I previously discussed how Spark and Flink could be used together. In this discussion I will be explaining how to use Spark and Cassandra. Apache Cassandra is one of the NoSQL databases I used in Applied Database Technologies class. This database is an open source, distributed, NoSQL database that presents a partitioned wide column storage model with eventually consistent semantics.

It has some limitations in terms of aggregate operations and data analysis and moving large amount of data. We can simply capture/store the data and then Spark worker nodes are co-located with Cassandra to perform data processing. The Spark worker understands how Cassandra distributes the data and reads only from the local node.

Afterwards, the integrations with Spark and Cassandra allows the map-reduce style behavior where some operations can and will be executed on local nodes, while others require data to be exchanged between the Spark workers. Finally, the results are written back in different locations like in RDBMS, Databases (Cassandra), local disks, HDFS (Hadoop) etc.

We can also see Homework 6 as an example, the csv data was stored in Apache HBase and then Spark was called to query through the HBase table. The role of this integration in big data application allows machine learning operations to execute easily even when combines with other programming languages like Scala. It also helps with the data processing tasks and lets us have storage location flexibility.

Apache Spark’s resilient distributed datasets (RDDs) provide fault-tolerant in-memory storage that makes it faster and more efficient than other big data processing frameworks. If we look to Google Cloud, we have a serverless Spark. A serverless Spark helps to write Spark applications and pipelines that auto scale without any manual tunning (this is cost effective). You can also run Spark from BigQuery, Dataplex etc without custom integrations and explorations. Spark in AWS (Amazon EMR) helps you simplify the deployment and management of Spark clusters. We also have Cloud ML integrated in Google Cloud ML Engine, AWS, Azure. By storing datasets in-memory during a job, Spark has great performance for iterative queries common in machine learning workloads. You can enhance Amazon SageMaker capabilities by connecting the notebook instance to an Apache Spark cluster running on Amazon EMR, with Amazon SageMaker Spark for easily training models and hosting models.