



DA 410 Drill Essentials Lab Guide

Q1-15

This Lab Guide (the “Lab Guide”) is protected under U.S. and international copyright laws, and is the exclusive property of MapR Technologies, Inc. © 2015, MapR Technologies, Inc. All rights reserved.



Get Started

Lab Overview

The MapR Sandbox with Apache Drill is a fully functional single-node, virtual machine cluster that can be used to get an overview of Apache Drill in a Hadoop environment. Note that the standard MapR sandbox does not include Apache Drill by default. The MapR Sandbox with Apache Drill was designed to provide a separate sandbox with Apache Drill preinstalled, and all of the data needed to perform the labs in this document. In this lab you will:

- Lab 0.1: Download and Install the MapR Sandbox with Drill
- Lab 0.2: Connect to the MapR Sandbox

System Requirements

The MapR Sandbox with Apache Drill runs on VMware Player and VirtualBox, free desktop applications with Windows, MacOS, or Linux. Before you install the MapR Sandbox, verify that the host system meets the following prerequisites:

- VMware Player or VirtualBox is installed.
- At least 20 GB free hard disk space, at least 2 physical cores, and 8 GB of RAM is available. Performance will increase with more CPU cores, RAM and free hard disk space.
- Uses one of the following 64-bit x86 architectures:
 - A 1.3 GHz or faster AMD CPU with segment-limit support in long mode
 - A 1.3 GHz or faster Intel CPU with VT-x support
- If you have an Intel CPU with VT-x support, verify that VT-x support is enabled in the host system BIOS. The BIOS settings that must be enabled for VT-x support vary depending on the system vendor. See the VMware knowledge base article at <http://kb.vmware.com/kb/1003944> for information about how to determine if VT-x support is enabled.
- For Linux, MacOS, or Windows, download the free [VMware Player](#) or [VirtualBox](#). Optionally, you can purchase [VMware Fusion](#) for MacOS.
- You will need to use a Secure Shell (ssh) client to log into the Sandbox. [PuTTY](#), [Cygwin](#), or a bash shell on a Unix-like OS with OpenSSH installed, such as Terminal on MacOS, are all acceptable clients. If you do not have an ssh client, you can log in to the Sandbox using the virtual machine console mode.

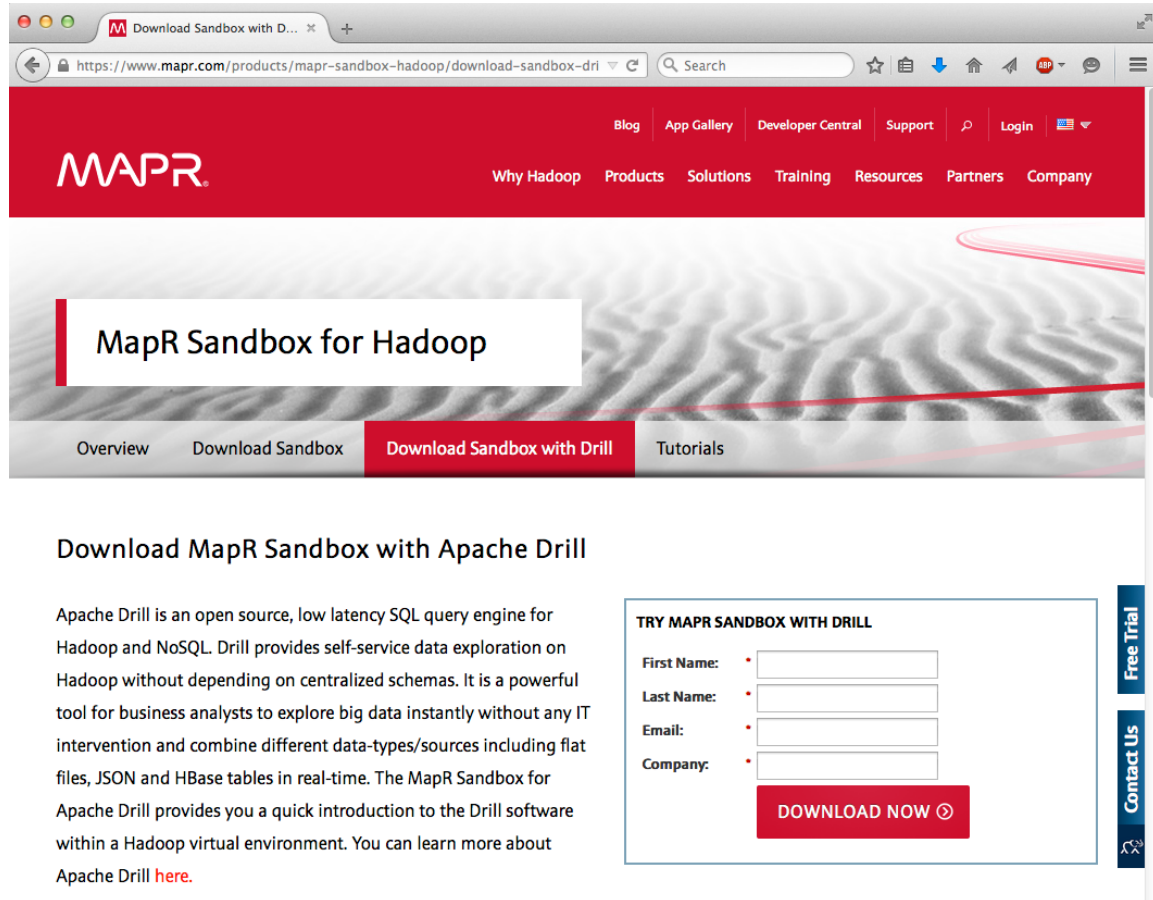
Lab Procedure

Lab 1: Download and Install the MapR Sandbox with Drill

1. Download the MapR Sandbox with Drill from the link below:

<https://www.mapr.com/products/mapr-sandbox-hadoop/download-sandbox-drill>

Figure 0.1: MapR Sandbox



Download MapR Sandbox with Apache Drill

Apache Drill is an open source, low latency SQL query engine for Hadoop and NoSQL. Drill provides self-service data exploration on Hadoop without depending on centralized schemas. It is a powerful tool for business analysts to explore big data instantly without any IT intervention and combine different data-types/sources including flat files, JSON and HBase tables in real-time. The MapR Sandbox for Apache Drill provides you a quick introduction to the Drill software within a Hadoop virtual environment. You can learn more about Apache Drill [here](#).

TRY MAPR SANDBOX WITH DRILL

First Name:
Last Name:
Email:
Company:

DOWNLOAD NOW ↻

2. Follow the instructions from the appropriate link to install and configure the sandbox.

To use the sandbox with **VMWare Player** or **VMWare Fusion**:

[Installing the MapR Sandbox with Apache Drill on VMware Player/VMware Fusion](#)

To use the sandbox with **VirtualBox**:

[Installing the MapR Sandbox with Apache Drill on VirtualBox](#)

Note: The MapR Sandbox for Apache Drill on VirtualBox comes with NAT port forwarding enabled. The MapR Sandbox for Apache Drill on VMware Player supports NAT, but does not support port forwards.

3. For more information about the MapR sandbox with Apache Drill, or installation procedures, visit the MapR Doc page at:

<http://doc.mapr.com/display/MapR/Installing+the+Apache+Drill+Sandbox>

Lab 2: Connect to the MapR Sandbox

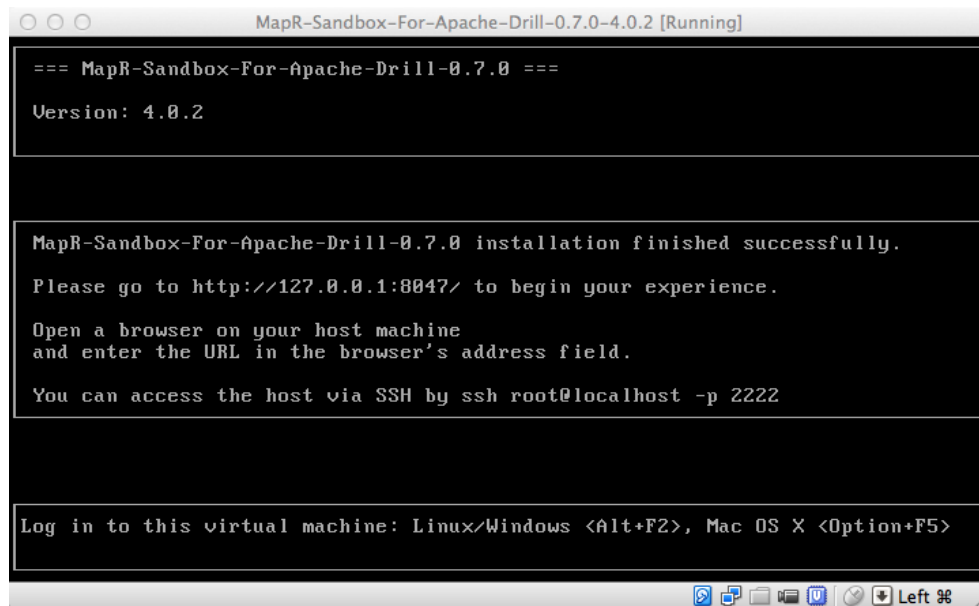
Apache Drill can be used to query data using the SSH terminal, as well as through a web interface at port 8047. Both options will be demonstrated in the lab exercises for this course.

1. Verify that your MapR Sandbox with Apache Drill is running, and make a note of the internal IP address for the sandbox (in this example: 127.0.0.1).

Figure 0.2: MapR Sandbox running in VirtualBox



Figure 0.3: Sandbox IP address



2. Open your SSH Client (PuTTY for Windows and Terminal for MacOS pictured)

Figure 0.4: PuTTY Configuration

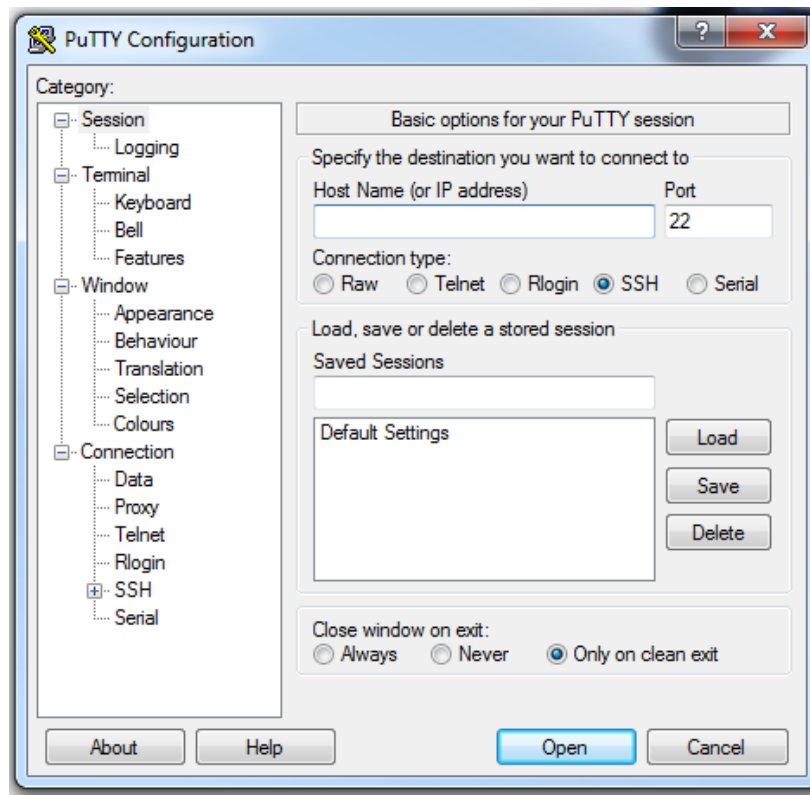
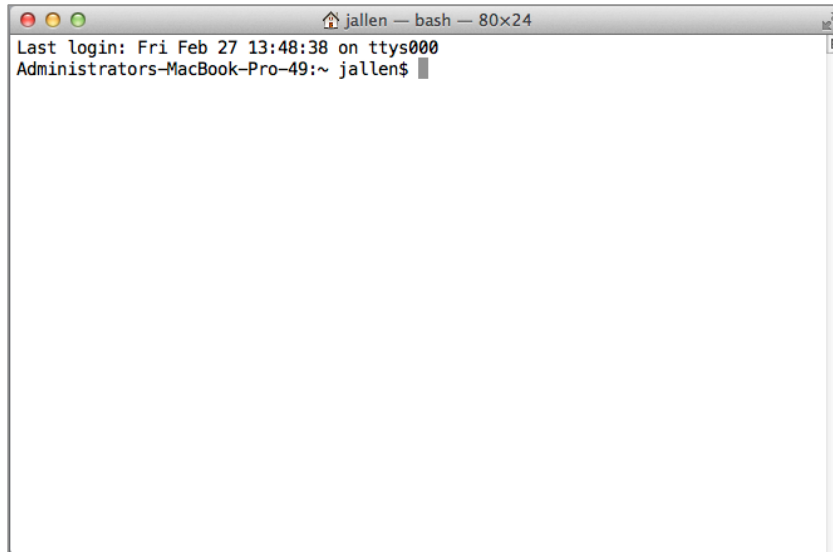


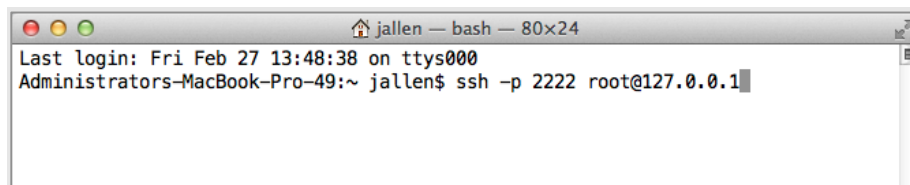
Figure 0.5: MacOS bash terminal



3. Log into the Sandbox as user 'root', with the password 'mapr'
 - a. When using PuTTY, type the IP address from step 1 into the Host Name (or IP address) input field, and select "Open". When prompted, log in as user 'root'.
 - b. When using Terminal, type at the prompt:

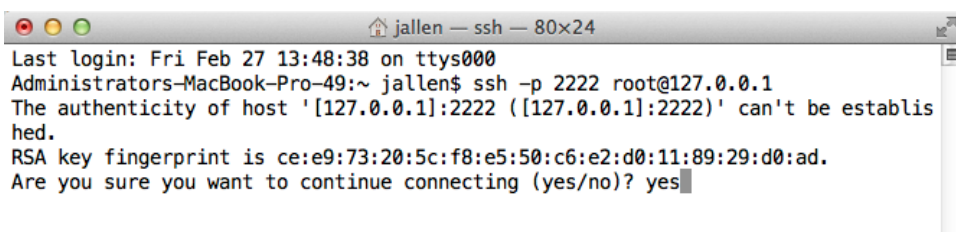
```
ssh -p 2222 root@127.0.0.1
```

Figure 0.6: Log into the Sandbox



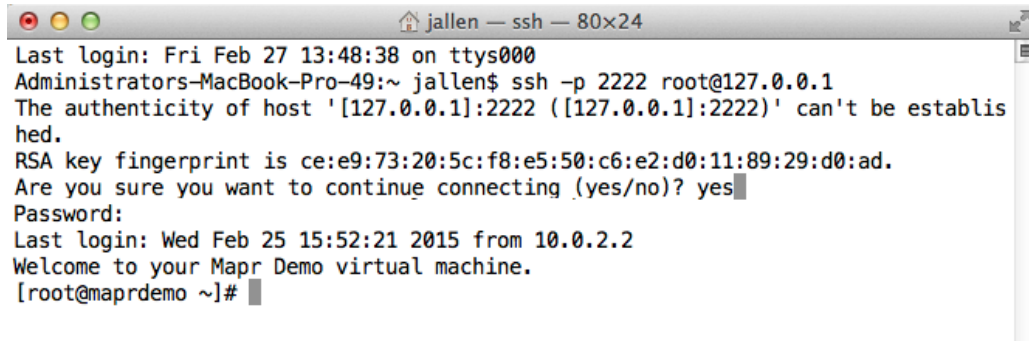
4. Accept the RSA key fingerprint

Figure 0.7: RSA Fingerprint



5. Complete the SSH login with the password 'mapr'

Figure 0.8: Logged into Sandbox



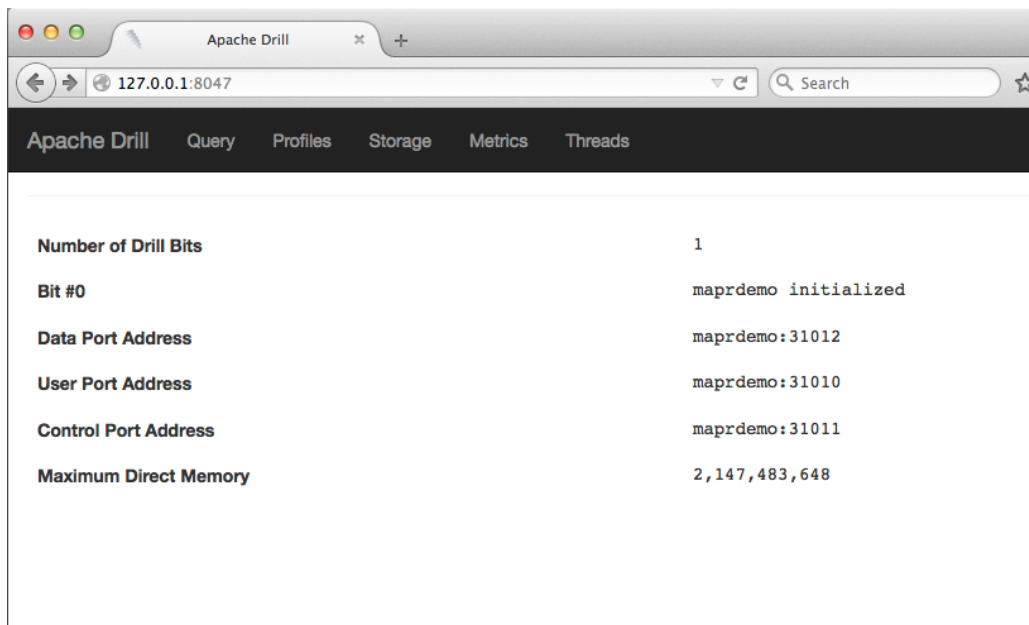
```

jallen — ssh — 80x24
Last login: Fri Feb 27 13:48:38 on ttys000
Administrators-MacBook-Pro-49:~ jallen$ ssh -p 2222 root@127.0.0.1
The authenticity of host '[127.0.0.1]:2222 ([127.0.0.1]:2222)' can't be established.
RSA key fingerprint is ce:e9:73:20:5c:f8:e5:50:c6:e2:d0:11:89:29:d0:ad.
Are you sure you want to continue connecting (yes/no)? yes
Password:
Last login: Wed Feb 25 15:52:21 2015 from 10.0.2.2
Welcome to your MapR Demo virtual machine.
[root@maprdemo ~]#

```

6. Access the Drill web interface by typing the IP address of the Drill sandbox into your browser, using port 8047.

Figure 0.9: Drill web interface



Conclusion

The MapR Sandbox is a convenient way to get experience with Hadoop. Most of the lab exercises in classes and courses available on MapR Academy are compatible with the MapR Sandbox. If you have any further questions about the MapR Sandbox, visit the MapR documentation:

Lesson 1: SQL Queries with Drill

Lab Overview

The objective of this lab is to get familiar with the features of Drill. We will start by performing familiar SQL queries on a structured Hive table, then expanding into data types that are not supported by other SQL tools. Finally, we will begin to explore data using the Drill Explorer.

Exercise	Required
1.1: Perform familiar SQL queries on structured data	Yes
1.2: Perform familiar SQL queries on a range of data types	Yes
1.3: Explore data using the Drill Explorer	No

Lab Procedures

Lab 1.1: Perform familiar SQL queries on structured data

In this first lab, we will introduce the ease of using drill to query data. You will:

- Launch Drill in the command line and web interface
- Perform some SQL queries on structure Hive data

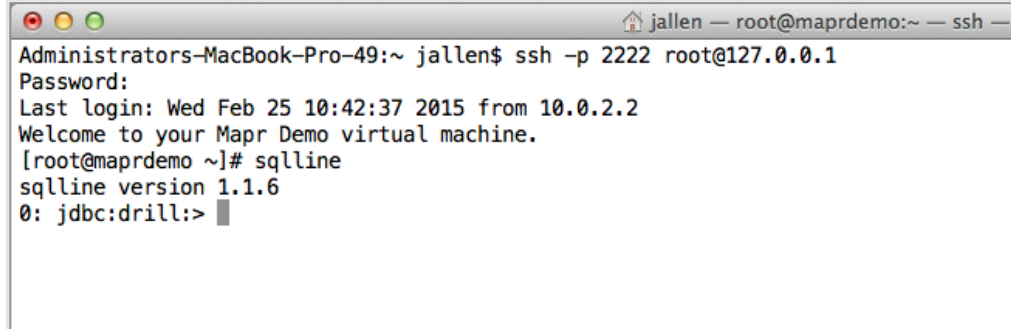
Launch Drill from the CLI

1. Power on your MapR Sandbox with Apache Drill. Log in as 'root' with the password 'mapr', and access the Drill web interface, as described in Lab 0.1 and Lab 0.2 above.
2. To launch Drill from the command line, at the CLI prompt type

```
sqlline
```

This command is an alias in the sandbox to launch Drill from the JDBC interface. As the versions of drill are updated, this alias will remain the same in the sandbox.

Figure 1.1: Launch Drill from the CLI



```
Administrators-MacBook-Pro-49:~ jallen$ ssh -p 2222 root@127.0.0.1
Password:
Last login: Wed Feb 25 10:42:37 2015 from 10.0.2.2
Welcome to your MapR Demo virtual machine.
[root@maprdemo ~]# sqlline
sqlline version 1.1.6
0: jdbc:drill:> █
```

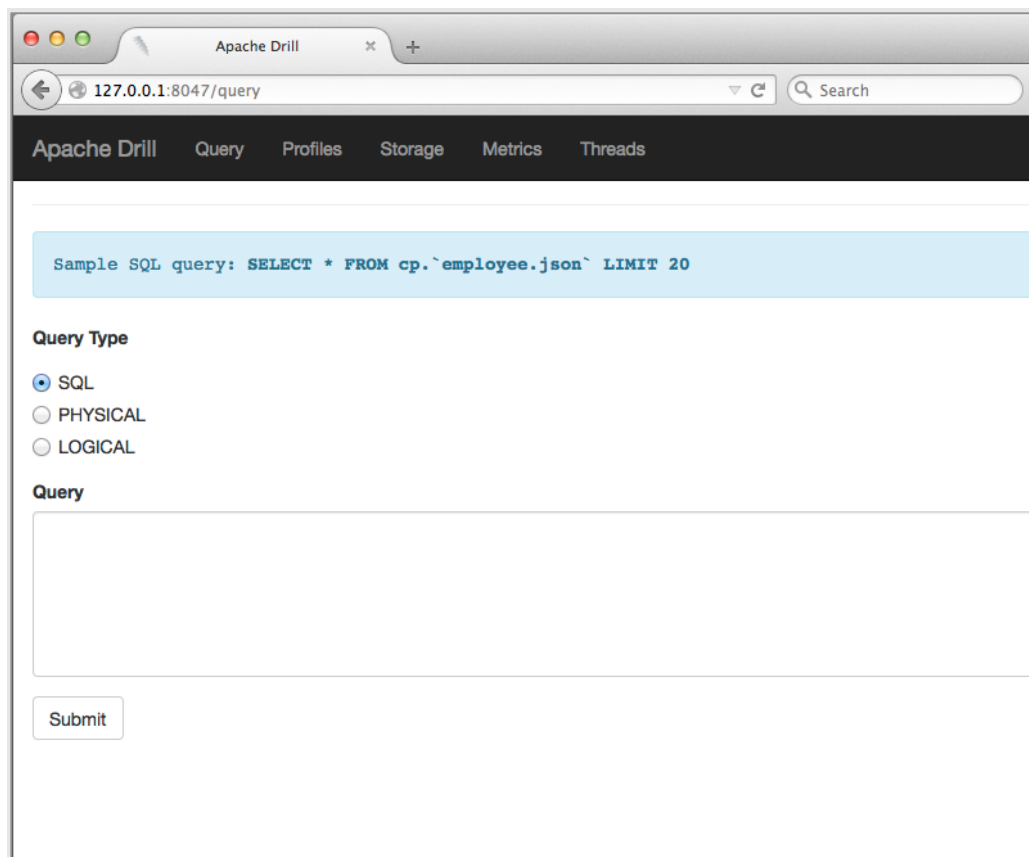
Launch Drill from the web interface

1. In a new browser window, type

```
<IP address of your Drill sandbox>:8047
```

2. Select the Query tab in the top navigation.

Figure 1.2: Launch the Drill Web Interface



Perform Queries on a Structured Hive Table

1. Type the following query at the CLI prompt, or into the Query window of the web interface.

IMPORTANT! The trailing ';' must be excluded from the query in the web interface. A query ending in a ';' will cause an error in the web interface.

```
SELECT `month`, SUM(order_total) as sales FROM hive.orders GROUP
BY `month` ORDER BY sales desc;
```

Figure 1.3: Query 1 - CLI

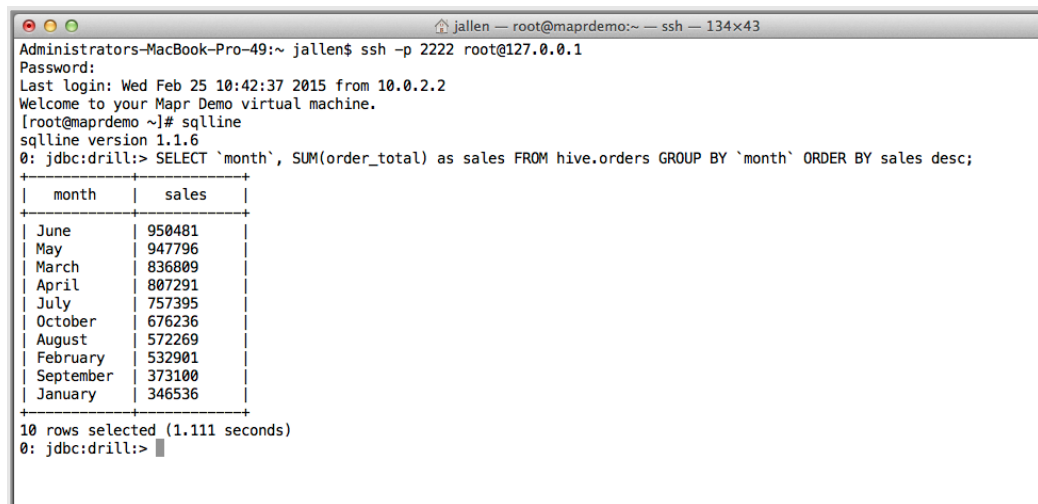


Figure 1.4: Query 1 - Web Interface

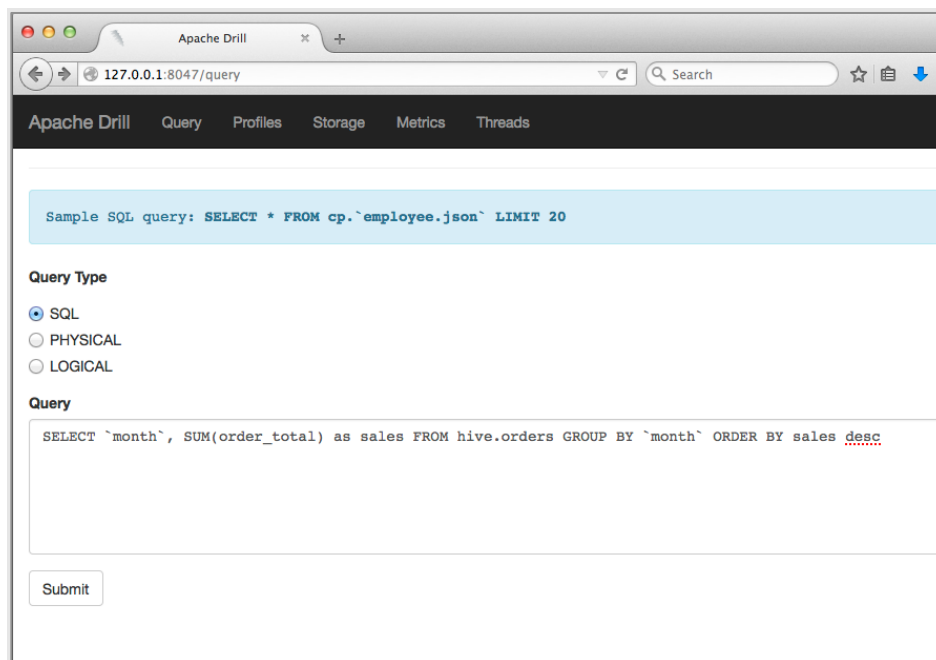
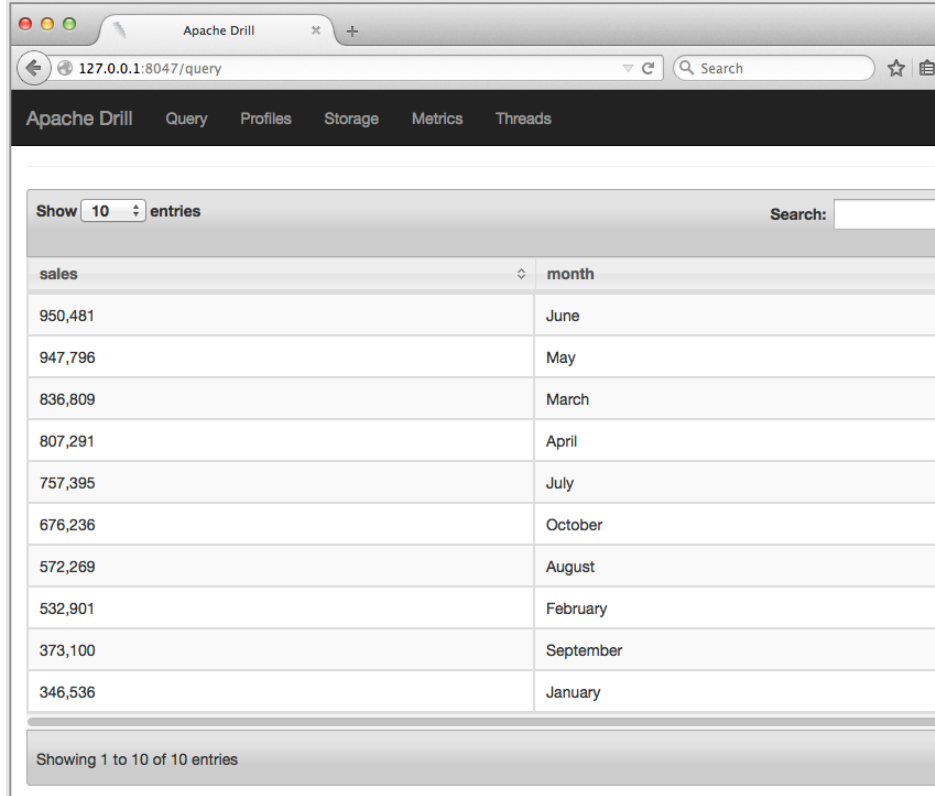


Figure 1.5: Query 1 Results - Web Interface



sales	month
950,481	June
947,796	May
836,809	March
807,291	April
757,395	July
676,236	October
572,269	August
532,901	February
373,100	September
346,536	January

2. Type the following query at the CLI prompt, and into the Query window of the web interface.

```
SELECT `month`, `state`, SUM(order_total) as sales FROM
hive.orders WHERE `month`='June' GROUP BY `month`, `state` ORDER
BY sales desc;
```

3. Type the following query at the CLI prompt, and into the Query window of the web interface.

```
SELECT `prod_id`, SUM(order_total) as sales FROM hive.orders
GROUP BY `prod_id` ORDER BY 2 desc limit 10;
```

Questions and Continuation

You just performed a series of ANSI SQL queries on a Hive table, using Drill.

What did you learn about the data?

List 3 additional queries you can make that will tell you more? Why are they helpful?

What did you learn about how Drill is able to query structured data with standard ANSI SQL?



Lab 1.2: Perform familiar SQL queries on a range of data types

In the second lab, you will continue what you learned in the first set of exercises. You will:

- A semi-structured HBase table
- A semi-structured JSON file that includes nested data
- A join on all three of these previous data sources

Query Semi-Structured HBase Table

1. Launch Drill with either the CLI or Web interface
2. What data is stored in the HBase table, “customers?”

3. Write SQL to query the HBase table and find out how many customers we have of each loyalty level, in each state. Refer to the Appendix for help, if needed.

Query Semi-Structured JSON File

1. Launch Drill with either the CLI or Web interface
2. What data is stored in the JSON click data?

3. Write SQL to query the JSON click data to determine how many times in April 2014 a customer looked at a product, but did not purchase it.

A copy of the click data saved in parquet format is linked in the Course Materials pdf.

4. Download this file.
5. Does it contain the same data as the JSON file? If not, how is it different?
6. Write the same query, this time using the parquet file on your local drive.

Join Structured and Semi-Structured Data Sources

1. Launch Drill with either the CLI or Web interface
2. What data is shared across our different data sources.



Write the SQL to query the HBase 'customers' table, the Hive 'products' table and the JSON 'clicks' file to see what products were viewed by not purchased by a customer from January 1, 2014 through March 31, 2014. Refer to the last query in lesson 1 for help.

Questions and Continuation

You just performed a series of ANSI SQL queries on a variety of different data sources, simple, complex, structured and semi-structured.

Can you query these data sources using ANSI SQL with Impala? Hive? Any other SQL tool?

What did you learn about using Drill to query different sources of data? What about data from locations other than a Hadoop cluster?

Write 3 more SQL queries, using these different data sources. Try joining data on the Hadoop cluster with data on your local drive. What other data sources do you have on your local drive that you can use? What about data from cloud storage?

Lab 1.3: Explore Data with Drill

In the third lab, you will get introduced to the concept of exploring data. The Drill Explorer is a Windows based GUI that connects to Drill through the ODBC interface. We used the Drill Explorer to show these concepts in the course material. You can use either the CLI or Drill Explorer to perform the exercises in this lab. You will:

- Use Drill to determine available data sources
- Discover the structure of these data sources
- Preview the data contents

Determine available data sources

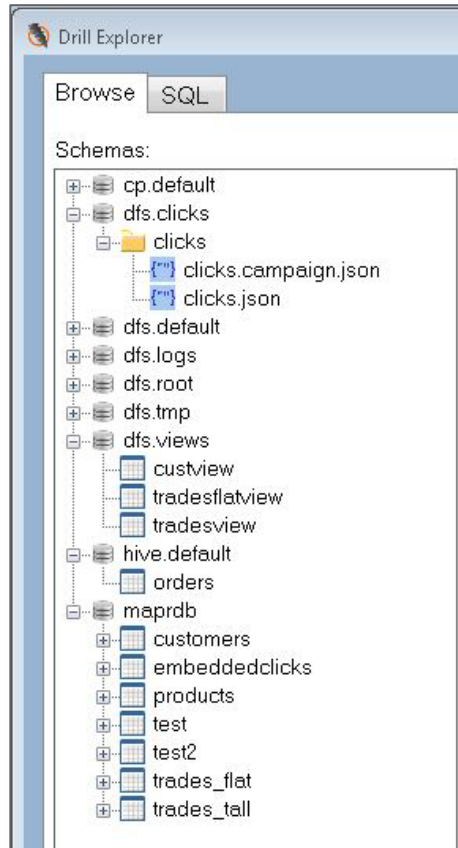
1. Launch Drill with either the CLI or open the Drill Explorer GUI. You can learn about how to connect to the ODBC driver on windows at:

<https://cwiki.apache.org/confluence/display/DRILL/Using+the+MapR+ODBC+Driver+on+Windows>

2. View the data sources available using Drill



Figure 3.1: Data Sources in Drill Explorer



To view the list of available data sources from the CLI, type:

```
SHOW DATABASES;
```

Figure 3.2: Data Sources in the Drill CLI

```
Welcome to your MapR Demo virtual machine.
[root@maprdemo ~]# sqlline
sqlline version 1.1.6
0: jdbc:drill:> SHOW DATABASES;

+-----+
| SCHEMA_NAME |
+-----+
| hive.default |
| dfs.default  |
| dfs.logs     |
| dfs.root     |
| dfs.views    |
| dfs.clicks   |
| dfs.tmp      |
| sys          |
| maprdb       |
| cp.default   |
| INFORMATION_SCHEMA |
+-----+
11 rows selected (2.237 seconds)
0: jdbc:drill:>
```



3. In the GUI, you can expand each data source to see the table and files available. In the CLI, select one of the data sources

```
use hive;
```

and then see the data tables and files in that source:

```
SHOW TABLES;
```

Figure 3.3: Hive tables in Drill CLI

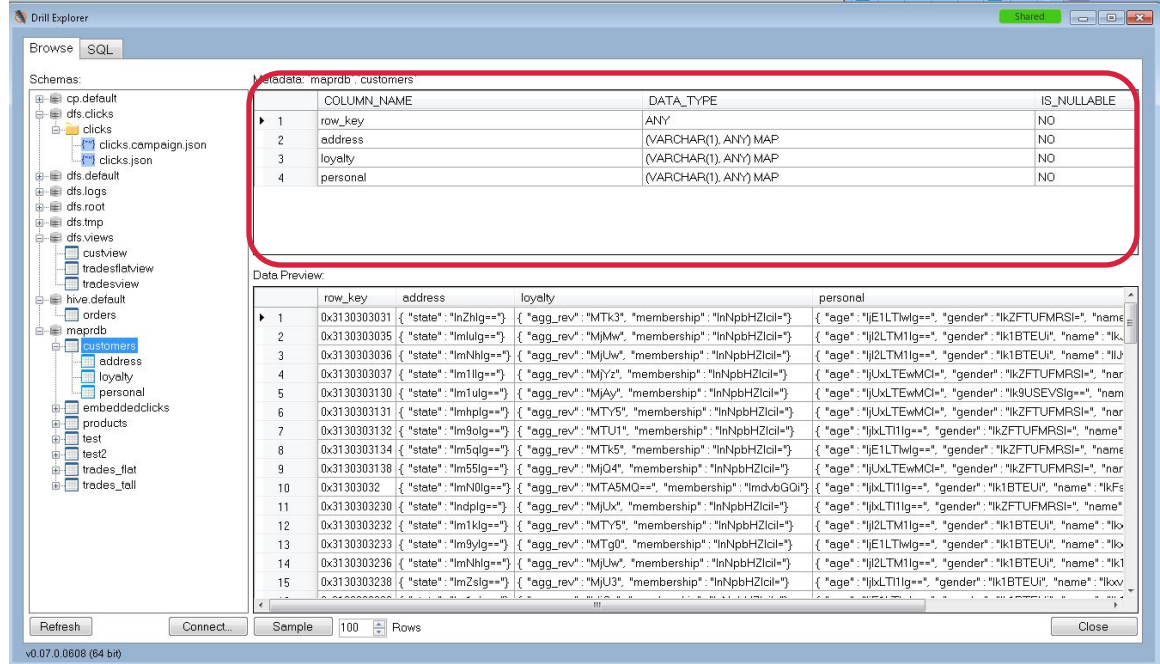
```
0: jdbc:drill:> use hive;
+-----+-----+
|      ok      | summary |
+-----+-----+
| true         | Default schema changed to 'hive' |
+-----+-----+
1 row selected (0.122 seconds)
0: jdbc:drill:> SHOW TABLES;
+-----+-----+
| TABLE_SCHEMA | TABLE_NAME |
+-----+-----+
| hive.default  | orders      |
| hive.default  | products    |
+-----+-----+
2 rows selected (2.548 seconds)
0: jdbc:drill:> █
```

Discover the structure of data

1. Launch Drill with either the CLI or open the Drill Explorer GUI.
2. Select one of the data sources in the GUI to see its schema in the 'Metadata' window



Figure 3.4: HBase Schema in the Drill Explorer



Metadata: maprdb: customers

	COLUMN_NAME	DATA_TYPE	IS_NULLABLE
1	row_key	ANY	NO
2	address	(VARCHAR(1), ANY) MAP	NO
3	loyalty	(VARCHAR(1), ANY) MAP	NO
4	personal	(VARCHAR(1), ANY) MAP	NO

Data Preview:

	row_key	address	loyalty	personal
1	0x3130303031	{ "state": "InZhlg==" }	{ "agg_rev": "MTk3", "membership": "InNpbHZicli=" }	{ "age": "JiE1LTWlg==", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
2	0x3130303035	{ "state": "InIulig==" }	{ "agg_rev": "MjMw", "membership": "InNpbHZicli=" }	{ "age": "Ji2LTm1lg==", "gender": "Ik1BTEUI", "name": "Ik1BTEUI" }
3	0x3130303036	{ "state": "InNhlgl==" }	{ "agg_rev": "MjUw", "membership": "InNpbHZicli=" }	{ "age": "Ji2LTm1lg==", "gender": "Ik1BTEUI", "name": "Ik1BTEUI" }
4	0x3130303037	{ "state": "InIilgl==" }	{ "agg_rev": "MjYz", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
5	0x3130303130	{ "state": "InIulgl==" }	{ "agg_rev": "MjYz", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
6	0x3130303131	{ "state": "InIulgl==" }	{ "agg_rev": "MTY5", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
7	0x3130303132	{ "state": "InIulgl==" }	{ "agg_rev": "MTU1", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
8	0x3130303134	{ "state": "InIulgl==" }	{ "agg_rev": "MTk5", "membership": "InNpbHZicli=" }	{ "age": "JiE1LTWlg==", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
9	0x3130303138	{ "state": "InIulgl==" }	{ "agg_rev": "MjQ4", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
10	0x31303032	{ "state": "InN0lg==" }	{ "agg_rev": "MTA5MO==", "membership": "InIulgl==" }	{ "age": "JiXLTewMCI=", "gender": "Ik1BTEUI", "name": "Ik1BTEUI" }
11	0x3130303230	{ "state": "InIulgl==" }	{ "agg_rev": "MjUw", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
12	0x3130303232	{ "state": "InIulgl==" }	{ "agg_rev": "MTY5", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
13	0x3130303233	{ "state": "InIulgl==" }	{ "agg_rev": "MTg0", "membership": "InNpbHZicli=" }	{ "age": "JiE1LTWlg==", "gender": "Ik1BTEUI", "name": "Ik1BTEUI" }
14	0x3130303236	{ "state": "InIulgl==" }	{ "agg_rev": "MjUw", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEUI", "name": "Ik1BTEUI" }
15	0x3130303238	{ "state": "InIulgl==" }	{ "agg_rev": "MjU3", "membership": "InNpbHZicli=" }	{ "age": "JiXLTewMCI=", "gender": "Ik1BTEUI", "name": "Ik1BTEUI" }

Or switch to the SCHEMA_INFORMATION

Figure 3.5: Show Schema Tables in the Drill CLI

```
0: jdbc:drill:> USE INFORMATION_SCHEMA;
+-----+-----+
| ok | summary |
+-----+-----+
| true | Default schema changed to 'INFORMATION_SCHEMA' |
+-----+-----+
1 row selected (0.076 seconds)
0: jdbc:drill:> SHOW TABLES;
+-----+-----+
| TABLE_SCHEMA | TABLE_NAME |
+-----+-----+
| INFORMATION_SCHEMA | VIEWS |
| INFORMATION_SCHEMA | COLUMNS |
| INFORMATION_SCHEMA | TABLES |
| INFORMATION_SCHEMA | CATALOGS |
| INFORMATION_SCHEMA | SCHEMATA |
+-----+-----+
5 rows selected (0.658 seconds)
0: jdbc:drill:>
```

and then query these tables to learn about the data source schema.



Figure 3.6: Data Structure in the Drill CLI

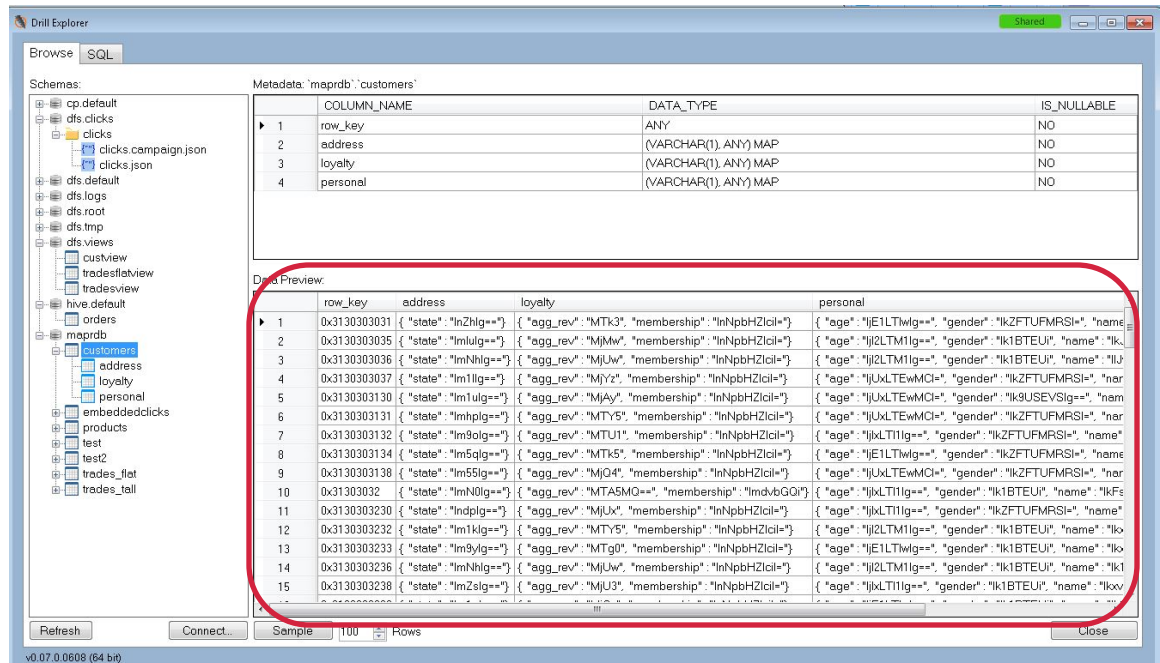
```
0: jdbc:drill:> select * from COLUMNS;
```

TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	ORDINAL_POSITION	IS_NULLABLE	DATA_TYPE	CHARACTER_MAXIMUM
DRILL	hive.default	orders	order_id	0	YES	BIGINT	-1
DRILL	hive.default	orders	month	1	YES	VARCHAR	65535
DRILL	hive.default	orders	cust_id	2	YES	BIGINT	-1
DRILL	hive.default	orders	state	3	YES	VARCHAR	65535
DRILL	hive.default	orders	prod_id	4	YES	BIGINT	-1
DRILL	hive.default	orders	order_total	5	YES	INTEGER	-1
DRILL	hive.default	products	prod_id	0	YES	BIGINT	-1
DRILL	hive.default	products	name	1	YES	VARCHAR	65535
DRILL	hive.default	products	category	2	YES	VARCHAR	65535
DRILL	hive.default	products	price	3	YES	INTEGER	-1
DRILL	dfs.clicks	custview	cust_id	0	NO	BIGINT	-1
DRILL	dfs.clicks	custview	name	1	NO	ANY	-1
DRILL	dfs.clicks	custview	gender	2	NO	ANY	-1
DRILL	dfs.clicks	custview	age	3	NO	ANY	-1
DRILL	dfs.clicks	custview	state	4	NO	ANY	-1
DRILL	dfs.clicks	custview	agg_rev	5	NO	DECIMAL	-1
DRILL	dfs.clicks	custview	membership	6	NO	ANY	-1
DRILL	sys	drillbits	host	0	NO	VARCHAR	1
DRILL	sys	drillbits	user_port	1	NO	INTEGER	-1

Preview data contents

1. Launch Drill with either the CLI or open the Drill Explorer GUI.
2. The Data Preview window in the GUI shows the data held in the table or file:

Figure 3.7: Data Preview in the Drill Explorer



Drill Explorer

Metadata: 'maprdb', 'customers'

	COLUMN_NAME	DATA_TYPE	IS_NULLABLE
1	row_key	ANY	NO
2	address	(VARCHAR(1), ANY) MAP	NO
3	loyalty	(VARCHAR(1), ANY) MAP	NO
4	personal	(VARCHAR(1), ANY) MAP	NO

Data Preview:

	row_key	address	loyalty	personal
1	0x3130303031	{ "state": "InZhlq==" }	{ "agg_rev": "MTk3", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "IkZFTUFMRSl=", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
2	0x3130303035	{ "state": "Im1ulg==" }	{ "agg_rev": "MjMw", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
3	0x3130303036	{ "state": "Im1Nhlq==" }	{ "agg_rev": "MjUw", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
4	0x3130303037	{ "state": "Im1llg==" }	{ "agg_rev": "MjYz", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
5	0x3130303130	{ "state": "Im1ulg==" }	{ "agg_rev": "MjYy", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
6	0x3130303131	{ "state": "Im1hplg==" }	{ "agg_rev": "MTY5", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
7	0x3130303132	{ "state": "Im19qlg==" }	{ "agg_rev": "MTU1", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
8	0x3130303134	{ "state": "Im15qlg==" }	{ "agg_rev": "MTk5", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
9	0x3130303136	{ "state": "Im155lg==" }	{ "agg_rev": "MjQ4", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
10	0x31303032	{ "state": "Im1Ndlg==" }	{ "agg_rev": "MTASMQ==", "membership": "ImdvbGGQ" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
11	0x3130303230	{ "state": "Im1ndplg==" }	{ "agg_rev": "MjUx", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
12	0x3130303232	{ "state": "Im1klg==" }	{ "agg_rev": "MTY5", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
13	0x3130303233	{ "state": "Im19ylg==" }	{ "agg_rev": "MTg0", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
14	0x3130303236	{ "state": "Im1Nhlq==" }	{ "agg_rev": "MjUw", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }
15	0x3130303238	{ "state": "Im1Zslg==" }	{ "agg_rev": "MjU3", "membership": "InNpbHZlcl=" }	{ "age": "jE1LTWlg==", "gender": "Ik1BTEU", "name": "Ik1BTEU", "membership": "InNpbHZlcl=" }

Refresh Connect... Sample 100 Rows Close

v0.07.0.0608 (64 bit)



or use SQL commands on the individual tables in the CLI to preview their content.

Figure 3.8: Hive Data Preview in the CLI

```
0: jdbc:drill:> select * from orders;
```

order_id	month	cust_id	state	prod_id	order_total
67212	June	10001	ca	909	13
70302	June	10004	ga	420	11
69090	June	10011	fl	44	76
68834	June	10012	ar	0	81
71220	June	10018	az	411	24
61287	June	1001	nj	104	134
68553	June	10021	ca	117	67
68109	June	10022	tx	337	10
68526	June	10025	mi	11	63
69362	June	10028	tx	430	65
68624	June	10030	fl	808	51
67250	June	10030	ma	449	63
68390	June	10031	va	7	42
68299	June	10038	ny	451	66
69995	June	10046	la	683	59
69989	June	10047	ny	19	42
70069	June	10047	oh	49	77
68028	June	10048	wi	296	51
70141	June	10049	il	113	67
68787	June	10057	fl	461	83
68226	June	10058	oh	198	79
63192	June	1005	ca	444	58
67606	June	10066	pa	507	77
71908	June	10068	pa	207	82
60583	June	1006	ut	30	111
69316	June	10075	fl	71	65
71555	June	10079	ky	433	32
71348	June	10079	tx	132	17
63524	June	1007	az	40	39
61830	June	1007	fl	919	107
70956	June	10080	tx	921	60
69942	June	10085	tx	261	44
68339	June	10087	oh	727	10
62844	June	1008	mo	564	129
68209	June	10090	nc	181	35
71168	June	10093	ct	300	91

Figure 3.9: HBase Data Preview in the CLI

```
0: jdbc:drill:> select * from customers;
```

row_key	address	loyalty	personal
006af0b6ad	{ "state": "InZiIg==" }	{ "agg_rev": "MTK3", "membership": "InNpbhZLciI==" }	{ "age": "J1E1LTiIg==", "gender": "IKZFTUPMRSI=", "name": "IK1bVcnJpbmUgTWJa" }
0070dc8b99	{ "state": "InNhiIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1J2LTiIg==", "gender": "IK1BTEUI=", "name": "IK3yaXN0W55iF8hcmS1" }
002bde0042	{ "state": "InNhiIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1J2LTiIg==", "gender": "IK1BTEUI=", "name": "IK1Jvc2UgtG9rZXk1" }
007be3830d	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IKZFTUPMRSI=", "name": "IK1pbhWzIEZv2dXlc" }
0074fe5d85	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK9USEVSIg==", "name": "IK1d1aWksXJtbyBbLb" }
0067ae02c4	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IKZFTUPMRSI=", "name": "IK1b021hcyBNVnZa" }
0065137c799	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IKZFTUPMRSI=", "name": "IK1b05bnRoaWg0mFyc" }
0024ec4987	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IKZFTUPMRSI=", "name": "IK1b1w1c1bWVdyY" }
0014c75101	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IKZFTUPMRSI=", "name": "IK1pY2hcn0g0nJ5Y" }
0036cf2353	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0059183a41	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IKZFTUPMRSI=", "name": "IK1b1w1c1bWVdyY" }
004598af19	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1J2LTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
002b0f6b01	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
00147611bd	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1J2LTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
00538699c9	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0029da85f6	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
004990eef1	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
004a3735f2	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
006eb6c8ee	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
002241ba0b	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0091d39bf1	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
005d9a2f45	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
006a758b15	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
004afe464d	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
007b0b9f11	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
005388eccc	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
00427aa4cc	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
001d96f2d4	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
006a11f44d	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0040be21a2	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0029233a33	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
004c8c7d66	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
001b7cd0cd	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0051809b0c	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0027497e	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
00e4975f1	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0019b1a8bb	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
005eb04ada	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
008025ec7	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
00539e5eac	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0031f1c826	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0047fe4f8	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0043447b5	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
00474f0cb	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }
0044f47a00	{ "state": "InMlIg==" }	{ "agg_rev": "MTK4", "membership": "InNpbhZLciI==" }	{ "age": "J1JxLTiIg==", "gender": "IK1BTEUI=", "name": "IK1b1w1c1bWVdyY" }



Questions and Continuation

You just used Drill to look at data before performing any SQL queries on the data.

How can exploring data be helpful to us when we encounter unknown data sources? What about if the data contains thousands, or even millions or rows?

Notice that the HBase content is stored in byte format, and unreadable in its current form. How can we use what we've already done in these lab exercises to know how to cast this data to make it readable?



Appendix A: Solutions

Lab 1.2 Solutions

HBase

1. The HBase table contains: HBASE_ROW_KEY, address:state(VARCHAR), loyalty:agg_rev(BIG INT), membership(VARCHAR), personal:name(VARCHAR), age(VARCHAR), gender(VARCHAR)

2. Start with:

```
select cast (t.loyalty.membership as varchar(20)) as loyalty,  
cast (t.address.state as varchar(20)) as state from customers t;
```

to get the full list of loyalty level by state.

JSON

1. A sample of the JSON data is shown below. The first part of each (x:y) pair is the meta data. The second entry is the data content.

```
{"trans_id":31920,"date":"2014-04-  
26","time":"12:17:12","user_info":{"cust_id":22526,"device":"IOS5","sta  
te":"il"},"trans_info":{"prod_id":[174,2],"purch_flag":"false"}}
```

