

삼성청년 SW·AI아카데미

관통프로젝트
배치처리 워크플로우 파이프라인 구축
(Airflow + Spark)

배치처리 워크플로우 파이프라인 구축 (Airflow + Spark)

챕터의 포인트

- 관통 프로젝트 안내

관통 프로젝트 안내

• 목표

- 이번 프로젝트의 목표는 Airflow를 활용한 스케줄링(Workflow Orchestration)과 Spark를 이용한 배치 처리를 구현하는 것입니다.
(예시: 매일 새벽 1시)에 실행되는 배치 작업을 통해, 전일 콘텐츠 수집 및 사용자 별 혹은 사용자 전체의 로그 데이터를 집계/분석하고, 키워드 트렌드나 카테고리별 통계를 산출하여 리포트를 자동 생성합니다.
- 배치가 완료된 후에는 원본 데이터(JSON 등)를 별도의 아카이빙 폴더로 이동하여, 전체 데이터 관리가 용이하도록 설계합니다. (이후 HDFS를 다루게 되면 HDFS 형태로 저장)
- Airflow DAG를 구성하여 배치 스케줄링(예시: 매일 새벽 1시 실행)할 수 있다.
- Spark를 활용하여 전처리/분석 작업을 수행할 수 있다.
- 전일 데이터를 집계 후, 트렌드/키워드 등의 리포트 PDF를 자동 생성할 수 있다.

• 준비사항

• 사용 데이터

- 로컬 파일 시스템에 저장된 데이터 (예: ../realtime/*.json) - 이후 hdfs로 이관
- Spark가 읽어올 JSON 형식(ex. 뉴스 내용, 작성일시, 키워드 정보 등)

• 개발언어/프로그램

- Python: 스크립트(배치 로직, Spark 처리 등)
- Apache Spark: 데이터프레임, RDD 기반 분석 및 시각화(로컬 파일 시스템에서 데이터 로드)
- Apache Airflow: 배치 스케줄링, Workflow Orchestration
- Matplotlib: 리포트용 시각화(차트, 그래프)
- 파일 및 디렉터리 관리: Python 표준 라이브러리(os, shutil) 활용

• 구현 방법

1) Airflow DAG 구성

- 주기적으로 특정 시점에 실행되는 DAG를 구성 (예시: 매일 새벽 1시)
- SparkSubmitOperator로 Spark 배치 스크립트 실행
- 분석 결과물(차트, PDF 리포트)을 특정 폴더에 저장

• 구현 방법

2) Spark 배치 처리

- 전 시점에 해당하는(EX. 전일 0시~24시) 데이터만 필터링하여 키워드, 트렌드 등을 집계
- Matplotlib 등 라이브러리를 통해 차트 시각화 및 리포트에 들어갈 내용 작성 후, PDF 형태 리포트를 자동 생성

• 구현 방법

3) 파일 아카이빙

- 분석에 사용된 JSON 파일들을 작업이 끝나면 ../long_term_archive 디렉터리로 이동 (이후 HDFS로 장기저장소로 변경할 것)
- Python의 os, shutil 라이브러리를 사용하여 파이썬 스크립트를 이용해 이동 처리
- 로그 또는 Airflow 태스크를 통해 이동 성공 여부 기록

• 구현 방법

4) 리포트 관리

- 생성된 daily_report_YYYYMMDD.pdf(또는 PNG, HTML 등)은 로컬 파일 시스템에 저장
- 필요 시 EmailOperator를 통해 보고서를 메일로 전송하거나, 협업 툴(MatterMost 등)에 업로드될 수 있도록 구성

• 관통 프로젝트 가이드

1) PJT 내용

- 데이터 수집
 - Kafka + Flink로 실시간 데이터를 로컬 디렉토리(../realtime/*.json)에 저장
- Spark 배치 처리
 - Airflow 스케줄러가 (예시: 매일 새벽 1시) 특정 시간에 Spark 스크립트를 실행
 - 전일 데이터(전날 0시~24시)만 필터링하여 키워드, 트렌드 등을 분석
- 리포트 생성
 - 상위 10개 키워드, 트렌드 등 차트 시각화 및 분석 리포트 생성 -> PDF로 저장
 - 파일명을 날짜 기준으로 생성(예시: daily_report_20251205.pdf)
- 데이터 아카이빙
 - 분석에 사용된 원본 JSON 파일을 ../long_term_archive 디렉토리로 이동
 - Airflow에서 테스크 순서대로 실행(분석 -> 이동 -> 완료)

• 관통 프로젝트 가이드

```
import pendulum
from datetime import datetime, timedelta
from airflow import DAG
from airflow.operators.bash import BashOperator

local_tz = pendulum.timezone("Asia/Seoul")

default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
}

with DAG(
    dag_id='daily_report_dag',
    default_args=default_args,
    description='매일 새벽 1시에 Spark를 이용해 뉴스 리포트 생성',
    schedule_interval='0 1 * * *', # 한국 시간 기준 새벽 1시
    start_date=datetime(2025, 2, 10, tzinfo=local_tz),
    catchup=False,
    tags=['daily', 'report', 'spark']
) as dag:
```

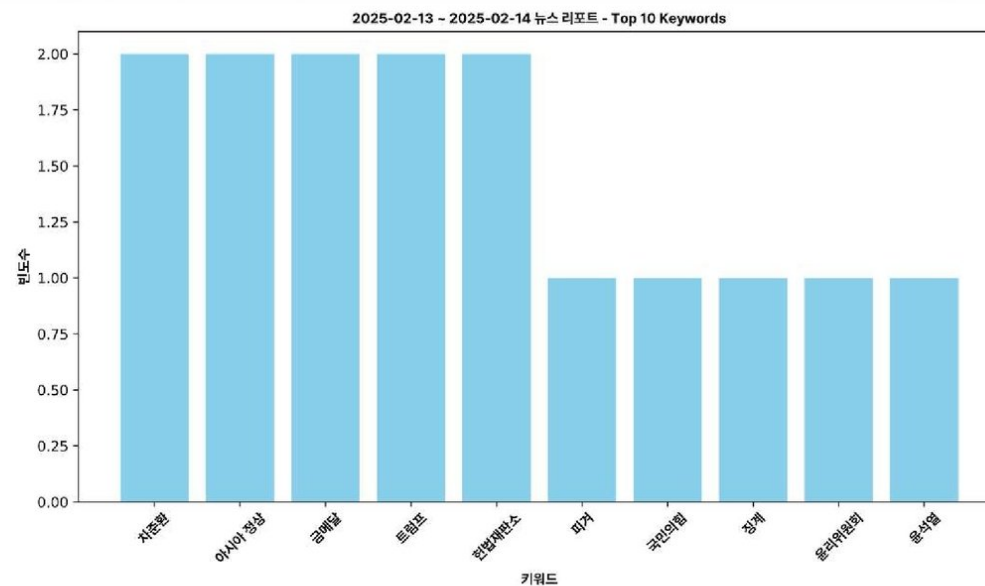
```
submit_spark_job = BashOperator(
    task_id='spark_daily_report',
    bash_command=(

    )
)

notify_report_generated = BashOperator(
    task_id='notify_report_generated',
    bash_command=(

    )
)

submit_spark_job >> notify_report_generated
```



요구사항

• 기본 기능

- Airflow + Spark를 이용한 배치 파이프라인 구축
 - 매일 특정 시간(예시: 자정, 새벽 1시 등)에 자동 실행
 - Spark로 로컬 디렉터리의 JSON 데이터를 필터링, 집계 분석

• 요청 조건

- Airflow DAG를 작성하여 정해진 스케줄에 자동 실행
- Spark 스크립트를 통해 전일 데이터만 집계/분석
- 키워드, 트렌드 통계 등 기본 로직 포함

• 결과

- 배치 완료 후 콘솔 or Airflow 로그에서 집계 결과 확인
- Airflow UI에서 DAG 성공/실패 상태 관리

요구사항

- 원본 데이터 아카이빙 + 2차 가공
 - 배치 처리 완료 후, 원본 파일을 ../long_term_archive 디렉터리로 이동
 - (선택) DB 적재 등 2차 가공 프로세스 확장을 통한 서비스에 필요한 데이터 이동
- 요청 조건
 - Python 표준 라이브러리(shutil.move) 등을 활용해 파일 이동
 - ETL 작업 확장을 위한 구조(예: DB 적재, 피처 엔지니어링 등)
- 결과
 - 배치 반복 시 ../realtime 디렉터리는 항상 최근 데이터만 유지
 - ../long_term_archive에 과거 데이터가 쌓여 관리

요구사항

- 트렌드 분석 및 PDF 리포트 자동 생성
- 요청 조건
 - 하루 치 데이터 집계 후, 시각화 (Ex. 상위 키워드 TOP 10 등)
 - PDF 파일 내부에 차트/그래프와 분석 내용이 포함될 수 있도록 구성
- 결과
 - 리포트 파일(예시: daily_report_20251205.pdf) 자동 생성
 - 정해진 디렉터리에 PDF 결과 저장

산출물

- Gitlab에 올라온 de-project 코드 기반의 레포지토리 지속적 커밋
- 이후 PJT도 해당 레포지토리에 이어 나가면서 개발

정리

구현 기능	비고
Airflow DAG 스케줄링	DAG구성, 스케줄링 설정, 작업 흐름 구성
데이터 아카이빙	파일 이동 처리 및 로컬 파일 적재
Spark 배치 분석	전일 데이터 필터링, 키워드/트렌드 집계, 통계적 분석 등
PDF 리포트 생성	Matplotlib 등 시각화 도구 활용 시각화 및 텍스트 작성을 고려한 적절한 리포트 생성
전체 통합 및 2차 가공 등 안정성 여부	DAG 전체 실행의 안정성 및 결과물 확인 가능 여부

내일 방송에서 만나요!

삼성청년SW·AI아카데미