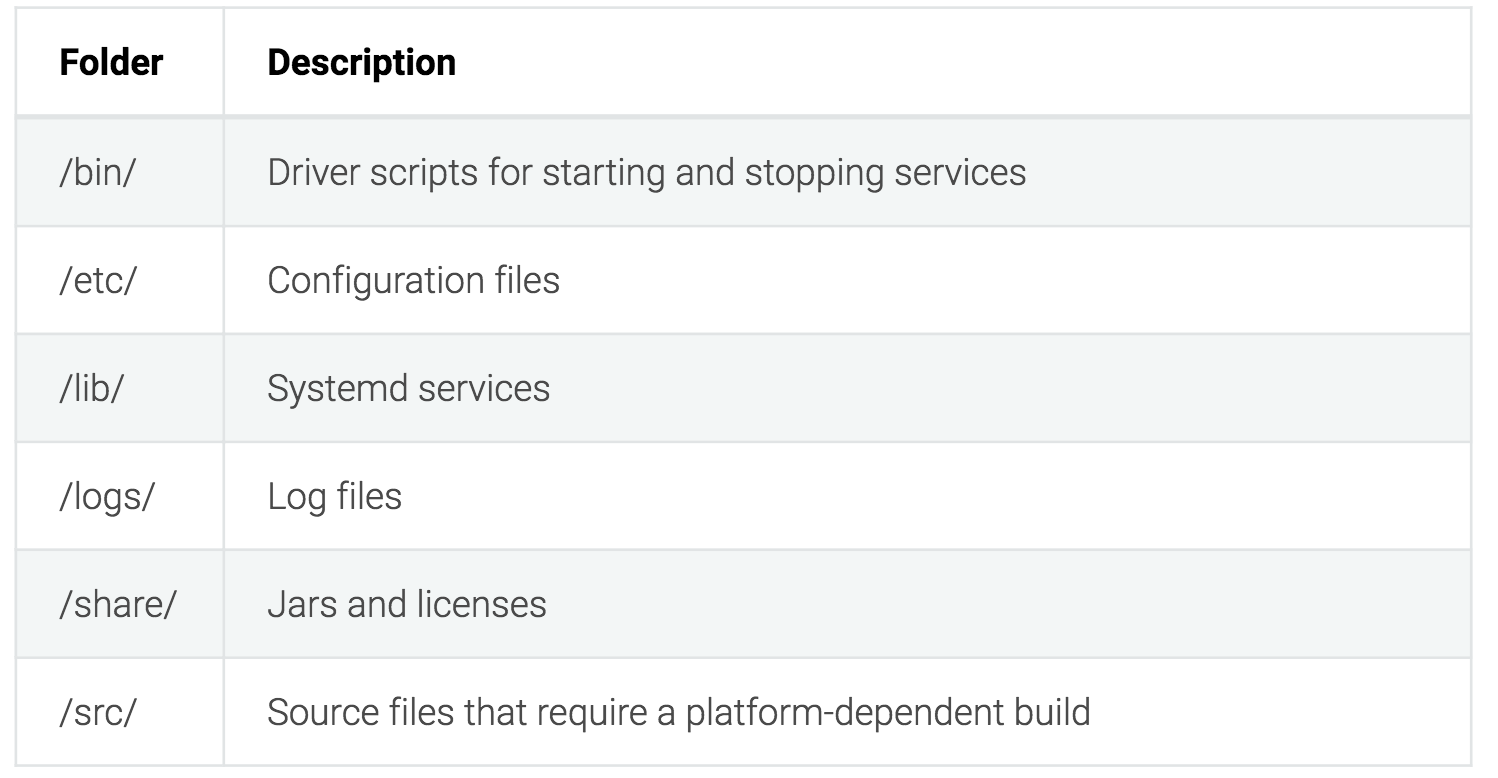
Exercice 1 :

Installation et lancement de kafka confluent 5.1.0

Solution :

Etape 1 :

Décompressez le fichier confluent-5.1.0-2.11.tar.gz, on y trouvera les dossiers suivant



Etape 2 :

Pour notre TP, nous allons installer le connecteur « source » **« kafka-connect-datagen »**

1. Allez dans le dossier de confluent après décompression
2. Lancez l’installation avec la commande suivante

<path-to-confluent>/bin/confluent-hub install --no-prompt confluentinc/kafka-connect-datagen:0.1.0

Etape 3 :

Lancer la plateforme confluent :

<path-to-confluent>/bin/confluent start

Vous devez avoir le resultat suivant :

Starting zookeeper

zookeeper **is** [UP]

Starting kafka

kafka **is** [UP]

Starting schema**-**registry

schema**-**registry **is** [UP]

Starting kafka**-**rest

kafka**-**rest **is** [UP]

Starting connect

connect **is** [UP]

Starting ksql**-**server

ksql**-**server **is** [UP]

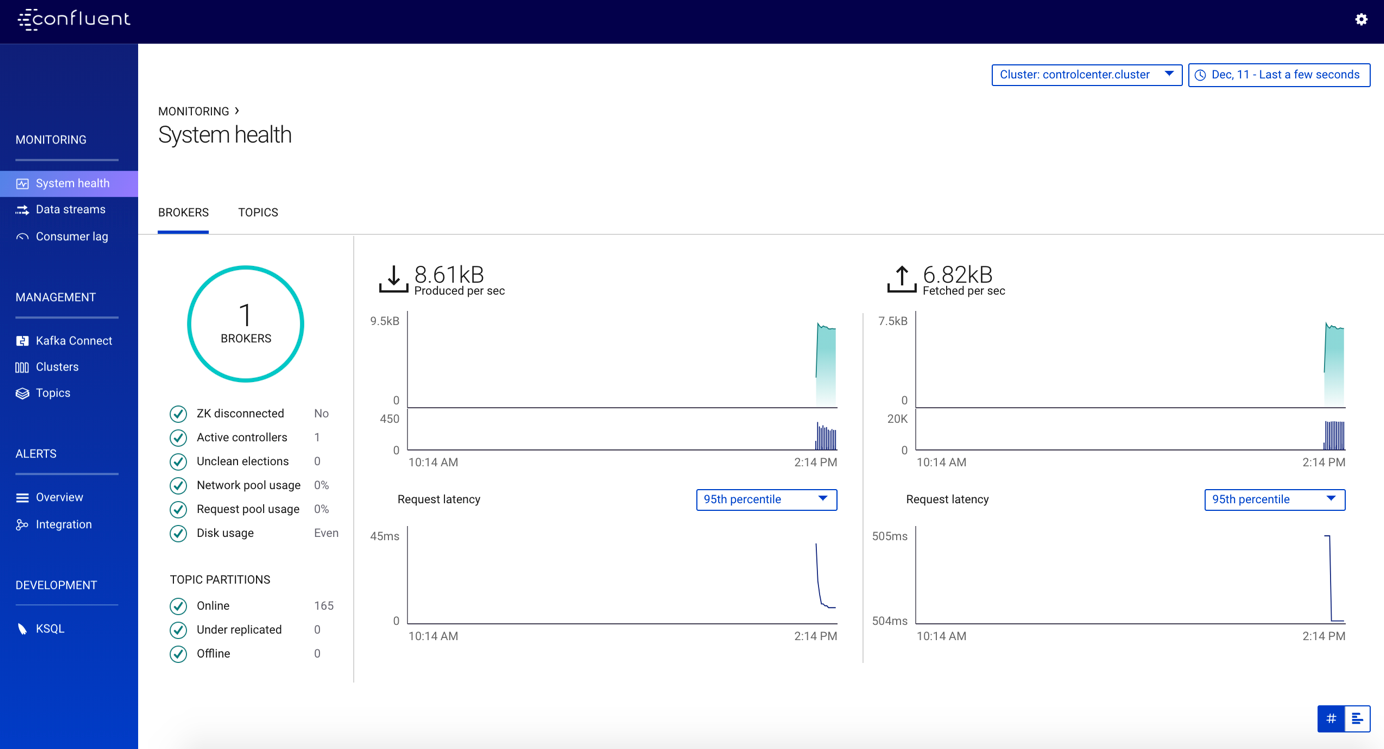
Starting control**-**center

control**-**center **is** [UP]

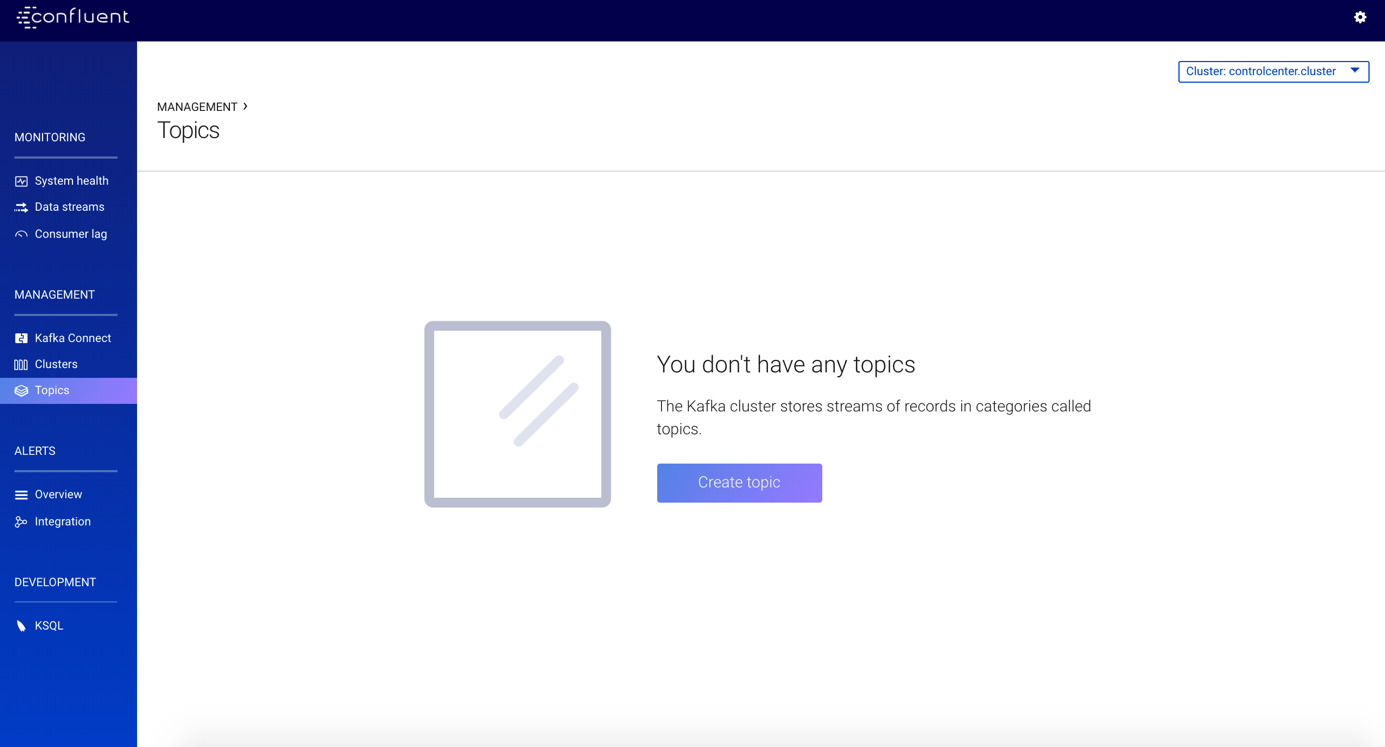
Exercice 2 :

La création de topics à partir de la plateforme confluent

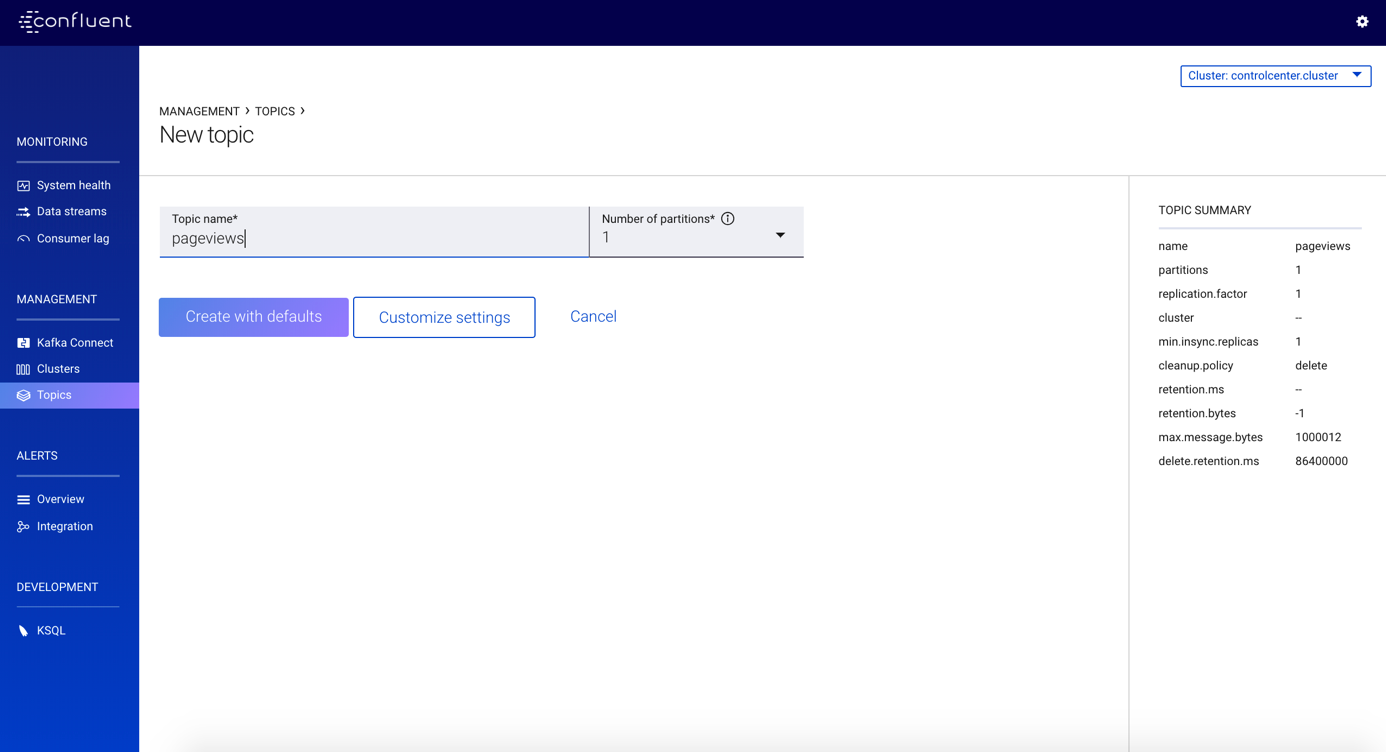
1. Connectez vous au Control Center web interface at <http://localhost:9021/>



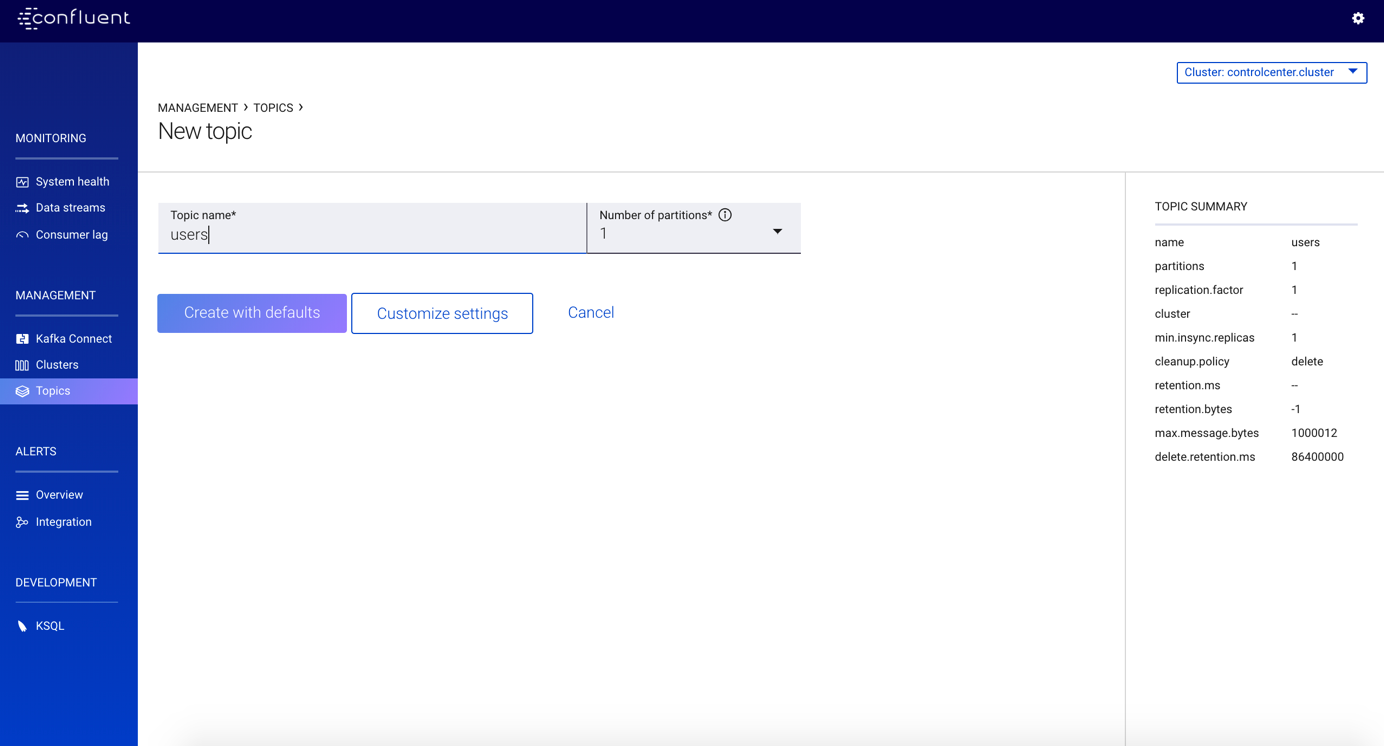
1. Sélectionnez **Management -> Topics** et cliquez sur **Create topic**.



1. Créez un topic et donnez-lui le namepageviews et cliquez sur **Create with defaults**.



1. Sélectionnez **Management -> Topics** et créez un topic users et après cliquez sur **Create with defaults**.



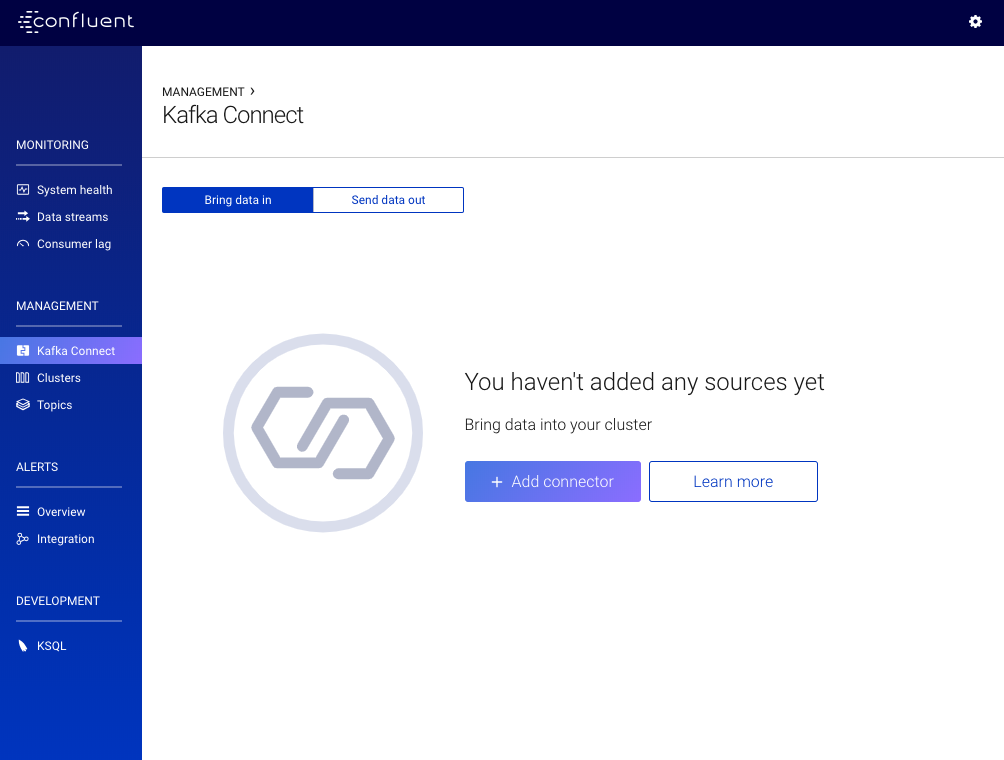
Exercice 3 :

Installer un connecteur kafka et générer un jeu de données, nous allons le connecteur code générateur qu’on a installé en début de l’exercice «kafka-connect-datagen». Ce connecteur permettra d’alimenter les données pour nos topics. « users » et « pageviews »

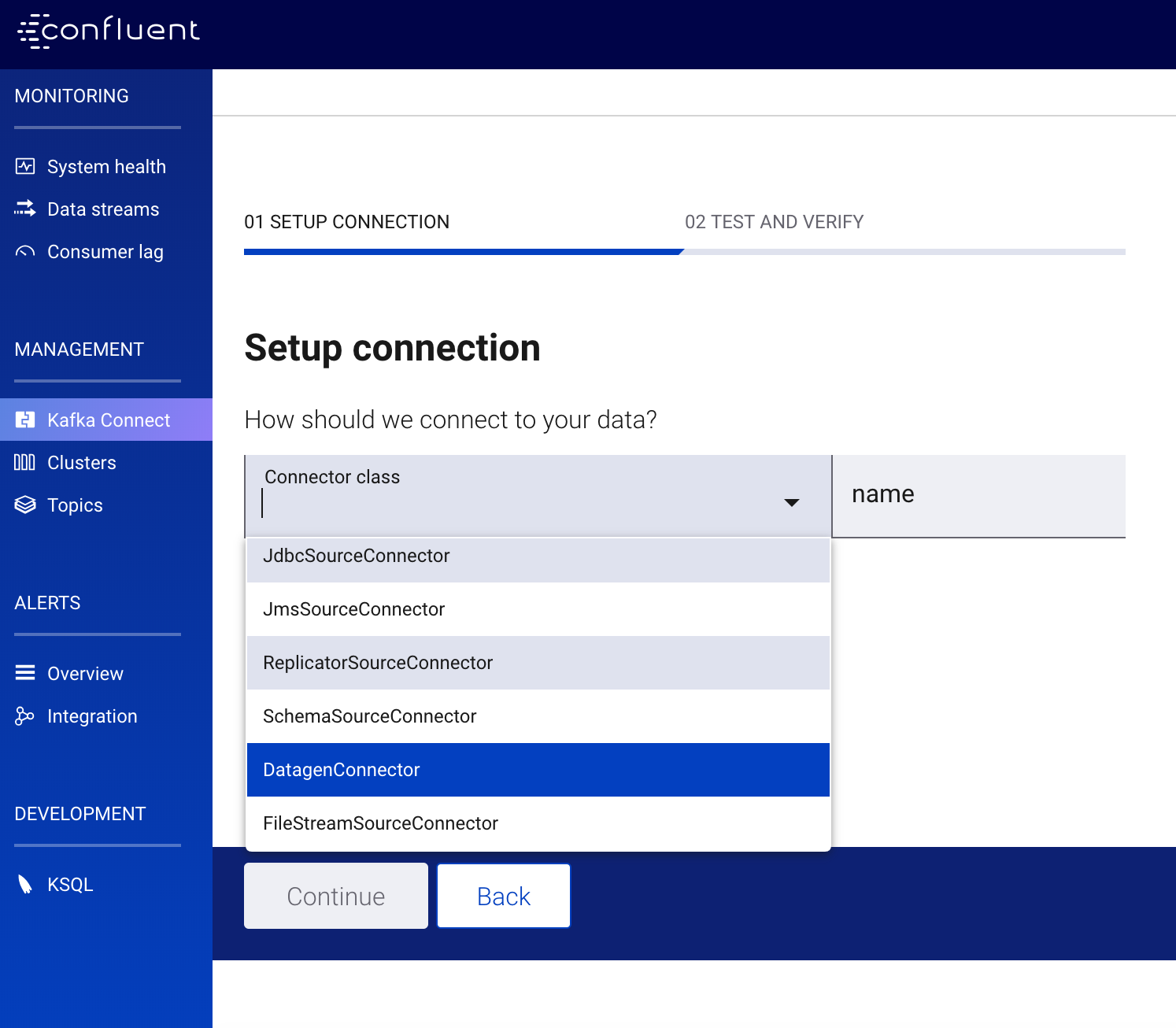
Etape 1 :

Lancer une instance du connecteur pour générer les données pour le topic « pageviews » en format « Avro »

1. From the Control Center interface, click **Management -> Kafka Connect**.

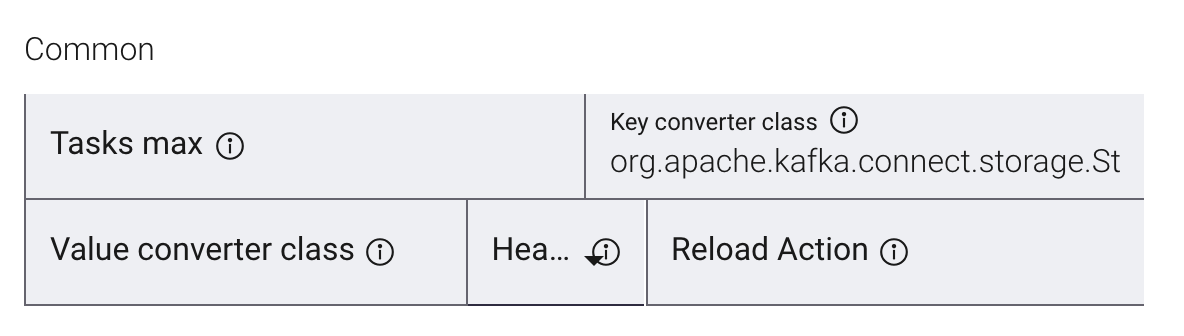


1. Click the **Add connector** button and scroll down to select **DatagenConnector**.

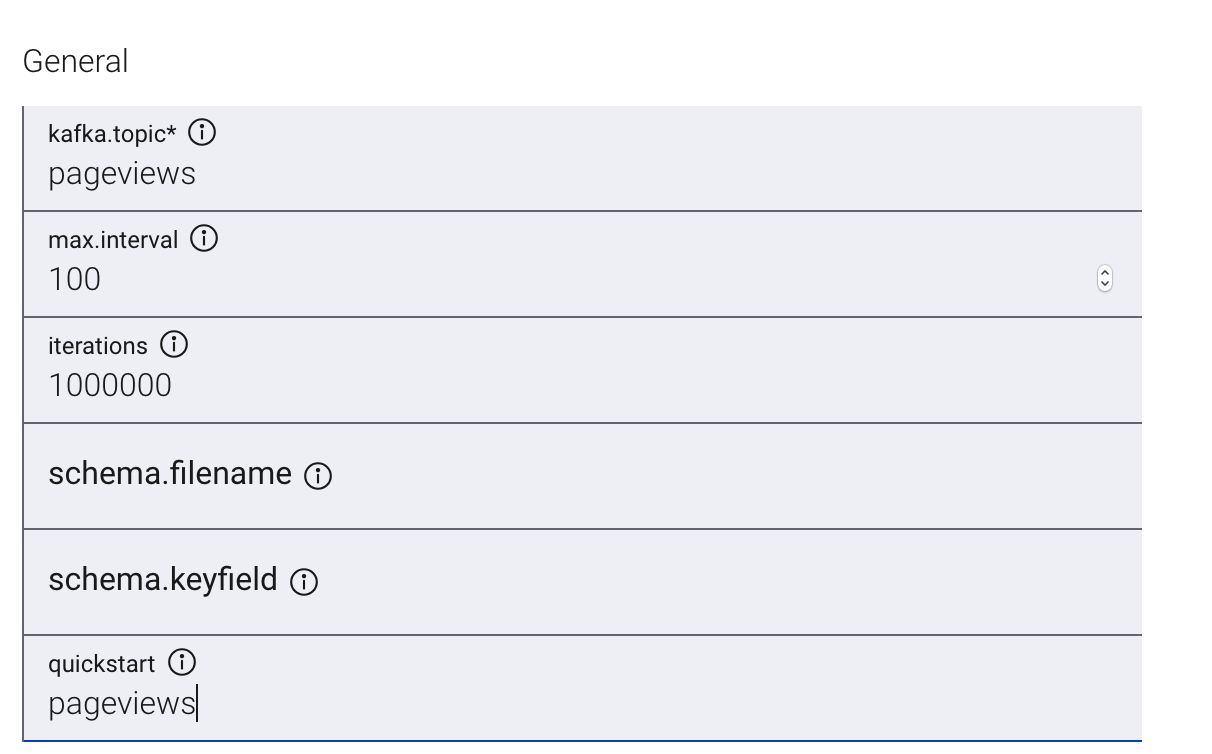


1. Name the connector datagen-pageviews. After you name the connector, new fields will appear. Scroll down and specify the following configurations.
   1. **Common**

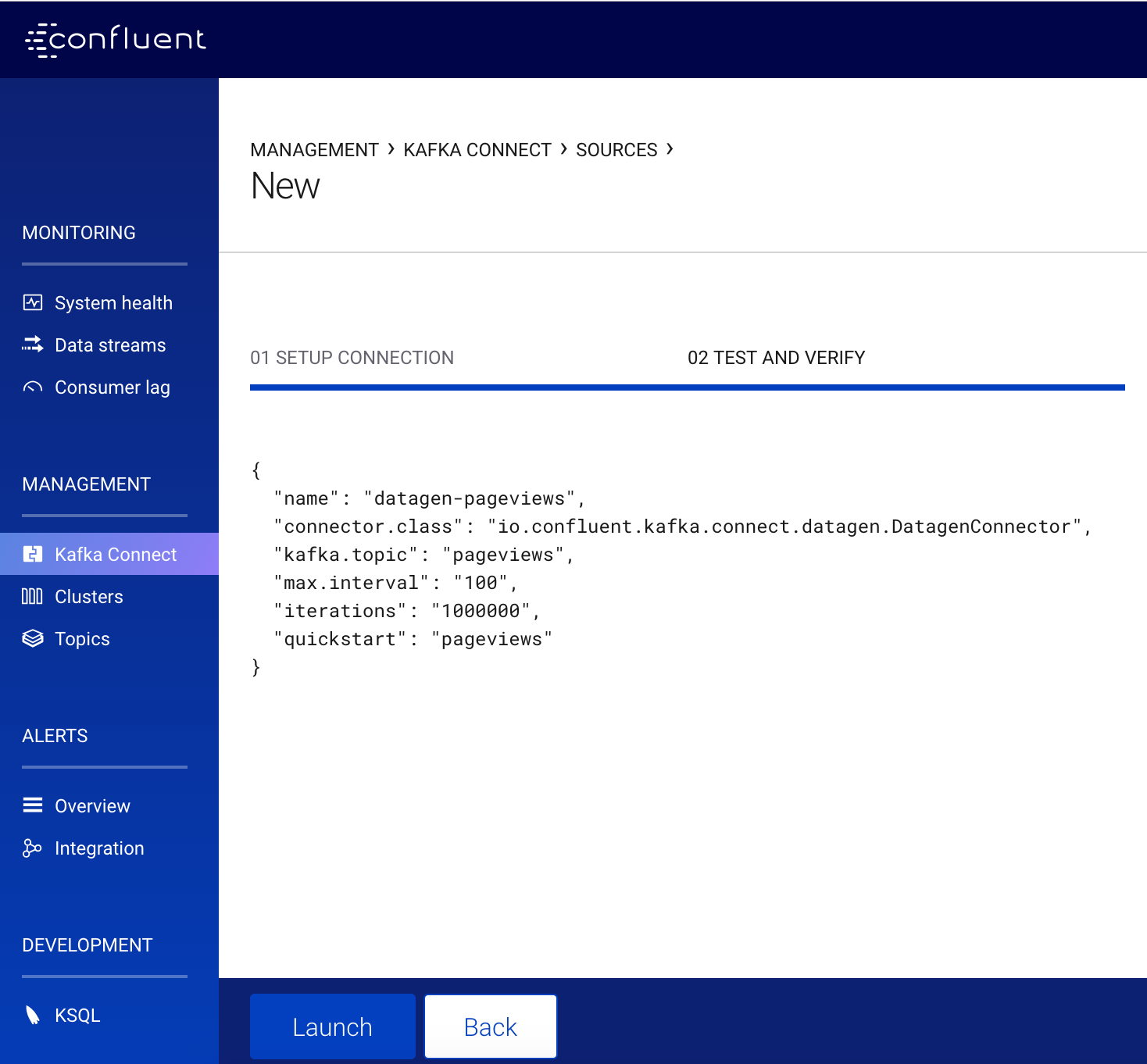
In the **Key converter class** field, type org.apache.kafka.connect.storage.StringConverter



* 1. **General**
     1. In the **kafka.topic** field, type pageviews.
     2. In the **max.interval** field, type 100.
     3. In the **iterations** field, type 1000000000.
     4. In the **quickstart** field, type pageviews.



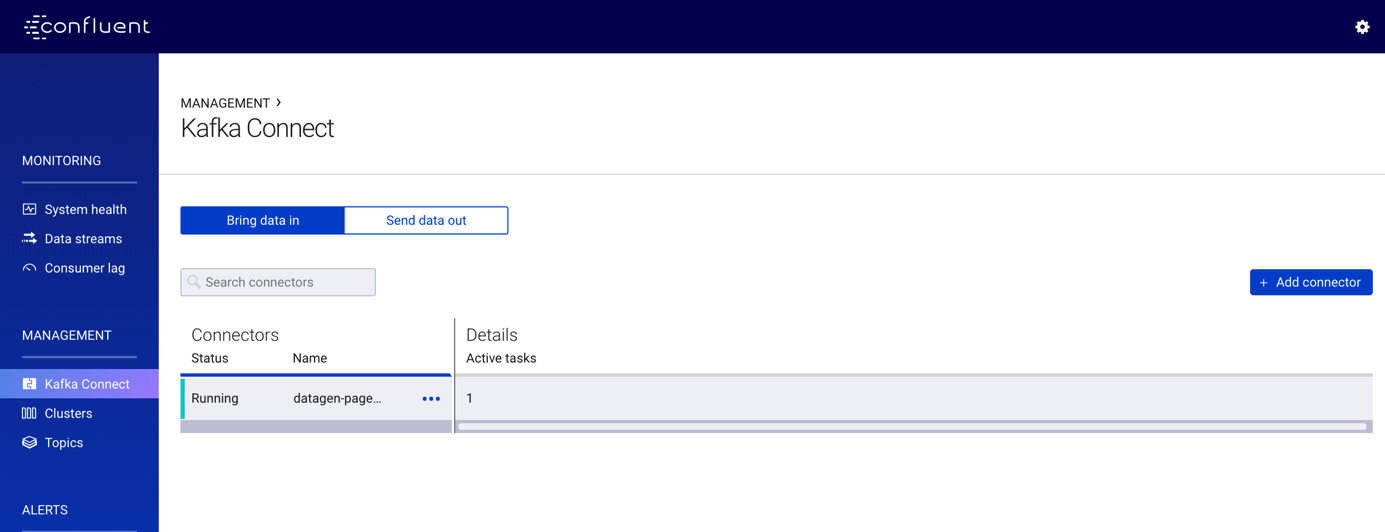
1. Click **Continue**.
2. Review the connector configuration and click **Launch**.



Etape 2 :

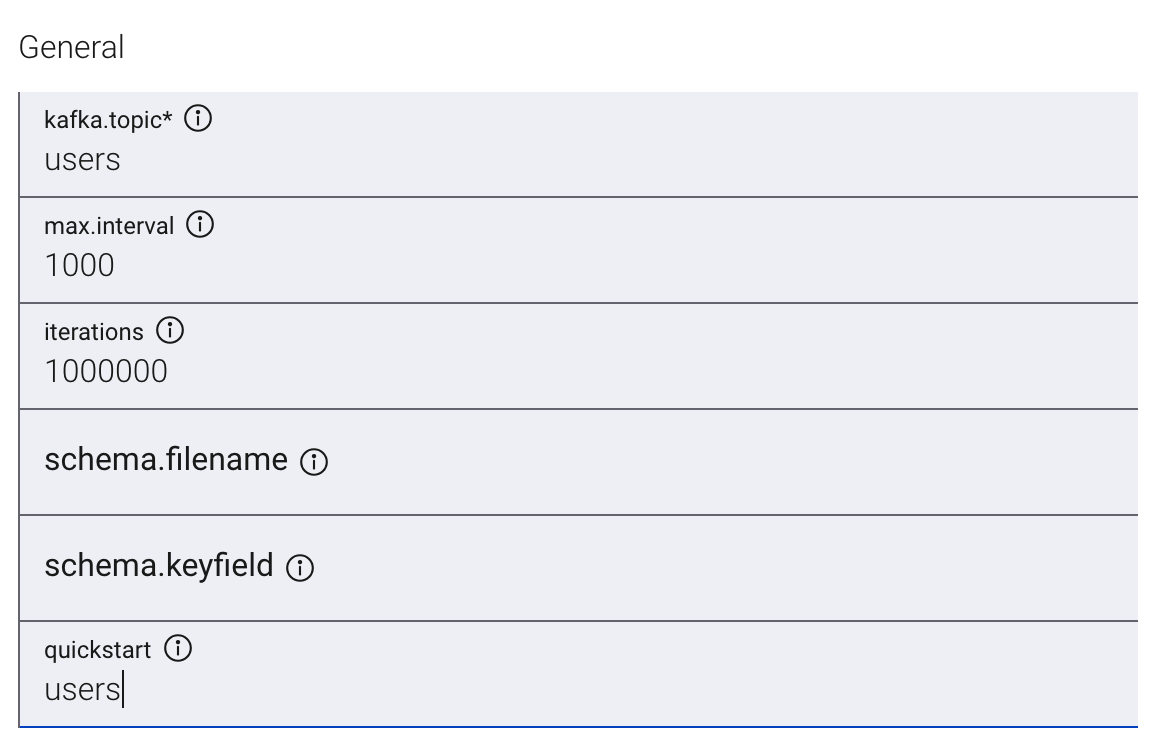
Lancer une autre instance du connecteur pour générer les données pour le topic « users » au format « Avro »

1. Click the **Add connector** button from the upper right-hand corner.

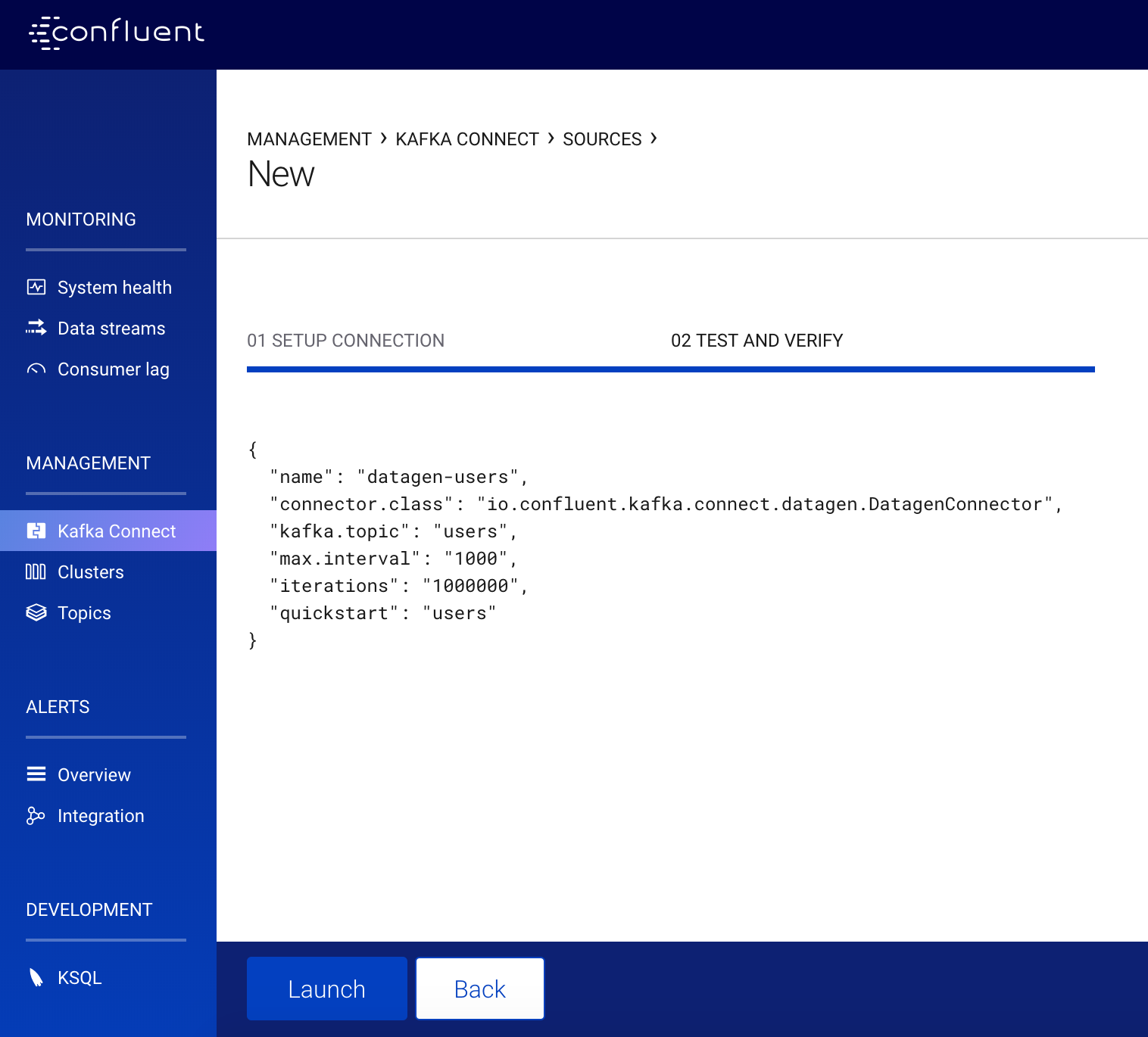


1. Click the **Add connector** button and scroll down to select **DatagenConnector**.
2. Name the connector datagen-users. After you name the connector, new fields will appear. Scroll down and specify the following configurations.

Redo the same configuration



1. Click **Continue**.
2. Review the connector configuration and click **Launch**.



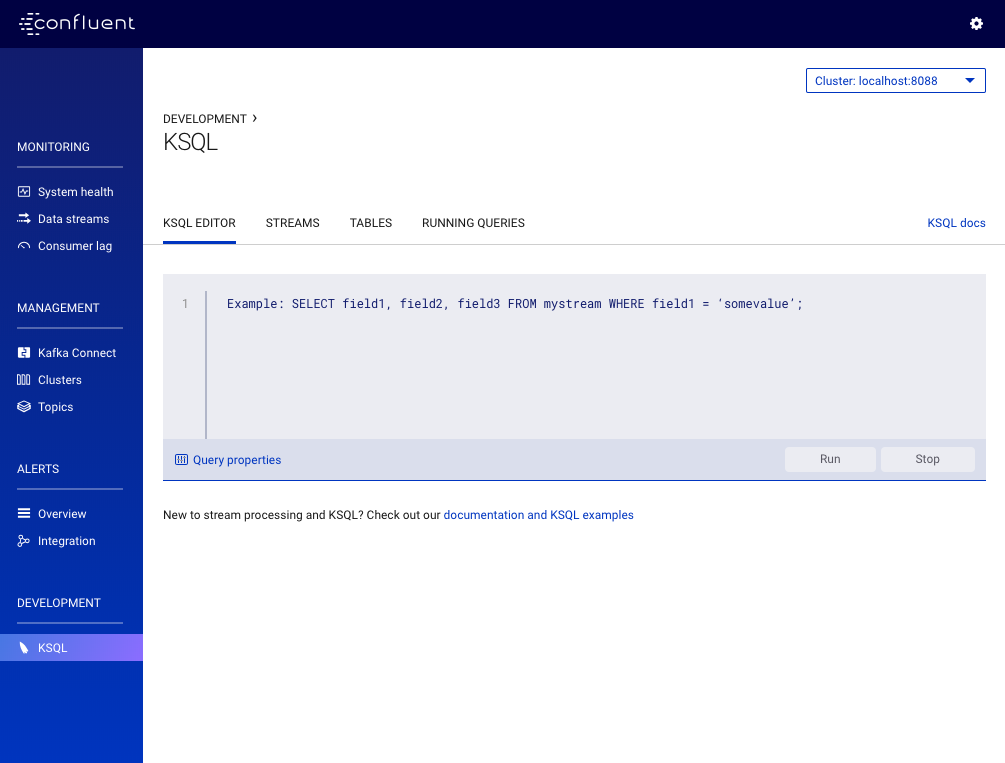
Exercice 4 :

Créer et écrire dans un stream et une table dans KSQL

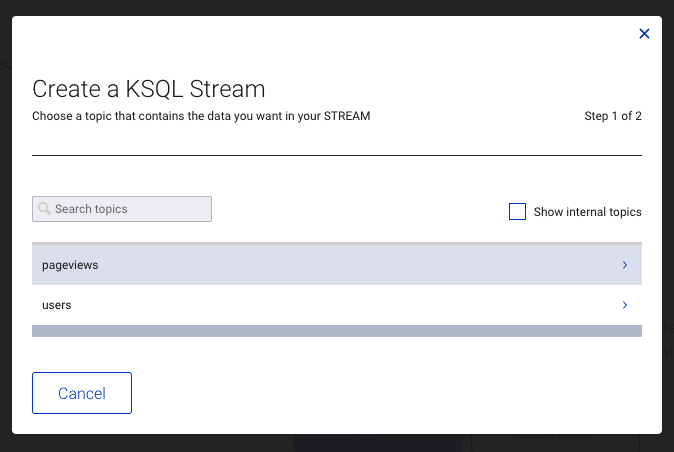
Etape 1 : Create Streams and Tables

In this step, KSQL is used to create streams and tables for the pageviews and users topics

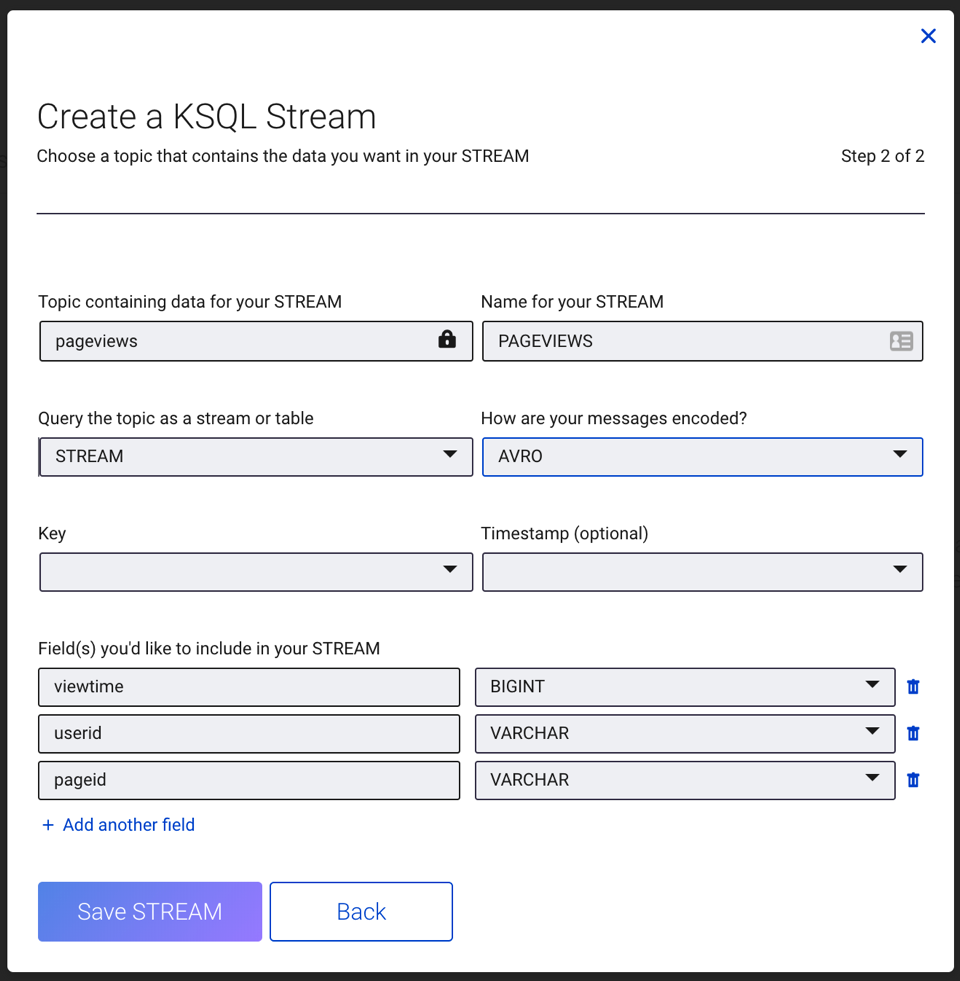
1. From the Control Center interface, click **Development -> KSQL**. By default your are on the **KSQL EDITOR** page



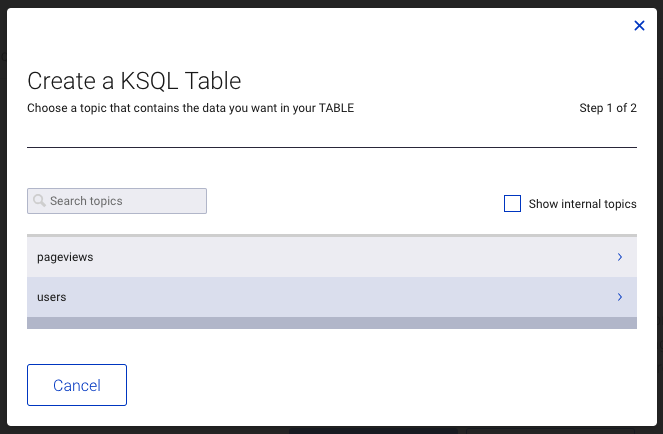
1. Click the **STREAMS** tab and **Add a stream** and select the pageviews topic



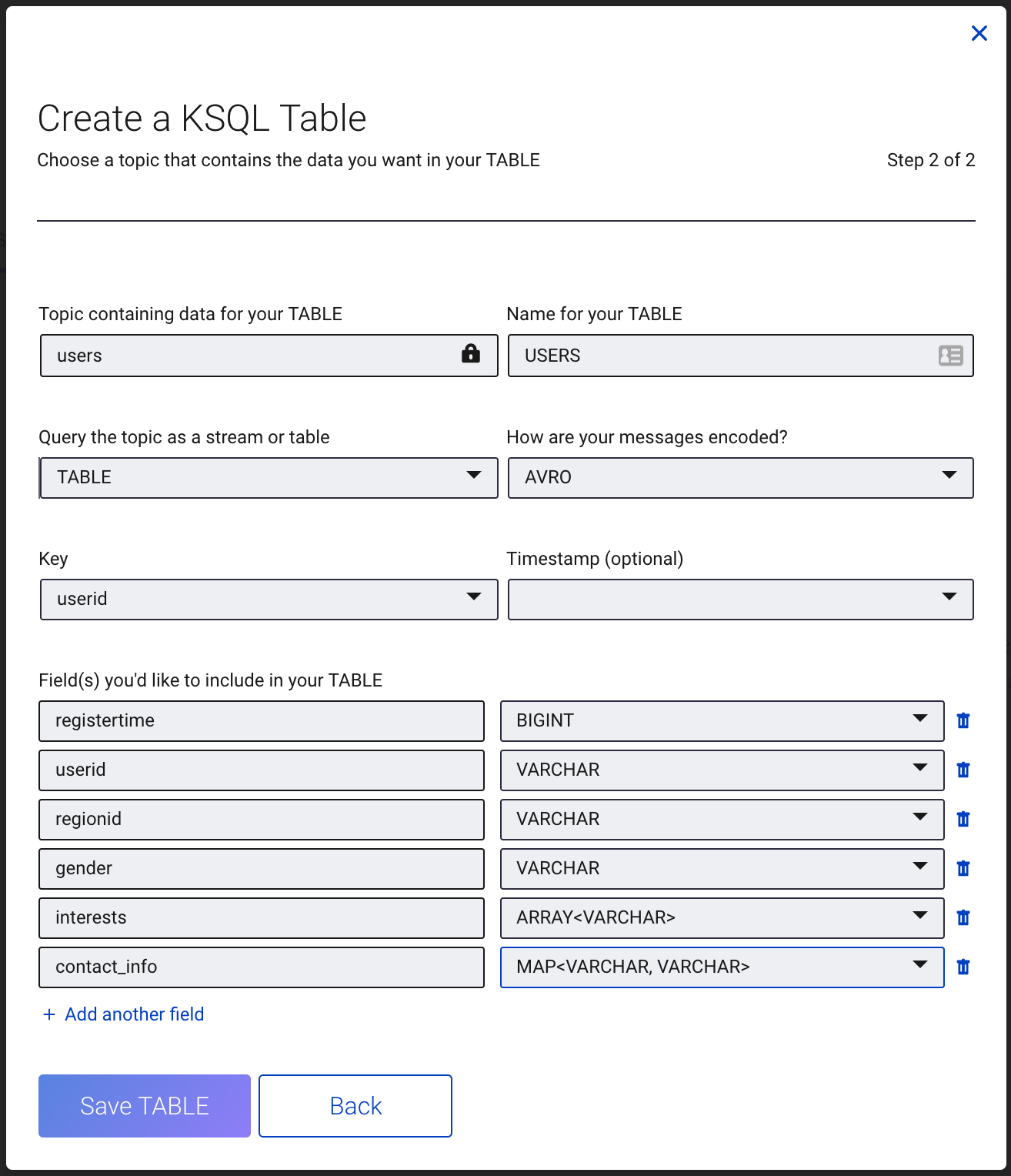
1. Choose your stream options and click **Save STREAM**
   1. In the **How are your messages encoded?** field, select AVRO.
   2. In the **Field(s) you’d like to include in your STREAM** field, add additional fields for:
      1. viewtime with type BIGINT
      2. userid with type VARCHAR
      3. pageid with type VARCHAR



1. Click the **TABLES** tab and **Add a table** and then select the users topic.

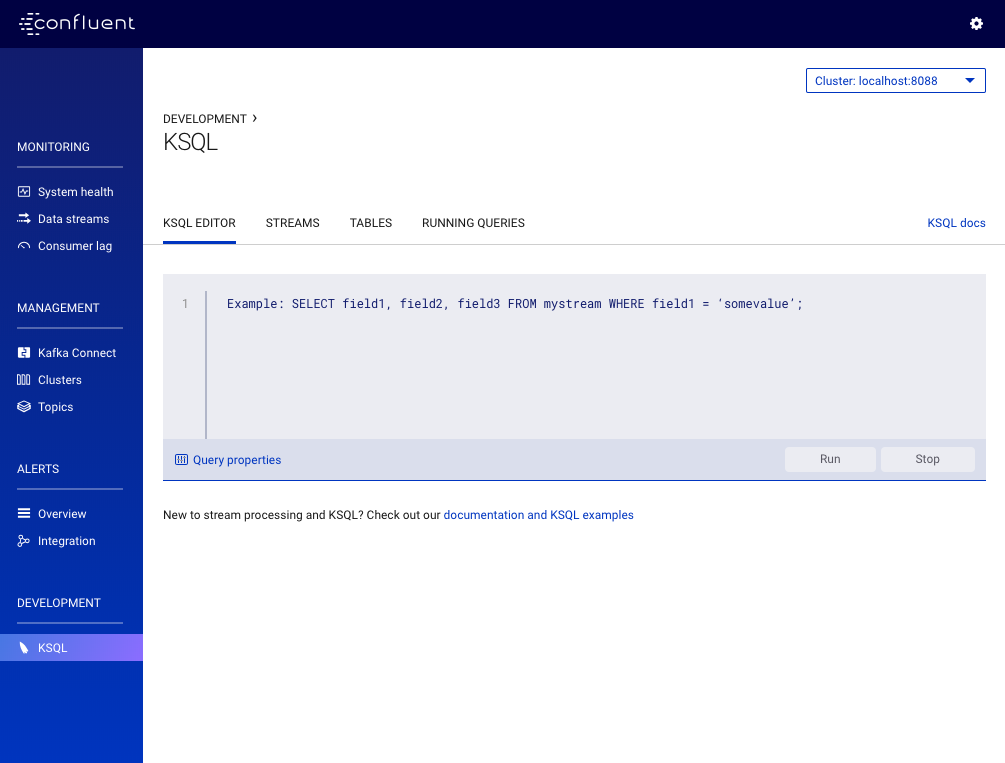


1. Choose your table options and click **Save TABLE**.
   1. In the **How are your messages encoded?** field, select AVRO.
   2. In the **Key** field, select userid.
   3. In the **Field(s) you’d like to include in your TABLE** field, add additional fields for:
      1. registertime with type BIGINT
      2. userid with type VARCHAR
      3. regionid with type VARCHAR
      4. gender with type VARCHAR
      5. interests with type ARRAY<VARCHAR>
      6. contact\_info with type MAP<VARCHAR, VARCHAR>



Etape 2 :Write Queries

1. From the Control Center interface, click **Development -> KSQL**. By default your are on the **KSQL EDITOR** page. Add the custom query property earliestfor the auto.offset.reset parameter. This instructs KSQL queries to read all available topic data from the beginning. This configuration is used for each subsequent query.



1. Run the following queries.
   1. Create a query that returns data from a stream with the results limited to three rows.

SELECT pageid FROM pageviews LIMIT 3;

* 1. Create a persistent query that filters for female users. The results from this query are written to the Kafka PAGEVIEWS\_FEMALE topic. This query enriches the pageviews STREAM by doing a LEFT JOIN with the users TABLE on the user ID, where a condition (gender = 'FEMALE') is met.

CREATE STREAM pageviews\_female AS SELECT users**.**userid AS userid, pageid, regionid, gender FROM pageviews LEFT JOIN users ON pageviews**.**userid **=** users**.**userid WHERE gender **=** 'FEMALE';

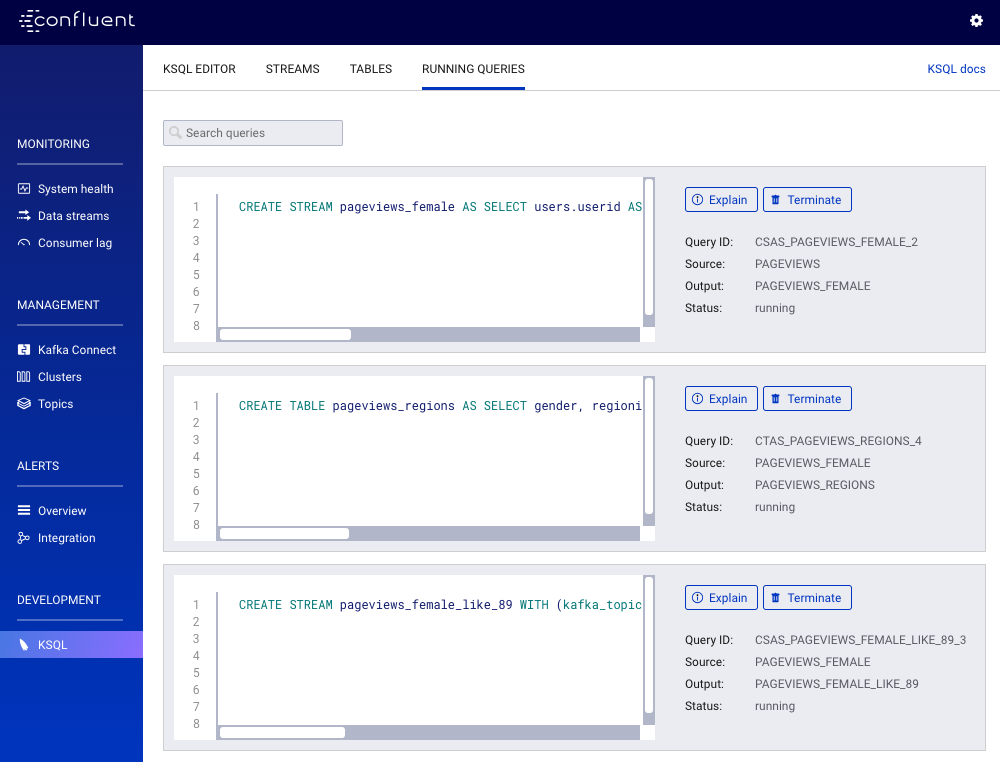
* 1. Create a persistent query where a condition (regionid) is met, using LIKE. Results from this query are written to a Kafka topic named pageviews\_enriched\_r8\_r9.

CREATE STREAM pageviews\_female\_like\_89 WITH (kafka\_topic**=**'pageviews\_enriched\_r8\_r9', value\_format**=**'AVRO') AS SELECT **\*** FROM pageviews\_female WHERE regionid LIKE '%\_8' OR regionid LIKE '%\_9';

* 1. Create a persistent query that counts the pageviews for each region and gender combination in a [tumbling window](https://docs.confluent.io/current/streams/developer-guide/dsl-api.html#windowing-tumbling) of 30 seconds when the count is greater than 1. Because the procedure is grouping and counting, the result is now a table, rather than a stream. Results from this query are written to a Kafka topic called PAGEVIEWS\_REGIONS.

CREATE TABLE pageviews\_regions AS SELECT gender, regionid , COUNT(**\***) AS numusers FROM pageviews\_female WINDOW TUMBLING (size 30 second) GROUP BY gender, regionid HAVING COUNT(**\***) **>** 1;

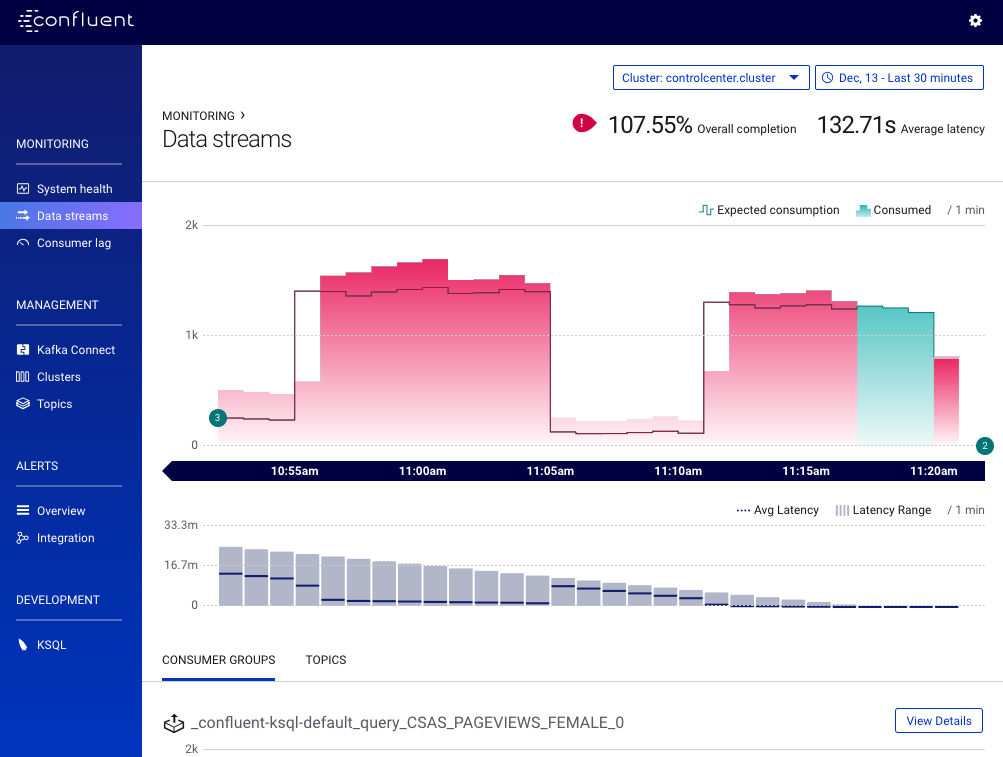
1. Click **RUNNING QUERIES** and you should see the following:



Exercice 5 :

Visualiser ton stream du Control center

1. View dashboard



1. View the consumers that have been created by KSQL

Click the **CONSUMER GROUPS** tab and then the **View Details** button for the \_confluent-ksql-default\_query\_CSAS\_PAGEVIEWS\_FEMALE\_0 consumer group.



This graph shows the messages being consumed by the stream query.

