ip-172-31-59-2", "nodes": 6}
[Dance ~/Downloads/submissions 20:32:14]$
```

Let's graph this!

Some python

Always use **datetime** module for dates & times (if possible)

UNIX timestamp is # of seconds since January 1, 1970 GMT

To convert a Unix timestamp to a datetime:

```
In [47]: import datetime

In [48]: datetime.datetime.fromtimestamp(1486085561.7805748)
Out[48]: datetime.datetime(2017, 2, 2, 20, 32, 41, 780575)
```

Read each line of the results file, convert to a dictionary, and append to data[]:

```
import json
data = []
for line in open(infile,"rU"):
    if line[0]=='{':
        data.append( json.loads( line ) )
```

Extract an array of timestamps:

```
dates = [ datetime.datetime.fromtimestamp( d['date']) for d in data]
```

It worked great until…

```
/Users/simsong/Downloads/submissions/*****/q3_results.txt

Traceback (most recent call last):
  File "dateplot.py", line 88, in <module>
    timestamp_cdf(args.infiles,args.outfile)
  File "dateplot.py", line 39, in timestamp_cdf
    data.append(json.loads(line))
  File "/opt/local/Library/Frameworks/Python.framework/Versions/3.5/
lib/python3.5/json/__init__.py", line 319, in loads
    return _default_decoder.decode(s)
  File "/opt/local/Library/Frameworks/Python.framework/Versions/3.5/
lib/python3.5/json/decoder.py", line 339, in decode
    obj, end = self.raw_decode(s, idx=_w(s, 0).end())
  File "/opt/local/Library/Frameworks/Python.framework/Versions/3.5/
lib/python3.5/json/decoder.py", line 357, in raw_decode
    raise JSONDecodeError("Expecting value", s, err.value) from None
json.decoder.JSONDecodeError: Expecting value: line 1 column 40
(char 39)
```

Use 'cat' to look at the problem file:

```
json.decoder.JSONDecodeError: Expecting value: line 1 column 40 (char 39)
$ cat q3_results.txt
{"node": "ip-172-31-37-198", "input:": "quazyilx3.txt", "output":
"output-2017-02-03T17-49-58", "date": 1486144771.9010673, "nodes": 2,
"seconds": 571.320051908493}
{"input:": "quazyilx3.txt", "seconds": XXXXXXX, "date":
1486147081.03975591, "nodes": XXXXXXXX, "output":
"output-2017-02-03T18-38-01”, "node": "ip-172-31-37-198"}
{"nodes": 9, "output": "output-2017-02-03T19-01-40", "date":
1486148683.8140492, "seconds": 180.88321089744568, "node":
"ip-172-31-37-198", "input:": "quazyilx3.txt"}
```

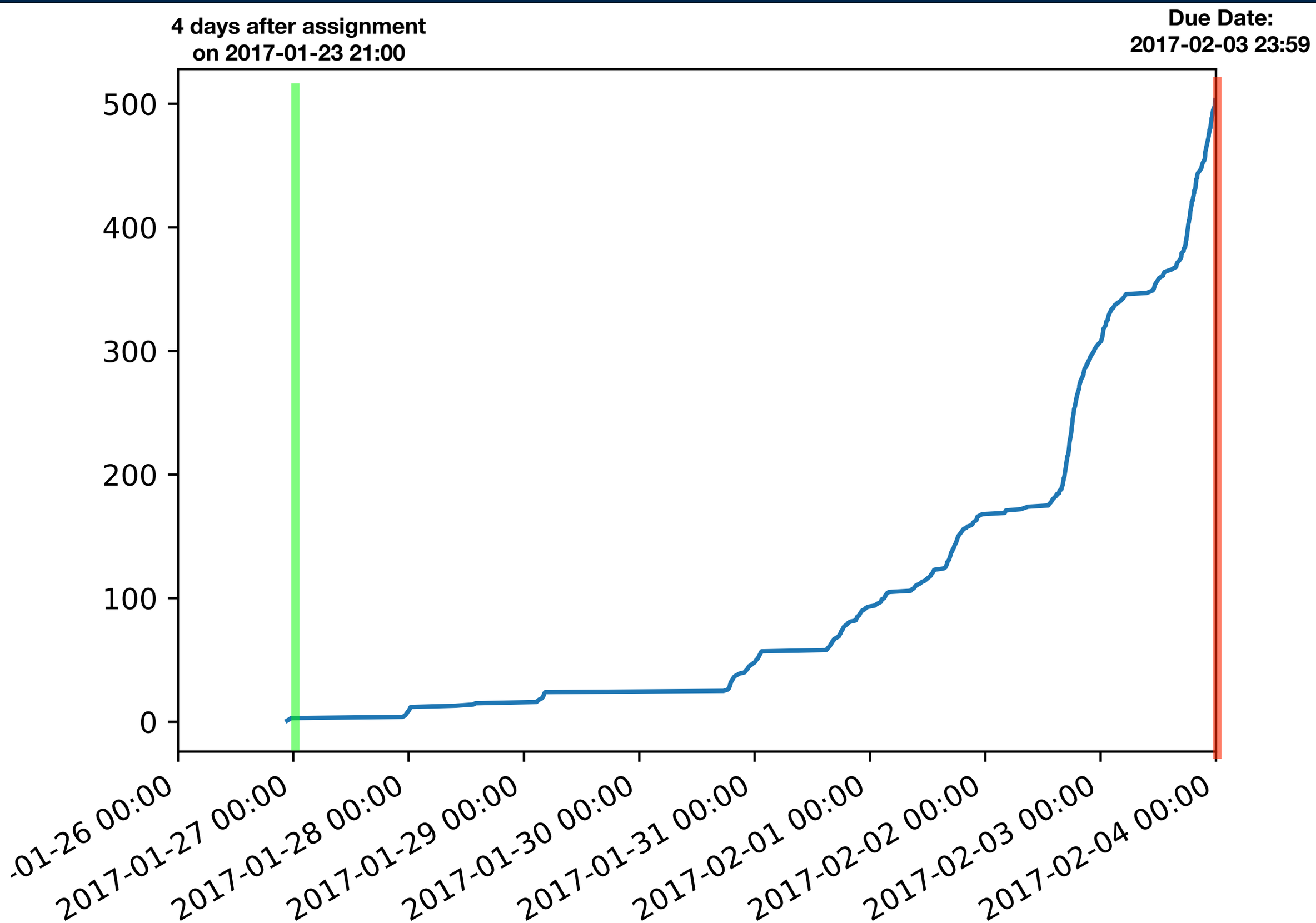
This isn't valid JSON. And one of the quotes are smart quotes too!

```
$ cat -v q3_results.txt
{"node": "ip-172-31-37-198", "input:": "quazyilx3.txt", "output":
"output-2017-02-03T17-49-58", "date": 1486144771.9010673, "nodes": 2,
"seconds": 571.320051908493}
{"input:": "quazyilx3.txt", "seconds": XXXXXXX, "date":
1486147081.03975591, "nodes": XXXXXXXX, "output":
"output-2017-02-03T18-38-01?M-^@M-^], "node": "ip-172-31-37-198"}
{"nodes": 9, "output": "output-2017-02-03T19-01-40", "date":
1486148683.8140492, "seconds": 180.88321089744568, "node":
"ip-172-31-37-198", "input:": "quazyilx3.txt"}
```

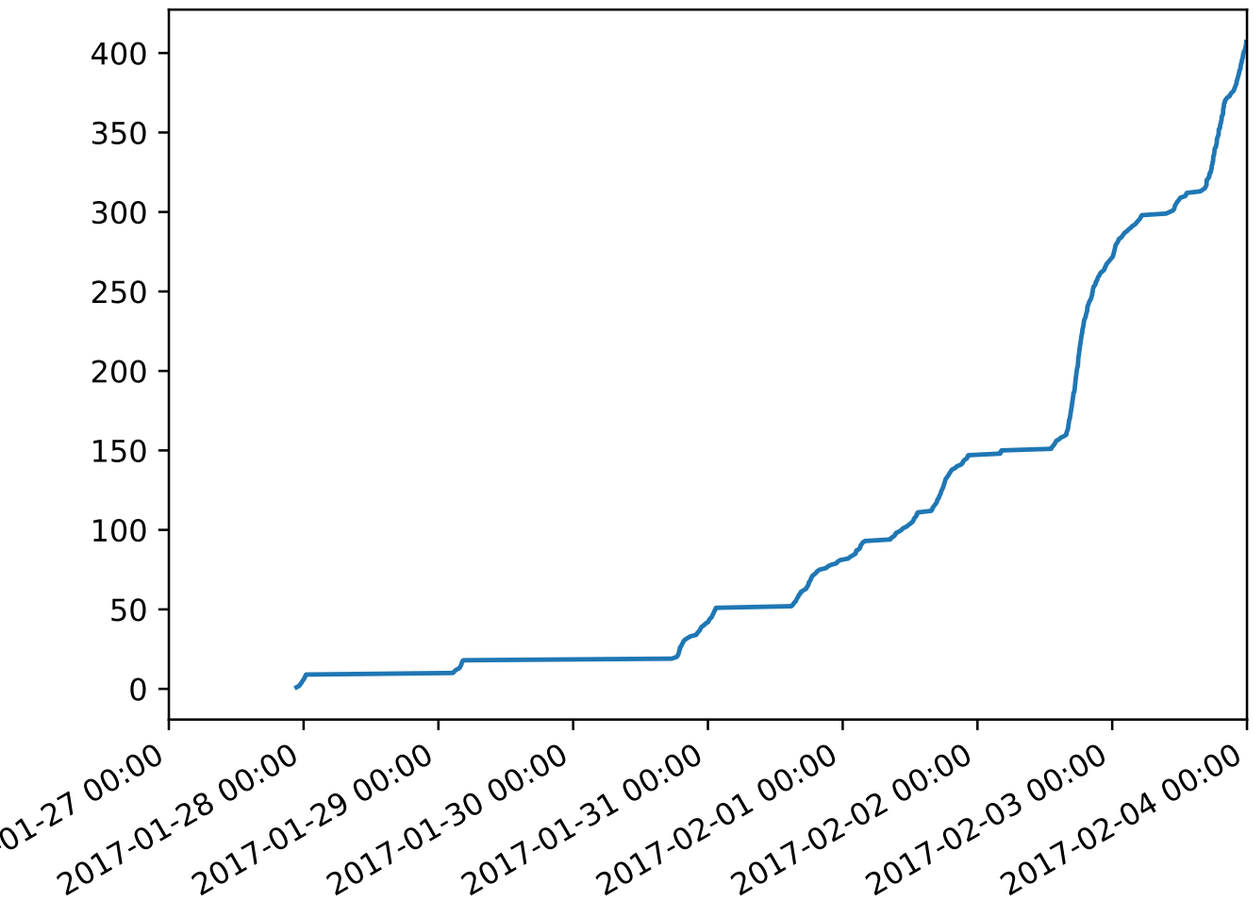
Reading results files with error handling:

```
# First create an array with all of the JSON objects...
for infile in infiles:
    for line in open(infile,"rU"):
        # only process lines beginning with JSON:
        if line[0]=='{':
            try:
                data.append(json.loads(line))
            except json.decoder.JSONDecodeError as e:
                print("  Invalid JSON in {}".format(infile))
```

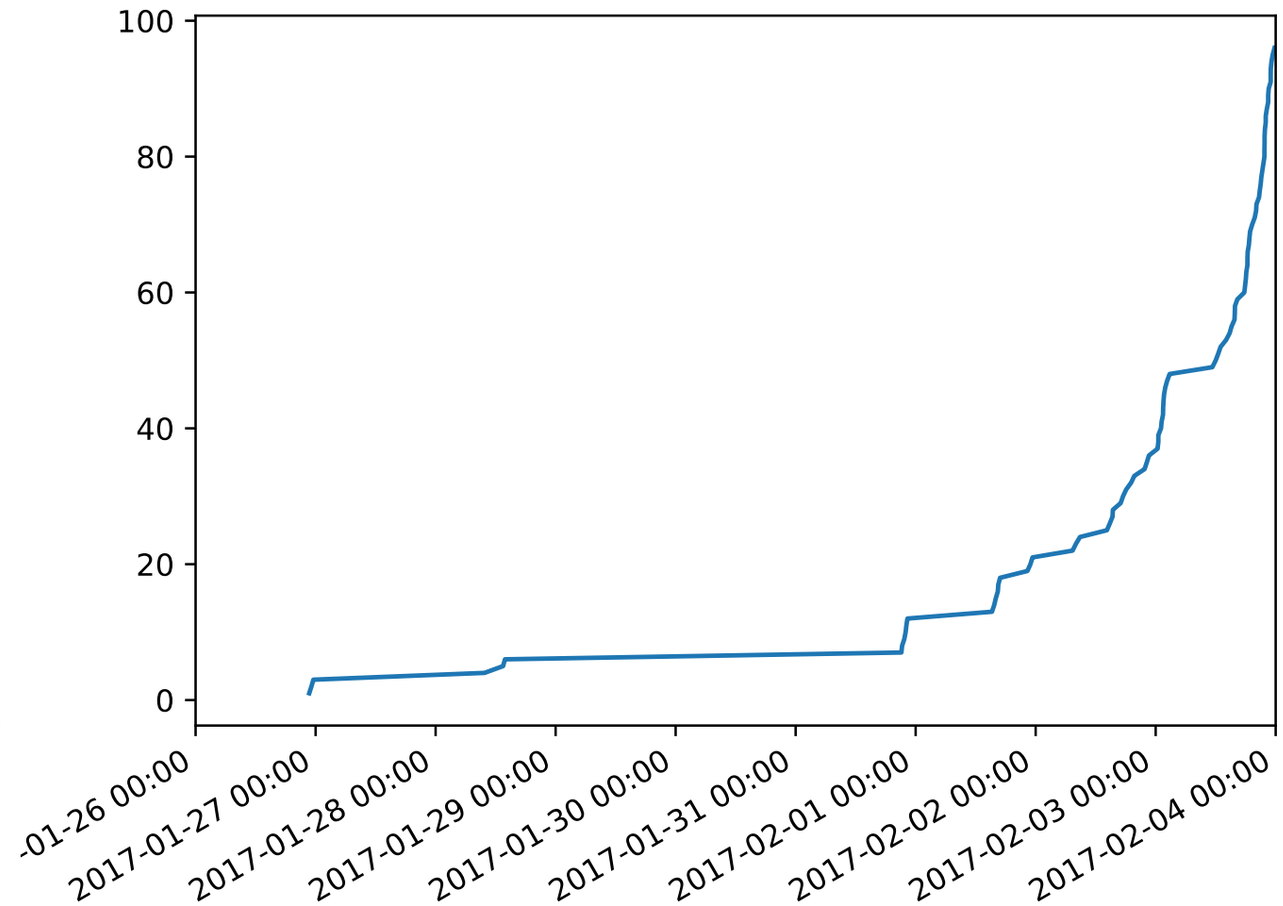
Graph of all of the timestamps!



Graph of Q2 vs. Q3



q2

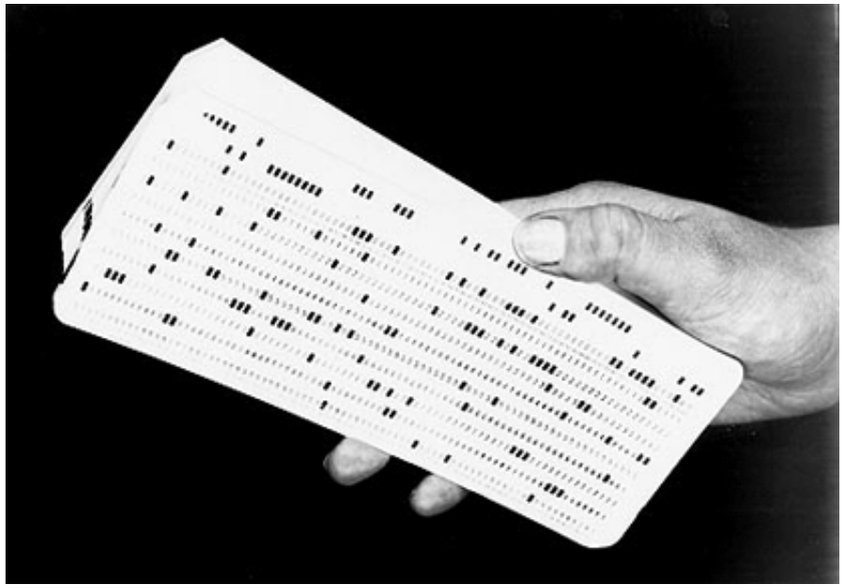


q3

Line Endings

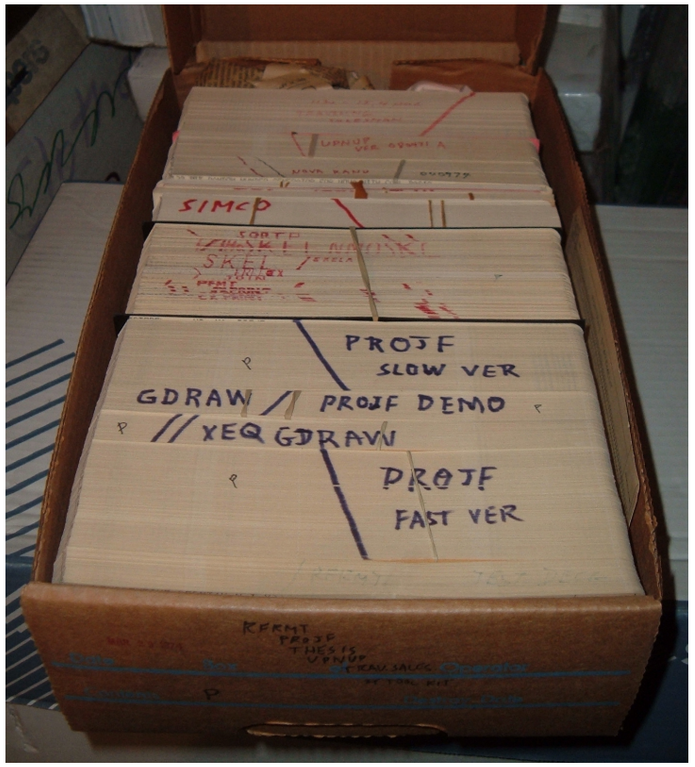
What are text files?

Originally: a stack of punch cards



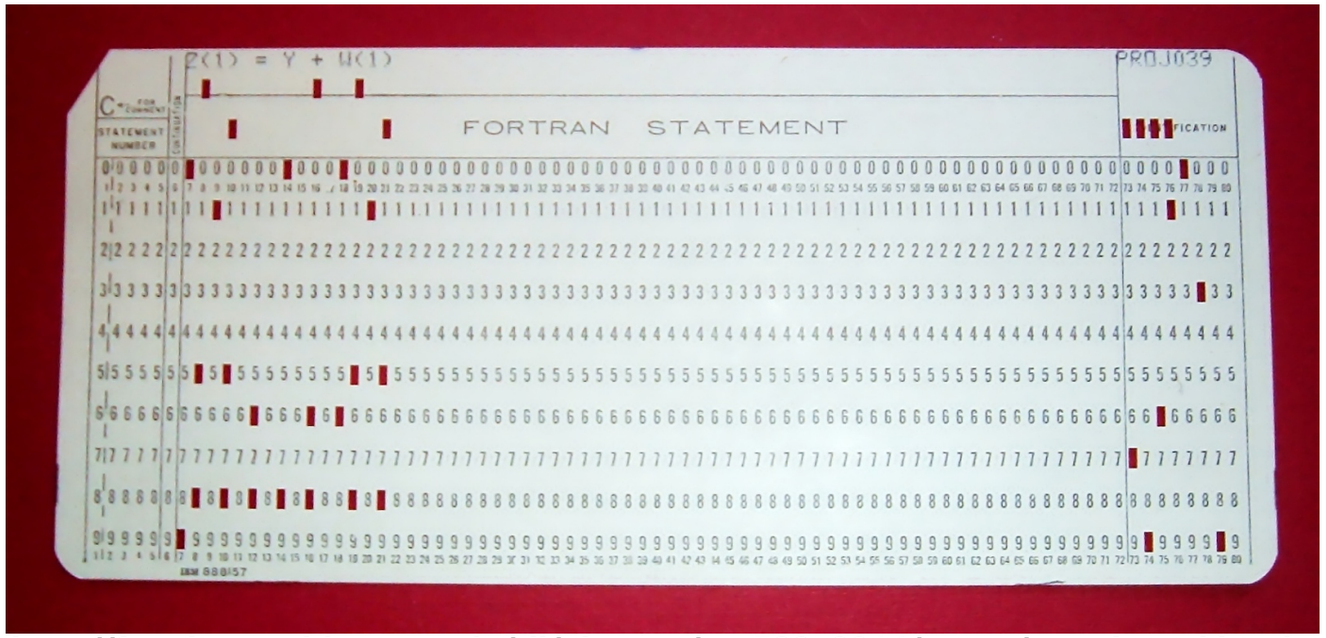
80 column punch cards

http://www.computerhistory.org/atchm/about-the-computer-history-museums-ibm-1401-machines/



http://faculty.washington.edu/rjl/classes/am583s2014/notes/punchcard.html

Each row is a character:



http://faculty.washington.edu/rjl/classes/am583s2014/notes/punchcard.html

In 1960, the Bell System created “ASCII”

American Standard Code for Information Interchange.

1963 — Teletype Corporation created the ASR33

https://en.wikipedia.org/wiki/Teletype_Model_33

ASCII was the dominant code from 1960 through 2000.

Each letter is described by a number.

"A" = 0010 00012 = 6510 = 01018 = 0x4116

"B" = 0010 00102 = 6610 = 01018 = 0x4116

Note:

'A' + 32 = 'a'

'A' - 64 = ^A

The decimal set:

0	nul	1	soh	2	stx	3	etx	4	eot	5	enq	6	ack	7	bel
8	bs	9	ht	10	nl	11	vt	12	np	13	cr	14	so	15	si
16	dle	17	dc1	18	dc2	19	dc3	20	dc4	21	nak	22	syn	23	etb
24	can	25	em	26	sub	27	esc	28	fs	29	gs	30	rs	31	us
32	sp	33	!	34	"	35	#	36	$	37	%	38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	\	93	]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124	|	125	}	126	~	127	del

Carriage Control Codes — aka "Control Codes" (The carriage is the thing that prints.)

8	BS	Back Space	— Move 1 character left
9	HT	Horizontal Tab	— Move to (8 - pos % 8) characters right
10	NL	Next Line	— Move carriage down one line
13	CR	Carriage Return	— Move carriage to the fight right

The decimal set:

0	nul	1	soh	2	stx	3	etx	4	eot	5	enq	6	ack	7	bel
8	bs	9	ht	10	nl	11	vt	12	np	13	cr	14	so	15	si
16	dle	17	dc1	18	dc2	19	dc3	20	dc4	21	nak	22	syn	23	etb
24	can	25	em	26	sub	27	esc	28	fs	29	gs	30	rs	31	us
32	sp	33	!	34	"	35	#	36	$	37	%	38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	\	93	]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124	|	125	}	126	~	127	del

There were two kinds of text files

Fixed record file

- Each record was group of N characters (e.g. 80)

Variable length record files

- Each record terminated with a special character.

UNIX (1971)	*new line* for the end of lines.
MSDOS (1980)	*carriage return, line feed.*
Macintosh (1984)	*carriage return.* (Only system 1 - 7)

Both MSDOS and Macintosh confused *logical representation* and *presentation.*

UNIX — *new line* represents the end of a line and the start of the next.

The way it is entered and printed depends on the program that's running.

MSDOS — Stored data in the file the way it was printed.

Macintosh — Stored data in the file the way it was

What this means for you

In this class we use \n as our end-of-line terminator.

If you edit text files with modern tools, you will be fine:

- EMACS
- vi / vim
- Textedit (Mac)

If you edit files with Notepad++ or another (windows) tool, be sure to ***set line mode to UNIX (NL).***

Reading a text file with Python — use Universal Line Ending:

```
for line in open("myfile.txt","rU"):
    sys.stdout.write(line)
```

—*Turns lines ending with \r, \r\n, or \n into \n*

Beware of smart quotes! Beware of smart dashes! Don't edit JSON!

screen

screen holds your session...

```
  Submitting tokens for job: job_1486318287298_0003
  Submitted application application_1486318287298_0003
  The url to track the job: http://ip-172-31-39-51.ec2.internal:20888/
proxy/application_1486318287298_0003/
  Running job: job_1486318287298_0003
  Job job_1486318287298_0003 running in uber mode : false
   map 0% reduce 0%
   map 1% reduce 0%
   map 2% reduce 0%
   map 3% reduce 0%
   map 4% reduce 0%
   map 5% reduce 0%
   map 6% reduce 0%
   map 7% reduce 0%
   map 8% reduce 0%
   map 9% reduce 0%
   map 10% reduce 0%
   map 11% reduce 0%
   map 12% reduce 0%
   map 13% reduce 0%
   map 14% reduce 0%
   map 15% reduce 0%
   map 16% reduce 0%
   map 17% reduce 0%
   map 18% reduce 0%
...
```

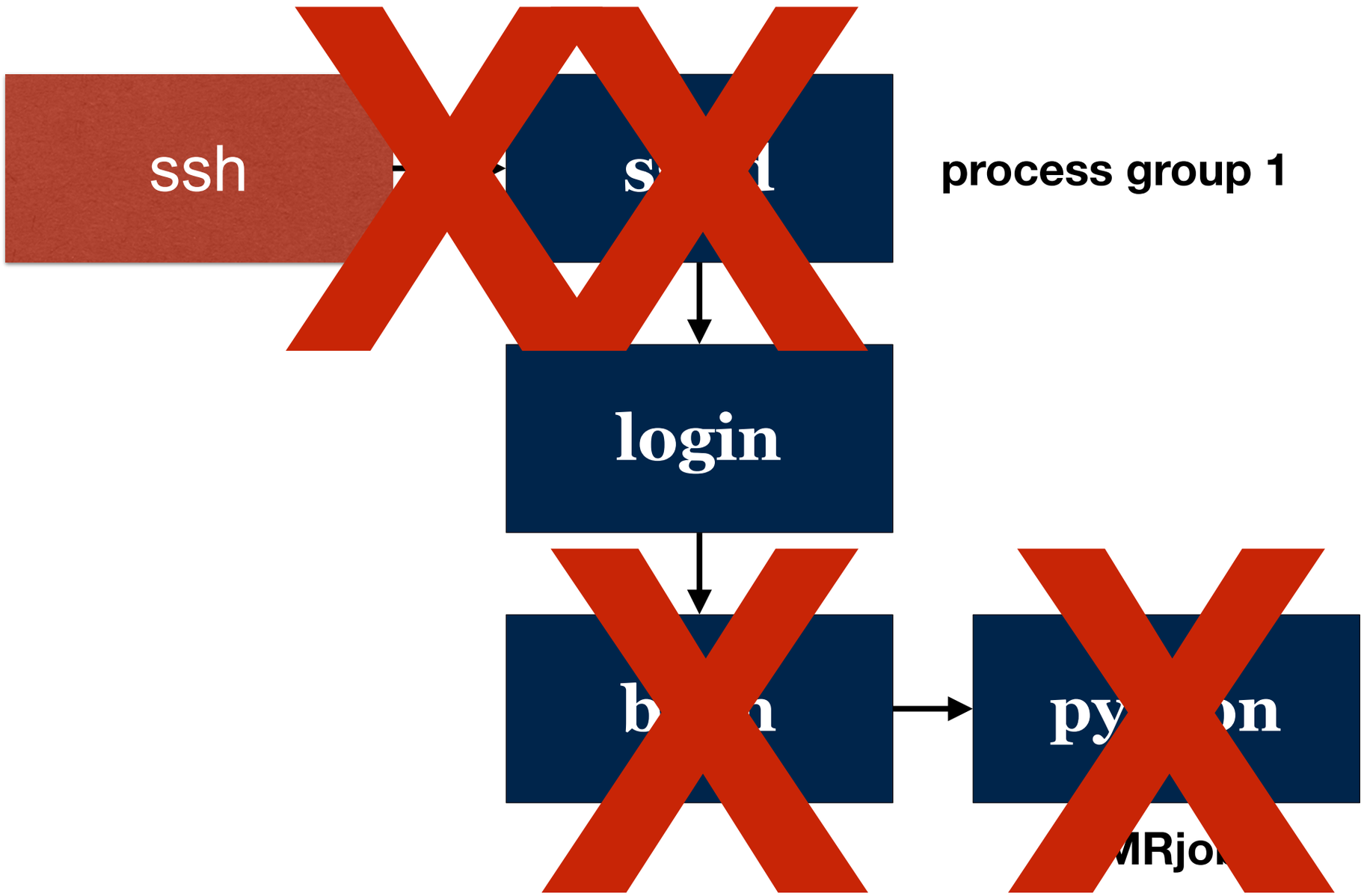
And then you turn off your laptop!

```
   map 100% reduce 22%
   map 100% reduce 25%
   map 100% reduce 31%
   map 100% reduce 39%
   map 100% reduce 44%
   map 100% reduce 45%
   map 100% reduce 67%
   map 100% reduce 68%
   map 100% reduce 69%
   map 100% reduce 70%
   map 100% reduce 71%
   map 100% reduce 72%
   map 100% reduce 73%
   map 100% reduce 74%
   map 100% reduce 75%
   map 100% reduce 76%
   map 100% reduce 77%
   map 100% reduce 78%
Timeout, server ec2-52-86-169-133.compute-1.amazonaws.com not
responding.
$
```

What happened

Your ssh connected to sushi, which ran login then bash.

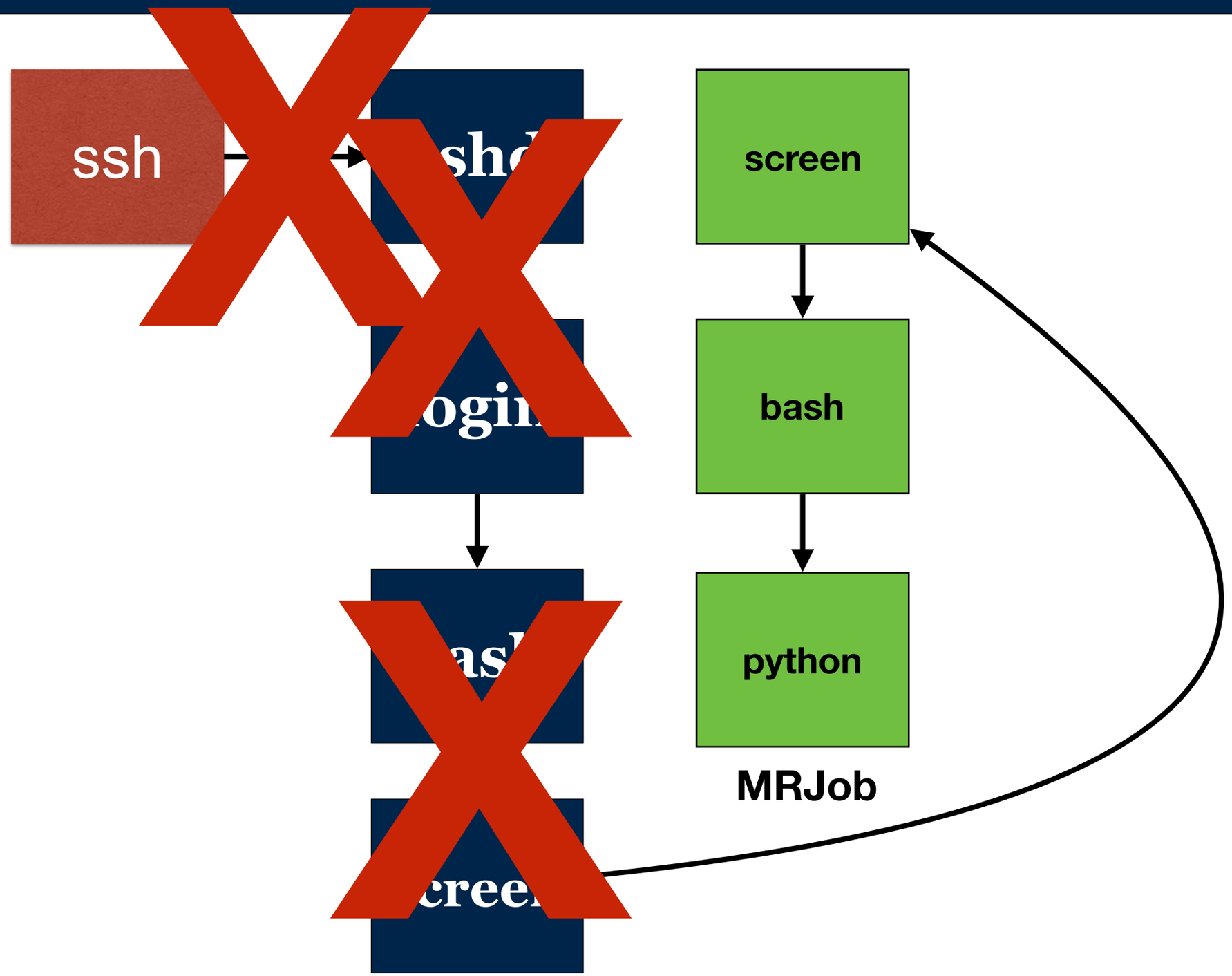
bash ran python (MRjob)



Kill signals propagate when sshd is terminated

screen starts a subshell on its own virtual terminal

```
$ echo $$
1692
$ screen
$ echo $$
45213
$
```



Kill signals do not cross the process group

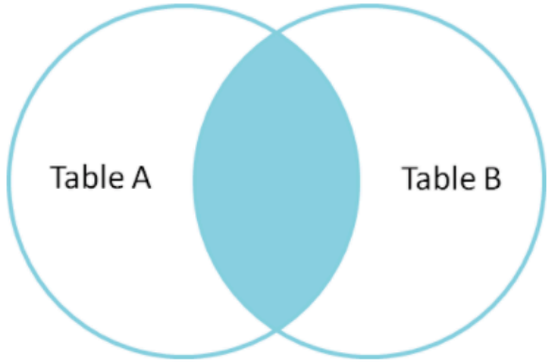
Fortunately, hadoop was still running..

```
[hadoop@ip-172-31-39-51 L04]$ aws s3 ls s3://anly502-slg/L04-1/
2017-02-06 13:57:22          0 _SUCCESS
2017-02-06 13:57:08   45785080 part-00000
2017-02-06 13:57:06   45788680 part-00001
2017-02-06 13:57:20   45784681 part-00002
[hadoop@ip-172-31-39-51 L04]$
```

Relational joins

Inner join

```
SELECT * FROM TableA
INNER JOIN TableB
ON TableA.name = TableB.name

id  name       id   name
--  ----       --   ----
1   Pirate     2    Pirate
3   Ninja      4    Ninja
```



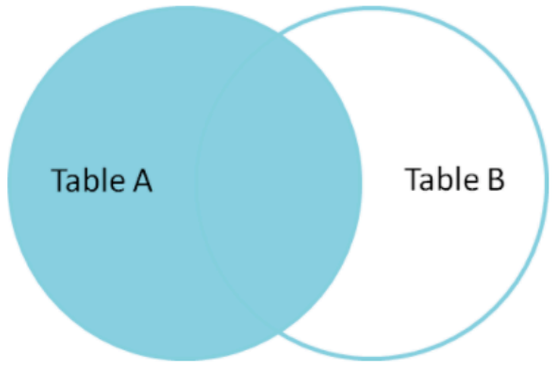
Other joins without own syntax in SQL

- Semi-join
- Anti-join
- Division

Cartesian join

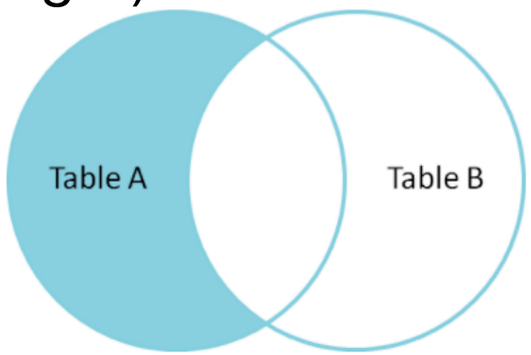
Left join (include all A)

```
SELECT * FROM TableA
LEFT OUTER JOIN TableB
ON TableA.name = TableB.name

id  name       id    name
--  ----       --    ----
1   Pirate     2     Pirate
2   Monkey     null  null
3   Ninja      4     Ninja
4   Spaghetti  null  null
```



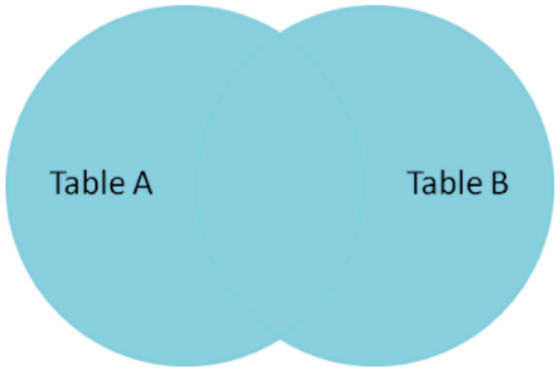
Left join (exclude missing B)

```
SELECT * FROM TableA
LEFT OUTER JOIN TableB
ON TableA.name = TableB.name
WHERE TableB.id IS null

id  name       id    name
--  ----       --    ----
2   Monkey     null  null
4   Spaghetti  null  null
```



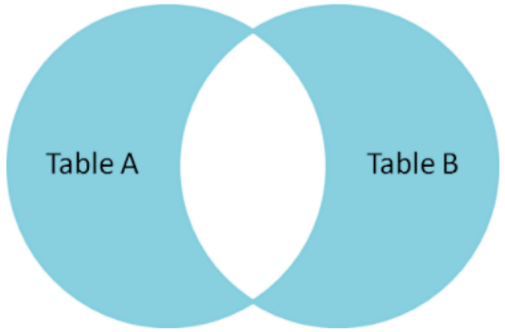
Full Outer join (everything)

```
SELECT * FROM TableA
FULL OUTER JOIN TableB
ON TableA.name = TableB.name

id    name       id    name
--    ----       --    ----
1     Pirate     2     Pirate
2     Monkey     null  null
3     Ninja      4     Ninja
4     Spaghetti  null  null
null  null       1     Rutabaga
null  null       3     Darth Vader
```



Full Outer join (exclude common)

```
SELECT * FROM TableA
FULL OUTER JOIN TableB
ON TableA.name = TableB.name
WHERE TableA.id IS null
OR TableB.id IS null

id    name       id    name
--    ----       --    ----
2     Monkey     null  null
4     Spaghetti  null  null
null  null       1     Rutabaga
null  null       3     Darth Vader
```



https://blog.codinghorror.com/a-visual-explanation-of-sql-joins/

Traditional database joins (with SQL)

A database "join" combines data from two tables:

Names		
NameID	Name	SchoolID
1	Alice	1
2	Bob	2
3	Claire	3
4	Donald	2

Schools		
SchoolID	Name	State
1	Georgeto	DC
2	Harvard	MA
3	Yale	CT
4	UC	CA

Example: print each person's name and where they go to school:

```
mysql> select Names.Name,Schools.Name,Schools.State from Names left join Schools
on Names.SchoolID=Schools.SchoolID;
+--------+------------+-------+
| Name   | Name       | State |
+--------+------------+-------+
| Alice  | Georgetown | DC    |
| Bob    | Harvard    | MA    |
| Donald | Harvard    | MA    |
| Claire | Yale       | CT    |
+--------+------------+-------+
4 rows in set (0.00 sec)
```

Why joins in Hadoop, MapReduce?

Why joins in Hadoop?

- Historically, most enterprise information systems were built on relational database models and optimized for operations in a normalized fashion.
  - *Parallel databases are quite expensive*
- In Hadoop, many kinds of data can be stored (structured, unstructured) from many sources and usually need to be combined for analytical purposes.

Why joins in MapReduce?

- OK, probably not the best tool to use since there are other higher level Hadoop based tools where joins are easier to implement like Hive and PIG.
- Because we can
- Useful to understand the algorithms that underlie basic relational operations

Three approaches for performing joins in MapRecduce

Reduce-side join:

- Mapper emits <*key-to-join*, *other-values*>
- Reducer combines all elements with same key, preserving the values.
  - —*requires reducer to keep state.*

Map-side join:

- Input data must be presorted by join key and both datasets must be partitioned equally
- Usually an element-by-element join
  - —*Remember — each core has its own mapper!*
- No reducer needed, save on shuffle/sort

Memory-Backed join (replicated join):

- One of the tables must fit in memory.
- Reducer may not be needed

```
names:
+--------+--------+----------+
| NameID | Name   | SchoolID |
+--------+--------+----------+
|      1 | Alice  |        1 |
|      2 | Bob    |        2 |
|      3 | Claire |        3 |
|      4 | Donald |        2 |
+--------+--------+----------+

schools:
+----------+-------------+-------+
| SchoolID | Name        | State |
+----------+-------------+-------+
|        1 | Georgetown  | DC    |
|        2 | Harvard     | MA    |
|        3 | Yale        | CT    |
|        4 | UC Berkeley | CA    |
+----------+-------------+-------+

mysql> select
Names.Name,Schools.Name,Schools.Sta
te from Names left join Schools on
Names.SchoolID=Schools.SchoolID;
+--------+------------+-------+
| Name   | Name       | State |
+--------+------------+-------+
| Alice  | Georgetown | DC    |
| Bob    | Harvard    | MA    |
| Donald | Harvard    | MA    |
| Claire | Yale       | CT    |
+--------+------------+-------+
4 rows in set (0.00 sec)
```

Joins in MapReduce are implemented by collecting values for keys.

This output:

```
mysql> select Names.Name,Schools.Name,Schools.State from Names left join Schools
on Names.SchoolID=Schools.SchoolID;
+--------+------------+-------+
| Name   | Name       | State |
+--------+------------+-------+
| Alice  | Georgetown | DC    |
| Bob    | Harvard    | MA    |
| Donald | Harvard    | MA    |
| Claire | Yale       | CT    |
+--------+------------+-------+
4 rows in set (0.00 sec)
```

Can be represented as:

```
Georgetown, DC • Alice
Harvard, MA    • Bob, Donald
Yale, CT       • Claire
```

This is implemented with a *reduce-side join.*

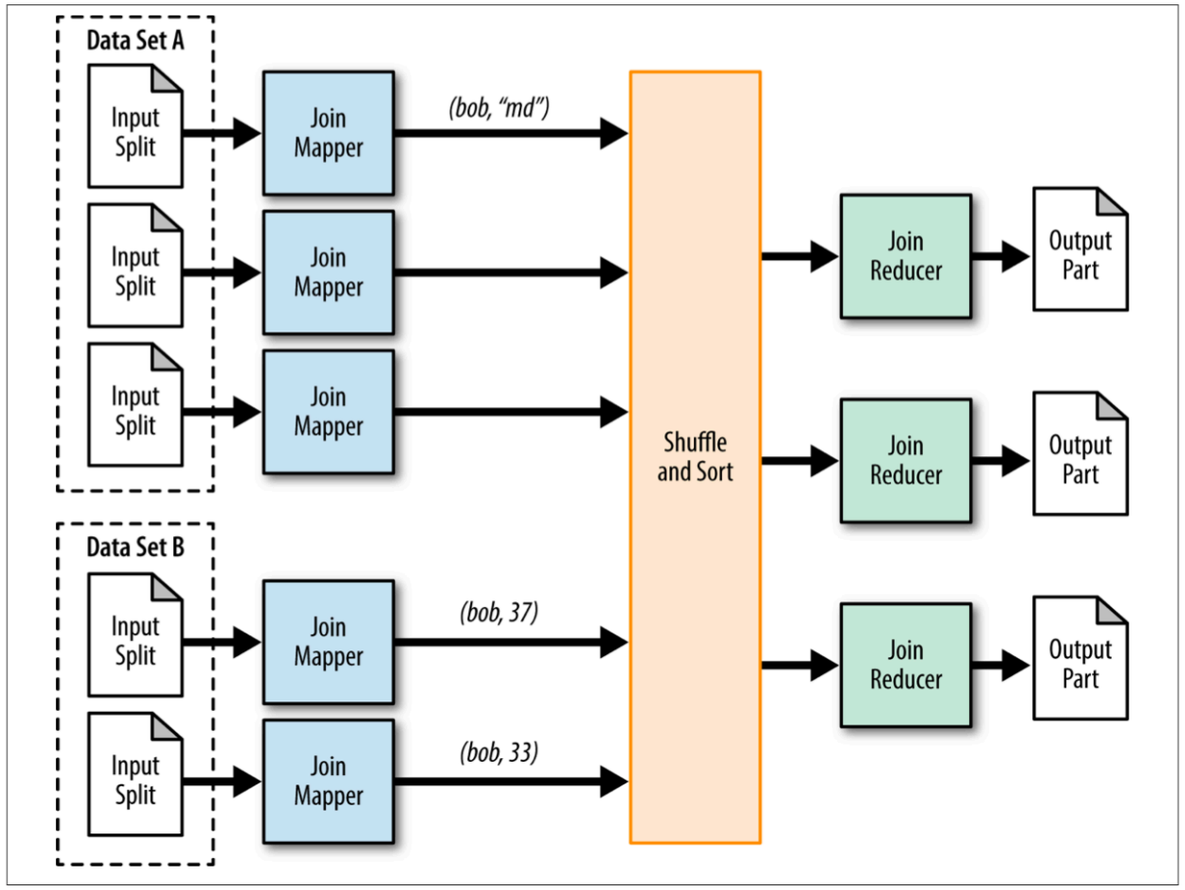
Reduce-Side Joins

- Easiest to implement
- Can be used for any of the join types
- Join as many datasets together as you need
- Hadoop sends identical keys to the same reducer, so the data is organized by key
- Trade-off in performance, because of shuffle/sort

One-to-One

- A value from dataset 'X' shares a common key with dataset 'Y'
- Key order is guaranteed on reducer, but value order is not
- Tag values

One-to-Many



Miner, Donald, and Adam Shook. MapReduce Design Patterns. Sebastopol, CA: Oreilly, 2013.

So how do we implement this in MapReduce?

Map reduce wants a *single input*, but we have two tables:

Names		
NameID	Name	SchoolID
1	Alice	1
2	Bob	2
3	Claire	3
4	Donald	2

Schools		
SchoolID	Name	State
1	Georgeto	DC
2	Harvard	MA
3	Yale	CT
4	UC	CA

Answer: prefix each row with the name of the table:

```
Names, 1, Alice, 1
Names, 2, Bob, 2
Names, 3, Claire, 3
Names, 4, Donald, 2
Schools, 1, Georgetown, DC
Schools, 2, Harvard, MA
Schools, 3, Yale, CT
Schools, 4, UC Berkeley, CA
```

```
1: Names, 1, Alice, 1
1: Schools, 1, Georgetown, DC
2: Names, 2, Bob, 2
2: Names, 4, Donald, 2
2: Schools, 2, Harvard, MA
3: Names, 3, Claire, 3
3: Schools, 3, Yale, CT
4: Schools, 4, UC Berkeley, CA
```

We might have data from two tables (data frames) to combine

ID, E#, Observation

```
100, 17, Yellow
101, 35, Red
102, 53, Purple
103, 29, Green
...
993243, 549003, Clear
```

E#, Name

```
17, Chlorine
29, Copper
35, Bromine
53, Iodine
...
549003, Adamantine
```

MAP

```
17, (Obs, (100, 17, Yellow))
35, (Obs, (101, 35, Red))
53, (Obs, (102, 53, Purple))
29, (Obs, (103, 29, Green))
...
549003, (Obs, (993243, 549003, Clear))
```

```
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
35, (Name, (35, Bromine))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
```

Mapped Data

ID, E#, Name, Observation

```
100, (17, Chlorine, Yellow)
101, (35, Bromine, Red)
102, (53, Iodine, Purple)
103, (29, Copper, Green)
...
993243, (549003, Adamantine, Clear)
```

Final Data

```
17, (Obs, (100, 17, Yellow))
35, (Obs, (101, 35, Red))
53, (Obs, (102, 53, Purple))
29, (Obs, (103, 29, Green))
...
549003, (Obs, (993243, 549003, Clear))
```

```
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
35, (Name, (35, Bromine))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
```

Mapped Data

Shuffle

```
17, (Obs, (100, 17, Yellow))
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
29, (Obs, (103, 29, Green))
35, (Obs, (101, 35, Red))
35, (Name, (35, Bromine))
53, (Obs, (102, 53, Purple))
53, (Name, (53, Iodine))
...
549003, (Name, (549003,
Adamantine))
549003, (Obs, (993243, 549003,
Clear))
```

Shuffled

```
17, (Obs, (100, 17, Yellow))
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
29, (Obs, (103, 29, Green))
35, (Obs, (101, 35, Red))
35, (Name, (35, Bromine))
53, (Obs, (102, 53, Purple))
53, (Name, (53, Iodine))
...
549003, (Name, (549003,
Adamantine))
549003, (Obs, (993243, 549003,
Clear))
```

Shuffled

Reduce

```
100, (17, Chlorine, Yellow)
101, (35, Bromine, Red)
102, (53, Iodine, Purple)
103, (29, Copper, Green)
...
993243, (549003, Adamantine,
```

Reduced

Join map code in mrjob

```
100, 17, Yellow
101, 35, Red
102, 53, Purple
103, 29, Green
...
993243, 549003, Clear
```

```
17, Chlorine
29, Copper
35, Bromine
53, Iodine
...
549003,
```

MAP

```
17, (Obs, (100, 17, Yellow))
35, (Obs, (101, 35, Red))
53, (Obs, (102, 53, Purple))
29, (Obs, (103, 29, Green))
...
549003, (Obs, (993243, 549003, Clear))
```

```
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
35, (Name, (35, Bromine))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
```

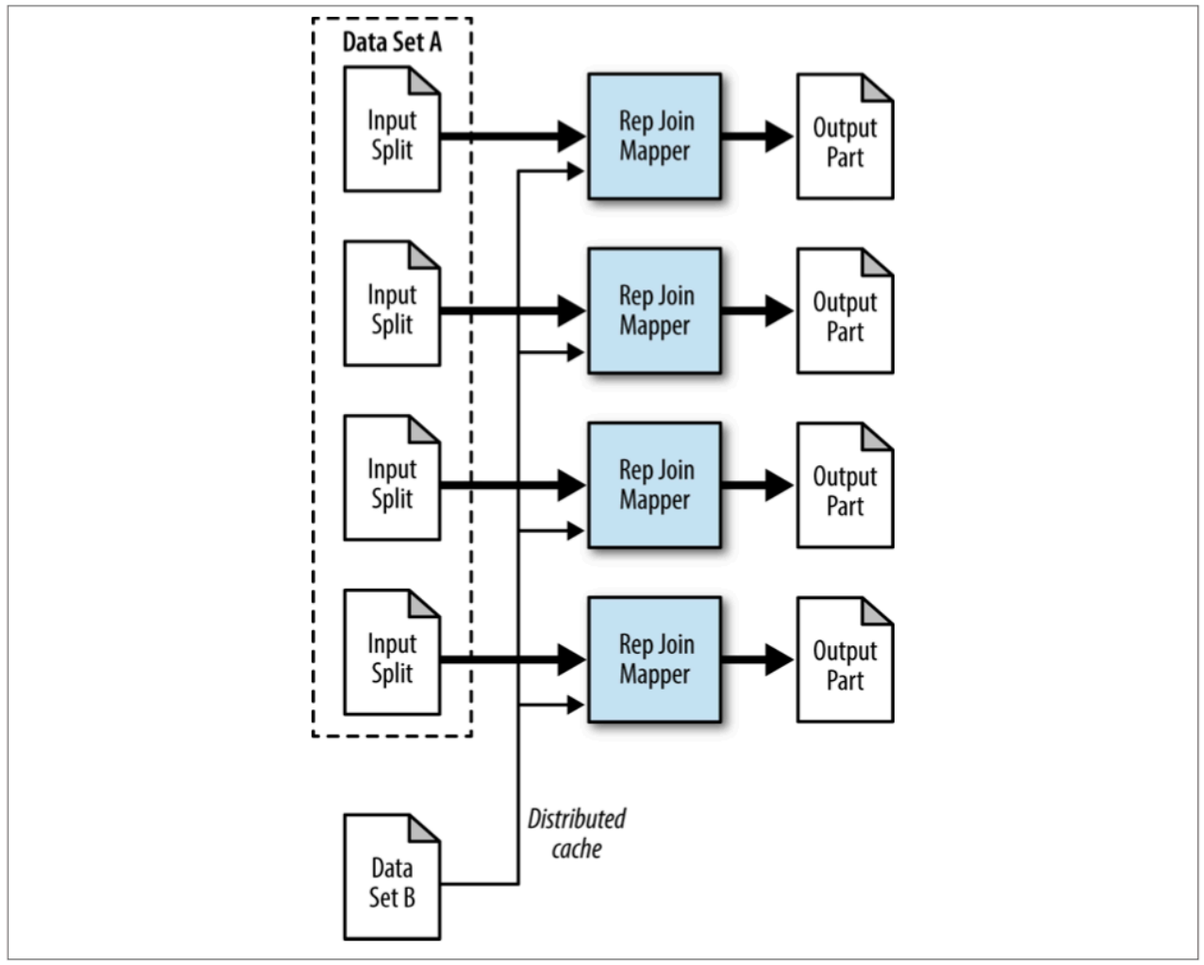
```
def mapper(self, _, line):
    fields = line.split(", ")
    if len(fields)==3:
        self.increment_counter("Info","Obs Count",1)
        yield fields[1], ("Obs", fields)
    elif len(fields)==2:
        self.increment_counter("Info","Name Count",1)
        yield fields[0], ("Name",fields)
    else:
        self.increment_counter("Warn","Invalid Data",1)
```

Counters for
“situational awareness”

Map-Side Joins

Replicated Join

- Join between one large and many small data sets that can be performed on the map side
- Eliminates the need for shuffle/sort
- Smaller datasets are read into memory during map task
- Large dataset(s) are the input(s) to the map task
- The dataset to be read in memory needs to be "packaged" with the job so it is available in every node and accessible to every map task. This small file is not read from HDFS (remember, program to the data)
- Applicability:
  - *Left outer join - where the larger of the datasets is the "left"*

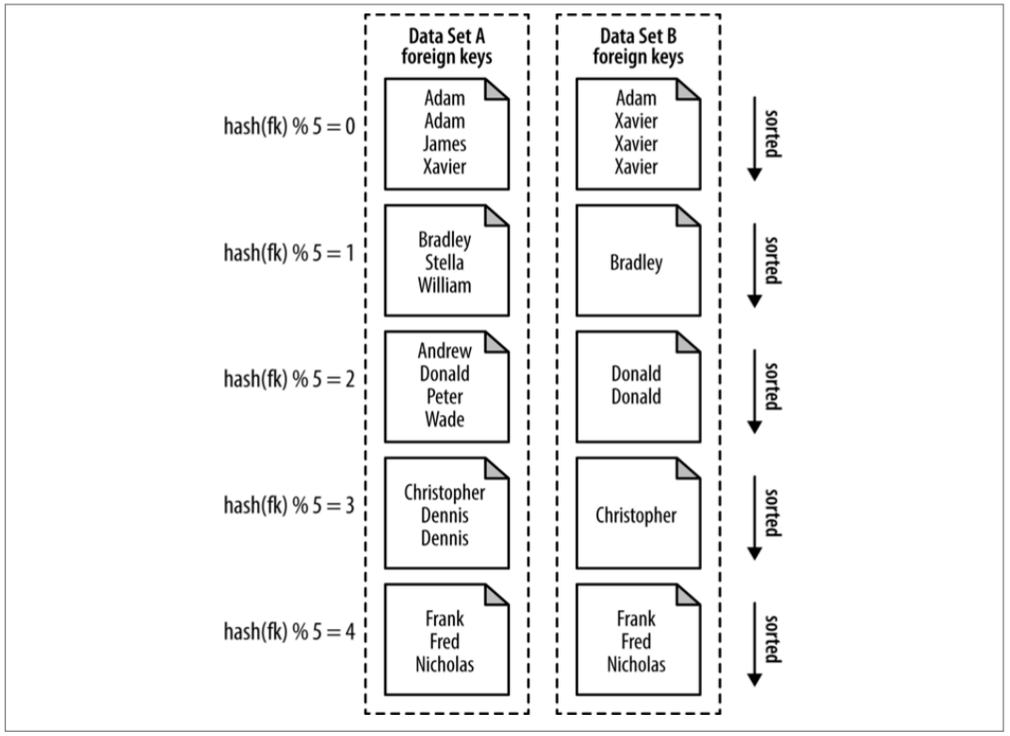


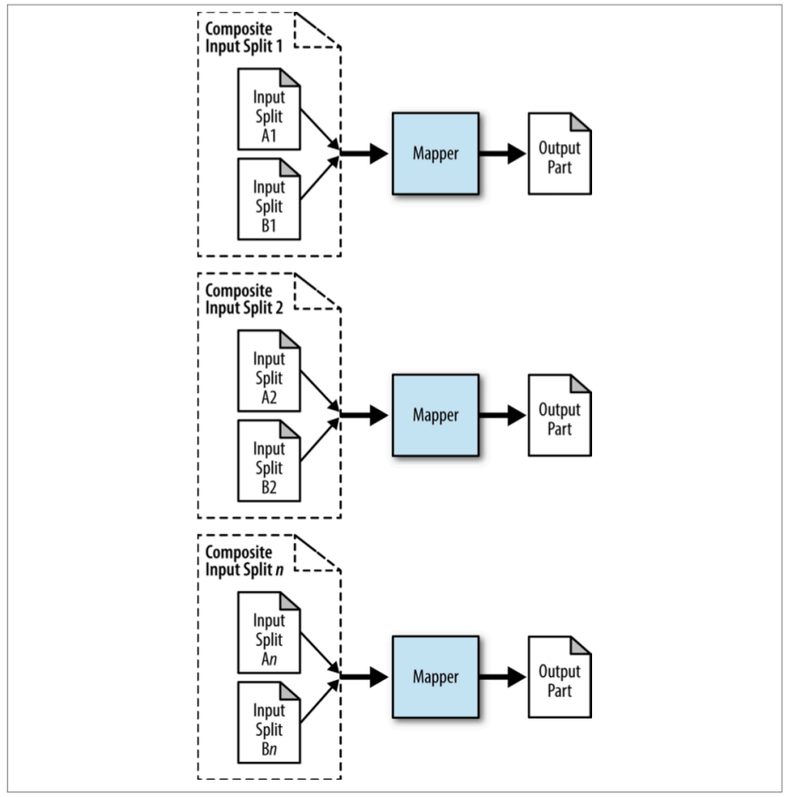
Miner, Donald, and Adam Shook. MapReduce Design Patterns. Sebastopol, CA: Oreilly, 2013.

Map-Side Joins

Composite Join

- Join between multiple large datasets only on the map side
- No reducer
- Data needs to be organized in specific way
- Applicability
  - —*Inner or full outer join*
  - —*All datasets are large*
  - —*Datasets have the same number of partitions*
  - —*Partitions are sorted equally by key*



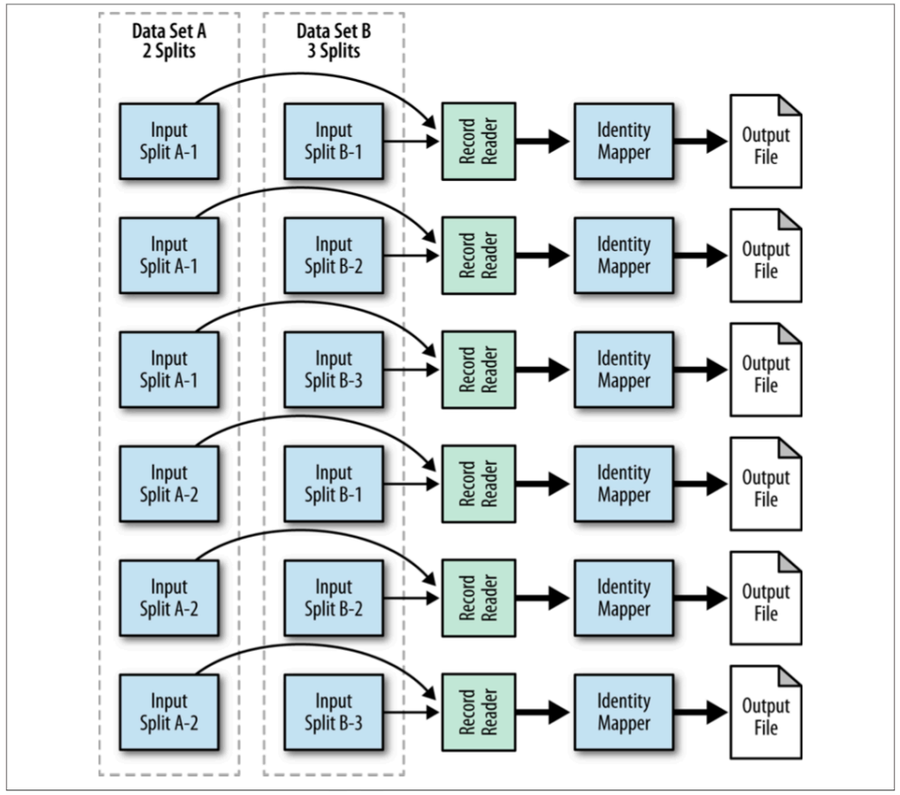


Miner, Donald, and Adam Shook. MapReduce Design Patterns. Sebastopol, CA: Oreilly, 2013.

Map-Side Joins

Cartesian Product

- Pairs up every single record with every other record in the dataset
- Very expensive operation
- Does not fit into the MapReduce paradigm very well because it is not intuitively splittable
- Perhaps there is no foreign key
- Use a cartesian product when:
  - *want/need to analyze relationships between all pairs of individual records*
  - *no constraints on execution time*
  - *all other methods have been exhausted*
- Massive explosion in data size



Miner, Donald, and Adam Shook. MapReduce Design Patterns. Sebastopol, CA: Oreilly, 2013.

Implementing joins in mrjob

```
$ more mrjob_school_join.py
from mrjob.job import MRJob
import heapq, csv

names_cols = 'Table,NameID,Name,SchoolID'.split(",")
schools_cols = 'Table,SchoolID,Name,State'.split(",")

class SchoolJoin(MRJob):
    def mapper(self, _, line):
        row = [a.strip() for a in csv.reader([li
        if row[0]=='Names':
            rowdict = dict(zip(names_cols,row))
            yield rowdict["SchoolID"], rowdict
        elif row[0]=='Schools':
            rowdict = dict(zip(schools_cols,row))
            yield rowdict["SchoolID"], rowdict
        else:
            self.increment_counter("warn","invalid input line",1)

    def reducer(self, key, values):
        # All of the values for the SchoolID will be grouped together
        # Find the school name, the school state, and all the students

        students = []
        schoolName  = None
        schoolState = None
        for v in values:
            if v["Table"]=="Schools":
                schoolName = v["Name"]
                schoolState = v["State"]
            elif v["Table"]=="Names":
                students.append(v["Name"])

        yield key, (schoolName,schoolState,students)

if __name__=="__main__":
    SchoolJoin.run()
```

```
Names, 1, Alice, 1
Names, 2, Bob, 2
Names, 3, Claire, 3
Names, 4, Donald, 2
Schools, 1, Georgetown, DC
Schools, 2, Harvard, MA
Schools, 3, Yale, CT
Schools, 4, UC Berkeley, CA
```

```
1: Names, 1, Alice, 1
1: Schools, 1, Georgetown, DC
2: Names, 2, Bob, 2
2: Names, 4, Donald, 2
2: Schools, 2, Harvard, MA
3: Names, 3, Claire, 3
3: Schools, 3, Yale, CT
4: Schools, 4, UC Berkeley, CA
```

Join output:

```
$ python2.7 mrjob_school_join.py  --strict-protocol  SchoolNames.txt
no configs found; falling back on auto-configuration
no configs found; falling back on auto-configuration
creating tmp directory /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/
mrjob_school_join.simsong.20151207.015133.385300
writing to /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/
mrjob_school_join.simsong.20151207.015133.385300/step-0-mapper_part-00000
Counters from step 1:
  (no counters found)
writing to /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/
mrjob_school_join.simsong.20151207.015133.385300/step-0-mapper-sorted
> sort /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/mrjob_school_join.simsong.
20151207.015133.385300/step-0-mapper_part-00000
writing to /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/
mrjob_school_join.simsong.20151207.015133.385300/step-0-reducer_part-00000
Counters from step 1:
  (no counters found)
Moving /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/mrjob_school_join.simsong.
20151207.015133.385300/step-0-reducer_part-00000 -> /var/folders/y0/
dbn9vvhd7dsg5_8f71c264vm0000gn/T/mrjob_school_join.simsong.20151207.015133.385300/
output/part-00000
Streaming final output from /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/
mrjob_school_join.simsong.20151207.015133.385300/output
"1"         ["Georgetown", "DC", ["Alice"]]
"2"         ["Harvard", "MA", ["Bob", "Donald"]]
"3"         ["Yale", "CT", ["Claire"]]
"4"         ["UC Berkeley", "CA", []]
removing tmp directory /var/folders/y0/dbn9vvhd7dsg5_8f71c264vm0000gn/T/
mrjob_school_join.simsong.20151207.015133.385300
$
```

Join reduce code in mrjob

```
17, (Obs, (100, 17, Yellow))
35, (Obs, (101, 35, Red))
53, (Obs, (102, 53, Purple))
29, (Obs, (103, 29, Green))
...
549003, (Obs, (993243, 549003, Clear))
```

```
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
35, (Name, (35, Bromine))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
```

```
100, (17, Chlorine, Yellow)
101, (35, Bromine, Red)
102, (53, Iodine, Purple)
103, (29, Copper, Green)
...
993243, (549003, Adamantine,
```

```
def reducer(self, key, values):
    name = None
    for v in values:
        if len(v)!=2:
            self.increment_counter("Warn","Invalid Join",1)
            continue
        if v[0]=='Name':
            name = v[1]
            continue
        if v[0]=='Obs':
            obs = v[1]
            if name:
                assert key==name[0]
                assert key==obs[1]
                yield obs[0],(obs[1],name[1],obs[2])
            else:
                self.increment_counter("Warn","Obs without Name")
                yield obs[0],(obs[1],"n/a",obs[2])
```

Runtime Error Checking

Lab

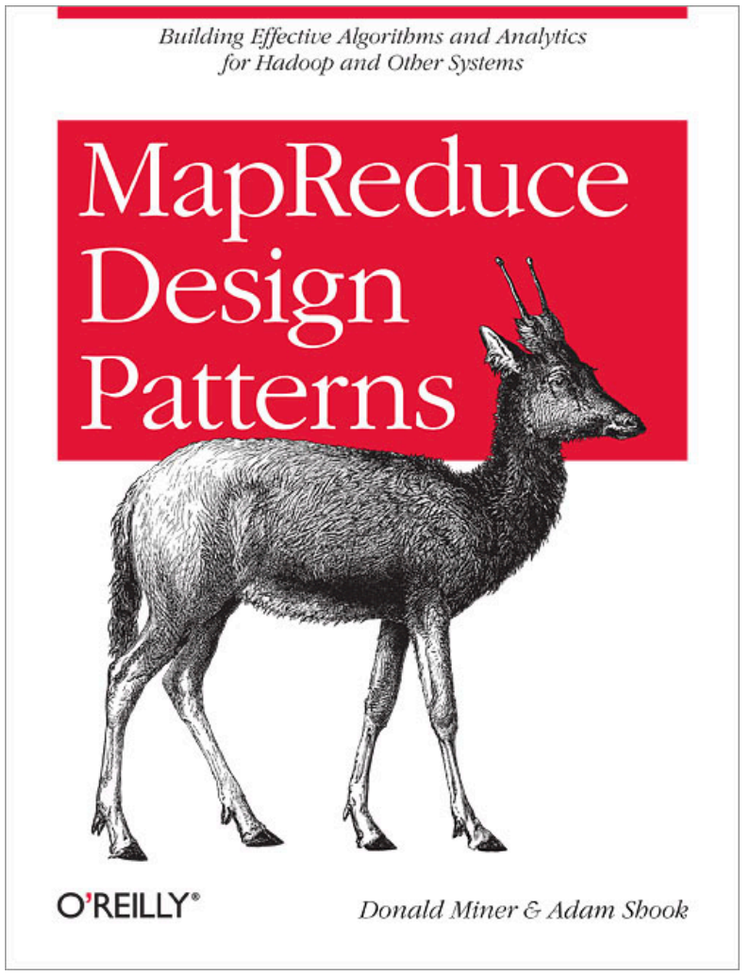
- Start a 1 node cluster (only master), don't forget to use the bootstrap script to get mrjob installed
- Clone the class repository
- Change to L04 directory
- There are two Python mrjob that do simple reduce-side joins scripts. Try them out. Look at the code
  - *Try mrjob_school_join.py (use this with names.csv and schools.csv as inputs)*
  - *Try mrjob_school_join.py (use this with tagged-names.csv and tagged-schools.csv as inputs)*
  - *Try myjob_join.py (use this with countries.dat and customers.dat)*
- Try out other mrjob functionality and see the output
  - *python34 myjob_join.py --mapper countries.dat customers.dat*
  - *python34 myjob_join.py --mapper countries.dat customers.dat | sort | python34 myjob_join.py --reducer countries.dat customers.dat*
  - *python34 myjob_join.py countries.dat customers.dat*
- Copy s3://gu-anly502/L04/L04files.zip to your node, and unzip (this will create a subdirectory called dell-dvd-dat
  - *Look at the README.txt file to get a sense of the files*
  - *Use this dataset to practice joins*

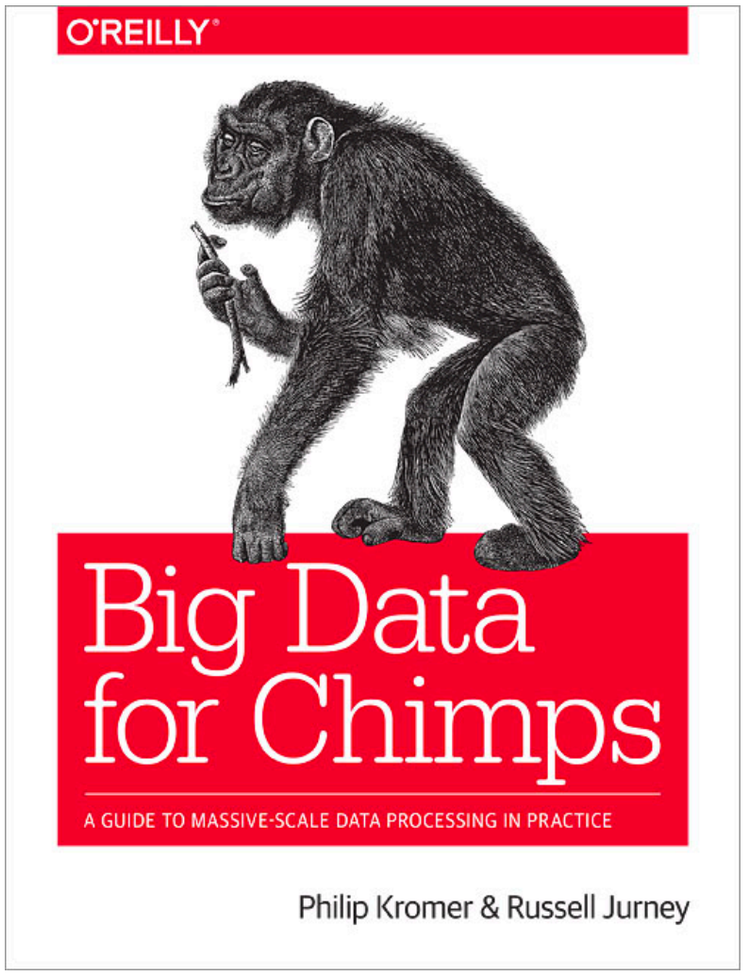
Further readings on design patterns

https://lintool.github.io/MapReduceAlgorithms/MapReduce-book-final.pdf

https://gist.github.com/rjurney/2f350b2cbed9862b692b

https://blog.matthewrathbone.com/2016/02/09/python-tutorial.html





Coming Up

A03 - Due Friday 2/17

Q4 - Due Friday 2/10

L05 - Data Wrangling