Orientation in Objects

CockroachDB

Jenseits von SQL und NoSQL

OIO-Hauskonferenz 2017

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Version: 1.0

Ihr Sprecher

Orientation in Objects

Tobias Polley

Trainer, Berater, Entwickler

Schwerpunkte
Integration
Microservices
Infrastructure Automation

Orientation in Objects

Die aktuelle Fassung dieser Folien gibt’s unter

https://github.com/rrayst/oio-hauskonferenz-2017

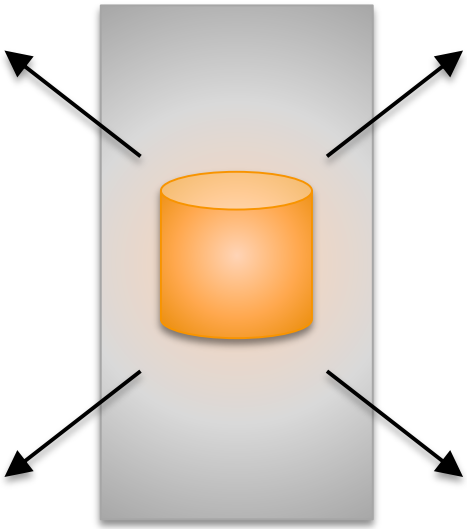
Gliederung

Orientation in Objects

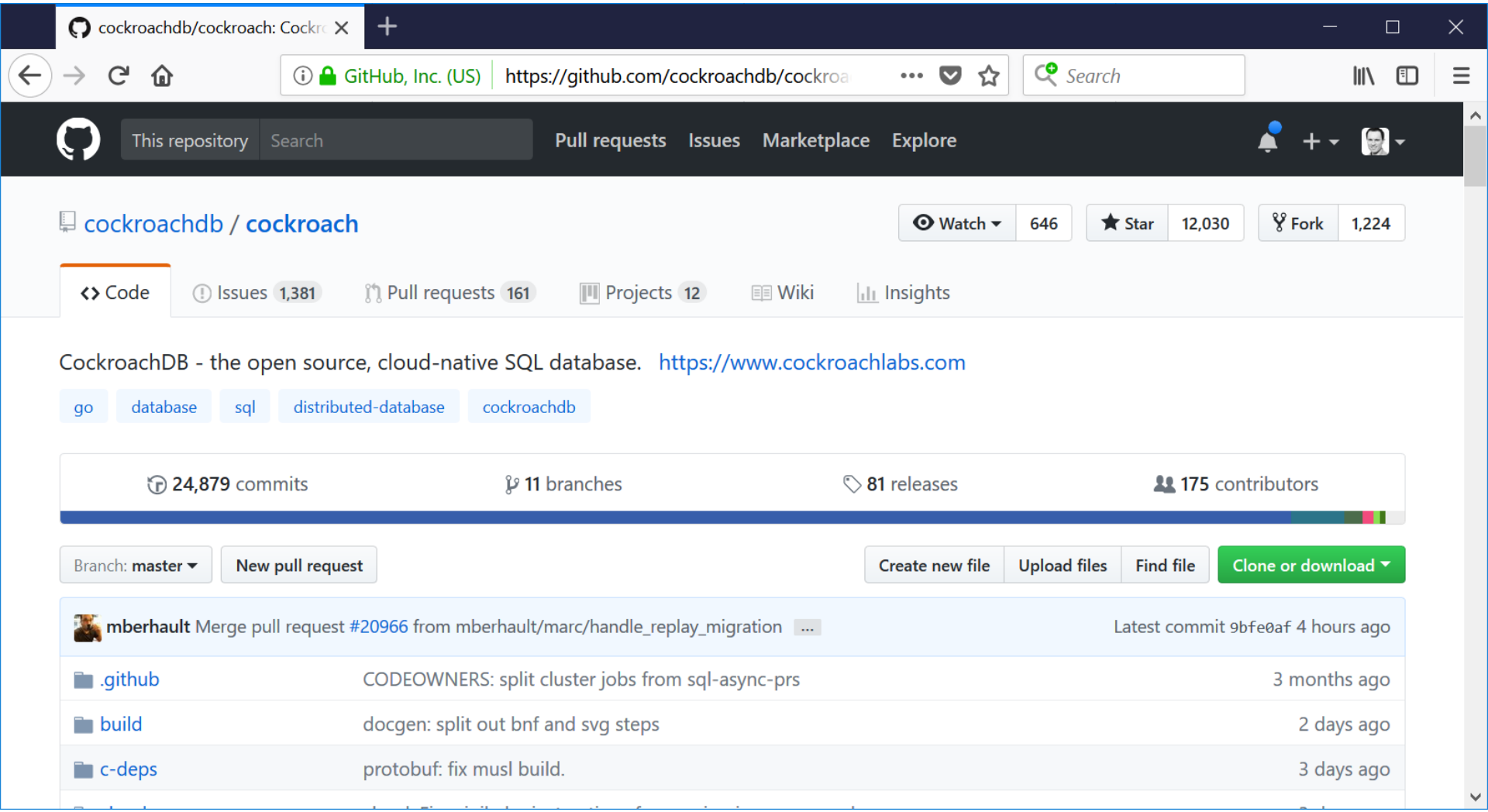
- Architektur und Konzepte
- Demo
- Skalieren
- Ausblick

Motivation

Orientation in Objects



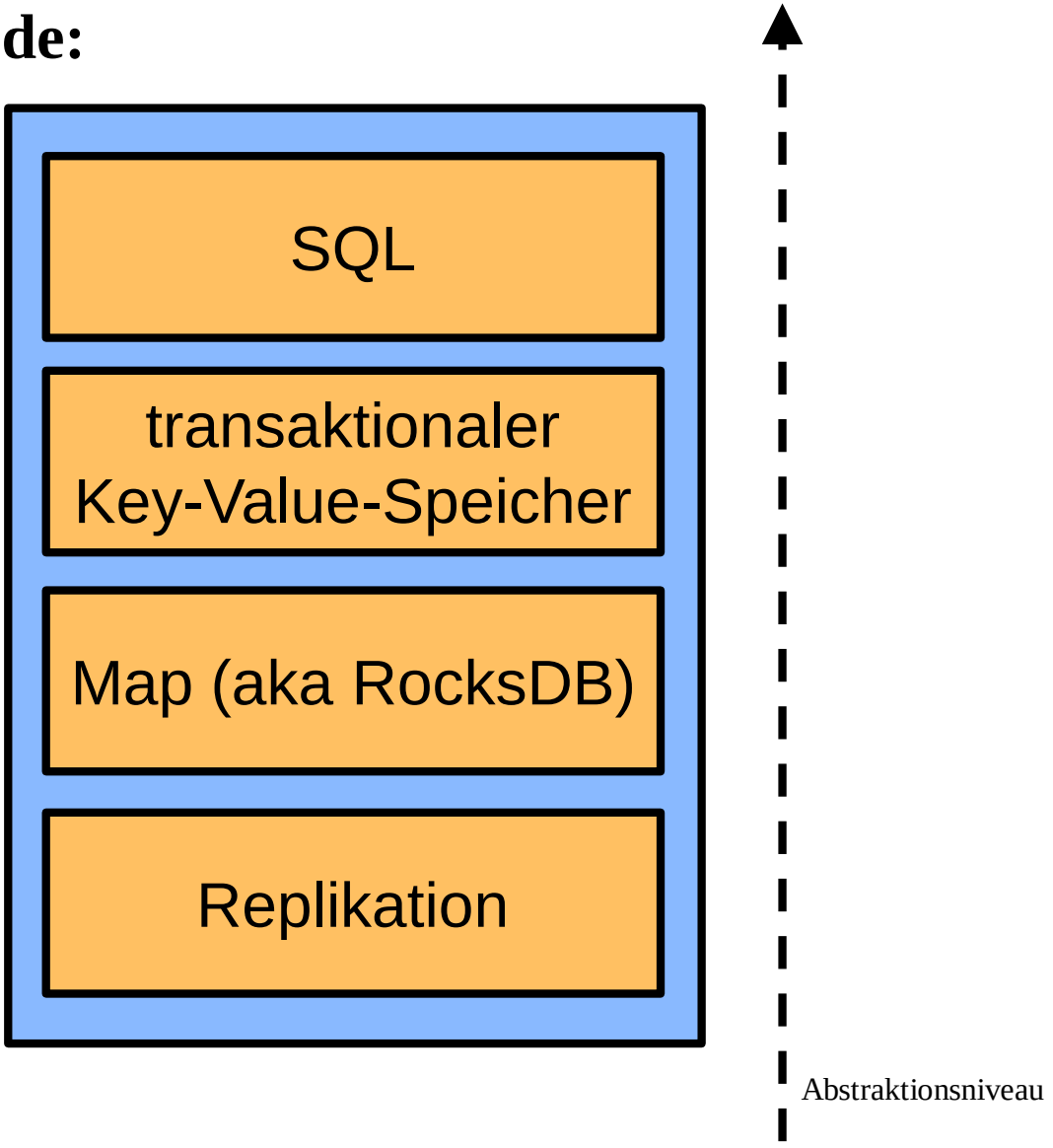
?


cockroachdb/cockroach: Cockr
GitHub, Inc. (US)
https://github.com/cockroachdb/cockroa
Search
This repository
Search
Pull requests
Issues
Marketplace
Explore
cockroachdb / cockroach
Watch 646
Star 12,030
Fork 1,224
Code
Issues 1,381
Pull requests 161
Projects 12
Wiki
Insights
CockroachDB - the open source, cloud-native SQL database. https://www.cockroachlabs.com
go
database
sql
distributed-database
cockroachdb
24,879 commits
11 branches
81 releases
175 contributors
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
mberhault Merge pull request #20966 from mberhault/marc/handle_replay_migration ...
Latest commit 9bfe0af 4 hours ago
.github
CODEOWNERS: split cluster jobs from sql-async-prs
3 months ago
build
docgen: split out bnf and svg steps
2 days ago
c-deps
protobuf: fix musl build.
3 days ago

Architektur

Orientation in Objects

Code:



```
CREATE TABLE test (
       id      INTEGER PRIMARY KEY,
       name    VARCHAR,
       price   FLOAT
);

INSERT INTO test VALUES (1, “ball”, 2.22);
```

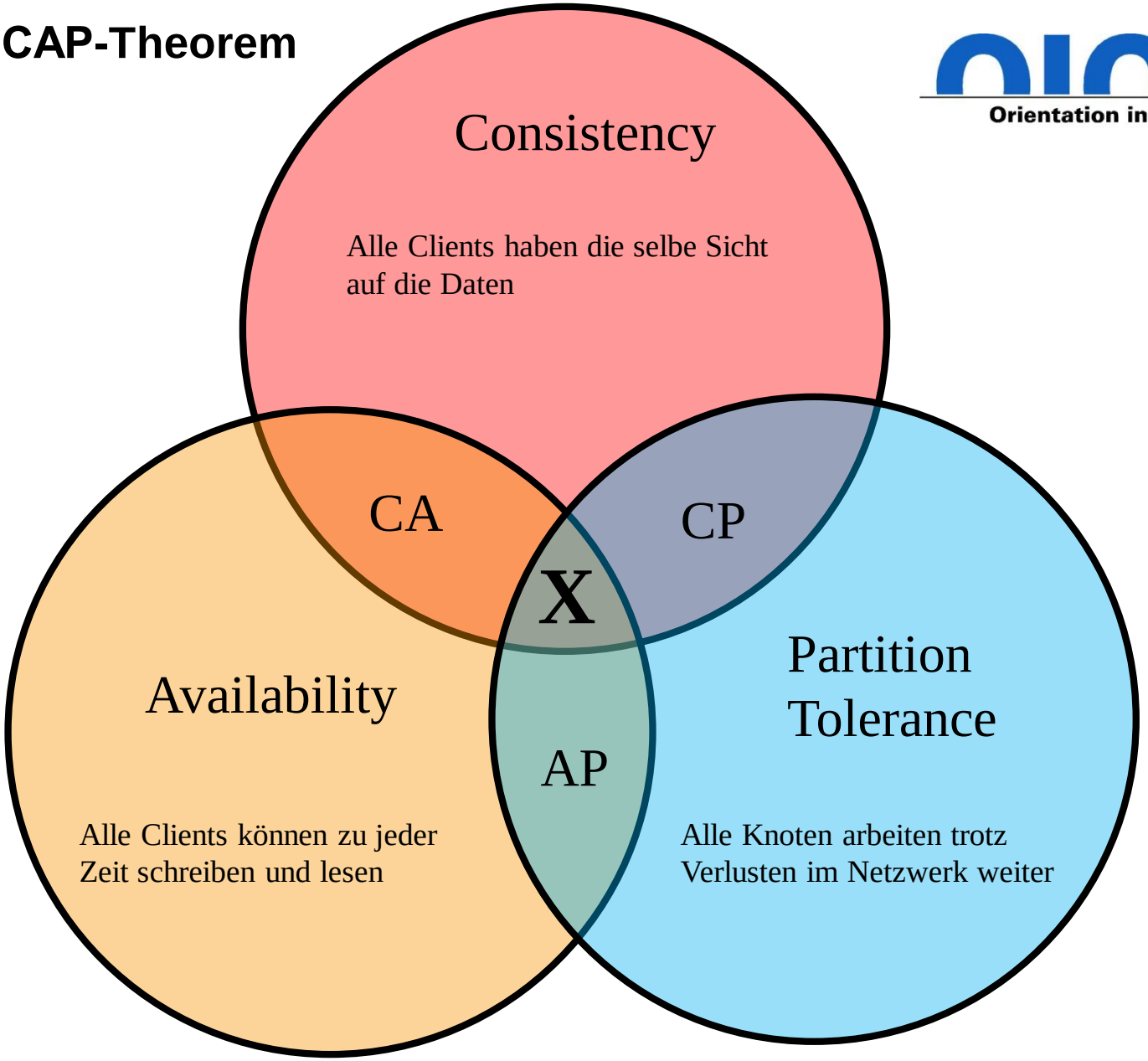
Key	Value
/test/primary/1/name	“ball”
/test/primary/1/price	2.22

```
CREATE TABLE test (
       id      INTEGER PRIMARY KEY,
       name    VARCHAR,
       price   FLOAT
);

INSERT INTO test VALUES (1, “ball”, 2.22);
INSERT INTO test VALUES (2, “apple”, 3.33);
```

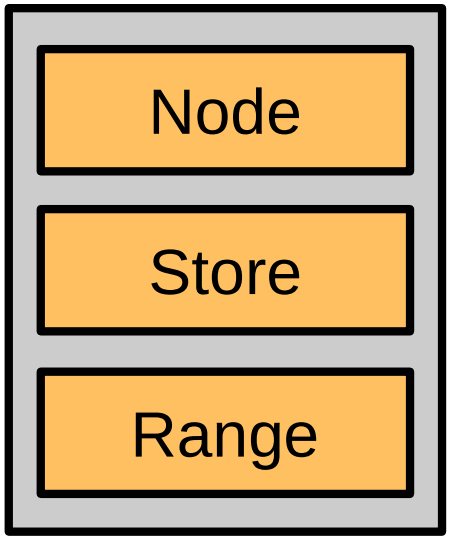
Key	Value
/test/primary/1/name	“ball”
/test/primary/1/price	2.22
/test/primary/2/name	“apple”
/test/primary/2/price	3.33

Brewer's CAP-Theorem



Orientation in Objects

Speicher:



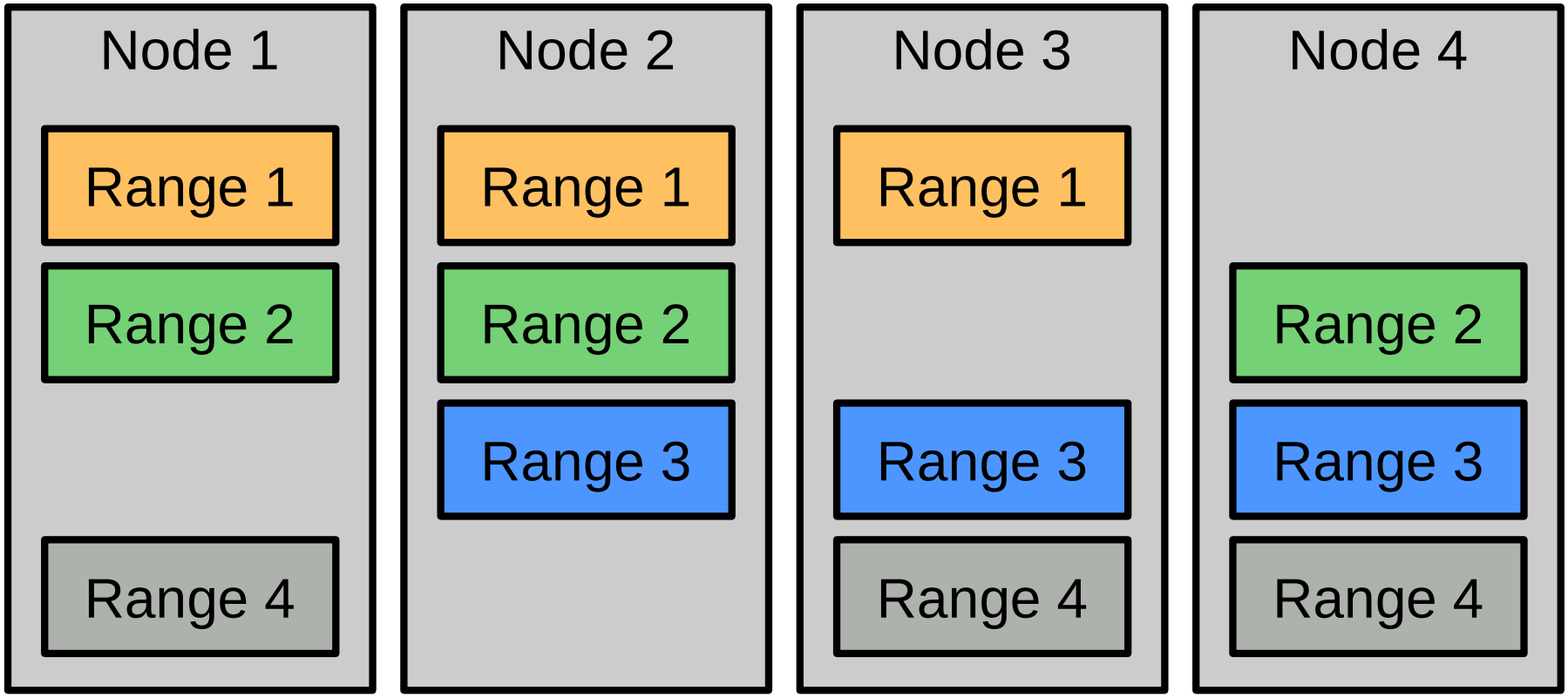
laufender `cockroach` Prozess

Datenverzeichnis, Festplatte

64MB-Auszug sortierter KV-Daten

Replikation der Daten

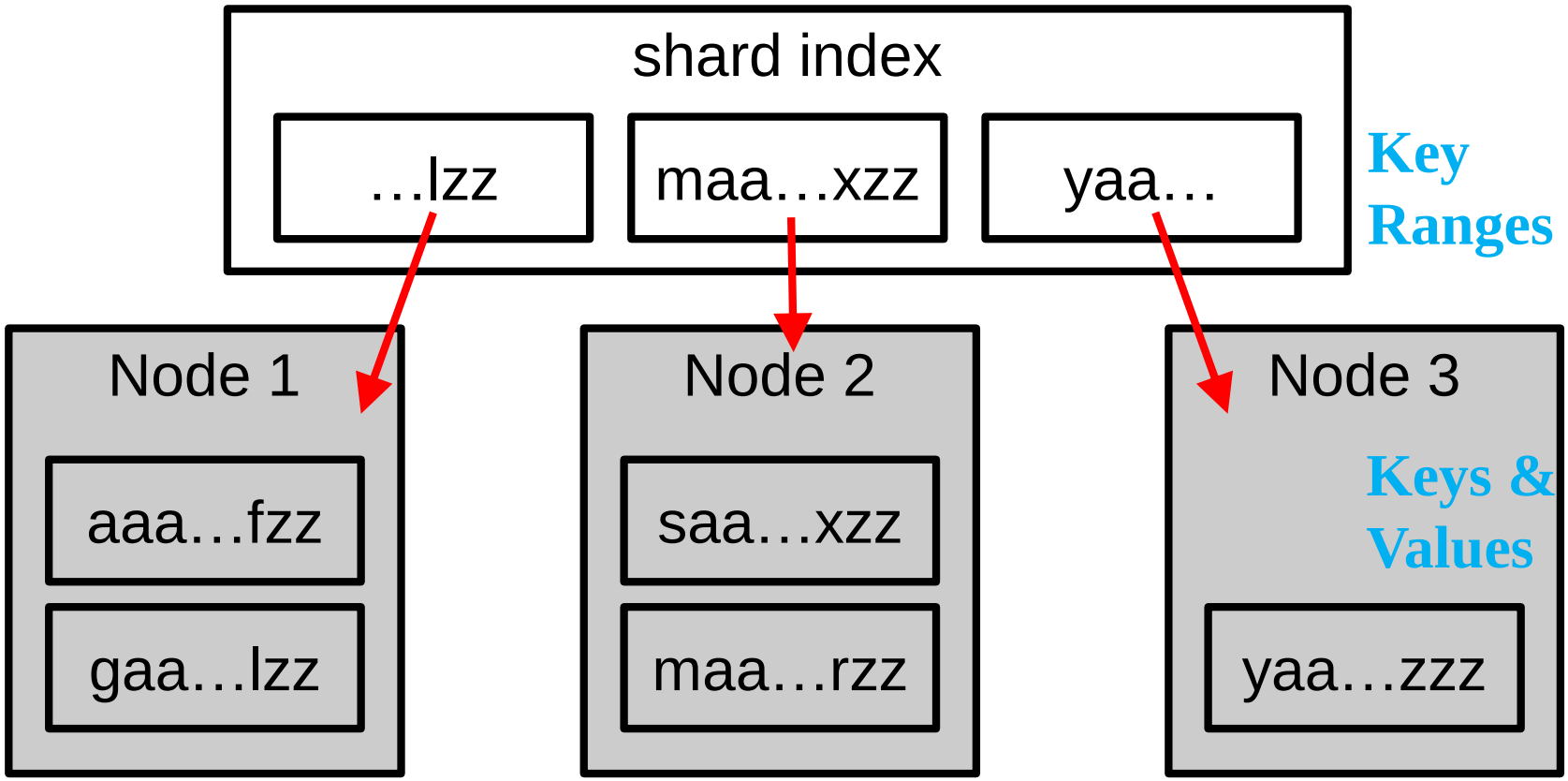
Orientation in Objects



Jeder Range wird 3x abgelegt. Diese Kopien bilden eine Raft-Gruppe.

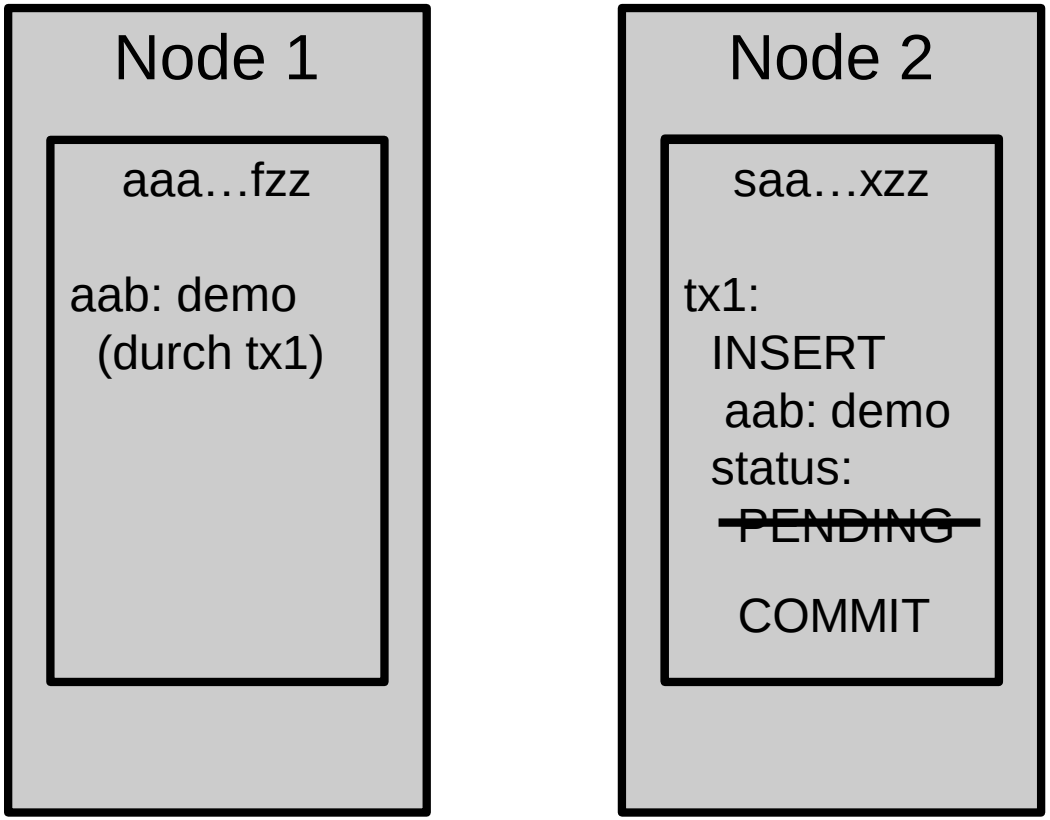
Verteilte Daten

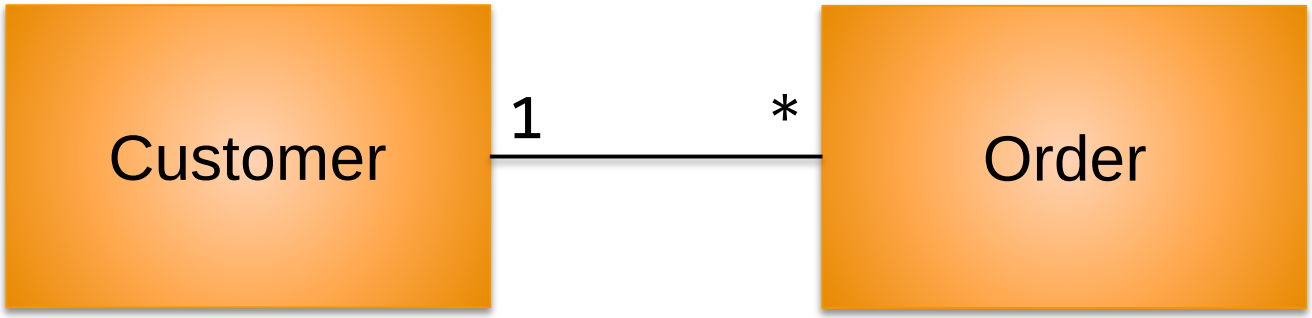
- Zuordnung auf Knoten *nicht* mittels Hash-Funktion
  - einfache Interaktion
- Stattdessen werden Daten geordnet in 64MB Ranges abgelegt.
  - zusätzlicher Index verwaltet die Zuordnung:



Transaktionen

Orientation in Objects




Customer
1
*
Order

Unser Beispiel

Orientation in Objects

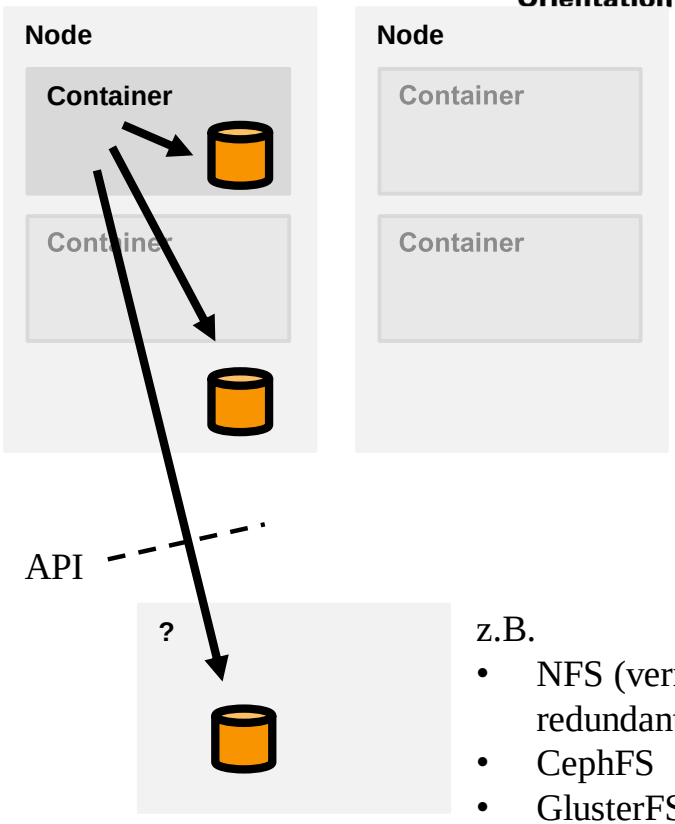
```
CREATE TABLE customers (
    id INT PRIMARY KEY,
    name STRING(50)
);

CREATE TABLE orders (
    customer INT,
    id INT,
    total DECIMAL(20, 5),
    PRIMARY KEY (customer, id),
    CONSTRAINT fk_customer FOREIGN KEY (customer)
                                        REFERENCES customers (id)
);
```

```
INSERT INTO customers (id, …) VALUES (:i, …);
INSERT INTO orders (…) VALUES (:i, …);
```

20 Threads = 20 TCP-Verbindungen
1 Transaktion pro 50 Customer
(also 100 Zeilen/TX, 0.5MB/TX)

Orientation in Objects



ZUSTAND

(aka. “wohin mit den Daten?”)

z.B.

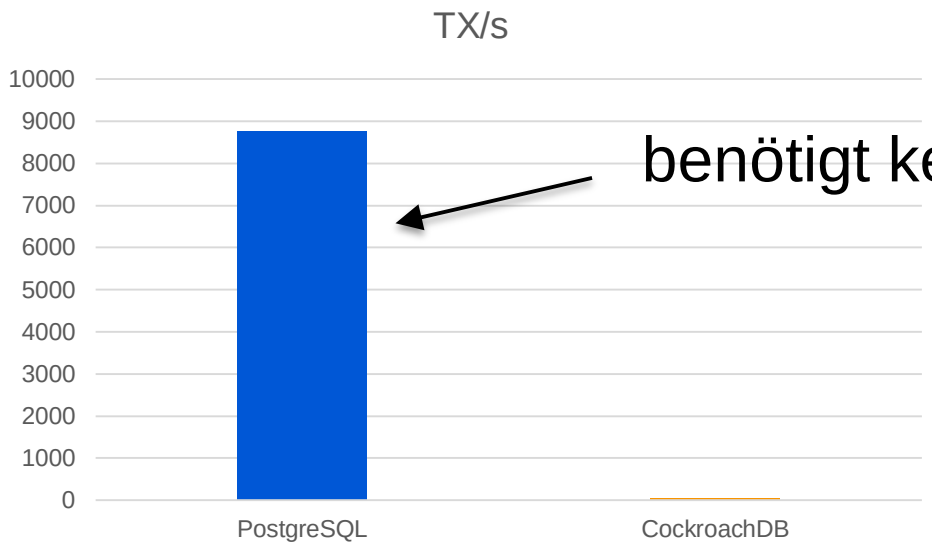
- NFS (vermutlich nicht redundant!)
- CephFS
- GlusterFS

Orientation in Objects

```
INSERT INTO customers (id, …) VALUES (:i, …);
INSERT INTO orders (…) VALUES (:i, …);
```

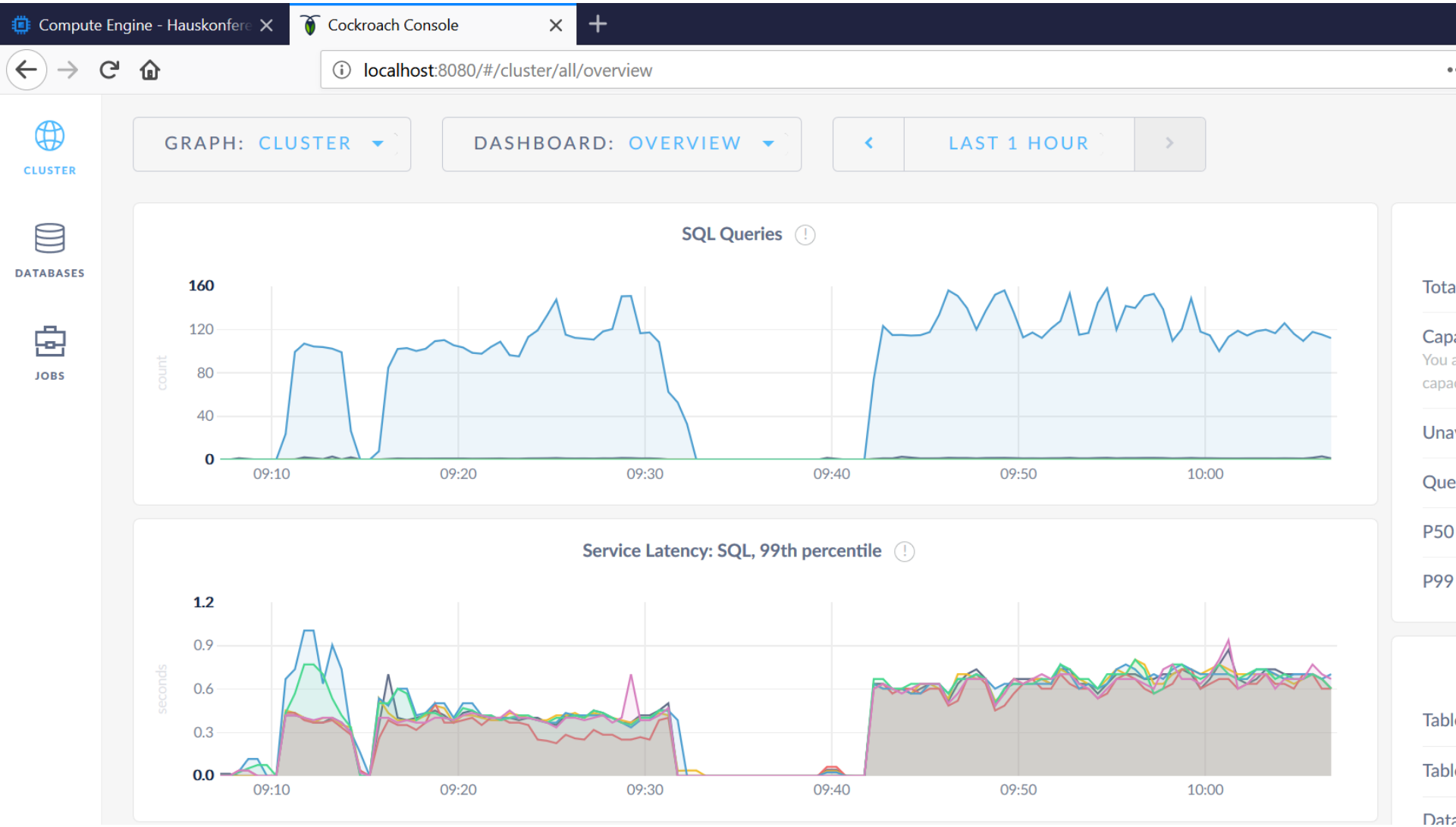
20 Threads = 20 TCP-Verbindungen
1 Transaktion pro 50 Customer
(also 100 Zeilen/TX, 0.5MB/TX)

läuft mit 53 TX/s



benötigt kein Netzwerk!

PostgreSQL	CockroachDB
1 Knoten	6 Knoten
mein Laptop	Google Cloud Platform
2.7GHz i7	2.6GHz Xeon 1 Kern
32GB RAM	3.75 GB RAM
500GB SSD	375GB SSD


Compute Engine - Hauskonfere
Cockroach Console
localhost:8080/#/cluster/all/overview
CLUSTER
DATABASES
JOBS
GRAPH: CLUSTER
DASHBOARD: OVERVIEW
LAST 1 HOUR
SQL Queries
160
120
80
40
0
count
09:10
09:20
09:30
09:40
09:50
10:00
Service Latency: SQL, 99th percentile
1.2
0.9
0.6
0.3
0.0
seconds
09:10
09:20
09:30
09:40
09:50
10:00


Compute Engine - Hauskonfere
Cockroach Console
localhost:8080/#/cluster/all/overview
CLUSTER
DATABASES
JOBS
Replicas per Node
count
60
40
20
0
09:30
09:40
09:50
10:00
10:10
10:20
Capacity
Capacity
Available
Used
terabytes
2.4
1.8
1.2
0.6
0.0
09:30
09:40
09:50
10:00
10:10
10:20

währenddessen…

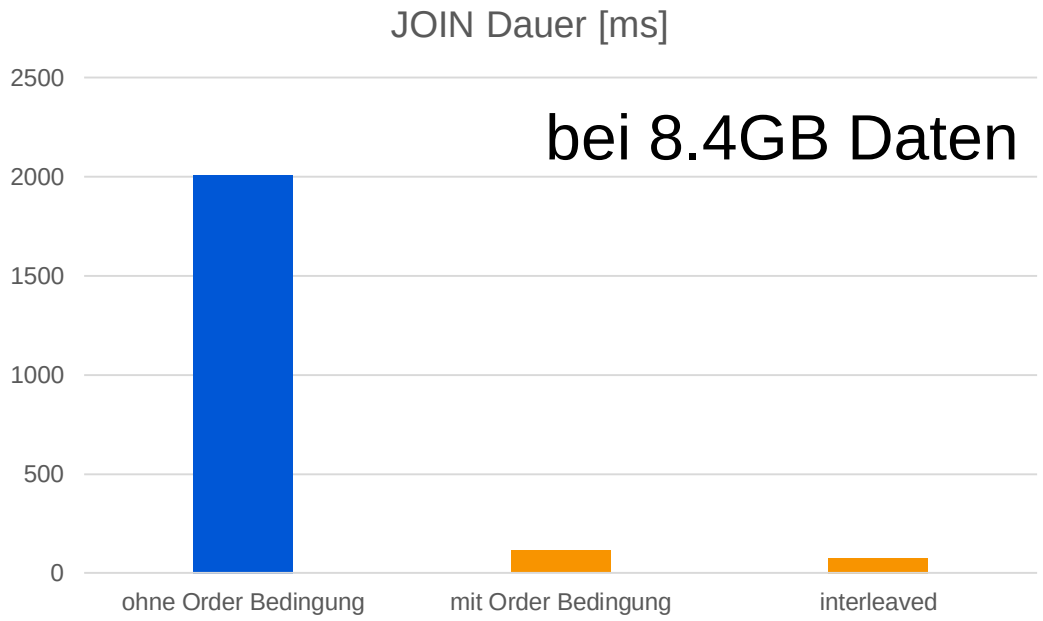
Orientation in Objects

```
org.postgresql.util.PSQLException: ERROR: restart
transaction: HandledRetryableTxnError:
TransactionRetryError: retry txn (RETRY_SERIALIZABLE):
"sql txn" id=2571f863 key=/Table/51/1/67019/0 rw=true
pri=0.04850156 iso=SERIALIZABLE stat=PENDING epo=0
ts=1513848638.487393417,1 orig=1513848638.411042960,0
max=1513848638.411042960,0 wto=false rop=false seq=2]
with root cause
```

GitHub: haben auch andere Leute unter Last beobachtet

```
select
  c.id, c.name, o.id, o.total
from
  customers as c
  inner join orders as o on c.id = o.customer
where
  c.id = ?
```

```
select
  c.id, c.name, o.id, o.total
from
  customers as c
  inner join orders as o on c.id = o.customer
where
  c.id = ?
  and o.customer = ?
```

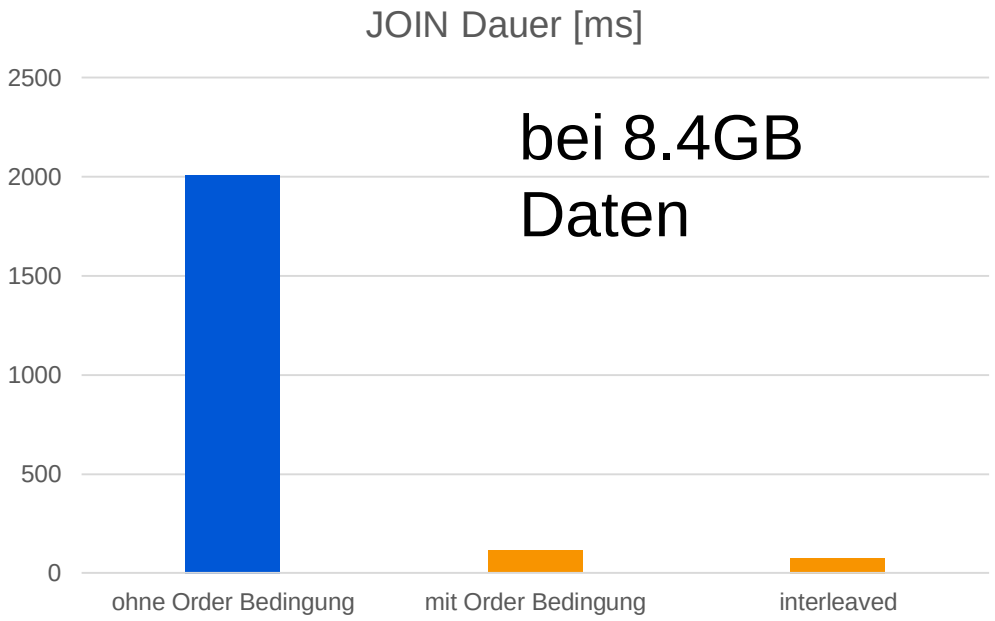


Interleaving

Orientation in Objects

- Daten, die häufig zusammen gelesen oder geschrieben werden, profitieren davon, nahe beisammen abgespeichert zu werden.
  - ähnlicher Key

```
/customers/1/
/customers/1/orders/1000
/customers/1/orders/1002
/customers/2
/customers/2/orders/1001
/customers/2/orders/1003
```



Interleaving

Orientation in Objects

- SQL-Erweiterung:

```
CREATE TABLE customers (
    id INT PRIMARY KEY,
    name STRING(50)
);

CREATE TABLE orders (
    customer INT,
    id INT,
    total DECIMAL(20, 5),
    PRIMARY KEY (customer, id),
    CONSTRAINT fk_customer FOREIGN KEY (customer)
                                        REFERENCES customers (id)
) INTERLEAVE IN PARENT customers (customer);
```

- ACID
  - Atomicity
  - Consistency
  - Isolation
  - Durability

- CockroachDB unterstützt die Isolationslevel `SERIALIZABLE` und `SNAPSHOT`.

Orientation in Objects

Einsatzszenarien

aktuelle Entwicklungen

Lizenz

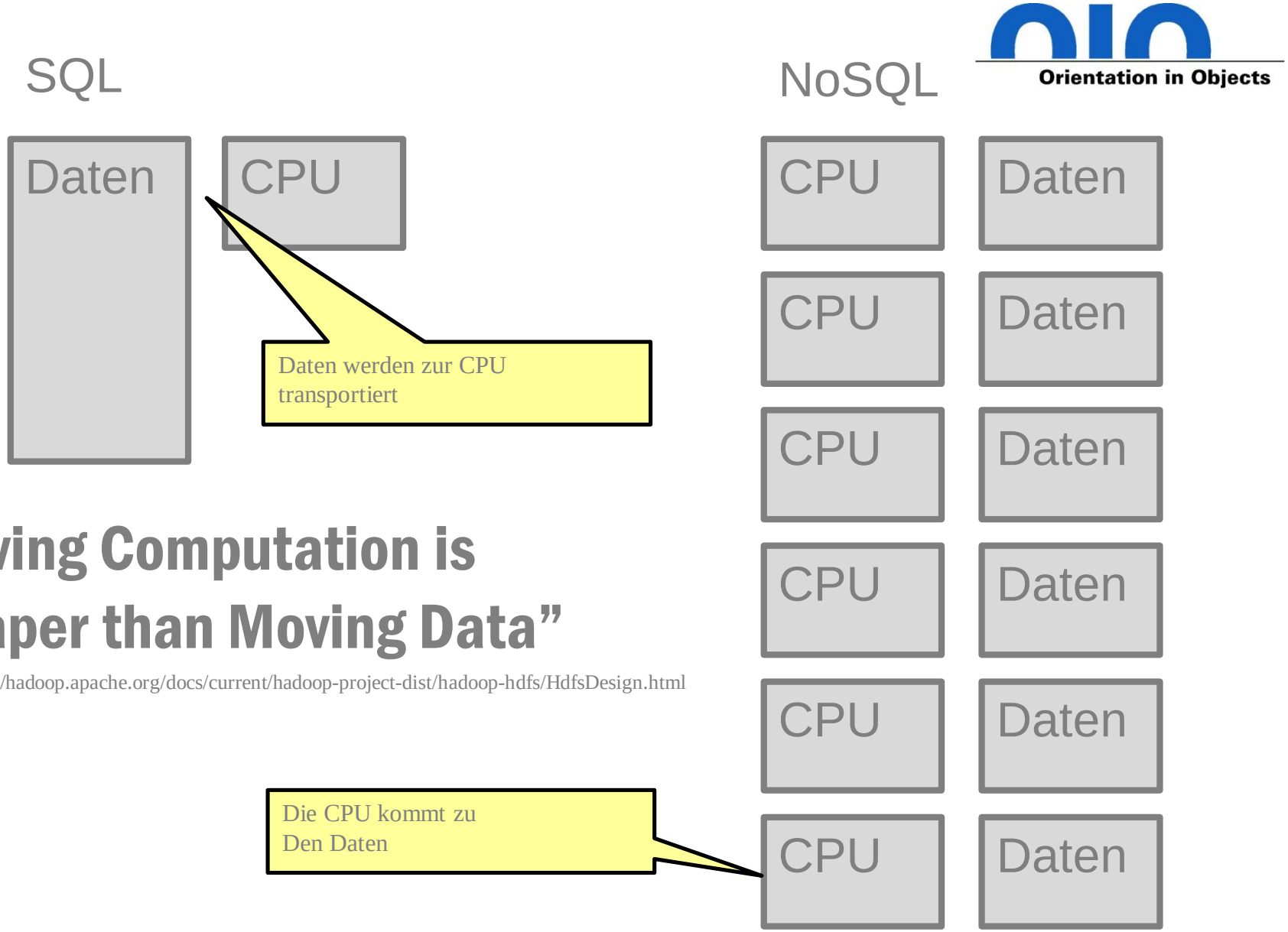
Orientation in Objects

Vielen Dank für Ihre Aufmerksamkeit !

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de



“Moving Computation is Cheaper than Moving Data”

Quelle: http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

Ressourcenverbrauch

Orientation in Objects

- Storage:
  - k1 × (DataSize × ReplicationFactor) / NumNodes + other
- Memory:
  - k2 × NumInFlightOperations / (NumNodes × NodeSpeed) + other

- siehe https://www.cockroachlabs.com/blog/memory-usage-cockroachdb/

Postgres-XL

Orientation in Objects

- central Global Transaction Manager (GTM) node.
- requires a fast interconnect between nodes
  - not suited to geographic distribution

Orientation in Objects

Fragen ?

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Orientation in Objects

Vielen Dank für Ihre Aufmerksamkeit !

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de