Step 1. Download and upzip the HDB file. Load the file by using `\l`.

In [1]:
```
\l C:\q\columbiaHdb\
```

Step 2. Define a variable `dt` to be the date `2019.01.03`.

In [2]:
```
dt: 2019.01.03
```

Step 3. Print the HDB table for the date `2019.01.03`.

In [3]:
```
tbl: select from bin10 where date = dt
```

In [4]:
```
tbl
```

Out[4]:
```
date       time     id trade     mid      spread        vol         adv      ..
-------------------------------------------------------------------------------..
2019.01.03 09:30:00 0  454071.8  91.715   0.001148717   0.000446367 1.122415e..
2019.01.03 09:30:10 0  -171833.3 91.72671 0.001072136   0.000446367 1.122415e..
2019.01.03 09:30:20 0  -301.2193 91.58853 7.658112e-005 0.000446367 1.122415e..
2019.01.03 09:30:30 0  -78884.65 91.65646 0.000612649   0.000446367 1.122415e..
2019.01.03 09:30:40 0  -24705.54 91.60259 0.0008423923  0.000446367 1.122415e..
2019.01.03 09:30:50 0  -92166.19 91.58385 0.0005871219  0.000446367 1.122415e..
2019.01.03 09:31:00 0  -22823.01 91.56276 0.000663703   0.000446367 1.122415e..
2019.01.03 09:31:10 0  -4940.182 91.59088 0.0005615949  0.000446367 1.122415e..
2019.01.03 09:31:20 0  -41691.88 91.59322 0.0005360678  0.000446367 1.122415e..
2019.01.03 09:31:30 0  -539826.4 91.60727 0.0007402842  0.000446367 1.122415e..
2019.01.03 09:31:40 0  -53883.89 91.5487  0.0004594867  0.000446367 1.122415e..
2019.01.03 09:31:50 0  316896.2  91.56745 0.0003063245  0.000446367 1.122415e..
2019.01.03 09:32:00 0  8767.784  91.82033 0.0004594867  0.000446367 1.122415e..
2019.01.03 09:32:10 0  132407.8  91.84138 0.000638176   0.000446367 1.122415e..
2019.01.03 09:32:20 0  -52051.64 92.02614 0.0005615949  0.000446367 1.122415e..
2019.01.03 09:32:30 0  441082.8  92.01445 0.0004339597  0.000446367 1.122415e..
2019.01.03 09:32:40 0  -182.6154 92.23874 0.000638176   0.000446367 1.122415e..
2019.01.03 09:32:50 0  4196.414  92.18269 0.0002807974  0.000446367 1.122415e..
2019.01.03 09:33:00 0  112899.5  92.36479 0.0007402842  0.000446367 1.122415e..
2019.01.03 09:33:10 0  -51038.96 92.27376 0.000612649   0.000446367 1.122415e..
..
```

Step 4. Print the column names of `tbl`. Bonus: read the documentations of `xcol` and `xcols` at https://code.kx.com/q/ref/cols/#cols.

In [5]:
```
cols tbl
```

Out[5]:
```
`date`time`id`trade`mid`spread`vol`adv
```

Step 5. For each stock `id`, find the closing price (`mid` at `16:00:00`) on `2019.01.03`.

In [6]:
```
select closing: last mid by id from tbl
```

Out[6]:
```
id| closing
--| --------
0 | 89.00147
1 | 68.89127
2 | 43.22651
3 | 61.05507
4 | 51.32667
5 | 89.21663
6 | 73.56235
7 | 14.53625
8 | 825.9875
9 | 50.55819
10| 297.0247
11| 106.3285
12| 66.26467
13| 63.56045
14| 143.5618
15| 29.45335
16| 17.54937
17| 16.60063
18| 32.555
19| 41.46458
..
```

Step 6. Append the closing price to `tbl`.

In [7]:
```
tbl: update closing: last mid by id from tbl

/ reorder the columns
tbl: xcols[`date`time`id`trade`mid`closing; tbl]
tbl
```

Out[7]:
```
date       time     id trade     mid      closing  spread        vol         ..
-------------------------------------------------------------------------------..
2019.01.03 09:30:00 0  454071.8  91.715   89.00147 0.001148717   0.000446367 ..
2019.01.03 09:30:10 0  -171833.3 91.72671 89.00147 0.001072136   0.000446367 ..
2019.01.03 09:30:20 0  -301.2193 91.58853 89.00147 7.658112e-005 0.000446367 ..
2019.01.03 09:30:30 0  -78884.65 91.65646 89.00147 0.000612649   0.000446367 ..
2019.01.03 09:30:40 0  -24705.54 91.60259 89.00147 0.0008423923  0.000446367 ..
2019.01.03 09:30:50 0  -92166.19 91.58385 89.00147 0.0005871219  0.000446367 ..
2019.01.03 09:31:00 0  -22823.01 91.56276 89.00147 0.000663703   0.000446367 ..
2019.01.03 09:31:10 0  -4940.182 91.59088 89.00147 0.0005615949  0.000446367 ..
2019.01.03 09:31:20 0  -41691.88 91.59322 89.00147 0.0005360678  0.000446367 ..
2019.01.03 09:31:30 0  -539826.4 91.60727 89.00147 0.0007402842  0.000446367 ..
2019.01.03 09:31:40 0  -53883.89 91.5487  89.00147 0.0004594867  0.000446367 ..
2019.01.03 09:31:50 0  316896.2  91.56745 89.00147 0.0003063245  0.000446367 ..
2019.01.03 09:32:00 0  8767.784  91.82033 89.00147 0.0004594867  0.000446367 ..
2019.01.03 09:32:10 0  132407.8  91.84138 89.00147 0.000638176   0.000446367 ..
2019.01.03 09:32:20 0  -52051.64 92.02614 89.00147 0.0005615949  0.000446367 ..
2019.01.03 09:32:30 0  441082.8  92.01445 89.00147 0.0004339597  0.000446367 ..
2019.01.03 09:32:40 0  -182.6154 92.23874 89.00147 0.000638176   0.000446367 ..
2019.01.03 09:32:50 0  4196.414  92.18269 89.00147 0.0002807974  0.000446367 ..
2019.01.03 09:33:00 0  112899.5  92.36479 89.00147 0.0007402842  0.000446367 ..
2019.01.03 09:33:10 0  -51038.96 92.27376 89.00147 0.000612649   0.000446367 ..
..
```

Step 7. For each stock and each time, calculate the return from the current time to market close.

In [8]:
```
tbl: update rtn: (neg 1) + closing%mid from tbl
```

```
/ reorder the columns
tbl: xcols[`date`time`id`trade`mid`closing`rtn; tbl]
tbl
```

Out[8]:

```
date       time     id trade     mid      closing  rtn         spread        ..
-------------------------------------------------------------------------------..
2019.01.03 09:30:00 0  454071.8  91.715   89.00147 -0.02958659 0.001148717   ..
2019.01.03 09:30:10 0  -171833.3 91.72671 89.00147 -0.02971043 0.001072136   ..
2019.01.03 09:30:20 0  -301.2193 91.58853 89.00147 -0.02824664 7.658112e-005 ..
2019.01.03 09:30:30 0  -78884.65 91.65646 89.00147 -0.02896681 0.000612649   ..
2019.01.03 09:30:40 0  -24705.54 91.60259 89.00147 -0.02839575 0.0008423923  ..
2019.01.03 09:30:50 0  -92166.19 91.58385 89.00147 -0.02819692 0.0005871219  ..
2019.01.03 09:31:00 0  -22823.01 91.56276 89.00147 -0.02797313 0.000663703   ..
2019.01.03 09:31:10 0  -4940.182 91.59088 89.00147 -0.02827149 0.0005615949  ..
2019.01.03 09:31:20 0  -41691.88 91.59322 89.00147 -0.02829635 0.0005360678  ..
2019.01.03 09:31:30 0  -539826.4 91.60727 89.00147 -0.02844544 0.0007402842  ..
2019.01.03 09:31:40 0  -53883.89 91.5487  89.00147 -0.02782386 0.0004594867  ..
2019.01.03 09:31:50 0  316896.2  91.56745 89.00147 -0.02802287 0.0003063245  ..
2019.01.03 09:32:00 0  8767.784  91.82033 89.00147 -0.03069975 0.0004594867  ..
2019.01.03 09:32:10 0  132407.8  91.84138 89.00147 -0.030922   0.000638176   ..
2019.01.03 09:32:20 0  -52051.64 92.02614 89.00147 -0.03286751 0.0005615949  ..
2019.01.03 09:32:30 0  441082.8  92.01445 89.00147 -0.03274466 0.0004339597  ..
2019.01.03 09:32:40 0  -182.6154 92.23874 89.00147 -0.03509664 0.000638176   ..
2019.01.03 09:32:50 0  4196.414  92.18269 89.00147 -0.03450997 0.0002807974  ..
2019.01.03 09:33:00 0  112899.5  92.36479 89.00147 -0.03641345 0.0007402842  ..
2019.01.03 09:33:10 0  -51038.96 92.27376 89.00147 -0.03546287 0.000612649   ..
..
```

Step 8. Reset `tbl` to be `bin10` on `dt`. For each stock, aggregate the absolute volume.

In [9]:

```
tbl: select from bin10 where date = dt

select sum abs trade by id from tbl
```

Out[9]:

```
id| trade
--| -------------
0 | 1.122415e+008
1 | 3.607207e+007
2 | 1.38682e+007
3 | 1.192783e+007
4 | 2.019054e+007
5 | 7931115
6 | 1.790894e+007
7 | 9922401
8 | 3.452761e+007
9 | 1.862319e+007
10| 8.626539e+007
11| 5.807214e+007
12| 2.996236e+007
13| 1.538175e+007
14| 9.883369e+008
15| 7.305665e+007
16| 1.933401e+008
17| 8571269
18| 8691514
19| 1.143187e+008
..
```

Step 9. Extract the last entry of `adv` from `tbl` by `id`. Compare with Step 8.

In [10]:
```
select last adv by id from tbl
```

Out[10]:
```
id| adv
--| -------------
0 | 1.122415e+008
1 | 3.607207e+007
2 | 1.38682e+007
3 | 1.192783e+007
4 | 2.019054e+007
5 | 7931115
6 | 1.790894e+007
7 | 9922401
8 | 3.452761e+007
9 | 1.862319e+007
10| 8.626539e+007
11| 5.807214e+007
12| 2.996236e+007
13| 1.538175e+007
14| 9.883369e+008
15| 7.305665e+007
16| 1.933401e+008
17| 8571269
18| 8691514
19| 1.143187e+008
..
```

Step 10. For each `id`, calculate the standard deviation of 10-second returns and compare with `vol`.

In [11]:
```
select std: sdev (neg 1) + mid % prev mid, last vol by id from tbl
```

Out[11]:
```
id| std          vol
--| -------------------------
0 | 0.000446376  0.000446367
1 | 0.0003582123 0.000358252
2 | 0.0004633929 0.0004827488
3 | 0.00044749   0.0004667023
4 | 0.000346852  0.0003663134
5 | 0.0005029934 0.000516633
6 | 0.0004704837 0.0004753984
7 | 0.0004011879 0.0004223277
8 | 0.0007291334 0.0007492939
9 | 0.0006666224 0.0006668529
10| 0.0007129224 0.0007148106
11| 0.0003856844 0.0003864019
12| 0.0003662671 0.0003828439
13| 0.0004642847 0.0004774389
14| 0.0005108918 0.0005112646
15| 0.0003700215 0.000372391
16| 0.0009079489 0.0009286111
17| 0.000644153  0.0006442204
18| 0.0006410766 0.0006423178
19| 0.0004128916 0.0004340049
..
```