PySpark

Pyspark is an Apache Spark library. written in Python to run Python applications using Apache Spark capabilities.

It is a python API which is an analytical processing engine for large-scale powerful distributed data processing & machine learning application.

Apache Spark

- Open-source application.
- fast, flexible, easy to use, processing large-scale datasets.
- ~~singl~~ It can be runned on single node or multi-node machines.
- created to handle the limitations of MapReduce by doing in-memory processing.

PySpark feature

* In-memory computation
* Distributed processing using parallelize
* Fault-tolerant
* Immutable
* Cache & persistence
* Lazy evaluation
* Inbuild-optimization when using Dataframes.
* Supports ANSI-SQL.

Apache Kafka is an open-source distributed event streaming platform.

Advantages of PySpark

Applications running on PySpark are 100x faster

It process data from Hadoop HDFS, AWS S3 & many file systems.

It is used to process real-time data using ~~streaming~~ streaming & Kafka.

It has natively Machine learning & graph libraries.

Versions supported with Pyspark 3.5

Python - 3.8 & newer
Java - 8, 11 & 17 (versions prior to 8u371 has been deprecated)
Scala - 2.12 & 2.13 beyond.

PySpark Architecture

works on master-slave architecture

Pyspark Modules & packages.

PySpark RDD (pyspark.rdd)
PySpark Dataframe & SQL (pyspark.sql)
PySpark Streaming (pyspark.streaming)
PySpark MLlib (pyspark.ml, pyspark.mllib)
PySpark GraphFrames (GraphFrames)
PySpark Resource (pyspark.resource) new in PySpark 3.0.

Uploading file in databricks

catalog - default - tables - create table

upload file

create table in notebook.

How to read the data of a file in PySpark (local):

val filePath = "

Resilient Distributed Datasets

Create RDD

1). By using parallelize() function → function used to create an RDD from a list collection

```
from pyspark.sql import SparkSession

spark = SparkSession \
    .builder
    .appName("Py Spark Create RDD ex") \
    .config("spark.some.config.option", "some-value") \
    .getOrCreate()

df = spark.sparkContext.parallelize()
```