

# KIP-960: Support single-key\_single-timestamp interactive queries (IQv2) for versioned state stores

- [Status](#)
- [Motivation](#)
- [Proposed Changes](#)
- [Examples](#)
- [Compatibility, Deprecation, and Migration Plan](#)
- [Test Plan](#)

## Status

**Current state:** Accepted

**Discussion thread:** [here](#)

**JIRA:** [KAFKA-15346](#)

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

## Motivation

The main goal is supporting interactive queries in presence of versioned state stores ([KIP-889](#)) in AK. For this KIP, the following query types are considered to be implemented. This KIP discusses single-key, single-timestamp queries. Other types of IQs are explained in the following KIPs ([KIP-968](#) and [KIP-969](#))

**Key Queries with single timestamp:**

1. single-key latest-value lookup
2. single-key lookup with asOf timestamp

## Proposed Changes

In this KIP we introduce the class **VersionedKeyQuery** with an *Optional* field to store the *asOfTimestamp* value. The method *asOf* creates key queries having *asOfTimestamp* value as well.

Defining the *latest()* method is not needed since returning the latest value has been always the default assumption. In other words, If a query is created without calling the *asOf()* method, it will return the latest value of the key.

## VersionedKeyQuery.java

```
package org.apache.kafka.streams.query;

/**
 * Interactive query for retrieving a single record from a versioned state store based on its key and
 * timestamp.
 * <p>
 * See KIP-960 for more details.
 */
@Evolving
public final class VersionedKeyQuery<K, V> implements Query<VersionedRecord<V>> {

    private final K key;
    private final Optional<Instant> asOfTimestamp;

    private VersionedKeyQuery(final K key, final Optional<Instant> asOfTimestamp) {
        this.key = Objects.requireNonNull(key);
        this.asOfTimestamp = asOfTimestamp;
    }

    /**
     * Creates a query that will retrieve the latest record from a versioned state store identified by {@code
     * key} if the key exists
     * (or {@code null} otherwise).
     * @param key The key to retrieve
     * @throws NullPointerException if @param key is null
     * @param <K> The type of the key
     * @param <V> The type of the value that will be retrieved
     * @throws NullPointerException if @param key is null
     */
    public static <K, V> VersionedKeyQuery<K, V> withKey(final K key);

    /**
     * Specifies the as of timestamp for the key query. The key query returns the record
     * with the greatest timestamp <= asOfTimestamp
     * @param asOfTimestamp The as of timestamp for timestamp
     * if @param asOfTimestamp is null, it will be considered as Optional.empty()
     */
    public VersionedKeyQuery<K, V> asOf(final Instant asOfTimestamp);

    /**
     * The key that was specified for this query.
     */
    public K key();

    /**
     * The timestamp of the query, if specified
     */
    public Optional<Instant> asOfTimestamp();
}
```

## Examples

The following example illustrates the use of the VersionedKeyQuery class to query a versioned state store.

```
builder.table(
    "my_topic",
    Consumed.with(Serdes.Integer(), Serdes.Integer()),
    Materialized.as(Stores.persistentVersionedKeyValueStore(
        "my_store",
        Duration.ofMillis(HISTORY_RETENTION)
    ))
);

final VersionedKeyQuery<Integer, Integer> query = VersionedKeyQuery.withKey(1).asOf(Instant.parse("2023-08-03T10:37:30.00Z"));

final StateQueryRequest<VersionedRecord<Integer>> request =
    inStore("my_store").withQuery(query);

final StateQueryResult<VersionedRecord<Integer>> result = kafkaStreams.query(request);
```

## Compatibility, Deprecation, and Migration Plan

- Since this is a completely new set of APIs, no backward compatibility concerns are anticipated.
- Since nothing is deprecated in this KIP, users have no need to migrate unless they want to.

## Test Plan

The single-key\_single-timestamp interactive queries will be tested in versioned stored IQv2 integration test (like non-versioned key queries).