Couchbase

DynamoDB Users:

Why Switching to Couchbase Will Save You Time and Money

EXECUTIVE SUMMARY

Amazon's DynamoDB database gives Amazon Web Services (AWS) customers easy access to a NoSQL system. Many AWS-based projects store their data in DynamoDB because of the presumed benefits of its tight integration with other AWS tools – but that doesn't always make it the best tool for the job, especially since the quality of integration depends entirely on the willingness and technical know-how of the customers who choose it.

DynamoDB setups can run into performance issues and ballooning costs as the workload scales up, and successful implementation requires a lot of detailed planning. The system's eventual consistency model also causes problems when your data needs to be available in real time, and the DynamoDB platform's proprietary query language may force you to retrain programmers and DBAs whose SQL-based experience simply doesn't apply to this system.

Choosing Couchbase from the start can help you avoid these unexpected issues and vendor lock-in, preserving your ability to move data at will across clouds, an increasingly important feature in the hybrid world in which today's enterprises operate.

In this paper, you will see how Couchbase was built from the ground up to provide excellent performance under many different workloads, with real-time data consistency on a global scale and a SQL for JSON query language that lets your staff apply the lessons they learned in the SQL-based world of relational databases.

Challenges With Amazon's DynamoDB

AWS is the backend platform for many web apps, cloud computing services, and other modern IT applications. AWS is a proven foundation for many use cases, ranging from hobbyist trifles to a full enterprise-scale replacement for traditional datacenters. Amazon's virtual IT shop includes a number of database options, including the company's homespun versions of relational and NoSQL databases.

Naturally, the company will try to sell its own database platforms at every turn. Amazon's landing page for AWS databases[1] lists third-party solutions such as Oracle, Microsoft SQL Server, MongoDB, or Couchbase in small print, making them look like lesser alternatives. Not surprisingly, it pushes Amazon's in-house versions of these database platforms with bolded, headline-style bullet points. An informed user can still set up a Couchbase or Oracle server in AWS – Amazon just doesn't make this obvious at a glance.

Hence, many projects simply stick with default choices like Amazon Aurora (relational) or DynamoDB (NoSQL) even if another database would serve them better. In many cases, the Couchbase NoSQL system beats DynamoDB in terms of operating costs, data quality, and ease of use. For example, AWS isn't built to effectively handle data at the edge. (And really, can you imagine uploading data to Amazon's cloud from an oil platform in the Persian Gulf?) Couchbase is built to scale at every instance, whether hosted in the cloud or on premises.

To be clear, DynamoDB is a great product for certain use cases. Supported by Amazon since 2012, the system manages both document-oriented databases and flat key-value stores. Administrators have fine-grained control over the system's security and access features. When the workload consists of more reading than writing and only basic key-value queries, using data sized at less than 400 kilobytes per object, and without a need for instant data consistency across load-balanced zones, DynamoDB might work for you.

But even that list of favorable conditions for use couldn't help but reveal some of DynamoDB's shortcomings.

Data items can't be larger than 400 KB. Even below that hard limit, Amazon charges significantly higher operating prices when items exceed 1 KB in size.

DynamoDB runs across arrays of load-sharing servers and solid-state storage devices. That's a big upgrade over magnetic-platter hard drives but nowhere near as fast as Couchbase's in-memory processing model and built-in caching functions.

You can add caching to a DynamoDB installation, but that will raise your costs again while making the whole process more difficult to set up and manage.

DynamoDB defaults to an eventual consistency approach[2] with an option for strongly consistent reads across geographic regions and availability zones. The developer guide notes that new or changed data should be consistent throughout your systems within one second or so. That's not always good enough. Most use cases will require strongly consistent reads.

[1] https://aws.amazon.com/products/databases/

[2] https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.ReadConsistency.html

This platform is also firmly tied to Amazon and AWS. Today, most enterprises are looking at a hybrid or a multi-cloud strategy to run workloads in the best execution environment or to meet security, governance, and compliance requirements across the globe. Single-cloud database-as-a-service offerings like DynamoDB and Microsoft CosmosDB do not satisfy these customer requirements, instead locking them into one platform.

Couchbase's cloud strategy, on the other hand, preserves choice while giving customers complete control and ownership of their data. Couchbase was built using a cloud-native architecture as a scale-out distributed database that can be deployed anywhere across physical, virtual, or containerized infrastructures. Couchbase's cloud offerings leverage these architectural advantages to focus on enabling customers to address their hybrid and multi-cloud requirements with a consistent data platform that can be deployed across on premises, edge, and cloud deployments.

Furthermore, database administrators and developers with training and experience in traditional SQL queries will find that DynamoDB relies on simple key-value lookups with some filtering[3] and is only based on JSON data structures.

You need to have a clear idea of what your database workload will look like before signing up for a DynamoDB server. Otherwise, Amazon's rigid provisioning of system assets could become a real headache. Hitting the upper limits of your provisioned throughput simply stops your data flow with the error message **ProvisionedThroughputExceededException** until the limit resets. For apps with dynamic, unpredictable, or spiky traffic schedules, that's a show-stopper.

Examples

This is where Couchbase comes in.

Top-shelf performance is at the heart of Couchbase's design. The database evolved from an in-memory data caching system before merging with parts of the established CouchDB database platform.[4] Memory caching is the name of the game here, keeping Couchbase data in the speediest data store available on modern computer systems.

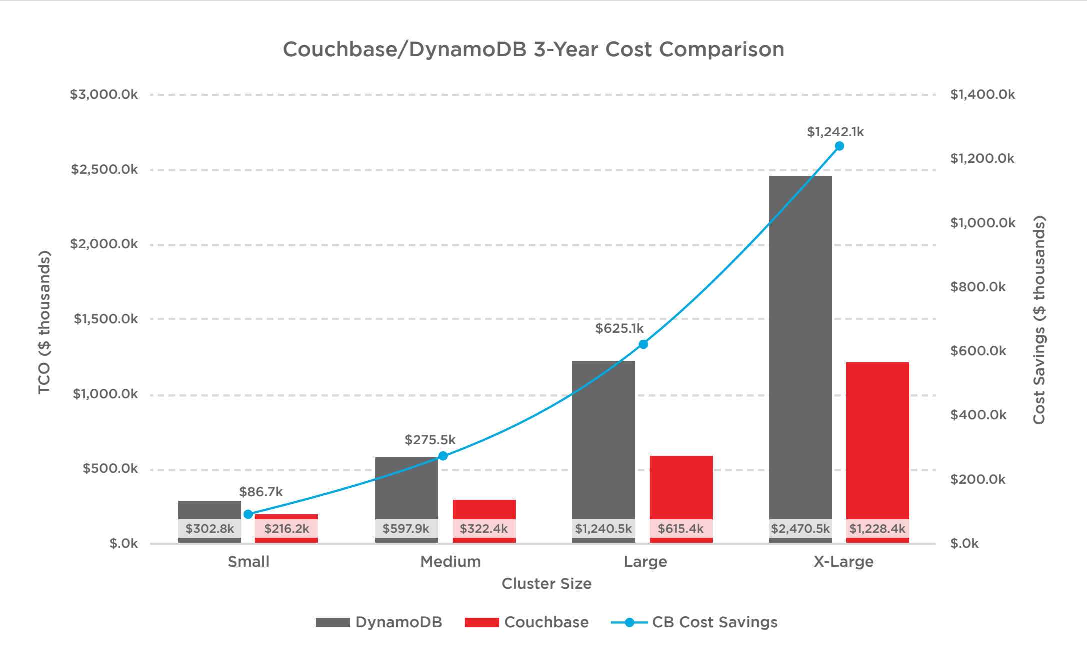
That's why Couchbase consistently delivers extreme performance, even under stress tests with large datasets. A simple 4-node cluster can process more than 30,000 operations per second (OPS) when running a database filter. A 10-node installation blazes through 80,000 OPS. Tests of a DynamoDB system under similar circumstances and handling similar throughput could cost you **over a half-million dollars** in license and support fees for less performance.[5]

Consider the toll that might take over three years of running essential workloads. Choose AWS and DynamoDB and, as the chart below shows, you'll pay **double** the cost of Couchbase at the highest tier – processing up 250,000 reads and writes per second – and 29% more running just 30,000 reads and writes per second. **With Couchbase, the savings grow as your workloads scale.**

[3] https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/SQLtoNoSQL.ReadData.Query.html

[4] https://en.wikipedia.org/wiki/Couchbase_Server

[5] https://www.scylladb.com/2018/12/13/scylla-vs-amazon-dynamodb/



The reasons for making the switch to Couchbase don't stop there. Couchbase's SQL-based query language is compliant with ANSI-92 SQL standards, allowing experienced developers and database administrators to make use of their hard-won query building skills. The database is available in a free, open source version for running tests or just getting your feet wet, as well as a fully supported enterprise offering. You can design your Couchbase clusters and regions in many different ways, making use of both cloud-based and self-hosted servers in a hybrid cloud model.

Conclusion

In most cases, Couchbase will both give your cloud-based application better performance and run at a lower cost than DynamoDB. You get the best of both worlds while relying on a more familiar query language and a more flexible model for systems management.

To learn more, contact your Couchbase sales representative today or visit:
https://www.couchbase.com/partners/amazon

About Couchbase

Couchbase's mission is to be the database platform that enables a revolution in application innovation. To make this possible, Couchbase created an enterprise-class NoSQL database to help deliver ever-richer and ever more personalized customer and employee experiences. Built with the most powerful NoSQL technology, Couchbase was architected on top of an open source foundation for the massively interactive enterprise. Our geo-distributed database provides unmatched developer agility and manageability, as well as unparalleled performance at any scale, from any cloud to the edge.

Couchbase has become pervasive in our everyday lives; our customers include industry leaders Amadeus, AT&T, BD (Becton, Dickinson and Company), Carrefour, Cisco, Comcast, Disney, DreamWorks Animation, eBay, Marriott, Neiman Marcus, Tesco, Tommy Hilfiger, United, Verizon, Wells Fargo, as well as hundreds of other household names. For more information, visit www.couchbase.com.