스터디 12.10.(화) 21:30

참고 :

https://nightlies.apache.org/flink/flink-docs-release-1.15/docs/learn-flink/overview/

카프카, 플링크, 스파크, + 에어플로우

스트림 처리의 이해

데이터는 자연스럽게 흐름.

실시간으로 흘러가는 데이터 스트림의 예를 들면

웹사이트에서 발생하는 클릭 로그, 증권거래소의 주식 거래, 공장의 센서 데이터 등

이런 데이터를 처리하는 방식 :

첫 번째는 배치 처리.

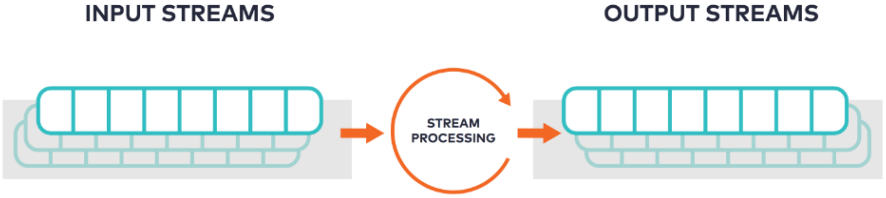
정해진 크기의 데이터를 모아서 한꺼번에 처리하는 방식.

예를 들어, 하루 동안의 판매 데이터를 모아서 밤중에 분석하는 것

→ 이 방식은 실시간성이 떨어짐.

두 번째는 스트림 처리. 데이터가 발생하자마자 처리하는 방식이기에 실시간으로 의미 있는 결과를 도출할 수 있음.

예를 들어, 신용카드 부정 거래를 실시간으로 감지.



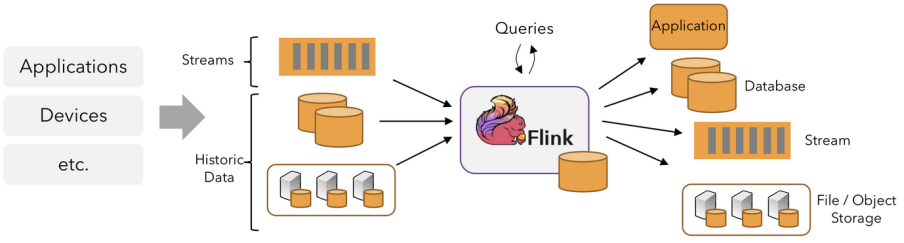
Apache Flink란 무엇인가?

Flink는 실시간으로 들어오는 데이터를 처리하기 위한 분산 처리 시스템

플링크(Flink)는 독일어로 민첩함을 뜻하는 단어. Exactly-once의 이벤트 처리를 보장하는 네이티브 스트림 방식으로, 지연 발생이 적고 처리량은 높으며 비교적 사용하기 쉬운 이점이 있음...

일괄처리 기능도 제공하지만 스트림 프로세싱을 목적으로 주로 사용.

1. **Java**
    - Flink의 주요 개발 언어로, 가장 완벽한 기능 지원
    - DataStream API, DataSet API, Table API 모두 사용 가능
2. **Scala**
    - Java와 동일한 수준의 기능 지원
    - 간결한 문법으로 코드 작성 가능
3. **Python (PyFlink)**
    - Table API와 DataStream API 지원
    - 일부 기능에 제한이 있을 수 있음



유사한 빅데이터 처리 프레임워크

Spark vs Flink

핵심 차이점:

- **처리 방식**
  - Spark: 미니 배치 처리 방식으로 작은 단위로 모아서 처리
  - Flink: 리얼 스트리밍 처리로 데이터를 즉시 처리
- **지연시간(지연 시간이 낮을수록 데이터 전송이 빠르다고 보면 될 것.)**
  - Spark: 수초 단위 지연 (미니배치 처리 때문)
  - Flink: 밀리초(1초 X 천 분의 1) 단위 지연 (실시간 처리)
- **상태 관리**
  - Spark: RDD를 통한 상태 관리
  - Flink: 더 효율적인 내장 상태 관리 시스템

적합한 사용 사례

- Spark: 대규모 데이터 분석, 머신러닝, 배치 처리
- Flink: 실시간 처리, 이벤트 기반 애플리케이션, 연속 ETL

  → 특히, 사기 탐지, 이상 탐지, 실시간 UX 개인화,품질 모니터링, 라이브 데이터의 임시 분석

선택 기준

1. 실시간성이 매우 중요한 경우 → Flink
2. 배치 처리가 주요한 경우 → Spark
3. 기존 시스템과의 통합이 중요한 경우 → Spark (더 큰 생태계)

Flink는 설정이 더 간단하고 강력한 윈도우 연산자를 제공, Spark는 더 넓은 생태계와 통합이 용이한 장점이 있음.

Flink 핵심 특징

1. **Exactly-once(정확하게 한번) 처리 보장**

- 데이터 손실이나 중복 처리 없음
- 장애 상황에서도 정확한 처리 보장

2. **장애 복구 메커니즘**
    - Checkpoint: 주기적인 상태 저장
    - Savepoint: 수동으로 생성하는 백업 포인트
    - 장애 발생 시 최근 상태에서 자동 복구
3. **상태 관리 방식**
    - 내부 상태를 효율적으로 관리
    - 메모리나 디스크에 상태 저장 가능
    - 분산 환경에서도 일관성 유지
4. **확장성 (Scalability)**
    - 클러스터 환경에서 수평 확장 가능해서 대규모 데이터 처리에 적합.

어느 BE 엔지니어 : “바퀴가 이미 있는데, 바퀴를 새로 만들 필요가 없다.”

→ 공식 Docker 이미지 활용

2가지 이유

1. 안정성

: 공식 이미지들은 각자 맡은 팀에서 직접 관리하고 테스트 하고 있고,

버전 관리가 잘 되어 있기에 필요한 버전을 쉽게 취사선택 가능

2. 효율성

: 환경 설정 시간을 줄이고 실질적으로 다른 곳에 리소스 등을 집중할 수 있고, 설정 오류나 호환성 및 의존성 문제를 막을 수 있다. 또한 이렇게 실무적으로 많이 사용하고 있기에 이렇게 사용하는 경험도 중요하다.

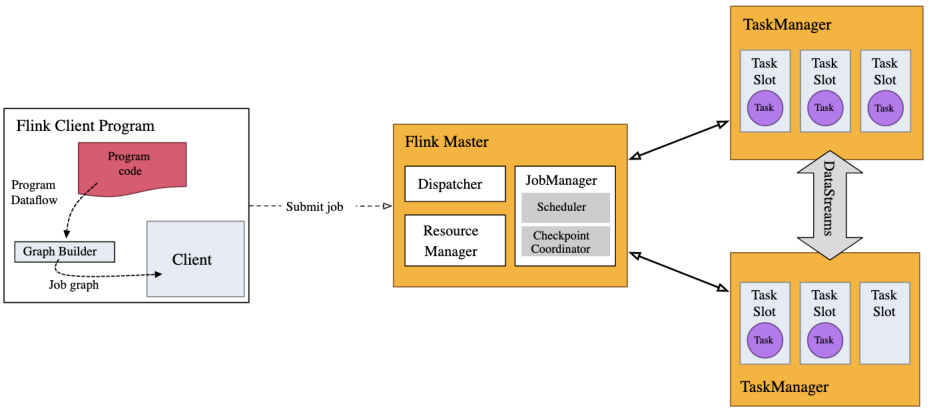
(+ 대규모 데이터 처리를 하지 않도록 노력해야 한다고도 얘기해주셨음. 글로벌 캐싱 전략(레디스 등), 로컬 캐싱 전략(코드 단) , 허용되지 않는 요청 거부 등을 통해서 노력하고, 그래도 대규모 데이터 처리를 해야한다면 고려해야 하는 것들.)

플링크 설치를 위한 예시 코드

```
services:
  jobmanager:
    image: apache/flink:1.16.1
    ports:
      - "8081:8081"
    command: jobmanager
    environment:
      - JOB_MANAGER_RPC_ADDRESS=jobmanager

  taskmanager:
    image: apache/flink:1.16.1
    depends_on:
      - jobmanager
    command: taskmanager
    environment:
      - JOB_MANAGER_RPC_ADDRESS=jobmanager
```

1. **JobManager (작업 관리자)**
   - 전체 작업 감독
   - TaskManager에 작업 분배
   - 웹 UI(8081 포트)로 모니터링 제공
2. **TaskManager (작업 수행자)**
   - 실제 데이터 처리 수행
   - 여러 개 실행 가능
   - JobManager와 지속적 통신



주요 포트

- 8081: 웹 대시보드 포트

- 6123: JobManager RPC 포트

→ Remote Procedure Call, JobManager가 TaskManager들과 통신할 때 사용하는 포트

→ 작업 할당, 상태 업데이트, 하트비트 체크 등의 내부 통신에 사용

→ 예시: JobManager가 TaskManager에게 새로운 작업을 할당하거나, TaskManager가 작업 상태를 보고할 때 사용

- 6124: BlobServer 포트

→ 대용량 파일이나 객체(Binary Large Object)를 전송할 때 사용하는 포트

  - JAR 파일 전송 (Flink 작업을 포함한 실행 파일)
  - 큰 설정 파일 전송
  - TaskManager들 간의 대용량 데이터 전송

→ 예시: 새로운 Flink 작업을 제출할 때 JAR 파일을 각 TaskManager에 배포하는 데 사용

- 6125: QueryableState 포트
- Flink 작업의 현재 상태를 외부에서 조회할 때 사용하는 포트
- 실시간으로 작업의 상태를 확인하고 싶을 때 사용
- 예시:
  - 현재 처리 중인 데이터의 상태 확인
  - 특정 키에 대한 최신 집계 결과 조회
  - 디버깅이나 모니터링 목적으로 상태 정보 조회

1. **애플리케이션 제출**: 사용자가 작성한 Flink 애플리케이션은 Client를 통해 JobManager에게 제출됩니다.
2. **작업 계획 생성**: JobManager는 애플리케이션을 실행 계획(Execution Plan)으로 변환하고, 이를 실행 그래프(Execution Graph)로 변환합니다.

3. **태스크 할당**: JobManager는 Execution Graph를 여러 태스크(Task)로 나누어 TaskManager에게 할당합니다.
4. **태스크 실행**: TaskManager는 할당받은 태스크를 실행하며, 태스크 간 데이터를 주고받습니다.
5. **상태 관리**: 상태 저장이 필요한 태스크는 State Backend를 통해 상태를 관리합니다.
6. **체크포인트**: 주기적으로 체크포인트를 생성하여 장애 복구를 대비합니다.
7. **장애 복구**: 장애 발생 시, 체크포인트를 이용해 작업을 복구하고 실행을 재개합니다.
8. **결과 출력**: 처리된 데이터는 최종적으로 싱크(Sink)에 저장되거나 다른 시스템으로 전달됩니다.

Flink 대시보드 띄우기는 추후에.