



Extending Structured Streaming Made Easy with Algebra

Erik Erlandson
Red Hat, Inc.

eje@redhat.com
@manyangled

#SAISDev2

In The Beginning



In The Beginning



1 Data Set

In The Beginning

1 File



1 Data Set

In The Beginning

1 File



1 Data Set

1 Machine

Sum

2

3

5

Sum

$s = 0$



Sum



2
3
5

$$s = s + 2 \quad (2)$$

Sum

2
3
5

$s = s + 3$ (5)

Sum



2
3
5

$$s = s + 5 \quad (10)$$

Commodity Scale-Out

2003

2004

Google FS
MapReduce

Commodity Scale-Out

2003

Google FS

2004

MapReduce

2006

Hadoop & HDFS

Commodity Scale-Out

2003

Google FS

2004

MapReduce

2006

Hadoop & HDFS

2009

Spark & RDD

Commodity Scale-Out

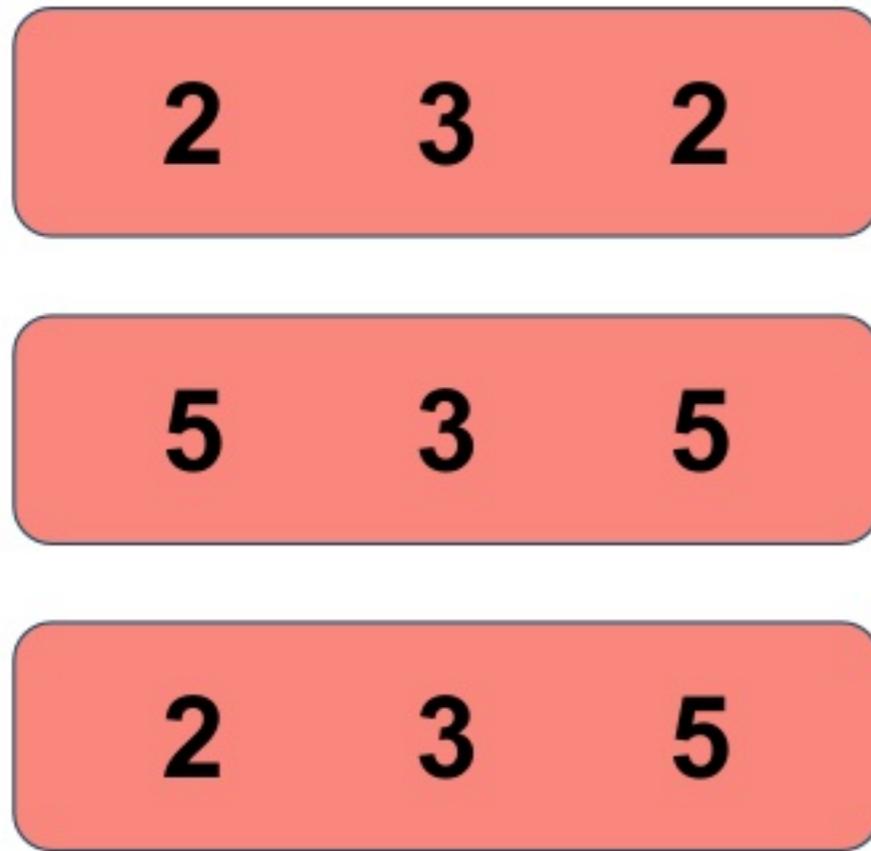
2003	Google FS
2004	MapReduce
2006	Hadoop & HDFS
2009	Spark & RDD
2015	DataFrame
2016	Structured Streaming

Our Scale-Out World

logical

2
3
2
5
3
5
2
3
5

Our Scale-Out World

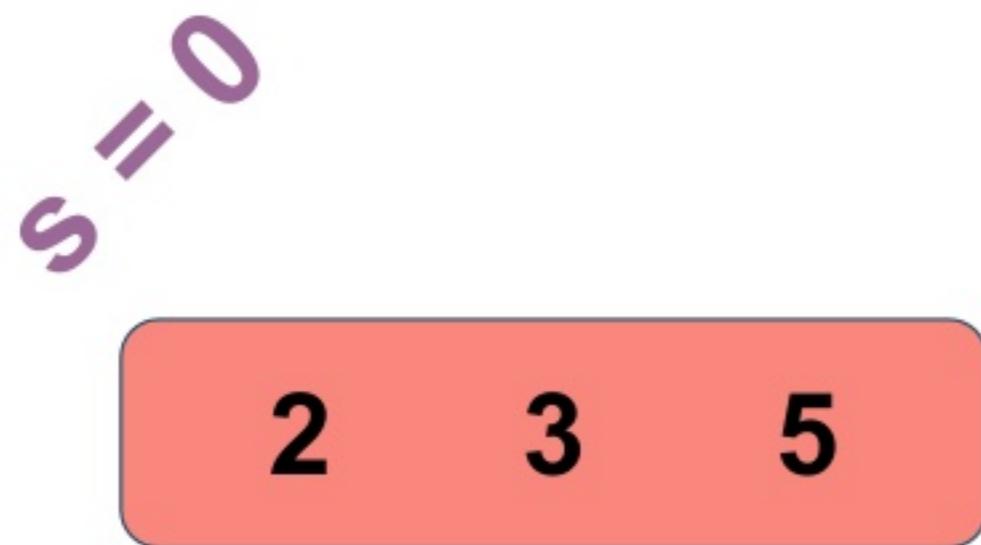


physical

logical



Scale-Out Sum

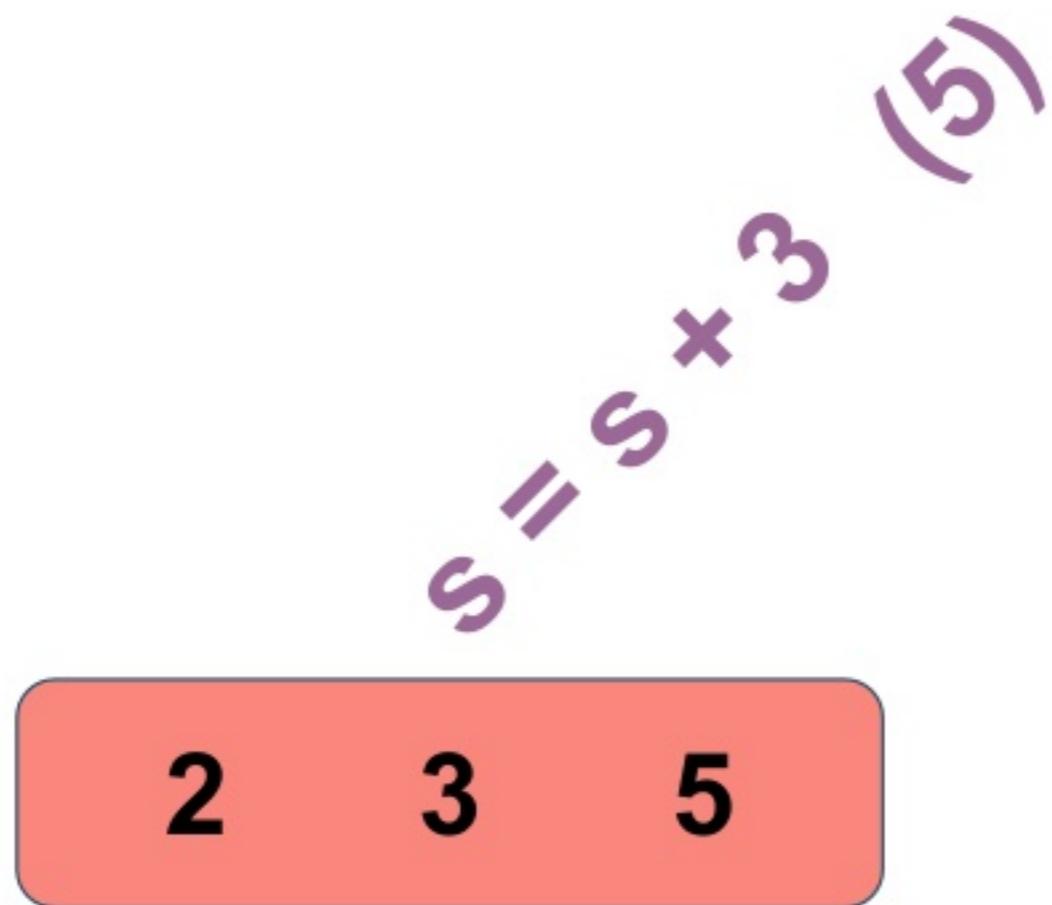


Scale-Out Sum

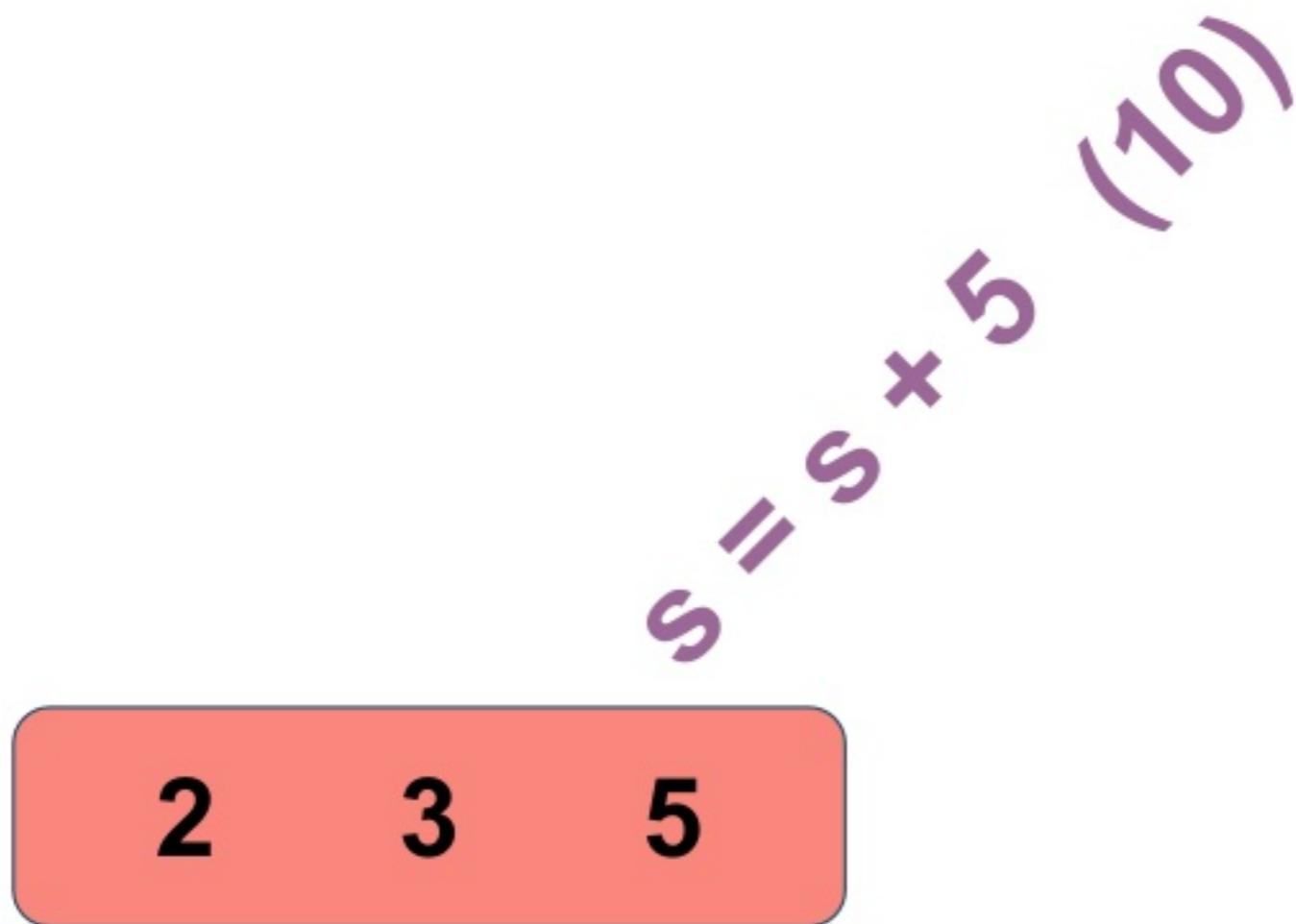
$$s = s \times 2^{(2)}$$

2 3 5

Scale-Out Sum



Scale-Out Sum



Scale-Out Sum

2 3 5 10

Scale-Out Sum

5 3 5 13

2 3 5 10

Scale-Out Sum

2 3 2 7

5 3 5 13

2 3 5 10

Scale-Out Sum

2 3 2

5 3 5

$$13 + 7 = 20$$

2 3 5

10

Scale-Out Sum

2 3 2

5 3 5

2 3 5

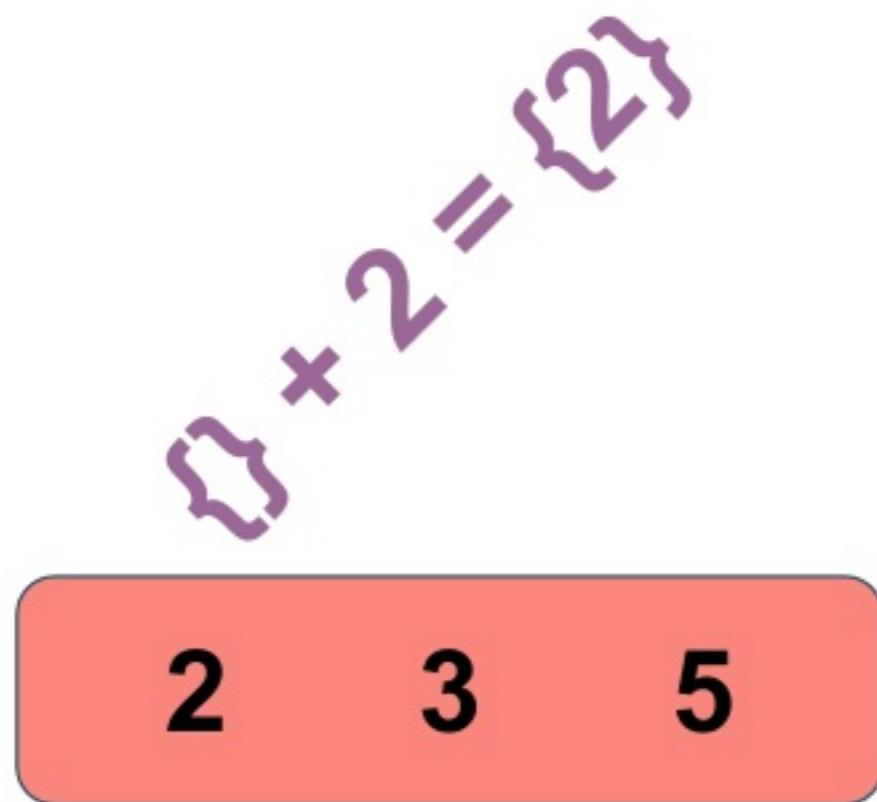
$$10 + 20 = 30$$

Unique



2 3 5

Unique



Unique

$$\{2\} \times \{3\} = \{2, 3\}$$

2 3 5

Unique

2 3 5

$$\{2,3\} \times 5 = \{2,3,5\}$$

Unique

2 3 5

{2,3,5}

Unique

5 3 5

{3,5}

2 3 5

{2,3,5}

Unique

2 3 2

{2,3}

5 3 5

{3,5}

2 3 5

{2,3,5}

Unique

2 3 2

5 3 5

2 3 5

$$\{3,5\} \cup \{2,3\} = \{2,3,5\}$$

$$\{2,3,5\}$$

Unique

2 3 2

5 3 5

2 3 5

$$\{2,3,5\} \cup \{2,3,5\} = \{2,3,5\}$$

Patterns

Examples	Sum	Unique	Pattern
<code>s = 0</code> <code>s = {}</code>	0	{ }	zero (aka identity)

Patterns

Examples	Sum	Unique	Pattern
<code>s = 0</code> <code>s = {}</code>	0	{ }	zero (aka identity)
<code>2 + 3 = 5</code> <code>{2} + 3 = {2, 3}</code>	addition	set insertion	update (aka reduce)

Patterns

Examples	Sum	Unique	Pattern
$s = 0$ $s = \{\}$	0	{ }	zero (aka identity)
$2 + 3 = 5$ $\{2\} + 3 = \{2, 3\}$	addition	set insertion	update (aka reduce)
$13 + 7 = 20$ $\{3,5\} \cup \{2,3\} = \{2,3,5\}$	addition	set union	merge (aka combine)

Spark Operators

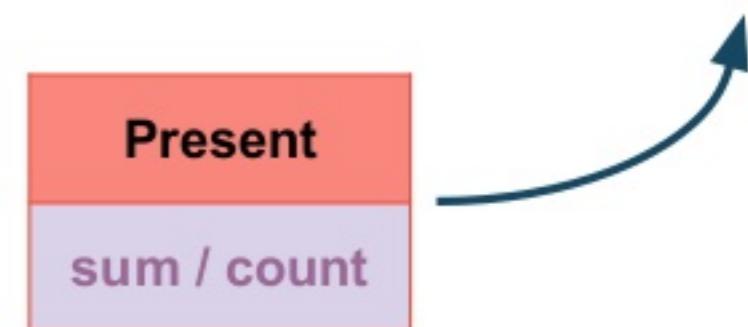
Operation	Data	Accumulator	Zero	Update	Merge
Sum	Numbers	Number	0	$a + x$	$a1 + a2$

Spark Operators

Operation	Data	Accumulator	Zero	Update	Merge
Sum	Numbers	Number	0	$a + x$	$a1 + a2$
Max	Numbers	Number	$-\infty$	$\max(a, x)$	$\max(a1, a2)$

Spark Operators

Operation	Data	Accumulator	Zero	Update	Merge
Sum	Numbers	Number	0	$a + x$	$a_1 + a_2$
Max	Numbers	Number	$-\infty$	$\max(a, x)$	$\max(a_1, a_2)$
Average	Numbers	(sum, count)	(0, 0)	(sum + x, count + 1)	(s1 + s2, c1 + c2)



Shhhhh....

Operation	Data	Accumulator	Zero	Update	Merge
Sum	Numbers	Number	0	$a + x$	$a_1 + a_2$
Max	Numbers	Number	$-\infty$	$\max(a, x)$	$\max(a_1, a_2)$
Average	Numbers	(sum, count)	(0, 0)	(sum + x, count + 1)	$(s_1 + s_2, c_1 + c_2)$

We're secretly algebras!

Algebras are Pattern Checklists

Object Sets
(data types) 

Algebras are Pattern Checklists

- Object Sets
(data types) ✓
- Operations ✓

Algebras are Pattern Checklists

- Object Sets
(data types) ✓
- Operations ✓
- Properties ✓

DataFrame Aggregations...

```
records.show(5)
```

```
+-----+-----+
| user_id|wordcount|
+-----+-----+
| 6458791872|      12|
| 7699035787|       5|
| 2509155359|       9|
| 9914782373|      18|
| 7816616846|      12|
+-----+-----+
```

```
records.groupBy($"user_id")
    .agg(avg($"wordcount").alias("avg"))
    .orderBy($"avg".desc)
    .show(5)
```

```
+-----+-----+
| user_id| avg|
+-----+-----+
| 9438801796| 42.0|
| 0837938601| 41.0|
| 0004926696| 40.0|
| 7439949213| 39.0|
| 2505585758| 39.0|
+-----+-----+
```

DataFrame Aggregations...

```
records.show(5)
```

```
+-----+-----+
| user_id|wordcount|
+-----+-----+
| 6458791872|      12|
| 7699035787|       5|
| 2509155359|       9|
| 9914782373|      18|
| 7816616846|      12|
+-----+-----+
```

```
records.groupBy($"user_id")
    .agg(avg($"wordcount").alias("avg"))
    .orderBy($"avg".desc)
    .show(5)
```

```
+-----+-----+
| user_id| avg|
+-----+-----+
| 9438801796| 42.0|
| 0837938601| 41.0|
| 0004926696| 40.0|
| 7439949213| 39.0|
| 2505585758| 39.0|
+-----+-----+
```

* Are Algebras! *

Aggregator Algebra

Data Type

Aggregator Algebra

Data Type
Accumulator Type

Aggregator Algebra

Data Type
Accumulator Type
Zero

Aggregator Algebra

Data Type
Accumulator Type
Zero
Update

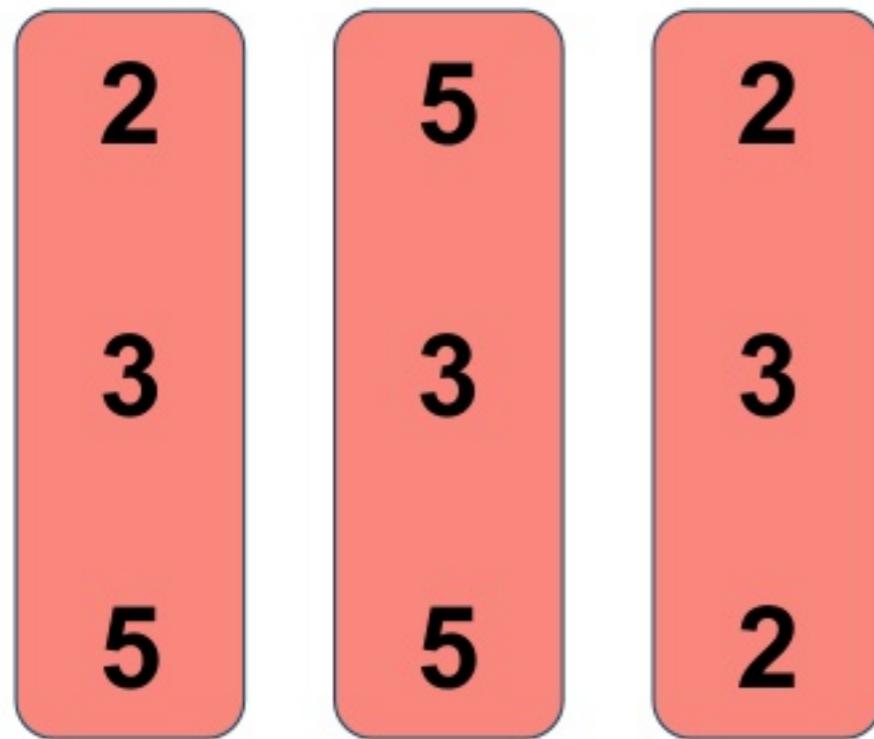
Aggregator Algebra

Data Type
Accumulator Type
Zero
Update
Merge

Aggregator Algebra

Data Type
Accumulator Type
Zero
Update
Merge
Present

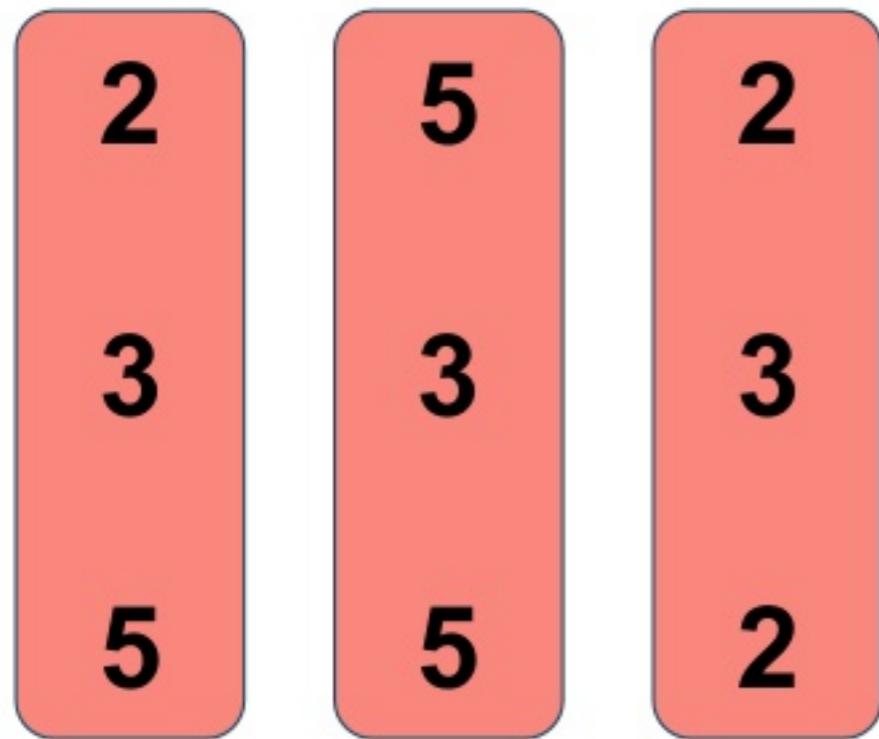
Merge Is Associative



10 13 7

$$(10 + 13) + 7 = 30$$

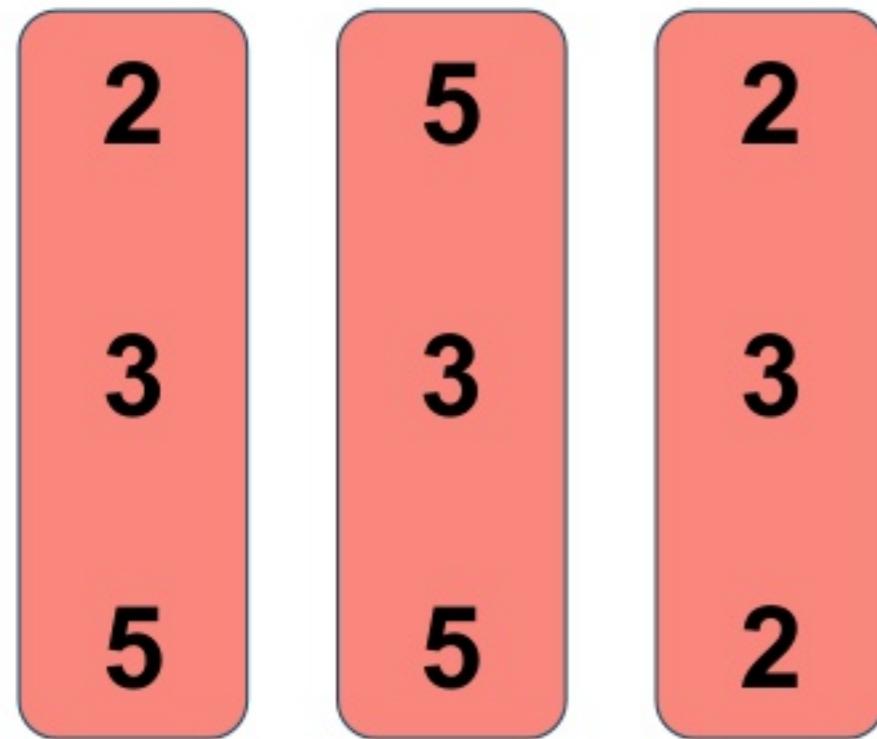
Merge Is Associative



$$(10 + 13) + 7 = 30$$

$$10 + (13 + 7) = 30$$

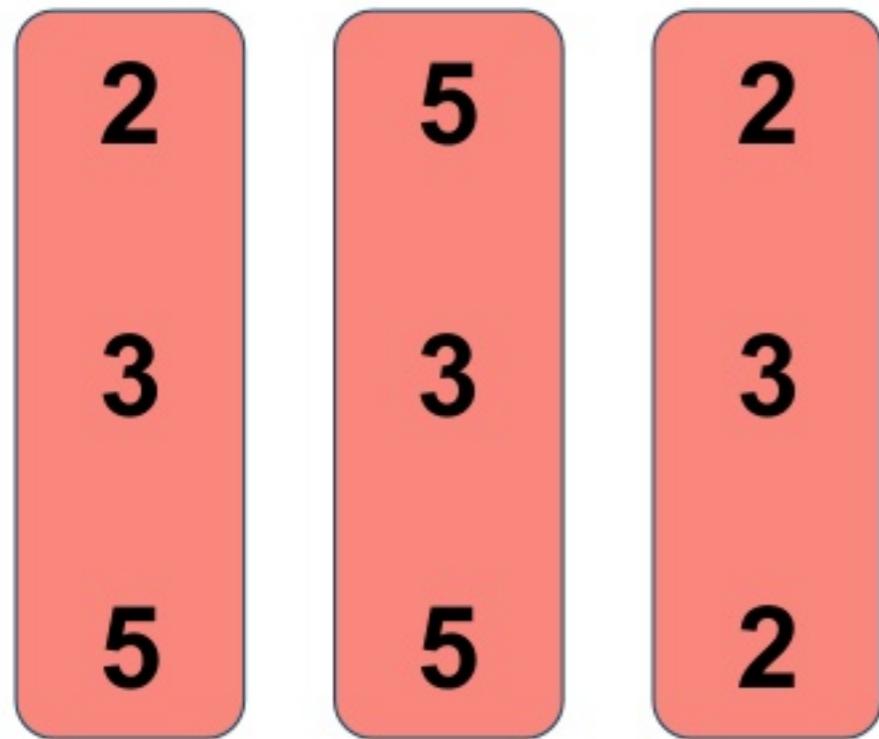
Merge Is (usually) Commutative



$$10 + 13 + 7 = 30$$

10 13 7

Merge Is (usually) Commutative

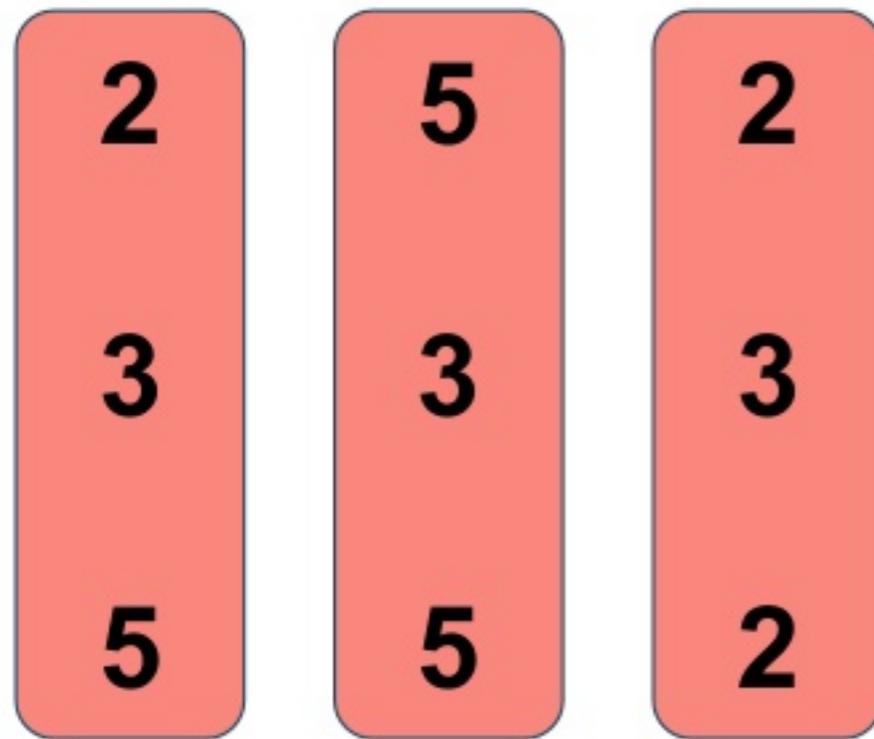


10 13 7

$$10 + 13 + 7 = 30$$

$$13 + 7 + 10 = 30$$

Merge Is (usually) Commutative



10 13 7

sum	$a_1 + a_2 = a_2 + a_1$
max	$\max(a_1, a_2) = \max(a_2, a_1)$
avg	$(s_1+s_2, n_1+n_2) = (s_2+s_1, n_2+n_1)$

Structured Streaming

user	wordcount
a	5
c	7



a	5
c	7

Logical

a	5
c	7

Aggregations

Structured Streaming

user	wordcount
a	5
c	7

b	8
c	10



a	5
c	7

Discarded

a	5
c	7
b	8
c	10

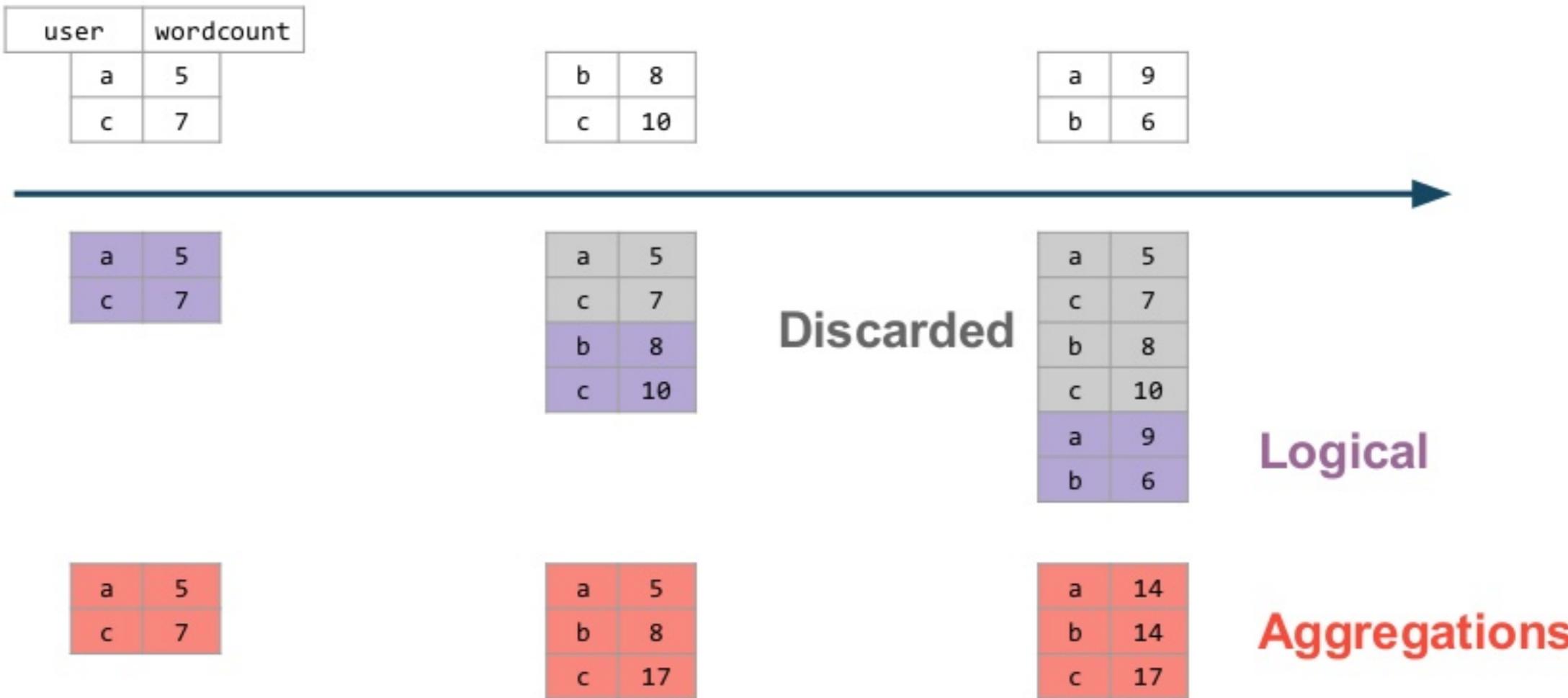
Logical

a	5
c	7

a	5
b	8
c	17

Aggregations

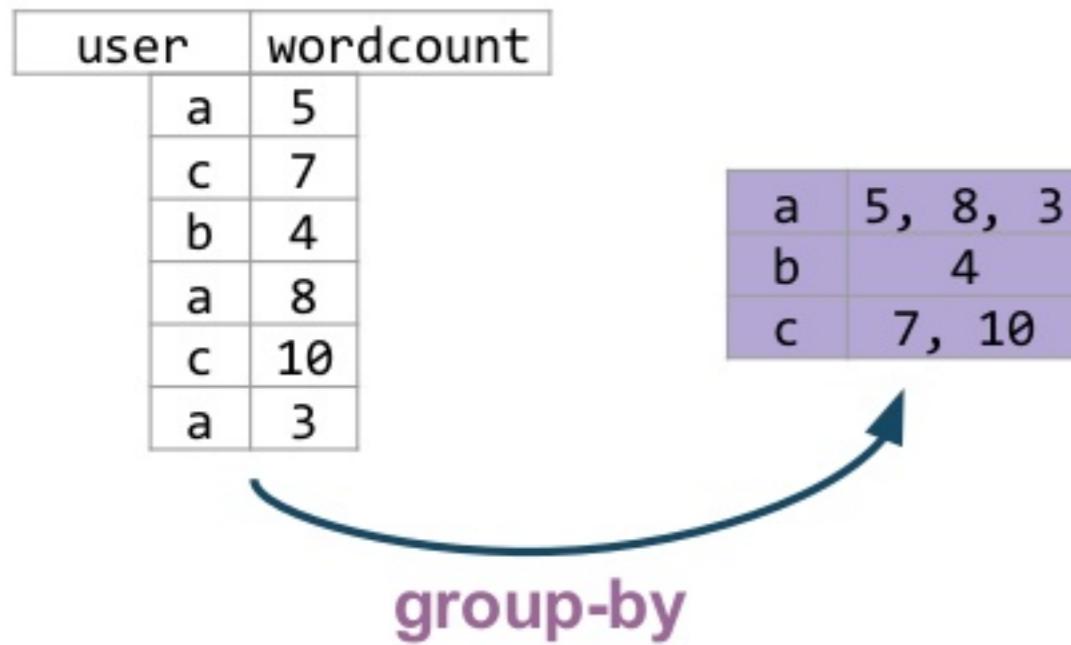
Structured Streaming



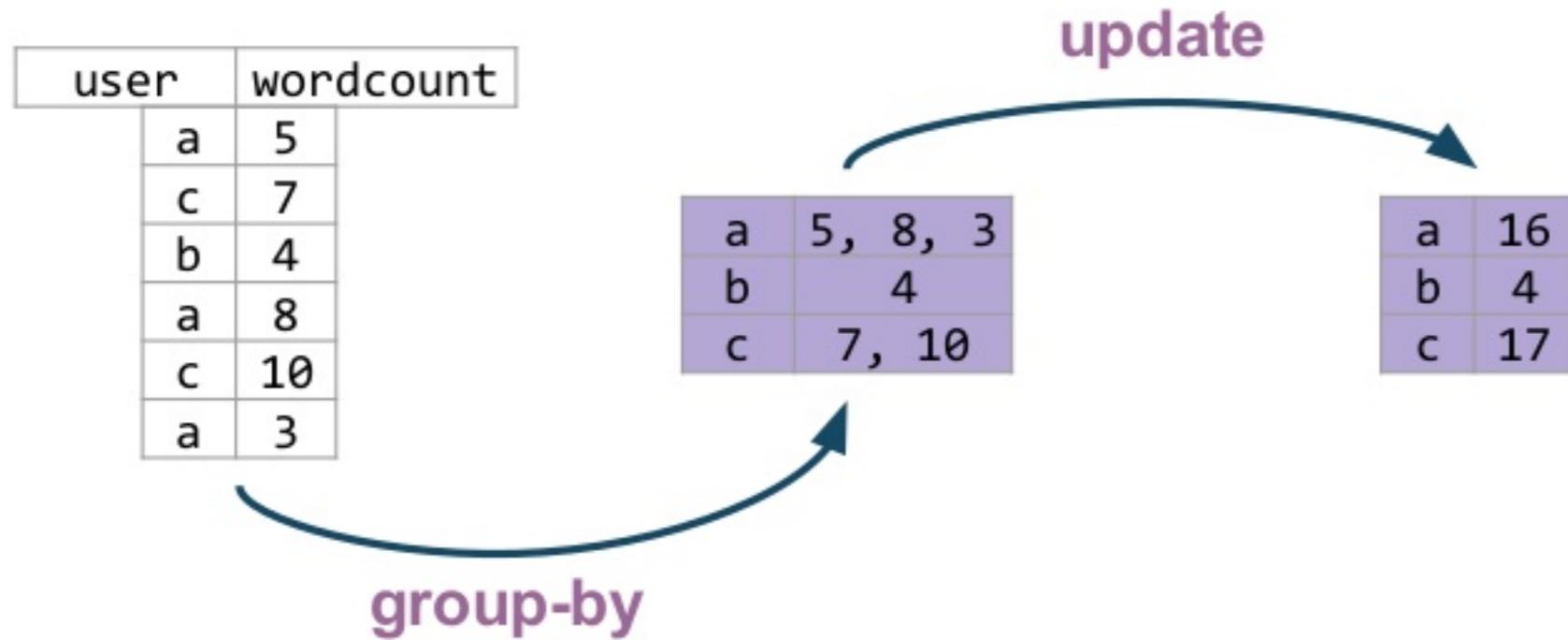
Structured Streaming

user	wordcount
a	5
c	7
b	4
a	8
c	10
a	3

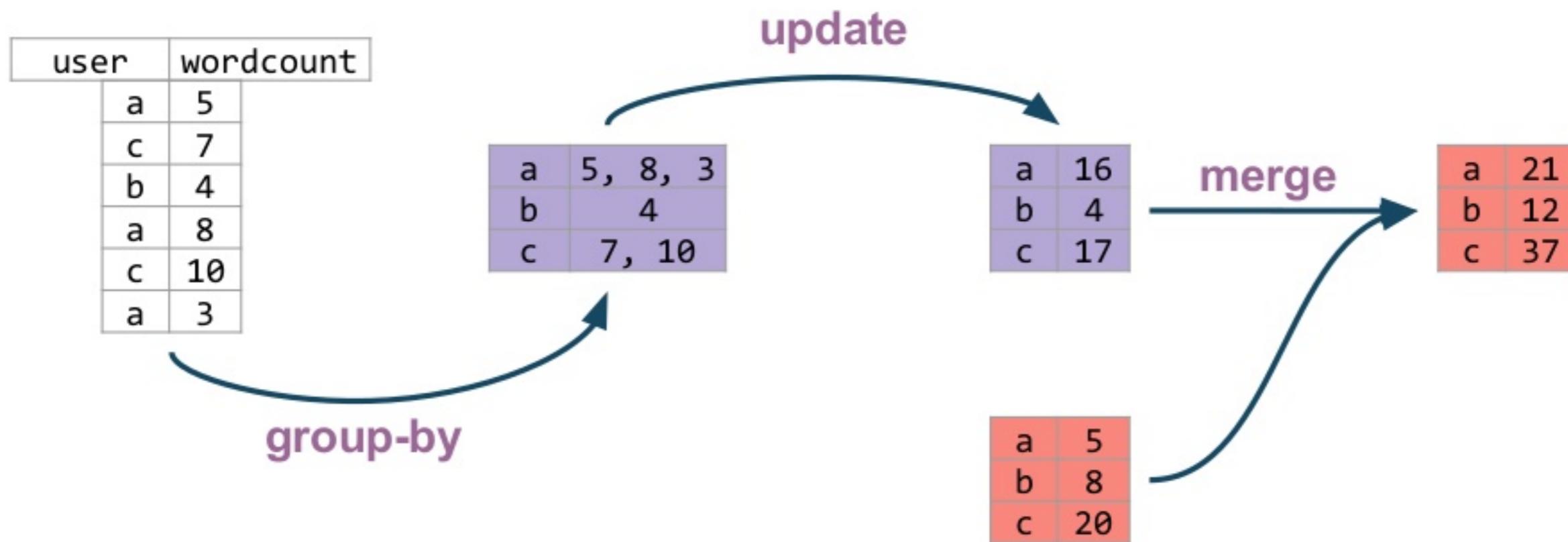
Structured Streaming



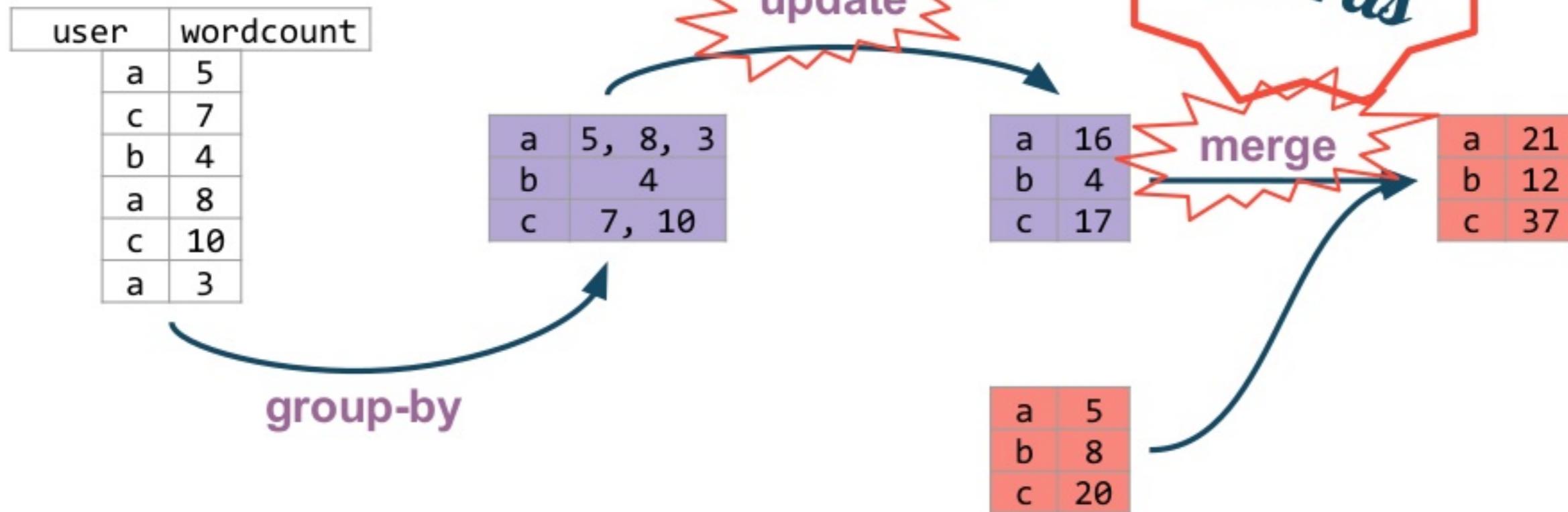
Structured Streaming



Structured Streaming



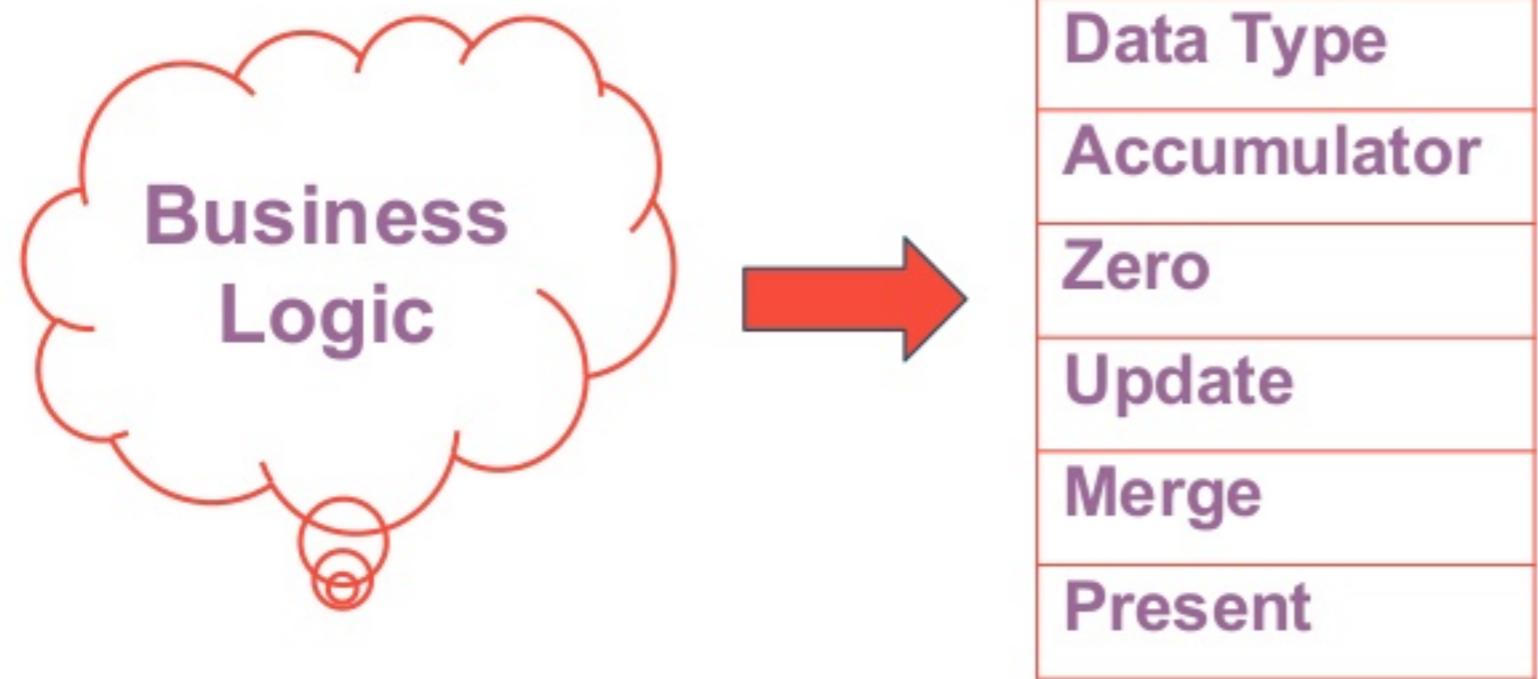
Structured Streaming



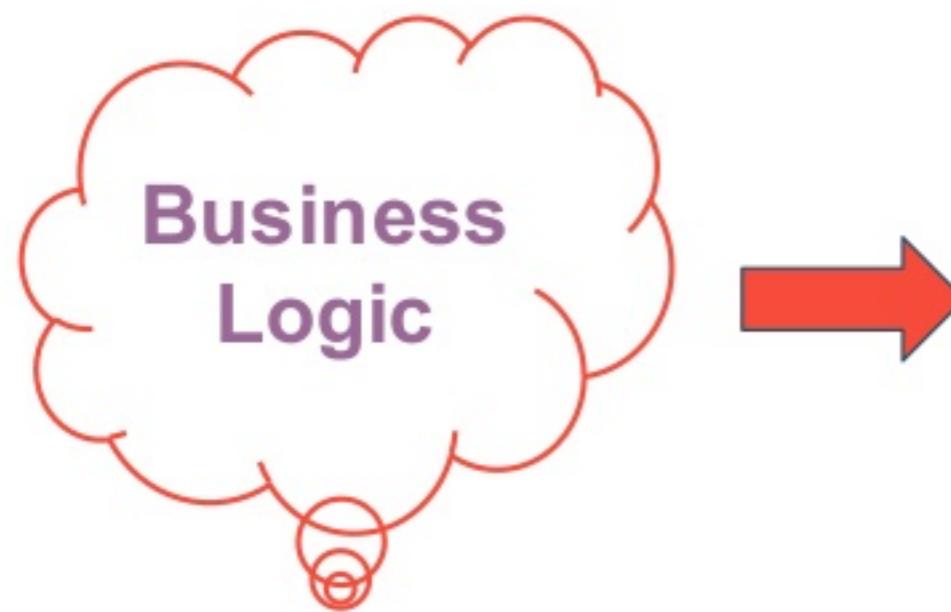
Algebras Around Us



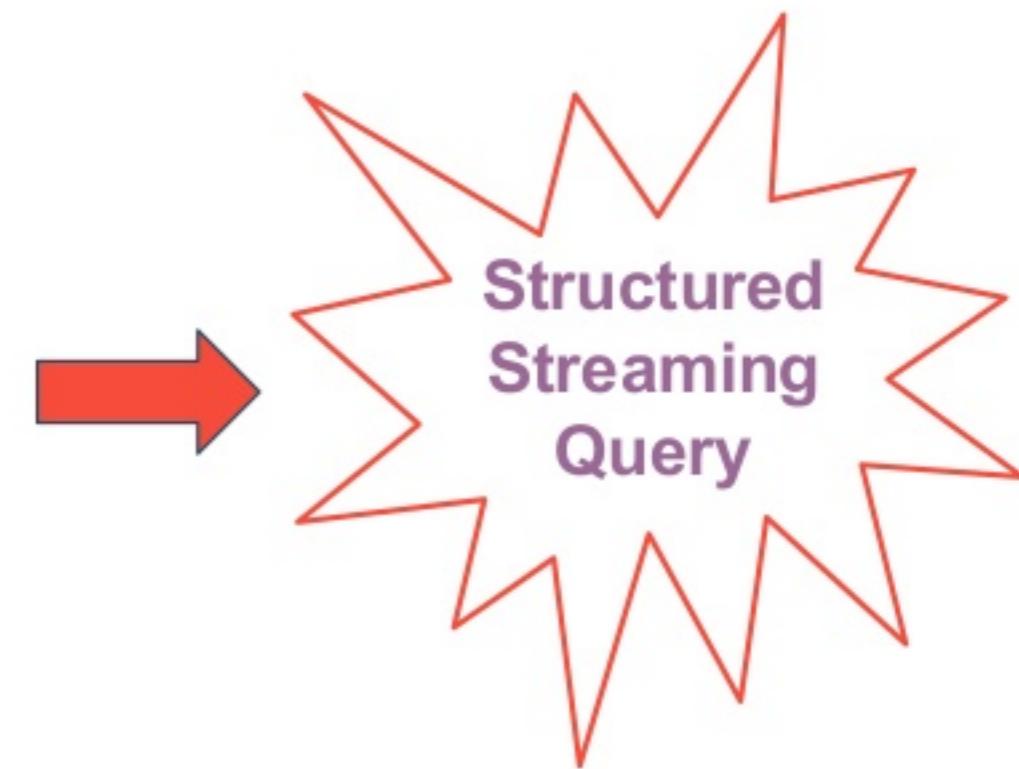
Algebras Around Us



Algebras Around Us



Data Type
Accumulator
Zero
Update
Merge
Present



Aggregating Quantiles



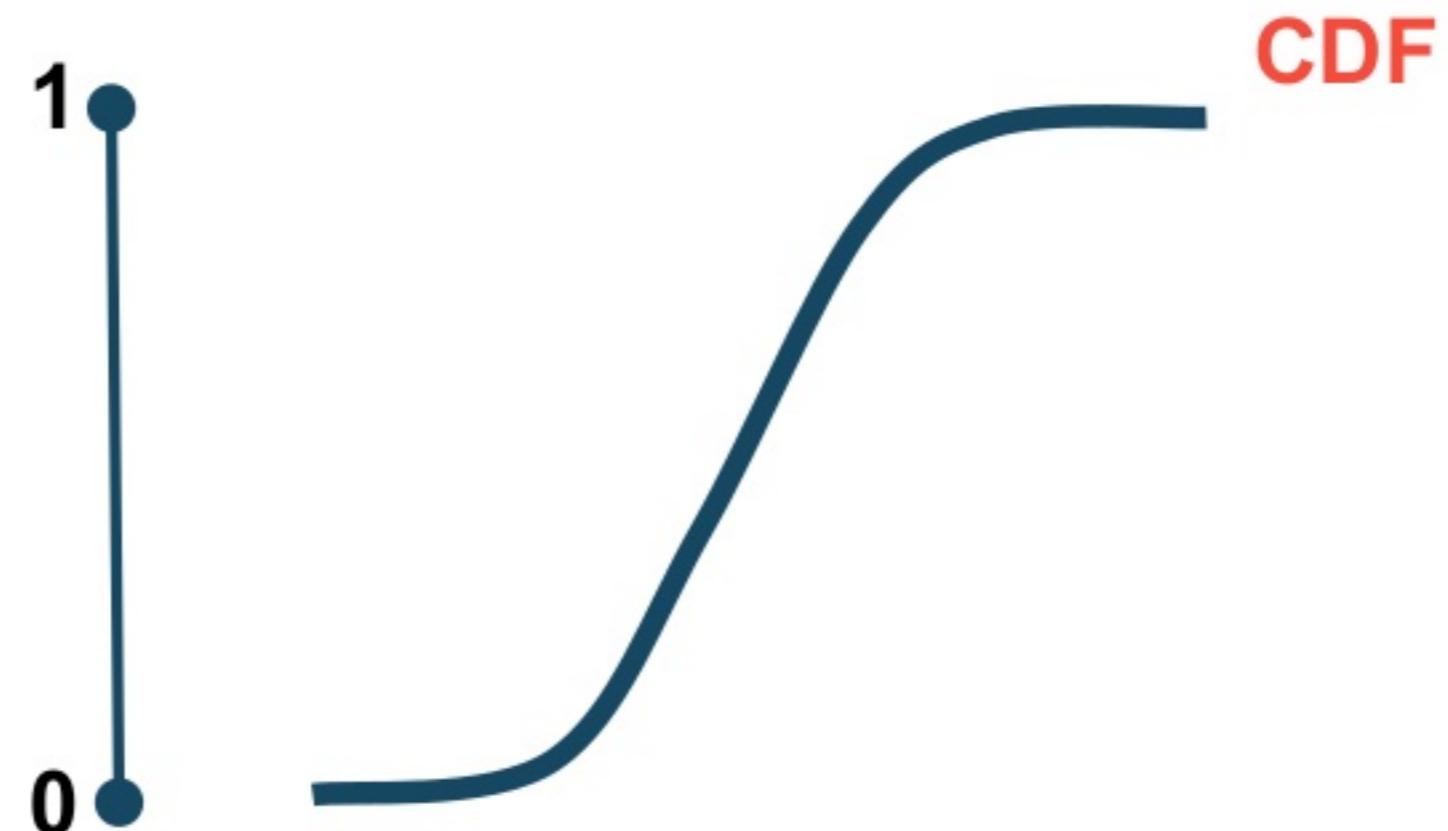
aggregating
query



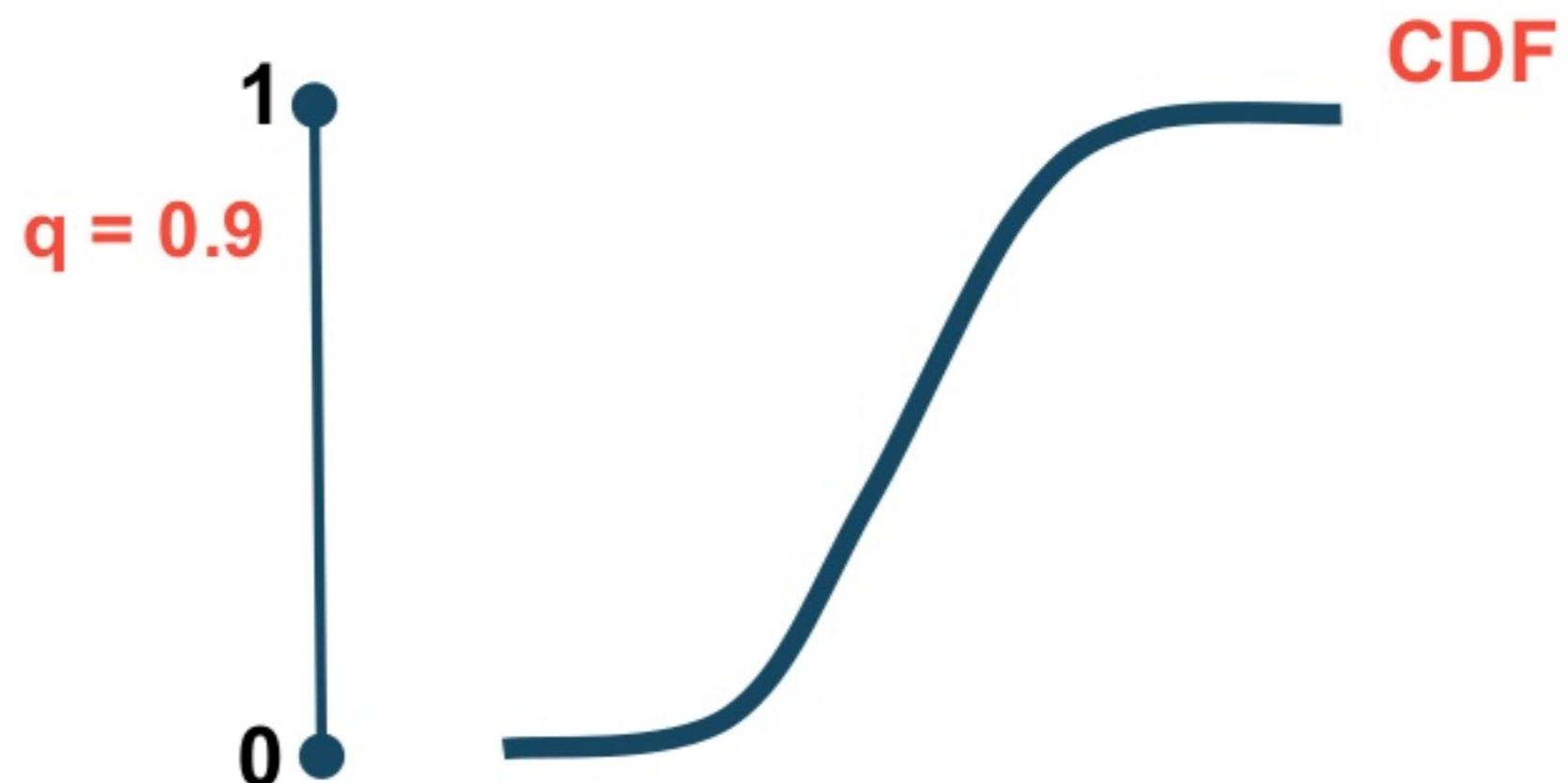
The median of
my data is 11

The 90th
percentile of
my data is 25

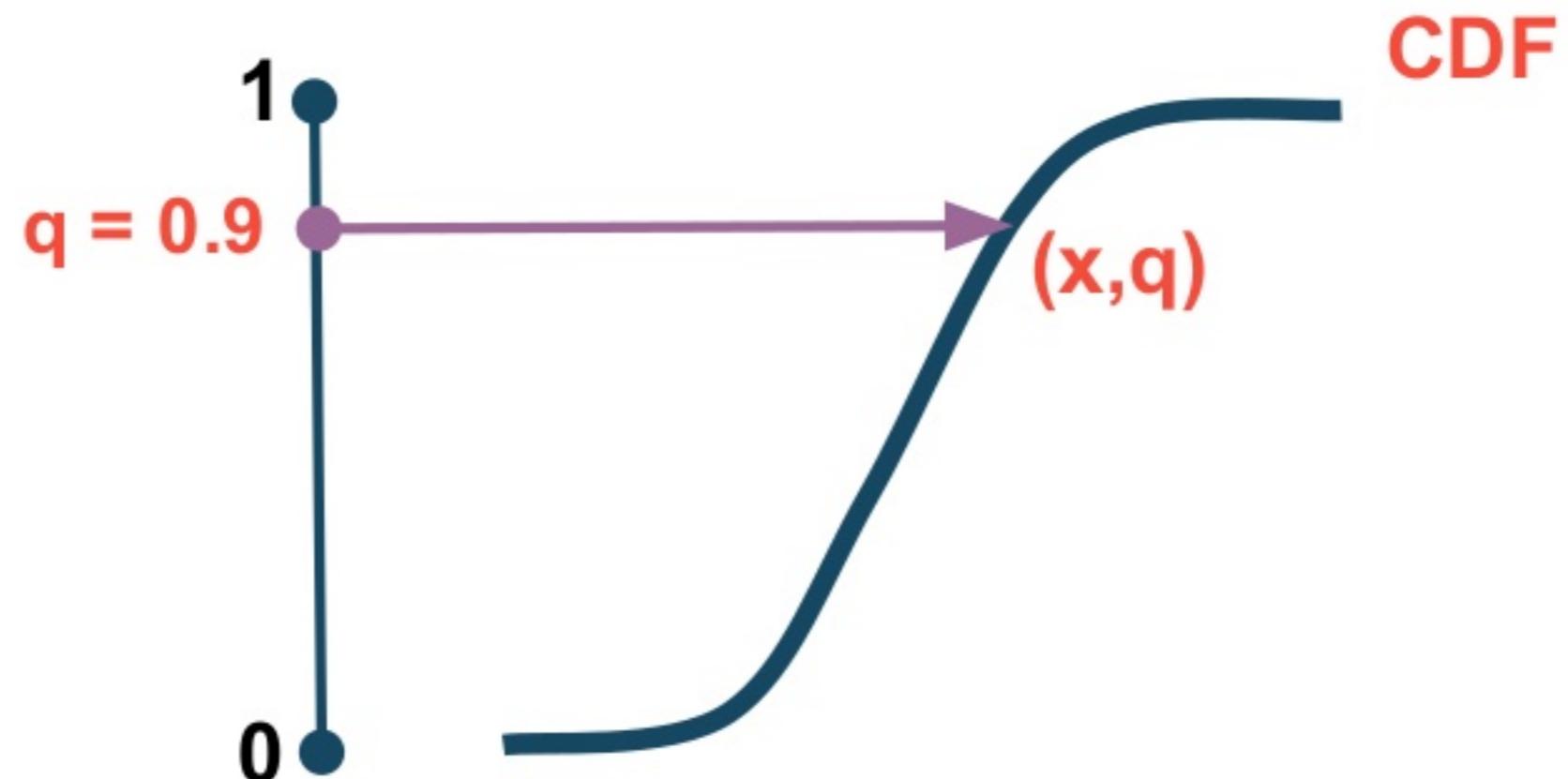
Distribution Sketch: T-Digest



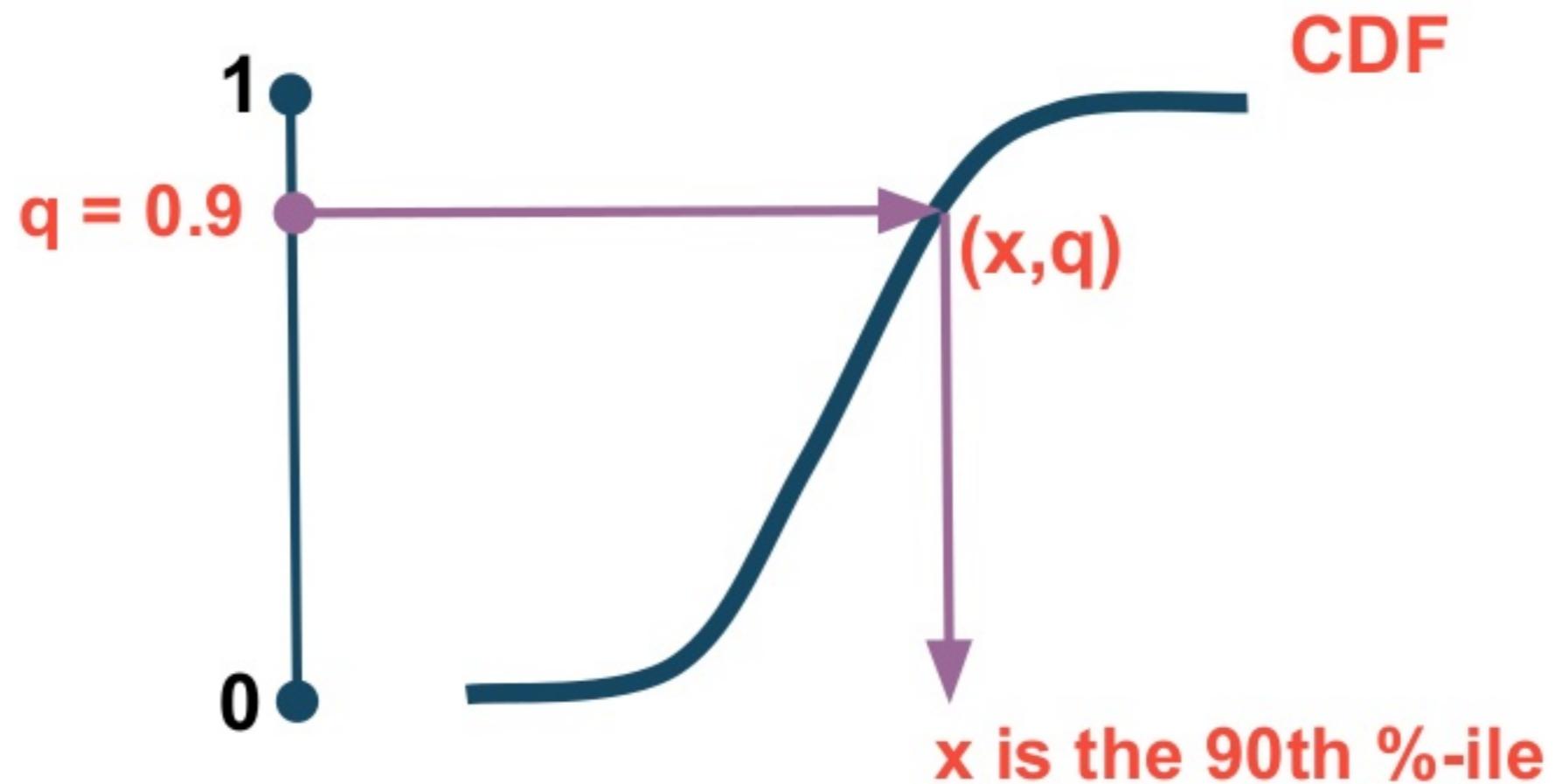
Distribution Sketch: T-Digest



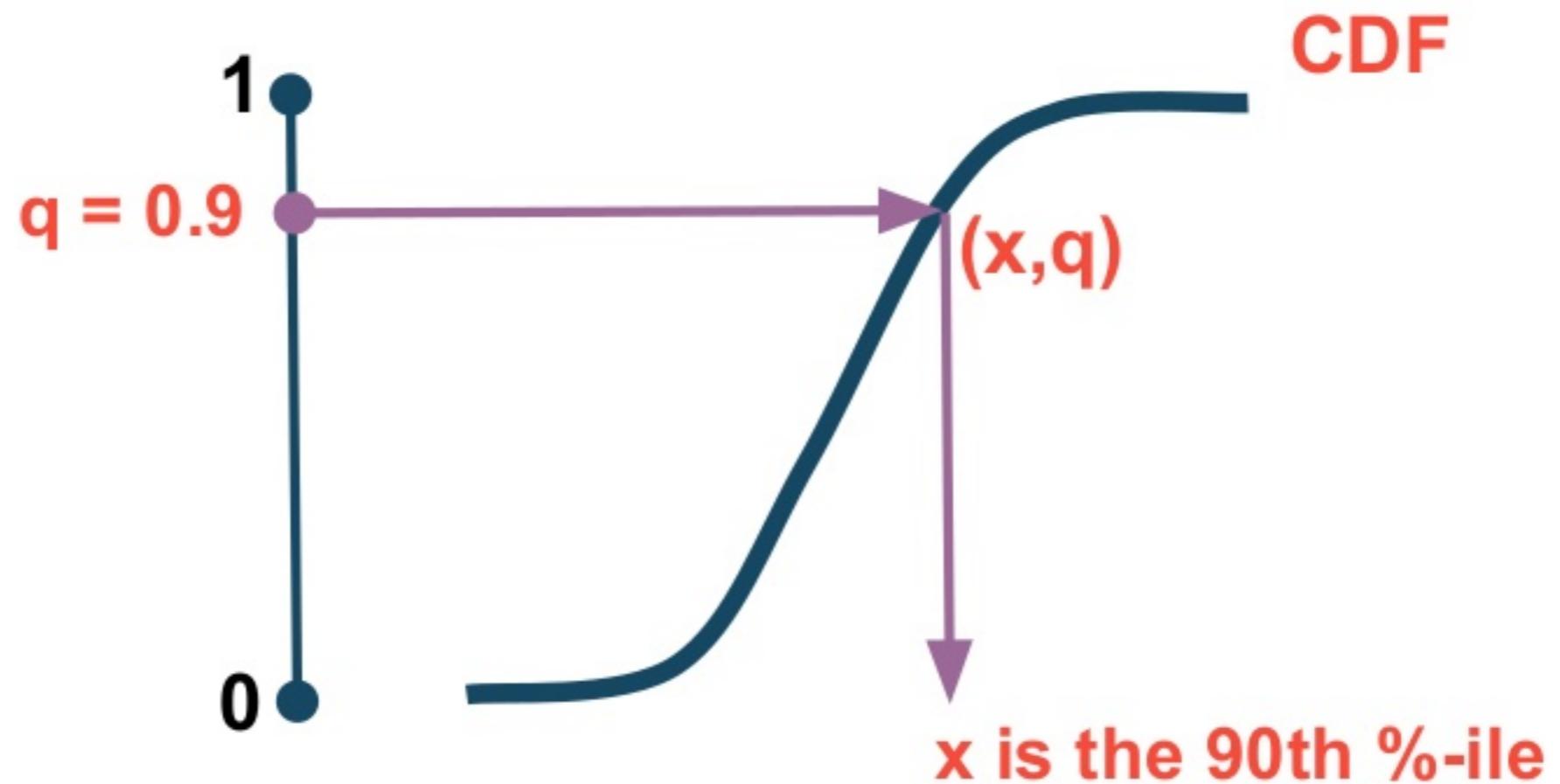
Distribution Sketch: T-Digest



Distribution Sketch: T-Digest



Distribution Sketch: T-Digest



Is T-Digest an Algebra?

Data Type

Numeric

Is T-Digest an Algebra?

Data Type	Numeric
Accumulator Type	T-Digest Sketch

Is T-Digest an Algebra?

Data Type	Numeric
Accumulator Type	T-Digest Sketch
Zero	Empty T-Digest

Is T-Digest an Algebra?

Data Type	Numeric
Accumulator Type	T-Digest Sketch
Zero	Empty T-Digest
Update	<code>tdigest + x</code>

Is T-Digest an Algebra?

Data Type	Numeric
Accumulator Type	T-Digest Sketch
Zero	Empty T-Digest
Update	<code>tdigest + x</code>
Merge	<code>tdigest1 + tdigest2</code>

Is T-Digest an Algebra?

Data Type	Numeric
Accumulator Type	T-Digest Sketch
Zero	Empty T-Digest
Update	<code>tdigest + x</code>
Merge	<code>tdigest1 + tdigest2</code>
Present	<code>tdigest.cdfInverse(quantile)</code>

Is T-Digest an Algebra?



Data Type	Numeric
Accumulator Type	T-Digest Sketch
Zero	Empty T-Digest
Update	<code>tdigest + x</code>
Merge	<code>tdigest1 + tdigest2</code>
Present	<code>tdigest.cdfInverse(quantile)</code>

User Defined Aggregator Function

```
val sketchCDF = tdigestUDAF[Double]
```

```
spark.udf.register("p50",  
  (c:Any)=>c.asInstanceOf[TDigestSQL].tdigest.cdfInverse(0.5))
```

```
spark.udf.register("p90",  
  (c:Any)=>c.asInstanceOf[TDigestSQL].tdigest.cdfInverse(0.9))
```

User Defined Aggregator Function

```
val sketchCDF = tdigestUDAF[Double]
```



```
spark.udf.register("p50",  
    (c:Any)=>c.asInstanceOf[TDigestSQL].tdigest.cdfInverse(0.5))
```

```
spark.udf.register("p90",  
    (c:Any)=>c.asInstanceOf[TDigestSQL].tdigest.cdfInverse(0.9))
```

Streaming Percentiles

```
val query = records  
  .writeStream //...
```

wordcount
12
5
9
18
12

```
val r = records.withColumn("time", current_timestamp())  
  .groupBy(window($"time", "30 seconds"))  
  .agg(sketchCDF($"wordcount").alias("CDF"))  
  .select(callUDF("p50", $"CDF").alias("p50"),  
          callUDF("p90", $"CDF").alias("p90"))  
val query = r.writeStream //...
```

p50	p90
15.6	31.0
16.0	30.8
15.8	30.0
15.7	31.0
16.0	31.0

Streaming Percentiles

```
val query = records  
  .writeStream //...
```

```
+-----+  
|wordcount|  
+-----+  
| 12 |  
| 5 |  
| 9 |  
| 18 |  
| 12 |  
+-----+
```

```
val r = records.withColumn("time", current_timestamp())  
  .groupBy(window($"time", "30 seconds"))  
  .agg(sketchCDF($"wordcount").alias("CDF"))  
  .select(callUDF("p50", $"CDF").alias("p50"),  
          callUDF("p90", $"CDF").alias("p90"))  
val query = r.writeStream //...
```

```
+----+----+  
| p50 | p90 |  
+----+----+  
| 15.6 | 31.0 |  
| 16.0 | 30.8 |  
| 15.8 | 30.0 |  
| 15.7 | 31.0 |  
| 16.0 | 31.0 |  
+----+----+
```



Most-Frequent Items

#DogRates

#YOLO

#SAIRocks

#TruthBomb

#Mondays

aggregating
query



#tag	frequency
#DogRates	1000000000
#Blockchain	780000
#TaylorSwift	650000

Heavy-Hitter Sketch

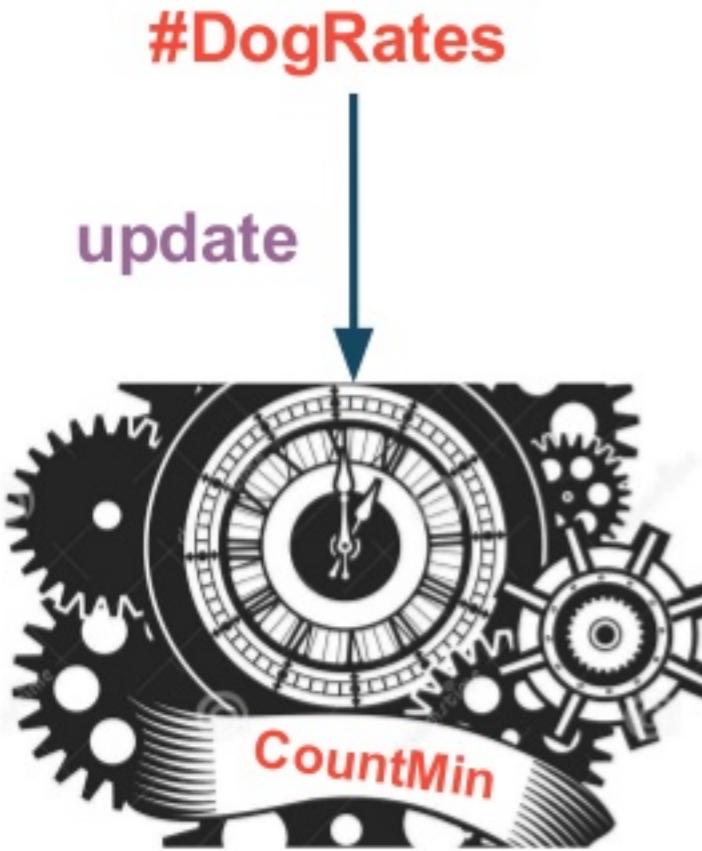


I estimate frequencies of objects!

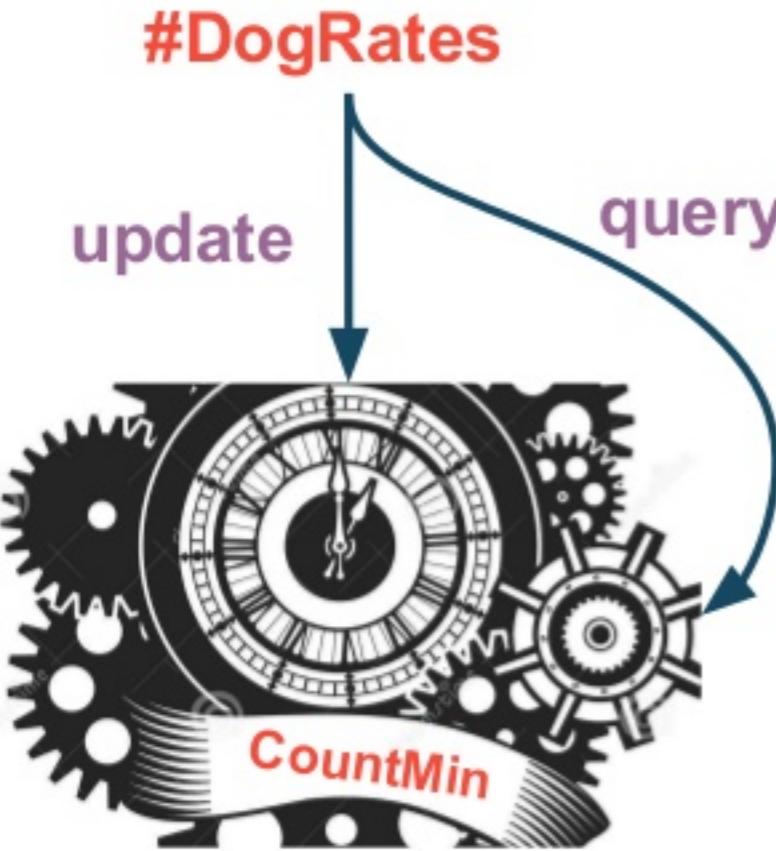


I store objects in sorted order!

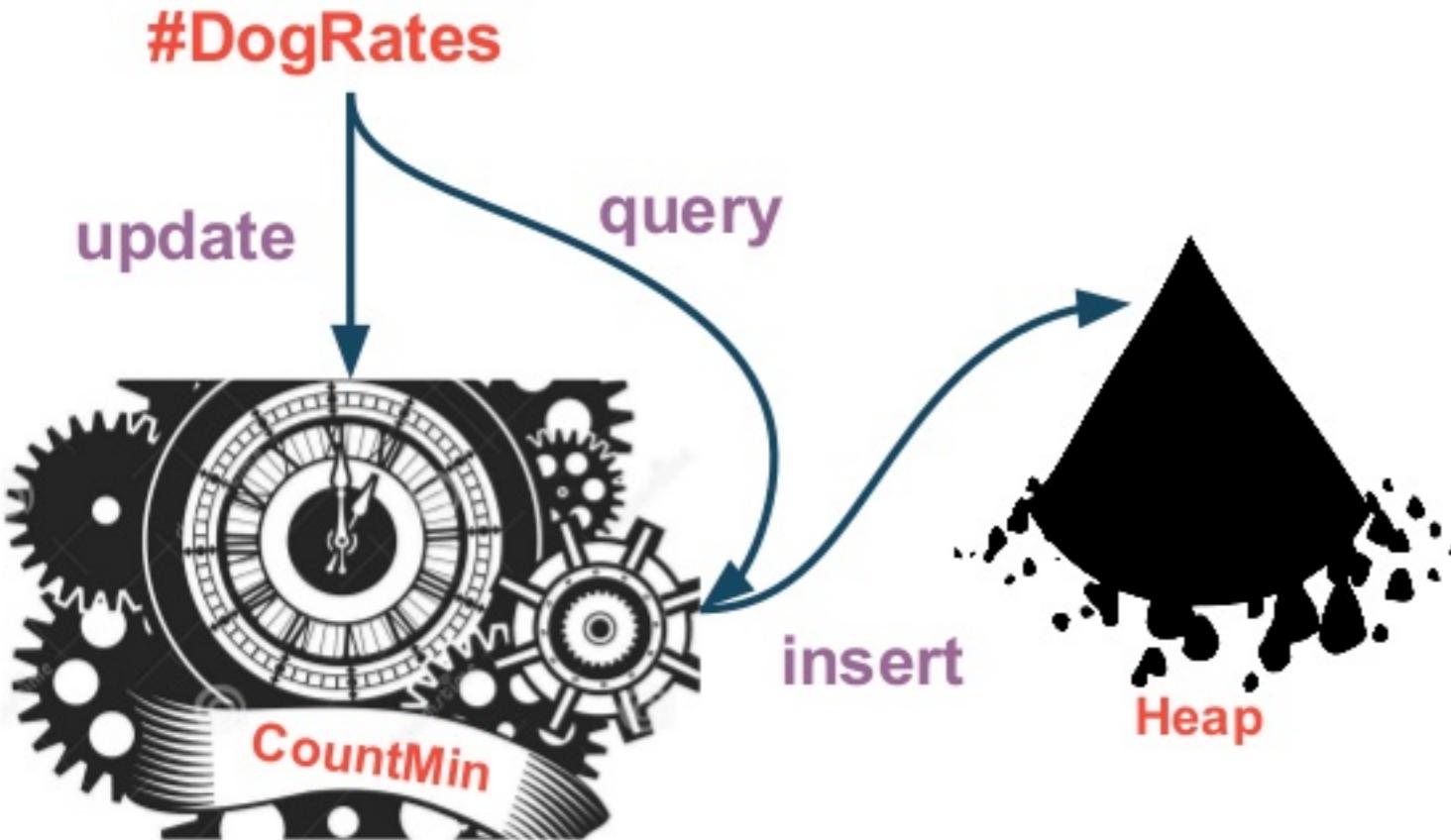
Heavy-Hitter Sketch



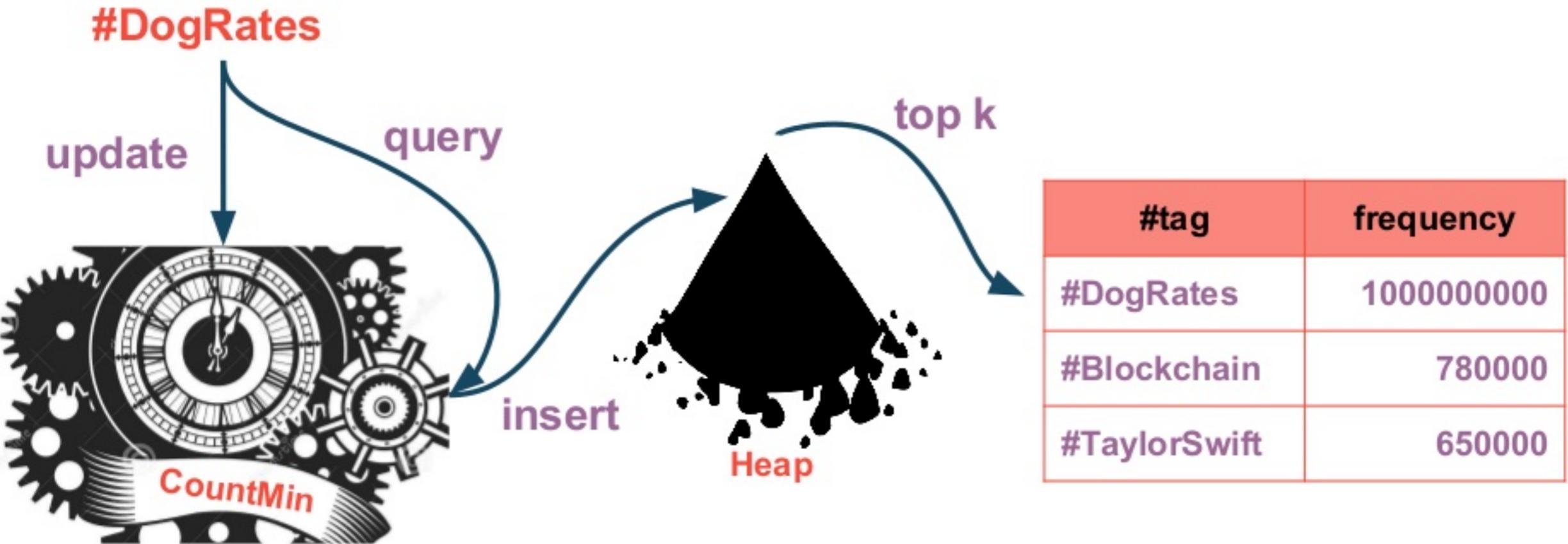
Heavy-Hitter Sketch



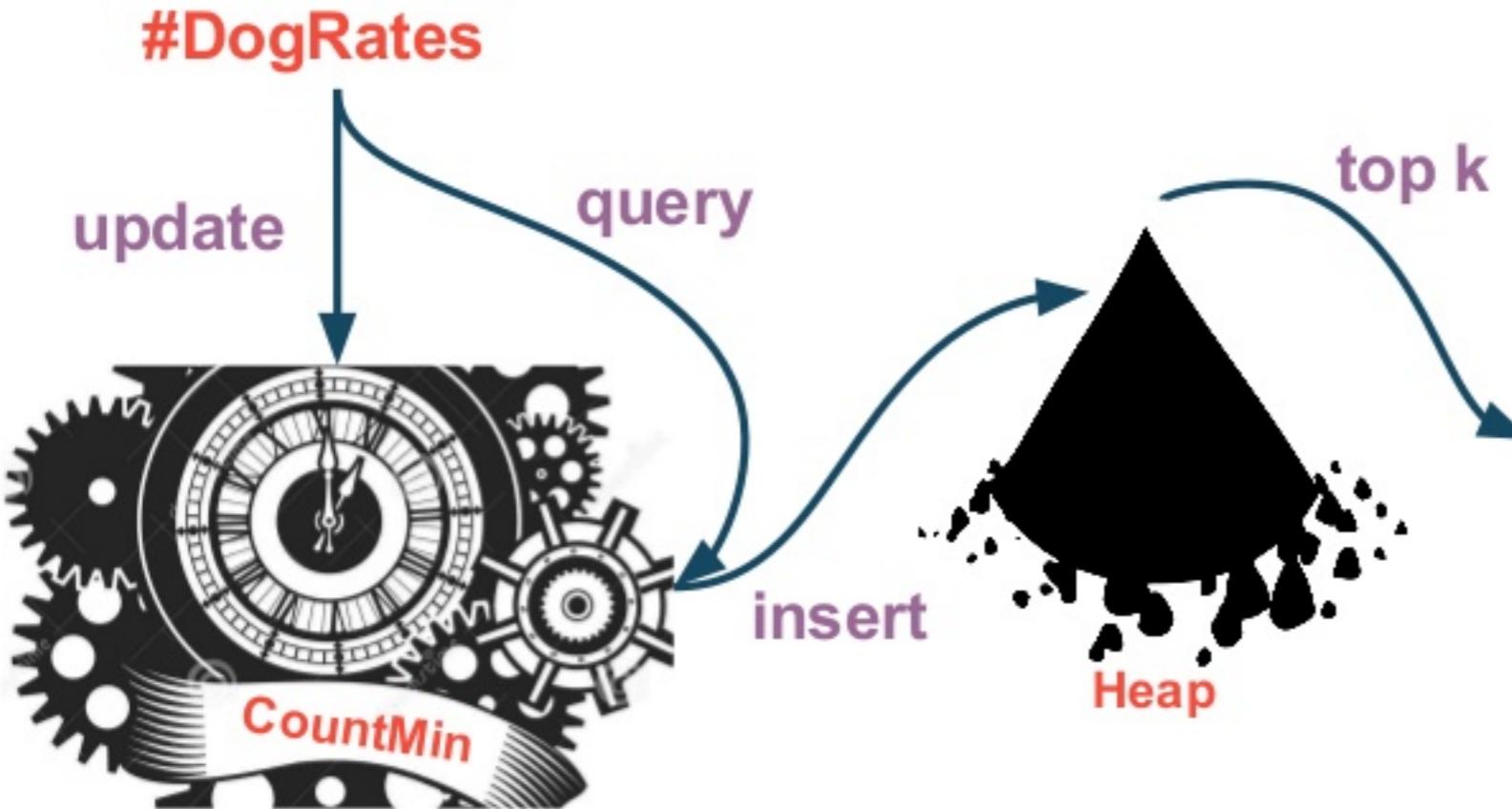
Heavy-Hitter Sketch



Heavy-Hitter Sketch



Heavy-Hitter Sketch



#tag	frequency
#DogRates	1000000000
#Blockchain	780000
#TaylorSwift	650000

Is Heavy-Hitter an Algebra?

Data Type	Items

Is Heavy-Hitter an Algebra?

Data Type	Items
Accumulator Type	countmin sketch heap

Is Heavy-Hitter an Algebra?

Data Type	Items
Accumulator Type	<code>countmin sketch</code> <code>heap</code>
Zero	<code>empty countmin</code> <code>empty heap</code>

Is Heavy-Hitter an Algebra?

Data Type	Items
Accumulator Type	<code>countmin sketch</code> <code>heap</code>
Zero	<code>empty countmin</code> <code>empty heap</code>
Update	<code>countmin.update(x)</code> <code>heap.insert(x, frequency)</code>

Is Heavy-Hitter an Algebra?

Data Type	Items
Accumulator Type	<code>countmin sketch heap</code>
Zero	<code>empty countmin empty heap</code>
Update	<code>countmin.update(x) heap.insert(x, frequency)</code>
Merge	<code>countmin1 + countmin2 update(heap1 + heap2)</code>

Is Heavy-Hitter an Algebra?

Data Type	Items
Accumulator Type	<code>countmin sketch</code> <code>heap</code>
Zero	<code>empty countmin</code> <code>empty heap</code>
Update	<code>countmin.update(x)</code> <code>heap.insert(x, frequency)</code>
Merge	<code>countmin1 + countmin2</code> <code>update(heap1 + heap2)</code>
Present	<code>heap.top(k)</code>

Is Heavy-Hitter an Algebra?

Data Type	Items
Accumulator Type	countmin sketch heap
Zero	empty countmin empty heap
Update	countmin.update(x) heap.insert(x, frequency)
Merge	countmin1 + countmin2 update(heap1 + heap2)
Present	heap.top(k)



Streaming Most-Frequent Items

```
val query = records  
  .writeStream //...
```

```
+-----+  
| hashtag |  
+-----+  
| #Gardiner |  
| #PETE |  
| #Nutiva |  
| #Fairfax |  
| #Darcy |  
+-----+
```

```
val windowBy60 =  
  windowing.windowBy[(Timestamp, String)](_.1, 60)  
val top3 = new TopKAgg[(Timestamp, String)](_.2)  
val r = records.withColumn("time", current_timestamp())  
  .as[(Timestamp, String)]  
  .groupByKey(windowBy60).agg(top3.toColumn)  
  .map { case (_, tk) => tk.toList.toString }  
val query = r.writeStream //...  
  
+-----+  
+ value |  
+-----+  
| List((#Iddesleigh,3), (#Gardiner,2), (#Willoughby,2)) |  
| List((#Elizabeth,3), (#HERPRESENT,1), (#upforhours,1)) |  
| List((#PETE,4), (#Nutiva,1), (#Fairfax,1)) |  
+-----+
```

Review

2 3 2

5 3 5

$$13 + 7 = 20$$

2 3 5

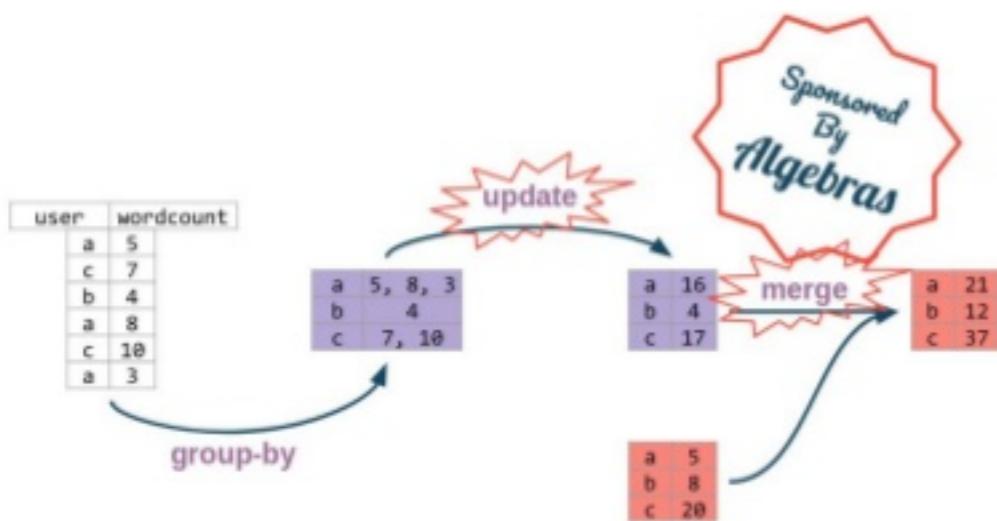
10

Review

2 3 2

5 3 5 $13 + 7 = 20$

2 3 5 10



Review

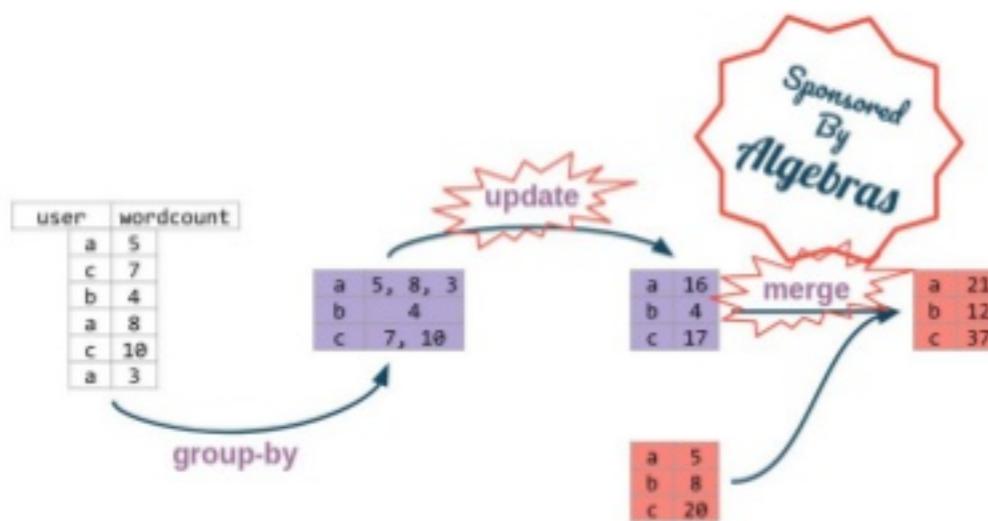
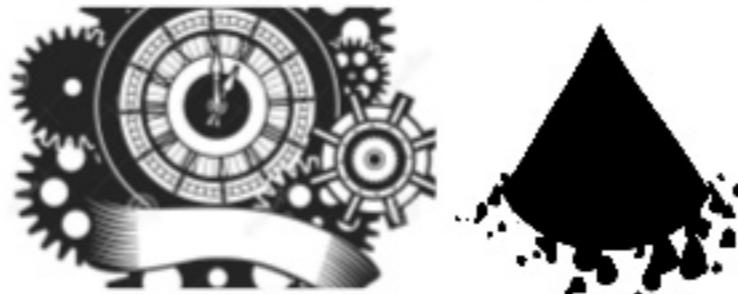
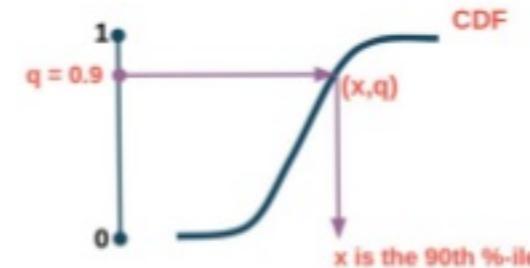
2 3 2

5 3 5

$$13 + 7 = 20$$

2 3 5

10



Review

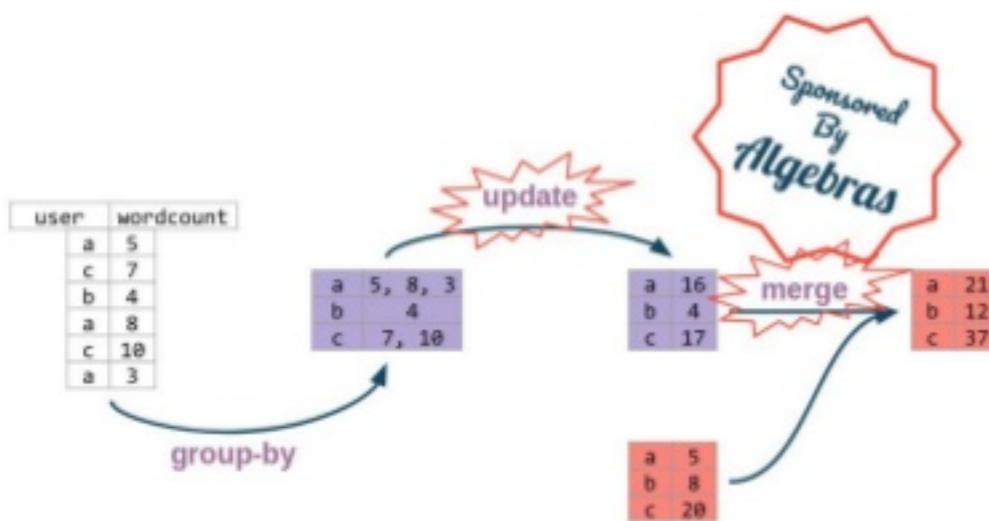
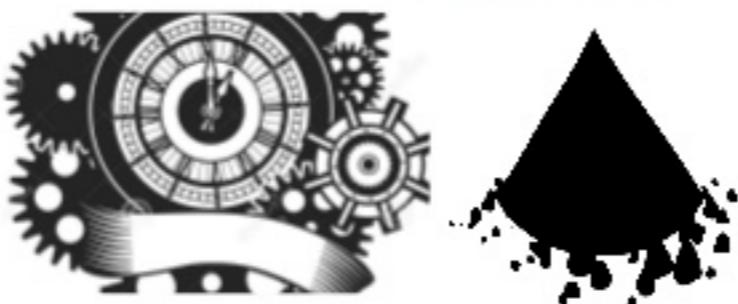
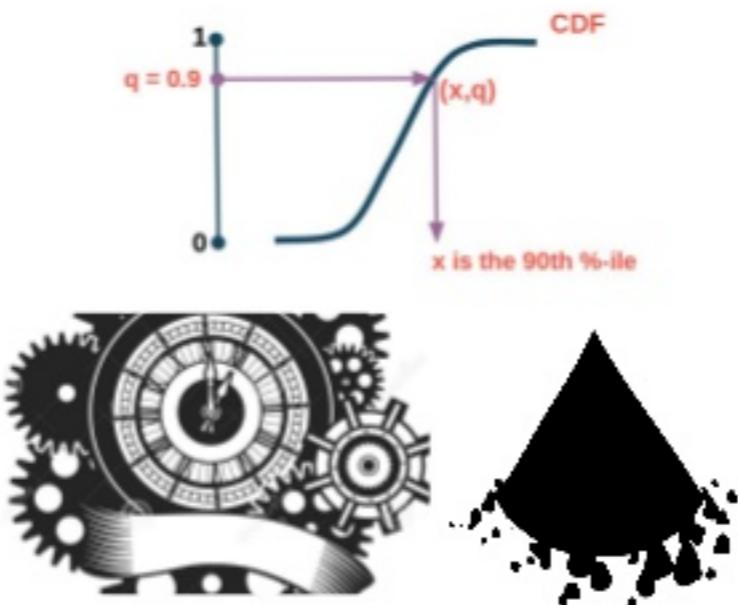
2 3 2

5 3 5

$$13 + 7 = 20$$

2 3 5

10



```
val windowBy60 =  
  windowing.windowBy[(Timestamp, String)](_-1, 60)  
val top3 = new TopKAgg[(Timestamp, String)](_-2)  
val r = records.withColumn("time", current_timestamp())  
  .as[(Timestamp, String)]  
  .groupByKey(windowBy60).agg(top3.toColumn)  
  .map { case (_, tk) => tk.toList.toString }  
val query = r.writeStream //...
```

```
+-----+  
+ value  
+-----+  
|List((#Iddesleigh,3), (#Gardiner,2), (#Willoughby,2))|  
|List((#Elizabeth,3), (#HERPRESENT,1), (#upforhours,1))|  
|List((#PETE,4), (#Nutiva,1), (#Fairfax,1))|  
+-----+
```

Review

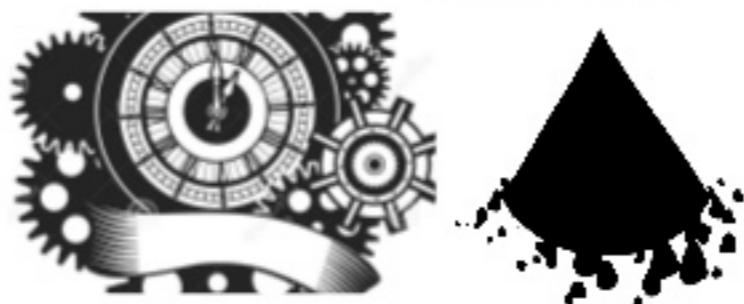
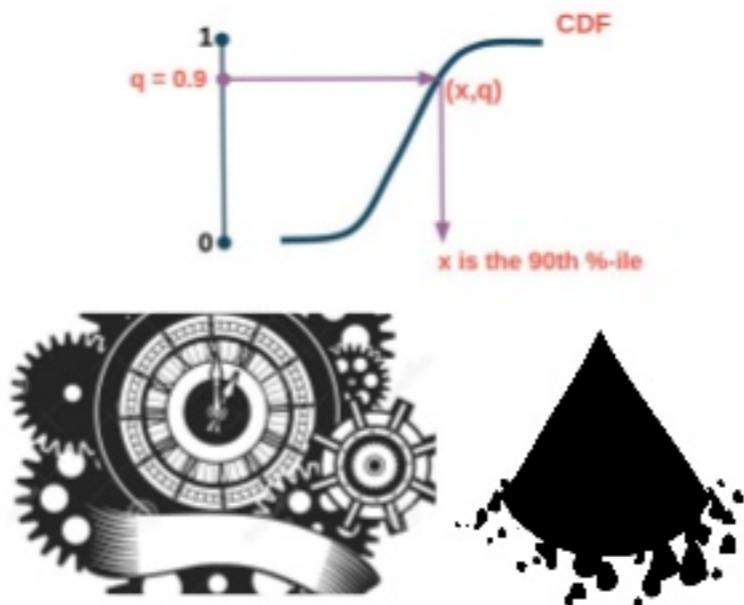
2 3 2

5 3 5

$$13 + 7 = 20$$

2 3 5

10



user	wordcount
a	5
c	7
b	4
a	8
c	18
a	3

group-by

update

group

Sponsored
By
Algebras

merge



#SAISDev2

```
val windowBy60 =  
  windowing.windowBy[(Timestamp, String)](_-1, 60)  
val top3 = new TopKAgg[(Timestamp, String)](_-2)  
val r = records.withColumn("time", current_timestamp())  
  .as[(Timestamp, String)]  
  .groupByKey(windowBy60).agg(top3.toColumn)  
  .map { case (_, tk) => tk.toList.toString }  
val query = r.writeStream //...
```

```
+-----+  
+ value  
+-----+  
|List((#Iddesleigh,3), (#Gardiner,2), (#Willoughby,2))|  
|List((#Elizabeth,3), (#HERPRESENT,1), (#upforhours,1))|  
|List((#PETE,4), (#Nutiva,1), (#Fairfax,1))|  
+-----+
```

eje@redhat.com
@manyangled