**Spark**

**Fundamentals of Databricks**

* When comparison has been made from the 1980's to till different kinds of architecture has been made from modern to traditional warehousing of project.
* During the late 1980’s, data warehouse architecture was introduced and rdbms and dbms were introduced in the 90's.
* In traditional database projects most of the sources will be structured databases like ERP and OLTP databases.
* Using ETL the databases are loaded into the OLAP warehouse.
* Where the BI+reports team will analyze the data and submit the report to business people.
* Here both the source and target of the database are structured data.
* In 2011 data lake came into picture mainly on on-premises big data combination or distributed file system HDFS hadoop and later spark has introduced.
* Database stores structured,semi-structured or unstructured data.
* Once data is available on data lake,ETL tool is available for loading. It is like a scala,java based project,but also having a spyspark project which is less when compared to others.
* After loading the data on ETL, reads the data in the data warehouse and converts it into structured format in the separate data warehouse.
* Traditional reporting teams are used here for providing reports.
* When advanced analytics has come into picture,data has been taken directly from a data lake,where it can store any type of data with unlimited storage as a plus point.
* Since it was having advantages and major disadvantages like metadata, an important thing comes to the analytics part which is missing.
* Then data lakehouse came into picture which is a combination of data bricks and delta.
* Data will be present on the databricks and delta will enable the database features in addition to the database.
* Doesn’t need any separate warehouse because it is getting both the database and data lake features.
* Delta converts the structured,semi structured,unstructured into analytics format.

**Session State in Spark**

* In Apache Spark, a "session" typically refers to a SparkSession, which is the entry point for programming with Spark.
* It helps you work with data by providing a unified interface for various Spark functionalities

**Difference Between Hadoop and Spark**

* HADOOP: Data is processed by batch processing /batch-oriented processing.Takes more lines of code, code can be written in Java only.
* Support Kerberos authentication architecture. (Always needs authentication when user needs to access the server)
* Write the data,read the data in disk.
* Spark:Data is processed in real streaming (real-time) and also supports batch processing.
* Fewer lines of code and can be written in Scala, python, or any other language also can use the OOPS concept.
* Supports YARN architecture: Yet another resource negotiator.
* Everything is done on RAM, load the data on memory whenever any action is performed, storage whole process done in RAM. Therefore it is very easy to preserve, compute and access the data quicker.

Apache spark

* The Spark follows the master-slave architecture. Its cluster consists of a single master and multiple slaves.
* Is an open-source data processing engine such as we have numpy, tensorflow, etc.
* Same as Apache Spark is a data processing engine where the data is processed in real-time in various clusters of computers with the help of programming constructors.
* Use programming languages such as Python, java, R, and scala.
* In the data domain use R or Python.
* API:Application Programming interface.

**Functions of spark**

* Query the data
* Analyze the data
* Transform the data

**3vs of big data**

* VOLUME: amount of data
* VELOCITY: speed of data processing
* VARIETY: number or type of data.

**API’S of Spark**

* RDD: (Resilient distributed dataset)
* Dataframe
* Dataset

**RDD: (Resilient distributed dataset)**

* It is a fundamental data structure in Apache Spark, representing an immutable, distributed collection of objects that can be processed in parallel across a cluster of machines. RDDs provide fault tolerance and enable parallel processing of data in a distributed computing environment.

**Types of RDD**

* **Transformation:**Operations that create a new RDD from an existing one.

1. In transformation they are lazy,i.e the results are not immmediately computed.

* **Action:**Compute a result based on RDD and either returned or saved on the external storage device.

1. They are eager,their result are immediately computed.

**Narrow Transformation**

* transformations can be executed in parallel, and Spark does not need to shuffle data between partitions.

**Wide Transformation**

* Wide transformations, such as groupByKey, may require shuffling data between partitions.

## **Directed Acyclic Graph (DAG)**

* The Directed Acyclic Graph (DAG) represents the flow of operations, optimizing their execution for better performance.
* Transformation Creation: When you apply a transformation on an RDD, Spark doesn't immediately execute it. Instead, it adds the transformation to its logical execution plan.
* DAG Construction: As you chain transformations, Spark constructs a DAG of stages. Each stage represents a set of transformations that can be executed in parallel.
* Action Triggers Execution: When an action is called (e.g., collect, count), Spark examines the DAG and determines the optimal physical execution plan.
* Task Execution: Spark divides the computation into tasks, which are the smallest unit of work. These tasks are then executed on the cluster.
* Data Movement (Shuffling): For wide transformations, where data needs to be redistributed, Spark performs a shuffle, which involves moving data across partitions.

•**Logical plan:** to give a high-level operation; modify the data using operations like filter union group etc.

•**Physical plan:** works to fulfill the requirement of a logical plan. how the data is getting partitioned and how the data is transformed based on shuffling after logical plan.