

# Amazon DynamoDB



Amazon ist nicht nur der grösste Internet-Shop. Amazon hat auch super interessante Technologien im Einsatz. Vor kurzem habe ich zum Beispiel über den Suchalgorithmus für "Andere Kunden kauften auch" gelesen.

Aber noch viel spannender ist das Thema [Amazon DynamoDB](#). Dynamo ist ein alternatives Datenbankkonzept, das auch unter dem Oberbegriff "NoSQL" auftaucht. Dabei geht es um eine absolut ausfallsichere Technologie zum Speichern von riesigen Datenmengen und einfache Skalierbarkeit.

Man kann sich denken, dass auf Amazon alle diese Anforderungen in hohem Masse zutreffen. Doch was steckt hinter dem Begriff DynamoDB?

## Verfügbarkeit

Nun, für Amazon ist das wichtigste überhaupt, dass die Daten immer verfügbar sind, selbst wenn ein ganzes Rechenzentrum ausfallen sollte. Um das zu gewährleisten, verteilt Amazon die eingebundenen physikalischen Rechner auf verschiedene Rechenzentren, sodass die Gefahr eines Hardware-Ausfalls minimiert wird.

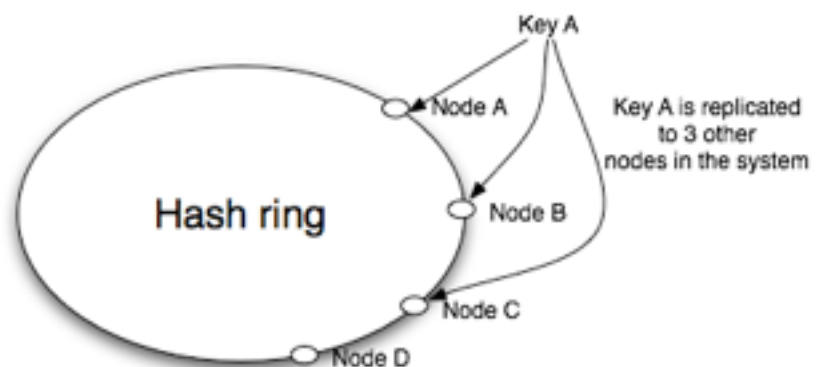
Jeder dieser Rechner beherbergt mehrere virtuelle Rechner, die auch als Knoten bezeichnet werden. Diese Knoten sind logisch in einen Kreis angeordnet, wobei der letzte Knoten wieder auf den ersten Knoten verweist.

## Datensicherheit

Alle Knoten in diesem Ring sind gleichberechtigt, wobei immer ein Knoten für eine Transaktion die Rolle des "Koordinators" übernimmt. Dieser Koordinator sorgt dafür, dass die Transaktion auf mehrere Knoten verteilt wird. Wird zum Beispiel etwas über Knoten A gespeichert, dann sorgt Knoten A dafür, dass diese Daten auch von Knoten B und von Knoten C gespeichert werden.

Durch die Anordnung der Knoten und physikalischen Rechner ist damit gewährleistet, dass die Daten immer auf mehreren Rechnern und in verschiedenen Rechenzentren gespeichert werden.

Fällt ein Knoten aus, dann schickt der Koordinator das



Datenpaket einfach zum nächsten Knoten, der dieses Paket dann temporär für den ausgefallenen Knoten speichert. Sobald der ausgefallene Knoten wieder verfügbar ist, bekommt er alle temporär für ihn gespeicherten Datenpakete wieder zurück.

## Skalierbarkeit

Wenn das System überlastet ist, werden einfach zusätzliche Knoten in den Ring eingefügt und so die Last aufgeteilt. Das Verteilen der Datenpakete auf die einzelnen Knoten geschieht über das Zuordnen von Hash-Werten. Das ist wie in einer Adresskartei. Alle Adressen beginnend mit A sind in Kartei 1, alle Adressen beginnend mit B in Kartei 2, usw.

Amazon DynamoDb ist eine Key / Value-Datenbank. Das bedeutet, alle Datenpakete haben einen eindeutigen Schlüssel und einen beliebigen Wert. Dies unterscheidet DynamoDb von herkömmlichen relationalen Datenbanken, ist aber die Voraussetzung dafür, um die Daten so verteilt zu speichern.

Die Schlüssel werden mit MD5 in einen Hash-Wert umgerechnet und der Hash-Wert ist der Verweis auf den Knoten, auf dem die Daten gespeichert werden. Alle Hash-Werte, die mit "00" beginnen, könnten zum Beispiel auf Knoten 1 und redundant auf Knoten 2 und 3 gespeichert werden.

Wenn ein Knoten überlastet ist, wird der Wertebereich des Hash, der dem Knoten zugeordnet ist, auf weitere Knoten aufgeteilt. In unserem Beispiel wäre dann Knoten 1 nur mehr für die Hash-Werte "000" bis "005" zuständig, und ein neuer Knoten für alle weiteren Werte im Bereich beginnend mit "00".

## Datenkonsistenz

Verfügbarkeit steht bei DynamoDb über Konsistenz. Aus diesem Grund werden die Daten dupliziert. Das führt aber unter Umständen dazu, dass unterschiedliche Versionen eines Datenpakets vorhanden sind, wenn zum Beispiel ein Knoten vorübergehend ausgefallen ist.

Aus diesem Grund wird ein Datenpaket auch beim Lesen von verschiedenen Knoten angefordert und deren Konsistenz überprüft. Falls die Pakete nicht übereinstimmen, wird entweder das neueste zurückgeliefert, oder die Applikation sorgt dafür, dass der Konflikt aufgelöst wird.

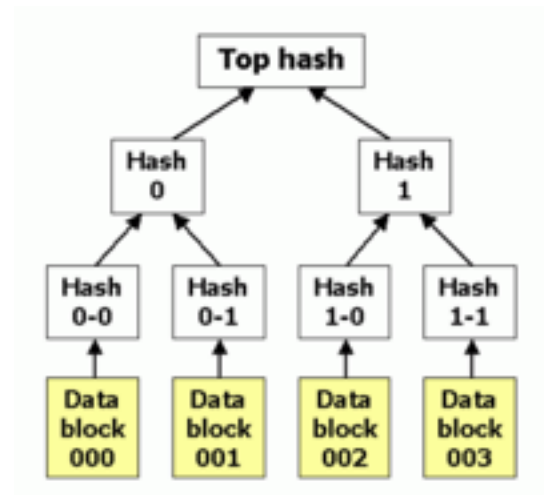
Um die Konsistenz überprüfen zu können, werden die Datenpakete mit einer Versionsnummer gespeichert. DynamoDb ist sogar so flexibel, dass festgelegt werden kann, auf wie vielen Knoten (N) die Datenpakete gespeichert werden müssen, wie viele Schreibvorgänge (W) und wie viele Lesevorgänge (R) erfolgreich sein müssen.

Normalerweise ist das System eingestellt auf (3,2,2). Die Datenpakete werden also auf 3 Knoten geschrieben und von 3 Knoten gelesen, und die Transaktion gilt aber als erfolgreich, wenn beim Schreiben bzw. beim Lesen 2 Knoten erfolgreich sind.

# Merkle-Tree

Wenn ein Knoten ausfällt, werden die Datenpakete, für die er zuständig ist, temporär von einem anderen Knoten übernommen.

Sobald der Knoten wieder aktiv ist, muss er überprüfen, ob seine Datenpakete aktuell sind. Dazu vergleicht der Knoten den Hash-Wert seiner Hash-Werte (Merkle-Tree) mit den Werten seiner Partner-Knoten. Wenn ein Wert nicht übereinstimmt, wird der Merkle-Tree so lange hochgearbeitet, bis das inkonsistente Datenpaket gefunden wird.



## Zusammenfassung

Das klingt jetzt alles gar nicht so kompliziert. Eine genauere Beschreibung gibt es bei Amazon selbst nachzulesen. Ein detaillierter Artikel ist auch auf [Wikipedia](#) zu finden.