[ENGLISH]

Hello Cloud Gurus and welcome to this lecture.

And this lecture is all about

Kinesis shards relative to consumers.

So this is going to be just a quick recap

on Kinesis to begin with, and hopefully you remember

that Kinesis is a service which allows you to collect

and process and analyze real time streaming data.

So for example, that could be unstructured data

like click stream data, video or audio data,

application logs et cetera.

And if you remember, a Kinesis stream is made up of a set

of multiple shards and each shard

is a sequence of data records,

each with their own unique sequence number.

And the data capacity of your stream

is the sum total of the capacity of all of its shards.

And each shard gives you five read transactions per second,

up to a maximum of two megabytes per second.

And you also get 1,000 write records per second,

up to a maximum of one megabyte per second.

So as your data rate goes up,

as it increases, then you will also need

to increase the number of shards

to handle the increase in your data.

And when we increase the number of shards,

that is known as resharding.

But what about the consumers?

And when we say consumers it really means the EC2 instances

which are consuming data from your stream.

And on the consumer you've got the Kinesis Client Library.

And the Kinesis Client Library tracks the number of shards

that exist in your stream

and it also discovers when new shards are added.

Say you increase the number of shards from four to six,

it's the Kinesis Client Library

which is going to detect that and respond accordingly.

So the Kinesis Client Library ensures that for every shard

there is a record processor,

and it's the record processors

which actually process the data

which is being streamed on your Kinesis stream.

And the client library

manages the number of record processors

relative to the number of shards and consumers.

So if you only have one consumer instance,

then the Kinesis Client Library is going to create

all of the record processors

on that single consumer instance.

However, if you have two or more consumer instances,

it's going to load balance across all of those

consumer instances and it will create an equal number

of record processors on each instance.

So for example, if you have two consumer instances,

it's going to load balance across those two,

and it will create half the processors

on one instance and half on the other.

But let's have a look at the diagram again

and hopefully it will make a lot more sense.

So in this example, we have one Kinesis data stream

and we've got four shards in the stream.

And then the right-hand side,

you got one consumer instances

and that one consumer instance is processing

all four shards of the stream.

And therefore, you have four record processors

which are running on that single consumer.

But what happens if we add another consumer instance?

So say we add another one for fail over purposes,

so we've got two consumers

and they still processing four shards

within a single Kinesis stream,

and you got the Kinesis client libraries running on

each of the consumer instances.

And you have two record processors running on each consumer

and the Kinesis Client Library

will handle the load balancing of the load

between the two consumers.

But what about scaling out the consumers?

Well, with the Kinesis Client Library

generally, you should ensure that the number of instances

does not exceed the number of shards

except of course for failure or standby purposes.

And you never need to use multiple instances

to handle the processing load of one shard.

And if you remember the processing load of one shard,

it's only up to a maximum of two megabytes

of read per transactions per second

and one megabyte of write transactions per second.

So you definitely are not going to need to have

more than one EC2 instance to process

that small amount of load per shard.

And of course, one worker

or one consumer can definitely process multiple shards

as we've seen in this diagram here

where we have two consumers

and each of them are processing two shards each.

So it's definitely fine if the number of shards

exceeds the number of instances,

and do not think that just because you reshard

and you add more shards to your Kinesis stream that

that means that you should add more instances

because it definitely does not mean that.

And instead you should use CPU utilization as the factor

that should drive the quantity of consumer instances

you have, definitely not the number of shards

in your Kinesis stream.

And of course the best practice would be

to use an auto scaling group,

and base your scaling decisions

on CPU load of your consumers.

So let's take another look at the diagram

and this diagram just shows you what happens if we reshard.

So we started off with four shards in our stream

and we've resharded and there's now a total of six shards

in our Kinesis stream.

And we've still got our two EC2 instances

which are the consumers.

So we want to load balance between those equally

and we've increased the number of record processors

to three on each of our consumers.

And of course that is all handled automatically

by the Kinesis Client Library,

you don't have to do anything yourself.

So let's move on to our exam tips for Kinesis shards.

And for the exam, you do not need to know

exactly how to set this up

or anything like that,

you don't need to know a low level detail.

You just really need to be aware

that on your consumer instances,

it's the Kinesis Client Library

that's doing all of this work

of managing the number of record processors.

And the Kinesis Client Library creates a record processor

for each shard which is being consumed by your instance.

And if you increase the number of shards

so you reshard on your stream,

the Kinesis Client Library is going to add

more record processors on your consumers,

and it will split them equally

between the number of consumers that you have.

And then finally, always use CPU utilization

as the driver for increasing the number

of consumer instances you have.

Don't think that just because you're resharding

you should increase the number of consumer instances.

And for best practice use an auto scaling group

and base those automatic scaling decisions

on CPU load on your consumers.

So that is everything that you should need to know

about Kinesis shards for the exam.

If you have any questions, please do let me know.

If not, feel free to move on to the next lecture.

Thank you.

[SPANISH]

Hola Cloud Gurus y bienvenidos a esta conferencia.

Y esta conferencia se trata de

Fragmentos de Kinesis en relación con los consumidores.

Así que esto va a ser solo un resumen rápido

en Kinesis para empezar, y espero que recuerdes

que Kinesis es un servicio que le permite recopilar

y procesar y analizar datos de transmisión en tiempo real.

Entonces, por ejemplo, podrían ser datos no estructurados

como datos de transmisión de clics, datos de video o audio,

de aplicaciones , etcétera.

Y si recuerda, un flujo de Kinesis se compone de un conjunto

de múltiples fragmentos y cada fragmento

es una secuencia de registros de datos,

cada uno con su propio número de secuencia único.

Y la capacidad de datos de su flujo

es la suma total de la capacidad de todos sus fragmentos.

Y cada fragmento le brinda cinco transacciones de lectura por segundo,

hasta un máximo de dos megabytes por segundo.

Y también obtiene 1000 registros de escritura por segundo,

hasta un máximo de un megabyte por segundo.

Entonces, a medida que aumenta su tarifa de datos,

a medida que aumenta, entonces también necesitarás

para aumentar el número de fragmentos

para manejar el aumento de sus datos.

Y cuando aumentamos el número de fragmentos,

eso se conoce como resharding .

Pero, ¿y los consumidores?

Y cuando decimos consumidores, en realidad nos referimos a las instancias EC2

que están consumiendo datos de su transmisión.

Y en el consumidor, tiene Kinesis Client Library.

Y Kinesis Client Library rastrea la cantidad de fragmentos

que existen en tu corriente

y también descubre cuándo se agregan nuevos fragmentos.

Digamos que aumenta la cantidad de fragmentos de cuatro a seis,

es la biblioteca de clientes de Kinesis

que va a detectar eso y responder en consecuencia.

Por lo tanto, Kinesis Client Library garantiza que para cada fragmento

hay un procesador de registros,

y son los procesadores de discos

que realmente procesan los datos

que se está transmitiendo en su flujo de Kinesis.

Y la biblioteca del cliente

gestiona el número de procesadores de registro

en relación con el número de fragmentos y consumidores.

Entonces, si solo tiene una instancia de consumidor,

entonces Kinesis Client Library va a crear

todos los procesadores de registro

en esa única instancia de consumidor.

Sin embargo, si tiene dos o más instancias de consumidor,

va a equilibrar la carga en todos esos

de consumidor y creará un número igual

de procesadores de registros en cada instancia.

Entonces, por ejemplo, si tiene dos instancias de consumidor,

va a equilibrar la carga entre esos dos,

y creará la mitad de los procesadores

en un caso y la mitad en el otro.

Pero echemos un vistazo al diagrama de nuevo.

y espero que tenga mucho más sentido.

Entonces, en este ejemplo, tenemos un flujo de datos de Kinesis

y tenemos cuatro fragmentos en la secuencia.

Y luego el lado derecho,

tienes una instancia de consumidor

y que una instancia de consumidor está procesando

los cuatro fragmentos de la corriente.

Y por lo tanto, tiene cuatro procesadores de registro

que se ejecutan en ese único consumidor.

Pero, ¿qué sucede si agregamos otra instancia de consumidor?

Digamos que agregamos otro para fines de conmutación por error,

así que tenemos dos consumidores

y todavía procesan cuatro fragmentos

dentro de una sola secuencia de Kinesis,

y tiene las bibliotecas de cliente de Kinesis ejecutándose en

cada una de las instancias de consumidor.

Y tiene dos procesadores de registro ejecutándose en cada consumidor

y la biblioteca de clientes de Kinesis

manejará el equilibrio de carga de la carga

entre los dos consumidores.

Pero, ¿qué pasa con el escalado de los consumidores?

Bueno, con Kinesis Client Library

en general , debe asegurarse de que el número de instancias

no excede el número de fragmentos

excepto, por supuesto, para fallas o propósitos de espera.

Y nunca necesita usar múltiples instancias

para manejar la carga de procesamiento de un fragmento.

Y si recuerda la carga de procesamiento de un fragmento ,

es solo hasta un máximo de dos megas

de lectura por transacciones por segundo

y un megabyte de transacciones de escritura por segundo.

Así que definitivamente no vas a necesitar tener

más de una instancia EC2 para procesar

esa pequeña cantidad de carga por fragmento .

Y por supuesto, un trabajador

o un consumidor definitivamente puede procesar múltiples fragmentos

como hemos visto en este diagrama aquí

donde tenemos dos consumidores

y cada uno de ellos está procesando dos fragmentos cada uno.

Así que definitivamente está bien si la cantidad de fragmentos

excede el número de instancias,

y no creas que solo porque te reshard

y agrega más fragmentos a su flujo de Kinesis que

eso significa que debes agregar más instancias

porque definitivamente no significa eso.

Y en su lugar, debe utilizar la utilización de la CPU como factor

eso debería impulsar la cantidad de instancias de consumidores

tienes , definitivamente no la cantidad de fragmentos

en su flujo de Kinesis.

Y, por supuesto, la mejor práctica sería

para usar un grupo de escalado automático,

y base sus decisiones de escalado

en la carga de la CPU de sus consumidores.

Así que echemos otro vistazo al diagrama.

y este diagrama solo muestra lo que sucede si recortamos .

Así que comenzamos con cuatro fragmentos en nuestra transmisión.

y hemos repartido y ahora hay un total de seis fragmentos

en nuestra corriente de Kinesis.

Y todavía tenemos nuestras dos instancias EC2

cuales son los consumidores.

Así que queremos equilibrar la carga entre aquellos igualmente

y hemos aumentado la cantidad de procesadores de registros

a tres en cada uno de nuestros consumidores.

Y, por supuesto, todo se maneja automáticamente.

por Kinesis Client Library,

usted no tiene que hacer nada usted mismo.

Así que pasemos a nuestros consejos de examen para fragmentos de Kinesis.

Y para el examen no hace falta saber

exactamente cómo configurar esto

o algo por el estilo,

no es necesario conocer un detalle de bajo nivel.

Solo necesitas ser consciente

que en sus instancias de consumidor,

es la biblioteca de clientes de Kinesis

que está haciendo todo este trabajo

de gestionar el número de procesadores de registros.

Y Kinesis Client Library crea un procesador de registros

para cada fragmento que está siendo consumido por su instancia.

Y si aumentas el número de fragmentos

por lo que reshard en su corriente,

Kinesis Client Library va a agregar

más procesadores de discos en sus consumidores,

y los dividirá por igual

entre el número de consumidores que tienes.

Y finalmente, siempre use la utilización de la CPU

como motor para aumentar el número

de instancias de consumidor que tiene.

No creas que solo porque estás resharding

debe aumentar el número de instancias de consumidor.

Y como práctica recomendada, utilice un grupo de escalado automático.

y basar esas decisiones de escalado automático

en la carga de la CPU en sus consumidores.

Así que eso es todo lo que deberías saber

sobre fragmentos de Kinesis para el examen.

Si tiene alguna pregunta, por favor hágamelo saber.

Si no, siéntase libre de pasar a la siguiente lección.

Gracias.