

So DynamoDB is a noSQL database. Traditionally, it's often referred to as a key value store that's not strictly true. It would be more accurate to describe it as a wide column store. It's a key value database, but it's two dimensional key value. It supports the concepts of tables, rows, and columns, which are called tables, items, and attributes within DynamoDB. Now tables in DynamoDB are actually collections of items. An item is like a row in other databases. So a table has zero or more items, and DynamoDB does not enforce a rigid schema across all of those items. So if you recall from earlier lessons, where I talked about relational databases required you to define the schema upfront. So that's the keys that are used, the attributes, and the data types. With DynamoDB, you don't need to do that. The only thing that you need to define upfront is the primary key for the table. So that's the partition key or the partition key and sort key and I'll talk about what that means next.

So every item in a **table needs to be uniquely identified. You can't have two items in a table that don't have unique identification and the way that that's done is using the table's primary key. So each item needs a unique value for that primary key. Now the primary key can either be a simple primary key, which just consists of one attribute known as a partition key or it could be a composite key, which means it has both a partition and a sort key. The partition key is also known as the hash key and the sort key is also known as the range key.** So try and keep those things in-memory when you're taking the exam. Generally, the exam will refer to the partition key and the source key but if you do hear hash key mentioned you know, that means the same as the partition key and if you hear a range key mentioned, that's the same as the sort key.

So let's look at an example. I'm going to move over to the DynamoDB console and DynamoDB, like a lot of other AWS services, is regional based, so you have to make sure that you select the correct region. So I'm currently in the North Virginia region, and I'm going to go ahead and create a table. So I click "Create Table" and I'm going to call the table weather\_data. It's going to be a table that's going to store weather data from weather stations that are distributed globally. Now we start with the default of having to provide a primary key for this table and by default, it's a simple primary key, which just consists of the partition key. Now a suitable partition key for this weather data table might well be station ID. So let's assume that this is a numeric ID for a given weather station. So we might have 0001, 0002, and so on. Each of those IDs would uniquely represent a weather station. But let's say that we wanted the weather station to report into DynamoDB once every half hour. Well, if we did that, just having this station ID as a simple primary key wouldn't be enough because if a single station reported in twice in the same hour, then the second entry would overwrite the first because only the station ID would exist to represent a unique value for the item. So what we need is to add what's known as a sort key. So by clicking this button and defining a sort key as a second part of what is now a composite primary key, we add another attribute to ensure uniqueness. Now let's say that I'm going to call this sort key date and time, so date and time. This means that a given station ID can now report multiple data points each for a specific date and time and so, assuming that we don't have one single station reporting data more frequently than the granularity of this provides, we're always going to ensure that it's unique. Now I'm going to be talking throughout this topic about all of the important settings that you need to be aware of for the exam. If I untick this use default settings for a moment, then there are a lot of other options that can be defined on a table basis. So we've got things like the read and write capacity mode, secondary indexes, provisioned capacity, auto scaling, encryption settings, and there are many other settings that you need to be aware of. Now I just want you to be aware of the fact they exist for now. But for this example, I'm going to tick use default settings and go ahead and create this table. Because DynamoDB is a regional service, it means that every table in a specific region needs to be unique. So when I create a table, I give it a table name and that table name needs to be unique in this region, in my account. It's not like S3, which has a global namespace across all AWS accounts. With DynamoDB, as long as the table name is unique in the region in your account, then that's enough. Now every table, when it's allocated is given a unique ARN or Amazon resource name and then that's the way that you can reference this table across any region in any AWS account. It's a wholly unique identifier for this specific table. So this is the region I'm currently in North Virginia. This is the account idea of the account I'm using. It's a table resource, and it's called Weather Data. Now, don't worry, I'll be going through all of the important elements of tables as I go through the lessons in this topic of the course. For now, I want to focus on items. Now this is an example item structure on the diagram on the right of my screen. An item consists of the primary key for the table. Now again, the primary key can either be just the partition key or it can be the partition and sort key. So essentially every item starts with those two attributes, but it could also have additional attributes. So in this example, I've got an item in a table, which has this as the ID. So this is a unique identifier. It's got an attribute called name and a value Adrian and attribute called age of value 40 and then an attribute called Father Foods, which is a list which has ice cream and cheesecake. I've got another item from a different table, which has a partition key of customer and a value of 0001, a sort of date and time and it's got the date and time represented in a full format as its value. It's got an attribute called title, 101 best Xmas Gifts is the value and then it's got attribute call payment, which is a list of JSON documents and don't worry, as we go through the course, you'll understand all of this. Now that's the theory but I want to show you how you can actually create an item inside DynamoDB. So to do that, I'm going to go ahead and click on "Create Item." Now the only two mandatory attributes are the keys for the table. So the partition key that we defined of station ID and the sort key which is date and time. So I could create an item with just those two. So let's say I wanted to create an entry for Station ID 0001 and then for date and time, let's say it was 2019. Maybe it's August, and it's the fifth of August and then the time is 8:00 a.m. and let's say the time zone is UTC, so that's referenced by Z for Zulu. So that creates an item in the table with just the primary keys. So the partition key and the sort key and that's perfectly valid. The only thing that's enforced in a DynamoDB table are the primary key or keys of the table. So I could also go ahead and create another item again for station ID 0001 and as long as it had a different date and time value for the sort key it would be added as a brand new item. So let's do that so again 2019 and it's August. It's the fifth of August. and then as the time, let's say it's 08:30 so 8:30 a.m. and again Z meaning UTC time zone. So I could go ahead and say that, and that will be added as a separate item. So remember, as long as the primary key is unique then it could be added as a new item and the primary key can either be the partition key alone or the partition key and the sort key. Now, in addition, I could add items which have additional attributes. So let's do that. I'll create an item again I'm going to add 0001. For date and time it's going to occur a little bit later in the day, so it's going to be the same day. So 2019 and then the fifth of August. For the time value, it's going to be 9 a.m. so 09:00:00 and then Z. Now, in addition to these primary keys, I'm also going to add an additional attribute and I can do that by going to append and then appending an additional string attribute. Now, an attribute in DynamoDB can be one of various different types of data. It could be a string, a binary, a number, a list, or set of strings the same for number and binary. It could be a map, a list, a boolean, or a null value. We've got lots of different available data types, but to keep things simple, I'm going to make this a string. I'm going to call it TempC for temperature in Celsius and I'm going to set it to 15 Celsius and hit "Save." Now items in a table don't need to have the same attributes. DynamoDB does not enforce a fixed or rigid schema on all items in a table. So I could go ahead and create a new item. I'm going to add it for the same station ID so 0001. It's going to be a different date and time value so 2019 again the fifth of August, and then it's going to be 09:30. So, again, it's going to be a different time than the previous items that we've added to this table. I'm going to add a new attribute. It's going to be a string. It's going to be the temperature in Celsius again but this time it's going to be a different value. But unlike the previous item, this is going to have an additional attribute. So again it's going to be a string attribute. The attribute name is going to be rain and the value is going to be zero. So I've have had no rain at this location and I'm going to save that item. So you can see straightaway the items in the table can have a varied structure in terms of their attributes. They do not need to have the same attributes. Now the limit for an item inside the DynamoDB table is that the keys for the item, plus all of the attribute names, and the tribute value, so everything combined, can only be up to 400 kb in size. So that's an important thing to remember for the exam. Everything combined the keys, the attribute names, and the attribute values they all total up and the maximum size for an item is 400 kb in size. Now, when you write an item, known as putting an item, or read an item, known is getting an item, you're storing all of the data for the item or reading all of the data for the item. You cannot put or get partial items. So if the item that you're interacting with is 400 kb in size then when you read the item, you have to read the full 400 kb and when you write the item, you have to write the full 400 kb. So that's also important to know for the exam. It becomes more important as we go through the lessons in this topic, and I start talking about how DynamoDB handles capacity. Now the simplest form of operations on a DynamoDB table is the get item which retrieves data and the put item which adds data. Each of those operations require that you specify explicit values for the single or the jewel key that the table uses. So if I wanted to retrieve one of these items using get item, I would need to specify the station ID and a value for date and time. If I go ahead and click on one of these items, let's say, for example, this bottom item that essentially performs a get item operation. So by specifying the station ID and date and time which I've simulated by clicking in the UI and performing get item operation. Now this represents it in a nice graphical view, but it is possible to show this item in text. So it's essentially just a JSON document. If a click on this check box to select DynamoDB JSON that I get a fuller form of this item. So this stores the attribute name and the attribute value as well as the data type that's used, so s for string. So essentially, **if you're interacting with DynamoDB, you can do so from the console UI, from the API, or the CLI** but in essence, you're always interacting with DynamoDB and updating the data that's stored in it.