Q LANGUAGE - ATTRIBUTES

Lists, dictionaries, or columns of a table can have attributes applied to them. Attributes impose certain properties on the list. Some attributes might disappear on modification.

Types of Attributes

Sorted `s#

`s# means the list is sorted in an ascending order. If a list is explicitly sorted by asc *orxasc*, the list will automatically have the sorted attribute set.

```
q)L1: asc 40 30 20 50 9 4

q)L1
`s#4 9 20 30 40 50
```

A list which is known to be sorted can also have the attribute explicitly set. **Q** will check if the list is sorted, and if is not, an **s-fail** error will be thrown.

```
q)L2:30 40 24 30 2

q)`s#L2
's-fail
```

The sorted attribute will be lost upon an unsorted append.

Parted `p#

`p# means the list is parted and identical items are stored contiguously.

The range is an **int** or **temporal type** having an underlying int value, such as years, months, days, etc. You can also partition over a symbol provided it is enumerated.

Applying the parted attribute creates an index dictionary that maps each unique output value to the position of its first occurrence. When a list is parted, lookup is much faster, since linear search is replaced by hashtable lookup.

```
q)L:`p# 99 88 77 1 2 3

q)L
`p#99 88 77 1 2 3

q)L,:3

q)L
99 88 77 1 2 3 3
```

Note −

- The parted attribute is not preserved under an operation on the list, even if the operation preserves the partitioning.
- The parted attribute should be considered when the number of entities reaches a billion and most of the partitions are of substantial size, i.e., there is significant repetition.

Grouped `g#

`g# means the list is grouped. An internal dictionary is built and maintained which maps each unique item to each of its indices, requiring considerable storage space. For a list of length **L** containing **u** unique items of size **s**, this will be $L \times 4 + u \times s$ bytes.

Grouping can be applied to a list when no other assumptions about its structure can be made.

The attribute can be applied to any typed lists. It is maintained on appends, but lost on deletes.

```
q)L: `g# 1 2 3 4 5 4 2 3 1 4 5 6

q)L
`g#1 2 3 4 5 4 2 3 1 4 5 6

q)L,:9

q)L
`g#1 2 3 4 5 4 2 3 1 4 5 6 9

q)L _:2

q)L
1 2 4 5 4 2 3 1 4 5 6 9
```

Unique `#u

Applying the unique attribute `u# to a list indicates that the items of the list are distinct. Knowing that the elements of a list are unique dramatically speeds up **distinct** and allows **q** to execute some comparisons early.

When a list is flagged as unique, an internal hash map is created to each item in the list. Operations on the list must preserve uniqueness or the attribute is lost.

```
q)LU:`u#`MSFT`SAMSUNG`APPLE

q)LU
`u#`MSFT`SAMSUNG`APPLE

q)LU,:`IBM                          /Uniqueness preserved

q)LU
`u#`MSFT`SAMSUNG`APPLE`IBM

q)LU,:`SAMSUNG                      / Attribute lost

q)LU
`MSFT`SAMSUNG`APPLE`IBM`SAMSUNG
```

Note −

- `u# is preserved on concatenations which preserve the uniqueness. It is lost on deletions and non-unique concatenations.
- Searches on `u# lists are done via a hash function.

Removing Attributes

Attributes can be removed by applying `#.

Applying Attributes

Three formats for applying attributes are −

- **L: `s# 14 2 3 3 9**   / Specify during list creation
- **@[`.; `L ; `s#]**   / Functional apply, i.e. to the variable list L

  / in the default namespace $i.e.$ `. apply

/ the sorted `s# attribute

- **Update `s#time from `tab**

/ Update the table *tab* to apply the

/ attribute.

Let’s apply the above three different formats with examples.

```
q)/ set the attribute during creation

q)L:`s# 3 4 9 10 23 84 90

q)/apply the attribute to existing list data

q)L1: 9 18 27 36 42 54

q)@[`.;`L1;`s#]
`.

q)L1                  / check
`s#9 18 27 36 42 54

q)@[`.;`L1;`#]       / clear attribute
`.

q)L1
9 18 27 36 42 54

q)/update a table to apply the attribute

q)t: ([] sym:`ibm`msft`samsung; mcap:9000 18000 27000)

q)t:([]time:09:00 09:30 10:00t;sym:`ibm`msft`samsung; mcap:9000 18000 27000)

q)t

    time          sym    mcap
---------------------------------
  09:00:00.000   ibm    9000
  09:30:00.000   msft   18000
  10:00:00.000  samsung 27000

q)update `s#time from `t
`t

q)meta t                / check it was applied

    c  | t f a
------ | -----
  time | t s
  sym  | s
  mcap | j

Above we can see that the attribute column in meta table results shows the time column is
sorted (`s#)
```

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