数据库引擎技术架构

李海翔 @那海蓝蓝

PostgreSQL, MySQL, Greenplum, Informix, etc

@那海蓝蓝 Blog : http://blog.163.com/li_hx/

《数据库查询优化器的艺术: 原理解析与SQL性能优化》

10月27日 下午 专场4：数据库平台架构及变迁（上）			
时间	主持人	李海翔	华胜信泰数据库架构师
13:30-14:20	数据库引擎技术架构	李海翔	华胜信泰数据库架构师
14:20-15:10	数据之大，云动未来——传统企业从IT到DT的互联网创新最佳实践	王伟	爱可生解决方案总监
15:10-16:00	腾讯云数据库CDB技术演进之路	程彬	腾讯基础架构部数据库研发负责人
16:00-16:20	茶歇，展示		
16:20-17:10	图数据库Neo4J的实践之路	魏佳	Linkedin China Engineer Supervisor
17:10-18:00	企业级云数据库管控架构设计与实践	宁海元	袋鼠云CTO

SQL执行过程

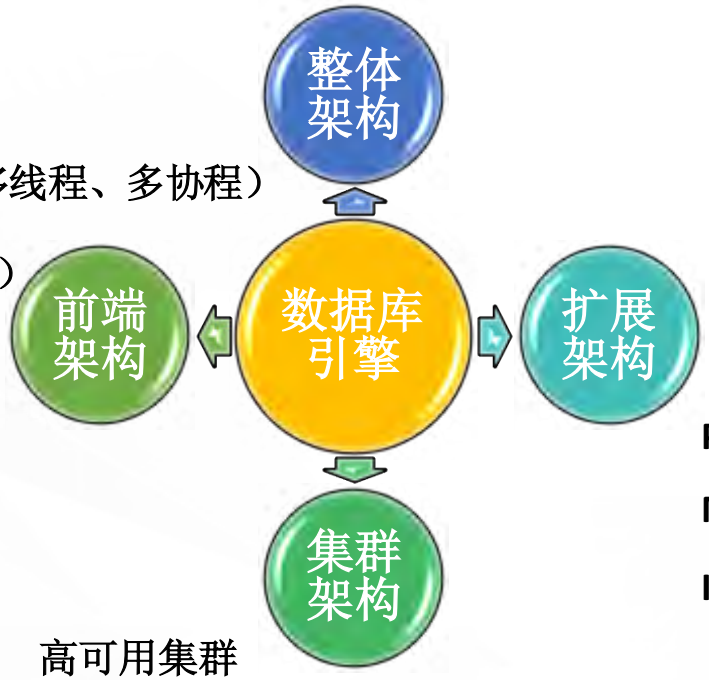
事务管理与并发控制、日志

数据存储

连接（APP、工具、接口）与协议

面向连接的服务器架构（多进程、多线程、多协程）

安全防范（用户鉴别、防止DoS攻击）



PostgreSQL的扩展

MySQL的扩展

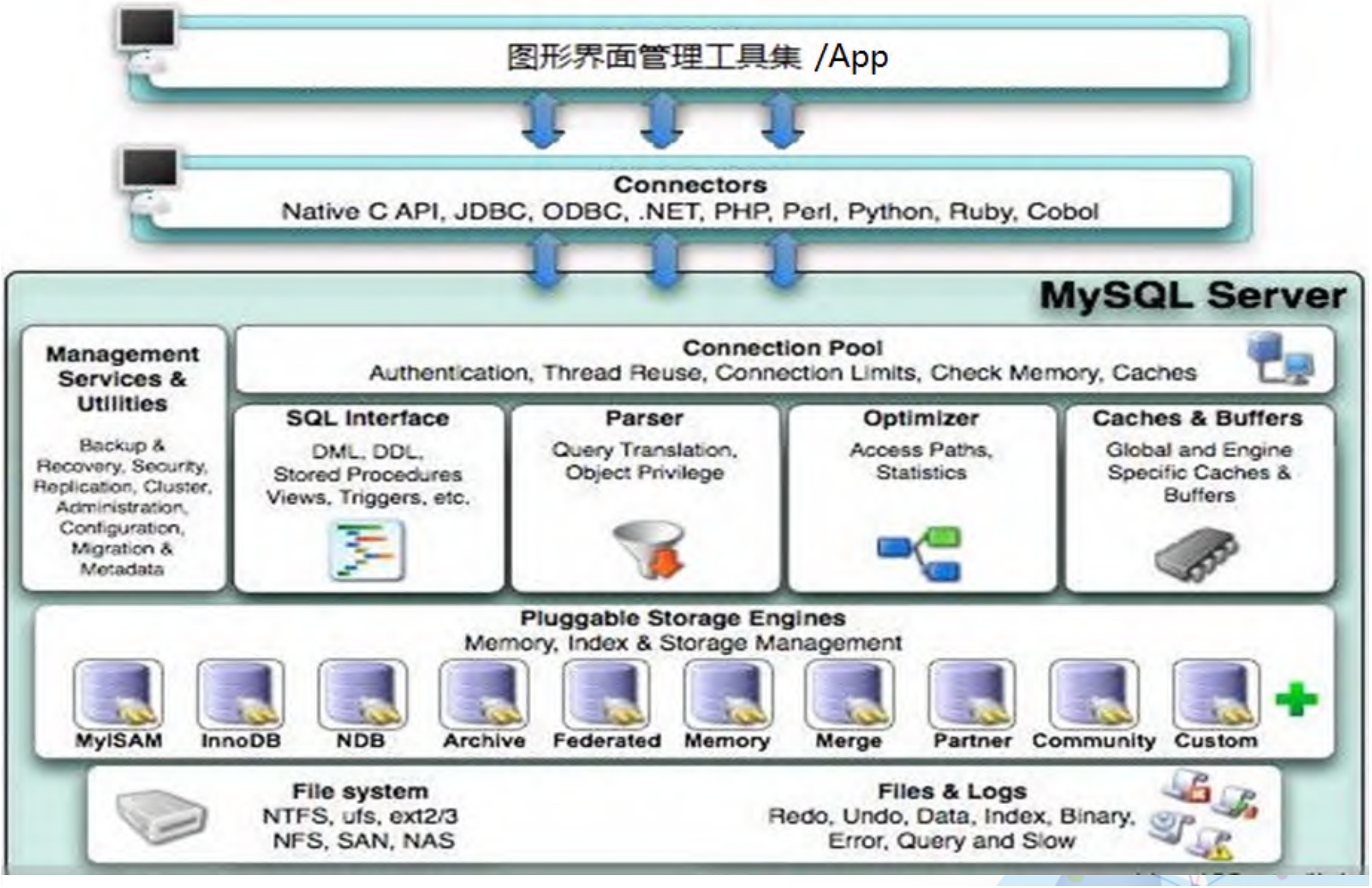
Infromix的扩展--数据刀片

高可用集群

高可靠高可用集群

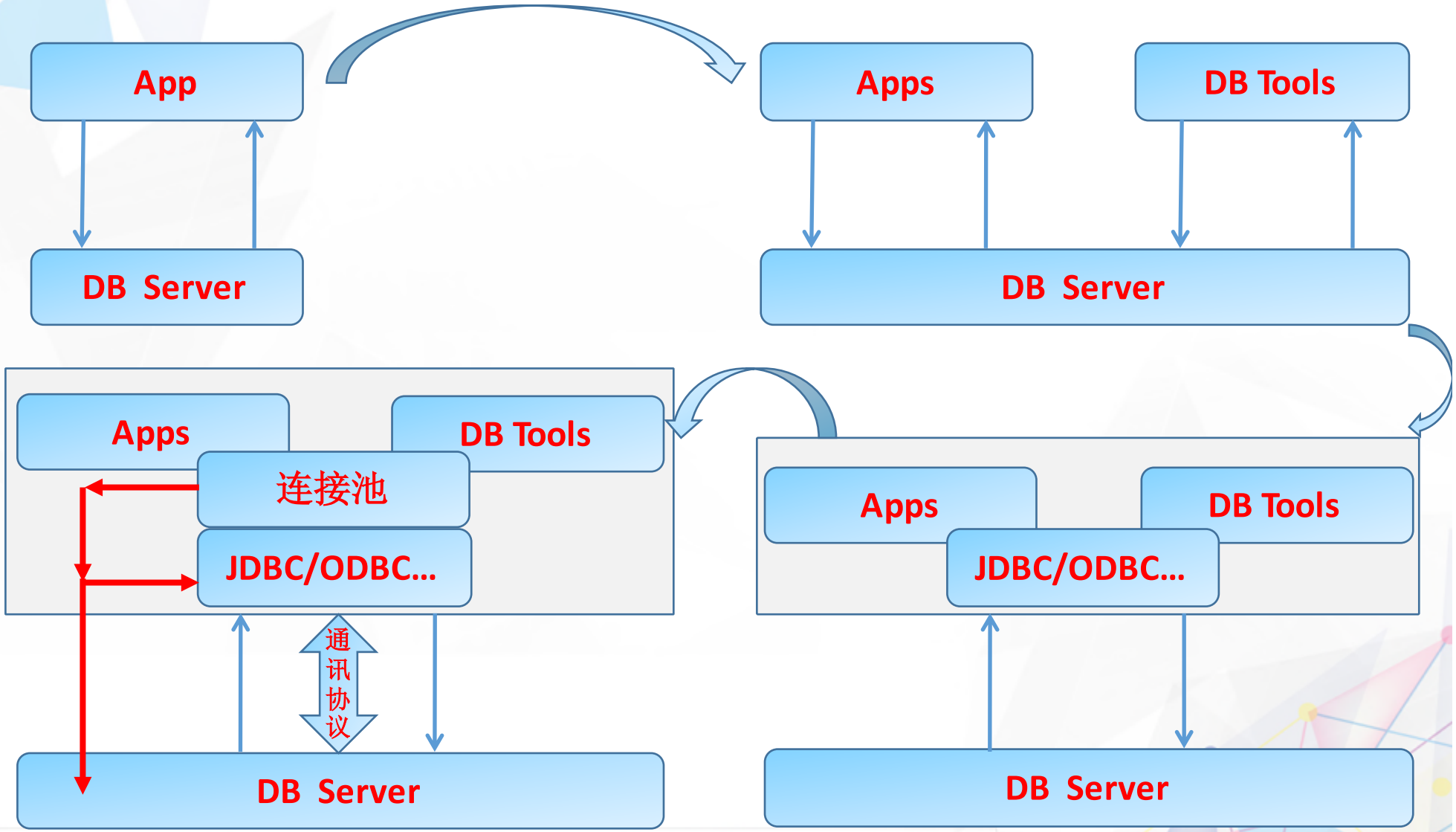
分布式NewSQL的三条路

1 前端架构


图形界面管理工具集 /App
Connectors
Native C API, JDBC, ODBC, .NET, PHP, Perl, Python, Ruby, Cobol
MySQL Server
Management Services & Utilities
Backup & Recovery, Security, Replication, Cluster, Administration, Configuration, Migration & Metadata
Connection Pool
Authentication, Thread Reuse, Connection Limits, Check Memory, Caches
SQL Interface
DML, DDL, Stored Procedures Views, Triggers, etc.
Parser
Query Translation, Object Privilege
Optimizer
Access Paths, Statistics
Caches & Buffers
Global and Engine Specific Caches & Buffers
Pluggable Storage Engines
Memory, Index & Storage Management
MyISAM
InnoDB
NDB
Archive
Federated
Memory
Merge
Partner
Community
Custom
File system
NTFS, ufs, ext2/3 NFS, SAN, NAS
Files & Logs
Redo, Undo, Data, Index, Binary, Error, Query and Slow

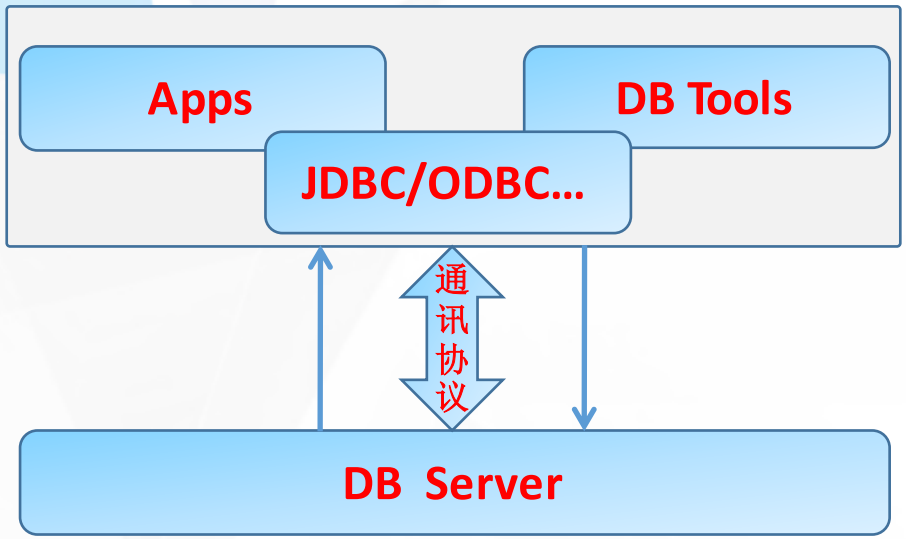
1 前端架构

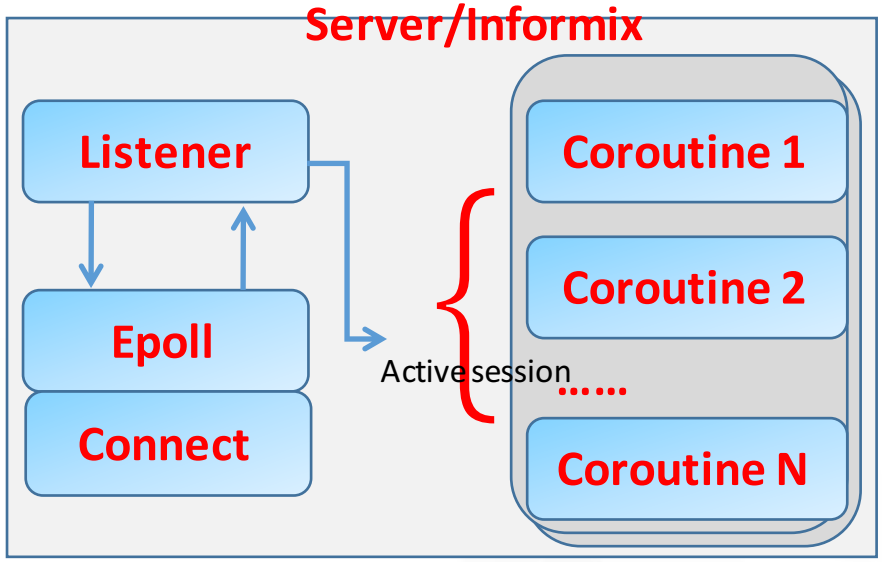
1.1 连接（APP、工具、接口）与协议

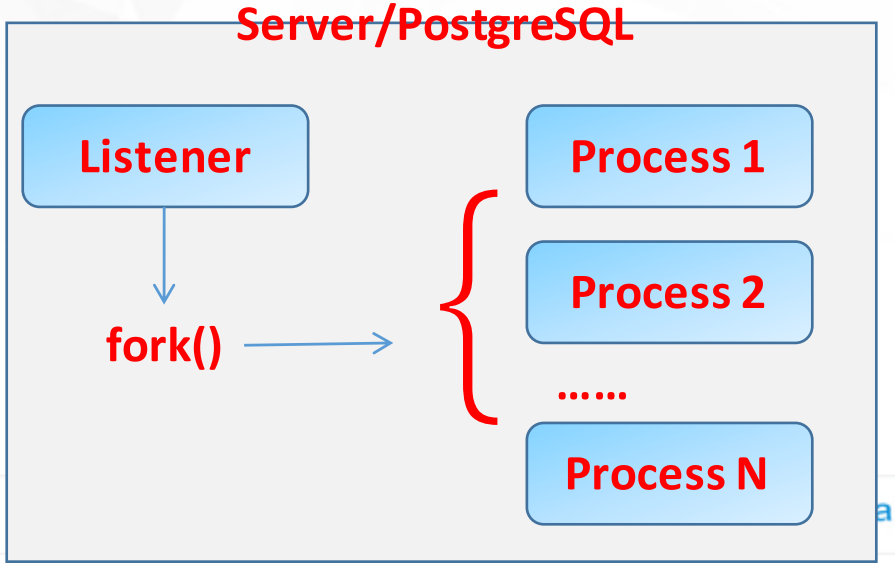


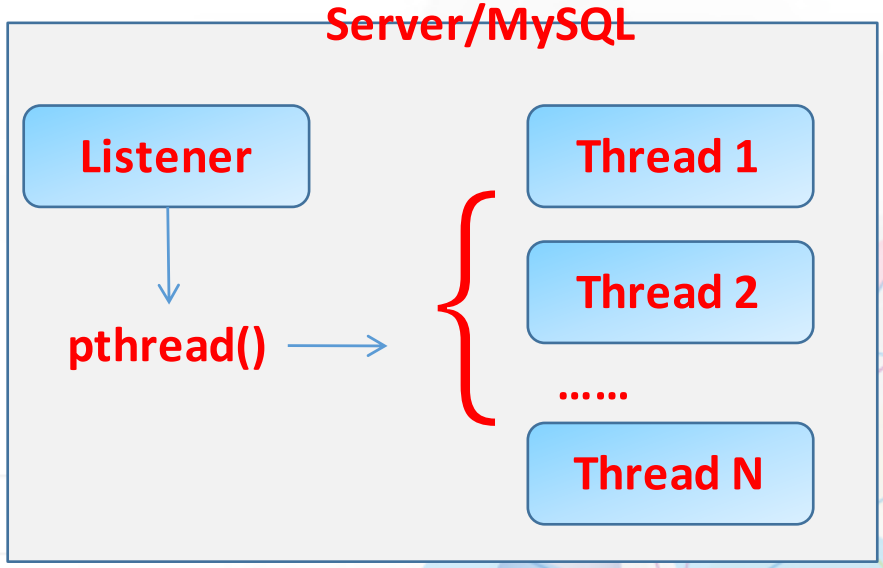
1 前端架构

1.2 面向连接的服务器架构（多进程、多线程、多协程）





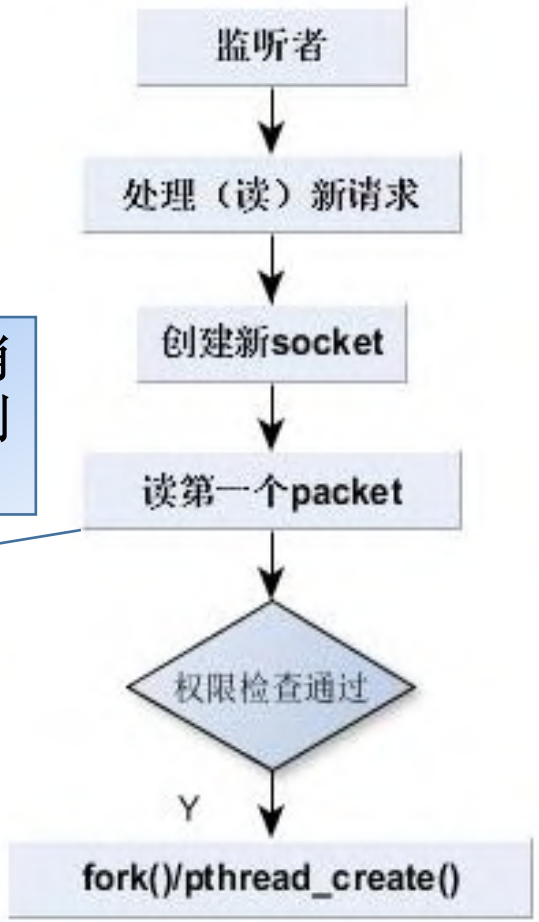


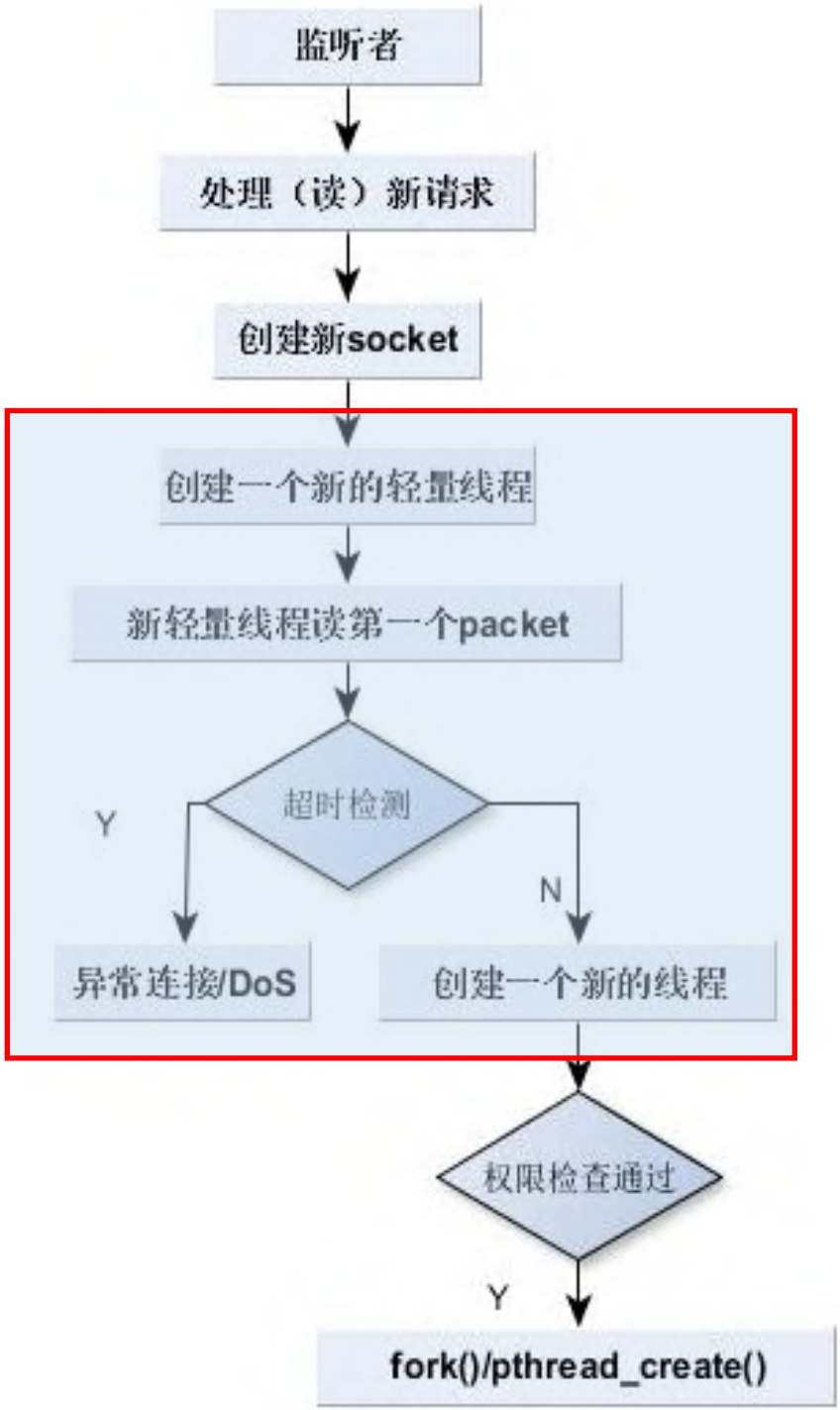


1 前端架构

1.3 安全防范（用户鉴别、防止Dos攻击）

客户端不发消息，监听者则阻塞于此





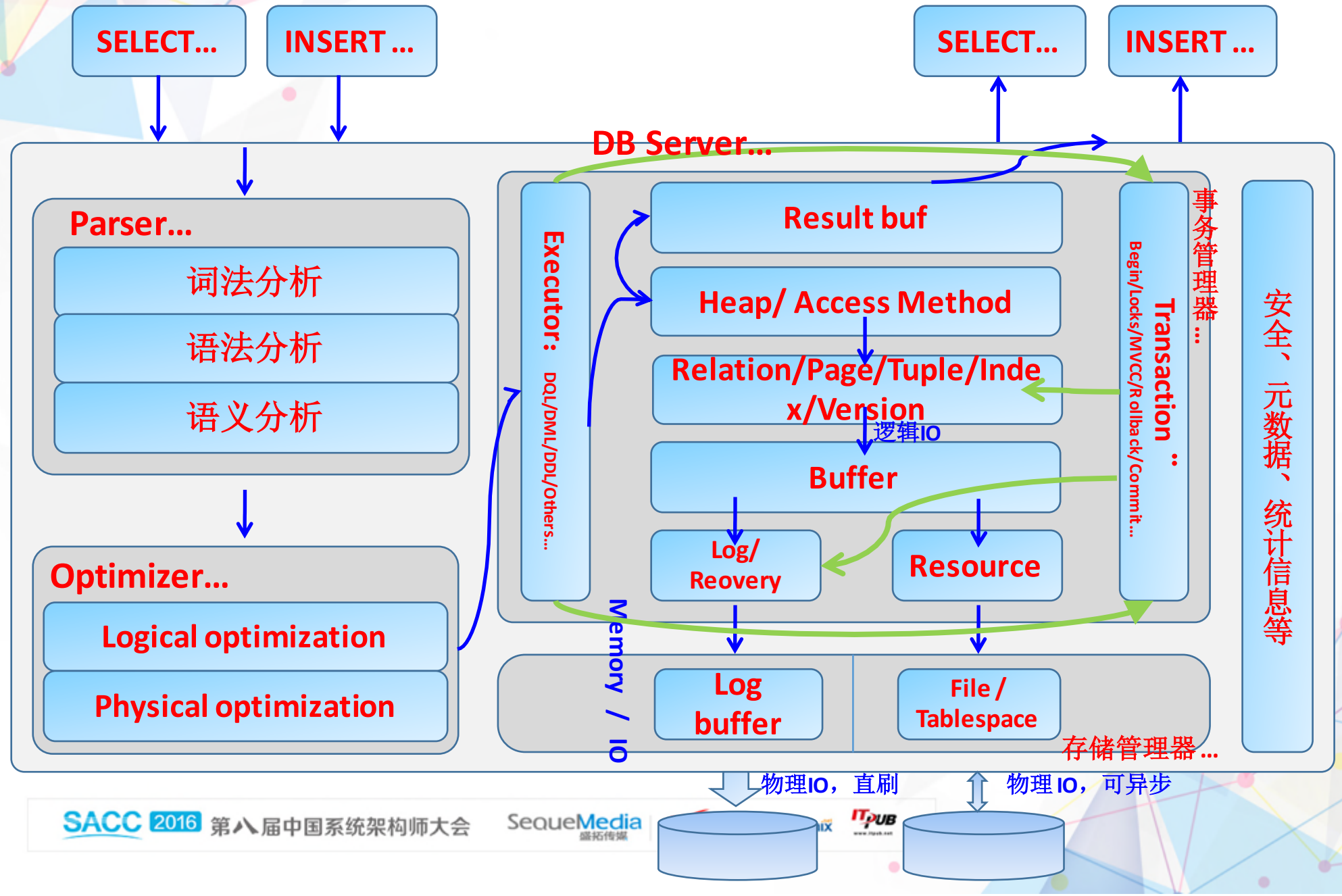
1 前端架构

前端架构需要考虑的问题：

- 多用户访问数据库
- 高效
- 安全

2 整体架构

2.1数据库引擎的构成



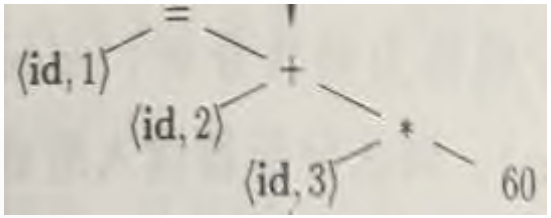
2 整体架构

2.2数据库引擎的分析器

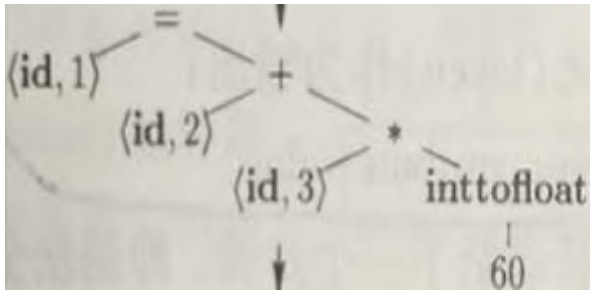
```
SELECT SUM(a1), id1
FROM t1 LEFT JOIN t2 on a1=a2 LEFT JOIN t3 on b2=b3
WHERE k1 IN (SELECT k3 FROM t3 AS t WHERE a3<30)
    AND b2=10 AND k2=10
GROUP BY id1;
```

1 词法分析

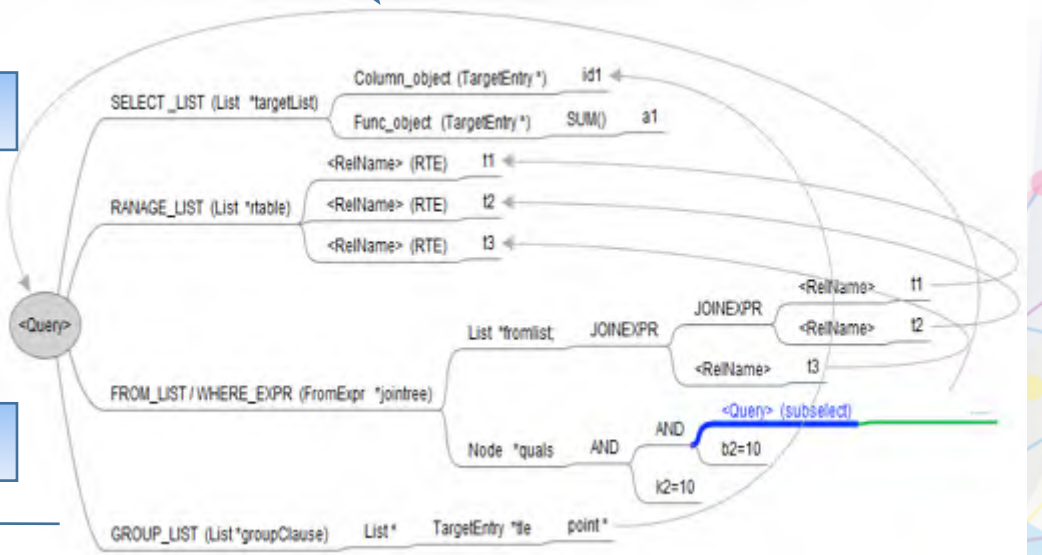
2 语法分析



3 语义分析



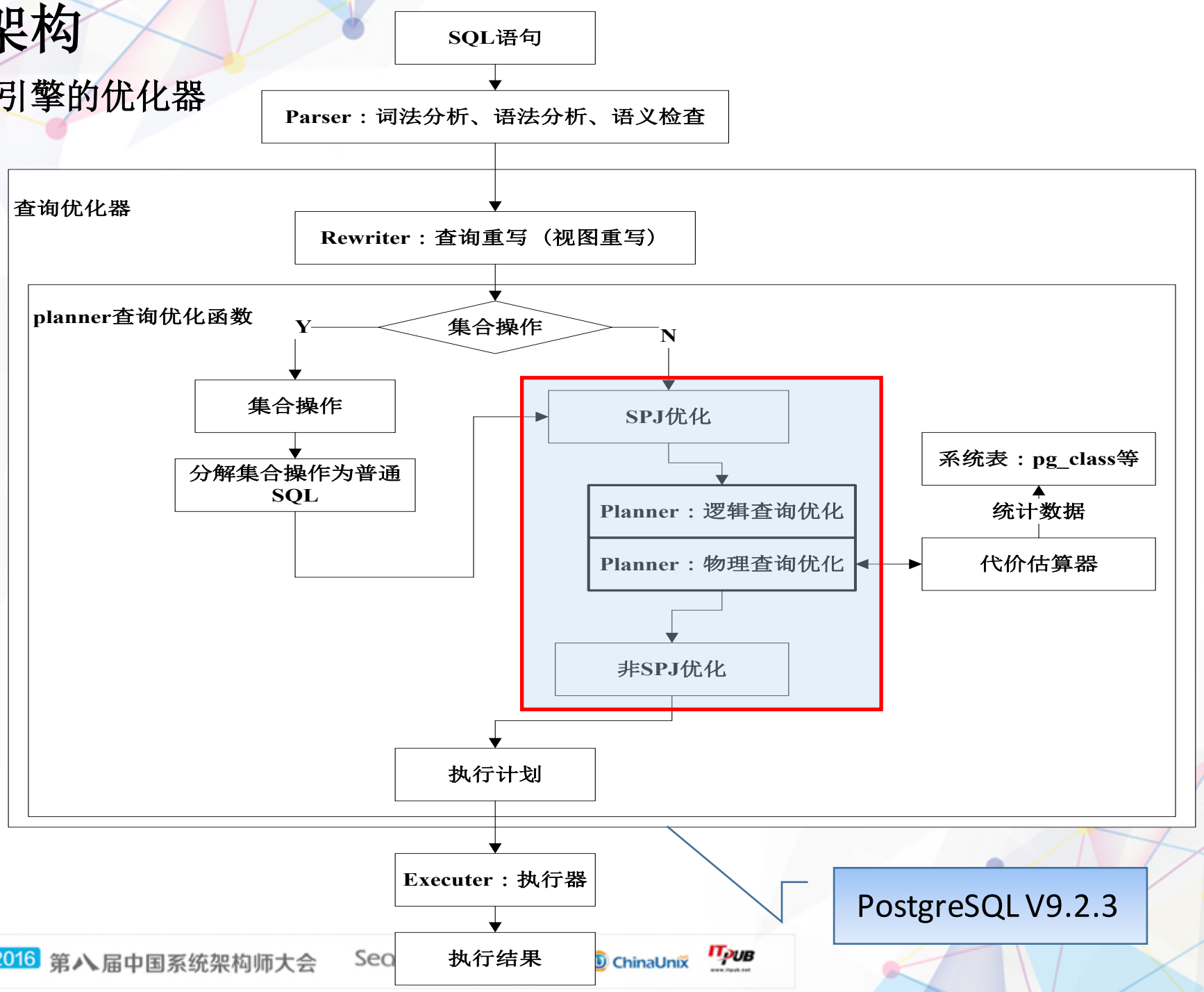
4 分析阶段产物：语法树



5
SQL分析 == 编译器前半段

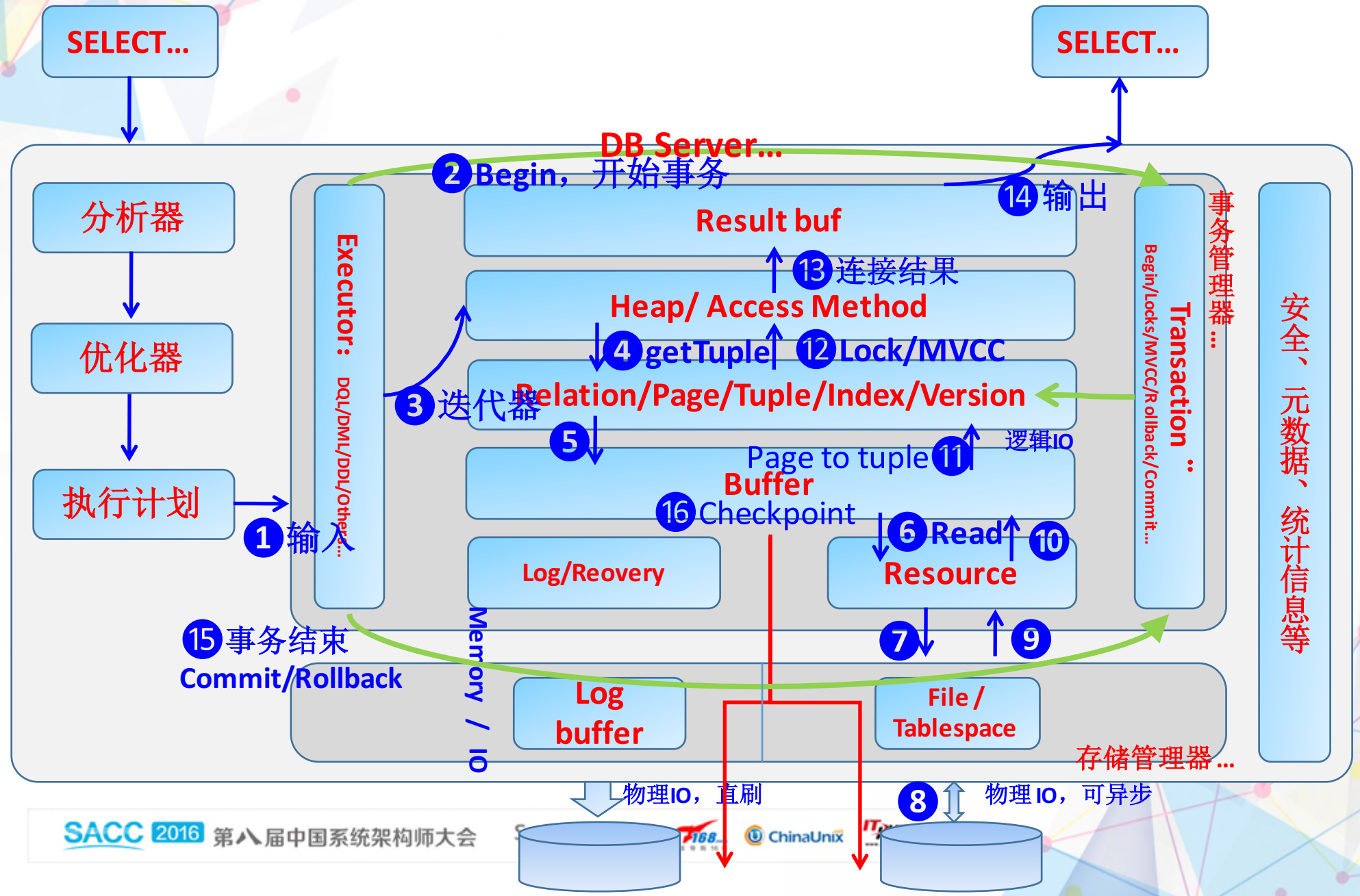
2 整体架构

2.3数据库引擎的优化器



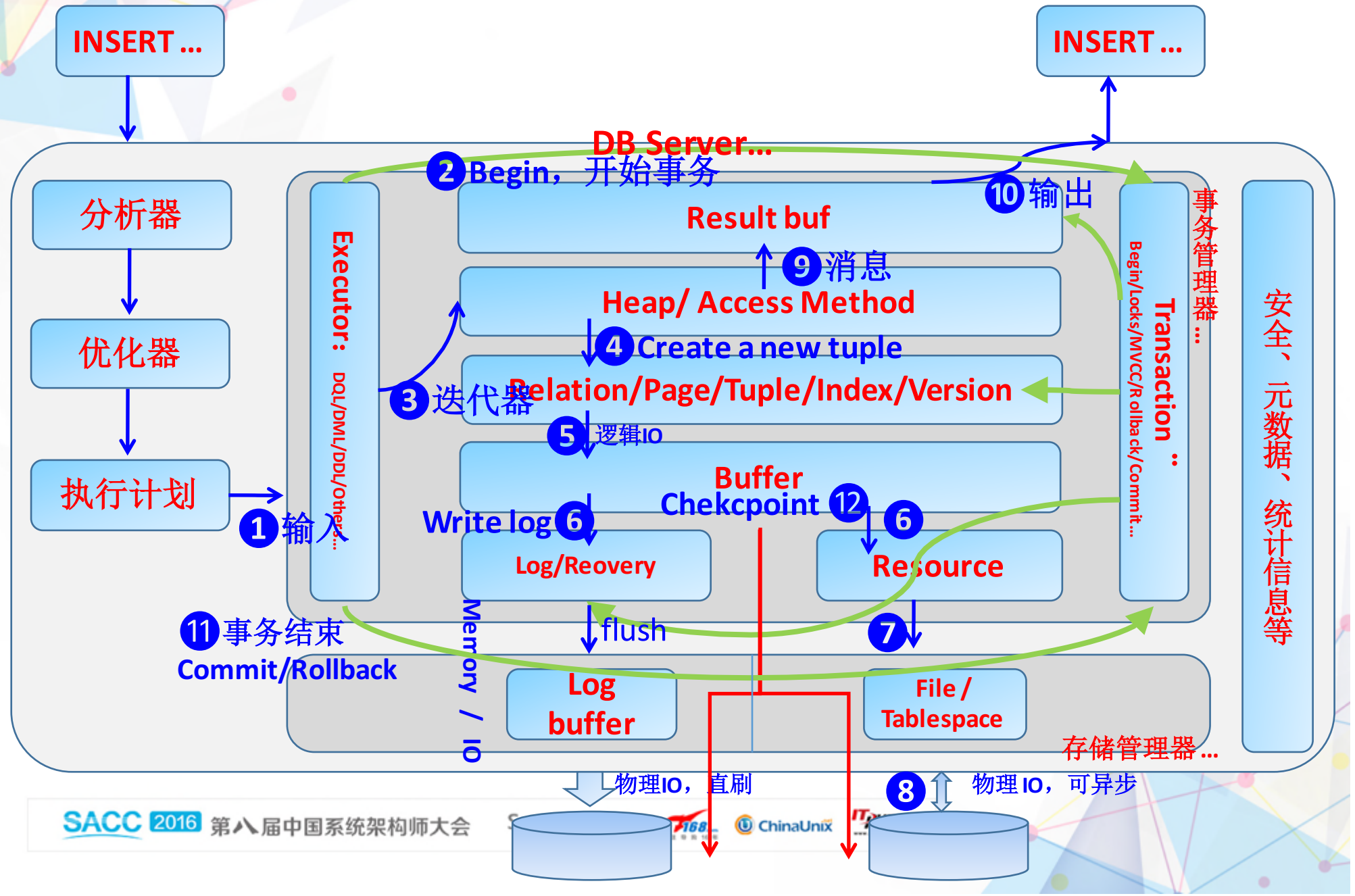
2 整体架构

2.4数据库引擎的执行器---查询语句的执行过程



2 整体架构

2.5数据库引擎的执行器—插入语句的执行过程



2 整体架构

整体架构需要考虑的问题：

- ACID特性保证—事务机制
- 数据可被并发访问——高效

3 扩展架构

扩展粒度细，层次多

闭源系统，少有扩展

类别	比较项	PostgreSQL	MySQL	Informix
存储	支持可替换整个存储引擎	不支持	通过handler实现	不支持
	数据存储底层的文件操作可替换	通过smgr层实现	不支持	支持
	用户可自定义数据类型	支持	不支持	支持
优化器	支持可替换整个优化器	支持	不支持	不支持
	支持可替换单表扫描的方法	支持	不支持	不支持
执行器	支持可替换整个执行器	支持	不支持	不支持
数据访问	用户可自定义索引	支持	不支持	不支持
	用户可自定义操作符	支持	不支持	不支持
	用户可自定义外部数据源	支持	不支持	不支持
	可以自定义数据采样方法	支持	不支持	不支持
进程	支持附加用户进程到服务器共享内存	支持 通过BackgroundWorker实现	不支持	支持
用户功能	自定义函数	支持	支持	支持
	存储过程	支持多种语言	单一语言	支持多种语言
	触发器	支持	支持	支持

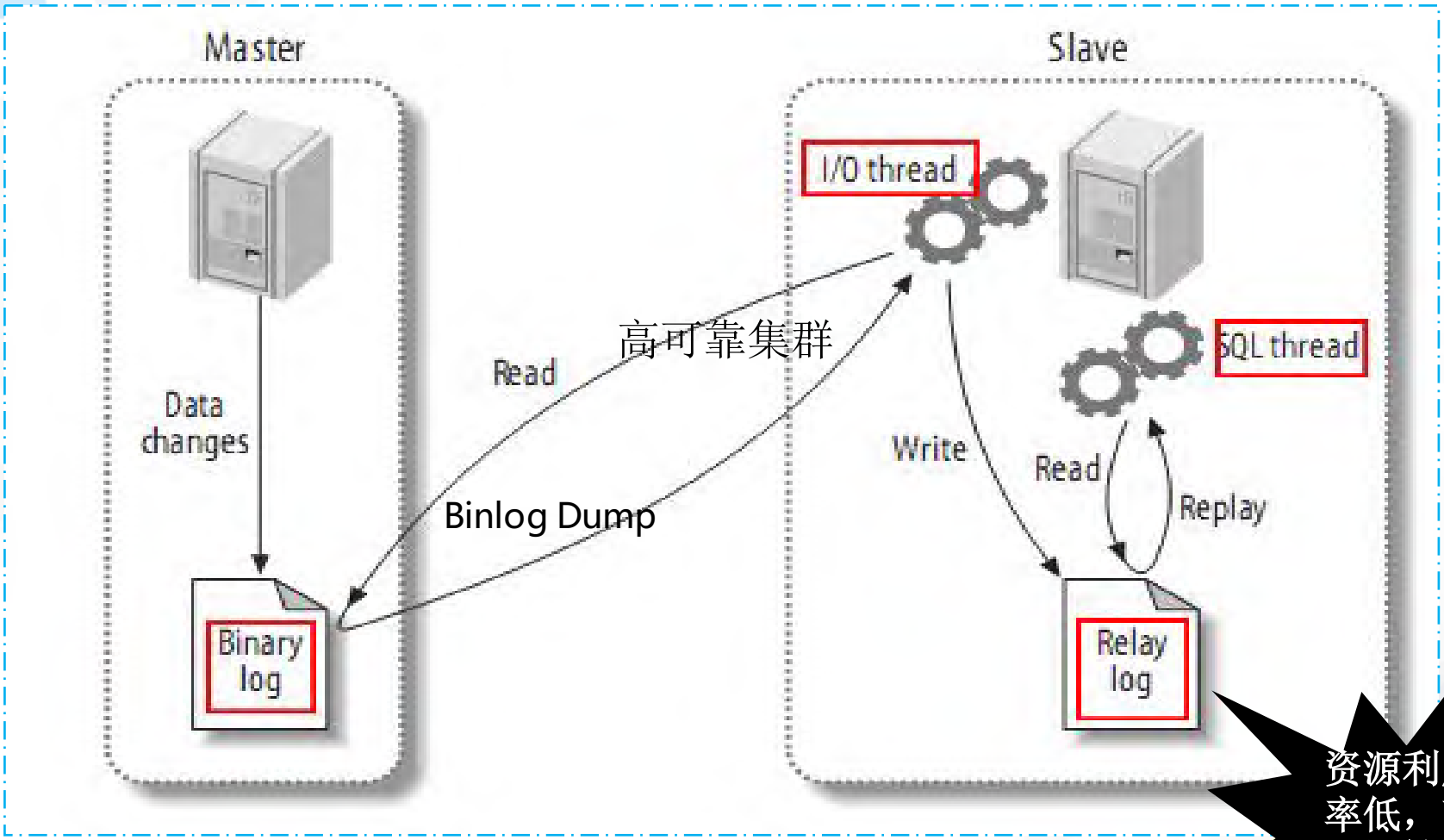
扩展粒度粗，面向存储引擎

4 集群架构

单机系统
单点故障

方案一：主从复制

MySQL复制原理



资源利用率低，高可用性差

4 集群架构

4.1 高可用集群

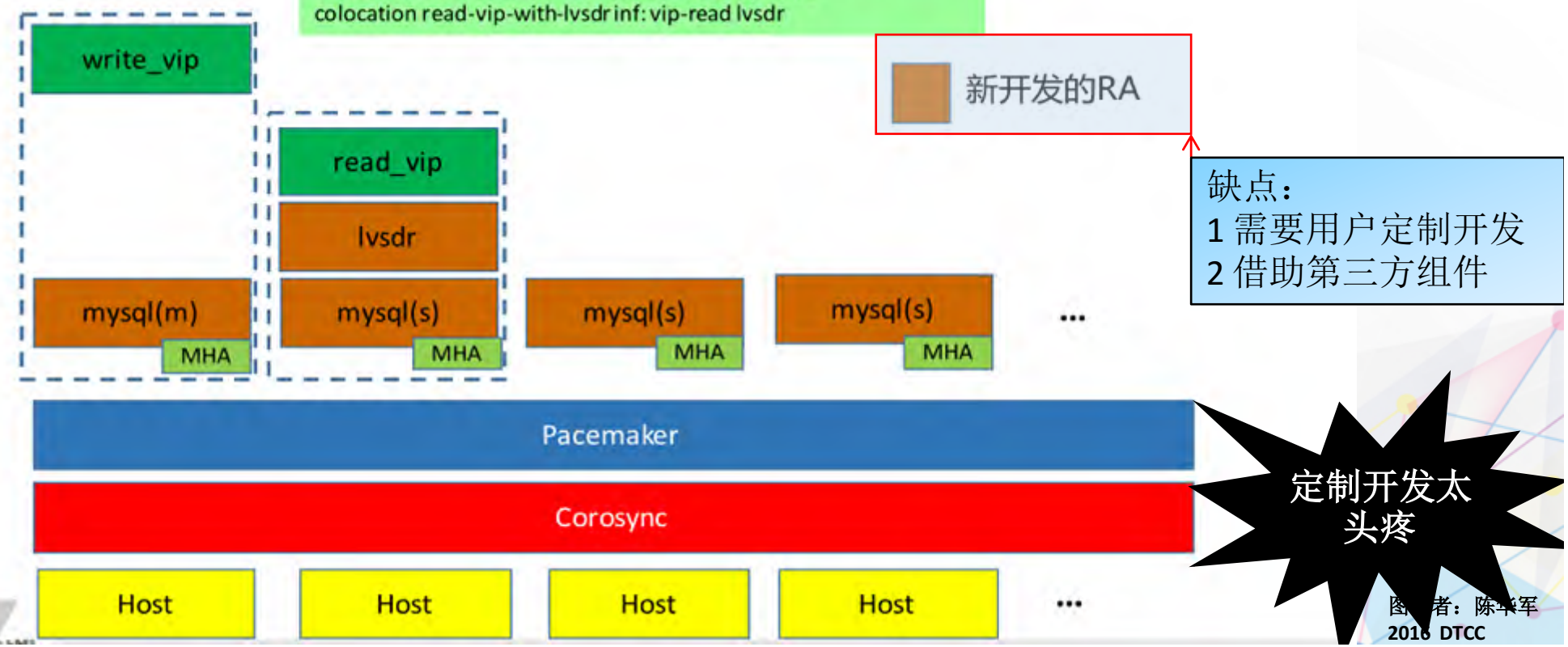
方案二：一主多从复制

实现架构

- 新开发MySQL和LVS的Resource Agent进行资源监控
- 每个节点部署MHA node做实际的failover
- 通过约束使写VIP和Master动态绑定，读VIP+LVS和Slave动态绑定

为保证数据一致性和部署的灵活性，没有采用ClusterLabs开源项目自带的mysql和ldirectord RA。

```
colocation write-vip-with-master inf: vip-write msMysql:Master
colocation lvsdr-with-slave inf: lvsdr msMysql:Slave
colocation read-vip-with-lvsdr inf: vip-read lvsdr
```



新开发的RA

缺点：
1 需要用户定制开发
2 借助第三方组件

定制开发太头疼

4 集群架构

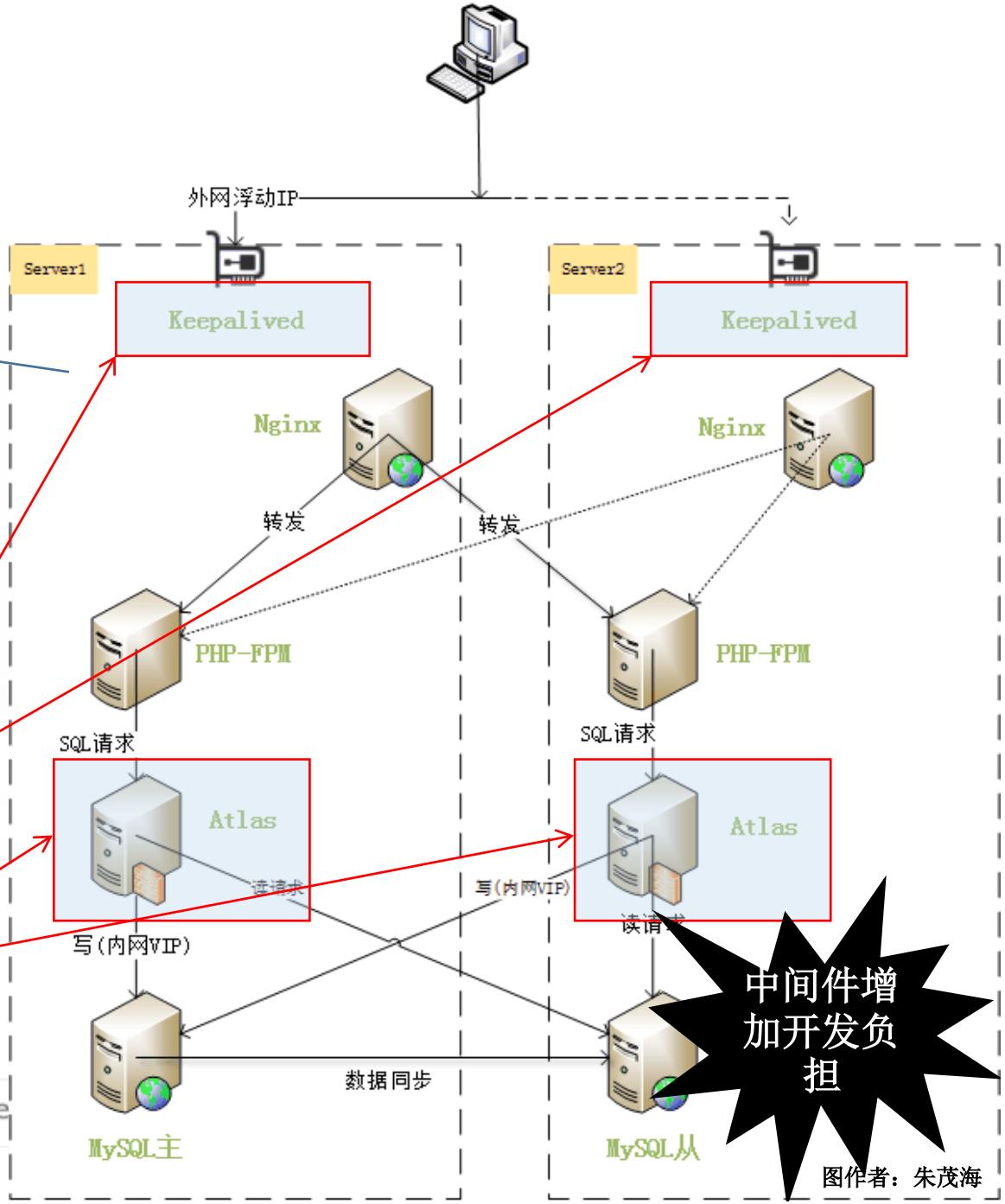
4.1 高可用集群

方案三：读写分离架构

读写分离架构

问题：
主备之间的网络出现故障时，集群脑裂，导致数据双写，VIP来回切换

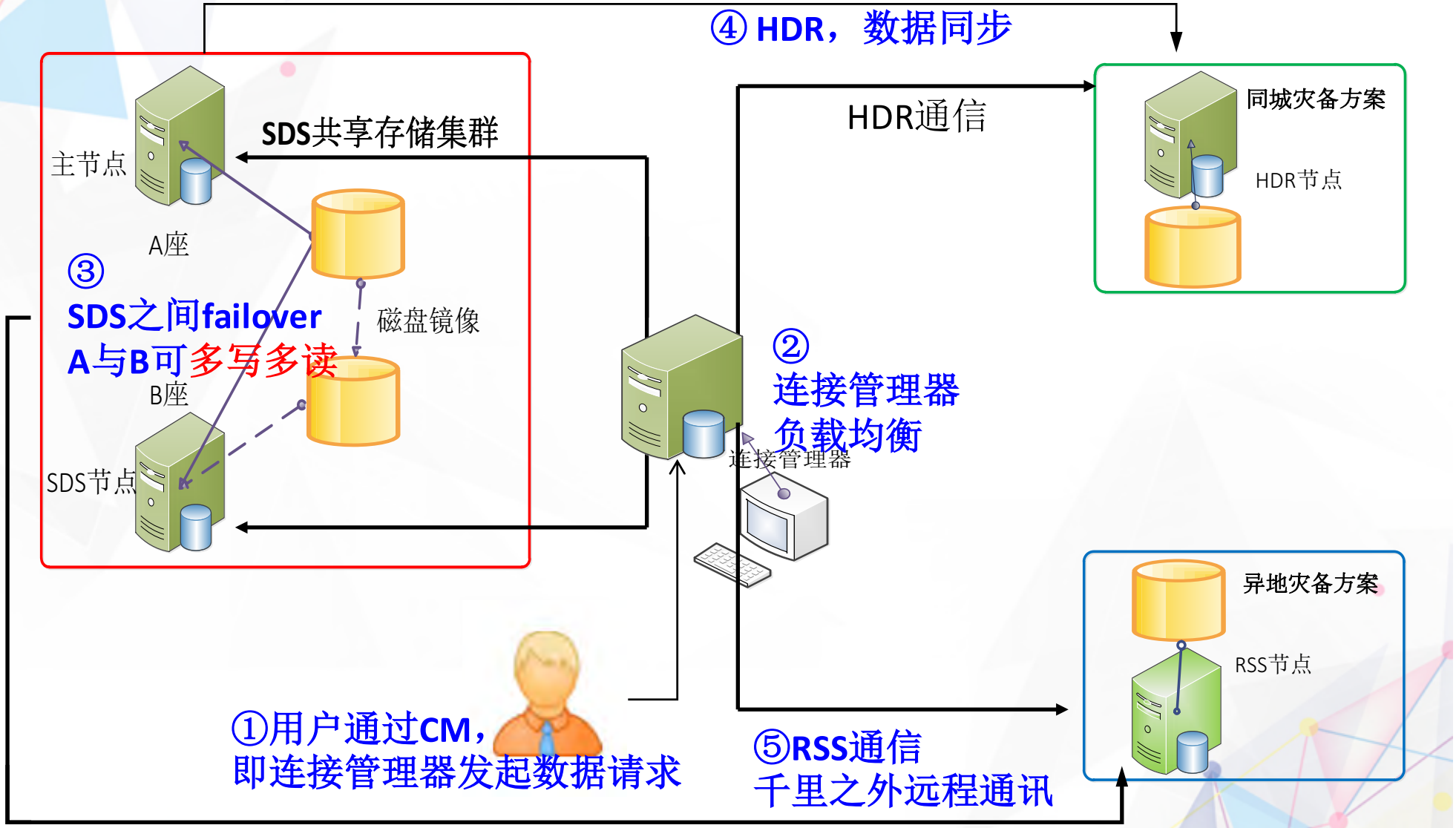
缺点：
1 failover依赖于第三方组件
2 读写分离依赖于MyCat/Atlas等分布式中间件
3 数据需要人工/自动分片



图作者：朱茂海

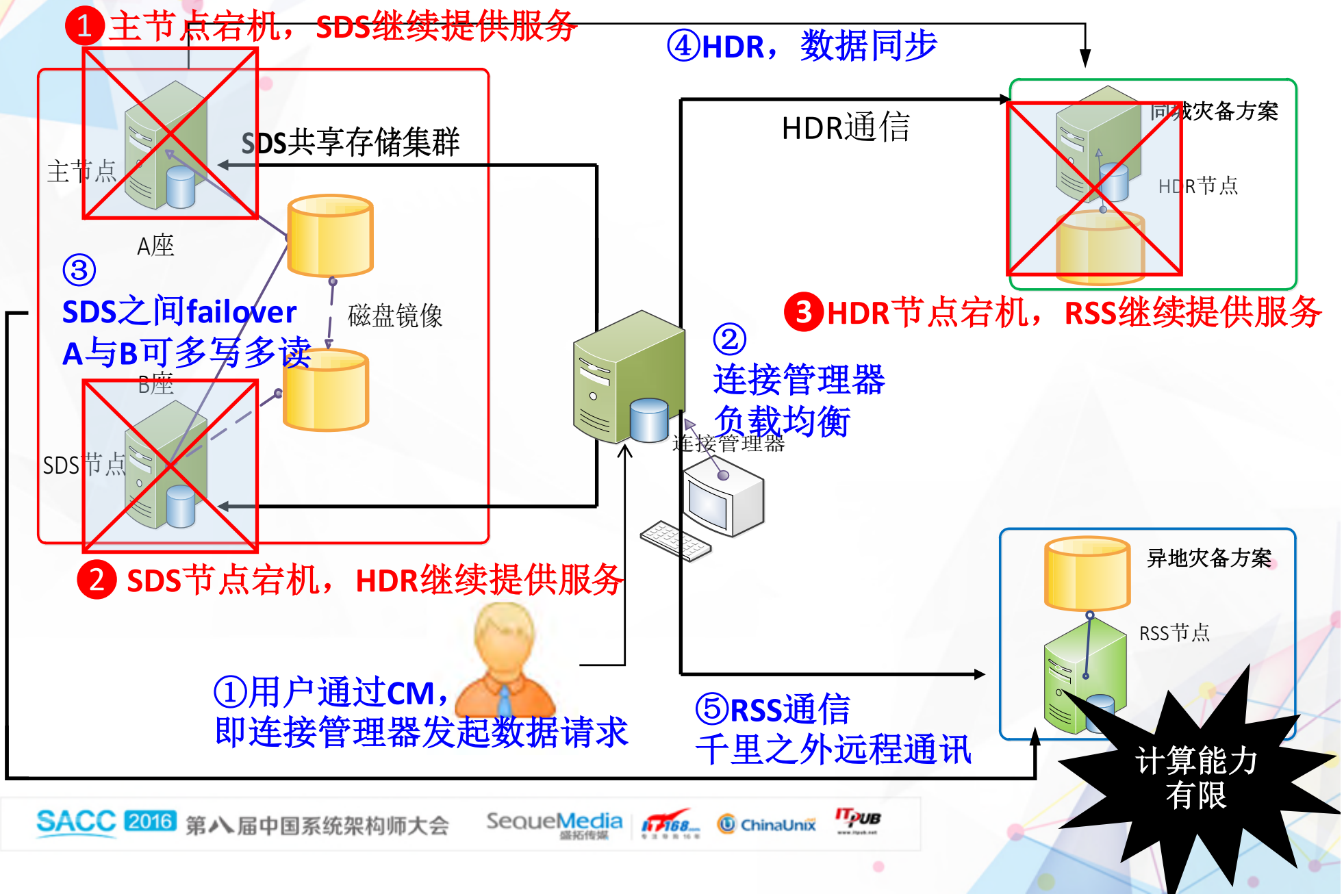
4 集群架构

4.2 高可靠高可用集群



4 集群架构

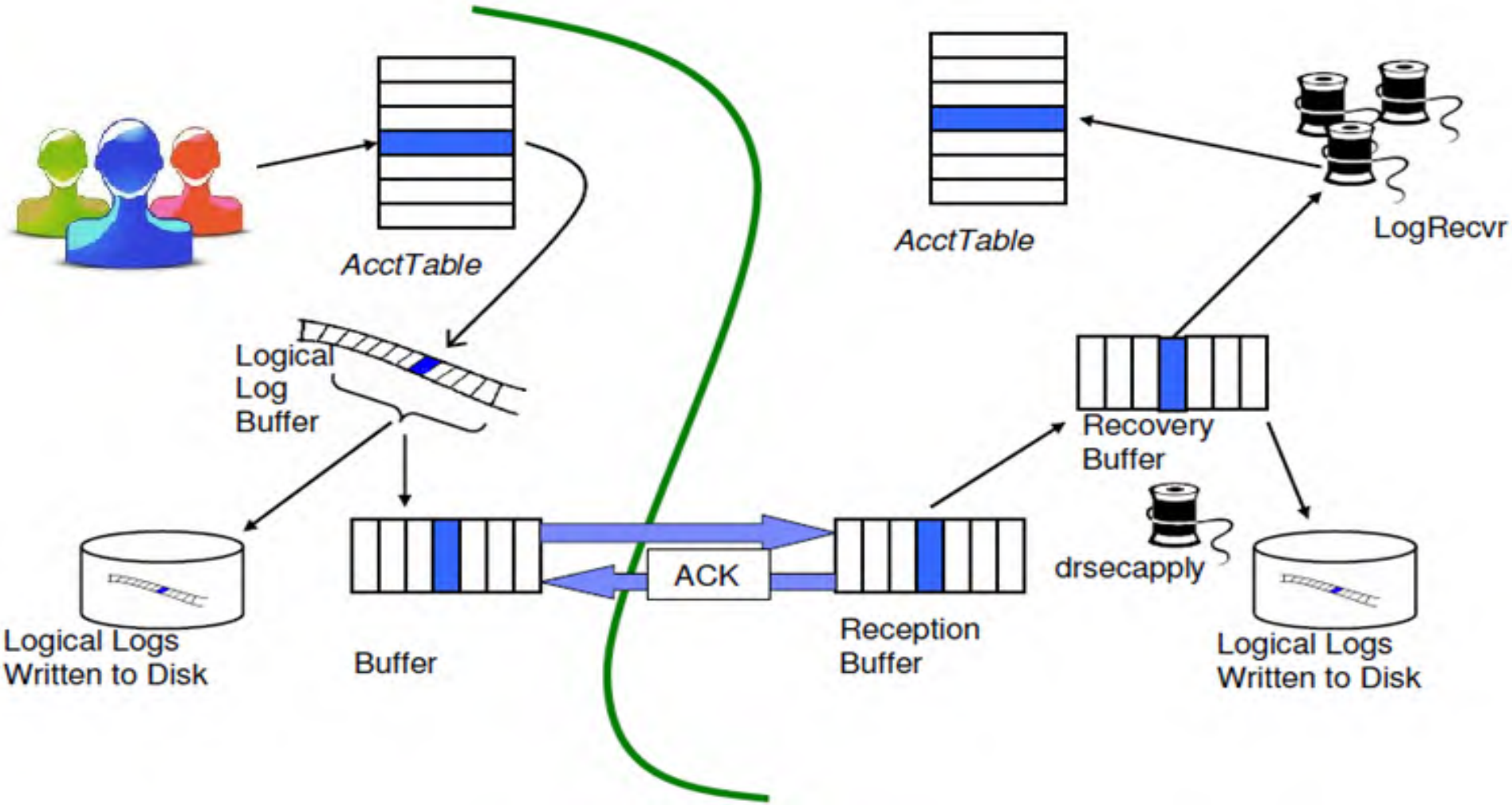
4.2 高可靠高可用集群



4 集群架构

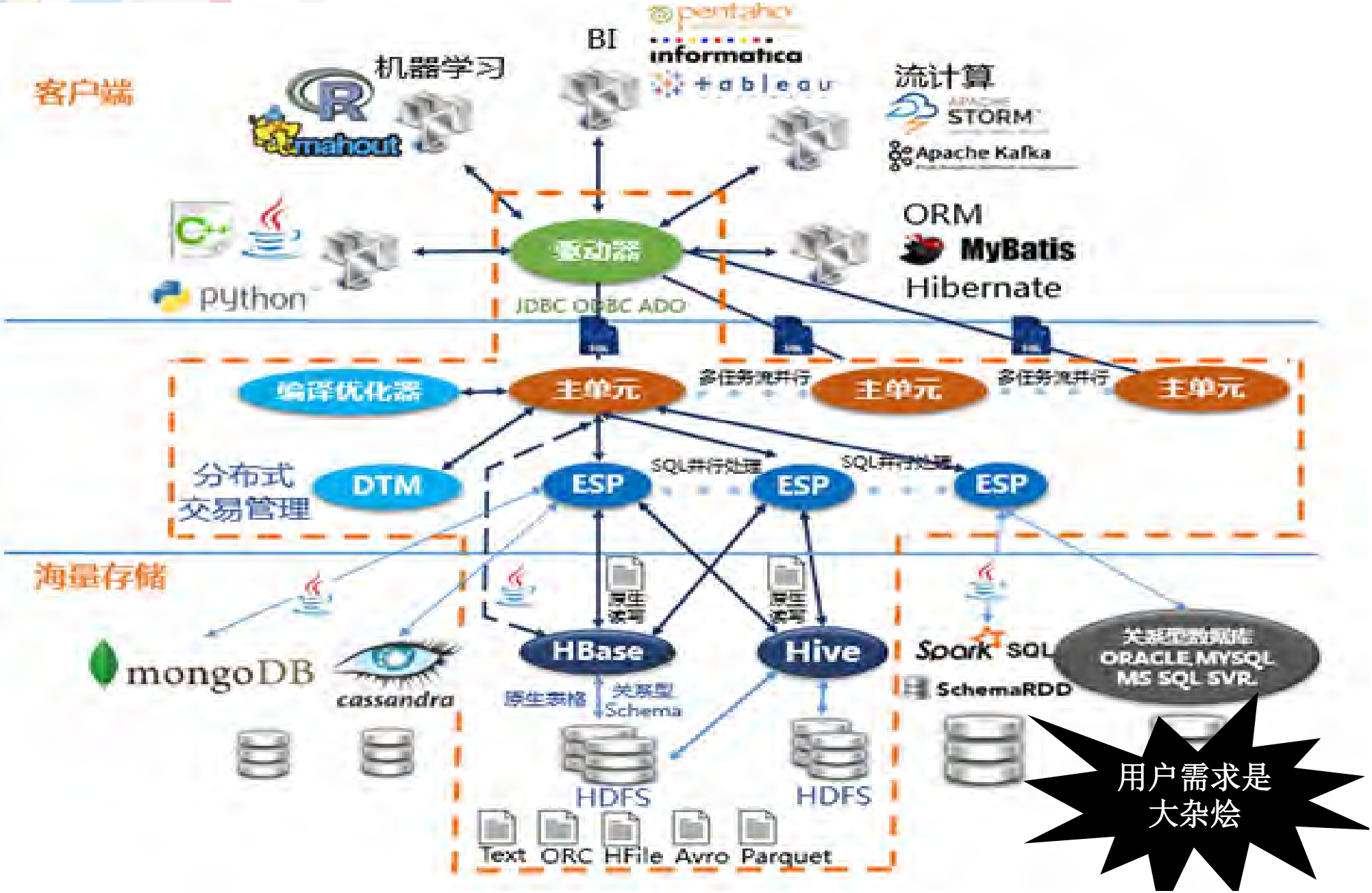
4.2 高可靠高可用集群

Infromix两地三中心架构---复制原理



4 集群架构

4.3 分布式NewSQL的三条路



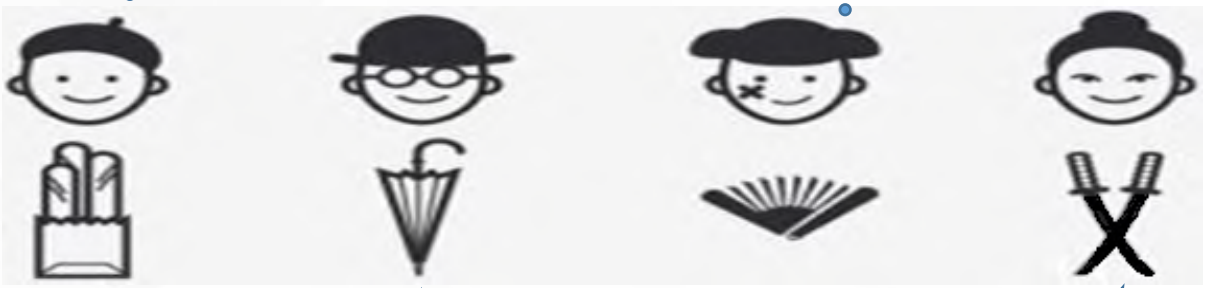
4 集群架构

4.3 分布式NewSQL的三条路

用户期望

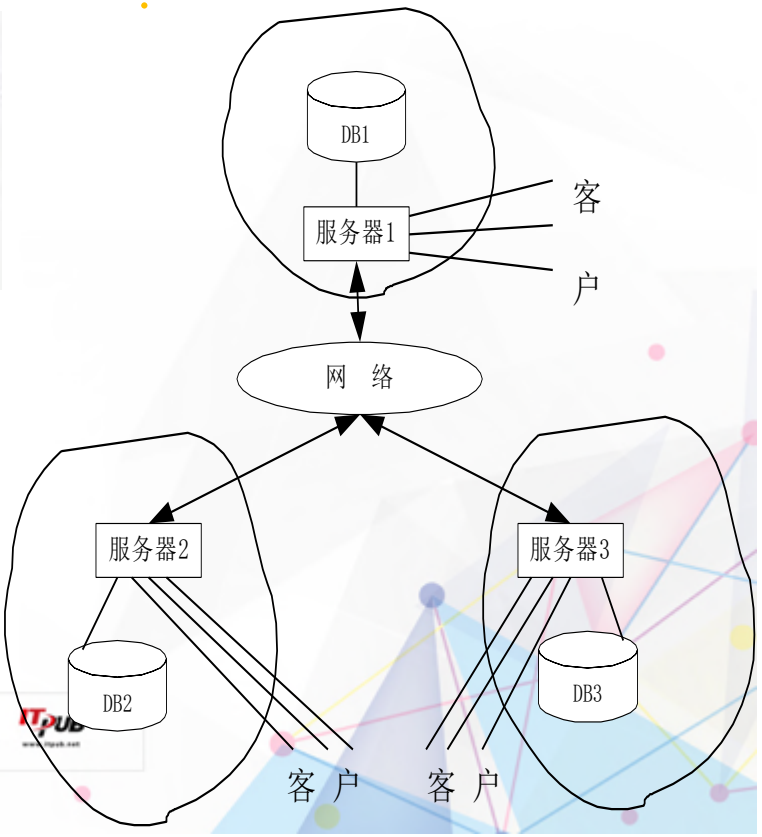
1 简化开发
2 海量存储

5 自动增容、减容
6 几乎无failover



3 高性能计算
4 结构化、半结构化数据等

7 有事务（ACID）特性
8 提供SQL接口



4 集群架构

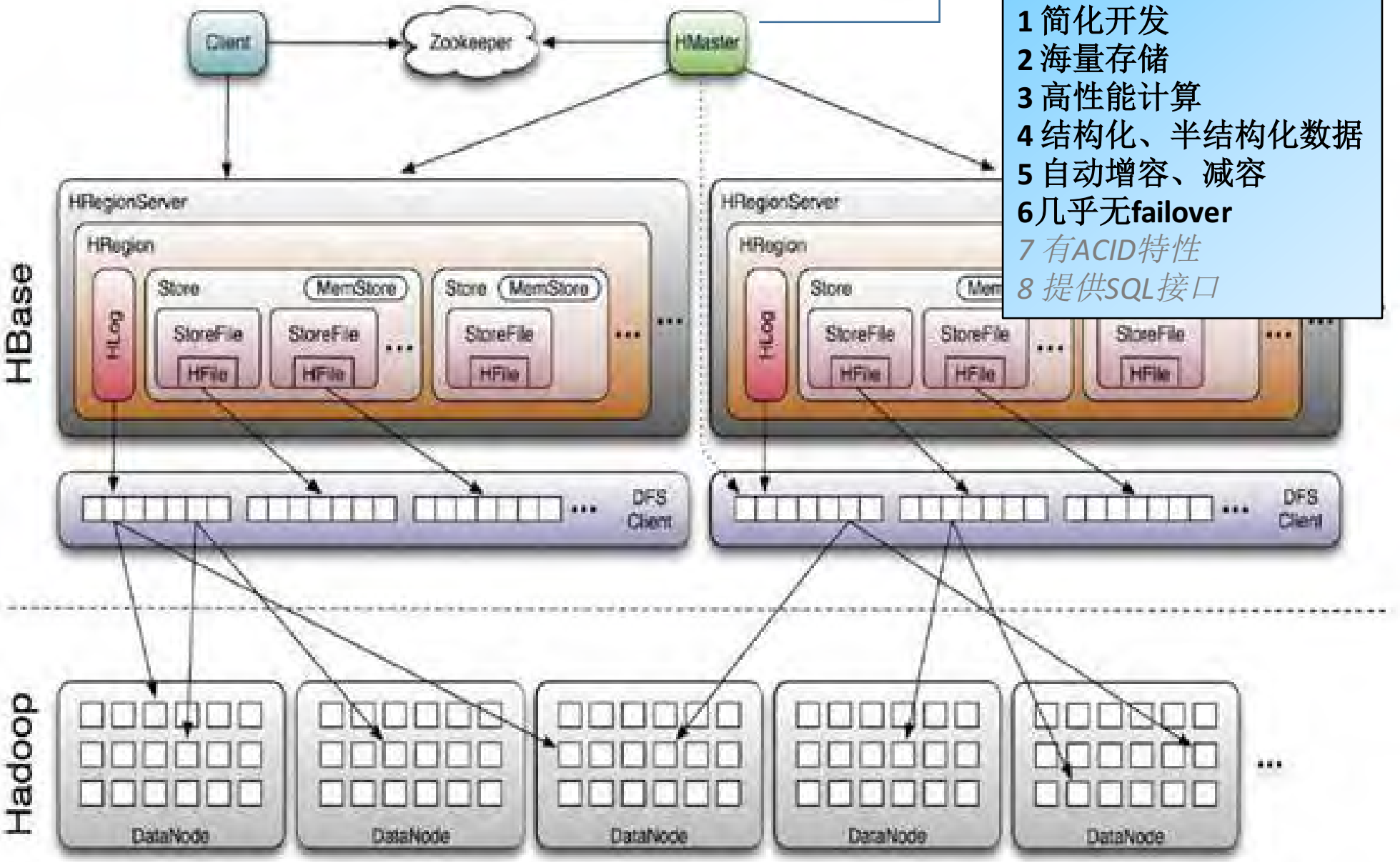
4.3 分布式NewSQL的三条路

第一条路：

NoSQL走向NewSQL：快速原型法，技术上越走越艰难

考虑的问题

1 简化开发
2 海量存储
3 高性能计算
4 结构化、半结构化数据
5 自动增容、减容
6几乎无failover
7 有ACID特性
8 提供SQL接口

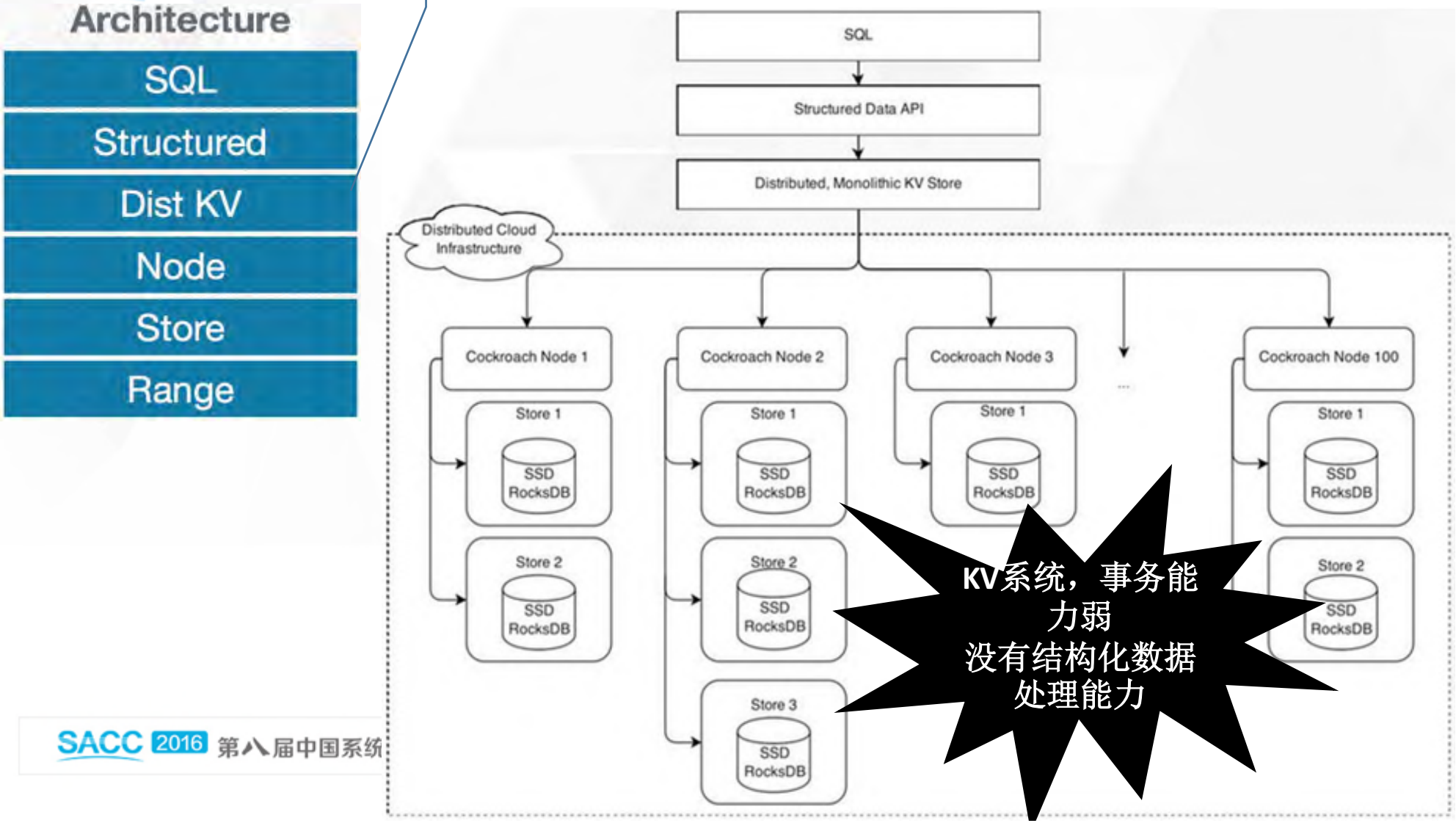


4 集群架构

4.3 分布式NewSQL的三条路

第一条路：
NoSQL走向NewSQL：快速原型法，技术上越走越艰难

CockroachDB，存储层是KV系统

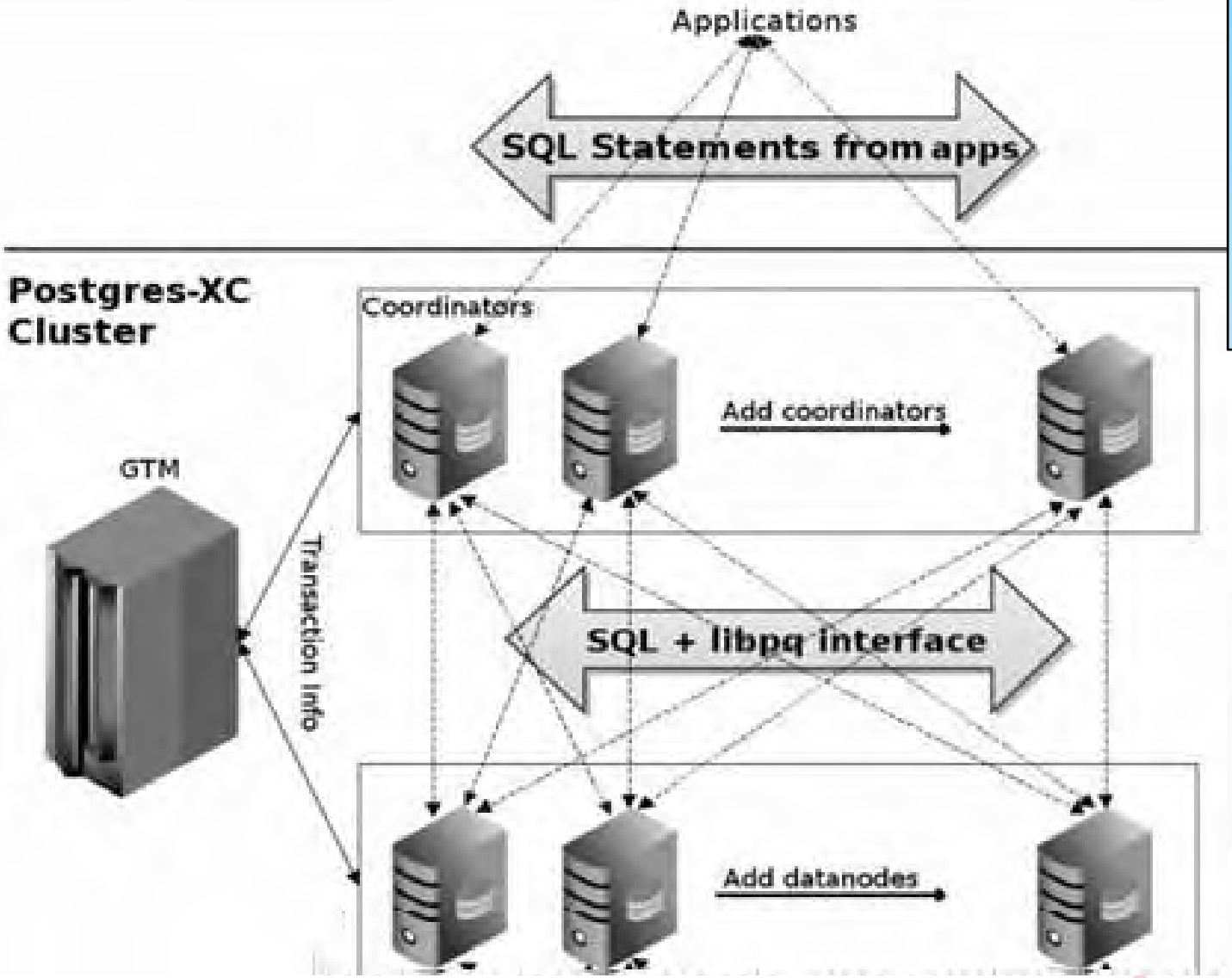


4 集群架构

4.3 分布式NewSQL的三条路

第二条路：
传统数据库走向NewSQL：架构改动法。

考虑的问题
1 简化开发
2 海量存储
3 高性能计算
4 结构化、半结构化数据
5 自动增容、减容
6 几乎无failover
7 有ACID特性
8 提供SQL接口



4 集群架构

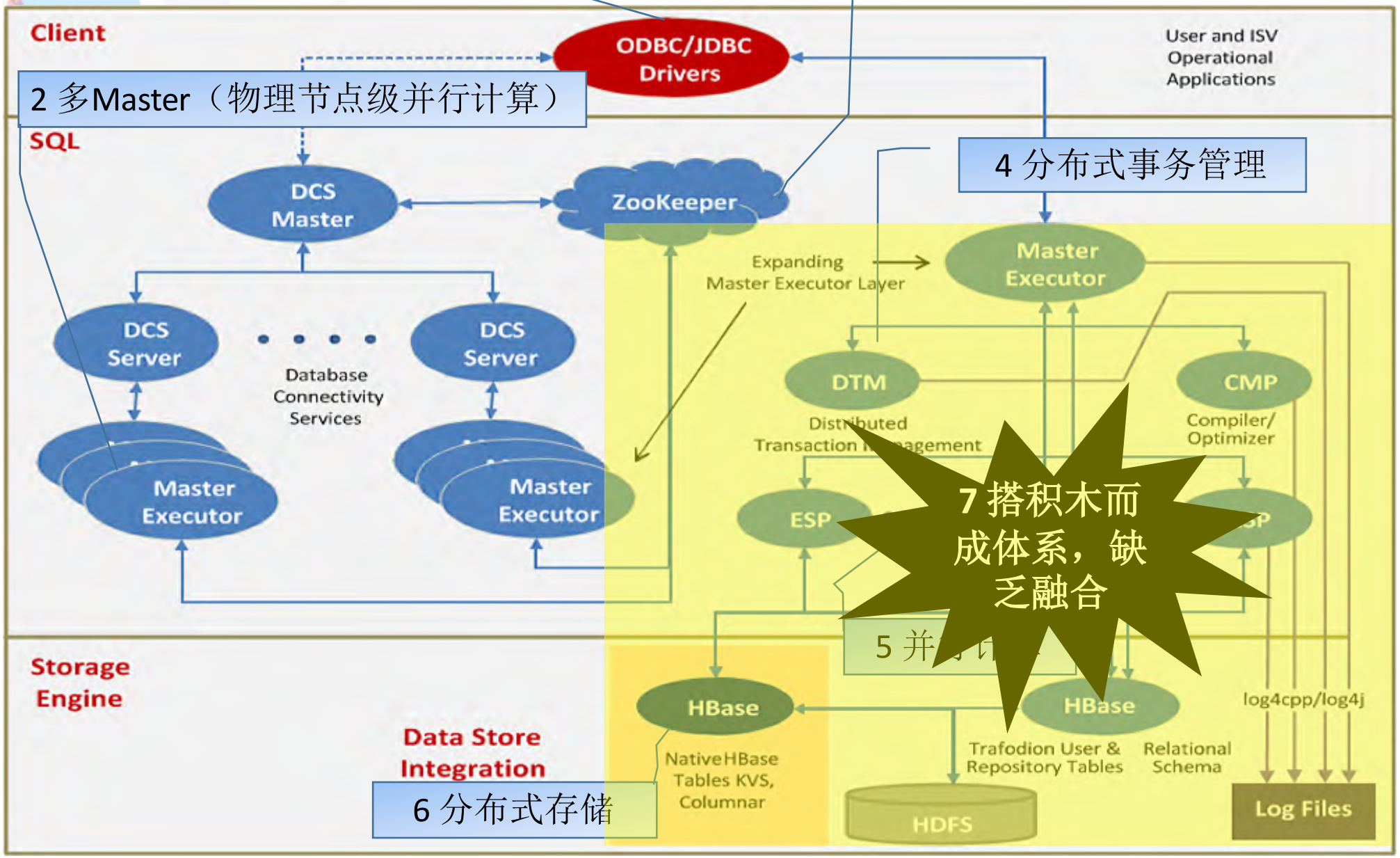
4.3 分布式NewSQL的三条路

混搭架构： Trafodion

1 用户接入简单（支持SQL）

第三条路：
混搭架构。

3 多主协调避免Master的单点故障

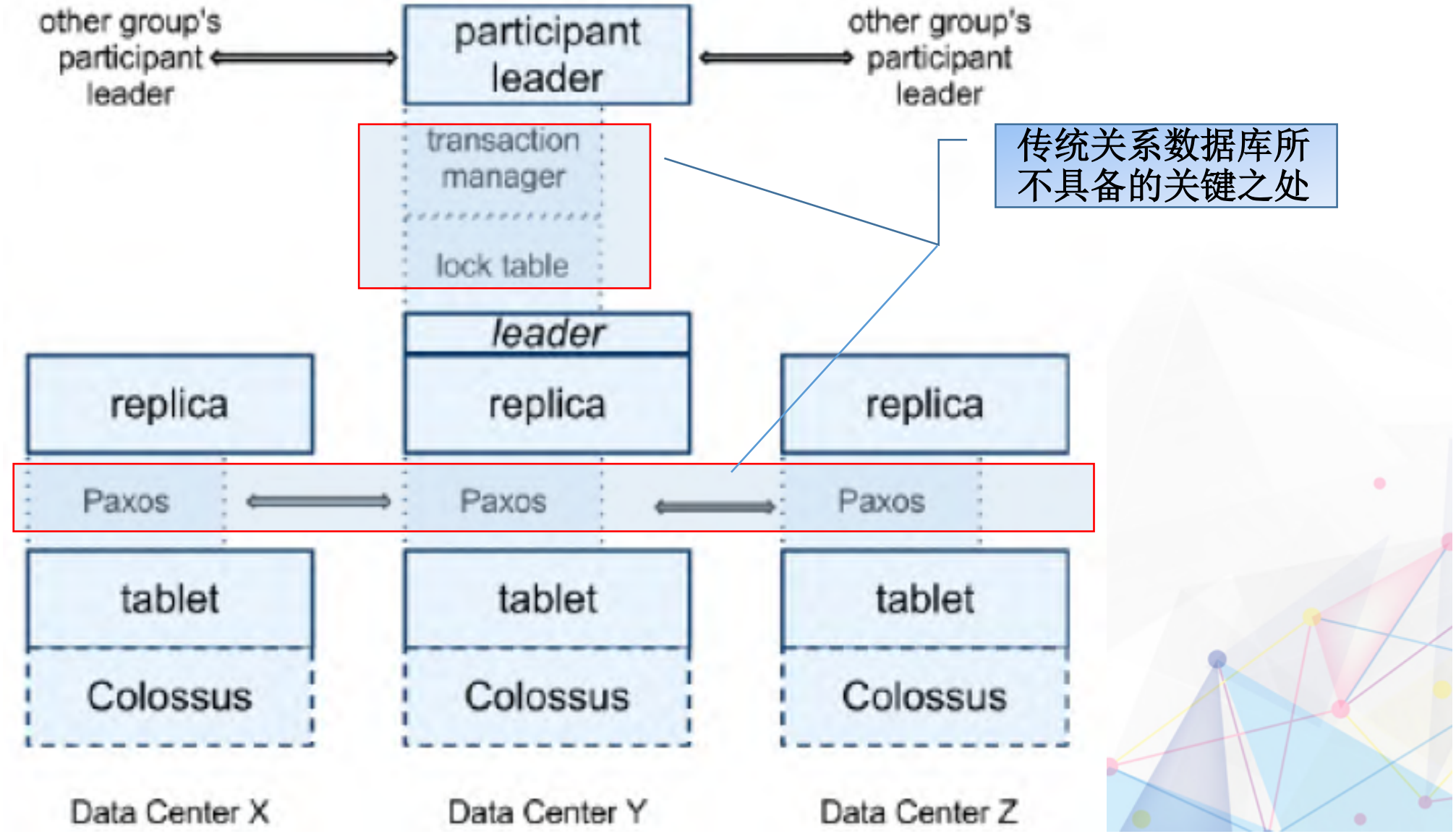


4 集群架构

4.3 分布式NewSQL的三条路

Google： Spanner

第三条路：
混搭架构。



4 集群架构

集群架构需要考虑的问题：

- 分布式存储+分布式计算——满足海量数据存储和处理的需求，数据不丢失
- 支持ACID——分布式事务特性
- 多点写、多点读——资源利用率高
- HTAP = OLTP + OLAP——多种类计算需求
- 无单点或半数以下节点故障——高可用
- 支持结构化、半结构化、非结构化数据——多种数据存储

PostgreSQL, MySQL, Greenplum, Informix, etc

@那海蓝蓝 Blog : http://blog.163.com/li_hx/

《数据库查询优化器的艺术: 原理解析与SQL性能优化》

Database_xx@126.com

本次分享的Ppt位于:
http://pan.baidu.com/s/1jI6MBg6

THANKS

SequeMedia 盛拓传媒 | ChinaUnix | ITPUB