



Universidad Nacional de Asunción
Facultad Politécnica
Diplomado Big Data y Business Analytics

Tarea Data Streaming

Unidad V: Procesamiento de Flujo de Datos Masivos
Profesor: Rodrigo Manuel Parra Zacarías
Alumna: Claudia Noemí Palacios Romero – CI: 3.710.469

Identificación

Unidad V: Procesamiento de Flujo de Datos Masivos

Curso: Big Data y Business Analytics

Objetivo

El objetivo de esta tarea es fijar los contenidos desarrollados durante el módulo de Data Streaming, incluyendo la aplicación de herramientas como Redpanda y KSQLDB para la solución de un problema.

Instrucciones

La tarea consiste en construir un servicio que consuma la API en tiempo real de Finnhub: <https://finnhub.io/docs/api/websocket-trades> y consuma actualizaciones para los siguientes símbolos:

- AAPL
- AMZN
- BINANCE:BTCUSDT

De modo a procesar las actualizaciones, deben seguirse los siguientes pasos:

1. Instalar un cluster Redpanda de manera local utilizando un archivo docker-compose.yml
2. Implementar un producer utilizando kafka-python, de acuerdo a la documentación de Redpanda y similar al ejemplo desarrollado en clase, para suscribirse a los eventos de la API.
3. Instalar KSQLDB modificando el archivo docker-compose.yml, de acuerdo a la documentación de Redpanda.
4. Ejecutar ksqldb-cli, definir los streams y tablas necesarios para responder a las siguientes preguntas:
 1. ¿Cuál fue el promedio ponderado de precio de una unidad por cada uno de los símbolos procesados? (e.j. AAPL)
 2. ¿Cuántas transacciones se procesaron por símbolo?
 3. ¿Cuál fue el máximo precio registrado por símbolo?
 4. ¿Cuál fue el mínimo precio registrado por símbolo?

Presentación

El proyecto debe desarrollarse en los grupos previamente formados. El entregable debe incluir:

- Enlace a repositorio público de Github incluyendo:
 - docker-compose.yml
 - código fuente del producer
 - archivos .sql con las 4 consultas utilizadas para responder las preguntas de la sección anterior
- Una grabación de pantalla mostrando el sistema en funcionamiento y explicando brevemente los pasos que se siguieron para su implementación.

Fecha límite de entrega: 21-08-2023 23:59

Archivo: docker-compose.yml contruido en función a los requerimientos planteados

```
---
version: "3.7"
name: redpanda-cpalacios
networks:
  redpanda_network-1:
    driver: bridge
volumes:
  redpanda-1: null
services:
  redpanda-1:
    command:
      - redpanda
      - start
      - --kafka-addr internal://0.0.0.0:9092,external://0.0.0.0:19092
      - --advertise-kafka-addr internal://redpanda-1:9092,external://localhost:19092
      - --pandaproxy-addr internal://0.0.0.0:8082,external://0.0.0.0:18082
      - --advertise-pandaproxy-addr internal://redpanda-1:8082,external://localhost:18082
      - --schema-registry-addr internal://0.0.0.0:8081,external://0.0.0.0:18081
      - --rpc-addr redpanda-1:33145
      - --advertise-rpc-addr redpanda-1:33145
      - --smp 1
      - --memory 1G
      - --mode dev-container
      - --default-log-level=debug
    image: docker.redpanda.com/redpandadata/redpanda:v23.2.6
    container_name: redpanda-1
    volumes:
      - redpanda-1:/var/lib/redpanda/data
    networks:
      - redpanda_network-1
    ports:
      - 18081:18081
      - 18082:18082
      - 19092:19092
      - 19644:9644

  console-1:
    container_name: redpanda-console-1
    image: docker.redpanda.com/redpandadata/console:v2.3.1
    entrypoint: /bin/sh
    command: -c 'echo "$${CONSOLE_CONFIG_FILE}" > /tmp/config.yml; /app/console'
    environment:
      CONFIG_FILEPATH: /tmp/config.yml
      CONSOLE_CONFIG_FILE: |
        kafka:
          brokers: ["redpanda-1:9092"]
          schemaRegistry:
            enabled: true
            urls: ["http://redpanda-1:8081"]
        redpanda:
          adminApi:
            enabled: true
            urls: ["http://redpanda-1:9644"]
    depends_on:
      - redpanda-1
    networks:
      - redpanda_network-1
    ports:
      - 8080:8080

  ksqldb-server-1:
    image: confluentinc/ksqldb-server:0.25.1
    hostname: ksqldb-server-1
    container_name: ksqldb-server-1
    depends_on:
      - redpanda-1
    networks:
      - redpanda_network-1
    ports:
      - "8088:8088"
    environment:
      KSQL_LISTENERS: "http://0.0.0.0:8088"
      KSQL_BOOTSTRAP_SERVERS: "redpanda-1:9092"
      KSQL_KSQL_SCHEMA_REGISTRY_URL: "http://schema-registry:8081"
```

```

KSQL KSQL_LOGGING_PROCESSING_STREAM_AUTO_CREATE: "true"
KSQL_KSQL_LOGGING_PROCESSING_TOPIC_AUTO_CREATE: "true"

ksqldb-cli-1:
image: confluentinc/cp-ksqldb-cli:latest
container_name: ksqldb-cli-1
depends_on:
  - redpanda-1
  - ksqldb-server-1
networks:
  - redpanda_network-1
entrypoint: /bin/sh
tty: true

```

Inicio los contenedores:

docker-compose up -d

```

PS E:\ad_bigdata\tarea-cpalacios> docker-compose up -d
[+] Building 0.8s (0/0)
[+] Running 5/5
✔ Network redpanda_cpacios_redpanda_network-1 Created 0.1s
✔ Container redpanda-1 Started 1.0s
✔ Container ksqldb-server-1 Started 1.8s
✔ Container redpanda-console-1 Started 1.7s
✔ Container ksqldb-cli-1 Started 2.1s
PS E:\ad_bigdata\tarea-cpalacios>

```

redpanda-cpalacios	Running (4/4)	3 minutes ago	3.67%
redpanda-1	Running	3 minutes ago	2.96%
redpanda-console-1	Running	3 minutes ago	0%
ksqldb-server-1	Running	3 minutes ago	0.71%
ksqldb-cli-1	Running	3 minutes ago	0%

Prueba de Redpanda Console:

Cluster Overview

Running 6.94 KiB Redpanda v23.2.6 1 of 1 3 3
Cluster Status Cluster Storage Size Cluster Version Brokers Online Topics Replicas

BROKER DETAILS

ID	Status	Size
0	Running	6.94 KiB

CLUSTER DETAILS

SERVICES

Kafka Connect	Not configured
Schema Registry	Healthy 0 schemas

STORAGE

Total Bytes	6.94 KiB
Primary	6.94 KiB
Replicated	0 B

SECURITY

Service Accounts	0
ACLs	1

Licensing

Console Open Source
Redpanda Open Source

Redpanda Console (Platform Version v23.2) (Built August 01, 2023) 3f51f89

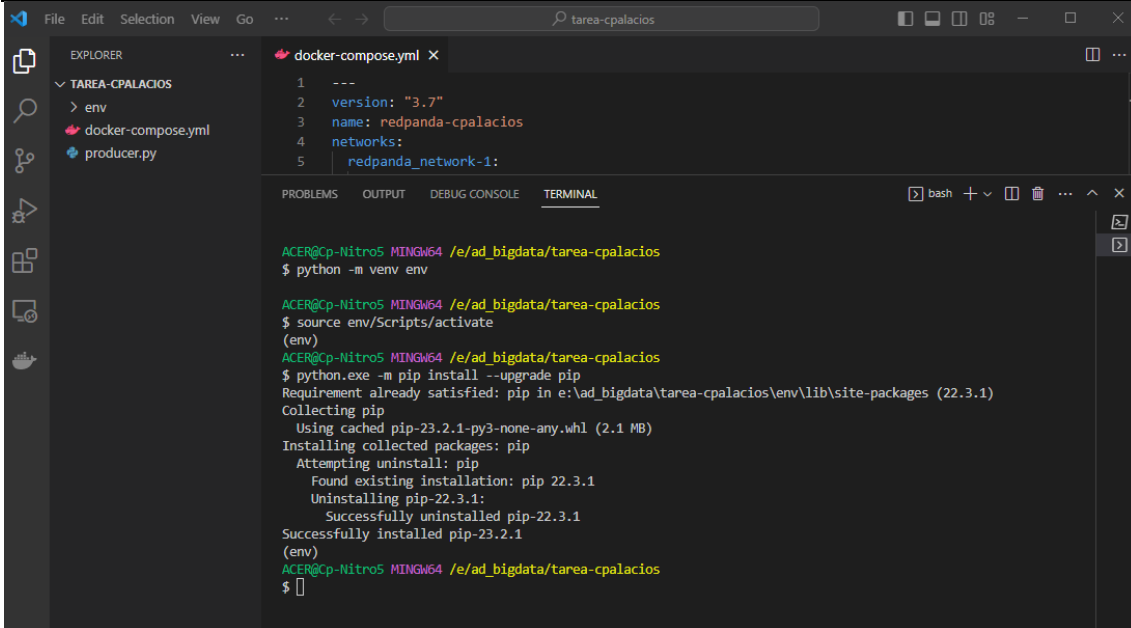
Preparación de entorno Python

En Windows:

```

python -m venv env
source env/Scripts/activate
python.exe -m pip install --upgrade pip

```



```

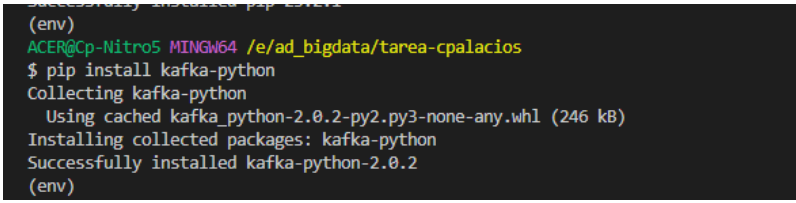
1  ---
2  version: "3.7"
3  name: redpanda-cpalacios
4  networks:
5  redpanda_network-1:

ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ python -m venv env

ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ source env/Scripts/activate
(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ python.exe -m pip install --upgrade pip
Requirement already satisfied: pip in e:\ad_bigdata\tarea-cpalacios\env\lib\site-packages (22.3.1)
Collecting pip
  Using cached pip-23.2.1-py3-none-any.whl (2.1 MB)
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.3.1
    Uninstalling pip-22.3.1:
      Successfully uninstalled pip-22.3.1
  Successfully installed pip-23.2.1
(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$
  
```

Instalación de requerimientos:

pip install kafka-python



```

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ pip install kafka-python
Collecting kafka-python
  Using cached kafka_python-2.0.2-py2.py3-none-any.whl (246 kB)
Installing collected packages: kafka-python
Successfully installed kafka-python-2.0.2
(env)
  
```

pip install websocket-client



```

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ pip install websocket-client
Collecting websocket-client
  Obtaining dependency information for websocket-client from https://files.pythonhosted.org/packages/d3/a3/63e9329c8c9be0153e919e1700ef5b60537fed78564872951b95bcc17c/websocket_client-1.6.1-py3-none-any.whl.metadata
  Using cached websocket_client-1.6.1-py3-none-any.whl.metadata (7.6 kB)
  Using cached websocket_client-1.6.1-py3-none-any.whl (56 kB)
Installing collected packages: websocket-client
Successfully installed websocket-client-1.6.1
(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$
  
```

Evaluación de estado:

docker exec -it redpanda-1 rpkm cluster info --brokers=127.0.0.1:19092

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ docker exec -it redpanda-1 rpk cluster info --brokers=127.0.0.1:19092
CLUSTER
=====
redpanda.46dbbe38-62af-434c-a56a-28b4b91d0e90

BROKERS
=====
ID      HOST      PORT
0*      localhost 19092

TOPICS
=====
NAME                                PARTITIONS  REPLICAS
_confluent-ksql-default__command_topic  1            1
_schemas                                1            1
default_ksql_processing_log              1            1

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ 

```

Estado del Cluster

```
docker exec -it redpanda-1 rpk cluster info
```

```

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ docker exec -it redpanda-1 rpk cluster info
CLUSTER
=====
redpanda.46dbbe38-62af-434c-a56a-28b4b91d0e90

BROKERS
=====
ID      HOST      PORT
0*      redpanda-1 9092

TOPICS
=====
NAME                                PARTITIONS  REPLICAS
_confluent-ksql-default__command_topic  1            1
_schemas                                1            1
default_ksql_processing_log              1            1

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ 

```

Creación de un topic llamado “reg-updates”:

```
docker exec -it redpanda-1 rpk topic create reg-updates
```

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ docker exec -it redpanda-1 rpk topic create reg-updates
TOPIC      STATUS
reg-updates OK
(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ 

```

```
(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$ docker exec -it redpanda-1 rpk cluster info
CLUSTER
=====
redpanda.46dbbe38-62af-434c-a56a-28b4b91d0e90

BROKERS
=====
ID      HOST      PORT
0*      redpanda-1  9092

TOPICS
=====
NAME                                PARTITIONS  REPLICAS
_confluent-ksql-default__command_topic  1           1
_schemas                               1           1
default_ksql_processing_log             1           1
reg-updates                             1           1

(env)
ACER@Cp-Nitro5 MINGW64 /e/ad_bigdata/tarea-cpalacios
$
```



Implementación de producer utilizando kafka-python, de acuerdo a la documentación de Redpanda para suscribirse a los eventos de la API.

Archivo: **producer.py**

```
import json
import traceback
from datetime import datetime

import websocket
from kafka import KafkaProducer
from kafka.errors import KafkaError

producer = KafkaProducer(
    bootstrap_servers="localhost:19092",
    value_serializer=lambda m: json.dumps(m).encode("ascii"),
)

topic = "reg-updates"

# Redpanda handlers
def on_success(metadata):
    print(f"Mensaje producido para topic '{metadata.topic}' con offset {metadata.offset}")

def on_error(e):
    print(f"Error al enviar mensaje: {e}")

# WS handlers
def on_ws_message(ws, message):
    data = json.loads(message)["data"]
    records = [
        {
            "symbol": d["s"],
            "price": d["p"],
            "volume": d["v"],
            "timestamp": datetime.utcfromtimestamp(d["t"] / 1000).strftime(
                "%Y-%m-%d %H:%M:%S"
            ),
        },
    ],
```

```

    }
    for d in data:
        1

for record in records:
    future = producer.send(topic, value=record)
    future.add_callback(on_success)
    future.add_errback(on_error)
    producer.flush()

def on_ws_error(ws, error):
    print(error)

def on_ws_close(ws, close_status_code, close_msg):
    producer.close()
    print("### Cerrado ###")

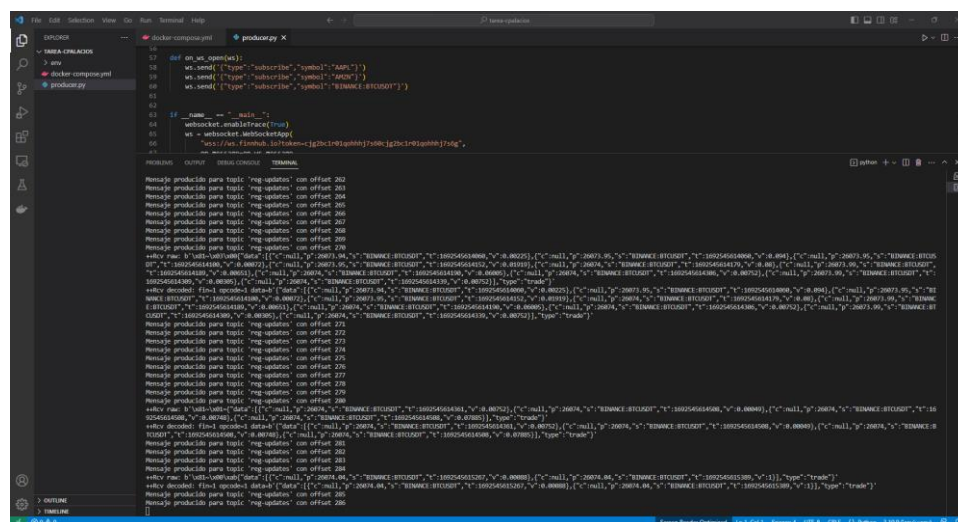
def on_ws_open(ws):
    ws.send({'type': 'subscribe', 'symbol': 'AAPL'})
    ws.send({'type': 'subscribe', 'symbol': 'AMZN'})
    ws.send({'type': 'subscribe', 'symbol': 'BINANCE:BTCUSD'})

if __name__ == '__main__':
    websocket.enableTrace(True)
    ws = websocket.WebSocketApp(
        "wss://ws.finnhub.io?token=cjg2bclr01qohhhj7s60cjg2bclr01qohhhj7s6g",
        on_message=on_ws_message,
        on_error=on_ws_error,
        on_close=on_ws_close,
    )
    ws.on_open = on_ws_open
    ws.run_forever()

```

Correr el producer:

python producer.py



Esto va a empezar a cargar el topic **reg-updates**

Cluster > Topics > **reg-updates**

91.8 kiB 545 delete 7 days Infinite
Size Messages cleanup.policy retention.ms retention.bytes

Offset	Partition	Timestamp	Key	Value
544	0	2018/2023, 11:34:26		["symbol":"BINANCE-BTCUSDT","price":126076,"volume":0.02299,"timestamp":"2023-08-28 15:34:26"]
543	0	2018/2023, 11:34:26		["symbol":"BINANCE-BTCUSDT","price":126076.01,"volume":0.00274,"timestamp":"2023-08-28 15:34:26"]
542	0	2018/2023, 11:34:27		["symbol":"BINANCE-BTCUSDT","price":126076,"volume":0.00763,"timestamp":"2023-08-28 15:34:27"]
541	0	2018/2023, 11:34:27		["symbol":"BINANCE-BTCUSDT","price":126076,"volume":0.00226,"timestamp":"2023-08-28 15:34:27"]
540	0	2018/2023, 11:34:26		["symbol":"BINANCE-BTCUSDT","price":126076,"volume":0.00286,"timestamp":"2023-08-28 15:34:26"]
539	0	2018/2023, 11:34:26		["symbol":"BINANCE-BTCUSDT","price":126076,"volume":0.00675,"timestamp":"2023-08-28 15:34:26"]
538	0	2018/2023, 11:34:26		["symbol":"BINANCE-BTCUSDT","price":126076.01,"volume":0.00059,"timestamp":"2023-08-28 15:34:26"]
537	0	2018/2023, 11:34:24		["symbol":"BINANCE-BTCUSDT","price":126076.01,"volume":0.00059,"timestamp":"2023-08-28 15:34:27"]
536	0	2018/2023, 11:34:23		["symbol":"BINANCE-BTCUSDT","price":126076.01,"volume":0.00076,"timestamp":"2023-08-28 15:34:27"]
535	0	2018/2023, 11:34:22		["symbol":"BINANCE-BTCUSDT","price":126076.01,"volume":0.00052,"timestamp":"2023-08-28 15:34:27"]
534	0	2018/2023, 11:34:22		["symbol":"BINANCE-BTCUSDT","price":126076.01,"volume":0.00054,"timestamp":"2023-08-28 15:34:27"]
533	0	2018/2023, 11:34:22		["symbol":"BINANCE-BTCUSDT","price":126076.01,"volume":0.00052,"timestamp":"2023-08-28 15:34:27"]
532	0	2018/2023, 11:34:21		["symbol":"BINANCE-BTCUSDT","price":126076,"volume":0.00115,"timestamp":"2023-08-28 15:34:27"]
531	0	2018/2023, 11:34:21		["symbol":"BINANCE-BTCUSDT","price":126076,"volume":0.00223,"timestamp":"2023-08-28 15:34:27"]

Nota: en el experimento, solo se vieron llegar de BINANCE:BTCUSDT

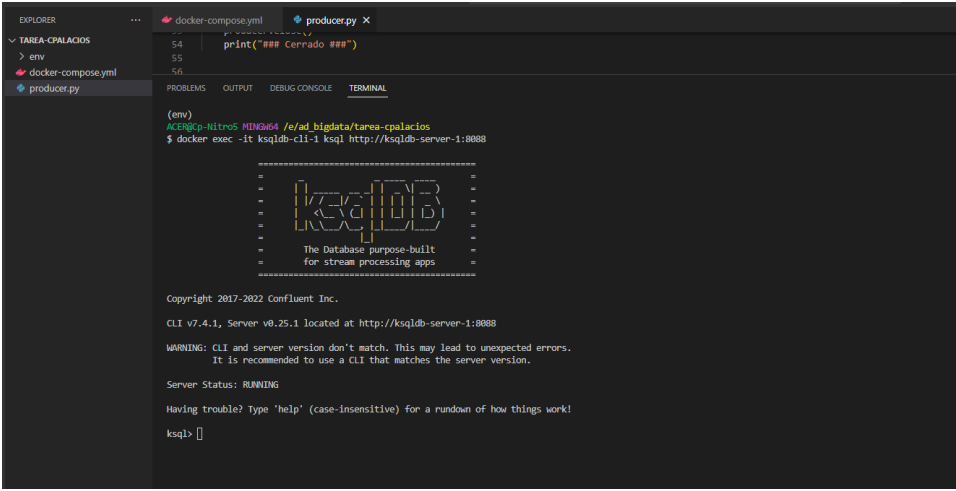
El consumer

Ejecutar ksqldb-cli, definir los streams y tablas necesarias para responder a las siguientes preguntas:

1. ¿Cuál fue el promedio ponderado de precio de una unidad por cada uno de los símbolos procesados? (e.j. AAPL)
2. ¿Cuántas transacciones se procesaron por símbolo?
3. ¿Cuál fue el máximo precio registrado por símbolo?
4. ¿Cuál fue el mínimo precio registrado por símbolo?

Iniciar ksqldb y acceder a su interfaz

```
docker exec -it ksqldb-cli-1 ksql http://ksqldb-server-1:8088
```



```

EXPLORER
  TAREA-CPALACIOS
    env
    docker-compose.yml
    producer.py

producer.py
  54 print("### Cerrado ###")
  55
  56

TERMINAL
(env)
ACER@cp-htnro5 MINGW64 /e/ad/bigdata/tarea-cpalacios
$ docker exec -it ksqldb-cli-1 ksql http://ksqldb-server-1:8088

=====
| ksqldb |
|=====|
| The Database purpose-built |
| for stream processing apps |
|=====|

Copyright 2017-2022 Confluent Inc.

CLI v7.4.1, Server v0.25.1 located at http://ksqldb-server-1:8088

WARNING: CLI and server version don't match. This may lead to unexpected errors.
It is recommended to use a CLI that matches the server version.

Server Status: RUNNING

Having trouble? Type 'help' (case-insensitive) for a rundown of how things work!

ksql>

```

Revisamos si ya tenemos topics creados:

```
ksql> SHOW TOPICS;
```

```
ksql> SHOW TOPICS;
```

Kafka Topic	Partitions	Partition Replicas
default_ksql_processing_log	1	1
reg-updates	1	1

```
ksql> []
```

Creación del stream en ksqlDB

Este comando crea no solo un **STREAM**, sino también un topic de Redpanda, si aún no existe. Si el tema ya existe, el comando define la secuencia, que luego se puede seleccionar con sintaxis SQL.

Para este caso ya creemos el topic

```
CREATE STREAM registros (symbol VARCHAR, price DOUBLE, volume DOUBLE, timestamp STRING)
WITH (kafka_topic='reg-updates', value_format='json', partitions=1);
```

```
ksql> SHOW TOPICS;
```

Kafka Topic	Partitions	Partition Replicas
default_ksql_processing_log	1	1
reg-updates	1	1

```
ksql> CREATE STREAM registros (symbol VARCHAR, price DOUBLE, volume DOUBLE, timestamp STRING)
>WITH (kafka_topic='reg-updates', value_format='json', partitions=1);
```

```
Message
```

```
Stream created
```

```
ksql> SHOW TOPICS;
```

Kafka Topic	Partitions	Partition Replicas
default_ksql_processing_log	1	1
reg-updates	1	1

```
ksql> []
```

```
ksql> SET 'auto.offset.reset' = 'earliest';
```

```
ksql> SET 'auto.offset.reset' = 'earliest';
Successfully changed local property 'auto.offset.reset' to 'earliest'. Use the UNSET command to revert your change.
ksql> []
```

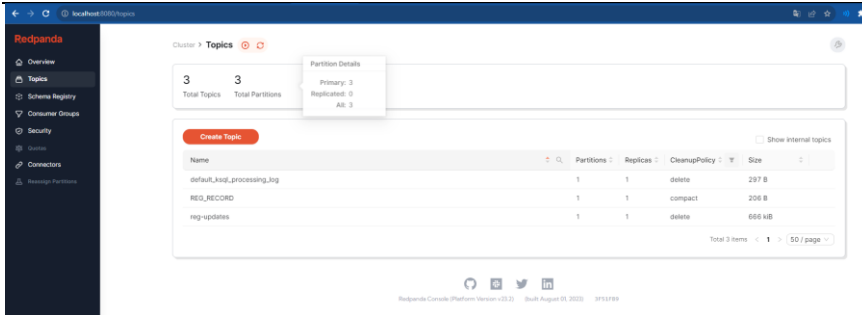
```
ksql> SELECT * FROM registros;
```

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL
BINANCE:BTCUSD	26897.41	0.00044	2023-08-20 15:59:29
BINANCE:BTCUSD	26897.42	0.0045	2023-08-20 15:59:29
BINANCE:BTCUSD	26897.41	0.00709	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.41	0.00031	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.42	0.00181	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.42	0.00383	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.42	0.0054	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.42	0.00646	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.42	0.0063	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.42	0.00483	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.47	0.00179	2023-08-20 15:59:30
BINANCE:BTCUSD	26897.48	0.00229	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00306	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00456	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00306	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00077	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00078	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00117	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00459	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00306	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00126	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00337	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00306	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00114	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00076	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00076	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00084	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00604	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00306	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00078	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.47	0.00381	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.48	0.00408	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.48	0.00116	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.48	0.00575	2023-08-20 15:59:31
BINANCE:BTCUSD	26897.76	0.00057	2023-08-20 15:59:31
BINANCE:BTCUSD	26898.18	0.00595	2023-08-20 15:59:31
BINANCE:BTCUSD	26898.26	0.0017	2023-08-20 15:59:31
BINANCE:BTCUSD	26898.6	0.00078	2023-08-20 15:59:31
BINANCE:BTCUSD	26898.77	0.00219	2023-08-20 15:59:31
BINANCE:BTCUSD	26898.77	0.00086	2023-08-20 15:59:31
BINANCE:BTCUSD	26898.78	0.01915	2023-08-20 15:59:31
BINANCE:BTCUSD	26899.0	0.026	2023-08-20 15:59:31
BINANCE:BTCUSD	26899.09	0.00039	2023-08-20 15:59:31

Creación de vistas materializadas

```
CREATE TABLE reg_record AS
SELECT symbol,
       count(price) AS total_registros
FROM registros
GROUP BY symbol
EMIT CHANGES;
```

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL
ksql> SHOW TOPICS;			
	Kafka Topic	Partitions	Partition Replicas
	default_ksql_processing_log	1	1
	reg-updates	1	1
ksql> CREATE TABLE reg_record AS			
> SELECT symbol,			
> count(price) AS total_registros			
> FROM registros			
> GROUP BY symbol			
> EMIT CHANGES;			
	Message		
	Created query with ID CTAS_REG_RECORD_5		
ksql> SHOW TOPICS;			
	Kafka Topic	Partitions	Partition Replicas
	REG_RECORD	1	1
	default_ksql_processing_log	1	1
	reg-updates	1	1
ksql>			



Cluster > Topics

3 Total Topics, 3 Total Partitions

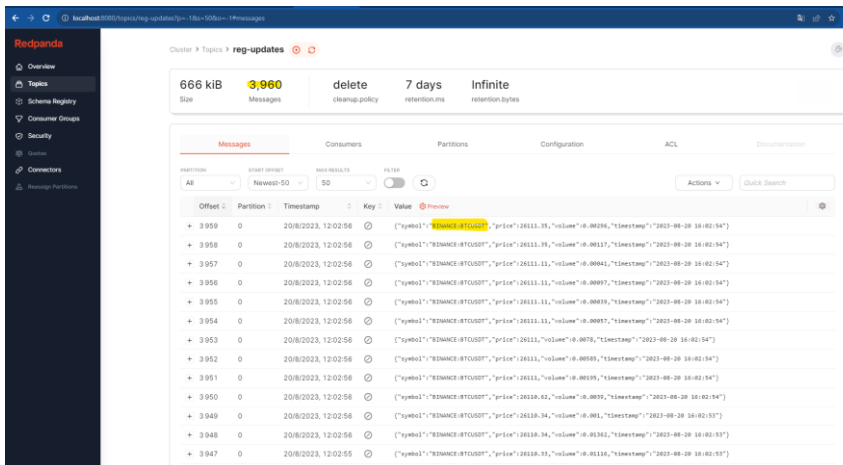
Primary: 3, Replicated: 0, All: 3

Create Topic

Name	Partitions	Replicas	CleanupPolicy	Size
default_ksql_processing_log	1	1	delete	297 B
REG_RECORD	1	1	compact	206 B
reg-updates	1	1	delete	666 KiB

Total 3 items < 1 > 50 / page

Redpanda Console (Platform Version v23.2) (Built August 01, 2023) 3F51F89



Cluster > Topics > reg-updates

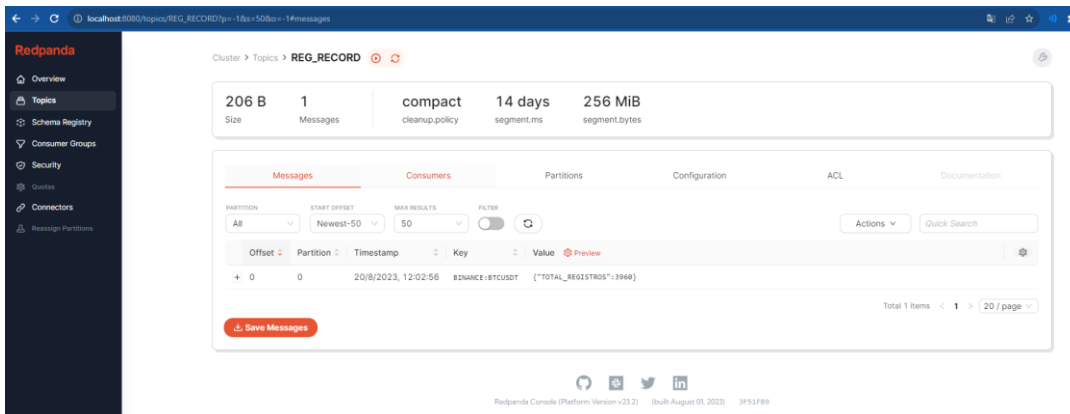
666 KiB Size, 3,960 Messages, delete cleanup.policy, 7 days retention.ms, Infinite retention.bytes

Messages, Consumers, Partitions, Configuration, ACL, Documentation

Offset	Partition	Timestamp	Key	Value	Preview
+ 3959	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111.35, "volume": 0.00296, "timestamp": "2023-08-20 16:02:56"}]	
+ 3958	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111.39, "volume": 0.00117, "timestamp": "2023-08-20 16:02:56"}]	
+ 3957	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111.11, "volume": 0.00041, "timestamp": "2023-08-20 16:02:56"}]	
+ 3956	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111.11, "volume": 0.00097, "timestamp": "2023-08-20 16:02:56"}]	
+ 3955	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111.11, "volume": 0.00039, "timestamp": "2023-08-20 16:02:56"}]	
+ 3954	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111.11, "volume": 0.00057, "timestamp": "2023-08-20 16:02:56"}]	
+ 3953	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111, "volume": 0.0078, "timestamp": "2023-08-20 16:02:56"}]	
+ 3952	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111, "volume": 0.00089, "timestamp": "2023-08-20 16:02:56"}]	
+ 3951	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26111, "volume": 0.00089, "timestamp": "2023-08-20 16:02:56"}]	
+ 3950	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26110.42, "volume": 0.0039, "timestamp": "2023-08-20 16:02:56"}]	
+ 3949	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26110.34, "volume": 0.001, "timestamp": "2023-08-20 16:02:53"}]	
+ 3948	0	20/8/2023, 12:02:56		[{"symbol": "Binance:BTCUSD", "price": 26110.34, "volume": 0.01362, "timestamp": "2023-08-20 16:02:53"}]	
+ 3947	0	20/8/2023, 12:02:55		[{"symbol": "Binance:BTCUSD", "price": 26110.33, "volume": 0.01116, "timestamp": "2023-08-20 16:02:53"}]	

```
SELECT * FROM reg_record;
```

```
ksql> SELECT * FROM reg_record;
+-----+-----+-----+-----+
|SYMBOL|          |TOTAL_REGISTROS|
+-----+-----+-----+-----+
|Binance:BTCUSD|          |3960|
Query terminated
ksql>
```



Cluster > Topics > REG_RECORD

206 B Size, 1 Messages, compact cleanup.policy, 14 days segment.ms, 256 MiB segment.bytes

Messages, Consumers, Partitions, Configuration, ACL, Documentation

Offset	Partition	Timestamp	Key	Value	Preview
+ 0	0	20/8/2023, 12:02:56	Binance:BTCUSD	[{"TOTAL_REGISTROS": 3960}]	

Total 1 items < 1 > 20 / page

Save Messages

Redpanda Console (Platform Version v23.2) (Built August 01, 2023) 3F51F89

¿Cuál fue el promedio ponderado de precio de una unidad por cada uno de los símbolos procesados? (e.j. AAPL)

Tabla:

```
CREATE TABLE reg_prom AS
SELECT symbol,
       count(price) AS total_registros,
       sum(price*volume) AS weighted_price,
       sum(volume) AS total_volume,
       sum(price*volume)/sum(volume) AS promedio
FROM registros
GROUP BY symbol
EMIT CHANGES;
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

ksql> CREATE TABLE reg_prom AS
> SELECT symbol,
>       count(price) AS total_registros,
>       sum(price*volume) AS weighted_price,
>       sum(volume) AS total_volume,
>       sum(price*volume)/ sum(volume) AS promedio
> FROM registros
> GROUP BY symbol
> EMIT CHANGES;

Message
-----
Created query with ID CTAS_REG_PROM_8
-----

ksql> []
```

```
ksql> SHOW TOPICS;

Kafka Topic          | Partitions | Partition Replicas
-----
REG_PROM              | 1          | 1
REG_RECORD            | 1          | 1
default_ksql_processing_log | 1          | 1
reg-updates           | 1          | 1
-----

ksql> []
```

General:

```
SELECT * FROM reg_prom;
```

```
ksql> SELECT * FROM reg_prom;
+-----+-----+-----+-----+-----+
|SYMBOL|TOTAL_REGISTROS|WEIGHTED_PRICE|TOTAL_VOLUME|PROMEDIO|
+-----+-----+-----+-----+-----+
|BINANCE:BTCUSD|3960|4091799.6740312055|156.85854000000032|26085.922220308803|
+-----+-----+-----+-----+-----+
Query terminated
ksql> []
```

Consulta por symbol:

```
SELECT * FROM reg_prom
WHERE symbol = 'BINANCE:BTCUSD';
```

```
SELECT * FROM reg_prom
WHERE symbol = 'AAPL';
```

```
SELECT * FROM reg_prom
WHERE symbol = 'AMZN';
```

```
Query terminated
ksql> SELECT * FROM reg_prom
>WHERE symbol = 'BINANCE:BTCUSD';
```

SYMBOL	TOTAL_REGISTROS	WEIGHTED_PRICE	TOTAL_VOLUME	PROMEDIO
BINANCE:BTCUSD	3960	4091799.6740312055	156.85854000000032	26085.922220308803

```
Query terminated
ksql> SELECT * FROM reg_prom
>WHERE symbol = 'AAPL';
```

SYMBOL	TOTAL_REGISTROS	WEIGHTED_PRICE	TOTAL_VOLUME	PROMEDIO
BINANCE:BTCUSD				

```
Query terminated
ksql> SELECT * FROM reg_prom
>WHERE symbol = 'AMZN';
```

SYMBOL	TOTAL_REGISTROS	WEIGHTED_PRICE	TOTAL_VOLUME	PROMEDIO
BINANCE:BTCUSD				

```
Query terminated
ksql> []
```

¿Cuántas transacciones se procesaron por símbolo?

Tabla:

```
CREATE TABLE reg_record AS
SELECT symbol,
       count(price) AS total_registros
FROM registros
GROUP BY symbol
EMIT CHANGES;
```

General:

```
SELECT * FROM reg_record;
```

```
ksql> SELECT * FROM reg_record;
```

SYMBOL	TOTAL_REGISTROS
BINANCE:BTCUSD	6150

```
Query terminated
ksql> SELECT * FROM reg_record;
```

SYMBOL	TOTAL_REGISTROS
BINANCE:BTCUSD	6315

```
Query terminated
ksql> SELECT * FROM reg_record;
```

SYMBOL	TOTAL_REGISTROS
BINANCE:BTCUSD	6316

```
Query terminated
ksql> []
```

Consulta por symbol:

```
SELECT * FROM reg_record
WHERE symbol = 'BINANCE:BTCUSD';
```

```
SELECT * FROM reg_record
WHERE symbol = 'AAPL';
```

```
SELECT * FROM reg_record
WHERE symbol = 'AMZN';
```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
ksql> SELECT * FROM reg_record
>WHERE symbol = 'BINANCE:BTCUSD';
+-----+-----+
|SYMBOL|TOTAL_REGISTROS|
+-----+-----+
|BINANCE:BTCUSD|3960|
+-----+-----+
Query terminated
ksql> SELECT * FROM reg_record
>WHERE symbol = 'AAPL';
+-----+-----+
|SYMBOL|TOTAL_REGISTROS|
+-----+-----+
Query terminated
ksql> SELECT * FROM reg_record
>WHERE symbol = 'AMZN';
+-----+-----+
|SYMBOL|TOTAL_REGISTROS|
+-----+-----+
Query terminated
ksql> 

```

¿Cuál fue el máximo precio registrado por símbolo?

Tabla:

```

CREATE TABLE maxprice_record AS
SELECT symbol,
       max(price) AS precio_maximo
FROM registros
GROUP BY symbol
EMIT CHANGES;

```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
ksql> CREATE TABLE maxprice_record AS
> SELECT symbol,
>       max(price) AS precio_maximo
> FROM registros
> GROUP BY symbol
> EMIT CHANGES;

Message
-----
Created query with ID CTAS_MAXPRICE_RECORD_11
-----
ksql> SHOW TOPICS;

Kafka Topic          | Partitions | Partition Replicas
-----
MAXPRICE_RECORD      | 1          | 1
REG_PROM              | 1          | 1
REG_RECORD            | 1          | 1
default_ksql_processing_log | 1          | 1
reg-updates           | 1          | 1
-----
ksql> 

```

General:

```
SELECT * FROM maxprice_record;
```

```

ksql> SELECT * FROM maxprice_record;
+-----+-----+
|SYMBOL|PRECIO_MAXIMO|
+-----+-----+
|BINANCE:BTCUSD|26111.35|
+-----+-----+
Query terminated
ksql> 

```

Consulta por symbol:

```
SELECT * FROM maxprice_record
```

```
WHERE symbol = 'BINANCE:BTCUSD';
```

```
SELECT * FROM maxprice_record
WHERE symbol = 'AAPL';
```

```
SELECT * FROM maxprice_record
WHERE symbol = 'AMZN';
```

```
ksql> SELECT * FROM maxprice_record
>WHERE symbol = 'BINANCE:BTCUSD';
+-----+-----+
|SYMBOL|PRECIO_MAXIMO|
+-----+-----+
|BINANCE:BTCUSD|26111.35|
+-----+-----+
Query terminated
ksql> SELECT * FROM maxprice_record
>WHERE symbol = 'AAPL';
+-----+-----+
|SYMBOL|PRECIO_MAXIMO|
+-----+-----+
Query terminated
ksql> SELECT * FROM maxprice_record
>WHERE symbol = 'AMZN';
+-----+-----+
|SYMBOL|PRECIO_MAXIMO|
+-----+-----+
Query terminated
ksql> []
```

¿Cuál fue el mínimo precio registrado por símbolo?

Tabla:

```
CREATE TABLE minprice_record AS
SELECT symbol,
       min(price) AS precio_minimo
FROM registros
GROUP BY symbol
EMIT CHANGES;
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

ksql> CREATE TABLE minprice_record AS
> SELECT symbol,
>       min(price) AS precio_minimo
> FROM registros
> GROUP BY symbol
> EMIT CHANGES;

Message
-----
Created query with ID CTAS_MINPRICE_RECORD_14
-----

ksql> SHOW TOPICS;

Kafka Topic      | Partitions | Partition Replicas
+-----+-----+
MAXPRICE_RECORD  | 1          | 1
MINPRICE_RECORD  | 1          | 1
REG_PROM         | 1          | 1
REG_RECORD       | 1          | 1
default_ksql_processing_log | 1          | 1
reg-updates      | 1          | 1
+-----+-----+

ksql> []
```

General:

```
SELECT * FROM minprice_record;
```

```
ksql> SELECT * FROM minprice_record;
+-----+-----+
|SYMBOL|PRECIO_MINIMO|
+-----+-----+
|BINANCE:BTCUSD|26024.01|
+-----+-----+
Query terminated
ksql> []
```

Consulta por symbol:

```
SELECT * FROM minprice_record  
WHERE symbol = 'BINANCE:BTCUSD';
```

```
SELECT * FROM minprice_record  
WHERE symbol = 'AAPL';
```

```
SELECT * FROM minprice_record  
WHERE symbol = 'AMZN';
```

```
ksql> SELECT * FROM minprice_record  
>WHERE symbol = 'BINANCE:BTCUSD';  
+-----+-----+  
|SYMBOL|PRECIO_MINIMO|  
+-----+-----+  
|BINANCE:BTCUSD|26624.01|  
Query terminated  
ksql> SELECT * FROM minprice_record  
>WHERE symbol = 'AAPL';  
+-----+-----+  
|SYMBOL|PRECIO_MINIMO|  
+-----+-----+  
Query terminated  
ksql> SELECT * FROM minprice_record  
>WHERE symbol = 'AMZN';  
+-----+-----+  
|SYMBOL|PRECIO_MINIMO|  
+-----+-----+  
Query terminated  
ksql>
```