GOPHER CHINA 2020

中国 上海 / 2020-11.21-22

探探分布式存储的实践

彭亮

Agenda

- Why tantan db
- What is tantan db
- Golang
- Roadmap

Why do we need tantan db?

tantan

左滑无感，右滑喜欢

破冰利器，附近动态

擦肩而过，回眸一笑

tantan

业务特性

- 大数据量
- 数据快速增长
- 低时延

数据特性

- 分区
- 聚集性
  Clustered

需求

架构师

- 分布式
- 可用性
- 扩展性
- 定制化

使用者

- SQL
- BASE vs ACID
- PACELC

DBA

- 存储成本
- 运维友好
- 容灾能力

开源方案

NoSQL

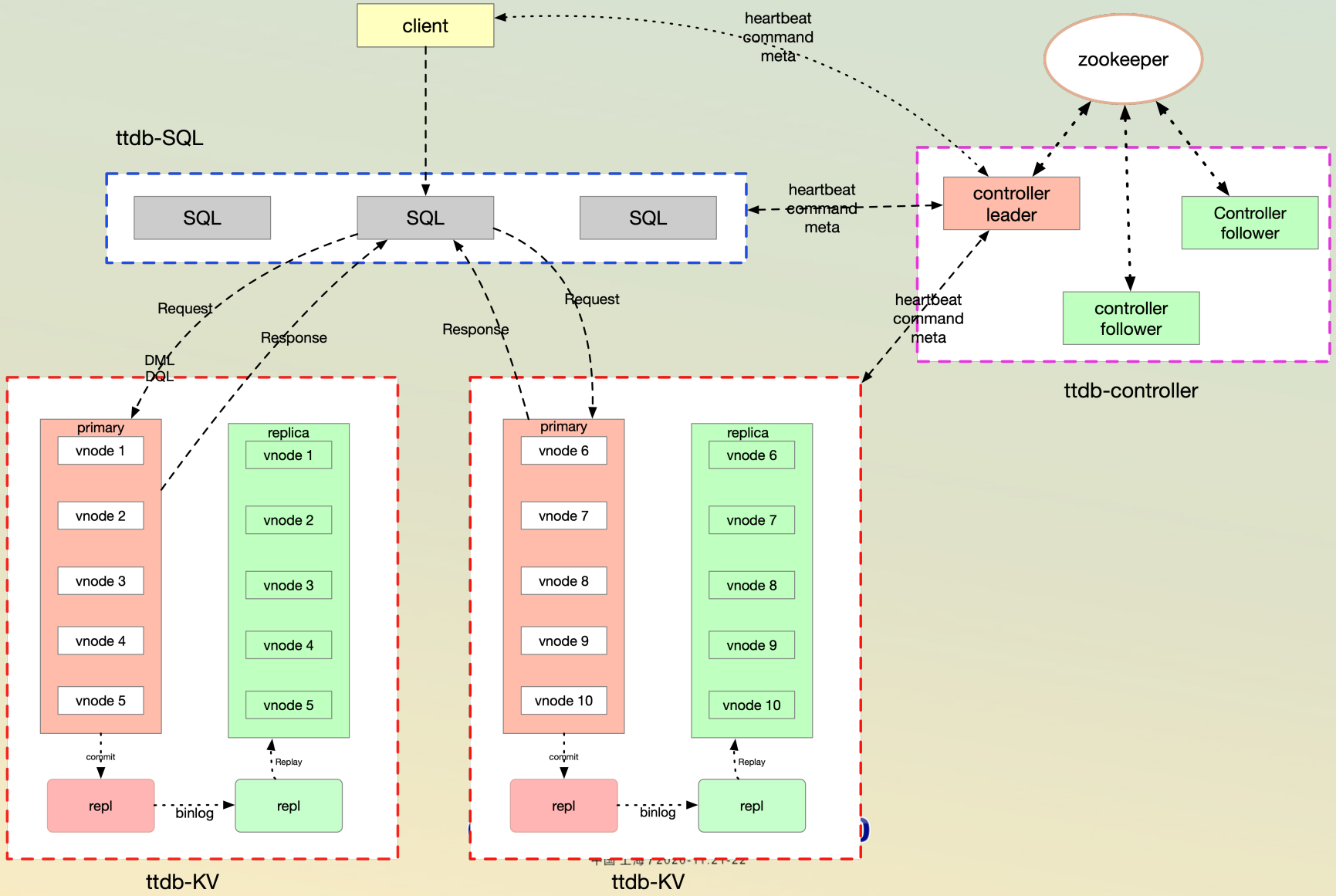
- SQL限制
- 存储成本
- 定制化

NewSQL

- 存储成本
- 强一致性
- ACID
- 延迟
- 定制化

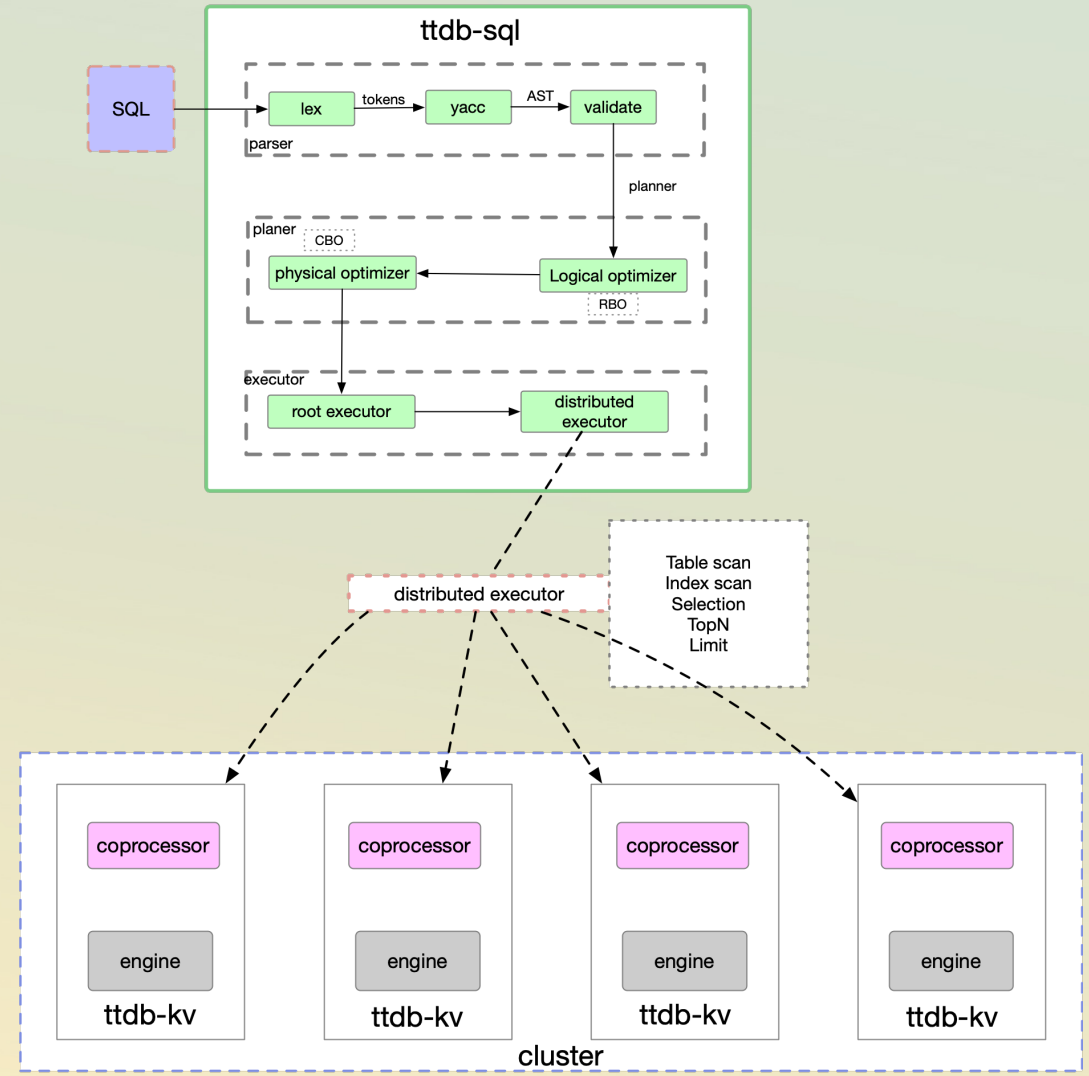
What is tantan db(ttdb)?

架构


client
heartbeat
command
meta
zookeeper
ttdb-SQL
SQL
SQL
SQL
heartbeat
command
meta
controller
leader
Controller
follower
controller
follower
heartbeat
command
meta
ttdb-controller
Request
Response
Response
Request
DML
DQL
primary
vnode 1
vnode 2
vnode 3
vnode 4
vnode 5
replica
vnode 1
vnode 2
vnode 3
vnode 4
vnode 5
commit
Replay
repl
binlog
repl
ttdb-KV
primary
vnode 6
vnode 7
vnode 8
vnode 9
vnode 10
replica
vnode 6
vnode 7
vnode 8
vnode 9
vnode 10
commit
Replay
repl
binlog
repl
ttdb-KV

SQL

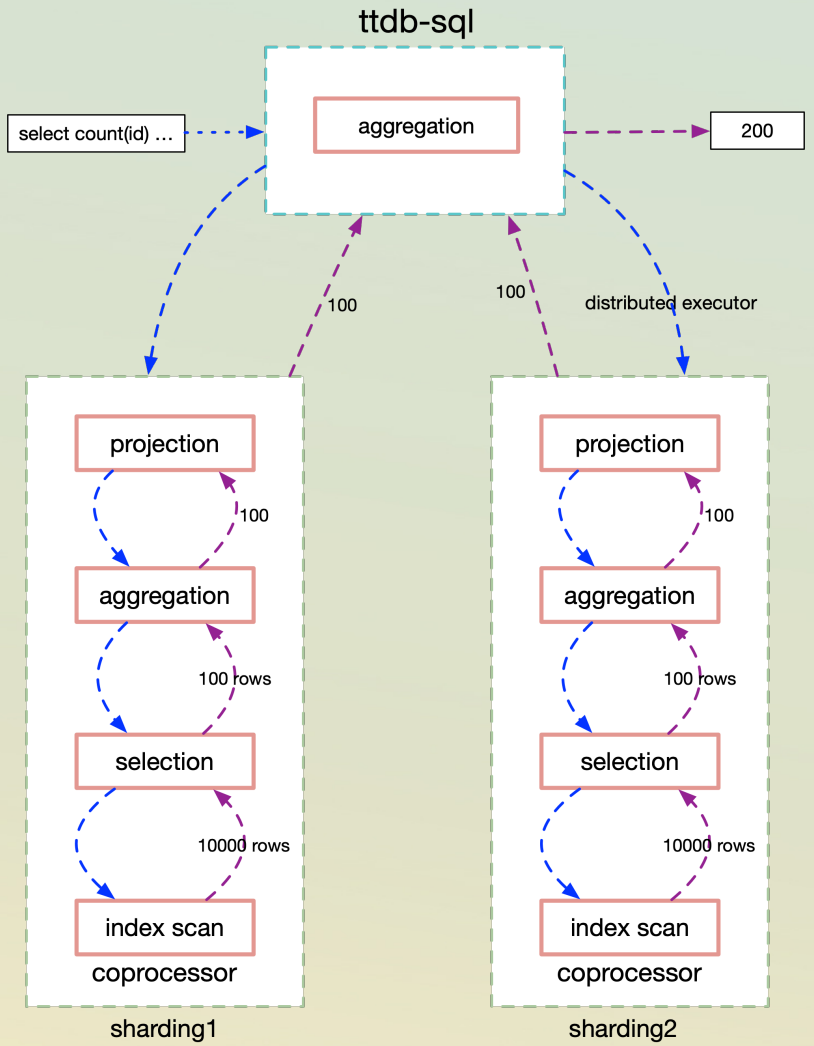
SQL 执行


ttdb-sql
SQL
lex
tokens
yacc
AST
validate
parser
planner
planer
CBO
physical optimizer
Logical optimizer
RBO
executor
root executor
distributed executor
distributed executor
Table scan
Index scan
Selection
TopN
Limit
coprocessor
engine
ttdb-kv
coprocessor
engine
ttdb-kv
coprocessor
engine
ttdb-kv
coprocessor
engine
ttdb-kv
cluster

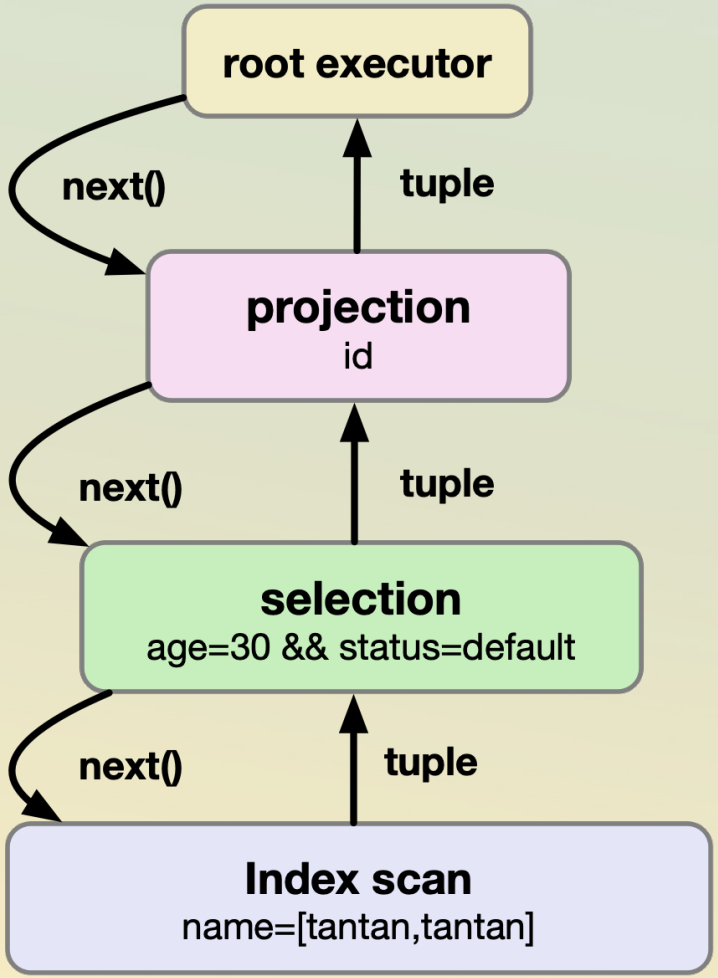
SQL 优化器

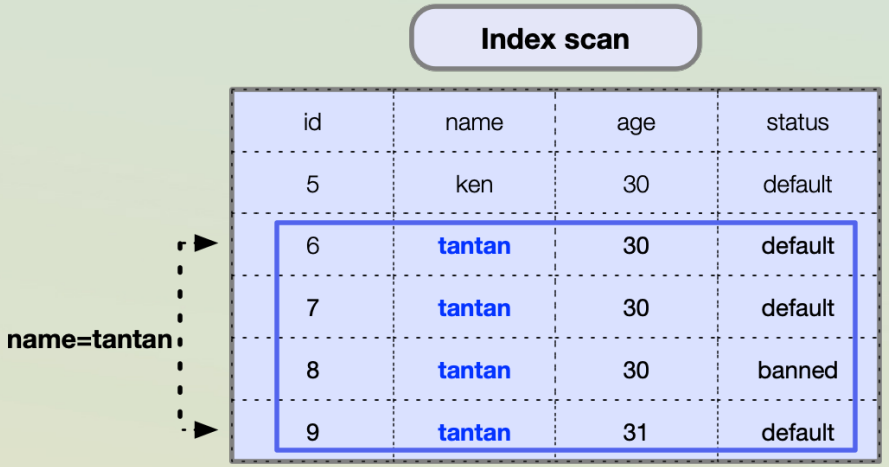
Rule based optimizer

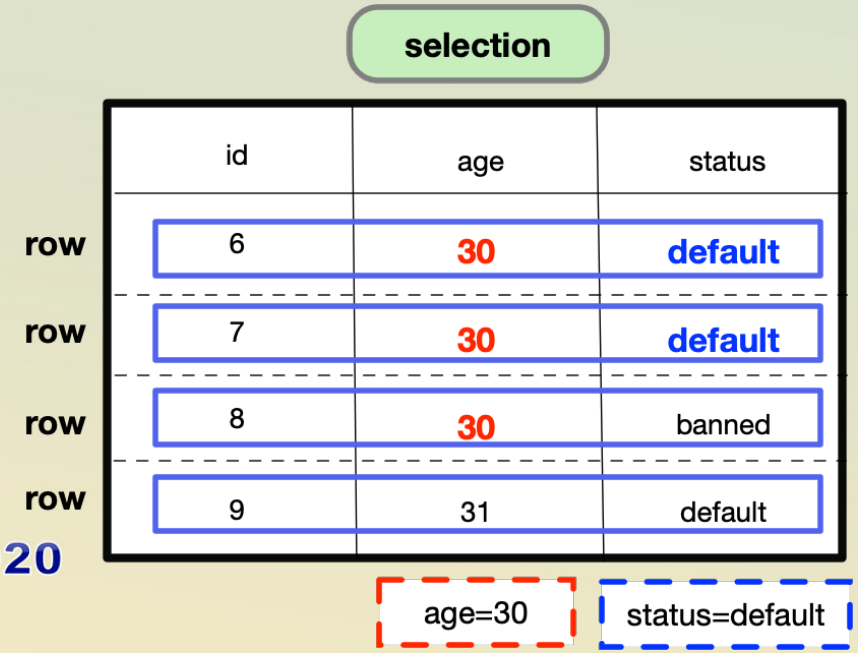
- 列裁剪(prune columns)
- 谓词下推(push down predicate)
- 聚合下推(push down aggregation)
- topN下推 (push down topN)



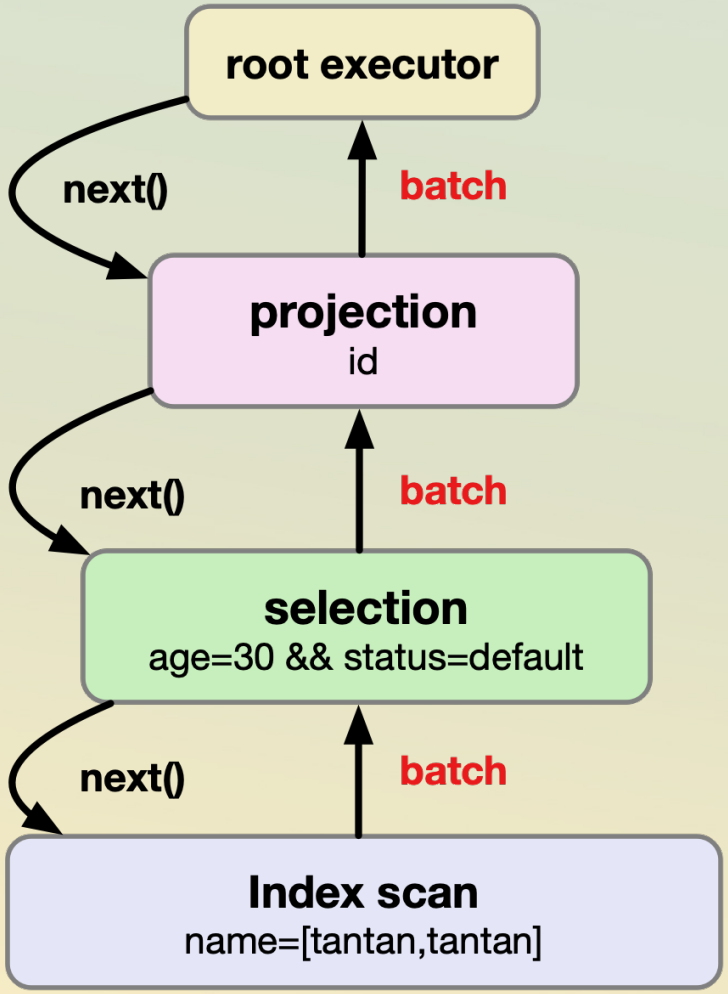
Volcano model executor

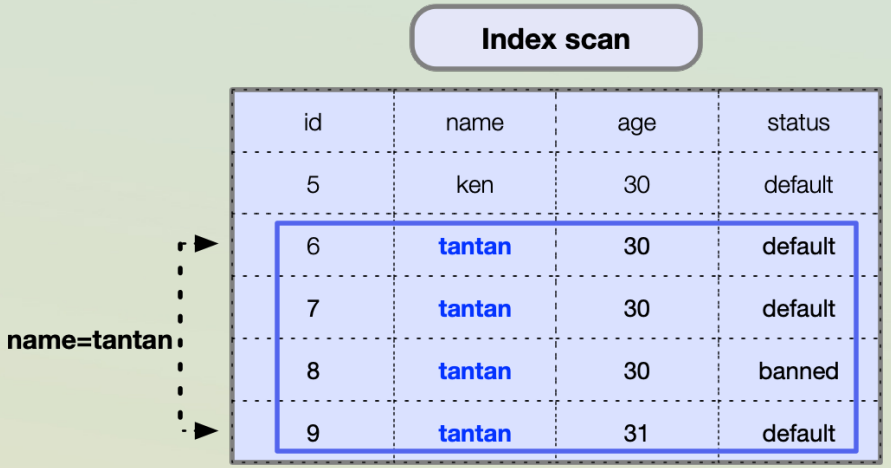


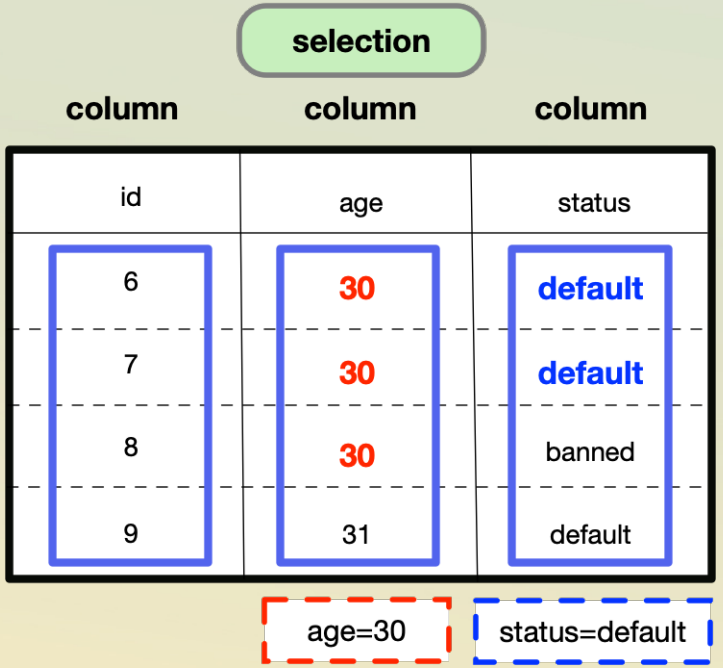




Vectorization model executor

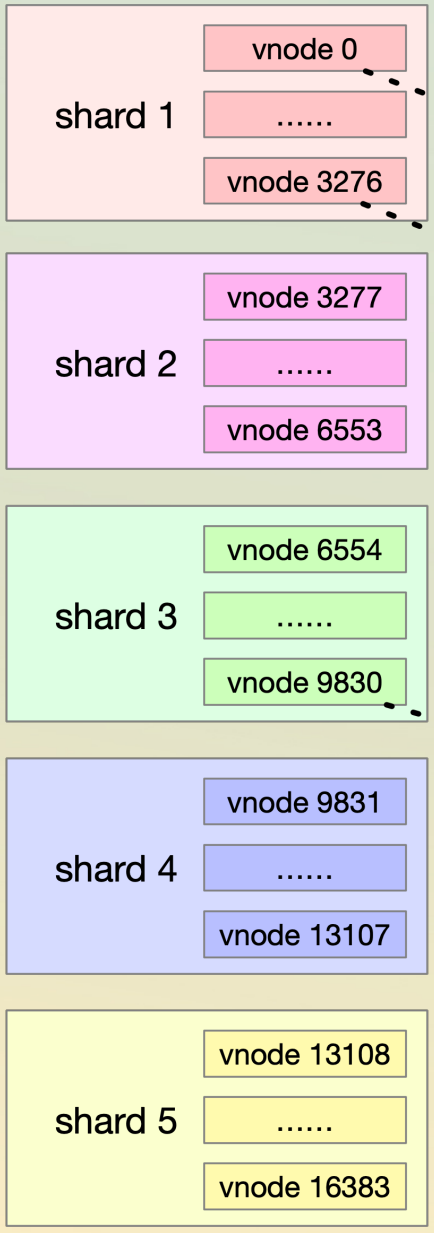






数据分片

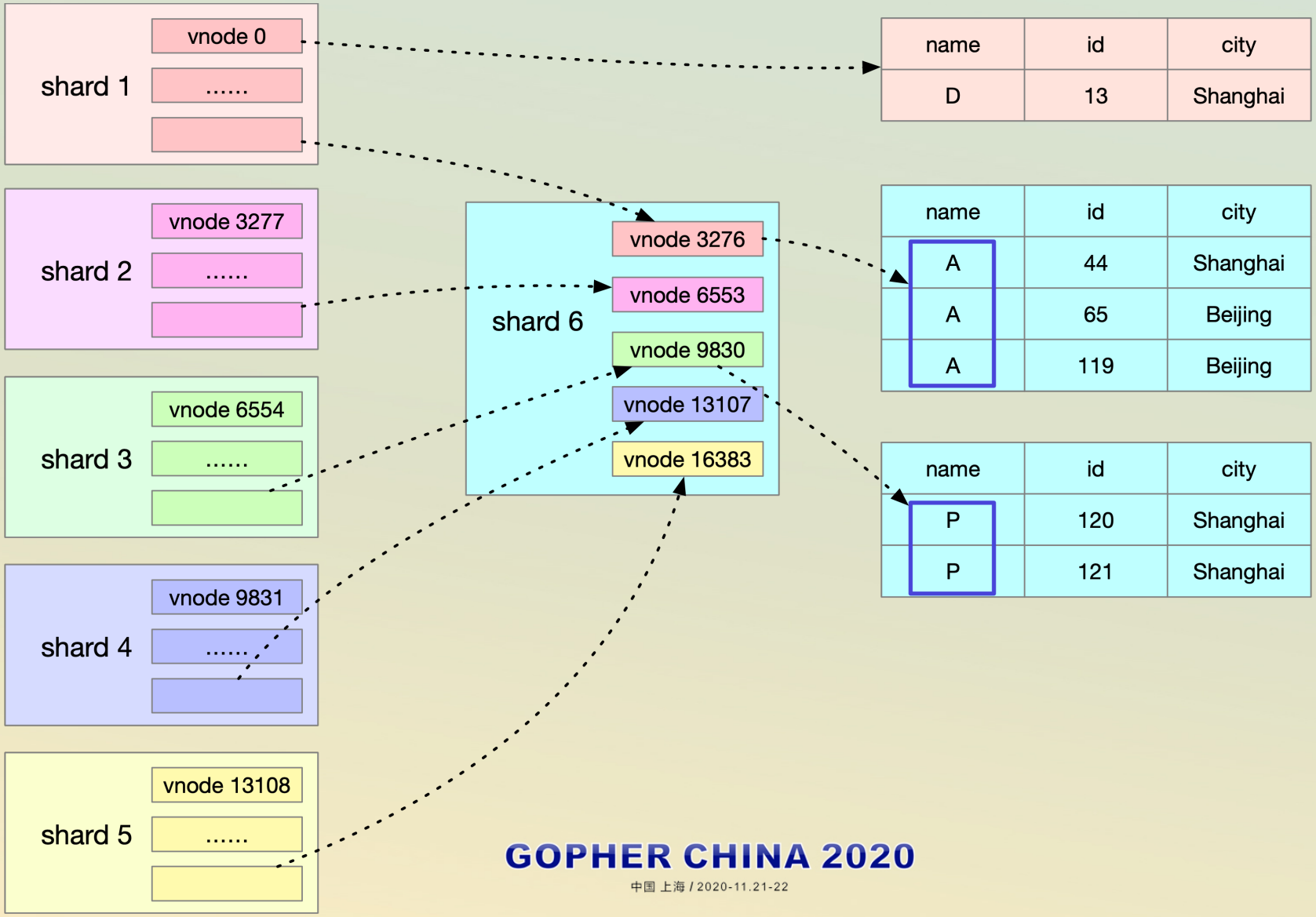
数据分片



name	id	city
D	13	Shanghai
A	44	Shanghai
A	65	Beijing
A	119	Beijing

name	id	city
P	120	Shanghai
P	121	Shanghai
B	140	Beijing
B	141	Beijing

扩容


shard 1
vnode 0
.....
shard 2
vnode 3277
.....
shard 3
vnode 6554
.....
shard 4
vnode 9831
.....
shard 5
vnode 13108
.....
shard 6
vnode 3276
vnode 6553
vnode 9830
vnode 13107
vnode 16383
name id city
D 13 Shanghai
name id city
A 44 Shanghai
A 65 Beijing
A 119 Beijing
name id city
P 120 Shanghai
P 121 Shanghai

高可用

故障检测

中心化

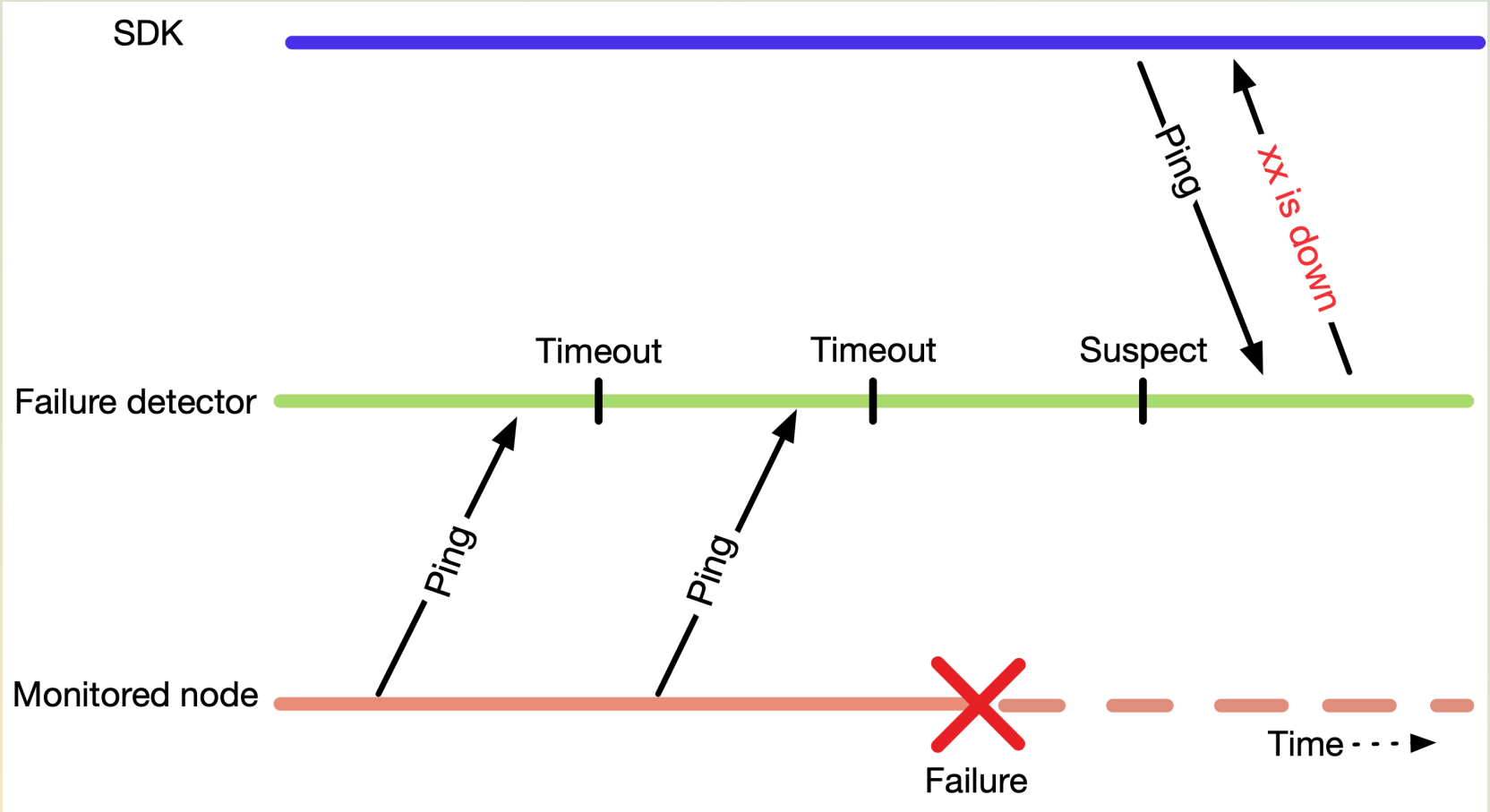
- Timeout(binary or accrual) vs false positive
- Network partition

无中心化

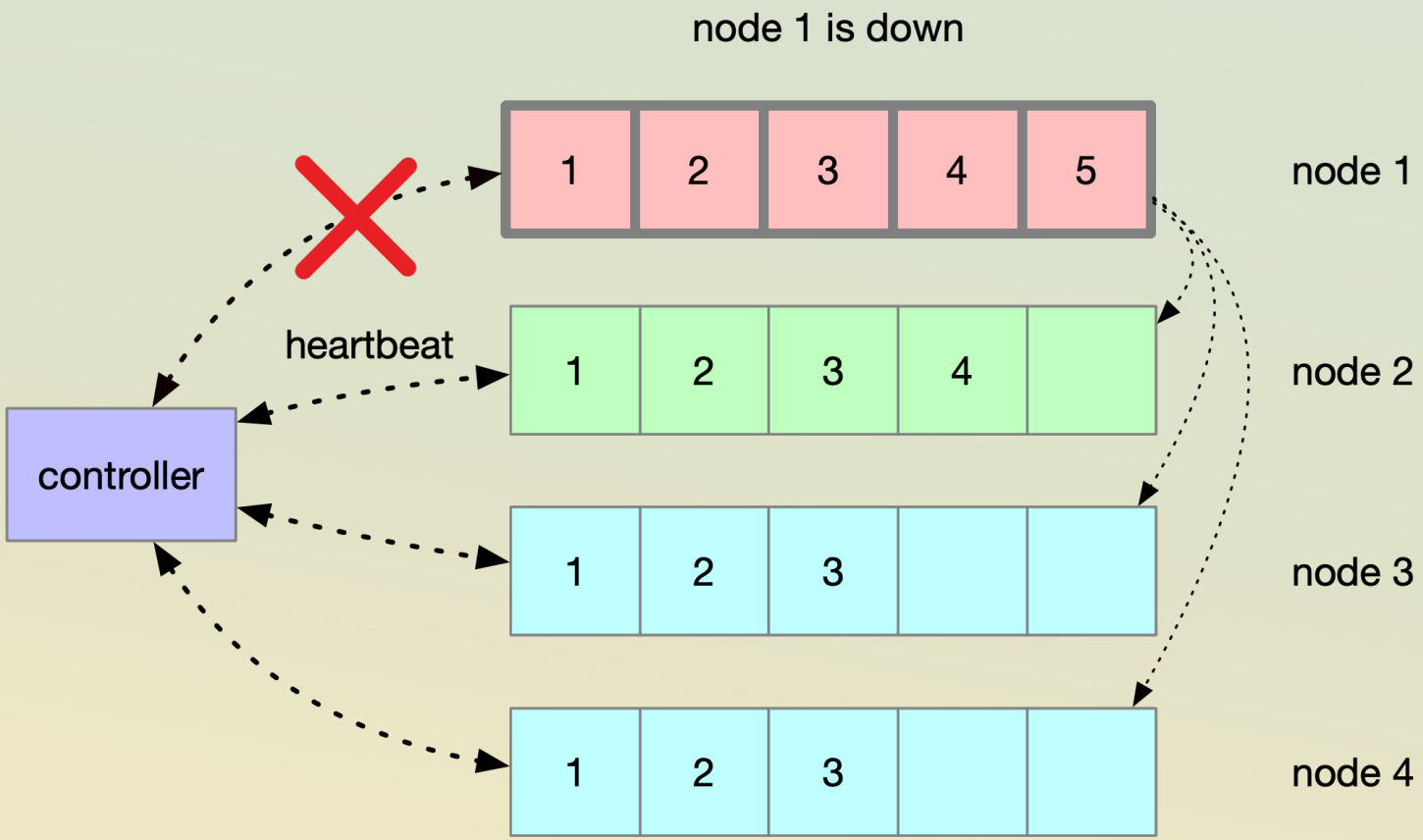
Gossip and Failure Detection

- P2P
- Outsourced
- Convergence

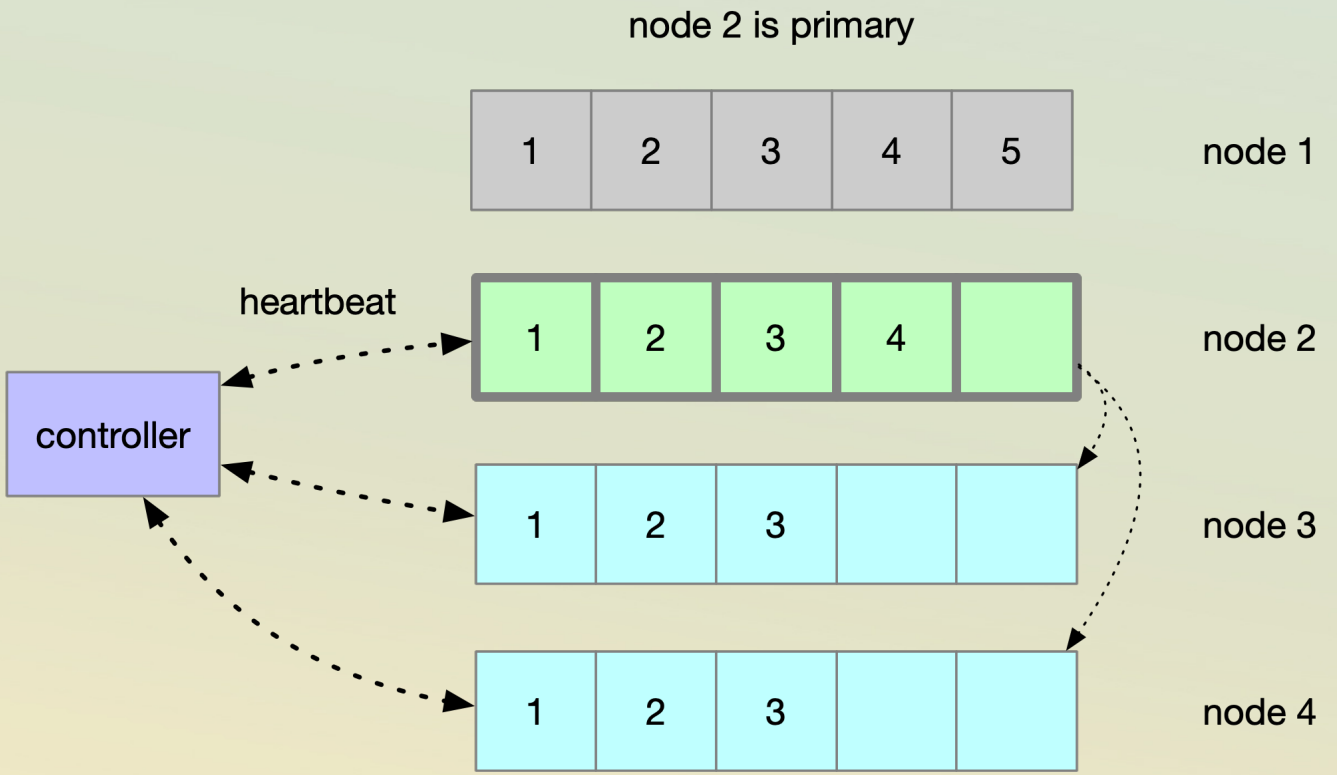
故障检测


SDK
Ping
xx is down
Timeout
Timeout
Suspect
Failure detector
Ping
Ping
Monitored node
Failure
Time

故障恢复


node 1 is down
1
2
3
4
5
node 1
heartbeat
1
2
3
4
node 2
controller
1
2
3
node 3
1
2
3
node 4

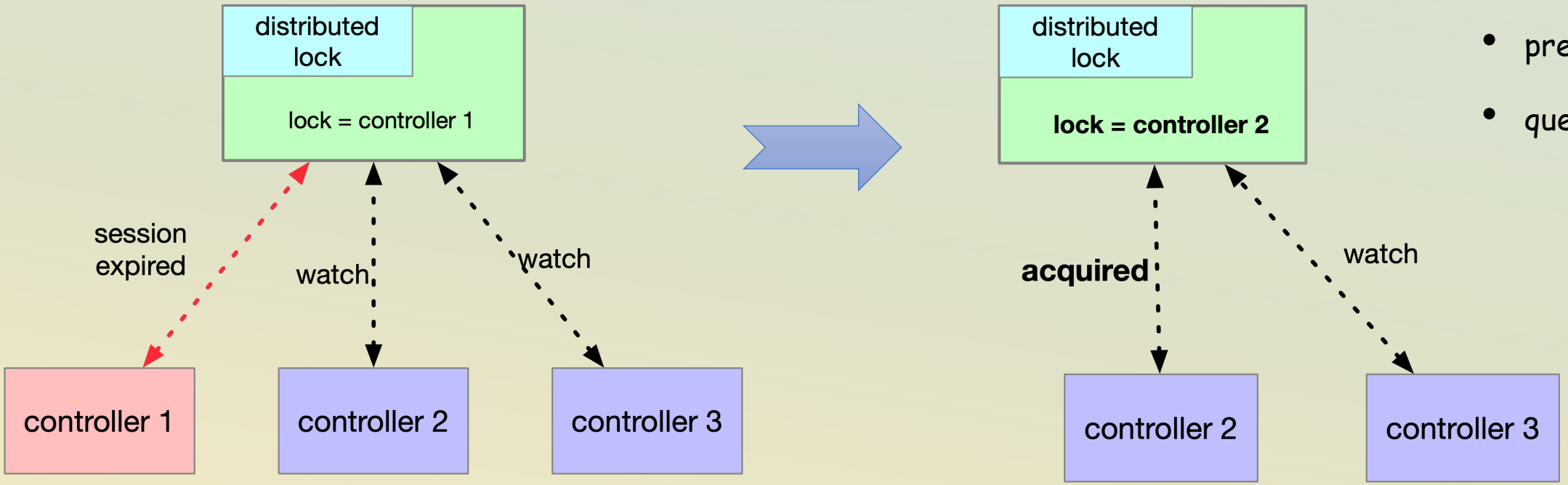
故障恢复



Stateful

- up-to-date

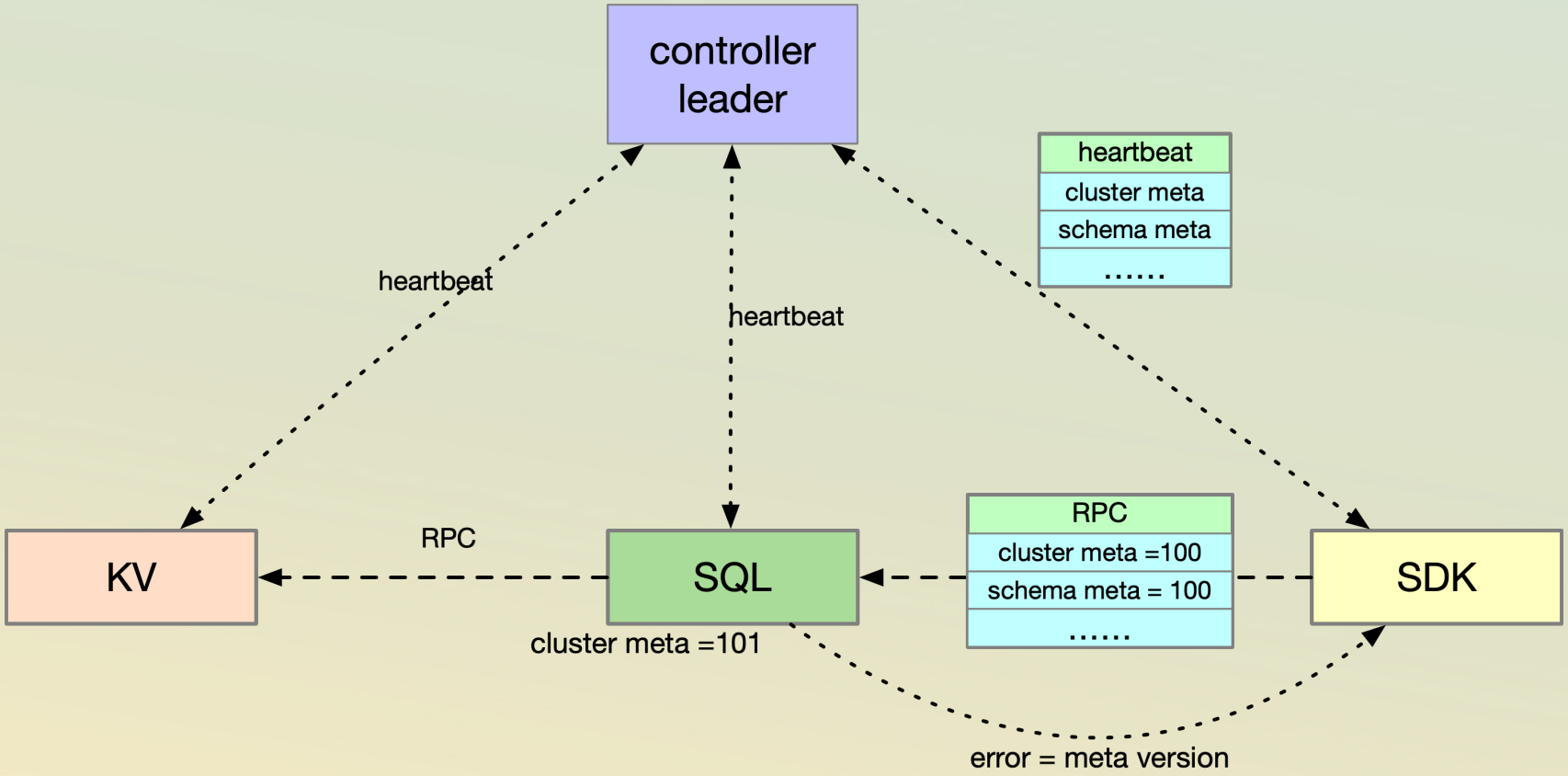
故障恢复



Stateless

- preemptive
- queue

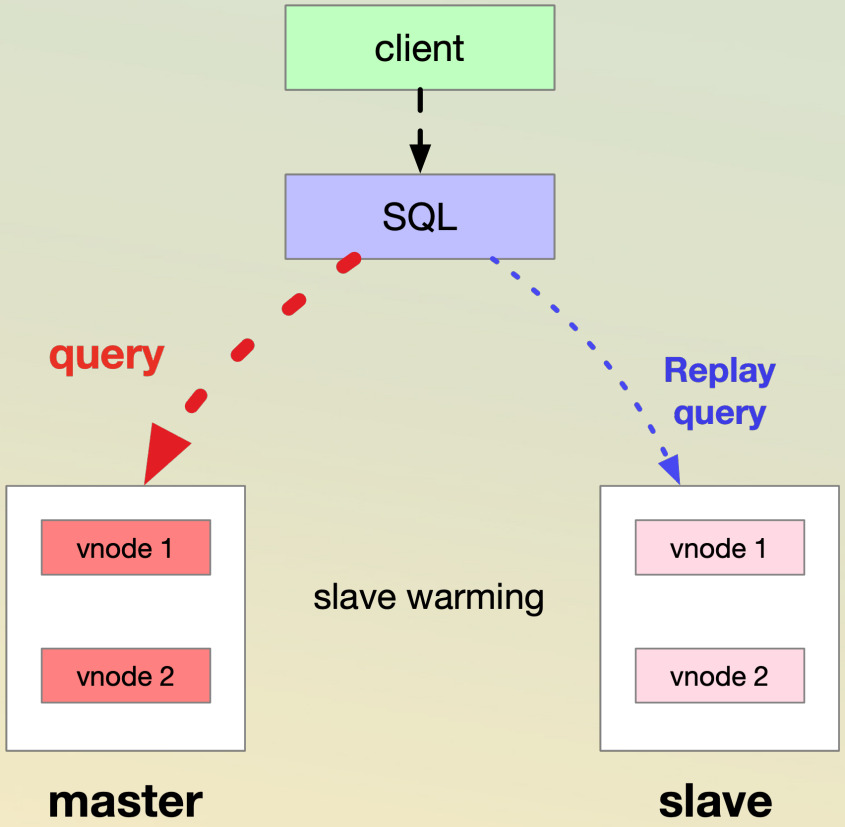
重新配置


controller
leader
heartbeat
cluster meta
schema meta
……
heartbeat
heartbeat
RPC
KV
SQL
RPC
cluster meta =100
schema meta = 100
……
SDK
cluster meta =101
error = meta version

故障恢复了?

cache miss

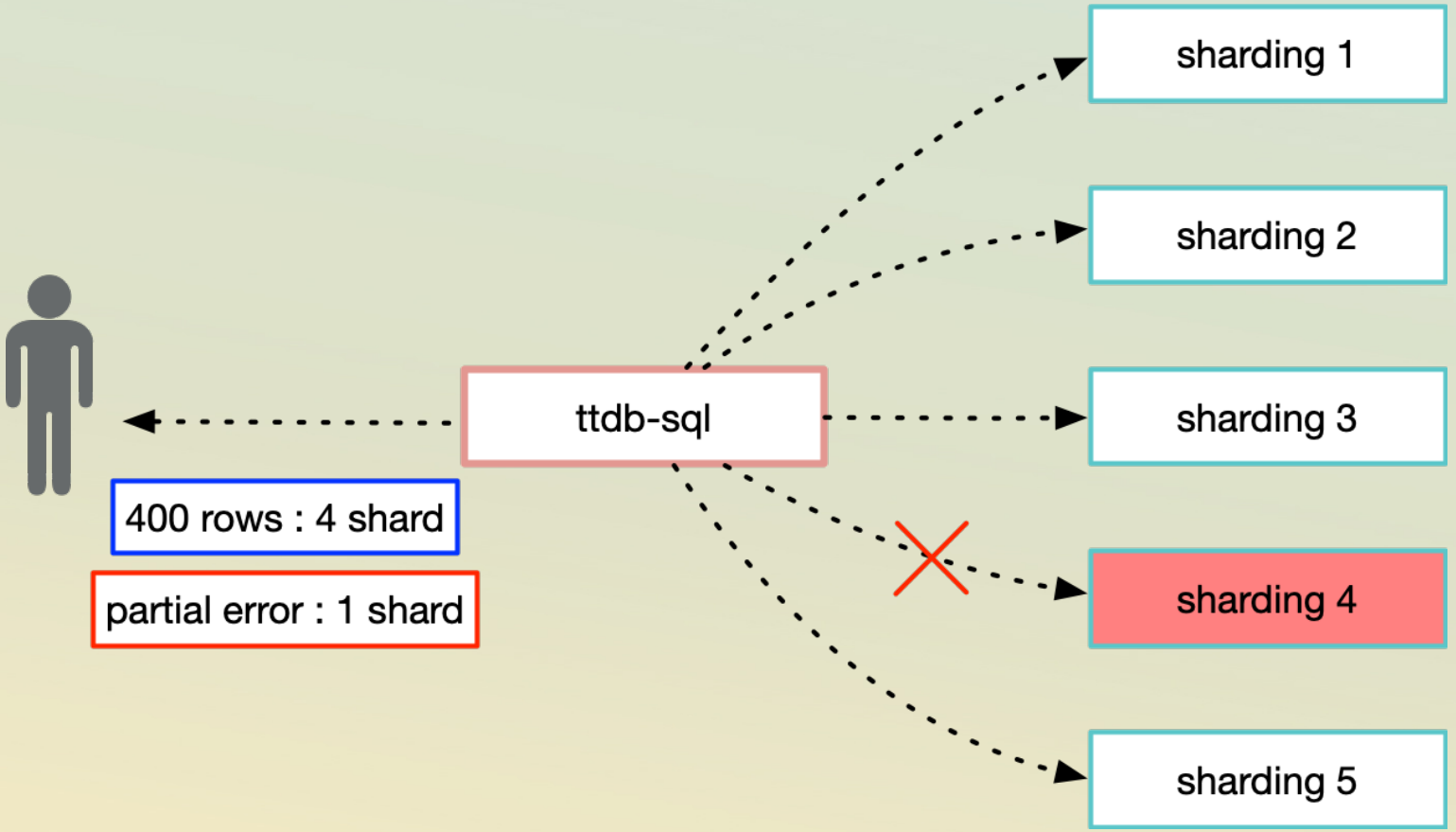
故障恢复了?



cache hit

结果与产出

一致性与可用性



业务应用

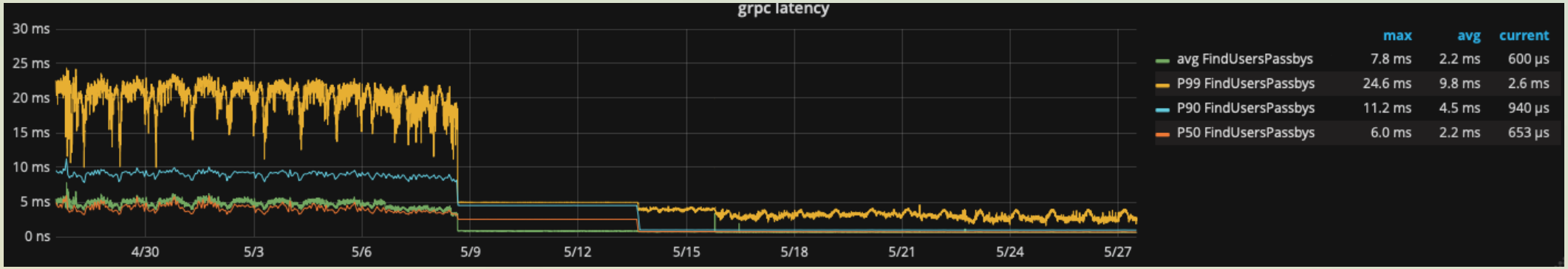
擦肩而过

user_id	other_user_id	location	time
user_A	user_B	142.654, 71.892	2020-01-25 11:00:00
user_A	user_C	142.617, 71.903	2020-01-25 11:05:00
user_A	user_D	142.598, 72.061	2020-01-25 11:07:00

```
ttdb> explain select user_id, other_user_id from location.passby where user_id=1 and other_user_id in (2,3,4) and count > 2;
plan                                type                 info
-------------------------------------------------------------------------------------------------------------------------
RootReader                          root
    Projection                      dist                 location.passby.user_id, location.passby.other_user_id
        Selection                   dist                 gt(location.passby.count, 2)
            TableScan               dist                 table:passby, range:[1 2,1 2], [1 3,1 3], [1 4,1 4]

ttdb>
```

擦肩而过



selection latency	P99	P90	Avg
ttdb	3.4ms	2.1ms	960us
cassandra	19ms	8.6ms	4ms

	1TB存储的数据行数
ttdb	41M
cassandra	7.8M

Golang

Why golang

- 学习成本
- 开发效率
- 标准库
- Goroutine
- GC

Goroutine 限制和泄漏

context, timeout, channel, select

- https://github.com/uber-go/goleak
- runtime.NumGoroutine()
- pprof/goroutine

GC 优化

减少对象分配

- Reuse
- Stack
- Preallocation

降低scan成本

- Pointers

Roadmap

Todo

- 开源 (2021)
- 物理优化
- 同步复制
- Range 分区
- 生态

GOPHER CHINA 2020

中国 上海 / 2020-11.21-22

Thanks