Gefangen im CAP Theorem

Warum Datenbanken nicht gut skalieren

Quest Software ist jetzt Teil von Dell.

Gefangen im CAP Theorem
Agenda

- Das CAP Theorem
- Zugriffe in der Oracle Datenbank
  - Lesezugriffe ohne Sperren?
- Alternativen zu relationalen Datenbanken
- Verbessern der Skalierbarkeit
  - Anwendungsdesign
  - Datenbankfeatures
  - Antwortzeiten
- Fazit

Das CAP Theorem

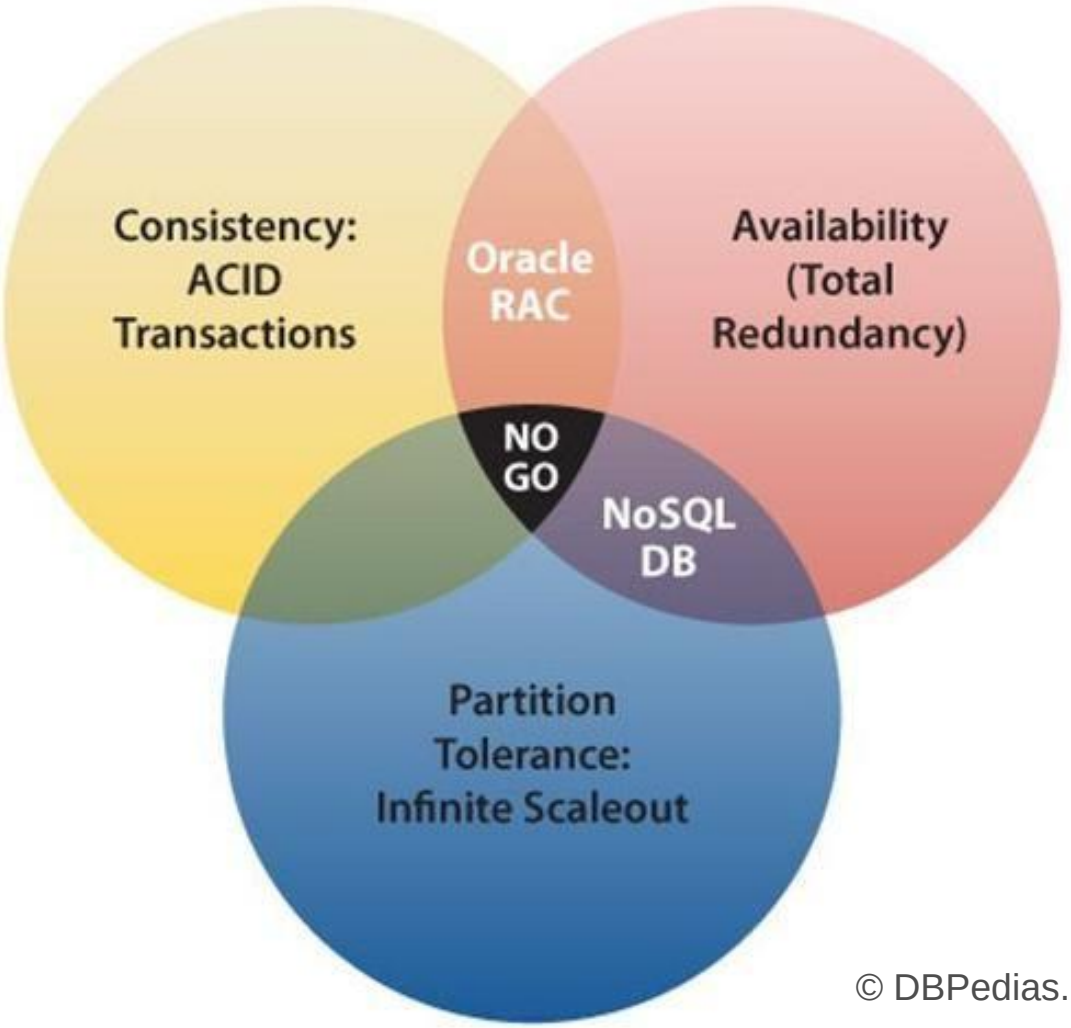
Das CAP Theorem
Relationale Datenbanken

- Betrifft Systeme mit ACID konformen Transaktionen, also auch viele relationale Datenbanken.

- ACID
  - **A**tomar
    - Eine Transaktion wird ganz oder gar nicht durchgeführt.
  - **C**onsistent
    - Die Daten sind zueinander stets konsistent.
  - **I**solated
    - Transaktionen werden nicht von anderen Transaktionen beeinflusst.
  - **D**urable / Dauerhaft
    - Änderungen werden persistent gespeichert.

Das CAP Theorem

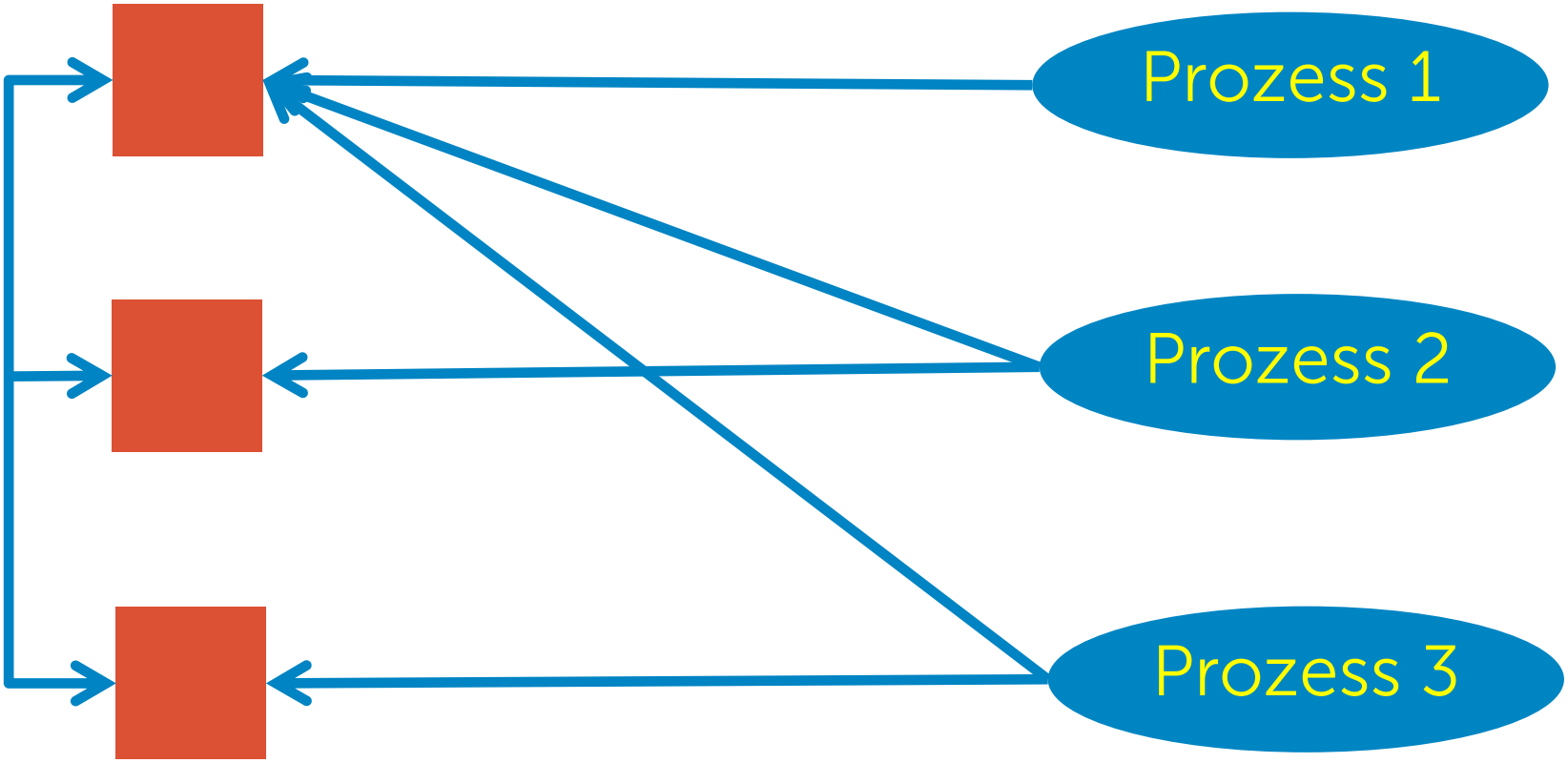
Relationale Datenbanken



© DBPedias.com

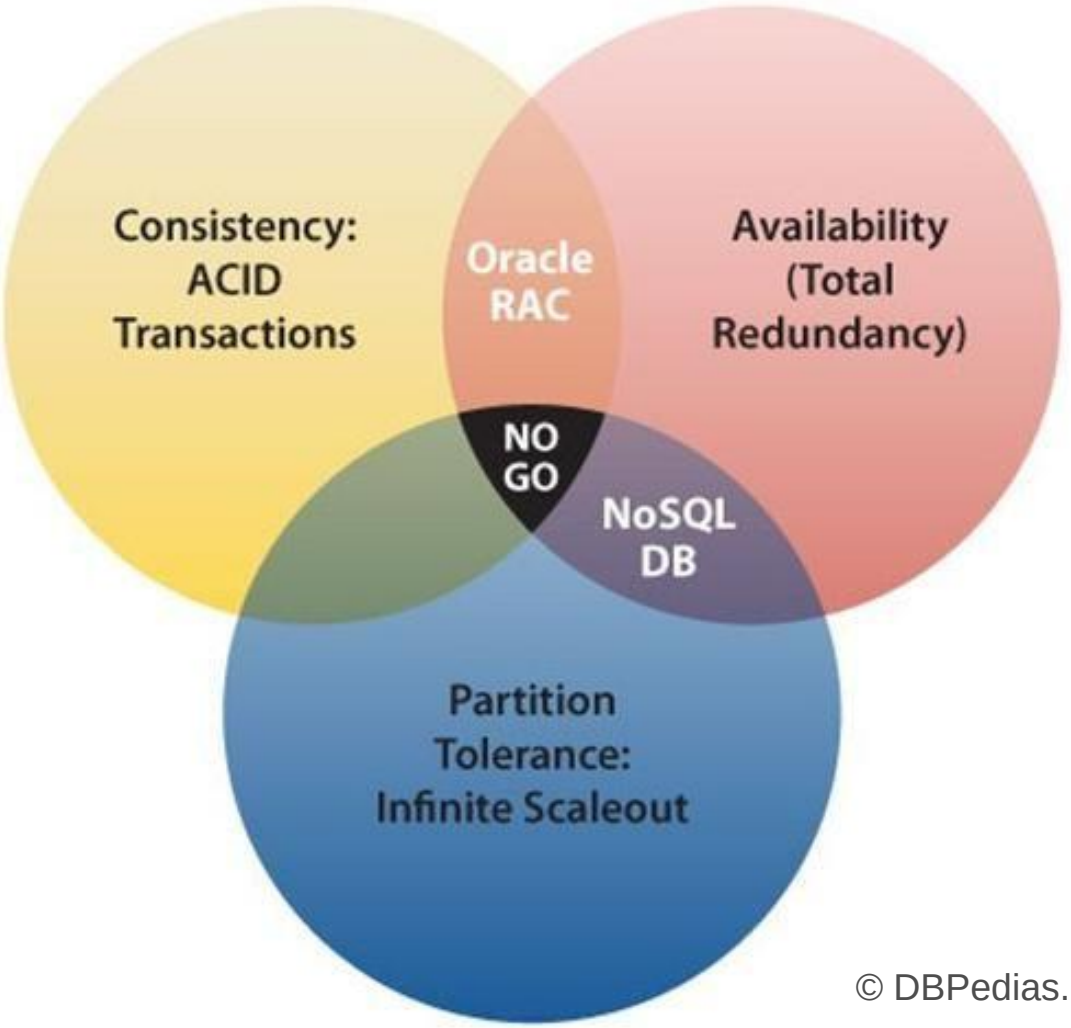
Das CAP Theorem

Relationale Datenbanken



Das CAP Theorem

Relationale Datenbanken



© DBPedias.com

Das CAP Theorem

Relationale Datenbanken

- Datenbanken skalieren nicht, weil Skalierbarkeit nicht wichtig ist.
  - … zumindest nicht so wichtig wie Konsistenz und Verfügbarkeit.
- NoSQL Systeme skalieren gut, weil sie die Daten nicht konsistent halten müssen.
- Systeme, die nicht verfügbar sein müssen, skalieren auch gut.
  - Man hat nur meist nichts davon…

Zugriffe in der Oracle Datenbank

Zugriffe in der Oracle Datenbank
Datenänderungen

- Wird ein Datensatz geschrieben, ist er für weitere schreibende Zugriffe gesperrt.
  - Aber nicht für lesende Zugriffe
- Lesende Zugriffe machen keine Sperren.
  - Wie geht das?
    - UNDO Segmente

- Man sollte also vermeiden, von vielen Sessions den gleichen Datensatz zu schreiben.

Zugriffe in der Oracle Datenbank

Lesezugriffe ohne Sperren?

- Gibt es bei Lesezugriffen keine Sperren?
  - Doch, aber keine Locks.
- Latches sorgen für Serialisierung
  - Prozessynchronisation
- Es werden auch beim Lesen Informationen geschrieben.
  - Eventuelle Buffer Cache Operationen
  - LRU Liste
  - ...

Zugriffe in der Oracle Datenbank

Was bedeutet das für die Skalierbarkeit?

- Auch Lesezugriffe finden nacheinander statt.
  - Keine echten Parallelzugriffe
- Auch hier treten Skalierungseffekte auf.

- Sperren von schreibenden Zugriffen sind schwerwiegender, treten aber meist nicht so häufig auf.

Alternativen zu relationalen Datenbanken

Alternativen zu relationalen Datenbanken

Was also tun, wenn Skalierbarkeit wichtig ist?

- Zugunsten von Skalierbarkeit müssen andere Eigenschaften eingeschränkt werden.
  - Konsistenz
  - Verfügbarkeit
- Ist Verfügbarkeit nicht wichtig, kann man mit synchroner Replikation arbeiten.
  - Das dürfte eher selten der Fall sein.
- Ist Konsistenz nicht wichtig, werden NoSQL Systeme interessant.
  - Auch hier sind die Daten meist konsistent, nur nicht immer.

Alternativen zu relationalen Datenbanken
Häufig genutzte NoSQL Systeme

- Hadoop (HDFS + MapReduce)
  - HDFS: Dateisystem für verteilte Datenhaltung
  - MapReduce: Framework für Berechnungen auf großen Datenmengen
  - Erweiterungen
    - HBase für Datenbankfunktionalitäten
    - Hive für Data Warehouse Funktionalitäten
- Cassandra
  - Verwaltet große, strukturierte Datenbanken
  - Ermöglicht SQL-ähnliche Abfragen

Alternativen zu relationalen Datenbanken

Häufig genutzte NoSQL Systeme

- MongoDB
  - Dokumentenspeicher ohne Schemastrukturen
  - Ermöglicht geschachtelte, hierarchische Strukturen und Indizes
- Amazon Dynamo
  - Verteilte Hashtabelle
  - Ermöglicht Verteilung über Rechner und Rechenzentren

- Es gibt noch viele weitere NoSQL Datenbanken.

Verbessern der Skalierbarkeit

Verbessern der Skalierbarkeit

Ordentliches Anwendungsdesign

- Überarbeitung des Datenmodells
  - Normalisierung
  - Möglichst kurze Tabellen
  - Sinnvolle Datenverteilung
- Weniger informationsgierige Anwendungen
  - Nur die benötigten Daten abfragen
- Optimierte Abfragen
  - Frühzeitige Selektionen vor Join Operationen

- Design ist ein Themenkomplex für sich...

Verbessern der Skalierbarkeit

Datenbankfeatures nutzen

- Nutzung von Indizes
  - Sinnvolle Indizes nutzen
  - Sinnlose Indizes droppen
- Daten puffern, statt wieder abzufragen
  - Client- und Serverside Resultsets nutzen
  - Materialized Views wo sinnvoll
- Wichtige Daten im Cache halten

- Tuning ist ein Themenkomplex für sich...

Verbessern der Skalierbarkeit

Antwortzeiten kurz halten

- Lange Antwortzeiten erhöhen die Gefahr von Kollisionen
- Gute Ausführungspläne erhalten
  - Statistiken aktuell halten
  - Systemstatistiken berücksichtigen
- Gute Statements schreiben
  - Indizes (ja/nein), Join Reihenfolge und Algorithmen
  - Bindevariablen

- Tuning ist ein Themenkomplex für sich...

Fazit

Fazit

Was haben wir gesehen?

- Relationale Datenbanken skalieren nicht, weil Skalierbarkeit das unwichtigste Kriterium ist.
  - Konsistenz und Verfügbarkeit sind wichtiger.
- Trotzdem gibt es Möglichkeiten, die Skalierbarkeit zu verbessern.
- Eventuell sind andere Speicherformen wie NoSQL Systeme sinnvoll.
  - Wenn ständige Konsistenz nicht gefordert wird.

Welche Fragen haben Sie?