

1. Tổng quan về Spark Streaming

Spark Streaming là một phần mở rộng của Spark API core cho phép xử lý luồng dữ liệu trực tiếp có khả năng mở rộng, thông lượng cao, có khả năng chịu lỗi. Spark Streaming được sử dụng trong các trường hợp sử dụng đòi hỏi xử lý nhanh một khối lượng lớn dữ liệu thời gian thực. Ví dụ các trường hợp sử dụng thời gian thực là:

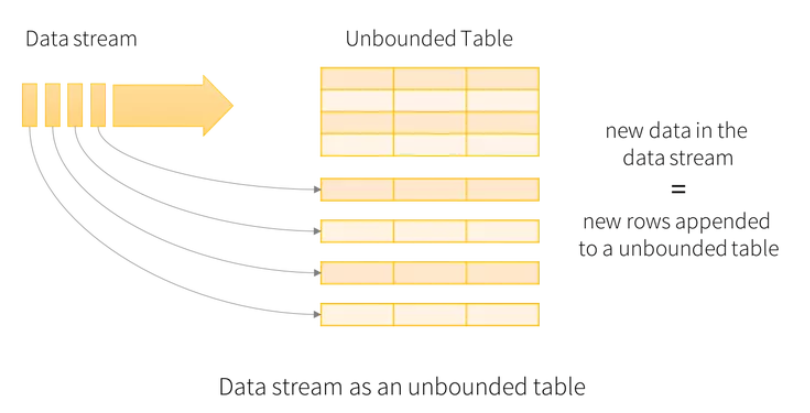
* Giám sát trang web, giám sát mạng
* Phát hiện gian lận
* Nhấp chuột trên web
* Quảng cáo
* Cảm biến Internet of Things

Spark Streaming hỗ trợ các nguồn dữ liệu như thư mục HDFS, Kafka, Flume, Twitter, v.v. Luồng dữ liệu có thể được xử lý bằng APIS lõi của Spark, SQL DataFrames hoặc API học máy và có thể được lưu vào hệ thống tệp, HDFS, cơ sở dữ liệu hoặc bất kỳ nguồn dữ liệu nào cung cấp Hadoop OutputFormat.

1. Tổng quan về Structured Streaming.

Từ phiên bản 2.x của Spark, Structured Streaming được hỗ trợ. Được xây dựng trên Spark SQL. Structured Streaming là một cách khác để xử lý luồng dữ liệu với Spark. Mô hình của luồng dữ liệu dựa trên API Dataframe và Dataset. Từ đó, ta có thể áp dụng bất kỳ truy vấn SQL nào (sử dụng API Dataframe) hoặc các hoạt động Scala (sử dụng API Dataset) trên luồng dữ liệu.

1. So sánh Spark Streaming và Structured Streaming.
   1. Truyền phát dữ liệu thời gian thực

* Spark Streaming: Dữ liệu được vào ra, xử lý theo từng đợt (batch) một. Mỗi batch đại diện cho một RDD.
* Structured Streaming: Đối với Structured Streaming, khái niệm batch được bỏ. Dữ liệu nhận được trong một kích hoạt được gắn với luồng dữ liệu và luồng này chảy liên tục. Mỗi hàng của luồng dữ liệu được xử lý và kết quả được cập nhật vào một bảng không giới hạn. Mỗi kết quả được coi là một hàng của bảng đó.
  1. API.
* Spark Streaming : Hoạt động trên API Dstream sử dụng RDD.
* Structured Streaming : Hoạt động trên API DataFrame và Dataset.

Khi sử dụng API RDD, Spark không tối ưu hóa chuỗi chuyển đổi, không tối ưu hóa truy vấn. Còn đối với API Dataframe, sự trừu tượng được cung cấp ở mức cao hơn, cho phép sử dụng ngôn ngữ truy vấn để thao tác dữ liệu, tận dụng được các chức năng của SQL.

* 1. Xử lý dữ liệu thời gian thực.

Event-time : thời gian mà một sự kiện thực sự xảy ra.

Late Data : dữ liệu được nhận chậm hơn so với thời gian thực sự xảy ra nó.

* Spark Streaming chỉ hoạt động với thời gian khi Spark nhận được dữ liệu. Dựa trên thời gian nhận được dữ liệu, Spark Streaming đặt dữ liệu theo từng batch ngay cả khi dữ liệu này là Late Data mà đúng ra nó phải thuộc về batch trước đó. Điều này dẫn đến việc thông tin kém chính xác do mất dữ liệu.
* Structured Streaming cung cấp chức năng xử lý dữ liệu trên cơ sở event-time một khi event-time được bao gồm trong dữ liệu nhận được.
  1. Đảm bảo đầu cuối.

Mọi ứng dụng đều yêu cầu khả năng chịu lỗi và đảm bảo đầu cuối của việc cung cấp dữ liệu. Bất cứ khi nào ứng dụng bị lỗi, nó phải có khả năng khởi động lại từ cùng một điểm mà nó bị lỗi để tránh mất dữ liệu và trùng lặp. Để cung cấp khả năng chịu lỗi, Spark Streaming và Structured Streaming đều sử dụng điểm kiểm tra để lưu tiến trình của công việc. Nhưng phương pháp này vẫn còn nhiều lỗ hổng có thể gây mất dữ liệu.

Khác với điểm kiểm tra, Structured Streaming đã áp dụng hai điều kiện để khôi phục từ bất kỳ lỗi nào:

* Nguồn phải được phát lại.
* Các sinks phải hỗ trợ các hoạt động bình thường để xử lý lại trong trường hợp thất bại.
  1. Hạn chế và sự linh hoạt.
* Spark Streaming : không có sự hạn chế về bất kỳ loại sinks nào. Nó cung cấp phương thức foreachRDD để thực hiện một số hành động trên luồng (trả về các RDD theo từng batch và có thể lưu trữ RDD cũng như gửi dữ liệu tới nhiều cơ sở dữ liệu).
* Structured Streaming : Cho đến phiên bản 3.3, số lượng sinks đầu ra bị giới hạn. Với mỗi một sink, chỉ có một số thao tác được thực hiện. Ngoài ra, việc lưu trữ dữ liệu đầu ra vào nhiều kho lưu trữ bên ngoài.