KIP-399: Extend ProductionExceptionHandler to cover serialization exceptions

This KIP is aimed at improving the error-handling semantics in Kafka Streams when Kafka Steams fails to serialize a message to the downstream sink by providing an interface that can provide custom messaging of the error (e.g. report to a custom metrics system) and indicate to Streams whether or not it should re-throw the Exception, thus causing the application to fall over.

Status

Current state: *Accepted*

Discussion thread: *here*

JIRA: *KAFKA-7499*

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

In KIP-210, an exception handler for the write path was introduced. This exception handler covers exception that are raised in the producer callback.

However, serialization happens before the data is handed to the producer with Kafka Streams itself and the producer uses `byte[]/byte[]` key-value-pair types.

Thus, we might want to extend the ProductionExceptionHandler to cover serialization exception, too, to skip over corrupted output messages. An example could be a "String" message that contains invalid JSON and should be serialized as JSON.

Public Interfaces

We are proposing addition of a new method in ProductionExceptionHandler interface, `handleSerializationException`, that has the following signature:

```
ProductionExceptionHandlerResponse handleSerializationException(ProducerRecord record, Exception exception);
```

To accept different types of records from multiple topologies, `ProducerRecord` is defined without generics. The above interface method will have a `default` implementation which returns `ProductionExceptionHandlerResponse.FAIL`

Proposed Changes

This implementation will override the new method, handleSerializationException, in the following class:

- `AlwaysContinueProductionExceptionHandler` and returns response as `CONTINUE`
- No need to implement in `DefaultProductionExceptionHandler,` as the response is set to `FAIL` by default.

We'll implement the following error handling logic to the send in RecordCollectorImpl. The new method, `handleSerializationException`, in ProductionExceptionHandler will not be invoked for

1. `ClassCastException` is thrown while serializing record key / value. We will continue to throw this exception and not invoke the new method. This will allow the current behavior to continue as this can help identify misconfigured serdes

It will be invoked for

1. Any other unchecked exceptions, that thrown during record key / value serialization.
   a. If the result is `CONTINUE`, log a note at `WARN` that we received that result and are not failing Streams as a result.
   b. If the result is `FAIL`, log a message at `ERROR` that we received that result and set `sendException` so Streams will fail.

Earlier, we are invoking the error handler only when there are any exceptions in producer callback. Now, we also invoke the handler when hitting the serialization exception. As explained in KIP-210, this will facilitate a number of error handling scenarios.

Compatibility, Deprecation, and Migration Plan

The default behavior will be consistent with the existing behavior. The new method, `handleSerializationException`, will have a implementation that is set to `FAIL` by default.

Rejected Alternatives

We have considered to reuse the existing `handle(ProducerRecord<byte[], byte[]> record, Exception exception)` method in `ProductionExceptionHandler`, but it has the following limitation:

1. The parameter `ProducerRecord` key and value type is set to `byte[],` on hitting the serialization exception the record key and value type may not be `byte[].`