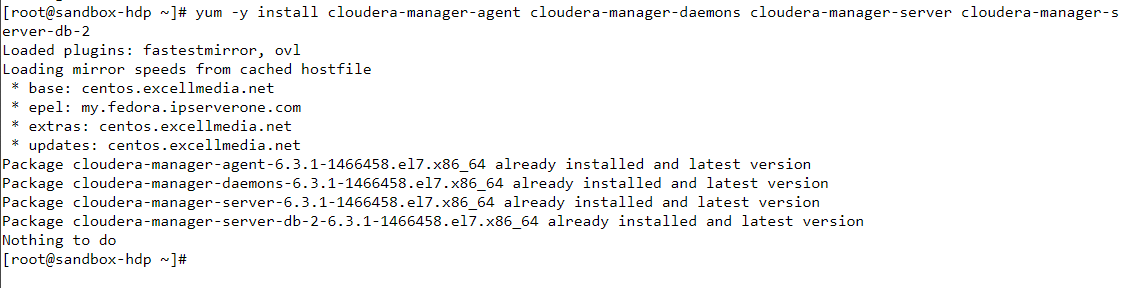
**Introduction to Big Data - Task 1**

**Cloudera Manager:**

According to Cloudera, Cloudera Manager is the best way to **install**, **configure**, **manage**, and **monitor** the Hadoop stack.

It provides:

1. Automated deployment and configuration
2. Customizable monitoring and reporting
3. Effortless robust troubleshooting
4. Zero – Downtime maintenance

****

**Components of Cloudera Manager:**

1. **Cloudera Manager Agent:**

The Cloudera Manager is a Cloudera Manager Component that **works with the Cloudera Manager Server to manage the process that map to role instances**. It is responsible for **Starting and Stopping processes**, **unpacking configurations**, **triggering installations**, and **monitoring all hosts in a cluster**.

1. **Starting Service of Cloudera Manager Agent:**



In the above image, I have stated the Cloudera Manager Agent

1. **Stopping Cloudera Manager Agent:**



In the above image, I have stopped the Cloudera Manager Agent

1. **Cloudera Manager Daemons:**

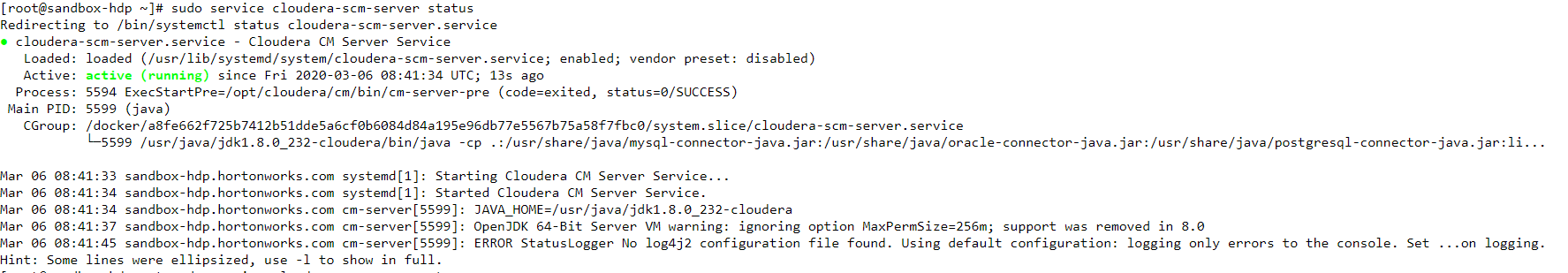
Daemons are the services that are runs on the Cloudera Manager.

1. **Cloudera Manager Server:**
2. **Starting Cloudera Manager Server:**



In the above image, I have stated the Cloudera Manager Server

1. **Status of Cloudera Manager Server:**



In the above image, Cloudera Manager Server is currently running.

1. **Stopping Cloudera Manager Server:**



In the above image, I have stopped the Cloudera Manager Server

**Some Services of Cloudera Manager:**

1. **Metric Collection:**

To do its monitoring, **Cloudera Manager collects metrics**. Metrics are simply **numeric values like name**, **timestamp**, etc. Most of the **services are collect by the agent**. Then **Agent forward them to service monitor**.

Special Metrices are collected by the service monitor.

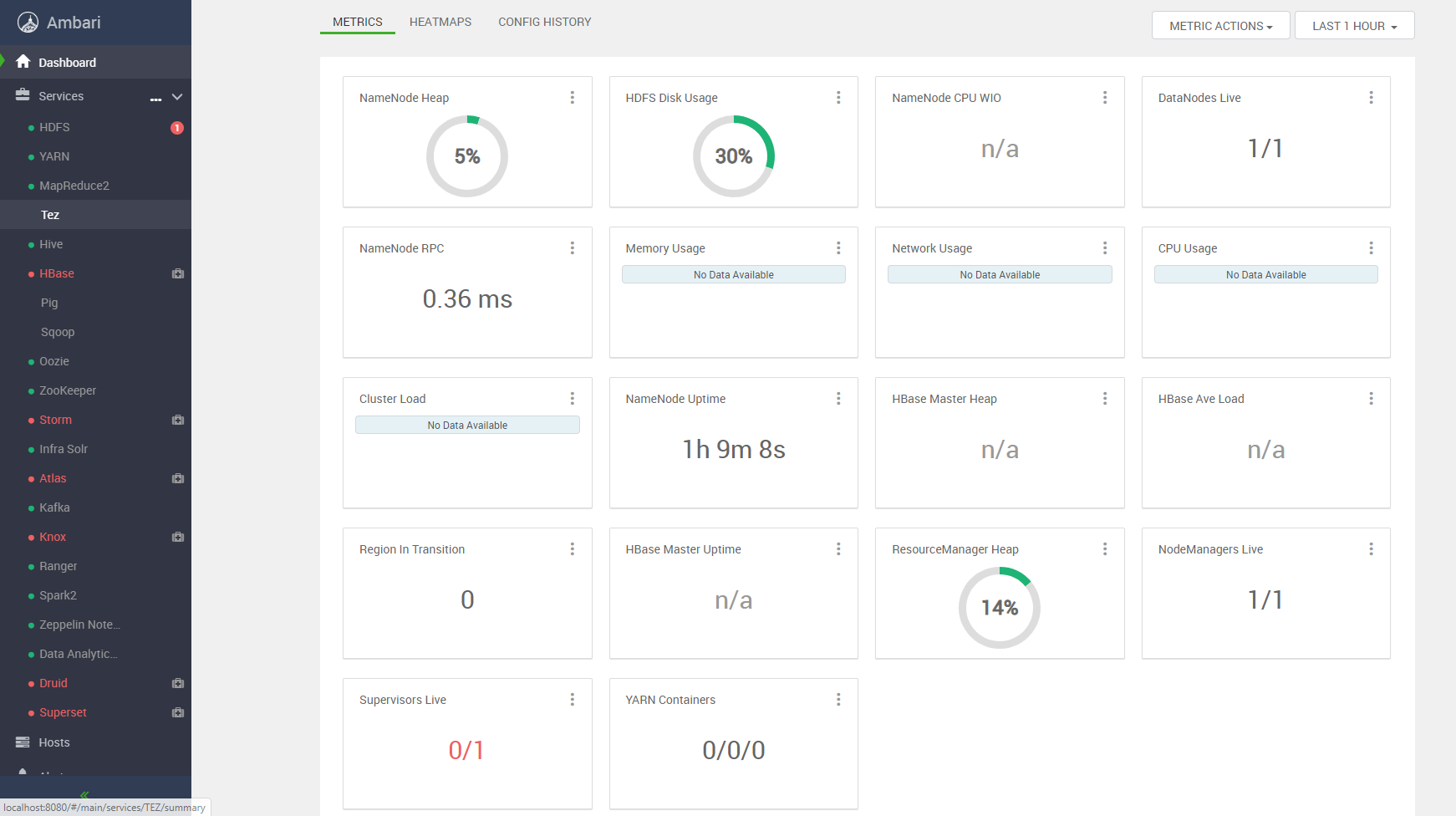
1. **Health Check:**

Service monitor continuously **checks the health of every node**. A simple one **asks whether there’s enough disk space in every Name Node data directory**. A more complicated health check may **evaluate when the last checkpoint for HDFS** was compared to a threshold or whether a Data Node is connected to a Name Node.

**Ambari UI:**

The Apache Ambari project is aimed at making Hadoop management simpler by developing software for provisioning, managing, securing and monitoring Apache Hadoop clusters.

It provides an intuitive, easy-to-use Hadoop management web UI backed by its RESTful APIs.



In the above image, Ambari UI is in currently running State.

**Components of Ambari UI:**

**Views in Apache Ambari:**

1. **Tez View:** The Tez View **helps you understand** and **optimize your cluster resource usage**. Using the view, you can optimize and accelerate individual SQL queries or Pig jobs to get the best performance in a multi-tenant Hadoop environment.
2. **Hive View:** Hive **View allows the user to write & execute SQL queries on the cluster**. It shows the history of all Hive queries executed on the cluster whether run from Hive view or another source such as JDBC/ODBC or CLI. It also provides graphical view of the query execution plan. This helps the user debug the query for correctness and for tuning the performance. It integrates Tez View that allows the user to debug any Tez job, including monitoring the progress of a job (whether from Hive or Pig) while it is running.
3. **Pig View:** Pig View is like the Hive View. **It allows writing and running a Pig script.** It has support for saving scripts and loading and using existing UDFs in scripts.
4. **Capacity Scheduler View:** Capacity Scheduler View **helps a Hadoop operator setup YARN workload management easily to enable multi-tenant and multi-workload processing**. This view provisions cluster resources by **creating and managing YARN queues**.
5. **Files View:** Files View allows the user to **manage**, **browse** and **upload files** **and** **folders in HDFS**.

**Apache Hive:**

Apache Hive provides SQL interface to **query data stored in various databases and files systems that integrate with Hadoop**. Hive enables analysts familiar with SQL to run queries on large volumes of data.

Hive has three main functions:

1. **data summarization**
2. **query data**
3. **data analysis.**

Hive provides tools that enable easy data extraction, transformation and loading.

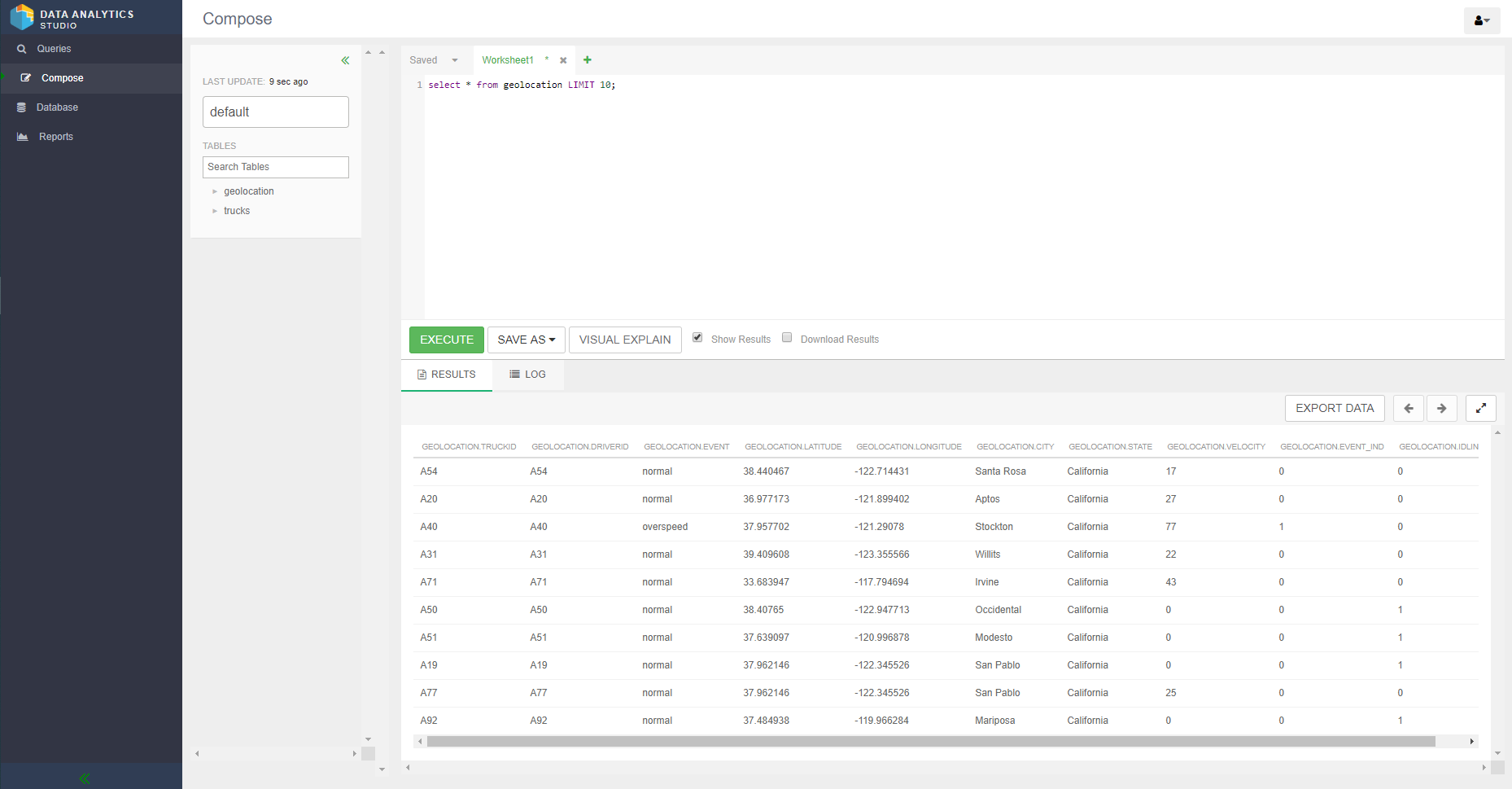
**Hive vectorization features:**

**Vectorization** is a type of **parallel processing**. It enables more computer hardware to be devoted to performing the computation, so the computation is done **faster**.

**Data Analytics Studio:**

There are 4 tabs to interact with Data Analytics Studio:

1. **Queries**: This view allows you to search previously executed SQL queries. You can also see commands each user issues.
2. **Compose**: From this view you can execute SQL queries and observe their output. Additionally, visually inspect the results of your queries and download them as csv files.
3. **Database**: Database allows you to add new Databases and Tables. Furthermore, this view grants you access to advanced information about your databases.
4. **Reports**: This view allows you keep track of Read and Write operations and shows you a Join Report of your tables.



In the above image, I have composed the query in Data Analytics Studio.

**Apache ORC:**

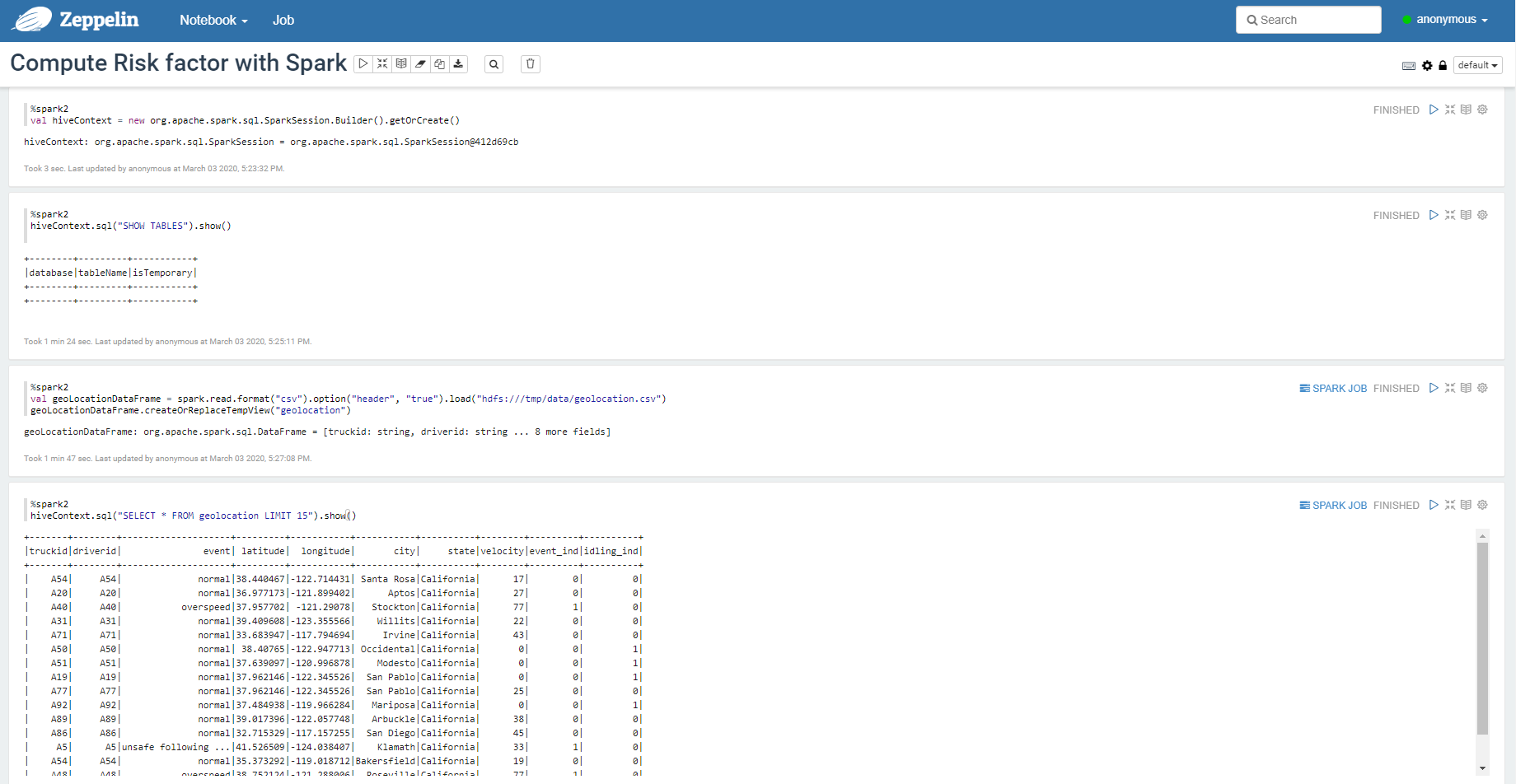
Apache ORC is a **fast-columnar storage file format for Hadoop workloads**.

The Optimized Row Columnar file format provides a highly efficient way to store Hive data. It was designed to overcome limitations of the other Hive file formats. Using ORC files improves performance when Hive is reading, writing, and processing data.

**Apache Zeppelin:**

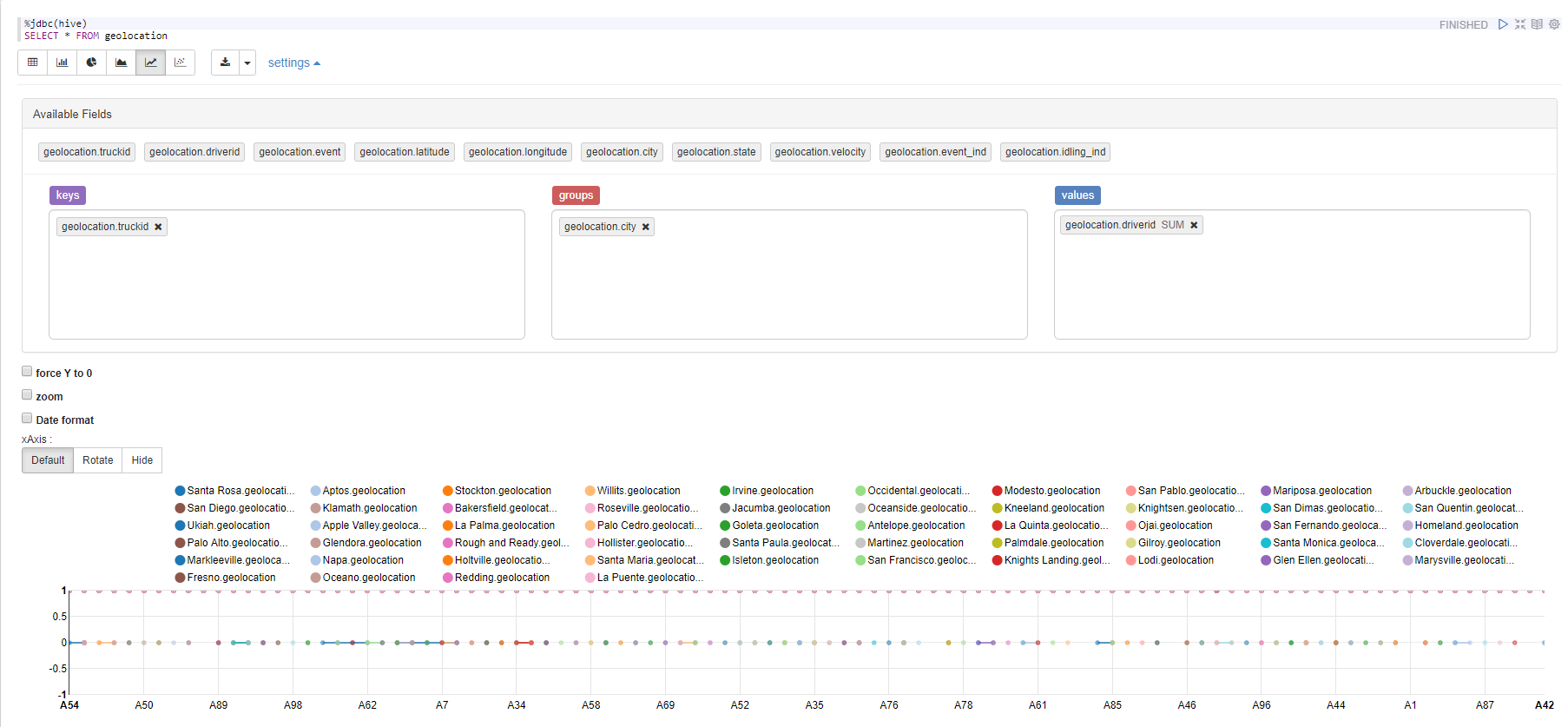
Apache Zeppelin provides a **powerful web-based notebook platform for data analysis and discovery**. Behind the scenes it supports Spark distributed contexts as well as other language bindings on top of Spark.

**Apache Spark with HDP Sandbox:**



I the above image, we can load, select data in Spark.

**Apache Spark Graphs in Zeppelin:**



I the above image, I have drawn graphs using zeppelin.