یاسمن گودرزی ۹۹۳۱۱۰۰

تمرین سری سوم فاز ۰ و ۱

نمایش کانتینرهای ایجاد شده:

توضیح وظایف هر یک از کانتینر های ایجاد شده:

کانتینر spark:

این کانتینر برای اجرای کدها و عملیات‌های مربوط به برنامه spark است. از وظایف آن می‌توان به اجرای کدهای نوشته شده به زبان برنامه‌نویسی اشاره کرد که شامل عملیات‌های مورد نظر ما بر روی پردازش داده‌های بزرگ است.

از دیگر وظایف آن می‌توان به مدیریت منابع اشاره کرد. این فرایند شامل تخصیص حافظه، پردازش داده‌ها و دیگر منابع است.

اجرای عملیات پردازش بر روی منابع مختلف با استفاده از کانتینرهای مجازی شده اسپارک ممکن است. در واقع این کانتینر عملیات‌های پیاده‌سازی شده را انجام میدهد.به طور کلی، کانتینرهای Spark مسئول اجرای و مدیریت برنامه‌ها و منابع مورد نیاز آنها برای اجرای پردازش‌های توزیع شده در سیستم Spark هستند.

کانتینر hadoop:

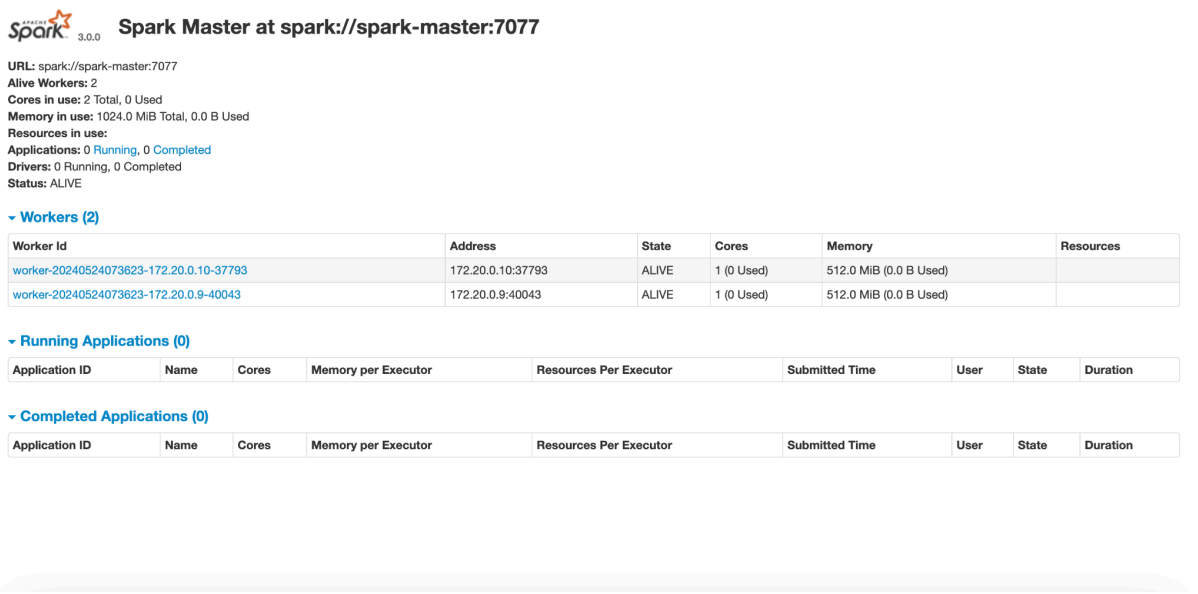
این کانتینر عملیات مختلفی که به پردازش داده‌ها و مدیریت منابع مربوط است را انجام می‌دهد. این کانتینر هم یکی از ابزار مدیریت و پردازش داده است. از وظایف آن میتوان اشاره کرد:

اجرای کدها و وظایف نوشته شده در MAPREDUCE در واقع این کانتینر کدهای نوشته شده در mapper و reducer را اجرا می‌کند. از طرفی مدیریت منابع و و ذخیره‌سازی داده‌ها در فضای ذخیره‌سازی توزی شده توسط این کانتینر انجام می‌شود. در نهایت نتیجه میگیریم که این کانتینر مسئول ایجاد یک محیط ایزوله و مجازی برای اجرای تسک‌ها و سرویس های مربوط به هدوپ است.

کانتینر Jupyter :

یک محیط اجرایی مجازی است که اجرای سریع و آسان برنامه‌های Jupyter Notebook و JupyterLab را فراهم می‌کند. در واقع این کانتینر به کاربران این امکان را میدهند تا کدهای مبتنی بر python و دیگر زبان‌های برنامه نویسی را در یک محیط ایزوله اجرا کنند. علاوه بر این از امکان تحلیل داده ها و پردازش داده ها را فراهم میکنند. در واقع این کانتینر یک محیط ایزوله با دیباگر ها و کامپایل های مرتبط با زبان برنامه نویسی را فراهم میکنند تا کاربران بتوانند کدهای دلخواه خود را اجرا کنند و نتایج را ببینند و در یک محیط امن آن ها را اشتراک بگذارند.

تصویر ui :



Spark 3.0.0 **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)

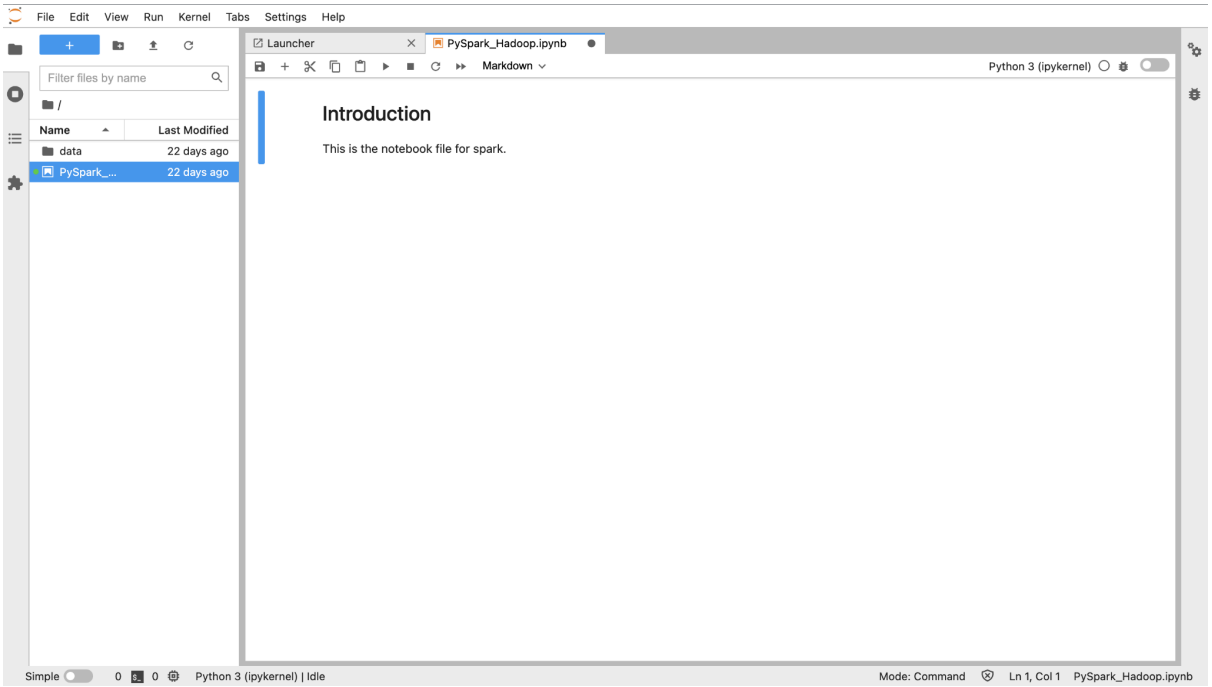
Worker Id	Address	State	Cores	Memory	Resources
worker-20240524073623-172.20.0.10-37793	172.20.0.10:37793	ALIVE	1 (0 Used)	512.0 MiB (0.0 B Used)	
worker-20240524073623-172.20.0.9-40043	172.20.0.9:40043	ALIVE	1 (0 Used)	512.0 MiB (0.0 B Used)	

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



Non DFS Used:	7.34 GB
DFS Remaining:	48.03 GB (82.29%)
Block Pool Used:	92.05 KB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	1 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	9
Number of Blocks Pending Deletion (including replicas)	0
Block Deletion Start Time	Mon May 20 21:50:59 +0800 2024
Last Checkpoint Time	Mon May 20 21:51:00 +0800 2024
Enabled Erasure Coding Policies	RS-6-3-1024k

با توجه به تصویر منابع استفاده شده یک گیگ مموری و دو هسته توسط دوتا ورکر استفاده شده است.

همچنین برای تعداد نودها دو تا نود با یک ادرس و دو پورت متفاوت داریم. وضعیت این دوتا ورکر alive است.

بخش دوم

```python
#!/usr/bin/env python
"""mapper.py - This mapper script reads text input from standard input, in the format "document_id,text".
It processes each line to extract words and emits each word along with the document ID it appeared in.
Each word is emitted only once per document to ensure uniqueness.
"""
import sys
for line in sys.stdin:
    document_id, text = line.strip().split(',', 1)
    words_list = text.split(" ")
    for word in words_list:
        print(f"{word.strip()},{document_id.strip()}")
```

در کد نوشته شده در مپر ابتدا ورودی از فایل گرفته میشود و سپس با , رشته رو جدا میکنیم که بخش اول آن مربوط به شناسه داکیومنت و بخش دوم کل جمله نوشته شده است.

سپس روی یک اسپیس اسپلیت رو انجام میدهیم تا لیست کلمات داخل یک فایل در بیاید و در یک حلقه فور به صورت خروجی یک مجموعه دوتایی از لغت و شناسه فایل را نشان می دهیم.

```python
#!/usr/bin/env python
"""reducer.py - Processes key-value pairs received from the mapper, where each key is a word
and each value is a document ID where the word appeared. This script aggregates the document IDs
for each word and outputs the word alongside the set of unique document IDs where the word was found.
The reducer reads lines of input from standard input, where each line has a format of 'word,document_id'.
The input lines are expected to be sorted by the word. The reducer uses this sorted order to efficiently
aggregate document IDs by using a set data structure, transitioning between different words as it processes
the input lines one-by-one.
"""

import sys
current_word = None
current_documents = set()
for line in sys.stdin:
    word, document_id = line.strip().split(',', 1)
    if current_word == word:
        current_documents.add(document_id)
    else:
        if current_word:
            print(f"{current_word}\t{', '.join(current_documents)}")
        current_word = word
        current_documents = set([document_id])
if current_word:
    print(f"{current_word}\t{', '.join(current_documents)}")

```

در ردیوسر ورودی های تولید شده در مپر را گرفته و در یک حلقه کلمه وشناسه فایل را در یک لیست ذخیره میکنیم. اگر کلمه که داریم همان کلمه قدیمی باشه به شناسه های آن ، شناسه فایل را اضافه میکنیم در غیر اینصورت اون کلمه رو با شناسه جدی به لیست اضافه میکنیم

خروجی:

```
yasaman@yasaman-VivoBook-ASUSLaptop-X509JP-R521JP:~/hadoop/hind$ cat ./input.txt|./mapper.py| sort -k1,1|./reducer.py
5g      doc16
accessibility   doc9
advanced        doc25
advancements    doc22
advances        doc10
agricultural    doc10
analysis        doc3, doc20, doc21
analytics       doc11, doc1, doc18, doc8
and     doc9, doc6, doc1, doc5, doc12, doc8, doc25, doc13, doc16, doc10, doc19, doc2, doc7, doc21, doc4, doc23, doc20
are     doc14, doc21
artificial      doc22, doc2
automates       doc12
automation      doc15
autonomous      doc2
behavior        doc8
benefits        doc23
big     doc8
blockchain      doc7
business        doc21
change  doc3
climate doc3
cloud   doc13
collection      doc12
combat  doc3
complex doc25, doc4
computing       doc25, doc13, doc4
connectivity    doc16
consumer        doc8
critical        doc14
cryptocurrency  doc7
cybersecurity   doc14
data    doc11, doc14, doc9, doc1, doc13, doc18, doc22, doc4, doc23, doc5, doc6, doc15, doc7, doc3, doc17, doc12, doc19, doc2, doc24, doc20, doc8, doc25, doc16, doc10, doc21
decisions       doc24
digital doc14
driven  doc6, doc15, doc24
drives  doc2
e-commerce      doc20
educational     doc6
education       doc18
energy  doc24
engineering     doc10
enhances        doc7, doc16, doc4
environmental   doc17
environments    doc5
evolves doc22
expand  doc18
experiences     doc6, doc20
```