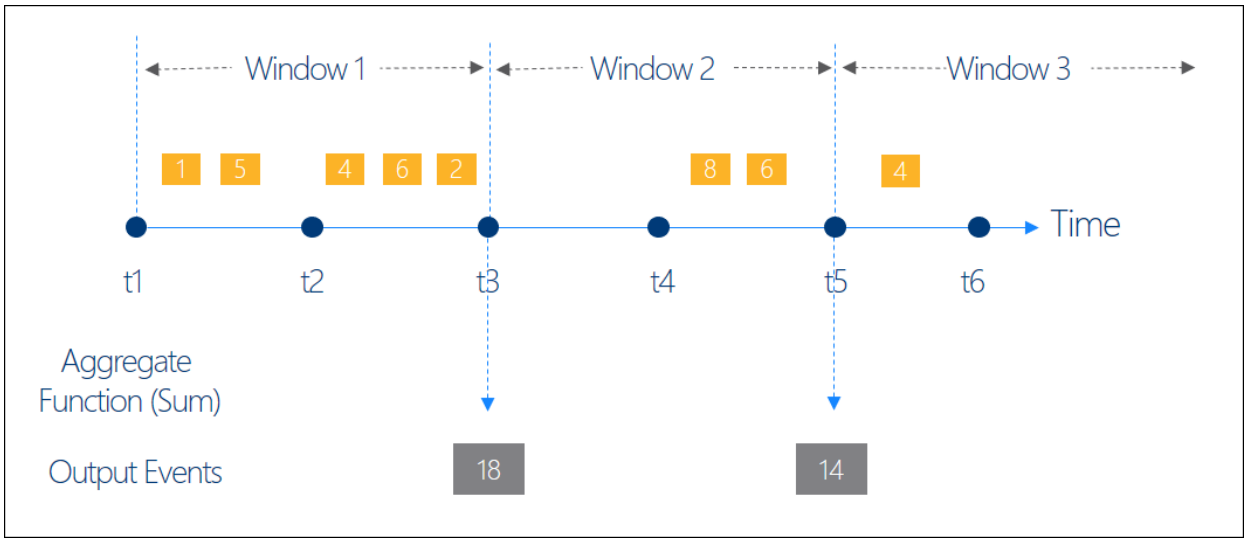
title	description	author	ms.author	ms.service	ms.topic
Introduction to Azure Stream Analytics windowing functions	This article describes four windowing functions (tumbling, hopping, sliding, session) that are used in Azure Stream Analytics jobs.	jseb225	jeanb	stream-analytics	conceptual

Introduction to Stream Analytics windowing functions

In time-streaming scenarios, performing operations on the data contained in temporal windows is a common pattern. Stream Analytics has native support for windowing functions, enabling developers to author complex stream processing jobs with minimal effort.

There are five kinds of temporal windows to choose from: **Tumbling**, **Hopping**, **Sliding**, **Session**, and **Snapshot** windows. You use the window functions in the **GROUP BY** clause of the query syntax in your Stream Analytics jobs. You can also aggregate events over multiple windows using the **Windows()** function.

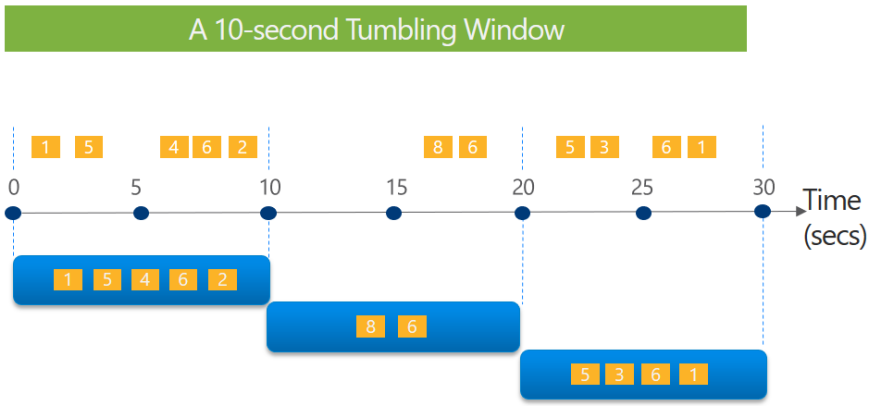
All the windowing operations output results at the **end** of the window. Note that when you start a stream analytics job, you can specify the *Job output start time* and the system will automatically fetch previous events in the incoming streams to output the first window at the specified time; for example when you start with the *Now* option, it will start to emit data immediately. The output of the window will be single event based on the aggregate function used. The output event will have the time stamp of the end of the window and all window functions are defined with a fixed length.



Tumbling window

Tumbling window functions are used to segment a data stream into distinct time segments and perform a function against them, such as the example below. The key differentiators of a Tumbling window are that they repeat, do not overlap, and an event cannot belong to more than one tumbling window.

Tell me the count of Tweets per time zone every 10 seconds

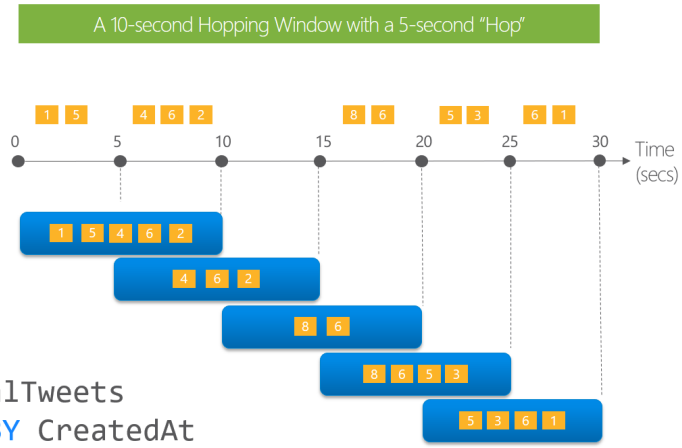



```
SELECT TimeZone, COUNT(*) AS Count
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY TimeZone, TumblingWindow(second,10)
```

Hopping window

Hopping window functions hop forward in time by a fixed period. It may be easy to think of them as Tumbling windows that can overlap and be emitted more often than the window size. Events can belong to more than one Hopping window result set. To make a Hopping window the same as a Tumbling window, specify the hop size to be the same as the window size.

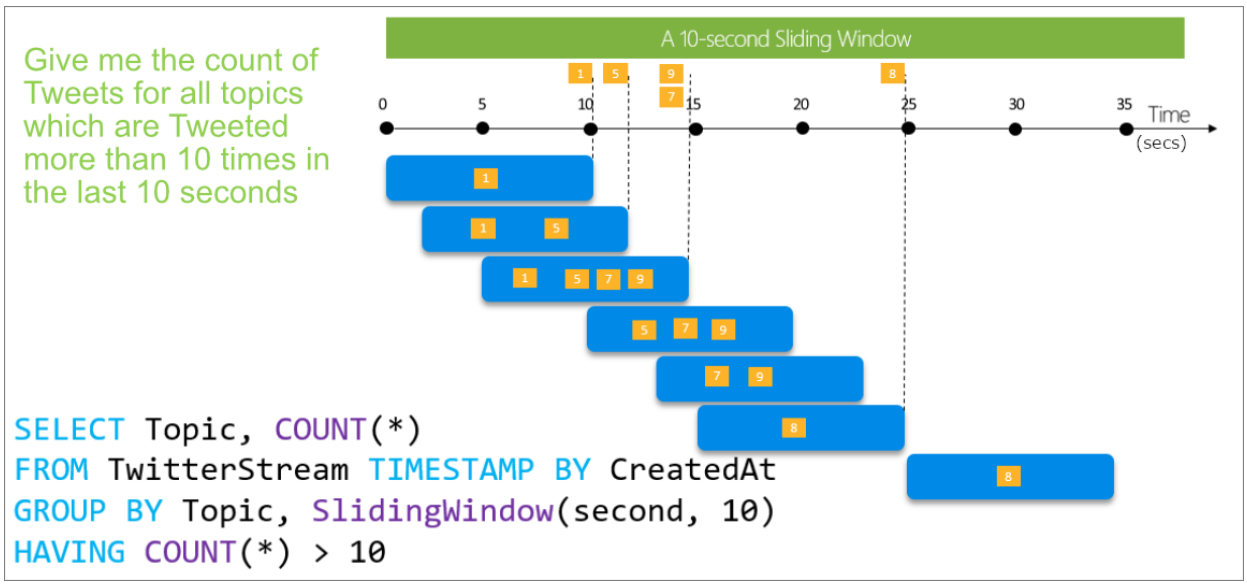
Every 5 seconds give me the count of Tweets over the last 10 seconds




```
SELECT Topic, COUNT(*) AS TotalTweets
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY Topic, HoppingWindow(second, 10 , 5)
```

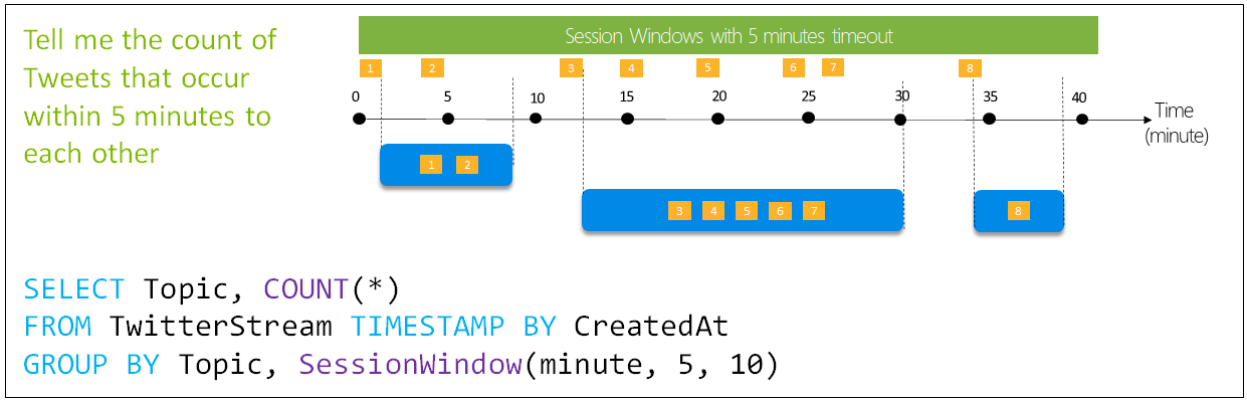
Sliding window

Sliding windows, unlike Tumbling or Hopping windows, output events only for points in time when the content of the window actually changes. In other words, when an event enters or exits the window. So, every window has at least one event. Similar to Hopping windows, events can belong to more than one sliding window.



Session window

Session window functions group events that arrive at similar times, filtering out periods of time where there is no data. It has three main parameters: timeout, maximum duration, and partitioning key (optional).



A session window begins when the first event occurs. If another event occurs within the specified timeout from the last ingested event, then the window extends to include the new event. Otherwise if no events occur within the timeout, then the window is closed at the timeout.

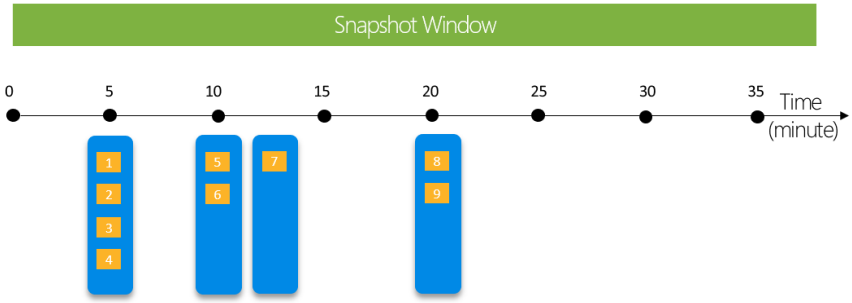
If events keep occurring within the specified timeout, the session window will keep extending until maximum duration is reached. The maximum duration checking intervals are set to be the same size as the specified max duration. For example, if the max duration is 10, then the checks on if the window exceed maximum duration will happen at t = 0, 10, 20, 30, etc.

When a partition key is provided, the events are grouped together by the key and session window is applied to each group independently. This partitioning is useful for cases where you need different session windows for different users or devices.

Snapshot window

Snapshot windows groups events that have the same timestamp. Unlike other windowing types, which require a specific window function (such as SessionWindow(), you can apply a snapshot window by adding System.Timestamp() to the GROUP BY clause.

Give me the count of tweets with the same topic type that occur at exactly the same time




```
SELECT Topic, COUNT(*)
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY Topic, System.Timestamp()
```

Next steps

- Introduction to Azure Stream Analytics
- Get started using Azure Stream Analytics
- Scale Azure Stream Analytics jobs
- Azure Stream Analytics Query Language Reference
- Azure Stream Analytics Management REST API Reference