Методичні вказівки

до виконання лабораторної роботи №4

"Використання засобів для високошвидкісної обробки, потокової аналітики та інтеграції даних"

з курсу

“Обробка надвеликих масивів даних”

Київ – 2020

Ціль роботи:

Навчитись налаштовувати систему потокової обробки та менеджменту даних та інтегрувати її із «виробниками» та «споживачами» інформації. Навчитись працювати з застуванням Apache Kafka.

Зрозуміти принципи роботи Apache Kafka та можливості застосування для надвеликих масивів даних.

Завдання:

1. Запустити Apache Kafka у середовищі контейнеризації, як приклад Docker.
2. Написати «Виробника» даних, який передаватиме дані через Apache Kafka.
3. Написати «Споживача» даних, який зчитуватиме дані з Apache Kafka.
4. Дати відповіді на контрольні запитання.

Варіанти

Типи використання Kafka:

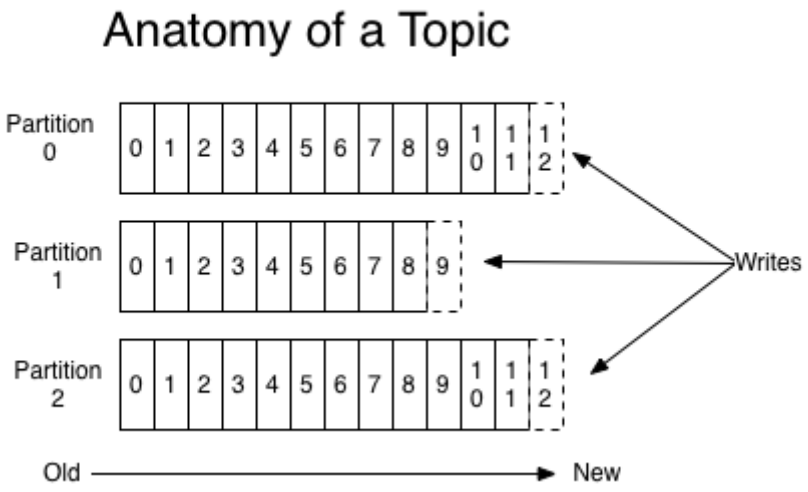
1. Логування: виробник передав повідомлення через Apache Kafka, споживач отримав через Apache Kafka і зберіг повідомлення;

2. Підрахунок метрик: виробник передає час обробки одного клієнта через Apache Kafka, споживач отримує ці дані і обраховує середній час обробки всіх клієнтів;

3. Посередник передачі повідомлень: виробник надсилає команду, а споживач отримує її і виконують певну операцію (приклад: виробник - "show hello world", споживачі - "print hello world")

Номер	Тип використання Kafka	Тип компанії	К-сть тем (Topic)	К-сть розділів (Partition)
1	1	Інтернет магазин	3	1

2	2	IT компанія	1	2
3	3	Теплова електростанція	2	1
4	1	Тютюнова фабрика	3	2
5	2	Телеком компанія	1	1
6	3	Логістична компанія	2	2
7	1	Машинобудівна компанія	3	1
8	2	Компанія виготовлення програмного забезпечення	2	2
9	3	Компанія виготовлення апаратного забезпечення	2	1



Короткі теоретичні відомості:

Apache Kafka - це розподілена потокова платформа, яка складається із серверів та клієнтів та дозволяє:

- Публікувати та підписуватись на потоки записів, подібні до черги повідомлень або корпоративної системи обміну повідомленнями
- Зберігати потоки записів стійким до відмов довговічним способом.
- Обробляти потоки записів у міру їх надходження.

Kafka офіційно підтримує такі мови програмування:

- C/C++
- Go
- Java
- .NET
- Python
- Scala

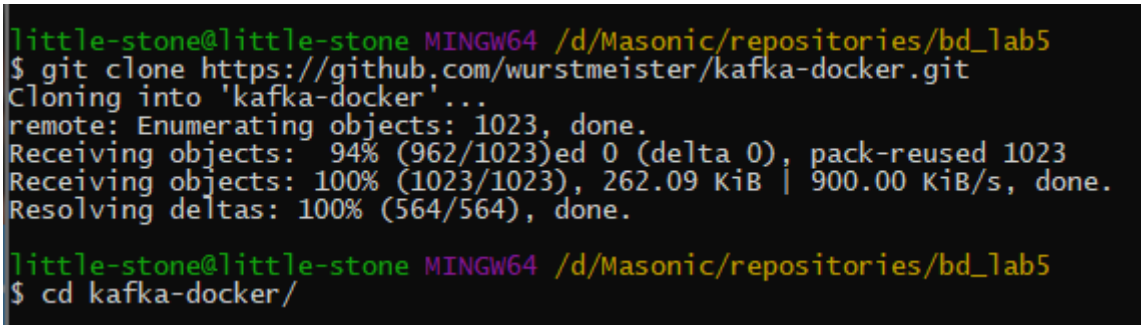
Деякі сервери називаються брокерами або посередниками, і вони утворюють рівень зберігання. Інші сервери запускають Kafka Connect для імпорту та експорту даних як потоків подій, щоб інтегрувати Kafka з існуючою системою у безперервному режимі.

З іншого боку, клієнти дозволяють створювати програми, які читають, записують та обробляють потоки подій. Клієнтом може бути виробник або споживач . Виробник пише (виробляє) події в Kafka, тоді як споживач читає та обробляє (споживає) події з Kafka.

План виконання роботи із прикладом:

1. Встановити Apache Kafka використовуючи Docker.

   Для цього можна скористатися готовими Docker Image.

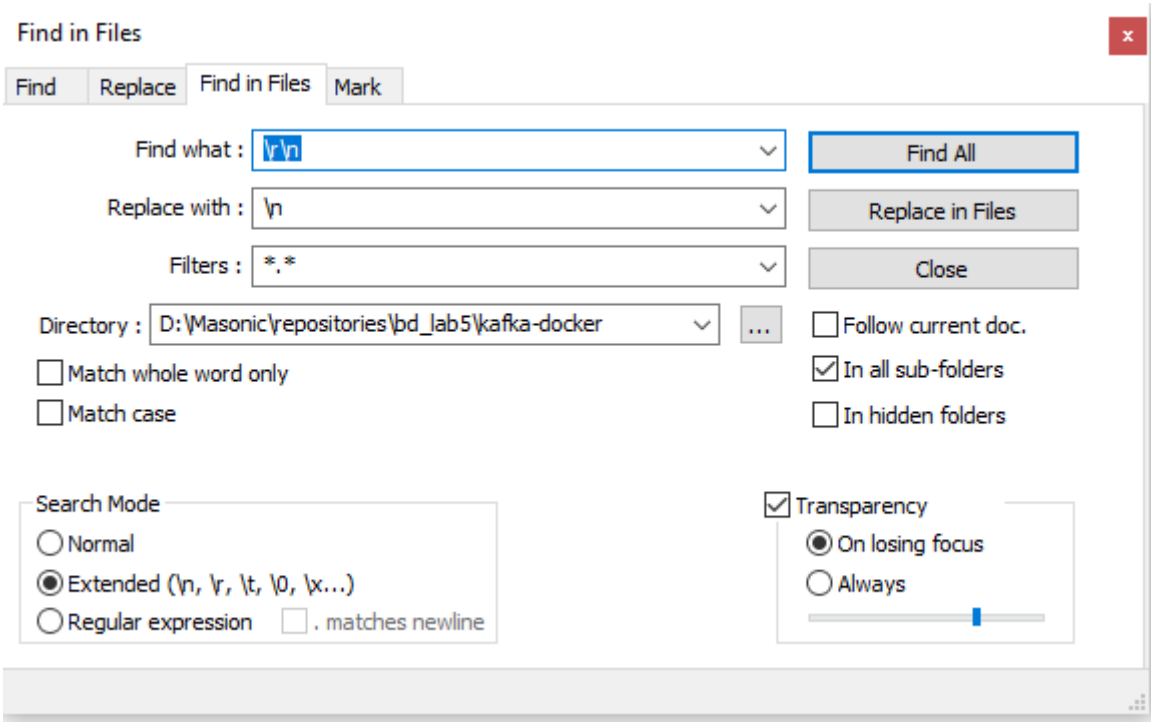


2. Створюємо документ **docker-compose-expose.yml –** визначає сервіси, мережі та томи.

```
version: '2'
services:
  zookeeper:
    image: wurstmeister/zookeeper:3.4.6
    ports:
     - "2181:2181"
  kafka:
    build: .
    ports:
     - "9092:9092"
    expose:
     - "9093"
    environment:
      KAFKA_ADVERTISED_LISTENERS:
INSIDE://kafka:9093,OUTSIDE://localhost:9092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP:
INSIDE:PLAINTEXT,OUTSIDE:PLAINTEXT
      KAFKA_LISTENERS: INSIDE://0.0.0.0:9093,OUTSIDE://0.0.0.0:9092
      KAFKA_INTER_BROKER_LISTENER_NAME: INSIDE
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
      KAFKA_CREATE_TOPICS: "topic_test:1:1"
    volumes:
     - /var/run/docker.sock:/var/run/docker.sock
```

3. Перед запуском контейнера на Windows на основі завантаженого репозиторію потрібно відформатувати символи закінчення рядка із «\r\n» на «\n»для всіх Shell Scripts. Для цього можна встановити Notepad++ і виконуємо наступні кроки
   1) Menu: Search -> Find in Files...
   2) Directory = the directory you want to be converted to Unix format, recursively.
   3) Find what = \r\n
   4) Replace with = \n
   5) Search Mode = Extended
   6) Press "Replace in Files"

   Має вийти наступна картина:

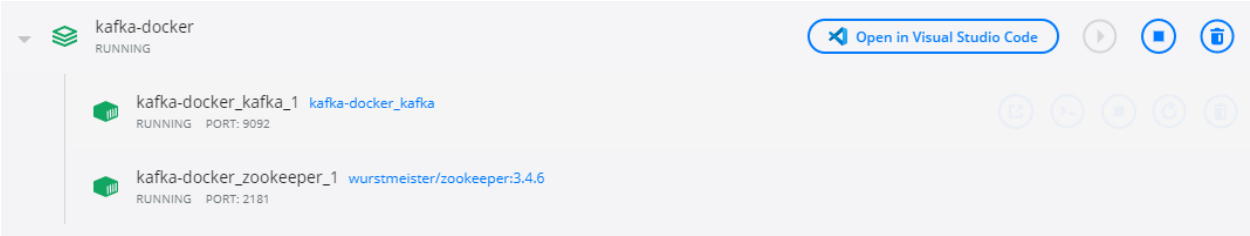


4. Виконуємо наступну команду для запуску контейнера

```
docker-compose -f docker-compose-expose.yml up
```

Результат виконання команди:

```
D:\Masonic\repositories\bd_lab5\kafka-docker>docker-compose -f docker-compo
se-expose.yml up
Creating network "kafka-docker_default" with the default driver
Creating kafka-docker_zookeeper_1 ... done
Creating kafka-docker_kafka_1     ... done
```



5. Для програм написаних на Python використовується бібліотека “kafka-python” (не використовувати бібліотеку «kafka»).

```
pip install kafka-python
```

6. Створюємо «Виробника» даних

```
from time import sleep
from json import dumps
from kafka import KafkaProducer


producer = KafkaProducer(
    bootstrap_servers=['localhost:9092'],
    value_serializer=lambda x: dumps(x).encode('utf-8')
)


for j in range(9999):
    print("Iteration", j)
    data = {'counter': j}
    producer.send('topic_test', value=data)
    sleep(0.5)
```

7. Створюємо «Споживача» даних

```
from kafka import KafkaConsumer
from json import loads
from time import sleep


consumer = KafkaConsumer(
    'topic_test',
    bootstrap_servers=['localhost:9092'],
    auto_offset_reset='earliest',
    enable_auto_commit=True,
    group_id='my-group-id',
    value_deserializer=lambda x: loads(x.decode('utf-8'))
)


for event in consumer:
    event_data = event.value
    # Do whatever you want
    print(event_data)
    sleep(2)
```

8. Запускаємо «Виробника» та «Споживача» та результат роботи:

```
D:\Masonic\repositories\bd_lab5>python py/producer.py
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7
Iteration 8
Iteration 9
Iteration 10
Iteration 11
```

```
D:\Masonic\repositories\bd_lab5>python py/consumer.py
{'counter': 0}
{'counter': 1}
{'counter': 2}
{'counter': 3}
{'counter': 4}
{'counter': 5}
{'counter': 6}
{'counter': 7}
{'counter': 8}
{'counter': 9}
{'counter': 10}
{'counter': 11}
```

Контрольні запитання:

1. Що таке Apache Kafka?
2. Типи API у Apache Kafka?
3. Структура запису у потоці даних Apache Kafka?
4. На якому протоколі реалізована комунікація між вузлами системи Apache Kafka?
5. Як досягається підвищений рівень відмовостійкості при обробці даних?
6. У якій послідовності «Споживачі» можуть обробляти дані?
7. Яку роль виконують “Partitions of a Topic” Apache Kafka?
8. Як реалізована розподіленість Apache Kafka?
9. Як відбувається синхронізація серверів Apache Kafka, «виробників» та «споживачів»?
10. Що таке принцип відсутності копіювання (англ. zero-copy)? Як він використовується у Apache Kafka.
11. Сфери застосування Apache Kafka?

Список посилань:

1) Офіційна документація Apache Kafka – https://kafka.apache.org/20/documentation.html
2) Мануал по швидкому старті із Apache Kafka - https://kafka.apache.org/quickstart

3) Мануал по швидкому старті із Apache Kafka на Docker - https://kafka.apache.org/quickstart-docker
4) How to install Kafka using Docker - https://itnext.io/how-to-install-kafka-using-docker-a2b7c746cbdc
5) Running Kafka using Docker - https://blog.k2datascience.com/running-kafka-using-docker-332207aec73c
6) Kafka-python - https://kafka-python.readthedocs.io/en/master/usage.html
7) Стаття опису прикладу зображеному у ході роботи - https://towardsdatascience.com/kafka-docker-python-408baf0e1088
8) Стаття з описом зв'язків вузлів у прикладі зображеному у ході роботи - https://github.com/wurstmeister/kafka-docker/wiki/Connectivity