포팅매뉴얼(데이터서버)

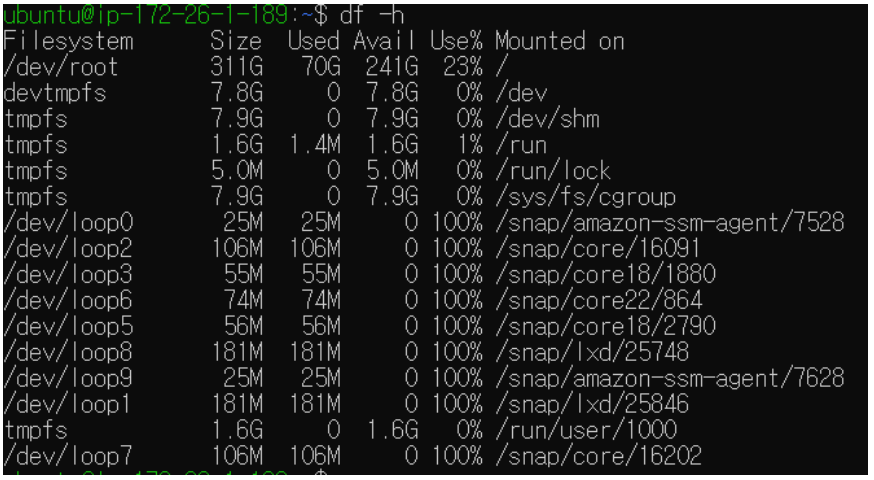
🛠️개발환경

서버 인스턴스 사양

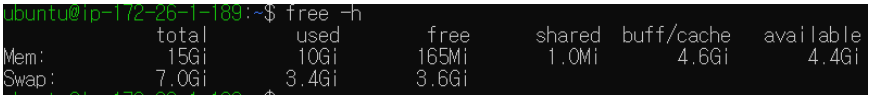
- CPU 정보: Intel Xeon(R) Core 4개

- RAM : 16GB

- Disk : 300GB



- 현재 메모리 사용량



OS

- Ubuntu 20.04

이슈 관리

- Jira

Communication

- Mattermost
- Notion

DB

- MongoDB 4.4.24

API Server

- FastAPI

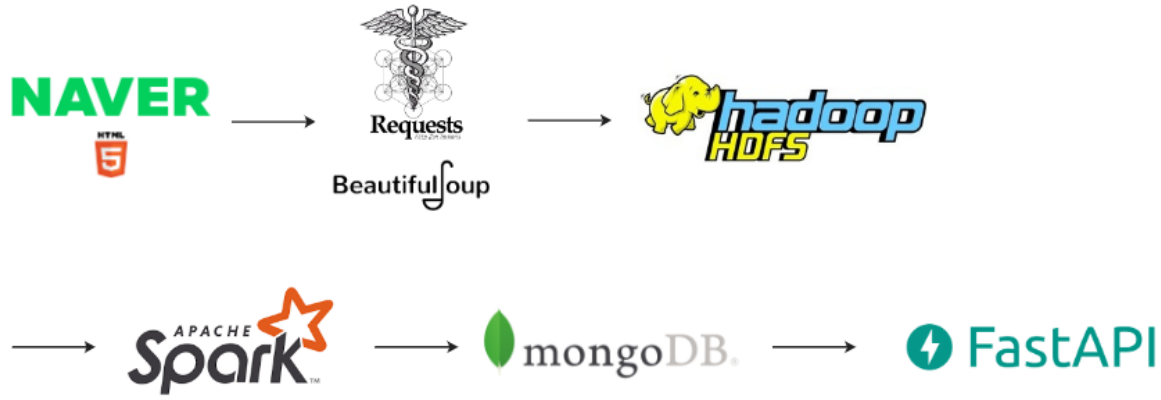
IDE

- Visual Studio Code : 1.75.0

Infra

- Web Server : Nginx

시스템 아키텍처



🔨배포 과정

📂빌드 환경설정

Hadoop 설정

참고

하둡 공식 문서

하둡 설치

```
wget [https://dlcdn.apache.org/hadoop/common/stable/hadoop-3.2.4.tar.gz]
(https://dlcdn.apache.org/hadoop/common/stable/hadoop-3.3.6.tar.gz)
```

`tar -xzvf hadoop-3.2.4.tar.gz`

하둡 설정(Pseudo-distributed Mode)

hadoop root 디렉터리로 이동해서 진행

- `etc/hadoop/core-site.xml`

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://localhost:9000</value>
    </property>
</configuration>
```

- `etc/hadoop/hdfs-site.xml`

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
</configuration>
```

- `etc/hadoop/mapred-site.xml`

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <name>mapreduce.application.classpath</name>
        <value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/share/
hadoop/mapreduce/lib/*</value>
    </property>
</configuration>
```

- `etc/hadoop/yarn-site.xml`

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
```

```
        <name>yarn.nodemanager.env-whitelist</name>
        <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH
_PREPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_HOME,PATH,LANG,TZ,HADOOP_MAPRED_HOME</value
>
    </property>
</configuration>
```

- `etc/hadoop/hadoop-env.sh` (하둡에 자바 환경변수 적용)

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

SSH 키 생성

원격 서버 액세스 전용(의사 분산 모델에서는 그저 `localhost`

- SSH키 생성 및 인증 설정

`ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa` `cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys` `chmod 0600 ~/.ssh/authorized_keys`

- 원격 서버 접속

`ssh localhost`

MapReduce Job 로컬 테스트

- 파일 시스템 포맷(최초 1회만 진행)

`bin/hdfs namenode -format`

- 네임노드 데몬 실행

`sbin/start-dfs.sh`

`sudo ufw allow 9870` 후 `http://localhost:9870/`로 이동하면 네임노드 대시보드로 이동 가능

- HDFS 디렉터리 생성

`bin/hdfs dfs -mkdir -p /user/ubuntu`

- HDFS에 input file 삽입

`bin/hdfs dfs -mkdir input`

`bin/hdfs dfs -put etc/hadoop/*.xml input`

- 샘플 파일 실행

`bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.4.jar grep input output 'dfs[a-z.]+'`

- output 확인

`bin/hdfs dfs -get output output`

`cat output/*`

- 네임노드 데몬 종료

`sbin/stop-dfs.sh`

Input 파일 저장

Hadoop root 디렉터리에서 진행

- hdfs 디렉터리 생성

`bin/hdfs -mkdir -p /user/<유저네임>`

`bin/hdfs dfs -mkdir input`

- 파일 저장

`bin/hdfs dfs -put <저장할 파일 위치> <저장할 hdfs 디렉터리>`

- 파일 생성시 safemode 오류

`put: Cannot create file/user/ubuntu/input/articles.csv.*COPYING*. Name node is in safe mode.`

→ namenode가 safemode에 있기 때문에 발생

`bin/hdfs dfsadmin -safemode leave` safemode 탈출

`bin/hdfs dfsadmin -safemode forceExit` safemode 강제 탈출

`bin/hdfs dfsadmin -safemode get` safemode 상태 확인

탈출 시 시간이 소요될 수 있음(기다려야 함)

탈출 후에도 지속적으로 safemode에 진입하는 경우, hdfs 파일 간 무결성에 문제가 있을 수 있음(로그 확인 및 `hdfs-site.xml` 파일 확인 요)

`bin/hdfs dfsadmin -safemode enter` safemode로 진입

→ 포맷

- 파일 저장 위치 확인

`bin/hdfs dfs -ls` 현재 hdfs 디렉터리 확인

`bin/hdfs dfs -ls <디렉터리>` 해당 디렉터리의 상태를 확인

위의 경우 `/user/ubuntu/input`에 input 파일이 들어가게 됨

`hdfs getconf -confKey fs.defaultFS`실행중인 namenode의 주소 확인

PySpark 실행 오류

- no live nodes

실행중인 노드가 없을 경우 → Missing block의 수 확인 후 블록 복구 시행

`hdfs dfsadmin -report` 노드 상태 확인

`hdfs dfsadmin -recoverBlocks` missing blocks 복구

`hdfs balancer` under replicated block 복구(클러스터 블록 효율적 분산)

- 인코딩

json 이용할 경우 인코딩 문제 해결

엥간하면 csv 말고 json 사용

- 가용한 네임노드가 부족할 때

`hdfs-site.xml` 에서 replication value를 1로 설정해줌

```
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
```

- PySpark 코드에서 namenode에 접근할 수 없을 때

`FileNotFoundError: [Errno 2] No such file or directory: 'hdfs://localhost:9000/user/ubuntu/article_data.json'`

- hdfs 명령어 자체가 실행 안될 때(ls, report 등 안먹힐때)

포맷

FastAPI

환경설정

`pip install fastapi[all]` fastapi

`pip install uvicorn` fastapi 실행 서버

`pip install pymongo` 몽고디비 연동

`pip install pydantic` 데이터 검증 패키지

가상환경

`python3 -m venv myenv` 가상환경 구성

`source myenv/bin/activate` 가상환경 작동

Run

```
uvicorn main:app --host 0.0.0.0 --port 8000 --reload
```

- 코드 내에 `app = FastAPI()` 를 선언해줘야함
- 모든 소스로부터 접근 허용
- 8000번 포트로 실행
- 백그라운드 실행

```
nohup uvicorn main:app --host 0.0.0.0 --port 8000 --reload > uvicorn.log 2>&1 &
```

Airflow

환경설정

```
pip3 install apache-airflow
```

```
cd airflow
```

```
airflow db init
```

```
mkdir dags
```

```
airflow users create -u admin -p admin -f MOA -l MORE -r Admin -e admin@admin.com
```

- 터미널 2개에서 각각 실행

```
airflow webserver -p 8080
```

```
airflow scheduler
```

이제 airflow 웹서버에 접속 가능해야 함

- 로그인 정보

ID: admin

PW: admin