Q LANGUAGE - QUERIES

Queries in **q** are shorter and simpler and extend the capabilities of sql. The main query expression is the 'select expression', which in its simplest form extracts sub-tables but it can also create new columns.

The general form of a **Select expression** is as follows −

```
Select columns by columns from table where conditions
```

****Note − by & where** phrases are optional, only the 'from expression' is mandatory.

In general, the syntax will be −

```
select [a] [by b] from t [where c]
update [a] [by b] from t [where c]
```

The syntax of **q** expressions look quite similar to SQL, but **q** expressions are simple and powerful. An equivalent sql expression for the above **q** expression would be as follows −

```
select [b] [a] from t [where c] [group by b order by b]
update t set [a] [where c]
```

All the clauses execute on the columns and therefore **q** can take advantage of order. As Sql queries are not based on order, they cannot take that advantage.

q relational queries are generally much smaller in size as compared to their corresponding sql. Ordered and functional queries do things that are difficult in sql.

In a historical database, the ordering of the **where** clause is very important because it affects the performance of the query. The **partition** variable *date/month/day* always comes first followed by the sorted and indexed column *generallythesymcolumn*.

For example,

```
select from table where date in d, sym in s
```

is much faster than,

```
select from table where sym in s, date in d
```

Basics Queries

Let's write a query script in notepad *asbelow*, save *as ∗ . q*, and then load it.

```
sym:asc`AIG`CITI`CSCO`IBM`MSFT;
ex:"NASDAQ"
dst:`$":c:/q/test/data/";            /database destination

@[dst;`sym;:;sym];
n:1000000;

trade:([]sym:n?`sym;time:10:30:00.0+til
n;price:n?3.3e;size:n?9;ex:n?ex);

quote:([]sym:n?`sym;time:10:30:00.0+til
n;bid:n?3.3e;ask:n?3.3e;bsize:n?9;asize:n?9;ex:n?ex);

{@[;`sym;`p#]`sym xasc x}each`trade`quote;
d:2014.08.07 2014.08.08 2014.08.09 2014.08.10 2014.08.11; /Date vector can also be
changed by the user
```

```
dt:{[d;t].[dst;(`$string d;t;`);:;value t]};
d dt/:\:`trade`quote;

Note: Once you run this query, two folders .i.e. "test" and "data" will be created under
"c:/q/", and date partition data can be seen inside data folder.
```

Queries with Constraints

* Denotes HDB query

Select all IBM trades

```
select from trade where sym in `IBM
```

*Select all IBM trades on a certain day

```
thisday: 2014.08.11
select from trade where date=thisday,sym=`IBM
```

Select all IBM trades with a price > 100

```
select from trade where sym=`IBM, price > 100.0
```

Select all IBM trades with a price less than or equal to 100

```
select from trade where sym=`IBM,not price > 100.0
```

*Select all IBM trades between 10.30 and 10.40, in the morning, on a certain date

```
thisday: 2014.08.11
select from trade where
date = thisday, sym = `IBM, time > 10:30:00.000,time < 10:40:00.000
```

Select all IBM trades in ascending order of price

```
`price xasc select from trade where sym =`IBM
```

*Select all IBM trades in descending order of price in a certain time frame

```
`price xdesc select from trade where date within 2014.08.07 2014.08.11, sym =`IBM
```

Composite sort − sort ascending order by sym and then sort the result in descending order of price

```
`sym xasc `price xdesc select from trade where date = 2014.08.07,size = 5
```

Select all IBM or MSFT trades

```
select from trade where sym in `IBM`MSFT
```

*Calculate count of all symbols in ascending order within a certain time frame

```
`numsym xasc select numsym: count i by sym from trade where date within 2014.08.07
2014.08.11
```

*Calculate count of all symbols in descending order within a certain time frame

```
`numsym xdesc select numsym: count i by sym from trade where date within 2014.08.07
2014.08.11
```

* What is the maximum price of IBM stock within a certain time frame, and when does this first happen?

```
select time,ask from quote where date within 2014.08.07 2014.08.11,
sym =`IBM, ask = exec first ask from select max ask from quote where
sym =`IBM
```

Select the last price for each sym in hourly buckets

```
select last price by hour:time.hh, sym from trade
```

Queries with Aggregations

* Calculate vwap *VolumeWeightedAveragePrice* of all symbols

```
select vwap:size wavg price by sym from trade
```

* Count the number of records *inmillions* for a certain month

```
(select trade:1e-6*count i by date.dd from trade where date.month=2014.08m) + select
quote:1e-6*count i by date.dd from quote where date.month=2014.08m
```

* HLOC – Daily High, Low, Open and Close for CSCO in a certain month

```
select high:max price,low:min price,open:first price,close:last price by date.dd from
trade where date.month=2014.08m,sym =`CSCO
```

* Daily Vwap for CSCO in a certain month

```
select vwap:size wavg price by date.dd from trade where date.month = 2014.08m ,sym =
`CSCO
```

* Calculate the hourly mean, variance and standard deviation of the price for AIG

```
select mean:avg price, variance:var price, stdDev:dev price by date, hour:time.hh from
trade where sym = `AIG
```

Select the price range in hourly buckets

```
select range:max[price] – min price by date,sym,hour:time.hh from trade
```

* Daily Spread $average bid - ask$ for CSCO in a certain month

```
select spread:avg bid-ask by date.dd from quote where date.month = 2014.08m, sym = `CSCO
```

* Daily Traded Values for all syms in a certain month

```
select dtv:sum size by date,sym from trade where date.month = 2014.08m
```

Extract a 5 minute vwap for CSCO

```
select size wavg price by 5 xbar time.minute from trade where sym = `CSCO
```

* Extract 10 minute bars for CSCO

```
select high:max price,low:min price,close:last price by date, 10 xbar time.minute from
trade where sym = `CSCO
```

* Find the times when the price exceeds 100 basis points $100e-4$ over the last price for CSCO for a certain day

```
select time from trade where date = 2014.08.11,sym = `CSCO,price > 1.01*last price
```

*** Full Day Price and Volume for MSFT in 1 Minute Intervals for the last date in the database**

```
select last price,last size by time.minute from trade where date = last date, sym =
`MSFT
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js