**Title:** Redis XDEL

**Excerpt:** Redis XDEL command is used to delete an entry from a Redis stream. This command takes in the stream key and the entry ID. Furthermore, the XDEL command can delete multiple entries within a stream at once by specifying the preferred number of entry IDs.

**Permalink:** redis-xdel

**Category:** Redis

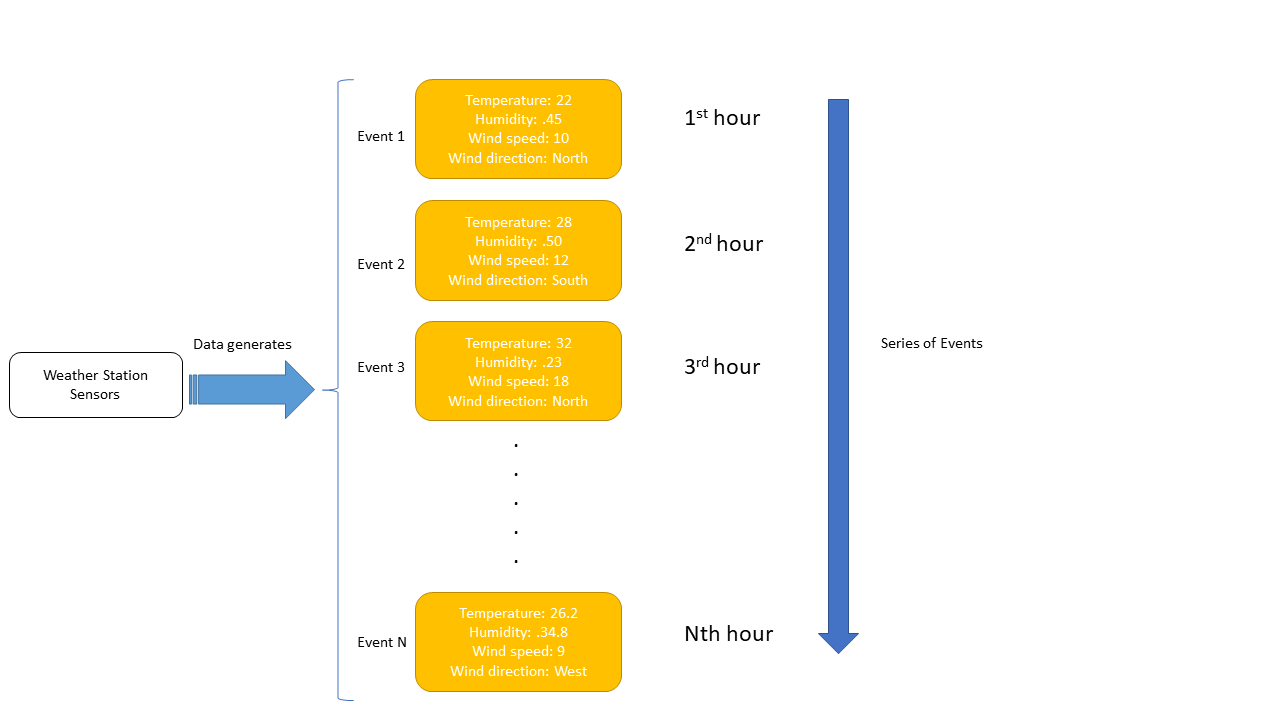
# 

# 

# 

# Data streaming with Redis

In the modern-day world, data is generated continuously from social media platforms, websites, and IoT devices. Hence, the data streaming is unstoppable. A stream can be identified as a series of events generated from multiple concurrent sources. As an example, a weather station generates weather data at a given location hourly. Each event consists of weather data such as temperature, humidity, wind speed and direction.

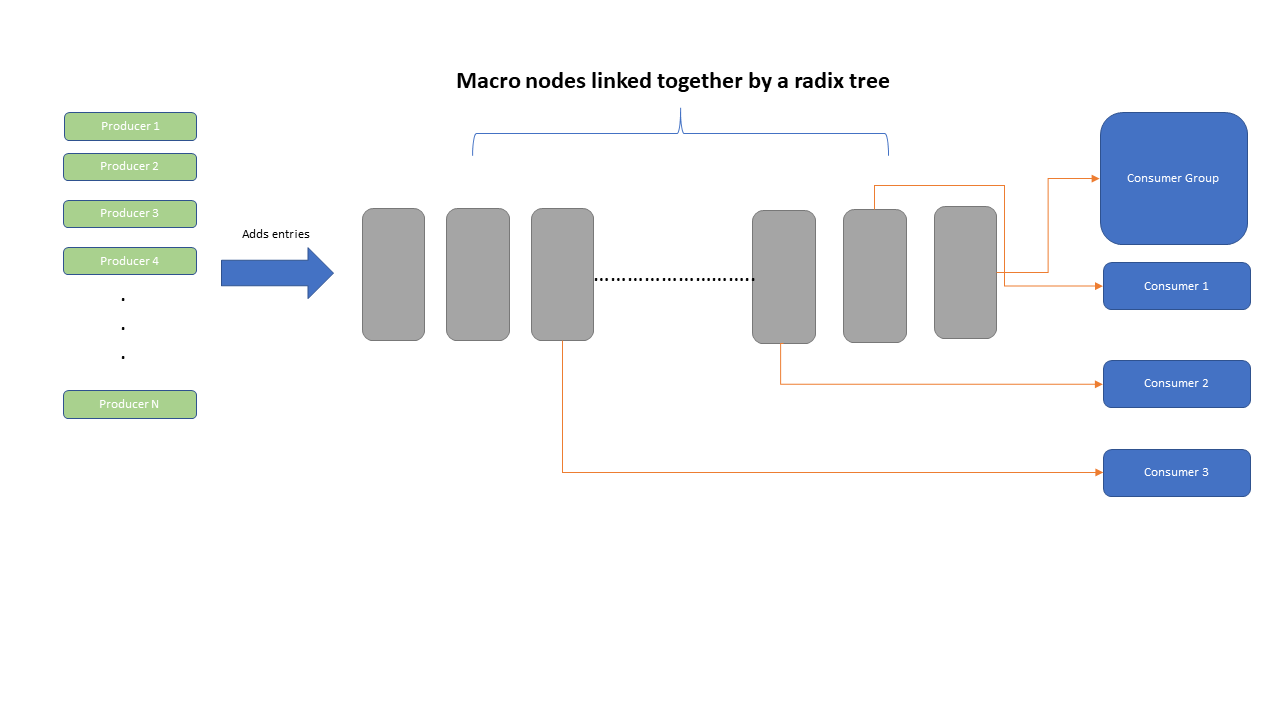


Redis streams help capture, manage, and make sense of this large amount of streamed data.

Usually, the sources that generate data or write to a stream are called producers. Each producer adds a new entry to the Redis stream with a unique event id where an event consists of one or more field-value pairs similar to a Redis hash. On the other end, consumers are reading from the stream to generate some meaningful insights. The producers and consumers do not need to know about each other’s implementation details which make Redis streams more robust.

# Redis stream entries and macro nodes

Redis stream type is based on widely used radix-tree data structures. Usually, a radix-tree consists of several macro nodes where one macro node can store multiple stream entries. Since the radix tree is a memory-optimized prefix tree, Redis streams can be used with low memory consumption. Furthermore, the underlying radix trees enable streams to access data randomly or within a specified range at higher efficiency.



The XADD command is used to add a new entry to a macro node in the stream. When you add an entry it is immutable. Hence, it is called an append-only data structure.

# The XDEL command

In some cases, maintaining millions of historical stream data might not be worth it due to data expirations. Therefore, Redis streams support the deletion of entries from the data structure and free up some memory. We can use the XDEL command to delete an entry from a specified stream.

## Syntax

| XDEL <**stream\_key**> <**entry\_id**> [entry\_id …] |
| --- |

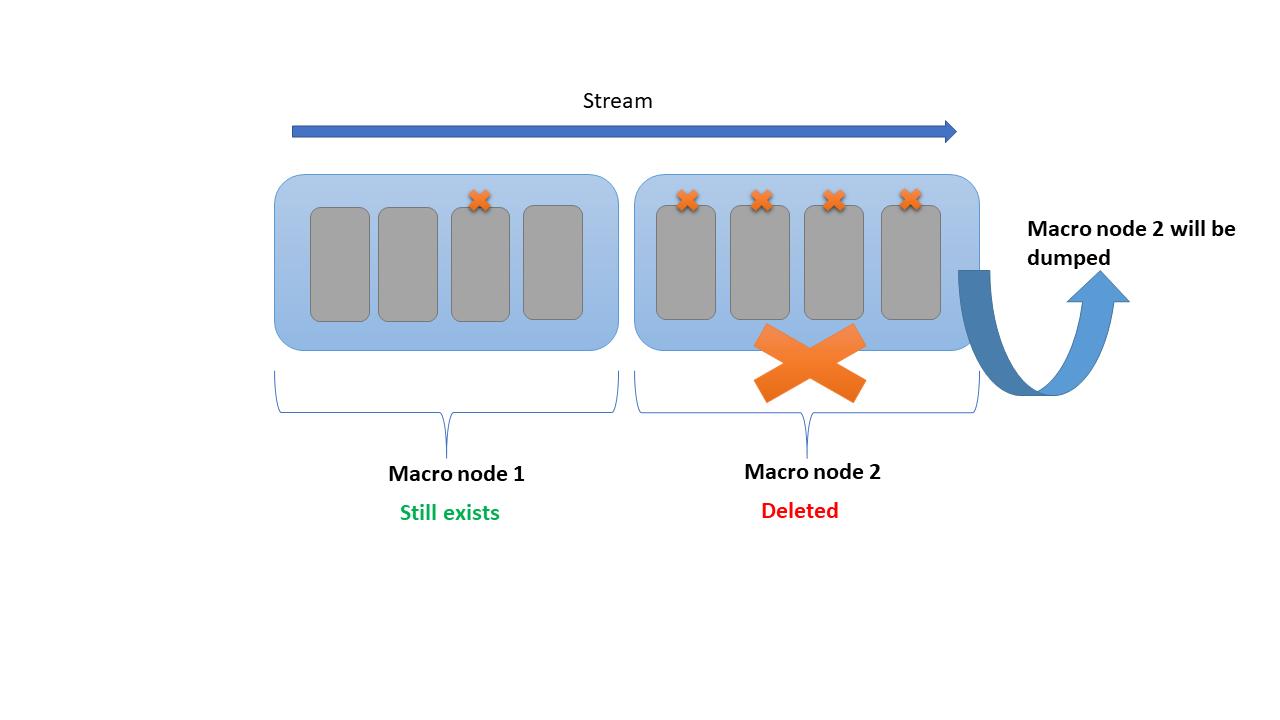
stream\_key: The Redis key which is used to uniquely identify a stream

entry\_id: One stream can have several entries which belong to multiple macro nodes. The ‘entry\_id’ is the unique identifier to identify an entry.

The XDEL command accepts multiple entry\_ids. Hence you can delete multiple entries per command. This command returns the count of the deleted stream entries. Whenever the given entry id does not exist in a stream, the returned count is less than the specified id count.

## Redis stream entry deletion mechanism

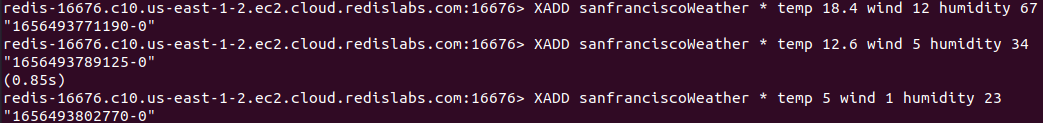
The XDEL command does not dump the stream's entries straightaway. Instead, it marks the entry as deleted first. Hence, the stream entry will keep utilizing the allocated memory. The memory will be released when all the entries belonging to a macro node have been marked as deleted. It would force the Redis server to release the memory for that macro node.



## Example 01 - Delete old weather data entry from a stream

Let’s first create a stream to store weather data in the san-Francisco city using the XADD command. Assume that the sensors produce data daily at noon. Hence, we will be adding data for three days in this example.

| XADD sanfranciscoWeather \* temp 18.4 wind 12 humidity 67  XADD sanfranciscoWeather \* temp 12.6 wind 5 humidity 34  XADD sanfranciscoWeather \* temp 5 wind 1 humidity 23 |
| --- |



Let’s assume the first entry is from the last month and we do not need to keep track of last month's data now.

Hence, we can use the XDEL command to delete the first entry with the id of *1656493771190-0*.

| XDEL sanfranciscoWeather 1656493771190-0 |
| --- |



The XDEL command returned the integer 1 which means only one entry has been deleted from the stream.

Let’s use the XRANGE command to read the stream and verify that the entry with an id of *1656493771190-0* has been marked as deleted.

| XRANGE sanfranciscoWeather - + |
| --- |



As expected, only the last two entries are available and the first entry has been deleted.

## Example 02 - Delete multiple weather entries from a stream

Let’s say we need to delete multiple entries from a stream at once. The XDEL command supports it. You can specify the entry ids as shown in the following.

We’ll delete the remaining two entries from the sanfranciscoWeather stream that we have created in the previous example. The 1656493789125-0 and 1656493802770-0 are the ids of the remaining two entries.

| XDEL sanfranciscoWeather 1656493789125-0 1656493802770-0 |
| --- |



As expected, the command return value is 2 which indicates two entries have been deleted.

Let’s read the stream.



The sanfranciscoWeather stream is empty.

# Conclusion

Redis streams can be identified as a series of events produced by concurrent sources. A stream can hold several entries where each entry has a unique id and consists of field-value pairs. Redis streams are based on radix tree data structure which makes them highly efficient and memory-optimized. The entries added to a stream can be deleted using the Redis XDEL command. This command marks the entry as deleted instead of dumping the entry and releasing memory. It reclaims the memory when all the entries of a macro node are marked as deleted.