Spark Streaming

데이터 처리

- 배치 처리
  - 고정된 과거의 데이터 처리
  - 처리하는 데이터의 크기가 크고 작업 수행시간이 상대적으로 긴 처리
  - 작업 도중 처리에 실패하더라도 재작업을 통해 동일한 최종 결과 재산출 가능

- 실시간 처리
  - 데이터 생성 시점과 처리되는 시점 사이의 간격이 짧은 처리
  - 생성된 후 가능한 빠르게 처리해야 유의미한 결과를 얻을 수 있는 경우

스파크 스트리밍

- 실시간 처리가 필요한 데이터를 다루기 위한 스파크의 서브 모듈

- 실시간으로 변하는 데이터를 (배치 처리보다) 짧은 주기에 맞춰 빠르고 안정적으로 처리하는 데 필요한 기능 제공

- 일정한 주기마다 연속적으로 발생한 새로운 데이터를 읽어서 처리
  - 이전에 발생한 데이터와 새로 발생한 데이터를 결합해서 필요한 처리 수행
  - 이 작업을 애플리케이션이 종료될 때까지 무한 반복

- 배치 처리에 비해 처리 주기 사이의 시간 간격에 대한 정보가 추가로 필요

주요 객체

- StreamingContext 객체
  - RDD와 데이터셋을 사용하기 위해 SparkContext와 SparkSession을 사용한 것처럼 스파크 스트리밍 작업을 수행하기 위해 StreamingContext 객체 필요

```
import org.apache.spark._
import org.apache.spark.streaming._

val ssc = new StreamingContext(sc, Seconds(5))
```

- DStream(Discretized Streams)
  - 고정되지 않고 끊임없이 생성되는 연속된 데이터를 나타내기 위한 추상 모델
  - 일정 시간마다 데이터를 모아서 RDD 생성 → 이 RDD로 만들어진 시퀀스

데이터 소스의 종류

- 기본 데이터 소스
  - 스파크 단독으로 처리할 수 있는 데이터 소스
  - 소켓, 파일, RDD 큐 등

- 어드밴스드 데이터 소스
  - 외부 라이브러리의 도움이 필요한 데이터 소스
  - 카프카, 플럼, 키니시스, 트위터 등

주요 연산

연산	설명
print()	DStream에 포함된 각 RDD의 내용을 콘솔에 출력 기본 10개 → 개수 지정 가능
map(func)	DStream의 RDD에 포함된 각 원소에 func 함수를 적용한 결괏값으로 구성된 새로운 DStream 반환
flatMap(func)	RDD의 flatMap()과 같으며 DStream의 RDD에 포함된 각 원소에 func 함수를 적용 하나의 입력이 0 ~ N개의 출력으로 변환
reduce(func), reduceByKey(func)	DStream에 포함된 RDD 값들을 집계해서 하나의 값으로 변환 RDD가 key ~ value 형식의 튜플인 경우 reduceByKey를 사용해서 key 별로 집계 수행 가능 반환 값은 새로운 DStream
filter	DStream의 모든 요소에 함수를 적용하고 그 결과가 true인 요소만 포함한 새로운 DStream 반환
union	두 개의 DStream 요소를 모두 포함하는 새로운 DStream 생성
join, leftOuterJoin, rightOuterJoin, fullOuterJoin	키와 값 쌍으로 구성된 두 개의 DStream을 키를 이용해 결합(Join)

주요 연산

연산	설명
transform(func)	DStream 내부의 RDD에 func를 적용하고 그 결과로 새로운 DStream 반환
updateStateByKey	새로 생성된 데이터와 이전 배치의 최종 상태 값을 함께 전달해서 현재까지의 누적 연산 수행
window, reduceByWindow, reduceByKeyAndWindow	윈도우 연산. 현재 배치 주기의 데이터와 집계되지 않은 이전 배치 주기의 입력 데이터와 함께 처리 가능
saveAsTextFiles. saveAsObjectFiles, saveAsHadoopFiles	DStream의 데이터를 텍스트, 객체 등의 형식으로 파일에 저장
foreachRDD	DStream에 포함된 각 RDD 별로 연산 수행
CheckPoint	잡이 실행되는 동안 생성된 중간 결과물을 안정성이 높은 저장소에 저장 → 장애 등의 복구에 사용
cache, persist	반복적으로 사용되는 데이터를 저장해두고 재사용