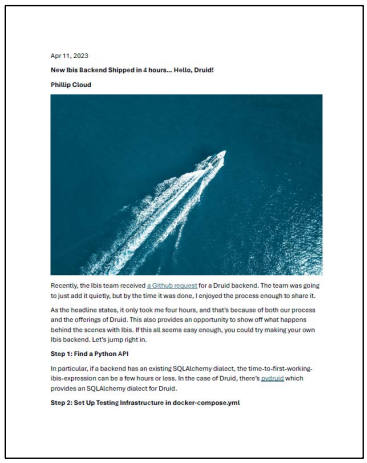
VOLTRON DATA

carahsoft



Apr 11, 2023

New Ibis Backend Shipped in 4 hours... Hello, Druid!

Phillip Cloud

Recently, the Ibis team received a GitHub request for a Druid backend. The team was going to just add it quietly, but by the time it was done, I enjoyed the process enough to share it.

As the headline states, it only took me four hours, and that's because of both our process and the offerings of Druid. This also provides an opportunity to show off what happens behind the scenes with Ibis. If this all seems easy enough, you could try making your own Ibis backend. Let's jump right in.

Step 1: Find a Python API

In particular, if a backend has an existing SQLAlchemy dialect, the time-to-first-working-ibis-expression can be a few hours or less. In the case of Druid, there's pydruid which provides an SQLAlchemy dialect for Druid.

Step 2: Set Up Testing Infrastructure in docker-compose.yml

New Ibis Backend Shipped in 4 hours… Hello, Druid!

Thank you for downloading this Voltron Data resource. Carahsoft is the distributor for Voltron Data's AI and ML solutions available via NASA SEWP V, ITES-SW2, NASPO ValuePoint, and many more contract vehicles.

To learn how to take the next step toward acquiring Voltron Data's solutions, please check out the following resources and information:

For Voltron Data overview:
carah.io/Voltron-Data

For upcoming events:
carah.io/Voltron-Events

For additional resources:
carah.io/Voltron-Resources

For additional Artificial Intelligence:
carah.io/ai-solutions

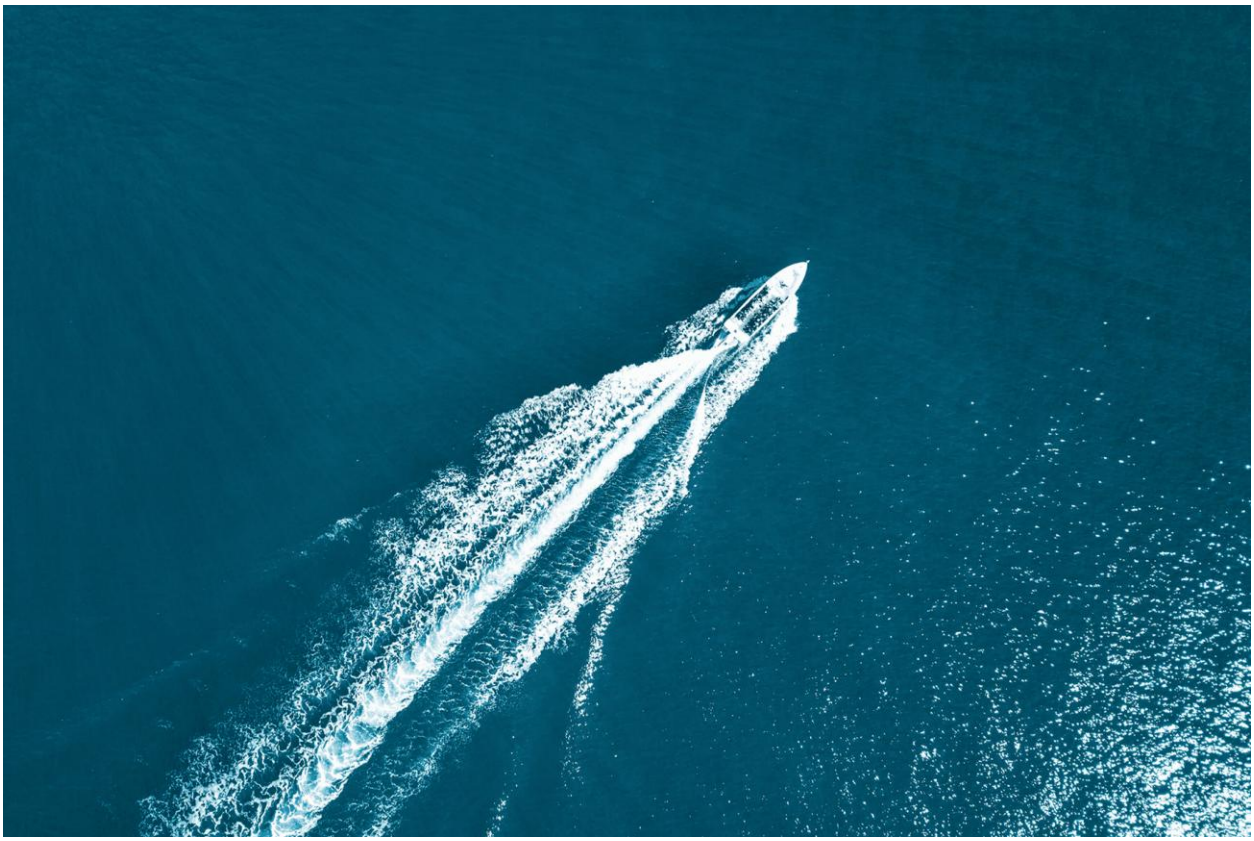
To set up a meeting:
VoltronData@carahsoft.com

To purchase, check out the contract vehicles available for procurement:
carah.io/procurement

Apr 11, 2023

New Ibis Backend Shipped in 4 hours... Hello, Druid!

Phillip Cloud



Recently, the Ibis team received a Github request for a Druid backend. The team was going to just add it quietly, but by the time it was done, I enjoyed the process enough to share it.

As the headline states, it only took me four hours, and that's because of both our process and the offerings of Druid. This also provides an opportunity to show off what happens behind the scenes with Ibis. If this all seems easy enough, you could try making your own Ibis backend. Let's jump right in.

Step 1: Find a Python API

In particular, if a backend has an existing SQLAlchemy dialect, the time-to-first-working-ibis-expression can be a few hours or less. In the case of Druid, there's pydruid which provides an SQLAlchemy dialect for Druid.

Step 2: Set Up Testing Infrastructure in docker-compose.yml

When adding a new backend, the first major lift is running the backend as a service so devs can test locally.

It's not always necessary to do this. For example, with the DuckDB backend, it doesn't make sense to set up a service since DuckDB follows an in-process deployment model.

It's also not possible in some cases—like for Snowflake and BigQuery—since both are proprietary SaaS offerings that execute code on your behalf in the cloud.

Druid follows a client-server model and is not a cloud-based SaaS, so I looked for an existing way to spin up all the services required to start executing queries against it. Luckily, the folks working on Druid have done 99% of the work. The remaining micro-step was to roll their setup into Ibis's docker-compose.yml.

Steps 3 to *n*: Iterate Until the Test Suite Stops Failing

This step is usually the most time-consuming and is the final step needed to merge a backend into the upstream Ibis repo.

Here, I put Druid through the wringer with the Ibis backend test suite, which automatically runs all known backends through a large set of functional tests. As you can imagine, not every backend passes every test so we have some tooling in the repo to help devs mark tests as a known failure.

As soon as all unmarked tests passed, I was done. The Druid backend was now in Ibis, and ready for the world.

Wrapping Up

When Ibis first started, adding a backend could be daunting, but with this one, I fully appreciate the payoff of what we've accomplished here. Druid brings us to **16 backends**, and we're ready for the next one. Remember, you can always leave requests on our issue tracker for a new backend!

If you want to get started with the Druid backend, the docs are up here: https://ibis-project.org/backends/Druid/. And, if you're interested in contributing, check out the "Contribution" docs on the Ibis page and be sure to review the Code of Conduct.

Photo by Louis Morgner