Optimizing, Structured Streaming & Spark 2.x

Justin Pihony

@JustinPihony|justin-pihony.blogspot.com

Course Overview

DataFrames

Datasets

Spark Streaming

Optimizing Towards Fast Data

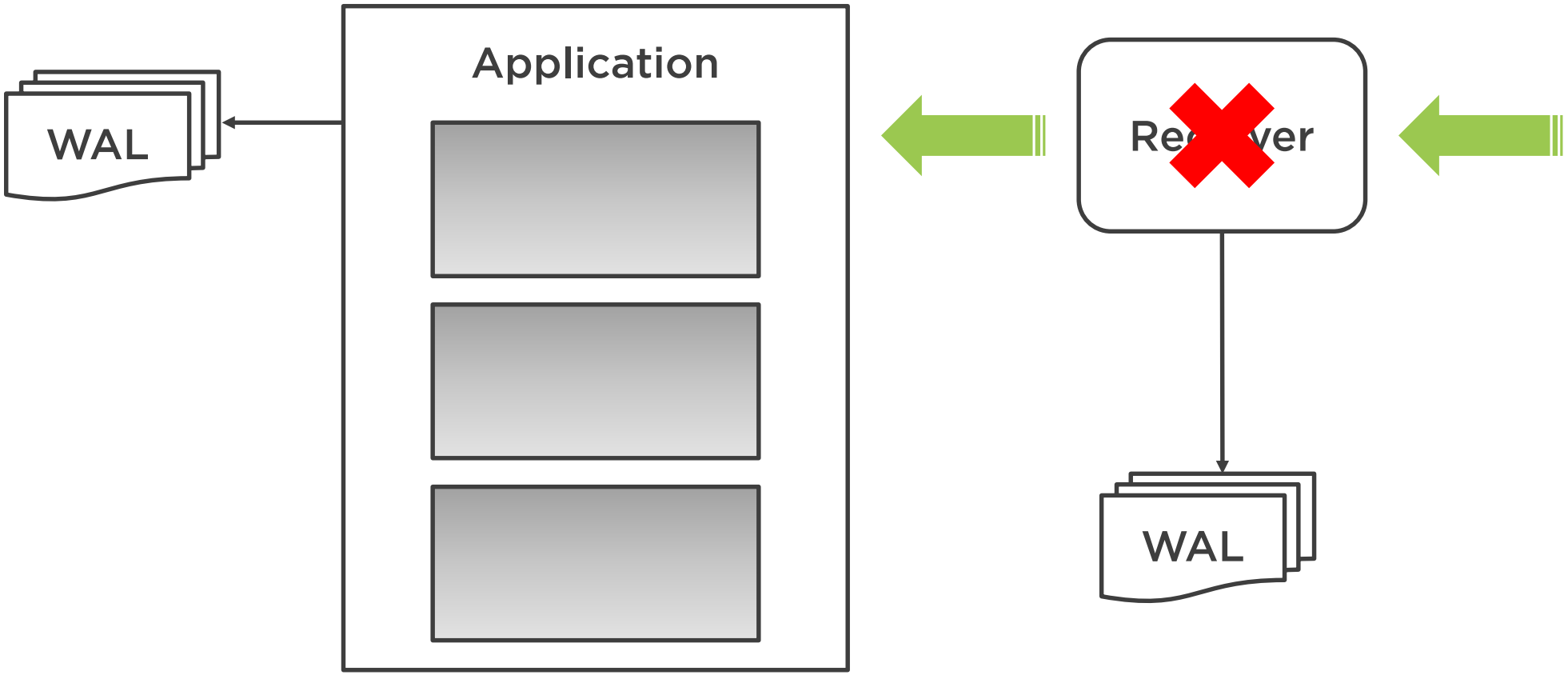
Module Overview

Optimizing Towards Fast Data

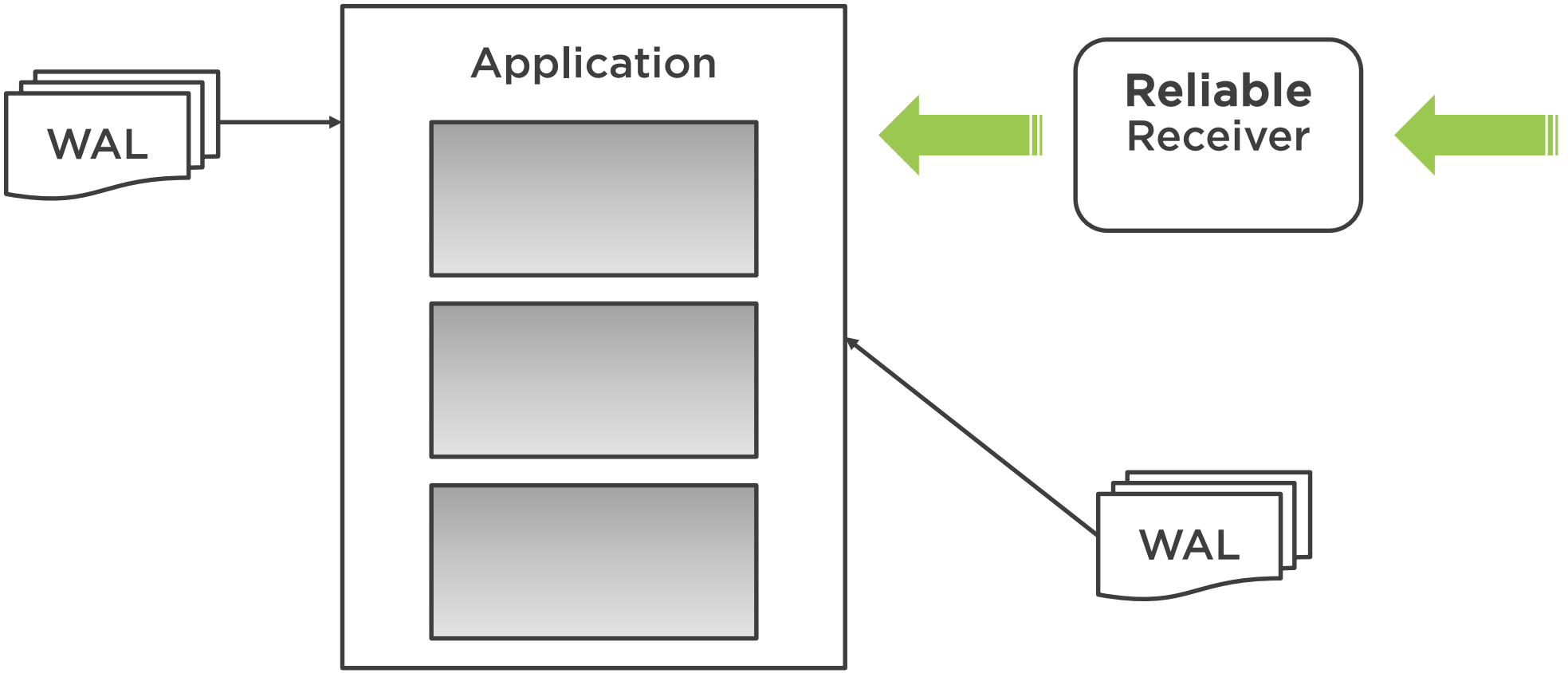
- Better Stream Recovery Options
- Optimizations
- Structured Streaming
- The Future: Spark 2.x

Increasing Stream Resiliency

Write Ahead Logging (WAL)



Write Ahead Logging (WAL)



Write Ahead Logging (WAL)

```
                    ceiver.writeAheadLog.enable
                    iver.writeAheadLog.closeFileAfterWrite
                    ceiver.writeAheadLog.closeFileAfterWrite
dStream.persist(StorageLevel.MEMORY_AND_DISK_SER)
```

Boosting Performance

Spark Streaming

- Batch Interval
  - “Total delay: ....”
  - Streaming UI
    - `spark.locality.wait`
    - Partitioning
  - Consistency
- Stream Rate Limiting
  - `spark.streaming.receiver.maxRate`
  - `spark.streaming.kafka.maxRatePerPartition`
  - `spark.streaming.backpressure.enabled`
    - http://bit.ly/2uciLaT
- Garbage Collection
  - `--XX:+UseConcMarkSweepGC`
- Serialized caching?
- Accumulators and Broadcast
  - .../streaming-programming-guide.html#accumulators-broadcast-variables-and-checkpoints

Spark SQL

Join Optimization

Query `size < spark.sql.autoBroadcastJoinThreshold`

```
                                                st(smallerDF), “key”)


BroadcastHashJoin … BuildRight
…
```

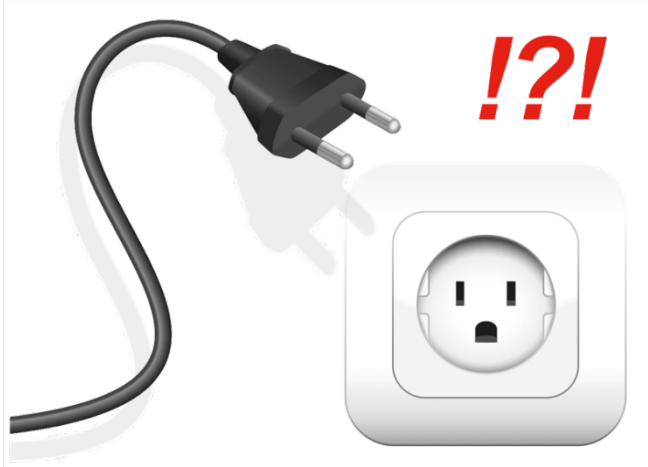
Spark SQL

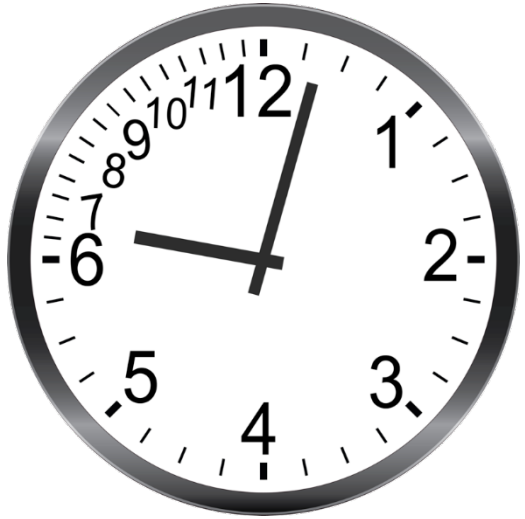
Tuning

- **Query Plan Review**
  - `df.queryExecution`
- **Memory Management Unification: SPARK-10000**
- `spark.sql.shuffle.partitions`
- **Tungsten Encoding!**
  - `spark.sql.inMemoryColumnarStorage.batchSize`

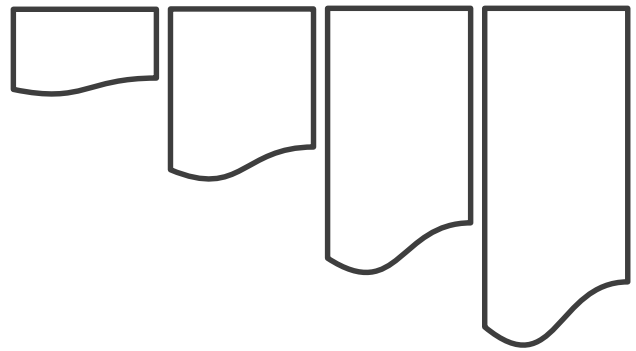
Structured Streaming

DStreams





1 →



Structured

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis

Structured Streams

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis

.....

Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex

.....

Nec	Mollis	Molestie	Nunc

.....

Structured Streaming

Batch Logic

```
s
...
...
...
.
```

Structured Streaming

Batch Logic

```
spark.read.format(…).load

…

…[processing logic]…

…

.write.format(…).save
```

Stream Logic

```
spark.readStream.format(…).load

…

…[processing logic]…

…

.writeStream.format(…).start
```

Structured Streaming

Batch Logic

```
spark.read.format(…).load

…

…[processing logic]…

…

.write.format(…).save
```

Stream Logic

```
spark.readStream.format(…).load

…

…[processing logic]…

…

.writeStream.format(…).start
```

Structured Streaming

Batch Logic

```
spark.read.format(…).load

…

…[processing logic]…

…

.write.format(…).save
```

Stream Logic

```
spark.readStream.format(…).load

…

…[processing logic]…

…

.writeStream.format(…).start
```

Structured Streaming

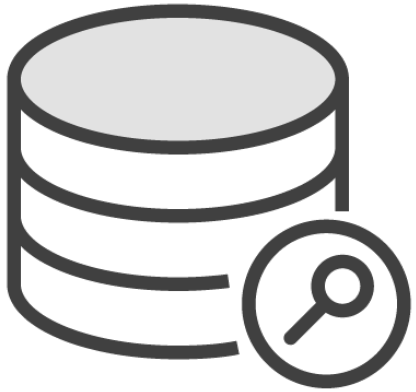
Batch Logic

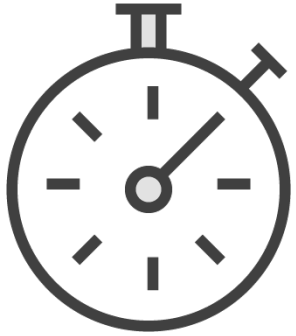
```
spark.read.format(…).load

…

…[processing logic]…

…

.write.format(…).save
```

Stream Logic

```
spark.readStream.format(…).load

…

…[processing logic]…

…

.writeStream.format(…).start
```

Structured Streaming





A Deeper Dive into Structured Streaming

Exactly Once Source to Sink

Files

Sockets TESTING ONLY

Exactly Once Source to Sink

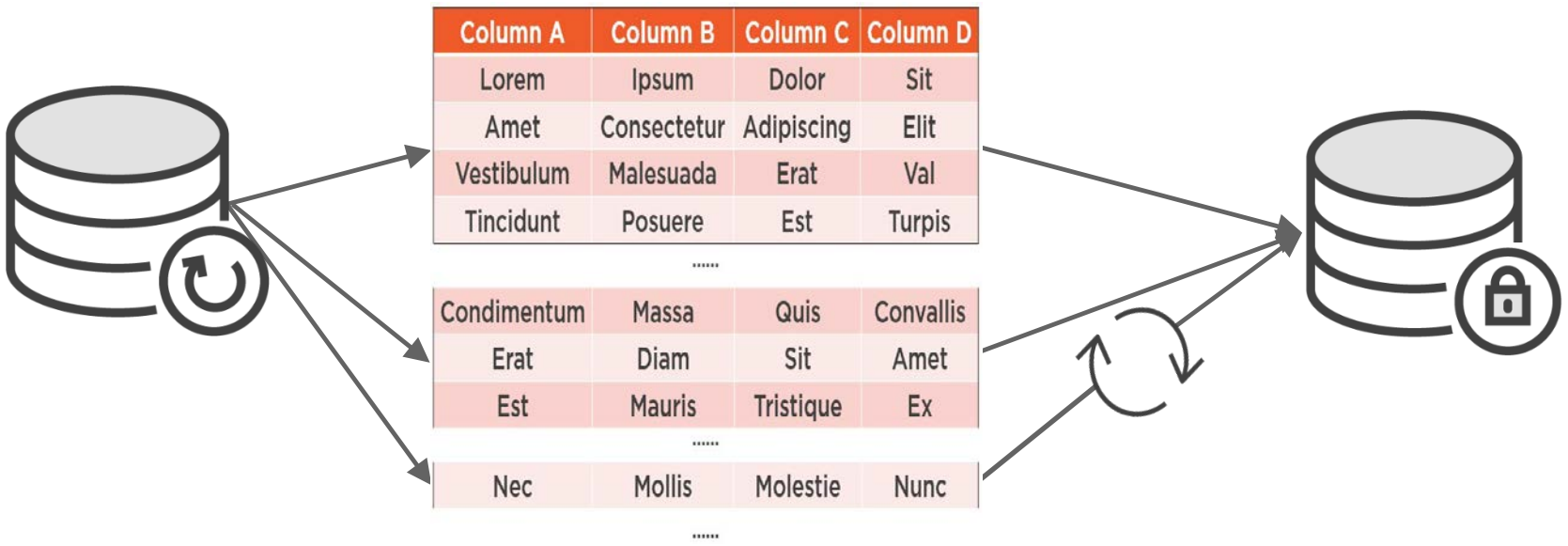
Files

Sockets

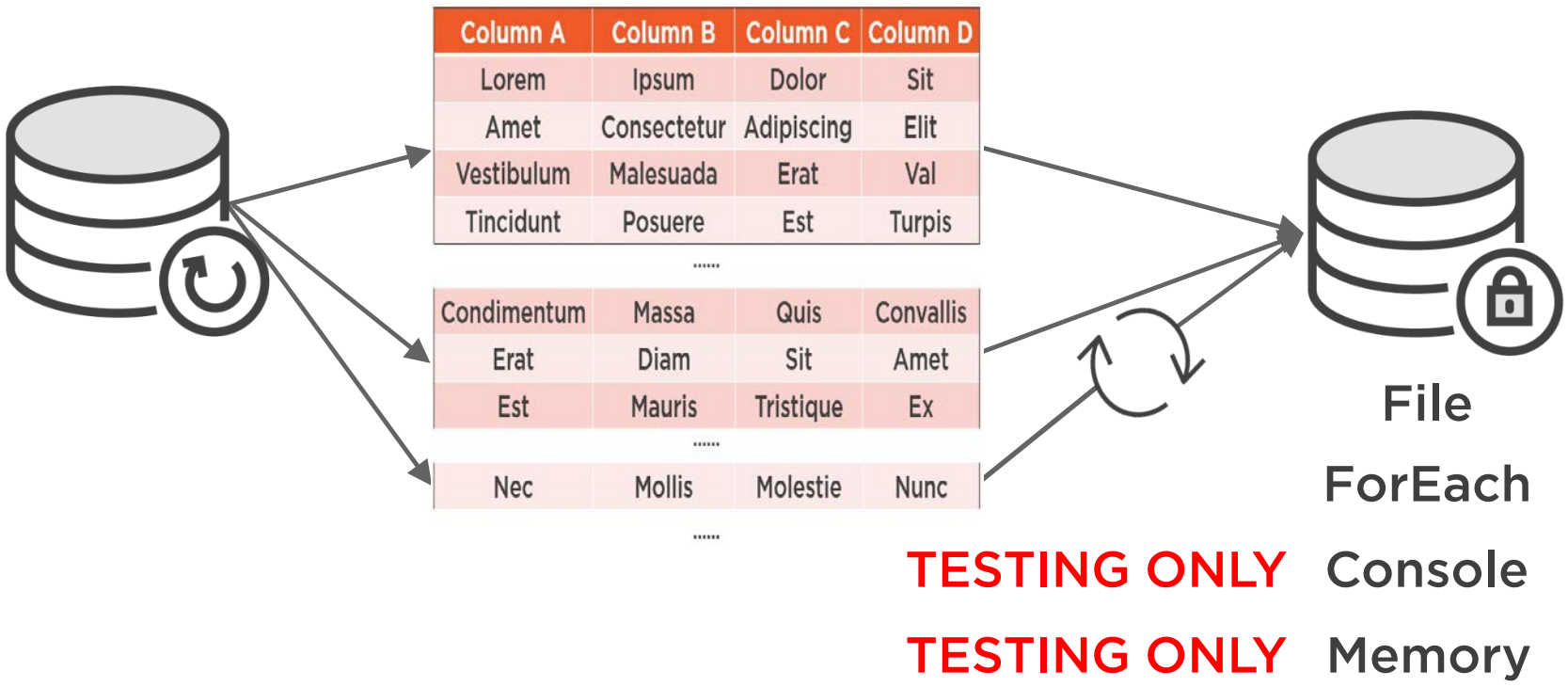
Kafka (Spark 2.1)

JDBC (Spark 2.?)

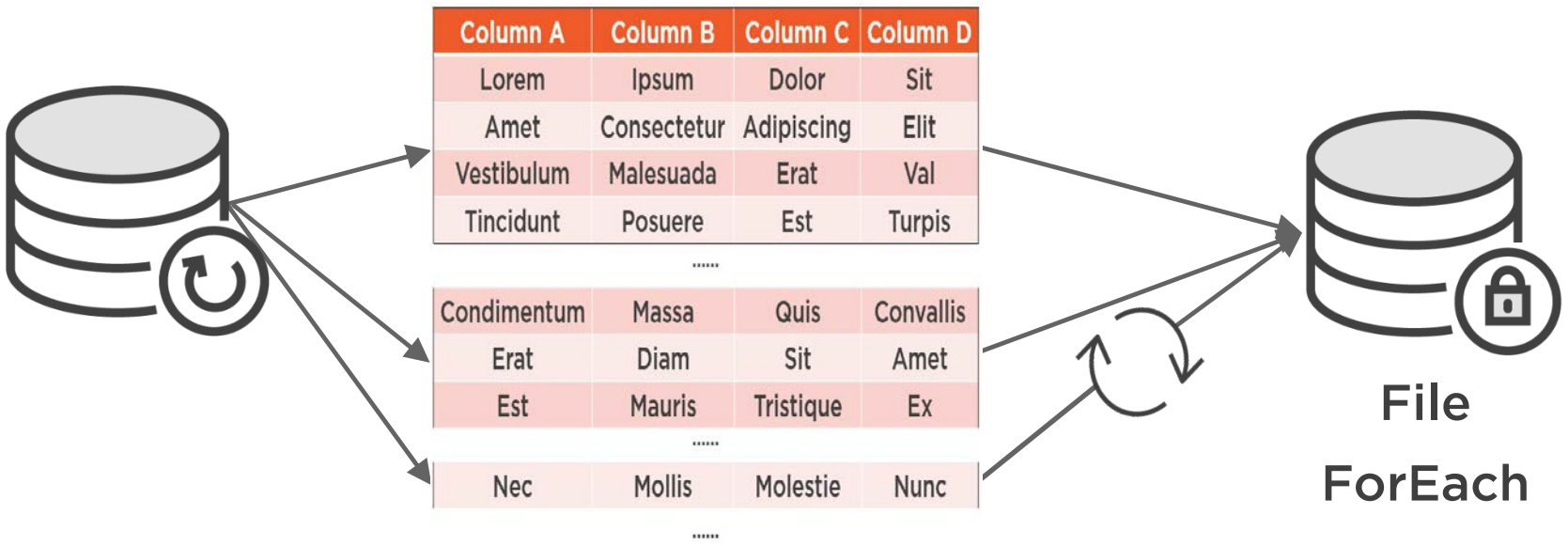
Exactly Once Source to Sink



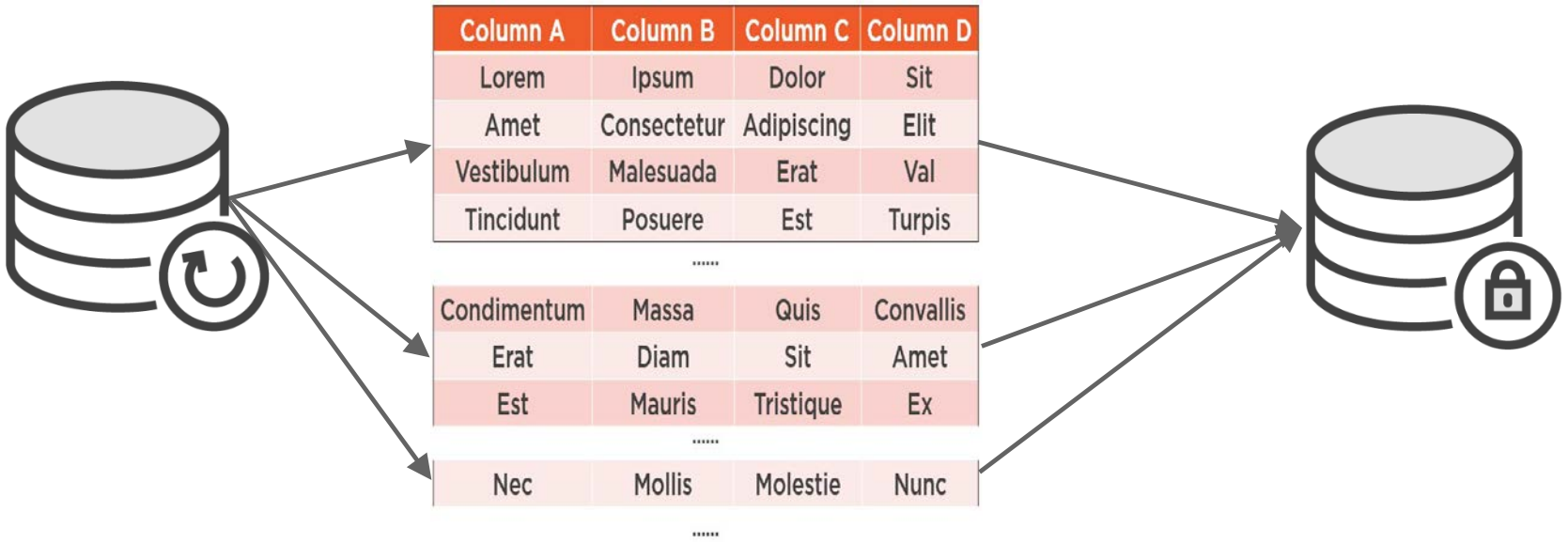
Exactly Once Source to Sink



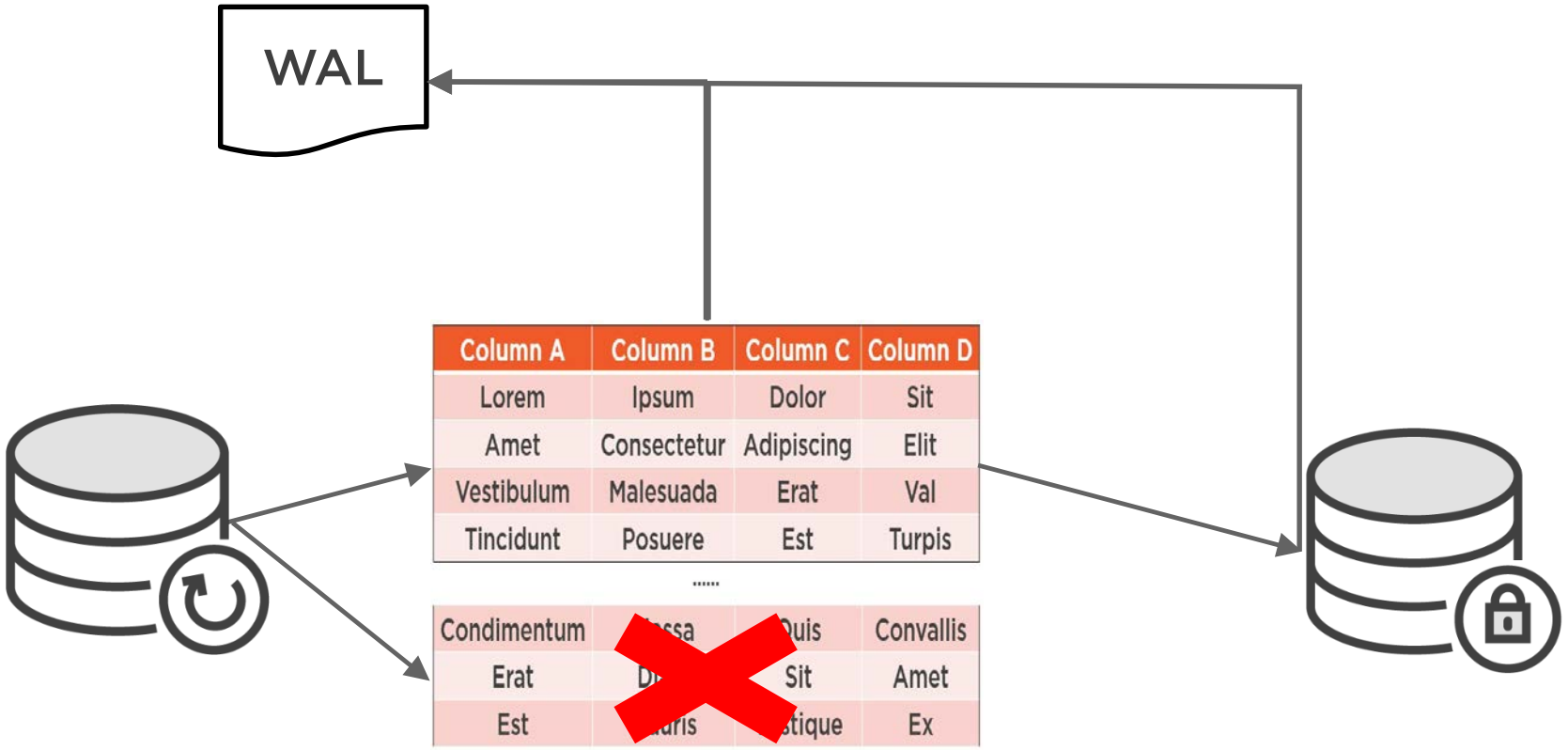
Exactly Once Source to Sink



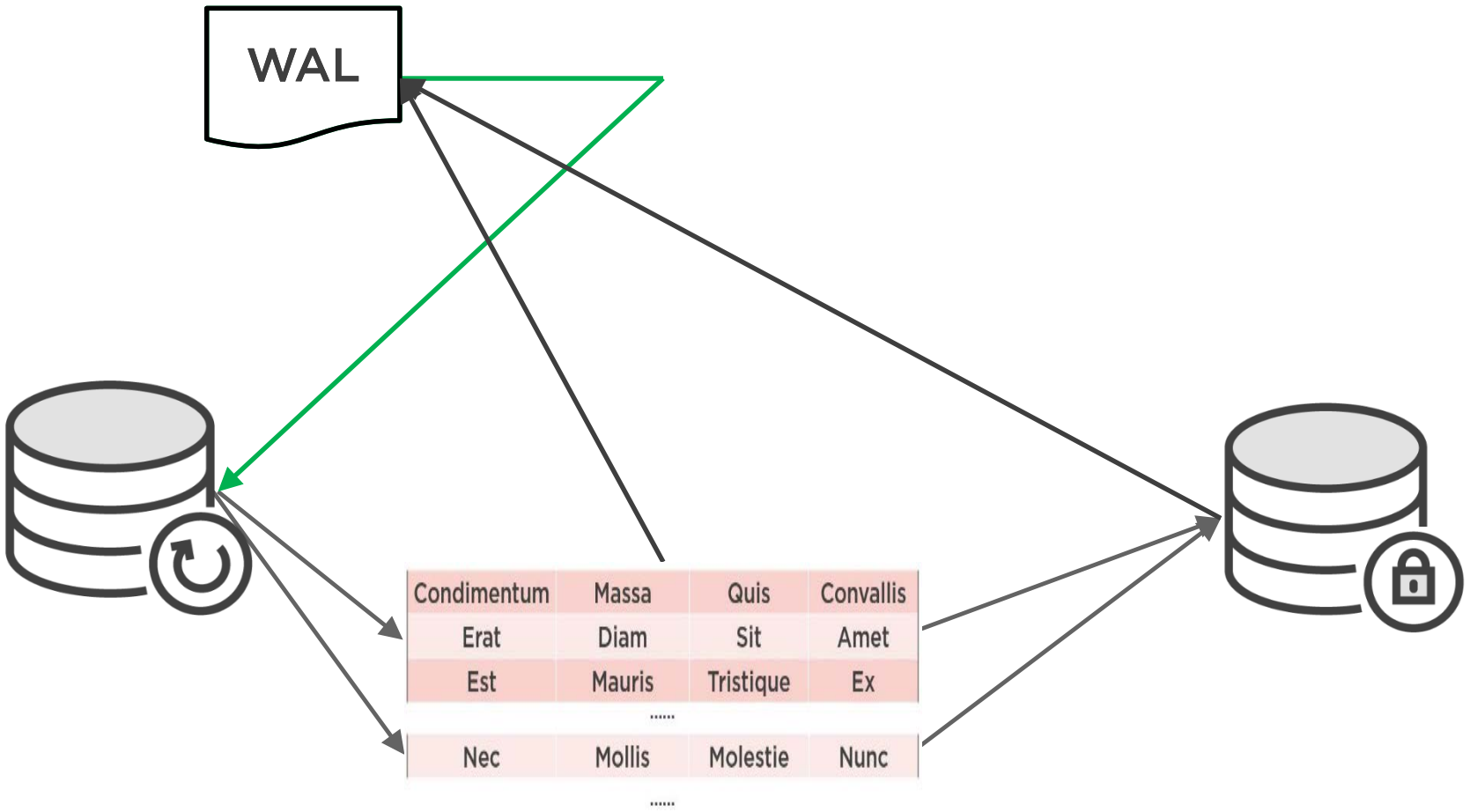
Recovery


Column A
Column B
Column C
Column D
Lorem
Ipsum
Dolor
Sit
Amet
Consectetur
Adipiscing
Elit
Vestibulum
Malesuada
Erat
Val
Tincidunt
Posuere
Est
Turpis
.....
Condimentum
Massa
Quis
Convallis
Erat
Diam
Sit
Amet
Est
Mauris
Tristique
Ex
.....
Nec
Mollis
Molestie
Nunc
.....

Recovery

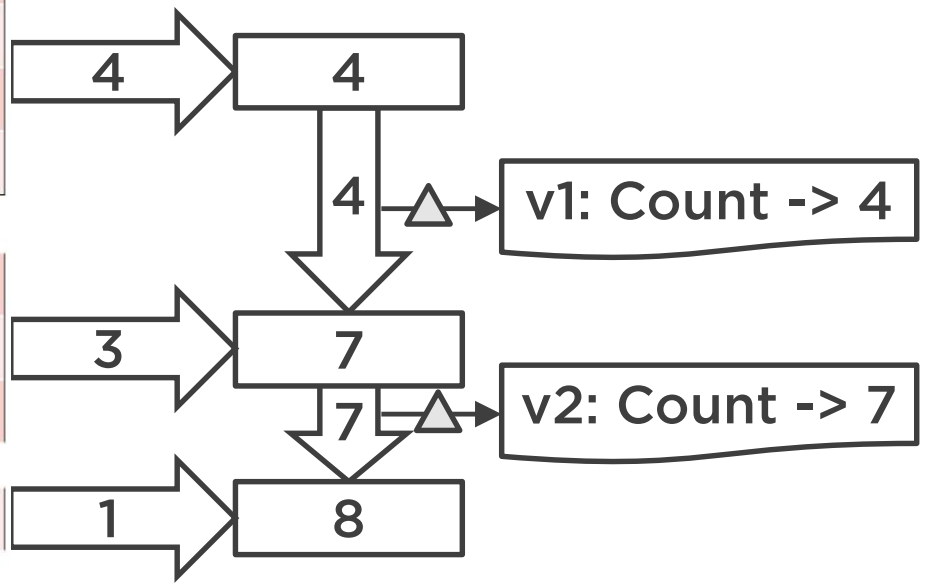

WAL
Column A
Column B
Column C
Column D
Lorem
Ipsum
Dolor
Sit
Amet
Consectetur
Adipiscing
Elit
Vestibulum
Malesuada
Erat
Val
Tincidunt
Posuere
Est
Turpis
.....
Condimentum
Quis
Convallis
Erat
Sit
Amet
Est
Ex

Recovery


WAL
Condimentum Massa Quis Convallis
Erat Diam Sit Amet
Est Mauris Tristique Ex
.....
Nec Mollis Molestie Nunc
.....

State

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis
.....			
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex
.....			
Nec	Mollis	Molestie	Nunc
.....			



State

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis
.....			
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex
.....			
Nec	Mo	Molestie	Nunc
.....			

v1: Count -> 4

v2: Count -> 7

State

WAL

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis
.....			
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex
.....			
Nec	Mollis	Molestie	Nunc
.....			

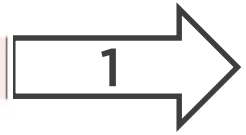
1

v1: Count -> 4

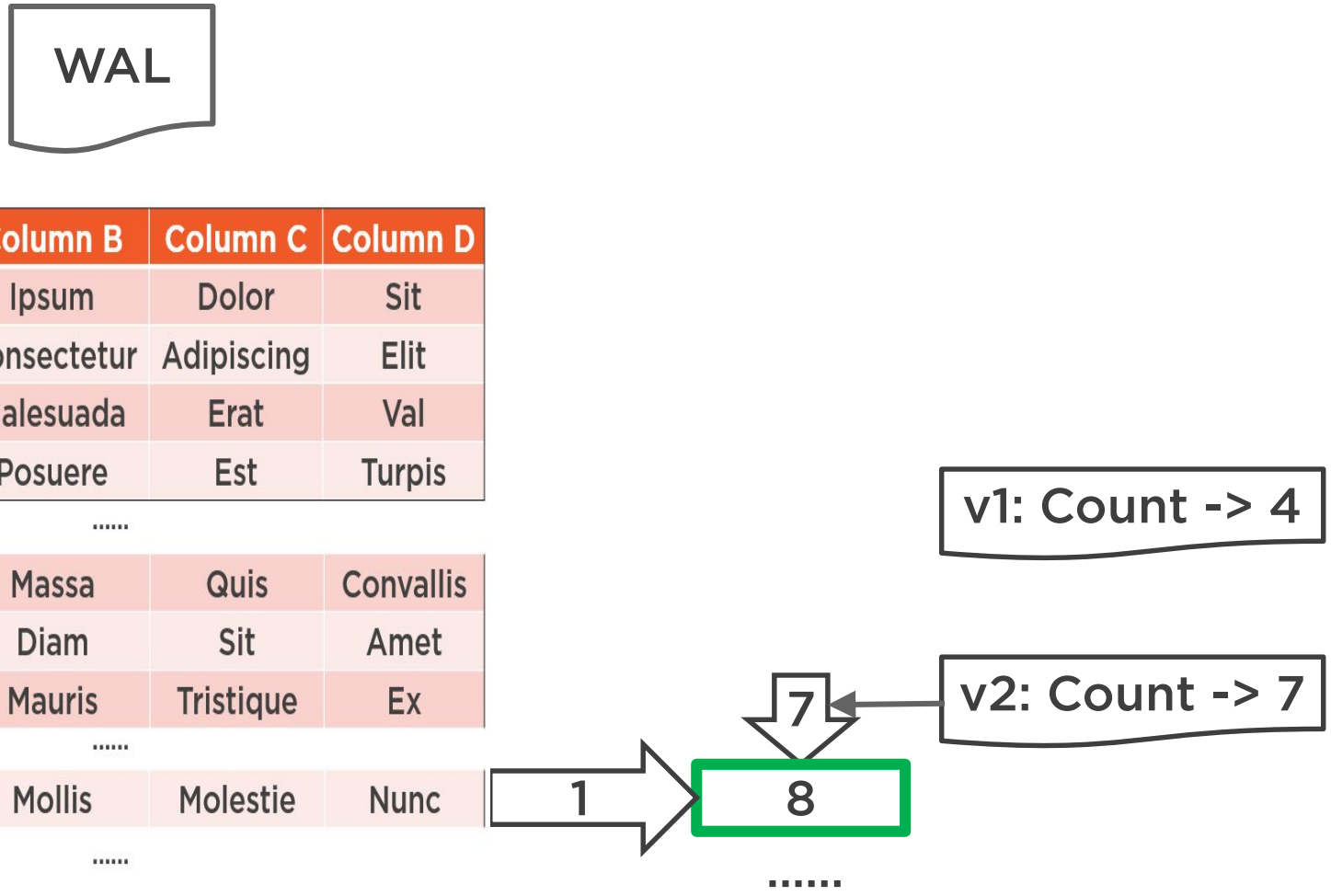
v2: Count -> 7

State

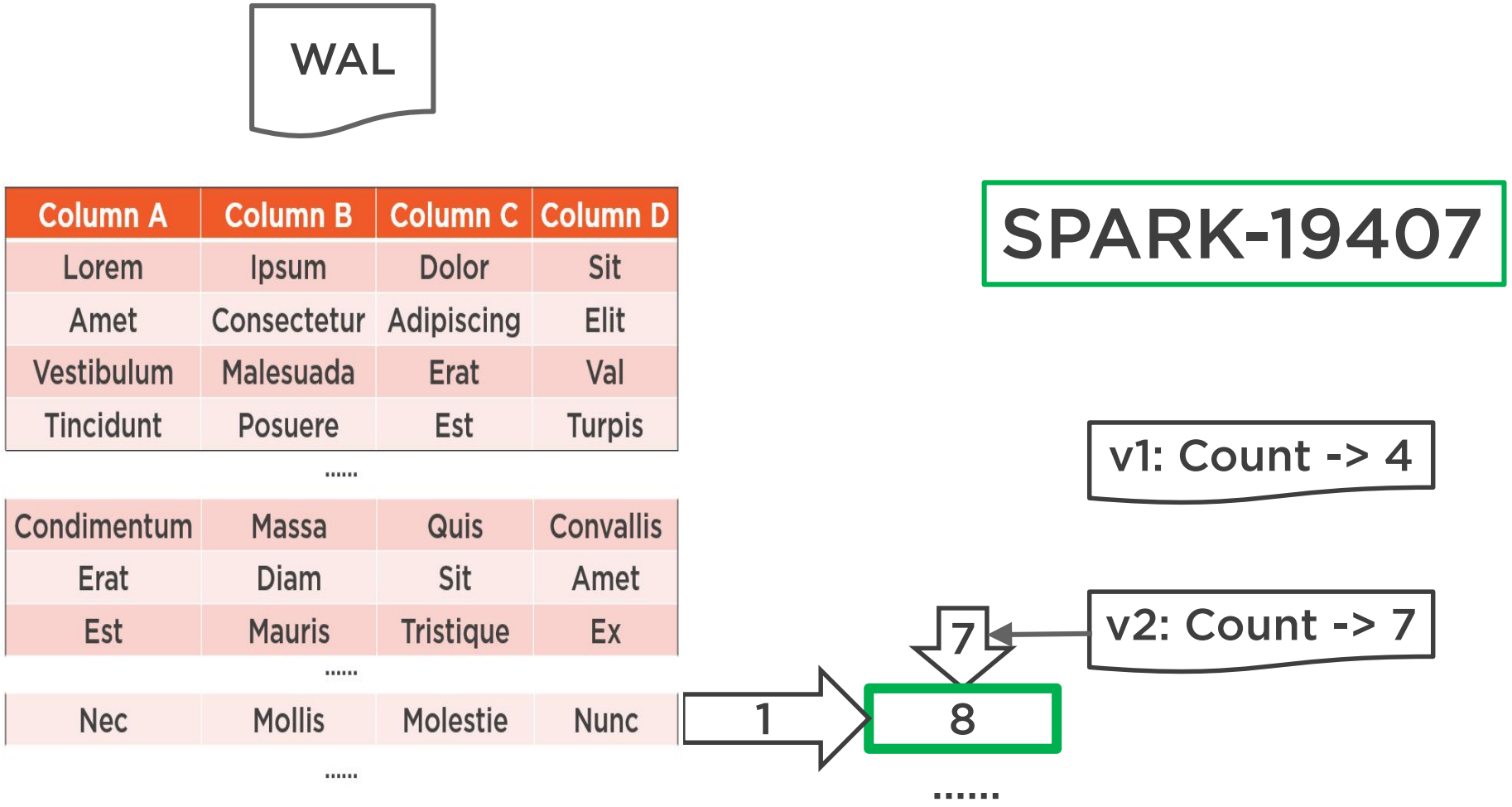
Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis
.....			
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex
.....			
Nec	Mollis	Molestie	Nunc
.....			



State

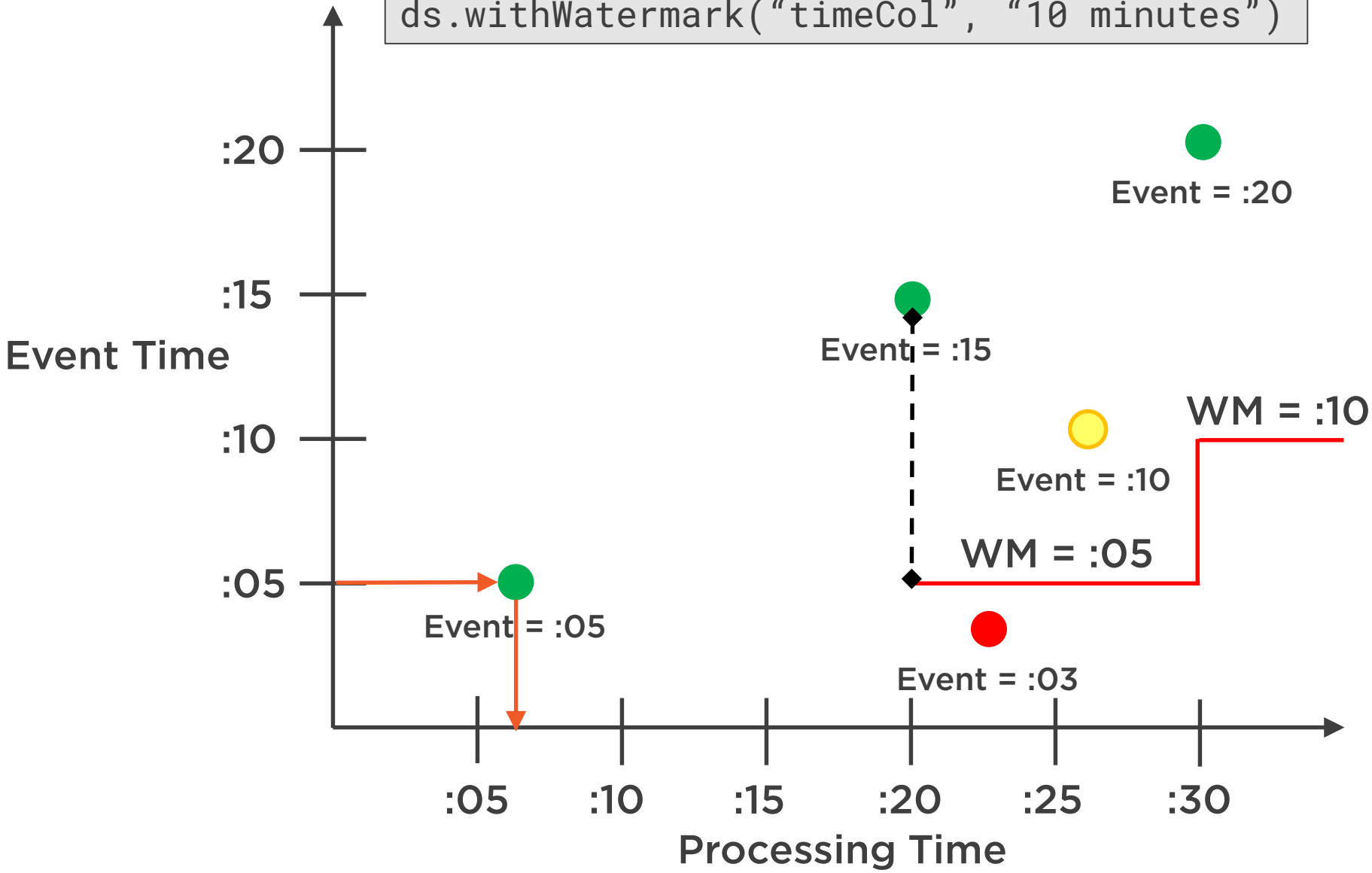

WAL
Column A
Column B
Column C
Column D
Lorem
Ipsum
Dolor
Sit
Amet
Consectetur
Adipiscing
Elit
Vestibulum
Malesuada
Erat
Val
Tincidunt
Posuere
Est
Turpis
.....
Condimentum
Massa
Quis
Convallis
Erat
Diam
Sit
Amet
Est
Mauris
Tristique
Ex
.....
Nec
Mollis
Molestie
Nunc
.....
1
7
8
.....
v1: Count -> 4
v2: Count -> 7

State


WAL
Column A
Column B
Column C
Column D
Lorem
Ipsum
Dolor
Sit
Amet
Consectetur
Adipiscing
Elit
Vestibulum
Malesuada
Erat
Val
Tincidunt
Posuere
Est
Turpis
.....
Condimentum
Massa
Quis
Convallis
Erat
Diam
Sit
Amet
Est
Mauris
Tristique
Ex
.....
Nec
Mollis
Molestie
Nunc
.....
SPARK-19407
v1: Count -> 4
v2: Count -> 7
7
1
8
.....

Watermark **Spark 2.1+**

```
ds.withWatermark("timeCol", "10 minutes")
```



Output Models

Append Mode

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis

Column A	Column B	Column C	Column D
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex

Column A	Column B	Column C	Column D
Nec	Mollis	Molestie	Nunc

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex
Nec	Mollis	Molestie	Nunc

Output Models

Complete Mode

Column A	Column B	Column C	Column D
Lorem	Ipsum	Dolor	Sit
Amet	Consectetur	Adipiscing	Elit
Vestibulum	Malesuada	Erat	Val
Tincidunt	Posuere	Est	Turpis

Count = 4

Column A	Column B	Column C	Column D
Condimentum	Massa	Quis	Convallis
Erat	Diam	Sit	Amet
Est	Mauris	Tristique	Ex

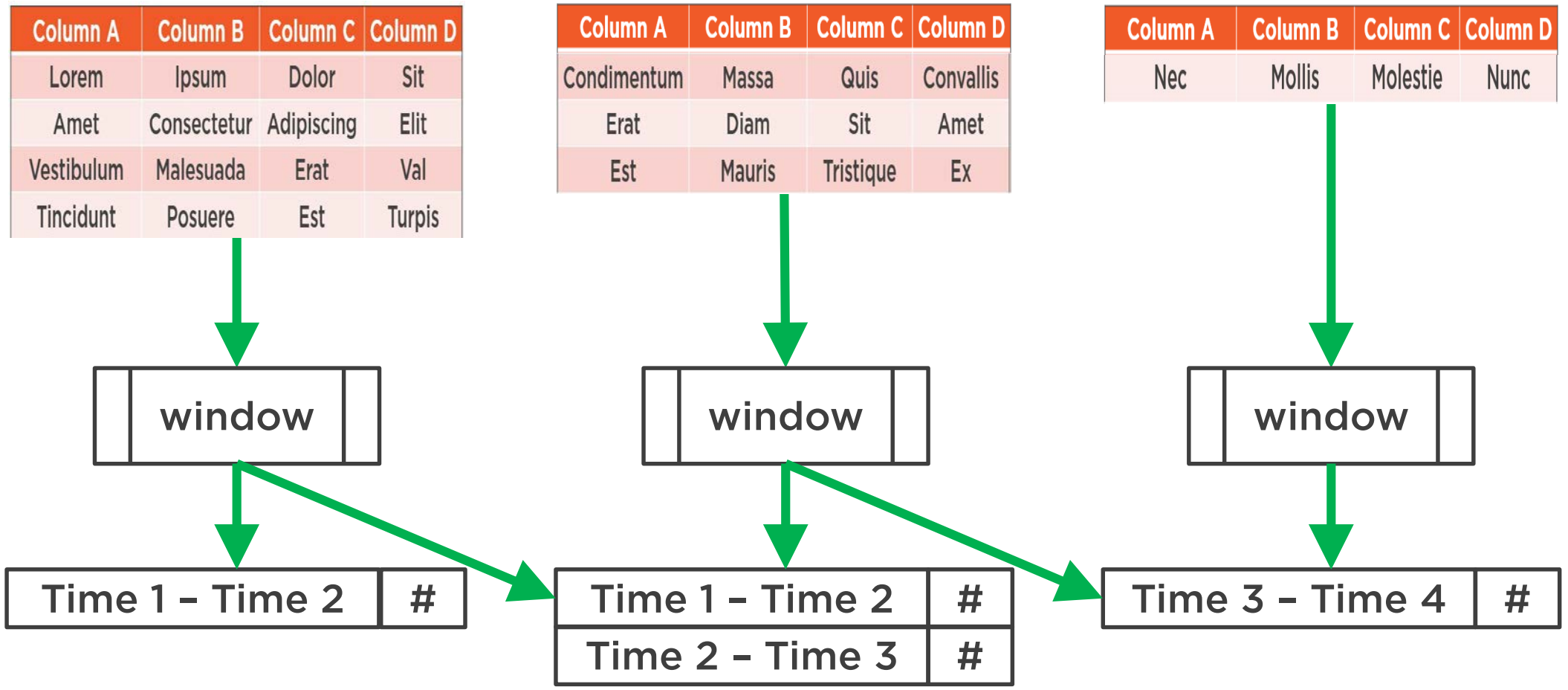
Count = 7

Column A	Column B	Column C	Column D
Nec	Mollis	Molestie	Nunc

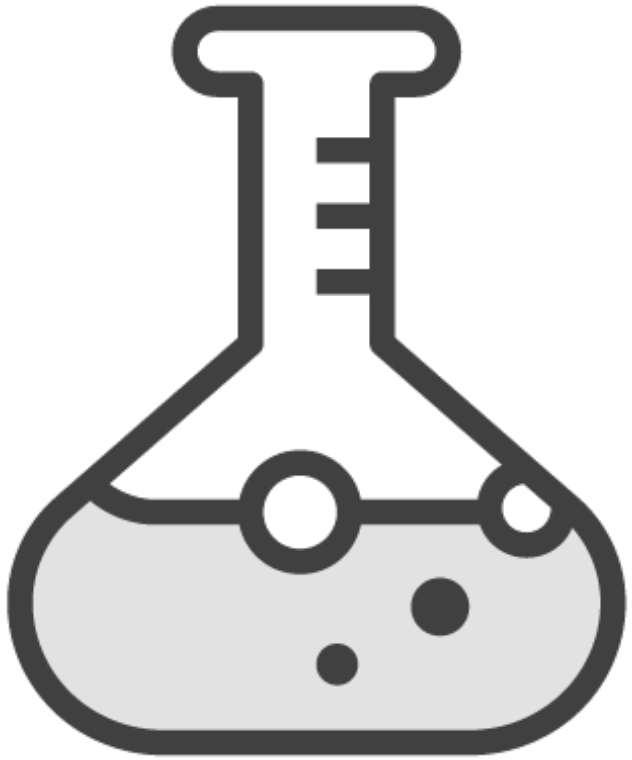
Count = 8

Output Models

Update Mode **Spark 2.1.1+**



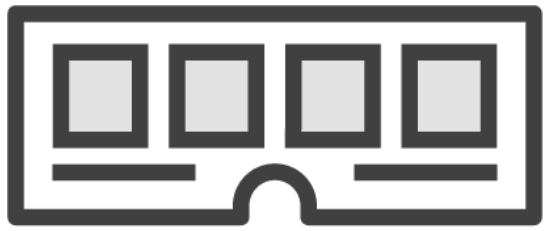
Structured Streaming

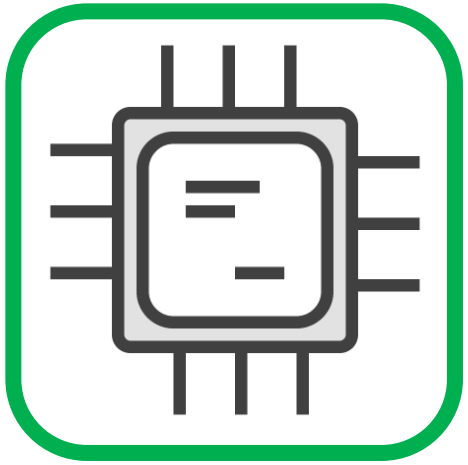


../structured-streaming-programming-guide.html#unsupported-operations

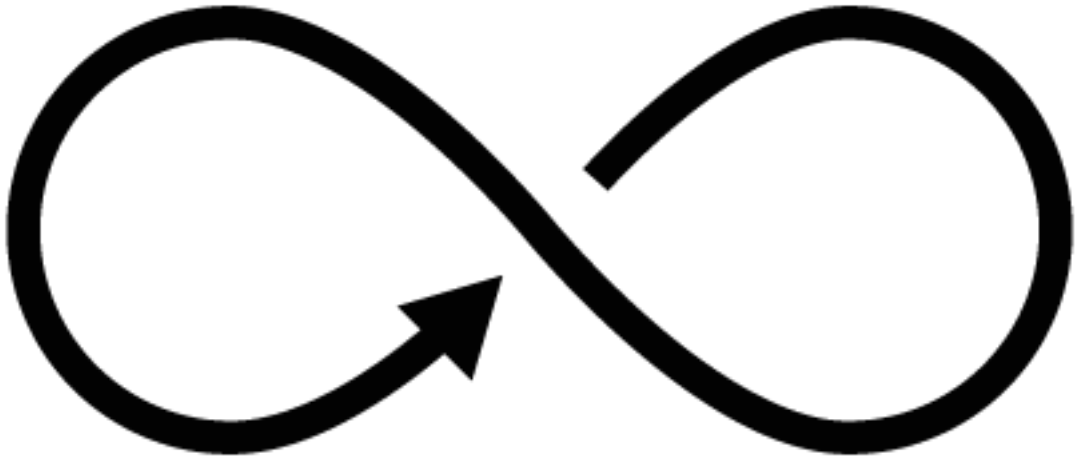
Spark 2.x

Focus





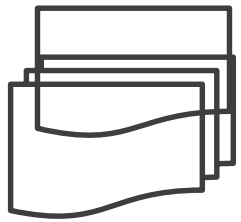
Focus



Spark 2.[0-1]

- AccumulatorV2
  - SPARK-14654
  - spark.apache.org/docs/latest/programming-guide.html#accumulators
- MIMA
  - github.com/apache/spark/blob/master/project/MimaExcludes.scala

Spark 2.[0-1]



Spark 2.[0-1]

Spark 2.[0-1]

Whole-stage Codegen


```
var count = 0
for(x in db) {
  if(x+1 > 0) {
    count = count + 1
  }
}
```

Spark 2.[0-1]

Whole-stage Codegen


```
var count = 0
for(x in db) {
  if(x+1 > 0) {
    count = count + 1
  }
}
```

Spark 2.[0-1]

Whole-stage Codegen


```
var count = 0
for(x in db) {
  if(x+1 > 0) {
    count = count + 1
  }
}
```

~5-100x

Coming 2.2+

- `mapGroupWithState` (SPARK-19067)
- EventTime based sessionization (SPARK-10816)
- Drizzle - https://github.com/amplab/drizzle-spark
  - SPARK-20928: Continuous Processing
- Structured Streaming, Structured Streaming, Structured Streaming...
- RISELab - https://rise.cs.berkeley.edu/

Resources

- Tuning Java Garbage Collection for Apache Spark Applications: Intel Big Data
  - databricks.com/blog/2015/05/28/tuning-java-garbage-collection-for-spark-applications
- SparkLint: a Tool for Monitoring, Identifying and Tuning Inefficient Spark Jobs: Simon Whitear
  - youtube.com/watch?v=reGerTzcvoA
- Problem Solving Recipes Learned from Supporting Spark: Justin Pihony
  - youtube.com/watch?v=Oq1_3BekIFE
- Structured Streaming Docs
  - spark.apache.org/docs/latest/structured-streaming-programming-guide
- Apache Spark as a Compiler: Databricks
  - databricks.com/blog/2016/05/23/apache-spark-as-a-compiler-joining-a-billion-rows-per-second-on-a-laptop
- Spark's Performance: The Past, Present, and Future: Sameer Agarwal
  - youtube.com/watch?v=RlbBPrWJEEM
- Drizzle: Low Latency Execution for Apache Spark: Shivaram Venkataraman
  - youtube.com/watch?v=Dm98w7gkKIA

Summary

Recovery Options

Optimizations

Structured Streaming

Spark 2.x