# Introduction

## What is Spark Streaming?

Using spark streaming we can process live streams of data.

Examples of Stream of datass

1. Logs generated by user activities in mobile application
2. Log files generated by production web servers.

Spark streaming provides api called as Dstream which is similar to RDD api of spark core. So we can apply spark operations such as transformations and actions.

Because of this programmer can learn spark streaming and switch between spark streaming application development and spark application that process the data stored in memory or disk.

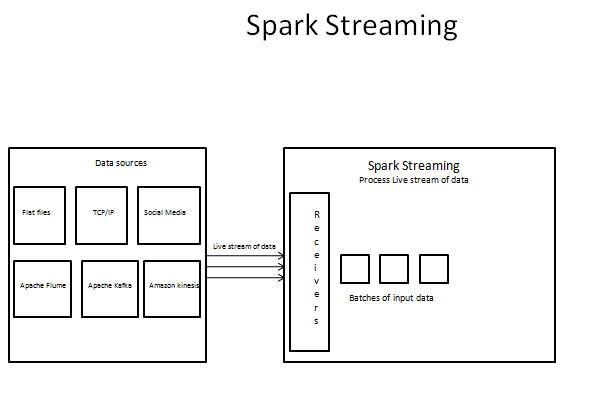
## Name some of the use cases of spark streaming?

1. Credit card fraud detection
2. Click stream analytics and recommendations: In DMP project we are doing this. When user perform some operations on website, we need to recommend him based on his activities in the site.
3. Ad recommendations
4. Understand user behaviour on the go
5. Spam filtering: As and when emails are coming in need to identify spam.
6. Real time social media analytics: When people are tweeting and posting something we need to analyse it.
7. Network intrusion detection: We need to keep listening to web logs and network logs to identify if somebody is trying to intrude.
8. Stock market analytics

## Name some of the use cases of DMP project?

Maybe we need to know the person who searched for particular mobile, who’s age is between 20-30 and who uses non H mobile.

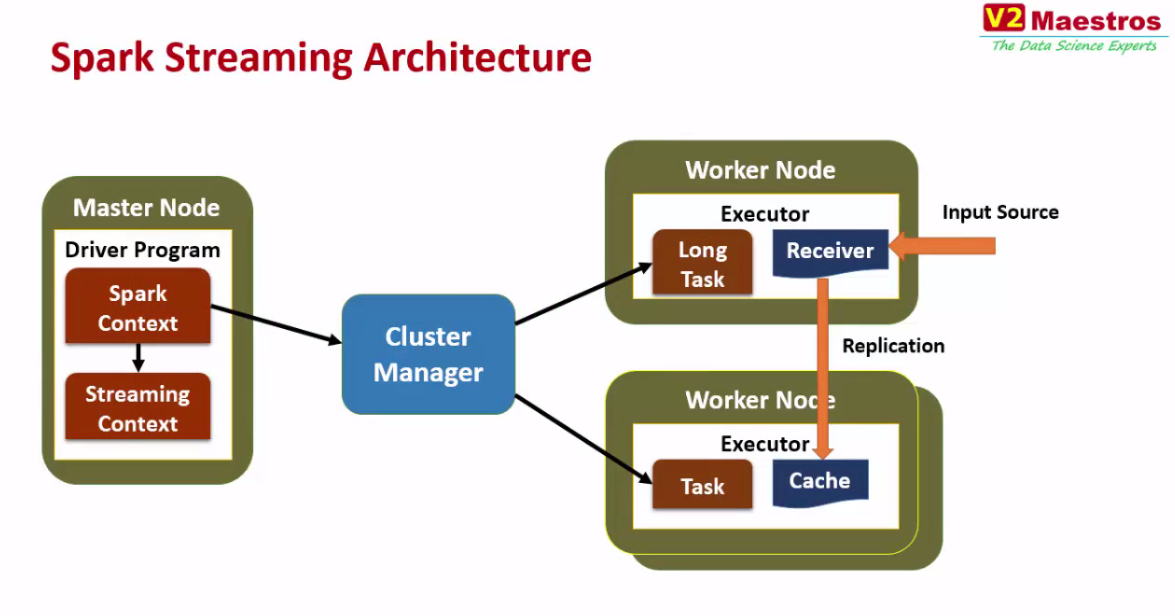
1. Category: H has e-commerce web application. Like other e commerce application, it will have categories like mobiles, smart watches, tabs etc. They want to know the person who viewed some mobile or generally person who viewed mobiles, person who viewed smart watches.
2. Keyword: Also, we need to know the person who searched for mobiles in search tab.
3. Device attributes: Number of users who are using mobile.
4. User attributes: Users who are living city aged between 10 to 20



## What are the data sources that spark streaming supports?

1. Flat files: As and when the flat file are created spark streaming application can look at them and make decisions.
2. TCP/IP: Spark streaming application can hook on to TCP/IP port and It can use whatever messages coming to that port
3. Apache Flume:
4. Apache Kafka:
5. Amazon Kinesis:
6. Real time feed from Twitter, Facebook and other social media: Libraries are available in spark for all of them we can use those libraries and hook on to social media

## What is the architecture of spark streaming? How Spark Streaming Works Internally?



We know that there will be Master node with Driver program and spark context.

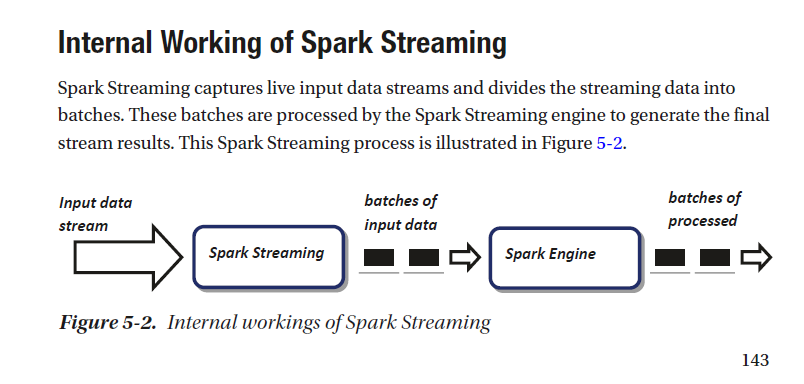
To trigger streaming first we need to create streaming context.

What happens when we create streaming context?

When we create streaming context one of the worker node in the cluster will be assigned the job of listening to data source. To listen spark is going to create a long task (long task is the task that will be keep running all the time) which intern creates a receiver which will be keep listening to input data source. As and when input data source generates data receiver receives data. The data received will be treated as Dstream. Dstream is nothing but sequence of RDDs

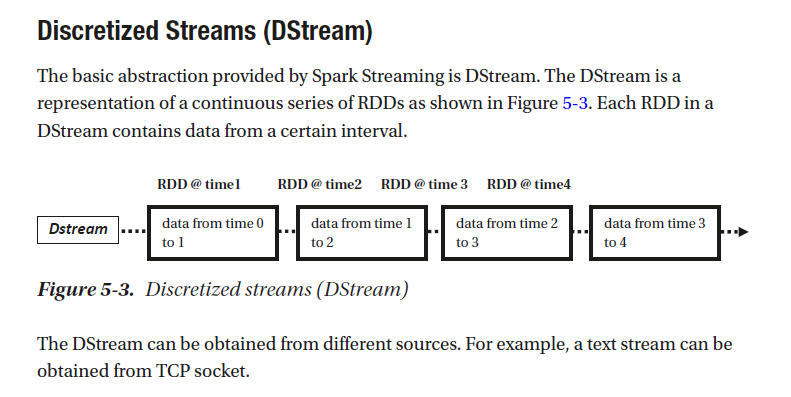
Receiver then takes care of propagating the data received to all the worker nodes.

Then there will be normal task in other worker nodes which will act up on data.



## Basic Concepts of Spark Stream

## DStream (micro batch)

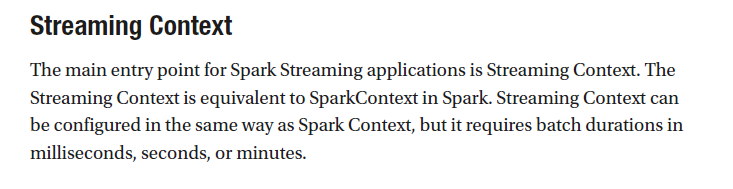


## What are DStreams in case of spark streaming?

DStream means Discretized stream.

To DStream all data is received, and all process occurs. On this Stream we can attach operations (transformations and actions) so that whenever data comes to this stream those operations will be applied.

## Streaming context

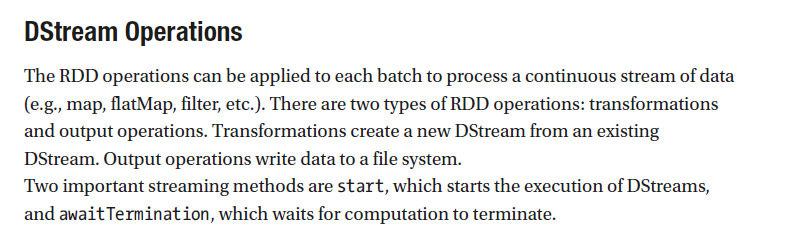


## How will you create Streaming Context?

Streaming context is created from spark context.

Dstreams are created from streaming context.

## DStream Operations?



## How does this operation will be applied to Dstreams ?

A micro batch window is setup say 3 seconds

Any data received is accumulated for 3 seconds and this 3 second data is processed as batch these batches are called as micro batches.

Each accumulated micro batch is an RDD so we can apply any of the RDD operations on DStreams.

## Dstream processing?

Spark collects incoming data for each of the interval the interval is nothing but micro batch.

The data received for that interval is collected as RDD. As it is RDD we can perform spark operations like transformation and action on the DStream.

Suppose we want to know something across Dstrems say counting incoming stream of data we can have Global variable can be used to track data across Dstreams

There are some windowing functions available for computing across multiple DStreams

## Spark Streaming Example Using TCP Socket

## IMP: What is Stateful streaming?

When data are tracked on each RDD, this is known as stateless streaming.

When data are tracked across all RDD, it is known as stateful streaming.

State full: **Remember something (state),** Say for example we must calculate in last 30 seconds how many users visited some website. And we must calculate this every 5 second.

In such case last 25 seconds data + this 5 second of data we must keep in memory and perform operations on that. so, these kind operations that are performed on previous batches are called stateful operations.

State less **(do not remember state process and forget)**

In case of stateless every batch is computed and discarded. In this case current batch will not have any dependency on previous batch. These kinds of operations are called as stateless operations or transformations

## IMP: What are the use cases of stateful transformation?

Trend detection. We need to analyse some trend.

## In stateful transformation instead of having window length 30 second Why not have batch interval of 30 seconds? (additional benefit of stateful over stateless)

For example in spark streaming window length of 30 seconds are given sliding interval is 5 seconds instead of that I can create batch interval of 30 seconds right .

If we create batch interval as 30 seconds then there will be only one batch of 30 seconds we cannot put dependency on previous batch. So if we configure window time 30 seconds slide time as 5 seconds we will get one batch every 5 seconds and 6 batches are kept in memory for 30 seconds. So we can perform operations on last 6 bathes and we can yield the result. When next batch of 5seconds data comes the oldest batch will be dropped.

## Are the stateful transformations are memory intensive?

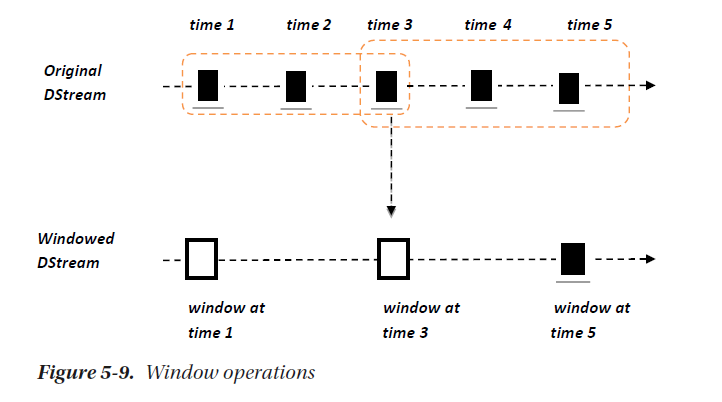
Yes, it is memory intensive because we must keep data of previous batches in memory. It again depends on how much old data we need.

## How many types of Stateful streaming are there?

Window-based.

Full-session-base.ss

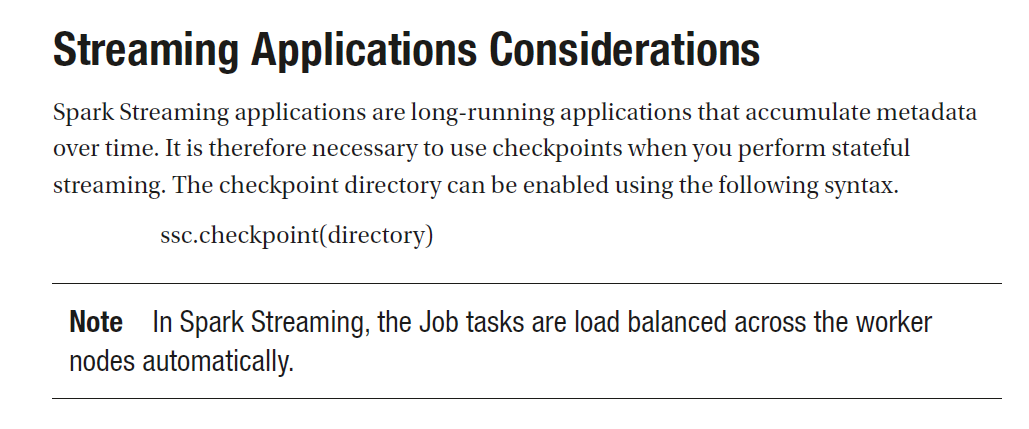
## Window-based Streaming?



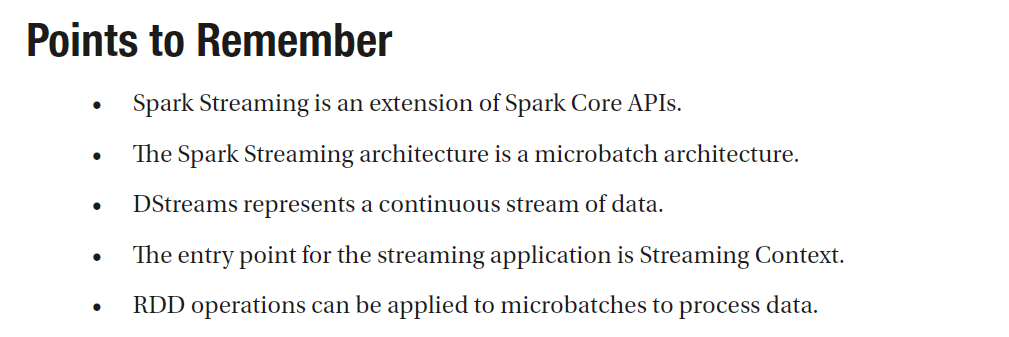
In this case we can perform, transformations over a sliding window of data.

## Full-Session-Based Streaming

## Streaming application considerations?



## Points to Remember in Spark Streaming?

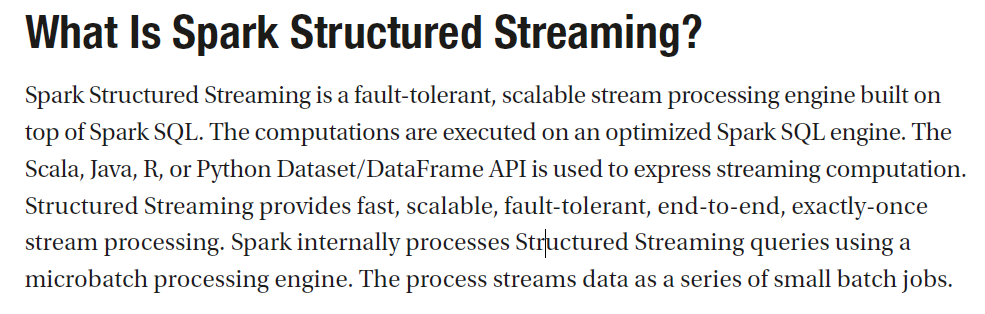


# Structured Streaming

## How Spark structured streaming is built?

structured stream processing built on top of the Spark SQL engine.

## What Is Spark Structured Streaming?



1. **How is spark streaming is different from batch processing?**

In Streaming primary abstraction is DStream which is nothing but sequence of RDDs. Where as in spark batch processing we will take one RDD or multiple RDDs we will jpin or do some preprocessing and arrive at some result.

1. **Is spark streaming a real time streaming?**

Spark streaming is not real time it is near real time. It means as soon as data comes, we are not processing data will be accumulated for micro batch interval configured. And then that batch will be processed.

1. **How much the batch interval could be?**

Batch interval varies from seconds to microseconds. It can also be minutes but it loses significance of stream processing.

1. **Is there a limit for number of messages that are accumulated in a batch interval?**

No there is no limit as such

For kafka one topic if there is multiple partition and one Dstream there could be lesser amount or data. We can create one Dstream for one partition by creating number of Dstreams equals to number of partitions in the topic this is in case of receiver based approach.

In case of kafka direct stream Number of partitions of the RDD(Dstream) is equal to number partition of kafka topic. So each partition of RDD will be

1. What is the relationship between number of batches and number of RDDs?

One batch is one RDD

1. What is the importance of spark.streaming.blockinterval? (In case of spark one block is like on partition of RDD)

By default the value of this configuration is 200 ms. It means data received every 200 ms will become one partition data of RDD(Dstream).

1. **What is the relationship between batch interval block interval and number of partitions?**

In case of receiver based approach

Partitions = (receivers \* batch interval)/blockinterval.

**So if there is only one receiver and batch interval is 1 sec (1000 ms) and block interval is 200ms then number of partitions of RDD will be 5.**

In case of direct stream number of partitions of RDD(Dstream) will be equal to number of partitions of Kafka topic.

1. **Details on receiver based approach in spark streaming?**

**How to achieve read parallelism in receiver based approach?**

In case of receivers based approach number of partitions will also be depending on number of receivers (receivers are used to achieve parallelism) .

If one topic is having 5 partitions if we create only one Dstream per topic so it will be reading from all the 5 partitions it will be too much of burden on dstream so in this case we are not achiving full parallelism here.

We can achieve parallelism using below 2 ways.

Create multiple Dstream one per partition: In this case one Dstream reads from one partition, we can union all dstreams and do the processing

Create one stream with multiple threads : There is disadvantage of creating multiple threads per dstream there multiple threads will be corresponding to on receiver . Here each thread will try to read the data and form an RDD so all threads will be acting upon same RDD so threads will take lock on RDD as result lot of time consumption happens in context switching

1. **How to achieve processing parallelism in receiver-based approach?**

We can increase number of partitions

1. What are the disadvantages of receiver based approach?

If the receiver is lost there is no way to know whether it is because of what issue whether is it because of spark or Kafka. So there is data lost seen.

1. **Difference between receiver and direct based approach in spark streaming?**

**What is the need for Kafka direct streaming?**

Receiver based approach: In case of receiver-based approach a long run task will be created which in turn creates a receiver this will be running till the end of spark streaming. The disadvantage is if the receiver is lost then there is no way to understand if it is due to kafka or if it is due to spark so there will be data lost

Direct based approach: Here there will be no receiver data will be directly consumed from kafka partition.

Advantages are

Simplfied parallelism: No need to create multiple kafka streams and then union them, with this approach RDD will be created with number of partitions as kafka

Achiveing zero data lose with receiver based there will be data lost if receiver is lost.

1. **Are the RDDs replicated in streaming application ?**

**Yes. Default replication is 2.because streaming data is loaded into memory so there is need to replicating RDDs in memory**

**Fault tolerant:**

1. **What is the major requirement from spark streaming?**

Spark streaming application should run 24/7.

As a result spark streaming application should be fault tolerant

1. **Fault tolerance in spark streaming? IMP**

Fault tolerance in spark streaming is achieved using check pointing

1. **What is Check pointing? IMP**

Check pointing saves application state timely to reliable storage (HDFS).When the driver is restarted the recovery takes place.

If any data is lost the recovery of the data should be speedy Spark streaming uses check pointing for this purpose.

Check pointing is feature or mechanism used to achieve fault tolerance in Spark Streaming.

What does this mechanism do?

periodically save data about the application to a reliable storage system, such as HDFS or Amazon S3. Later this data is used for recovery in case of application failure.

1. **What are the types of data that we check point in spark? IMP**
2. **MetaData Check pointing:**

Metadata check pointing includes *Configurations*: Configuration used to create streaming.

*DStreams operations*: operations applied on Stream or Operations that define Streaming application.

*Incomplete batches*: Batches which are in the queue but not complete.

1. **Data Check Pointing:**

Save the RDD to reliable storage.

This is to be done when RDD of One batch depends on the RDD of Previous batch?

1. **What are the types of CheckPointing in spark? IMP**

**Reliable Check pointing:**

The actual RDD is saved in reliable distributed file system, e.g. HDFS. To set the checkpoint directory call: SparkContext.setCheckpointDir(directory: String)

When running our application in cluster mode the directory should be HDFS path since the driver tries to recover the check pointed directory from local file but the checkpoint files are actually on the executor machine.

**Local Check pointing:**

RDD is persisted to local storage in the executor.

Note:

In case of task failure or executor program failure?

Spark streaming can recompute state using lineage graph of transformations but check pointing controls how far back up must go.

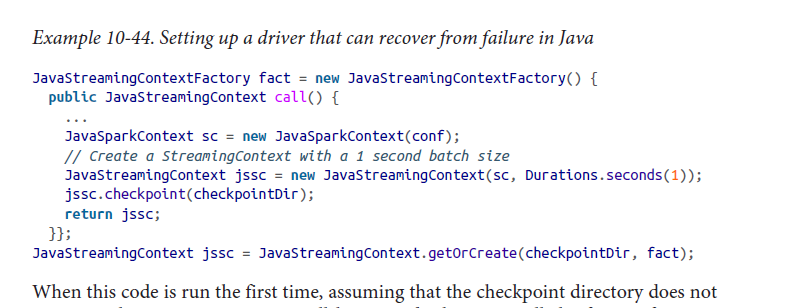
What if driver program fails?

1. **Driver Fault tolerance or How Driver failure is handled in spark? IMP**

We can re-launch the driver program again and ask driver program to recover from checkpoint.

In such case Spark streaming identifies how far the previous run processed data and take overs from there.

To achieve this in code instead of simply creating new StreamingContext StreamingContext.getOrCreate() function.



1. **How to monitor Driver program? IMP**

We can monitor the spark driver program using tools like **monit** and restart it.

We can also write shell scripts to monitor driver program.

1. **Worker fault tolerance or how worker failure is handled in spark? IMP**

Data Received from external sources is replicated among workers.

RDDs created through transformations of this data are tolerant to failures, using RDDs lineage we can recomputed the lost data using surviving replica of the data.

1. **Fault tolerance of receivers in spark streaming?**

We know that in case of spark streaming multiple receivers will be running in multiple worker nodes.

When some receiver fails spark streaming restarts the receiver in another nodes in the cluster.

Whether it loses received data or depends on nature of source and implementation of receiver. From source side whether source can resend the data and from receiver side whether it updates the source about received data or not.

Example Flume, there are 2 types of receivers.

1. Receiver-pull-from-sink Model

Spark removes the elements only once they have been replicated inside Spark.

1. Push-to-receiver Model,

If the receiver fails before the data is replicated some data can be lost.

1. **What are the guarantees that are provided by receivers in case of Spark Streaming? IMP**
2. All data read from a reliable filesystemis reliable, because underlying file system is replicated (e.g., with StreamingContext.hadoop Files).

Spark streaming will remember which data it processed in its check points and will pick it again where it left if my application crashes.

1. For unreliable sources such as Kafka, push-based Flume, or Twitter, Spark replicates the input data to other nodes, but it can briefly lose data if a receiver task is down.

In Spark 1.1 and earlier, received data was replicated in-memory only to executors, so it could also be lost if the driver crashed (in which case all executors disconnect). In Spark 1.2, received data can be logged to a reliable file system like HDFS so that it is not lost on driver restart.

Note: learning spark streaming is suggesting doing streaming from HDFS data do some research on it.

1. **How Spark Streaming provides data processing guarantees?**

We know that spark provides Worker tolerance.

If some worker fails some data gets reprocessed, the final transformed data will be same as the data if the data was processed exactly once.

The actual problem arises when the final result is to be pushed to external systems. In this case the task which pushes the data to external system may execute multiple times because of failure and data can be pushed multiple times. Since this involves external system spark does not provide any features to handle this kind of situations. So we need to handle this case at application level code.

So we can use transactions to push data to external systems, that is atomically push one RDD partition at a time.

Or we should design in idempotent way that is even if you do same operation multiple times the result should not change.

For example, Spark Streaming’s saveAs...File operations automatically make sure only one copy of each output file exists, by atomically moving a file to its final destination when it is complete

1. **What is the need for transform in spark streaming?**

In case of spark streaming if we attach transformation to Dstream before returning particular RDD batch Transformation will be applied and resultant RDD in the form of Dstream will be returned .

1. **Need for foreach RDD?**

For Each RDD is like getting RDD out of Dstream and on top of that RDD we can apply operations it will not return any thing

1. **What is the difference between for each and transform?**

**Where as in case of transformation we get RDD during transformation ultimately we get resultant form as DStream**

1. **What is the impact of persisting RDDs in streaming application?**

**In case of Streaming Data will be continuously coming and and for each window time one RDD is formed. So once if we persist one RDD and if RDD is not unpersisted it will be keep on persisiting incoming RDDs . Because DStream is sequence of RDDs so if we do RDD persist of one RDD it will persist all RDDs if we have not written code for unpersisitng**

1. **Check pointing ? Meat data vs data?**

**There are 2 kinds of check pointing.**

1. **Meta data check pointing.**
2. **Data check pointing.**
3. **What is the impact of check pointing at too frequent intervals?**

**Too much of IO as a result slow processing.**

1. **Performance tuning (setting right batch interval)?**

**We need to avoid by key transformations.**

**Setting right batch interval based on your incoming data. If processing interval is more than batch interval newly coming job will be in queue.**

1. **Memory tuning (persist and unpersiting the data, serialization)**

**Persisting the data when to persist RDD. Even in case of batch processing it is fine. If you persist and do not unpersisit after the application ends it is automatically cleans up the memory. Whereas in spark streaming if you persist and do not unpersisit the RDD It keep on persisting all the RDDs.**

**Serialization:**

* **Spark streaming with kafka**

1. **What are the approaches to connect to kafka in spark streaming?**

There are two approaches.

1. Receiver based approach
2. Direct approach (receiver less)
3. **Does spark streaming uses zookeeper to store the kafka offset?**

No, Consumed offsets are tracked by streaming itself.

It will not use zookeeper to store the offset.

1. **Many Kafka monitoring tools depends on zookeeper for offsets, but spark streaming does not maintain offsets in zookeeper then how kafka monitoring tools will work?**

In that case develepoer needs to handle that from spark streaming application that he write.He has to update zookeeper/Kafka from streaming application.

Developer can access the offsets used in each batch from the generated RDDs (see HasOffsetRanges).

1. **How failure Recovery is done? How spark streaming is fault tolerant?**

Need to Enable check pointing from SparkStreamingContext in order to recover from driver failures

The information on consumed offset can be recovered from the checkpoint

1. **In spark streaming what is the check pointing interval?**

Batch interval configured

1. **Which approach does D project is using?**

Direct approach, receiver less.

1. **How data is being consumed by RTS?**

Need to read java doc of create direct stream api

* **DMP Project Spark streaming applications.**

**RTS**