First derivatives plc

CMTP

Advanced Kdb+ Exam

**Exercise 1 - Tick**

Create a project with the following components and requirements. The basis of the exercise will be tick scripts available here:

<https://github.com/kxsystems/kdb-tick>

| Number | Component | Requirements |
| --- | --- | --- |
|  | Ticker Plant | Create a tickerplant which contains the schema for the below tables:   1. Trade 2. Quote 3. Aggregation table which will contain following metrics by symbol    1. Max/min trade prices    2. Traded volume    3. Top of book details   The tickerplant should also log every minute:   1. number of messages that have been processed by table up until that time 2. subscription details of subscribers |
| 2. | RDB | Create 2 RDB which subscribe as follows:   1. Subscribes only to Trade and Quote tables 2. Subscribes to aggregation table |
| 3. | Feed Handler | Create a mock feedhandler that will supply trades and quotes to the above system. |
| 4. | CEP | Create a Complex Event Processer/Calculation Engine which will subscribe to trades and quote from tickerplant and then calculate metrics for the aggregation table and publish the data back to the Ticker Plant. |
| 5. | Logging | Write a logging script that can be loaded into the above components and will log:   1. details of connections opened 2. details of connections closed 3. all logging statements should include username of calling process where applicable and memory usage details from .Q.w[] 4. functions should be available so that can write internal logging statements to write to standard out and error |
| 6. | Startup/shutdown scripts | Create a startup script in whichever language you prefer e.g. ksh, bash, perl etc that will work in 3 main modes:   1. START - give the ability to start up the components in 1-4. You should be able to start all or any specific one 2. STOP - give the ability to stop the components in 1-4. You should be able to start all or any specific one 3. TEST – test which components are currently running and supply details 4. Make the script as configurable as you see fit e.g. how port numbers will be assigned, log locations, db directories |
| 7. | Ticker Plant log replay | Write a script that reads in a tickerplant log file which contains trade and quote updates and creates a new tickerplant log file which only contains the trade updates for ibm.n. |
| 8. | CSV File load | Create a script that will load a csv file and publish the contents to the Ticker plant. |
| 9. | EOD Process | Write a script which will take the Ticker Plant log and create a daily partitioned HDB in which all columns are compressed with the exception of sym and time. |
| 10. | Schema Change | Discuss the effect a schema change to the trade table in the above system e.g. a sequence number column was added to both trade and quote tables. How you would plan a turnover to update the schema. |
|  | Answer | If your data stream for trade/quote now contained some sequence number then you would need to make some changes to your feed handler to accurately transform the new messages. As well as this you would add the column to the schema file. Now everything will be set for the RDB but if you ever had an API that collected data and merged it from the HDB it would break. Or querying the hdb after the first write down with the new column occurs because different partitions will have different schemas.  To fix this, I would make the following changes while the system is offline.  - Add the column to both tables in the schema file. - Update feed handler to expect this data. - Go back and add this column to each date partition with null types for each entry. - Update any query code where necessary. |

**Exercise 2 - Debugging**

Debug the following

| Number | Component | Requirements |
| --- | --- | --- |
|  | TP/RDB problem | The below tickerplant log contains and error.   * + 1. Locate the error     2. Fix the error   Show how each was performed. |
| Ans. 1 |  | The second last entry was not enlisted.  Started debugging by checking the update function for each entry:  {0N! first each x} get .tp.log //.tp.log is logfile handle  Then I saw some unexpected results and drilled into it to see the un-enlisted values.  I removed the data and had another look and then I saw some data type errors as well.  2 additional entries with char type for sym and 1 entry where size was a float. |
|  | Splay Table problem | The 3 tables in the below zip file contain errors. Try to ascertain the error and recommend a suitable fix for each. |
| Ans. 3 |  | -T2 is missing a price for the final row. Could use the previous price for that sym, or a size weighted average over all vals/ -T1 is missing the .d file so it can’t create the table. Copy over the .d file from table with same meta, ie. T2. -T3’s sym column has not been enumerated. Enumerate the table and save it down again. |
| 3. | Blocking calls | In your current system you have a historical database and a single hdb kdb+ instance to host user queries. Users are complaining about slowness. What is the probable cause of the slowness and suggest 2 changes that could be made to alleviate the problem. Maintain a single entry point to the system if possible. |
| Ans. 3 |  | User queries are being executed in order on a single thread.   * Add slaves to the hdb process. * Add another replica hdb process that can take some of the load. You will also need some query router if you want to keep a single point of entry. * Additionally if some calls are not looking for data to be returned you could make them asynchronous. |
| 4. | Query Performance | Improve the performance of function  StringtoDate:{[x]{$[10h~abs type x;x:"D"$x;-14h~ type x;x:x;`date$x]}'[x]} given input: raze 500000#enlist("2010.01.01";2010.01.02). Do not use .Q.fu. |
|  |  | If string the cast using “D”, anything else (including date) cast with `date.  No need to do additional checks.  StringtoDate :{[x]{`date$x}'[x]} |

**Exercise 3 - API**

| Number | Task | Requirements |
| --- | --- | --- |
|  | Create a small script in another language to accomplish requirements listed. Select language from:   1. Unix shell 2. Perl 3. Python 4. Scala | The script should read in a csv file and publish each row to a tickerplant. |
| 1. | Answer | See attached python dir. |
| 2. | Create a small script in another language to accomplish requirements listed. Select language from:   1. C/C++ 2. Java 3. C# | The script should read in a csv file and publish each row to a tickerplant. |
| 2 | Answer | See Java dir attached. |
| 3. | Web Interface – HTML5 | Create a simple web interface using HTML5 that would be able to query a trade table, filtering on symbol input by user into a text box and display the results in table. |
| 3 | Answer | See html dir attached. |