




Unique IDs with Causality


We'll cover the following




- Causality
- Use UNIX time stamps
 - Pros
 - Cons
- Twitter Snowflake
 - Pros
 - Cons
- Using logical clocks
 - Lamport clocks
 - Vector clocks
- TrueTime API
 - Pros
 - Cons
- Summary

Causality

In the [previous](#) lesson, we generated unique IDs to differentiate between various events. Apart from having unique identifiers for events, we're also interested in finding the sequence of these events. Let's consider an example where Peter and John are two Twitter users. John posts a comment (event A), and Peter replies to John's comment (event B). Event B is dependent on event A and can't happen before it. The events are not concurrent here. 

We can also have concurrent events—that is, two events that occur independently of each other. For example, if Peter and John comment on two different Tweets, there's no 

happened-before relationship or causality between them. It's essential to identify the dependence of one event over the other but not in the case of concurrent events.



Note: The scenario described above can also be handled by assigning a unique ID and encoding the dependence of events using a social graph. We might also use a separate time data structure and a simple unique ID. However, we want a unique ID to do double duty—provide unique identification and also help with the causality of events.

The following slides provide a visualization of concurrent and nonconcurrent events.

