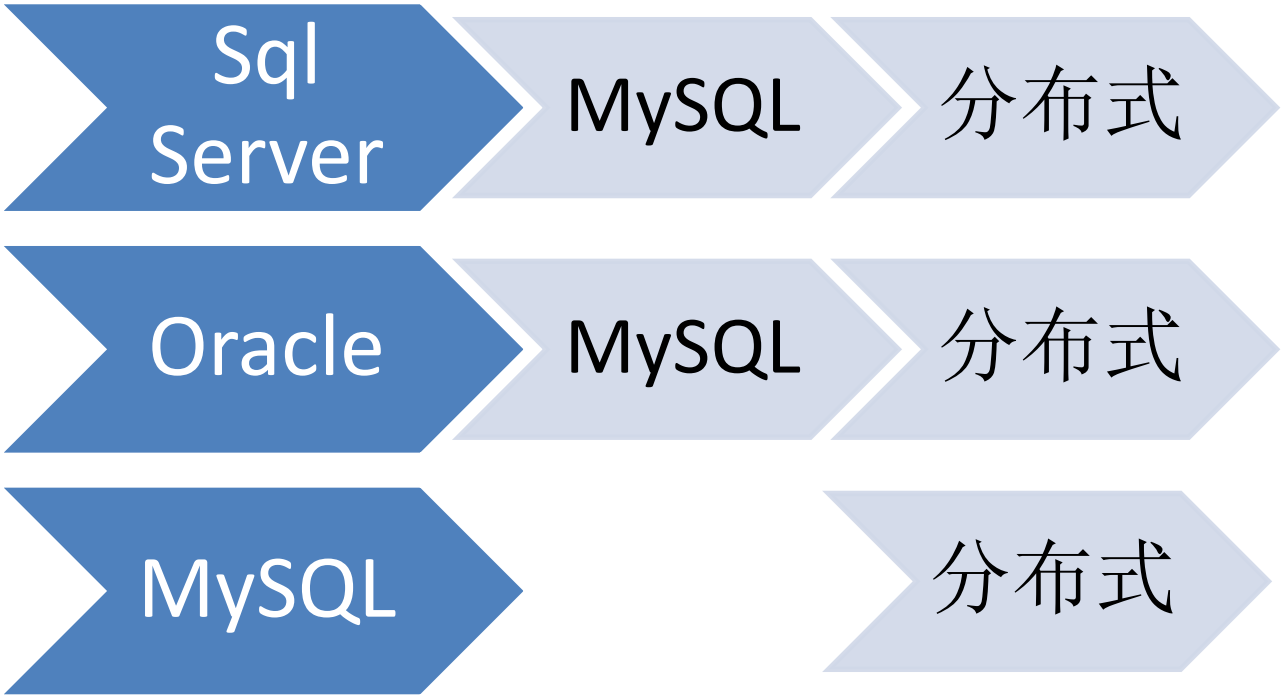
SDCC 2016
中国软件开发者大会
SOFTWARE DEVELOPER CONFERENCE CHINA

京东分布式数据库系统演进之路

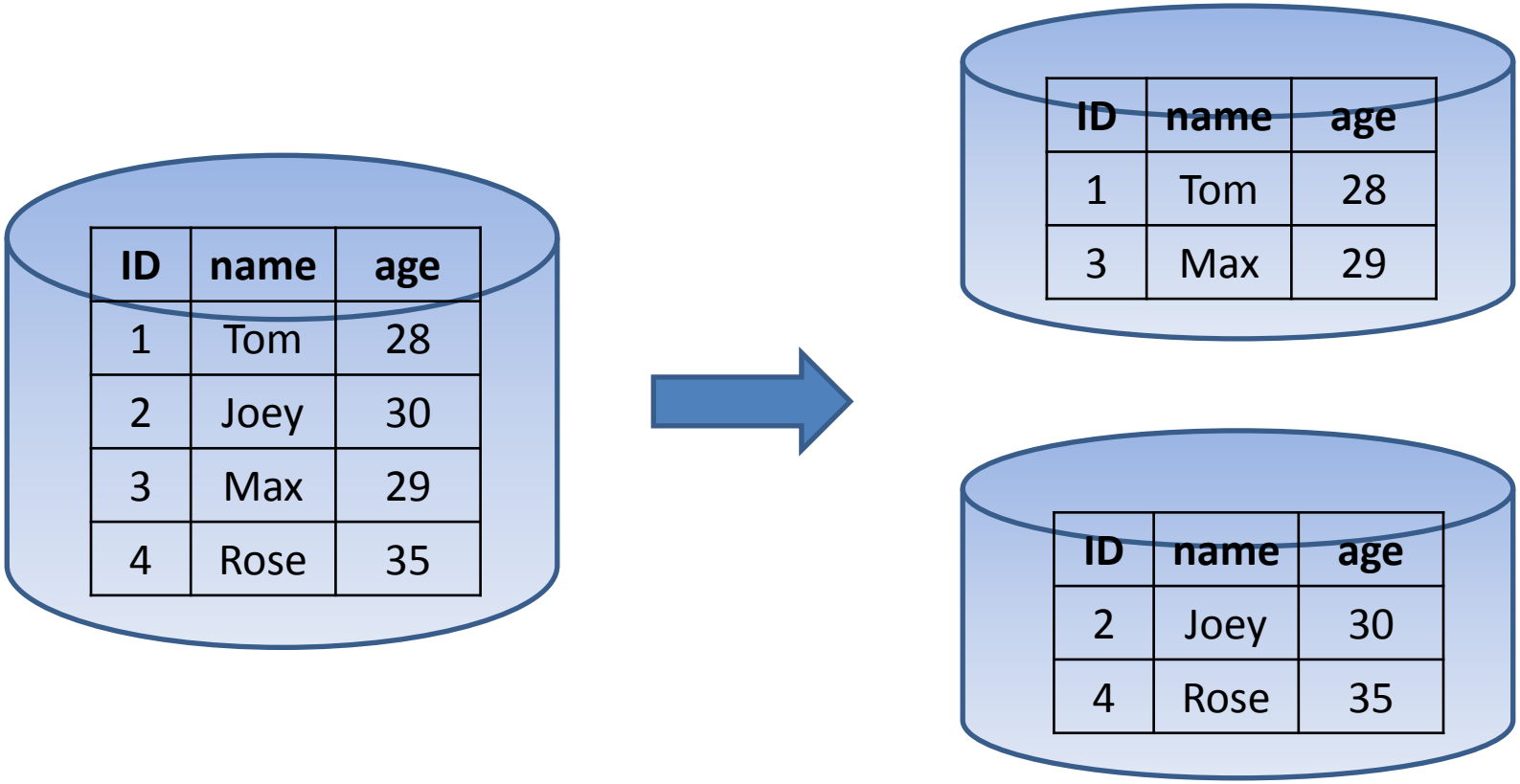
京东—张成远

- 目录
- 发展历程
- 实践经验
- 困难与挑战
- 小结

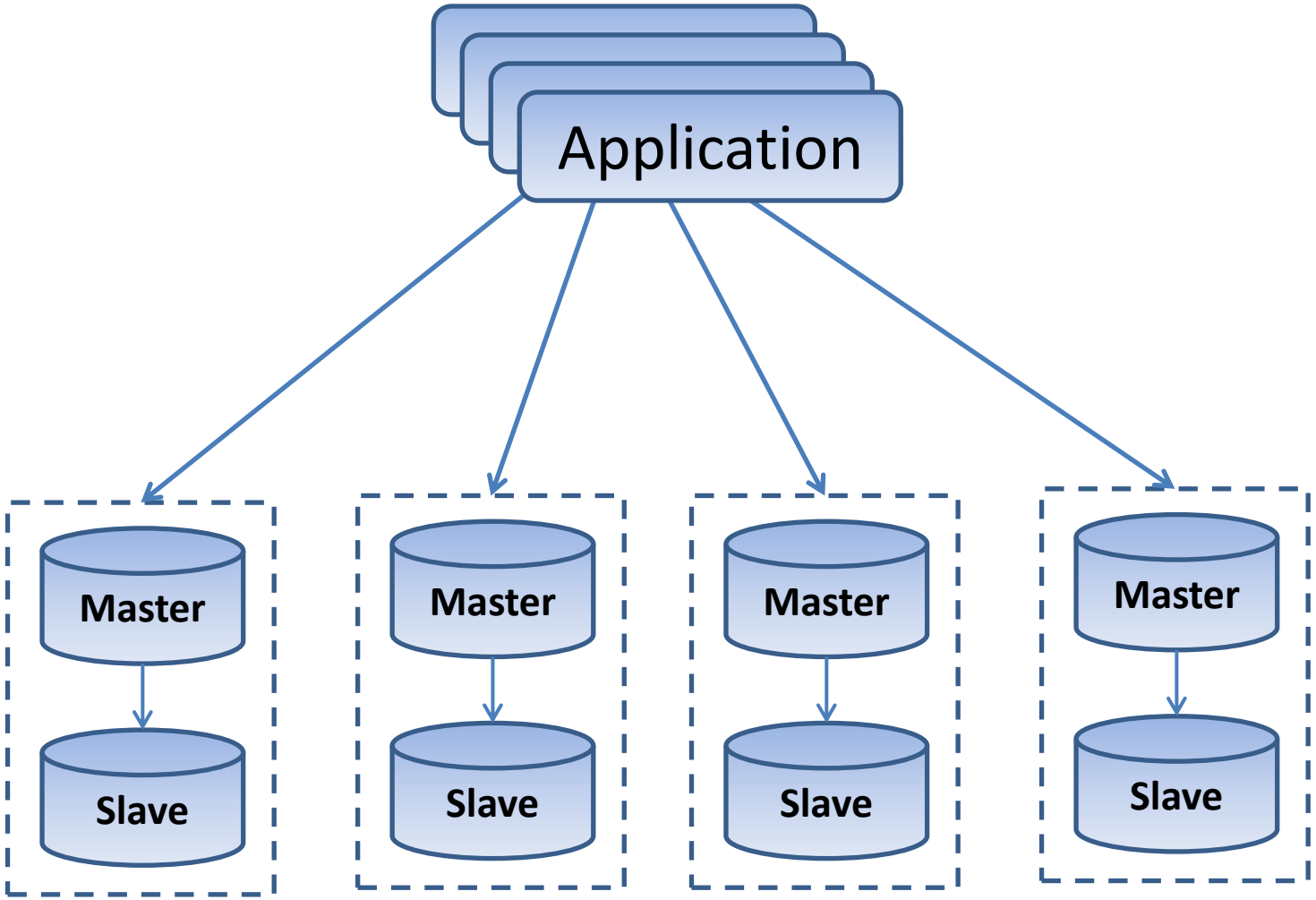
变化趋势



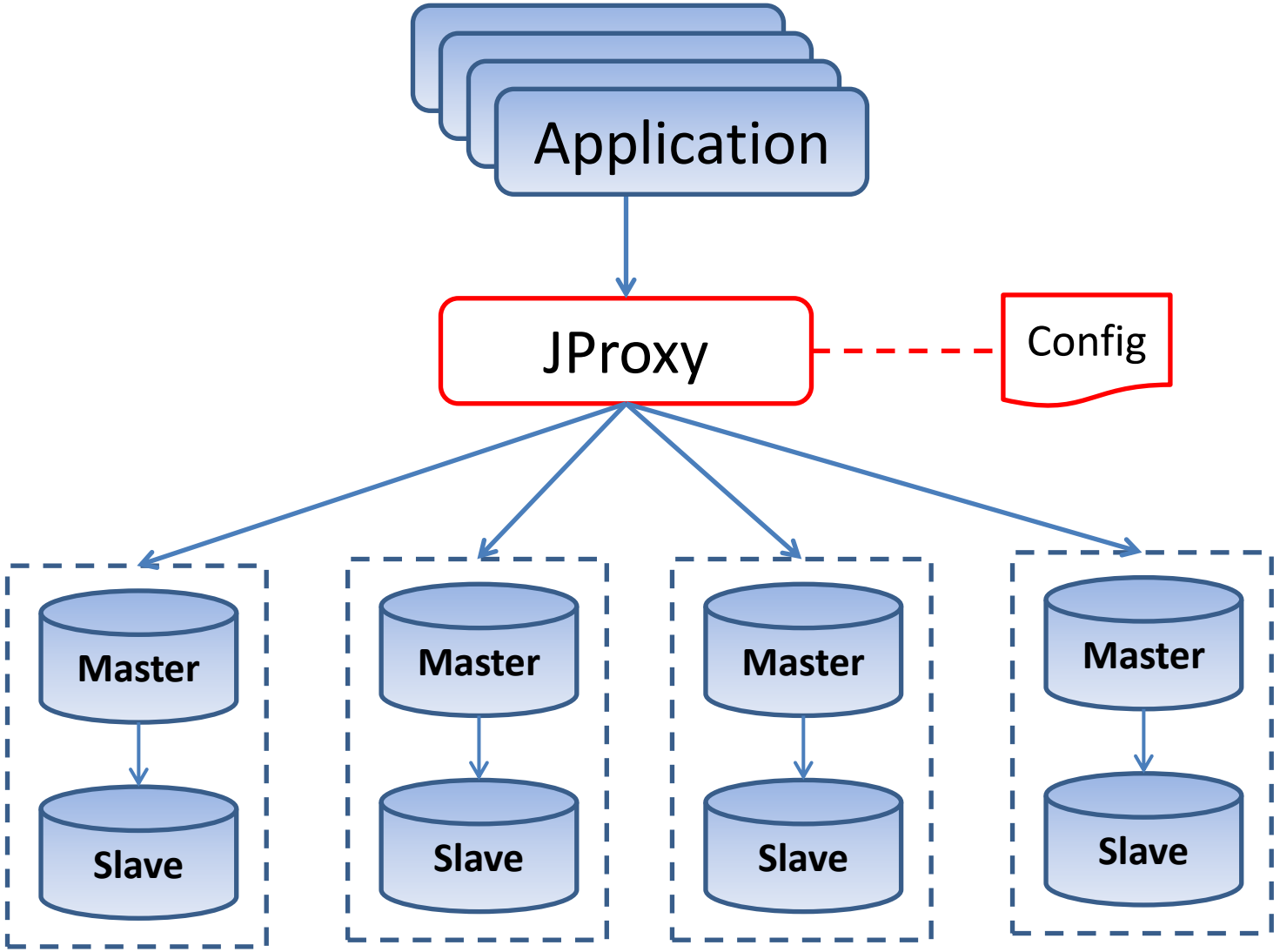
业务层面拆分


ID name age
1 Tom 28
2 Joey 30
3 Max 29
4 Rose 35
ID name age
1 Tom 28
3 Max 29
ID name age
2 Joey 30
4 Rose 35

业务层面拆分

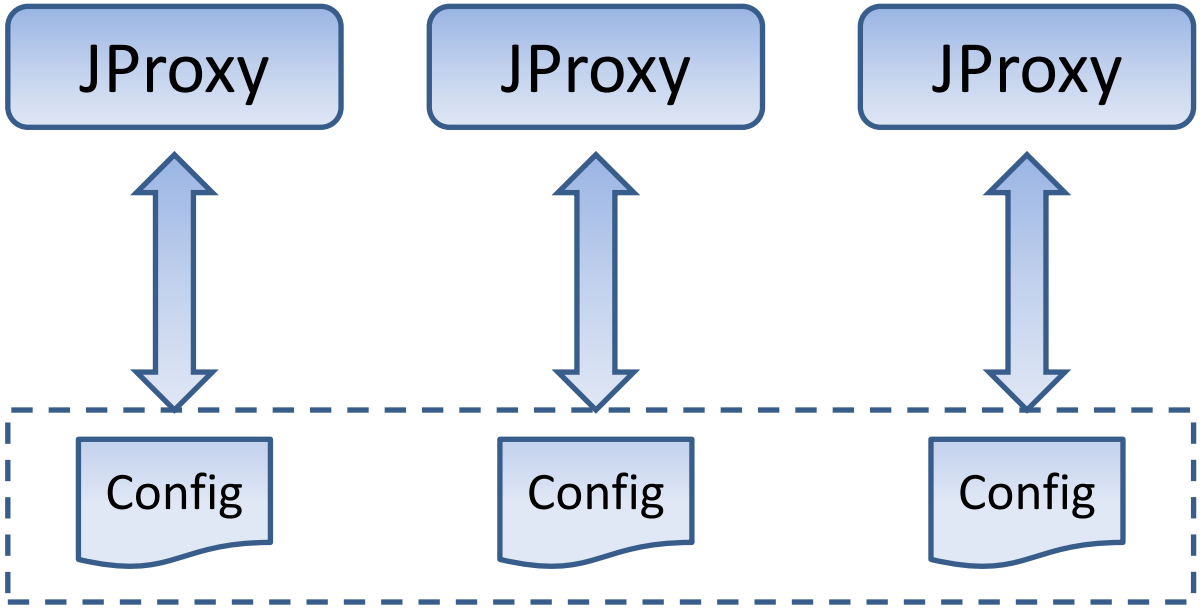

Application
Master
Slave
Master
Slave
Master
Slave
Master
Slave

提供统一的解决方案


Application
JProxy
Config
Master
Slave
Master
Slave
Master
Slave
Master
Slave

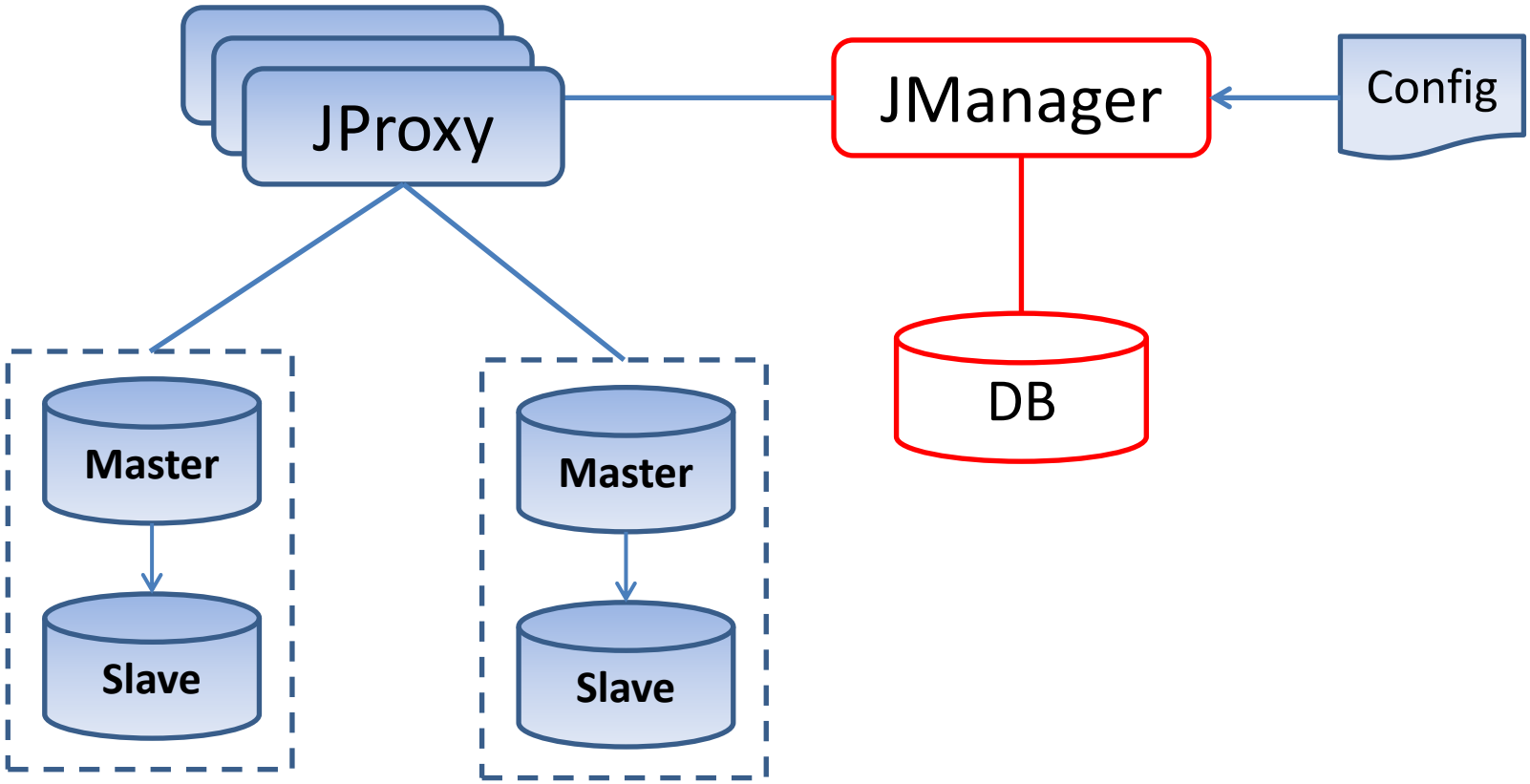
维护配置文件之痛

多份配置文件一致性如何保证？

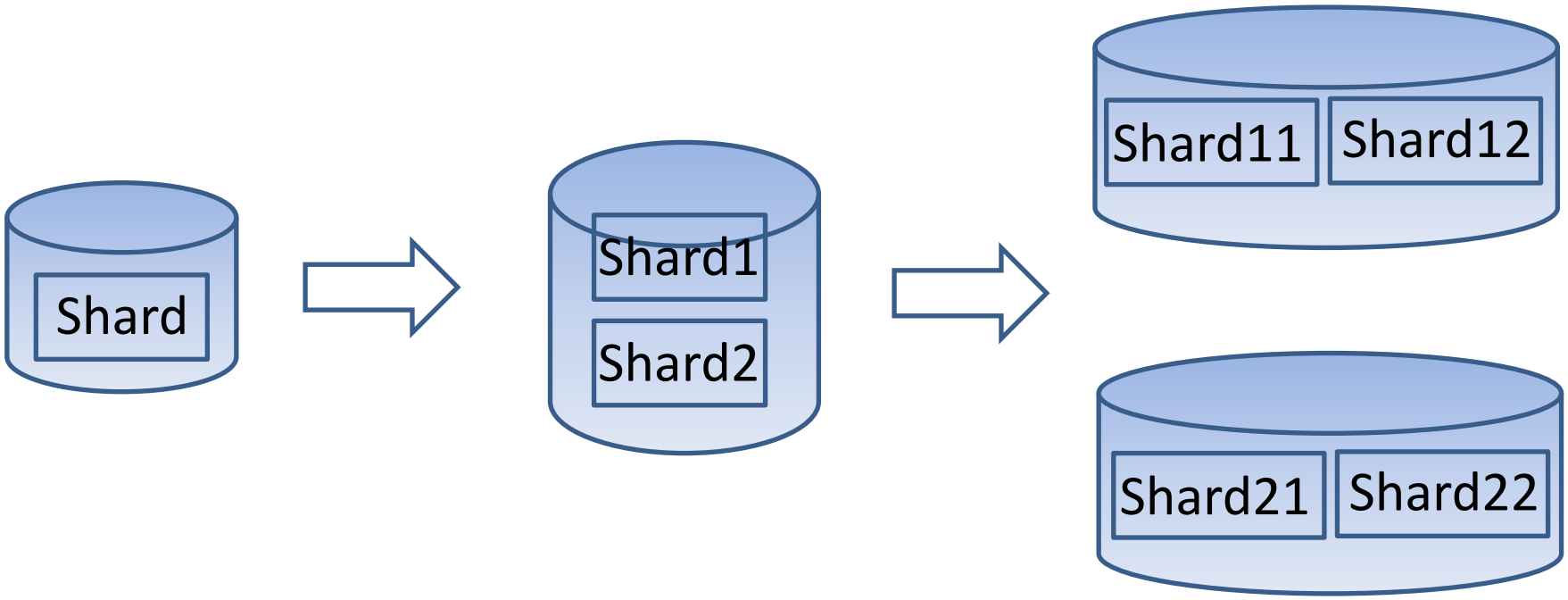


维护配置文件之痛

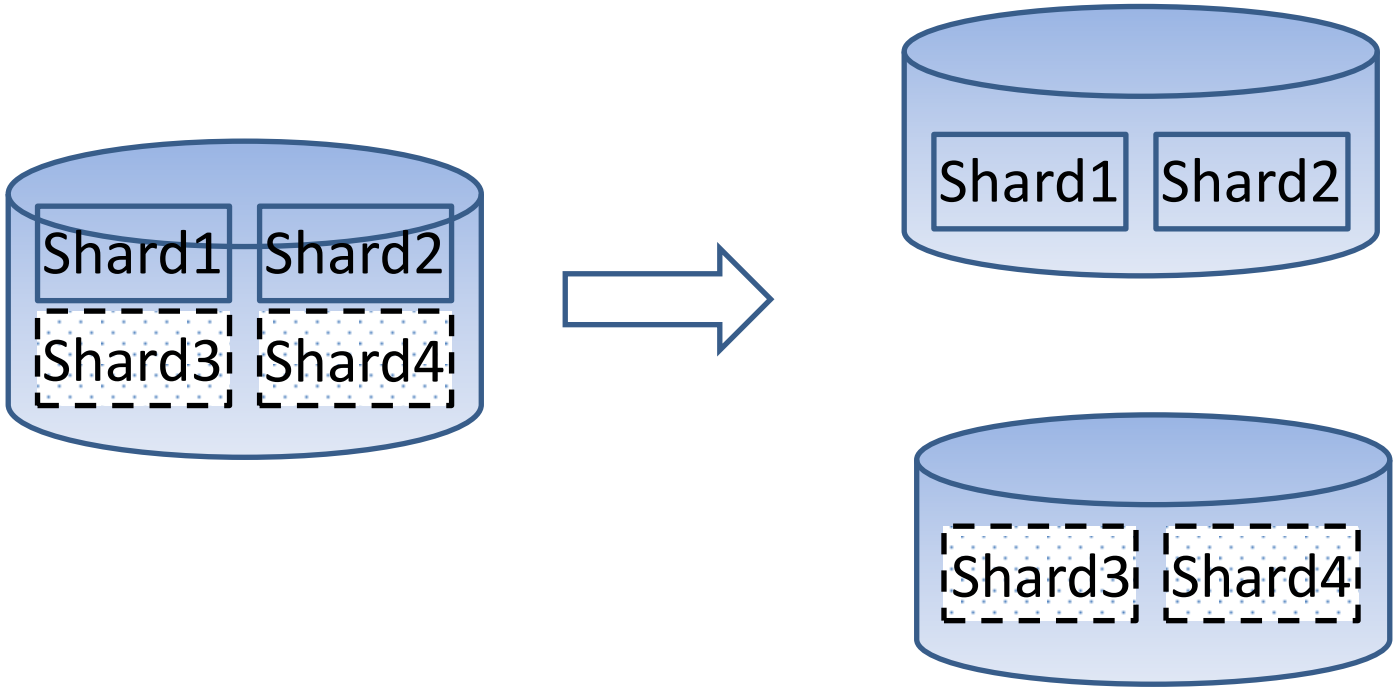
配置文件中心化存储



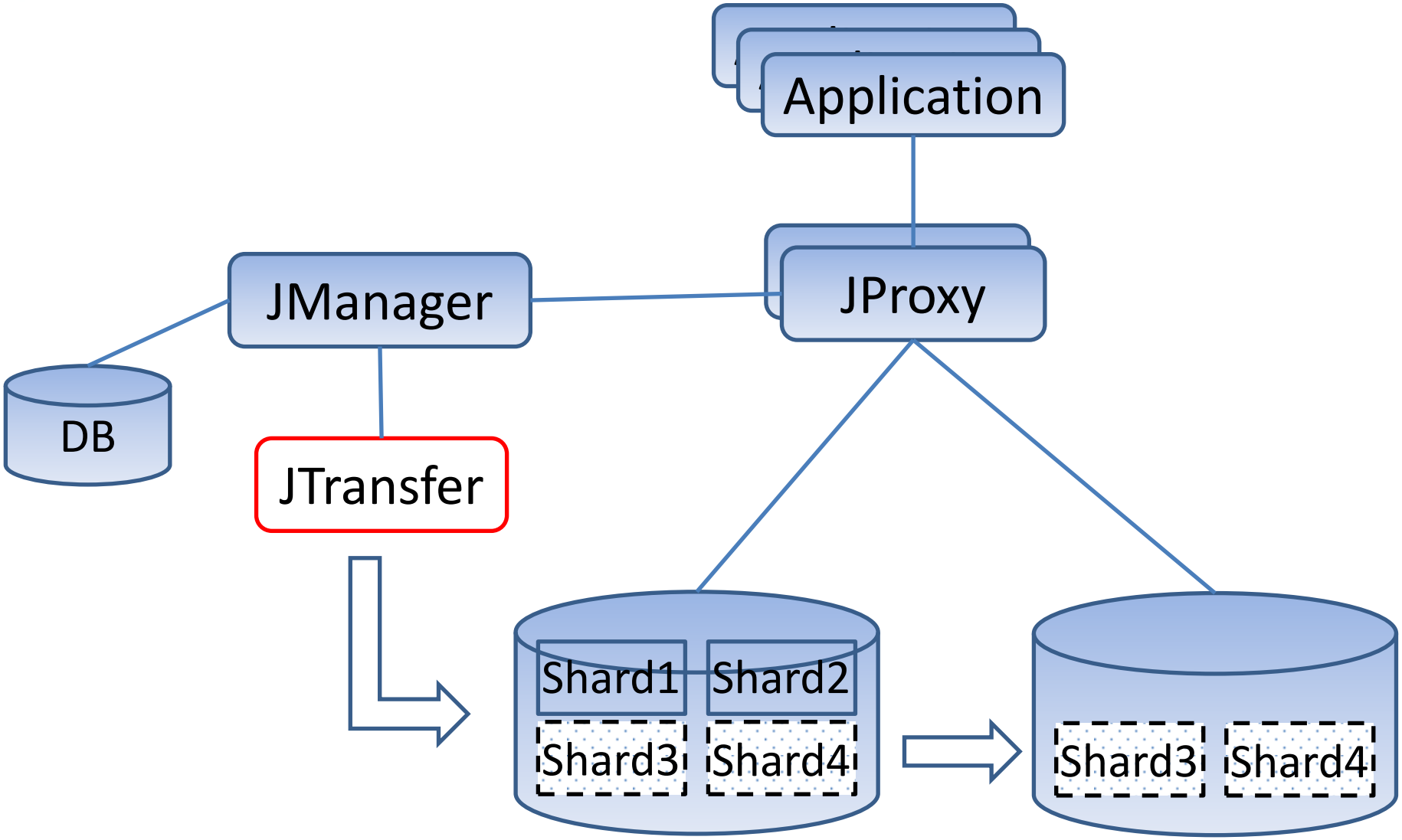
扩容方案— re-sharding



扩容方案— pre-sharding



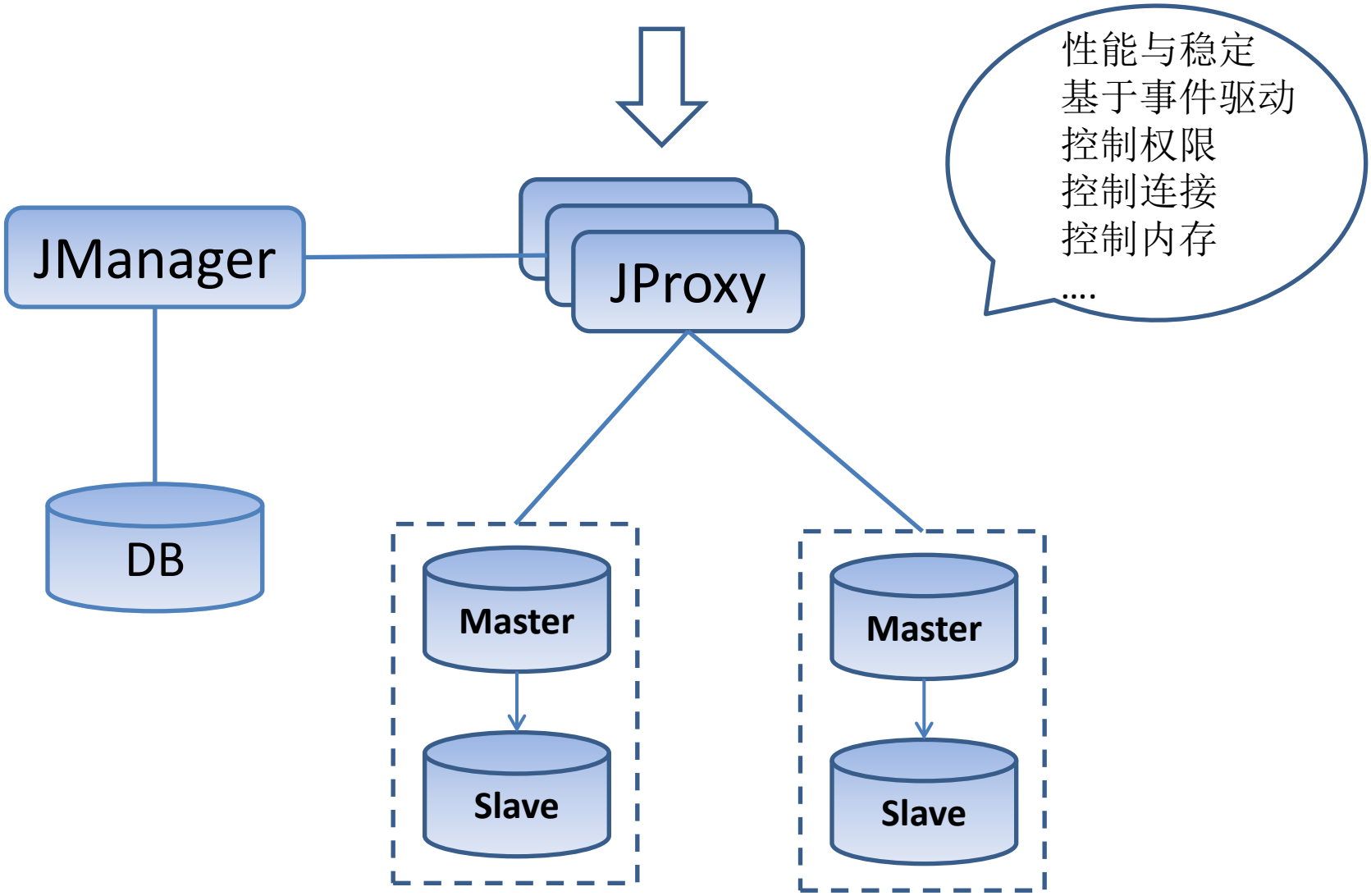
在线扩容


Application
JProxy
JManager
DB
JTransfer
Shard1
Shard2
Shard3
Shard4
Shard3
Shard4

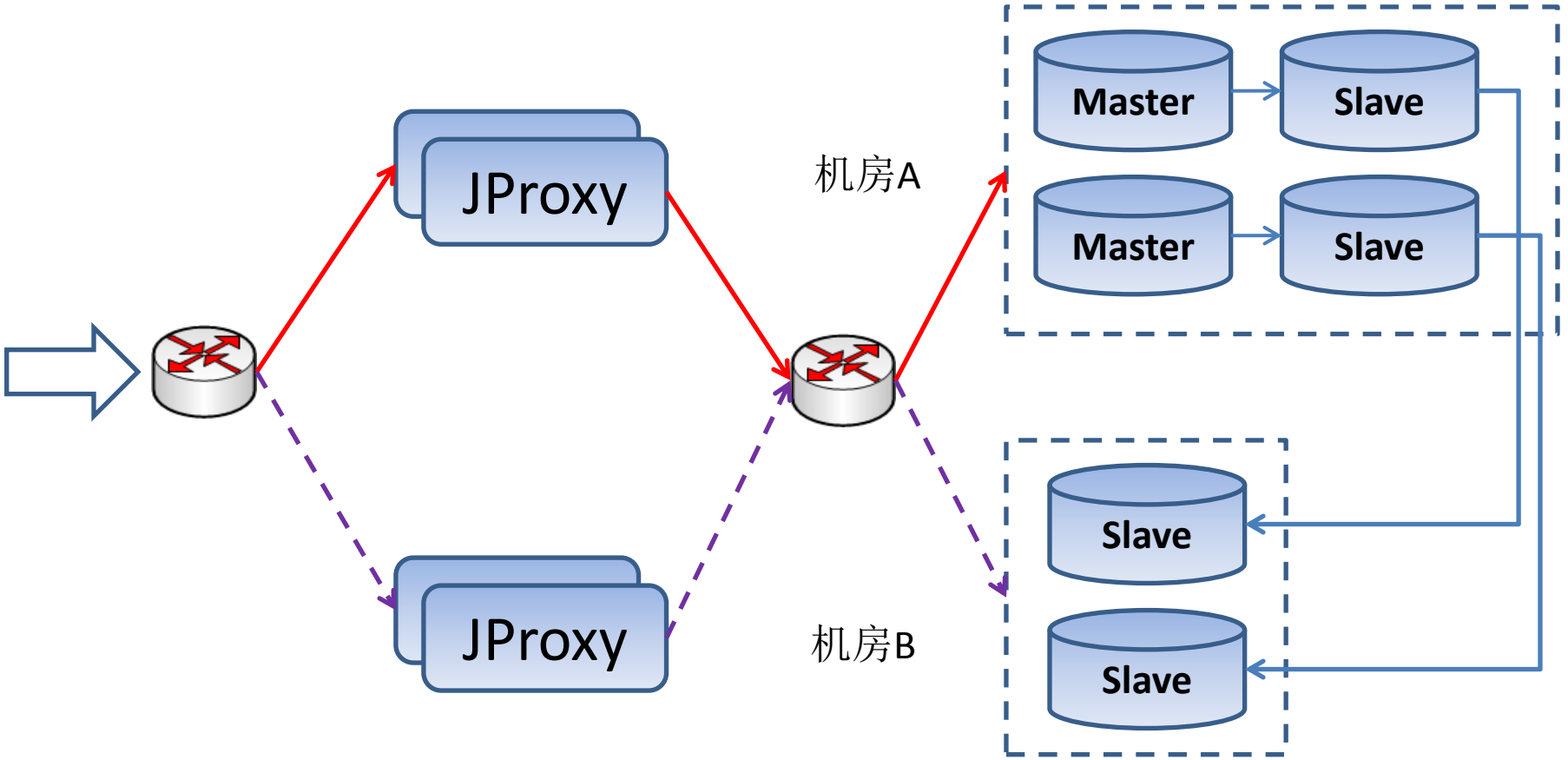
实践经验

- 活下来
- 活得更好
- 经验小结

活下来——性能与稳定

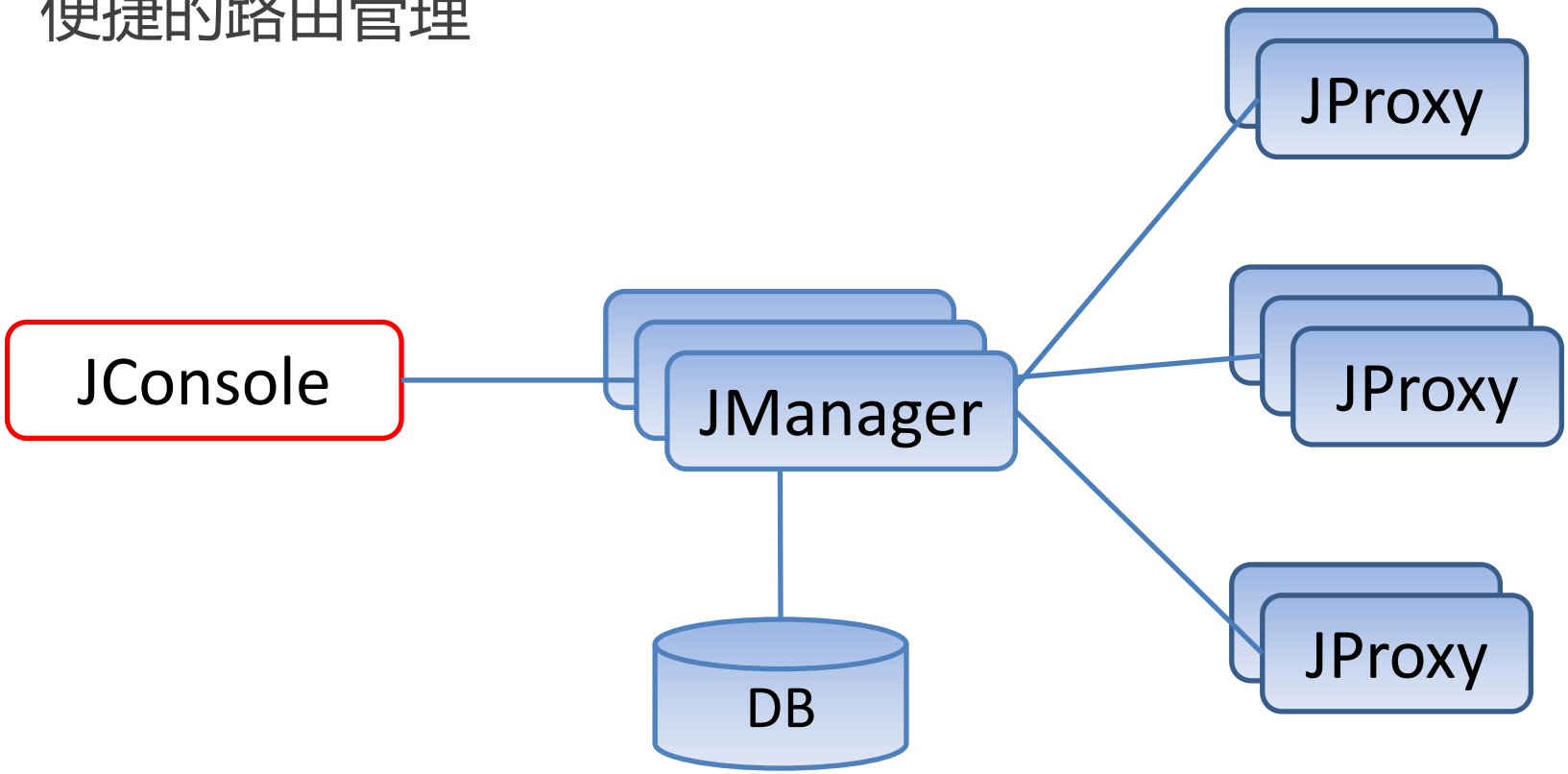

性能与稳定
基于事件驱动
控制权限
控制连接
控制内存
....
JManager
JProxy
DB
Master
Slave
Master
Slave

活下来—高可用/高可靠方案

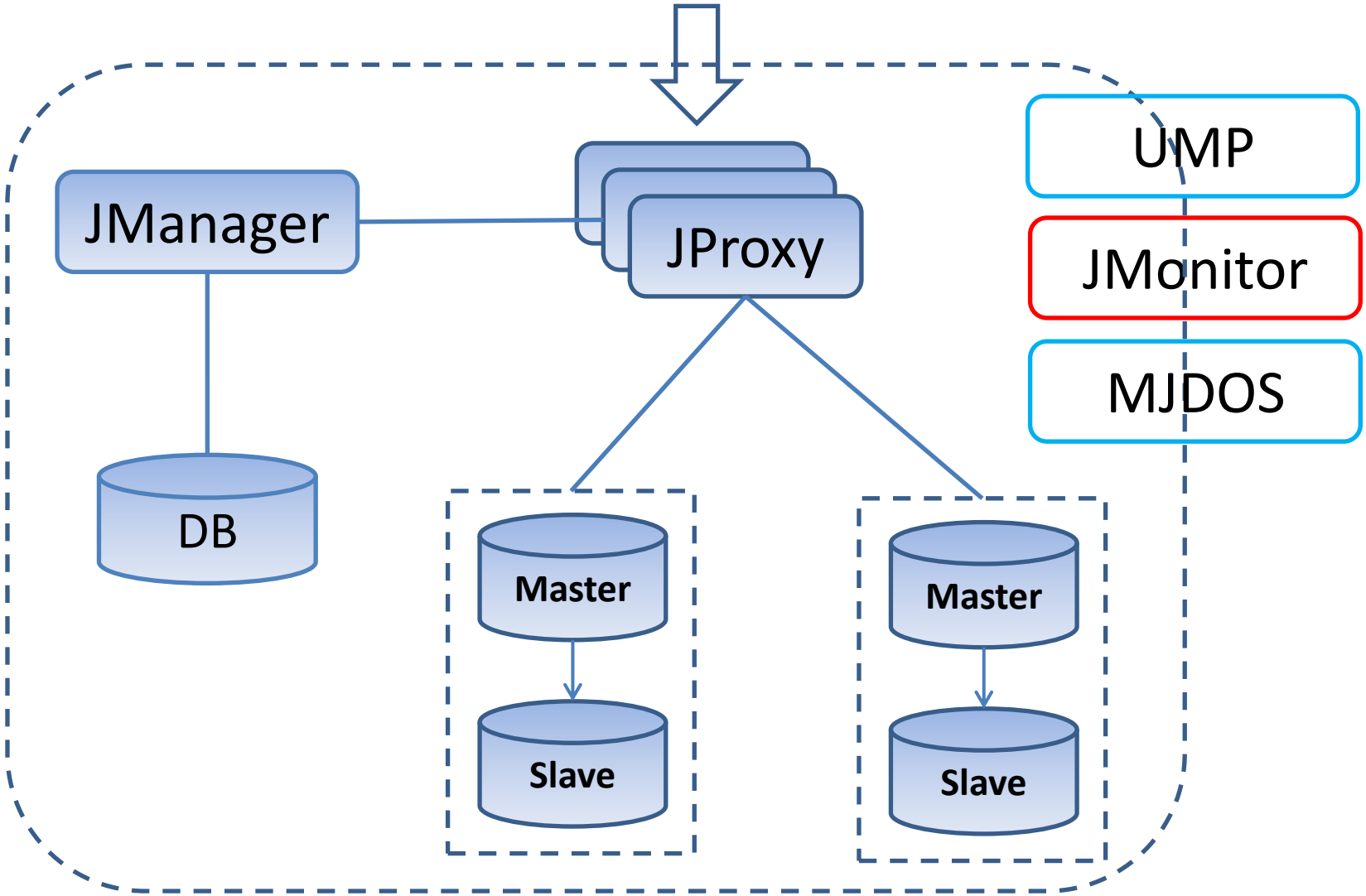


活得更好——路由管理

- 完善的路由检查机制
- 便捷的路由管理



活得更好——监控系统


JManager
JProxy
DB
Master
Slave
Master
Slave
UMP
JMonitor
MJDOS

活得更好—监控系统

- 精细化记录慢SQL
- 支持慢SQL分析功能
- 跨库SQL/事务的监控
- 数据分布情况
- JProxy/MySQL连接数
- 网络指标
- 内存/cpu/磁盘/机器负载等

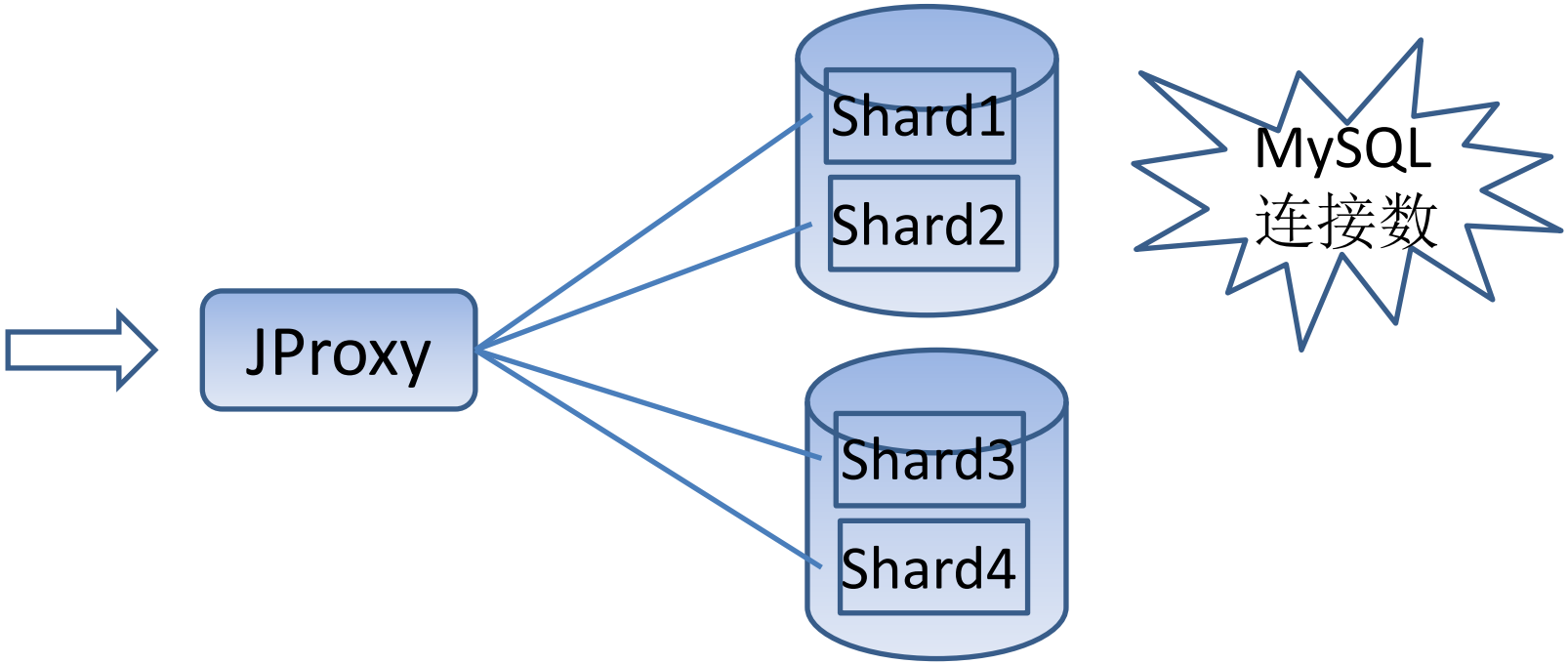
实践经验

- 避免跨库的SQL

select * from table

select * from table where id in(x, y, z)

select * from table where id=x or id=y



实践经验

- 避免跨库的事务

```
begin;
select * from table where id = x; //shard1
select * from table where id = y; //shard2
update table set price=xx where id=z; //shard3
commit;
```

- 每条SQL不跨库，事务依然可能跨库
- 大面积死锁风险（短期内未必马上显现）
- MySQL连接数依然会压力比较大

实践经验

- 特殊业务特殊对待

  - 全国分拣中心数量有限
  - 各个分拣中心数据量有较大出入
  - 普通拆分方式可能导致分布不均匀

  - 开发路由，支持特殊拆分，大的分拣中心可以落在指定MySQL实例上，确保压力分散均匀

实践经验

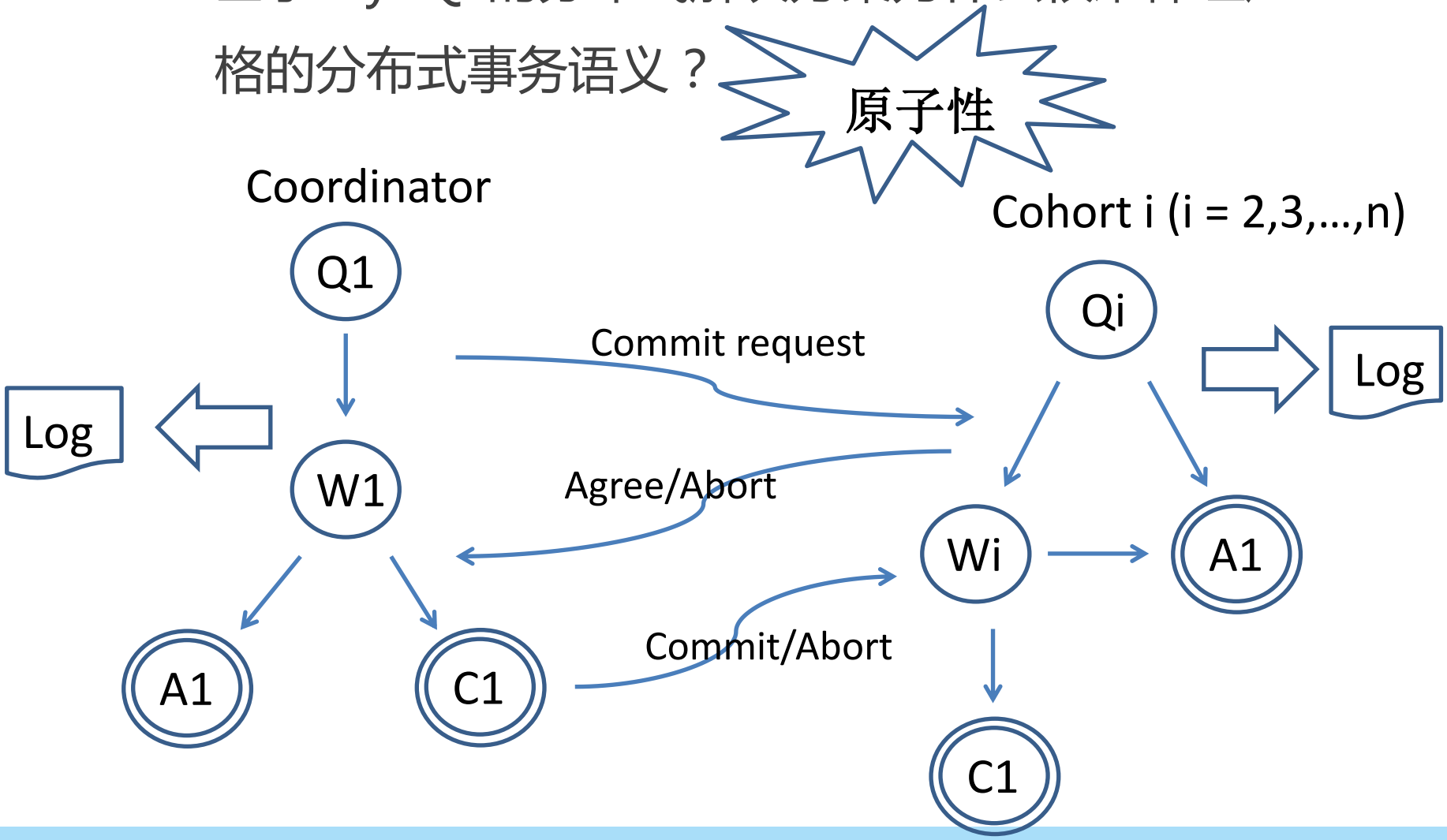
- **系统尽量独享**
  - 关键业务可以接受部分的资源浪费，确保业务稳定是第一要素
  - 避免重要业务彼此影响产生的连锁效应
- **使用容器提高资源利用率**
  - 部署层面可以解决的问题，没必要在系统层面去解决
  - 确保系统足够健壮可控比引入很多高大上的功能来得靠谱

实践经验

- 选择合适的拆分字段/拆分方式
- 重写SQL，尽量带拆分字段
- 避免跨库的事务
- 每条SQL都有拆分字段无法保证避免跨库事务
- 跨库事务最痛苦的事情是死锁，大面积死锁
- 特殊业务特殊需求特殊对待
- 容器+物理机提高资源利用率，可以在部署层面搞定的事情没必要在系统层面去解决

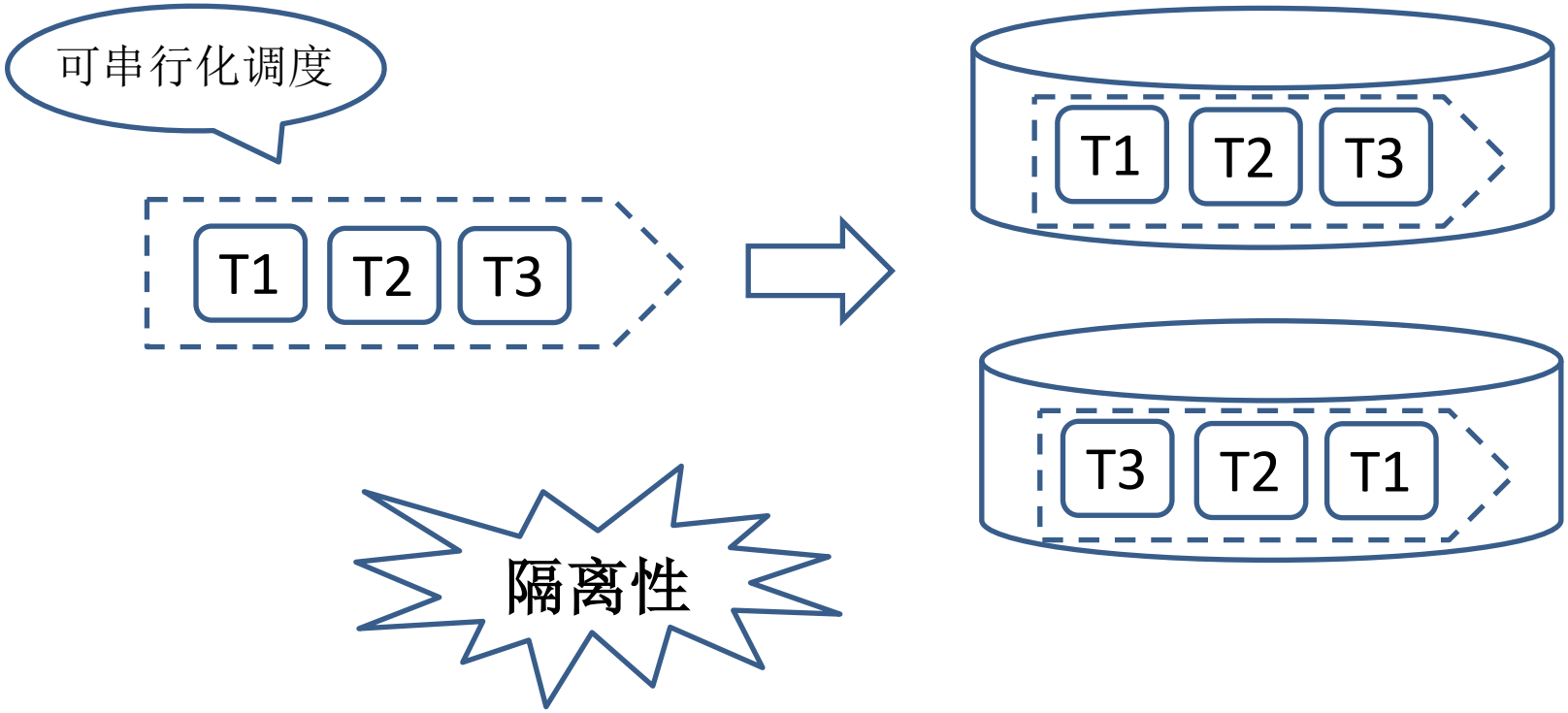
困难与挑战

- 基于MySQL的分布式解决方案为什么很难保证严格的分布式事务语义？



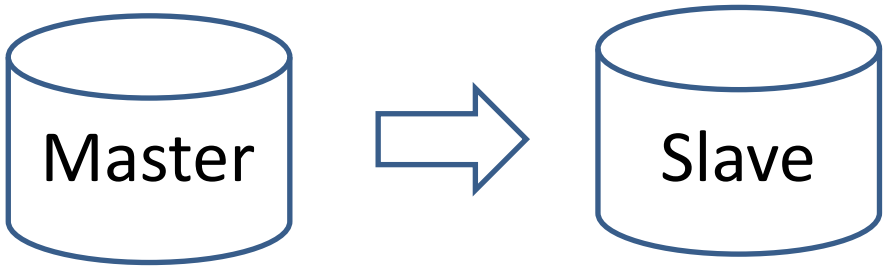
困难与挑战

- 基于MySQL的分布式解决方案为什么很难保证严格的分布式事务语义？



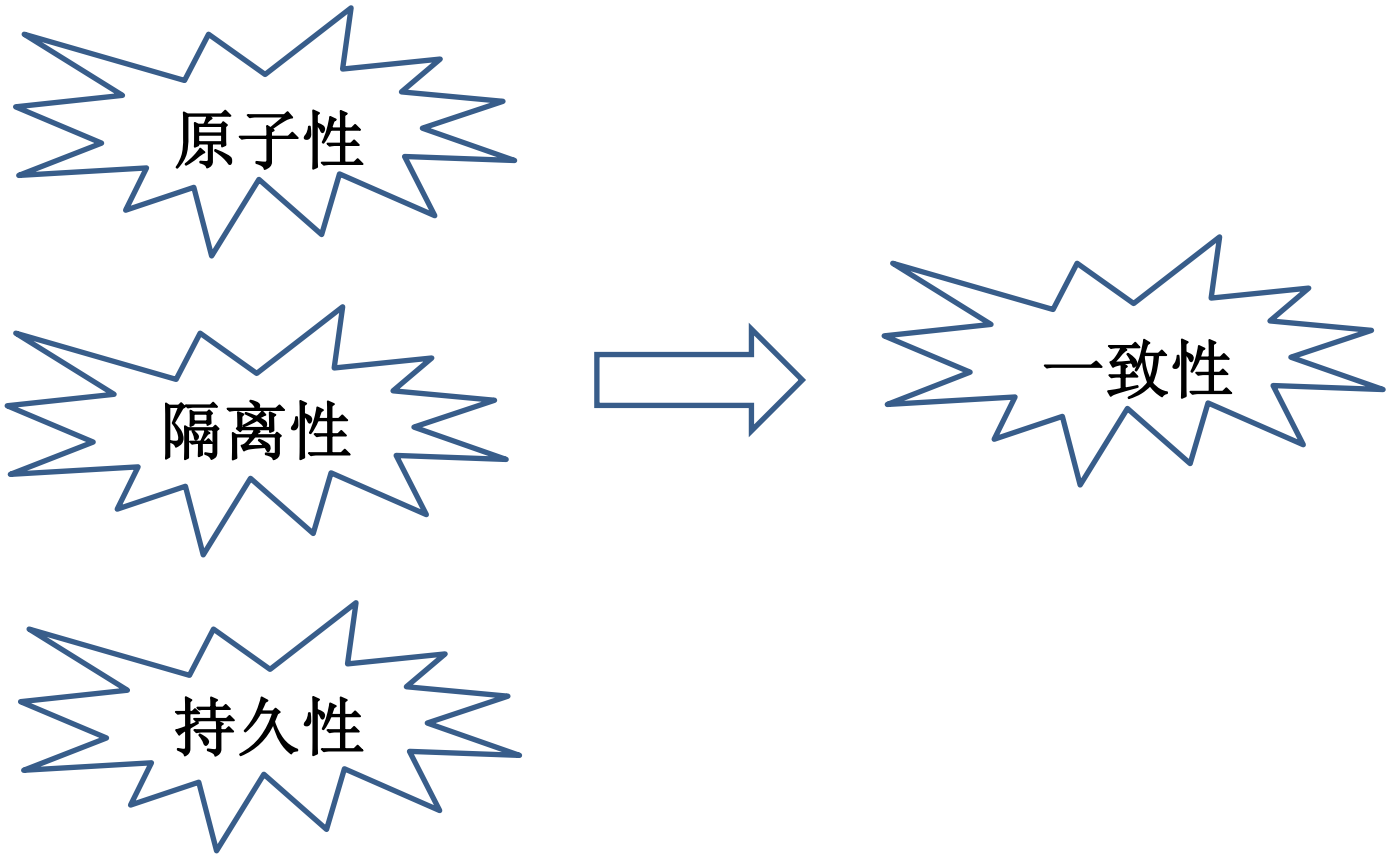
困难与挑战

- 基于MySQL的分布式解决方案为什么很难保证严格的分布式事务语义？



困难与挑战

- 基于MySQL的分布式解决方案为什么很难保证严格的分布式事务语义？



困难与挑战—解决思路

- 协调者引入两阶段提交，保证原子性
- 引入全局事务ID，保证隔离性
- 考虑采用Raft等强一致协议保证多副本强一致，从而保证事务的持久性
- 前三者满足以后，自然就保证了事务的一致性

小结

- 配置文件中心化管理
- 自动化迁移完成扩容
- 要有良好的自我保护功能
- 生产环境监控非常关键
- 要有高可用/高可靠的解决方案
- 分布式事务（本质上是大事务）容易引起大规模死锁
- 基于MySQL的解决方案很难保证严格的分布式事务语义

SDCC 2016

中国软件开发者大会

SOFTWARE DEVELOPER CONFERENCE CHINA

谢 谢！