Transformations, Windowing, and Aggregations

Justin Post

Transformations, Windowing, and Aggregations

Justin Post

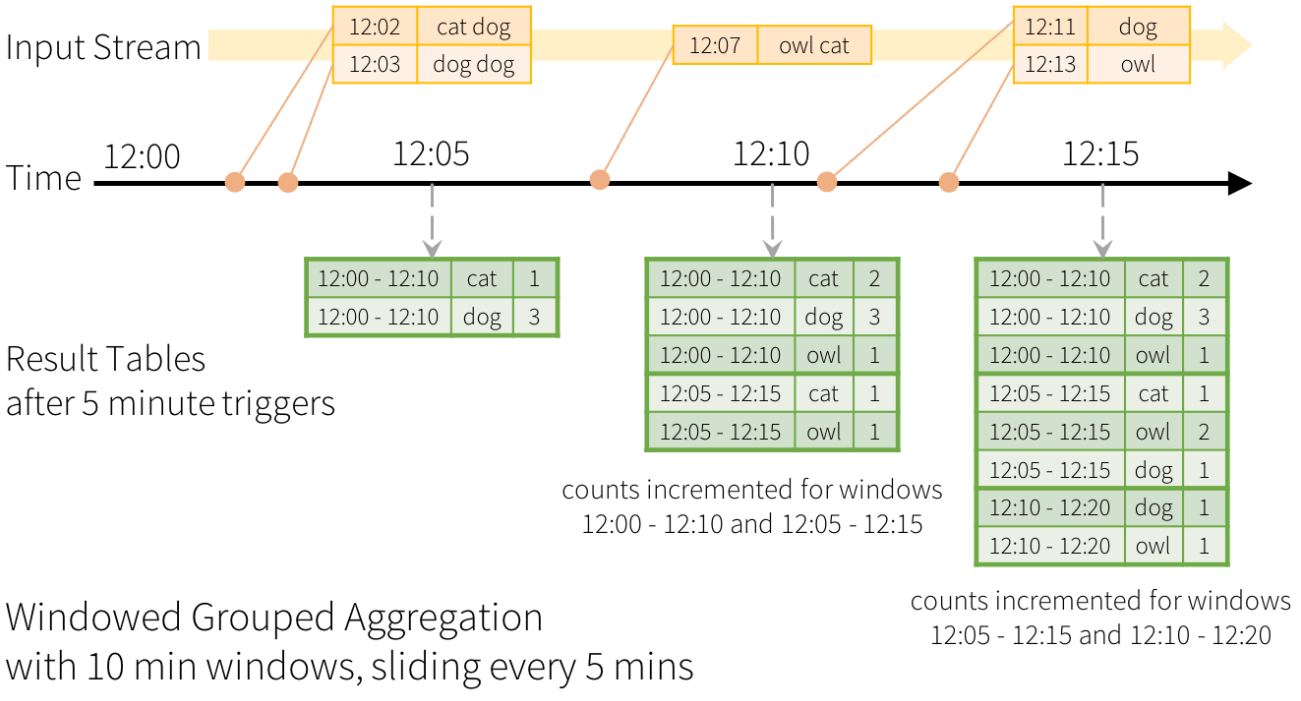
Recap

We'll use Spark Structured Streaming to handle our streaming data (Guide)

- Create a spark session

1. Read in a stream
    - Stream from a file, terminal, or use something like kafka
2. Set up transformations/aggregations to do (mostly using SQL type functions)
    - Perhaps over windows
3. Set up writing of the query to an output source
    - Console (for debugging)
    - File (say .csv)
    - Database
4. `query.start()` the query!
    - Continues listening until terminated (`query.stop()`)

Aggregations over an Event-time Window



https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html

Aggregations over an Event-time Window

- Need a **time stamp** or **event-time** on the data
  - Note: Event-time may be different than time the event is received by spark!

Aggregations over an Event-time Window

- Need a **time stamp** or **event-time** on the data
  - Note: Event-time may be different than time the event is received by spark!
- Easy to do windowing
  - Use `groupBy()` and specify the window size and update time

```
df.groupBy(
  window(df.timestamp, "1 minute", "30 seconds"), #2nd arg is window size, 3rd update time
  other_grouping_var_if_desired
  ).aggregation()
```

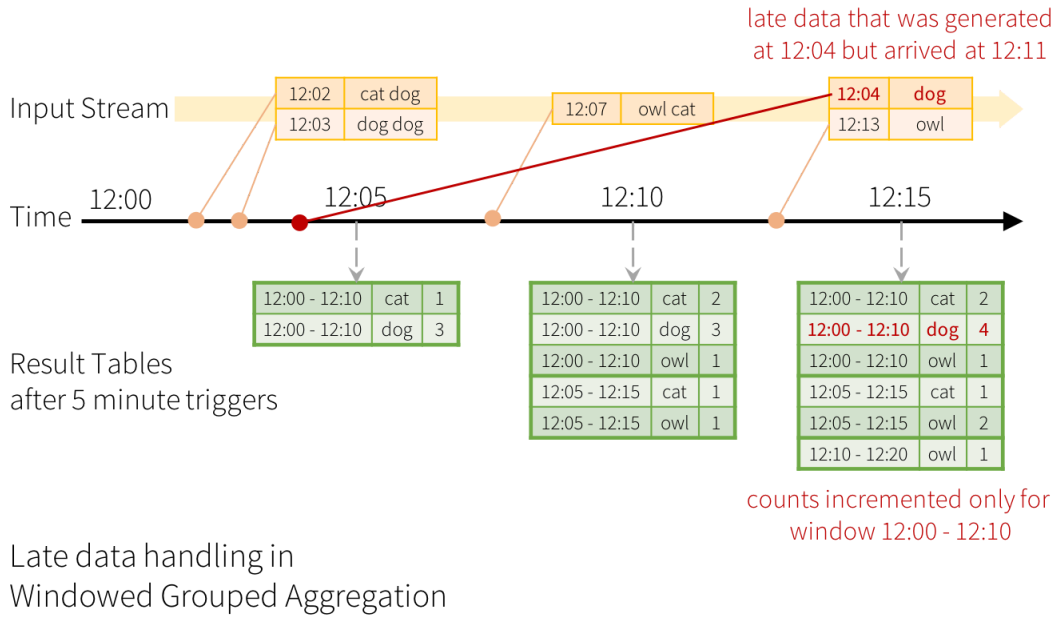
Aggregations over an Event-time Window

Let's jump into pyspark and aggregate some data over windows!

Late Data

Spark can also handle *late* data

- Event-time (time stamp) is when the event occurred
- Not always the same as when the data is received

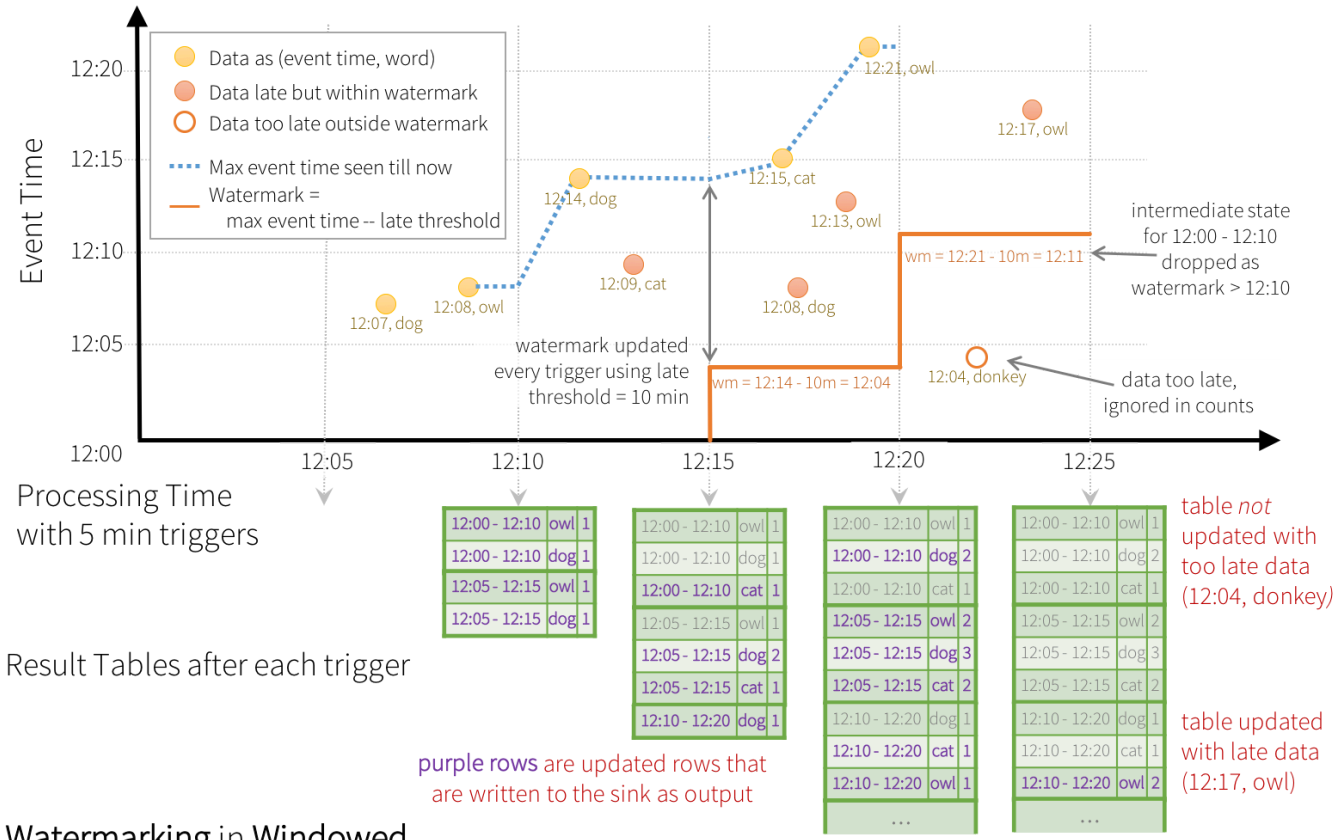


https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html

Watermarking

- Past Data is usually discarded once the event-time window for computation closes
- Can provide a **watermark**
  - Specifies a threshold on how late our event-time data can be
  - Data states maintained until the window plus that threshold is reached

Watermarking



Watermarking in **Windowed Grouped Aggregation** with **Update Mode**

Watermarking

- Past Data is usually discarded once the event-time window for computation closes
- Can provide a **watermark**
  - Specifies a threshold on how late our event-time data can be
  - Data states maintained until the window plus that threshold is reached

```
df \
  .withWatermark("timestamp", "20 seconds") \ #accept data 20 seconds past the close of the window
  .groupBy(
    window(df.timestamp, "1 minute", "30 seconds"), #2nd arg is window size, 3rd update time
    other_grouping_var_if_desired
  ) \
  .aggregation()
```

- Note: There are conditions for using watermarks given in the guide

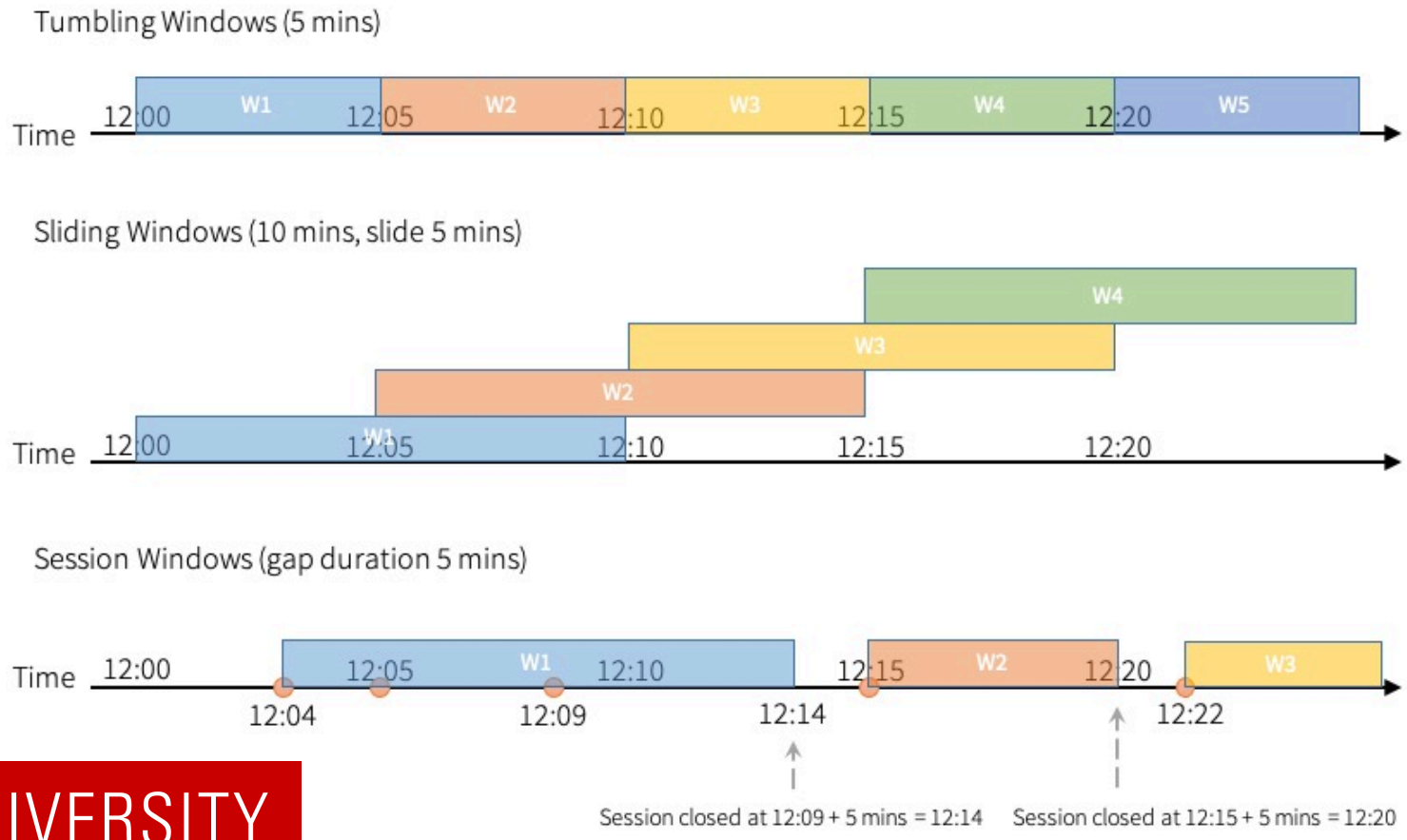
Watermarking

Let's jump back into pyspark and include a watermark!

This will allow us to write to a .csv file with our aggregation

Types of Time Windows

Three types of time windows in spark:



Recap

- Create a spark session

1. Read in a stream
    - Stream from a file, terminal, or use something like kafka
2. Set up transformations/aggregations to do (mostly using SQL type functions)
    - Perhaps over windows
    - Use a watermark to allow for late data
3. Set up writing of the query to an output source
    - Console (for debugging)
    - File (say .csv)
    - Database
4. `query.start()` the query!
    - Continues listening until terminated (`query.stop()`)