S&P Global

Self introduction

3.5 rate in pyspark

1.spark Architecture
PySpark is similar to Apache Spark which works as master-slave architecture pattern.
- Here, the master node is called the Driver and the slave nodes are called the workers.
- When a Spark application is run, the Spark Driver creates SparkContext, which acts as an entry point to the spark a pplication.
- Here, all the operations are executed on the worker nodes.
- The resources required for executing the operations, on the worker nodes are managed by the Cluster Managers.

The Spark follows the master-slave architecture. Its cluster consists of a single master and multiple slaves.
The Spark architecture depends upon two abstractions:
Resilient Distributed Dataset (RDD)
Directed Acyclic Graph (DAG)

Resilient Distributed Datasets (RDD)

This Datasets are the group of data items, that can be stored in-memory on worker nodes. Here,
Resilient: Restore the data on failure.
Distributed: Data is distributed among different nodes.
Dataset: Group of data.

Directed Acyclic Graph (DAG)

(Directed Acyclic Graph)

- DAG in Apache Spark is a set of Vertices and Edges, where vertices represent the RDDs, and the edges represent the Operation to be applied on RDD.
- So here, DAG is a finite direct graph, that performs a sequence of computations on data. Each node is an RDD par tition, and the edge is a transformation on top of data.

We have few components in the spark Architecture those are:
Driver node
worker node
Executor
Cluster manager
Task

Driver Node

- This Program is a process runs the main() function of the application, and creates the SparkContext object. The pu rpose of SparkContext is, to coordinate the spark applications, running as independent sets of processes on a cluster.
- To run on a cluster, the SparkContext connects to a different type of cluster managers, and then perform the tasks:
-
- It acquires executors on nodes in the cluster.
- Then, it sends your application code to the executors. Here, the application code can be defined by JAR or Python files passed to the SparkContext.
- At last, the SparkContext sends tasks to the executors to run.
Cluster Manager:

- Cluster manager is a platform (cluster mode) where we can run Spark

- The role of the cluster manager is to allocate resources across applications. The Spark is capable enough of runnin g on a large number of clusters.
- It consists of various types of cluster managers such as Hadoop YARN, Apache Mesos, and Standalone Scheduler .

Standalone – This is a simple cluster manager that is included with Spark.
Apache Mesos – This manager can run Hadoop MapReduce and PySpark apps.
Hadoop YARN – This manager is used in Hadoop2.
Kubernetes – This is an open-source cluster manager that helps in automated deployment, scaling and automatic m anagement of containerized apps.

- Here, the Standalone Scheduler is a standalone spark cluster manager that facilitates to install Spark on an empty s et of machines.

Worker Node

The worker node is a slave node
Its role is to run the application code in the cluster.

Executor

An executor is a process launched for an application on a worker node.
It runs tasks and keeps data in memory or disk storage across them.
It read and write data to the external sources.
Every application contains its executor.

Task

A unit of work that will be sent to one executor.
****
2.what is Spark Driver
The spark driver is that the program that defines the transformations and actions on RDDs of knowledge and submits request to the master. Spark driver is a program that runs on the master node of the machine which declares transfor mations and actions on knowledge RDDs.

In easy terms, the driver in Spark creates SparkContext, connected to a given Spark Master.It conjointly delivers the RDD graphs to Master, wherever the standalone cluster manager runs.

A Spark driver (aka an application's driver process) is a JVM process that hosts SparkContextfor a Spark application . It is the master node in a Spark application.
It is the cockpit of jobs and tasks execution (using DAGScheduler and Task Scheduler).
It hosts Web UI for the environment.
It splits a Spark application into tasks and schedules them to run on executors.
A driver is where the task scheduler lives and spawns tasks across workers.
A driver coordinates workers and overall execution of tasks.

- This Program is a process runs the main() function of the application, and creates the SparkContext object. The pur pose of SparkContext is, to coordinate the spark applications, running as independent sets of processes on a cluster.
- To run on a cluster, the SparkContext connects to a different type of cluster managers, and then perform the tasks: -
- It acquires executors on nodes in the cluster.
- Then, it sends your application code to the executors. Here, the application code can be defined by JAR or Python files passed to the SparkContext.
- At last, the SparkContext sends tasks to the executors to run.
****

**
3.Whom will decide the division of data in the spark driver and worker node
****
***
4.how will you filter the records from the particular data frame, like df name is city, He wants the records.
We use isin() method for that here we can use left join for better perfomance.
In Spark isin() function is used to check if the DataFrame column value exists in a list/array of values. To use IS N OT IN, use the NOT operator to negate the result of the isin() function
****
****
5.If you have 2df and large in size, and you can not join them in your system, then what will be your approach towar ed that.
We will
****
****
6. Caching techniques in spark and y we use caching techniques?
Caching is an optimization technique in spark computations.
Coming to this, caching techniques is used to save the Spark RDD, Dataframe, and Dataset's. RDD cache() method d efault saves it to memory (MEMORY_ONLY)

Memory only : Keeps data in memory only
Disk only : Keeps data in disk only
Memory and disk : If the memory is over, it will save data into disk
- Cost efficient – Spark computations are very expensive hence reusing the computations are used to save cost.
- Execution time – Saves execution time of the job and we can perform more jobs on the same cluster.
- Time efficient – Reusing the repeated computations saves lots of time.
****
7. when we have Squit data(key value format) and specific key is in multiple keys then how will you process the dat a and we can't delete the duplicate data? (salting techniques)

Duplicate rows could be remove or drop from Spark SQL DataFrame using distinct() and dropDuplicates() functions , distinct() can be used to remove rows that have the same values on all columns whereas dropDuplicates() can be us ed to remove rows that have the same values on multiple selected columns.

salting technique
The process of using SALT in Spark can be breakdown into: Add a new field and populate it with random numbers . Combine this new field and the existing keys as a composite key, perform any transformation. Once the processing is done, combine the final result
- It is a technique that adds random values, to push Spark partition data evenly. It is usually good to adopt for wide t ransformation which requires shuffling like join operation.
****
8.Optimization techniques in spark

There are
1)Hive optimization techniques
*Hive partitions
*BUcketing
*Indexing
*Vectorization

2)Spark optimization techniques
*Data filtering
*Data partitioning
*Data serialization
*caching and persistance

Coming to optimization techniques in spark there are several techinques like:

1. Serialization:Serialization is performance for any distributed application. By default, Spark uses Java serializer
2. API selection:Spark introduced three types of API to work upon – RDD, DataFrame, DataSet
   RDD is used for low level operation with less optimization
   DataFrame is best choice in most cases due to its catalyst optimizer and low garbage collection (GC) overhead.
   Dataset is highly type safe and use encoders. It uses Tungsten for serialization in binary format.
3. Cache and Persist:Spark provides its own caching mechanisms like persist() and cache().
   cache() and persist() will store the dataset in memory.
   When you have a small dataset which needs be used multiple times in your program, we cache that dataset.
   cashe() no – Always in Memory
   Persist() – Memory and disks
4. ByKey Operation:Shuffles are heavy operation which consume a lot of memory.
   While coding in Spark, the user should always try to avoid shuffle operation.
   High shuffling may give rise to an OutOfMemory Error; To avoid such an error, the user can increase the level of parallelism.
   Use reduceByKey instead of groupByKey.
   Partition the data correctly.
5. File Format selection:Spark supports many formats, such as CSV, JSON, XML, PARQUET, ORC, AVRO, etc.
   Spark jobs can be optimized by choosing the parquet file with snappy compression which gives the high performance and best analysis.
   Parquet file is native to Spark which carries the metadata along with its footer.
6. Garbage Collection Tuning:JVM garbage collection can be a problem when you have large collection of unused objects.
   The first step in GC tuning is to collect statistics by choosing – verbose while submitting spark jobs.
7. Advance Variable:Broadcasting plays an important role while tuning Spark jobs.
   Broadcast variable will make small datasets available on nodes locally.
8. Level of Parallelism

Parallelism plays a very important role while tuning spark jobs.
Every partition ~ task requires a single core for processing.
There are two ways to maintain the parallelism:
Repartition: Gives equal number of partitions with high shuffling
Coalesce: Generally reduces the number of partitions with less shuffling.

****

9.How did you connect to the table using Pyspark.

Step 1: Import the modules. In this scenario, we are going to import the pyspark and pyspark SQL modules and also specify the app name

```
import pyspark
from pyspark.sql import SparkSession
from pyspark.sql import Row
appName= "hive_pyspark"
master= "local"
```

Step 2: Create Spark Session:create a spark session and enable the Hive support to interact with the hive database.

```
spark = SparkSession.builder \.master(master).appName(appName).enableHiveSupport().getOrCreate()
```

Step 3: Verify the databases:Here we are going to verify the databases in hive using pyspark as

```
df=spark.sql("show databases")
```

Step 4: Verify the Table:Here we are going to verify the table in hive using pyspark as

```
tables = spark.sql("show tables").show()
```

Step 5: Fetch the rows from the table:Here we are going to fetch rows from the table in hive using pyspark and store them in the dataframe as

```
df1=spark.sql("select * from drivers_table limit 5")
df1.show()
```

Step 6: Print the schema of the table:Here we are going to print the schema of the table in hive using pyspark as

```
df1.printSchema()
```

****

**

10. Suppose sale table, with few employee columns emp_name, designation, dept, sales. Now i want top3 emp od ev ery dept who did most sale

df.groupby('dept')

****

11.Debugging

Debugging in Python is facilitated by pdb module(python debugger) which comes built-in to the Python standard lib rary. It is actually defined as the class Pdb which internally makes use of bdb(basic debugger functions) and cmd(su pport for line-oriented command interpreters) modules. The major advantage of pdb is it runs purely in the command line thereby making it great for debugging code on remote servers when we don't have the privilege of a GUI-based debugger.

pdb supports-

Setting breakpoints
Stepping through code
Source code listing
Viewing stack traces

A debugger is a program that can help you find out what is going on in a computer program. You can stop the execut ion at any prescribed line number, print out variables, continue execution, stop again, execute statements one by one, and repeat such actions until you have tracked down abnormal behavior and found bugs.

****

12.Class method in python

The classmethod() is an inbuilt function in Python, which returns a class method for a given function.;
Class method: Used to access or modify the class state. In method implementation, if we use only class variables, the n such type of methods we should declare as a class method. The class method has a cls parameter which refers to th e class.

A class method is a method which is bound to the class and not the object of the class. They have the access to the st ate of the class as it takes a class parameter that points to the class and not the object instance. It can modify a class s tate that would apply across all the instances of the class

****

13.Lambda functions.

We use lambda functions when we require a nameless function for a short period of time. In Python lambda function is a small anonymous function. A lambda function can take any number of arguments, but can only have one expres sion.we can use them as an anonymous function inside another function

****

14.Data skewness

It is nothing but a measure, whether the peak is centered in the middle of the distribution, +ve value represents the p eak is off to the left and -ve peak value represents the peak is off to right.

- Skewness means lack of symmetry, to have an idea about the shape of the curve in the normal distribution.

- So Skewness is basically an analytical term, which refers to the value distribution in a given dataset.
- When we say there is highly skewed data, it means that some column values have more rows and have very few rows, i.e., the data is not evenly distributed.

use case of data skewness?

- Usually, in Apache Spark, data skewness is caused by transformations, that change data partitioning like join, gro upBy, and orderBy.
- For example, joining on a key that is not evenly distributed across the cluster, which causes some partitions to be very large and not allowing Spark to process data in parallel.

****

14. repartion and colease

Spark repartition () is used to increase or decrease the RDD, DataFrame, Dataset partitions whereas the Spark coales ce () is used to only decrease the number of partitions in an efficient way.

coalesce may run faster than repartition, but unequal sized partitions are generally slower to work with than equal siz ed partitions. One additional point to note here is that, as the basic principle of Spark RDD is immutability. The repa rtition or coalesce will create new RDD.

****

Class method vs static method

Class method can access and modify the class state. Static Method cannot access or modify the class state. The class method takes the class as parameter to know about the state of that class. Static methods do not know about class stat e.

class method:

The class method takes cls (class) as first argument.

Class method can access and modify the class state.

The class method takes the class as parameter to know about the state of that class.

@classmethod decorator is used here.

Static method:

The static method does not take any specific parameter.

Static Method cannot access or modify the class state.

Static methods do not know about class state. These methods are used to do some utility tasks by taking some param eters.

@staticmethod decorator is used here.

****

Brodcast join

Broadcast join is an important part of Spark SQL's execution engine. When used, it performs a join on two relations by first broadcasting the smaller one to all Spark executors, then evaluating the join criteria with each executor's part itions of the other relation.

When should I broadcast join spark?

in spark when we want to join one small data frame with the large one. the requirement here is we should be able to store the small data frame easily in the memory so that we can join them with the large data frame in order to boost t he performance of the join

****

Accumulater variable

An accumulator is a variable that the program uses to calculate a sum or product of a series of values. A computer pr ogram does this by having a loop that adds or multiplies each

successive value onto the accumulator.In order to aggregate the information through associative and commutative op erations, we use them.

Code: class pyspark.Accumulator(aid, value, accum_para)

****

1.etl vs elt, which one is better?

1) Support for Data Warehouse:
Yes, ETL is the traditional process for transforming and integrating structured or relational data into a cloud-based o r on-premises data warehouse.

Yes, ELT is the modern process for transforming and integrating structured or unstructured data into a cloud-based d ata warehouse.

2) Support for Data Lake/Mart/Lakehouse:
No, ETL is not an appropriate process for data lakes, data marts or data lakehouses.

Yes, the ELT process is tailored to provide a data pipeline for data lakes, data marts or data lakehouses.

3) Size/type of data set:
ETL is most appropriate for processing smaller, relational data sets which require complex transformations and have been predetermined as being relevant to the analysis goals.

ELT can handle any size or type of data and is well suited for processing both structured and unstructured big data. S ince the entire data set is loaded, analysts can choose at any time which data to transform and use for analysis.

4) Implementation:
The ETL process has been around for decades and there is a mature ecosystem of ETL tools and experts readily avail able to help with implementation.

The ELT process is a newer approach and the ecosystem of tools and experts needed to implement it is still growing.

5) Transformation:
In the ETL process, data transformation is performed in a staging area outside of the data warehouse and the entire d ata must be transformed before loading. As a result, transforming larger data sets can take a long time up front but an alysis can take place immediately once the ETL process is complete.

In the ELT process, data transformation is performed on an as-needed basis in the target system itself. As a result, th e transformation step takes little time but can slow down the querying and analysis processes if there is not sufficient processing power.

6. Loading:
The ETL loading step requires data to be loaded into a staging area before being loaded into the target system. This multi-step process takes longer than the ELT process

In ELT, the full data set is loaded directly into the target system. Since there is only one step, and it only happens on e time, loading in the ELT process is faster than ETL.

7) Maintenance/Ease of Use:
ETL processes that involve an on-premise server require frequent maintenance by IT given their fixed tables, fixed ti melines and the requirement to repeatedly select data to load and transform. Newer automated, cloud-based ETL sol utions require little maintenance.

The ELT process typically requires low maintenance given that all data is always available and the transformation pr ocess is usually automated and cloud-based.

8) Cost:
ETL can be cost-prohibitive for many small and medium businesses.

ELT benefits from a robust ecosystem of cloud-based platforms which offer much lower costs and a variety of plan

options to store and process data.

9) Hardware:
The traditional, on-premises ETL process requires expensive hardware. Newer, cloud-based ETL solutions do not re quire hardware.

Given that the ELT process is inherently cloud-based, no additional hardware is required.

10) Compliance:
ETL is better suited for compliance with GDPR, HIPAA, and CCPA standards given that users can omit any sensitiv e data prior to loading in the target system.

ELT carries more risk of exposing private data and not complying with GDPR, HIPAA, and CCPA standards given t hat all data is loaded into the target system.
****
****

2.datalake vs datawarehouse
DATALAKE:
It is a type of DB which is designed to handle huge amount of data . it stores historical and current data.it stores struc tured,semi-structured data and unstructured data. it is OLAP
eg: amazon,s3,azure data lake
DATAWAREHOUSE:
which contains structured and semi structured data . data ware house will analyse data,looking for insights and create BI reports and dash boards
eg:snowflake and mysql
****
****

3.fact vs dimention
A fact table holds the data to be analyzed, and a dimension table stores data about the ways in which the data in the fact table can be analyzed.
****
****

4.what is spark context?
 PySpark SparkContext is an initial entry point of the spark functionality.
 It also represents Spark Cluster Connection and can be used for creating the Spark RDDs(Resilient Distributed D atasets) and broadcasting the variables on the cluster.

 When we want to run the Spark application, a driver program that has the main function will be started.
 From this point, the SparkContext that we defined gets initiated. Later on, the driver program performs operations inside the executors of the worker nodes.
 Additionally, JVM will be launched using Py4J which in turn creates JavaSparkContext. Since PySpark has defau lt SparkContext available as “sc”, there will not be a creation of a new SparkContext
****
****

Star Schema:
Star schema is the type of multidimensional model which is used for data warehouse. In star schema, The fact tables and the dimension tables are contained. In this schema fewer foreign-key join is used. This schema forms a star with fact table and dimension tables.

Snowflake Schema:
Composite key -is a Candidate key that consists of more than one attribute.
Foreign key -is an attribute which is a Primary key in its parent table but is included as an attribute in the host table.
Foreign keys- may accept non-unique and null values.
Snowflake Schema is also the type of multidimensional model which is used for data warehouse. In snowflake sche

ma, The fact tables, dimension tables as well as sub dimension tables are contained. This schema forms a snowflake with fact tables, dimension tables as well as sub-dimension tables.
****

1. Batch Processing :
Batch processing refers to processing of high volume of data in batch within a specific time span. It processes large volume of data all at once. Batch processing is used when data size is known and finite. It takes little longer time to processes data. It requires dedicated staffs to handle issues. Batch processor processes processes data in multiple pas ses. When data is collected overtime and similar data batched/grouped together then in that case batch processing is used.
****

2. Stream Processing :
Stream processing refers to processing of continuous stream of data immediately as it is produced. It analyzes stream ing data in real time. Stream processing is used when the data size is unknown and infinite and continuous. It takes f ew seconds or milliseconds to process data. In stream processing data output rate is as fast as data input rate. Stream processor processes data in few passes. When data stream is continuous and requires immediate response then in that case stream processing is used.
****

Foreign Key
A foreign key can contain duplicate values. There is no limitation in inserting the values into the table column. Whil e inserting any value in the foreign key table, ensure that the value is present into a column of a primary key
****

1.) ways to read a file in pysaprk and what are the file extension that pyspark support to read?
there are three ways to read text files into PySpark DataFrame.
1.Using spark.read.text()
2.Using spark.read.csv()
3.Using spark.read.format().load()

1. spark.read.text()
It is used to load text files into DataFrame whose schema starts with a string column. Each line in the text file is a new row in the resulting DataFrame. Using this method we can also read multiple files at a time.

syntax: spark.read.text(path)

2: Using spark.read.csv()
It is used to load text files into DataFrame. Using this method we will go through the input once to determine the i nput schema if inferSchema is enabled. To avoid going through the entire data once, disable inferSchema option or s pecify the schema explicitly using the schema.

Syntax: spark.read.csv(path)

3: Using spark.read.format()
It is used to load text files into DataFrame. The .format() specifies the input data source format as “text”. The .load() loads data from a data source and returns DataFrame.

****

3.) difference between filter and where?
PySpark filter() function is used to filter the rows from RDD/DataFrame based on the given condition or SQL expres sion, you can also use where() clause instead of the filter()

****

4.) how select is used in pyspark?
In PySpark, select() function is used to select single, multiple, column by index, all columns from the list and the nes ted columns from a DataFrame, PySpark select() is a transformation function hence it returns a new DataFrame with

the selected columns.

You can select the single or multiple columns of the Spark DataFrame by passing the column names you wanted to s elect to the select() function. Since DataFrame is immutable, this creates a new DataFrame with a selected columns. show() function is used to show the DataFrame contents

****

5.what is Caching and Persistence of DataFrame?

Caching or persistence are optimization techniques for (iterative and interactive) Spark computations.

Both caching and persisting are used to save the Spark RDD, Dataframe, and Dataset's. But, the difference is, RDD cache() method default saves it to memory (MEMORY_ONLY) whereas persist() method is used to store it to the us er-defined storage level.

both caching and persistance dataframe are used for saving the memory in RDD and dataframe. where in caching it s aves automatically whereas in persistance it will be saved by user defined level
****
What is an anti join?
An anti join returns the rows of the first table where it cannot find a match in the second table. Anti joins are a type o f filtering join, since they return the contents of the first table, but with their rows filtered depending upon the match conditions
****
Pivoting and unpivoting
PIVOT carries out an aggregation and merges possible multiple rows into a single row in the output. UNPIVOT doe sn't reproduce the original table-valued expression result because rows have been merged. Also, null values in the in put of UNPIVOT disappear in the output.
****