

Решение задания на проектирование интеграции через Kafka

1. Топики Kafka

- 1. **Order-Confirmation:** для сообщений о подтверждении заказа.
- 2. **Order-Status-Update:** для сообщений о смене статуса заказа.
- 3. **Order-Cancellation:** для сообщений об отмене заказа.
- 4. **Customer-Notification:** для уведомлений клиентам.

2. Формат сообщений

- **Order-Confirmation:**
 - Key: `order_id`
 - Value: `{ "customer_id": "XYZ", "products": [{"id": "A", "price": 10}, {"id": "B", "price": 20}], "status": "new" }`
- **Order-Status-Update:**
 - Key: `order_id`
 - Value: `{ "status": "processing/shipped/cancelled" }`
- **Order-Cancellation:**
 - Key: `order_id`
 - Value: `{ "reason": "customer request/stock unavailability" }`
- **Customer-Notification:**
 - Key: `customer_id`
 - Value: `{ "order_id": "123", "message": "Your order is now shipped." }`

3. Обработка ошибок и отказоустойчивость

- **Dead-letter queues:** для сообщений, которые не удалось обработать.
- **Retry policies:** автоматическая попытка повторной обработки в случае временной ошибки.
- **Monitoring & Alerts:** настроить систему мониторинга для отслеживания ошибок.

4. Хранение состояния

- Хранение текущего статуса каждого заказа в базе данных для быстрого доступа.
- Снапшоты статуса заказа для возможности восстановления.

Дополнительные задачи

1. Горизонтальное масштабирование:

- Увеличение числа партиций для каждого топика.
- Использование нескольких брокеров.

2. Аудит:

- Использование отдельного топика Kafka для хранения всех операций с заказами.
- Применение системы логгирования для аудита событий.

Бонус

1. Идемпотентность:

- Использование уникальных идентификаторов транзакций.
- Проверка на сервере, была ли операция уже выполнена для данного идентификатора.