Media Service: Kinesis

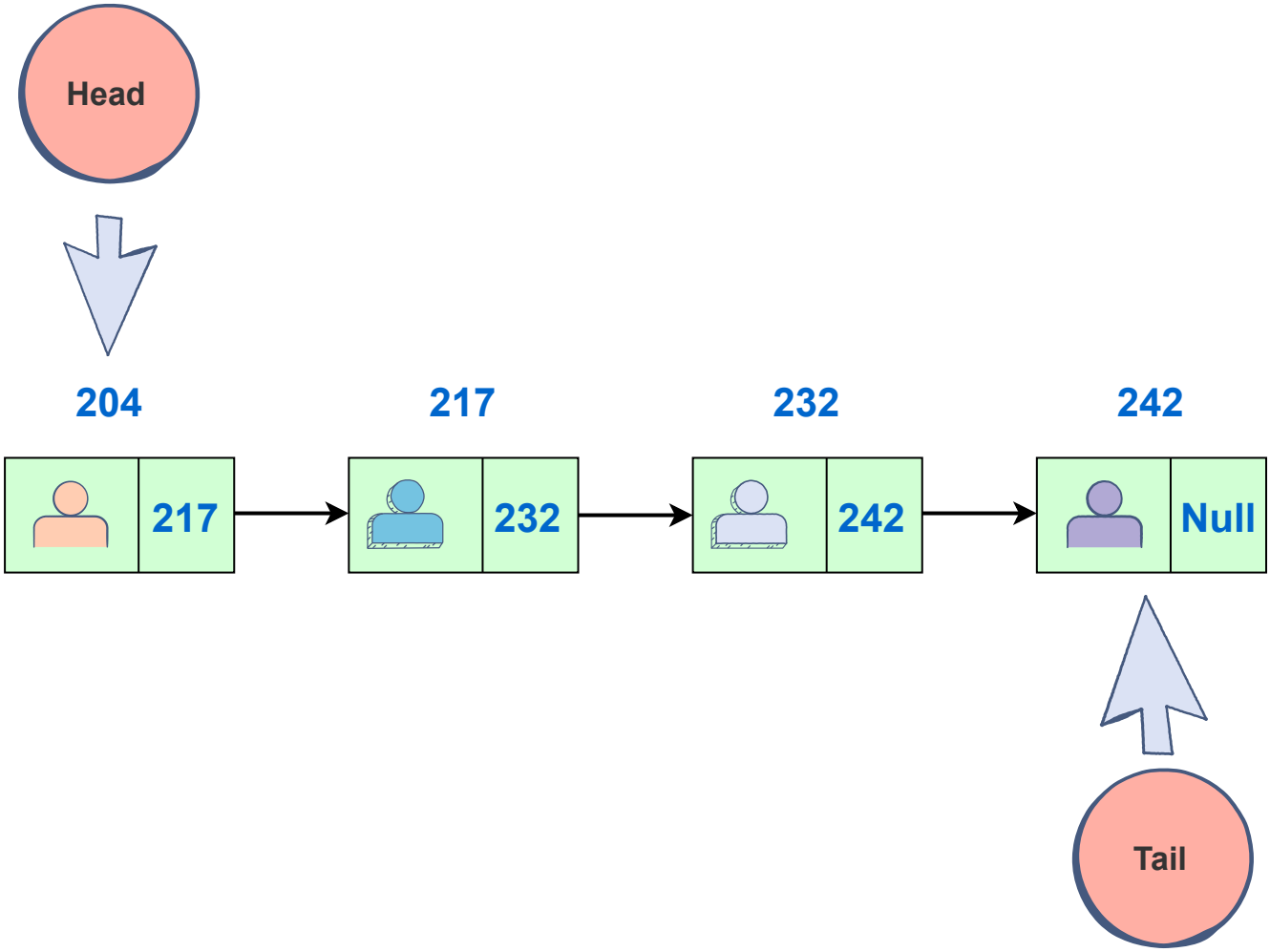
Kinesis: A linked list in the cloud and it's pros and cons will be discussed in this lesson.

We'll cover the following

- Kinesis vs SQS
- Cheaper/cost effective
- Not easy to use

A linked list in the cloud

You can think of a Kinesis stream as a highly-durable linked list in the cloud. The use cases for Kinesis are often similar to those of SQS—you would typically use either Kinesis or SQS when you want to enqueue records for asynchronous processing.



Linkedlist of people's data stored in memory

Kinesis vs SQS

The main difference between the two services is:

- SQS can only have one consumer, while Kinesis can have many.
- Once an SQS message gets consumed, it gets deleted from the queue. But Kinesis records get added to a list in a stable order, and any number of consumers can read a copy of the stream by keeping a cursor over this never-ending list.
- Multiple consumers don't affect each other, and if one falls behind, it doesn't slow down the other consumers.
- Whenever consumers read data out of Kinesis, they will always get their records in the same order.

Cheaper/cost effective

In addition to supporting multiple consumers, another benefit of Kinesis over SQS

is that it can be a lot cheaper. For example:

- Putting 1 KB messages in SQS at an average rate of 500 messages per second will cost you $34.56 per day. A Kinesis stream with 50% capacity headroom can handle that same volume for just $0.96 per day. So there can be about a massive difference in cost.

> **Let's do a fun exercise:** Use the below widget to calculate the cost of putting a certain size (KB) message at an average rate of certain number of messages per second per day to a Kinesis stream. $0.02 is the cost of putting a million messages to a Kinesis stream per day. You just have to put the *number* of **messages** per second, and the size of each message in KB. It will automatically generate the cost per day in the next cell.

No. of messages per second	Size of each message (KB)	Cost of using Kinesis stream per day ($)
1000	1	f 1.7280000000000002

The reason for this cost profile is simple:

- Kinesis streams are optimized for sequential reads and sequential writes.
- Records get added to the end of a file and read always happens sequentially from a pointer on that file.
- Unlike SQS, records in a Kinesis stream don't get deleted when consumed, so it's a pure append-only data structure behind the scenes.
- Data simply ages out of a Kinesis stream once it exceeds its retention period, which is 24 hours by default.

> **NOTE:** It is very important to note that the prices mentioned above are only for the purpose of understanding cost comparisons. These prices are subject to change anytime by AWS. Therefore, the most current prices should be referenced for final business decisions.

Not easy to use

On the other hand, Kinesis is not as easy to use as SQS. While with SQS you can simply enqueue as many messages as you want without having to worry about capacity or throttling limits, a Kinesis stream is made up of slices of capacity called shards, and it's up to you to figure out how many shards you need, monitor shard utilization, add shards, and figure out the best way to route records to shards so that they get approximately an equivalent amount of traffic. And unfortunately, all of this is a significant operational burden.

> That said, Kinesis can still be much easier to use when compared to other self-hosted technologies that provide the same streaming properties.

Once a Kinesis message gets consumed, it gets deleted from the queue as opposed to SQS where it is appended to the end of the file.

Retake Quiz

In the next lesson, we will take a small quiz to test the concepts you have learned so far in this first chapter.