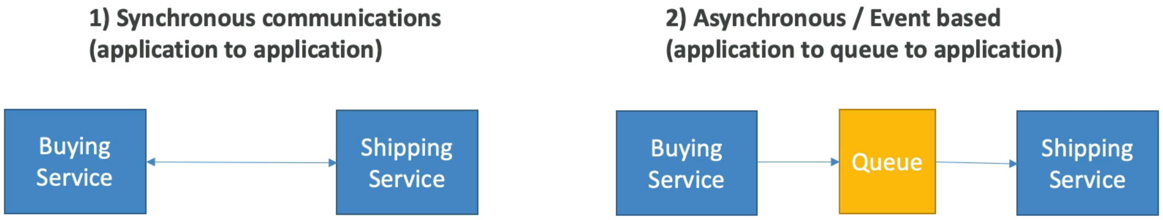
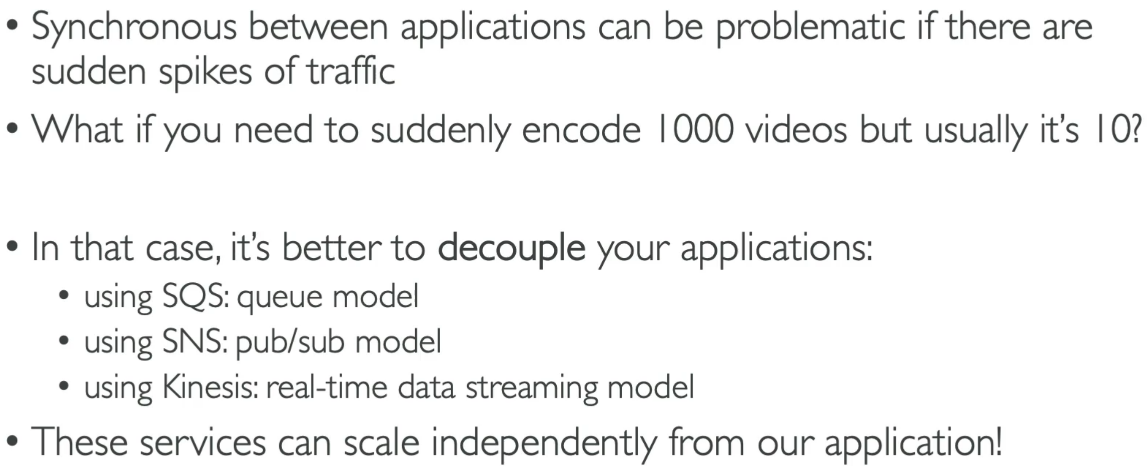
Cloud Integrations

- When we deploy multiple applications, they will inevitably need to communicate with one another.
- There are two patterns of application communication:



1) Synchronous communications (application to application)

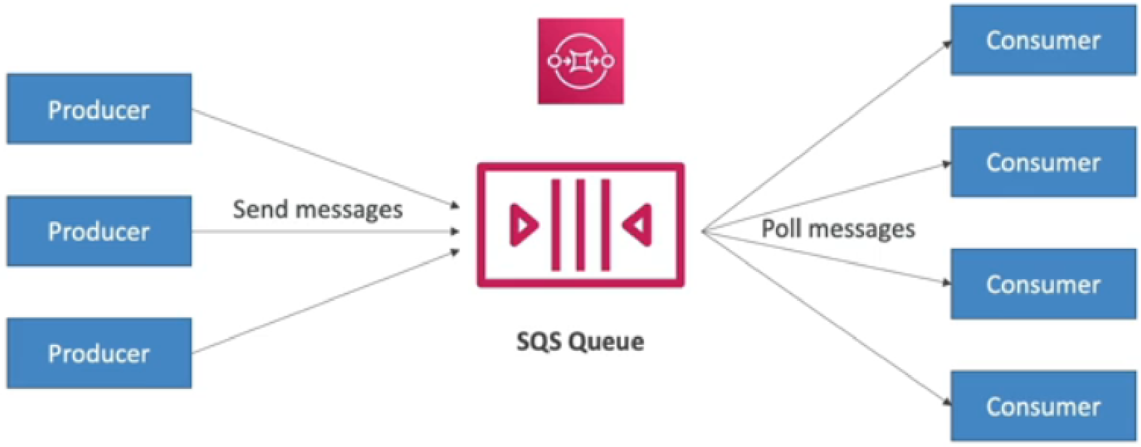
2) Asynchronous / Event based (application to queue to application)



- Synchronous between applications can be problematic if there are sudden spikes of traffic
- What if you need to suddenly encode 1000 videos but usually it's 10?

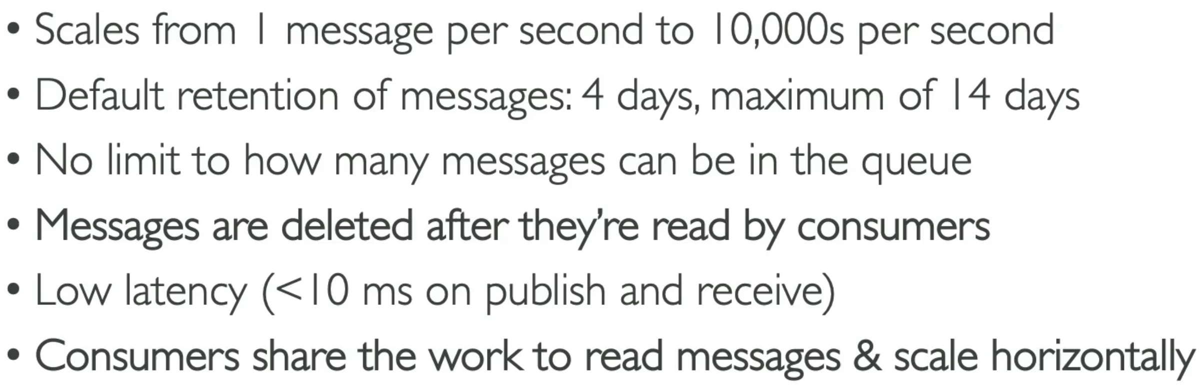
- In that case, it's better to **decouple** your applications:
  - using SQS: queue model
  - using SNS: pub/sub model
  - using Kinesis: real-time data streaming model
- These services can scale independently from our application!

Amazon SQS - Simple Queue Service



- Producers send messages into the queue.
- Consumers poll the queue for message retrieval.
- Multiple consumers share and process different messages. Once completely processed, messages are deleted from the queue.
- Producers and consumers operate independently (decoupled behaviour).They can have different speeds for message production and consumption.

- **SQS is AWS' oldest offering which aims to decouples applications.**
- It is a fully managed service (serverless).



- Scales from 1 message per second to 10,000s per second
- Default retention of messages: 4 days, maximum of 14 days
- No limit to how many messages can be in the queue
- Messages are deleted after they're read by consumers
- Low latency (<10 ms on publish and receive)
- Consumers share the work to read messages & scale horizontally

- SQS also offers the FIFO (first in first out) queue feature.

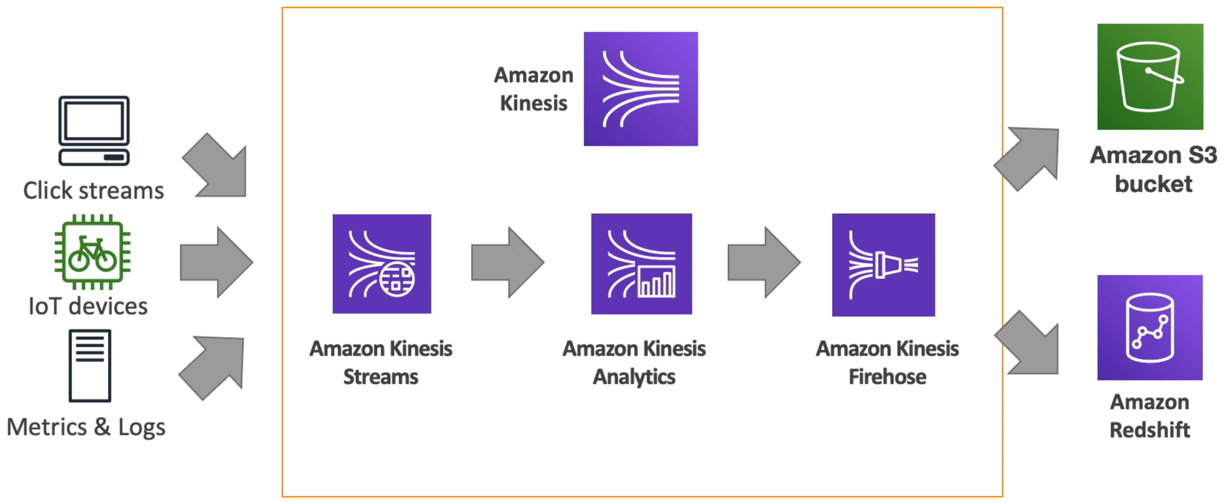
Amazon Kinesis

- **For exam: Kinesis = real-time big data streaming**

- Managed service to collect, process, and analyze real-time streaming data at any scale

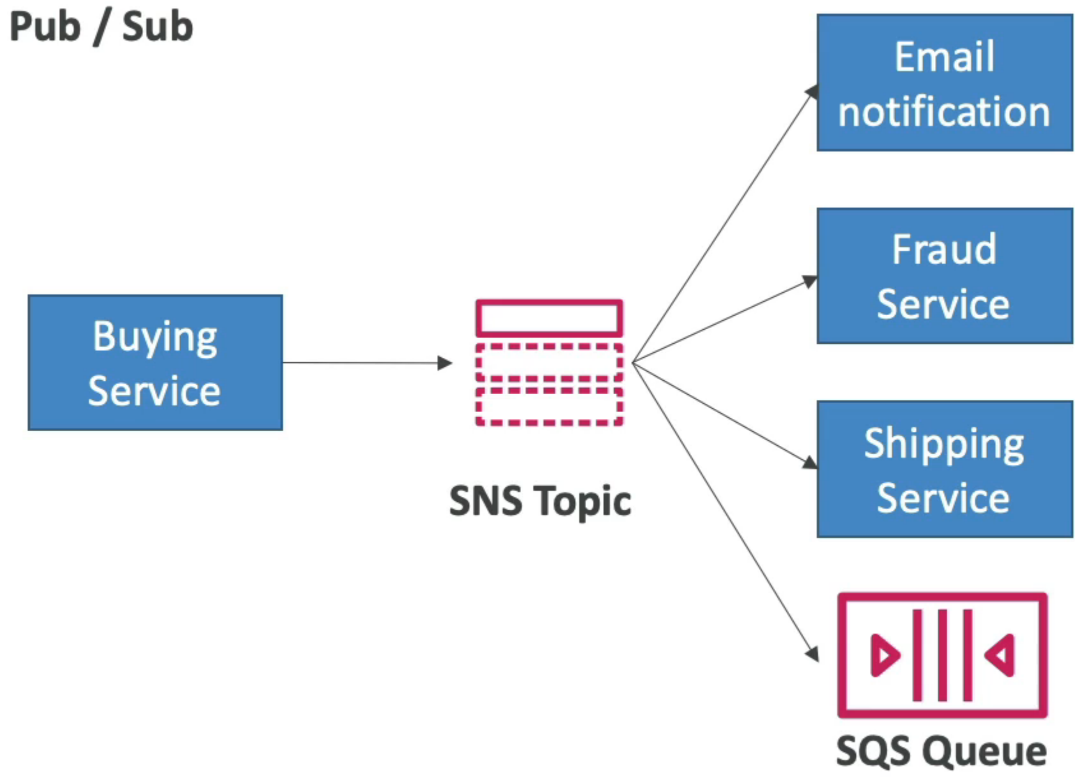
- Too detailed for the Cloud Practitioner exam but good to know:
  - Kinesis Data Streams: low latency streaming to ingest data at scale from hundreds of thousands of sources
  - Kinesis Data Firehose: load streams into S3, Redshift, ElasticSearch, etc…
  - Kinesis Data Analytics: perform real-time analytics on streams using SQL
  - Kinesis Video Streams: monitor real-time video streams for analytics or ML

Kinesis (high level overview)

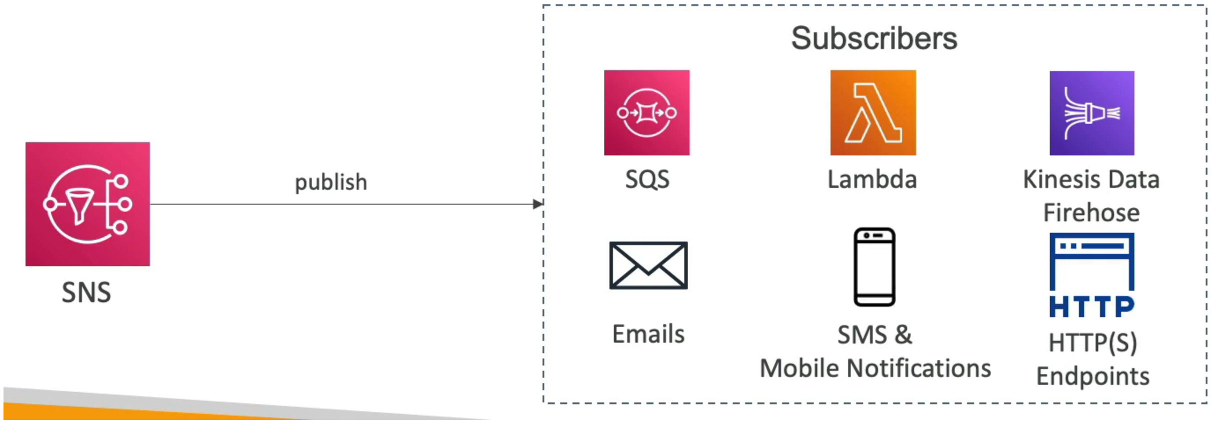


Amazon SNS - Simple Notification Service

- SNS is used to send one message to many receivers. Doing so directly can be complex, so instead, we use the pub/sub model provided by SNS.
- The **sender (or event publisher or pub)** only sends their message to a SNS topic. Now, you can have any number of **receivers (or event subscribers or subs)** as we want listening to the SNS topic notification.



- Here, each subscriber on the topic will get all the messages.



Subscribers

SNS

publish

SQS

Lambda

Kinesis Data Firehose

Emails

SMS & Mobile Notifications

HTTP(S) Endpoints

Amazon MQ

Amazon MQ

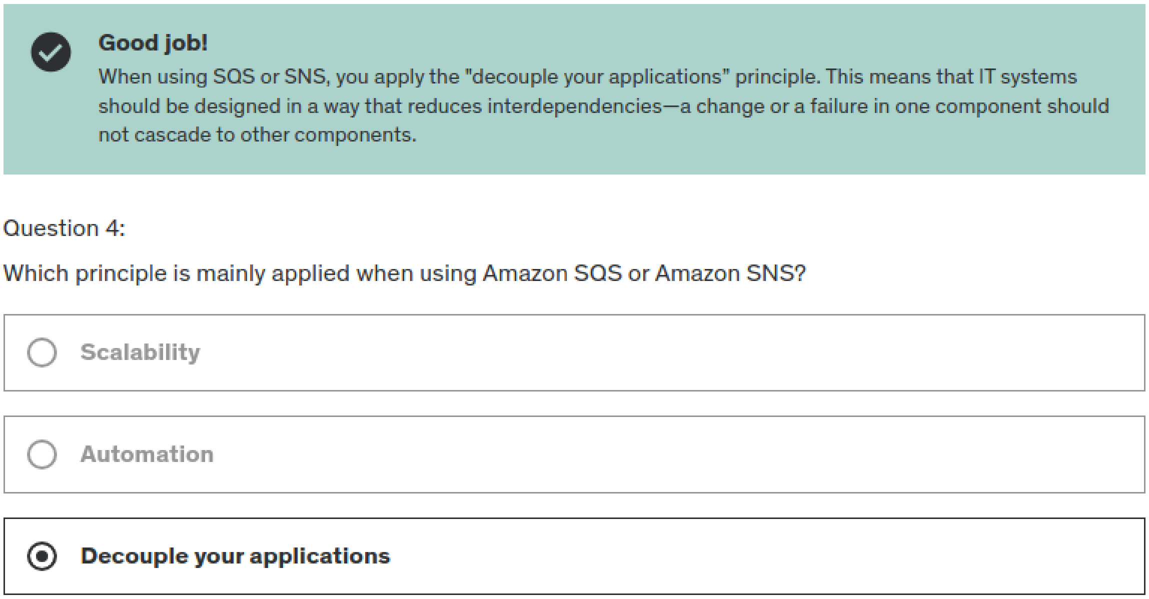
- SQS, SNS are "cloud-native" services: proprietary protocols from AWS
- Traditional applications running from on-premises may use open protocols such as: MQTT, AMQP, STOMP, Openwire, WSS
- **When migrating to the cloud**, instead of re-engineering the application to use SQS and SNS, we can use Amazon MQ
- **Amazon MQ is a managed message broker service for**

RabbitMQ ACTIVEMQ

- Amazon MQ doesn't "scale" as much as SQS / SNS
- Amazon MQ runs on servers, can run in Multi-AZ with failover
- Amazon MQ has both queue feature (~SQS) and topic features (~SNS)

Summary

- SQS:
  - Queue service in AWS
  - Multiple Producers, messages are kept up to 14 days
  - Multiple Consumers share the read and delete messages when done
  - Used to **decouple** applications in AWS
- SNS:
  - Notification service in AWS
  - Subscribers: Email, Lambda, SQS, HTTP, Mobile…
  - Multiple Subscribers, send all messages to all of them
  - No message retention
- Kinesis: real-time data streaming, persistence and analysis
- Amazon MQ: managed message broker for ActiveMQ and RabbitMQ in the cloud (MQTT, AMQP.. protocols)



✓ **Good job!**

When using SQS or SNS, you apply the "decouple your applications" principle. This means that IT systems should be designed in a way that reduces interdependencies—a change or a failure in one component should not cascade to other components.

Question 4:

Which principle is mainly applied when using Amazon SQS or Amazon SNS?

○ **Scalability**

○ **Automation**

● **Decouple your applications**

✓ **Good job!**
Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications. It uses a pull-based system.

Question 5:

Which service allows you to send, store, and receive messages between software components at any volume, without losing messages or requiring other services to be available, using a pull-based system?

○ Simple Notification Service (SNS)

● **Simple Queue Service (SQS)**

○ Auto Scaling Groups (ASG)