

KIP-666: Add Instant-based methods to ReadOnlySessionStore

- [Status](#)
- [Motivation](#)
- [Public Interfaces](#)
- [Proposed Changes](#)
- [Compatibility, Deprecation, and Migration Plan](#)
- [Rejected Alternatives](#)

Status

Current state: Adopted

Vote thread: [here](#)

Discussion thread: [here](#)

JIRA: [here](#)

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

Motivation

Interactive-Query APIs are designed for usability. Instant-based methods were introduced in [KIP-358](#) to represent time ranges.

On the other hand, read-write interfaces used by Processor API are tuned for performance. Therefore, long-based methods are kept to avoid performance penalties caused by object allocation.

ReadOnlySessionStore API is misaligned with the improvements introduced into ReadOnlyWindowStore in [KIP-358](#).

This proposal aims to solve this issue, adding Instant-based alternative methods to the interactive query API for Session Stores.

[KIP-617](#) is already moving read methods from SessionStore into ReadOnlySessionStore

Public Interfaces

- org.apache.kafka.streams.state.ReadOnlySessionStore
 - add method: KeyValueIterator<Windowed<K>, AGG> findSessions(final K key, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)
 - add method: KeyValueIterator<Windowed<K>, AGG> findSessions(final K keyFrom, final K keyTo, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)
 - add method: AGG fetchSession(final K key, final Instant sessionStartTime, final Instant sessionEndTime)
 - add method: KeyValueIterator<Windowed<K>, AGG> backwardFindSessions(final K key, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)
 - add method: KeyValueIterator<Windowed<K>, AGG> backwardFindSessions(final K keyFrom, final K keyTo, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)
- org.apache.kafka.streams.state.SessionStore
 - add method with default implementation calling long-based method:
 - KeyValueIterator<Windowed<K>, AGG> findSessions(final K key, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)
 - KeyValueIterator<Windowed<K>, AGG> findSessions(final K keyFrom, final K keyTo, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)
 - AGG fetchSession(final K key, final Instant sessionStartTime, final Instant sessionEndTime)
 - KeyValueIterator<Windowed<K>, AGG> backwardFindSessions(final K key, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)
 - KeyValueIterator<Windowed<K>, AGG> backwardFindSessions(final K keyFrom, final K keyTo, final Instant earliestSessionEndTime, final Instant latestSessionStartTime)

Proposed Changes

Changes to ReadOnlySessionStore, considering new methods added on [KIP-617](#):

```
public interface ReadOnlySessionStore<K, AGG> {  
    // Moving read functions from SessionStore to ReadOnlySessionStore  
    KeyValueIterator<Windowed<K>, AGG> findSessions(final K key, final Instant earliestSessionEndTime, final  
    Instant latestSessionStartTime);
```

[illegible]

```

final long latestSessionStartTime);

@Override
default KeyValueTypeIterator<Windowed<K>, AGG> backwardFindSessions(final K key,
                                                                    final Instant earliestSessionEndTime,
                                                                    final Instant latestSessionStartTime) {

    return backwardFindSessions(
        key,
        ApiUtils.validateMillisecondInstant(earliestSessionEndTime, prepareMillisCheckFailMsgPrefix
(earliestSessionEndTime, "earliestSessionEndTime")),
        ApiUtils.validateMillisecondInstant(latestSessionStartTime, prepareMillisCheckFailMsgPrefix
(latestSessionStartTime, "latestSessionStartTime")));
    }

    @Override
    default AGG backwardFetchSession(final K key, final Instant sessionStartTime, final Instant sessionEndTime)
    {
        return backwardFetchSession(
            key,
            ApiUtils.validateMillisecondInstant(sessionStartTime, prepareMillisCheckFailMsgPrefix
(sessionStartTime, "sessionStartTime")),
            ApiUtils.validateMillisecondInstant(sessionEndTime, prepareMillisCheckFailMsgPrefix(sessionEndTime,
"sessionEndTime"))));
    }
}

```

Compatibility, Deprecation, and Migration Plan

- Custom IQ Session Stores will have to implement new methods if haven't implement read-write APIs yet.

Rejected Alternatives

None yet.