Hello Cloud Gurus and welcome to this lecture

which is going to introduce ElastiCache.

So what is ElastiCache?

Well, as you probably know, memory is faster than disk.

And wouldn't it be great if we could speed up

database queries by storing frequently accessed data

in an in-memory cache?

Well, ElastiCache enables you to do exactly that.

So ElastiCache is an in-memory cache.

Unlike many data stores in AWS,

it is a key value data store.

An ElastiCache makes it really easy to deploy,

operate, and scale an in-memory cache in the cloud.

And it's designed to improve database performance.

So it allows you to retrieve information

from fast in memory caches,

instead of slower disc based storage.

And it's great for read heavy database workloads.

So it allows you to catch the results

of I/O intensive database queries,

and it's also really useful for storing session data

for distributed applications.

But it might make a little bit more sense

if we consider an example.

So imagine you have a busy database,

serving thousands of queries a day,

and many of these queries are requesting the same data.

Well, in this scenario, you can use an ElastiCache cluster

to cache frequently accessed data.

So frequently accessed data will be stored

here in ElastiCache to enable faster access.

And then when your application needs to access the data,

it can query ElastiCache instead of the database.

And by doing this, we can reduce the load on the database

and speed up data access for our application.

And there are two types of ElastiCache available.

Firstly, we have Memcached,

and this is great for basic object caching.

It scales horizontally,

but there is no persistence for the data

and there is no Multi-AZ or failover option either.

However, is a good choice if you just want basic caching

and you want your caching model to be as simple as possible.

And then the second option that we have is Redis.

And this is a more sophisticated solution

with enterprise features like persistence,

replication, Multi-AZ and failover.

and it also supports sorting and ranking data

for example, for gaming leaderboards.

And it also supports complex data types

like lists and hashes.

But don't get too hung up on complex data types,

the main thing to remember is we've got two options,

you've got Memcache D for the basic object caching,

and Redis, is just the more sophisticated solution

with enterprise features.

So let's take a look at a typical scenario

that you might see in the exam.

And you might be given a scenario

where a particular database is under a lot of stress

and you'll be asked to find a solution.

So you might be asked at which service

should you use to alleviate this.

And you will need to know when to use ElastiCache.

So ElastiCache is a really good choice

if your database is particularly read-heavy

and the data is not prone to frequent changing.

Because if the data is changing too frequently,

then ElastiCache will struggle

to have the latest data available

and it will not be a very effective solution.

And you should also be aware of when ElastiCache can't help.

And it's not gonna help if your database

is struggling with heavy write loads.

So caching will not help alleviate heavy write loads

so in that case you may need to scale up

your database instead.

And if you're running online, analytical processing

or OLAP queries, and your database is feeling stress

because you're performing online, analytical processing,

then ElastiCache is not going to help with that

because you really need a data warehouse for that

rather than a database.

And you should think about using Redshift instead.

So onto my exam tips,

ElastiCache is an in-memory cache designed to improve

read performance for read-heavy databases.

And we've got two different flavors,

so we've got Memcache D and Redis.

And Memcache D is an in-memory key value data store.

And it's the one to use if object caching

is your primary goal.

And you want to keep things as simple as possible.

You don't need persistence or Multi-AZ features

and you don't need to support advanced data types

or sorting your data.

And then we have Redis as well,

and it's also an in-memory key value data store.

And this is the one to use

if you're performing data sorting, and ranking,

the kind of thing that you would need

for gaming leaderboards.

If you've got advanced data types such as lists and hashes,

and you need your data to be persistent,

and you need it to be Multi-AZ.

So that's it for this lecture.

If you have any questions, please let me know,

otherwise I will see you in the next lecture.

Thank you.