Ce code met en œuvre un système de simulation et de traitement de données de trafic routier en utilisant Apache Kafka, un système de messagerie distribuée, et Python. Voici une analyse détaillée de ses différentes composantes et de leur fonctionnement :

Configuration et Initialisation de Kafka

Kafka Producer

- Configuré pour se connecter à un serveur Kafka spécifique (`ec2-100-25-131-75.compute-1.amazonaws.com:9092`).
- Utilisé pour envoyer des données au topic Kafka nommé `road1-traffic-data`.
- Serialize les données en JSON avant de les envoyer.

Kafka Consumer

- Établit une connexion au même topic pour consommer les données.
- Regroupé sous `road1-traffic-data-group` pour maintenir un suivi cohérent des messages consommés.
- Deserialize les messages de JSON.

Simulation de Données de Trafic

Génération de Données de Trafic

- Utilise des appels API pour simuler les taux d'entrée (`inflow`) et de sortie (`outflow`) du trafic.
- Envoie ces données, ainsi que d'autres paramètres (capacité de la route, trafic initial, etc.), au topic Kafka toutes les 5 secondes.

Calcul du Trafic

- Calcule la congestion du trafic en fonction du trafic entrant et sortant, de la capacité de la route et du temps écoulé.

Traitement des Données

Gestion des Messages en File d'Attente

- Utilise une file d'attente pour gérer les messages de manière thread-safe.
- Les messages sont consommés du topic Kafka et placés dans cette file d'attente.

Traitement des Messages

- Des threads supplémentaires sont lancés pour traiter les messages en file d'attente.
- Calcule le ratio de congestion du trafic pour chaque message et écrit les résultats dans un fichier JSON.

Multithreading

Threads Producteur et Consommateur

- Un thread séparé est utilisé pour envoyer et un autre pour consommer les données de trafic.
- Cela assure une exécution simultanée de la production et de la consommation de données.

Threads de Traitement

- Plusieurs threads sont utilisés pour traiter les données en file d'attente.

- Permet un traitement parallèle des données, améliorant l'efficacité.

Utilisation d'API Externes

API pour les Taux de Trafic

- Le code utilise des API externes pour obtenir des taux de trafic aléatoires.
- Cela simule une situation réelle où les taux de trafic peuvent fluctuer.

Points Clés

Robustesse et Extensibilité

- Le code est conçu pour être robuste (gestion des erreurs) et extensible (facile à adapter pour d'autres routes ou paramètres).

Dépendance Externe

- Le système dépend des réponses des API externes et du serveur Kafka.

Objectif:

- Le script vise à simuler, produire, consommer et traiter des données de trafic en temps réel, fournissant une base pour des analyses de trafic plus complexes.

En résumé, ce code illustre une application pratique de Kafka pour le traitement de flux de données en temps réel, avec une simulation de trafic routier comme cas d'utilisation.