**Please note: You need to either have a local editor open - because you will need to modify some of the commands before paste in the CLI in order to personalize the topics name - OR copy locally this file and make modifications directly in it.**

**Working with Kafka CLI**

**ssh student@learn.academyofdata.com**

**Passw: OroJul21**

**We work with Confluent Open Source**

**TO ENTER KSQL TYPE BELOW COMMAND IN CONFLUENT OPEN SOURCE CLI:**

LOG\_DIR=./tmp/ksql\_logs opt/confluent/bin/ksql

LIST FUNCTIONS; // to see all the functions in KSQL

**To see all topics:**

SHOW TOPICS ;

**Exercise 1: get started with KSQL**

**STOP HERE and LETS CREATE A TOPIC YOUR\_NAME IN WHICH YOU WILL CONSTANTLY TYPE DATA AND LETS SEE IF IT’S VISIBLE IN KSQL. IN THE TOPIC WE WOULD LIKE TO HAVE DATA: NAME,COUNTRY**

**Example of create topic and produce in Confluent:**

/opt/confluent/bin/kafka-topics --zookeeper localhost:2181 --create --partitions 4 --replication-factor 1 --topic your\_name

/opt/confluent/bin/kafka-topics --zookeeper localhost:2181 --list

/opt/confluent/bin/kafka-console-producer --broker-list localhost:9092 --topic your\_name

**>maria,uk**

**>gabriel,us**

**>john,uk**

**>ion,ro**

**>tad,fr**

**>felix,ro**

**>maria,fr**

**>traian,ro**

**>george,fr**

**Recommendation: have two windows open - one with the topic producer and one with KSQL, in order to easily monitor the data being produced and what you see in KSQL.**

**Now, in KSQL to see the content of a topic from the beginning of the topic:**

Print ‘your\_name’; //you should see in real time what you are typing in the producer console. Please note - of your topic name doesn’t contain characters like - or \_ then you can use prin topicname without quotes.

print 'your\_name' from beginning;

print ‘your\_name’ from beginning limit 1;

print ‘your\_name’ from beginning interval 2 limit 2;

print ‘your\_name’ from beginning sample 3;

OR JUST

Print ‘your\_name’; // but if no new events nothing will be shown if no new data is entered in the topic

**Use CTRL+C to stop the query if needed (if you use limit the query will stop automatically after the limit is passed).**

**Using KSQL lets create a stream based on the ‘your\_name’ topic.**

create stream your\_name\_stream (name VARCHAR,country VARCHAR) with (KAFKA\_TOPIC='your\_name', VALUE\_FORMAT='DELIMITED');

LIST STREAMS; //SHOW STREAMS works as well

Describe extended your\_name\_stream; //you can see the underlying topic and the replication factor and partitions defined

select \* from your\_name\_stream emit changes; // nothing will show. Start typing data in the topic and you will see data in the stream as well.

**Note - regarding the emit changes syntax:**

**Pull queries**: query the current state of the system, return a result, and terminate.

**Push queries**: query the state of the system in motion and continue to output results until they meet a LIMIT condition or are terminated by the user.

**'EMIT CHANGES' i**s used to indicate a query is a push query. To convert a pull query into a push query, which was the default behavior in older versions of KSQL, add `EMIT CHANGES` to the end of the statement before any LIMIT clause.

**Back to our stream, if you would like to see the data from beginning of the topic - you need to indicate that you would like to start reading from the earliest offset:**

SET 'auto.offset.reset'='earliest'; // Determines what to do when there is no initial offset in Apache Kafka or if the current offset does not exist on the server. The default value in KSQL is latest, which means all Kafka topics are read from the latest available offset. This setting is per CLI session.

select \* from your\_name\_stream emit changes;

select \* from your\_name\_stream emit changes limit 3;

**Lets try some operations and aggregations**

select \* from your\_name\_stream where country='ro' emit changes;

select country,count(\*) from your\_name\_stream group by country emit changes;

**Start typing more data into your topic and see how the result changes.**

**--- stop here -----**

**Let’s see the time format in a more readable way**

select TIMESTAMPTOSTRING(rowtime, 'yyyy-MM-dd HH:mm:ss.SSS'),country,name from your\_name\_stream emit changes;

**Note: If we would like to drop the stream and the underlying topic:**

Drop stream name\_stream; // we will need our stream moving forward so don’t drop it just yet, or drop but you need to redefine it

Drop stream if exists name\_stream delete topic;

Show streams;

Show topics;

**----------------stop here -----**

**Our stream has no key - let’s create another stream with key from this one:**

CREATE STREAM name\_stream\_key \

WITH(KAFKA\_TOPIC='name') AS \

SELECT CAST(country as VARCHAR) as country, name \

FROM your\_name\_stream \

PARTITION BY country;

select \* from name\_stream\_key where rowkey = 'uk' emit changes; // lets see all people from uk

**Lets try as well an aggregation, lets see how many people per country are registered:**

select country, count(\*) from name\_stream\_key group by country emit changes;

**Before we move on with the exercises, let’s stop and see a detailed explanation of the last stream created:**

Describe extended name\_stream\_key; **// you can see which query writes into this stream, you can see the underlying topic created by this stream (yourname\_stream\_key). If you try to drop this stream it will not work:**

Show queries; **// you can see all queries including your query**

Now, if you try to drop the recently created stream - it will not work:

Drop stream name\_stream\_key; **// cannot drop this stream since you have running queries that read/write into it.**

**Cannot drop NAME\_STREAM\_KEY.**

**The following queries read from this source: [].**

**The following queries write into this source: [CSAS\_NAME\_STREAM\_KEY\_326].**

**You need to terminate them before dropping NAME\_STREAM\_KEY.**

**// so in order to drop NAME\_STREAM\_KEY you need to stop the query writing into it.**

Show queries; //will show you all running queries. **If you want to stop a query: Terminate query\_name;**

**Also, lets see the topic created by this new stream: you can see the underlying topic in describe extended applied on your stream.**

Print name from beginning;

**---stop here---**

**Let’s create a table that stores the result of the count aggregations per country (aggregation on a stream results in a table):**

Create table name\_table as select country, count(\*) from name\_stream\_key group by country;

select \* from name\_table emit changes;

select \* from name\_table where rowkey = 'nz';

Insert another person in nz and redo the query. The query was terminated so this new event is not visible in the console.

Describe extended name\_table; **//you can see that create table -- created a new Kafka topic named the same as your table name**

**----------stop here---**

**Now, based on the topic created by name\_stream\_key (which is a topic with key - in our example is called name) we can build a table that will show only the latest value per each key - in this case the key is country. Let’s create this table based on an existing topic -- and this table will basically record only the last entries in each country:**

CREATE TABLE name\_table\_1 (country\_string VARCHAR, name VARCHAR) \

WITH (KAFKA\_TOPIC='name', \

VALUE\_FORMAT='DELIMITED', \

KEY='country\_string');

SELECT \* FROM name\_table\_1 emit changes;

SELECT \* FROM name\_table\_1 where rowkey=’nz’ emit changes;

**-----------stop here ----**

**DEMO topic,**

**mere,ana,4**

**Cine consuma cele mai multe fructe**

**Care fructe sunt cele mai populare**

**Now. let’s perform a join:**

CREATE TABLE countries\_list (ROWKEY VARCHAR KEY, city VARCHAR) WITH (kafka\_topic='countries-all', partitions=4, value\_format='DELIMITED');

INSERT INTO countries\_list (rowkey, city) VALUES ('fr', 'Paris');

INSERT INTO countries\_list (rowkey, city) VALUES ('ro', 'Bucharest');

INSERT INTO countries\_list (rowkey, city) VALUES ('nz', 'Auckland');

INSERT INTO countries\_list (rowkey, city) VALUES ('uk', 'London');

INSERT INTO countries\_list (rowkey, city) VALUES ('ir', 'Dublin');

select \* from countries\_list emit changes;

INSERT INTO name\_stream\_key (rowkey, country, name) VALUES ('ir', 'ir', 'Joe');

INSERT INTO name\_stream\_key (rowkey, country, name) VALUES ('fr', 'fr', 'Edith');

INSERT INTO name\_stream\_key (rowkey, country, name) VALUES ('ro', 'ro', 'Gina');

select \* from name\_stream\_key LEFT JOIN countries\_list ON name\_stream\_key.rowkey = countries\_list.rowkey emit changes;

create stream joined\_data as SELECT name\_stream\_key.name as name, name\_stream\_key.country as country, countries\_list.city as city from name\_stream\_key LEFT JOIN countries\_list ON name\_stream\_key.rowkey = countries\_list.rowkey;

select \* from joined\_data emit changes;

**Window queries:**

select country,count(country) as count \

from name\_stream\_key \

WINDOW TUMBLING (SIZE 1 MINUTE) \

group by country emit changes;