NAME- Devansh Mehta

ID - 202318053

Assignment Report: Building a Real-Time E-commerce Order Processing System Using Kafka

Introduction:

This report outlines the steps and best practices for developing a Kafka-based system to manage e-commerce orders in real-time. The system involves setting up Kafka, implementing producers and consumers, and integrating message filtering logic to ensure efficient order processing.

Step 1: Setting Up Kafka

1. Installing Kafka: Start by ensuring Kafka is successfully installed and running on the designated system or server.

2. Creating Kafka Topics: Establish two Kafka topics, namely "inventory_orders" and "delivery_orders," which will serve as the channels for producers to send relevant messages.

Step 2: Implementing Kafka Producers

1. Inventory Orders Producer (inventory_orders_producer):

   - Develop a producer specifically designed to filter messages with the type field set to "inventory."

   - Utilize Kafka to read inventory-related events from various sources such as databases or event streams and direct messages with the "inventory" type to the "inventory_orders" topic.

2. Delivery Orders Producer (delivery_orders_producer):

- Create a producer that filters messages based on the type field, specifically targeting "delivery" messages.
- Implement functionality to read delivery-related events and transmit messages with the "delivery" type to the "delivery_orders" topic.

Step 3: Implementing Kafka Consumers

1. Inventory Data Consumer (inventory_data_consumer):

- Configure a Kafka consumer to subscribe to the "inventory_orders" topic.
- Develop processing logic to handle inventory messages received, including updating relevant databases or systems.

2. Delivery Data Consumer (delivery_data_consumer):

- Set up a Kafka consumer for the "delivery_orders" topic.
- Implement logic to process delivery-related messages, such as scheduling deliveries, updating delivery statuses, and notifying customers accordingly.

Step 4: Developing Message Filtering Logic

1. Producer Message Filtering:

- Integrate message filtering logic within each producer (inventory_orders_producer and delivery_orders_producer) to selectively send messages based on the type field from incoming data sources.
- Ensure that only messages matching the desired types (inventory or delivery) are transmitted to Kafka.

Additional Considerations:

- Error Handling: Implement robust error handling mechanisms within producers and consumers to gracefully manage exceptions or failed operations.
- Scalability: Design the system with scalability in mind, considering Kafka partitioning, consumer groups, and effective scaling strategies to accommodate increasing loads.
- Monitoring and Logging: Utilize Kafka monitoring tools and logging frameworks to monitor system performance, detect anomalies, and troubleshoot issues effectively.

Conclusion:

By following these steps and incorporating best practices, a resilient Kafka-based e-commerce order management system capable of real-time inventory management and delivery processing can be successfully developed.