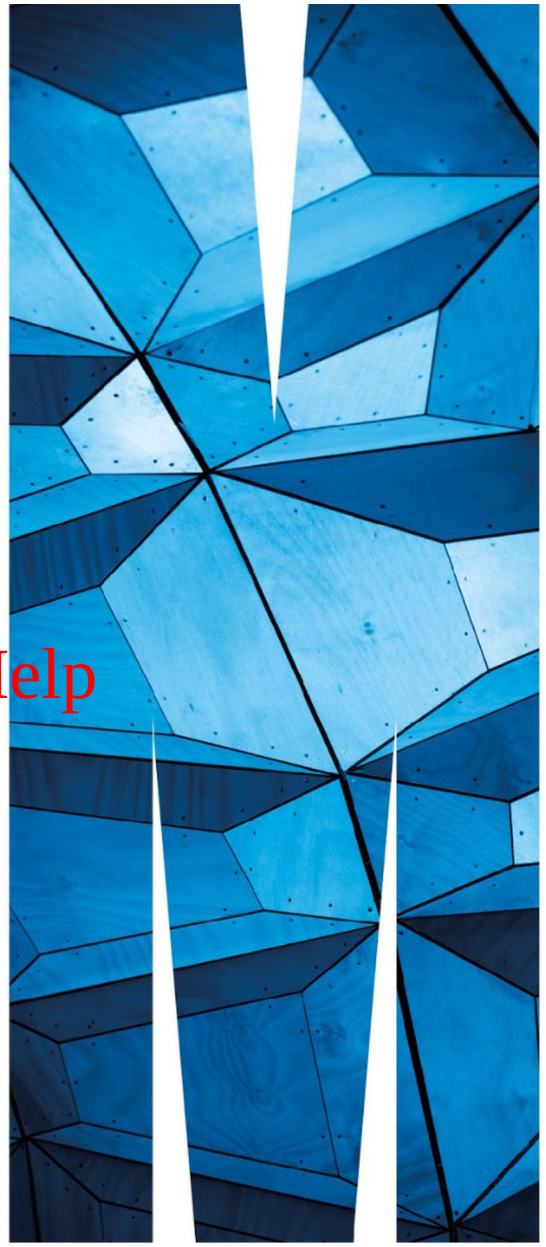
MONASH University

MONASH
INFORMATION
TECHNOLOGY



FIT5202 – Data Processing for Big Data

Stream Data Processing

Prajwol Sangat

Edited by Ting Chee Ming (7 May 2021)

GROUP OF EIGHT AUSTRALIA

Last Lecture

- Collaborative Filtering

This Week

- **Reminder:** Student Ev[illegible]Teaching and Units (SETU)
- Stream data processin[illegible]
- Streaming processing technology

Data Stream Applications

#trending

Track monitoring

#Stock market

#Security

Characteristics of the application

- Unbounded (infinite) data: keeps coming in without boundary
- Real time query at any given time
  - Get response quickly
  - Differ from batch query applied to entire table/data block
- Data comes continuously
  - Uniform rate (e.g., data comes every 1 second)
  - Bursty (e.g., data records come in one single reading)
- Interested in an "event" or trends across time.

Data stream

A data stream is a real-tim[illegible]us, ordered (implicitly by arrival time or explicitly by timestamp) se[illegible]tems.

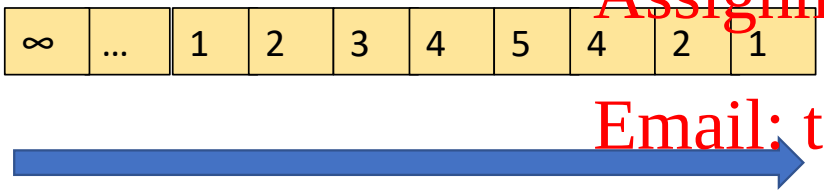
t_n t_0

Data Stream Query - Examples

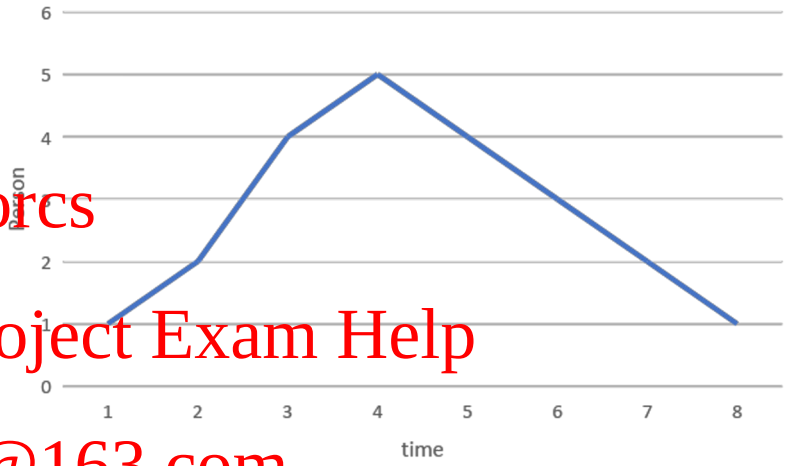
What is the total number of attende

What is the average number of atte
(at time tx)

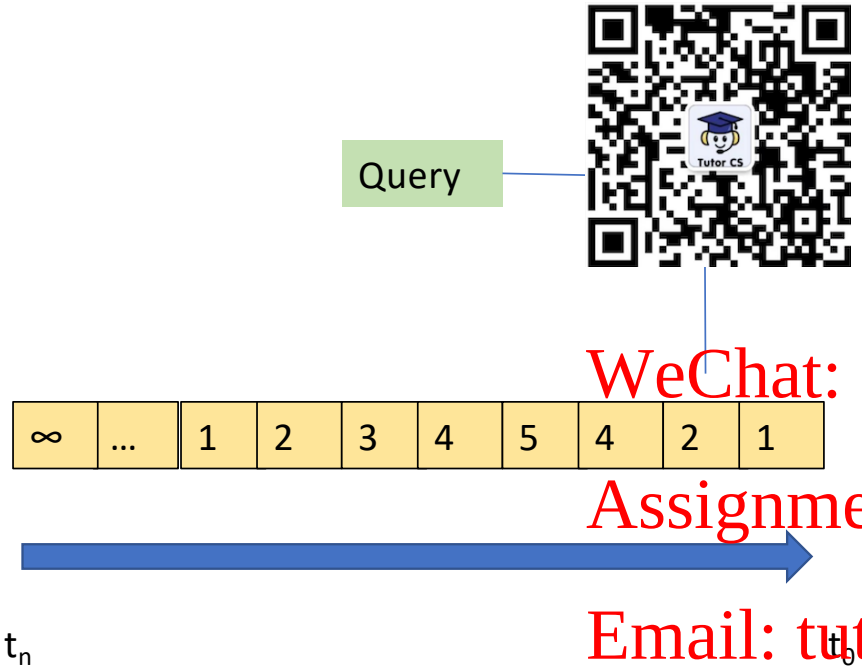
Will the next number be lower or higher than the current reading?



t_n t_0



What is the total number of attendees up to tx?



Assume memory can only keep 5 tuples.

Keep an accumulator for the sum. There is no need to keep the tuples.

Possible issues:

- What to do with incoming tuples if the incoming tuple arrival rate > processing time?

https://aseigneurin.github.io/2018/08/27/kafka-streams-processing-late-events.html

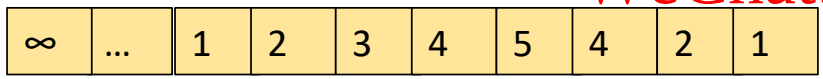
Data streams: Can't keep all incoming data because data is unbounded
Database: Have persistent storage to store all data

→ Rely on accumulator as a summary of the stream

What is the average number of attendees so far? (at time tx)

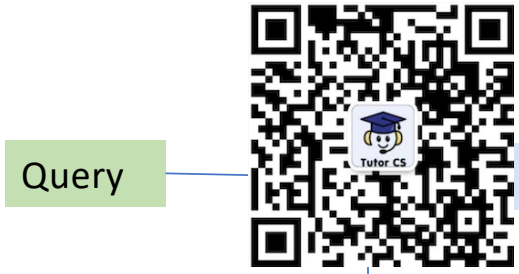
Query

Keep two accumulators for the sum and count. There is no need to keep the tuples.



t_n t_0

Will the next number be lower or higher than the current reading?



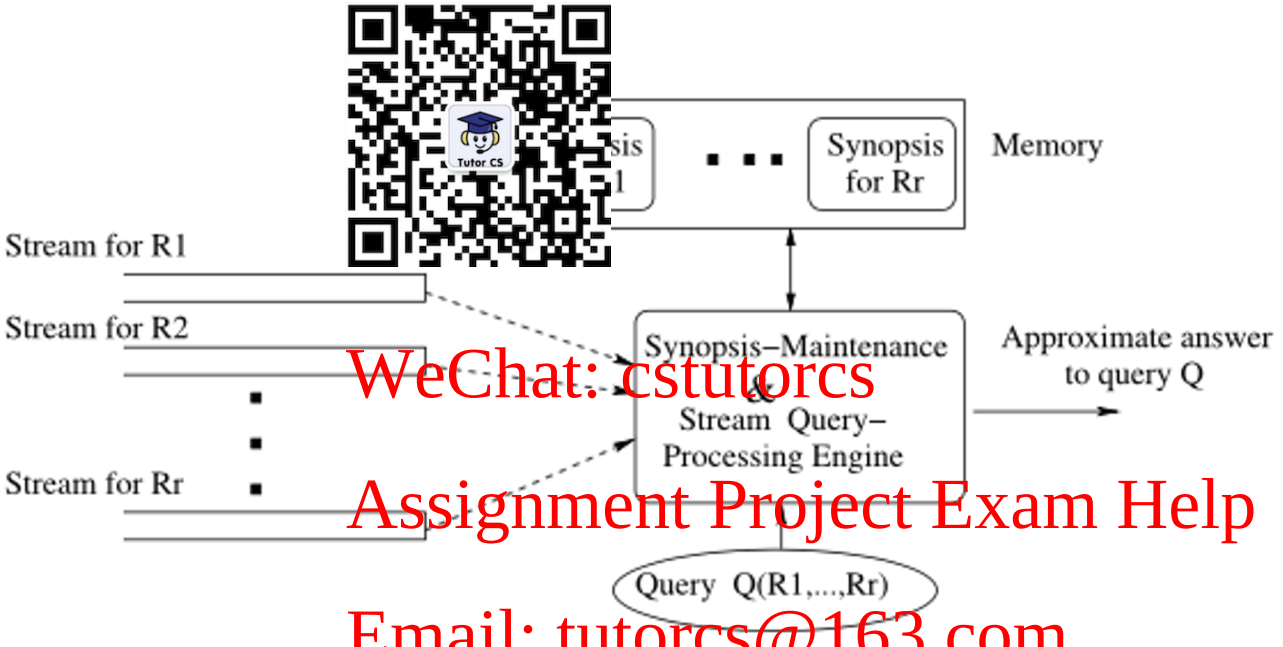
∞	…	1	2	3	4	5	4	2	1

t_n t_0

Prediction may need all/some previous tuples.
Memory can only keep 5 tuples.
What can we do?

Learn from previous data (create summary/model based on previous data)

Stream Processing System Architecture

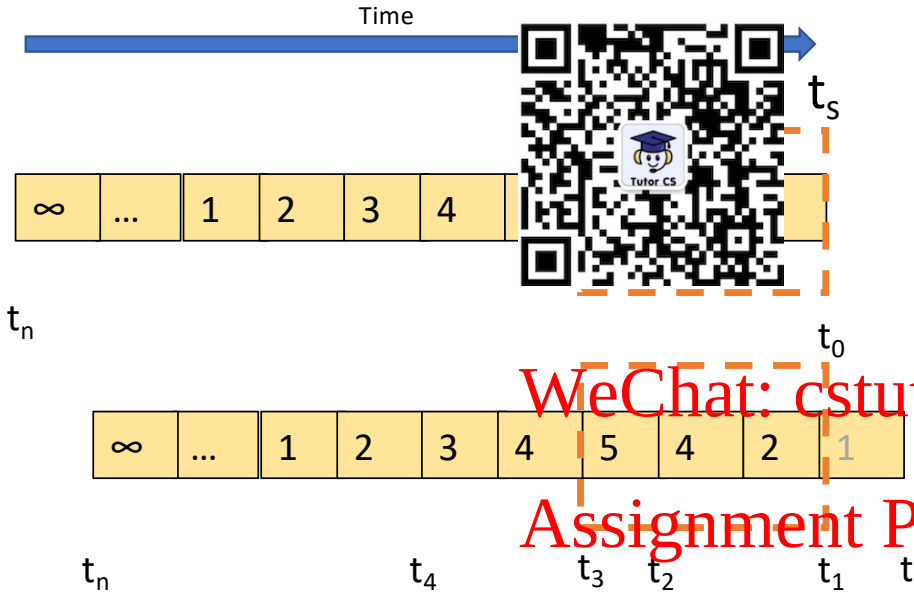


Garofalakis M., Gehrke J., Rastogi R. (2016) Data Stream Management: A Brave New World. In: Garofalakis M., Gehrke J., Rastogi R. (eds) Data Stream Management. Data-Centric Systems and Applications. Springer, Berlin, Heidelberg

Time Decay

- Is it possible to reduce th[illegible]ce of older tuples, without eliminating their influence, on the results [illegible]ysis?
- Popular approach: windo[illegible]
- Query processing is performed only on the tuples inside the window.
- Window types:
  - Time-based
  - Tuple-based (count-based)

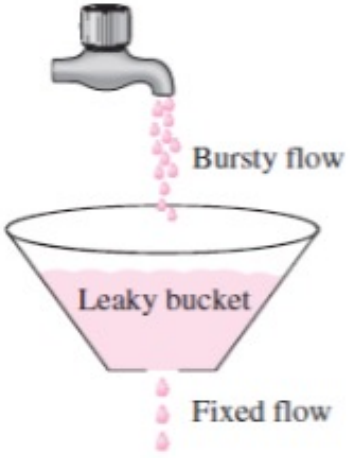
Stream Window – Time Based Window

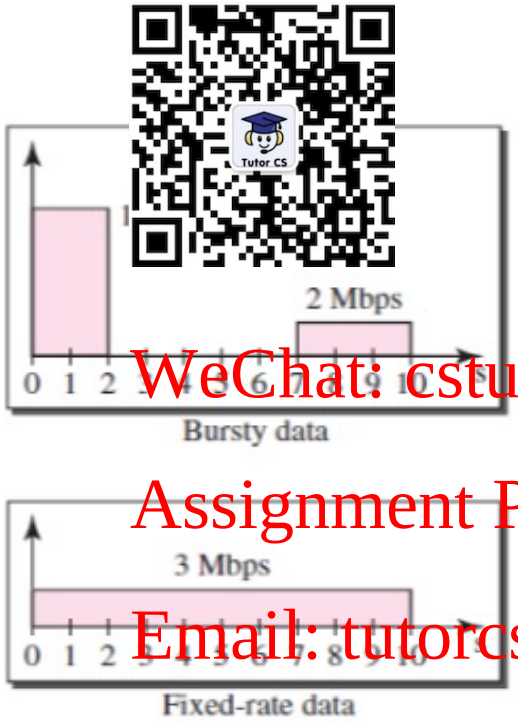


Window size 2 seconds
Slides 1 second.

- Size of window is specified by fixed time duration
- Number of tuples in windows may vary depending on how many tuples are available while processing
- It may be high due to bursty arrival of data due to delay in data transfer
- Tuples in the windows will be processed, no matter how many tuples in the windows

Bursty data

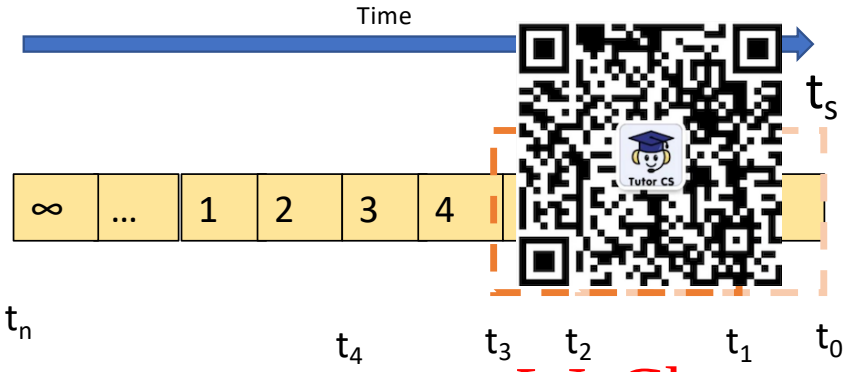




Data rate change suddenly in short time

Image source: https://www.quora.com/What-is-the-difference-between-token-bucket-and-leaky-bucket-algorithms

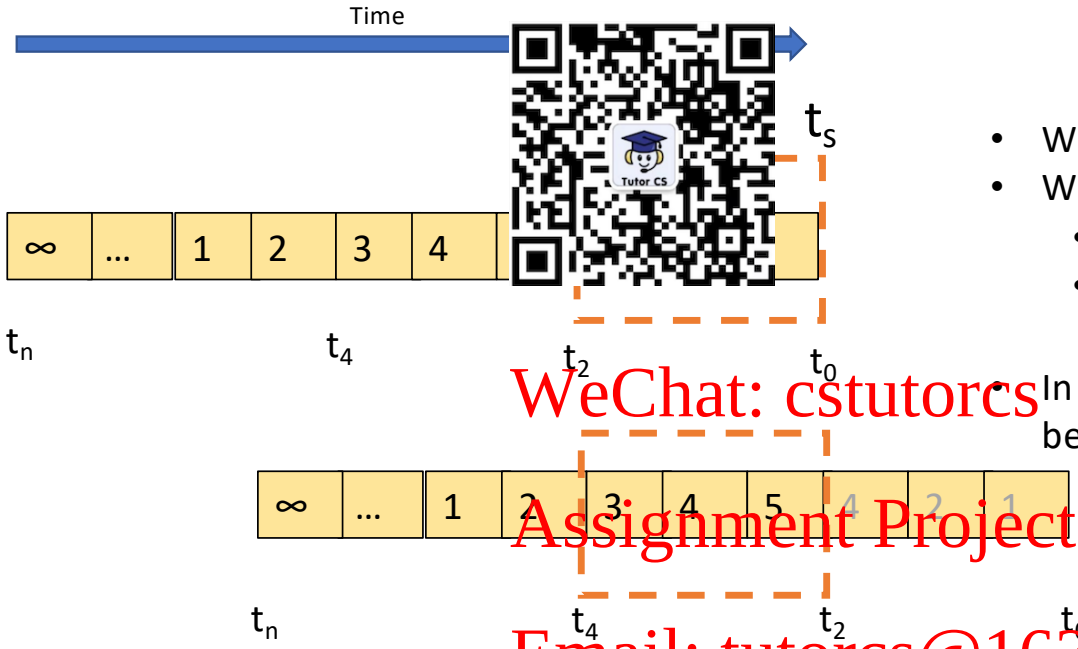
Stream Window – Time Based Window



Window size 2 seconds
Slides 1 second.

Overlapping Sliding Window

Stream Window – Time Based Window

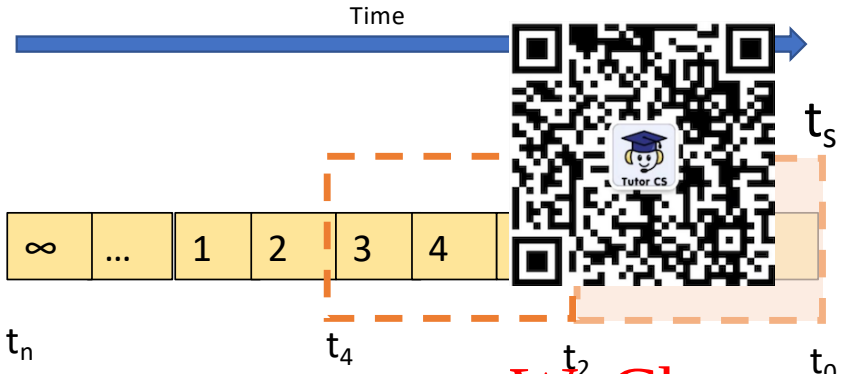


- Window size is based on time, eg 2 seconds
- Window can be advanced by:
  - t_e-t_s (window size)
  - Duration less than the window size (sliding window).
- In uniform data rate, the number of tuples will be the same for each window.

Non-Overlapping Sliding Window

- Slide of window is equal to size of window
- Two windows are disjoint
- No influence of data in previous window on current window

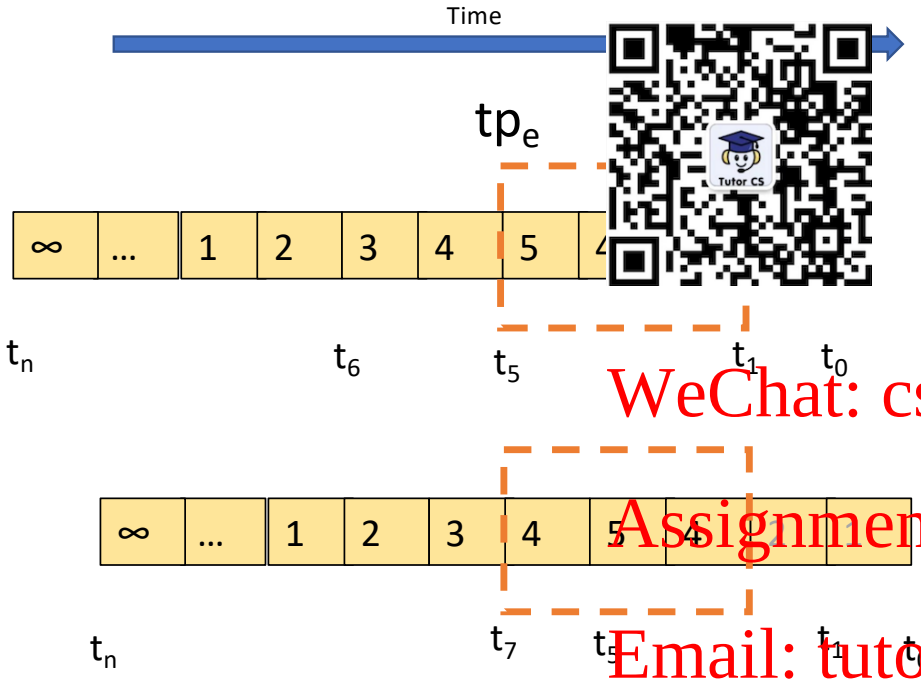
Stream Window – Time Based Window



Non-Overlapping Sliding Window

- Window size is based on time, eg 2 seconds
- Window can be advanced by:
  - t_e-t_s (window size)
  - Duration less than the window size (sliding window).
- In uniform data rate, the number of tuples will be the same for each window.

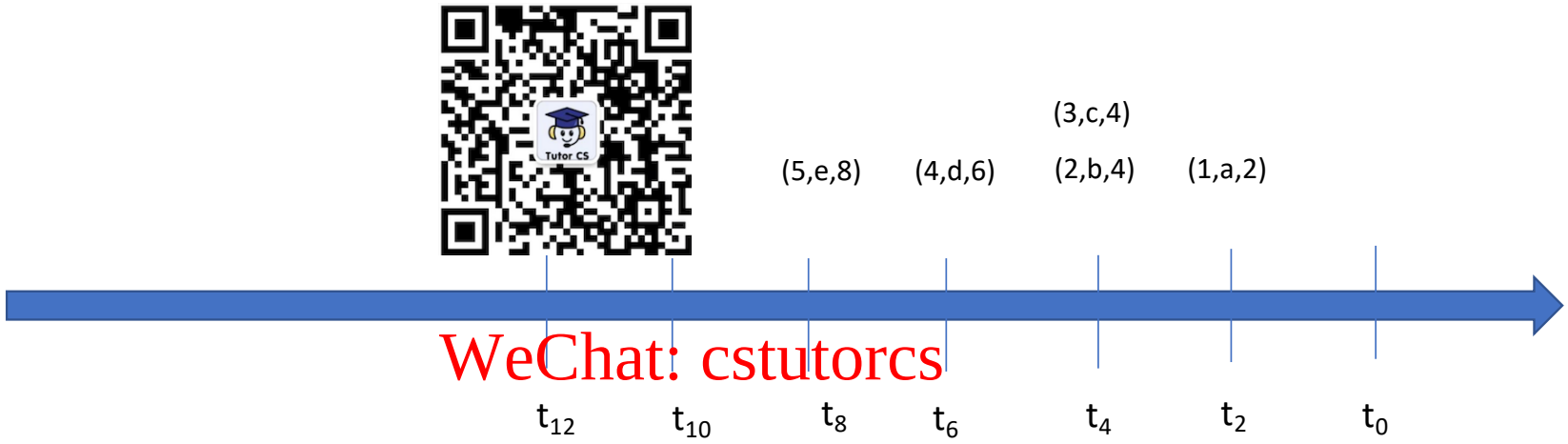
Stream Window – Tuple Based Window



Window size is 3 tuples
Slide window by 1 tuple

- Size of window is specified by number of tuples in the window
- The window always has a fixed number of tuples
- We count the number of tuples and create windows. If the window has required numbers of tuples, then that window will be processed
- The time duration of windows may vary

Example of Burst Data Arrival



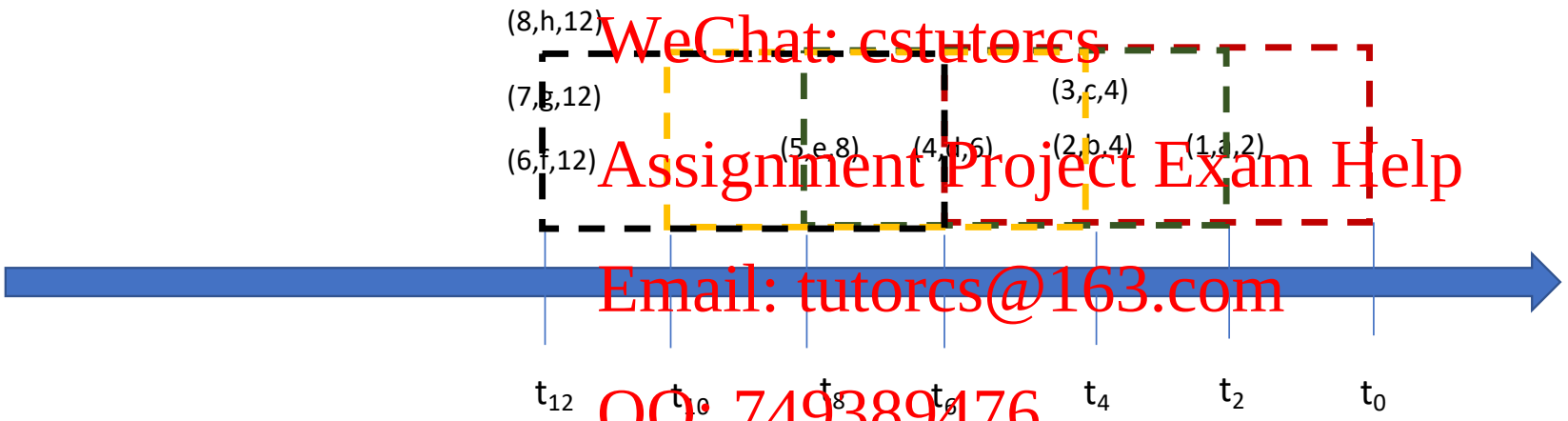
(eventTime, value, processingTime)

Event Time: The time stamp when the data is generated
Processing Time: the time stamp when the data arrived at the processing.

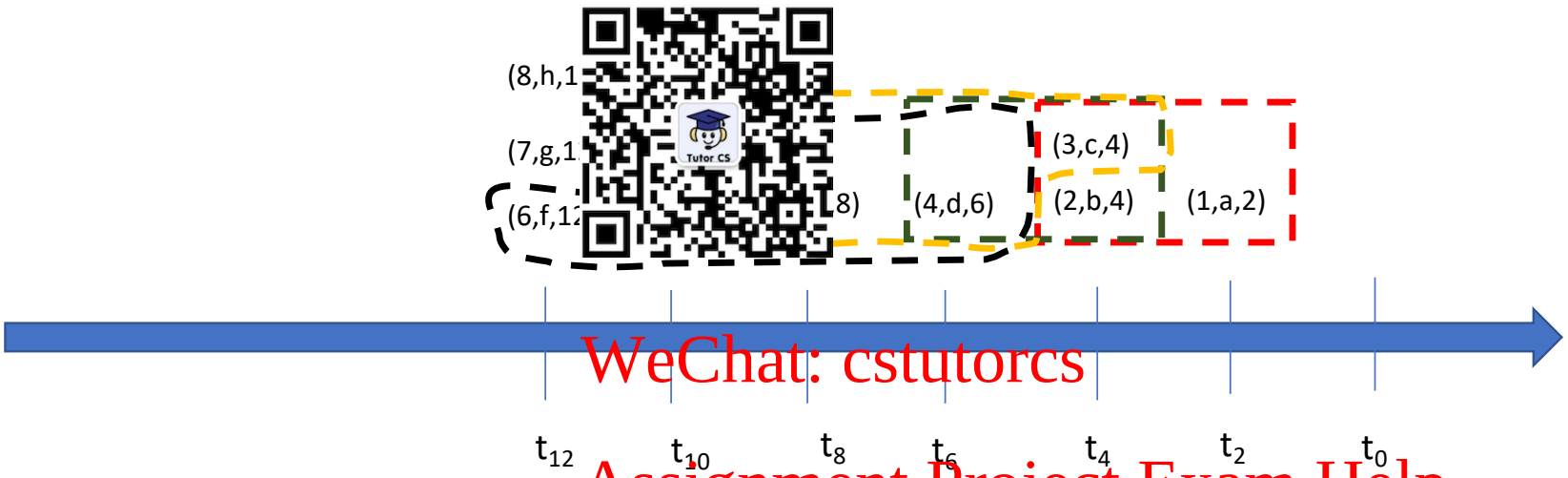
Time Based Window

- Processing time window,
- Window size 6 units, slid[illegible]s

W_1\{(1,a,2),(2,b,4),(3,c,4),(4,d,6)\}
W_2\{(1,a,2),(2,b,4),(3,c,4),(4,d,6),(5,e,8)\}
W_3\{(2,b,4),(3,c,4),(4,d,6),(5,e,8)\}
W_4\{(4,d,6),(5,e,8),(6,f,12),(7,g,12),(8,h,12)\}



Tuple Based Window



Window size is 3 tuples.
Slide by 1 tuple

W_1 {(1,a,2),(2,b,4),(3,c,4)}
W_2 {(2,b,4),(3,c,4),(4,d,6)}
W_3 {(3,c,4),(4,d,6),(5,e,8)}
W_4 {(4,d,6),(5,e,8),(6,f,12)}
}

→ Need to drop some of the tuple with processing timestamp 12

Practice Question

A data stream application processe[illegible]ta from a data source. The data source emits a single tuple per second (constant rate). T[illegible]ceived by the data stream application in the format of *(eventTime,value)*. Once it arrives [illegible]he processing timestamp is added to the tuple. The final tuple format after the addition of th[illegible]mestamp is *(eventTimestamp, value, processingTimestamp)*.

Examples of tuples are as followed:

{1,a,1}
{2,b,4}
{3,c,4}
{5,e,6}
{6,f,8}
{7,g,8}
{8,h,9}
{9,i,10}
{10,j,11}

Based on the above data and scenario answer next exercise.

Exercise 2

- Let's say we follow a ti[illegible]windowing mechanism. The window size is three seconds a[illegible]e is two seconds. Assume the timing start at processing timestamp 1. What will be the content of the third window?

- A. {1,a,1}, {2,b,4}, {3,c,4}
- B. {2,b,4}, {3,c,4}, {5,e,6}
- C. {5,e,6}, {6,f,8}, {7,g,8}
- D. {3,c,4}, {5,e,6}, {6,f,8}

Event vs Processing time

- Event time: the time whe is produced by the source.
- Processing time: the time a is arrived at the processing server.
- In ideal situation, event t essing time.
- In real world event time is earlier than the processing time due to network delay.
- The delay can be uniformed (ideal situation) or non-uniform (most of real network situation).
- Data may arrive in “burst” (bursty network).

Database vs Stream Processing

- Bounded data (assess data).
- Relatively static data
- Complex, ad-hoc query
- Possible to backtrack during processing
- Exact answer to a query
- Tuples arrival rate is low

- Unbounded data (data keeps coming without end).
- Dynamic data.
- Simple, continuous query
- No backtracking, single pass operation.
- Approximate answer to a query
- Tuples arrival rate is high

Sliding Window: Produce an approximate answer to a data stream query is to evaluate the query not over the entire past history of data streams, but rather only over sliding windows of recent data from the streams.

MONASH University

Stream Processing Technology

Streaming Data Processing

Prajwol Sangat

Updated by Ting Chee Ming

Streaming Platforms

Real-Time Streaming Architecture

Kafka-Spark Streaming Integration

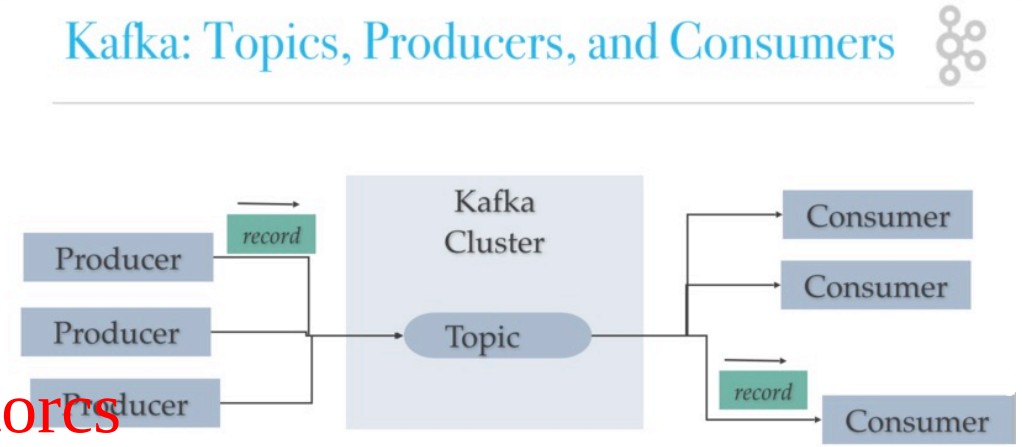
Kafka Producer
- Send data from p sing side
- publish data into

Data stream
- Examples: web host, server, sensors, IOT devices which generate data

Consumer & Spark Streaming
- Listen/subscribe to topic and consumes the data

What is Apache Kafka?

- Publish-subscribe messaging syst
- Scalable, Fault-tolerant
- Enables distributed applications
- Powers web-scale Internet compa
  LinkedIn, NetFlix, AirBnB, and many others.



https://dzone.com/articles/kafka-architecture

- ❑ Producer publish streams of data records into topics
- ❑ Consumers subscribe to the topics and process the stream of records

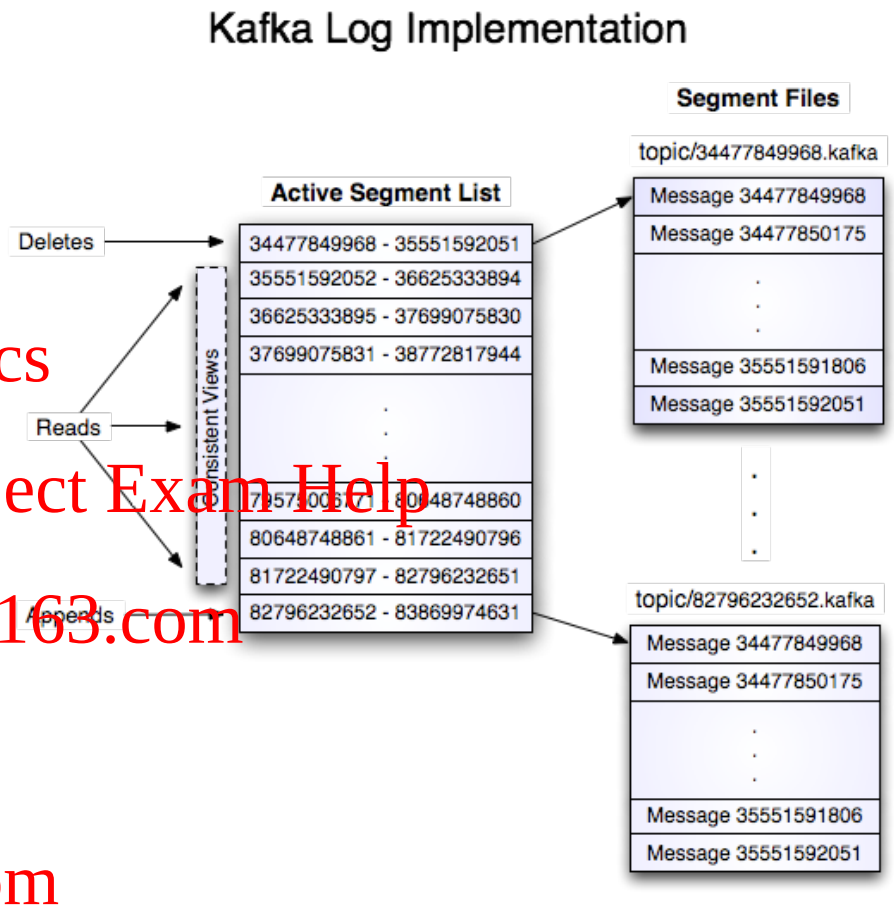
- ❑ Data in Kafka is stored in topic
  - Topic is category/feed name where records are stored
  - A topic is associated with a log – data structure on disk
  - Each topic is indexed and stored with timestamp

- ❑ Kafka is run as a cluster comprised of one or more servers (called brokers)

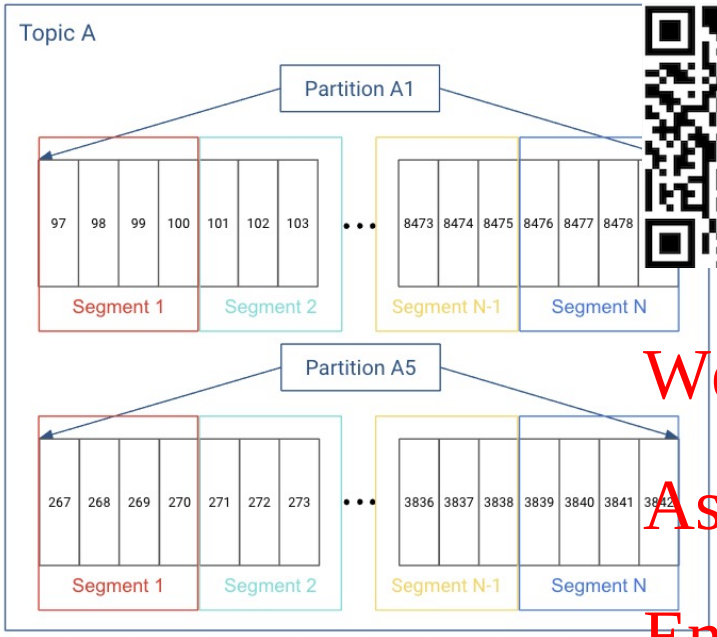
  - A broker receive messages from producers and stores them on disk keyed by unique offset.
  - A broker allows consumers to fetch messages by topic, partition and offset

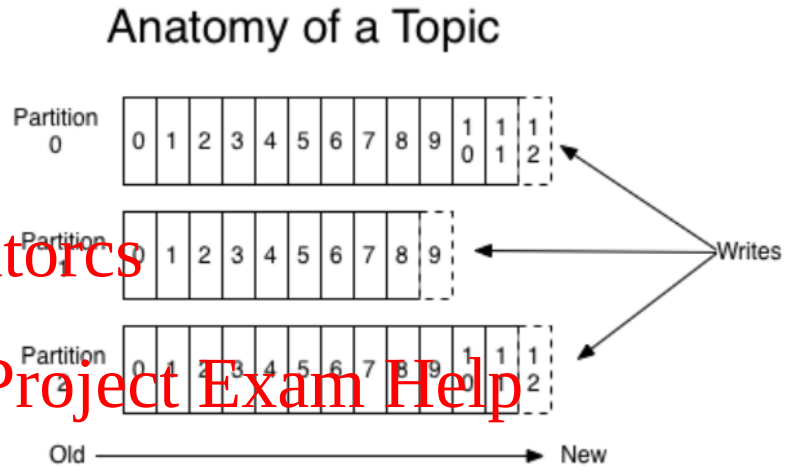
How does Kafka Work?

- Topics represent commit log data ... ed on disk
  - Topics are divided into partitions ... n is further divided into segments
  - Each segment has a log file to store the actual message
  - Log – time-ordered, sequence of messages that is continually appended (each log entry can be array of bytes)
  - Partitions are replicated and distributed over servers in Kafka cluster (brokers) for fault tolerance (when a server in the cluster fails so messages remain available)
  - Messages stay around for a configurable period of time (i.e., 7 days, 30 days, etc.).
    - Can recover lost messages during time out or lost connection



Kafka Storage Structure





Messages are written to it in an append-only fashion

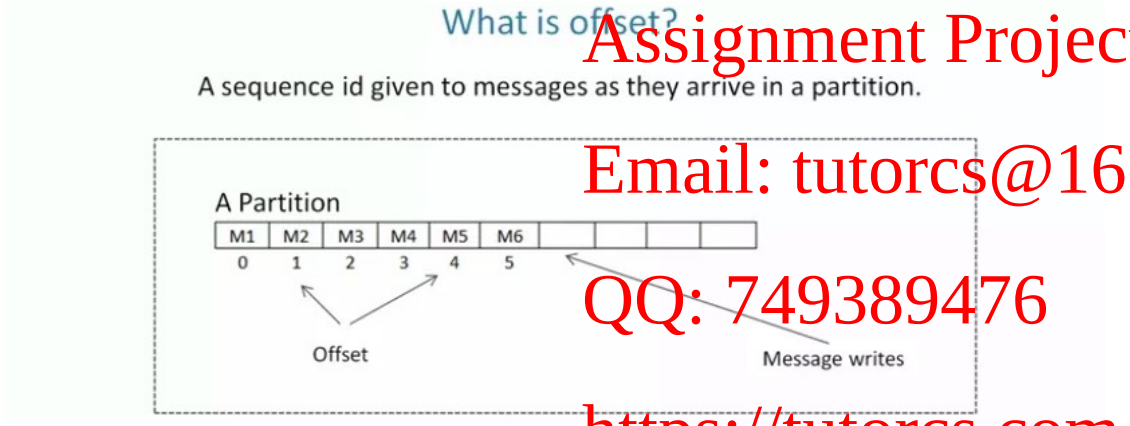
- Topic is logical grouping
- Partition is actual unit of data storage

- Every piece of data stored in segment file is a message
- Each message in partition is uniquely identified by an ID called offset

Image source: https://docs.cloudera.com/runtime/7.2.8/kafka-overview/topics/kafka-overview-logs-and-log-segments.html
https://kafka.apache.org/081/documentation.html

What Kafka doesn't do?

- Kafka does not have in[illegible]essage IDs. Messages are simply addressed by their offs[illegible]g.
- Kafka also does not tra[illegible]sumers that a topic has or who has consumed what messages.
- There are no deletes. Kafka keeps all parts of the log for the specified time.



https://shubhamagtech.home.blog/2019/07/31/kafka-offset-management/

Kafka and big data at web-scale companies

- Big Data ingesti… Netflix

https://www.slideshare.net/SparkSummit/spark-and-spark-streaming-at-netfix-sedakar-daxini

Kafka and big data at web-scale companies

- **BP OIL USE CA**

https://www.linkedin.com/pulse/real-time-streaming-data-pipelines-apache-kafka-spark-steven-murhula/

Kafka and big data at web-scale companies

- **Transform, Store, Explore Healthcare Dataset**

https://mapr.com/blog/streaming-data-pipeline-transform-store-explore-healthcare-dataset-mapr-db/

Should you use Apache Kafka?

- Kafka fits a class of problem that a lot of web-scale companies and enterprises have, but just as the traditional message broker is not a one size fits all, neither is Kafka.
- If you're looking to build a set of resilient data services and applications, Kafka can serve as the source of truth by collecting and keeping all of the "facts" or "events" for a system.

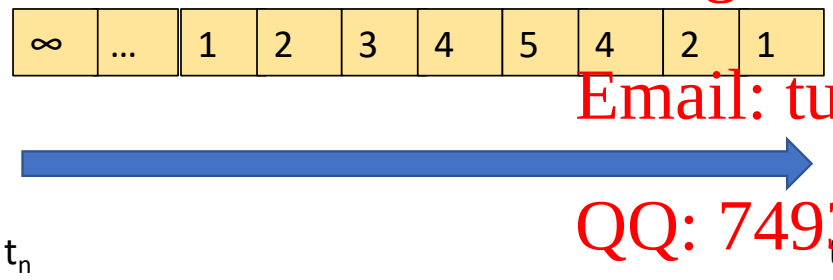
Real-Time Streaming Architecture

DEMO

- What is the total nu[illegible]attendees up to tx?

Assume memory can only keep 5 tuples.

Keep an accumulator for the sum. There is no need to keep the tuples.



Thank You

Questions?