Big Data Analytics
(Dealing with streams of data)

Problem Statement:

We want to create an application that ingests data from continuous stream, processes it in Realtime and output metrics on a dashboard for the business user to see.

More concretely, in past 2 seconds, output the product name that has between 4.2 – 5.0 stars out of 5 stars review rating, has at least 5 reviews and is running low on stocks (below 5 products left only).

When the application is running, the business user has an automated script running for him. he/she takes the product name generated by your application and submits a stock regeneration request. The caveat here is that we want to keep some information across streams/batches/windows/micro-batch. i.e., if the same product comes up in two consecutive batches, the output is not sent to the business user.

Dataset:

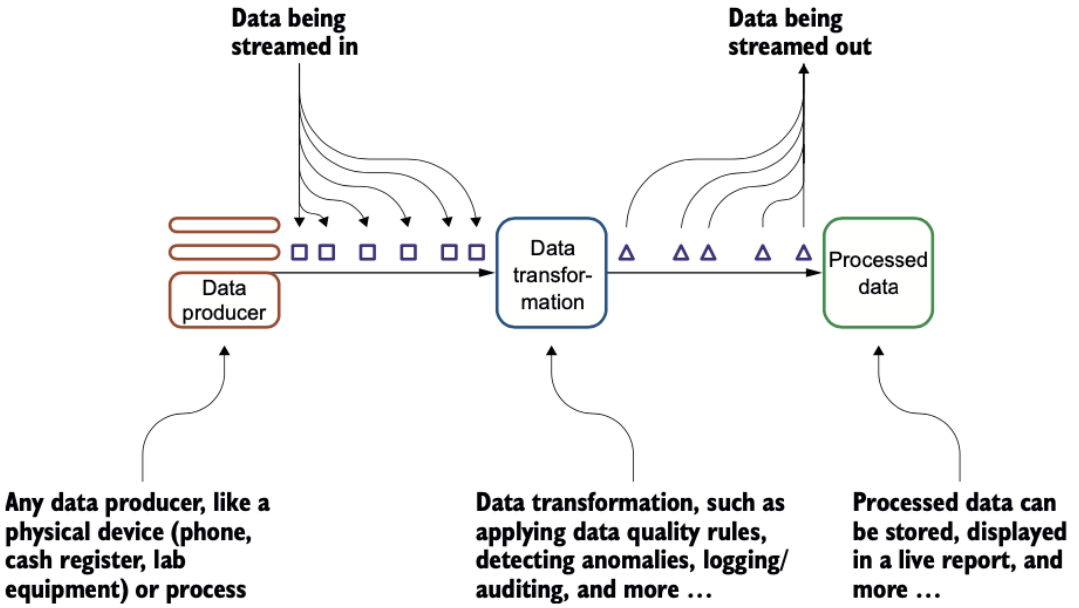
(10,000 Records)
https://data.world/promptcloud/fashion-products-on-amazon-com
For bigger version of the same dataset (7 million records):
https://www.promptcloud.com/datastock-access-ready-to-use-datasets?utm_source=data-world&utm_medium=referral

Technical details:

Part-1:

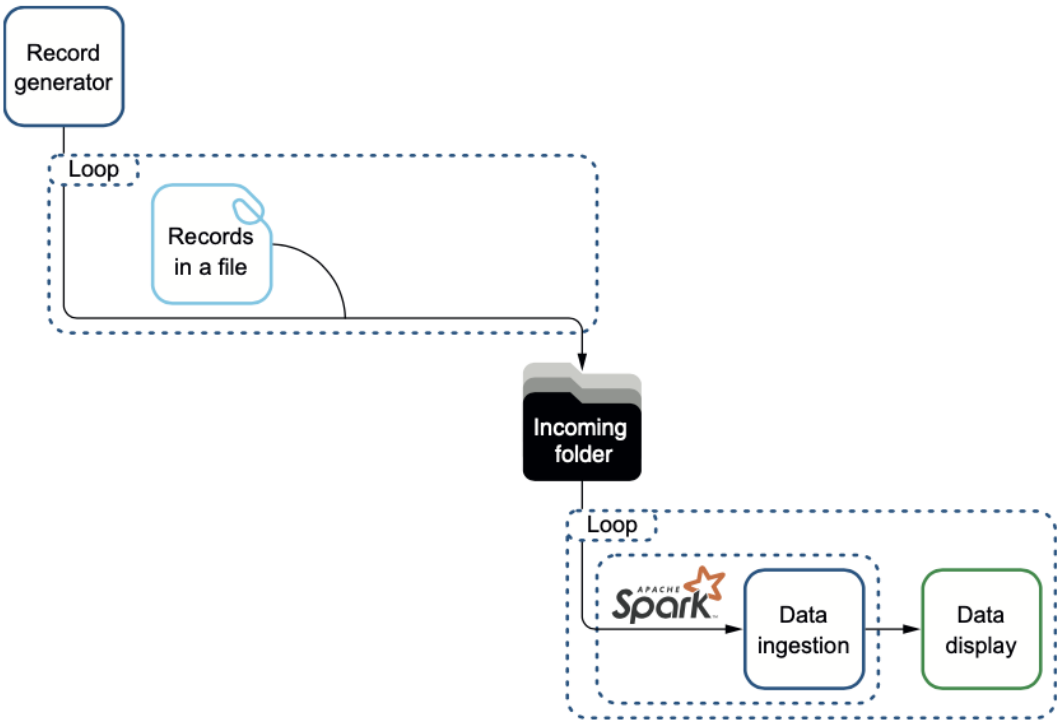
There are two kind of streams network and file. Here we would be emulating file stream. In a file streaming scenario, files are dropped in a directory (sometimes called a landing zone or staging area) and consumer takes the files from this directory as they come in. In the second scenario, data is sent over the network.

In our scenario, we are going to run two applications. The order in which you start the application does not matter for file streams. One application will produce the stream, which contains records describing products. The other application will use Spark to consume the generated stream. We will start with the data generator and then build the consumer. See the architecture below:



In this typical streaming scenario, data is being produced, sent as atomic elements, transformed on the fly, and finally is available in processed form for live reporting, storage, and more.

For part one of the application, you need to create the following design to emulate streaming data:

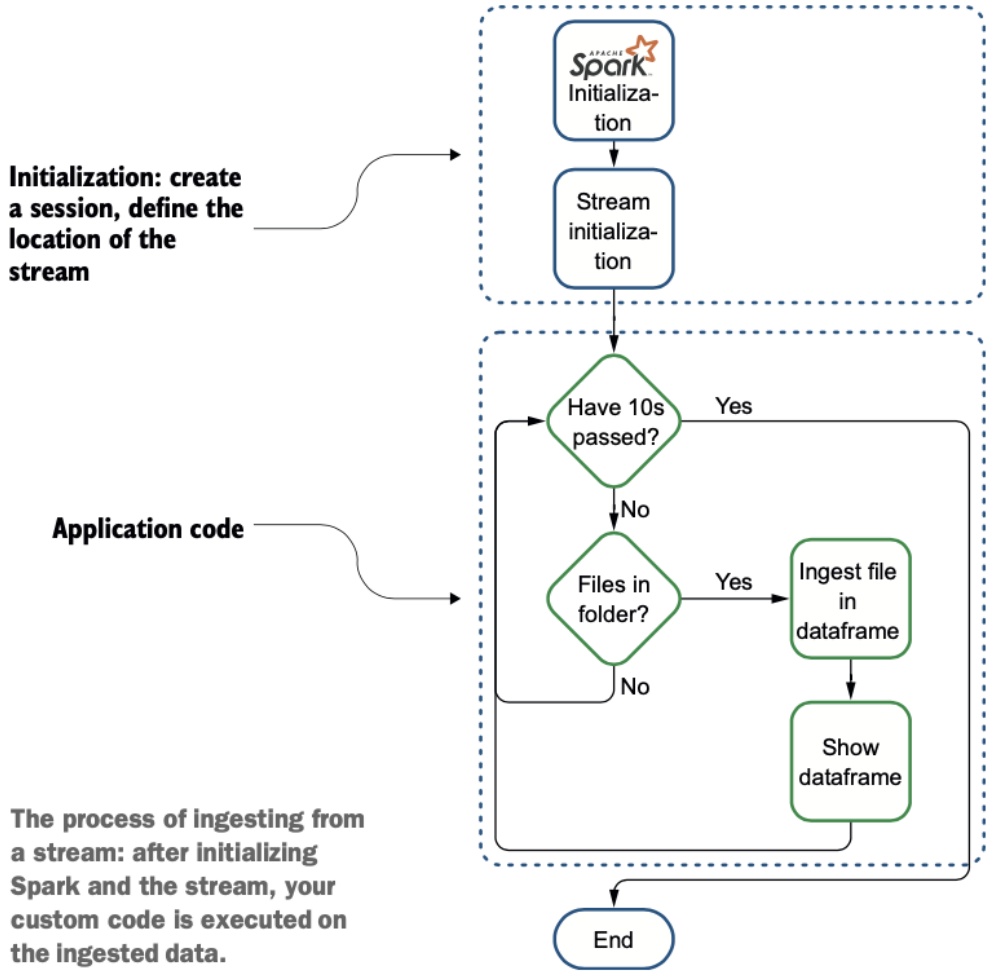


In this streaming scenario, one application (the generator) generates records in a file and places those files in a folder. A Spark application (the consumer) reads the files in that folder and displays the data.

Implementation details of part-1:

1. Your generator continuously loops over the dataset again and in a while loop.
2. Stream duration is 10 seconds
3. Batch size is 10 data points
4. Point 2 & 3 means in 10 seconds generate a batch of 10 records maximum store them in 10 separate files (.txt, csv, xml, json)
5. Wait time of 20 seconds to be implemented i.e., it is the duration the generator waits between two events before generating the next batch.

Implementation details of part-2:



The process of ingesting from a stream: after initializing Spark and the stream, your custom code is executed on the ingested data.

As the data comes in you need to:

1. Follow the above architecture diagram for implementation of your consumer in spark.
2. Read a single batch using spark.
3. Apply transformations on this emulated Realtime data according to the requirements (you will need to, as the data is not in the form that can be useful for different checks).

4. Output the product names that follow the criteria (when the next batch executes the dashboard must refresh).

Sidebar discussion:

The dashboard could be a simple continuous stream of outputs in the console (if you develop your consumer as a terminal application).

All the sanity checks should be in place such as the producer and the consumer are running continuously, the consumer waits and checks whether the directory is filled and has enough datapoints to construct a batch.

Finally, and most importantly, use the wait time in the generator to clear all the files in the directory on the 15^{th} second to make way for the next batch i.e., at any given time only a single batch is present in the directory from where the consumer is reading the files.