

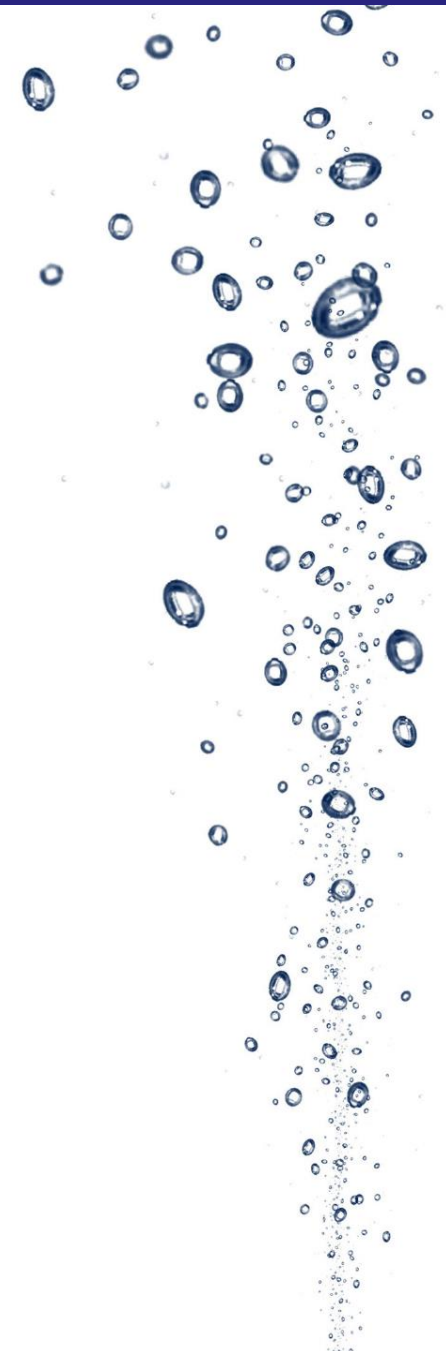


Grafana and kdb+

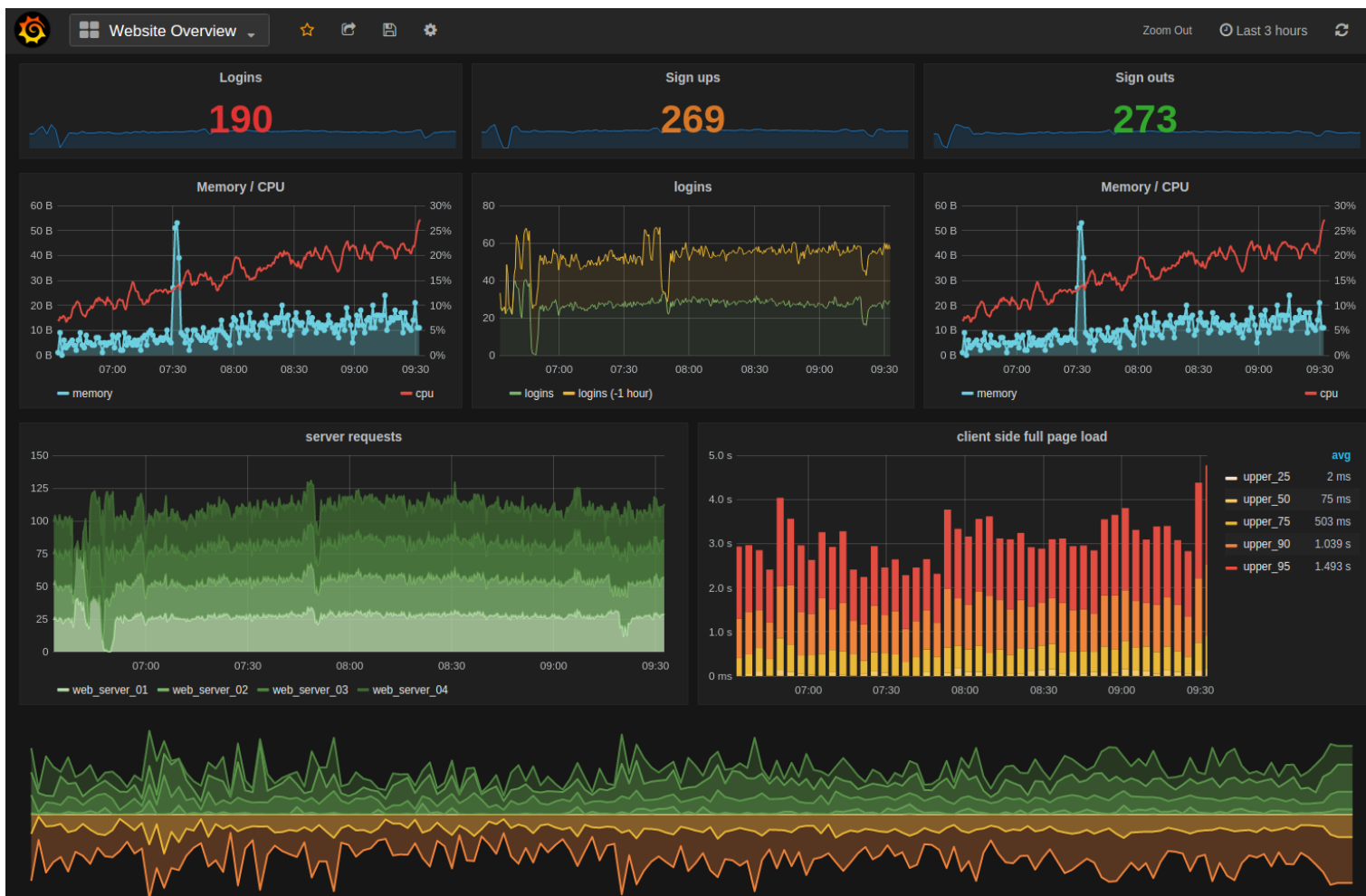
Experts in fast data solutions  
for demanding environments

# Overview

1. What is Grafana?
2. Simple JSON Extension
3. Objectives for a New Adaptor
4. Version 1.0
5. Demonstration
6. Development Road Map
7. Questions



# What is Grafana?

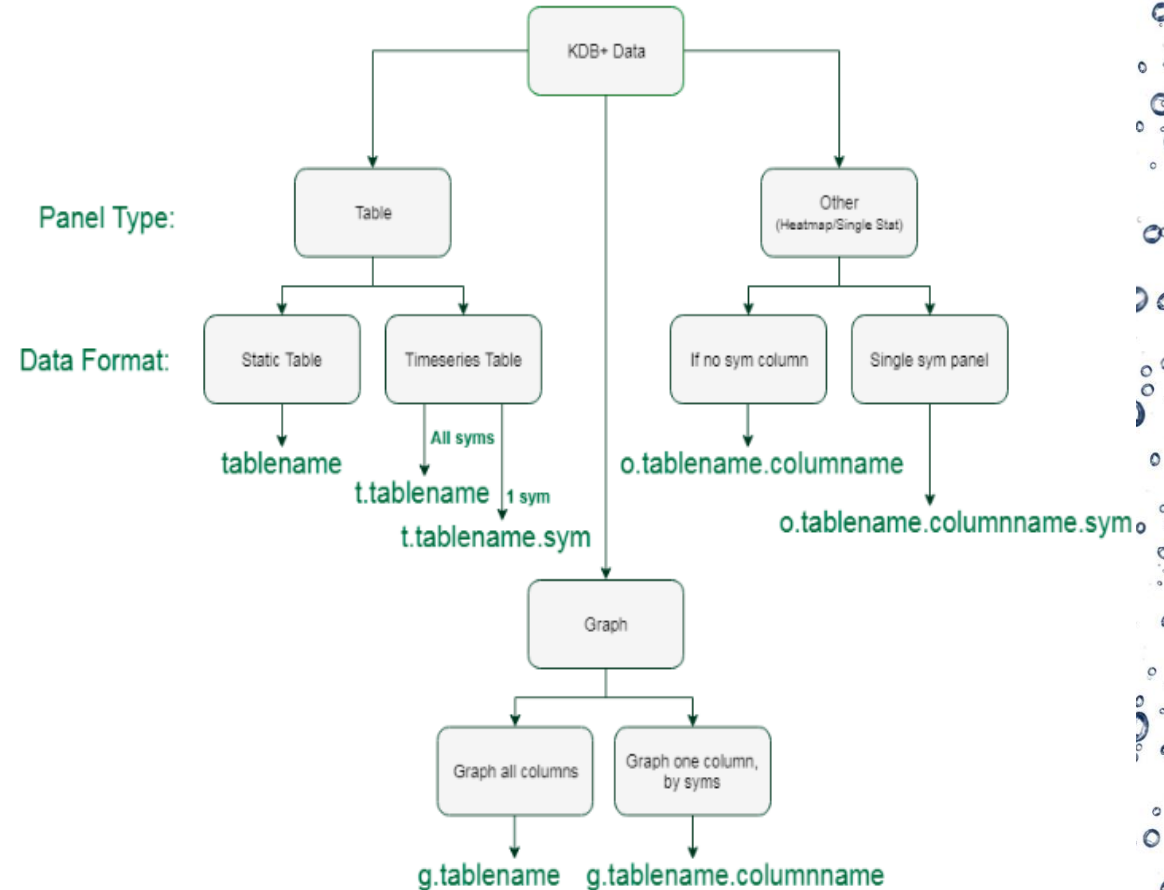


- First released in 2014
- Data visualisation tool
- Open source
- Multi-platform
- Supports dev of:
  - a) Data-sources
  - b) Panels
  - c) Apps



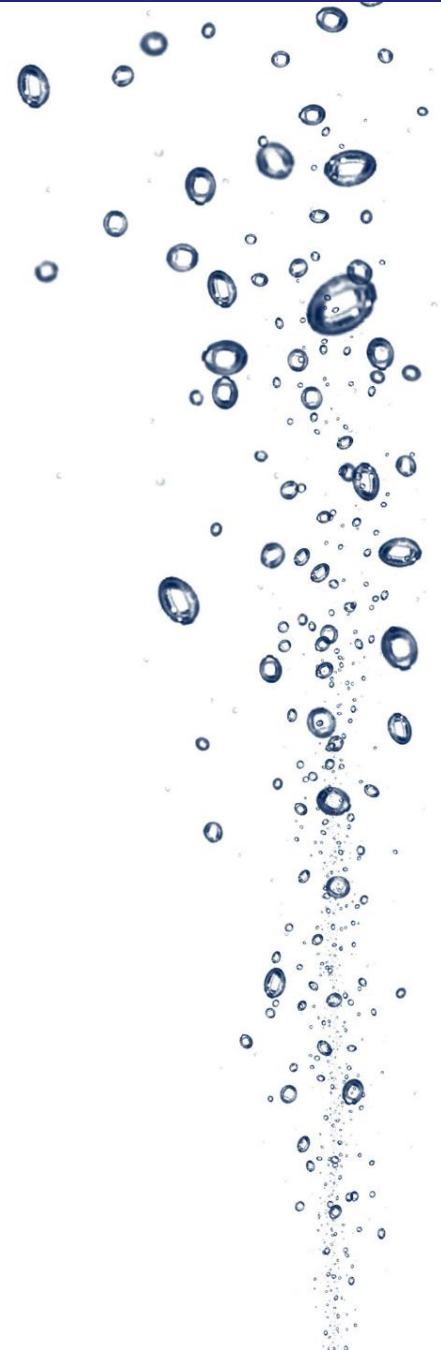
# Initial kdb+ Grafana Adaptor – Simple JSON

- Released in late 2018
- Leverages Simple JSON Grafana data-source (HTTP)
- Uses .j.k. kdb+ function to translate inbound requests to dictionaries
- Grafana.q script needs to be run to set up



# Requirements for Full Grafana Plugin

- Easily usable by someone with no understanding of q
- Minimal server side code
- User friendly query-builder
- Support for aggregation options and column aliasing
- Support for returning the results of custom queries, functions etc
- Use of web-socket protocol





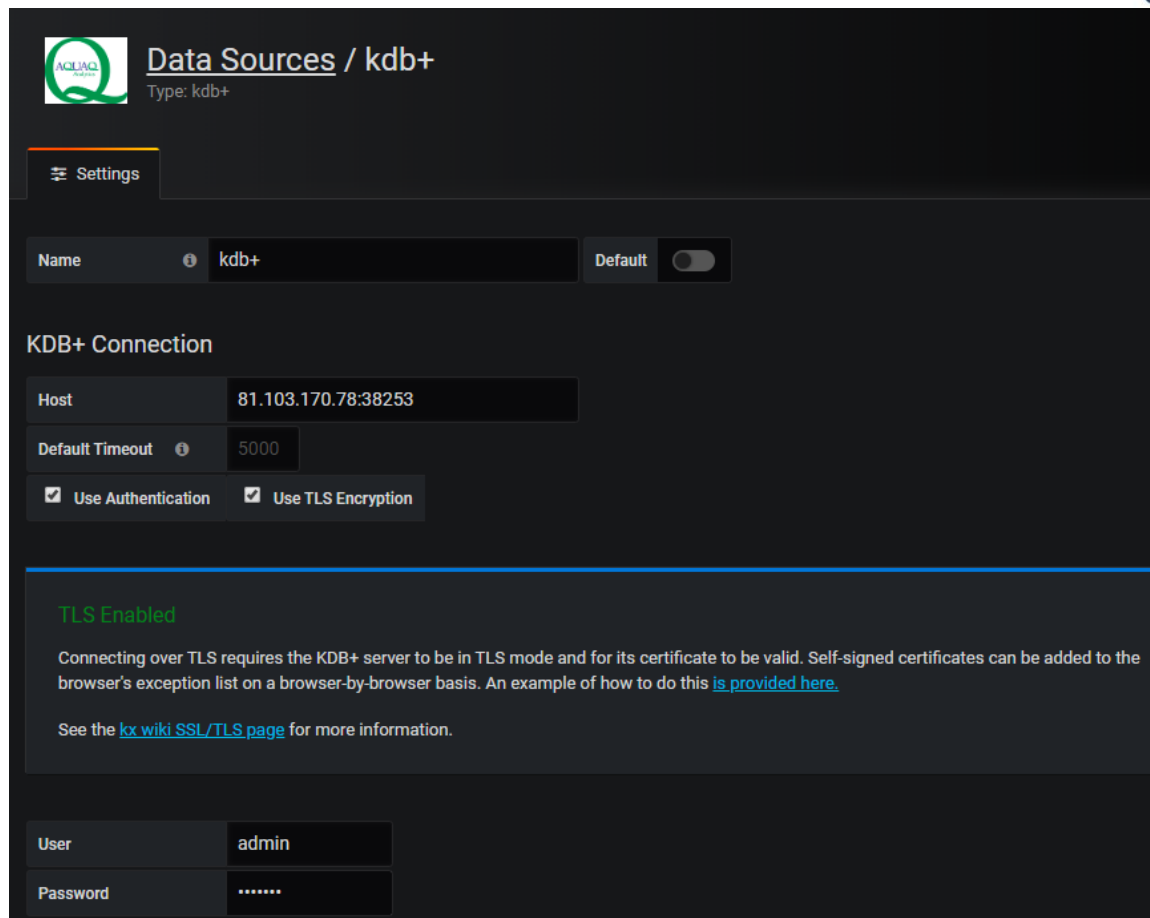
# Kdb+ Grafana Adaptor - Development

- Work started in mid-2019
- Involved a number of people within AquaQ
- From late Jan 2020 there were weekly internal release cycles to fine tune development
- Automated unit testing of each element of the kdb+ function
- 'Test Complete' used to automatically test the plugin's UI prior to each release



# kdb+ Grafana Plugin Version 1.0

- Added to Grafana plugin repository on 14<sup>th</sup> April 2020
- Available from:
  - Grafana:  
<https://grafana.com/grafana/plugins/aquaanalytics-kdbadaptor-datasource>
  - AquaQ Github:  
<https://github.com/AquaQAnalytics/grafana-kdb-datasource-ws>

A screenshot of the Grafana web interface showing the configuration page for the 'kdb+' data source. The page has a dark theme. At the top, there's a header with the AQUAQ logo and the text 'Data Sources / kdb+' and 'Type: kdb+'. Below this is a 'Settings' tab. The main configuration area includes a 'Name' field set to 'kdb+', a 'Default' toggle switch, and a 'KDB+ Connection' section. This section contains a 'Host' field with the value '81.103.170.78:38253', a 'Default Timeout' field set to '5000', and two checked checkboxes: 'Use Authentication' and 'Use TLS Encryption'. A green message box states 'TLS Enabled' and provides instructions on connecting over TLS. At the bottom, there are fields for 'User' (set to 'admin') and 'Password' (masked with dots).

**Data Sources / kdb+**  
Type: kdb+

Settings

Name *i* kdb+ Default ☐

**KDB+ Connection**

Host 81.103.170.78:38253

Default Timeout *i* 5000

☒ Use Authentication ☒ Use TLS Encryption

**TLS Enabled**

Connecting over TLS requires the KDB+ server to be in TLS mode and for its certificate to be valid. Self-signed certificates can be added to the browser's exception list on a browser-by-browser basis. An example of how to do this [is provided here](#).

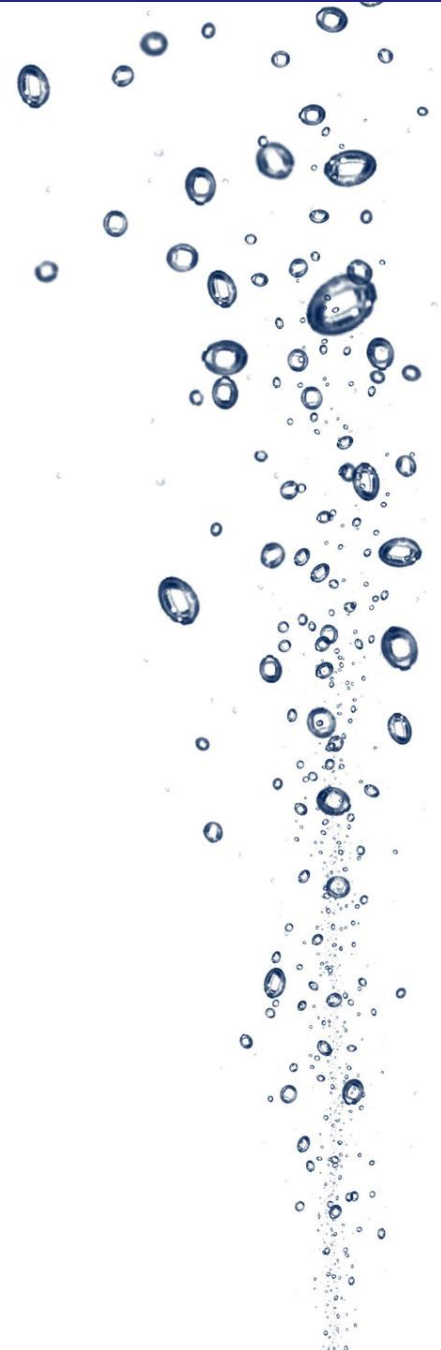
See the [kx wiki SSL/TLS page](#) for more information.

User admin

Password .....

# Version 1.0 - Features

- Uses web-socket protocol to communicate with kdb+
- Only configuration required on kdb+ instances is the declaration of a web-socket handler (.z.ws)
- Support for TLS protocol to enable secure connections
- Free for query builder that supports grouping/aggregation and conflation
- Functional query builder to allow users with knowledge of q to write their own more complex queries





# kdb+ Plugin – Integration with kdb+

- .z.ws needs to be configured:

```
q).z.ws:{ds:-9!x;neg[.z.w] -8! `o`ID!(@[value;ds[`i];{`$""",x}];ds[[]ID])}
```

- Requests sent down in a form which enables them to easily be translated to a kdb+ dictionary.
- Function to process the dictionary passed down with each request
- Both the function and dictionary vary depending on the type of query being executed
- Function transforms query results into format expected by Grafana (largely)



# kdb+ Function Structure

- PARENT EXECUTE
- TAKES DICTIONARY AS ARGUMENT
- DEFINES CHILD FUNCTIONS
- WBUILD, CBUILD, BBUILD
- CONC, CONB
- FORMAT

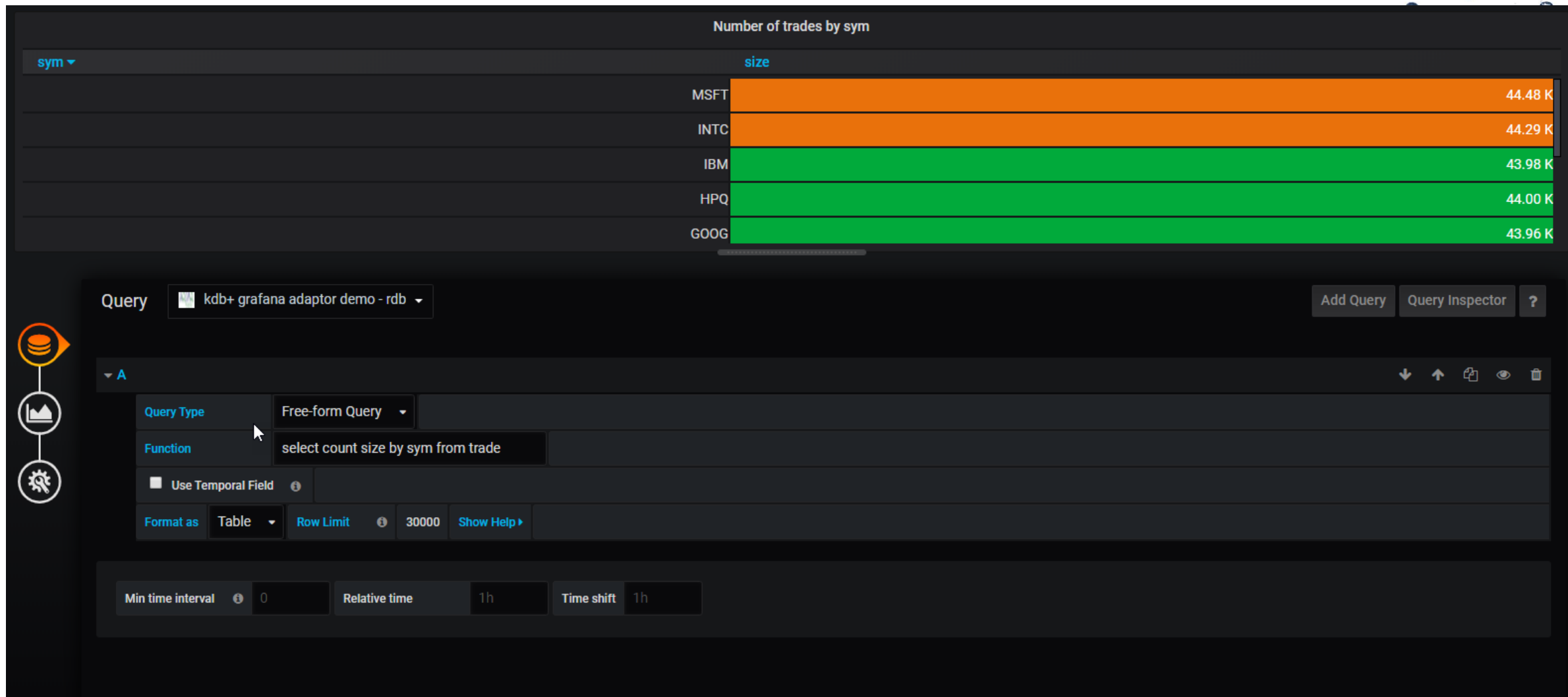
```
where      | (">";`amount;100)
by         | ()
table      | `trade
column     | ("avg";`price)
temporal_field | `time
temporal_range | 09:00:00 17:00:00
query_type  | `timeseries
grouping    | `sym
conflation  | `val`agg! ("10000";"avg")
query      | `type`value!(`select;())
```

```
wbuild:([d]
//Where clause builder
tr:enlist ("within";d[`temporal_field];enlist d[`temporal_range]); //builds where for Grafana time window
wc:tr,([x]${0=type first x;x;enlist x})d[`where]; //prepends time where to any wheres passed in dictionary
:([x](value first x),1_x) each wc; //converts string keywords into their q object counterparts
);
```

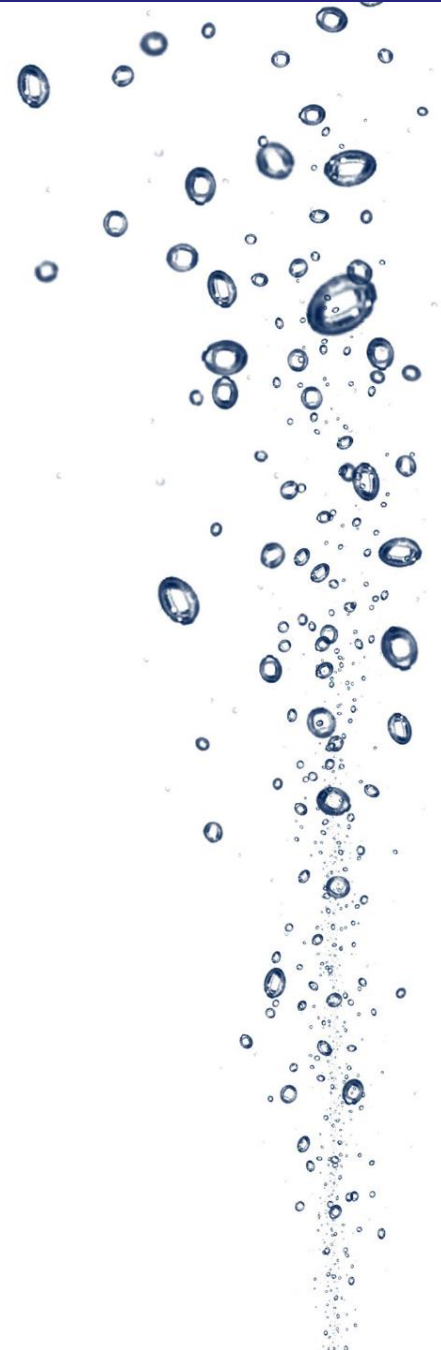
# kdb+ Plugin – Query Builder



# kdb+ Plugin – Free Form Queries



# Demonstration





# Development Roadmap

- Support for Grafana variables
- Integration with candlestick panel type
- Support for complex grouping in free-form queries
- Other 'behind the scenes' improvements
- Anything else?



Thanks!

Q+A

21<sup>st</sup> May: kdb+ 4.0

28<sup>th</sup> May: kdb+ in Containers

