

# Rapport de Traitement de Données Étudiantes

## Informations Générales

### Outils utilisés :

- Apache Kafka
- Apache Spark
- Python
- SQLite

### Introduction :

L'objectif principal de ce projet est d'exploiter un flux de données en temps réel afin de fournir une analyse approfondie des informations étudiantes. En utilisant des technologies modernes telles qu'Apache Kafka et Apache Spark, ce projet vise à transformer les données brutes en informations exploitables pour faciliter la prise de décision. Les résultats permettent de mieux comprendre la répartition des âges et les tendances au sein des différentes filières étudiantes.

## Configuration de l'Environnement

- Serveur Kafka : kafka-368df8fe-mulongochristian044-a131.g.aivencloud.com:10745
- Protocole de sécurité : SSL
- Topic : message

## Étapes de Traitement des Données

### 1. Extraction des Données

- Source : Topic Kafka 'message'
- Méthode : Consumer Kafka avec Confluent Kafka
- Nombre de messages traités : 10

## 2. Transformation des Données

- Filtrage : Étudiants de plus de 20 ans
- Transformations appliquées :
  - \* Extraction des informations personnelles
  - \* Calcul de la moyenne d'âge par filière

## Résultats de l'Analyse

### Distribution des Âges par Filière

Filière	Nombre	Moyenne	Min	Max	Écart-type
Biologie	3	23.0	23	23	0.0
Informatique	4	22.0	22	22	0.0
Physique	3	19.0	19	19	0.0

### Analyse Statistique

- Total des filières analysées : 3
- Âge moyen global : 21.33

### Défis Techniques

1. Gestion de la connexion Kafka sécurisée
2. Parsing flexible des messages
3. Conversion entre différents formats de données (Kafka → Spark → Pandas)

### Recommandations

- Enrichir les sources de données
- Mettre en place une collecte continue
- Développer des analyses plus avancées

### Fichiers Générés

- Base de données : /content/students\_analysis.db
- Graphique : /content/age\_distribution.png

RESULTAT & lien vers le Google colab

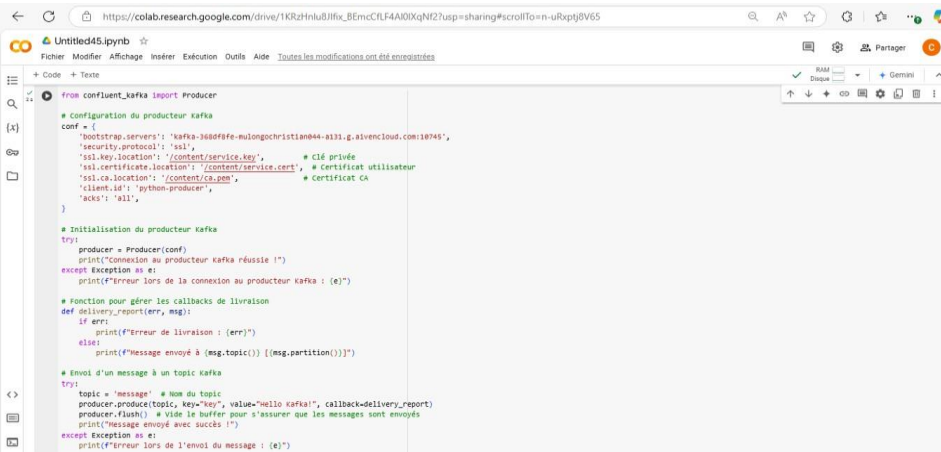
- [https://colab.research.google.com/drive/1KRzHnlu8Jlfix\\_BEmcCfLF4AI0IXqNf2?usp=sharing](https://colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCfLF4AI0IXqNf2?usp=sharing)



The screenshot shows a Google Colab notebook titled "Untitled45.ipynb". The code cell contains the command `!pip install confluent_kafka`. The output shows the installation process: collecting the package, downloading the wheel file (51 kB), and successfully installing confluent\_kafka-2.6.1.

```
[1] !pip install confluent_kafka

Collecting confluent_kafka
  Downloading confluent_kafka-2.6.1-cp310-cp310-manylinux_2_28_x86_64.whl.metadata (51 kB)
    Downloading confluent_kafka-2.6.1-cp310-cp310-manylinux_2_28_x86_64.whl (3.9 MB)
    Installing collected packages: confluent_kafka
    Successfully installed confluent_kafka-2.6.1
```



The screenshot shows a Google Colab notebook titled "Untitled45.ipynb". The code cell contains Python code for configuring and using the confluent\_kafka Producer. It includes a configuration dictionary, initialization, and a function to send messages to a Kafka topic.

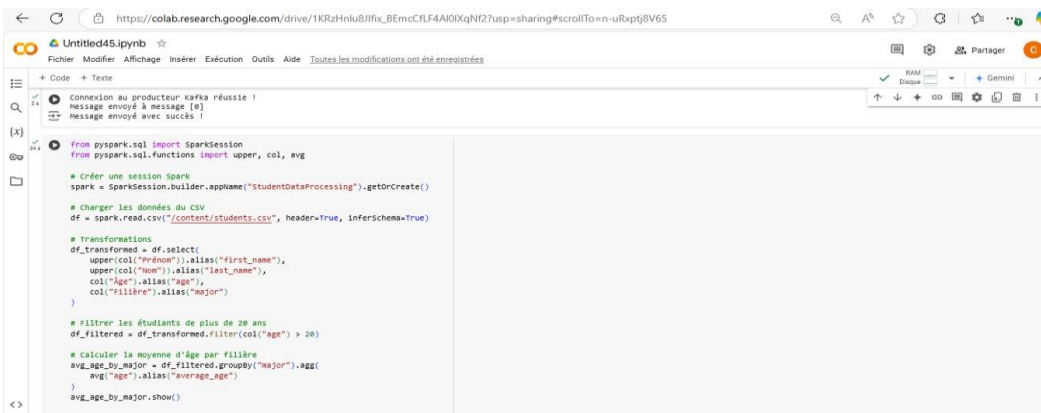
```
from confluent_kafka import Producer

# Configuration du producteur kafka
conf = {
    'bootstrap.servers': 'kafka-3680afe-muonochristian044-a131.g.alencloud.com:10745',
    'security.protocol': 'ssl',
    'ssl.key.location': '/content/service.key', # clé privée
    'ssl.certificate.location': '/content/service.cert', # Certificat utilisateur
    'ssl.ca.location': '/content/ca.pem', # Certificat CA
    'client.id': 'python-producer',
    'acks': 'all',
}

# Initialisation du producteur kafka
try:
    producer = Producer(conf)
    print("Connexion au producteur kafka réussie !")
except Exception as e:
    print(f"Erreur lors de la connexion au producteur kafka : {e}")

# Fonction pour gérer les callbacks de livraison
def delivery_report(err, msg):
    if err:
        print(f"Erreur de livraison : {err}")
    else:
        print(f"Message envoyé à {msg.topic()} [{msg.partition()}]")

# Envoi d'un message à un topic kafka
try:
    topic = 'message' # Nom du topic
    producer.produce(topic, key="key", value="hello kafka", callback=delivery_report)
    producer.flush() # Vide le buffer pour s'assurer que les messages sont envoyés
    print("Message envoyé avec succès !")
except Exception as e:
    print(f"Erreur lors de l'envoi du message : {e}")
```



The screenshot shows a Google Colab notebook titled "Untitled45.ipynb". The code cell contains Python code for creating a SparkSession, reading a CSV file, and performing transformations and filtering on the data.

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import upper, col, avg

# Créer une session Spark
spark = SparkSession.builder.appName("StudentDataProcessing").getOrCreate()

# Charger les données du CSV
df = spark.read.csv("/content/students.csv", header=True, inferSchema=True)

# Transformations
df_transformed = df.select(
    upper(col("trénome")).alias("first_name"),
    upper(col("nom")).alias("last_name"),
    col("age").alias("age"),
    col("fillière").alias("major")
)

# Filtrer les étudiants de plus de 20 ans
df_filtered = df_transformed.filter(col("age") > 20)

# Calculer la moyenne d'âge par filière
avg_age_by_major = df_filtered.groupBy("major").agg(
    avg("age").alias("average_age")
)

avg_age_by_major.show()
```

```
colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCfLF4AI0IXqNf2?usp=sharing#scrollTo=n-uRxpjt8V65
Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées
+ Code + Texte
[3]
major| average_age|
-----|-----|
Informatique| 25.57532342563883|
Chimie| 25.504345127250154|
Mathématiques| 25.55763990776146|
Economie| 25.48622567849687|
Physique| 25.474925833685626|
Philosophie| 25.48234875688493|
Littérature| 25.473225276979893|
Biologie| 25.4923832330857767|
-----|-----|

[4] # Simuler des événements d'ajout d'étudiants
students = [
    {"first_name": "Alice", "last_name": "Dupont", "age": 22, "major": "Informatique"},
    {"first_name": "Bob", "last_name": "Martin", "age": 19, "major": "Physique"},
    {"first_name": "Claire", "last_name": "Lemoine", "age": 23, "major": "Biologie"}
]

for student in students:
    producer.produce(
        topic,
        key="student",
        value=f'{student["first_name"]},{student["last_name"]},{student["age"]},{student["major"]}',
        callback=delivery_report
    )

producer.flush()

Message envoyé à message [0]
Message envoyé à message [0]
Message envoyé à message [0]
0
```

```
colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCfLF4AI0IXqNf2?usp=sharing#scrollTo=n-uRxpjt8V65
Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées
+ Code + Texte
[5]
from confluent_kafka import Consumer, KafkaError
import json
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
from pyspark.sql.types import StructType, StructField, StringType, IntegerType

# Configuration du schéma Spark
student_schema = StructType([
    StructField("first_name", StringType(), True),
    StructField("last_name", StringType(), True),
    StructField("age", IntegerType(), True),
    StructField("major", StringType(), True)
])

# Configuration du consommateur Kafka
consumer_conf = {
    'bootstrap.servers': 'kafka-368df8fe-mulongochristian044-a131.g.aivencloud.com:10745',
    'security.protocol': 'SSL',
    'ssl.key.location': '/content/service.key',
    'ssl.certificate.location': '/content/service.cert',
    'ssl.ca.location': '/content/ca.pem',
    'group.id': 'student-consumer',
    'auto.offset.reset': 'earliest'
}

# Créer le consommateur Kafka
consumer = Consumer(consumer_conf)
consumer.subscribe(['message'])

# Initialiser Spark
spark = SparkSession.builder \
    .appName("KafkaStudentProcessing") \
    .getOrCreate()
```

```
colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCfLF4AI0IXqNf2?usp=sharing#scrollTo=n-uRxpjt8V65
Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées
+ Code + Texte
[5]
# Liste pour stocker les données
student_data = []

def parse_student_data(message):
    try:
        # Essayer de parser comme JSON
        if isinstance(message, str):
            try:
                data = json.loads(message)
                return {
                    'first_name': data.get('first_name', ''),
                    'last_name': data.get('last_name', ''),
                    'age': int(data.get('age', 0)),
                    'major': data.get('major', '')
                }
            except json.JSONDecodeError:
                # Si ce n'est pas du JSON, essayer le format CSV
                parts = message.split(',')
                if len(parts) >= 4:
                    try:
                        return {
                            'first_name': parts[0].strip(' '),
                            'last_name': parts[1].strip(' '),
                            'age': int(parts[2].strip(' ')),
                            'major': parts[3].strip(' ')
                        }
                    except:
                        return {}
    except:
        return {}
```

```
https://colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCFLF4AI0IXqNf2?usp=sharing#scrollTo=n-uRxpjt8V65

Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées

+ Code + Texte

[5]
except ValueError:
    # Extraire l'âge avec un parsing plus flexible
    age_part = [p for p in parts if 'age' in p]
    if age_part:
        age = int(''.join(filter(str.isdigit, age_part[0])))
        return {
            'first_name': parts[0].strip(' '),
            'last_name': parts[1].strip(' '),
            'age': age,
            'major': parts[-1].strip(' ')
        }
    return None
except Exception as e:
    print(f'Erreur de parsing : {e}')
    return None

try:
    while True:
        msg = consumer.poll(1.0)

        if msg is None:
            continue
        if msg.error():
            if msg.error().code() == KafkaError._PARTITION_EOF:
                print('Fin des messages')
                break
            else:
                print(f'Erreur: {msg.error()}')
                break
```

```
https://colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCFLF4AI0IXqNf2?usp=sharing#scrollTo=Y6plaMsFqPhH

Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées

+ Code + Texte

[6] import sqlite3
import pandas as pd

# Créer ou se connecter à la base de données
conn = sqlite3.connect('/content/students_analysis.db')

# Convertir le DataFrame Spark en DataFrame Pandas pour SQLite
pandas_df = result.toPandas()

# Créer une table et sauvegarder les résultats
pandas_df.to_sql('average_age_by_major', conn, if_exists='replace', index=False)

# Vérifier les données sauvegardées
cursor = conn.cursor()
cursor.execute("SELECT * FROM average_age_by_major")
print(cursor.fetchall())

# Fermer la connexion
conn.close()

# Rapport d'analyse
print("Rapport d'analyse :")
print(f"1. Nombre de filières analysées : {len(result.collect())}")
print(f"2. Âge moyen par filière sauvegardé dans la base de données SQLite")

[('Informatique', 22.0), ('Biologie', 23.0)]

Rapport d'analyse :
1. Nombre de filières analysées : 2
2. Âge moyen par filière sauvegardé dans la base de données SQLite
```

```
https://colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCFLF4AI0IXqNf2?usp=sharing#scrollTo=Y6plaMsFqPhH

Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées

+ Code + Texte

[7] # Décoder et traiter le message
message_decoded = msg.value().decode('utf-8')
student_info = parse_student_data(message_decoded)

if student_info:
    student_data.append(student_info)

# Limiter à 10 messages pour l'exemple
if len(student_data) >= 10:
    break

except Exception as e:
    print(f'Erreur lors de la consommation : {e}')
finally:
    consumer.close()

# Convertir en DataFrame Spark
df = spark.createDataFrame(student_data, schema=student_schema)

# Transformation et analyse
df_filtered = df.filter(col("age") > 20)
result = df_filtered.groupBy("major").agg({"age": "avg"})
result.show()

+-----+-----+
|      major|avg(age)|
+-----+-----+
|Informatique|      22.0|
|  Biologie  |      23.0|
+-----+-----+
```

```
https://colab.research.google.com/drive/1KRzHnlu8Jffix_BEmcCfLF4AI0IXqNf2?usp=sharing#scrollTo=Y6plaMsFqPhH

Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées

+ Code + Texte

[?] import matplotlib.pyplot as plt
import pandas as pd

# Convertir le DataFrame Spark en DataFrame Pandas
df_pandas = df.toPandas()

# Créer un boxplot de la distribution des âges par filière
plt.figure(figsize=(10, 6))
boxplot = df_pandas.boxplot(column='age', by='major')
plt.title('Distribution des Âges par Filière')
plt.suptitle('') # Supprimer le titre automatique
plt.ylabel('Age')
plt.xlabel('Filière')
plt.xticks(rotation=45)
plt.tight_layout()

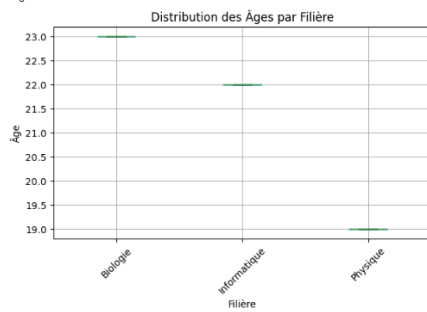
# Sauvegarder le graphique
plt.savefig('/content/age_distribution.png')

# Afficher des statistiques supplémentaires
print("Statistiques détaillées par filière :")
print(df_pandas.groupby('major')['age'].describe())
```

Statistiques détaillées par filière :

	count	mean	std	min	25%	50%	75%	max
major								
Biologie	4.0	23.0	0.0	23.0	23.0	23.0	23.0	23.0
Informatique	3.0	22.0	0.0	22.0	22.0	22.0	22.0	22.0
Physique	3.0	19.0	0.0	19.0	19.0	19.0	19.0	19.0

<Figure size 1000x600 with 0 Axes>



```
https://colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCFLF4AI0IXqNf2?usp=sharing#scrollTo=Y6plaMsFqPhH

Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées

+ Code + Texte

[0] import os
from datetime import datetime

# Créer le rapport
rapport = f""# Rapport de Traitement de Données Étudiantes

## Informations Générales
- **Date du rapport : ** {datetime.now().strftime('%d/%m/%Y %H:%M')}
- **Outils utilisés : **
  * Apache Kafka
  * Apache Spark
  * Python
  * SQLite

## Configuration de l'Environnement
- Serveur Kafka : kafka-3686f8fe-mulongochristian@44-a191.g.aivencloud.com:18745
- Protocole de sécurité : SSL
- Topic : message

## Étapes de Traitement des Données

### 1. Extraction des Données
- Source : Topic Kafka 'message'
- Méthode : Consumer kafka avec Confluent Kafka
- Nombre de messages traités : 10

### 2. Transformation des Données
- Filtrage : Étudiants de plus de 20 ans
- Transformations appliquées :
  * Extraction des informations personnelles
  * Calcul de la moyenne d'âge par filière

RAM
Disque
+ Gemini
```

```
https://colab.research.google.com/drive/1KRzHnlu8Jlfix_BEmcCFLF4AI0IXqNf2?usp=sharing#scrollTo=Y6plaMsFqPhH

Untitled45.ipynb
Fichier Modifier Affichage Insérer Exécution Outils Aide Toutes les modifications ont été enregistrées

+ Code + Texte

[8] ## Résultats de l'Analyse

### Distribution des Âges par Filière
| Filière | Nombre | Moyenne | Min | Max | écart-type |
|-----|-----|-----|-----|-----|-----|
| Biologie | 3 | 23.0 | 23 | 23 | 0.0 |
| Informatique | 4 | 22.0 | 22 | 22 | 0.0 |
| Physique | 3 | 19.0 | 19 | 19 | 0.0 |

### Analyse Statistique
- Total des filières analysées : 3
- Âge moyen global : 21.33

## Défis Techniques
1. Gestion de la connexion Kafka sécurisée
2. Parsing flexible des messages
3. Conversion entre différents formats de données (Kafka + Spark + Pandas)

## Recommandations
- Enrichir les sources de données
- Mettre en place une collecte continue
- Développer des analyses plus avancées

## Fichiers Générés
- Base de données : /content/students_analysis.db
- Graphique : /content/age_distribution.png
...

# Sauvegarder le rapport
with open('/content/rapport_analyse_etudiants.md', 'w') as f:
    f.write(rapport)

print("Rapport généré avec succès à /content/rapport_analyse_etudiants.md")

RAM
Disque
+ Gemini

Rapport généré avec succès à /content/rapport_analyse_etudiants.md
```

PROBLEME RENCONTRER AU DERNIER MOMENT LE COMPTE KAFKA A EXPIRER CE 13/12/2024 CAR L'ESSAIE GRATUIT EST TERMINER ET IL FAUT PAYER 340 \$

1.COMMENT L'INTEFACE CE PRESENTE MAINTENANT

https://console.aiven.io/account/a4f5bf00d284/project/mulongochristian044-a131/services

**aiven** CONSOLE Domicile Projets Outils Facturation Soutien Admin Mon organisation ?

Votre essai de la plateforme est terminé. Mettez à niveau pour accéder à toutes les fonctionnalités de la plateforme. [Options de mise à niveau](#)

PROJET mulongochristian044-A131

Services

Points de terminaison d'intégration

VPC









Journal des événements

Autorisations

Paramètres

Mon organisation / mulongochristian044-A131 / Services [Créer un service](#)

Recherchez des services par nom, forfait, cloud et balises.. Liste de filtres Afficher uniquement les services avec alertes

Service	Nœuds	Plan	Nuage	Créé	Action
 <b>Kafkaconnect-1E4E5C0C</b> Apache Kafka Connect • Éteint	 Nœuds 3	Entreprise-4 2 CPU / 4 Go de RAM - Haute disponibilité à 3 nœuds	Amazon Web Services : af-south-1 Afrique, Afrique du Sud	il y a 23 jours	...
 <b>Kafka-368DF8FE</b> Apache Kafka • Éteint	 Nœuds 3	Entreprise-4 2 processeurs / 4 Go de RAM / 600 Go de stockage - Haute disponibilité à 3 nœuds	Amazon Web Services : af-south-1 Afrique, Afrique du Sud	il y a 23 jours	...
 <b>data-pipeline-test-dvd-rental-kafka</b> Apache Kafka • Éteint data-pipeline : test-dv...	 Nœuds 3	Entreprise-4 2 processeurs / 4 Go de RAM / 600 Go de stockage - Haute disponibilité à 3 nœuds	Amazon Web Services : af-south-1 Afrique, Afrique du Sud	il y a 23 jours	...
 <b>data-pipeline-test-dvd-rental-flink</b> Apache Flink • Éteint data-pipeline : test-dv...	 Nœuds 3	Entreprise-4 2 CPU / 4 Go de RAM - Haute disponibilité à 3 nœuds	Amazon Web Services : af-south-1 Afrique, Afrique du Sud	il y a 23 jours	...



## 2.COMMENT L'INTEFACE CE PRESENTER AVANT

The screenshot displays the Aiven console interface. At the top, the navigation bar includes the Aiven logo, a breadcrumb trail (Foyer, Projets, Outils, Facturation, Appui, Admin), and user information (Mon Organisation, ?). A banner below the navigation bar states: "L'essai de la plateforme Aiven est actif. Il reste 30 jours pour utiliser des crédits d'essai de 30 dollars des États-Unis sur les services de plan non gratuits. Options de mise à niveau".

The left sidebar contains a menu with the following items: Retour au projet, Vue d', Integrations, Métrages, Logements, Utilisateurs (highlighted), LCA, Stockage à niveau, Thèmes, Sauvegardes, and Connecteurs.

The main content area is titled "Utilisateurs" and shows the path: Mon Organisation / mulongocharien044-a131 / kafka-368df8fe / Utilisateurs. A button "Ajouter un utilisateur de services" is located in the top right corner.

A note box states: "Note Vous pouvez télécharger le certificat CA pour ce service sur la page Vue d'ensemble. Aller à la page Vue d'ensemble".

Below the note is a search bar labeled "Search by user name".

The main content is a table with the following columns: Nom d'utilisateur, Mot de passe, Certificat, and Validité du certificat.

Nom d'utilisateur	Mot de passe	Certificat	Validité du certificat
avnadmin	- [eye icon] [copy icon]	<a href="#">Montrer la clé d'accès</a> <a href="#">Afficher le certificat d'accès</a>	Mise à jour

Créer un nouveau service Mon organisation / mulongochristian044-A131 / Sélectionnez le service / Apache Kafka®

Ensemble haute disponibilité à 3 nœuds

4. Stockage hiérarchisé

Stockage hiérarchisé

Cette fonctionnalité n'est pas disponible dans votre plan de service actuel et dans Inactif votre région. Choisissez une autre région à activer.

5. Nommez et étiquetez ce service

Le nom du service ne peut pas être modifié par la suite.

Nom\*

kafka-2bfc1b3

Ajouter une balise à ce service

Région

Afrique, Afrique du Sud - Amazon Web Services : Le Cap

Plan

Start-up 2

2 Processeur 2 Go de RAM

90 Go de stockage pas de sauvegardes

Ensemble haute disponibilité à 3 nœuds

Stockage hiérarchisé\*

Non disponible

Prix mensuel estimé\*

**Essai gratuit**

340 \$ / mois une fois les crédits d'essai consommés\*

\*Le prix mensuel estimé est basé sur 730 heures d'utilisation.

Créer un service

## Create topic



Topic name\*

message

Enable advanced configuration

No

Cancel

Create topic

Topic	Partitions	Replication	Min. insync replicas	Retention hours	Retention bytes	Cleanup p
__connect_configs-67785088-484b-4d41-ba96-56871fc2c8c4	1	3	1	168 hours	unlimited	compact
__connect_offsets-67785088-484b-4d41-ba96-56871fc2c8c4	25	3	1	168 hours	unlimited	compact
__connect_status-67785088-484b-4d41-ba96-56871fc2c8c4	5	3	1	168 hours	unlimited	compact
message	1	3	2	168 hours	unlimited	delete