Project 2

I have used Hadoop Map-Reduce method to get the output in this project.

The advantage of Hadoop Map-Reduce is very fast and also, we have done homework 4 on Hadoop map-reduce, so it would much easier to use this method instead of learning a new one.

The following is the output for the project.

Average TMAX, Average TMIN, Minimum Temperature & Maximum Temperature according to year:

Year	Average Tmax	Average Tmin	Minimum Temp	Maximum Temp
2000	17.55	4.43	-57.8	52.2
2001	17.8	4.79	-52.8	52.8
2002	17.7	4.6	-47.2	53.3
2003	17.70	4.86	-50.0	53.3
2004	17.45	4.9	-53.3	51.7
2005	17.76	4.980	-55.0	53.9
2006	18.07	5.0	-52.8	52.8
2007	17.82	4.97	-53.9	53.9
2008	17.00	4.07	-57.8	52.8
2009	16.86	4.33	-55.6	53.3
2010	17.15	4.72	-53.3	51.7
2011	17.24	4.60	-51.7	51.1
2012	18.39	5.34	-54.4	53.7
2013	16.73	4.30	-52.8	53.9
2014	16.84	4.38	-50.0	52.2
2015	17.72	5.35	-52.8	55.6
2016	17.90	5.51	-46.9	53.9
2017	17.66	5.28	-52.0	52.8
2018	17.13	4.94	-47.8	52.8
2019	7.303	-3.65	-49.4	41.7

5 hottest and 5 coldest stations according to the year with their codes:

Year	Tag	Station1 code	Station2 code	Station3 code	Station4 code	Station5 code
2000	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2000	Cold	'USC00501684'	'USC00505644'	'USC00505644'	'USC00501684'	'USC00508140'
2001	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2001	Cold	'USW00026508'	'USR0000ABCA'	'USW00026508'	'USS0051R01S'	'USW00026508'
2002	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2002	Cold	'USR0000AKAI'	'USS0051R01S'	'USS0050S01S'	'USR0000ABEV'	'USC00503212'
2003	Hot	'USC00042319'	'USR0000CMEA'	'USC00042319'	'USC00042319'	'USR0000AHAV'
2003	Cold	'USC00501492'	'USC00501492'	'USS0051R01S'	'USW00026533'	'USS0050S01S'
2004	Hot	'USC00042319'	'USC00042319'	'USC00024761'	'USC00024761'	'USC00042319'
2004	Cold	'USC00501684'	'USC00501684'	'USC00502568'	'USS0045O10S'	'USS0045O10S'
2005	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2005	Cold	'USC00501684'	'USC00509313'	'USC00501684'	'USC00501684'	'USC00501684'
2006	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2006	Cold	'USR0000ASEL'	'USC00501492'	'USC00501492'	'USC00501492'	'USC00501492'
2007	Hot	'USC00042319'	'USC00042319'	'USW00053139'	'USC00042319'	'USC00042319'
2007	Cold	'USC00501684'	'USC00501684'	'USC00501684'	'USS0045R01S'	'USC00501684'
2008	Hot	'USC00044297'	'USC00042319'	'USC00042319'	'USC00024761'	'USC00044297'
2008	Cold	'USC00501684'	'USC00501684'	'USC00501684'	'USC00501684'	'USC00501684'
2009	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2009	Cold	'USC00501684'	'USC00502101'	'USC00502101'	'USC00501684'	'USC00502101'
2010	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USR0000AHAV'
2010	Cold	'USC00501684'	'USC00502101'	'USC00501684'	'USC00502101'	'USS0051R01S'
2011	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2011	Cold	'USC00509869'	'USS0045R01S'	'USS0045R01S'	'USS0051R01S'	'USS0051R01S'
2012	Hot	'USS0005N23S'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2012	Cold	'USC00503165'	'USC00503165'	'USC00503165'	'USC00503212'	'USS0051R01S'
2013	Hot	'USC00042319'	'USW00004134'	'USC00042319'	'USC00042319'	'USC00044297'
2013	Cold	'USC00502339'	'USC00501684'	'USC00501684'	'USC00501684'	'USC00502339'
2014	Hot	'USC00042319'	'USC00042319'	'USW00053139'	'USC00042319'	'USC00042319'
2014	Cold	'USC00501684'	'USC00501684'	'USC00501684'	'USC00501684'	'USC00501684'
2015	Hot	'USR0000HKAU'	'USR0000HKAU'	'USR0000HKAU'	'USC00042319'	'USC00042319'
2015	Cold	'USC00502339'	'USC00502339'	'USC00501684'	'USC00501684'	'USC00502339'
2016	Hot	'USR0000CBEV'	'USC00042319'	'USC00042319'	'USC00040924'	'USC00042319'
2016	Cold	'USS0051R01S'	'USR0000ACHL'	'USC00501684'	'USR0000ACHL'	'USR0000ACHL'
2017	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00021050'
2017	Cold	'USS0051R01S'	'USR0000ASLC'	'USW00026529'	'USW00026529'	'USS0051R01S'
2018	Hot	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'	'USC00042319'
2018	Cold	'USC00501684'	'USC00501684'	'USR0000ANOR'	'USR0000AKAV'	'USW00096406'
2019	Hot	'USW00022010'	'USC00415048'	'USW00012907'	'USC00417624'	'USR0000TFAL'
2019	Cold	'USC00509891'	'USC00501684'	'USC00211840'	'USC00218618'	'USC00211840'

Hottest and Coldest Temperature with the station code:

Temperature	Station Code
55.6	USR0000HKAU
-57.8	USC00501684

Map1.py

```
#!/usr/bin/env python
import sys
def function(a):
    return {'ID':a[0],
            'DATE':a[1],
            'TYPE':a[2],
            'VALUE':a[3],
            'MFLAG':a[4],
            'QFLAG':a[5],
            'SFLAG':a[6],
            'OBSTIME':a[7]}
for l in sys.stdin:
    data = l.strip().upper().split(',')
    c = function(data)
    if 'TMAX' != c['TYPE'] and 'TMIN' != c['TYPE']:
        continue
    if c['VALUE'] == -9999:
        continue
    if c['SFLAG'] == '':
        continue
    if c['QFLAG'] != '':
        continue
```

```
    if c['MFLAG'] == 'P':
        contiune
    print '%s,%s,%s,%s,' % (c['DATE'],c['ID'],c['TYPE'],c['VALUE'])
```

Reduce1.py

```
#!/usr/bin/env python
import sys
import operator

countmax = 0
countmin = 0
avgmax = 0
avgmin = 0
hot = []
cold = []
max = (-9999,'','')
min = (9999,'','')
currentyear = None
for l in sys.stdin:
    l = l.strip().split(',')

    date = l[0]
    year = date[:4]
    id = l[1]
    met = l[2]
    val = l[3]

    try:
        val = int(val)
    except ValueError:
        continue
```

```
if currentyear is None:
    currentyear = year

if currentyear != year:
    print 'Year: %s' % currentyear
    print 'Average TMAX: %s' % (avgmax * 0.1 / countmax)
    print 'Average TMIN: %s' % (avgmin * 0.1 / countmin)

    print 'Hottest Day: day: %s  val: %s  loc: %s' %(hot[0][2], hot[0][0], hot[0][1])
    print 'Coldest Day: day: %s  val: %s  loc: %s' %(cold[0][2], cold[0][0], cold[0][1])

    print 'Hottest Stations %s' % ([y[1] for y in hot])
    print 'Hottest Station Temperatures %s' % ([y[0] for y in hot])
    print 'Coldest Stations %s' % ([y[1] for y in cold])
    print 'Coldest Station Temperatures %s' % ([y[0] for y in cold])
    currentyear = year
    countmax = 0
    countmin = 0
    avgmax = 0
    avgmin = 0
    hot = []
    cold = []
if met == 'TMAX':
    avgmax += val
    countmax += 1
    if max[0] < val:
        max = (val,id,date)

    hot.append((val,id,date))
```

```
        if len(hot) > 5:
            hot = sorted(hot, key=operator.itemgetter(0), reverse=True)
            hot.pop(len(hot) - 1)

    elif met == 'TMIN':
        avgmin += val
        countmin += 1
        if min[0] > val:
            min = (val,id,date)

        cold.append((val,id,date))

        if len(cold) > 5:
            cold = sorted(cold, key=operator.itemgetter(0))
            cold.pop(len(cold) - 1)



print 'Year: %s' % currentyear
print 'Average TMAX: %s' % (avgmax * 0.1 / countmax)
print 'Average TMIN: %s' % (avgmin * 0.1 / countmin)

print 'Hottest Day: day: %s  val: %s  loc: %s' %(hot[0][2], hot[0][0], hot[0][1])
print 'Coldest Day: day: %s  val: %s  loc: %s' %(cold[0][2], cold[0][0], cold[0][1])

print 'Hottest Stations %s' % ([y[1] for y in hot])
print 'Hottest Station Temperatures %s' % ([y[0] for y in hot])
print 'Coldest Stations %s' % ([y[1] for y in cold])
print 'Coldest Station Temperatures %s' % ([y[0] for y in cold])



print 'Max TMAX: %s | Station: %s' % (max[0],max[1])
```

```
print 'Min TMIN: %s | Station: %s' % (min[0],min[1])
```