Internet of Things

Lecture 10

Data 3

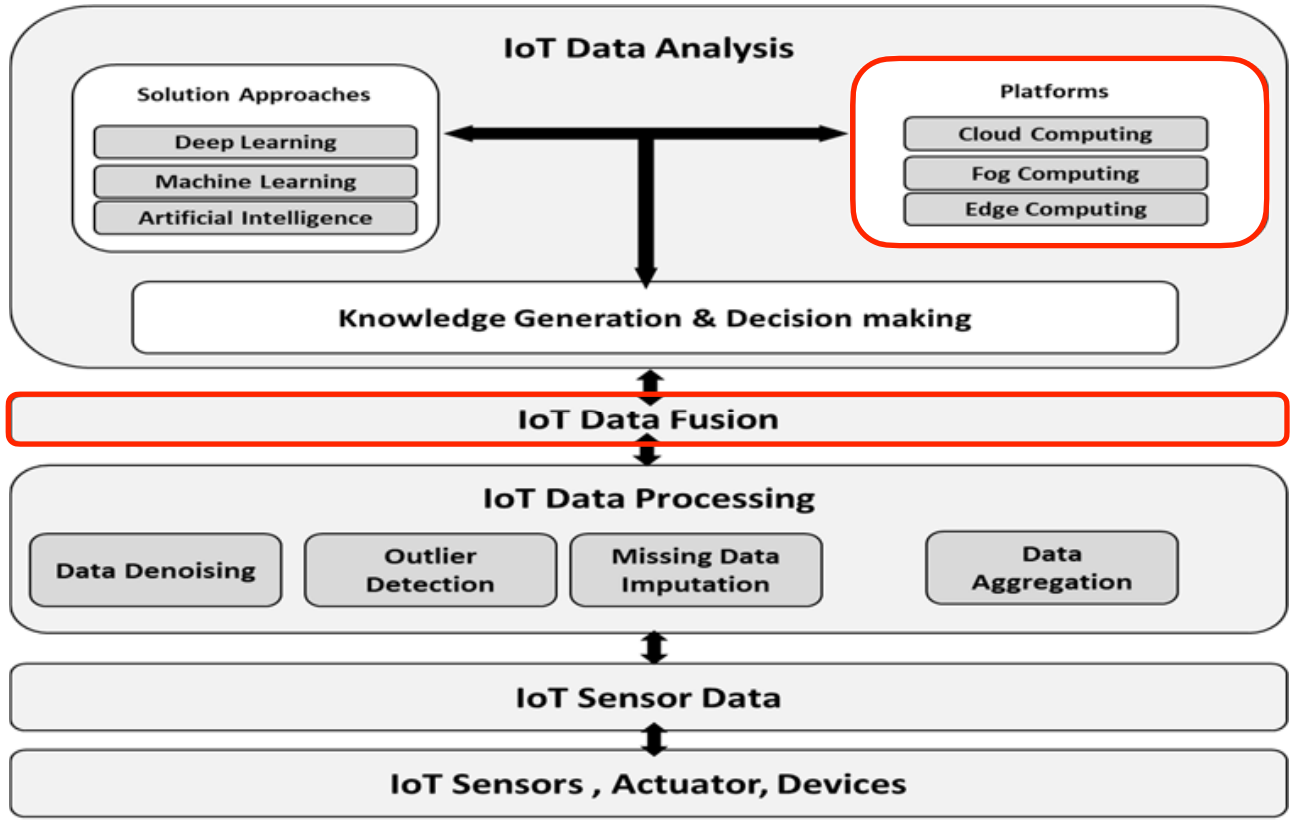
Data storage and processing

Michael Engel

Lecture slides licensed under
CC-by-SA 4.0 (unless noted otherwise)

Storing data in the IoT

- How can data be stored in IoT applications?



Databases overview

- Relational databases
  - SQL query language

- NoSQL databases – non-traditional database approaches
  - Key-value stores
  - Document stores
  - Wide-column stores
  - Time series databases
  - Graph databases

- Stream processing

Relational databases

- Based on the relational data model [1]
  - data is organized into one or more tables ("relations") of columns and rows, with a unique key identifying each row
    - rows are also called records or tuples
    - columns are also called attributes
- Each table/relation represents one *entity type*
  - e.g. a customer or a product
- Standard Query Language: SQL
- **Challenge:** The amount of useful data in some application areas has become so vast that it cannot be stored or processed by traditional database solutions
  - Examples:
    - user-generated content in social networks
    - data retrieved from large sensor networks
- General trend: **Big Data**

Challenges for IoT data storage

Data in IoT applications is characterized by

- Heterogeneity of data
- Variable data formats
- Real-time requirements
- Data streams and data fusion

- Traditional relational databases don’t cover IoT use cases well
  - Difficult to support heterogeneous data structures
  - Large data (e.g. blobs like pictures/video/audio) not well supported
  - Real-time applications require guaranteed response times
  - Streaming data is possibly only relevant for a short amount of time

NoSQL databases [2]

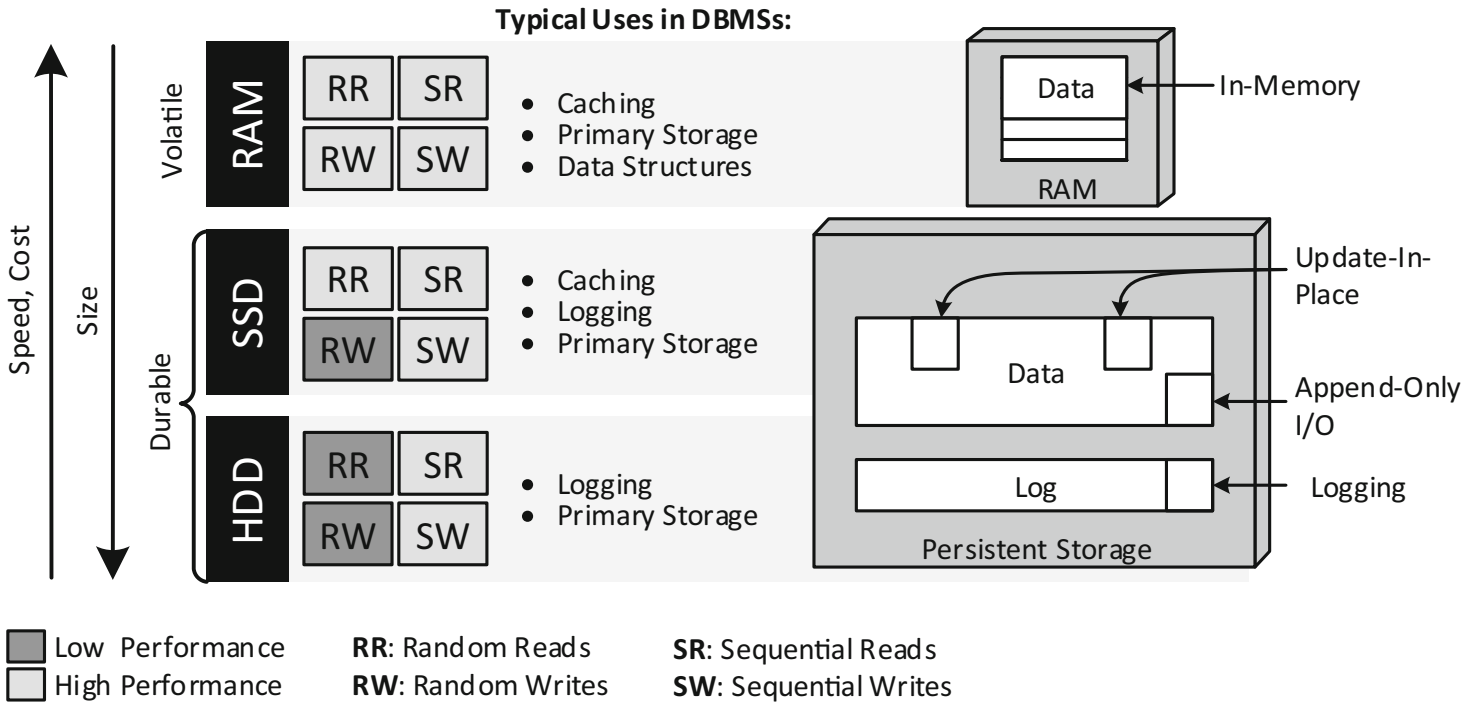
- Class of non-relational databases covering a wide range of use cases
  - specifically, databases able to cope with Big Data

- Many NoSQL databases offer horizontal scalability and higher availability than relational databases
  - by sacrificing querying capabilities and consistency guarantees

- The trade-offs are pivotal for service-oriented computing and as-a-service models
  - any stateful service can only be as scalable and fault-tolerant as its underlying data store.

NoSQL storage pyramid

- Different levels of storage in the **storage hierarchy** have different use cases:
  - from fast, expensive, small (RAM) to large, inexpensive, slow (hard disk, HDD)



NoSQL databases – key-value stores

- Consist of a set of **key-value pairs** with **unique keys**
- Simple structure: only supports get and put operations

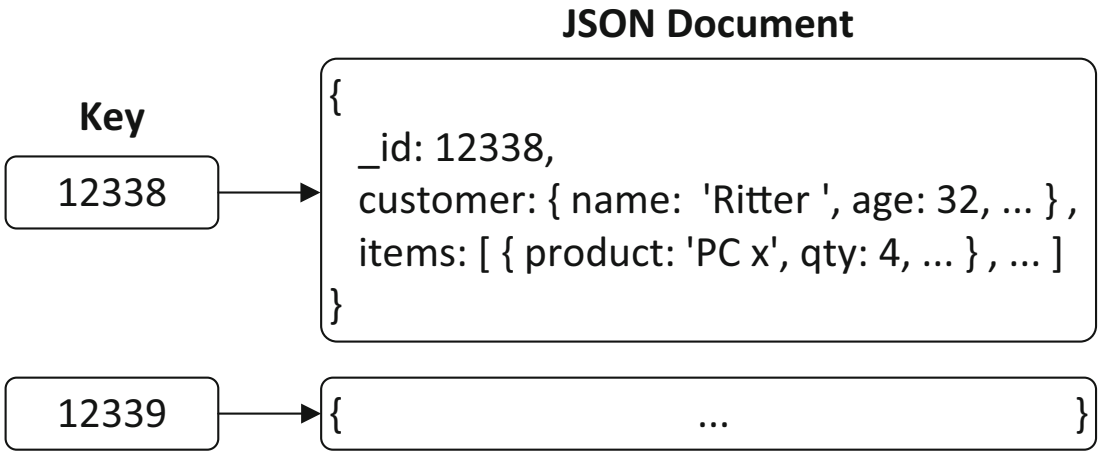
- Pure key-value stores do not support operations beyond simple **CRUD** (Create, Read, Update, Delete)

- Key-value stores are **schemaless**:
  - any assumptions about the structure of stored data are implicitly encoded in the application logic

Key	Value (Opaque)
User:2:friends	{23, 76, 233, 11}
User:2:settings	Theme: dark, cookies: false
User:3:friends	[234, 3466, 86, 55]

NoSQL databases – document stores

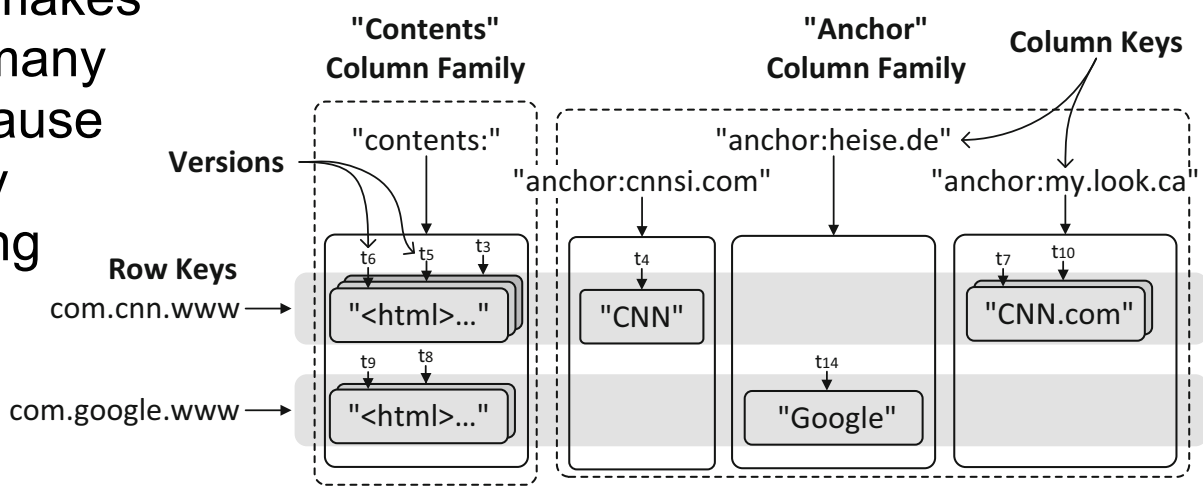
- A document store is a key-value store that restricts values to semi-structured formats such as JSON documents



- This restriction in comparison to key-value stores brings great flexibility in accessing the data
  - It is not only possible to fetch an entire document by its ID, but also to retrieve only parts of a document
  - E.g. the age of a customer, and to execute queries like aggregation, query-by-example or even full-text search

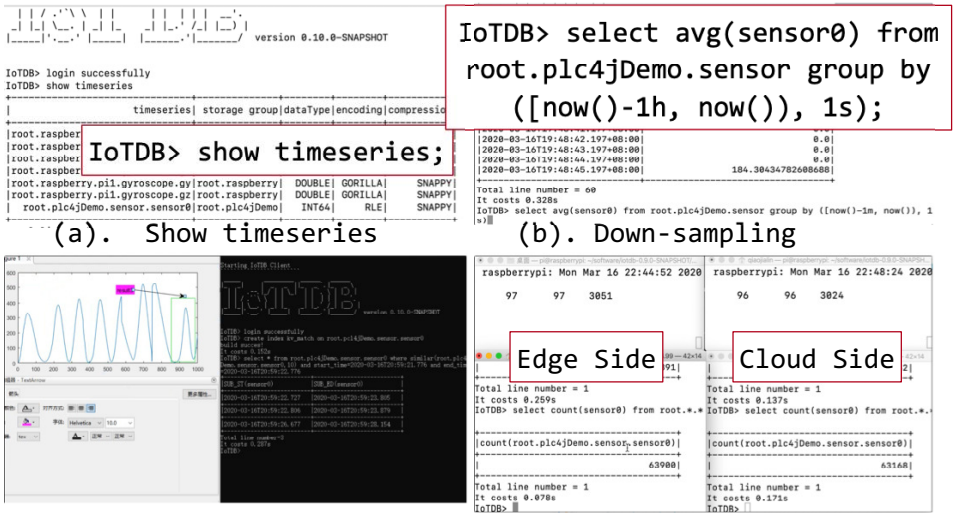
NoSQL databases – wide-column stores

- Wide-column stores are named according to the underlying data model: a relational table with many sparse columns
- Technically a wide-column store is closer to a distributed multi-level sorted map
  - first-level keys identify rows which themselves consist of key-value pairs
    - first-level keys are called row keys
    - second-level keys are called column keys
- This storage scheme makes tables with arbitrarily many columns feasible, because there is no column key without a corresponding value
- Null values can be stored without any space overhead



NoSQL databases – time series

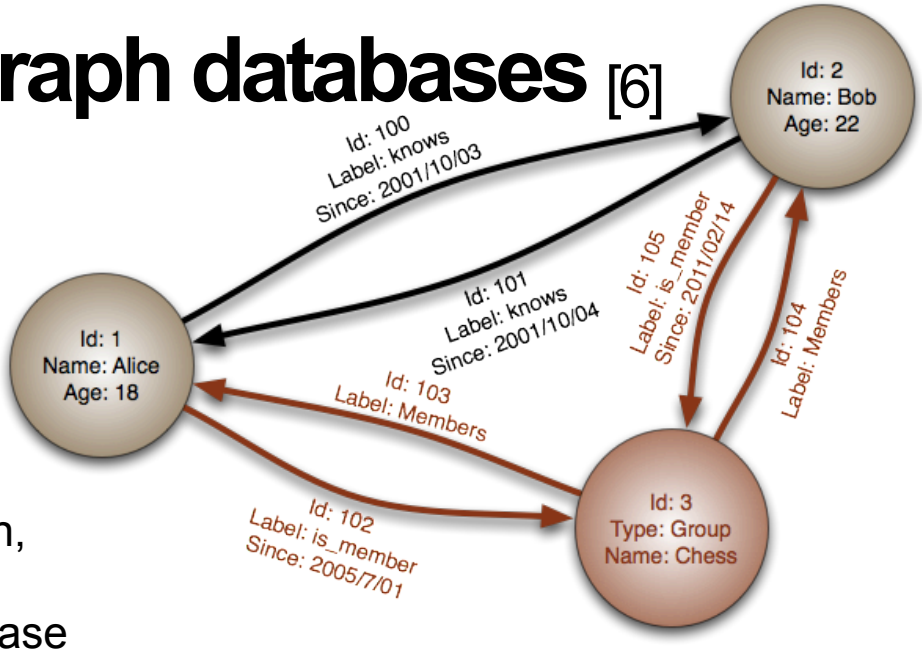
- A time series database (TSDB) is a software system that is optimized for storing and serving time series through associated pairs of time(s) and value(s) [3,4]
  - Time series datasets are relatively large and uniform compared to other datasets–usually being composed of a timestamp and associated data
  - Time series datasets can also have fewer relationships between data entries in different tables and don't require indefinite storage of entries
  - In some fields, time series may be called profiles, curves, traces or trends
- IoT example: Queries in Apache IoTDB [5]



(a). Show timeseries (b). Down-sampling

(c). KV-match index (d). File Sync

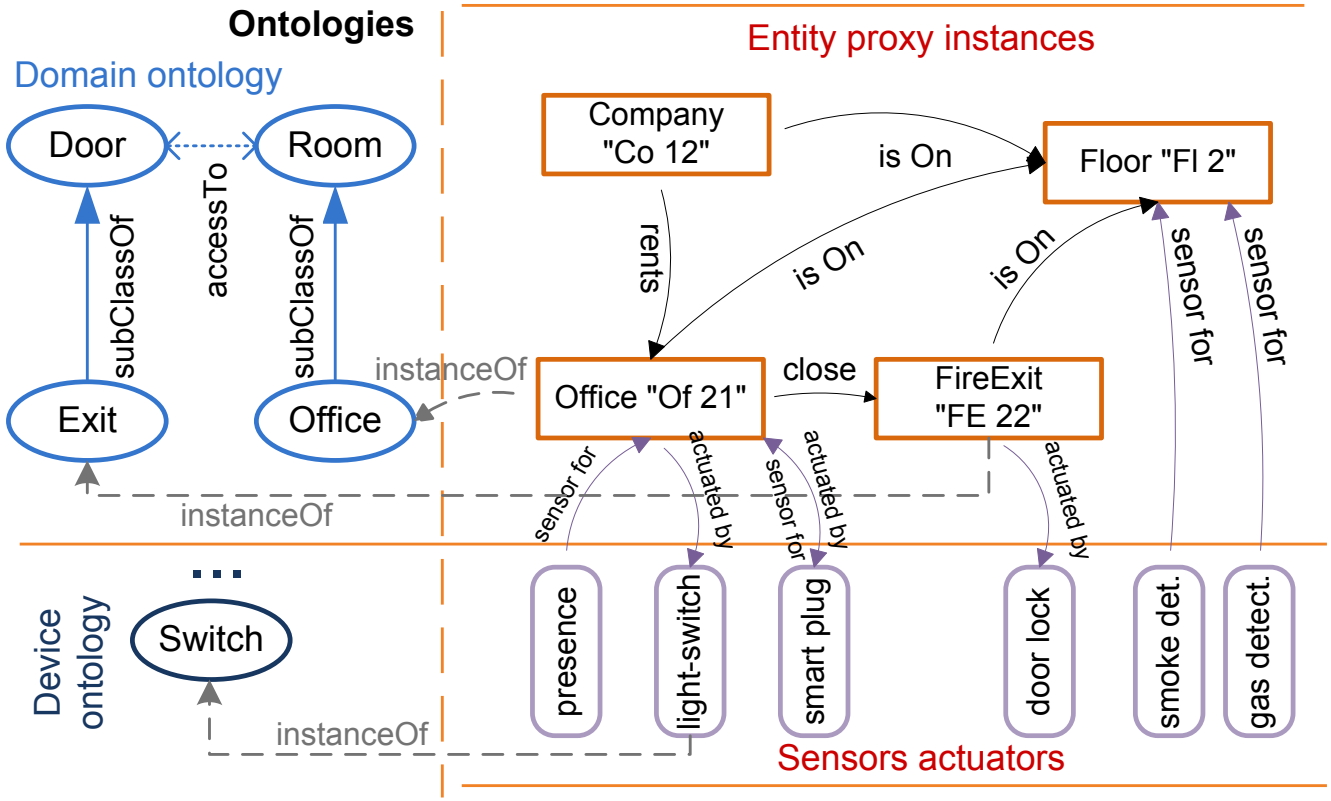
NoSQL databases – graph databases [6]



- Databases based on graph theory
- It consists of a set of **objects**, which can be a **node** or an **edge**
- **Nodes** represent entities or instances such as people, businesses, accounts, or any other item to be tracked
  - Rough equivalent of a record, relation, or row in a relational database, or a document in a document-store database
- **Edges** connect nodes to other nodes; representing the relationship between them
  - Edges can either be directed or undirected
  - In an undirected graph, an edge connecting two nodes has a single meaning
  - In a directed graph, the edges connecting two different nodes have different meanings, depending on their direction
  - Edges are the key concept in graph databases, representing an abstraction that is not directly implemented in a relational model or a document-store model.
- **Properties** are information associated to nodes
  - For example, if **Wikipedia** were a node, it might be tied to properties such as website, reference material, or words that starts with the letter 'w'

IoT Graph database example

- Capturing the structure of IoT systems in a smart home environment (from [7]), e.g. in the Neo4j graph database [8]

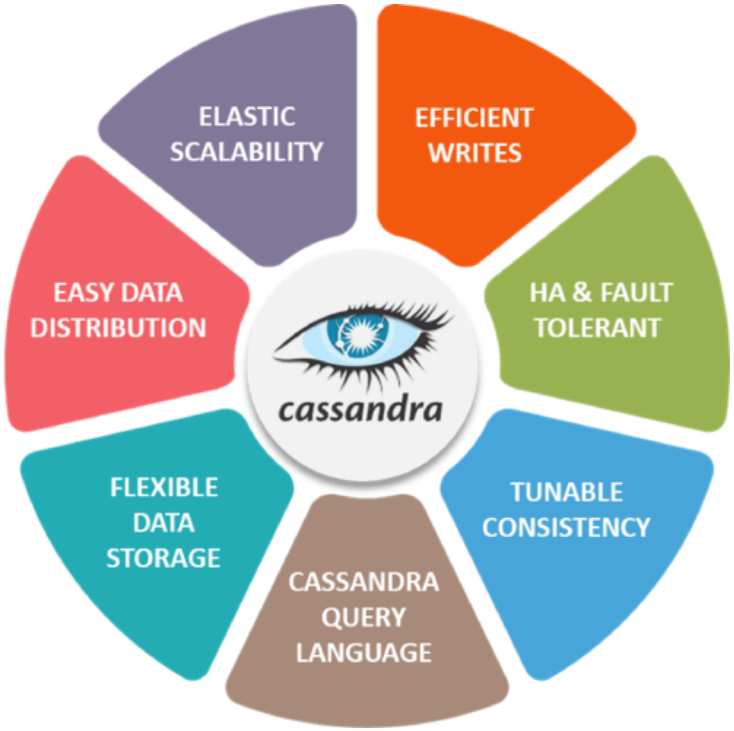


Distributed databases – Cassandra [9]

- Distributed, wide-column store, NoSQL database

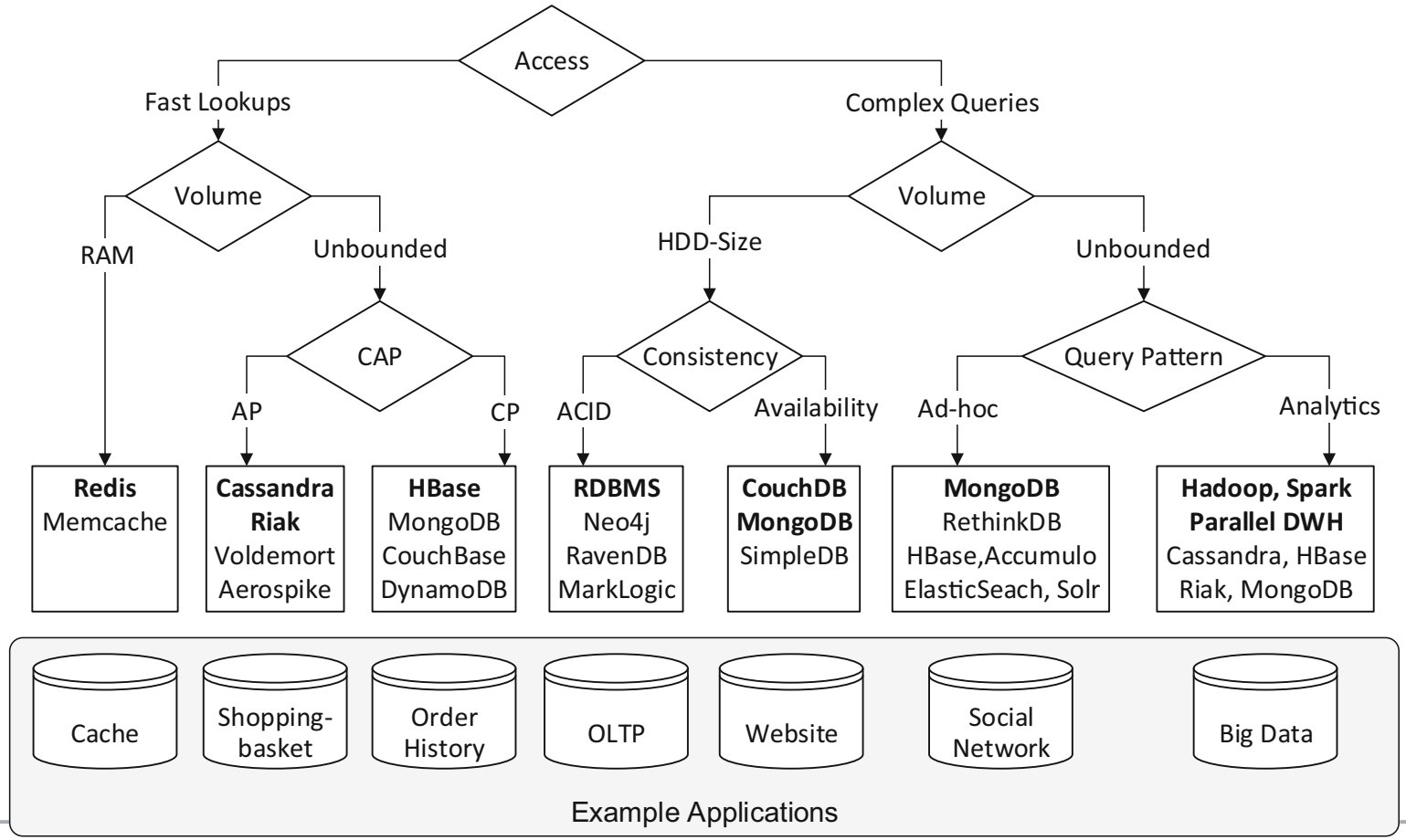
- Designed to handle large amounts of data across many commodity servers
  - high availability with no single point of failure

- Support for clusters spanning multiple data centers
  - asynchronous masterless replication allowing low latency operations for all clients

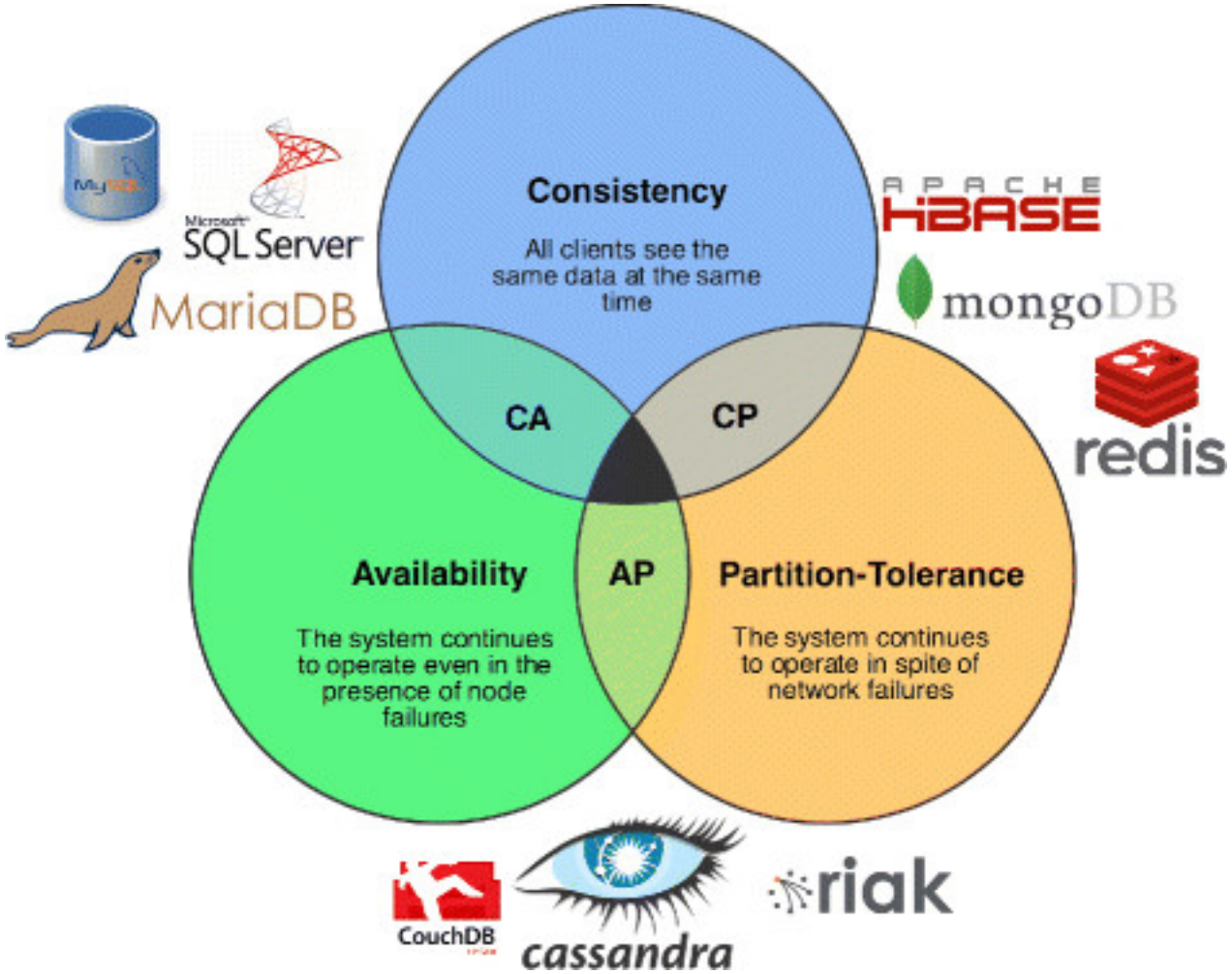


NoSQL decision tree

- Different trade-offs to be considered [1]
  - leaf nodes: applications from simple caching (left) to big data analytics (right)
  - differentiation according to access patterns, query complexity and data volume



Properties of different NoSQL databases



from [10]

Conclusion

- Traditional relational databases not well suited for variable, streaming, heterogeneous big data use cases
  - data typical in the IoT

- NoSQL databases are a class of databases
  - different types for specific use cases:
    key-value stores, document stores, wide-column stores, time series DB, graph databases

- Distributed data bases increase properties such as availability, capacity, latency/throughput and fault tolerance

References

[1] E. F. Codd, *A Relational Model of Data for Large Shared Data Banks*, Communications of the ACM. 13 (6): 377–387, 1970

[2] F. Gessert, W. Wingerath, S. Friedrich, et al. *NoSQL database systems: a survey and decision guidance,* Comput Sci Res Dev 32, 353–365 (2017). https://doi.org/10.1007/s00450-016-0334-3

[3] W. Wingerath, N. Ritter, F. Gessert, *Real-Time & Stream Data Management - Push-Based Data in Research & Practice*, Springer Briefs in Computer Science, 1/77, 2019, ISBN 978-3-030-10554-9

[4] A. Mueen, E. Keogh, Q. Zhu, Qiang. S. Cash, B. Westover, *Exact Discovery of Time Series Motifs*, University of California, Riverside. 2009: 473–484

[5] C. Wang et al. *Apache IoTDB: time-series database for internet of things*, Proceedings of the VLDB Endowment 13.12 (2020): 2901-2904

[6] R. Angles, C. Gutierrez, *Survey of graph database models*, ACM Computing Surveys (CSUR) 40.1 (2008): 1-39.

[7] Gilles Privat and Dana Popovici, *Capturing the Structure of IoT Systems with Graph Databases*, Orange Labs 2015 https://www.iaria.org/conferences2015/filesDBKDA15/graphsm_privat_popovici.pdf

[8] J. J. Miller, *Graph database applications and concepts with Neo4j*, Proceedings of the southern association for information systems conference, Atlanta, GA, USA. Vol. 2324. No. 36. 2013

[9] E. Hewitt, *Cassandra: the definitive guide*, O'Reilly, 2020, ISBN 978-1098115166

[10] https://thecustomizewindows.com/2019/08/apache-cassandra-vs-apache-hbase/