



**TECH  
STARTER**

NoSQL -  
Datenbanken

NOSQL



TECH  
STARTER

# Datenmodell

## SQL

- Relationales Datenmodell, das auf Tabellen basiert
- In tabellenartigen Formen organisiert, und Beziehungen zwischen den Tabellen werden durch Fremdschlüssel definiert

## NoSQL

- Verschiedene Datenmodelle
- Key-Value-Stores (wie Redis)
- Graphenmodelle (wie Neo4j)

# Skalierbarkeit

## SQL

- Skalieren oft vertikal, indem mehr Leistung (CPU, RAM) auf einem einzelnen Server hinzugefügt wird

## NoSQL

- Oft bessere horizontale Skalierbarkeit, indem sie auf mehrere Server verteilt werden
- Erleichtert das Hinzufügen von Servern bei Bedarf, um die Datenbankleistung zu verbessern

# Schema

## SQL

- Haben ein festes Schema, das die Struktur der Datenbank und die Art der Daten, die gespeichert werden können, definiert
- Änderungen am Schema erfordern oft aufwändige Migrationen.

## NoSQL

- Sind oft schemafrei oder haben ein flexibleres Schema, das es ermöglicht, Daten ohne vorherige Definition von Tabellenstrukturen zu speichern
- Dies erleichtert die Anpassung an sich ändernde Anforderungen

# BASE vs. ACID

## BASE

- Basically Available, Soft state, Eventually consistent
- Basically Available (Grundlegend verfügbar): Das System bleibt auch unter widrigen Bedingungen (z. B. Netzwerkausfall oder Datenbankpartition) grundlegend verfügbar  
→ Es gibt keine Garantie für sofortige Konsistenz.
- Eventually Consistent (Schließlich konsistent): Das System wird schließlich in einen konsistenten Zustand übergehen, wenn es keine weiteren Änderungen gibt  
→ Es akzeptiert vorübergehend inkonsistente Zustände

# BASE vs. ACID

## ACID

- Atomicity, Consistency, Isolation, Durability
- Atomicity (Atomarität): Eine Transaktion wird als eine atomare Einheit betrachtet, die entweder vollständig ausgeführt wird oder gar nicht. Es gibt keinen Zwischenzustand
- Consistency (Konsistenz): Eine Transaktion bringt die Datenbank von einem konsistenten Zustand in einen anderen konsistenten Zustand
- Isolation (Isolation): Die Ausführung einer Transaktion ist unabhängig von anderen Transaktionen  
→ Jede Transaktion sieht die Datenbank, als ob sie die einzige Transaktion wäre.
- Durability (Dauerhaftigkeit): Nach Abschluss einer Transaktion bleiben die Ergebnisse dauerhaft, auch wenn es zu einem Systemausfall kommt

# Transaktionen

## SQL

- Bieten oft ACID-Transaktionen (Atomicity, Consistency, Isolation, Durability),  
→ stellen sicher, dass Datenbankoperationen konsistent und zuverlässig sind

## NoSQL

- Bieten im Allgemeinen eine flexiblere Herangehensweise an Konsistenz und Transaktionen.
- Einige NoSQL-Datenbanken priorisieren Verfügbarkeit und Partitionstoleranz (CAP-Theorem) und opfern möglicherweise etwas Konsistenz.



# Anwendungsbereiche

## SQL

Werden oft in traditionellen Unternehmensanwendungen verwendet, bei denen komplexe Abfragen und Transaktionen erforderlich sind.

## NoSQL

Werden häufig in Big Data-Anwendungen, Webanwendungen, Echtzeit-Analytik und anderen Szenarien eingesetzt, in denen eine flexible Datenmodellierung und Skalierbarkeit wichtig sind.

Habt ihr Ideen für Anwendungsfälle SQL vs. NoSQL?

JSON



TECH  
STARTER

# Mehr zu SQL vs NoSQL

[Mehr zu SQL vs. NoSQL](#)