**Druid Quickstart**

In this quickstart, we will download Druid, set up it up on a single machine, load some data, and query the data.

**Prerequisites**

You will need:

* Java 7 or higher
* Linux, Mac OS X, or other Unix-like OS (Windows is not supported)
* 8G of RAM
* 2 vCPUs

On Mac OS X, you can use [Oracle's JDK 8](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html) to install Java.

On Linux, your OS package manager should be able to help for Java. If your Ubuntu- based OS does not have a recent enough version of Java, WebUpd8 offers [packages for those OSes](http://www.webupd8.org/2012/09/install-oracle-java-8-in-ubuntu-via-ppa.html).

**Getting started**

To install Druid, issue the following commands in your terminal:

curl -O http://static.druid.io/artifacts/releases/druid-0.9.2-bin.tar.gz

tar -xzf druid-0.9.2-bin.tar.gz

cd druid-0.9.2

In the package, you should find:

* LICENSE - the license files.
* bin/ - scripts useful for this quickstart.
* conf/\* - template configurations for a clustered setup.
* conf-quickstart/\* - configurations for this quickstart.
* extensions/\* - all Druid extensions.
* hadoop-dependencies/\* - Druid Hadoop dependencies.
* lib/\* - all included software packages for core Druid.
* quickstart/\* - files useful for this quickstart.

**Start up Zookeeper**

Druid currently has a dependency on [Apache ZooKeeper](http://zookeeper.apache.org/) for distributed coordination. You'll need to download and run Zookeeper.

curl http://www.gtlib.gatech.edu/pub/apache/zookeeper/zookeeper-3.4.6/zookeeper-3.4.6.tar.gz -o zookeeper-3.4.6.tar.gz

tar -xzf zookeeper-3.4.6.tar.gz

cd zookeeper-3.4.6

cp conf/zoo\_sample.cfg conf/zoo.cfg

./bin/zkServer.sh start

**Start up Druid services**

With Zookeeper running, return to the druid-0.9.2 directory. In that directory, issue the command:

bin/init

This will setup up some directories for you. Next, you can start up the Druid processes in different terminal windows. This tutorial runs every Druid process on the same system. In a large distributed production cluster, many of these Druid processes can still be co-located together.

java `cat conf-quickstart/druid/historical/jvm.config | xargs` -cp "conf-quickstart/druid/\_common:conf-quickstart/druid/historical:lib/\*" io.druid.cli.Main server historical

java `cat conf-quickstart/druid/broker/jvm.config | xargs` -cp "conf-quickstart/druid/\_common:conf-quickstart/druid/broker:lib/\*" io.druid.cli.Main server broker

java `cat conf-quickstart/druid/coordinator/jvm.config | xargs` -cp "conf-quickstart/druid/\_common:conf-quickstart/druid/coordinator:lib/\*" io.druid.cli.Main server coordinator

java `cat conf-quickstart/druid/overlord/jvm.config | xargs` -cp "conf-quickstart/druid/\_common:conf-quickstart/druid/overlord:lib/\*" io.druid.cli.Main server overlord

java `cat conf-quickstart/druid/middleManager/jvm.config | xargs` -cp "conf-quickstart/druid/\_common:conf-quickstart/druid/middleManager:lib/\*" io.druid.cli.Main server middleManager

You should see a log message printed out for each service that starts up.

Later on, if you'd like to stop the services, CTRL-C to exit from the running java processes. If you want a clean start after stopping the services, delete the var directory and run the init script again.

Once every service has started, you are now ready to load data.

**Load batch data**

We've included a sample of Wikipedia edits from September 12, 2015 to get you started.

This section shows you how to load data in batches, but you can skip ahead to learn how to [load streams in real-time](http://druid.io/docs/latest/tutorials/quickstart.html#load-streaming-data). Druid's streaming ingestion can load data with virtually no delay between events occurring and being available for queries.

The [dimensions](https://en.wikipedia.org/wiki/Dimension_%28data_warehouse%29) (attributes you can filter and split on) in the Wikipedia dataset, other than time, are:

* channel
* cityName
* comment
* countryIsoCode
* countryName
* isAnonymous
* isMinor
* isNew
* isRobot
* isUnpatrolled
* metroCode
* namespace
* page
* regionIsoCode
* regionName
* user

The [measures](https://en.wikipedia.org/wiki/Measure_%28data_warehouse%29), or *metrics* as they are known in Druid (values you can aggregate) in the Wikipedia dataset are:

* count
* added
* deleted
* delta
* user\_unique

To load this data into Druid, you can submit an *ingestion task* pointing to the file. We've included a task that loads the wikiticker-2015-09-12-sampled.json file included in the archive. To submit this task, POST it to Druid in a new terminal window from the druid-0.9.2 directory:

curl -X 'POST' -H 'Content-Type:application/json' -d @quickstart/wikiticker-index.json localhost:8090/druid/indexer/v1/task

Which will print the ID of the task if the submission was successful:

{"task":"index\_hadoop\_wikipedia\_2013-10-09T21:30:32.802Z"}

To view the status of your ingestion task, go to your overlord console: <http://localhost:8090/console.html>. You can refresh the console periodically, and after the task is successful, you should see a "SUCCESS" status for the task.

After your ingestion task finishes, the data will be loaded by historical nodes and available for querying within a minute or two. You can monitor the progress of loading your data in the coordinator console, by checking whether there is a datasource "wikiticker" with a blue circle indicating "fully available": <http://localhost:8081/#/>.

Once the data is fully available, you can immediately query it— to see how, skip to the [Query data](http://druid.io/docs/latest/tutorials/quickstart.html#query-data) section below. Or, continue to the [Load your own data](http://druid.io/docs/latest/tutorials/quickstart.html#load-your-own-data) section if you'd like to load a different dataset.

**Load streaming data**

To load streaming data, we are going to push events into Druid over a simple HTTP API. To do this we will use [Tranquility], a high level data producer library for Druid.

To download Tranquility, issue the following commands in your terminal:

curl -O http://static.druid.io/tranquility/releases/tranquility-distribution-0.8.0.tgz

tar -xzf tranquility-distribution-0.8.0.tgz

cd tranquility-distribution-0.8.0

We've included a configuration file in conf-quickstart/tranquility/server.json as part of the Druid distribution for a *metrics* datasource. We're going to start the Tranquility server process, which can be used to push events directly to Druid.

bin/tranquility server -configFile <path\_to\_druid\_distro>/conf-quickstart/tranquility/server.json

This section shows you how to load data using Tranquility Server, but Druid also supports a wide variety of [other streaming ingestion options](http://druid.io/docs/latest/ingestion/stream-ingestion.html#stream-push), including from popular streaming systems like Kafka, Storm, Samza, and Spark Streaming.

The [dimensions](https://en.wikipedia.org/wiki/Dimension_%28data_warehouse%29) (attributes you can filter and split on) for this datasource are flexible. It's configured for *schemaless dimensions*, meaning it will accept any field in your JSON input as a dimension.

The metrics (also called [measures](https://en.wikipedia.org/wiki/Measure_%28data_warehouse%29); values you can aggregate) in this datasource are:

* count
* value\_sum (derived from value in the input)
* value\_min (derived from value in the input)
* value\_max (derived from value in the input)

We've included a script that can generate some random sample metrics to load into this datasource. To use it, simply run in your Druid distribution repository:

bin/generate-example-metrics | curl -XPOST -H'Content-Type: application/json' --data-binary @- http://localhost:8200/v1/post/metrics

Which will print something like:

{"result":{"received":25,"sent":25}}

This indicates that the HTTP server received 25 events from you, and sent 25 to Druid. Note that this may take a few seconds to finish the first time you run it, as Druid resources must be allocated to the ingestion task. Subsequent POSTs should complete quickly.

Once the data is sent to Druid, you can immediately [query it](http://druid.io/docs/latest/tutorials/quickstart.html#query-data).

**Query data**

**Direct Druid queries**

Druid supports a rich [family of JSON-based queries](http://druid.io/docs/latest/querying/querying.html). We've included an example topN query in quickstart/wikiticker-top-pages.json that will find the most-edited articles in this dataset:

curl -L -H'Content-Type: application/json' -XPOST --data-binary @quickstart/wikiticker-top-pages.json http://localhost:8082/druid/v2/?pretty

**Visualizing data**

Druid is ideal for power user-facing analytic applications. There are a number of different open source applications to visualize and explore data in Druid. We recommend trying [Pivot](https://github.com/implydata/pivot), [Caravel](https://github.com/airbnb/caravel), or [Metabase](https://github.com/metabase/metabase) to start visualizing the data you just ingested.

If you installed Pivot for example, you should be able to view your data in your browser at [localhost:9090](http://localhost:9090/).

**SQL and other query libraries**

There are many more query tools for Druid than we've included here, including SQL engines, and libraries for various languages like Python and Ruby. Please see [the list of libraries](http://druid.io/docs/latest/development/libraries.html) for more information.

**Clustered setup**

This quickstart sets you up with all services running on a single machine. The next step is to [load your own data](http://druid.io/docs/latest/tutorials/ingestion.html). Or, you can skip ahead to [running a distributed cluster](http://druid.io/docs/latest/tutorials/cluster.html).

窗体顶端

|  |  |
| --- | --- |
| |  | | --- | |  | |

窗体底端

**Getting Started**

* [Concepts](http://druid.io/docs/latest/design/)
* [Quickstart](http://druid.io/docs/latest/tutorials/quickstart.html)
* [Loading Data](http://druid.io/docs/latest/tutorials/ingestion.html)
  + [Loading from Files](http://druid.io/docs/latest/tutorials/tutorial-batch.html)
  + [Loading from Streams](http://druid.io/docs/latest/tutorials/tutorial-streams.html)
  + [Loading from Kafka](http://druid.io/docs/latest/tutorials/tutorial-kafka.html)
* [Clustering](http://druid.io/docs/latest/tutorials/cluster.html)

**Data Ingestion**

* [Data Formats](http://druid.io/docs/latest/ingestion/data-formats.html)
* [Data Schema](http://druid.io/docs/latest/ingestion/index.html)
* [Schema Design](http://druid.io/docs/latest/ingestion/schema-design.html)
* [Schema Changes](http://druid.io/docs/latest/ingestion/schema-changes.html)
* [Batch File Ingestion](http://druid.io/docs/latest/ingestion/batch-ingestion.html)
* [Stream Ingestion](http://druid.io/docs/latest/ingestion/stream-ingestion.html)
  + [Stream Push](http://druid.io/docs/latest/ingestion/stream-push.html)
  + [Stream Pull](http://druid.io/docs/latest/ingestion/stream-pull.html)
* [Updating Existing Data](http://druid.io/docs/latest/ingestion/update-existing-data.html)
* [FAQ](http://druid.io/docs/latest/ingestion/faq.html)

**Querying**

* [Overview](http://druid.io/docs/latest/querying/querying.html)
* [Timeseries](http://druid.io/docs/latest/querying/timeseriesquery.html)
* [TopN](http://druid.io/docs/latest/querying/topnquery.html)
* [GroupBy](http://druid.io/docs/latest/querying/groupbyquery.html)
* [Time Boundary](http://druid.io/docs/latest/querying/timeboundaryquery.html)
* [Segment Metadata](http://druid.io/docs/latest/querying/segmentmetadataquery.html)
* [DataSource Metadata](http://druid.io/docs/latest/querying/datasourcemetadataquery.html)
* [Search](http://druid.io/docs/latest/querying/searchquery.html)
* [Select](http://druid.io/docs/latest/querying/select-query.html)
* Components
  + [Datasources](http://druid.io/docs/latest/querying/datasource.html)
  + [Filters](http://druid.io/docs/latest/querying/filters.html)
  + [Aggregations](http://druid.io/docs/latest/querying/aggregations.html)
  + [Post Aggregations](http://druid.io/docs/latest/querying/post-aggregations.html)
  + [Granularities](http://druid.io/docs/latest/querying/granularities.html)
  + [DimensionSpecs](http://druid.io/docs/latest/querying/dimensionspecs.html)
  + [Context](http://druid.io/docs/latest/querying/query-context.html)
* [Multi-value dimensions](http://druid.io/docs/latest/querying/multi-value-dimensions.html)
* [SQL](http://druid.io/docs/latest/querying/sql.html)
* [Lookups](http://druid.io/docs/latest/querying/lookups.html)
* [Joins](http://druid.io/docs/latest/querying/joins.html)
* [Multitenancy](http://druid.io/docs/latest/querying/multitenancy.html)
* [Caching](http://druid.io/docs/latest/querying/caching.html)
* [Sorting Orders](http://druid.io/docs/latest/querying/sorting-orders.html)

**Design**

* [Overview](http://druid.io/docs/latest/design/design.html)
* Storage
  + [Segments](http://druid.io/docs/latest/design/segments.html)
* Node Types
  + [Historical](http://druid.io/docs/latest/design/historical.html)
  + [Broker](http://druid.io/docs/latest/design/broker.html)
  + [Coordinator](http://druid.io/docs/latest/design/coordinator.html)
  + [Indexing Service](http://druid.io/docs/latest/design/indexing-service.html)
  + [Realtime](http://druid.io/docs/latest/design/realtime.html)
* Dependencies
  + [Deep Storage](http://druid.io/docs/latest/dependencies/deep-storage.html)
  + [Metadata Storage](http://druid.io/docs/latest/dependencies/metadata-storage.html)
  + [ZooKeeper](http://druid.io/docs/latest/dependencies/zookeeper.html)

**Operations**

* [Good Practices](http://druid.io/docs/latest/operations/recommendations.html)
* [Including Extensions](http://druid.io/docs/latest/operations/including-extensions.html)
* [Data Retention](http://druid.io/docs/latest/operations/rule-configuration.html)
* [Metrics and Monitoring](http://druid.io/docs/latest/operations/metrics.html)
* [Alerts](http://druid.io/docs/latest/operations/alerts.html)
* [Updating the Cluster](http://druid.io/docs/latest/operations/rolling-updates.html)
* [Different Hadoop Versions](http://druid.io/docs/latest/operations/other-hadoop.html)
* [Performance FAQ](http://druid.io/docs/latest/operations/performance-faq.html)

**Configuration**

* [Common Configuration](http://druid.io/docs/latest/configuration/index.html)
* [Indexing Service](http://druid.io/docs/latest/configuration/indexing-service.html)
* [Coordinator](http://druid.io/docs/latest/configuration/coordinator.html)
* [Historical](http://druid.io/docs/latest/configuration/historical.html)
* [Broker](http://druid.io/docs/latest/configuration/broker.html)
* [Realtime](http://druid.io/docs/latest/configuration/realtime.html)
* [Configuring Logging](http://druid.io/docs/latest/configuration/logging.html)

**Development**

* [Overview](http://druid.io/docs/latest/development/overview.html)
* [Libraries](http://druid.io/docs/latest/development/libraries.html)
* [Extensions](http://druid.io/docs/latest/development/extensions.html)
* [JavaScript](http://druid.io/docs/latest/development/javascript.html)
* [Build From Source](http://druid.io/docs/latest/development/build.html)
* [Versioning](http://druid.io/docs/latest/development/versioning.html)
* [Integration](http://druid.io/docs/latest/development/integrating-druid-with-other-technologies.html)
* Experimental Features
  + [Overview](http://druid.io/docs/latest/development/experimental.html)
  + [Approximate Histograms and Quantiles](http://druid.io/docs/latest/development/extensions-core/approximate-histograms.html)
  + [Datasketches](http://druid.io/docs/latest/development/extensions-core/datasketches-aggregators.html)
  + [Geographic Queries](http://druid.io/docs/latest/development/geo.html)
  + [Router](http://druid.io/docs/latest/development/router.html)
  + [Kafka Indexing Service](http://druid.io/docs/latest/development/extensions-core/kafka-ingestion.html)

**Misc**

* [Papers & Talks](http://druid.io/docs/latest/misc/papers-and-talks.html)
* [Thanks](http://druid.io/thanks.html)