Sprawozdanie

6th May 2024

1 Producent; skrypty inicjujące i zasilający

Klaster należy uruchomić poleceniem:

```
gcloud dataproc clusters create ${CLUSTER_NAME} \
--enable-component-gateway --region ${REGION} --subnet default \
--master-machine-type n1-standard-2 --master-boot-disk-size 50 \
--num-workers 2 --worker-machine-type n1-standard-2 --worker-boot-disk-size 50 \
--image-version 2.1-debian11 --optional-components DOCKER,ZOOKEEPER,ZEPPELIN \
--project ${PROJECT_ID} --max-age=3h \
--metadata "run-on-master=true" \
--initialization-actions \
gs://goog-dataproc-initialization-actions-${REGION}/kafka/kafka.sh
```

Należy załadować dane na klaster z `http://www.cs.put.poznan.pl/kjankiewicz/bigdata/stream_project` - folder stocks_result.zip i plik csv symbols_valid_meta.csv. Na klaster należy załadować również jar z projektem - stock_stats_app.jar

Należy załadować na klaster również pliki z folderu projekt2.

Poniżej są opisane skrypty inicjujące i zasilające wraz z ich użyciem:

1.1 skrypt inicjujący tematy kafki i resetujący środowisko - initialize_reset_kafka_topics.sh

Sposób uruchomienia: sh initialize_reset_kafka_topics.sh

1.2 skrypt zasilający temat kafki kafka-input plikami z folderu stocks_result - send_to_kafka_input_topic.sh

Należy nadać uprawnienia dla skryptu csvtotxt.sh, który jest wykonywany w środku skryptu send_to_kafka_input_topic.sh. Zamienia on każdą linię w pliku csv na odpowiednią formę, która zostanie wysłana do tematu kafki: <klucz>:<wartość w formacie json>

chmod +x csvtotxt.sh

Sposób uruchomienia: sh send_to_kafka_input_topic.sh

1.2.1 Zrzut ekranu pokazujący format danych wysłany do tematu kafki kafka-input


```
JNJ:{"Stock": "JNJ","Date": "1965-09-20T00:00:00.000Z","Open": "0.0","High": "0.3524305522441864"
,"Low": "0.3489583432674408","Close": "0.35011574625968933","Volume": "345600.0"}
GT:{"Stock": "GT","Date": "1965-09-21T00:00:00.000Z","Open": "0.0","High": "12.40625","Low": "12.
0625","Close": "12.1875","Volume": "21200.0"}
HPQ:{"Stock": "HPQ","Date": "1965-09-21T00:00:00.000Z","Open": "0.1197419986128807","High": "0.12
195944041013718","Low": "0.1197419986128807","Close": "0.1197419986128807","Volume": "3748600.0"}
GE:{"Stock": "GE","Date": "1965-09-21T00:00:00.000Z","Open": "1.1343148946762085","High": "1.1443
309783935547","Low": "1.1242989301681519","Close": "1.1293069124221802","Volume": "1627300.0"}
MO:{"Stock": "MO","Date": "1965-09-21T00:00:00.000Z","Open": "0.0","High": "0.1649305522441864","
Low": "0.1629774272441864","Close": "0.1644965261220932","Volume": "2649600.0"}
KO:{"Stock": "KO","Date": "1965-09-21T00:00:00.000Z","Open": "0.4108072817325592","High": "0.4134
114682674408","Low": "0.41015625","Close": "0.41015625","Volume": "364800.0"}
JNJ:{"Stock": "JNJ","Date": "1965-09-21T00:00:00.000Z","Open": "0.0","High": "0.35300925374031067
","Low": "0.3506944477558136","Close": "0.35300925374031067","Volume": "259200.0"}
GT:{"Stock": "GT","Date": "1965-09-22T00:00:00.000Z","Open": "0.0","High": "12.375","Low": "12.12
5","Close": "12.375","Volume": "22000.0"}
HPQ:{"Stock": "HPQ","Date": "1965-09-22T00:00:00.000Z","Open": "0.1197419986128807","High": "0.12
151595205068588","Low": "0.1197419986128807","Close": "0.12151595205068588","Volume": "1522000.0"
}
GE:{"Stock": "GE","Date": "1965-09-22T00:00:00.000Z","Open": "1.1293069124221802","High": "1.1418
26868057251","Low": "1.1280548572540283","Close": "1.1405749320983887","Volume": "1208000.0"}
MO:{"Stock": "MO","Date": "1965-09-22T00:00:00.000Z","Open": "0.0","High": "0.1644965261220932","
Low": "0.1627604216337204","Close": "0.1636284738779068","Volume": "518400.0"}
KO:{"Stock": "KO","Date": "1965-09-22T00:00:00.000Z","Open": "0.41015625","High": "0.4114583432674
4408","Low": "0.4075520932674408","Close": "0.4114583432674408","Volume": "249600.0"}
JNJ:{"Stock": "JNJ","Date": "1965-09-22T00:00:00.000Z","Open": "0.0","High": "0.35648149251937866
","Low": "0.3524305522441864","Close": "0.35532405972480774","Volume": "561600.0"}
GT:{"Stock": "GT","Date": "1965-09-23T00:00:00.000Z","Open": "0.0","High": "12.3125","Low": "12.1
5625","Close": "12.25","Volume": "16000.0"}
HPQ:{"Stock": "HPQ","Date": "1965-09-23T00:00:00.000Z","Open": "0.12151595205068588","High": "0.1
2195944041013718","Low": "0.11929851025342941","Close": "0.11929851025342941","Volume": "1493800.
0"}
GE:{"Stock": "GE","Date": "1965-09-23T00:00:00.000Z","Open": "1.1405749320983887","High": "1.1518
429517745972","Low": "1.1217948198318481","Close": "1.123046875","Volume": "229600.0"}
MO:{"Stock": "MO","Date": "1965-09-23T00:00:00.000Z","Open": "0.0","High": "0.1640625","Low": "0.
1631944477558136","Close": "0.1631944477558136","Volume": "1094400.0"}
KO:{"Stock": "KO","Date": "1965-09-23T00:00:00.000Z","Open": "0.4114583432674408","High": "0.4166
666567325592","Low": "0.41015625","Close": "0.41015625","Volume": "556800.0"}
JNJ:{"Stock": "JNJ","Date": "1965-09-23T00:00:00.000Z","Open": "0.0","High": "0.36342594027519226
","Low": "0.35879629850387573","Close": "0.36342594027519226","Volume": "604800.0"}
GT:{"Stock": "GT","Date": "1965-09-24T00:00:00.000Z","Open": "0.0","High": "12.4375","Low": "12.1
875","Close": "12.4375","Volume": "30400.0"}
```


1.3 skrypt zasilający temat kafki kafka-input-metadata plikiem symbols_valid_meta.csv - save_to_metadata_topic.sh

Należy nadać uprawnienia dla skryptu csvtotxtmetadata.sh, który jest wykonywany w środku skryptu save_to_metadata_topic.sh. Zamienia on każdą linię

w pliku csv na odpowiednią formę, która zostanie wysłana do tematu kafki: <klucz>:<wartość w formacie json>

chmod +x csvtotxtmetadata.sh

Sposób uruchomienia: sh save_to_metadata_topic.sh <name of csv file>

1.3.1 Zrzut ekranu pokazujący format danych wysłany do tematu kafki kafka-input-metadata


```
KBLMU:{"Symbol": "KBLMU","SecurityName": "KBL Merger Corp. IV - Unit"}
KBWD:{"Symbol": "KBWD","SecurityName": "Invesco KBW High Dividend Yield Financial ETF"}
KBWR:{"Symbol": "KBWR","SecurityName": "Invesco KBW Regional Banking ETF"}
KCE:{"Symbol": "KCE","SecurityName": "SPDR S&P Capital Markets ETF"}
KCNY:{"Symbol": "KCNY","SecurityName": "KraneShares E Fund China Commercial Paper ETF"}
KEM:{"Symbol": "KEM","SecurityName": "KEMET Corporation Common Stock"}
KEP:{"Symbol": "KEP","SecurityName": "Korea Electric Power Corporation Common Stock"}
KEX:{"Symbol": "KEX","SecurityName": "Kirby Corporation Common Stock"}
KFS:{"Symbol": "KFS","SecurityName": "Kingsway Financial Services"}
KFY:{"Symbol": "KFY","SecurityName": "Korn Ferry Common Stock"}
KFYP:{"Symbol": "KFYP","SecurityName": "KraneShares CICC China Leaders 100 Index ETF"}
KGC:{"Symbol": "KGC","SecurityName": "Kinross Gold Corporation Common Stock"}
KGRN:{"Symbol": "KGRN","SecurityName": "KraneShares MSCI China Environment Index ETF"}
KHC:{"Symbol": "KHC","SecurityName": "The Kraft Heinz Company - Common Stock"}
KIN:{"Symbol": "KIN","SecurityName": "Kindred Biosciences"}
KIO:{"Symbol": "KIO","SecurityName": "KKR Income Opportunities Fund Common Shares"}
KLCD:{"Symbol": "KLCD","SecurityName": "KFA Large Cap Quality Dividend Index ETF"}
KMF:{"Symbol": "KMF","SecurityName": "Kayne Anderson Midstream Energy Fund"}
KMI:{"Symbol": "KMI","SecurityName": "Kinder Morgan"}
KMT:{"Symbol": "KMT","SecurityName": "Kennametal Inc. Common Stock"}
KN:{"Symbol": "KN","SecurityName": "Knowles Corporation Common Stock"}
KNDI:{"Symbol": "KNDI","SecurityName": "Kandi Technologies Group"}
KNOP:{"Symbol": "KNOP","SecurityName": "KNOT Offshore Partners LP Common Units representin
g Limited Partner Interests"}
KNOW:{"Symbol": "KNOW","SecurityName": "Direxion All Cap Insider Sentiment Shares"}
KNSA:{"Symbol": "KNSA","SecurityName": "Kiniksa Pharmaceuticals"}
KNX:{"Symbol": "KNX","SecurityName": "Knight-Swift Transportation Holdings Inc."}
KOIN:{"Symbol": "KOIN","SecurityName": "Innovation Shares NextGen Protocol ETF"}
KOP:{"Symbol": "KOP","SecurityName": "Koppers Holdings Inc. Koppers Holdings Inc. Common S
tock"}
KOPN:{"Symbol": "KOPN","SecurityName": "Kopin Corporation - Common Stock"}
KORP:{"Symbol": "KORP","SecurityName": "American Century Diversified Corporate Bond ETF"}
KRC:{"Symbol": "KRC","SecurityName": "Kilroy Realty Corporation Common Stock"}
KRE:{"Symbol": "KRE","SecurityName": "SPDR S&P Regional Banking ETF"}
KRMA:{"Symbol": "KRMA","SecurityName": "Global X Conscious Companies ETF"}
KRO:{"Symbol": "KRO","SecurityName": "Kronos Worldwide Inc Common Stock"}
KRP:{"Symbol": "KRP","SecurityName": "Kimbell Royalty Partners Common Units Representing L
imited Partner Interests"}
KRTX:{"Symbol": "KRTX","SecurityName": "Karuna Therapeutics"}
KSCD:{"Symbol": "KSCD","SecurityName": "KFA Small Cap Quality Dividend Index ETF"}
KSM:{"Symbol": "KSM","SecurityName": "DWS Strategic Municipal Income Trust"}
KSU:{"Symbol": "KSU","SecurityName": "Kansas City Southern Common Stock"}
KT:{"Symbol": "KT","SecurityName": "KT Corporation Common Stock"}
KTB:{"Symbol": "KTB","SecurityName": "Kontoor Brands"}
KTCC:{"Symbol": "KTCC","SecurityName": "Key Tronic Corporation - Common Stock"}
KTOS:{"Symbol": "KTOS","SecurityName": "Kratos Defense & Security Solutions"}
```

2 Utrzymanie obrazu czasu rzeczywistego – transformacje

```
//dane załadowane do tematu kafki
KStream<String, InputScores> data = builder.stream( topic: "kafka-input", Consumed.with(Serdes.String(), inputScoresSerde));
```

Stream zawierający dane z wejściowego tematu kafki

```
//agregacja
KTable<Windowed<String>, AggregationData> aggData = data
        .groupByKey()
        .windowedBy(TimeWindows.of(Duration.ofDays(30))).
        aggregate(
                () -> {
                    AggregationData aggregatedData = new AggregationData();
                    aggregatedData.Stock = "";
                    aggregatedData.name = "";
                    aggregatedData.closeValues = new ArrayList<>();
                    aggregatedData.minLow = Long.MAX_VALUE;
                    aggregatedData.maxHigh = 0;
                    aggregatedData.sumVolume = 0;

                    return aggregatedData;
                },
                (aggKey, newValue, aggValue) -> {
                    aggValue.Stock = newValue.Stock;
                    aggValue.closeValues.add(Double.parseDouble(newValue.Close));
                    aggValue.minLow = Math.min(aggValue.minLow, Double.parseDouble(newValue.Low));
                    aggValue.maxHigh = Math.max(aggValue.maxHigh, Double.parseDouble(newValue.High));
                    aggValue.sumVolume += Double.parseDouble(newValue.Volume);
                    return aggValue;
                },
                Materialized.with(Serdes.String(), aggregationDataSerde)
        );
```

Agregacje przeprowadzane na danych

```
//finalny wynik agregacji, sprowadzony do streamu
KStream<String, AggregationDataFinal> aggResult = aggData.toStream()
        .map(
                (key, value) -> {
                    AggregationDataFinal aggrData = new AggregationDataFinal();
                    aggrData.Stock = value.Stock;
                    aggrData.averageClose = value.getAverageClose();
                    aggrData.minLow = value.minLow;
                    aggrData.maxHigh = value.maxHigh;
                    aggrData.sumVolume = value.sumVolume;
                    return KeyValue.pair(key.key(), aggrData);

                }
        );
```

Finalny wynik agregacji

3 Utrzymanie obrazu czasu rzeczywistego – obsługa trybu A

Tu nic nie trzeba było robić, ponieważ w Kafka Streams takie zachowanie jest domyślne.

4 Utrzymanie obrazu czasu rzeczywistego – obsługa trybu C

Aby uwzględnić brak aktualizacji wyników obrazu czasu rzeczywistego, do agregacji został dodany .suppress(Suppressed.untilWindowCloses(unbounded()))

```
else if(DELAY.equals("C")) {

    KTable<Windowed<String>, AggregationData> aggData  = data
            .groupByKey()
            .windowedBy(TimeWindows.of(Duration.ofDays(30))).

            aggregate(
                    () -> {
                        AggregationData aggregatedData = new AggregationData();
                        aggregatedData.Stock = "";
                        aggregatedData.name = "";
                        aggregatedData.closeValues = new ArrayList<>();
                        aggregatedData.minLow = Long.MAX_VALUE;
                        aggregatedData.maxHigh = 0;
                        aggregatedData.sumVolume = 0;

                        return aggregatedData;
                    },
                    (aggKey, newValue, aggValue) -> {
                        aggValue.Stock = newValue.Stock;
                        aggValue.closeValues.add(Double.parseDouble(newValue.Close));
                        aggValue.minLow = Math.min(aggValue.minLow, Double.parseDouble(newValue.Low));
                        aggValue.maxHigh = Math.max(aggValue.maxHigh, Double.parseDouble(newValue.High));
                        aggValue.sumVolume += Double.parseDouble(newValue.Volume);
                        return aggValue;
                    },
                    Materialized.with(Serdes.String(), aggregationDataSerde)
            ).suppress(Suppressed.untilWindowCloses(unbounded()));
```

Agregacje przeprowadzane na danych w trybie C

5 Wykrywanie anomalii

```
//dane załadowane do tematu kafki
KStream<String, InputScores> data = builder.stream( topic: "kafka-input", Consumed.with(Serdes.String(), inputScoresSerde));
```

Stream zawierający dane z wejściowego tematu kafki

```
//wykrywanie anomalii
KTable<Windowed<String>, StockStats> stats = data
        .groupByKey().
        windowedBy(TimeWindows.of(Duration.ofDays(D))).
        aggregate(
                () -> {
                    StockStats hs = new StockStats();
                    hs.stock = "";
                    hs.min_score = Long.MAX_VALUE;
                    hs.max_score = 0;
                    hs.difference = 0;
                    return hs;
                },
                (aggKey, newValue, aggValue) -> {
                    aggValue.stock = newValue.Stock;
                    aggValue.min_score = Math.min(aggValue.min_score, Double.parseDouble(newValue.Low));
                    aggValue.max_score = Math.max(aggValue.max_score, Double.parseDouble(newValue.High));
                    aggValue.difference = (Double.parseDouble(newValue.High) - Double.parseDouble(newValue.Low)) * 100 / Double.parseDouble(newValue.High);
                    return aggValue;
                },
                Materialized.with(Serdes.String(), stockStatsSerde)
        );
```

Wykrywanie anomalii na danych

```
KStream<String, StockAlerts> resultStream = stats.toStream()
        .filter((key, value) -> {
            return value.difference > P;
        })
        .map(
                (key, value) -> {
                    StockAlerts ha = new StockAlerts();
                    ha.start_ts = key.window().startTime().toString();
                    ha.end_ts = key.window().endTime().toString();
                    ha.stock = value.stock;
                    ha.min_score = value.min_score;
                    ha.max_score = value.max_score;
                    ha.difference = value.difference;
                    return KeyValue.pair(key.key(), ha);

                }
        );

resultStream.to( s: "kafka-output", Produced.with(Serdes.String(), stockAlertsSerde));
```

Przygotowanie danych do raportu i na końcu wysłanie do docelowego tematu kafki

6 Program przetwarzający strumienie danych; skrypt uruchamiający

Przed uruchomieniem skryptu należy najpierw skonfigurować Kafka Connect W tym celu należy wykonać w terminalu następujące komendy:

(A przed konfiguracją Kafka Connect polecam wykonać najpierw punkt 7, czyli stworzenia miejsca utrzymania obrazów czasu rzeczywistego, potem wrócić tu)

```
wget https://repo1.maven.org/maven2/com/mysql/mysql-connector-j/\
8.0.33/mysql-connector-j-8.0.33.jar

sudo cp mysql-connector-j-8.0.33.jar /usr/lib/kafka/libs

wget https://packages.confluent.io/maven/io/confluent/kafka-connect-jdbc/\
10.7.0/kafka-connect-jdbc-10.7.0.jar

sudo mkdir /usr/lib/kafka/plugin
sudo cp kafka-connect-jdbc-10.7.0.jar /usr/lib/kafka/plugin

vim connect-standalone.properties
```

Plik connect-standalone.properties należy uzupełnić następującą treścią i zapisać:

```
plugin.path=/usr/lib/kafka/plugin
bootstrap.servers=localhost:9092
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.json.JsonConverter
key.converter.schemas.enable=false
value.converter.schemas.enable=true
offset.storage.file.filename=/tmp/connect.offsets
offset.flush.interval.ms=10000

vim connect-jdbc-sink.properties
```

Plik connect-jdbc-sink.properties należy uzupełnić następującą treścią i zapisać:

```
connection.url=jdbc:mysql://localhost:6033/kafkadatabase
connection.user=streamuser
connection.password=stream

tasks.max=1
name=kafka-to-mysql-task
connector.class=io.confluent.connect.jdbc.JdbcSinkConnector
topics=kafka-output
table.name.format=data_sink
delete.enabled=true
pk.mode=record_key
pk.fields=id
```

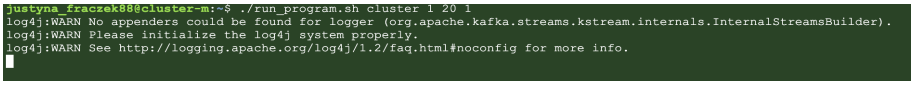
```
sudo cp /usr/lib/kafka/config/tools-log4j.properties \
 /usr/lib/kafka/config/connect-log4j.properties

echo "log4j.logger.org.reflections=ERROR" | \
 sudo tee -a /usr/lib/kafka/config/connect-log4j.properties

/usr/lib/kafka/bin/connect-standalone.sh connect-standalone.properties \
 connect-jdbc-sink.properties
```

W ten sposób uruchomiliśmy Kafkę Connect w trybie standalone. teraz w nowym terminalu możemy uruchomić program.

- skrypt uruchamiający program - run_program.sh

Sposób uruchomienia: sh run_program.sh <nazwa klastra> <D> <P> <DELAY>

Przykładowe uruchomienie:



gdzie D to długość okresu czasu wyrażoną w dniach, P to stosunek (minimalny) różnicy pomiędzy najwyższym kursem akcji danej spółki a najniższym do najwyższego kursu, a DELAY to tryb w jakim ma działać aplikacja, podawany w postaci int jako 1 lub 2, 1 dla tryba A, 2 dla trybu C.

7 Miejsce utrzymywania obrazów czasu rzeczywistego – skrypt tworzący

- skrypt tworzący miejsce utrzymywania obrazów czasu rzeczywistego - configure_mysql_docker.sh

Należy nadać uprawnienia: chmod +x configure_mysql_docker.sh

Ten skrypt tworzy kontener dockerowy MySQL.

Teraz stworzymy usera streamuser z hasłem stream, który posłuży nam za klienta do stworzenia tabeli aggData, oraz stworzymy bazę danych kafkadatabase.

docker exec -it mymysql bash

mysql -uroot -pmy-secret-pw

```
CREATE USER 'streamuser'@'%' IDENTIFIED BY 'stream';
CREATE DATABASE IF NOT EXISTS kafkadatabase CHARACTER SET utf8;
GRANT ALL ON kafkadatabase.* TO 'streamuser'@'%';
```

Wychodzimy z mysql poleceniem exit;

Teraz zalogujemy się jako użytkownik, którego właśnie stworzyliśmy:

mysql -ustreamuser -pstream kafkadatabase

Teraz jak znajdujemy się w konsoli mysql to należy wykonać następujące polecenie:

```
CREATE TABLE aggData (id INT AUTO_INCREMENT PRIMARY KEY,
key_col VARCHAR(255),average_close DECIMAL(10, 2),
min_low DECIMAL(10, 2),
max_high DECIMAL(10, 2),
sum_volume DECIMAL(15, 4));
```

Mamy już gotową tabelę, w której będą przechowywane dane. Pamiętaj żeby wcześniej przed uruchomieniem programu skonfigurować także Kafka Connect (opisane w punkcie 6)

Ten terminal z terminalem klienta w mysql może zostać otwarty, z niego będzie można odczytywać potem zagregowane dane z tabeli poleceniem:

SELECT * FROM aggData;

7.1 Zrzuty ekranu wykonanych powyższych poleceń w konsoli

```
justyna_fraczek88@cluster-m:~$ docker exec -it mymysql bash
root@19c3bf3079c3:/# mysql -uroot -pmy-secret-pw
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE USER 'streamuser'@'%' IDENTIFIED BY 'stream';
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE DATABASE IF NOT EXISTS kafkadatabase CHARACTER SET utf8;
Query OK, 1 row affected, 1 warning (0.01 sec)

mysql> GRANT ALL ON kafkadatabase.* TO 'streamuser'@'%';
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> exit;
Bye
root@19c3bf3079c3:/# mysql -ustreamuser -pstream kafkadatabase
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE TABLE aggData (id INT AUTO_INCREMENT PRIMARY KEY,
    -> key_col VARCHAR(255),average_close DECIMAL(10, 2),
    -> min_low DECIMAL(10, 2),
    -> max_high DECIMAL(10, 2),
    -> sum_volume DECIMAL(15, 4));
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT * FROM aggData;
Empty set (0.01 sec)

mysql>
```

```
aggResult.foreach((key, value) -> {
    try (Connection connection = DriverManager.getConnection(MYSQL_URL, MYSQL_USERNAME, MYSQL_PASSWORD)) {
        PreparedStatement statement = connection
                .prepareStatement( sql: "INSERT INTO " + TABLE_NAME + "(key_col, average_close, min_low, max_high, sum_volume) VALUES (?, ?, ?, ?, ?)");
        statement.setString( parameterIndex: 1, key);
        statement.setDouble( parameterIndex: 2, value.averageClose);
        statement.setDouble( parameterIndex: 3, value.minLow);
        statement.setDouble( parameterIndex: 4, value.maxHigh);
        statement.setDouble( parameterIndex: 5, value.sumVolume);
        statement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
});
```

Wstawianie zagregowanych danych do tabeli w bazie mysql - fragment kodu

8 Miejsce utrzymywania obrazów czasu rzeczywistego – cechy

Obrazy czasu rzeczywistego są zapisywane do bazy danych MySQL.

Baza MySQL jest dobrym wyborem jako miejsce utrzymywania obrazu czasu rzeczywistego dla danych przetwarzanych przez Kafka Streaming ze względu na swoje solidne wsparcie dla transakcji, skalowalność, wydajność, bogate możliwości zapytań, dojrzałość i stabilność oraz integrację z Kafka.

9 Konsument: skrypt odczytujący wyniki przetwarzania

Wyniki anomalii są zapisywane do tematu wyjściowego kafki, a wyniki obrazu czasu rzeczywistego do bazy danych.

9.1 skrypt odczytujący wyniki obrazu czasu rzeczywistego z bazy danych - read_from_database.sh

Polecenie SELECT * FROM aggData; z terminala klienta w mysql (opisane w punkcie 7)

9.2 skrypt odczytujący wyniki anomalii z wynikowego tematu kafki - odczytanie_anomalii.sh

Sposób uruchomienia: sh odczytanie_anomalii.sh

9.3 Zrzuty ekranu przedstawiające przykładowe wyniki (lokalnie i z GCP):

9.3.1 Agregacje (wyniki po uruchomieniu read_from_database.sh):

id	key_col	average_close	min_low	max_high	sum_volume
1	DIS	140.18	113.21	148.20	2225498600.00
2	HPQ	21.66	20.17	23.93	2424174100.00
3	ARNC	30.92	28.16	34.27	871776100.00
4	AA	16.98	12.83	21.86	1134272800.00
5	MRO	11.81	7.54	14.07	2858444100.00
6	MO	47.51	38.57	51.78	1958558500.00
7	IP	42.93	35.60	46.55	571103800.00
8	BA	324.76	269.60	349.95	1471954200.00
9	HON	176.92	150.62	184.06	525933200.00
10	CUZ	40.91	34.81	42.99	132837200.00
11	CMCSA	44.88	38.97	47.74	3991504300.00
12	KO	57.29	51.58	60.13	2790996000.00
13	CNP	26.25	22.48	27.53	1112333800.00
14	JNJ	147.28	130.82	154.50	1527245000.00
15	AXP	128.22	107.00	138.13	762353400.00
16	CAT	139.11	119.03	150.55	702546000.00
17	DLX	45.14	32.27	50.40	60378700.00
18	UNP	180.02	152.79	188.96	603012800.00
19	KBAL	19.21	15.26	21.76	19422200.00
20	CMTL	33.44	27.21	37.34	44821100.00
21	BK	46.69	38.10	51.60	1168441700.00
22	BMY	64.63	56.37	68.34	2924106700.00
23	DD	55.62	42.06	65.16	1525259800.00
24	MCD	209.75	188.81	218.39	758950200.00
25	DXC	32.34	21.83	38.37	795013800.00
26	PNR	44.43	37.39	47.43	232409700.00
27	ORI	22.54	19.22	23.62	327412800.00
28	MUX	1.17	0.83	1.37	689994300.00
29	Y	798.18	666.69	847.95	10191600.00

79214	DTE	27.60	17.13	38.63	20251500.0000
79215	MMM	7.86	4.44	11.11	261108000.0000
79216	NYT	1.05	0.99	1.14	351600.0000
79217	MDT	0.35	0.32	0.38	32742400.0000
79218	CL	1.92	1.72	2.03	23537600.0000
79219	BAC	4.69	4.44	4.94	5036800.0000
79220	SIF	2.23	2.00	2.42	95700.0000
79221	JCP	21.85	19.13	25.25	5241200.0000
79222	MKC	1.48	1.22	1.75	6070400.0000
79223	TER	1.01	0.83	1.21	2976000.0000
79224	OGE	6.54	6.25	6.81	2053200.0000
79225	NEE	9.35	8.31	10.00	4788400.0000
79226	AEM	4.29	3.38	6.38	1913900.0000
79227	VMI	0.52	0.47	0.56	569600.0000
79228	SXI	2.08	1.83	2.31	390400.0000
79229	SCX	9.65	8.13	12.50	176800.0000
79230	ALX	5.34	3.88	6.75	826200.0000
79231	MSA	0.71	0.52	0.84	2664000.0000
79232	PHI	0.34	0.29	0.36	808000.0000
79233	TXN	1.81	1.55	2.10	360225600.0000
79234	SNE	6.66	5.80	7.93	21325800.0000
79235	SVT	4.19	3.59	4.65	64000.0000
79236	AWR	2.32	2.21	2.44	328200.0000
79237	TGT	0.38	0.34	0.44	36412800.0000
79238	NFG	2.57	2.40	2.67	362400.0000
79239	EGP	13.59	12.00	14.25	142900.0000
79240	ECL	2.33	2.08	2.50	12985600.0000
79241	IDA	14.89	14.25	15.63	895800.0000
79242	SR	5.36	5.03	5.63	240000.0000
79243	ALCO	3.78	3.56	4.25	799200.0000
79244	ARNC	6.21	3.37	8.87	361021800.0000
79245	IP	8.13	5.47	11.34	315906500.0000
79246	BA	1.27	0.40	3.70	3106416800.0000
79247	F	2.19	1.80	2.67	495604500.0000
79248	AXP	3.80	3.11	4.48	97206100.0000
79249	CAT	3.35	1.23	5.89	675702400.0000
79250	DXC	0.39	0.19	0.63	9034603.0000
79251	PNR	0.98	0.87	1.16	656300.0000
79252	WMT	0.06	0.04	0.07	286822400.0000
79253	CLX	4.28	2.31	5.66	28940000.0000
79254	WHR	14.87	12.25	17.00	2800600.0000

9.3.2 Anomalie (wyniki po uruchomieniu odczytanie_anomalii.sh):


```
CLGN:{"stock":"CLGN","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":5.624000072479248,"max_score":14.550000190734863,"difference":10.451612164897304}
NIO:{"stock":"NIO","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":3.130000114440918,"max_score":5.650000095367432,"difference":13.941018184114624}
INOD:{"stock":"INOD","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":1.100000023841858,"max_score":1.25,"difference":10.399999618530273}
HUSA:{"stock":"HUSA","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.14000000059604645,"max_score":0.2800000011920929,"difference":28.571427811165247}
NGD:{"stock":"NGD","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.7400000095367432,"max_score":1.0700000524520874,"difference":11.76470656983444}
CORV:{"stock":"CORV","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.20999999344348907,"max_score":0.6499999761581421,"difference":14.583332169180089}
SNSS:{"stock":"SNSS","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.3400000035762787,"max_score":1.1299999952316284,"difference":13.953488694454537}
CBAT:{"stock":"CBAT","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.44999998807907104,"max_score":1.1599999666213999,"difference":10.909091106131052}
CBLI:{"stock":"CBLI","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.5199999809265137,"max_score":5.0,"difference":12.499997671693512}
AEZS:{"stock":"AEZS","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.7699999809265137,"max_score":1.5399999618530273,"difference":10.204083866896262}
SSNT:{"stock":"SSNT","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":2.690000057220459,"max_score":4.820000171661377,"difference":14.41860689479886}
BIOC:{"stock":"BIOC","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.2800000011920929,"max_score":0.7900000214576721,"difference":22.500000558793538}
ONTX:{"stock":"ONTX","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.2899999916553497 3,"max_score":0.75,"difference":18.000000715255737}
LIFE:{"stock":"LIFE","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":3.4700000286102295,"max_score":7.619999885559082,"difference":16.639999389648438}
TRPX:{"stock":"TRPX","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":0.4050000011920929,"max_score":1.1380000114440918,"difference":13.513516561230885}
REKR:{"stock":"REKR","start_ts":"2023-06-04T00:00:00Z","end_ts":"2023-06-05T00:00:00Z","min_score":3.2300000190734863,"max_score":4.829999923706055,"difference":13.813459733270701}
```

```
HPQ:{"stock":"HPQ","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.07539311051368713,"max_score":0.13127270340919495,"difference":12.371128753344074}
MO:{"stock":"MO","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.1189236119389534,"max_score":0.1953125,"difference":12.31999724902337}
GT:{"stock":"GT","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":7.875,"max_score":11.4375,"difference":15.25974025974026}
HPQ:{"stock":"HPQ","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.07140170782804489,"max_score":0.13127270340919495,"difference":16.145832467451672}
KO:{"stock":"KO","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.1822916716337204,"max_score":0.2701822817325592,"difference":13.312689088819972}
HPQ:{"stock":"HPQ","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.06696681678295135,"max_score":0.13127270340919495,"difference":12.716767608653768}
HPQ:{"stock":"HPQ","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.056766577064991,"max_score":0.13127270340919495,"difference":10.48950991442376}
HPQ:{"stock":"HPQ","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.056766577064991,"max_score":0.13127270340919495,"difference":10.48950991442376}
HPQ:{"stock":"HPQ","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.054105643182992935,"max_score":0.13127270340919495,"difference":13.705584448964979}
BA:{"stock":"BA","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.5802469253540039,"max_score":0.9320987462997437,"difference":13.761468214997128}
IBM:{"stock":"IBM","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":4.733333110809326,"max_score":7.7133331298828125,"difference":10.579352621515774}
BA:{"stock":"BA","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":0.5781893134117126,"max_score":0.9320987462997437,"difference":23.224044840342}
CAT:{"stock":"CAT","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":1.2916666269302368,"max_score":1.7708333730697632,"difference":10.144930300505266}
DTE:{"stock":"DTE","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":22.75,"max_score":30.6875,"difference":11.650485436893204}
IBM:{"stock":"IBM","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":4.0,"max_score":7.7133331298828125,"difference":10.16442542512812}
XOM:{"stock":"XOM","start_ts":"2023-06-05T00:00:00Z","end_ts":"2023-06-06T00:00:00Z","min_score":1.41796875,"max_score":1.7578125,"difference":11.029411764705882}
```