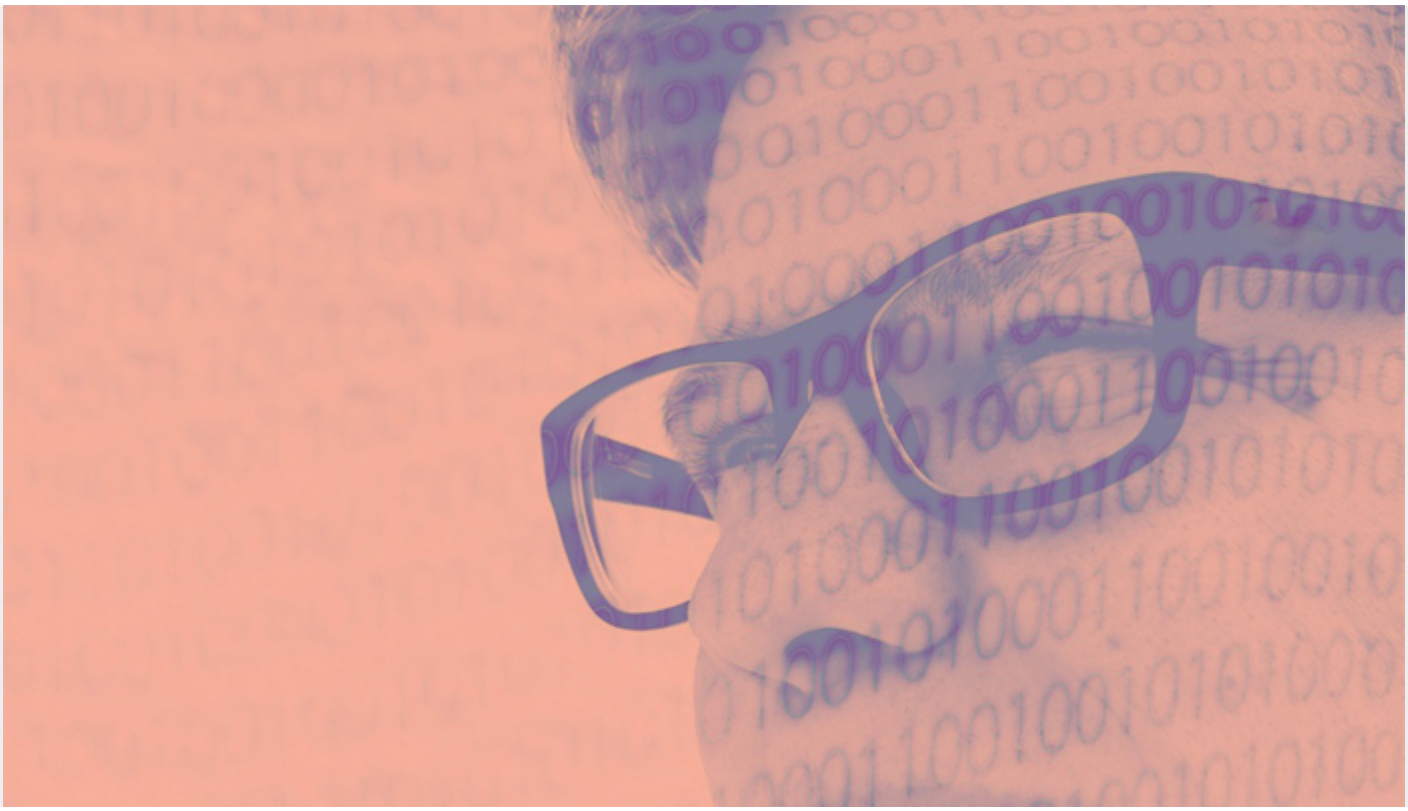
INTRODUCTION TO AMAZON DYNAMODB



DynamoDB is a key-value, noSQL database developed by Amazon. It's unlike some other products offered by Amazon and other vendors in that it's not just an open source system, like Spark, hosted on the vendor's platform. Amazon wrote this for their own internal needs and now they make it available to their customers.

(This tutorial is part of our DynamoDB Guide. Use the right-hand menu to navigate.)

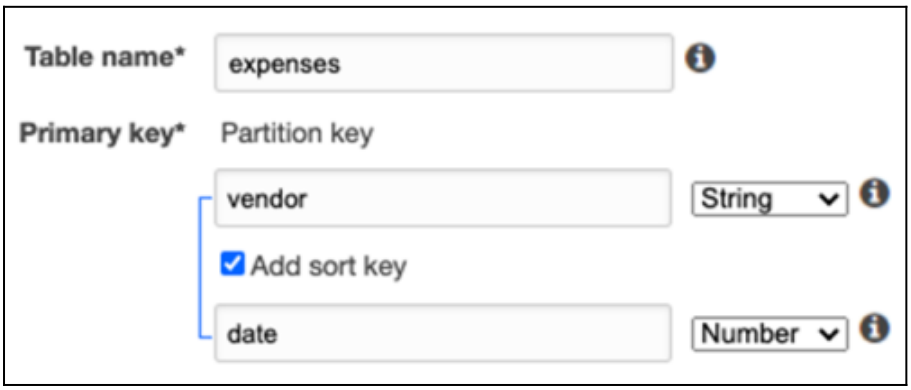
How does DynamoDB work?

DynamoDB looks just like JSON, with the one difference being that each JSON record must include the record key. That has the advantage that it lets you do updates on a record. In a JSON database, like MongoDB, you cannot update records. Instead you must delete them then add back the changed version to effect the same change.

DynamoDB also lets you work with transactions, something that MongoDB supports as well. Not all noSQL databases let you do that. This is important as certain database operations logically must go together. For example, a sales transaction must both decrement inventory and increase cash-on-hand. If one of those two operations failed then the sales and inventory systems would be out of balance.

You work with the database using the AWS command line client, APIs for different programming languages, their NoSQL workbench desktop tool, or on the Amazon AWS website. For example, the screen below shows how you create a table.

Notice that you just create the key. That's because it's JSON, meaning there's no structure, no schema. So all the other attributes can be anything.



DynamoDB Definitions

DynamoDB has these concepts and more:

- **Table**: a collection of items
- **Item**: a collection of attributes. (Other databases call these records or documents.)
- **Stream**: like a cache that holds changes in memory until they are flushed to storage.
- **Partition key**: the primary key. It must be unique.
- **Partition key and sort key:** a composite primary key, meaning a partition key with more than one attribute, like employee name and employee ID (necessary because two employees could have the same name).
- **Secondary indexes**: you can index other attributes that you frequently query to speed up reads.

API and SDK

As with most cloud systems, DynamoDB exposes its services via web services. But that does not mean you have to format your data to JSON and then post it using HTTP. Instead they provide software development kits (SDKs). The SDK takes the requests you send it and then translates that to HTTP calls behind the scenes. In this way, the SDK provides a more natural and far less wordy way to work with the database. The SDK lets you work with DynamoDB as you would work with regular objects.

The SDK has these methods:

- PutItem
- BatchWriteItem
- GetItem
- BatchGetItem
- Query
- Scan
- UpdateItem
- DeleteItem
- ListStreams
- GetShardIterator
- GetRecords

- TransactWriteItems
- TransactGetItems

AWS CLI

As with other Amazon products you can use the AWS command line client. That lets you run database operations from the command line without having to write a program. You use JSON to work with DynamoDB.

For example, there are these operations and a few more:

- aws dynamodb create-table
- aws dynamodb put-item

SDKs

DynamoDB has SDKs for these programing languages:

- Java
- JavaScript
- .NET
- js
- PHP
- Python
- Ruby
- C++
- Go,
- Android
- iOS

For Java and .NET, they provide objects. Those let you work with table items as if they were objects in those programming languages. And it translates from data types, letting you use, for example, you can use dates and bytes instead of being limited to facsimiles of those as strings or numbers, as you would with JSON.

For example, with Java you add this @annotation then can work with the table as if it were a Java object. That way you don't have to do something that is wordier, like using SQL or JSON.

```
@DynamoDBTable(tableName = "ProductCatalog")
    public static class CatalogItem {
        private Integer id;
        private String title;
        private String ISBN;
        private Set bookAuthors;
```

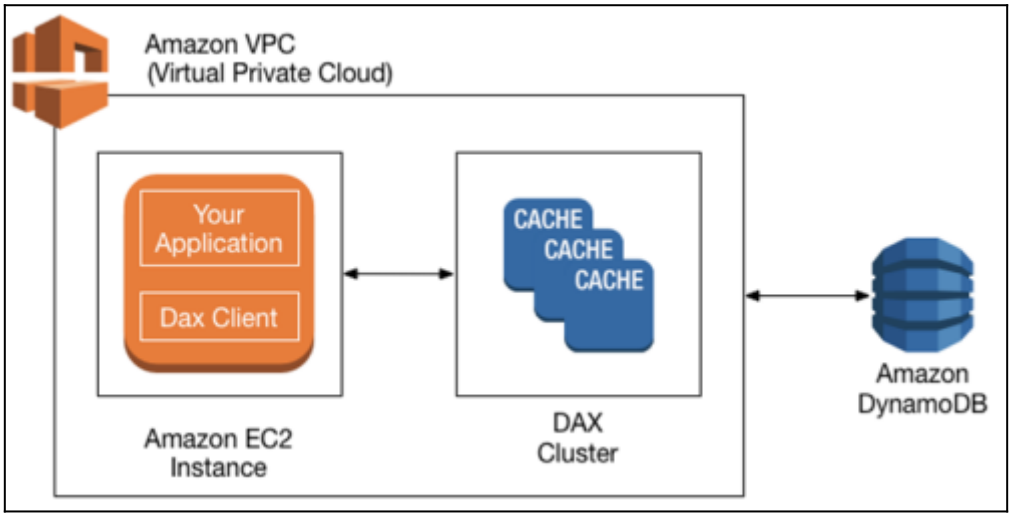
DynamoDB is downloadable

You can also download DynamoDB and run it on your local system. Their web site does not mention whether this is free, meaning whether you could use it forever and never pay. The idea is you use it to test code locally, and thus save money, before committing it to the cloud.

You can, for example, add it as a Maven dependency in your Java project. That means when you run your Java code in the code editor it will download DynamoDB and spin up an instance for you with little to no configuration required.

Amazon DynamoDB Accelerator (DAX)

DAX is an optional feature that turns DynamoDB into an in-memory database.



Source

Integration with other systems

- **Amazon Glue.** In this case you pull data from DynamoDB into Amazon Glue. There you to ETL and then write it out to other systems like the Amazon Redshift data warehouse.
- **Apache Hive on Amazon EMR.** This is more a back-and-forth interface. You can use this to, for example, using HiveQL (the Hive SQL language) to query DynamoDB tables. And you can copy data into Hadoop. (Hive rides atop Hadoop to support EMR, which is the mapReduce operation.) You can also join DynamoDB tables in Hive.

Quotas

DynamoDB users are subject to quotas by Amazon. These are given in **capacity units**. The user can request an additional quota. For example, for a write transaction 1 capacity unit = one write per second up to 1KB.

Transactions, reads, and stream operations are all capacity units as well.

Pricing

As with EC2, there are separate prices for on-demand and provisioned (i.e., dedicated) resources. For a provisioned instance, a write capacity unit is $0.00065. Read is $0.00013.

Storage prices vary according to the Amazon data center. US-East in Ohio is $0.25 per 25 GB.

DAX is priced by the hour and varies according to virtual CPU and memory size. For example, 1 vCPU and 1.5 GB memory is $0.04 per hour.

Additional resources

For more on this topic, explore the BMC Big Data & Machine Learning Blog or check out these resources:

- AWS Guide, with 15+ articles and tutorials on AWS
- Availability Regions and Zones for AWS, Azure & GCP
- Databases on AWS: How Cloud Databases Fit in a Multi-Cloud World
- An Introduction to Database Reliability
- MongoDB Guide & Apache Cassandra Guide, each with articles and tutorials