what is spark

```
spark is a distributed big data processing engine. it can process batch as well as realt
```

Spark Performance Tuning – Best Guidelines & Practices

```
Use DataFrame/Dataset over RDD.
Use coalesce() over repartition()
Use mapPartitions() over map()
Use Serialized data format's.
Avoid UDF's (User Defined Functions)
Caching data in memory.
Reduce expensive Shuffle operations.
Disable DEBUG & INFO Logging.
```

why dataframe is faster than rdd

RDD – RDD API is slower to perform simple grouping and aggregation operations. DataFrame – DataFrame API is very easy to use. It is faster for exploratory analysis,
because rdd not

Spark – Difference between Cache and Persist?

```
Using cache() and persist() methods, Spark provides an optimization mechanism to store the intermediate

Both caching and persisting are used to save the Spark RDD, Dataframe, and Dataset’s. But, the differen
```

PySpark Broadcast Variables

https://sparkbyexamples.com/pyspark/pyspark-broadcast-variables/

```
In PySpark RDD and DataFrame, Broadcast variables are read-only shared variables that are cached and ava

follow->interview_preprations/spark_tutorial/Spark_RDD_Tutorial
```

diff between Accumulator and Broadcast Variables

```
An accumulator is also a variable that is broadcasted to the worker nodes.
The key difference between a broadcast variable and an accumulator is that while
the broadcast variable is read-only, the accumulator can be added to

Accumulator can be used to implement counters (as in MapReduce) or sums.
Spark natively supports accumulators of numeric types, and programmers can add support for new types.
```

spark executor(imp)

https://spark.apache.org/docs/latest/cluster-overview.html

find second higest salary

```
select quantity from orders order by quantity desc limit 1 offset 1;
```

interview links:

```
apache spark interview question

https://www.simplilearn.com/top-apache-spark-interview-questions-and-answers-article

https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/
```

spark example run

```
https://sparkbyexamples.com/spark/spark-how-to-kill-running-application/
```

rdd of spark

```
https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.RDD.html
```

increase and decrease no of partitions and parallelism

```
spark.conf.set("spark.sql.shuffle.partitions", "1000")
spark.conf.set("spark.default.parallelism", "1000")

When you run a PySpark RDD, DataFrame applications that have the Broadcast variables def

PySpark breaks the job into stages that have distributed shuffling and actions are execu
Later Stages are also broken into tasks
Spark broadcasts the common data (reusable) needed by tasks within each stage.
The broadcasted data is cache in serialized format and deserialized before executing eac
You should be creating and using broadcast variables for data that shared across multipl
```

show full text in df

```
use df.show(truncate=False)
```

spark coding links

```
https://sparkbyexamples.com/spark/spark-web-ui-understanding/

https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.RDD.html
```

what is job scheduling in spark

```
By default, Spark's scheduler runs jobs in FIFO fashion. Each job is divided into “stage
```

spark executor(imp) and spark archetictute

https://www.youtube.com/watch?v=JSvzGJC8vjQ

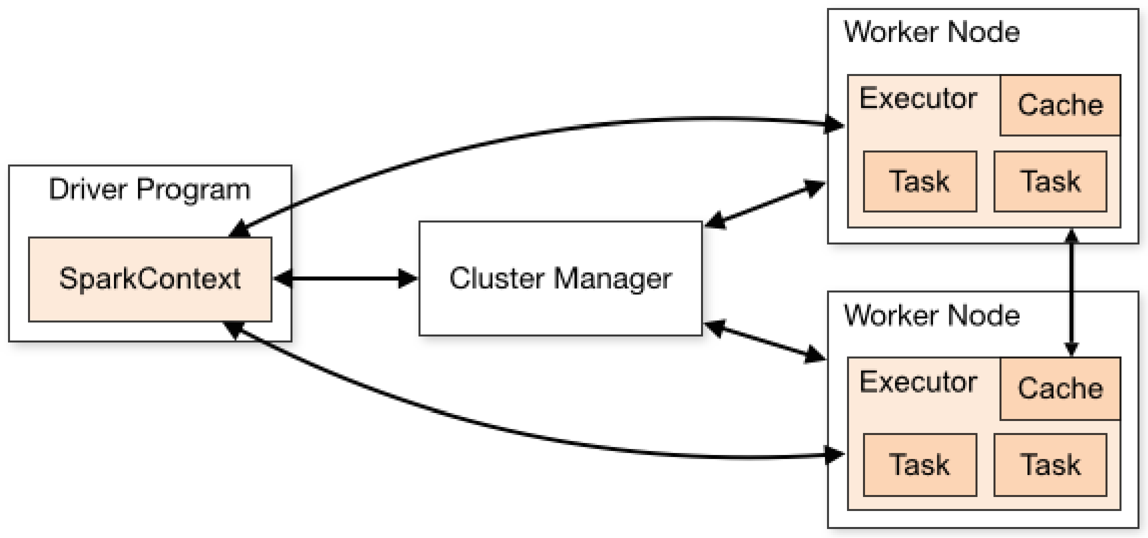
```
spark core component :spark ML-LIB,spark-GraphX,spark-streaming,spark-sql

spark executor : standalone(local),kubenetes ,hadoop yarn ,apche messos

    standalone : this is very simple cluster manager include with spark

    hadoop-yarn : the resource manager of hadoop 2 which stand for yet another negotiator

    apache-mesos : general cluster manager that can run hadoop mapreduce and service as well
```



```
spark Archetiecture define into three part
1.driver program/sparkSession
2.cluster manager
3.worker node (Executor + task)

1.driver program /spark session =>
    this is the main entry point of spark application. the main method of program runs in driver
    driver create sparkSession/ sparkContext
    driver perform transformation and action

    main role of driver program is that
    1. convert user program into task
    2.scheduling task in the executor

2.cluster manager =>
    cluster manager are responsibe for resource allocation anmd launch executor.
    cluster manager is pluggable component of spark
    cluster manager can dynamically adjust resouce used by the spark application

3.Spark Executor:
    the individual task given by the spark job are runs in the spark executor
    Executor Launch once in the begning of the spark application and they run for the entire life cycle

    the two main role play the executor
    1. runs the task that makes up the spark application and return the result to the driver
    2. proivde in memory storage for RDD/DataFrame/DataSets that are cached by the users
```

how to optimize spark job

```
1.repartitions (repartitions split and suffle the data accorss all the node in a cluster
2.broadcating
3.Avoid UDF and UDAF (please try to avoid user define function and use spark optimize fu

4.Dynamic allocation
    According to the workloads it is possible to scale up and scale down the number of e

    $ spark.dynamicAllocation.enabled //Set the value to true to enable dynamic allocati

5.Parallelism
    In spark higher level APIs like dataframe and datasets use the following parameters

    $ spark.sql.files.maxPartitionBytes //Specify maximum partition size by default 128m
```

what is dataset in spark

```
Dataset is a new interface added in Spark 1.6 that provides the benefits of RDDs (strong

the basic difference between dataframe and dataset is
Type Safety: Dataset provides compile-time type safety. It means that the application’s
```

. Python does not have the support for the Dataset API.

what is parquet and why it is used and how this is works internelly

```
https://www.upsolver.com/blog/apache-parquet-why-use
    what is row based and column based
    let's take an example we have a file in which have 5 column and 100 record then what happen
    in csv make 100 files in which one full information of a row but other hand parquet store data into
```

pyspark join

```
pyspark dataframe support sql type join like inner,outer,left,right
df.join(df2,df.id==df2.id,'inner')
```

mapPartitions()

is used to provide heavy initialization for each partition instead of applying
to all elements this is the main difference

data is skewed to remove by using repartitions

Use option maxRecordsPerFile if you want to control the number of records for each partition. This is particularly helpful when your data is skewed (Having some partitions with very low records and other partitions with high number of records).

spark suffling and partitions

https://sparkbyexamples.com/spark/spark-shuffle-partitions/

Based on your dataset size, number of cores, and memory, Spark shuffling can benefit or harm your jobs. When you dealing with less amount of data, you should typically reduce the shuffle partitions otherwise you will end up with many partitioned files with a fewer number of records in each partition. which results in running many tasks with lesser data to process.

On other hand, when you have too much of data and having less number of partitions results in fewer longer running tasks and some times you may also get out of memory error.

Getting a right size of the shuffle partition is always tricky and takes many runs with different value to achieve the optimized number. This is one of the key property to look for when you have performance issues on Spark jobs.

diff between ETL AND ELT

diff.
Transformations are done in ETL server/staging area. but in ELT Transformations are performed in the target system

ETL model used for on-premises, relational and structured data.Used in scalable cloud infrastructure which supports structured, unstructured data sources.