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

分布式数据库CDS-原理与实践

嘉宾：王义林

议题

- 数据分片概述
- CDS架构及关键特性
- 实践：平滑实施分库分表

数据分片概述

- 遇到容量瓶颈了？

Scale Up

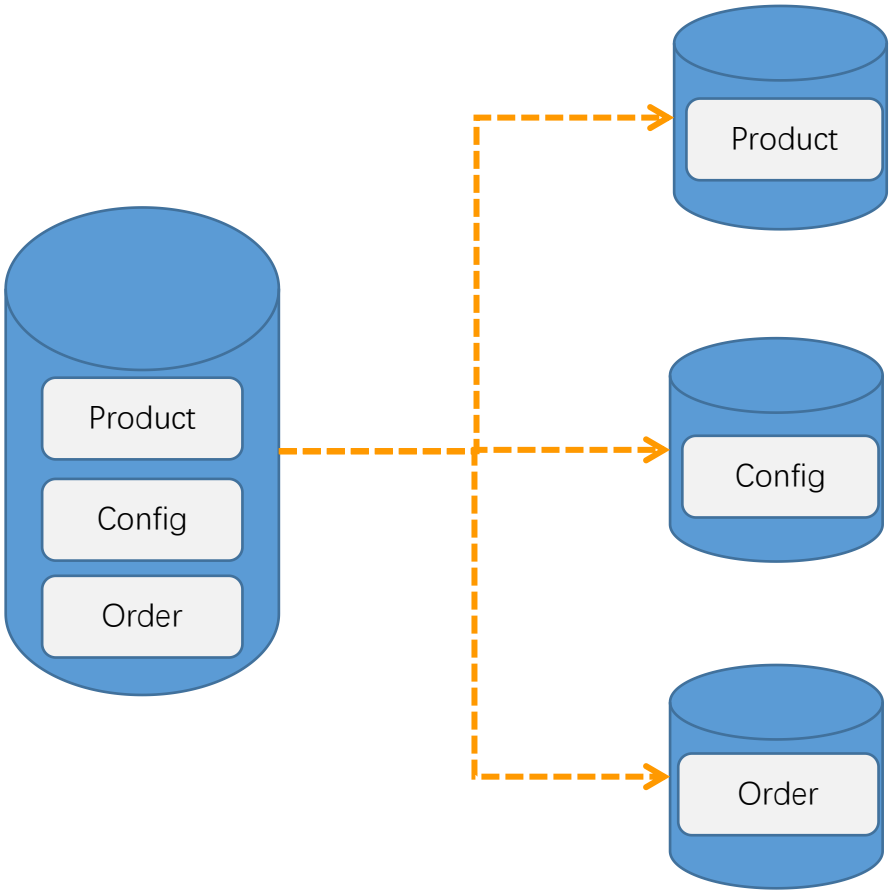
VS

Scale Out

- 集中式
  - ✓ Scale Up
  - ✓ 成本高
  - ✓ 扩展性差，到顶
  - ✓ 易管理
- 分布式
  - ✓ Scale Out
  - ✓ 成本较低
  - ✓ 扩展性好
  - ✓ 较难管理

数据分片概述

- 垂直拆分先行！



- 垂直拆分
  - ✓ 按功能模块拆分，分散数据库压力；
  - ✓ 结合服务化改造，强耦合的系统拆分成多个弱耦合的服务；
  - ✓ 可以规避跨库表之间的join场景；
  - ✓ 单服务下，也避免了分布式事务场景；

数据分片概述

- 读写分离 or 水平拆分 ？

- 读写分离
  - ✓ 数据量在千万级别以内；
  - ✓ 对数据一致性要求较弱，可接受数据延时
  - ✓ 访问量大，写少读多；
- 不用拆分
  - ✓ 数据量小，访问量大，考虑缓存；
- 水平拆分
  - ✓ 数据为流水性质，数据量剧增；
  - ✓ 两组业务数据间依赖关系弱(流水之间)
  - ✓ 访问量大，重要性高，影响可用率；
- 水平拆分 + 读写分离 ？

数据分片概述

- 读写分离实施简单，但需要注意场景

不适用读写分离场景	适用读写分离场景
• 对数据有强一致性要求； • 流水数据，写多； • 依据当前状态做下一步处理的场景；	• 对数据能接受一定延时 • 主数据、配置数据查询 • 运营查询

Tips：

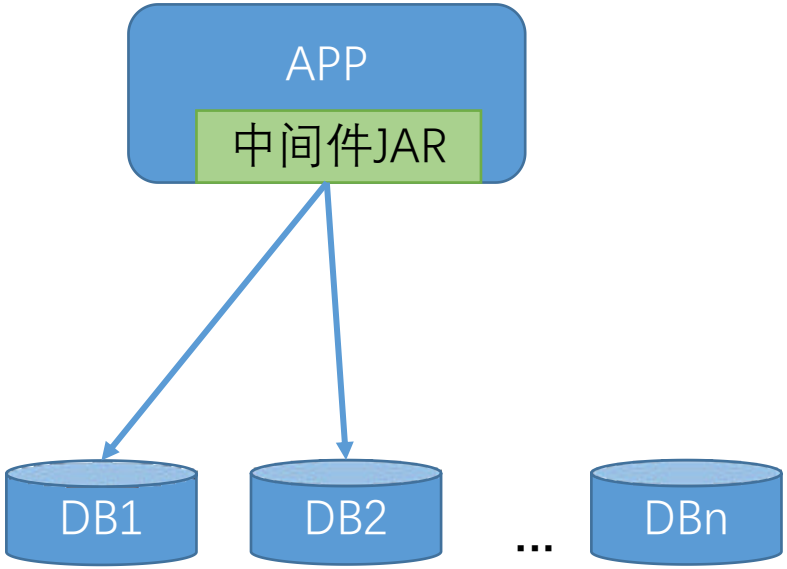
✓ 很多能做读写分离的场景也可以使用缓存

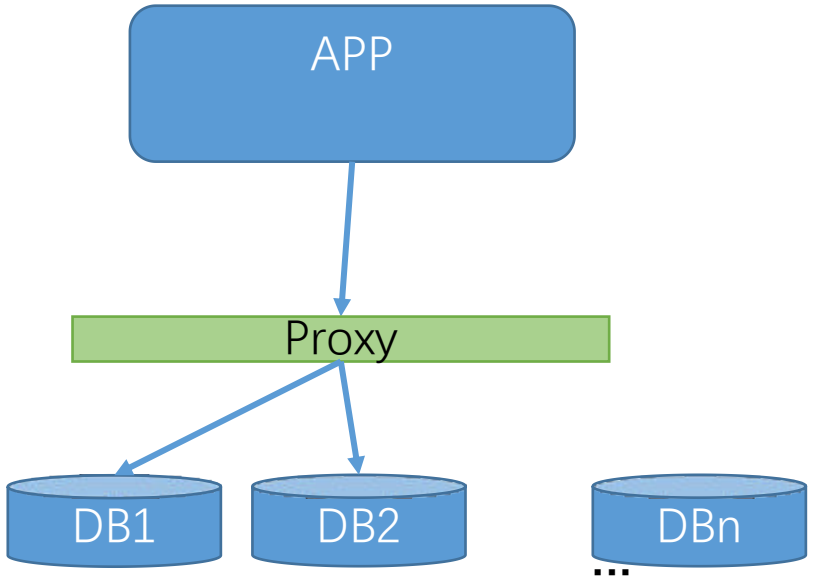
数据分片概述

- 分库分表实施步骤
  - ✓第1步 - 必要性评估
  - ✓第2步 - 配置表处理
  - ✓第3步 - 拆分维度确定
  - ✓第4步 - 分表、分库数量确定
  - ✓第5步 - 事务处理
  - ✓第6步 - 跨分片join处理
  - ✓第7步 - 全局唯一id生成
  - ✓第8步 - 聚合处理

数据分片概述

- 客户端中间件 or 服务端代理层





- 客户端类型
  - ✓ 兼容多种数据库类型
  - ✓ 拓扑结构简单，可用性好，集群间无耦合
  - ✓ JAR包嵌入，对应用有细微影响

- 服务端(代理)类型
  - ✓ 支持多种开发语言；应用透明；
  - ✓ 可用性稍差，集群间耦合
  - ✓ 针对特定数据库；

议题

- 数据分片概述
- CDS架构及关键特性
- 实践：平滑实施分库分表

简介

- 基于京东金融业务特点、发展现状

简称	• Completed Database Sharding
功能	• 数据分片、读写分离，海量数据存取； • 运维控制台；数据迁移、扩容平台； • 离线准实时查询、统计；
特点	• 兼容JDBC，接入成本低； • 支持MySQL/Oracle/SQL Server； • 形成数据分片运维体系的生态圈；

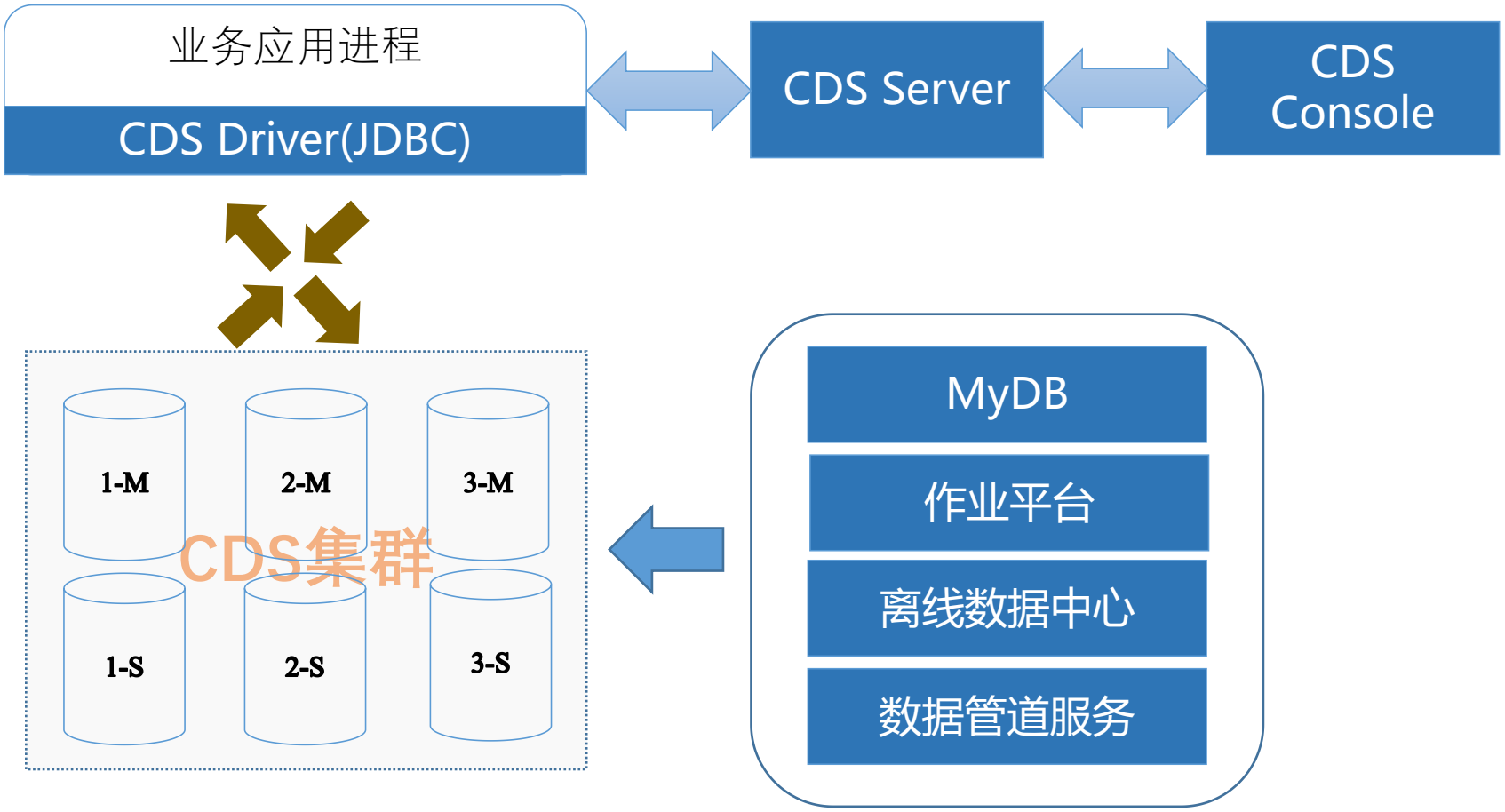
版本演进

分库分表，读写分离，Failover。

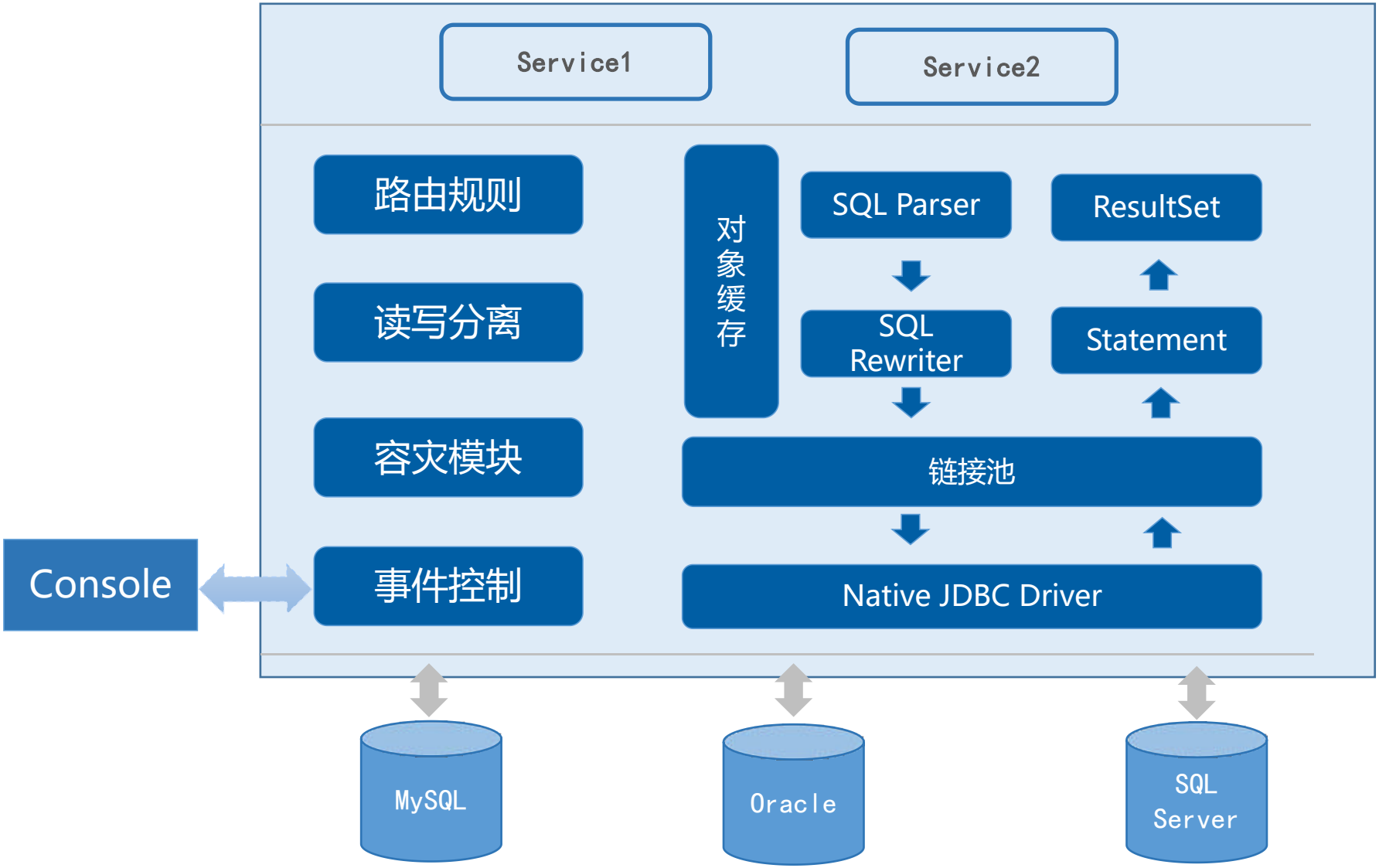
运维监控，自定义路由，级联路由，重读双写，集群配置在线推送，弱XA事务支持，作业平台，离线数据集中平台。

优化sql解析模块，宏定义语法，groovy路由算法，数据管道服务，分布式事务服务。

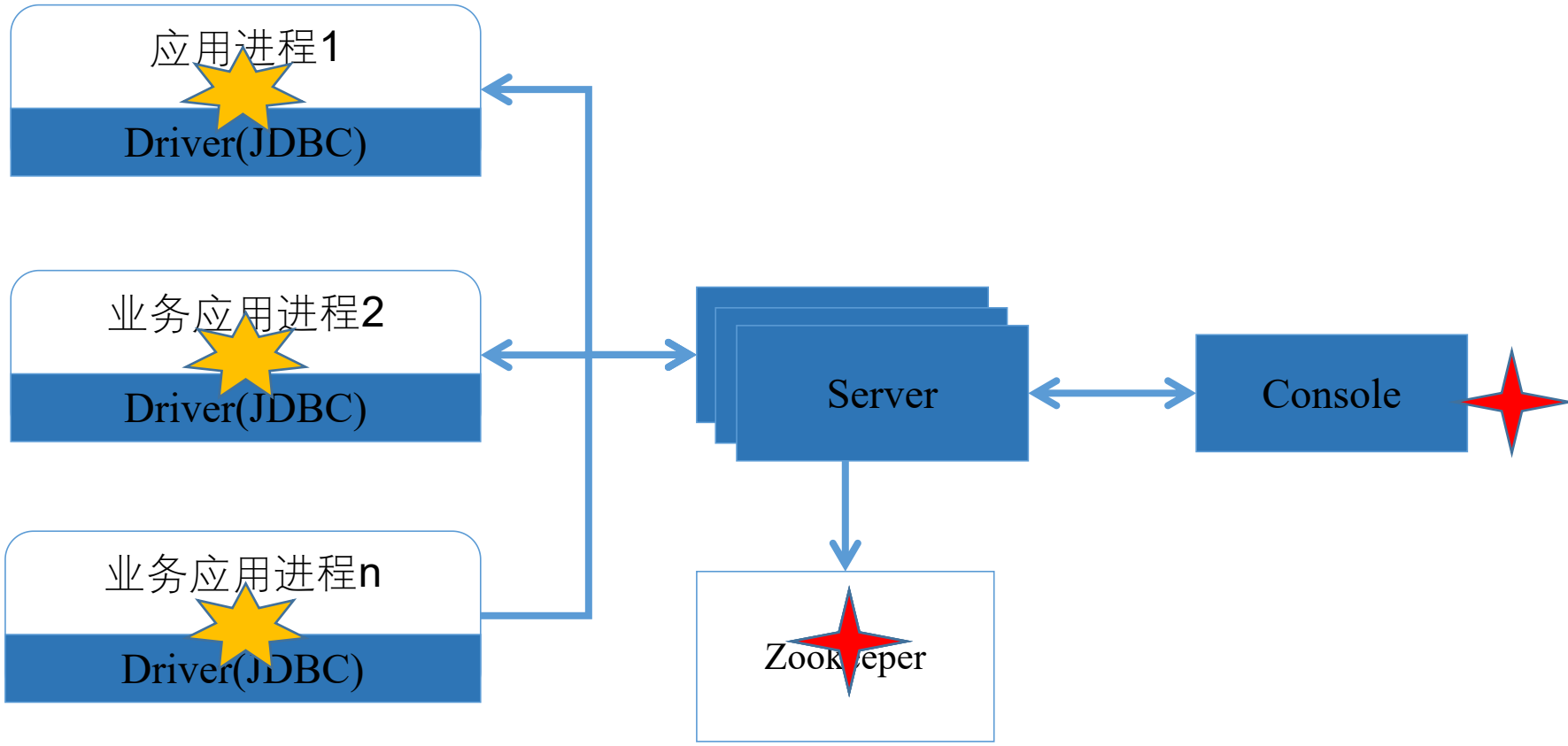
系统架构


业务应用进程
CDS Driver(JDBC)
CDS Server
CDS Console
1-M
2-M
3-M
CDS集群
1-S
2-S
3-S
MyDB
作业平台
离线数据中心
数据管道服务

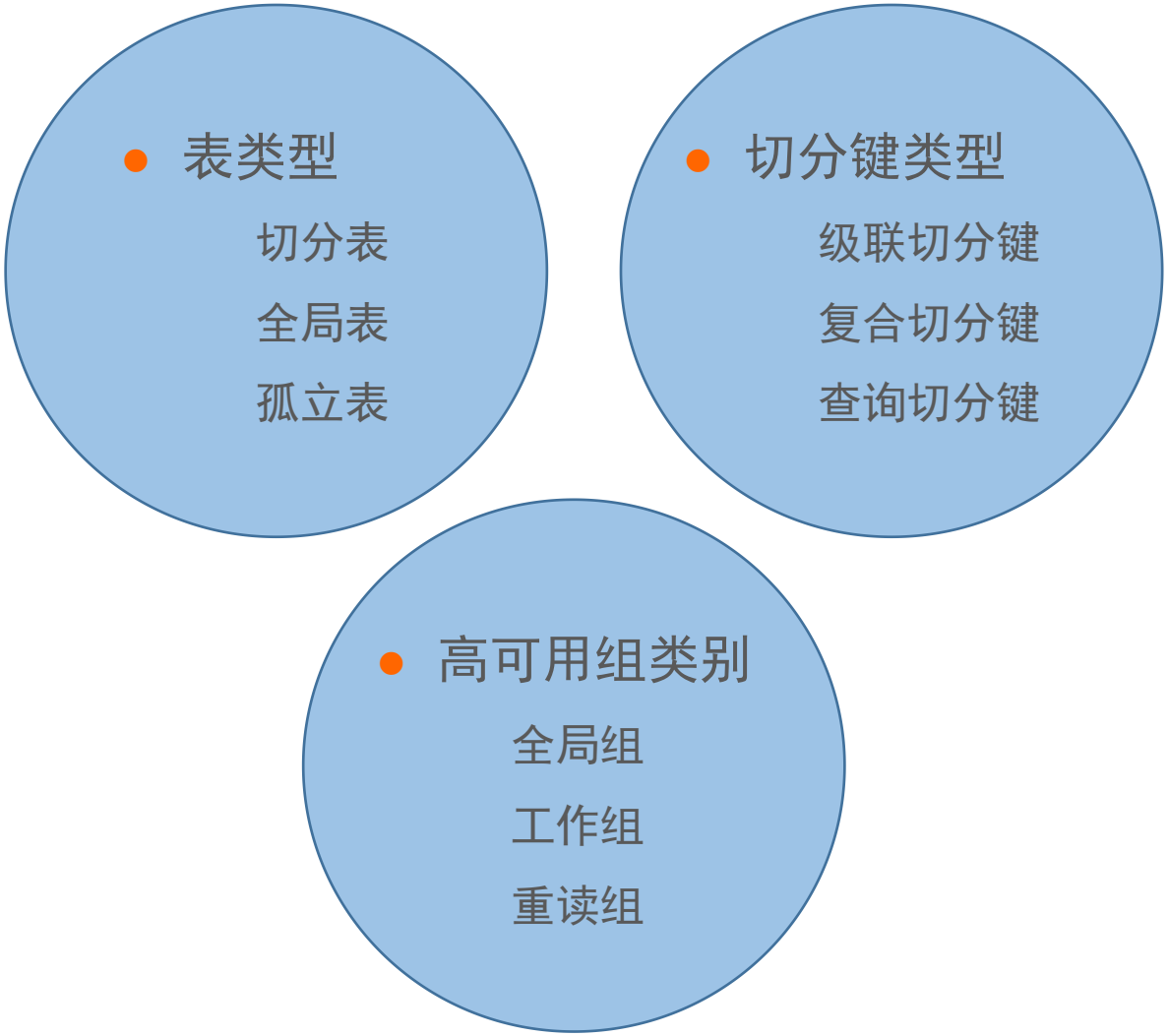
01 中间件：驱动架构


Service1
Service2
路由规则
读写分离
容灾模块
事件控制
对象缓存
SQL Parser
SQL Rewriter
ResultSet
Statement
链接池
Native JDBC Driver
Console
MySQL
Oracle
SQL Server

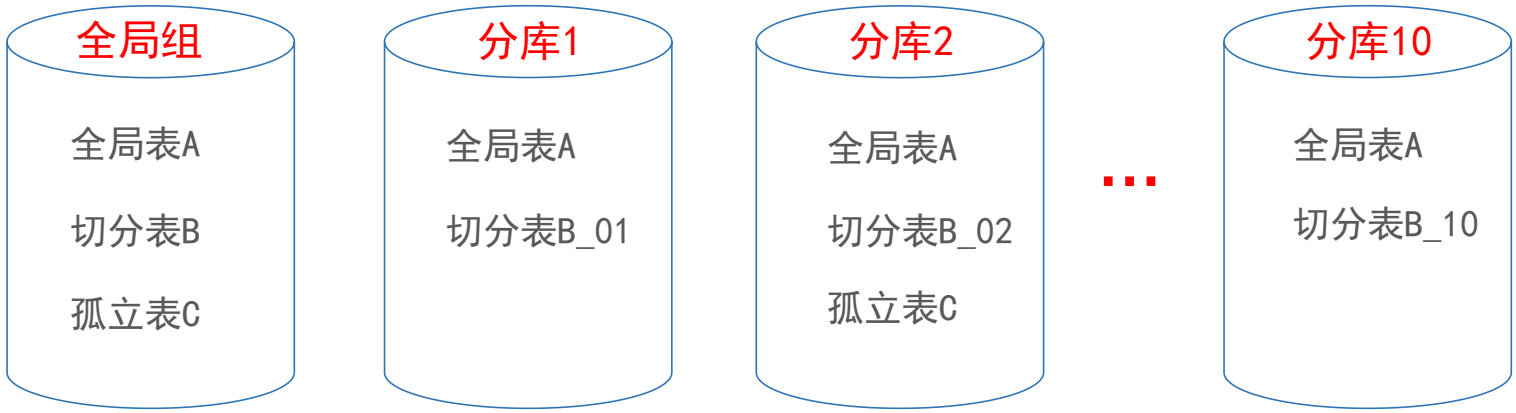
事件控制过程


应用进程1
Driver(JDBC)
业务应用进程2
Driver(JDBC)
业务应用进程n
Driver(JDBC)
Server
Console
ZookKeper

CDS集群元素



02. CDS集群结构



CDS集群

全局表：每个工作组都存在相同的数据备份(比如. 城市、类别）

切分表：插入数据时根据切分键插入到不同分表（比如. 业务流水表）

孤立表：可以存放在任意工作组包含数据表（比如. 配置表）

CDS关键特性(一)

- 兼容JDBC标准
- 支持MySQL/Oracle/SQL Server
- 内嵌高性能连接池组件
- 读写分离
  - ✓权重定义
  - ✓故障读节点自动摘除及恢复
- 水平拆分
  - ✓Range、Hash、List、支持自定义路由规则
  - ✓全局id
  - ✓常见聚合函数
  - ✓多级路由

CDS特性：兼容JDBC标准


```
<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
   <property name="driverClassName" value="com.wangyin.cds.driver.CdsDriver"/>
   <property name="url" value= "jdbc:cds://127.0.0.1:8088/CardOrder " />
   <property name="username" value="cardorder" />
   <property name="password" value="xxxxx" />
   <property name="connectionProperties">
        <props>
          <prop key="dbpool.class">com.wangyin.commons.cp.WangyinCPDataSource</prop>
          <prop key="dbpool.minConnections">2</prop>
          <prop key="dbpool.maxConnections">50</prop>
          <prop key="wycds.sql">true</prop>
             <prop key="urlparam.allowMultiQueries">true</prop>
        </props>
   </property>
</bean>
```

CDS关键特性(一)

- 兼容JDBC标准
- 支持MySQL/Oracle/SQL Server
- 内嵌高性能连接池组件
- 读写分离
  - ✓权重定义
  - ✓故障读节点自动摘除及恢复
- 水平拆分
  - ✓Range、Hash、List、支持自定义路由规则
  - ✓全局id
  - ✓常见聚合函数
  - ✓多级路由

CDS关键特性(二)

- 图形化控制台
  - ✓集群配置
  - ✓管理及监控
  - ✓事件控制
- 工具集
  - ✓MyDB，统一查询路口，分库分表对开发人员透明；
  - ✓作业平台，内置存量数据迁移和增量数据同步作业；
  - ✓离线数据中心；
  - ✓数据管道服务 ；
  - ✓分布式事务服务；

议题

- 数据分片概述
- CDS架构及关键特性
- 实践：平滑实施分库分表

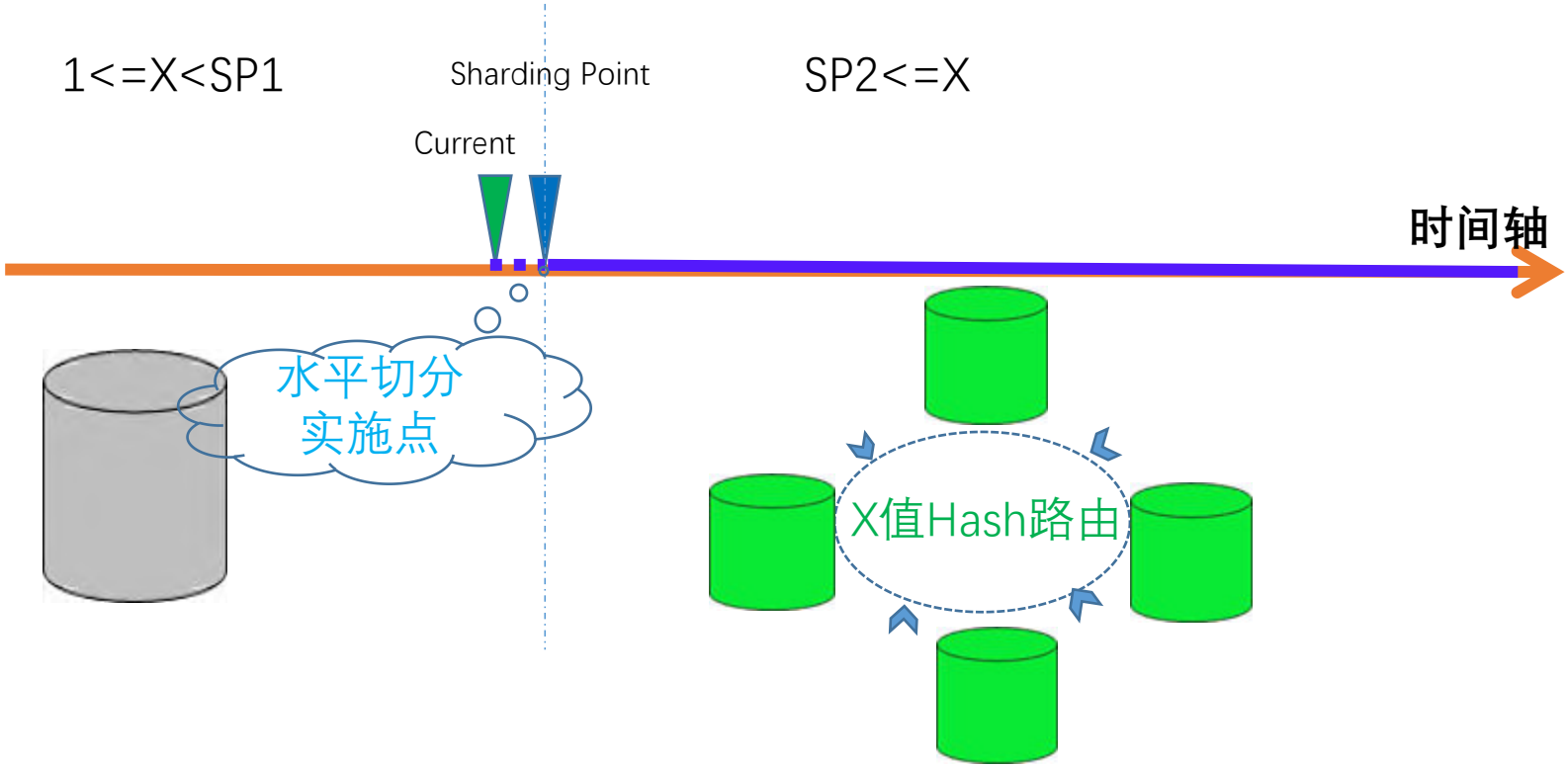
实践：平滑实施分库分表

- 平滑实施分库分表

  - 采用1+N方式；
  - 模拟MySQL主从切换方式；

1+N：介绍

- 切分列值在时间维度是增序的，范围路由；
- 支持多级路由: 一级路由提供横向扩展的能力，二级路由扩展计算能力



1+N：满足前提

- 订单号前8位提供分区表分区键功能；
- 系统版本位提供一级路由功能，分库分表位提供二级路由功能；
- 分库分表位+8位sequence保证订单号全局唯一性，单库每天订单容量1亿；
- 根据订单号进行水平切分

位置	8位日期	数据版本位	系统版本位	3位业务标识位	1位新应用标示位	3位分库分表位	8位sequence
例	20150101	0	0	001	1	055	00000001

分区键

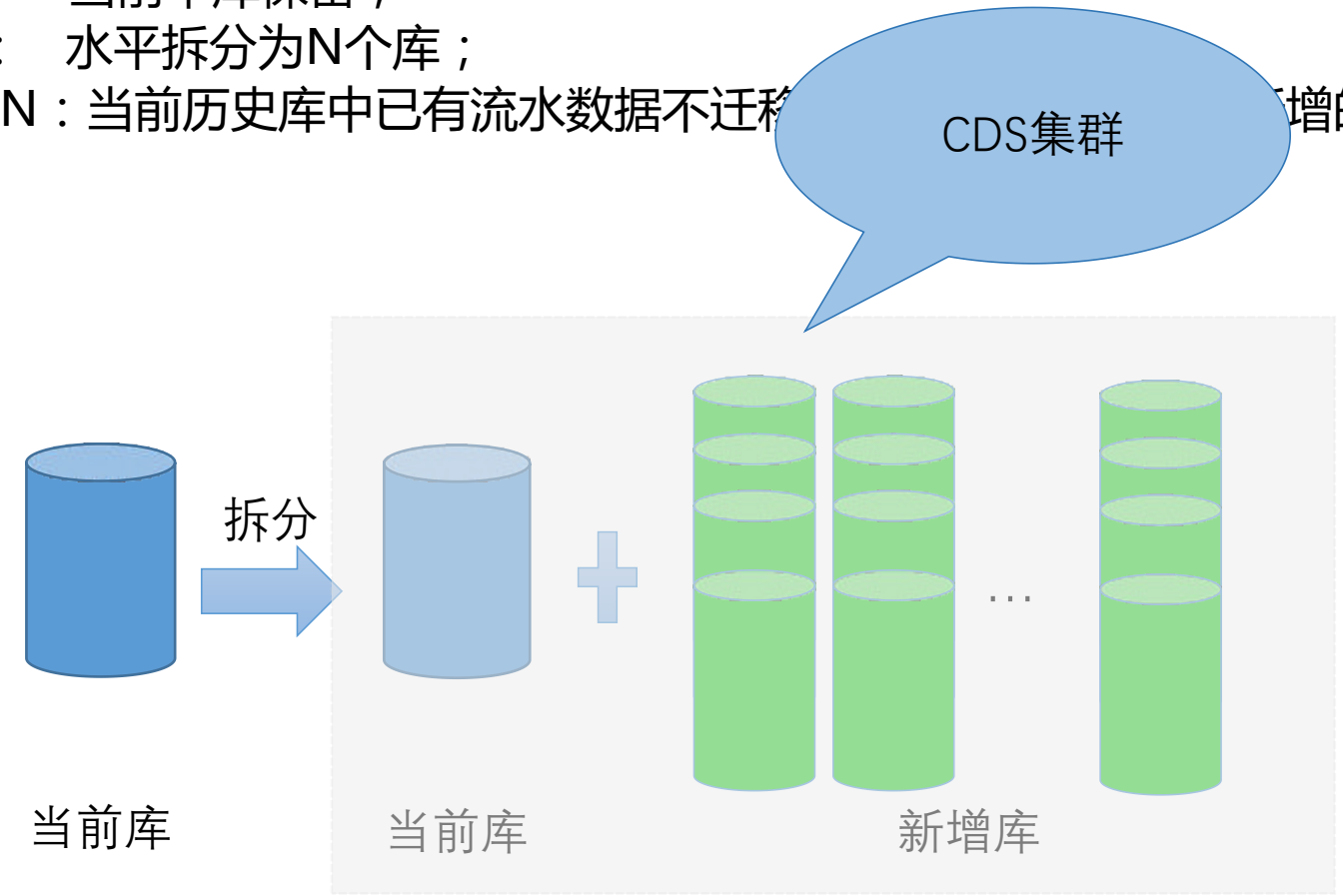
一级路由

二级路由

固定长度，每天单库订单容量1亿

1+N：关键设计

- 采用1+N方式
  - ✓1： 当前单库保留；
  - ✓N： 水平拆分为N个库；
  - ✓1+N：当前历史库中已有流水数据不迁移，新增的N台库中。



1+N：关键设计

- 当前库
  - ✓ 历史订单数据、孤立表数据
- 新增库
  - ✓ 新增订单数据、外部订单号和内部订单号的映射表
- 优点
  - ✓ 当前库无新正单数据进入,解决空间压力，过一段时间后当前库可直接作为历史库
  - ✓ 新数据均匀进入新增库，历史数据不用迁移至新增库
  - ✓ 当前库和新增库均能提供读写服务，对应用功能无影响

1+N：两级路由

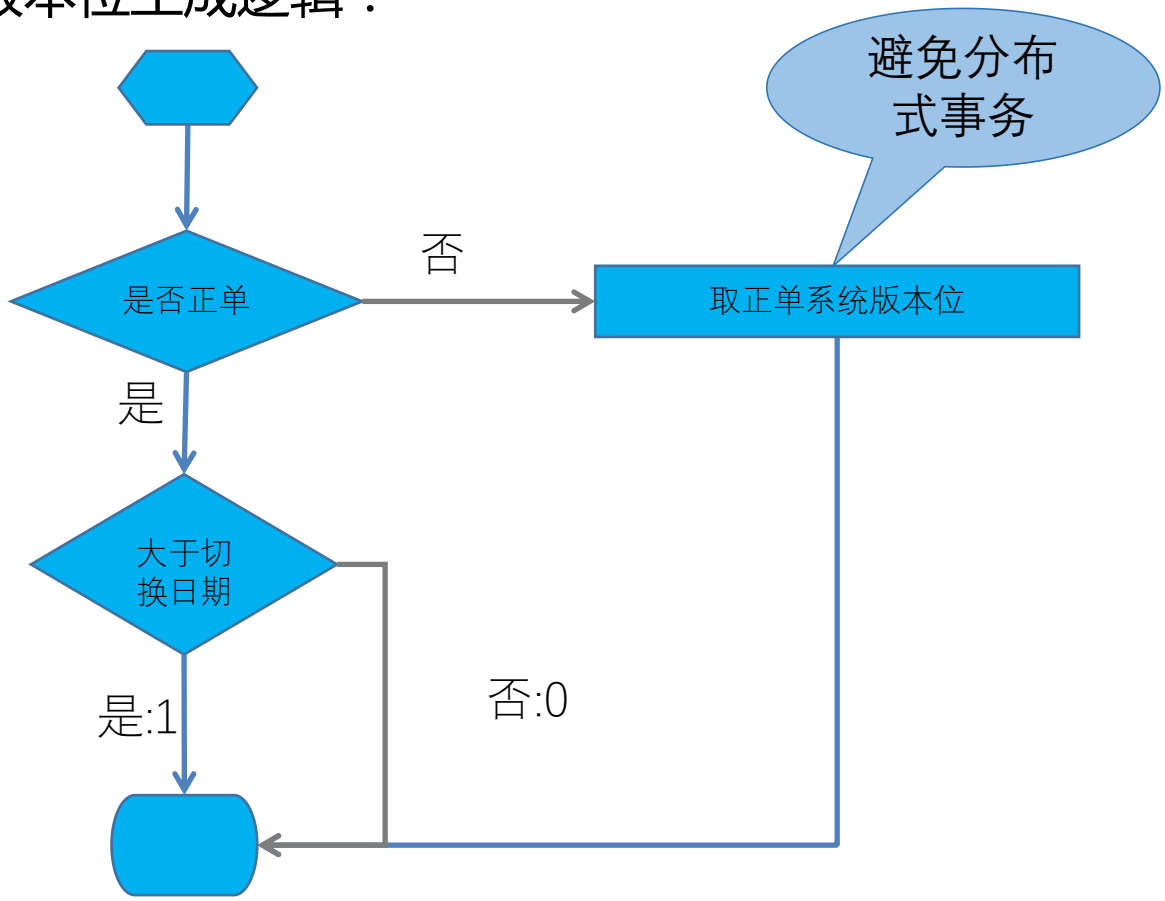
- 主流水表第一级路由：系统版本位
  - ✓避免一级路由引发正单和退款间分布式事务，正单退款订单需要在同一库中；
  - ✓考虑不停机发布，一级路由需延时生效；
  - ✓新流水数据进入新库，当前库上无新正单记录写入；

		日期条件	系统版本位	路由所在库
正单	老正单	--	0	当前库
	新正单	<某日期	0	当前库
		>=某日期	1	新库
退款	老退款	--	0	当前库
	新退款	--	取正单位→0	当前库
		--	取正单位→1	新库

注：某日期为晚于应用上线时间点的日期

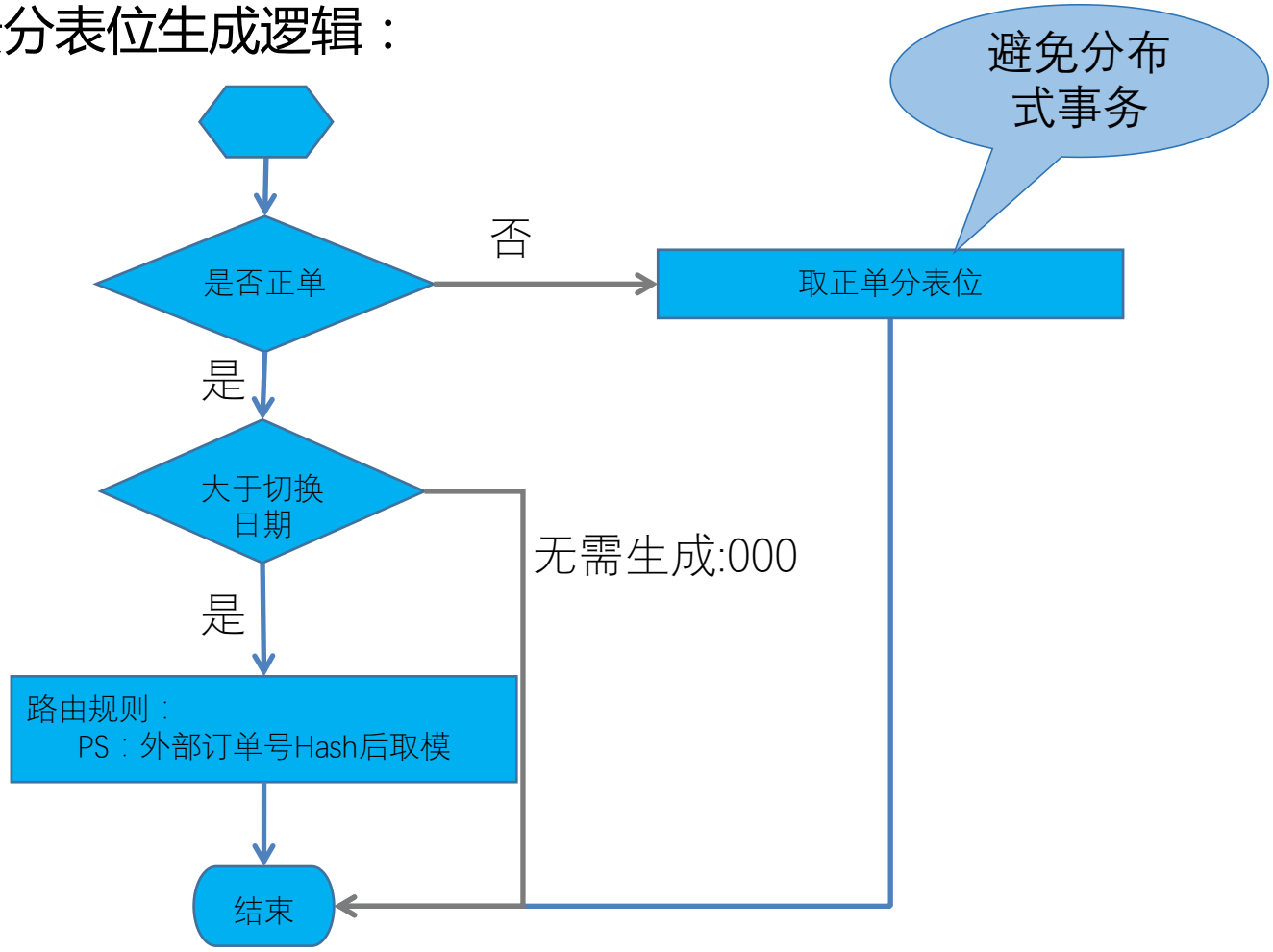
1+N：两级路由

- 应用负责系统版本位生成逻辑：



1+N：两级路由

- 中间件负责分表位生成逻辑：

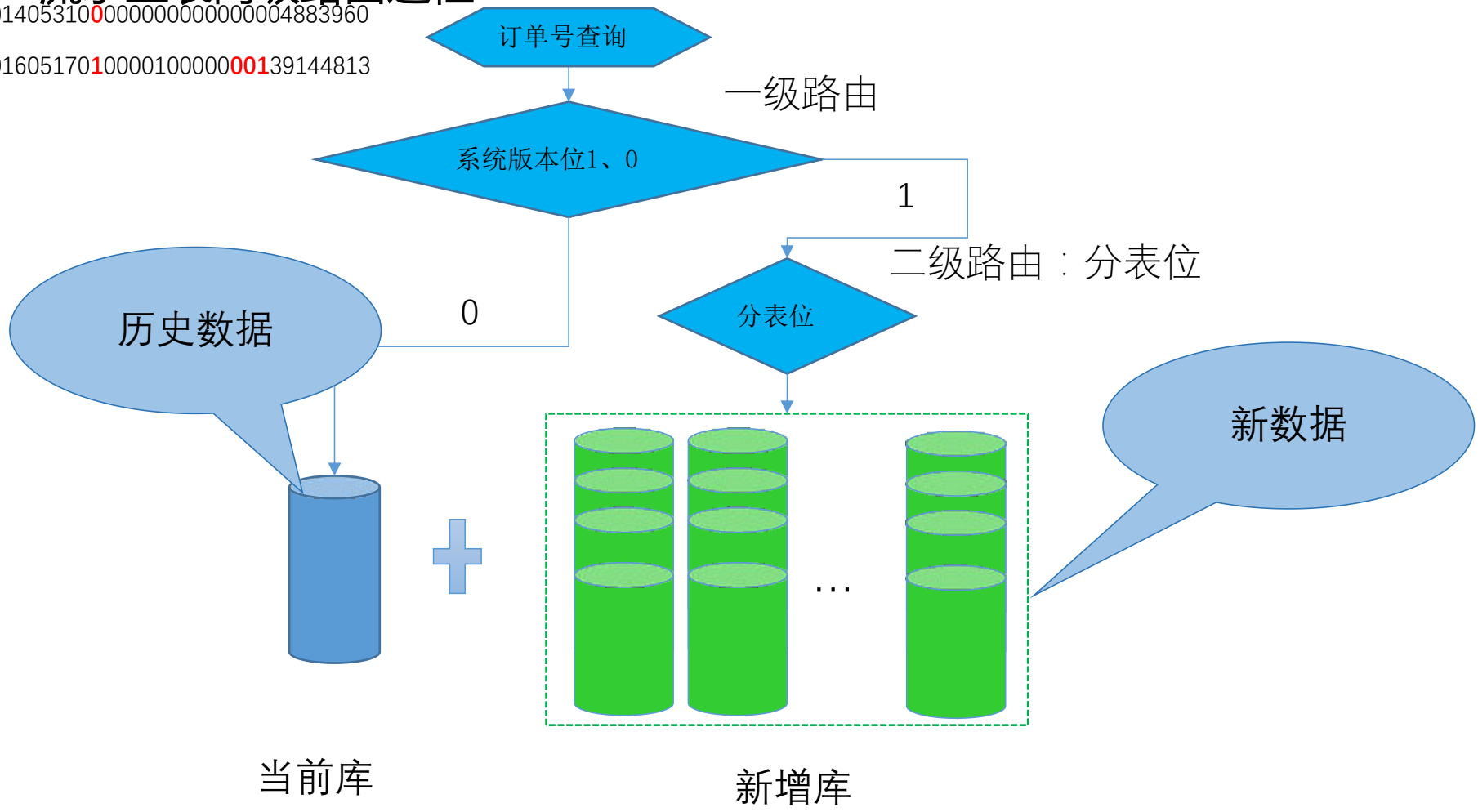


1+N：两级路由

- 流水主表两级路由过程

2014053100000000000000004883960

2016051701000010000001139144813



1+N：不停机关键

- **不停机方案的关键在于解决升级过程新老版本应用并存时，延时生效时间点前完成所有应用发布上线：**

✓增量映射数据的查询、写入→ 重查、双写

✓新增流水数据入库的精确时间控制→系统版本位值延时生效

✓防止资损风险→停退款服务

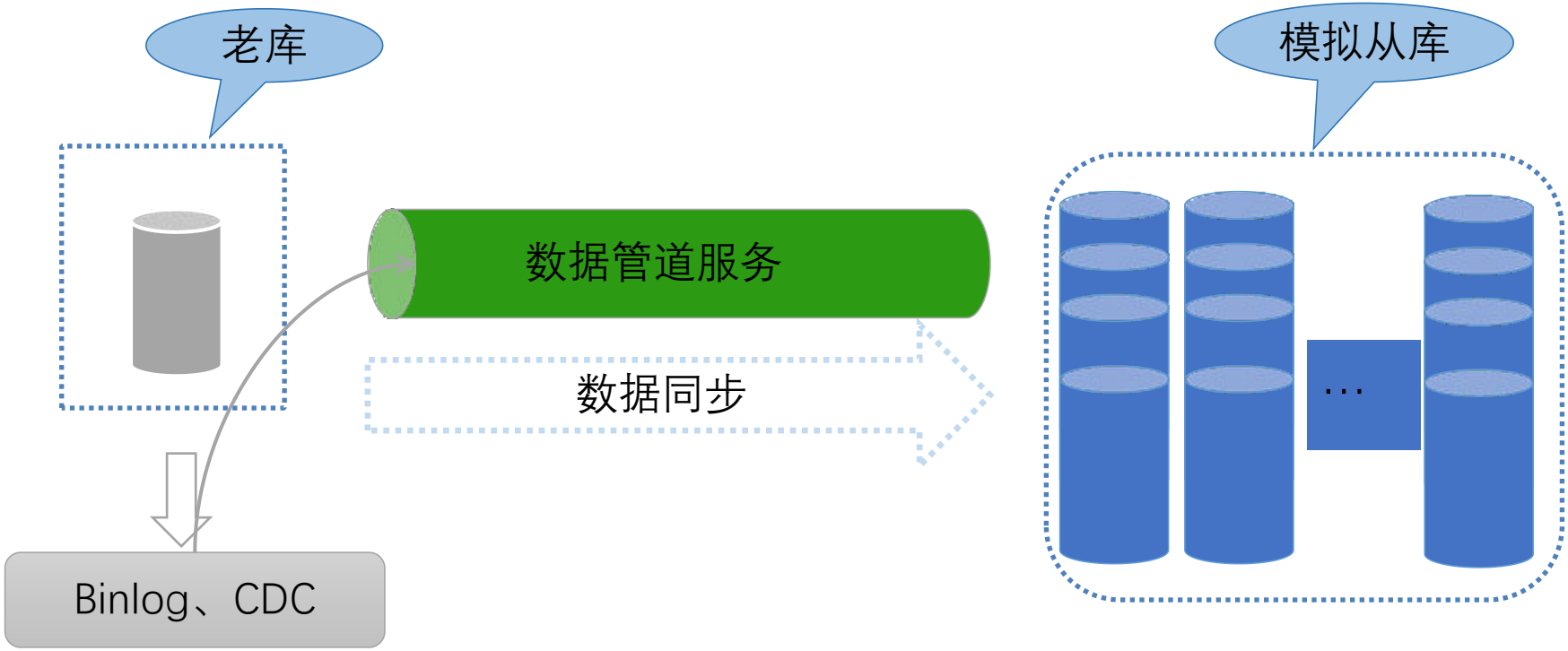
1+N：步骤小结

- Step1. 全量映射数据迁移;
- Step2. 开始应用发布上线，新老应用并行阶段；
- Step3. 老应用没有变化，新应用对映射数据实现重查双写，新应用将流水数据写入当前库
- Step4. 全部完成应用升级后做增量映射数据迁移，此时所有映射数据均在新库存在
- Step5. 第一次推送切换时间点，流水数据一级路由的系统版本位置根据时间自动切换到新增库；
- Step6. 第二次推送新路由规则，实现去除新应用对当前库映射数据的查询和写入，由重查双写改为单查单写；

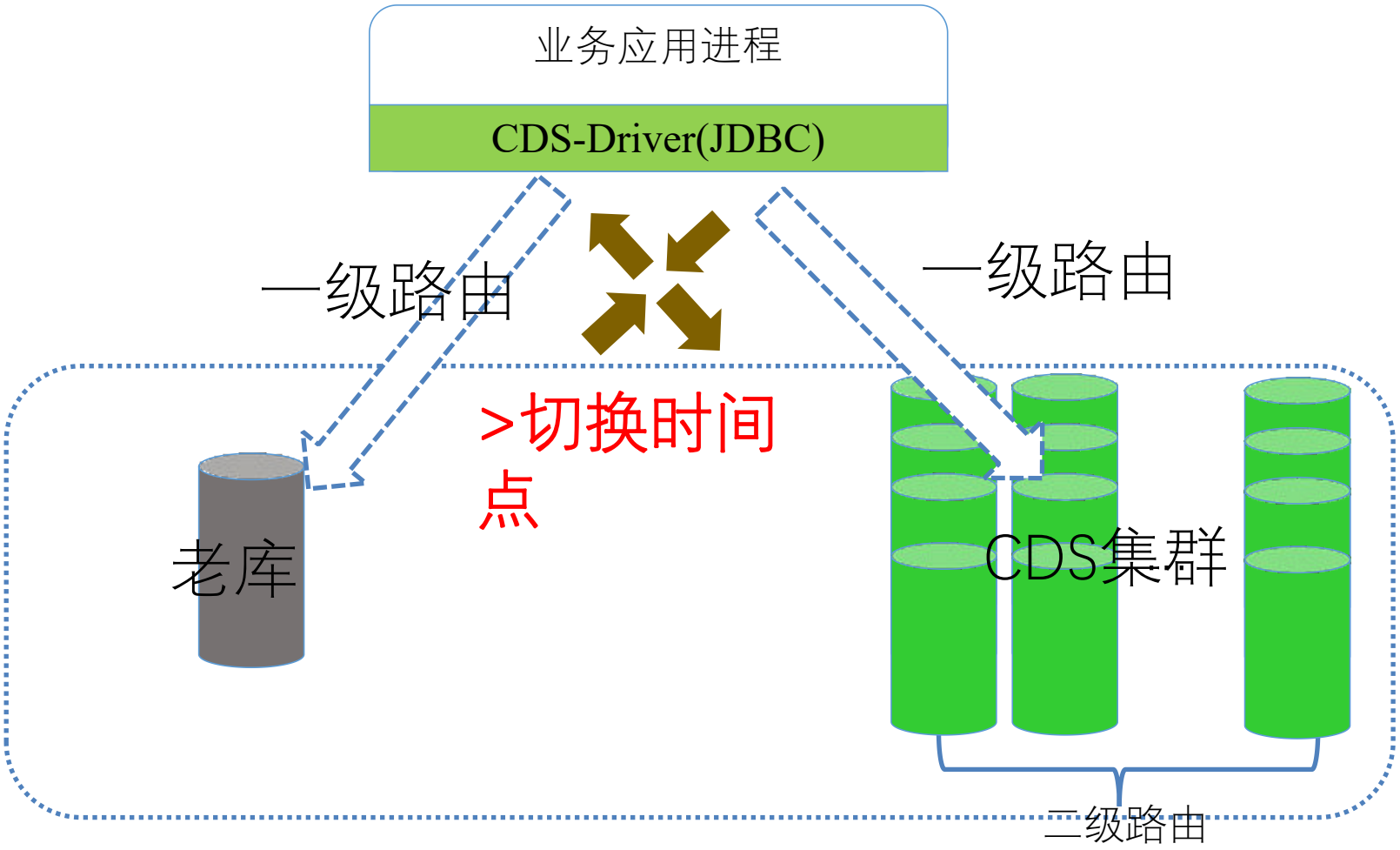
主从切换：概要

- 并不是所有切分列值都是在时间维度上增序的；
- 省去双写过程，减少切换边界；
- 通过Binlog解析, CDS集群模拟成老库的从库；
- 切换过程就像mysql主从切换；

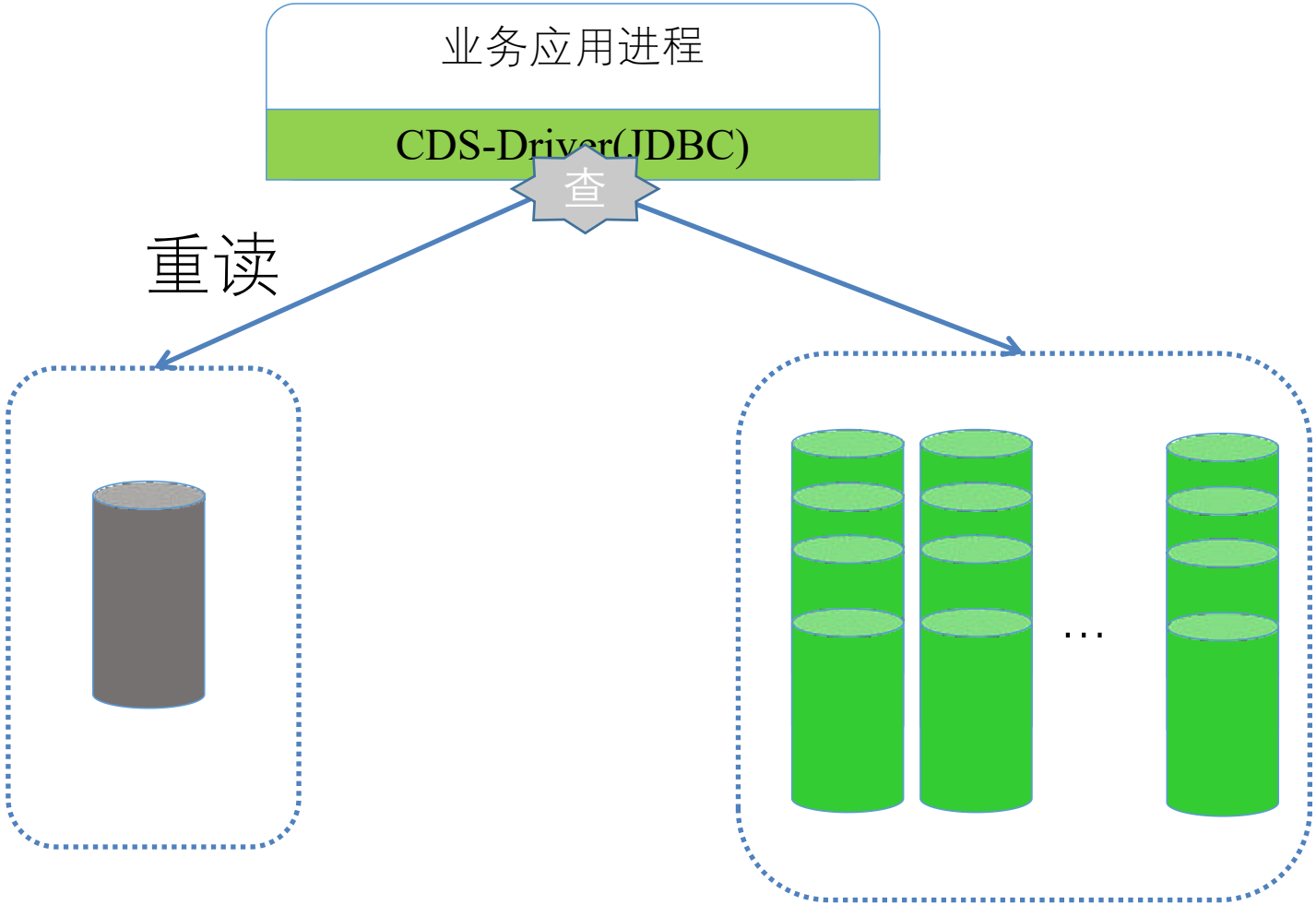
主从切换：增量同步



模拟从库：切换

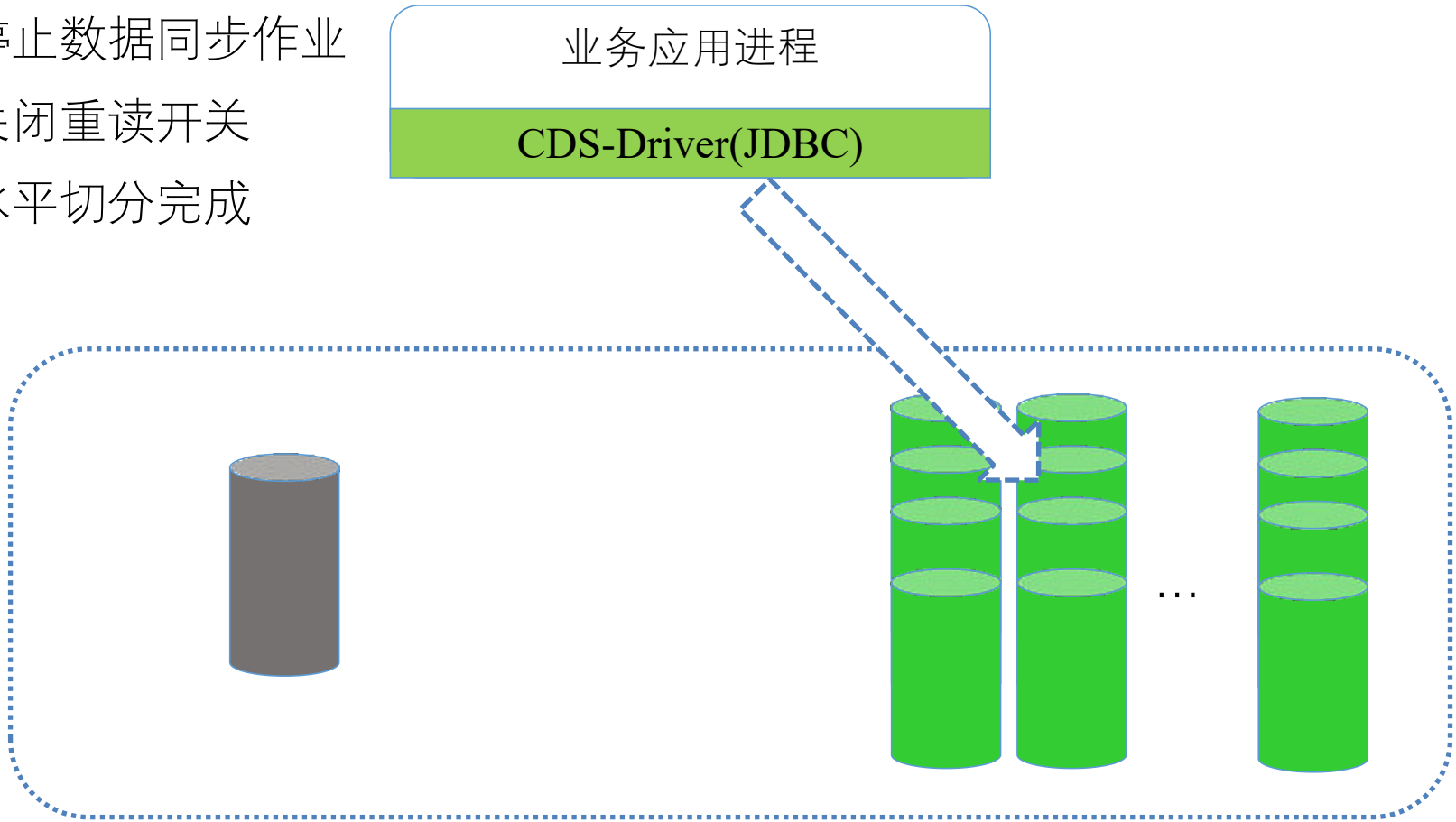


模拟从库：重读



模拟从库：在线控制

1、停止数据同步作业
2、关闭重读开关
3、水平切分完成



主从切换：步骤小结

- Step1. 做1次全量数据迁移，迁移前记录binlog当前位置；
- Step2.启动增量数据同步作业；
- Step3. 应用灰度上线（数据源是CDS数据源，一级路由还是访问老库）；
- Step4. 等待增量数据同步作业，执行速度追上最新提交的事务；
- Step5. 在CDS控制台中，推送切换时间（一个大于当前时间的时间点）；
- Step6. 切换时间生效后，一级路由转为访问分库分表集群，二级路由Hash分片；
- Step7. 在CDS控制台中，关闭增量数据同步作业，推送关闭重读命令；
- Step8. 完成水平分库分表上线。

平滑拆分：小结

- 支持重读、双写机制；
- 配置在线推送生效；
- 多级路由功能支持；
- 作业平台：存量数据迁移；
- 数据管道服务支持；

团队介绍

- 分布式数据库各部件基本功能完成；
- 正在研发数据管道平台3.0；
- 人手比较紧张……

SDCC 2016

中国软件开发者大会

SOFTWARE DEVELOPER CONFERENCE CHINA

谢 谢！