# KIP-216: IQ should throw different exceptions for different errors

## Status

**Current state**: *Accepted(vote)*

**Discussion thread**: *here*

**JIRA**: KAFKA-5876 - Getting issue details... STATUS

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

## Motivation

Currently, IQ throws ***InvalidStateStoreException*** for any types of error, that means a user cannot handle different types of error.

Because of that, we should throw different exceptions for each type.

## Proposed Changes

To distinguish different types of error, we need to handle all ***InvalidStateStoreException*** better during these public methods invoked. The main change is to introduce new exceptions that extend from ***InvalidStateStoreException***. ***InvalidStateStoreException*** is not thrown at all anymore, but only new sub-classes.

```
public class StreamsNotStartedException extends InvalidStateStoreException
public class StreamsRebalancingException extends InvalidStateStoreException
public class StateStoreMigratedException extends InvalidStateStoreException
public class StateStoreNotAvailableException extends InvalidStateStoreException
public class UnknownStateStoreException extends InvalidStateStoreException
public class InvalidStateStorePartitionException extends InvalidStateStoreException
```

- **StreamsNotStartedException**: will be thrown when stream thread state is *CREATED*, the user can retry until to *RUNNING*.
- **StreamsRebalancingException**: will be thrown when stream thread is not running and stream state is *REBALANCING*, the user just retry and wait until rebalance finished (*RUNNING*).
- **StateStoreMigratedException**: will be thrown when state store already closed and stream state is *RUNNING*. The user need to rediscover the store and cannot blindly retry as the store handle is invalid and a new store handle must be retrived.
- **StateStoreNotAvailableException**: will be thrown when state store closed and stream state is *PENDING_SHUTDOWN* / *NOT_RUNNING* / *ERROR*. The user cannot retry when this exception is thrown.
- **UnknownStateStoreException**: will be thrown when passing an unknown state store. The user cannot retry when this exception is thrown.
- **InvalidStateStorePartitionException**: will be thrown when user requested partition is not available on the stream instance.

The following is the public methods that users will call to get state store instance:

- *KafkaStreams*
  - @Deprecated store(storeName, queryableStoreType)
  - store(storeQureyParams)

All the above methods could be throw exceptions:

***StreamsNotStartedException***, ***StreamsRebalancingException***, ***StateStoreMigratedException***, ***StateStoreNotAvailableException,*** ***UnknownStateStoreException***, **InvalidStateStorePartitionException**

The following is the public methods that users will call to get store values:

- *interface ReadOnlyKeyValueStore(class CompositeReadOnlyKeyValueStore)*
  - get(key)
  - range(from, to)
  - all()
  - approximateNumEntries()
- *interface ReadOnlySessionStore(class CompositeReadOnlySessionStore)*
  - fetch(key)
  - fetch(from, to)
- *interface ReadOnlyWindowStore(class CompositeReadOnlyWindowStore)*
  - fetch(key, time)
  - fetch(key, from, to)
  - fetch(from, to, fromTime, toTime)
  - all()
  - fetchAll(from, to)
  - @Deprecated fetch(key, timeFrom, timeTo)
  - @Deprecated fetch(from, to, timeFrom, timeTo)
  - @Deprecated fetchAll(timeFrom, timeTo)
- *interface KeyValueIterator(class DelegatingPeekingKeyValueIterator)*
  - next()
  - hasNext()
  - peekNextKey()

All the above methods could be throw following exceptions:

***StreamsRebalancingException***, ***StateStoreMigratedException***, ***StateStoreNotAvailableException, InvalidStateStorePartitionException***

## Compatibility, Deprecation, and Migration Plan

- All new exceptions extend from InvalidStateStoreException, this change will be fully backward compatible.

## Rejected Alternatives

None.