TAREA DATA STREAMING

GRUPO 19:

- Carlos Emanuel Aquino Peralta
- Hugo Alberto Díaz Parra
- Hugo Ismael Gonzalez
- Rodolfo Paredes Monges
- Elio Emanuel Romero López

PRE-REQUISITOS

- 1. **Python 3.x** con pip actualizado
- 2. **Docker** con docker compose actualizado

INSTRUCCIONES

- 1. **Crear los contenedores**, para ello utilizar el docker compose utilizando el siguiente comando
 - \$ docker compose -d up
- 2. Comprobar los contenedores, para ello ejecutar el comando
 - \$ docker stats
 - se deben visualizar en ejecucion los contenedores
 - redpanda
 - redpanda-console
 - ksqldb-server
 - ksqldb-cli
- 3. Crear el ambiente de trabajo para Python3, para ello ejecutar:
 - \$ python3 -m venv env
 - \$ source env/bin/activate
 - \$ pip install --upgrade pip
 - \$ pip install kafka-python
- 4. Crear el topic en RedPanda
 - \$ docker exec -it redpanda rpk topic create stock-updates
- 5. **Actualizar el API Key de Finnhub**, para registrarse en https://finnhub.io/ y obtener el API Key, luego editar y actualizar el archivo producer.py en la linea 55 **Finnhub Stock API KEY:** cjfcc21r01qhblojr5s0cjfcc21r01qhblojr5sg
- 6. **Cargar los mensajes a RedPanda**, ejecutando el producer.py \$ python3 stock-monitor/producer.py
- 7. **Verificar la carga de los datos**, para ello ingresar a la consola de RedPanda desde el navegador a la siguiente URL http://127.0.0.1:8080/overview, se debe visualizar el topic **stock-updates** y los mensajes.
- 8. Ingresar a KSQL, con el siguiente comando
 - \$ sudo docker exec -it ksqldb-cli ksql http://ksqldb-server:8088
- 9. Crear el stream stock_updates en KLSQL
 - > CREATE STREAM stock_updates (symbol VARCHAR,

price DOUBLE,

volume DOUBLE.

timestamp VARCHAR)

WITH (

kafka topic='stock-updates',

value format='json');

- 10. Consultar los datos del stream,
 - > SELECT * FROM STOCK UPDATES;
- 11. Crear la tabla de stock_resume en KLSQL,
 - > CREATE TABLE stock_resume AS SELECT symbol AS simbolo,

MAX(price) AS maximo,

MIN(price) AS minimo,

SUM(price*volume)/SUM(volume) AS promedio,

COUNT(*) AS cantidad

FROM stock updates

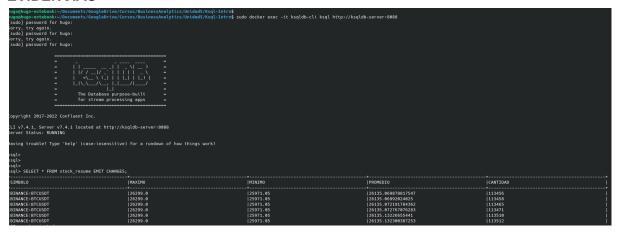
GROUP BY symbol;

12. Consultar la tabla de resumen,

```
> SELECT * FROM STOCK RESUME EMIT CHANGES;
```

la información resumida va ir actualizando según el productor carga los mensajes a RedPanda.

EVIDENCIAS



REPOSITORIO GIT

https://github.com/ninjadiaz/redpanda/tree/main/Ksql-Intro

REFERENCIAS

https://finnhub.io/

https://docs.redpanda.com/docs/get-started/guick-start/

https://redpanda.com/blog/python-redpanda-kafka-api-tutorial

https://redpanda.com/blog/ksqldb-materialized-cache