Q LANGUAGE - TABLES ON DISK

Data on your hard disk *alsocalledhistoricaldatabase* can be saved in three different formats − Flat Files, Splayed Tables, and Partitioned Tables. Here we will learn how to use these three formats to save data.

Flat file

Flat files are fully loaded into memory which is why their size *memoryfootprint* should be small. Tables are saved on disk entirely in one file *sosizematters*.

The functions used to manipulate these tables are **set/get** −

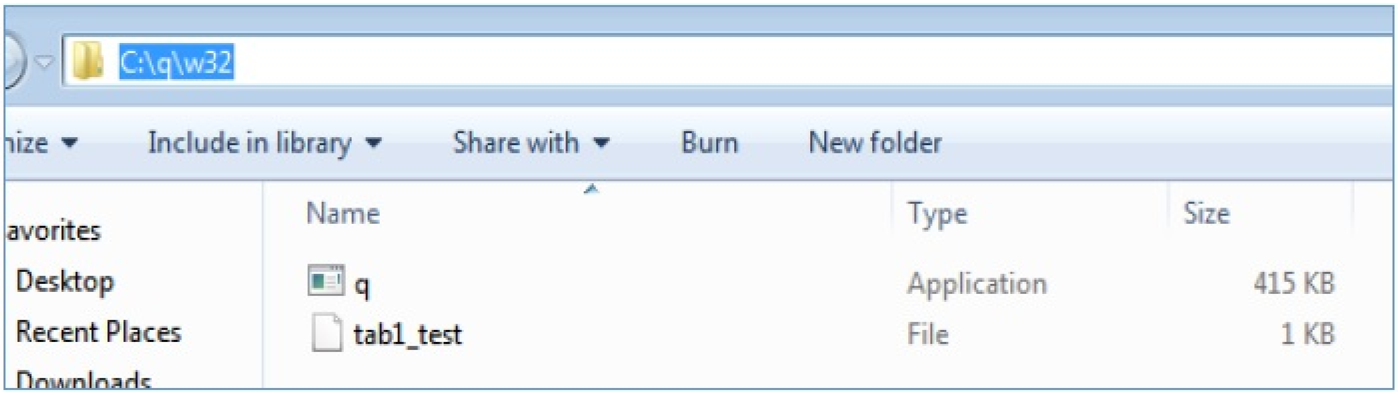
```
`:path_to_file/filename set tablename
```

Let’s take an example to demonstrate how it works −

```
q)tables `.
`s#`t`tab`tab1

q)`:c:/q/w32/tab1_test set tab1
`:c:/q/w32/tab1_test
```

In Windows environment, flat files are saved at the location − **C:\q\w32**



Name	Type	Size
q	Application	415 KB
tab1_test	File	1 KB

Get the flat file from your disk *historicaldb* and use the **get** command as follows −

```
q)tab2: get `:c:/q/w32/tab1_test

q)tab2

   sym     |   time           price   size
--------- | -------------------------------
  APPLE    | 11:16:39.779   8.388858   12
  MSFT     | 11:16:39.779   19.59907   10
  IBM      | 11:16:39.779   37.5638    1
 SAMSUNG   | 11:16:39.779   61.37452   90
  APPLE    | 11:16:39.779   52.94808   73
```

A new table is created **tab2** with its contents stored in **tab1_test** file.

Splayed Tables

If there are too many columns in a table, then we store such tables in splayed format, i.e., we save them on disk in a directory. Inside the directory, each column is saved in a separate file under the same name as the column name. Each column is saved as a list of corresponding type in a kdb+ binary file.

Saving a table in splayed format is very useful when we have to access only a few columns frequently out of its many columns. A splayed table directory contains **.d** binary file which contains the order of the columns.

Much like a flat file, a table can be saved as splayed by using the **set** command. To save a table as splayed, the file path should end with a backlash −

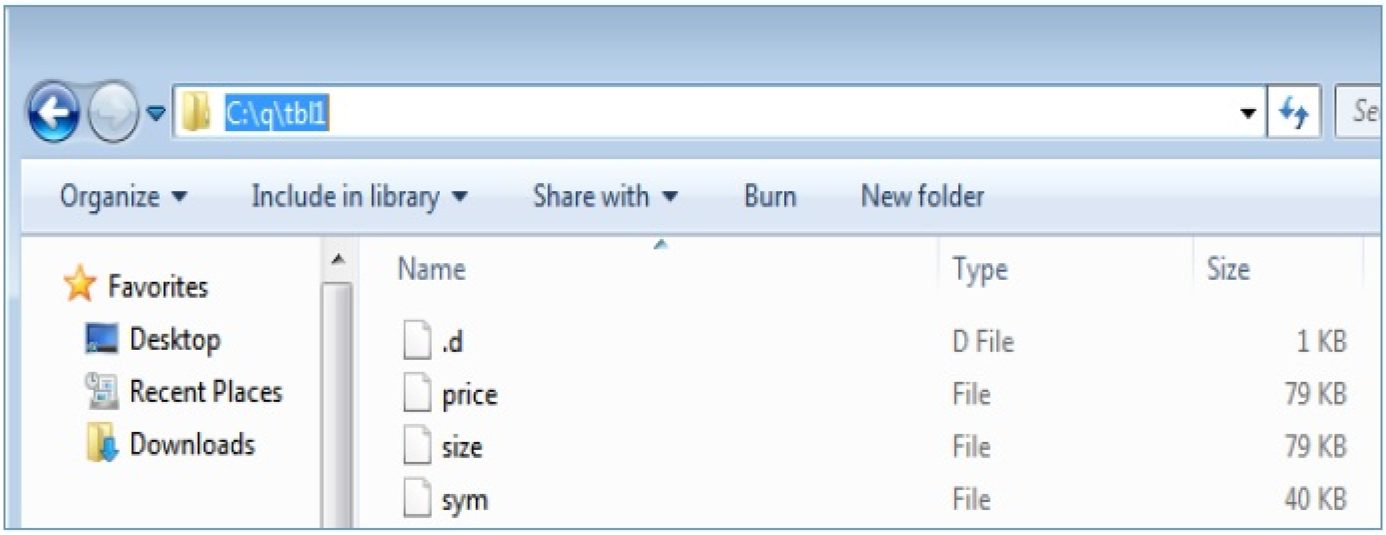
```
`:path_to_filename/filename/ set tablename
```

For reading a splayed table, we can use the **get** function −

```
tablename: get `:path_to_file/filename
```

Note − For a table to be saved as splayed, it should be un-keyed and enumerated.

In Windows environment, your file structure will appear as follows −



Name	Type	Size
.d	D File	1 KB
price	File	79 KB
size	File	79 KB
sym	File	40 KB

Partitioned Tables

Partitioned tables provide an efficient means to manage huge tables containing significant volumes of data. Partitioned tables are splayed tables spread across more partitions *directories*.

Inside each partition, a table will have its own directory, with the structure of a splayed table. The tables could be split on a day/month/year basis in order to provide optimized access to its content.

To get the content of a partitioned table, use the following code block −

```
q)get `:c:/q/data/2000.01.13              // “get” command used, sample folder

quote| +`sym`time`bid`ask`bsize`asize`ex!(`p#`sym!0 0 0 0 0 0 0 0 0 0 0
0 0 0….

trade| +`sym`time`price`size`ex!(`p#`sym!0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 ….
```

Let’s try to get the contents of a trade table −

```
q)get `:c:/q/data/2000.01.13/trade

   sym    time            price      size    ex
--------------------------------------------------
    0   09:30:00.496    0.4092016    7       T
    0   09:30:00.501    1.428629     4       N
    0   09:30:00.707    0.5647834    6       T
    0   09:30:00.781    1.590509     5       T
    0   09:30:00.848    2.242627     3       A
    0   09:30:00.860    2.277041     8       T
    0   09:30:00.931    0.8044885    8       A
    0   09:30:01.197    1.344031     2      A
    0   09:30:01.337    1.875        3      A
    0   09:30:01.399    2.187723     7      A
```

Note − The partitioned mode is suitable for tables with millions of records per day *i. e. timeseriesdata*

Sym file

The sym file is a kdb+ binary file containing the list of symbols from all splayed and partitioned tables. It can be read with,

```
get `:sym
```

par.txt file *optional*

This is a configuration file, used when partitions are spread on several directories/disk drives, and contain the paths to the disk partitions.

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