



AWS News Blog

New – Auto Scaling for Amazon DynamoDB

by [Jeff Barr](#) | on 14 JUN 2017 | in [Amazon DynamoDB](#), [Launch](#), [News](#) | [Permalink](#) | [Share](#)

[Amazon DynamoDB](#) has more than one hundred thousand customers, spanning a wide range of industries and use cases. These customers depend on DynamoDB's consistent performance at any scale and presence in 16 geographic regions around the world. A recent trend we've been observing is customers using DynamoDB to power their serverless applications. This is a good match: with DynamoDB, you don't have to think about things like provisioning servers, performing OS and database software patching, or configuring replication across availability zones to ensure high availability – you can simply create tables and start adding data, and let DynamoDB handle the rest.

DynamoDB provides a provisioned capacity model that lets you set the amount of read and write capacity required by your applications. While this frees you from thinking about servers and enables you to change provisioning for your table with a simple API call or button click in the [AWS Management Console](#), customers have asked us how we can make managing capacity for DynamoDB even easier.

Today we are introducing Auto Scaling for DynamoDB to help automate capacity management for your tables and global secondary indexes. You simply specify the desired target utilization and provide upper and lower bounds for read and write capacity. DynamoDB will then monitor throughput consumption using [Amazon CloudWatch](#) alarms and then will adjust provisioned capacity up or down as needed. Auto Scaling will be on by default for all new tables and indexes, and you can also configure it for existing ones.

Even if you're not around, DynamoDB Auto Scaling will be monitoring your tables and indexes to automatically adjust throughput in response to changes in application traffic. This can make it easier to administer your DynamoDB data, help you maximize availability for your applications, and help you reduce your DynamoDB costs.

Let's see how it works...

Using Auto Scaling

The DynamoDB Console now proposes a comfortable set of default parameters when you create a new table. You can accept them as-is or you can uncheck Use default settings and enter your own parameters:

Table settings

Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created.

☒ Use default settings

- No secondary indexes.
- Auto Scaling capacity set to 70% target utilization, at **NEW!** minimum capacity of 5 reads and 5 writes

Additional charges may apply if you exceed the AWS Free Tier levels for CloudWatch or Simple Notification Service. Advanced alarm settings are available in the CloudWatch management console.

Cancel

Create

Here's how you enter your own parameters:

Auto Scaling

☒ Read capacity

☒ Write capacity

☐ Same settings as read

Target utilization

50

%

50

%

Minimum provisioned capacity

5

units

5

units

Maximum provisioned capacity

1000

units

1000

units

☒ Apply same settings to global secondary indexes

☒ Apply same settings to global secondary indexes

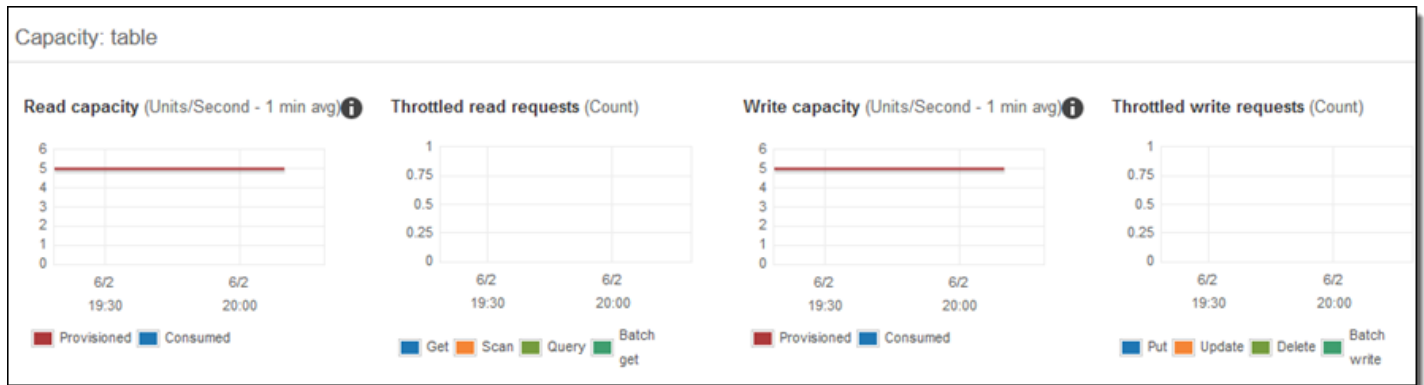
Target utilization is expressed in terms of the ratio of consumed capacity to provisioned capacity. The parameters above would allow for sufficient headroom to allow consumed capacity to double due to a burst in read or write requests (read [Capacity Unit Calculations](#) to learn more about the relationship between DynamoDB read and write operations and provisioned capacity). Changes in provisioned capacity take place in the background.

Auto Scaling in Action

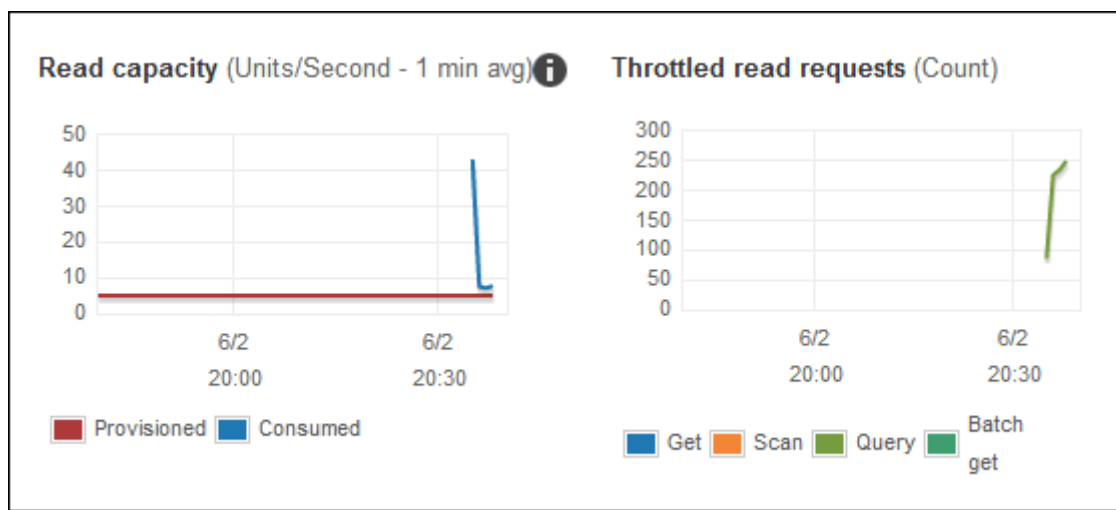
In order to see this important new feature in action, I followed the directions in the [Getting Started Guide](#). I launched a fresh EC2 instance, installed (`sudo pip install boto3`) and configured (`aws configure`) the [AWS SDK for Python](#). Then I used the code in the [Python and DynamoDB](#) section to create and populate a table with some data, and manually configured the table for 5 units each of read and write capacity.

I took a quick break in order to have clean, straight lines for the CloudWatch metrics so that I could show the

effect of Auto Scaling. Here's what the metrics look like before I started to apply a load:



I modified the code in [Step 3](#) to continually issue queries for random years in the range of 1920 to 2007, ran a single copy of the code, and checked the read metrics a minute or two later:



The consumed capacity is higher than the provisioned capacity, resulting in a large number of throttled reads. Time for Auto Scaling!

I returned to the console and clicked on the **Capacity** tab for my table. Then I clicked on **Read capacity**, accepted the default values, and clicked on **Save**:

Movies [Close](#)

[Overview](#)
[Items](#)
[Metrics](#)
[Alarms](#)
[Capacity](#)
[Indexes](#)
[Triggers](#)
[Access control](#)
[Tags](#)

▶ Scaling activities

Provisioned capacity

Read capacity units

Write capacity units

Table

⚠ Consumed read capacity ≥ 4 for 5 minutes

Estimated cost \$2.91 / month ([Capacity calculator](#))

Auto Scaling

☒ **Read capacity**
☐ **Write capacity**

Target utilization %

Minimum provisioned capacity units

Maximum provisioned capacity units

☒ Apply same settings to global secondary indexes

IAM Role I authorize DynamoDB to scale capacity using the following role:

☒ New role: **DynamoDBAutoscaleRole**
☐ Existing role with pre-defined policies [\[Instructions\]](#)

Role Name*

DynamoDB created a new IAM role (**DynamoDBAutoscaleRole**) and a pair of CloudWatch alarms to manage the Auto Scaling of read capacity:

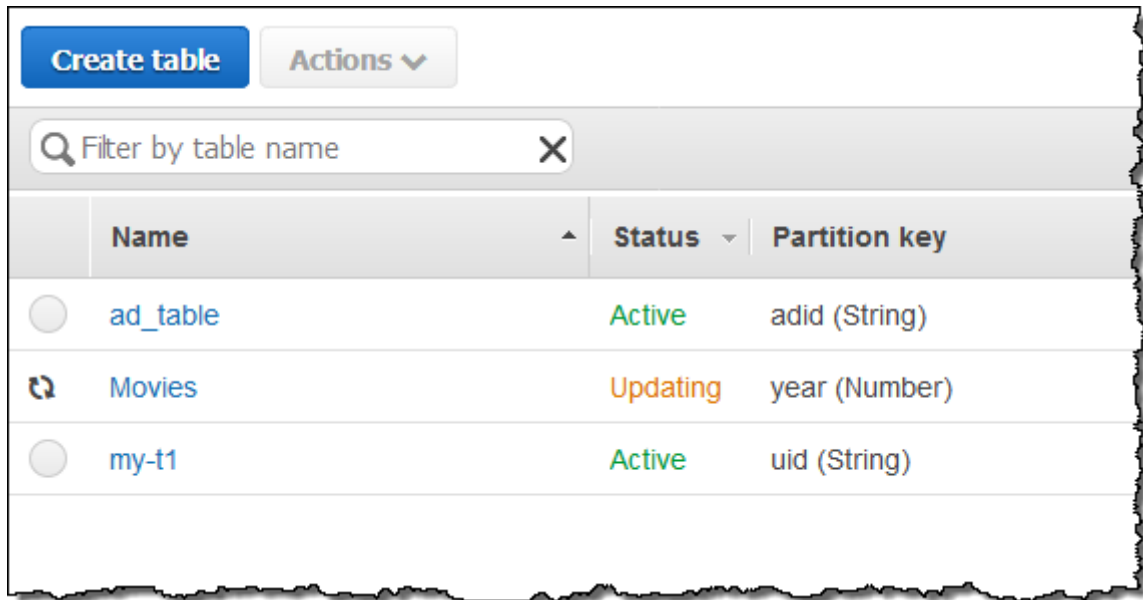
[Create Alarm](#)
[Add to Dashboard](#)
[Actions](#)

Filter: [State is OK](#)

State	Name	Threshold
<input checked="" type="checkbox"/> OK	TargetTracking-table/Movies-AlarmHigh-a5c84852-48ce-465e-a296-8aba285daece	ConsumedReadCapacityUnits > 504 for 5 minutes
<input type="checkbox"/> OK	TargetTracking-table/Movies-AlarmLow-7217b2b1-cd50-4594-96c7-6b1c551c828f	ConsumedReadCapacityUnits < 360 for 15 minutes

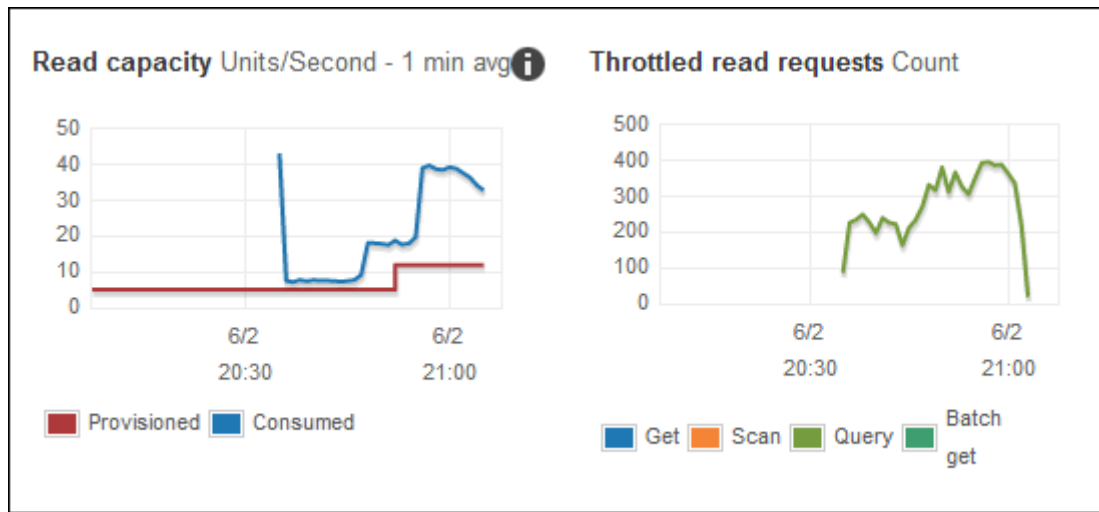
DynamoDB Auto Scaling will manage the thresholds for the alarms, moving them up and down as part of the

scaling process. The first alarm was triggered and the table state changed to **Updating** while additional read capacity was provisioned:

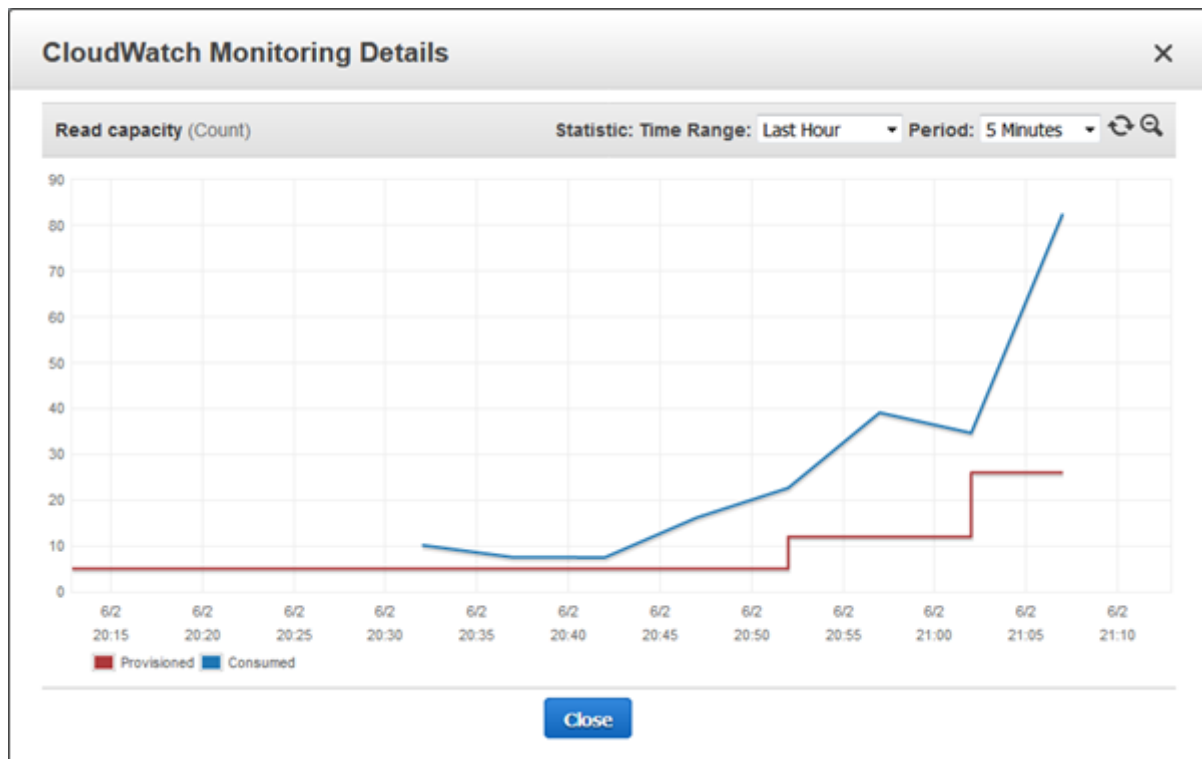


	Name	Status	Partition key
<input type="radio"/>	ad_table	Active	adid (String)
<input checked="" type="radio"/>	Movies	Updating	year (Number)
<input type="radio"/>	my-t1	Active	uid (String)

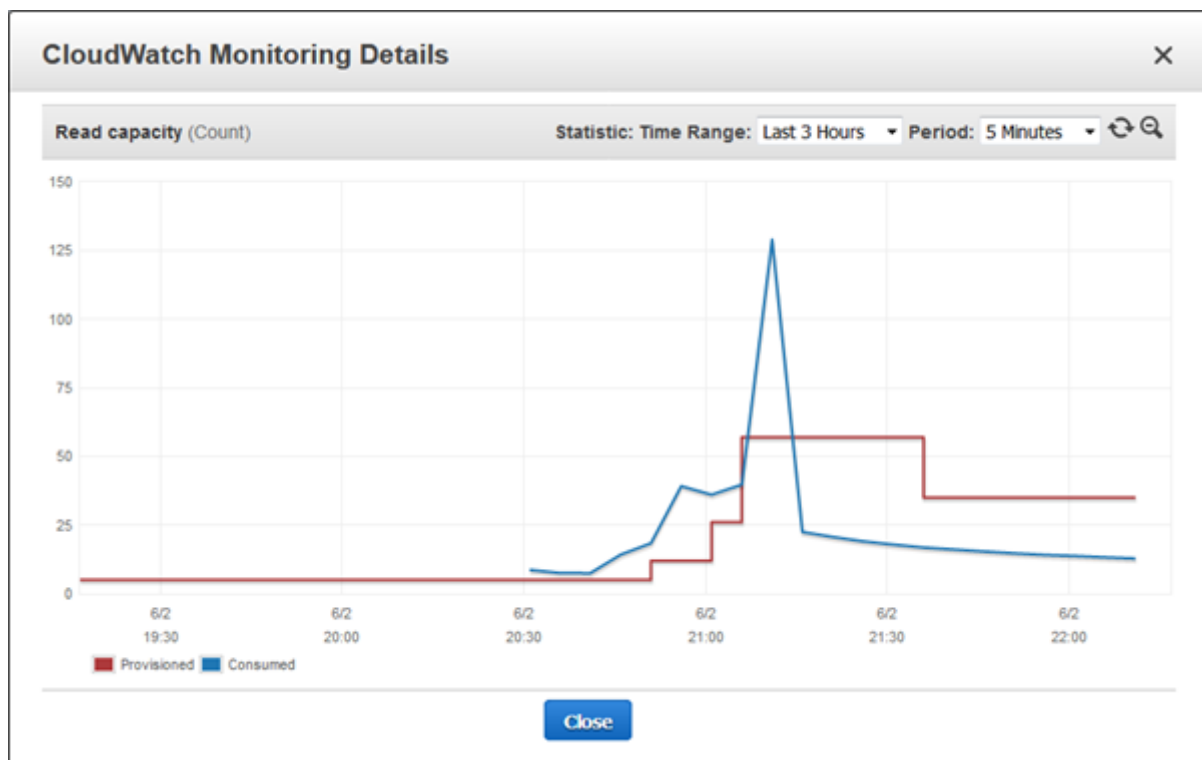
The change was visible in the read metrics within minutes:



I started a couple of additional copies of my modified query script and watched as additional capacity was provisioned, as indicated by the red line:



I killed all of the scripts and turned my attention to other things while waiting for the scale-down alarm to trigger. Here's what I saw when I came back:



The next morning I checked my **Scaling activities** and saw that the alarm had triggered several more times overnight:

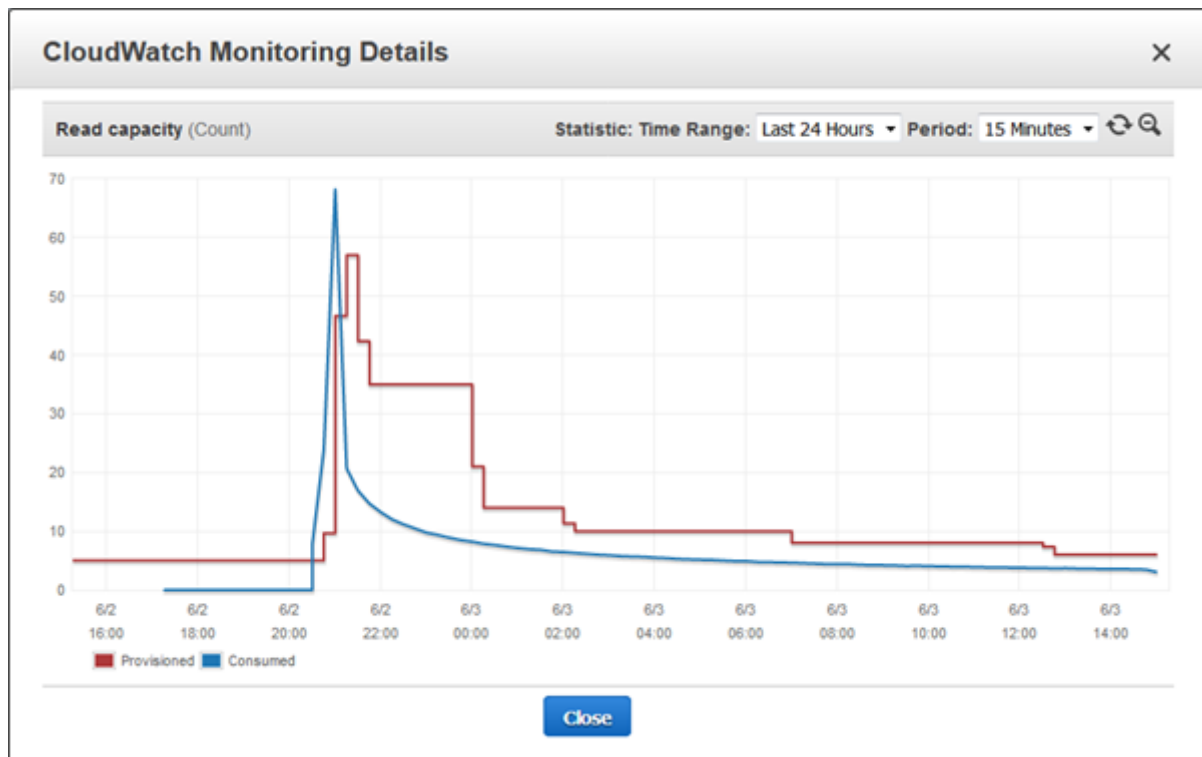
Movies [Close](#)

Overview Items Metrics Alarms **Capacity** Indexes Triggers Access control Tags

▼ Scaling activities

Start Time	End Time	Target	Capacity Unit	Scaling Activity
Sat Jun 03 05:39:20 GMT-700 2017	Sat Jun 03 05:39:57 GMT-700 2017	table/Movies	Read	Setting read capacity units to 6.
Fri Jun 02 23:54:04 GMT-700 2017	Fri Jun 02 23:54:40 GMT-700 2017	table/Movies	Read	Setting read capacity units to 8.
Fri Jun 02 19:04:03 GMT-700 2017	Fri Jun 02 19:04:35 GMT-700 2017	table/Movies	Read	Setting read capacity units to 10.
Fri Jun 02 17:00:30 GMT-700 2017	Fri Jun 02 17:01:03 GMT-700 2017	table/Movies	Read	Setting read capacity units to 14.

This was also visible in the metrics:



Until now, you would prepare for this situation by setting your read capacity well about your expected usage, and pay for the excess capacity (the space between the blue line and the red line). Or, you might set it too low, forget to monitor it, and run out of capacity when traffic picked up. With Auto Scaling you can get the best of both worlds: an automatic response when an increase in demand suggests that more capacity is needed, and another automated response when the capacity is no longer needed.

Things to Know

DynamoDB Auto Scaling is designed to accommodate request rates that vary in a somewhat predictable, generally periodic fashion. If you need to accommodate unpredictable bursts of read activity, you should use Auto Scaling in combination with [DAX](#) (read [Amazon DynamoDB Accelerator \(DAX\) – In-Memory Caching for](#)

[Read-Intensive Workloads](#) to learn more). Also, the AWS SDKs will detect throttled read and write requests and retry them after a suitable delay.

I mentioned the **DynamoDBAutoscaleRole** earlier. This role provides Auto Scaling with the privileges that it needs to have in order for it to be able to scale your tables and indexes up and down. To learn more about this role and the permissions that it uses, read [Grant User Permissions for DynamoDB Auto Scaling](#).

Auto Scaling has complete CLI and API support, including the ability to enable and disable the Auto Scaling policies. If you have some predictable, time-bound spikes in traffic, you can programmatically disable an Auto Scaling policy, provision higher throughput for a set period of time, and then enable Auto Scaling again later.

As noted on the [Limits in DynamoDB](#) page, you can increase provisioned capacity as often as you would like and as high as you need (subject to per-account limits that we can increase on request). You can decrease capacity up to nine times per day for each table or global secondary index.

You pay for the capacity that you provision, at the regular [DynamoDB prices](#). You can also purchase [DynamoDB Reserved Capacity](#) to further savings.

Available Now

This feature is available now in all regions and you can start using it today!

— [Jeff](#);

TAGS: [Amazon DynamoDB](#)



AWS Podcast

Subscribe for weekly AWS news and interviews

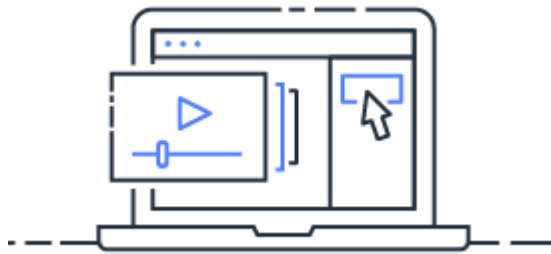
[Learn more »](#)



AWS Partner Network

Find an APN member to support your cloud business needs

[Learn more »](#)



AWS Training & Certifications

Free digital courses to help you develop your skills

[Learn more »](#)

Resources

[Getting Started](#)

[What's New](#)

[Top Posts](#)

[Official AWS Podcast](#)

[Case Studies](#)

Follow

 [Twitter](#)

[Facebook](#)[LinkedIn](#)[Twitch](#)[RSS Feed](#)[Email Updates](#)

New Launches From re:Invent

Discover the latest services and features from AWS

[Visit the News Blog »](#)

Related Posts

[Optimizing batch processing with custom checkpoints in AWS Lambda](#)

[Using AWS Lambda for streaming analytics](#)

[Detecting sensitive data in DynamoDB with Macie](#)

[BandLab welcomes users by the millions with AWS](#)

[ICYMI: Serverless pre:Invent 2020](#)

[Detect change points in your event data stream using Amazon Kinesis Data Streams, Amazon DynamoDB and AWS Lambda](#)

[Integrating Amazon ElastiCache with other AWS services: The serverless way](#)

[New – Export Amazon DynamoDB Table Data to Your Data Lake in Amazon S3, No Code Writing Required](#)

