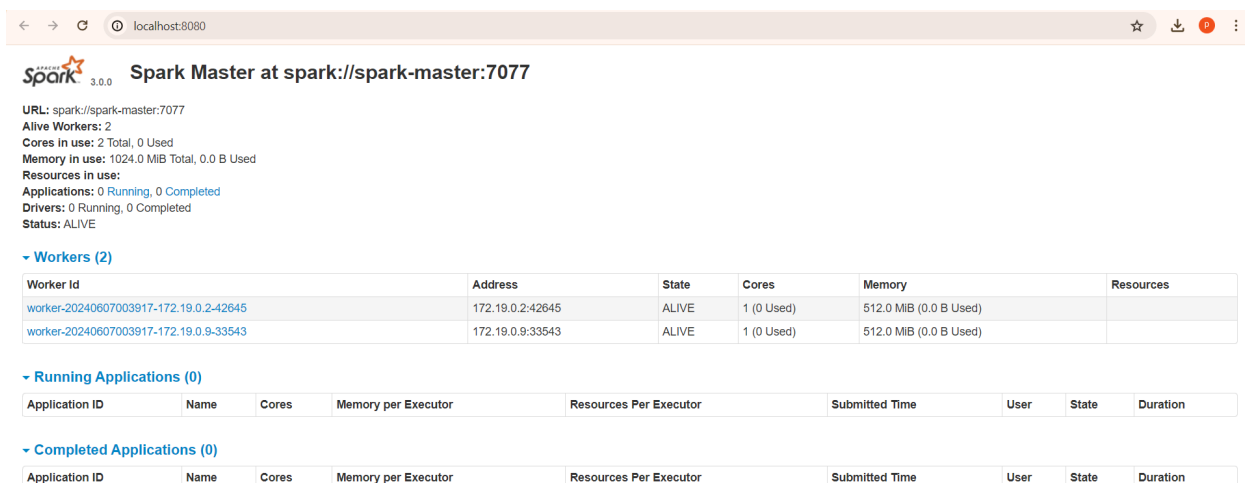


گزارش تمرین سوم فاز 0 و 1

پریا مهدیان 9931079

• نمایش UI برای Hadoop و Spark و Jupyter



Spark Master at spark://spark-master:7077

URL: spark://spark-master:7077
Alive Workers: 2
Cores in use: 2 Total, 0 Used
Memory in use: 1024.0 MIB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (2)

| Worker Id | Address | State | Cores | Memory | Resources |
|--|------------------|-------|------------|------------------------|-----------|
| worker-20240607003917-172.19.0.2-42645 | 172.19.0.2:42645 | ALIVE | 1 (0 Used) | 512.0 MIB (0.0 B Used) | |
| worker-20240607003917-172.19.0.9-33543 | 172.19.0.9:33543 | ALIVE | 1 (0 Used) | 512.0 MIB (0.0 B Used) | |

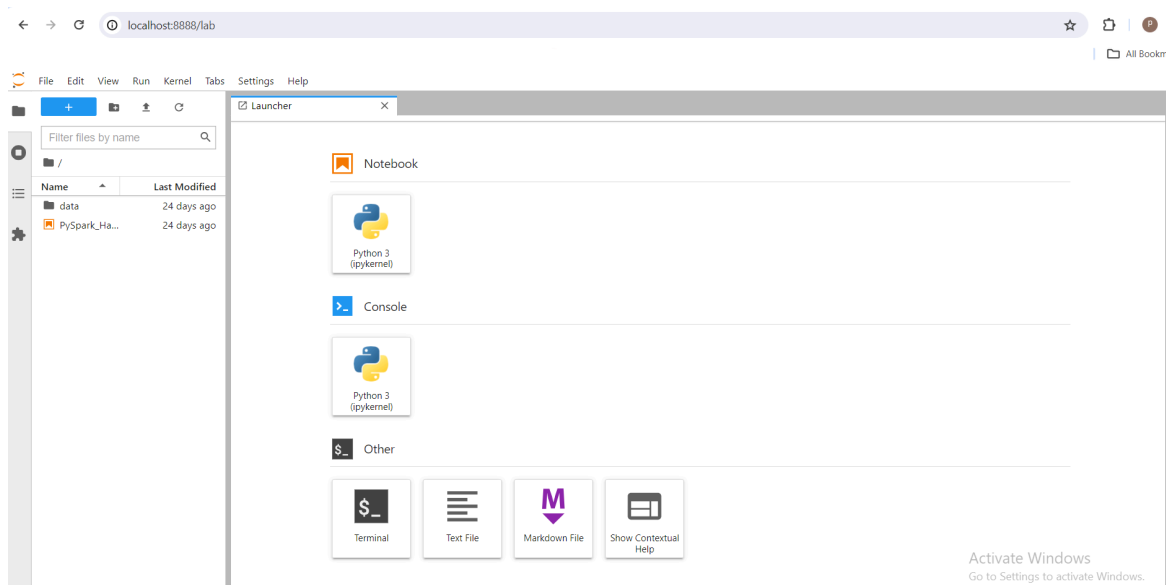
Running Applications (0)

| Application ID | Name | Cores | Memory per Executor | Resources Per Executor | Submitted Time | User | State | Duration |
|----------------|------|-------|---------------------|------------------------|----------------|------|-------|----------|
|----------------|------|-------|---------------------|------------------------|----------------|------|-------|----------|

Completed Applications (0)

| Application ID | Name | Cores | Memory per Executor | Resources Per Executor | Submitted Time | User | State | Duration |
|----------------|------|-------|---------------------|------------------------|----------------|------|-------|----------|
|----------------|------|-------|---------------------|------------------------|----------------|------|-------|----------|

تعداد گره های **Spark**: دو گره Spark worker در کلاستر داریم که با "spark-worker-1" و "spark-worker-2" نشان داده شده است.



JupyterLab interface showing the Launcher area. The Launcher area contains the following options:

- Notebook
- Python 3 (ipykernel)
- Console
- Python 3 (ipykernel)
- Other
- Terminal
- Text File
- Markdown File
- Show Contextual Help

Activate Windows
Go to Settings to activate Windows.

Overview 'hadoop-namenode:9000' (✓active)

| | |
|----------------|--|
| Started: | Tue May 21 23:36:53 -0700 2024 |
| Version: | 3.3.6, r1be78238728da9266a4f88195058f08fd012bf9c |
| Compiled: | Sun Jun 18 01:22:00 -0700 2023 by ubuntu from (HEAD detached at release-3.3.6-RC1) |
| Cluster ID: | CID-bee0ffe8-897b-481e-bf8e-161a47702406 |
| Block Pool ID: | BP-1600706988-172.18.0.8-1716359811310 |

Summary

Security is off.

Safemode is off.

15 files and directories, 7 blocks (7 replicated blocks, 0 erasure coded block groups) = 22 total filesystem object(s).

Heap Memory used 69.7 MB of 273 MB Heap Memory. Max Heap Memory is 1.27 GB.

Non Heap Memory used 51.57 MB of 53.38 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

| | |
|----------------------|------------|
| Configured Capacity: | 1006.85 GB |
|----------------------|------------|

امنیت: ویژگی های امنیتی در حال حاضر در محیط Hadoop خاموش هستند.

SafeMode: حالت امن، حالتی که در آن NameNode وظایف تعمیر و نگهداری را انجام می دهد، در حال حاضر غیرفعال است.

فایل ها و دایرکتوری ها: در کل 15 فایل و دایرکتوری در فایل سیستم Hadoop وجود دارد.

مجموع بلوک ها: 7 بلوک در سیستم موجود است.

Replication: از بین 7 بلوک، هر 7 بلوک تکراری هستند.

گروه های بلوک کد شده پاک سازی: در حال حاضر، 0 گروه بلوک کد شده پاک سازی وجود دارد.

اشیاء سیستم فایل: در مجموع 22 شیء سیستم فایل شامل فایل ها و دایرکتوری ها وجود دارد.

حافظه Heap استفاده شده: 69.7 مگابایت حافظه heap در حال حاضر استفاده می شود.

حداکثر حافظه Heap: حداکثر حافظه Heap اختصاص داده شده روی 1.27 گیگابایت تنظیم شده است.

● نمایش کانتینرهای ایجاد شده

```
C:\Windows\System32\cmd.exe
=> CACHED [jupyter-notebook 2/16] RUN mkdir -p /opt/workspace/data 0.0s
=> CACHED [jupyter-notebook 3/16] RUN mkdir -p /usr/share/man/man1 0.0s
=> CACHED [jupyter-notebook 4/16] RUN apt-get update -y 0.0s
=> CACHED [jupyter-notebook 5/16] RUN apt-get install -y curl python3 r-base 0.0s
=> CACHED [jupyter-notebook 6/16] RUN ln -s /usr/bin/python3 /usr/bin/python 0.0s
=> CACHED [jupyter-notebook 7/16] RUN curl https://downloads.lightbend.com/scala/2.12.10/scala-2.12.10.deb -k - 0.0s
=> CACHED [jupyter-notebook 8/16] RUN apt install -y ./scala.deb 0.0s
=> CACHED [jupyter-notebook 9/16] RUN rm -rf scala.deb /var/lib/apt/lists/* 0.0s
=> CACHED [jupyter-notebook 10/16] COPY workspace/ /workspace/ 0.0s
=> CACHED [jupyter-notebook 11/16] RUN apt-get update -y 0.0s
=> CACHED [jupyter-notebook 12/16] RUN apt-get install -y python3-pip python3-dev 0.0s
=> CACHED [jupyter-notebook 13/16] RUN pip3 install --upgrade pip 0.0s
=> CACHED [jupyter-notebook 14/16] RUN pip3 install wget==3.2 pyspark==3.0.0 jupyterlab==3.0.0 0.0s
=> CACHED [jupyter-notebook 15/16] RUN pip3 install pandas 0.0s
=> CACHED [jupyter-notebook 16/16] WORKDIR /workspace 0.0s
=> [jupyter-notebook] exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:29132af008cadbe4058db18321458bb9adae080c423c0990f6e9a12f4a9f83c7 0.0s
=> => naming to docker.io/library/hind-main-jupyter-notebook 0.0s
=> [spark-worker-1 internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 310B 0.0s
=> [spark-worker-1 internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [spark-worker-2 internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 310B 0.0s
=> [spark-worker-1] exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:78745f4b5b700a558e4f44b594917103f7f824812685efef3c7f2abc1157767a 0.0s
=> => naming to docker.io/library/hind-main-spark-worker-1 0.0s
=> [spark-worker-2 internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [spark-worker-2] exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:c14f0c3f78800bc09cec9755ccb7b579f315ec81ccd29399b5f11de6e3b06955 0.0s
=> => naming to docker.io/library/hind-main-spark-worker-2 0.0s
[+] Running 9/9
  Container spark-master      Started 1.4s
  Container spark-worker-2    Started 2.0s
  Container spark-worker-1    Started 1.8s
  Container hadoop-nodemanager-1 Started 0.8s
  Container hadoop-datanode    Started 0.7s
  Container hadoop-historyserver Started 1.1s
  Container hadoop-namenode    Started 1.3s
  Container jupyter-notebook   Started 0.9s
  Container hadoop-resource-manager Started 0.5s
```

● توضیح وظایف هر کدام از کانتینرهای ایجاد شده

:spark-master

این کانتینر معمولاً میزبان سرویس Spark Master است که اجرای کلی یک برنامه Spark را کنترل می کند. وظایف را در Spark Workers هماهنگ می کند و داده ها را در سراسر کلاستر توزیع می کند.

spark-worker-1 و 2:

این کانتینرها معمولاً میزبان خدمات Spark Worker هستند که وظیفه اجرای وظایف محول شده توسط Spark Master را بر عهده دارند. آنها محاسبات را انجام می دهند و داده ها را برای برنامه های Spark ذخیره می کنند.

:hadoop-nodemanager-1

این کانتینر معمولاً سرویس NodeManager را در یک کلاستر Hadoop اجرا می کند. NodeManager مسئول مدیریت منابع روی یک گره و اجرای وظایف اختصاص داده شده توسط ResourceManager است.

:hadoop-datanode

این کانتینر سرویس DataNode را در یک کلاستر Hadoop اجرا می کند. DataNode داده ها را در HDFS ذخیره می کند و به درخواست ها برای عملیات خواندن/نوشتن داده ها پاسخ می دهد.

:hadoop-historyserver

این کانتینر معمولاً میزبان JobHistory سرور در یک کلاستر Hadoop است. JobHistory Server اطلاعات تاریخی مربوط به کارهای تکمیل شده MapReduce را برای اشکال زدایی و تجزیه و تحلیل ذخیره می کند.

:hadoop-namenode

به عنوان سرور اصلی برای سیستم فایل توزیع شده HDFS (Hadoop) عمل می کند. NameNode ابرداده های HDFS مانند سلسله مراتب سیستم فایل و اطلاعات فراداده را حفظ می کند و دسترسی به داده های ذخیره شده در DataNodes را هماهنگ می کند.

:jupyter-notebook

این کانتینر سرور Jupyter Notebook را میزبانی می کند و به کاربران اجازه می دهد با کد پایتون، داده ها، visualizations و متن تعامل داشته باشند.

:hadoop-resourcemanager

این کانتینر معمولاً سرویس ResourceManager را در یک کلاستر Hadoop میزبانی می کند. ResourceManager مسئول مدیریت منابع در سرتاسر کلاستر و برنامه ریزی برنامه ها بر اساس منابع موجود است.

• کد mapper

```
#!/usr/bin/env python
import sys
```

```
for line in sys.stdin:
```

```
# ورودی را به id document و text تقسیم میکنیم.
```

```
document_id, text = line.strip().split(',', 1)
```

```
# متن را کلمه کلمه میکنیم
```

```
words = text.split()
```

```
# به ازای تمامی کلمات موجود در words کلمه را همراه با id چاپ میکنیم
```

```
for word in words:
```

```
    print(f'{word},{document_id}')
```

• کد reducer

```
#!/usr/bin/env python
import sys
```

```
current_word = None
```

```
current_docs = set()
```

```

def output_result(word, documents):
    # هر کلمه را همراه با document id یکتا چاپ میکنیم.
    print(f"{word}\t{','.join(documents)}")

for line in sys.stdin:
    # ورودی را تجزیه یا parse میکنیم و به word و document id تقسیم می کنیم.
    word, document_id = line.strip().split(',', 1)

    # بررسی می کنیم که آیا word تغییر کرده است یا نه
    if current_word != word:
        if current_word:
            output_result(current_word, current_docs)
            current_word = word
            current_docs = set()

    # به مجموعه، document id را اضافه میکنیم
    current_docs.add(document_id)

    # خروجی نتایج برای کلمه آخر
    if current_word:
        output_result(current_word, current_docs)

```

از دستور زیر برای قرار دادن فایل ورودی در hdfs استفاده میکنیم.

```
hdfs dfs -put input.txt /input/input.txt
```

ساختار دایرکتوری HDFS مستقیماً به عنوان پوشه های داخل فهرست پروژه قابل مشاهده نیست. این یک سیستم فایل توزیع شده است که توسط سرویس های Hadoop مانند namenode و datanode مدیریت می شود.