**MQT Lab 9**

**Lab 3**

1. Create two topics:

topic item in the left part of the menu, above ksqlDB.

Add topic --> ConfluentTextLinesTopic --> partitions 10

Add topic --> UppercasedTextLinesTopic --partitions 10

2. Produce some low-case values into ConfluentTextLinesTopic

topic item in left part of menu, above ksqlDB.

choose --> ConfluentTextLinesTopic --> Messages tab --> "+ Produce a new message to this topic"

"alpha" --> Produce

"beta" --> Produce

3. Run KSQL

ksqlDB item in left part of menu, under Topics.

ksqlDB\_cluster\_0 --> Editor

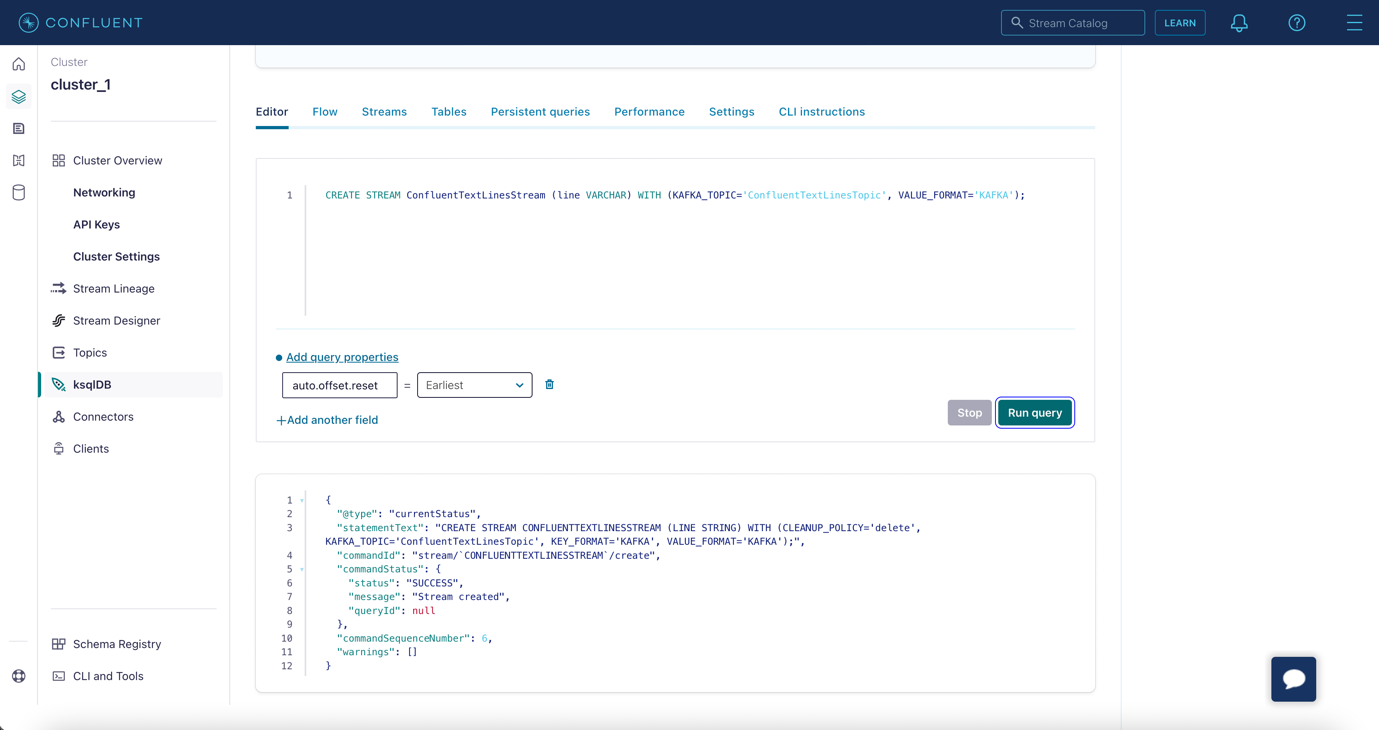
(or the name of cluster that you have created)

4. Run the commands inside KSQL

A screenshot of a computer

Description automatically generated

CREATE STREAM ConfluentTextLinesStream (line VARCHAR) WITH (KAFKA\_TOPIC='ConfluentTextLinesTopic', VALUE\_FORMAT='KAFKA');

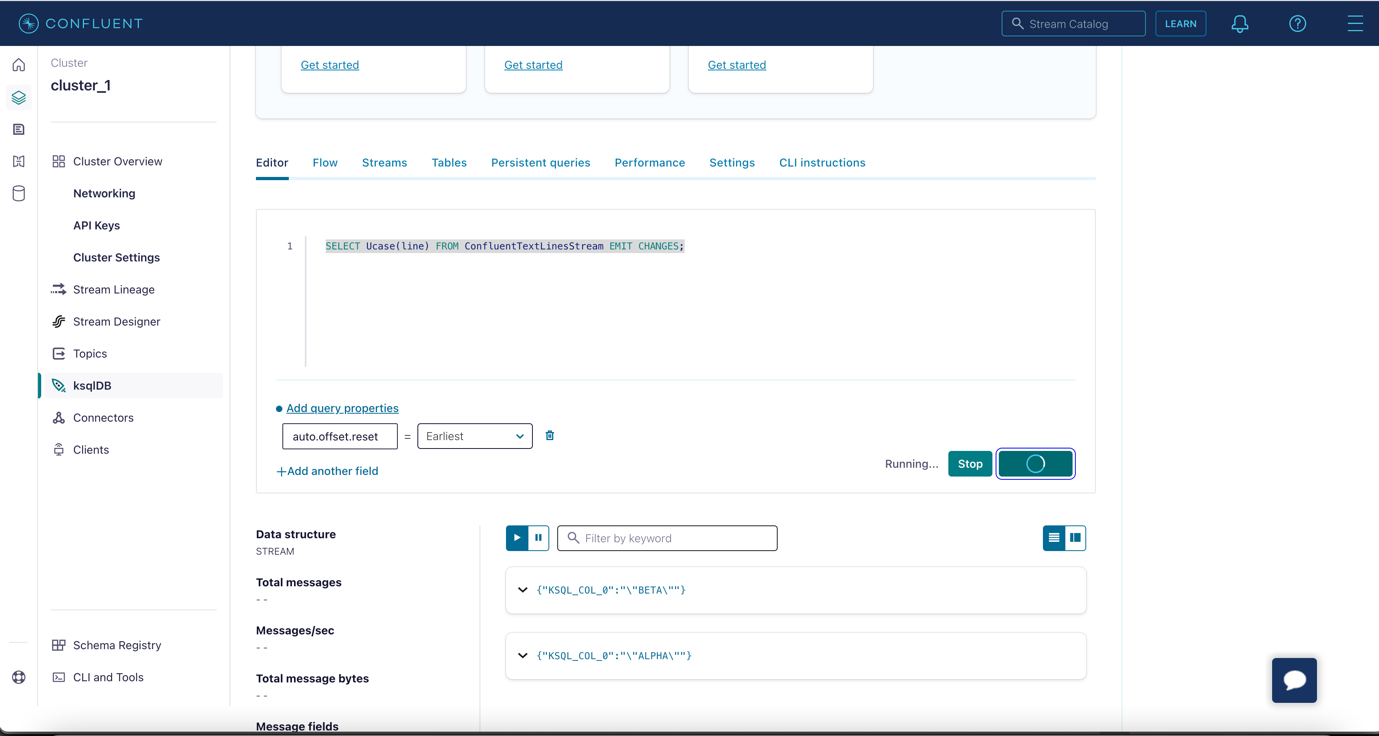


SELECT \* FROM ConfluentTextLinesStream EMIT CHANGES;

A screenshot of a computer

Description automatically generated

SELECT Ucase(line) FROM ConfluentTextLinesStream EMIT CHANGES;



CREATE STREAM UppercasedTextLinesStream WITH (KAFKA\_TOPIC='UppercasedTextLinesTopic', VALUE\_FORMAT='KAFKA') AS SELECT Ucase(line) FROM ConfluentTextLinesStream;

A screenshot of a computer

Description automatically generated

5. Check UppercasedTextLinesTopic topic;

SELECT \* FROM UppercasedTextLinesStream EMIT CHANGES;

A screenshot of a computer

Description automatically generated

**Lab 4**

1. Create topic:

readings --> partitions 6

2. create stream for topic readings

CREATE STREAM readings (

sensor VARCHAR KEY,

val DOUBLE,

location VARCHAR) WITH (

kafka\_topic='readings',

partitions=6,

value\_format='JSON');

A screenshot of a computer

Description automatically generated

3. Produce some events into readings topic

INSERT INTO readings (sensor, val, location) VALUES ('sensor-1', 45, 'wheel');

INSERT INTO readings (sensor, val, location) VALUES ('sensor-2', 41, 'motor');

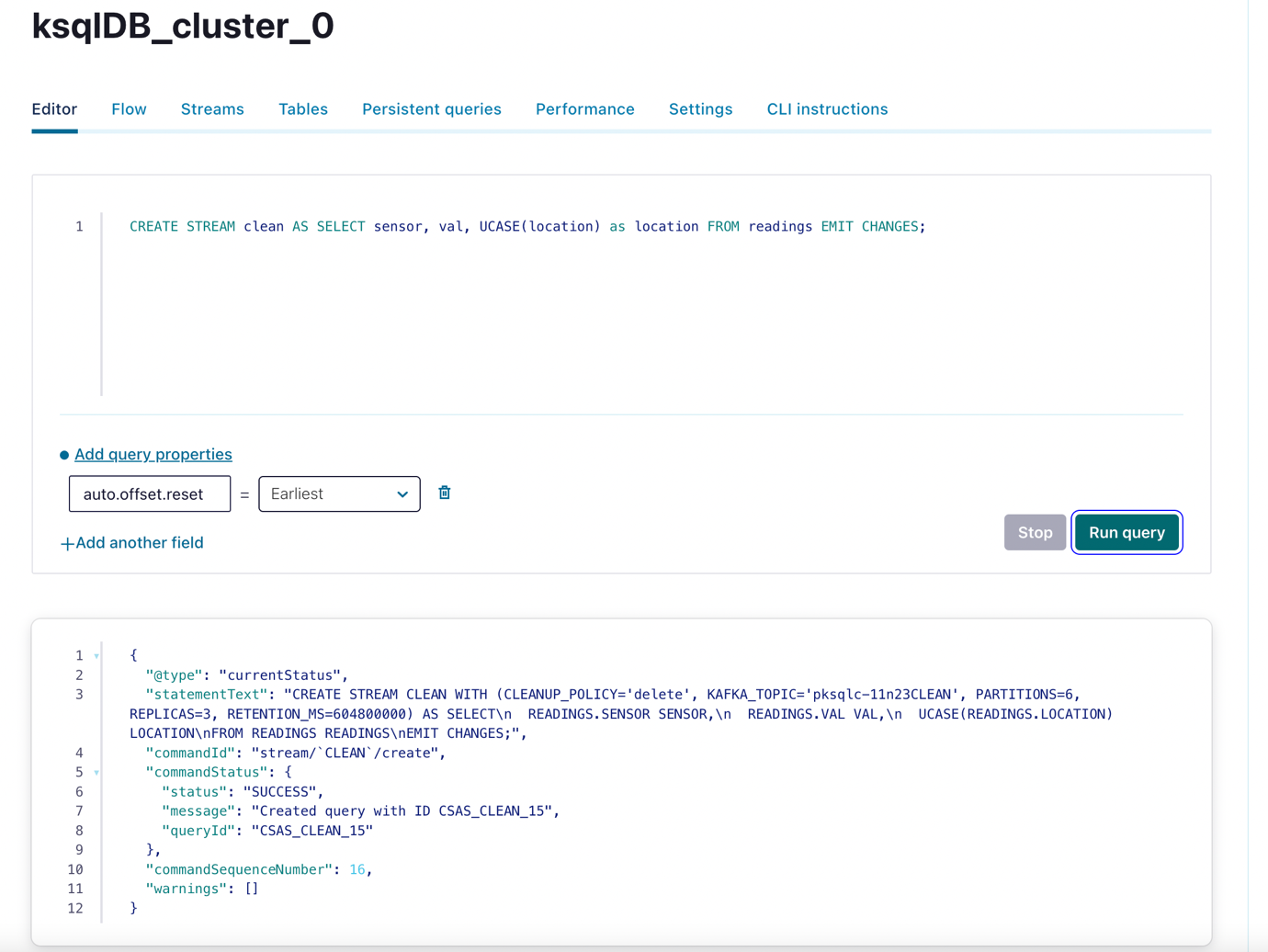
INSERT INTO readings (sensor, val, location) VALUES ('sensor-1', 42, 'wheel');

INSERT INTO readings (sensor, val, location) VALUES ('sensor-3', 42, 'muffler');

INSERT INTO readings (sensor, val, location) VALUES ('sensor-3', 48, 'muffler');

4. Run some queries

CREATE STREAM clean AS SELECT sensor, val, UCASE(location) as location FROM readings EMIT CHANGES;

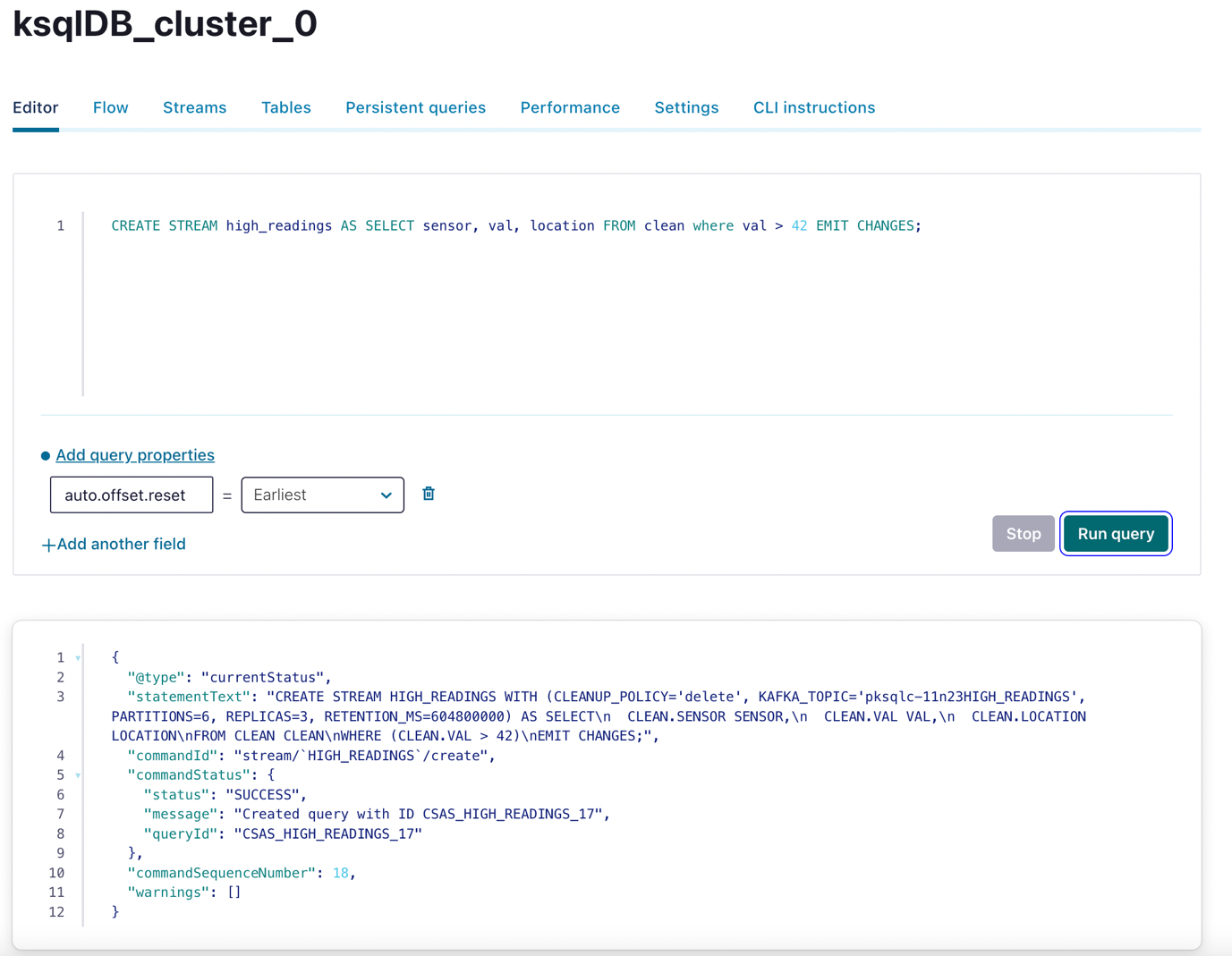


SELECT sensor, val, location FROM readings WHERE sensor='sensor-1' EMIT CHANGES;

A screenshot of a computer

Description automatically generated

CREATE STREAM high\_readings AS SELECT sensor, val, location FROM clean where val > 42 EMIT CHANGES;



5. Produce some events into readings topic

INSERT INTO readings (sensor, val, location) VALUES ('sensor-3', 36, 'motor');

A screenshot of a computer program

Description automatically generated

6. Run some queries

CREATE STREAM high\_pri AS SELECT sensor, val, UCASE(location) as location

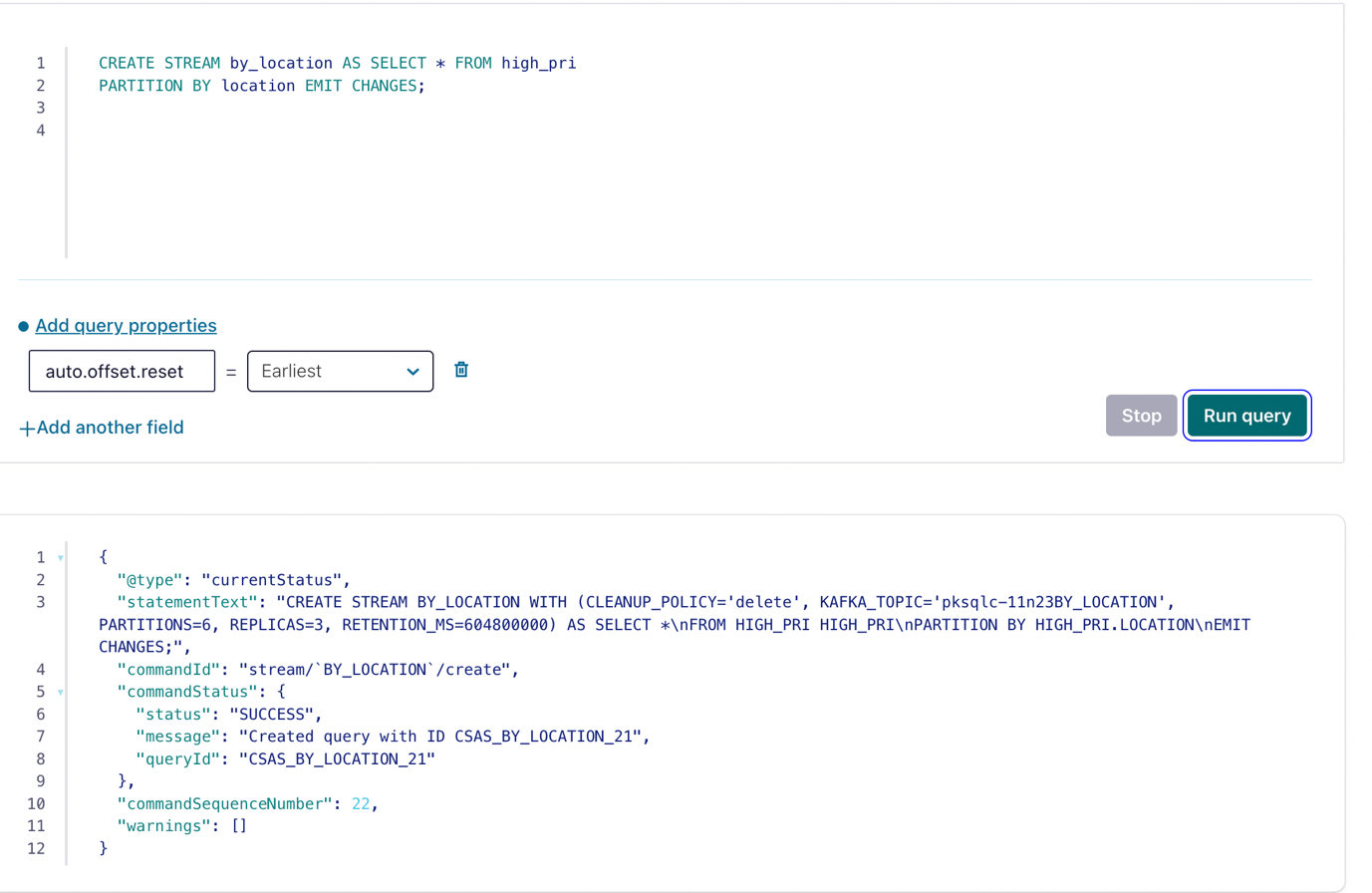
FROM readings where val > 42 EMIT CHANGES;

A screenshot of a computer program

Description automatically generated

CREATE STREAM by\_location AS SELECT \* FROM high\_pri

PARTITION BY location EMIT CHANGES;



CREATE STREAM by\_val AS SELECT \* FROM high\_pri

PARTITION BY val EMIT CHANGES;

A screenshot of a computer program

Description automatically generated

**Lab 5**

1. Considering the readings topic is already created

readings --> partitions 6

2. Produce some events into readings topic

INSERT INTO readings (sensor, val, location) VALUES ('sensor-1', 45, 'wheel');

INSERT INTO readings (sensor, val, location) VALUES ('sensor-2', 41, 'motor');

INSERT INTO readings (sensor, val, location) VALUES ('sensor-1', 42, 'wheel');

INSERT INTO readings (sensor, val, location) VALUES ('sensor-3', 42, 'muffler');

INSERT INTO readings (sensor, val, location) VALUES ('sensor-3', 48, 'muffler');

3. Create KTable:

CREATE TABLE avg\_readings AS

SELECT sensor, AVG(val) as avg

FROM readings

GROUP BY sensor

EMIT CHANGES;

A screenshot of a computer program

Description automatically generated

CREATE TABLE part\_avg AS

SELECT location, AVG(val) as avg

FROM readings

GROUP BY location

EMIT CHANGES;

