**SCORPIO CONTEXT BROKER**

**RUNNING CONTAINERS WITH DOCKER COMPOSE**

**# download the docker-compose file with the below mentioned url**

<https://github.com/ScorpioBroker/ScorpioBroker/blob/master/docker-compose-aaio.yml>

**# Change KAFKA\_ADVERTISED\_HOST\_NAME environment variable with your system IP in environment section of docker-compose file**

- KAFKA\_ADVERTISED\_HOST\_NAME: <YOUR IP HERE>

**CHECKING CONSUMER AND PRODUCER APIs (using DOCKER CONTAINERS)**

**# Create a topic**

$ docker-compose exec kafka bash -c '$KAFKA\_HOME/bin/kafka-topics.sh --create --topic <topic\_name> --partitions 1 --zookeeper $KAFKA\_ZOOKEEPER\_CONNECT --replication-factor 1'

**# Deleting a topic**

$ docker-compose exec kafka bash -c '$KAFKA\_HOME/bin/kafka-topics.sh --delete --topic <topic\_name>

**# Describe Topic**

$ docker-compose exec kafka bash -c '$KAFKA\_HOME/bin/kafka-topics.sh --describe --topic <topic\_name> --zookeeper $KAFKA\_ZOOKEEPER\_CONNECT'

**# Starting a Producer Console**

$ docker-compose exec kafka bash -c '$KAFKA\_HOME/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic <topic\_name>'

**# Starting a Consumer Console**

$ docker-compose exec kafka bash -c '$KAFKA\_HOME/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic <topic\_name> --from-beginning'

**CHECKING CONSUMER AND PRODUCER APIs (without DOCKER CONTAINERS)**

**# Download Kafka Libraries and Config files from below link**

http://mirrors.estointernet.in/apache/kafka/2.4.0/kafka\_2.11-2.4.0.tgz

**# Moving onto the directory**

$ cd kafka\_2.11-2.4.0

**# Starting Zookeeper Service**

$ ./bin/zookeeper-server-start.sh config/zookeeper.properties

**# Starting Kafka Server**

$ ./bin/kafka-server-start.sh config/server.properties

**# Creating a Topic**

$ ./bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic <topic\_name>

**# Deleting a topic**

$ ./bin/kafka-topics.sh --zookeeper localhost:2181 --delete --topic <topic\_name>

**# Describe Topics**

$ ./bin/kafka-topics.sh --describe --zookeeper localhost:2181

**# Starting producer console**

$ ./bin/kafka-console-producer.sh --broker-list localhost:9092 --topic <topic\_name>

**# Starting Consumer Console**

$ ./bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic chat --from-beginning

**STREAM API WORKFLOW**

**# start zookeeper service**

$ ./bin/zookeeper-server-start.sh config/zookeeper.properties

**# start kafka server**

./bin/kafka-server-start.sh config/server.properties

**# create input topic**

$ ./bin/kafka-topics.sh --create \

--bootstrap-server localhost:9092 \

--replication-factor 1 \

--partitions 1 \

--topic streams-plaintext-input

**# create output topic**

$ ./bin/kafka-topics.sh --create \

--bootstrap-server localhost:9092 \

--replication-factor 1 \

--partitions 1 \

--topic streams-wordcount-output \

--config cleanup.policy=compact

**# describing topic**

$ ./bin/kafka-topics.sh --bootstrap-server localhost:9092 --describe

**# Executing script for stream processing**

$ ./bin/kafka-run-class.sh org.apache.kafka.streams.examples.wordcount.WordCountDemo

**# Starting Producer Console with Input Topic**

$ ./bin/kafka-console-producer.sh --broker-list localhost:9092 --topic streams-plaintext-input

**# Starting Consumer Console with Output Topic**

$ ./bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 \

--topic streams-wordcount-output \

--from-beginning \

--formatter kafka.tools.DefaultMessageFormatter \

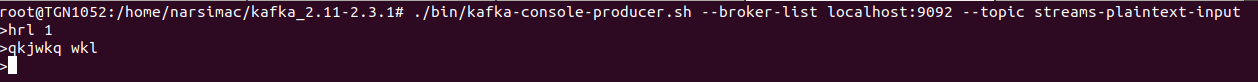
--property print.key=true \

--property print.value=true \

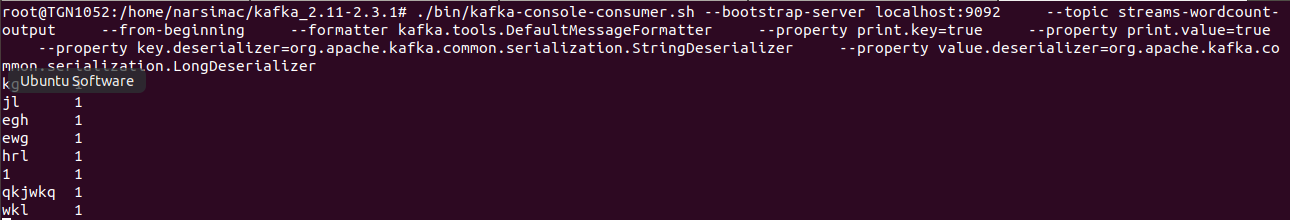
--property key.deserializer=org.apache.kafka.common.serialization.StringDeserializer \

--property value.deserializer=org.apache.kafka.common.serialization.LongDeserializer

**PRODUCER CONSOLE**



**CONSUMER CONSOLE**



**# Inserting an Entity on Scorpio Broker**

$ entity='{

"id": "urn:ngsi-ld:testunit:15",

"type": "AirQualityObserved",

"dateObserved": {

"type": "Property",

"value": {

"@type": "DateTime",

"@value": "2018-08-07T12:00:00Z"

}

},

"NO2": {

"type": "Property",

"value": 23,

"unitCode": "GP",

"accuracy": {

"type": "Property",

"value": 0.95

}

},

"refPointOfInterest": {

"type": "Relationship",

"object": "urn:ngsi-ld:PointOfInterest:RZ:MainSquare"

},

"@context": [

"<https://schema.lab.fiware.org/ld/context>",

"<https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld>"

]

}'

$ curl -iX POST http://localhost:9090/ngsi-ld/v1/entities -H 'Content-Type: application/ld+json' -d "$entity"

**VALIDATING DATA ON POSTGRES**

**# Executing Bash on Postgres Container**

$ docker exec –it <postgres\_container\_id> bash

**# Logging into Postgres user (Default user:- ngb, Default Password :- ngb, Default DB :- ngb)**

$ psql -U ngb -d ngb -W

**# Get Data of Entity**

$ select \* from entity;

**ADDITIONAL COMMANDS FOR POSTGRES**

**# to use db**

$ \c <db\_name>

**# to connect db as this user**

$ \c <db\_name> <user\_name>

**# list of databases**

$ \l

**# list of tables in db**

$ \dt

**# describe table**

$ \d <table\_name>

**# show schemas of selected db**

$ \dn

**# available functions in db**

$ \df

**# shows available views in db**

$ \dv

**# shows all the users**

$ \du

**# shows the current version of postgres**

$ SELECT version();

**# to execute the previous command**

$ \g

**# list the command history**

$ \s

**# listing the available commands**

$ \?

**# turning on the query execution time**

$ \timing

**# to quit psql**

$ \q

**# get all data from table**

$ SELECT \* FROM <table\_name>;

**# database creation**

$ create database ngb;

**# making user a superuser**

$ create user ngb with encrypted password 'ngb';

$alter user ngb with superuser;

**# granting all access to user for db**

$ grant all privileges on database ngb to ngb;