README

BitBeat is a new startup that is planning to take the record industry—and the world—by storm with our new product **BitBanger,** a web-based music mixer app.

BitBeat is becoming a recognizable name in the music industry. We want to experiment with voice recognition technologies. This means that we must research and test services that will allow us to build and sustain rapid growth. With a growing customer base, we must perform well, even during traffic spikes.

Your manager wants you to explore different database options. You have to determine which is the best solution to support future growth, SQL or NoSQL. A senior solutions architect on your team read an article about NoSQL databases. She has suggested Amazon DynamoDB could be the solution to meet evolving business requirements. She shared that DynamoDB is a fully managed proprietary NoSQL database service that has a reputation for helping companies easily scale and move towards better performance.
DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, ad-tech, Internet of Things (IoT), and many other applications.

BEFORE GETTING STARTED

Here's some important information to know before starting this hands- on activity.

Activity time: 60 min

Requirements: You must have an AWS Educate account.

Getting help: If you experience any issues as you complete this activity, please ask your instructor for assistance.

DID YOU KNOW

With DynamoDB, there are no servers to provision, patch, or manage, and no software to install, maintain, or operate. DynamoDB automatically scales up and down to adjust for capacity and maintain performance. Availability and fault tolerance are built in, eliminating the need to architect your applications for these capabilities. It's a fully managed, serverless service that only requires you to provide the capacity your application requires. DynamoDB takes that information and determines the perfect mix of resources to meet your requirements.

Task overview

In this hands-on activity, you are going to create a DynamoDB table as a proof of concept. You will populate the table with objects and their attributes to gain insight into how the database works.

Learning outcomes

After completing this activity, you should be able to:

- Create a DynamoDB table; assign its primary and sort keys
- Add items to the DynamoDB table
- Create and modify item attributes
- Perform both query and scan searches of the table

Let's get started!

Task 1: Create a new table

Start by creating a new table in DynamoDB called **Music**. Each table requires a primary key to partition data across DynamoDB servers. A table can also have a sort key. The combination of primary key and sort key will uniquely identify each item in our DynamoDB table.

1. In the AWS Management Console, click **Service**. Then, click **DynamoDB.**
2. Click **Create table.**
3. For **Table name,** type Music.
4. For **Primary key,** type Artist and leave **String** selected.
5. Select ☑ **Add sort key**, then in the new field type Song and leave **String** selected.

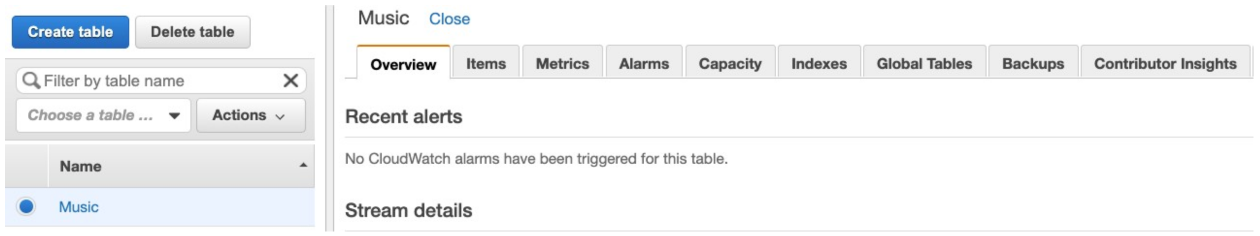
Important info

The table will have **items** with **attributes.** It's important that you understand what items and attributes are. Let's add some data to the table paying close attention to items and attributes.

Your table will use default settings for indexes and provisioned capacity.

6. Click **Create.**

The table will create in less than a minute. Once created, you have a database **table** ready for data to be added to it. When you create a table, you define a partition key attribute to uniquely identify each item in the table so that no two items can have the same key. You can also assign other attributes like a sort key attribute.



No CloudWatch alarms have been triggered for this table.

*AWS Management Console view of the **Music** table*

Task 2: Add data

Next, add data to the Music table. A **table** is a collection of data on a particular topic.

Each table contains multiple **items**.

> *An item is a group of attributes that are uniquely identifiable among all of the other items.* Items in a DynamoDB are similar in many ways to rows in other database systems. In DynamoDB, there is no limit to the number of items you can store in a table.

Each item is composed of one or more **attributes**.

> *An attribute is a fundamental data element, something that does not need to be broken down any further.* For example, an item in a *Music* table contains attributes such as *Song* and *Artist*. Attributes in DynamoDB are similar to columns in other database systems, but each item (row) can have different attributes (columns).

When you write an item to a DynamoDB table, only the primary key and sort key (*if used*) are required. The table does not require a schema, you can add attributes to one item that may be different to the attributes of other items.

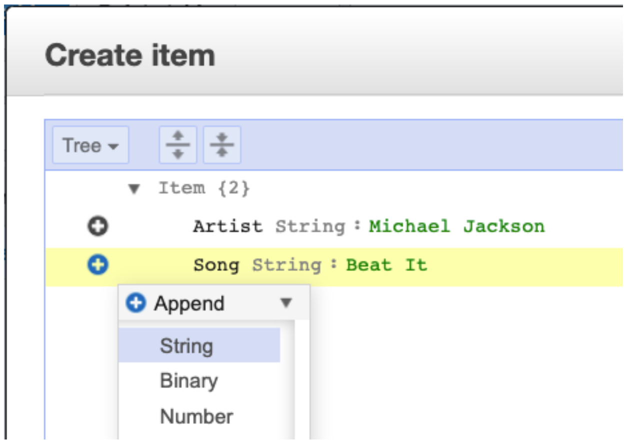
1. Click the **Items** tab. Then, click **Create item**.

If the table is still being created, refresh the webpage to display the *items* tab.

2. For **Artist** string, type Michael Jackson.
3. For **Song** string, type Beat It.

These are the only **required** attributes, but you will now add additional attributes.

4. To create an additional attribute, click the plus sign ⊕ to the left of Song. Then, click **Append.**



```
▼ Item {2}
    Artist String : Michael Jackson
    Song String : Beat It
```

AWS Management Console view of creating an item with attribute types.

5. In the drop-down list, select **String**.

A new attribute row will be added.

6. For the new attribute, enter:
   - In FIELD, type Album.
   - In VALUE, type: Thriller.

7. Add another new attribute by clicking the plus sign ⊕ to the left of Album. Then, click **Append**.

8. In the drop-down list, select **Number.**

A new number attribute will be added.

9. For the new attribute, enter:
   - In FIELD, type Year.
   - In VALUE, type: 1982.

10. Click **Save** to store the new item with its four attributes.

The item will appear in the AWS Management Console.

You have created one item so far. (*Refer to the above item and its attributes.*)

Quiz

1. What is the name of your **table**?
2. What is your table's **primary key?**
3. What is your table's **sort key**?

11. Now **create a second item**, using these attributes:
(Refer to previously used steps if needed.)

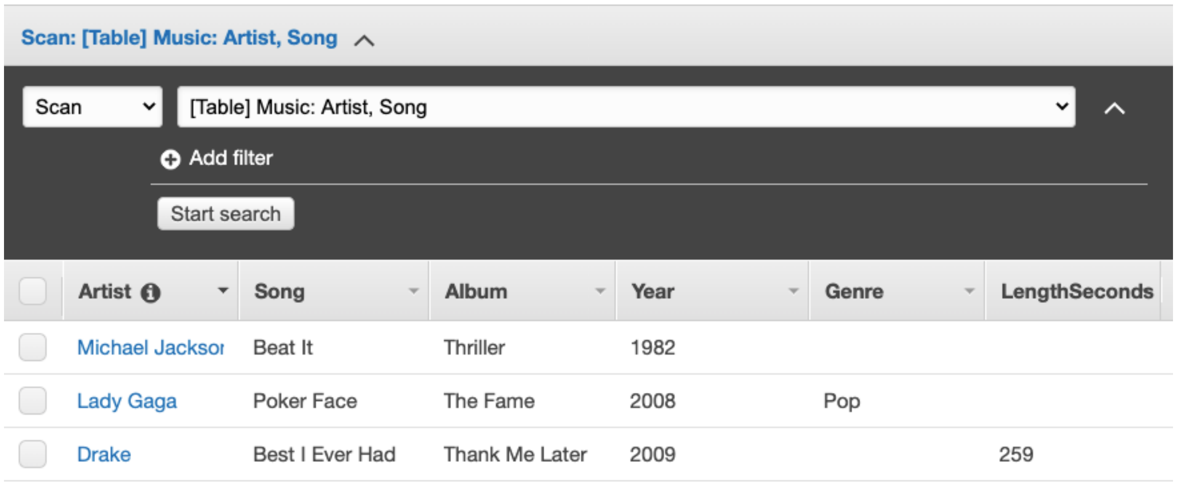
Attribute name	Attribute type	Attribute value
Artist	String	Lady Gaga
Song	String	Poker Face
Album	String	The Fame
Year	Number	2008
Genre	String	Pop

12. **Create a third item**, using these attributes:

Attribute name	Attribute type	Attribute value
Artist	String	Drake
Song	String	Best I Ever Had
Album	String	Thank Me Later
Year	Number	2009
LengthSeconds	Number	259

Note: The item above has a new LengthSeconds attribute (*identifying the length of the song*). This demonstrates the flexibility of the NoSQL database.

There are also faster ways to load data into DynamoDB, such as AWS Data Pipeline, programmatically loading data, or using one of the no-cost tools available on the internet.


Scan: [Table] Music: Artist, Song
Scan
[Table] Music: Artist, Song
Add filter
Start search

Artist	Song	Album	Year	Genre	LengthSeconds
Michael Jacksor	Beat It	Thriller	1982		
Lady Gaga	Poker Face	The Fame	2008	Pop	
Drake	Best I Ever Had	Thank Me Later	2009		259

AWS Management Console view of populated table

Task 3: Modify an existing item

You notice that there is an error in your data. In this task, you will modify an existing item.

1. Click **Drake**.
2. Mouse over 2009 in the **Year** column**,** click the pencil **and change** 2009 to 2010.
3. Click **Save**.

The item is now updated.

DID YOU KNOW

DynamoDB takes away one of the main stumbling blocks of scaling databases: *the management of database software and the provisioning of the hardware needed to run it*. You can deploy a nonrelational database in a matter of minutes. DynamoDB also synchronously replicates data across three facilities in an AWS Region, giving you high availability and data durability.

Task 4: Query the table

There are two ways to query a DynamoDB table: **Query** and **Scan.**

A **query** operation finds items based on primary key and optionally sort key. It's fully indexed, so it runs very quickly.

A **scan** operation reads every item in a table and returns all data attributes for every item in the table.

1. Click the drop-down list showing **Scan** (located below the *Create item* button) and change it to **Query.**

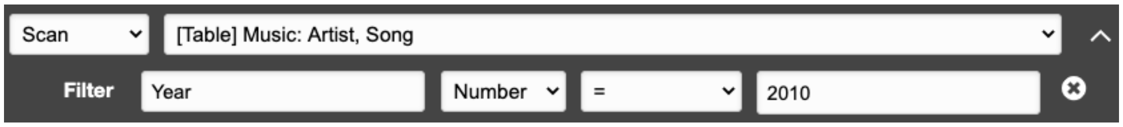
Fields for the partition key (which is the same as primary key) and sort key are now displayed.

2. Enter these details:
 Partition key: Drake **Sort key:** Best I Ever Had

3. Click **Start search**. (*You might need to scroll down to see it*.)

The song quickly appears in the list. A query is the most efficient way to retrieve data from a DynamoDB table. Alternatively, you can scan for an item. Scanning involves looking through every item in a table, so it's less efficient and can take a significant amount of time for larger tables.

4. Click the drop-down list showing **Query** and change it back to **Scan**.
5. Click ⊕ **Add filter,** then:



AWS Management Console view where query and scan search will be performed

- For **Enter attribute,** type Year.
- Change *string* to Number.
- For **Enter value,** type 2010.
- Click **Start Search**.

Now, only the song released in 2010 is displayed.

DID YOU KNOW

When reading data from DynamoDB, users can specify whether they want the read to be eventually consistent or strongly consistent:

- Eventually consistent reads (the default) – The eventual consistency option maximizes your read throughput. However, an eventually consistent read might not reflect the results of a recently completed write. All copies of data usually reach consistency within a second. Repeating a read after a short time should return the updated data.
- Strongly consistent reads — In addition to eventual consistency, Amazon DynamoDB also gives you the flexibility and control to request a strongly consistent read if your application, or an element of your application, requires it. A strongly consistent read returns a result that reflects all writes that received a successful response before the read.
- ACID transactions – DynamoDB transactions provide developers atomicity, consistency, isolation, and durability (ACID) across one or more tables within a single AWS account and region. You can use transactions when building applications that require coordinated inserts, deletes, or updates to multiple items as part of a single logical business operation.

(Learn more about read consistency:
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.ReadConsistency.html)

Great job!

You have successfully created a DynamoDB table and increased your knowledge in the value(s) of a NoSQL database.

Let's review

In this hands-on activity, you were able to create a DynamoDB table and populate the table with items and their attributes. You can now comfortably explain to your **BitBeat** colleagues the value of the NoSQL database, how simple it is to use, and how it appears to be a good fit for the ever-changing needs of **BitBeat**'s business.

In this activity, you:

- Created a DynamoDB table and assigned its primary and sort keys
- Added items to the DynamoDB table
- Created and modified item attributes
- Performed both query and scan searches of the table

Test your knowledge

- What is DynamoDB? Explain a few reasons why you would choose this database.____
- Explain or describe what a **table** is. ____
- What is a primary key and a sort key? Why are they important?_____
- How would you explain an **item**? ____
- How would you explain an **attribute**? ____
- Do attributes have single type? Y/N____
- If no, can you name three different attribute types you saw in this activity? _______

Don't forget the **bonus task** near the end of this activity.

DID YOU KNOW

DynamoDB items can have an optional sort key to store related attributes in a sorted order. This allows multiple items to be queried as a collection, which simplifies access patterns.

Each table also has a primary key, which represents the table's key or keys. If there is no sort key, the primary and partition keys are the same. If there is a sort key, the primary key is a combination of the partition and sort keys called a composite primary key. DynamoDB has two types of secondary indexes: local and global. These indexes improve the application's ability to access data quickly and efficiently. A local secondary index uses the table's partition key with a different sort key. You are allowed five per table. Local indexes must be created when you create the table.

	Attribute name	Attribute type	Attribute value
PRIMARY KEY	Artist	*String*	Artist Name
SORT KEY	Song	*String*	Song Title
	Album	*String*	Album Name
	Year	*Number*	Date of Year
	Genre	*String*	Music Genre
	LengthSeconds	*Number*	Seconds

Visual framework of database created

Bonus activity – Delete the table

In this task, you will delete the Music table, which will also delete all the data in the table.

1. Click **Delete table.** On the confirmation panel, click **Delete**.

The table will be deleted.

Resources: https://aws.amazon.com/dynamodb/
https://youtu.be/sI-zciHAh-4
https://aws.amazon.com/dynamodb/features/
https://aws.amazon.com/dynamodb/faqs/
https://docs.aws.amazon.com/search/doc-search.html?searchPath=documentation&searchQuery=dynamodb