Why should I use Amazon Kinesis and not SNS-SQS?

Asked 10 years, 1 month ago Modified 5 months ago

Viewed 128k times Part of AWS Collective

244

I have a use case where there will be stream of data coming and I cannot consume it at the same pace and need a buffer. This can be solved using an SNS-SQS queue. I came to know the Kinesis solves the same purpose, so what is the difference? Why should I prefer (or should not prefer) Kinesis?

amazon-web-services amazon-sqs amazon-kinesis

Share

Improve this question

Follow

asked Oct 29, 2014 at 6:01

Apoorv

2,725 ● 2 ● 15 ● 18

13 Answers Sorted by: Highest score (default)

103

Keep in mind this answer was correct for Jun 2015

After studying the issue for a while, having the same question in mind, I found that SQS (with SNS) is preferred for most use cases unless the order of the messages is important to you (SQS doesn't guarantee FIFO on messages). (EDIT: SQS now do support FIFO queue, but then you can have on 1 consumer at a time for the queue per topic, more about it in [fifo-topic-message-ordering])

There are 2 main advantages for Kinesis:

1. you can read the same message from several applications
2. you can re-read messages in case you need to.

Both advantages can be achieved by using SNS as a fan out to SQS. That means that the producer of the message sends only one message to SNS, Then the SNS fans-out the message to multiple SQSs, one for each consumer application. In this way you can have as many consumers as you want without thinking about sharding capacity.

Moreover, we added one more SQS that is subscribed to the SNS that will hold messages for 14 days. In normal case no one reads from this SQS but in case of a bug that makes us want to rewind the data we can easily read all the messages from this SQS and re-send them to the SNS. While Kinesis only provides a 7 days retention.

In conclusion, SNS+SQSs is much easier and provides most capabilities. IMO you need a really strong case to choose Kinesis over it.

Share Improve this answer Follow

edited Jul 1 at 7:36

answered Jun 16, 2015 at 10:25

Roee Gavirel
19.4k ● 13 ● 68 ● 97

3 FYI: You can have Kinesis retain for up to 7 days. – Didier A. Mar 22, 2016 at 22:25

40 Recently, AWS has announced SQS FIFO [docs.aws.amazon.com/AWSSimpleQueueService/latest/… which can serve the time-ordering of messages. – VijeshJain Dec 28, 2016 at 21:54

11 super minor comment - probably wouldn't use the word `split` in `SNS split the message to multiple SQSs` since it doesn't break down messages into pieces but copies it to multiple destinations. – Mobigital May 3, 2017 at 1:11

6 Kinesis is unsuitable for fan-out (pub-sub) use-cases due to limitations around the number of readers per shard/second. Whilst not relevant to the original enquiry, anyone relying on Kinesis scaling to n-readers should take this fact into consideration. forums.aws.amazon.com/message.jspa?messageID=760351 – codeasone Jun 7, 2017 at 14:07

7 Kinesis's order guarantee is per-shard, not per-stream. Once you have more than one shards, the whole stream would have no guarantee on order. For a SQS queue, when the throughput is relatively low, it is almost FIFO. Only when your throughput goes higher, the order is less followed. This

is regarding classic SQS queues, not FIFO queues. – yoroto Nov 13, 2018 at 22:30

75

Semantics of these technologies are different because they were designed to support different scenarios:

- SNS/SQS: the items in the stream **are not** related to each other
- Kinesis: the items in the stream **are** related to each other

Let's understand the difference by example.

1. Suppose we have a stream of orders, for each order we need to reserve some stock and schedule a delivery. Once this is complete, we can safely remove the item from the stream and start processing the next order. We are fully *done* with the previous order before we start the next one.
2. Again, we have the same stream of orders, but now our goal is to group orders by destinations. Once we have, say, 10 orders to the same place, we want to deliver them together (delivery optimization). Now the story is different: when we get a new item from the stream, we cannot finish processing it; rather we "wait" for more items to come in order to meet our goal. Moreover, if the processor process crashes, we must "restore" the state (so no order will be lost).

Once processing of one item cannot be separated from processing another one, we must have Kinesis semantics in order to handle all the cases safely.

Share Improve this answer Follow

edited Apr 7, 2018 at 15:43

answered Nov 2, 2017 at 20:58

Konstantin Triger
1,721 ● 15 ● 13

With SQS FIFO queue, we would have the messages ordered as they are sent. Does that make SQS similar to Kinesis on this aspect? – Andy Dufresne Sep 17, 2018 at 13:21

2 @AndyDufresne: this covers well the scenario where order is important. In the case (1) above, you may want to handle orders "in order". So if you run out of stock, later orders are rejected or delayed. FIFO semantics does not solve the core relativity (grouping) problem. – Konstantin Triger Sep 18, 2018 at 13:46

Great answer, thanks. I have a question about “if the processor process crashes, we must "restore" the state (so no order will be lost)” - isn’t this also achieved by SQS messages automatically reappearing if they aren’t explicitly removed? – Juraj Martinka Sep 25, 2023 at 7:26

@JurajMartinka: yes, both technologies support crash recovery for their primary use cases - single or related messages processing. – Konstantin Triger Sep 25, 2023 at 20:06

65

On the surface they are vaguely similar, but your use case will determine which tool is appropriate. IMO, if you can get by with SQS then you should - if it will do what you want, it will be simpler and cheaper, but here is a better explanation from the AWS FAQ which gives examples of appropriate use-cases for both tools to help you decide:

FAQ's

Share Improve this answer

Follow

edited Nov 19, 2015 at 8:03

Bazzinga...

1,004 ● 2 ● 17 ● 26

answered Oct 29, 2014 at 9:34

E.J. Brennan

46.8k ● 8 ● 93 ● 119

61

Kinesis support multiple consumers capabilities that means same data records can be processed at a same time or different time within 24 hrs at different consumers, similar behavior in SQS can be achieved by writing into multiple queues and consumers can read from multiple queues. However writing again into multiple queue will add sub seconds {few milliseconds} latency in system.

Second, Kinesis provides routing capability to selective route data records to different shards using partition key which can be processed by particular EC2 instances and can enable micro batch calculation {Counting & aggregation}.

Working on any AWS software is easy but with SQS is easiest one. With Kinesis, there is a need to provision enough shards ahead of time, dynamically increasing number of shards to manage spike load and decrease to save cost also required to manage. it's pain in Kinesis, No such things are required with SQS. SQS is infinitely scalable.

Share Improve this answer

Follow

edited Oct 10, 2015 at 20:10

Rohit Banga

18.9k ● 31 ● 118 ● 194

answered Nov 10, 2014 at 5:11

kartik

2,135 ● 3 ● 21 ● 31

14 Regarding you explanation on the SQS. You can achieve an easy way to send the same message to multiple SQSs by having an SNS before them. – Roee Gavirel Jun 16, 2015 at 10:15

8 app --> sns topic ---> sqs1, sqs2, sqs3... ? – kartik Jun 16, 2015 at 10:45

4 Yes, I was referring to this exactly approach. – Roee Gavirel Jun 17, 2015 at 8:29

@RoeeGavirel what about the request/second limitations for sns api? – Barbaros Alp Oct 27, 2016 at 8:02

@BarbarosAlp - I only aware of SMS (mobile text messages) limitation which is off topic here. this is the official documentation: docs.aws.amazon.com/general/latest/gr/… – Roee Gavirel Oct 30, 2016 at 12:32

51

Excerpt from AWS Documentation:

> **We recommend Amazon Kinesis Streams for use cases with requirements that are similar to the following:**
>
> - Routing related records to the same record processor (as in streaming MapReduce). For example, counting and aggregation are simpler when all records for a given key are routed to the same record processor.
> - Ordering of records. For example, you want to transfer log data from the application host to the processing/archival host while maintaining the order of log statements.
> - Ability for multiple applications to consume the same stream concurrently. For example, you have one application that updates a real-time dashboard and another that archives data to Amazon Redshift. You want both applications to consume data from the same stream concurrently and independently.
> - Ability to consume records in the same order a few hours later. For example, you have a billing application and an audit application that runs a few hours behind the billing application. Because Amazon Kinesis Streams stores data for up to 7 days, you

can run the audit application up to 7 days behind the billing application.

We recommend Amazon SQS for use cases with requirements that are similar to the following:

- Messaging semantics (such as message-level ack/fail) and visibility timeout. For example, you have a queue of work items and want to track the successful completion of each item independently. Amazon SQS tracks the ack/fail, so the application does not have to maintain a persistent checkpoint/cursor. Amazon SQS will delete acked messages and redeliver failed messages after a configured visibility timeout.
- Individual message delay. For example, you have a job queue and need to schedule individual jobs with a delay. With Amazon SQS, you can configure individual messages to have a delay of up to 15 minutes.
- Dynamically increasing concurrency/throughput at read time. For example, you have a work queue and want to add more readers until the backlog is cleared. With Amazon Kinesis Streams, you can scale up to a sufficient number of

shards (note, however, that you'll need to provision enough shards ahead of time).

- Leveraging Amazon SQS's ability to scale transparently. For example, you buffer requests and the load changes as a result of occasional load spikes or the natural growth of your business. Because each buffered request can be processed independently, Amazon SQS can scale transparently to handle the load without any provisioning instructions from you.

Share Improve this answer

Follow

edited May 4, 2017 at 3:17

Pang

10.1k ● 146 ● 85 ● 124

answered Jul 25, 2016 at 11:00

cloudtechnician

715 ● 10 ● 16

this answers the question succintly – Govind Rai Dec 2, 2021 at 16:52

44

The biggest advantage for me is the fact that Kinesis is a replayable queue, and SQS is not. So you can have multiple consumers of the same messages of Kinesis (or the same consumer at different times) where with SQS, once a message has been ack'd, it's gone from that queue. SQS is better for worker queues because of that.

Share Improve this answer

Follow

answered Oct 31, 2014 at 19:10

Matthew Curry

762 ● 6 ● 16

How do you ack message? Did you mean delete?
– Tom Raganowicz Apr 5, 2018 at 11:25

Yes, essentially. Saying that you are done with this message
– Matthew Curry Apr 5, 2018 at 12:51

1 It is not correct, you set deletion policy on your sqsListener, so you can say never delete the msg when read it – farhad Jul 29, 2021 at 7:48

3 Maybe that's a new thing @farhad , I answered this 7 years ago. Would a listener, with such a deletion policy in place, grab the same message once more, or would it allow you to skip messages? Regardless, the usual practice of SQS is still more of job-queue, despite being able to abuse it for other things – Matthew Curry Jul 30, 2021 at 18:21

21

The pricing models are different, so depending on your use case one or the other may be cheaper. Using the simplest case (not including SNS):

- SQS charges per message (each 64 KB counts as one request).
- Kinesis charges per shard per hour (1 shard can handle up to 1000 messages or 1 MB/second) and also for the amount of data you put in (every 25 KB).

Plugging in the current prices and not taking into account the free tier, if you send 1 GB of messages per day at the

maximum message size, Kinesis will cost much more than SQS ($10.82/month for Kinesis vs. $0.20/month for SQS). But if you send 1 TB per day, Kinesis is somewhat cheaper ($158/month vs. $201/month for SQS).

Details: SQS charges $0.40 per million requests (64 KB each), so $0.00655 per GB. At 1 GB per day, this is just under $0.20 per month; at 1 TB per day, it comes to a little over $201 per month.

Kinesis charges $0.014 per million requests (25 KB each), so $0.00059 per GB. At 1 GB per day, this is less than $0.02 per month; at 1 TB per day, it is about $18 per month. However, Kinesis also charges $0.015 per shard-hour. You need at least 1 shard per 1 MB per second. At 1 GB per day, 1 shard will be plenty, so that will add another $0.36 per day, for a total cost of $10.82 per month. At 1 TB per day, you will need at least 13 shards, which adds another $4.68 per day, for a total cost of $158 per month.

edited Sep 25, 2017 at 17:19

answered Sep 18, 2017 at 20:52

John Velonis
1,629 ● 1 ● 17 ● 16

I don't completely follow why the exponential increase in size, here, matters. Can you dig in a bit more? It sounds like you got some insight that I'd like to have. *Edit* Actually, looking at

Euguene Feingold's answer, there looks to be a pretty solid debate on this (?). – Thomas Sep 22, 2017 at 14:41

Sorry, I made some mistakes in my calculations (fixed now, I hope). – John Velonis Sep 25, 2017 at 17:19

right, but what if your average SQS message size is small, say 1kb or less? – mcmillab Jun 28, 2019 at 2:54

1 @mcmillab SQS will charge the same whether your message is 1 KB or 64 KB -- see Amazon's SQS pricing page. So if your messages are only 1 KB, SQS will cost 64x as much as the figures I gave above if you are sending the same total amount of data. However, a single request can contain up to 10 messages, so if you are able to batch messages together, it might only be 6x as much (depending how full your batches are). – John Velonis Jul 3, 2019 at 21:34

1 @JohnVelonis Above calculations for SQS pricing are missing a key piece. Extra care is needed to understand how SQS requests are charged. 1 request = 1 API Action. In order to process a single "message", it's necessary to perform at least 3 API actions: 1 send + 1 read + 1 delete. Other SQS features such as changing visibility will incur more API actions. This unexpected multiplier is quite nasty and typically results in SQS being 2-10x more expensive than Kinesis Streams for large data sets (say processing 100 million messages per month). – Vlad Poskatcheev Oct 29, 2019 at 23:00

20

~~Another thing: Kinesis can trigger a Lambda, while SQS cannot. So with SQS you either have to provide an EC2 instance to process SQS messages (and deal with it if it fails), or you have to have a scheduled Lambda (which doesn't scale up or down - you get just one per minute).~~

Edit: This answer is no longer correct. SQS can directly trigger Lambda as of June 2018

https://docs.aws.amazon.com/lambda/latest/dg/with-sqs.html

Share Improve this answer

Follow

edited Jul 2, 2019 at 23:39

shonky linux user

6,408 ● 5 ● 49 ● 75

answered Apr 22, 2016 at 2:52

DenNukem

8,234 ● 3 ● 41 ● 45

1 -1 Disagree. While Kinesis can trigger lambda this poses no advantage over a sheduled SQS lambda. The latter will scale seamlessly (ie if it takes longer than a minute a second lambda will get spun up). Price is per compute time so no appreciable difference there either. And if you need more than 5 concurrent lambdas then just add multiple triggers scheduled a few seconds apart (using cron). This is not a reason to use Kinesis over SNS/SQS. – Steven de Salas Dec 13, 2016 at 19:05

5 I'm not sure I agree with the disagreement ;] - you can schedule one lambda / minute, which would limit you to batch process the messages that arrived that interval. Kinesis would allow you to read the messages immediately. Or is something I misunderstood? – Moszi Feb 8, 2017 at 8:41

There is a huge difference between a couple of cloudwatch triggers and hundreds when needing to invoke the pulling lambda against SQS. – Coding Pig Dec 21, 2017 at 0:46

14 Lambda now supports SQS as a trigger! – sixty4bit Jun 30, 2018 at 18:24

13

Kinesis solves the problem of map part in a typical map-reduce scenario for streaming data. While SQS doesnt make sure of that. If you have streaming data that needs to be aggregated on a key, kinesis makes sure that all the data for that key goes to a specific shard and the shard can be consumed on a single host making the aggregation on key easier compared to SQS

Share Improve this answer

Follow

answered Nov 19, 2015 at 7:21

bhanu tadepalli

161 ● 1 ● 2

13

Kinesis Use Cases

- Log and Event Data Collection
- Real-time Analytics
- Mobile Data Capture
- “Internet of Things” Data Feed

SQS Use Cases

- Application integration
- Decoupling microservices
- Allocate tasks to multiple worker nodes

- Decouple live user requests from intensive background work
- Batch messages for future processing

Share Improve this answer Follow

answered Jan 30, 2020 at 14:45

Sharhabeel Hamdan
1,551 ● 18 ● 16

4 Can't Kinesis do all of the things SQS can do that you mentioned? – CodesInTheDark Aug 20, 2021 at 21:28

@CodesInTheDark For that matter, couldn't SNS do all the things Kinesis is suggested for too? I don't feel like this answers the question of "why" at all. – Casey Dec 8, 2022 at 20:42

2 SQS (optionally combined with SNS, for fan-out) is for loosely coupled microservices; while Kinesis Data Stream is for high throughtput ingestion of streams, and low-latency high throughput delivery of streams to consumers: -SQS FIFO can ingest "only" 300 transactions / second (and SQS FIFO high throughput 3000 TX/s); while Kinesis (which is FIFO too) can absorb 1000 TX/s and per shard, and you can create many shards! -SQS doesn't provide metrics on its delivery latency (the time between the ingestion of a record, and the time it's available for read), while for Kinesis it's on average 200ms – galeop Mar 6, 2023 at 11:44

I'll add one more thing nobody else has mentioned -- SQS is several orders of magnitude more expensive.

5

Share Improve this answer

Follow

answered Dec 3, 2015 at 19:43

Eugene Feingold

279 ● 2 ● 5

4 Are you sure? From my calculation Kinesis is much more expensive, but I've never been talented using the Amazon Simple Price Calculator. – Didier A. Mar 22, 2016 at 22:27

1 Looking at the current pricing examples on aws: Kinesis with 267M messages is around $60, while putting that amount of messages through SQS would result in $107. Obviously I just did a really quick comparison, and this highly differs with different use cases, but this answer definitely should deserve some credit. – Moszi Feb 8, 2017 at 8:48

3 Assume you are doing a fan out to say 2 consumers and 100 million messages a day. SNS cost is $50/day. SQS cost is $40/day/consumer or $80/day total. Kinesis is $1.4/day for PUTs and $0.36/shard. Even with 100 shards (100 MB/s in, 200 MB/s out) it's just $3.60/day + $1.40/day. So Kinesis at $4/day vs. SNS/SQS at $130/day. – Carlos Rendon Feb 27, 2017 at 4:44

7 I'd be interested to know why there is such a disparity in costs in this thread. – Thomas Sep 22, 2017 at 14:43

3 One gatcha with SQS pricing is that the stated cost rate per 1 million requests isn't actual messages. It's API actions, where 1 request = 1 API action. In order to process a single "message", it's necessary to perform at least 3 API actions: 1 send + 1 receive + 1 delete. Other SQS features such as

changing visibility will incur more API actions. This unexpected multiplier is quite nasty and typically results in SQS being 2-10x more expensive than Kinesis Streams for large data sets (say processing 100 million messages per month). – Vlad Poskatcheev Oct 29, 2019 at 23:11

2

Since these services are constantly changing and improving, I summarize main gaps I found in Kinesis that do not exist in SQS.

This information is correct as of April 2023.

Kinesis enables:

1. Processing and analyzing the data in real time.
2. Replay data for any point in time in the last 365 days.
3. Built-in support for several consumers simultaneously consuming the same information. (You can get a similar thing if you connect SQS+SNS)

aws.amazon.com/kinesis/data-streams/faqs/

Share Improve this answer

Follow

answered Apr 17, 2023 at 8:41

Refael

7,273 ● 10 ● 38 ● 54

-1

In very simple terms, and keeping costs out of the picture, the real intention of SNS-SQS are to make services loosely coupled. And this is only primary reason to use

SQS where the order of the msgs are not so important and where you have more control of the messages. If you want a pattern of job queue using an SQS is again much better. Kinesis shouldn't be used be used in such cases because it is difficult to remove messages from kinesis because kinesis replays the whole batch on error. You can also use SQS as a dead letter queue for more control. With kinesis all these are possible but unheard of unless you are really critical of SQS.

If you want a nice partitioning then SQS won't be useful.

Share Improve this answer

Follow

answered Jan 5, 2023 at 8:45

Ankur Kothari

908 ● 9 ● 11

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.