NoSQL-Datenbanken

Einführung

Johannes Zschache
Sommersemester 2019

Abteilung Datenbanken, Universität Leipzig
http://dbs.uni-leipzig.de

Inhaltsverzeichnis

- **Organisation**
- **Motivation**
  - Relationale Datenbanken
  - NoSQL-Datenbanken
- **Übersicht zur Vorlesung**
- **Verteilte Datenbanksysteme**
  - Verfügbarkeit und Serialisierbarkeit
  - Eventual Consistency, Sitzungsgarantien und kausale Konsistenz
  - ACID-Garantien
  - CAP-Theorem

Organisation

- 14 Vorlesungstermine: 4.4.2019 – 11.7.2019 (30.5. entfällt)
- Anmeldung zum Modul/Prüfung: AlmaWeb
- Aktuelle Informationen und Folien auf Webseite der Veranstaltung: http://dbs.uni-leipzig.de/stud/2019ss/nosql
- Vorlesungsbegleitende (praktische) Übungen
- Prüfung: Klausur (60 min) am Ende des Semesters
- Allgemeine Literatur (Verlinkungen auf Webseite):
  - L. Wiese: **"Advanced Data Management"**, 2015
  - L. Perkins, Redmond, E. and Wilson, J. R.: **"Seven Databases in Seven Weeks"**, 2nd Edition, 2018
- Weitere Hinweise zur Literatur auf Folien

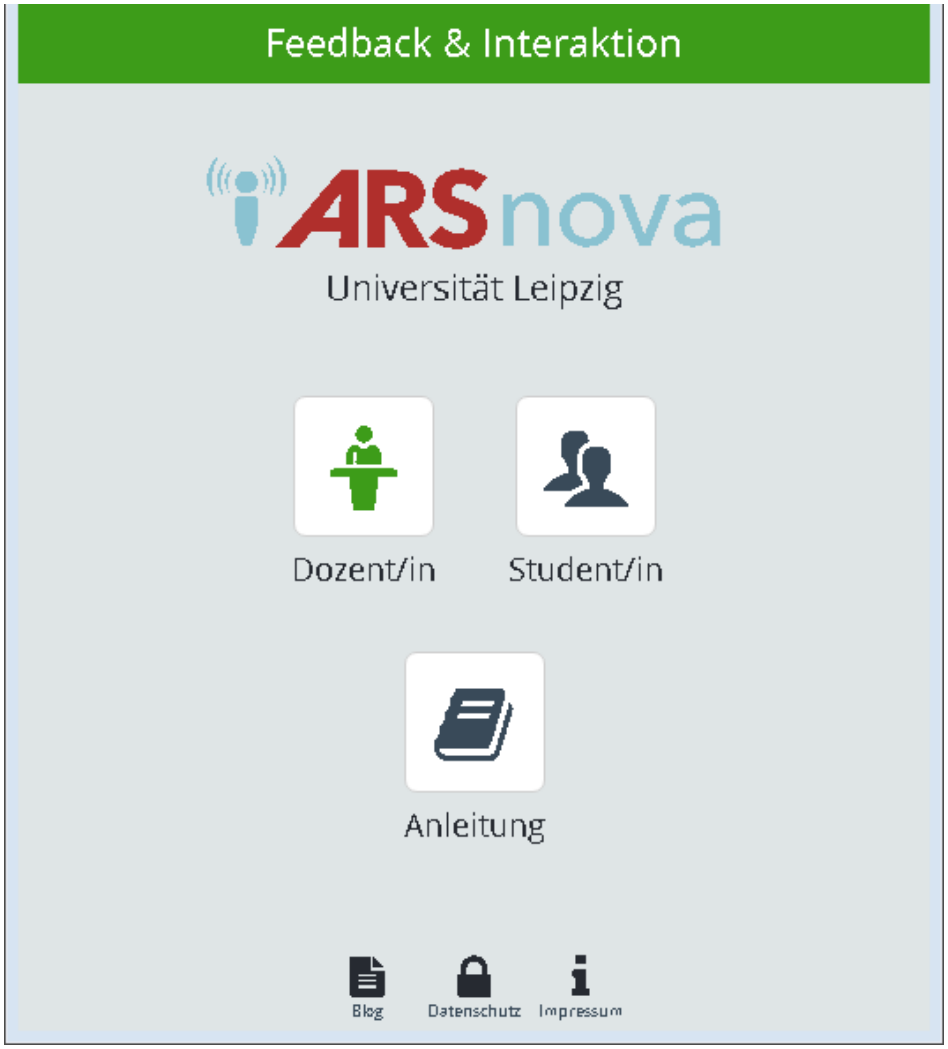
Lehrveranstaltungen SS 2019

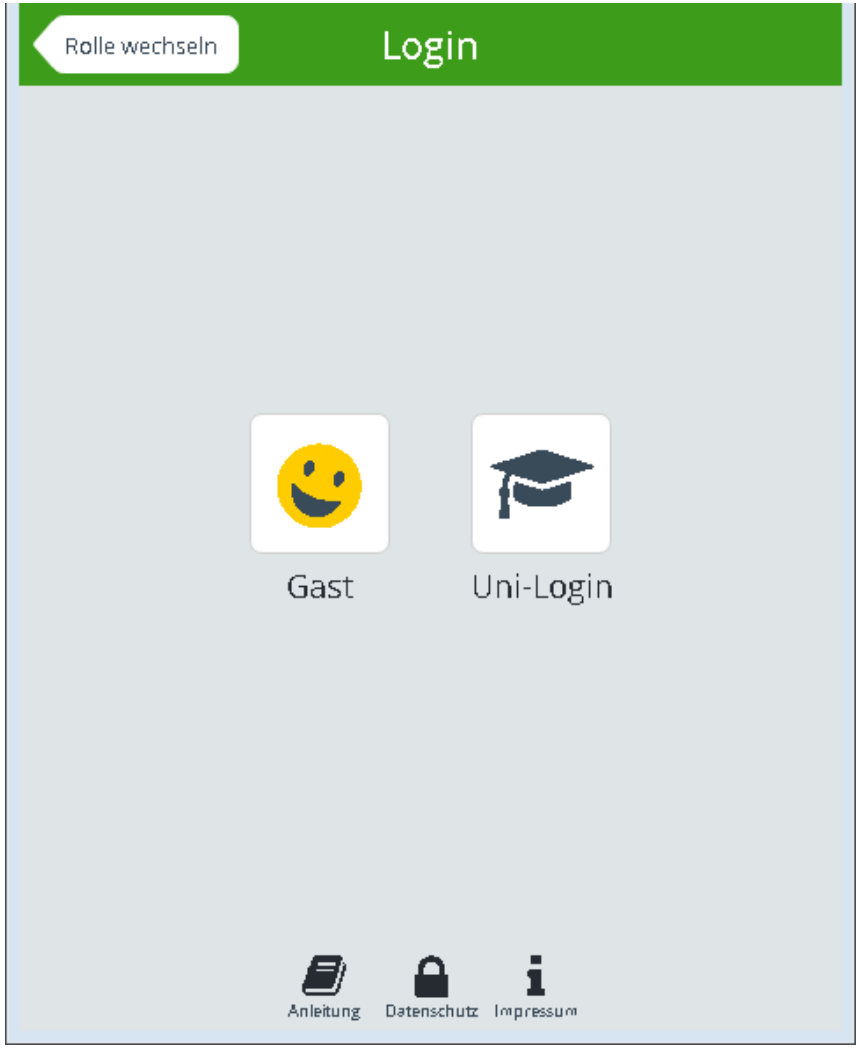
- Lehrveranstaltungen SS 2019: https://dbs.uni-leipzig.de/stud/2019ss
  - Vorlesung + Übung: Datenbanksysteme 2 (DBS2)
  - Relationales Datenbankpraktikum
  - Vorlesung: Cloud Data Management (CDM)
  - Vorlesung: NoSQL-Datenbanken
  - Big Data Praktikum: Einführung am 15.4. um 11:15 Uhr im Raum S 314
  - Bachelor-/Masterseminar: 3.-7.6.2019

- Verwendung der NoSQL-Vorlesung:
  - Bachelor-Modul “Realisierung von Informationssystemen” (CDM + NoSQL)
  - Master-Modul “Anwendungsbezogene Datenbankkonzepte”
    - (kleines) Kernmodul (5LP, zwei Vorlesungen: CDM, NoSQL, ggf. DBS2)
    - (großes) Vertiefungsmodul (10LP, zwei Vorlesungen + Big Data Praktikum)

- Empfohlen für Master Informatik mit Schwerpunkt Big Data

ARSnova: arsnova.rz.uni-leipzig.de 81 60 01 90





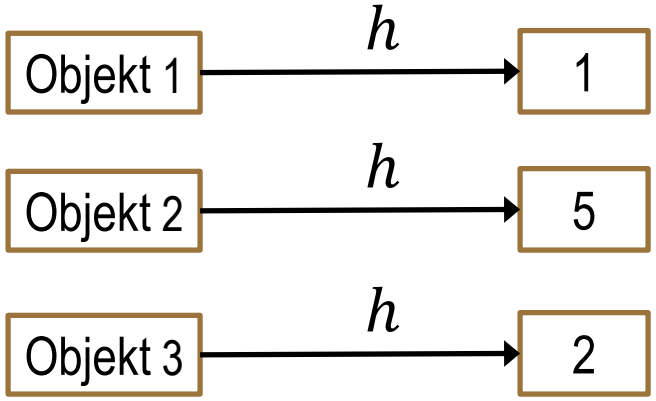
Inhaltsverzeichnis

- **Organisation**
- **Motivation**
  - Relationale Datenbanken
  - NoSQL-Datenbanken
- **Übersicht zur Vorlesung**
- **Verteilte Datenbanksysteme**
  - Verfügbarkeit und Serialisierbarkeit
  - Eventual Consistency, Sitzungsgarantien und kausale Konsistenz
  - ACID-Garantien
  - CAP-Theorem

Relationale Datenbanken

- Relation = Menge von Tupeln (Zeilen) = **Tabelle**
  - Relationale Algebra
  - **Joins**
- **SQL** – Structured Query Language
- **Schema**: Design der Datenbank
  - Feste Anzahl und Reihenfolge der Attribute
  - Vordefinierte Datentypen
  - Einschränkungen (z.B. Not Null, Unique)
  - Referentielle Integrität
  - Normalisierung
- Flexible Anwendbarkeit (Anfragen)
- **ACID** Garantien für Transaktionen
- Geschwindigkeit über Anfrageoptimierung und **Indizes**

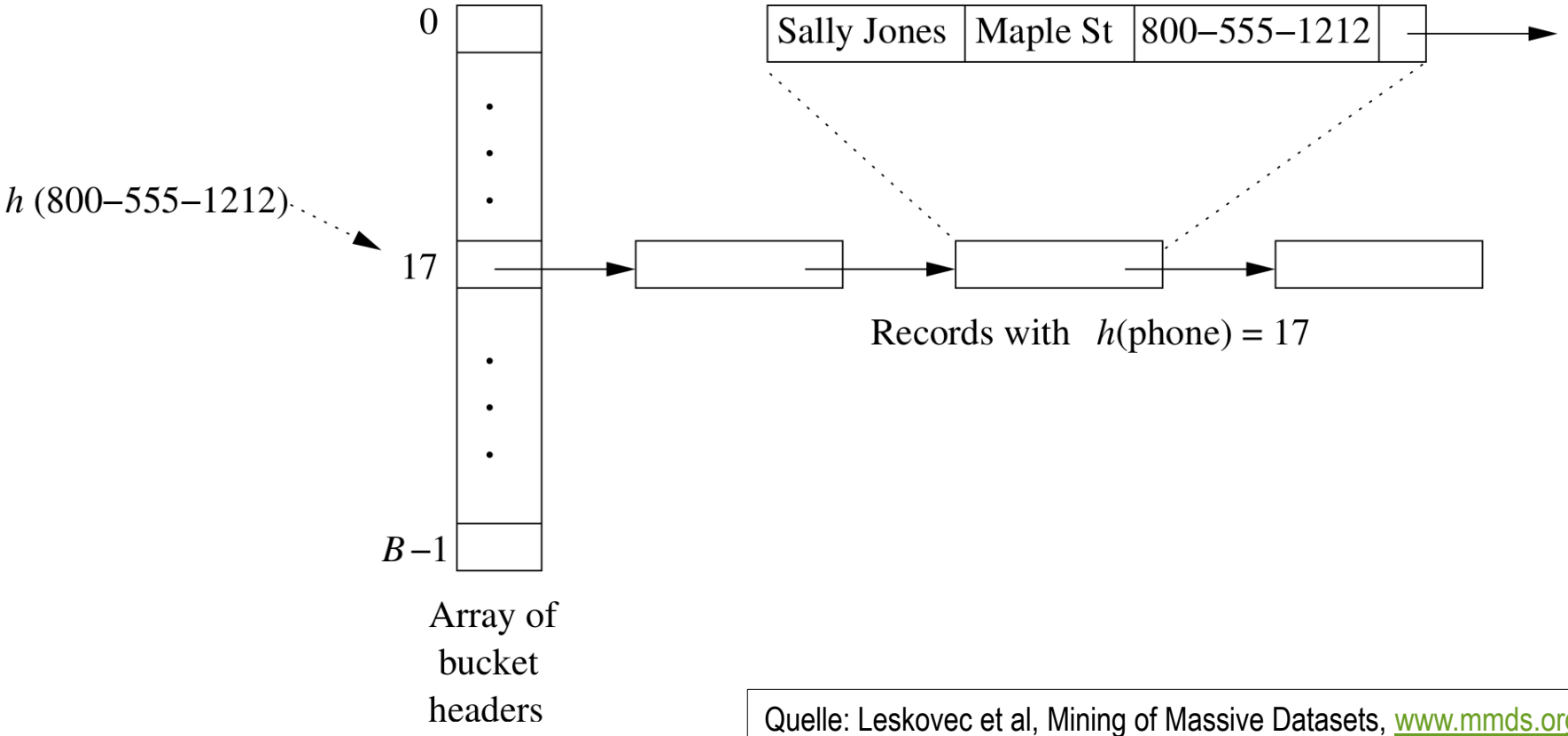
Wiederholung: Hashfunktion



- „Bucket“-Anzahl $B \in \mathbb{N}$
- Hashfunktion h: Objekt $\rightarrow \{0, \ldots, B-1\}$
- *Randomisierend*:
  - Gleichmäßige Aufteilung der Objekte über die Buckets
  - B ist kleiner als die Menge aller Objekte
- Beispiele
  - N modulo p, wobei p eine Primzahl und N eine natürliche Zahl
    - Zeichenketten: Summe über deren ASCII/Unicode-Repräsentation
    - Tupel: Summe über Hash-Werte der Elemente
  - MurmurHash: https://github.com/aappleby/smhasher/wiki
  - Liste mit Hash-Funktionen: https://en.wikipedia.org/wiki/List_of_hash_functions

Wiederholung: Hashindex

- Effiziente Suche eines Eintrags über Attribut
- Beispiel: Hashtabelle

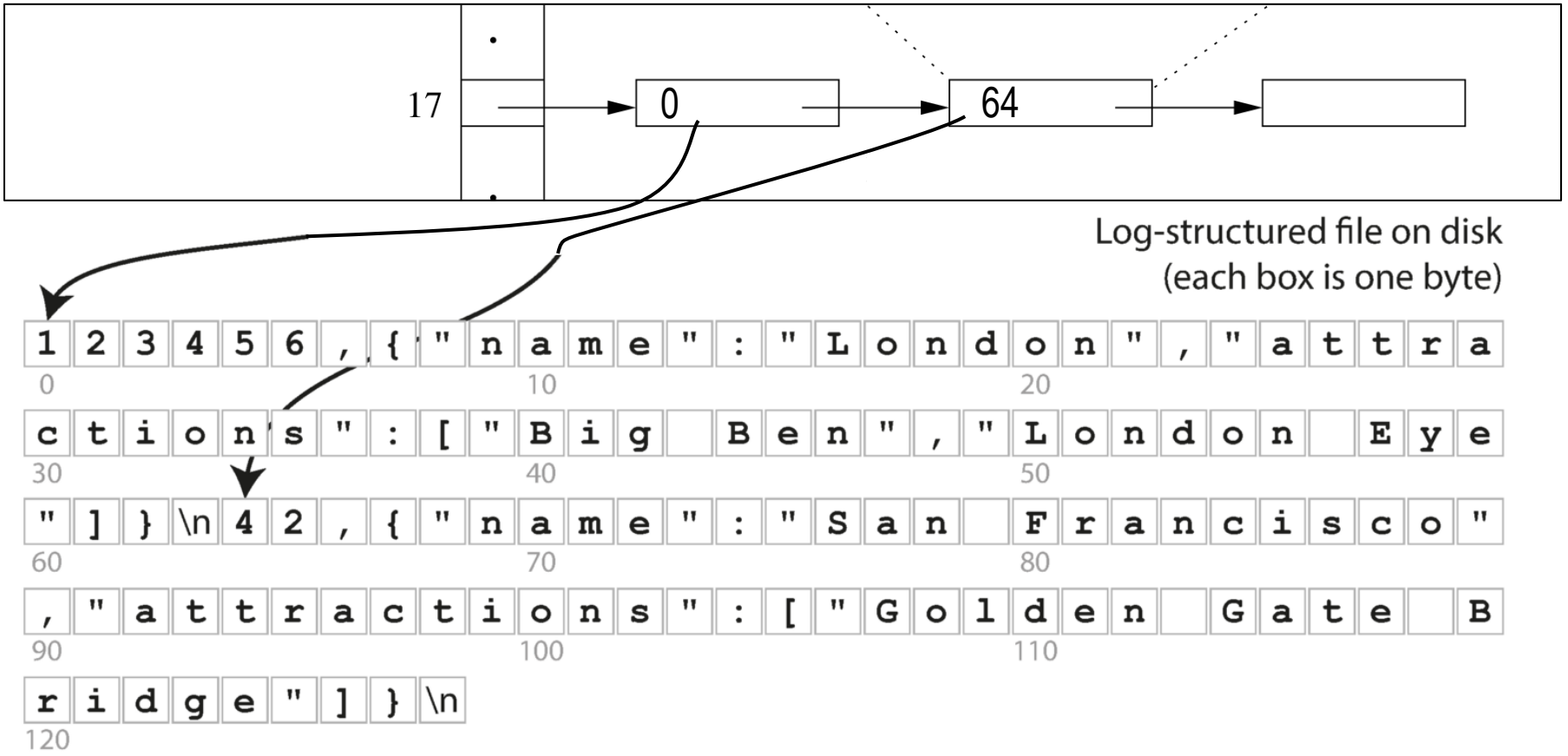


Quelle: Leskovec et al, Mining of Massive Datasets, www.mmds.org

- Je größer B, desto effizienter die Suche

Wiederholung: Hashindex

- Datenbanken: In-memory Hashtabelle mit Byte-Offsets als Werten
- Beispiel:



Quelle: https://medium.com/@shashankbaravani/database-storage-engines-under-the-hood-705418dc0e35

Stärken relationaler Datenbanken

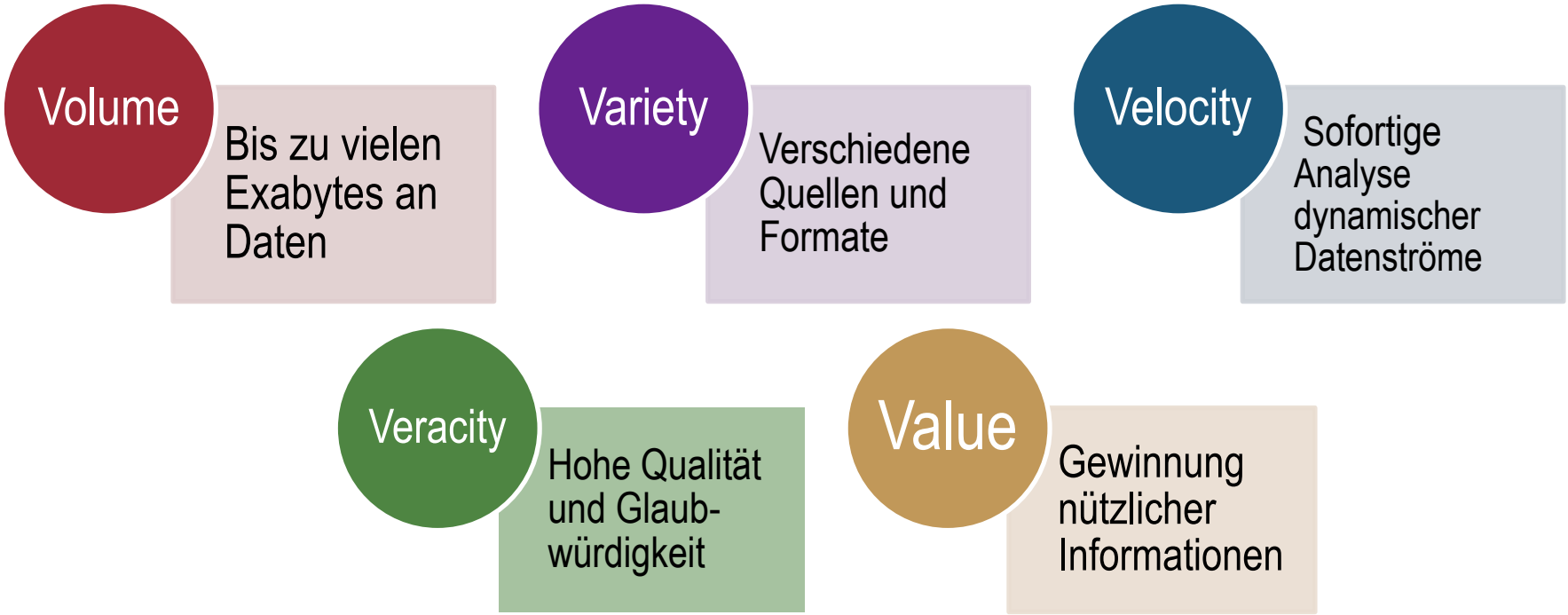
- Reife Technologie
- Mächtige Anfragesprache, insb. Join
- Schema
- Indizes
- Transaktionen mit ACID Garantien
- Mehrere Benutzer
- Authentifizierung
- Aktive Komponenten
- Unterstützung durch verschiedene Programmiersprachen/OS
- Automatische Parallelisierung
- Vertikale Skalierbarkeit

> "if you include the correct modules, tune your engine, and index well, it will perform amazingly well for *multiple terabytes of data* with very small resource consumption"
> (Redmond and Wilson, 2012)

Warum braucht man Alternativen zu relationalen Datenbanken?

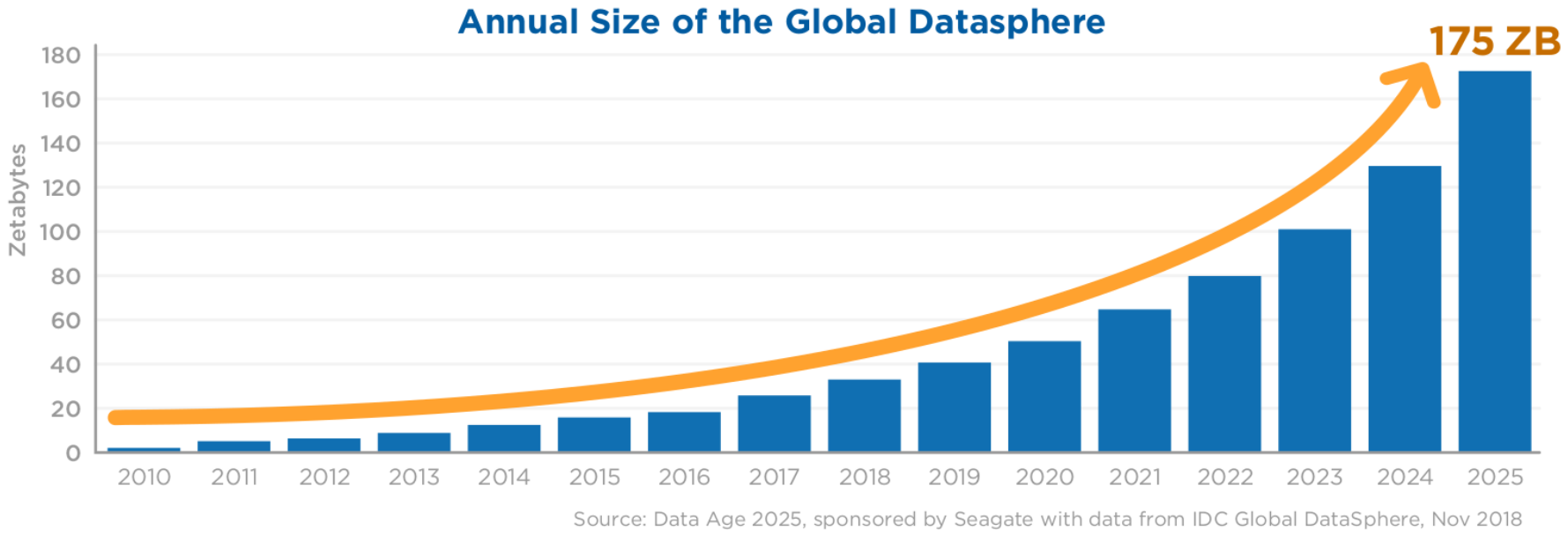
Neue Herausforderung: Big Data

"The term “Big Data” is an imprecise description of a rich and complicated set of characteristics, practices, techniques, ethical issues, and outcomes all associated with data.”
(https://www.aapor.org/Education-Resources/Reports/Big-Data.aspx#3.%20What%20is%20Big%20Data?)



Massives Datenwachstum

- International Data Corporation (IDC): Marktforschungsunternehmen (US)
- „Global Datasphere“: Maß für Umfang der neu hinzukommenden Daten (inkl. Replikationen) pro Jahr weltweit, z.B.
  - Unterhaltung (TV, Radio, Filme, Spiele, …)
  - Andere Bildaufnahmen (Medizin, Multifunktionsgeräte, Überwachung, …)
  - Produktion (PCs, Server, Metadaten, eingebettete Systeme, …)

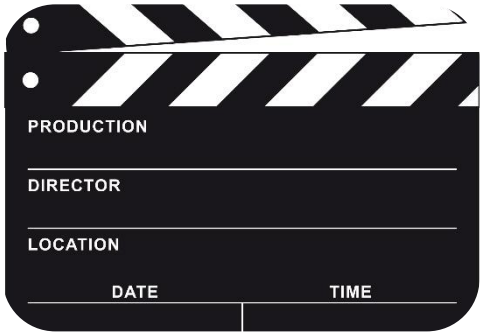


Quelle: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf

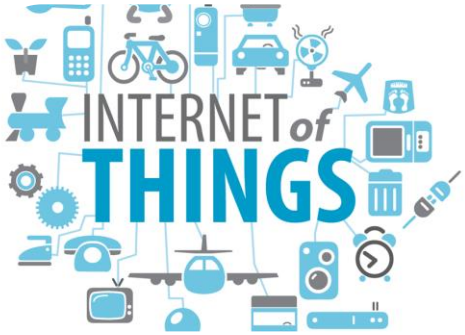
Datenquellen/-formate

```
<!-- created 2010-01-01 -->
 <head>
  <title>sample</title>
 </head>
 <body>
  <p>Voluptatem accusantium
   totam rem aperiam.</p>
 </body>
```

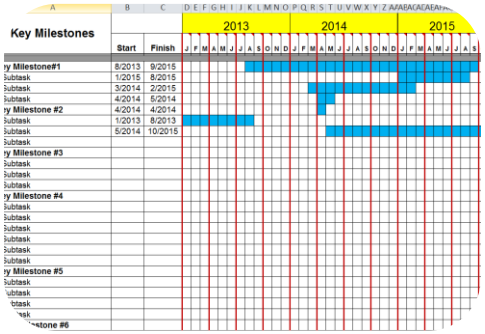
WWW



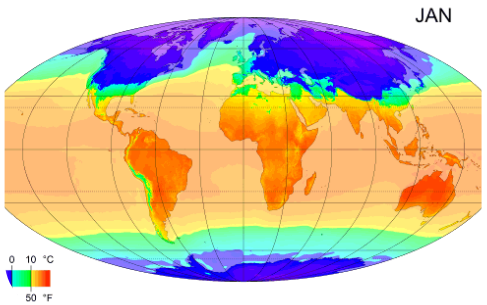
Medien



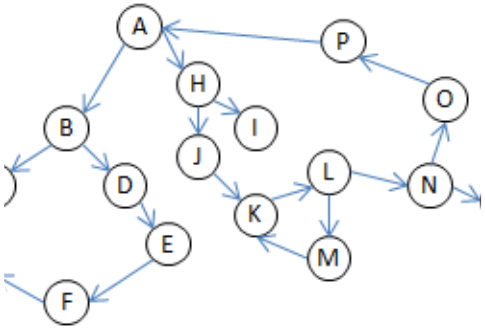
IoT



Unternehmen



Wissenschaft



Graphen

Grenzen der relationalen DB

Leistung

- Sehr große (web-scale) Datenmengen speichern und effizient verarbeiten
- Hohes Volumen konkurrierender Schreib- und Lesezugriffe, z.B. bei Millionen von Nutzern sozialer Medien
- Hohe Verfügbarkeit und geringe Latenzzeiten trotz Knoten- oder Netzwerkausfällen
- Flexible Skalierbarkeit: horizontal anstatt vertikal

Nicht-relationale Daten

- Semi-/unstrukturierte Daten:
  - HTML, XML, JSON
  - Texte, Log-Dateien, Bilder
  - Graphdaten
- Relationales Schema zu starr für einige Anwendungen
- Optionale Attribute/Datentypen
- Änderung der Anwendung führt evtl. zur Schemaänderung
- Inadäquate Repräsentation kann zu ineffizienter Verarbeitung der Anfragen, insb. Join

Inhaltsverzeichnis

- **Organisation**
- **Motivation**
  - Relationale Datenbanken
  - NoSQL-Datenbanken
- **Übersicht zur Vorlesung**
- **Verteilte Datenbanksysteme**
  - Verfügbarkeit und Serialisierbarkeit
  - Eventual Consistency, Sitzungsgarantien und kausale Konsistenz
  - ACID-Garantien
  - CAP-Theorem

NoSQL-Datenbanken

- Entstanden aufgrund unerfüllter Anforderungen von Webanwendungen
  - Google, Amazon, Facebook, Netflix, Uber: Millionen von Nutzern
  - Sehr aufwendige Verwendung von RDBS
- NoSQL-Datenbanken: Bezeichnung für nicht-relationale Datenbanken
- Abgrenzung teilweise schwierig: NoSQL vs NewSQL vs SQL

Neben den relationalen Datenbanken gibt es andere Datenbanken mit Vorteilen in speziellen Anwendungsbereichen

- Eigenschaften (je nach DB nur teilweise zutreffend)
  - Kein relationales Datenmodell, keine Normalisierung
  - "**N**ot **o**nly SQL" : Zugriff über REST API oder einfacher/SQL-ähnlicher Sprache
  - Schemafrei oder schwache Schemarestriktionen
  - Verteiltes, auf horizontale Skalierbarkeit ausgelegtes, System
  - Schwächere semantische Garantien (anstatt ACID)

DB Ranking: NoSQL-DB

Quelle: http://db-engines.com/en/ranking/

345 Systeme im Ranking, März 2019

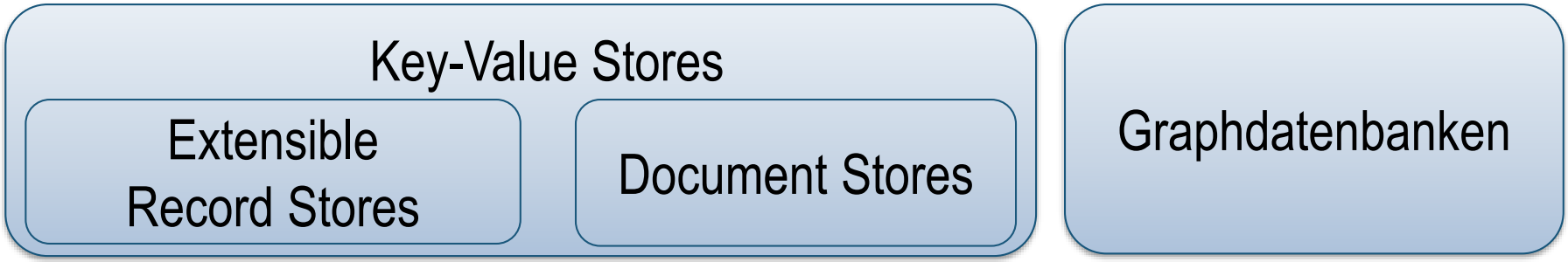
Rang Mär 2019	Rang Feb 2019	Rang Mär 2018	DBMS	Datenbankmodell	Punkte Mär 2019	Punkte Feb 2019	Punkte Mär 2018
1.	1.	1.	Oracle	Relational, Multi-Model	1279,14	+15,12	-10,47
2.	2.	2.	MySQL	Relational, Multi-Model	1198,25	+30,96	-30,62
3.	3.	3.	Microsoft SQL Server	Relational, Multi-Model	1047,85	+7,79	-56,94
4.	4.	4.	PostgreSQL	Relational, Multi-Model	469,81	-3,75	+70,46
5.	5.	5.	MongoDB	Document	401,34	+6,24	+60,82
6.	6.	6.	IBM Db2	Relational, Multi-Model	177,20	-2,23	-9,47
7.	↑ 9.	7.	Microsoft Access	Relational	146,20	+2,18	+14,26
8.	↓ 7.	8.	Redis	Key-value, Multi-Model	146,12	-3,32	+14,90
9.	↓ 8.	9.	Elasticsearch	Suchmaschine, Multi-Model	142,79	-2,46	+14,25
10.	10.	↑ 11.	SQLite	Relational	124,87	-1,29	+10,06
11.	11.	↓ 10.	Cassandra	Wide column	122,80	-0,58	-0,69
12.	12.	↑ 15.	MariaDB	Relational, Multi-Model	84,31	+0,89	+21,21
13.	13.	13.	Splunk	Suchmaschine	83,10	+0,29	+17,44
14.	14.	↓ 12.	Teradata	Relational	75,22	-0,75	+2,76
15.	15.	↑ 18.	Hive	Relational	73,00	+0,71	+16,00
16.	16.	↓ 14.	Solr	Suchmaschine	60,01	-0,95	-4,80
17.	17.	17.	HBase	Wide column	58,80	-1,48	-2,14
18.	18.	↑ 19.	FileMaker	Relational	58,13	+0,34	+3,00
19.	↑ 20.	↓ 16.	SAP Adaptive Server	Relational	56,03	+0,29	-6,58
20.	↓ 19.	20.	SAP HANA	Relational, Multi-Model	55,51	-1,03	+6,99
21.	21.	21.	Amazon DynamoDB	Multi-Model	54,49	-0,45	+12,04
22.	22.	22.	Neo4j	Graph	48,58	+0,72	+7,68
23.	23.	23.	Couchbase	Document	33,80	-1,78	+0,90
24.	24.	24.	Memcached	Key-value	28,73	-0,72	-2,62
25.	25.	↑ 26.	Microsoft Azure SQL Database	Relational, Multi-Model	27,93	+0,81	+3,31
26.	26.	↓ 25.	Informix	Relational, Multi-Model	26,90	+0,54	-0,61
27.	27.	↑ 31.	Microsoft Azure Cosmos DB	Multi-Model	24,83	-0,03	+8,07
28.	28.	↓ 27.	Vertica	Relational, Multi-Model	22,07	-0,74	+1,77
29.	29.	↑ 32.	Amazon Redshift	Relational	20,89	-0,10	+6,70
30.	↑ 32.	↓ 29.	Firebird	Relational	20,17	+0,82	+2,14

Inhaltsverzeichnis

- **Organisation**
- **Motivation**
  - Relationale Datenbanken
  - NoSQL-Datenbanken
- **Übersicht zur Vorlesung**
- **Verteilte Datenbanksysteme**
  - Verfügbarkeit und Serialisierbarkeit
  - Eventual Consistency, Sitzungsgarantien und kausale Konsistenz
  - ACID-Garantien
  - CAP-Theorem

Fazit & Übersicht

- NoSQL: Aufgeben von Eigenschaften relationaler DB
  - Relationales Schema
  - Standardisierte Anfragesprache
  - Logischer Einbenutzerbetrieb
  - Beständigkeit
- Im Austausch für Leistung (Verfügbarkeit, Skalierbarkeit) und Flexibilität bzgl. Datenstrukturen
- Inhalt der Vorlesung
  - Datenbankarchitekturen für die effiziente Verwaltung von teilweise sehr großen Mengen nicht-relationaler Daten
  - Am Beispiel ***relativ populärer open-source*** NoSQL-Datenbanken



Inhalt der Vorlesung

>225 NoSQL Datenbanken:
http://nosql-database.org/

Graphdatenbanken

- Repräsentation der Daten als Knoten und Kanten mit Properties

Key-Value Stores

- Kollektion von Key/Value-Paaren

Document Stores

- Speicherung semistrukturierter Daten als Dokumente (JSON)

Extensible Record Stores

- Tabellenartige Strukturen mit flexibler Erweiterung der Attribute

Andere nicht-relationale DBS

XML-Datenbanken

Objektdatenbanken

→ Datenbanksysteme 2

RDF Stores

→ Semantic Web

Zeitreihendatenbanken

- InfluxDB, TimescaleDB, M3, …

NewSQL-Datenbanken

- Spanner, YugaByte, FaunaDB, …

Suchmaschinen

- Elasticsearch, Solr, Lucene, …

Analytische Datenbanken

- Impala, Druid, AresDB, …

Offene Themen:
Studierende, welche eines dieser Themen im Rahmen einer Abschlussarbeit bearbeiten wollen, können sich bei mir melden!

Abgrenzung zu anderen Vorlesungen

Mehrrechnerdatenbanksysteme (evtl. nächstes Semester)

- Einsatz mehrerer Rechner zur koordinierten Verarbeitung von Daten
- Klassifikation: parallel, verteilt, föderiert
- Viele NoSQL-DB unterstützen eine **verteilte** Architektur

Implementierung von Datenbanksystemen (letztes Semester)

- Realisierungskonzepte von Datenbanksystemen
- Speicher-/Indexverwaltung, Synchronisation, Recovery, …
- Wichtige Themen auch für NoSQL-DB

Cloud Data Management (dieses Semester)

- Verteilte Dateisysteme, MapReduce, Large-scale Datenanalyse, Data Streaming
- NoSQL-Datenbanken sind weitere wichtige Bestandteile Big-Data-Ökosystems