rittmanmead (http://www.rittmanmead.com/)

An Introduction to Apache Drill

11 AUGUST 2016 on hdfs (/blog/tag/hdfs/), Oracle (/blog/tag/oracle/), Big Data (/blog/tag/big-data/), apache drill (/blog/tag/apache-drill/), drill (/blog/tag/drill/), sql (/blog/tag/sql/), Technical (/blog/tag/technical/)

Apache Drill (https://drill.apache.org/) is an engine that can connect to many different data sources, and provide a SQL interface to them. It's not just a wanna-be SQL interface that trips over at anything complex - it's a hugely functional one (https://drill.apache.org/docs/sql-reference-introduction/) including support for many built in functions as well as windowing functions (https://drill.apache.org/docs/sql-window-functions-introduction/). Whilst it can connect to standard data sources that you'd be able to query with SQL anyway, like Oracle or MySQL, it can also work with flat files such as CSV or JSON, as well as Avro and Parquet formats. It's this capability to run SQL against files that first piqued my interest in Apache Drill. I've been spending a lot of time looking at Big Data architectures and tools, including Big Data Discovery (https://speakerdeck.com/rmoff/unlock-the-value-in-your-big-data-reservoir-using-oracle-big-data-discovery-and-oracle-big-data-spatial-and-graph). As part of this, and experimenting with data pipeline options (http://www.rittmanmead.com/2016/07/stream-analytics-processing-kafka-oracle-stream-analytics/) one of the gaps that I've found is the functionality to dig through files in their raw state, before they've been brought into something like Hive which would enable their exploration through BDD and other tools.

In this article I'll walk through getting started with Apache Drill, and show some of the types of queries that I think are a great example of how useful it can be.

Getting Started

It's very simple to get going with Apache Drill - just download (https://drill.apache.org/download/) and unpack it, and run. Whilst it can run distributed across machines for performance, it can also run standalone on a laptop.

To launch it

```
cd /opt/apache-drill-1.7.0/
bin/sqlline -u jdbc:drill:zk=local
```

If you get `No current connection` or `com.fasterxml.jackson.databind.JavaType.isReferenceType()Z` then you have a conflicting JAR problem (http://rmoff.net/2016/06/20/apache-drill-not-connected/) (e.g. I encountered this on Oracle's BigDataLite VM), and should launch it with a clean environment

```
env -i HOME="$HOME" LC_CTYPE="${LC_ALL:-${LC_CTYPE:-$LANG}}" PATH="$PATH" USER="$USER" /opt/apache-drill-1.7.0/bin/drill-embedded
```

There's a built in dataset that you can use for testing:

```
USE cp;
SELECT employee_id, first_name FROM `employee.json` limit 5;
```

This should return five rows, in a very familiar environment if you're used to using SQL*Plus and similar tools:

```
0: jdbc:drill:zk=local> USE cp;
+-------+---------------------------------+
|  ok   |             summary             |
+-------+---------------------------------+
| true  | Default schema changed to [cp]  |
+-------+---------------------------------+
1 row selected (1.776 seconds)
0: jdbc:drill:zk=local>     SELECT employee_id, first_name FROM `employee.json` limit 5;
+--------------+-------------+
| employee_id  | first_name  |
+--------------+-------------+
| 1            | Sheri       |
| 2            | Derrick     |
| 4            | Michael     |
| 5            | Maya        |
| 6            | Roberta     |
+--------------+-------------+
5 rows selected (3.624 seconds)
```

So far, so SQL, so relational - so familiar, really. Where Apache Drill starts to deviate from the obvious is its use of storage handlers. In the above query `cp` is the 'database' that we're running our query against, but this is in fact a "classpath" (hence "cp") storage handler that's defined by default. Within a 'database' there are 'schemas' which are sub-configurations of the storage handler. We'll have a look at viewing and defining these later on. For now, it's useful to know that you can also list out the available databases:

```
0: jdbc:drill:zk=local> show databases;
+---------------------+
|     SCHEMA_NAME     |
+---------------------+
| INFORMATION_SCHEMA  |
| cp.default          |
| dfs.default         |
| dfs.root            |
| dfs.tmp             |
| sys                 |
+---------------------+
```

Note `databases` command is a synonym for `schemas`; it's the `<database>.<schema>` that's returned for both. In Apache Drill the backtick is used to enclose identifiers (such as schema names, column names, and so on), and it's quite particular about it. For example, this is valid:

```
0: jdbc:drill:zk=local> USE `cp.default`;
+-------+-----------------------------------------+
|  ok   |                 summary                 |
+-------+-----------------------------------------+
| true  | Default schema changed to [cp.default]  |
+-------+-----------------------------------------+
1 row selected (0.171 seconds)
```

whilst this isn't:

```
0: jdbc:drill:zk=local> USE cp.default;
Error: PARSE ERROR: Encountered ". default" at line 1, column 7.
Was expecting one of:
<EOF>
"." <IDENTIFIER> ...
"." <QUOTED_IDENTIFIER> ...
"." <BACK_QUOTED_IDENTIFIER> ...
"." <BRACKET_QUOTED_IDENTIFIER> ...
"." <UNICODE_QUOTED_IDENTIFIER> ...
"." "*" ...

SQL Query USE cp.default
```

This is because `default` is a reserved word, and hence must be quoted. Hence, you can also use

```
0: jdbc:drill:zk=local> use cp.`default`;
```

but not

```
0: jdbc:drill:zk=local> use `cp`.default;
```

Querying JSON data

On the Apache Drill website there's some useful tutorials (https://drill.apache.org/docs/tutorials/), including one using data provided by Yelp (http://www.yelp.com/dataset_challenge). This was the dataset that originally got me looking at Drill, since I was using it as an input to Big Data Discovery (https://speakerdeck.com/rmoff/unlock-the-value-in-your-big-data-reservoir-using-oracle-big-data-discovery-and-oracle-big-data-spatial-and-graph) (BDD) but struggling on two counts. First up was how best to define a suitable Hive table over it in order to ingest it to BDD. Following from this was trying to understand what value there might be in the data which would drive how long to spend perfecting the way in which I exposed the data in Hive. The examples below show the kind of complications that complex JSON can introduce when queried in a tabular fashion.

First up, querying a JSON file, with the schema inferred automagically. Pretty cool.

```
0: jdbc:drill:zk=local> select * from `/user/oracle/incoming/yelp/tip_json/yelp_academic_dataset_tip.json` limit 5;
+---------+------+-------------+-------+------+------+
| user_id | text | business_id | likes | date | type |
+---------+------+-------------+-------+------+------+
| -6rEfobYjMxpUWLNxszaxQ | Don't waste your time. | cE27W9VPgO88Qxe4ol6y_g | 0 | 2013-04-18 | tip |
| EZ0r9dKKtEGVx2CdnowPCw | Your GPS will not allow you to find this place. Put Rankin police department in instead. They are directly a
cross the street. | mVHrayjG3uZ_RLHkLj-AMg | 1 | 2013-01-06 | tip |
| xb6zEQCw9I-Gl0g06e1KsQ | Great drink specials! | KayYbHCt-RkbGcPdGOThNg | 0 | 2013-12-03 | tip |
| QawZN4PSW7ng_9SP7pjsVQ | Friendly staff, good food, great beer selection, and relaxing atmosphere | KayYbHCt-RkbGcPdGOThNg | 0 | 2015
-07-08 | tip |
| MLQre1nvUtW-RqMTc4iC9A | Beautiful restoration. | 1_lU0-eSWJCRvNGk78Zh9Q | 0 | 2015-10-25 | tip |
+---------+------+-------------+-------+------+------+
5 rows selected (2.341 seconds)
```

We can use standard SQL aggregations such as `COUNT`:

```
0: jdbc:drill:zk=local> select count(*) from `/user/oracle/incoming/yelp/tip_json/yelp_academic_dataset_tip.json`;
+---------+
| EXPR$0  |
+---------+
| 591864  |
+---------+
1 row selected (4.495 seconds)
```

as well as `GROUP BY` operation:

```
0: jdbc:drill:zk=local> select `date`,count(*) as tip_count from `/user/oracle/incoming/yelp/tip_json/yelp_academic_dataset_tip.json` g
roup by `date` order by 2 desc limit 5;
+-------------+------------+
|    date     | tip_count  |
+-------------+------------+
| 2012-07-21  | 719        |
| 2012-05-19  | 718        |
| 2012-08-04  | 699        |
| 2012-06-23  | 690        |
| 2012-07-28  | 682        |
+-------------+------------+
5 rows selected (7.111 seconds)
```

Digging into the data a bit, we can see that it's not entirely flat - note, for example, the `hours` column, which is a nested JSON object:

```
0: jdbc:drill:zk=local> select full_address,city,hours from `/user/oracle/incoming/yelp/business_json` b limit 5;
+--------------+------+-------+
| full_address | city | hours |
+--------------+------+-------+
| 4734 Lebanon Church Rd
Dravosburg, PA 15034 | Dravosburg | {"Friday":{"close":"21:00","open":"11:00"},"Tuesday":{"close":"21:00","open":"11:00"},"Thursday":
{"close":"21:00","open":"11:00"},"Wednesday":{"close":"21:00","open":"11:00"},"Monday":{"close":"21:00","open":"11:00"},"Sunday":{},"Sa
turday":{}} |
| 202 McClure St
Dravosburg, PA 15034 | Dravosburg | {"Friday":{},"Tuesday":{},"Thursday":{},"Wednesday":{},"Monday":{},"Sunday":{},"Saturday":{}} |
| 1 Ravine St
Dravosburg, PA 15034 | Dravosburg | {"Friday":{},"Tuesday":{},"Thursday":{},"Wednesday":{},"Monday":{},"Sunday":{},"Saturday":{}} |
| 1530 Hamilton Rd
Bethel Park, PA 15234 | Bethel Park | {"Friday":{},"Tuesday":{},"Thursday":{},"Wednesday":{},"Monday":{},"Sunday":{},"Saturday":{}} |
| 301 South Hills Village
Pittsburgh, PA 15241 | Pittsburgh | {"Friday":{"close":"17:00","open":"10:00"},"Tuesday":{"close":"21:00","open":"10:00"},"Thursday":
{"close":"17:00","open":"10:00"},"Wednesday":{"close":"21:00","open":"10:00"},"Monday":{"close":"21:00","open":"10:00"},"Sunday":{"clos
e":"18:00","open":"11:00"},"Saturday":{"close":"21:00","open":"10:00"}} |
+--------------+------+-------+
5 rows selected (0.721 seconds)
0: jdbc:drill:zk=local>
```

With Apache Drill we can simply use dot notation to access nested values. It's necessary to alias the table (`b` in this example) when you're doing this:

```
0: jdbc:drill:zk=local> select b.hours from `/user/oracle/incoming/yelp/business_json` b limit 1;
+-------+
| hours |
+-------+
| {"Friday":{"close":"21:00","open":"11:00"},"Tuesday":{"close":"21:00","open":"11:00"},"Thursday":{"close":"21:00","open":"11:00"},"We
dnesday":{"close":"21:00","open":"11:00"},"Monday":{"close":"21:00","open":"11:00"},"Sunday":{},"Saturday":{}} |
+-------+
```

Nested objects can themselves be nested - not a problem with Apache Drill, we just chain the dot notation further:

```
0: jdbc:drill:zk=local> select b.hours.Friday from `/user/oracle/incoming/yelp/business_json` b limit 1;
+------------------------------------+
|              EXPR$0                |
+------------------------------------+
| {"close":"21:00","open":"11:00"}  |
+------------------------------------+
1 row selected (0.238 seconds)
```

Note the use of backtick (`` ` ``) to quote the reserved `open` and `close` keywords:

```
0: jdbc:drill:zk=local> select b.hours.Friday.`open`,b.hours.Friday.`close` from `/user/oracle/incoming/yelp/business_json` b limit 1;
+---------+---------+
| EXPR$0  | EXPR$1  |
+---------+---------+
| 11:00   | 21:00   |
+---------+---------+
1 row selected (0.58 seconds)
```

Nested columns are proper objects in their own right in the query, and can be used as predicates too:

```
0: jdbc:drill:zk=local> select b.name,b.full_address,b.hours.Friday.`open` from `/user/oracle/incoming/yelp/business_json` b where b.ho
urs.Friday.`open` = '11:00' limit 5;
+------------------------+------------------------------------------------+---------+
|          name          |                  full_address                  | EXPR$2  |
+------------------------+------------------------------------------------+---------+
| Mr Hoagie              | 4734 Lebanon Church Rd
Dravosburg, PA 15034    | 11:00   |
| Alexion's Bar & Grill  | 141 Hawthorne St
Greentree
Carnegie, PA 15106  | 11:00   |
| Rocky's Lounge         | 1201 Washington Ave
Carnegie, PA 15106         | 11:00   |
| Papa J's               | 200 E Main St
Carnegie
Carnegie, PA 15106      | 11:00   |
| Italian Village Pizza  | 2615 Main St
Homestead, PA 15120              | 11:00   |
+------------------------+------------------------------------------------+---------+
5 rows selected (0.404 seconds)
```

You'll notice in the above output that the `full_address` field has line breaks in -- we can just use a SQL Function (https://drill.apache.org/docs/about-sql-function-examples/) to replace line breaks with commas:

```
0: jdbc:drill:zk=local> select b.name,regexp_replace(b.full_address,'\n',','),b.hours.Friday.`open` from `/user/oracle/incoming/yelp/bu
siness_json` b where b.hours.Friday.`open` = '11:00' limit 5;
+------------------------+------------------------------------------------+---------+
|          name          |                     EXPR$1                     | EXPR$2  |
+------------------------+------------------------------------------------+---------+
| Mr Hoagie              | 4734 Lebanon Church Rd,Dravosburg, PA 15034    | 11:00   |
| Alexion's Bar & Grill  | 141 Hawthorne St,Greentree,Carnegie, PA 15106  | 11:00   |
| Rocky's Lounge         | 1201 Washington Ave,Carnegie, PA 15106         | 11:00   |
| Papa J's               | 200 E Main St,Carnegie,Carnegie, PA 15106      | 11:00   |
| Italian Village Pizza  | 2615 Main St,Homestead, PA 15120               | 11:00   |
+------------------------+------------------------------------------------+---------+
5 rows selected (1.346 seconds)
```

Query Federation

So Apache Drill enables you to run SQL queries against data in a multitude of formats and locations, which is rather useful in itself. But even better than that, it lets you federate these sources in a single query. Here's an example of joining between data in HDFS and Oracle:

```
0: jdbc:drill:zk=local> select X.text,
. . . . . . . . . . . > Y.NAME
. . . . . . . . . . . > from hdfs.`/user/oracle/incoming/yelp/tip_json/yelp_academic_dataset_tip.json` X
. . . . . . . . . . . > inner join ora.MOVIEDEMO.YELP_BUSINESS Y
. . . . . . . . . . . > on X.business_id = Y.BUSINESS_ID
. . . . . . . . . . . > where Y.NAME = 'Chick-fil-A'
. . . . . . . . . . . > limit 5;
+--------------------------------------------------------------------+--------------+
|                                text                                |     NAME     |
+--------------------------------------------------------------------+--------------+
| It's daddy daughter date night here and they go ALL OUT!           | Chick-fil-A  |
| Chicken minis!  The best part of waking up Saturday mornings.  :)  | Chick-fil-A  |
| Nice folks as always unlike those ghetto joints                    | Chick-fil-A  |
| Great clean and delicious chicken sandwiches!                      | Chick-fil-A  |
| Spicy Chicken with lettuce, tomato, and pepperjack cheese FTW!     | Chick-fil-A  |
+--------------------------------------------------------------------+--------------+
5 rows selected (3.234 seconds)
```

You can define a view over this:

```
0: jdbc:drill:zk=local> create or replace view dfs.tmp.yelp_tips as select X.text as tip_text, Y.NAME as business_name from hdfs.`/use
r/oracle/incoming/yelp/tip_json/yelp_academic_dataset_tip.json` X inner join ora.MOVIEDEMO.YELP_BUSINESS Y on X.business_id = Y.BUSINES
S_ID ;
+-------+------------------------------------------------------------+
|  ok   |                          summary                           |
+-------+------------------------------------------------------------+
| true  | View 'yelp_tips' replaced successfully in 'dfs.tmp' schema  |
+-------+------------------------------------------------------------+
1 row selected (0.574 seconds)
0: jdbc:drill:zk=local> describe dfs.tmp.yelp_tips;
+----------------+--------------------+--------------+
|  COLUMN_NAME   |     DATA_TYPE      | IS_NULLABLE  |
+----------------+--------------------+--------------+
| tip_text       | ANY                | YES          |
| business_name  | CHARACTER VARYING  | YES          |
+----------------+--------------------+--------------+
2 rows selected (0.756 seconds)
```

and then query it as any regular object:

```
0: jdbc:drill:zk=local> select tip_text,business_name from dfs.tmp.yelp_tips where business_name like '%Grill' limit 5;
+------+------+
| text | NAME |
+------+------+
| Great drink specials! | Alexion's Bar & Grill |
| Friendly staff, good food, great beer selection, and relaxing atmosphere | Alexion's Bar & Grill |
| Pretty quiet here... | Uno Pizzeria & Grill |
| I recommend this location for quick lunches. 10 min or less lunch menu. Soup bar ( all you can eat)  the broccoli cheddar soup is del
icious. | Uno Pizzeria & Grill |
| Instead of pizza, come here for dessert. The deep dish sundae is really good. | Uno Pizzeria & Grill |
+------+------+
5 rows selected (3.272 seconds)
```

Querying Twitter JSON data

Here's an example of using Drill to query a local file holding some Twitter data. You can download the file here (https://gist.github.com/b81b40df61d3faa1f4bf05193cfeec6d) if you want to try querying it yourself.

To start with I switched to using the `dfs` storage plugin:

```
0: jdbc:drill:zk=local> use dfs;
+-------+----------------------------------+
|  ok   |             summary              |
+-------+----------------------------------+
| true  | Default schema changed to [dfs]  |
+-------+----------------------------------+
```

And then tried a select against the file. Note the `limit 5` clause - very useful when you're just examining the structure of a file.

```
0: jdbc:drill:zk=local> select * from `/user/oracle/incoming/twitter/geo_tweets.json` limit 5;
Error: DATA_READ ERROR: Error parsing JSON - Unexpected end-of-input within/between OBJECT entries

File  /user/oracle/incoming/twitter/geo_tweets.json
Record  2819
Column  3503
Fragment 0:0
```

An error? That's not supposed to happen. I've got a JSON file, right? It turns out the JSON file is one complete JSON object per line. Except that it's not on the last record. Note the record count given in the error above - 2819:

```
[oracle@bigdatalite ~]$ wc -l geo_tweets.json
2818 geo_tweets.json
```

So the file only has 2818 complete lines. Hmmm. Let's take a look at that record, using a head/tail bash combo :

```
[oracle@bigdatalite ~]$ head -n 2819 geo_tweets.json |tail -n1
{"created_at":"Sun Jul 24 21:00:44 +0000 2016","id":757319630432067584,"id_str":"757319630432067584","text":"And now @HillaryClinton hi
res @DWStweets: Honorary Campaign Manager across the USA #corruption #hillarysamerica  https://t.co/8jAGUu6w2f","source":"<a href=\"htt
p://www.handmark.com\" rel=\"nofollow\">TweetCaster for iOS</a>","truncated":false,"in_reply_to_status_id":null,"in_reply_to_status_id_
str":null,"in_reply_to_user_id":null,"in_reply_to_user_id_str":null,"in_reply_to_screen_name":null,"user":{"id":2170786369,"id_str":"21
70786369","name":"Patricia Weber","screen_name":"InnieBabyBoomer","location":"Williamsburg, VA","url":"http://lovesrantsandraves.blogsp
ot.com/","description":"Baby Boomer, Swing Voter, Conservative, Spiritual, #Introvert, Wife, Grandma, Italian, ♥ Books, Cars, Ferrari,
 F1 Race♥  #tcot","protected":false,"verified":false,"followers_count":861,"friends_count":918,"listed_count":22,"favourites_count":1
7,"statuses_count":2363,"created_at":"Sat Nov 02 19:13:06 +0000 2013","utc_offset":null,"time_zone":null,"geo_enabled":true,"lang":"e
n","contributors_enabled":false,"is_translator":false,"profile_background_color":"C0DEED","profile_background_image_url":"http://pbs.tw
img.com/profile_background_images/378800000107659131/3589f
```

That's the complete data in the file - so Drill is right - the JSON is corrupted. If we drop that last record and create a new file (`geo_tweets.fixed.json` (https://gist.github.com/9edca5e444143a00a8207ed41a364b39))

```
head -n2818 geo_tweets.json > geo_tweets.fixed.json
```

and query it again, we get something!

```
0: jdbc:drill:zk=local>  select text from `/users/rmoff/data/geo_tweets.fixed.json` limit 5;
+------+
| text |
+------+
| Vancouver trends now: Trump, Evander Kane, Munich, 2016HCC and dcc16. https://t.co/joI9GMfRim |
| We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade #Tucson, AZ #Job
s |
| Donald Trump accepted the Republican nomination last night. Isis claimed responsibility. |
| Obama: "We must stand together and stop terrorism"
Trump: "We don't want these people in our country"
� |
| Someone built a wall around Trump's star on the Hollywood Walk of Fame. #lol #nowthatsfunny @… https://t.co/qHWuJXnzbw |
+------+
5 rows selected (0.246 seconds)
```

`text` here being one of the json fields. I could do a `select *` but it's not so intelligable:

```
0: jdbc:drill:zk=local>  select * from `/users/rmoff/data/geo_tweets.fixed.json` limit 5;
+------------+----+--------+------+--------+-----------+------+-----+-------------+-------+-----------------+---------------+----------
------+----------+-----------+-----------+--------------------+--------------+------+--------------+----------+------------+-----------
+------------------+----------------------+--------------------+-------------------+-----------------------+---------------------+-----
------------+------------+---------------+---------------+------------+-----------+--------------------------------+-----------+-------
---+----------------+-------------------+---------------------------------+-----------------------+---------------------------+--------
-------------+-------------------------+-------------------------+------------------+-----------------------+------------------+-------
---------------+---------------+
| created_at | id | id_str | text | source | truncated | user | geo | coordinates | place | is_quote_status | retweet_count | favorite_
count | entities | favorited | retweeted | possibly_sensitive | filter_level | lang | timestamp_ms | @version | @timestamp | user_name
 | user_screen_name | user_followers_count | user_friends_count | user_listed_count | user_favourites_count | user_statuses_count | use
r_created_at | place_name | place_country | hashtags_list | urls_array | urls_list | user_mentions_screen_name_list | longitude | latit
ude | hashtags_array | extended_entities | user_mentions_screen_name_array | in_reply_to_status_id | in_reply_to_status_id_str | in_rep
ly_to_user_id | in_reply_to_user_id_str | in_reply_to_screen_name | retweeted_status | retweeted_screen_name | quoted_status_id | quote
d_status_id_str | quoted_status |
+------------+----+--------+------+--------+-----------+------+-----+-------------+-------+-----------------+---------------+----------
------+----------+-----------+-----------+--------------------+--------------+------+--------------+----------+------------+-----------
+------------------+----------------------+--------------------+-------------------+-----------------------+---------------------+-----
------------+------------+---------------+---------------+------------+-----------+--------------------------------+-----------+-------
---+----------------+-------------------+---------------------------------+-----------------------+---------------------------+--------
-------------+-------------------------+-------------------------+------------------+-----------------------+------------------+-------
---------------+---------------+
| Fri Jul 22 19:37:11 +0000 2016 | 756573827589545984 | 756573827589545984 | Vancouver trends now: Trump, Evander Kane, Munich, 2016HCC
and dcc16. https://t.co/joI9GMfRim | <a href="http://dlvr.it" rel="nofollow">dlvr.it</a> | false | {"id":67898674,"id_str":"6789867
4","name":"Vancouver Press","screen_name":"Vancouver_CP","location":"Vancouver, BC","url":"http://vancouver.cityandpress.com/","descrip
tion":"Latest news from Vancouver. Updates are frequent.","protected":false,"verified":false,"followers_count":807,"friends_count":1
3,"listed_count":94,"favourites_count":1,"statuses_count":131010,"created_at":"Sat Aug 22 14:25:37 +0000 2009","utc_offset":-25200,"tim
e_zone":"Pacific Time (US & Canada)","geo_enabled":true,"lang":"en","contributors_enabled":false,"is_translator":false,"profile_backgro
und_color":"FFFFFF","profile_background_image_url":"http://abs.twimg.com/images/themes/theme1/bg.png","profile_background_image_url_htt
ps":"https://abs.twimg.com/images/themes/theme1/bg.png","profile_background_tile":false,"profile_link_color":"8A1C3B","profile_sidebar_
border_color":"FFFFFF","profile_sidebar_fill_color":"FFFFFF","profile_text_color":"2A2C31","profile_use_background_image":false,"profil
e_image_url":"http://pbs.twimg.com/profile_images/515841109553983490/_t0QWPco_normal.png","profile_image_url_https":"https://pbs.twimg.
com/profile_images/515841109553983490/_t0QWPco_normal.png","profile_banner_url":"https://pbs.twimg.com/profile_banners/67898674/1411821
103","default_profile":false,"default_profile_image":false} | {"type":"Point","coordinates":[49.2814375,-123.12109067]} | {"type":"Poin
t","coordinates":[-123.12109067,49.2814375]} | {"id":"1e5cb4d0509db554","url":"https://api.twitter.com/1.1/geo/id/1e5cb4d0509db554.jso
n","place_type":"city","name":"Vancouver","full_name":"Vancouver, British Columbia","country_code":"CA","country":"Canada","bounding_bo
x":{"type":"Polygon","coordinates":[[[-123.224215,49.19854],[-123.224215,49.316738],[-123.022947,49.316738],[-123.022947,49.1985
4]]]},"attributes":{}} | false | 0 | 0 | {"urls":[{"url":"https://t.co/joI9GMfRim","expanded_url":"http://toplocalnow.com/ca/vancouver?
section=trends","display_url":"toplocalnow.com/ca/vancouver?s…","indices":[70,93]}],"hashtags":[],"user_mentions":[],"media":[],"symbol
s":[]} | false | false | false | low | en | 1469216231616 | 1 | 2016-07-22T19:37:11.000Z | Vancouver Press | Vancouver_CP | 807 | 13 |
 94 | 1 | 131010 | Sat Aug 22 14:25:37 +0000 2009 | Vancouver | Canada |  | ["toplocalnow.com/ca/vancouver?s…"] | toplocalnow.com/ca/va
ncouver?s… |  | -123.12109067 | 49.2814375 | [] | {"media":[]} | [] | null | null | null | null | null | {"user":{},"entities":{"user_m
entions":[],"media":[],"hashtags":[],"urls":[]},"extended_entities":{"media":[]},"quoted_status":{"user":{},"entities":{"hashtags":
[],"user_mentions":[],"media":[],"urls":[]},"extended_entities":{"media":[]}}} | null | null | null | {"user":{},"entities":{"user_ment
ions":[],"media":[],"urls":[],"hashtags":[]},"extended_entities":{"media":[]},"place":{"bounding_box":{"coordinates":[]},"attributes":
{}},"geo":{"coordinates":[]},"coordinates":{"coordinates":[]}} |
```

Within the twitter data there's root-level fields, such as `text`, as well as nested ones such as information about the tweeter in the `user` field. As we saw above you reference nested fields using dot notation. Now's a good time to point out a couple of common mistakes that you may encounter. The first is not quoting reserved words, and is the first thing to check for if you get an error such as `Encountered "."`:

```
0: jdbc:drill:zk=local>  select user.screen_name,text from `/users/rmoff/data/geo_tweets.fixed.json` limit 5;
Error: PARSE ERROR: Encountered "." at line 1, column 12.
[...]
```

Second is declaring the table alias when using dot notation - if you don't then Apache Drill thinks that the parent column is actually the table name (`VALIDATION ERROR: [...] Table 'user' not found`):

```
0: jdbc:drill:zk=local>  select `user`.screen_name,text from dfs.`/users/rmoff/data/geo_tweets.fixed.json` limit 5;
Aug 10, 2016 11:16:45 PM org.apache.calcite.sql.validate.SqlValidatorException <init>
SEVERE: org.apache.calcite.sql.validate.SqlValidatorException: Table 'user' not found
Aug 10, 2016 11:16:45 PM org.apache.calcite.runtime.CalciteException <init>
SEVERE: org.apache.calcite.runtime.CalciteContextException: From line 1, column 8 to line 1, column 13: Table 'user' not found
Error: VALIDATION ERROR: From line 1, column 8 to line 1, column 13: Table 'user' not found

SQL Query null

[Error Id: 1427fd23-e180-40be-a751-b6f1f838233a on 192.168.56.1:31010] (state=,code=0)
```

With those mistakes fixed, we can see the user's screenname:

```
0: jdbc:drill:zk=local>  select tweets.`user`.`screen_name` as user_screen_name,text from dfs.`/users/rmoff/data/geo_tweets.fixed.json`
tweets limit 2;
+------------------+------+
| user_screen_name | text |
+------------------+------+
| Vancouver_CP     | Vancouver trends now: Trump, Evander Kane, Munich, 2016HCC and dcc16. https://t.co/joI9GMfRim |
| tmj_TUC_skltrd   | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTra
de #Tucson, AZ #Jobs |
+------------------+------+
2 rows selected (0.256 seconds)
0: jdbc:drill:zk=local>
```

As well as nested objects, JSON supports arrays. An example of this in twitter data is hashtags, or URLs, both of which there can be zero, one, or many of in a given tweet.

```
0: jdbc:drill:zk=local>  select tweets.entities.hashtags from dfs.`/users/rmoff/data/geo_tweets.fixed.json` tweets limit 5;
+--------+
| EXPR$0 |
+--------+
| [] |
| [{"text":"hiring","indices":[6,13]},{"text":"Job","indices":[98,102]},{"text":"SkilledTrade","indices":[103,116]},{"text":"Tucson","i
ndices":[117,124]},{"text":"Jobs","indices":[129,134]}] |
| [] |
| [] |
| [{"text":"lol","indices":[72,76]},{"text":"nowthatsfunny","indices":[77,91]}] |
+--------+
5 rows selected (0.286 seconds)
```

Using the `FLATTEN` function each array entry becomes a new row, thus:

```
0: jdbc:drill:zk=local>  select flatten(tweets.entities.hashtags) from dfs.`/users/rmoff/data/geo_tweets.fixed.json` tweets limit 5;
+----------------------------------------------+
|                    EXPR$0                    |
+----------------------------------------------+
| {"text":"hiring","indices":[6,13]}           |
| {"text":"Job","indices":[98,102]}            |
| {"text":"SkilledTrade","indices":[103,116]}  |
| {"text":"Tucson","indices":[117,124]}        |
| {"text":"Jobs","indices":[129,134]}          |
+----------------------------------------------+
5 rows selected (0.139 seconds)
```

Note that the `limit 5` clause is showing only the first five array instances, which is actually just hashtags from the first tweet in the above list.

To access the text of the hashtag we use a subquery and the dot notation to access the `text` field:

```
0: jdbc:drill:zk=local> select ent_hashtags.hashtags.text from (select flatten(tweets.entities.hashtags) as hashtags from dfs.`/users/r
moff/data/geo_tweets.fixed.json` tweets) as ent_hashtags limit 5;
+---------------+
|    EXPR$0     |
+---------------+
| hiring        |
| Job           |
| SkilledTrade  |
| Tucson        |
| Jobs          |
+---------------+
5 rows selected (0.168 seconds)
```

This can be made more readable by using Common Table Expressions (https://drill.apache.org/docs/with-clause/) (CTE, also known as subquery factoring) for the same result:

```
0: jdbc:drill:zk=local> with ent_hashtags as (select flatten(tweets.entities.hashtags) as hashtags from dfs.`/users/rmoff/data/geo_twee
ts.fixed.json` tweets)
. . . . . . . . . . . > select ent_hashtags.hashtags.text from ent_hashtags
. . . . . . . . . . . > limit 5;
+---------------+
|    EXPR$0     |
+---------------+
| hiring        |
| Job           |
| SkilledTrade  |
| Tucson        |
| Jobs          |
+---------------+
5 rows selected (0.253 seconds)
```

Combining the flattened array with existing fields enables us to see things like a list of tweets with their associated hashtags:

```
0: jdbc:drill:zk=local> with tmp as ( select flatten(tweets.entities.hashtags) as hashtags,tweets.text,tweets.`user`.screen_name as use
r_screen_name from dfs.`/users/rmoff/data/geo_tweets.fixed.json` tweets) select tmp.user_screen_name,tmp.text,tmp.hashtags.text as hash
tag from tmp limit 10;
+------------------+------+---------+
| user_screen_name | text | hashtag |
+------------------+------+---------+
| tmj_TUC_skltrd | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade
#Tucson, AZ #Jobs | hiring |
| tmj_TUC_skltrd | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade
#Tucson, AZ #Jobs | Job |
| tmj_TUC_skltrd | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade
#Tucson, AZ #Jobs | SkilledTrade |
| tmj_TUC_skltrd | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade
#Tucson, AZ #Jobs | Tucson |
| tmj_TUC_skltrd | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade
#Tucson, AZ #Jobs | Jobs |
| johnmayberry | Someone built a wall around Trump's star on the Hollywood Walk of Fame. #lol #nowthatsfunny @… https://t.co/qHWuJXnzbw
| lol |
| johnmayberry | Someone built a wall around Trump's star on the Hollywood Walk of Fame. #lol #nowthatsfunny @… https://t.co/qHWuJXnzbw
| nowthatsfunny |
| greensboro_nc | #WinstonSalem Time and place announced for Donald Trump's visit to… https://t.co/6OVl7crshw #ws @winston_salem_ http
s://t.co/l5h220otj4 | WinstonSalem |
| greensboro_nc | #WinstonSalem Time and place announced for Donald Trump's visit to… https://t.co/6OVl7crshw #ws @winston_salem_ http
s://t.co/l5h220otj4 | ws |
| trendinaliaSG | 6. Hit The Stage
7. TTTT
8. Demi Lovato
9. Beijing
10. Donald Trump

2016/7/23 03:36 SGT #trndnl https://t.co/psP0GzBgZB | trndnl |
+------------------+------+---------+
10 rows selected (0.166 seconds)
```

We can also filter based on hashtag:

```
0: jdbc:drill:zk=local> with tmp as ( select flatten(tweets.entities.hashtags) as hashtags,tweets.text,tweets.`user`.screen_name as use
r_screen_name from dfs.`/users/rmoff/data/geo_tweets.fixed.json` tweets) select tmp.user_screen_name,tmp.text,tmp.hashtags.text as hash
tag from tmp where tmp.hashtags.text = 'Job' limit 5;
+------------------+------+---------+
| user_screen_name | text | hashtag |
+------------------+------+---------+
| tmj_TUC_skltrd | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade
#Tucson, AZ #Jobs | Job |
| tmj_VAL_health | Want to work at Genesis Rehab Services? We're #hiring in #Clinton, MD! Click for details: https://t.co/4lt7I4gMZk #J
ob #Healthcare #Jobs | Job |
| tmj_in_retail | Want to work in #Clinton, IN? View our latest opening: https://t.co/UiimnlubYs #Job #Retail #Jobs #Hiring #CareerArc
 | Job |
| tmj_la_hrta | Want to work at SONIC Drive-In? We're #hiring in #Clinton, LA! Click for details: https://t.co/aQ1FrWc7iR #Job #SONIC #
Hospitality #Jobs | Job |
| tmj_ia_hrta | We're #hiring! Click to apply: Department Manager - https://t.co/SnoKcwwHFk #Job #Hospitality #Clinton, IA #Jobs #Caree
rArc | Job |
+------------------+------+---------+
5 rows selected (0.207 seconds)
```

as well as summarise hashtag counts:

```
0: jdbc:drill:zk=local> with ent_hashtags as (select flatten(tweets.entities.hashtags) as hashtags from dfs.`/users/rmoff/data/geo_twee
ts.fixed.json` tweets)
. . . . . . . . . . . > select ent_hashtags.hashtags.text,count(ent_hashtags.hashtags.text) from ent_hashtags
. . . . . . . . . . . > group by ent_hashtags.hashtags.text
. . . . . . . . . . . > order by 2 desc;
+-----------------------------+---------+
|           EXPR$0            | EXPR$1  |
+-----------------------------+---------+
| Trump                       | 365     |
| trndnl                      | 176     |
| job                         | 170     |
| Hiring                      | 127     |
| Clinton                     | 108     |
| Yorkshire                   | 100     |
| CareerArc                   | 100     |
[...]
```

To filter out records that may not have array values (such as hashtags, which not every tweet has) and without with the query may fail, use `IS NOT NULL` against an attribute of first index of the array:

```
0: jdbc:drill:zk=local> select tweets.entities.hashtags from dfs.`/users/rmoff/data/geo_tweets.fixed.json` tweets where tweets.entitie
s.hashtags[0].text is not null limit 5;
+--------+
| EXPR$0 |
+--------+
| [{"text":"hiring","indices":[6,13]},{"text":"Job","indices":[98,102]},{"text":"SkilledTrade","indices":[103,116]},{"text":"Tucson","i
ndices":[117,124]},{"text":"Jobs","indices":[129,134]}] |
| [{"text":"lol","indices":[72,76]},{"text":"nowthatsfunny","indices":[77,91]}] |
| [{"text":"WinstonSalem","indices":[0,13]},{"text":"ws","indices":[92,95]}] |
| [{"text":"trndnl","indices":[89,96]}] |
| [{"text":"trndnl","indices":[92,99]}] |
+--------+
5 rows selected (0.187 seconds)
```

If you try and compare the array itself, it doesn't work:

```
0: jdbc:drill:zk=local> select tweets.entities.hashtags from dfs.`/users/rmoff/data/geo_tweets.fixed.json` tweets where tweets.entitie
s.hashtags is not null limit 5;
Error: SYSTEM ERROR: SchemaChangeException: Failure while trying to materialize incoming schema.  Errors:

Error in expression at index -1.  Error: Missing function implementation: [isnotnull(MAP-REPEATED)].  Full expression: --UNKNOWN EXPRES
SION--..

Fragment 0:0

[Error Id: 99ac12aa-f6b4-4692-b815-8f483da682c4 on 192.168.56.1:31010] (state=,code=0)
```

The above example demonstrates using array indexing, which is an alternative to `FLATTEN` for accessing individual objects in the array if you know they're going to exist:

```
0: jdbc:drill:zk=local> select tweets.entities.hashtags[0].text as first_hashtag,text from dfs.`/users/rmoff/data/geo_tweets.fixed.json
` tweets where tweets.entities.hashtags[0].text is not null limit 5;
+---------------+------+
| first_hashtag | text |
+---------------+------+
| hiring | We're #hiring! Click to apply: Bench Jeweler - SEC Oracle &amp; Wetmore - https://t.co/Oe2SHaL0Hh #Job #SkilledTrade #Tucso
n, AZ #Jobs |
| lol | Someone built a wall around Trump's star on the Hollywood Walk of Fame. #lol #nowthatsfunny @… https://t.co/qHWuJXnzbw |
| WinstonSalem | #WinstonSalem Time and place announced for Donald Trump's visit to… https://t.co/6OVl7crshw #ws @winston_salem_ http
s://t.co/l5h220otj4 |
```

Querying CSV files

JSON files are relatively easy to interpret because they have a semi-defined schema within them, including column names. CSV (and character delimited files in general), on the other hand, are a bit more of a 'wild west' when it comes to reliably inferring column names. You can configure Apache Drill (https://drill.apache.org/docs/text-files-csv-tsv-psv/) to ignore the first line of a CSV file (on the assumption that it's a header) if you want to, or to take them as column names. If you don't do this and query a CSV file that looks like this:

```
[oracle@bigdatalite ~]$ head nyc_parking_violations.csv
Summons Number,Plate ID,Registration State,Plate Type,Issue Date,Violation Code,Vehicle Body Type,Vehicle Make,Issuing Agency,Street Co
de1,Street Code2,Street Code3,Vehicle Expiration Date,Violation Location,Violation Precinct,Issuer Precinct,Issuer Code,Issuer Command,
Issuer Squad,Violation Time,Time First Observed,Violation County,Violation In Front Of Or Opposite,House Number,Street Name,Intersectin
g Street,Date First Observed,Law Section,Sub Division,Violation Legal Code,Days Parking In Effect    ,From Hours In Effect,To Hours In
 Effect,Vehicle Color,Unregistered Vehicle?,Vehicle Year,Meter Number,Feet From Curb,Violation Post Code,Violation Description,No Stand
ing or Stopping Violation,Hydrant Violation,Double Parking Violation
1360858775,PHW9801,OH,PAS,07/01/2015,20,SUBN,HONDA,P,61490,26160,26190,0,0044,44,44,929822,0044,0000,0653P,,BX,O,651,RIVER AVE,,0,408,
D,,BBBBBBB,ALL,ALL,,0,0,-,0,,,,,
```

You'll get two records, each one column wide, as an array:

```
0: jdbc:drill:zk=local> select *  from `/user/oracle/incoming/nyc_parking/nyc_parking_violations.csv` LIMIT 5;
+---------+
| columns |
+---------+
| ["Summons Number","Plate ID","Registration State","Plate Type","Issue Date","Violation Code","Vehicle Body Type","Vehicle Make","Issu
ing Agency","Street Code1","Street Code2","Street Code3","Vehicle Expiration Date","Violation Location","Violation Precinct","Issuer Pr
ecinct","Issuer Code","Issuer Command","Issuer Squad","Violation Time","Time First Observed","Violation County","Violation In Front Of
 Or Opposite","House Number","Street Name","Intersecting Street","Date First Observed","Law Section","Sub Division","Violation Legal Co
de","Days Parking In Effect    ","From Hours In Effect","To Hours In Effect","Vehicle Color","Unregistered Vehicle?","Vehicle Year","Me
ter Number","Feet From Curb","Violation Post Code","Violation Description","No Standing or Stopping Violation","Hydrant Violation","Dou
ble Parking Violation"] |
| ["1360858775","PHW9801","OH","PAS","07/01/2015","20","SUBN","HONDA","P","61490","26160","26190","0","0044","44","44","929822","004
4","0000","0653P","","BX","O","651","RIVER AVE","","0","408","D","","BBBBBBB","ALL","ALL","","0","0","-","0","","","","",""] |
```

To access the actual columns in the CSV file you need to use `columns[x]` syntax to reference them. Watch out that `columns` is case-sensitive, and the numbering is zero-based:

```
0: jdbc:drill:zk=local> select columns[1] as `PlateID`, columns[2] as `RegistrationState` from `/user/oracle/incoming/nyc_parking/nyc_p
arking_violations.csv` limit 5;
+----------+--------------------+
| PlateID  | RegistrationState  |
+----------+--------------------+
| AR877A   | NJ                 |
| 73268ME  | NY                 |
| 2050240  | IN                 |
| 2250017  | IN                 |
| AH524C   | NJ                 |
+----------+--------------------+
5 rows selected (0.247 seconds)
```

To make it easier to work with the data on a repeated basis you can define a view over the data:

```
0: jdbc:drill:zk=local> create view dfs.tmp.NYC_Parking_01 as select columns[1] as `PlateID`, columns[2] as `RegistrationState` from `/
user/oracle/incoming/nyc_parking/nyc_parking_violations.csv`;
+-------+-------------------------------------------------------------------+
|  ok   |                              summary                              |
+-------+-------------------------------------------------------------------+
| true  | View 'NYC_Parking_01' created successfully in 'dfs.tmp' schema   |
+-------+-------------------------------------------------------------------+
1 row selected (0.304 seconds)
```

This is using the `dfs` storage plugin and the `tmp` schema within it, which has the following storage configuration - note that `writeable` is true

```
"tmp": {
  "location": "/tmp",
  "writable": true,
  "defaultInputFormat": null
}
```

(if you use the wrong database [storage plugin] or schema you'll get `Schema [hdfs] is immutable.`)

Query the new view

```
0: jdbc:drill:zk=local> select * from dfs.tmp.NYC_Parking_01 limit 5;
+-----------+---------------------+
|  PlateID  |  RegistrationState  |
+-----------+---------------------+
| Plate ID  | Registration State  |
| PHW9801   | OH                  |
| K8010F    | TN                  |
| GFG6211   | NY                  |
| GHL1805   | NY                  |
+-----------+---------------------+
5 rows selected (0.191 seconds)
```

Through the view, or direct against the CSV path, you can also run aggregates:

```
0: jdbc:drill:zk=local> select PlateID,count(*) from dfs.tmp.NYC_Parking_01 group by PlateID having count(*) > 1 limit 1;
+----------+---------+
| PlateID  | EXPR$1  |
+----------+---------+
| 2050240  | 4       |
+----------+---------+
1 row selected (15.983 seconds)
```

Although this isn't rerunnable for the same result - probably because of the `limit` clause

```
0: jdbc:drill:zk=local> select PlateID,count(*) from dfs.tmp.NYC_Parking_01 group by PlateID having count(*) > 1 limit 1;
+----------+---------+
| PlateID  | EXPR$1  |
+----------+---------+
| AR877A   | 3       |
+----------+---------+
1 row selected (12.881 seconds)
```

Under the covers the view definition is written to `/tmp` - you'll want to move this path if you're wanting to preserve this data past reboot:

```
    [oracle@bigdatalite parking]$ cat /tmp/NYC_Parking_01.view.drill
    {
      "name" : "NYC_Parking_01",
      "sql" : "SELECT `columns`[1] AS `PlateID`, `columns`[2] AS `RegistrationState`\nFROM `/user/oracle/incoming/nyc_parking/nyc_parki
ng_violations.csv`",
      "fields" : [ {
        "name" : "PlateID",
        "type" : "ANY",
        "isNullable" : true
      }, {
        "name" : "RegistrationState",
        "type" : "ANY",
        "isNullable" : true
      } ],
      "workspaceSchemaPath" : [ "hdfs" ]
```

You can also create an actual table using CTAS (Create Table As Select):

```
0: jdbc:drill:zk=local> create table dfs.tmp.parking as select columns[1] as `PlateID`, columns[2] as `RegistrationState` from `/user/o
racle/incoming/nyc_parking/nyc_parking_violations.csv`;
+-----------+----------------------------+
| Fragment  | Number of records written  |
+-----------+----------------------------+
| 1_1       | 4471875                    |
| 1_0       | 4788421                    |
+-----------+----------------------------+
2 rows selected (42.913 seconds)
```

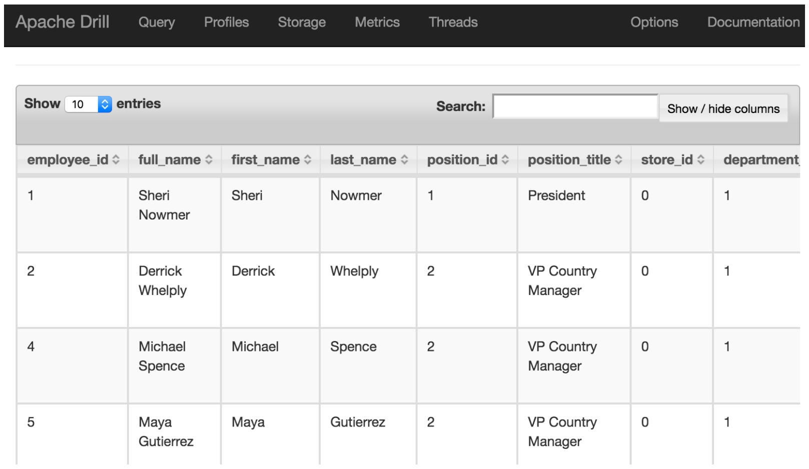
This is stored on disk (per the `dfs` config) and by default in Parquet format:

```
[oracle@bigdatalite parking]$ ls -l /tmp/parking/
total 76508
-rw-r--r--. 1 oracle oinstall 40623288 Aug 10 22:53 1_0_0.parquet
-rw-r--r--. 1 oracle oinstall 37717804 Aug 10 22:53 1_1_0.parquet
```

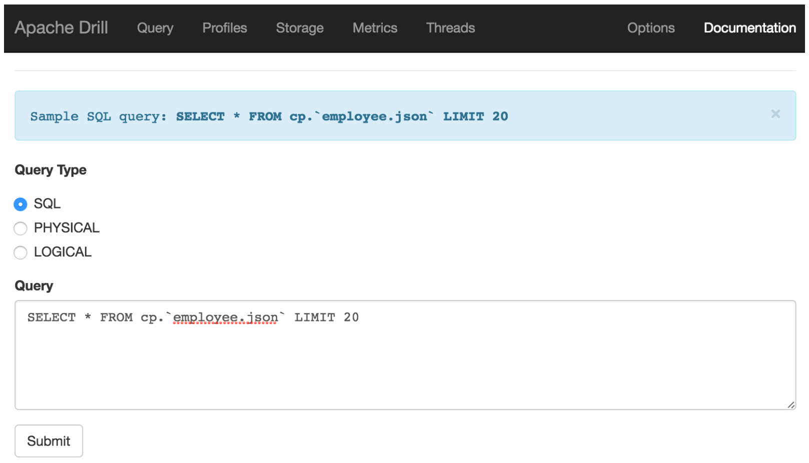
Drill's Web Interface

Drill comes with a web interface which you can access at `http://<IP>:8047/` and is useful for

- Issuing queries



employee_id	full_name	first_name	last_name	position_id	position_title	store_id	department
1	Sheri Nowmer	Sheri	Nowmer	1	President	0	1
2	Derrick Whelply	Derrick	Whelply	2	VP Country Manager	0	1
4	Michael Spence	Michael	Spence	2	VP Country Manager	0	1
5	Maya Gutierrez	Maya	Gutierrez	2	VP Country Manager	0	1



Sample SQL query: SELECT * FROM cp.`employee.json` LIMIT 20

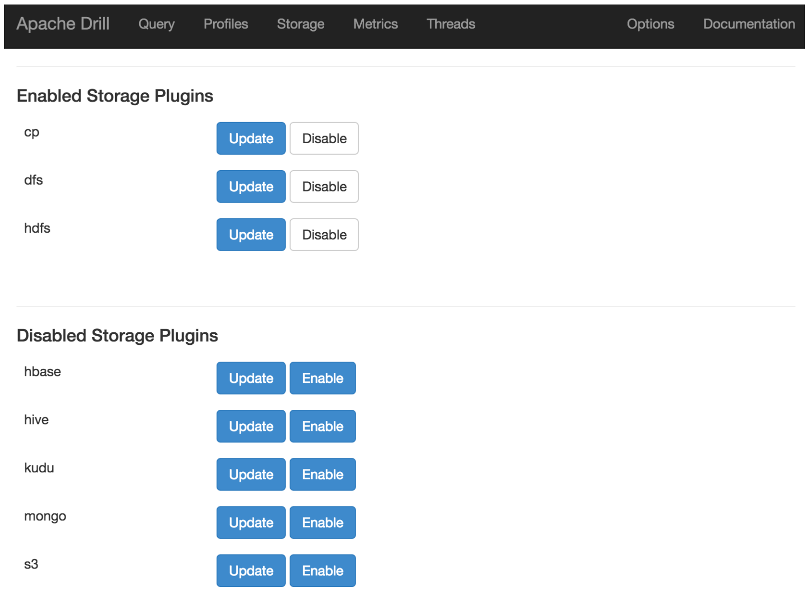
Query Type

[✓ SQL] PHYSICAL LOGICAL

Query

```
SELECT * FROM cp.`employee.json` LIMIT 20
```

- Configuring additional storage plugins (e.g. database (https://drill.apache.org/docs/rdbms-storage-plugin/), hdfs (https://drill.apache.org/docs/file-system-storage-plugin/), etc)



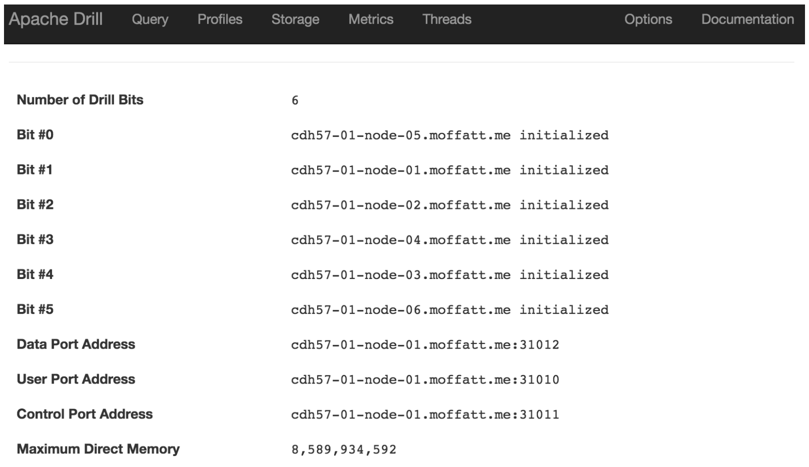
Enabled Storage Plugins

cp	Update	Disable
dfs	Update	Disable
hdfs	Update	Disable

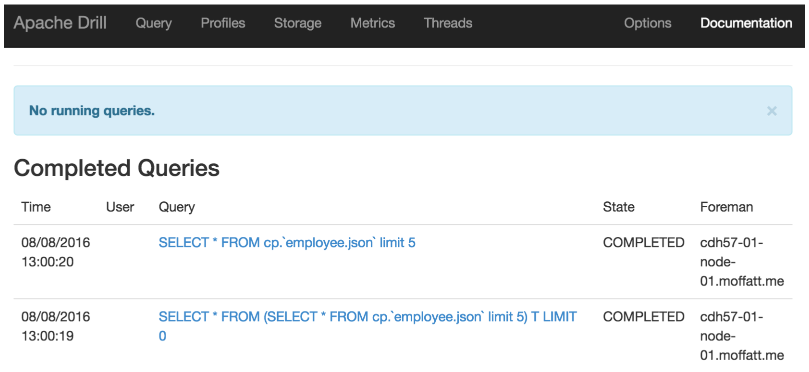
Disabled Storage Plugins

hbase	Update	Enable
hive	Update	Enable
kudu	Update	Enable
mongo	Update	Enable
s3	Update	Enable

- Metrics and debug



Number of Drill Bits	6
Bit #0	cdh57-01-node-05.moffatt.me initialized
Bit #1	cdh57-01-node-01.moffatt.me initialized
Bit #2	cdh57-01-node-02.moffatt.me initialized
Bit #3	cdh57-01-node-04.moffatt.me initialized
Bit #4	cdh57-01-node-03.moffatt.me initialized
Bit #5	cdh57-01-node-06.moffatt.me initialized
Data Port Address	cdh57-01-node-01.moffatt.me:31012
User Port Address	cdh57-01-node-01.moffatt.me:31010
Control Port Address	cdh57-01-node-01.moffatt.me:31011
Maximum Direct Memory	8,589,934,592



No running queries.

Completed Queries

Time	User	Query	State	Foreman
08/08/2016 13:00:20		SELECT * FROM cp.`employee.json` limit 5	COMPLETED	cdh57-01-node-01.moffatt.me
08/08/2016 13:00:19		SELECT * FROM (SELECT * FROM cp.`employee.json` limit 5) T LIMIT 0	COMPLETED	cdh57-01-node-01.moffatt.me

Defining Storage Plugins

From the Drill web interface you can view existing storage plugins, or define new ones. To create a new one, enter its name (for example, `hdfs`, but could be `fred` for all that it matters - it's just a label) under **New Storage Plugin** on the Storage page, and click on Create. Paste the necessary JSON definition in the Configuration box, and then click Create. If you don't want to use the GUI there's also a REST API (https://drill.apache.org/docs/rest-api/#storage).

Storage plugin configuration is stored either within Zookeeper (when running Drill distributed), or locally in the `sys.store.provider.local.path` path when running standalone. By default this is under `/tmp` which gets cleared down at server reboot. To persist custom storage configurations amend the `sys.store.provider.local.path` in `drill-override.conf`, for example:

```
drill.exec: {
    cluster-id: "drillbits1",
    zk.connect: "localhost:2181"
    sys.store.provider.local.path="/home/oracle/drill/"
}
```

Working with filesystem data

Here's an example of a storage configuration that enables Drill to access a CDH cluster's HDFS:

```
    {
      "type": "file",
      "enabled": true,
      "connection": "hdfs://cdh57-01-node-01:8020/",
      "config": null,
      "workspaces": {
        "root": {
          "location": "/",
          "writable": true,
          "defaultInputFormat": null
        }
      },
      "formats": {
        "csv": {
          "type": "text",
          "extensions": [
            "csv"
          ],
          "delimiter": ","
        },
        "json": {
          "type": "json",
          "extensions": [
            "json"
          ]
        }
      }
    }
```

As well as the `connection` parameter itself for HDFS, the important bit in this configuration is the `formats` section. This tells Drill how to interpet files that it finds, without the end-user having to explicitly declare their type.

For the filesystem-based plugin `dfs` (which can include local files, HDFS, even Amazon S3), you can browse the available "tables":

List the files in HDFS (previously selected with `use hdfs;`)

```
0: jdbc:drill:zk=local> show files;
+--------+--------------+---------+---------+--------+-------------+-------------+------------------------+--------------------------+
|  name  | isDirectory  | isFile  | length  | owner  |    group    | permissions |       accessTime       |     modificationTime     |
+--------+--------------+---------+---------+--------+-------------+-------------+------------------------+--------------------------+
| hbase  | true         | false   | 0       | hbase  | supergroup  | rwxr-xr-x   | 1969-12-31 19:00:00.0  | 2016-07-25 14:46:08.212  |
| share  | true         | false   | 0       | hdfs   | supergroup  | rwxrwxrwx   | 1969-12-31 19:00:00.0  | 2016-05-15 12:28:08.152  |
| solr   | true         | false   | 0       | solr   | solr        | rwxr-xr-x   | 1969-12-31 19:00:00.0  | 2016-06-01 18:34:50.716  |
| tmp    | true         | false   | 0       | hdfs   | supergroup  | rwxrwxrwt   | 1969-12-31 19:00:00.0  | 2016-06-24 04:54:41.491  |
| user   | true         | false   | 0       | hdfs   | supergroup  | rwxrwxrwx   | 1969-12-31 19:00:00.0  | 2016-06-21 15:55:59.084  |
| var    | true         | false   | 0       | hdfs   | supergroup  | rwxr-xr-x   | 1969-12-31 19:00:00.0  | 2016-05-11 17:53:29.804  |
+--------+--------------+---------+---------+--------+-------------+-------------+------------------------+--------------------------+
6 rows selected (0.145 seconds)
```

Show files in a given path:

```
0: jdbc:drill:zk=local> show files in `/user/oracle`;
+-----------------+--------------+---------+---------+---------+---------+--------------+------------------------+----------------------
----+
|      name       | isDirectory  | isFile  | length  |  owner  |  group  | permissions  |       accessTime       |     modificationTime
|
+-----------------+--------------+---------+---------+---------+---------+--------------+------------------------+----------------------
----+
| .Trash          | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-05-23 20:42:34.8
15  |
| .sparkStaging   | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-07-06 03:56:38.8
63  |
| .staging        | true         | false   | 0       | oracle  | oracle  | rwx------    | 1969-12-31 19:00:00.0  | 2016-06-01 18:37:04.0
05  |
| incoming        | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-08-03 05:34:12.3
8   |
| mediademo       | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-06-01 18:59:45.6
53  |
| moviedemo       | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-05-15 12:02:55.6
52  |
| moviework       | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-05-15 12:03:01.4
97  |
| oggdemo         | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-05-15 12:03:01.5
52  |
| oozie-oozi      | true         | false   | 0       | oracle  | oracle  | rwxr-xr-x    | 1969-12-31 19:00:00.0  | 2016-05-15 12:03:01.6
51  |
+-----------------+--------------+---------+---------+---------+---------+--------------+------------------------+----------------------
----+
9 rows selected (0.428 seconds)
```

You can also query across multiple files by specifying a wildcard match. Here's the truncated list of files available:

```
0: jdbc:drill:zk=cdh57-01-node-01.moffatt.me:> show files in `hdfs`.`/user/rmoff/incoming/twitter/2016/06/17/tweets/`;
+--------------------------+--------------+---------+----------+--------+--------+--------------+--------------------------+-----------
---------------+
|           name           | isDirectory  | isFile  |  length  | owner  | group  | permissions  |        accessTime        |     modifi
cationTime     |
+--------------------------+--------------+---------+----------+--------+--------+--------------+--------------------------+-----------
---------------+
| FlumeData.1466176113171  | false        | true    | 1055675  | rmoff  | rmoff  | rw-r--r--    | 2016-08-10 21:28:27.072  | 2016-06-17
16:08:38.023  |
| FlumeData.1466176113172  | false        | true    | 1051411  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.756  | 2016-06-17
16:08:40.597  |
| FlumeData.1466176113173  | false        | true    | 1054734  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.752  | 2016-06-17
16:08:43.33   |
| FlumeData.1466176113174  | false        | true    | 1050991  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.743  | 2016-06-17
16:08:44.361  |
| FlumeData.1466176113175  | false        | true    | 1053577  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.748  | 2016-06-17
16:08:45.162  |
| FlumeData.1466176113176  | false        | true    | 1051965  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.752  | 2016-06-17
16:08:46.261  |
| FlumeData.1466176113177  | false        | true    | 1049555  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.758  | 2016-06-17
16:08:47.425  |
| FlumeData.1466176113178  | false        | true    | 1050566  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.758  | 2016-06-17
16:08:48.23   |
| FlumeData.1466176113179  | false        | true    | 1051751  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.756  | 2016-06-17
16:08:49.381  |
| FlumeData.1466176113180  | false        | true    | 1052249  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.757  | 2016-06-17
16:08:50.042  |
| FlumeData.1466176113181  | false        | true    | 1055002  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.758  | 2016-06-17
16:08:50.896  |
| FlumeData.1466176113182  | false        | true    | 1050812  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.758  | 2016-06-17
16:08:52.191  |
| FlumeData.1466176113183  | false        | true    | 1048954  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.757  | 2016-06-17
16:08:52.994  |
| FlumeData.1466176113184  | false        | true    | 1051559  | rmoff  | rmoff  | rw-r--r--    | 2016-08-05 20:46:51.773  | 2016-06-17
16:08:54.025  |
[...]
```

Count number of records in one file (`FlumeData.1466176113171`):

```
0: jdbc:drill:zk=cdh57-01-node-01.moffatt.me:> SELECT count(*) FROM table(`hdfs`.`/user/rmoff/incoming/twitter/2016/06/17/tweets/FlumeD
ata.1466176113171`(type => 'json'));
+---------+
| EXPR$0  |
+---------+
| 277     |
+---------+
1 row selected (0.798 seconds)
```

In several files (`FlumeData.146617611317*`):

```
0: jdbc:drill:zk=cdh57-01-node-01.moffatt.me:> SELECT count(*) FROM table(`hdfs`.`/user/rmoff/incoming/twitter/2016/06/17/tweets/FlumeD
ata.146617611317*`(type => 'json'));
+---------+
| EXPR$0  |
+---------+
| 2415    |
+---------+
1 row selected (2.466 seconds)
```

In all files in the folder (`*`):

```
0: jdbc:drill:zk=cdh57-01-node-01.moffatt.me:> SELECT count(*) FROM table(`hdfs`.`/user/rmoff/incoming/twitter/2016/06/17/tweets/*`(typ
e => 'json'));
+---------+
| EXPR$0  |
+---------+
| 7414    |
+---------+
1 row selected (3.867 seconds)
```

And even across multiple folders:

```
0: jdbc:drill:zk=cdh57-01-node-01.moffatt.me:> SELECT count(*) FROM table(`hdfs`.`/user/flume/incoming/twitter/2016/06/*/*`(type => 'js
on'));
+---------+
| EXPR$0  |
+---------+
| 206793  |
+---------+
1 row selected (87.545 seconds)
```

Querying data without an identifying extension

Drill relies on the `format` clause of the storage extension configurations in orer to determine how to interpret files based on their extensions. You won't always have that luxury of extensions being available, or being defined. If you try and query such data, you'll not get far. In this example I'm querying data on HDFS that's in JSON format but without the `.json` suffix:

```
0: jdbc:drill:zk=cdh57-01-node-01.moffatt.me:> SELECT text FROM `hdfs`.`/user/rmoff/incoming/twitter/2016/06/17/tweets/FlumeData.146617
6113171` limit 5;
Error: VALIDATION ERROR: From line 1, column 18 to line 1, column 23: Table 'hdfs./user/rmoff/incoming/twitter/2016/06/17/tweets/FlumeD
ata.1466176113171' not found

SQL Query null
```

Fear not - you can declare them as part of the query syntax.

```
0: jdbc:drill:zk=cdh57-01-node-01.moffatt.me:> SELECT text FROM table(`hdfs`.`/user/rmoff/incoming/twitter/2016/06/17/tweets/FlumeData.
1466176113171`(type => 'json')) limit 5;
+------+
| text |
+------+
| RT @jjkukrl: susu bayi jg lagi mahal nih ugh ayah harus semangat cari duit ^^9 https://t.co/2NvTOShRbI |
| Oracle Java 1Z0-808 Web Exam Simulator https://t.co/tZ3gU8EMj3 |
| @TribuneSelatan ahaha kudu gaya atuh da arek lebarann ahahaha |
| Short impression of yesterday's speech. What a great day it was! #lifeatoracle #team #salesincentive #oracle https://t.co/SVK2ovOe3U
 |
| Want to work at Oracle? We're #hiring in New York! Click for details: https://t.co/NMTo1WMHVw #Sales #Job #Jobs #CareerArc |
+------+
5 rows selected (1.267 seconds)
```

Storage Configuration - Oracle

Per the documentation (https://drill.apache.org/docs/rdbms-storage-plugin/) it's easy to query data residing in a RDBMS, such as Oracle. Simply copy the JDBC driver into Apache Drill's jar folder:

```
cp /u01/app/oracle/product/12.1.0.2/dbhome_1/jdbc/lib/ojdbc7.jar /opt/apache-drill-1.7.0/jars/3rdparty/
```

And then add the necessary storage configuration, which I called `ora`:

```
    {
      "type": "jdbc",
      "driver": "oracle.jdbc.OracleDriver",
      "url": "jdbc:oracle:thin:moviedemo/welcome1@localhost:1521/ORCL",
      "username": null,
      "password": null,
      "enabled": true
    }
```

If you get an error `Please retry: error (unable to create/ update storage)` then check that the target Oracle database is up, the password is correct, and so on.

You can then query the data within Hive:

```
0: jdbc:drill:zk=local> use ora.MOVIEDEMO;
+-------+--------------------------------------------+
|  ok   |                  summary                   |
+-------+--------------------------------------------+
| true  | Default schema changed to [ora.MOVIEDEMO]  |
+-------+--------------------------------------------+
1 row selected (0.205 seconds)

0: jdbc:drill:zk=local> show tables;
+----------------+-----------------------------+
|  TABLE_SCHEMA  |         TABLE_NAME          |
+----------------+-----------------------------+
| ora.MOVIEDEMO  | ACTIVITY                    |
| ora.MOVIEDEMO  | BDS_CUSTOMER_RFM            |
| ora.MOVIEDEMO  | BUSINESS_REVIEW_SUMMARY     |
[...]

0: jdbc:drill:zk=local> select * from ACTIVITY limit 5;
+--------------+---------+
| ACTIVITY_ID  |  NAME   |
+--------------+---------+
| 3.0          | Pause   |
| 6.0          | List    |
| 7.0          | Search  |
| 8.0          | Login   |
| 9.0          | Logout  |
+--------------+---------+
5 rows selected (1.644 seconds)
```

If you get `Error: DATA_READ ERROR: The JDBC storage plugin failed while trying setup the SQL query.` then enable verbose errors in Apache Drill to see what the problem is:

```
0: jdbc:drill:zk=local> ALTER SESSION SET `exec.errors.verbose` = true;
+-------+-------------------------------+
|  ok   |            summary            |
+-------+-------------------------------+
| true  | exec.errors.verbose updated.  |
+-------+-------------------------------+
1 row selected (0.154 seconds)

0: jdbc:drill:zk=local> select * from ora.MOVIEDEMO.YELP_BUSINESS limit 1;
Error: DATA_READ ERROR: The JDBC storage plugin failed while trying setup the SQL query.

sql SELECT *
FROM "MOVIEDEMO"."YELP_BUSINESS"
plugin ora
Fragment 0:0

[Error Id: 40343dd5-1354-48ed-90ef-77ae1390411b on bigdatalite.localdomain:31010]

(java.sql.SQLException) ORA-29913: error in executing ODCIEXTTABLEOPEN callout
ORA-29400: data cartridge error
KUP-11504: error from external driver: MetaException(message:Could not connect to meta store using any of the URIs provided. Most recen
t failure: org.apache.thrift.transport.TTransportException: java.net.ConnectException: Connection refused
```

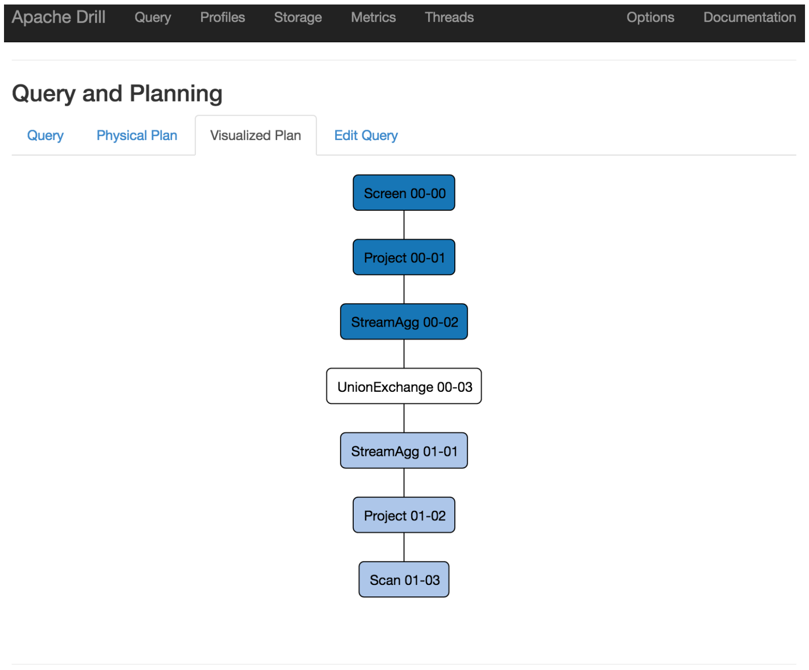
Here the problem was with the external table that Oracle was querying (`ORA-29913: error in executing ODCIEXTTABLEOPEN`). It's actually an Oracle external table over a Hive table, which obviously Drill could be querying directly - but hey, we're just sandboxing here...

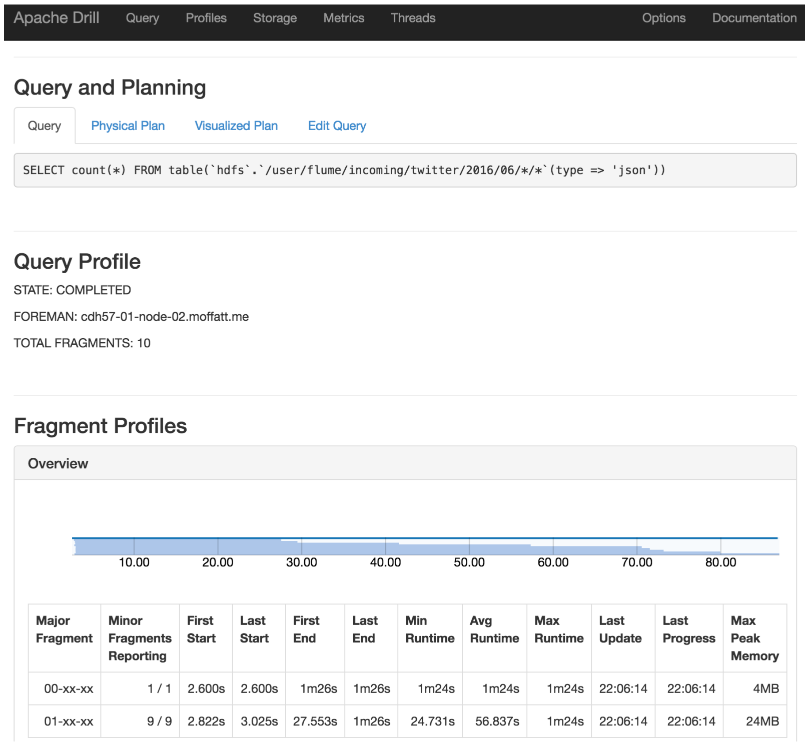
Query Execution

Just as Oracle has its Cost Based Optimiser (CBO) which helps it determine how to execute a query, and do so most efficiently, Apache Drill has an execution engine that determines how to actually execute the query you give it. This also includes how to split it up over multiple nodes ("drillbits") if available, as well as optimisations such as partition pruning (https://drill.apache.org/docs/how-to-partition-data/) in certain cases. You can read more about how the query execution works here (https://drill.apache.org/docs/drill-query-execution/), and watch a good explanation of it (https://www.youtube.com/watch?v=0rurIzOkTIg) here. To view the explain plan for a query use `explain plan` :

```
0: jdbc:drill:zk=local> !set maxwidth 10000
0: jdbc:drill:zk=local> explain plan for select `date`,count(*) as tip_count from `/user/oracle/incoming/yelp/tip_json/yelp_academic_da
taset_tip.json` group by `date` order by 2 desc limit 5;
+------+------+
| text | json |
+------+------+
| 00-00    Screen
00-01      Project(date=[$0], tip_count=[$1])
00-02        SelectionVectorRemover
00-03          Limit(fetch=[5])
00-04            SelectionVectorRemover
00-05              TopN(limit=[5])
00-06                HashAgg(group=[{0}], tip_count=[$SUM0($1)])
00-07                  HashAgg(group=[{0}], tip_count=[COUNT()])
00-08                    Scan(groupscan=[EasyGroupScan [selectionRoot=hdfs://localhost:8020/user/oracle/incoming/yelp/tip_json/yelp_aca
demic_dataset_tip.json, numFiles=1, columns=[`date`], files=[hdfs://localhost:8020/user/oracle/incoming/yelp/tip_json/yelp_academic_dat
aset_tip.json]]])
 | {
  "head" : {
    "version" : 1,
    "generator" : {
      "type" : "ExplainHandler",
      "info" : ""
    [...]
```

You can also use the Drill web interface to see information about how a query executed:





Query and Planning

```
SELECT count(*) FROM table(`hdfs`.`/user/flume/incoming/twitter/2016/06/*/*`(type => 'json'))
```

Query Profile

STATE: COMPLETED

FOREMAN: cdh57-01-node-02.moffatt.me

TOTAL FRAGMENTS: 10

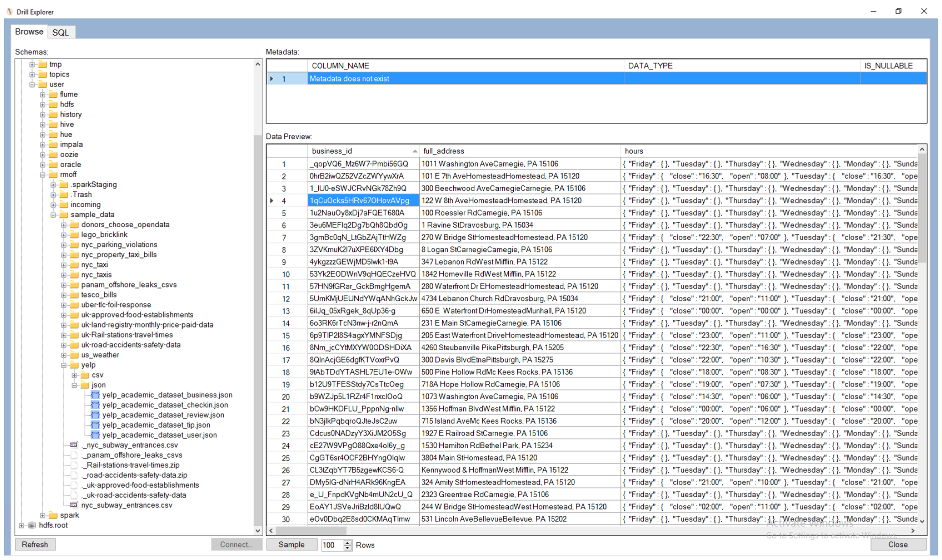
Fragment Profiles

Overview

Major Fragment	Minor Fragments Reporting	First Start	Last Start	First End	Last End	Min Runtime	Avg Runtime	Max Runtime	Last Update	Last Progress	Max Peak Memory
00-xx-xx	1 / 1	2.600s	2.600s	1m26s	1m26s	1m24s	1m24s	1m24s	22:06:14	22:06:14	4MB
01-xx-xx	9 / 9	2.822s	3.025s	27.553s	1m26s	24.731s	56.837s	1m24s	22:06:14	22:06:14	24MB

Drill Explorer

The MapR Drill ODBC driver comes with a tool called **Drill Explorer (https://drill.apache.org/docs/drill-explorer-introduction/)**. This is a GUI that enables you to explore the data by navigating the databases (==storage plugins) and folders/files within, previewing the data and even creating views on it.



```
tmp
topics
user
  flume
  hdfs
  history
  hive
  hue
  impala
  oozie
  oracle
  rmoff
    .sparkStaging
    .Trash
    incoming
    sample_data
      donors_choose_opendata
      lego_bricklink
      nyc_parking_violations
      nyc_property_taxi_bills
      nyc_taxi
      nyc_taxis
      panam_offshore_leaks_csvs
      tesco_bills
      uber-tlc-foil-response
      uk-approved-food-establishments
      uk-land-registry-monthly-price-paid-data
      uk-Rail-stations-travel-times
      uk-road-accidents-safety-data
      us_weather
      yelp
        csv
        json
          yelp_academic_dataset_business.json
          yelp_academic_dataset_checkin.json
          yelp_academic_dataset_review.json
          yelp_academic_dataset_tip.json
          yelp_academic_dataset_user.json
      ._nyc_subway_entrances.csv
      ._panam_offshore_leaks_csvs
      ._Rail-stations-travel-times.zip
      ._road-accidents-safety-data.zip
      ._uk-approved-food-establishments
      ._uk-road-accidents-safety-data
      nyc_subway_entrances.csv
  spark
hdfs.root
```

	COLUMN_NAME	DATA_TYPE	IS_NULLABLE
1	Metadata does not exist		

	business_id	full_address	hours
1	_qopVQ6_Mz6W7-Pmbi56GQ	1011 Washington AveCarnegie, PA 15106	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
2	0hrB2iwQZ52VZcZWYywXrA	101 E 7th AveHomesteadHomestead, PA 15120	{ "Friday" : { "close" : "16:30", "open" : "08:00" }, "Tuesday" : { "close" : "16:30", "ope
3	1_IU0-eSWJCRvNGk78Zh9Q	300 Beechwood AveCarnegieCarnegie, PA 15106	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
4	1qCuOcks5HRv67OHovAVpg	122 W 8th AveHomesteadHomestead, PA 15120	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
5	1u2NauOy8xDj7aFQET680A	100 Roessler RdCarnegie, PA 15106	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
6	3eu6MEFlq2Dg7bQh8QbdOg	1 Ravine StDravosburg, PA 15034	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
7	3gmBc0qN_LtGbZAjTtHWZg	270 W Bridge StHomesteadHomestead, PA 15120	{ "Friday" : { "close" : "22:30", "open" : "07:00" }, "Tuesday" : { "close" : "21:30", "ope
8	3ZVKmuK2l7uXPE6lXY4Dbg	8 Logan StCarnegieCarnegie, PA 15106	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
9	4ykgzzzGEWjMD5lwk1-l9A	347 Lebanon RdWest Mifflin, PA 15122	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
10	53Yk2EODWnV9qHQECzeHVQ	1842 Homeville RdWest Mifflin, PA 15122	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
11	57HN9fGRar_GckBmgHgemA	280 Waterfront Dr EHomesteadHomestead, PA 15120	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
12	5UmKMjUEUNdYWqANhGckJw	4734 Lebanon Church RdDravosburg, PA 15034	{ "Friday" : { "close" : "21:00", "open" : "11:00" }, "Tuesday" : { "close" : "21:00", "ope
13	6ilJq_05xRgek_8qUp36-g	650 E Waterfront DrHomesteadMunhall, PA 15120	{ "Friday" : { "close" : "00:00", "open" : "00:00" }, "Tuesday" : { "close" : "00:00", "ope
14	6o3RK6rTcN3nw-j-r2nQmA	231 E Main StCarnegieCarnegie, PA 15106	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
15	6p9TIP2l8S4agxYMNFSDjg	205 East Waterfront DriveHomesteadHomestead, PA 15120	{ "Friday" : { "close" : "23:00", "open" : "11:00" }, "Tuesday" : { "close" : "23:00", "ope
16	8Nm_jcCYtMXYW0ODSHDiXA	4260 Steubenville PikePittsburgh, PA 15205	{ "Friday" : { "close" : "22:30", "open" : "16:30" }, "Tuesday" : { "close" : "22:00", "ope
17	8QlnAcjGE6dgfKTVoxrPvQ	300 Davis BlvdEtnaPittsburgh, PA 15275	{ "Friday" : { "close" : "22:00", "open" : "10:30" }, "Tuesday" : { "close" : "22:00", "ope
18	9tAbTDdYTASHL7EU1e-OWw	500 Pine Hollow RdMc Kees Rocks, PA 15136	{ "Friday" : { "close" : "18:00", "open" : "08:30" }, "Tuesday" : { "close" : "18:00", "ope
19	b12U9TFESStdy7CsTtcOeg	718A Hope Hollow RdCarnegie, PA 15106	{ "Friday" : { "close" : "19:00", "open" : "07:30" }, "Tuesday" : { "close" : "19:00", "ope
20	b9WZJp5L1RZr4F1nxclOoQ	1073 Washington AveCarnegie, PA 15106	{ "Friday" : { "close" : "14:30", "open" : "06:00" }, "Tuesday" : { "close" : "14:30", "ope
21	bCw9HKDFLU_PppnNg-nllw	1356 Hoffman BlvdWest Mifflin, PA 15122	{ "Friday" : { "close" : "00:00", "open" : "06:00" }, "Tuesday" : { "close" : "00:00", "ope
22	bN3jlkPqbqroQJteJsC2uw	715 Island AveMc Kees Rocks, PA 15136	{ "Friday" : { "close" : "20:00", "open" : "12:00" }, "Tuesday" : { "close" : "20:00", "ope
23	Cdcus0NADzyY3XiJM2O5Sg	1927 E Railroad StCarnegie, PA 15106	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
24	cE27W9VPgO88Qxe4ol6y_g	1530 Hamilton RdBethel Park, PA 15234	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
25	CgGT6sr4OCF2BHYngOlqlw	3804 Main StHomestead, PA 15120	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
26	CL3tZqbYT7B5zgewKCS6-Q	Kennywood & HoffmanWest Mifflin, PA 15122	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
27	DMy5lG-dNrH4ARk96KngEA	324 Amity StHomesteadHomestead, PA 15120	{ "Friday" : { "close" : "21:00", "open" : "10:00" }, "Tuesday" : { "close" : "21:00", "ope
28	e_U_FnpdKVgNb4mUN2cU_Q	2323 Greentree RdCarnegie, PA 15106	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda
29	EoAY1JSVeJriBzld8lUQwQ	244 W Bridge StHomesteadWest Homestead, PA 15120	{ "Friday" : { "close" : "02:00", "open" : "11:00" }, "Tuesday" : { "close" : "02:00", "ope
30	eOv0Dbq2E8sd0CKMAqTlmw	531 Lincoln AveBellevueBellevue, PA 15202	{ "Friday" : { }, "Tuesday" : { }, "Thursday" : { }, "Wednesday" : { }, "Monday" : { }, "Sunda

Drill Client

Within the Drill client there are various settings available:

Share this Post

(https://twitter.com/intent/tweet?text=An%20Introduction%20to%20Apache%20Drill&url=https://www.rittmanmead.com/blog/2016/08/a introduction-to-apache-drill/) (https://www.facebook.com/sharer/sharer.php?u=https://www.rittmanmead.com/blog/2016/08/an-introduction-to-apache-drill/) (https://plus.google.com/share?url=https://www.rittmanmead.com/blog/2016/08/an-introduction-to-apache-drill/)

TECHNICAL INSIGHTS (/BLOG/TAG/TECHNICAL)

BUSINESS INSIGHTS (/BLOG/TAG/BUSINESS-INSIGHTS)

RITTMAN MEAD LIFE (/BLOG/TAG/RITTMAN-MEAD-LIFE)

Recent Posts

- OA Summit 2020: OA Roadmap Summary (/blog/2020/06/oa-summit-2020-oracle-analytics-roadmap-summary/)
- Data Virtualization: What is it About? (/blog/2020/06/data-virualization-what-is-it/)
- Getting Smart View to work with OAC (/blog/2020/05/getting-smart-view-to-work-with-oac/)
- Oracle Analytics: Everything you always wanted to know (But were afraid to ask) (/blog/2020/02/oracle-analytics-everything-you-always-wanted-to-know-but-were-afraid-to-ask/)
- Oracle Data Science - Accelerated Data Science SDK Configuration (/blog/2020/02/accelerated-data-science-sdk-configuration/)

Sign Up for Our Newsletter

email address

SUBSCRIBE

(/blog/2016/08/using-apache-drill-with-obiee-12c/)

READ THIS NEXT

Using Apache Drill with OBIEE 12c

Apache Drill enables querying with SQL against a multitude of datasources including things like JSON files, Parquet and Avro,…

(/blog/2016/07/obiee-12-2-1-1-0-new-feature-guide/)

YOU MIGHT ENJOY

OBIEE 12.2.1.1.0 - New Feature Guide

Oracle has recently released version 12.2.1.1.0 for OBIEE 12c, which has a variety of exciting…

```
0: jdbc:drill:zk=local> !set
autocommit          true
autosave            false
color               true
fastconnect         true
force               false
headerinterval      100
historyfile         /home/oracle/.sqlline/history
incremental         true
isolation           TRANSACTION_REPEATABLE_READ
maxcolumnwidth      15
maxheight           56
maxwidth            1000000
numberformat        default
outputformat        table
propertiesfile      /home/oracle/.sqlline/sqlline.properties
rowlimit            0
showelapsedtime     true
showheader          true
shownestederrs      false
showwarnings        true
silent              false
timeout             -1
trimscripts         true
verbose             false
```

To change one, such as the width of output displayed:

```
0: jdbc:drill:zk=local> !set maxwidth 10000
```

To connect to remote Drill specify the Zookeeper node(s) that store the Drillbit connection information:

```
rmoff@asgard-3:apache-drill-1.7.0> bin/sqlline -u jdbc:drill:zk=cdh57-01-node-01.moffatt.me:2181,cdh57-01-node-02.moffatt.me:2181,cdh57
-01-node-03.moffatt.me:2181
```

Conclusion

Apache Drill is a powerful tool for using familiar querying language (SQL) against different data sources. On a small scale, simply being able to slice and dice through structured files like JSON is a massive win. On a larger scale, it will be interesting to experiment with how Apache Drill compares when querying larger volumes of data across a cluster of machines, maybe compared to a tool such as Impala.

For more information about Apache Drill see how to access Drill from within OBIEE (http://www.rittmanmead.com/blog/2016/08/using-apache-drill-with-obiee-12c/), as well as bonus geeky blog coming soon explaining the debug tools I used to try and figure out why it wouldn't initially work...

(/blog/auth moffatt/)

Robin Moffatt (/blog/author/robin-moffatt/)

Read more posts (/blog/author/robin-moffatt/) by this author.

Yorkshire, UK *https://www.linkedin.com/in/robinmoffatt (https://www.linkedin.com/in/robinmoffatt)*

About Us

Rittman Mead is a data and analytics company who specialise in data visualisation, predictive analytics, enterprise reporting and data engineering.

(http://www.rittmanmead.com/feed/) (http://twitter.com/rittmanmead)

(https://www.facebook.com/rittmanmead/) (http://www.linkedin.com/company/rittman-mead)

Contact Us

Rittman Mead Consulting Ltd.
Platf9rm, Hove Town Hall
Tisbury Road,
Brighton, BN3 3BQ
United Kingdom

Tel: (Phone) +44 1273 053956
Email: (Email)info@rittmanmead.com (mailto:info@rittmanmead.com)