NoSQL Data Models

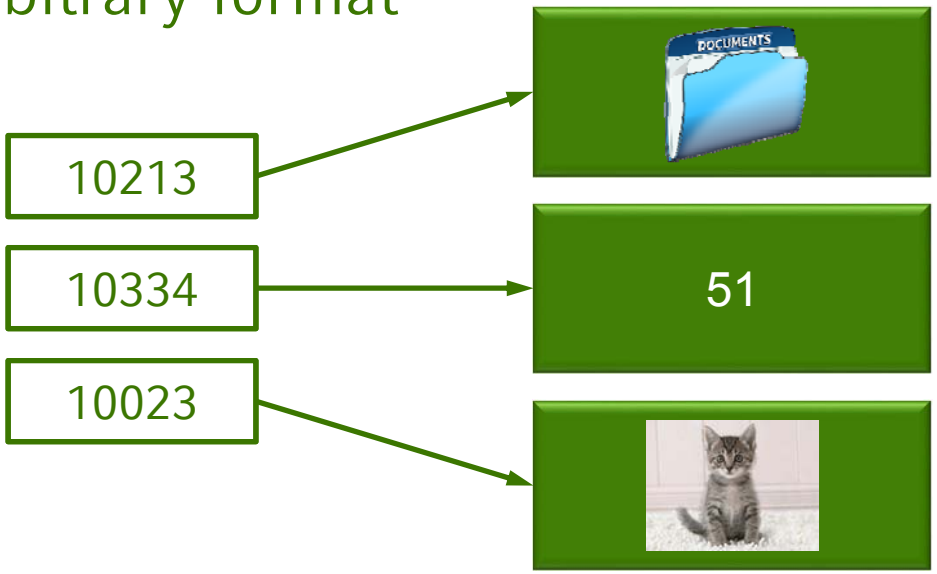
The 4 Main NoSQL Data Models:

- Key/Value Stores
- Document Stores
- Wide Column Stores
- Graph Databases

NoSQL Data Models

Key/Value Stores:

- Most simple form of database systems
- Store key/value pairs and retrieve values by keys
- Values can be of arbitrary format



NoSQL Data Models

Key/Value Stores:

- Consistency models range from *Eventual consistency* to *serializibility*
- Some systems support ordering of keys, which enables efficient querying, like range queries
- Some systems support in-memory data maintenance, some use disks

→ There are very heterogeneous systems

NoSQL Data Models

Key/Value Stores - Redis:

- In-memory data structure store with built-in replication, transactions and different levels of on-disk persistence
- Support of complex types like lists, sets, hashes, …
- Support of many *atomic* operations

```
>> SET val 1
>> GET val => 1
>> INCR val => 2
>> LPUSH my_list a (=> ‘a‘)
>> LPUSH my_list b (=> ‘b‘,‘a‘)
>> RPUSH my_list c (=> ‘b‘,‘a‘,‘c‘)
>> LRANGE my_list 0 1 => b,a
```

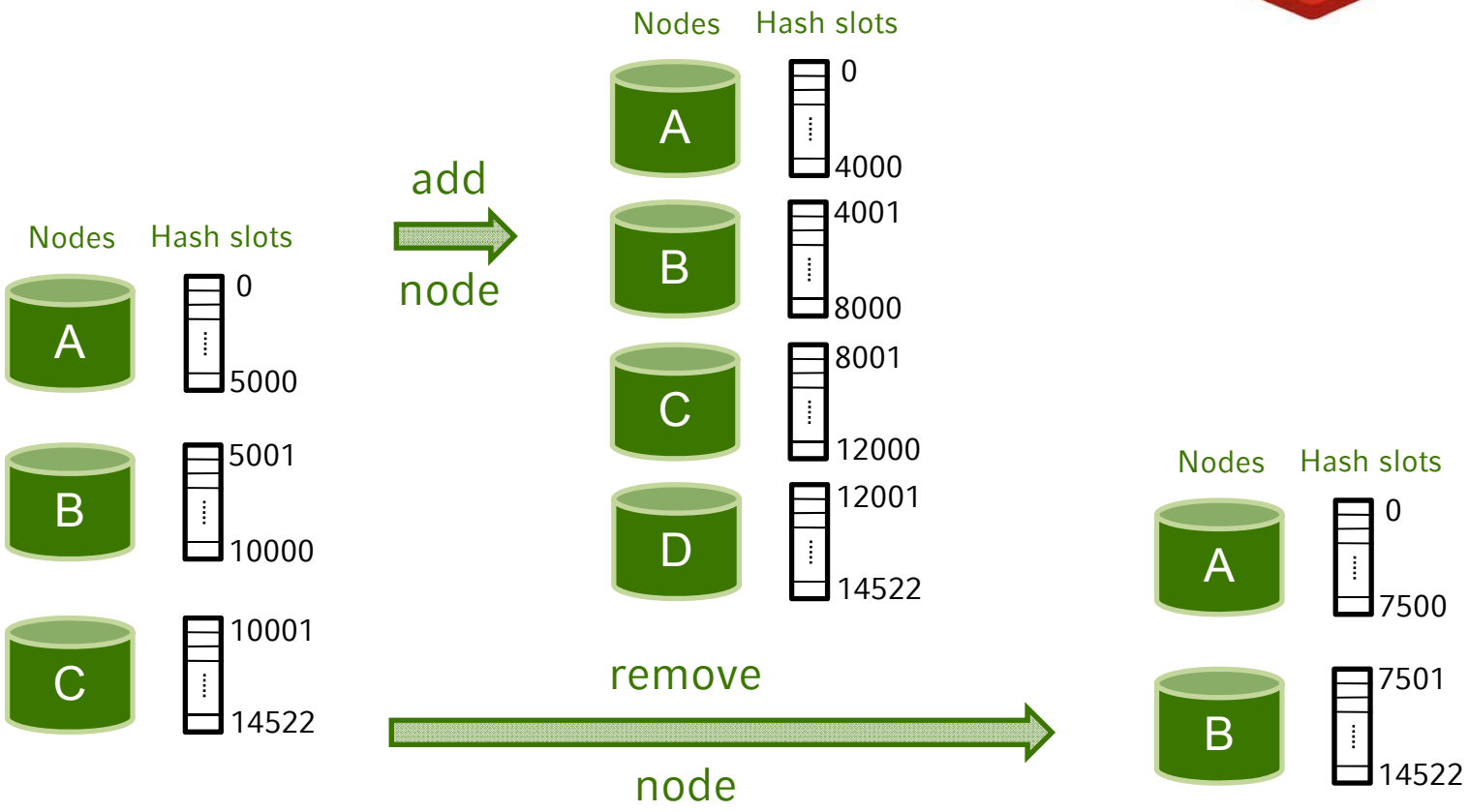
NoSQL Data Models

Key/Value Stores – The Redis cluster model:

- Data is automatically sharded across nodes
- Some degree of availability, achieved by master-slave architecture (but cluster stops in the event of larger failures)
- Easily extendable

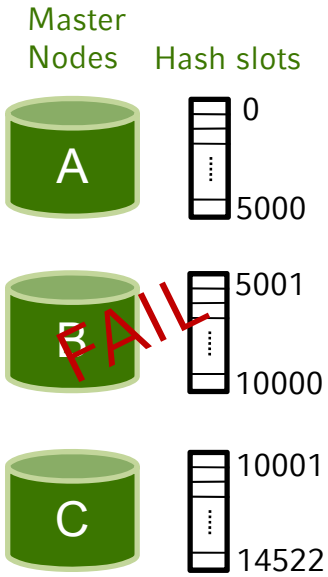
NoSQL Data Models

Key/Value Stores – The Redis cluster model:

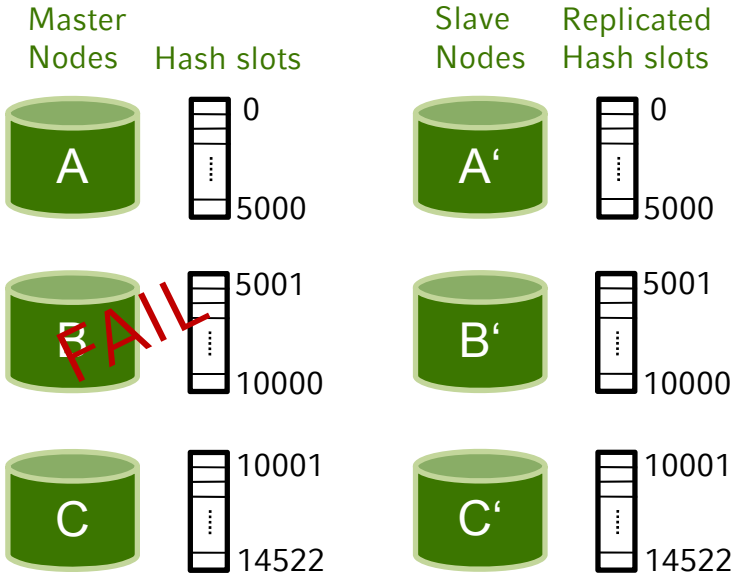


NoSQL Data Models

Key/Value Stores – The Redis cluster model:



Hash slots 5001 – 10000 cannot be used anymore



Slave node B‘ is promoted as the new master and hash slots 5001 – 10000 are still available

Big Picture

CAP Theorem:

C

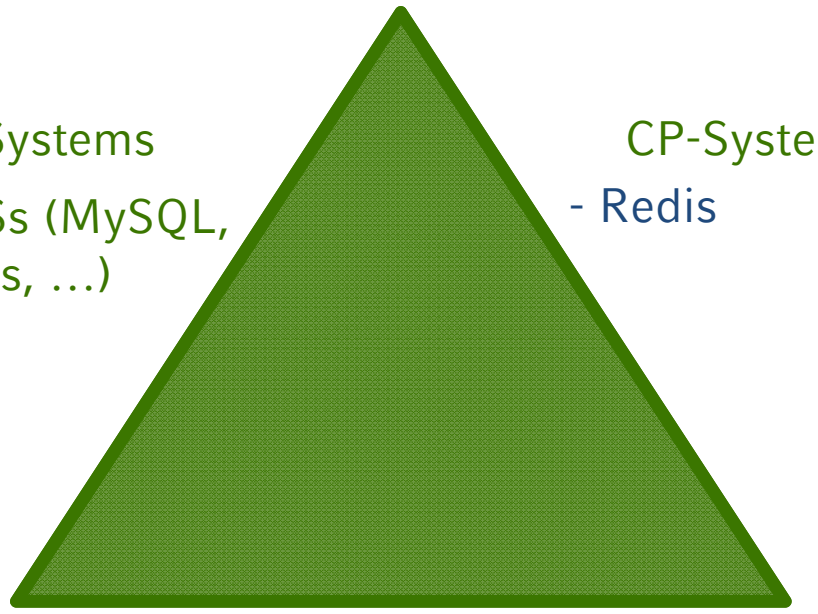
ACID

BASE

A

All clients always have the same view of the data

Key/Value Stores



AC-Systems

- RDBMSs (MySQL, Postgres, ...)

CP-Systems

- Redis

AP-Systems

- Dynamo

Each client can always read and write

The system works well despite physical network partitions

NoSQL Data Models

Document Stores:

- Store documents in form of XML or JSON
- Semi-structured data records that do not have a homogeneous structure
- Columns can have more than one value (arrays)
- Documents include internal structure, or metadata
- Data structure enables efficient use of indexes

NoSQL Data Models

Document Stores:

Given following text:

Max Mustermann
Musterstraße 12
D-12345 Musterstadt

```
<contact>
  <first_name>Max</first_name>
  <last_name>Mustermann</last_name>
  <street>Musterstraße 12</street>
  <city>Musterstadt</city>
  <zip>12345</zip>
  <country>D</country>
</contact>
```

→ Find all <contact>s where <zip> is “12345“

NoSQL Data Models

Document Stores:

- Data stored as documents in binary representation (BSON)
- Similarly structured documents are bundled in collections
- Provides own ad-hoc query language
- Supports ACID transactions on document level

NoSQL Data Models

Document Stores: mongoDB

MongoDB Data Management:

- Automatic data sharding
- Automatic re-balancing

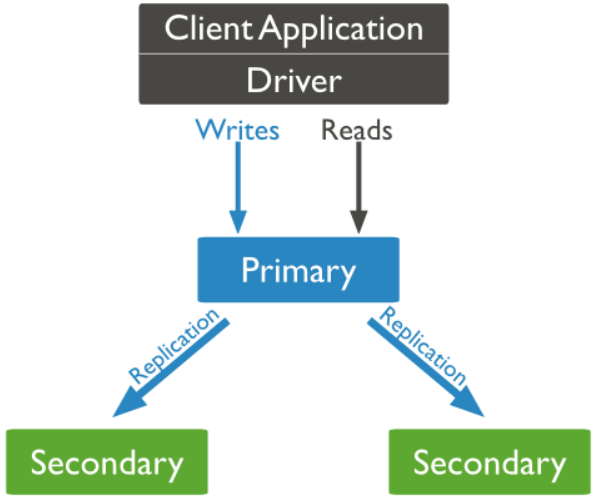
- Multiple sharding policies:
  - Hash-based sharding: Documents are distributed according to an MD5 hash → uniform distribution
  - Range-based sharding: Documents with shard key values close to one another are likely to be co-located on the same shard → works well for range queries
  - Location-based sharding: Documents are partitioned wrt to a user-specified configuration that associates shard key ranges with specific shards and hardware

NoSQL Data Models

Document Stores: mongoDB

MongoDB Consistency & Availabilty:

- Default: Strong consistency (but configurable)
- Increased availability through replication
  - *Replica sets* consist of one *primary* and multiple *secondary members*
  - MongoDB applies writes on the primary and then records the operations on the primary's *oplog*



Big Picture

CAP Theorem:

C

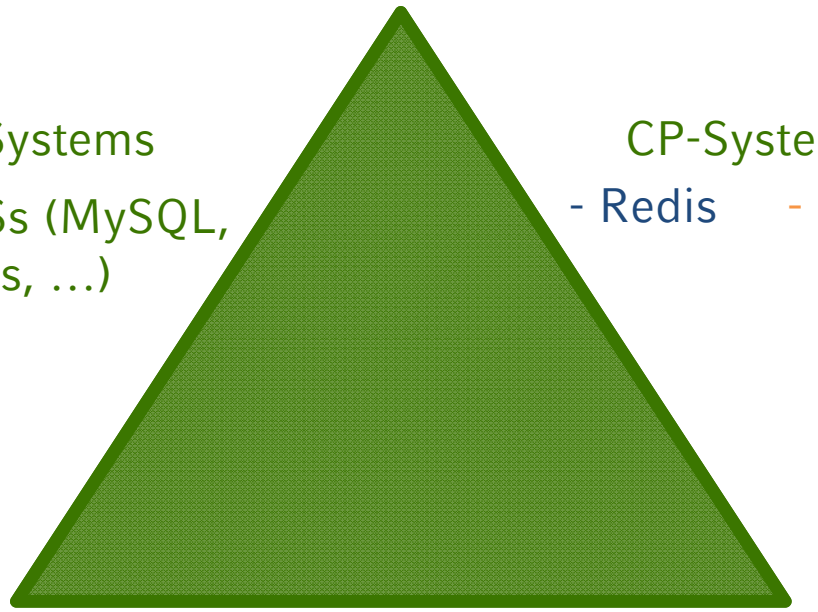
ACID

BASE

A

All clients always have the same view of the data

Key/Value Stores
Document Stores



AC-Systems

- RDBMSs (MySQL, Postgres, ...)

CP-Systems

- Redis
- MongoDB

Each client can always read and write

AP-Systems

- Dynamo
- CouchDB

The system works well despite physical network partitions

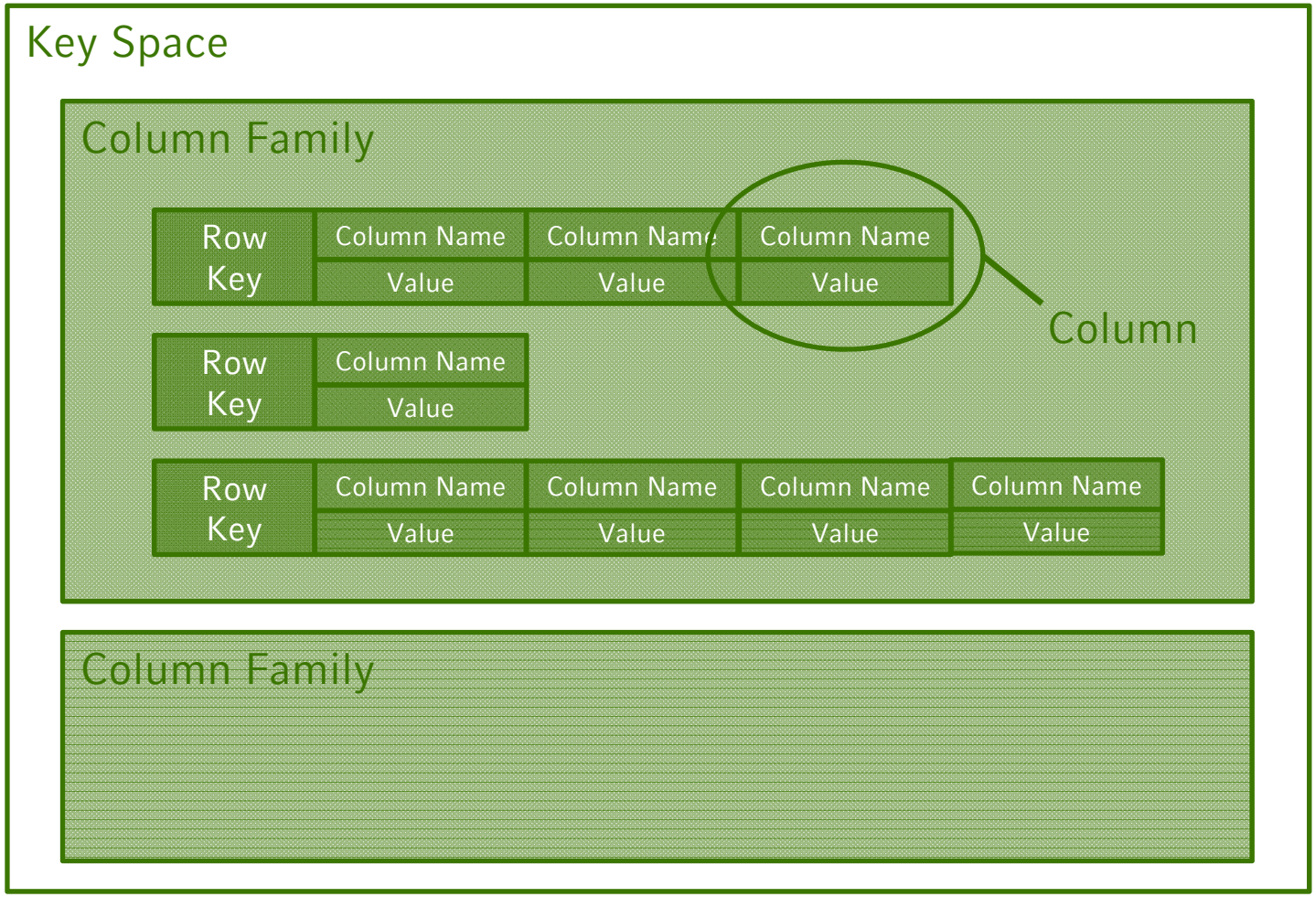
NoSQL Data Models

Wide Column Stores:

- Rows are identified by keys
- Rows can have different numbers of columns (up to millions)
- Order of rows depend on key values (locality is important!)
- Multiple rows can be summarized to *families* (or *tablets*)
- Multiple families can be summarized to a *key space*

NoSQL Data Models

Wide Column Stores:



NoSQL Data Models

Wide Column Stores:

Key Space „Edibles“

Column Family „Fruit“

Apple	color	weight	variety	
	„green“	95	„Granny Smith“	
Cherry	color			
	„red“			
Lemon	color	weight	origin	flavor
	„yellow“	50	„Egypt“	„sour“

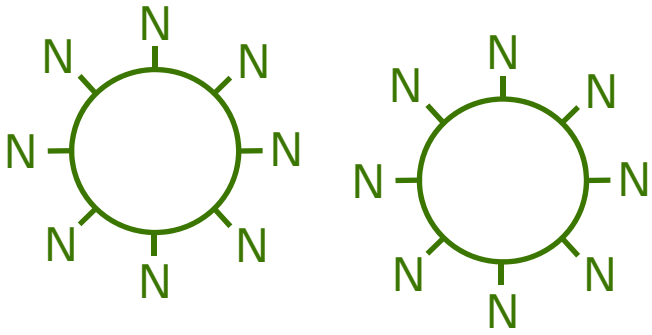
Column Family „Vegetable“

Carrot	2015-08-11	2015-08-12	…	2015-09-21
	65	50	…	87

NoSQL Data Models

Wide Column Stores: Cassandra

- Developed by Facebook, Apache project since 2009
- Cluster Architecture:
  - P2P system (ordered as rings)
  - Each node plays the same role (decentralized)
  - Each node accepts read/write operations



- User access through nodes via *Cassandra Query Language (CQL)*

NoSQL Data Models

Wide Column Stores: 

Consistency

- Tunable Data Consistency (choosable per operation)
- Read repair: if stale data is read, Cassandra issues a read repair → find most up-to-date data and update stale data
- Generally: Eventually consistent
- Main focus on availability!

Big Picture

CAP Theorem:

C

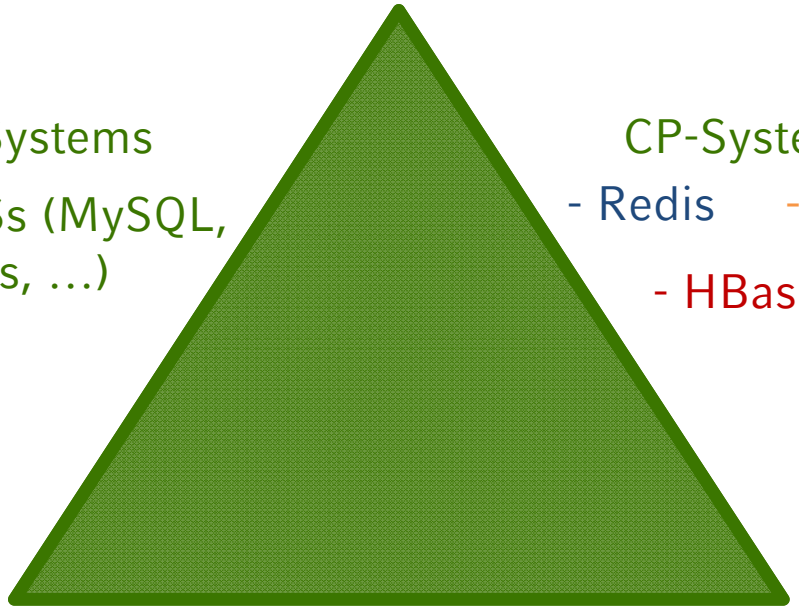
ACID

BASE

A

All clients always have the same view of the data

Key/Value Stores
Document Stores
Wide Column Stores



AC-Systems

- RDBMSs (MySQL, Postgres, ...)

CP-Systems

- Redis
- MongoDB
- HBase

Each client can always read and write

AP-Systems

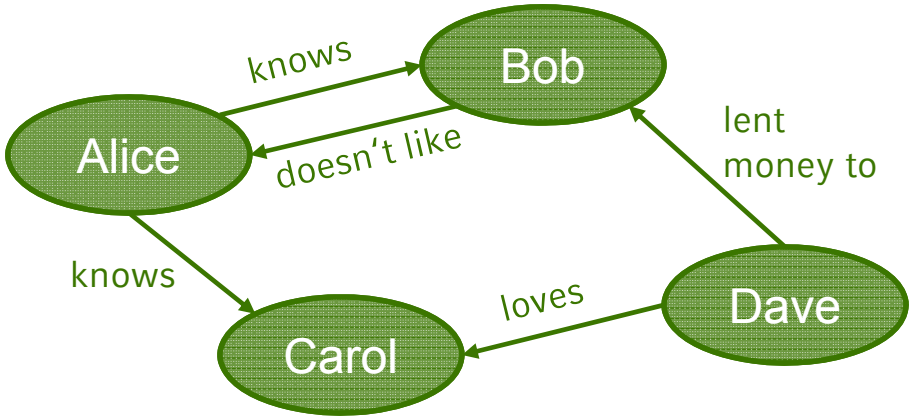
- Dynamo
- CouchDB
- Cassandra

The system works well despite physical network partitions

NoSQL Data Models

Graph Databases:

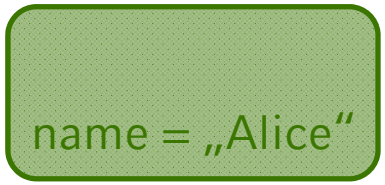
- Use graphs to store and represent relationships between entities
- Composed of *nodes* and *edges*
- Each node and each edge can contain *properties* (*Property-Graphs*)

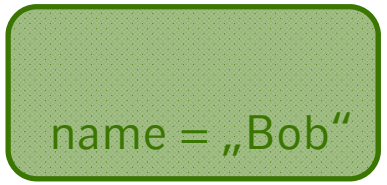


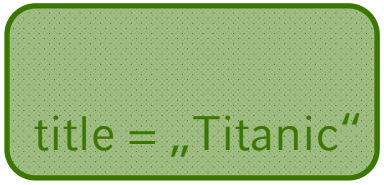
NoSQL Data Models

Graph Databases:

Alice is a friend of Bob and vice versa. They both love the movie „Titanic“.



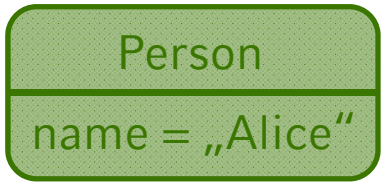


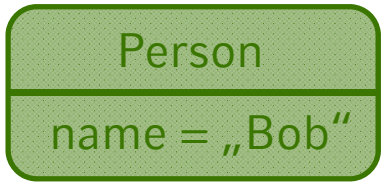


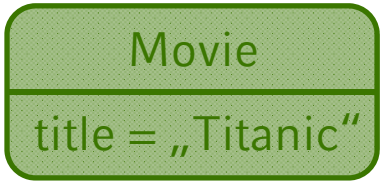
NoSQL Data Models

Graph Databases:

Alice is a friend of Bob and vice versa. They both love the movie „Titanic“.





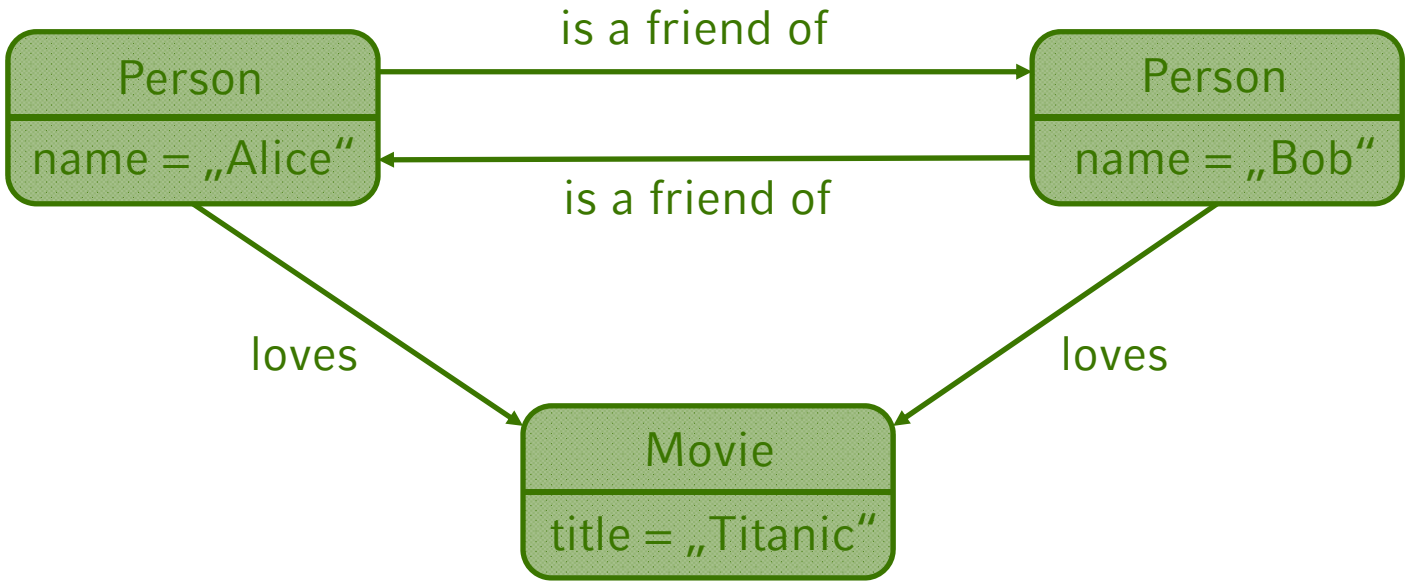


NoSQL Data Models

Graph Databases:

neo4j

Alice is a friend of Bob and vice versa. They both love the movie „Titanic“.



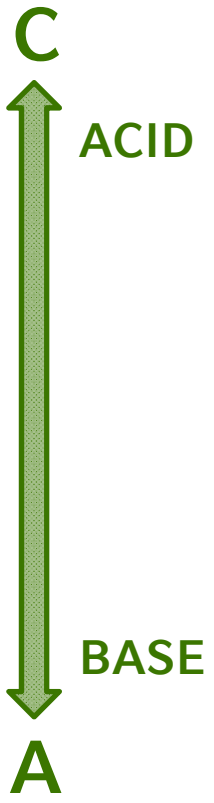
NoSQL Data Models

Graph Databases:

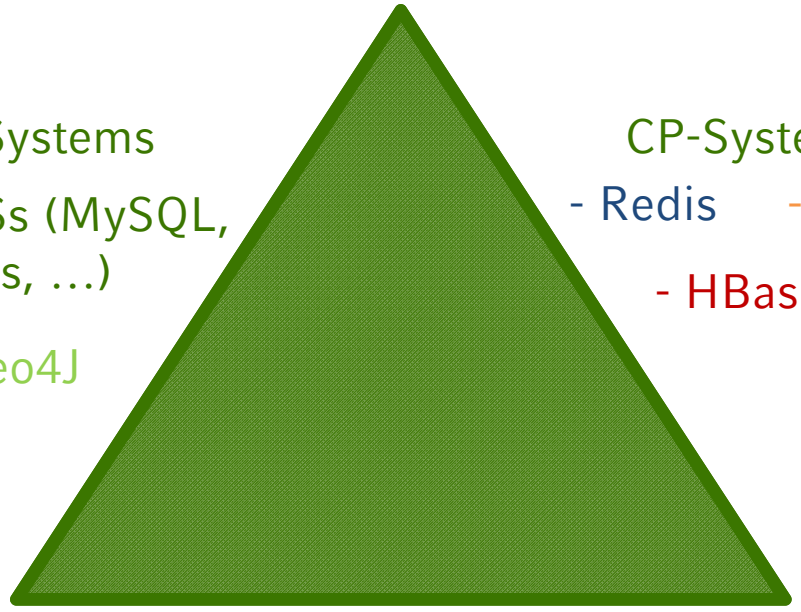
- Master-Slave Replication (no partitioning!)
- Consistency: Eventual Consistency (tunable to Immediate Consistency)
- Support of ACID Transactions
- Cypher Query Language
- Schema-optional

Big Picture

CAP Theorem:



All clients always have the same view of the data



Each client can always read and write

The system works well despite physical network partitions

AC-Systems
- RDBMSs (MySQL, Postgres, ...)
- Neo4J

CP-Systems
- Redis
- MongoDB
- HBase

AP-Systems
- Dynamo
- CouchDB
- Cassandra

Key/Value Stores
Document Stores
Wide Column Stores
Graph Databases