KIP-811: Add config repartition.purge.interval.ms to Kafka Streams

Status

Current state: Accepted

Discussion thread: here

JIRA: here

Pull Request: here

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

Motivation

Kafka Streams treats repartition topics differently to regular topics. Instead of setting arbitrary retention criteria and having the broker cleanup old records, Kafka Streams sets infinite retention on repartition topics and explicitly deletes records once they've been committed to the next topic in their Topology. Currently, this is done every time the Task is committed, resulting in explicit "delete records" requests being sent every `commit.interval.ms` milliseconds.

When `commit.interval.ms` is set very low, for example when `processing.guarantee` is set to `exactly_once_v2`, this causes delete records requests to be sent extremely frequently, potentially reducing throughput and causing a high volume of log messages to be logged by the brokers.

Public Interfaces

New configuration options

Name	Type	Importance	Default	Description
`repartition.purge.interval.ms`	Long	LOW	30000	The minimum interval in milliseconds with which to delete fully consumed records from repartition topics. Purging will occur after at least this value since the last purge, but may be delayed until later. (Note, unlike `commit.interval.ms`, the default for this value remains unchanged when `processing.guarantee` is set to `exactly_once_v2`).

Proposed Changes

Adding a new configuration option, `repartition.purge.interval.ms`, that configures the period of these explicit record deletions, will resolve the issue by enabling users to tune the `commit.interval.ms` and `repartition.purge.interval.ms` separately.

Compatibility, Deprecation, and Migration Plan

- The interval between explicit delete requests for repartition records will no longer be coupled to `commit.interval.ms`. Default behaviour is unchanged, however:
  - When `commit.interval.ms` is explicitly modified by the user, old repartition records will no longer be deleted on every commit.
  - When `processing.guarantee` is set to `exactly_once_v2`, since the default `commit.interval.ms` is changed internally to `100 ms`, old repartition records will no longer be deleted on every commit.
  - Users can regain this coupling by explicitly configuring both `commit.interval.ms` and `repartition.purge.interval.ms` to the same value.

Rejected Alternatives

- Purging after *exactly* the configured amount of time has elapsed was rejected, as it would necessitate a design that would likely have a negative performance or correctness impact.

- Purging after a specified multiple of commits was rejected, as it would be tightly coupled to the value of another config parameter (`commit.interval.ms`), which would cause a likely unintended change to purge behavior whenever the commit interval was reconfigured, including implicitly when the processing.guarantee is changed.