influxdata®

Understanding the New InfluxDB Storage Engine

Anais Dotis-Georgiou

January 2023

Anais Dotis-Georgiou
Developer Advocate

LinkedIn

Agenda

- What is the new InfluxDB Engine?
- What requirements does the new InfluxDB Engine meet?
    - Understanding the Apache Ecosystem
- Offerings and Release Timeline
- New InfluxDB Cloud Features
- SQL support
- Interoperability plans
- Survey
- Resources

The new storage engine that powers InfluxDB Cloud

InfluxDB’s new storage engine is built on

- Rust
- Apache Arrow
- Apache Parquet
- Arrow Flight
- DataFusion

Release Details

- Release Jan 31st
- Available in InfluxDB Cloud on AWS in:
  - AWS us-east-1
  - AWS eu-central-1

Requirements for the new storage engine

Requirement/Feature	Rust	Arrow	DataFusion	Parquet
1. No limits on cardinality. Write any kind of event data and don't worry about what a tag or field is.	X	X	X	X
2. Best-in-class performance on analytics queries in addition to our already well-served metrics queries.	X	X	X	X
3. Separate compute from storage and tiered data storage. The DB should use cheaper object storage as its long-term durable store.			X	X
4. Operator control over memory usage. The operator should be able to define how much memory is used for each buffering, caching, and query processing.			X	
5. Bulk data export and import.				X
6. Broader ecosystem compatibility. Where possible, we should aim to use and embrace emerging standards in the data and analytics ecosystem.	X	X	X	X
7. Run at the edge and in the datacenter. Federated by design.	X			X

Rust and InfluxDB Requirements

Rust and Requirement 1.

Requirement: No limits on cardinality. Write any kind of event data and don't worry about what a tag or field is.

Rust Contributions:

- InfluxDB’s new storage engine is built on the Rust implementation of Apache Arrow which contributes heavily to meeting this requirement.
- Handling unlimited cardinality use cases requires non-trivial CPU during query processing. Rust supports optimizing resources for increased performance.

Rust and Requirement 2.

Requirement: Best-in-class performance on analytics queries in addition to our already well-served metrics queries.

Rust Contributions:

- Arrow and DataFusion are built on Rust.

Rust and Requirement 4.

Requirement: Operator control over memory usage. The operator should be able to define how much memory is used for each buffering, caching, and query processing.

Rust Contributions:

- Rust is used for memory control.

Rust and Requirement 6.

Requirement: Broader ecosystem compatibility. Where possible, we should aim to use and embrace emerging standards in the data and analytics ecosystem.

Rust Contributions:

- Rust helps support the implementation of Arrow, DataFusion, and Parquet.

Rust and Requirement 7.

Requirement: Run at the edge and in the datacenter. Federated by design.

Rust Contributions:

- Optimizing your memory usage with Rust, means that InfluxDB Cloud will also contain these memory optimizations at the edge or in the datacenter.

Rust and Requirement 7.

Requirement: Run at the edge and in the datacenter. Federated by design.

Rust Contributions:

- Optimizing your memory usage with Rust, means that InfluxDB Cloud will also contain these memory optimizations at the edge or in the datacenter.

Arrow and InfluxDB Requirements

Arrow and Requirement 1.

Requirement: No limits on cardinality. Write any kind of event data and don't worry about what a tag or field is.

Arrow Contributions:

- Apache Arrow overcomes memory challenges associated with large-cardinality use cases by providing efficient columnar data exchange.

Advantages of Columnar Data Storage (sidebar)

Sidebar–Advantages of Columnar Data Storage

```
measurement1,tag1=tagvalue1 field1=1i timestamp1
measurement1,tag1=tagvalue2 field1=2i timestamp2
measurement1,tag2=tagvalue3 field1=3i timestamp3
measurement1,tag1=tagvalue1,tag2=tagvalue3 field1=4i,field2=true timestamp4
measurement1, field1=1i timestamp5
```

Sidebar–Advantages of Columnar Data Storage

Name: measurement1					
field1	**field2**	**tag1**	**tag2**	**tag3**	**time**
1i	null	tagvalue1	null	null	timestamp1
2i	null	tagvalue2	null	null	timestamp2
3i	null	null	tagvalue3	null	timestamp3
4i	true	tagvalue1	tagvalue3	tagvalue4	timestamp4
1i	null	null	null	null	timestamp5

Sidebar–Advantages of Columnar Data Storage

1i	2i	3i	4i	1i
null	null	null	true	null
tagvalue1	tagvalue2	null	tagvalue1	null
null	null	tagvalue3	tagvalue3	null
null	null	null	tagvalue4	null
timestamp1	timestamp2	timestamp3	timestamp4	timestamp5

```
1i, 2i, 3i, 4i, 1i;
null, null, null, true, null;
tagvalue1, tagvalue2, null, tagvalue1, null;
null, null, null, tagvalue3, tagvalue3, null;
null, null, null, tagvalue4, null;
timestamp1, timestamp2, timestamp3, timestamp4, timestamp5.
```

Arrow and Requirement 2.

Requirement: Best-in-class performance on analytics queries in addition to our already well-served metrics queries.

Arrow Contributions:

- Arrow offers best in class performance on analytics through the memory optimizations and efficient data exchange.

Arrow and Requirement 3.

Requirement: Separate compute from storage and tiered data storage. The DB should use cheaper object storage as its long-term durable store.

Arrow Contributions:

- Arrow provides the in-memory columnar storage while Parquet will provide the column-oriented data file format on disk.

Arrow and Requirement 4.

Requirement: Operator control over memory usage. The operator should be able to define how much memory is used for each buffering, caching, and query processing

Arrow Contributions:

- The Rust implementation of Apache Arrow provides fine grained memory control.

Arrow and Requirement 6.

Requirement: Broader ecosystem compatibility. Where possible, we should aim to use and embrace emerging standards in the data and analytics ecosystem.

Arrow Contributions:

- Leveraging Arrow is easier with the 12 libraries it supports for C, C++, Java, JavaScript, Python, Ruby, and more.

DataFusion and InfluxDB Requirements

DataFusion and Requirement 1.

Requirement: No limits on cardinality. Write any kind of event data and don't worry about what a tag or field is.

DataFusion Contributions:

- What use is unlimited cardinality data if you can’t query it? DataFusion provides the query, processing, and transformation of this data.

DataFusion and Requirement 2.

Requirement: Best-in-class performance on analytics queries in addition to our already well-served metrics queries.

DataFusion Contributions:

- What use is unlimited cardinality data if you can't query it? DataFusion provides the query, processing, and transformation of this data.

DataFusion and Requirement 3.

Requirement: Separate compute from storage and tiered data storage. The DB should use cheaper object storage as its long-term durable store.

DataFusion Contributions:

- DataFusion enables fast query against data stored on cheaper object store and separate compute.

DataFusion and Requirement 6.

Requirement: Broader ecosystem compatibility. Where possible, we should aim to use and embrace emerging standards in the data and analytics ecosystem.

DataFusion Contributions:

- DataFusion supports both a postgres compatible SQL and DataFrame API.

Parquet and InfluxDB Requirements

Parquet and Requirement 2.

Requirement: Best-in-class performance on analytics queries in addition to our already well-served metrics queries.

Parquet Contributions:

- Efficient compression and interoperability with ML and analytics tooling.

Parquet and Requirement 3.

Requirement: Separate compute from storage and tiered data storage. The DB should use cheaper object storage as its long-term durable store.

Parquet Contributions:

- Parquet files take up little disk space and are fast to scan.

Parquet and Requirement 5.

Requirement: Bulk data export and import.

Parquet Contributions:

- Parquet files enable bulk data export and import.

Parquet and Requirement 5.

Requirement: Broader ecosystem compatibility. Where possible, we should aim to use and embrace emerging standards in the data and analytics ecosystem.

Parquet Contributions:

- Parquet offers interoperability with modern ML and analytics tools.

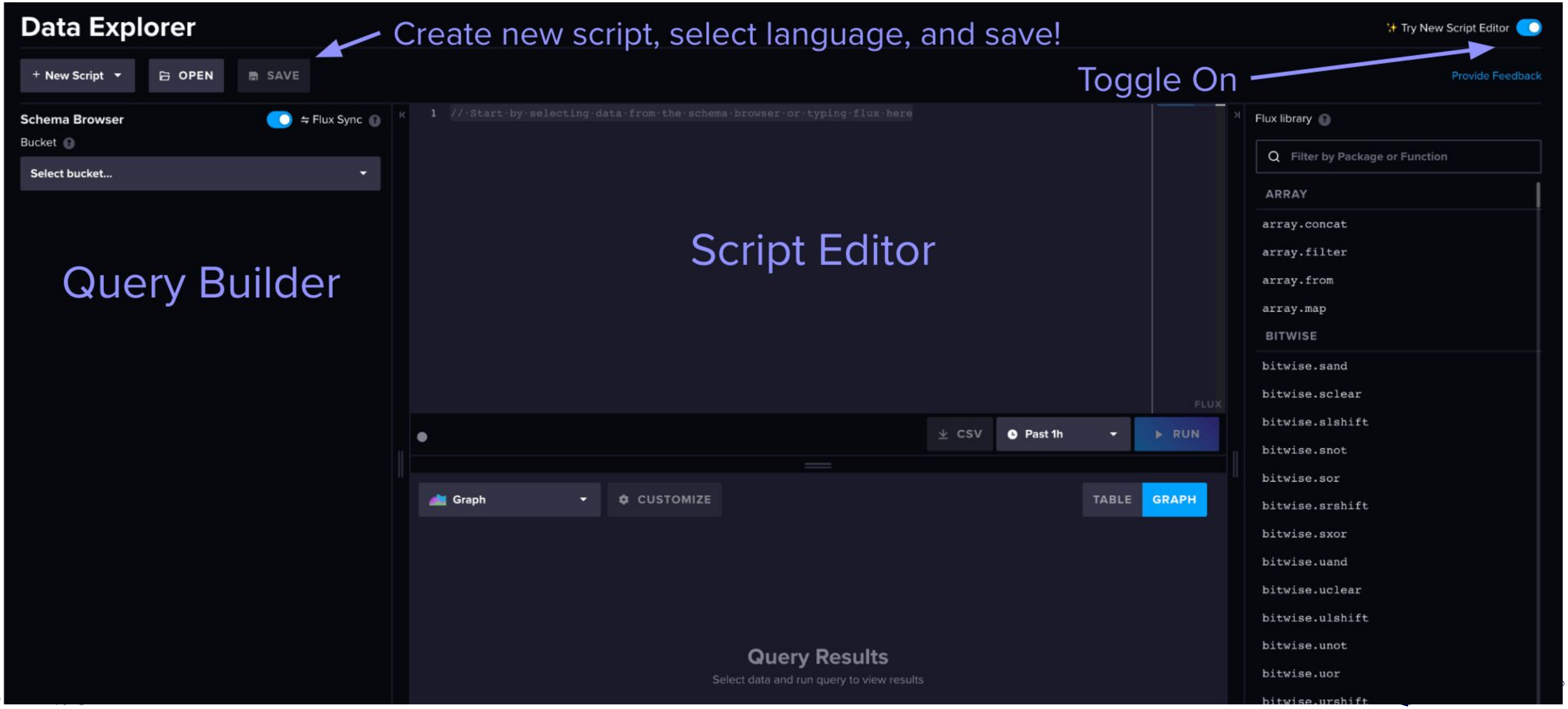
Parquet and Requirement 5.

Requirement: Run at the edge and in the datacenter. Federated by design.

Parquet Contributions:

- Because Parquet files are so efficient, they will facilitate and increase the capacity for data storage at the edge.

New Data Explorer



Data Explorer

Create new script, select language, and save!

Try New Script Editor

Toggle On

Provide Feedback

+ New Script | OPEN | SAVE

Schema Browser | Flux Sync

Bucket

Select bucket...

Query Builder

```
// Start by selecting data from the schema browser or typing flux here
```

Script Editor

FLUX

CSV | Past 1h | RUN

Graph | CUSTOMIZE | TABLE | GRAPH

Query Results

Select data and run query to view results

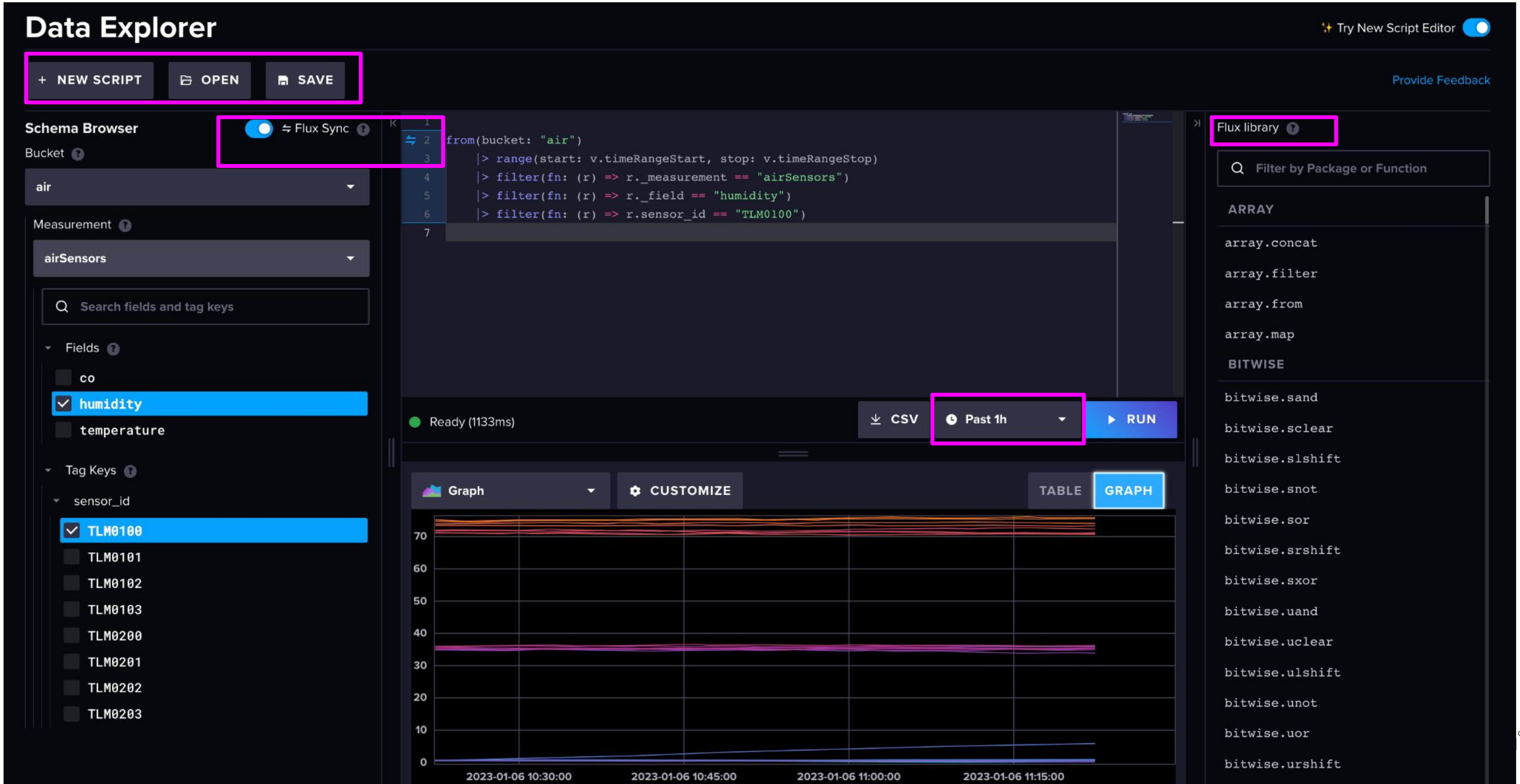
Flux library

Filter by Package or Function

ARRAY
- array.concat
- array.filter
- array.from
- array.map

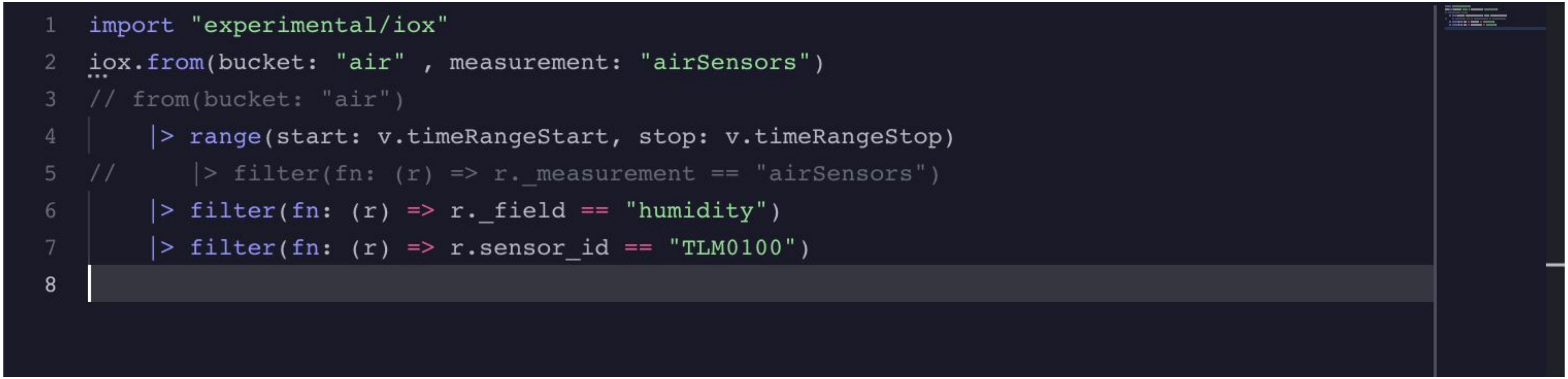
BITWISE
- bitwise.sand
- bitwise.sclear
- bitwise.slshift
- bitwise.snot
- bitwise.sor
- bitwise.srshift
- bitwise.sxor
- bitwise.uand
- bitwise.uclear
- bitwise.ulshift
- bitwise.unot
- bitwise.uor
- bitwise.urshift

New Data Explorer



```
from(bucket: "air")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r._measurement == "airSensors")
  |> filter(fn: (r) => r._field == "humidity")
  |> filter(fn: (r) => r.sensor_id == "TLM0100")
```

iox.from() vs from()



```
import "experimental/iox"
iox.from(bucket: "air" , measurement: "airSensors")
// from(bucket: "air")
    |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
//      |> filter(fn: (r) => r._measurement == "airSensors")
    |> filter(fn: (r) => r._field == "humidity")
    |> filter(fn: (r) => r.sensor_id == "TLM0100")
```

iox.from() vs from()

```
from(bucket: "anais-iox")
    |> range(start: 2022-12-01T19:05:41.000Z, stop: now())
    |> filter(fn: (r) => r._measurement == "airSensors")
    |> filter(fn: (r) => r._field == "temperature")
    |> filter(fn: (r) => r.sensor_id == "TLM0100")
```

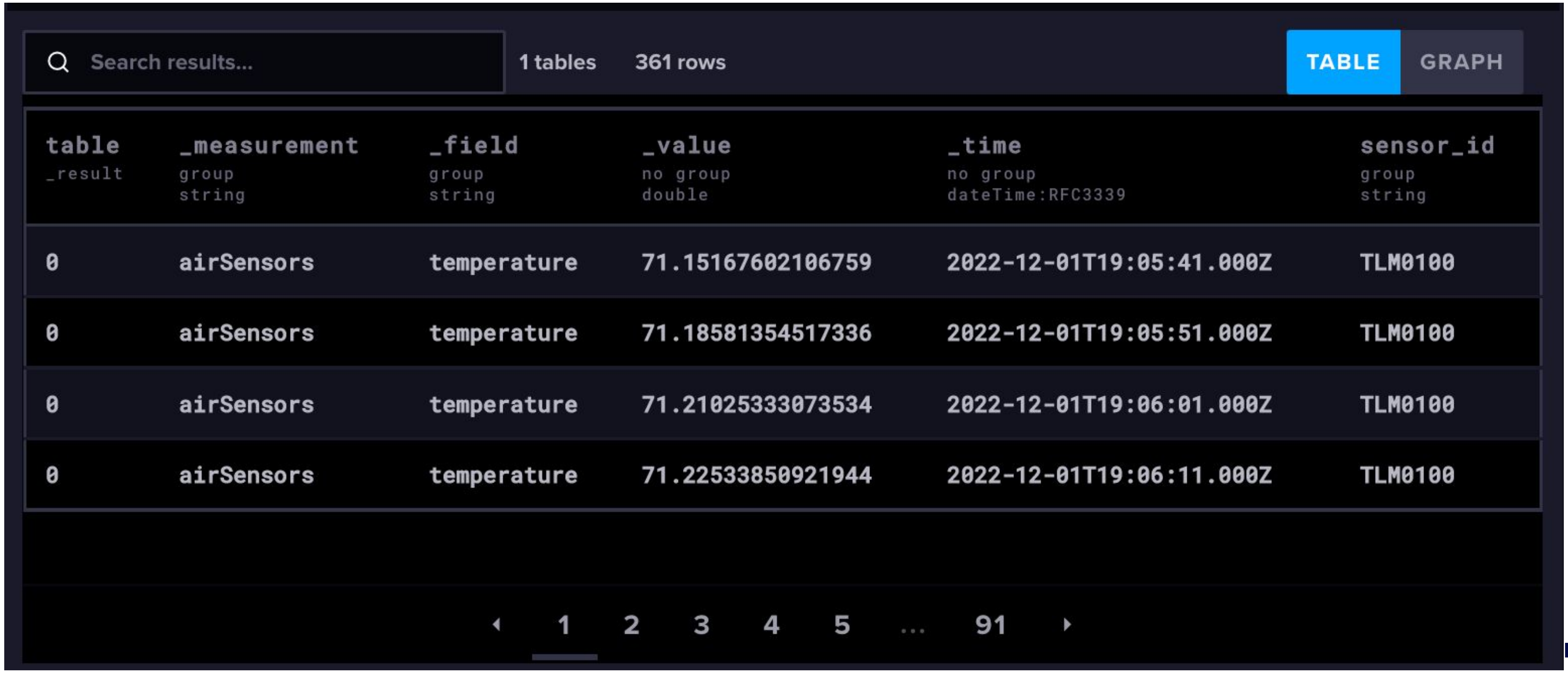
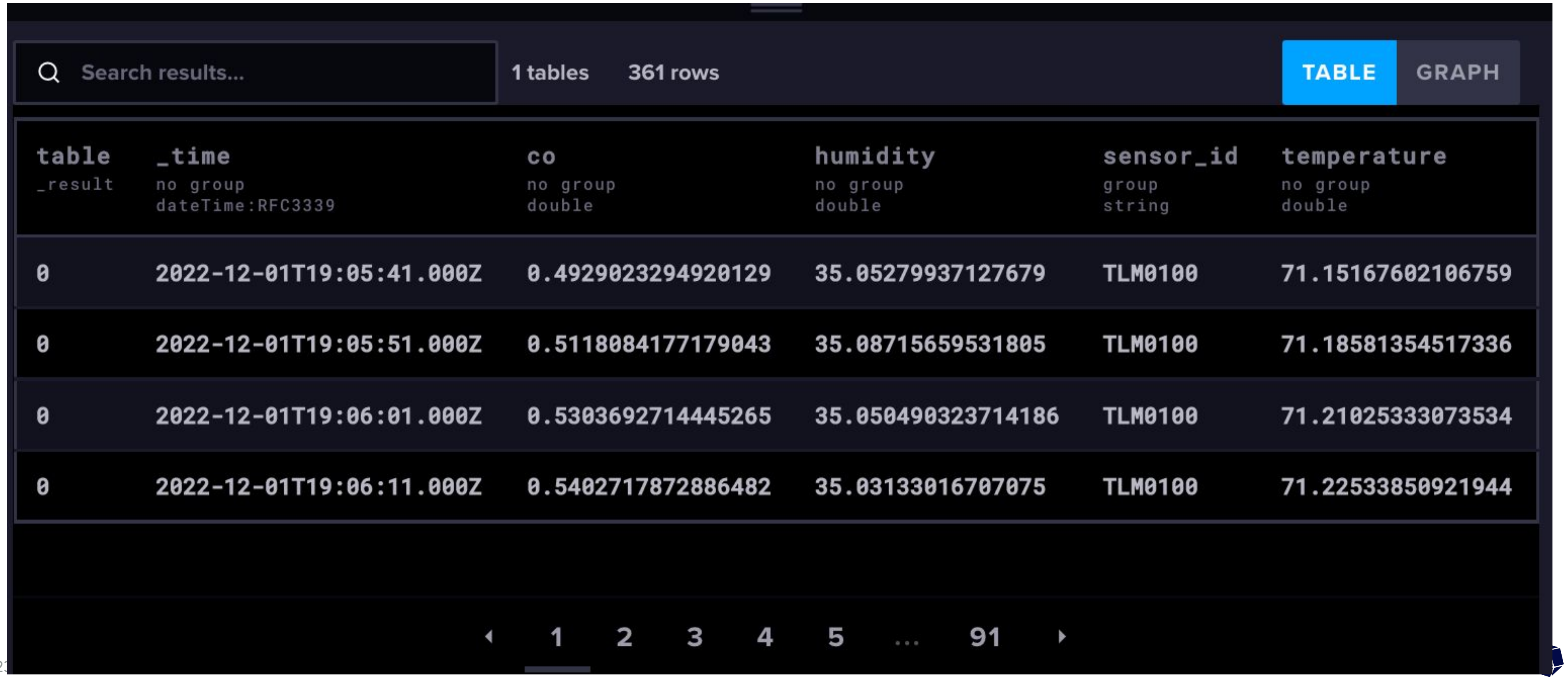


table _result	_measurement group string	_field group string	_value no group double	_time no group dateTime:RFC3339	sensor_id group string
0	airSensors	temperature	71.15167602106759	2022-12-01T19:05:41.000Z	TLM0100
0	airSensors	temperature	71.18581354517336	2022-12-01T19:05:51.000Z	TLM0100
0	airSensors	temperature	71.21025333073534	2022-12-01T19:06:01.000Z	TLM0100
0	airSensors	temperature	71.22533850921944	2022-12-01T19:06:11.000Z	TLM0100

nfluxdata®

iox.from() vs from()

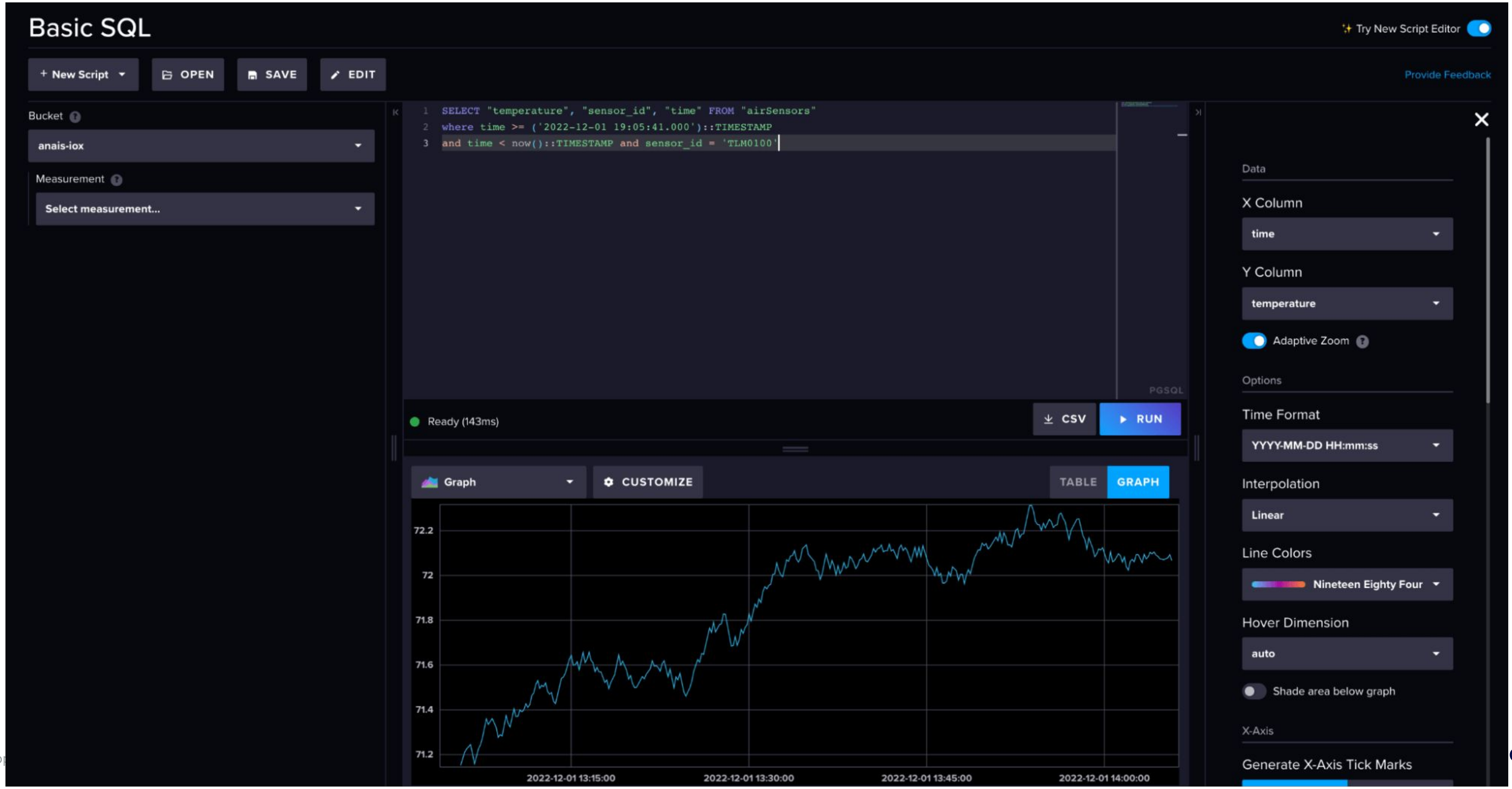
```
import "experimental/iox"
data = iox.from(bucket: "anais-iox", measurement: "airSensors")
|> range(start: 2022-12-01T19:05:41.000Z, stop: now())
|> filter(fn: (r) => r.sensor_id == "TLM0100")
|> yield()
```



1 tables 361 rows

table _result	_time no group dateTime:RFC3339	co no group double	humidity no group double	sensor_id group string	temperature no group double
0	2022-12-01T19:05:41.000Z	0.4929023294920129	35.05279937127679	TLM0100	71.15167602106759
0	2022-12-01T19:05:51.000Z	0.5118084177179043	35.08715659531805	TLM0100	71.18581354517336
0	2022-12-01T19:06:01.000Z	0.5303692714445265	35.050490323714186	TLM0100	71.21025333073534
0	2022-12-01T19:06:11.000Z	0.5402717872886482	35.03133016707075	TLM0100	71.22533850921944

SQL Support



Bucket: anais-iox

Measurement: Select measurement...

```
SELECT "temperature", "sensor_id", "time" FROM "airSensors"
where time >= ('2022-12-01 19:05:41.000')::TIMESTAMP
and time < now()::TIMESTAMP and sensor_id = 'TLM0100'
```

Ready (143ms)

Data

X Column: time

Y Column: temperature

Adaptive Zoom: on

Options

Time Format: YYYY-MM-DD HH:mm:ss

Interpolation: Linear

Line Colors: Nineteen Eighty Four

Hover Dimension: auto

Shade area below graph: off

X-Axis

Generate X-Axis Tick Marks

SQL Functions Supported by Jan 31

- **Supported Statements:** SELECT, FROM, WHERE, GROUP BY, ORDER BY, JOIN (left and inner), WITH clause, HAVING, UNION, LIMIT, OVER
- **Subqueries:** EXISTS, NOT EXISTS, IN, NOT IN
- **Agg Functions:** COUNT(), AVG(), MEAN(), SUM(), MIN(), MAX()
- **Time Series Functions:** TIME_BUCKET_GAPFILL(), DATEBIN(), NOW()
- **Other:** EXPLAIN, EXPLAIN ANALYZE

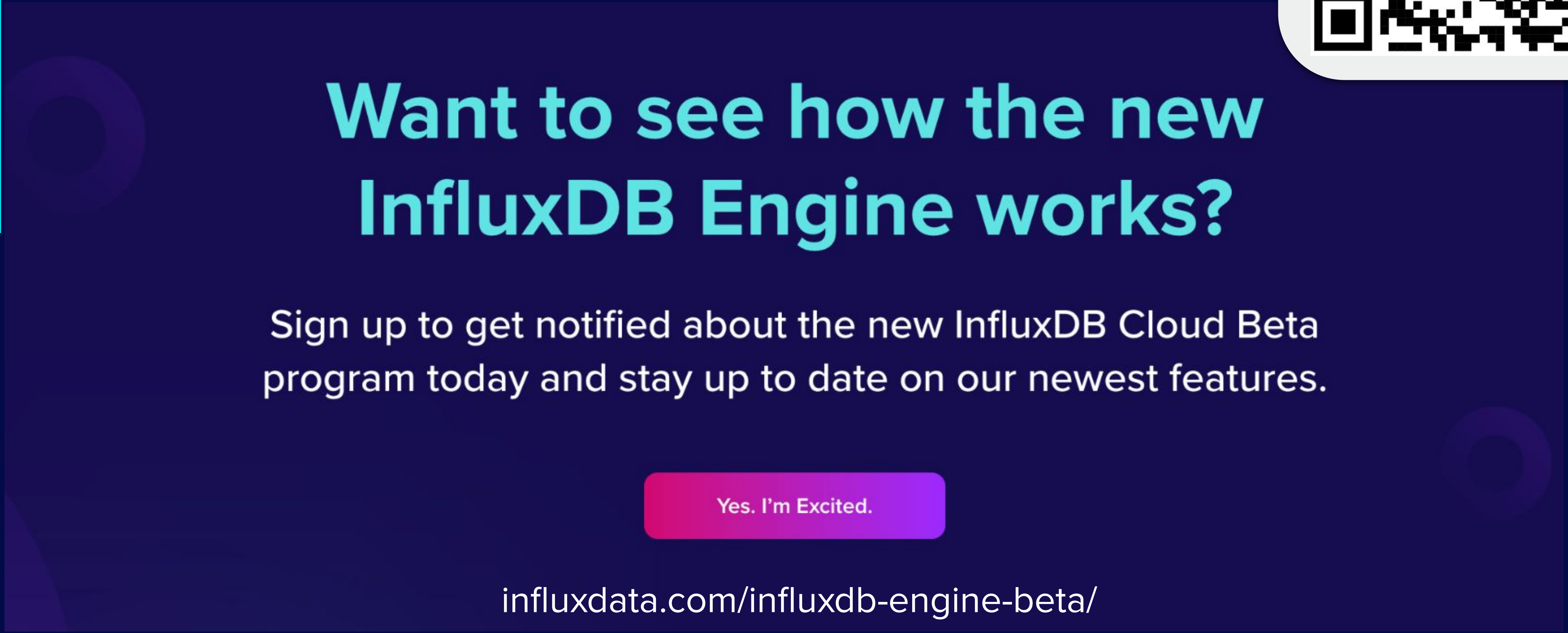
Interoperability plans

- FlightSQL plugins (Timeline TBD):
  - Apache Superset
  - Tableau
  - PowerBI
  - Grafana

Survey

- Please take 5 minutes to fill out this anonymous survey. Your feedback is extremely valuable to us.
- https://bit.ly/3imWP7Y

Get Started



Want to see how the new InfluxDB Engine works?

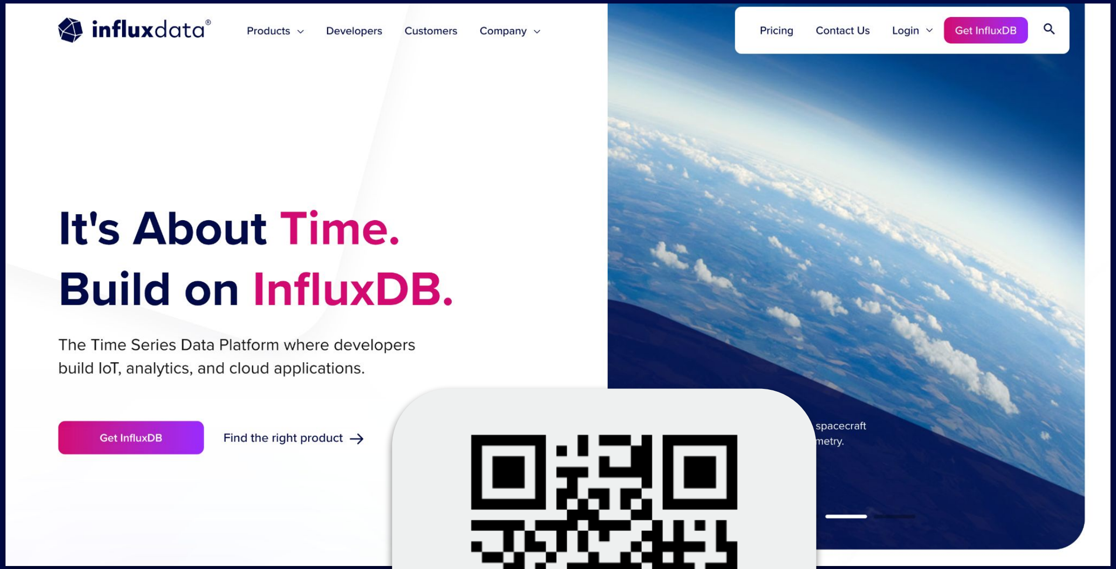
Sign up to get notified about the new InfluxDB Cloud Beta program today and stay up to date on our newest features.

influxdata.com/influxdb-engine-beta/

InfluxDB Community Slack workspace

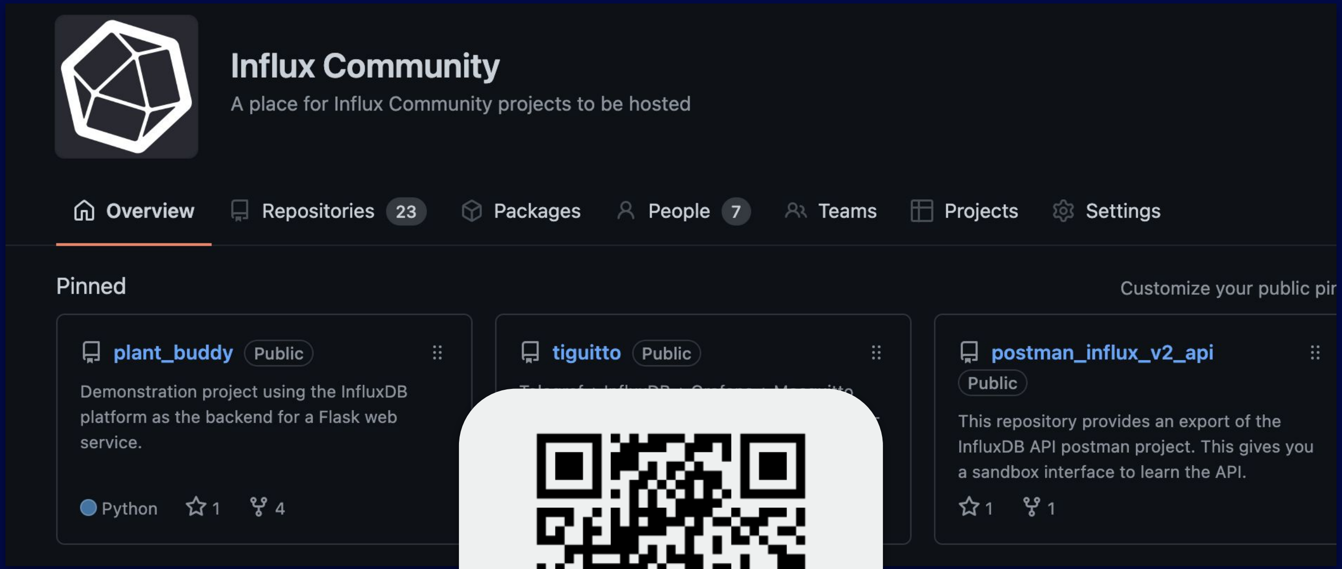
Please join us in the InfluxDB Community Slack at www.influxdata.com/slack.

To participate in conversations, join the #influxdb_iox channel.



It's About Time.
Build on InfluxDB.

The Time Series Data Platform where developers build IoT, analytics, and cloud applications.



Influx Community

A place for Influx Community projects to be hosted

plant_buddy Public

Demonstration project using the InfluxDB platform as the backend for a Flask web service.

Python 1 4

tiguitto Public

postman_influx_v2_api Public

This repository provides an export of the InfluxDB API postman project. This gives you a sandbox interface to learn the API.

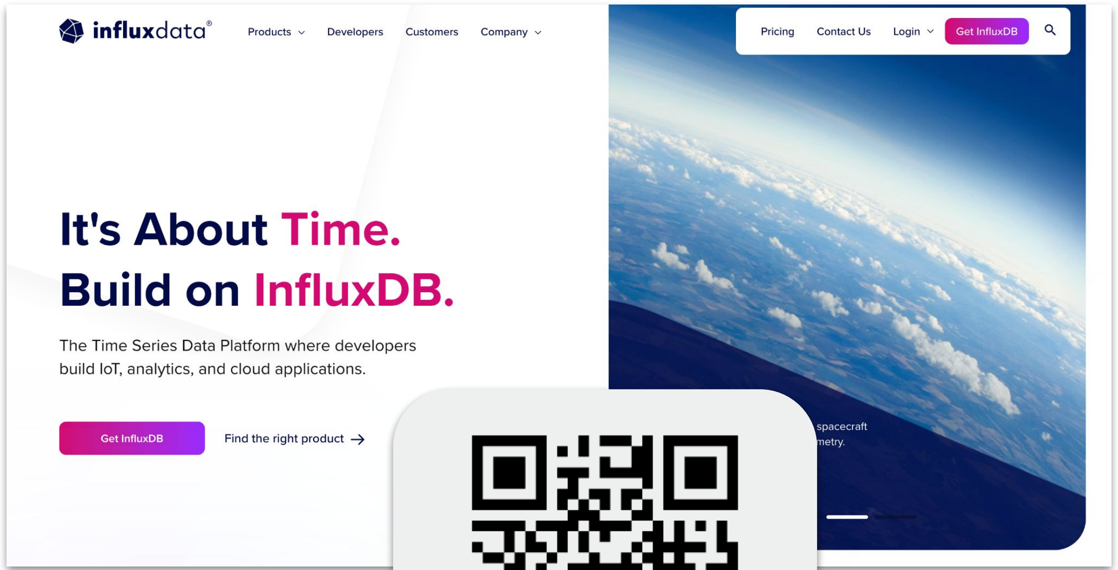
1 1

influxdata.com

https://github.com/InfluxCommunity

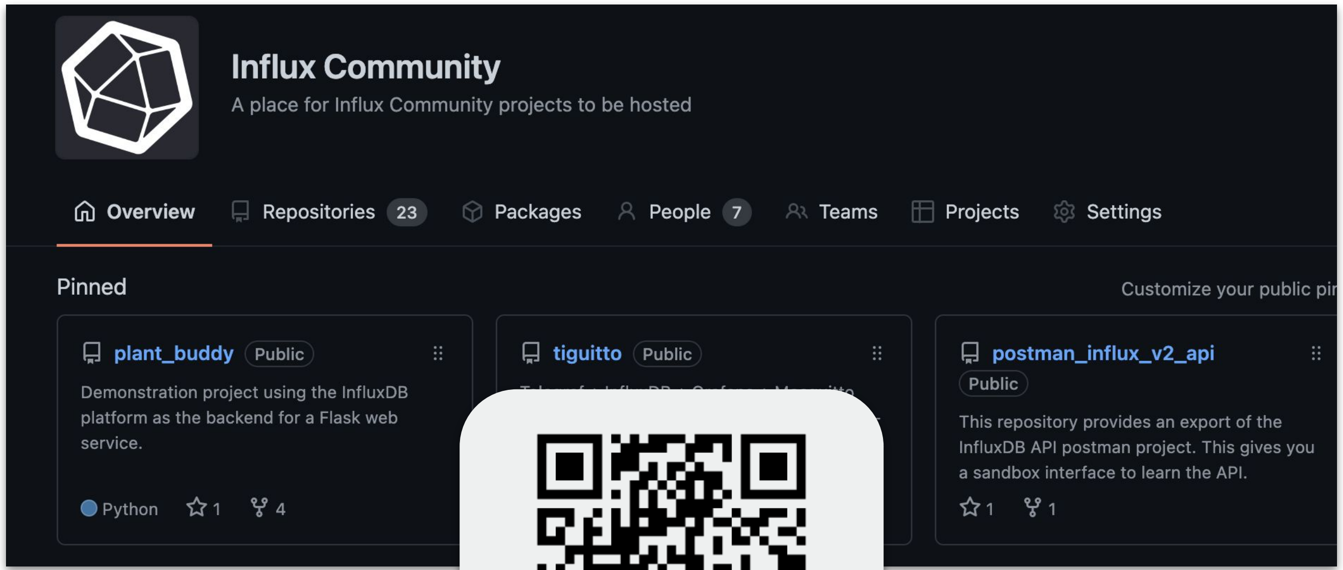
Try it yourself

Get Started



It's About Time.
Build on InfluxDB.

The Time Series Data Platform where developers build IoT, analytics, and cloud applications.



Influx Community

A place for Influx Community projects to be hosted

Pinned

plant_buddy Public

Demonstration project using the InfluxDB platform as the backend for a Flask web service.

Python 1 4

tiguitto Public

postman_influx_v2_api Public

This repository provides an export of the InfluxDB API postman project. This gives you a sandbox interface to learn the API.

1 1

Related Blogs

- Understanding InfluxDB IOx and the Commitment to Open Source
- Querying Data in InfluxDB using Flux and SQL
- Intro to InfluxDB IOx
- Welcome to InfluxDB IOx: InfluxData's New Storage Engine
- The Journey of InfluxDB by Paul Dix
- InfluxData Deploys Next-Generation InfluxDB Time Series Engine with Unlimited Scale
- Announcing InfluxDB IOx - The Future Core of InfluxDB Built with Rust and Arrow
- Evolving InfluxDB into the Smart Data Platform for Time Series
- InfluxData is Building a Fast Implementation of Apache Arrow in Go Using c2goasm and SIMD
- On InfluxData's New Storage Engine. Q&A with Andrew Lamb
- Apache Arrow, Parquet, Flight and Their Ecosystem are a Game Changer for OLAP

Get Help + Resources!

Forums: community.influxdata.com

Slack: influxcommunity.slack.com

GH: github.com/InfluxCommunity

Book: awesome.influxdata.com

Docs: docs.influxdata.com

Blogs: influxdata.com/blog

InfluxDB University: influxdata.com/university

Questions?

influxdata®

THANK YOU