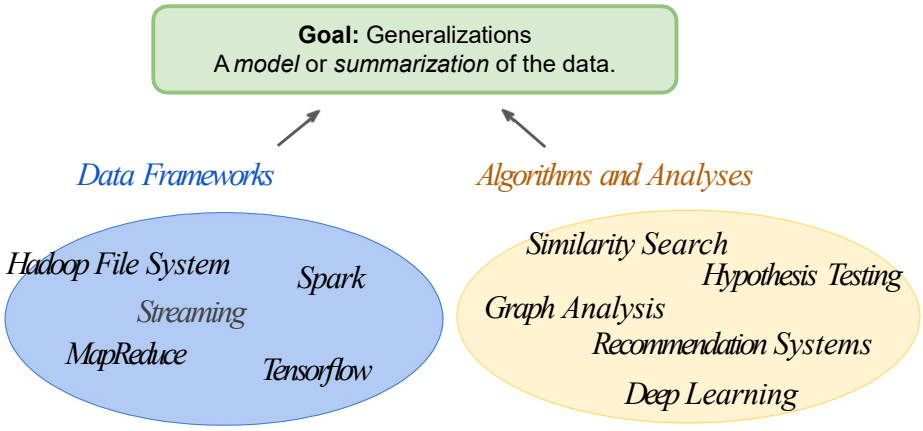
Streaming Algorithms: Data without a disk

1

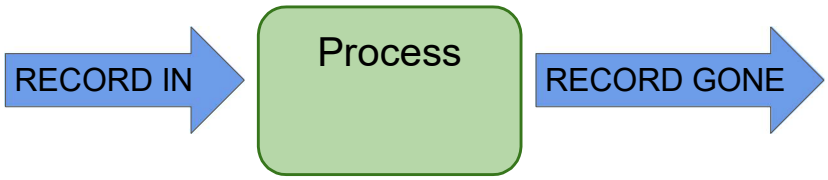
Big Data Analytics, The Class



2

What is Streaming?

Broadly:



3

Why Streaming?

(1) Direct: Often, data …

- … cannot be stored (too big, privacy concerns)
- … are not practical to access repeatedly (reading is too long)
- … are rapidly arriving (need rapidly updated "results")

4

Why Streaming?

(1) Direct: Often, data …

- … cannot be stored (too big, privacy concerns)
- … are not practical to access repeatedly (reading is too long)
- … are rapidly arriving (need rapidly updated "results")

Examples: Google search queries
Satellite imagery
data Text Messages, Status updates
Click Streams

5

Why Streaming?

(1) Direct: Often, data …

- … cannot be stored (too big, privacy concerns)
- … are not practical to access repeatedly (reading is too long)
- … are rapidly arriving (need rapidly updated "results")

(2) Indirect: The constraints for streaming data force one to solutions that are often efficient even when storing data.

Streaming Approx Random Sample

6

Why Streaming?

Often translates into *O(N)* or strictly *N* algorithms.

(2) **Indirect:** The constraints for streaming data force one to solutions that are often efficient even when storing data. *Streaming Approx Random Sample*

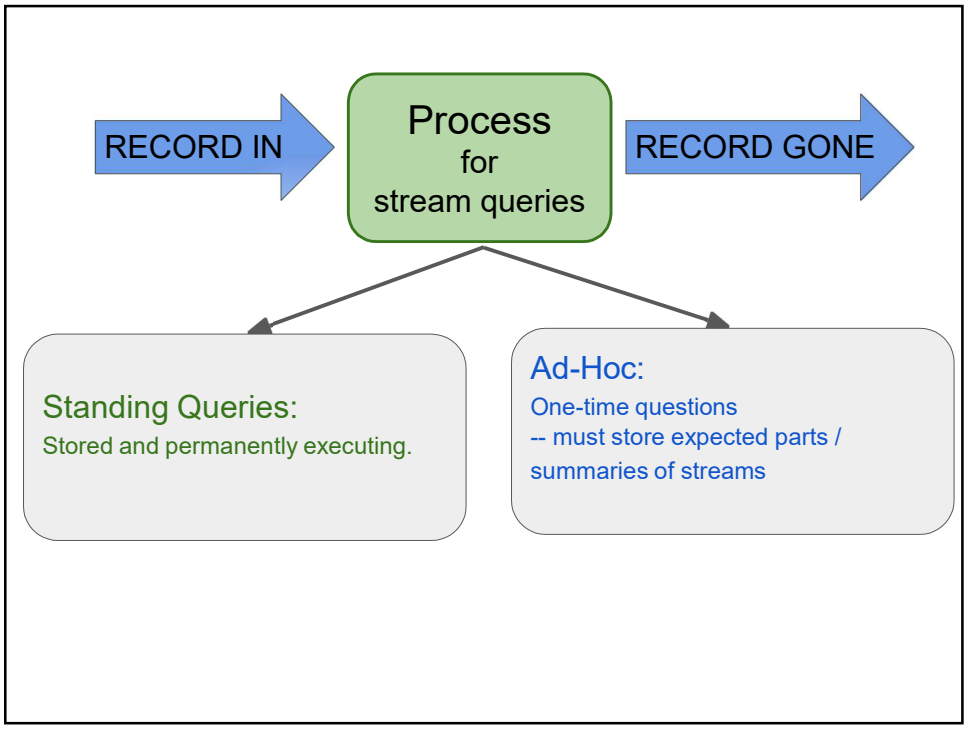
Distributed IO (MapReduce, Spark)

7

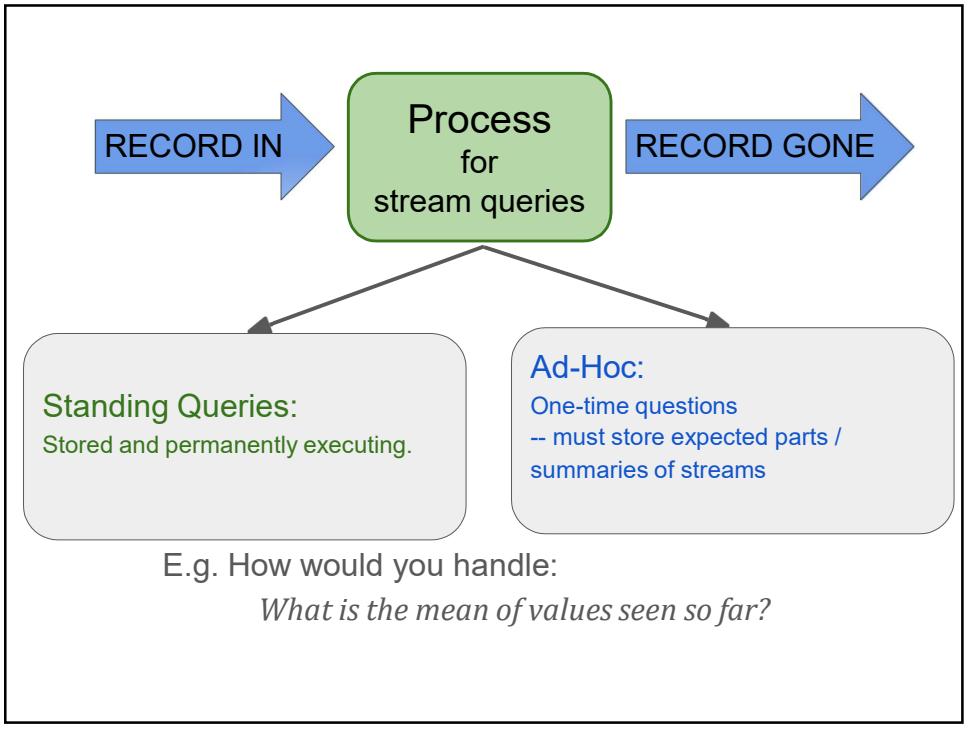
Streaming Topics

- General Stream Processing Model
- Sampling
- Counting Distinct Elements
- Filtering data according to a criteria

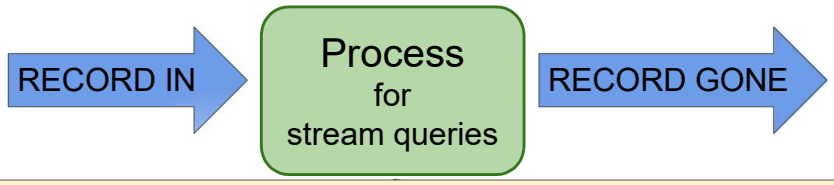
8


RECORD IN
Process
for
stream queries
RECORD GONE
Standing Queries:
Stored and permanently executing.
Ad-Hoc:
One-time questions
-- must store expected parts /
summaries of streams

9


RECORD IN
Process
for
stream queries
RECORD GONE
Standing Queries:
Stored and permanently executing.
Ad-Hoc:
One-time questions
-- must store expected parts /
summaries of streams
E.g. How would you handle:
What is the mean of values seen so far?

10



Important difference from typical database management:

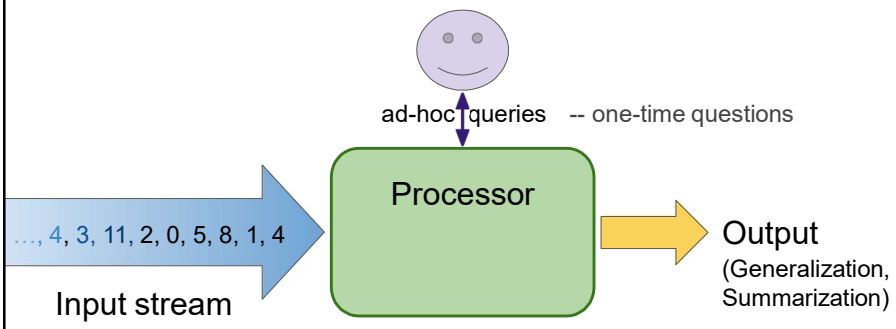
- Input is not controlled by system staff.
- Input timing/rate is often unknown, controlled by users.

E.g. How would you handle:

What is the mean of values seen so far?

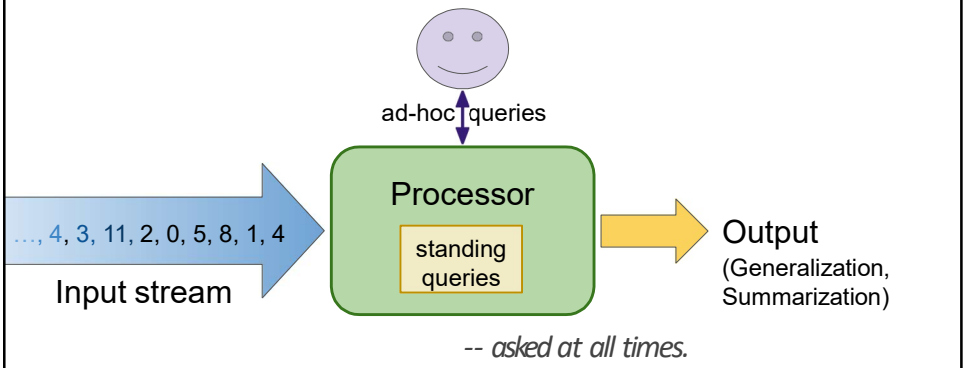
11

General Stream Processing Model



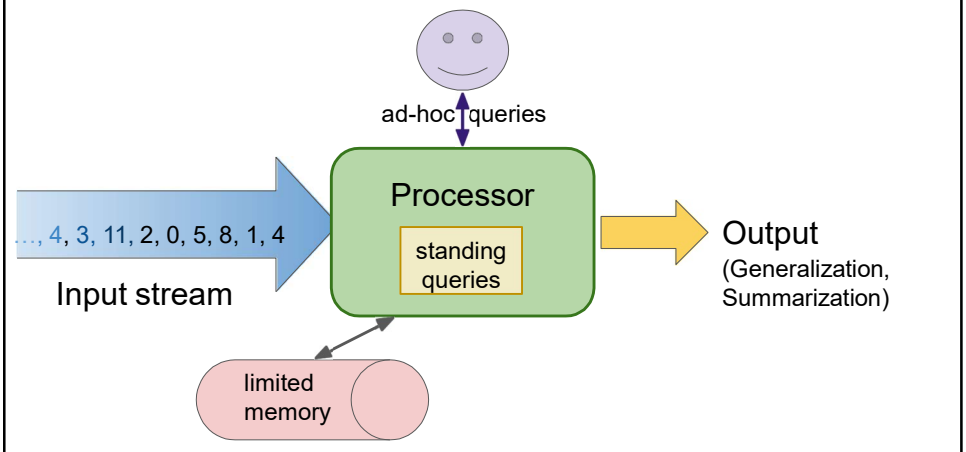
12

General Stream Processing Model



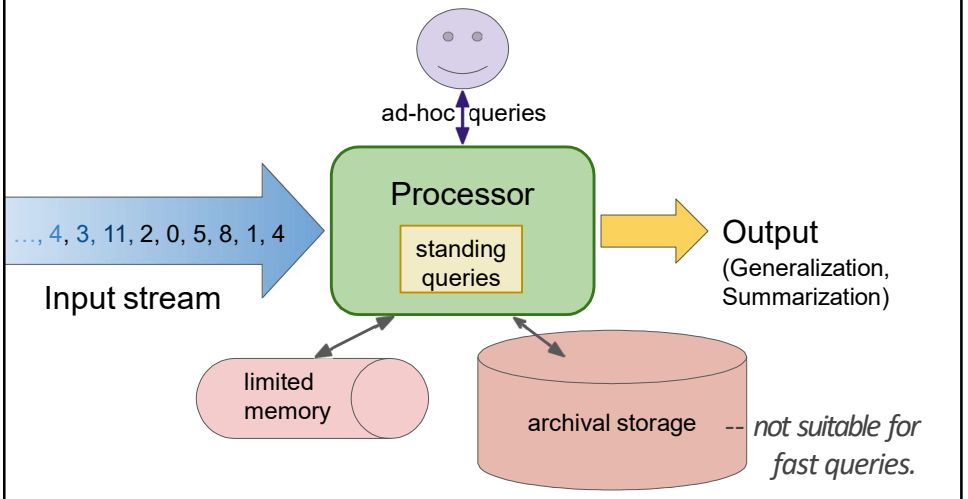
13

General Stream Processing Model



14

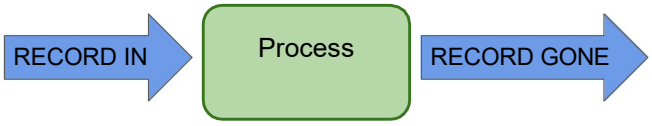
General Stream Processing Model



15

Sampling

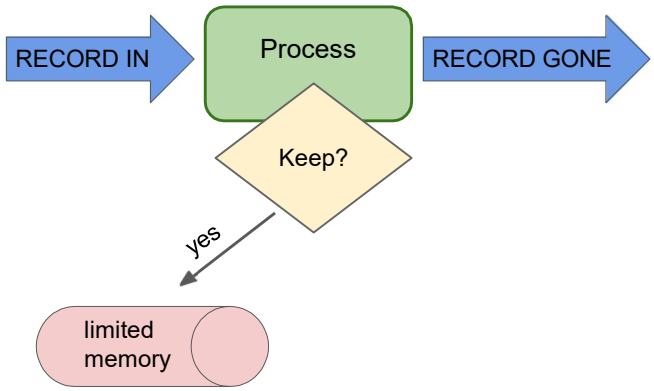
Create a random sample for statistical analysis.



16

Sampling

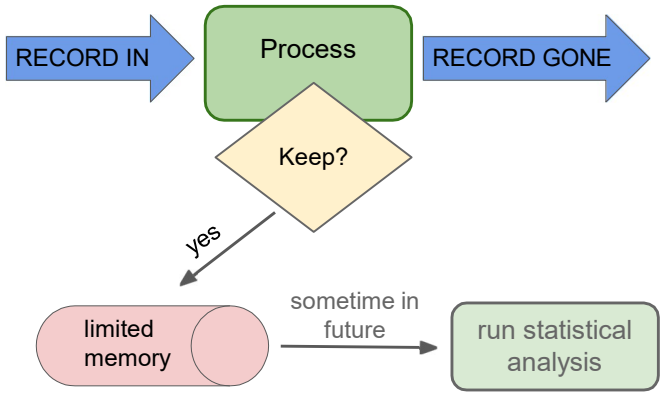
Create a random sample for statistical analysis.



17

Sampling

Create a random sample for statistical analysis.



18

Sampling: 2 Versions

Create a random sample for statistical analysis.

1. **Simple Sampling:** Individual records are what you wish to sample.

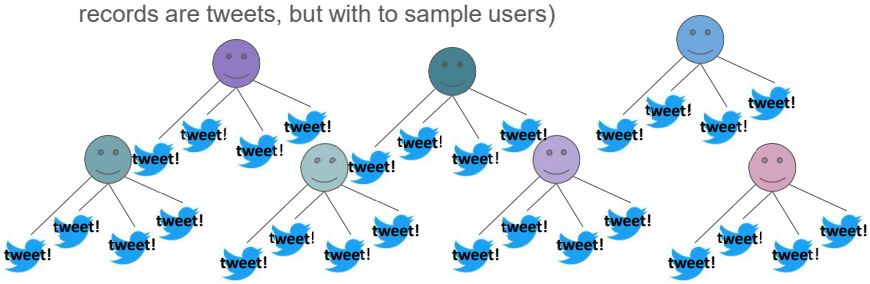
19

Sampling: 2 Versions

Create a random sample for statistical analysis.

1. **Simple Sampling:** Individual records are what you wish to sample.

2. **Hierarchical Sampling:** Sample an attribute of a record. (e.g. records are tweets, but with to sample users)



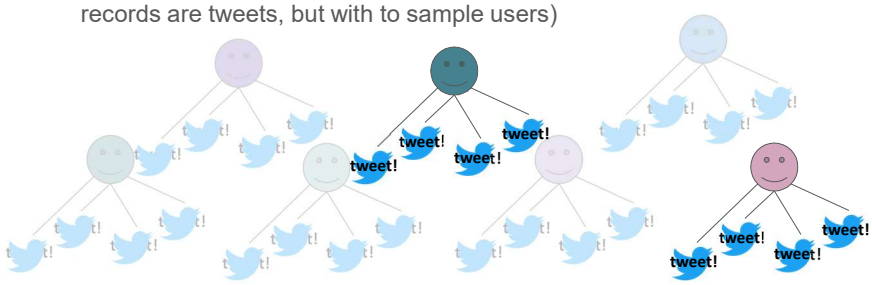
20

Sampling: 2 Versions

Create a random sample for statistical analysis.

1. **Simple Sampling:** Individual records are what you wish to sample.

2. **Hierarchical Sampling:** Sample an attribute of a record. (e.g. records are tweets, but with to sample users)



21

Sampling

Create a random sample for statistical analysis.

1. **Simple Sampling:** Individual records are what you wish to sample.

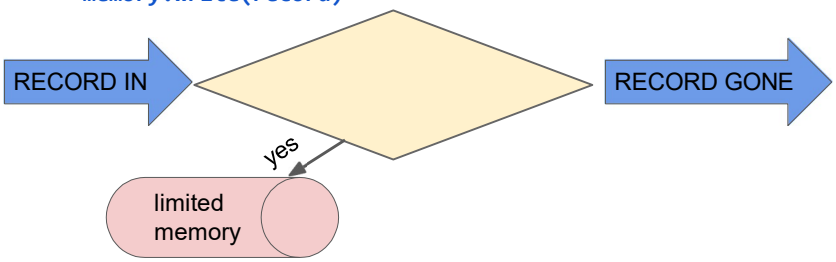
22

Sampling

Create a random sample for statistical analysis.

1. **Simple Sampling:** Individual records are what you wish to sample.

```
record = stream.next()
if ?: #keep: e.g., true 5% of the time
    memory.write(record)
```



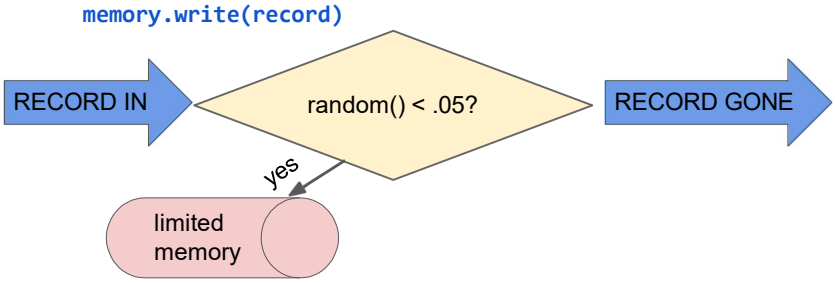
23

Sampling

Create a random sample for statistical analysis.

1. **Simple Sampling:** Individual records are what you wish to sample.

```
record = stream.next()
if random() <= .05: #keep: true 5% of the time
    memory.write(record)
```



24

Sampling

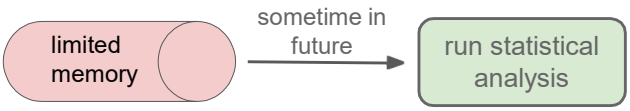
Create a random sample for statistical analysis.

1. **Simple Sampling:** Individual records are what you wish to sample.

```
record = stream.next()
if random() <= .05: #keep: true 5% of the time
    memory.write(record)
```

Problem: records/rows often are not units-of-analysis for statistical analyses

E.g. user_ids for searches, tweets; location_ids for satellite images



25

Sampling

2. **Hierarchical Sampling:** Sample an attribute of a record.

(e.g. records are tweets, but with to sample users)

```
record = stream.next()
if random() <= .05: #keep: true 5% of the time
    memory.write(record)
```

Solution: ?

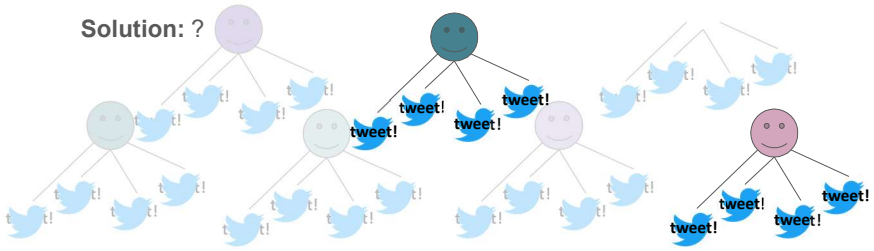
26

Sampling

2. **Hierarchical Sampling:** Sample an attribute of a record.
   (e.g. records are tweets, but with to sample users)

```
record = stream.next()
if ??: #keep
    memory.write(record)
```

Solution: ?



27

Sampling

2. **Hierarchical Sampling:** Sample an attribute of a record.
   (e.g. records are tweets, but with to sample users)

```
record = stream.next()
if ??: #keep:
    memory.write(record)
```

Solution: instead of checking random digit; hash the attribute being sampled.

– streaming: only need to store hash functions; may be part of standing query

28

Sampling

2. **Hierarchical Sampling:** Sample an attribute of a record.
   (e.g. records are tweets, but with to sample users)

```
record = stream.next()
if hash(record['user_id']) == 1: #keep
    memory.write(record)
```

Solution: instead of checking random digit; hash the attribute being sampled.
– streaming: only need to store hash functions; may be part of standing query

How many buckets to hash into?

29

Streaming Topics

- General Stream Processing Model
- Sampling
- Counting Distinct Elements
- Filtering data according to a criteria

30

Counting Moments

Moments:

- Suppose m_i is the count of distinct element i in the data
- The kth moment of the stream is $\sum_{i \in \text{Set}} m_i^k$

31

Counting Moments

Moments:

- Suppose m_i is the count of distinct element i in the data
- The kth moment of the stream is $\sum_{i \in \text{Set}} m_i^k$

- 0th moment: count of distinct elements
- 1st moment: length of stream
- 2nd moment: sum of squares
  (measures *uneveness;* related to variance)

32