Useful Commands

We **recommend** doing all the **hands on work** including registering for Instaclustr account **from Ubuntu VM** image that was provided to you. If you still don't have this image then **ask** and we will provide the image ova file to you. Opening Instaclustr platform console website in VM makes it easy to copy connection credentials when connecting to C* or Kafka clusters.

Signup for Instaclustr account

- Go to https://www.instaclustr.com/
- Sign up for a new account https://console.instaclustr.com/user/signup

Using Instaclustr Managed C* cluster

- Connect to cluster using already downloaded cqlsh tool using commands below

```
cd ~/Downloads/apache-cassandra-3.11.4/bin/

./cqlsh -u iccassandra <your cluster public IP>
```

  Enter password when prompted

- Creating Keyspace

```
CREATE KEYSPACE training_ks WITH replication = {'class':
'NetworkTopologyStrategy', '<DC name e.g. AWS_VPC_US_WEST_2>':
'3'};

DESCRIBE training_ks;
```

- Creating Table

```
CREATE TABLE training_ks.sensor_events (
      sensor_id text,
      event_ts timestamp,
      reading double,
      PRIMARY KEY (sensor_id, event_ts));
```

- Inserting records

```
INSERT INTO training_ks.sensor_events (sensor_id, event_ts,
reading)
VALUES ('S123456','2019-04-25T10:02:03.123',5.67);
INSERT INTO training_ks.sensor_events (sensor_id, event_ts,
reading)
VALUES ('S123456','2019-04-25T10:02:03.124',2.34);
INSERT INTO training_ks.sensor_events (sensor_id, event_ts,
reading)
```

```
VALUES ('S123457','2019-04-26T10:03:03.124',4.56);
```

- Selecting records
```
SELECT * FROM training_ks.sensor_events;
```

Using Instaclustr Managed Kafka Cluster

- Create kafka.properties file
```
cd ~/Downloads/kafka_client/bin/
```

Open an editor using command below
```
nano kafka.properties
```

Copy paste the following text into nano editor

```
security.protocol=SASL_PLAINTEXT
sasl.mechanism=SCRAM-SHA-256
sasl.jaas.config=org.apache.kafka.common.security.scram.ScramL
oginModule required \
    username="ickafka" \
    password="[USER PASSWORD]";
```

Save kafka.properties file.
NOTE: you can also use ***vi*** editor

- List all topics in cluster
```
./ic-kafka-topics.sh --bootstrap-server <cluster public
IP>:9092 --properties-file kafka.properties --list
```

- Create new topic
```
./ic-kafka-topics.sh --bootstrap-server <cliuster public
IP>:9092 --properties-file kafka.properties --create --topic
events --replication-factor 3 --partitions 3
```

You can list topics again and ensure new topic *'events'* is there

- Start Kafka console consumer
```
./ic-kafka-topics.sh --bootstrap-server <your_cluster_IP>:9092
--properties-file kafka.properties --list
```

Developing Java based Producer and Consumers

You can get the code repository locally by using following commands

```
mkdir -p ~/dev/training
cd ~/dev/training/
git clone https://github.com/dwivedialok/eventsproducer.git
git clone https://github.com/dwivedialok/eventsconsumer.git
```

Now you can import these projects into IntelliJ using pom.xml file and using “Import project” option of IntelliJ. However, we suggest creating project step by step as will be shown in the hands-on exercise session in the workshop. For the ease of referring to existing code, we are providing URLs of code repos on github.com

https://github.com/dwivedialok/eventsproducer
https://github.com/dwivedialok/eventsconsumer