Feature Request: Add Primary Key Data to Kafka Messages

Feature Request Description

Problem Statement:

OpenLogReplicator currently supports Kafka as a destination for change data capture (CDC). However, it only populates the value field in Kafka messages and does not utilize the key field. This limitation becomes problematic when dealing with high-volume data flows. Without a defined key, Kafka cannot guarantee that messages with the same primary key values are sent to the same partition. This can lead to issues with message ordering when consuming data with parallel consumers, potentially causing data integrity problems, such as processing updates out of order.

Proposed Solution:

We propose adding the capability to populate the Kafka key field with a string composed of the table name and primary key (PK) values, separated by a delimiter (e.g., tablename;PK1;PK2). This will ensure that messages with the same key are consistently sent to the same partition, maintaining proper order during parallel consumption.

For tables without defined primary keys, the key field can be left empty or filled with a unique index if available. Alternatively, a CRC32 hash of the key string can be used to address concerns about long keys, ensuring that messages with the same logical key are partitioned consistently.

Benefits:

1. Improved Data Consistency: Ensuring messages with the same primary key are sent to the same partition helps maintain data order and consistency when consuming Kafka messages with multiple consumers.
2. Enhanced Parallel Processing: By correctly partitioning data, it becomes possible to efficiently consume and process large volumes of data in parallel, reducing the time needed to handle high

throughput.

3. Optional Configuration: The feature can be made optional via configuration, allowing users to enable or disable key population based on their specific needs and constraints.

Call for Sponsorship:

We seek sponsorship to support the development and implementation of this feature. This enhancement will be particularly valuable for organizations with high data throughput requirements and those seeking to maintain data consistency in distributed systems. By sponsoring this development, companies can ensure that OpenLogReplicator meets their needs for reliable and efficient data replication.