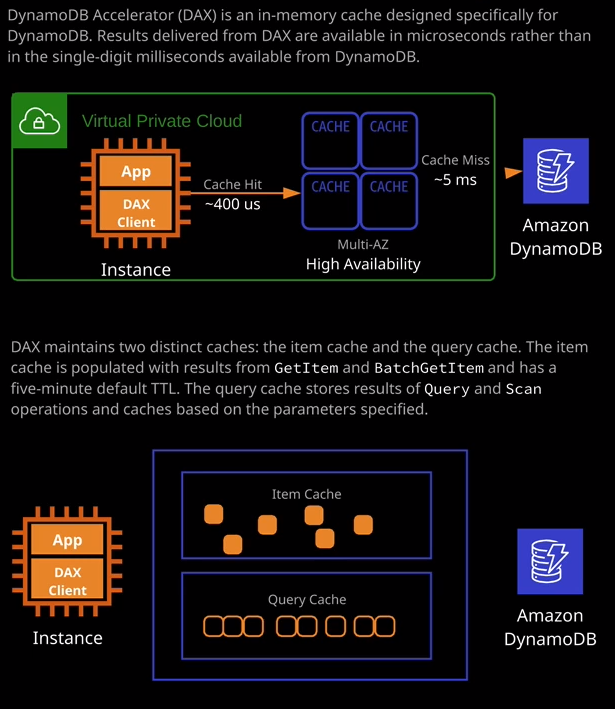
Welcome back, and in this lesson, I want to talk about DynamoDB Accelerator, also known as DAX.



**DynamoDB, is designed to be a low maintenance, high performance data base which provides access to data in milliseconds** and in most cases that's enough but there are edge cases solutions which need data faster, potentially microseconds, or when you have some really high-end read-heavy workloads and for those scenarios, you can benefit from an in-memory cache, which is what DAX provides. Now I don't want to go into huge amounts of detail in this lesson because you really don't need the implementation details for the exam but you do need to be aware at a high level of exactly how DAX works architecturally. **Now Dax runs inside a VPC and it uses a cluster architecture so you can have one or more nodes. Applications use a DAX client, which is generally installed on the same compute resources as the application itself. When you initially read items from DynamoDB those items, as well as being returned to the application are stored inside DAX. If an item is read again, it's returned from DAX without using DynamoDB, a process called a cache hit. If an item is not in DAX or it expires, it gets pulled from DynamoDB, known as a cache miss, and then it's delivered to the application and again it's stored inside DAX and that's pretty much the architecture that DAX uses. It lives on the same compute resources as your application. If your application needs data from within DynamoDB, it can be delivered from cache, often in microseconds rather than the single digit milliseconds that DynamoDB offers.**

So essentially, ***DAX is designed for really high read-heavy workloads or when your application is extremely latency sensitive.***

Now DAX maintains two distinct caches. **The item cache and the query cache**. **The item cache is populated with individual results from get item and batch get item, and it has a five minute default TTL. The query cache stored results of query and scan operations, and it caches these results based on the parameters specified so if you do a query for a certain partition key and a certain value or range of values for the sort key then the data returned is stored in the query cache along with the parameters that are used to retrieve that data.** So it operates in a similar way to how CloudFront works from the media caching perspective but it does so for DynamoDB. Now DAX isn't the only in-memory cache that you can use with DynamoDB. The next product that we're going to talk about, which is Elasticache historically, that was the product that was used as an in-memory cache but DAX has been specifically designed to work with DynamoDB, and so it is the preferred solution. Now again, I don't expect you to get many questions that are testing your knowledge on the specifics of DynamoDB Accelerator. What I might expect a question on is whether you can use DAX with other AWS products and what is the best caching product to you was with DynamoDB, and so I wanted to introduce the concept. I wanted to talk about the architecture, I mentioned how it has to operate inside a VPC and how its main benefit is this microsecond response, rather than the millisecond that DynamoDB can deliver by default. So if you've got applications that require the fastest possible response time for reads maybe **real time bidding, gaming, or trading applications DAX is an ideal solution**. **If you've got applications that read a small number of items much more frequently. For example, online stores, maybe fire sales, maybe busy sale periods, or popular products then DAX can offer significant benefits**. **If you've got applications that are read intensive but you don't want to allocate really high levels of RCU to a DynamoDB table then you can potentially look at using DAX. Applications that require the same type of access patterns against large sets of data. Again, you can look at using DAX.** Things that you wouldn't use DAX for are applications that require **strongly consistent reads. Any reads from DAX are going to be eventually consistent.** So that's critical to understand if the question mentions strongly consistent then you can't use DAX. If you've got applications that really don't require microsecond response, then DAX is probably not an ideal solution. You could just look at optimizing your existing access patterns against DynamoDB. If your applications are write intensive then probably again it's an antipattern for DAX because DAX only supports read caching and then lastly, if you've got applications that either for legacy reasons, or just compatibility reasons, already used a different caching solution than you're probably not going to get any benefit from DAX without significant retooling. So you need to keep all those things in mind if you do get any questions in the exam about utilizing a caching engine and that's everything I wanted to cover. I want to keep it high level because I wanted to give you just the level of information that you'll need for the exam. With that being said, that's everything I wanted to cover. Go ahead, mark this lesson as complete, and when you're ready, join me in the next.