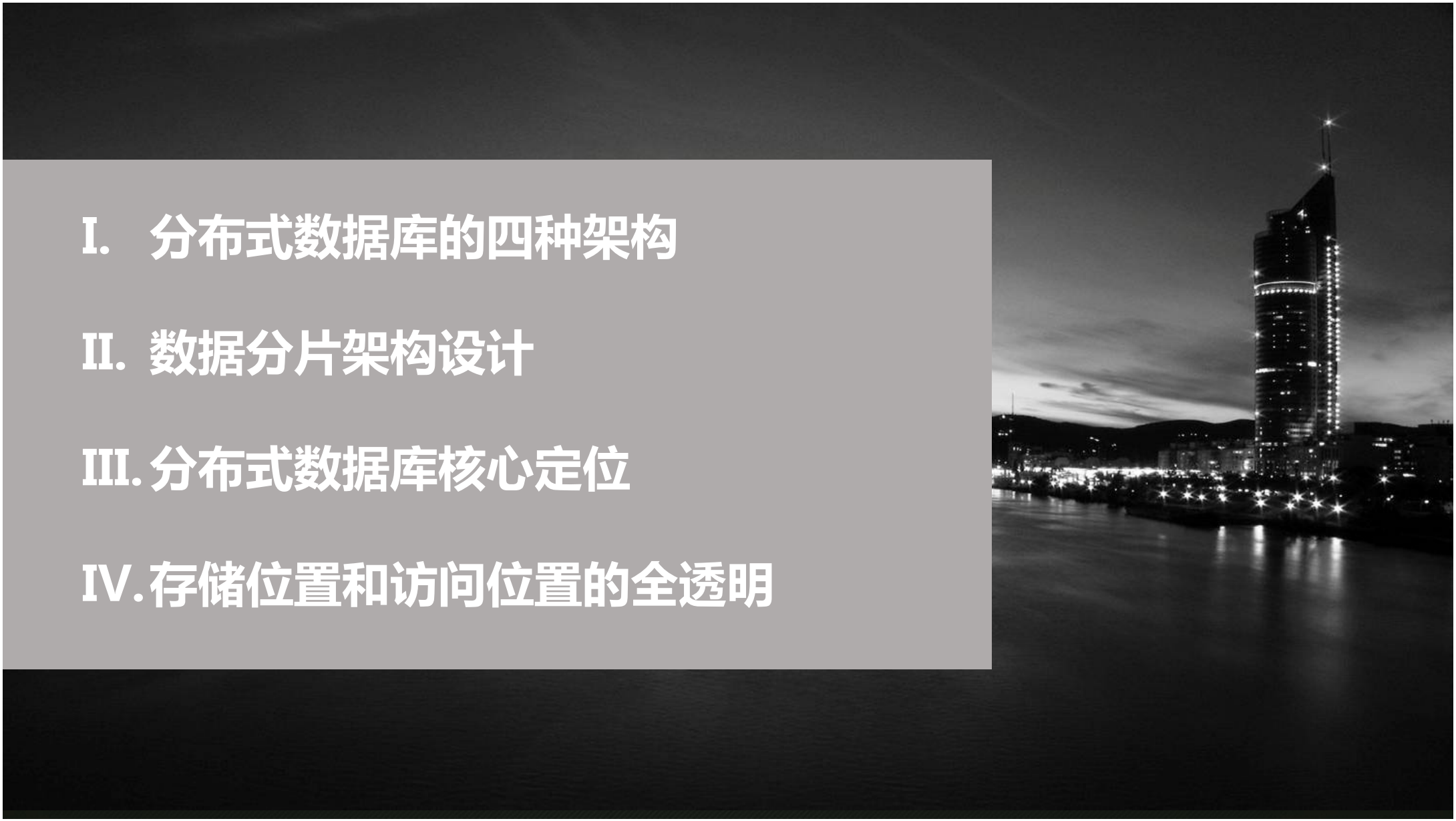
DTCC

2017第八届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2017

分布式数据库的架构与分片设计

热璞科技 金官丁



I. 分布式数据库的四种架构

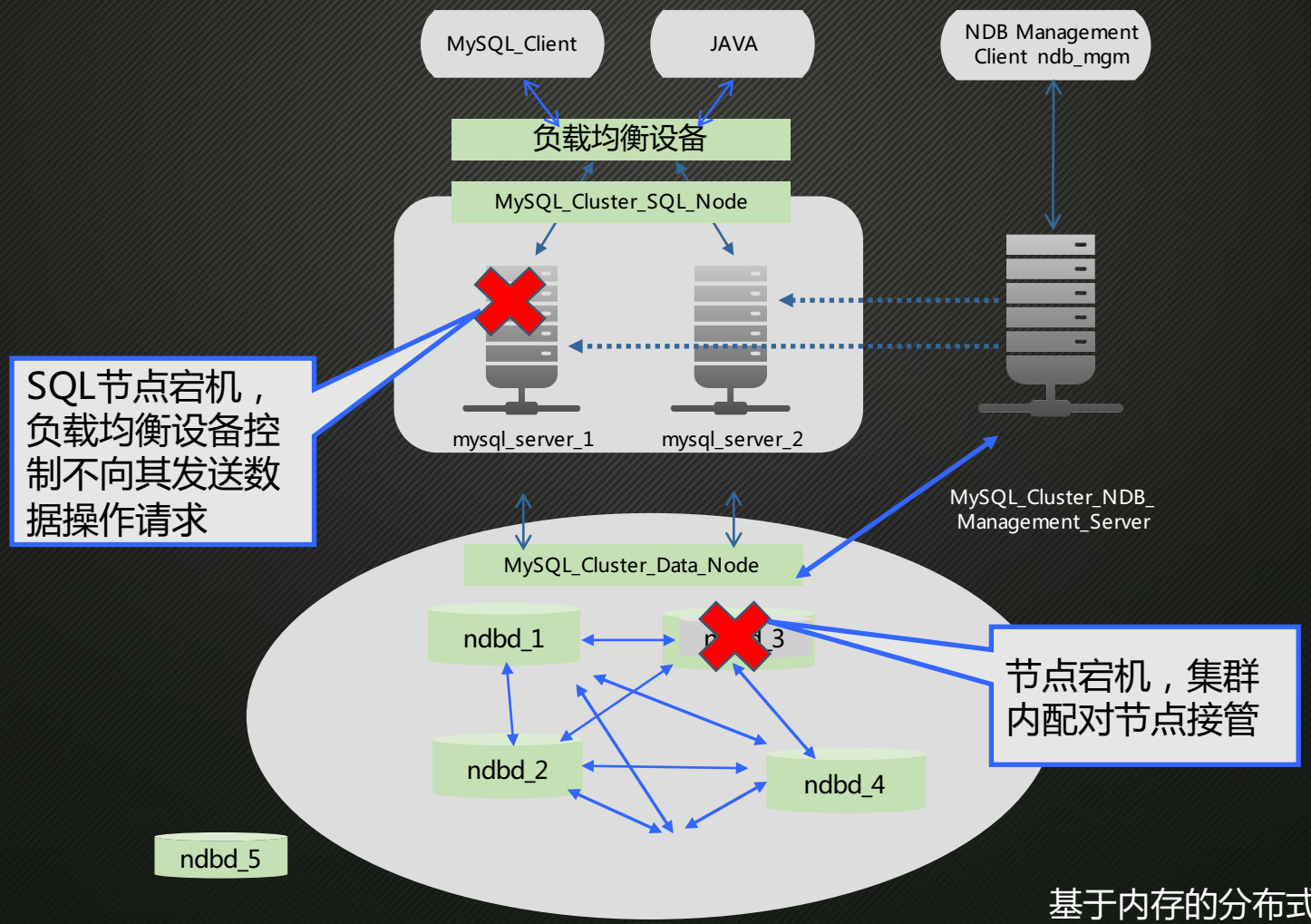
II. 数据分片架构设计

III. 分布式数据库核心定位

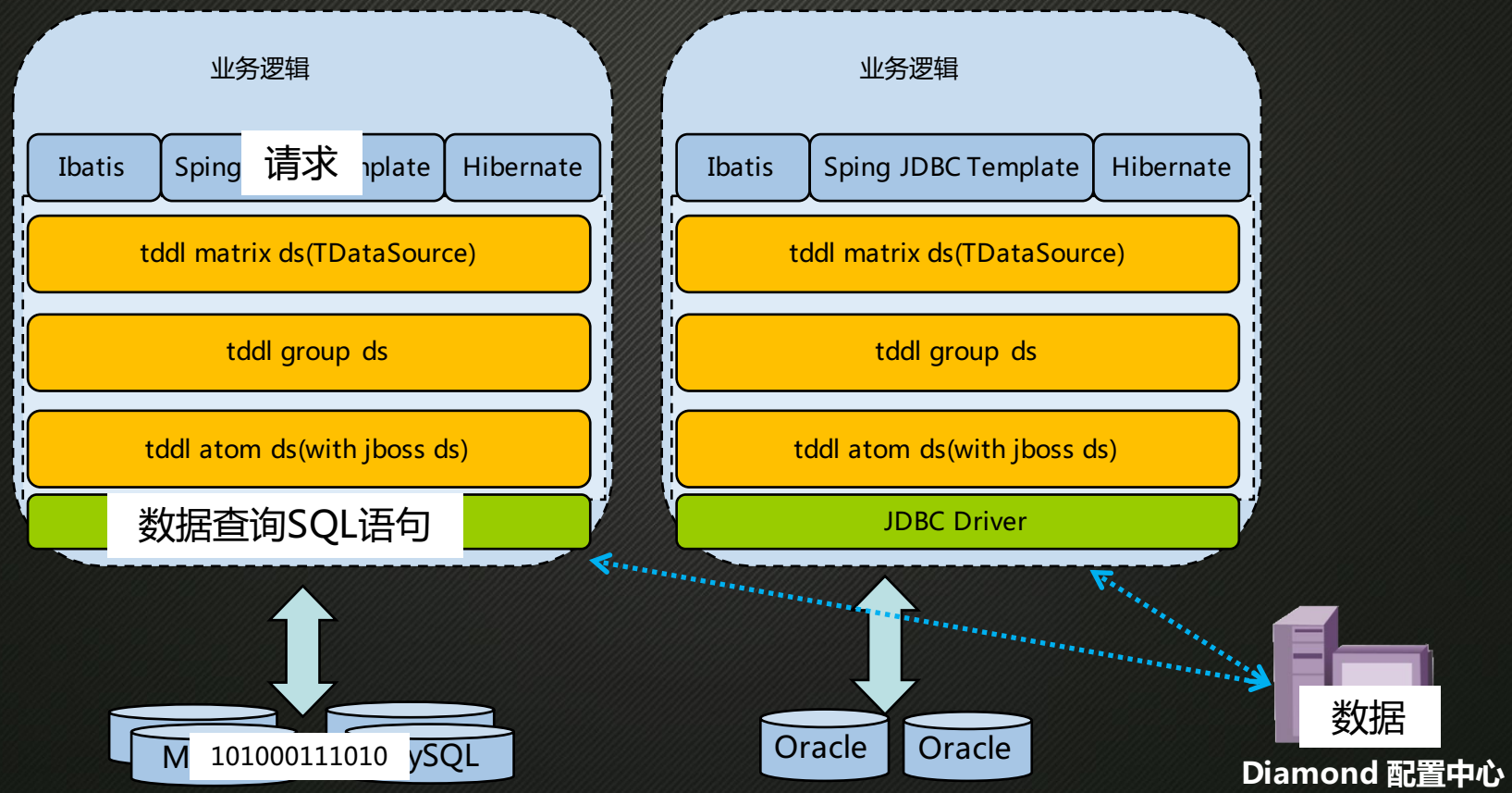
IV. 存储位置和访问位置的全透明

分布式数据库的四种架构

基于内存的分布式数据库

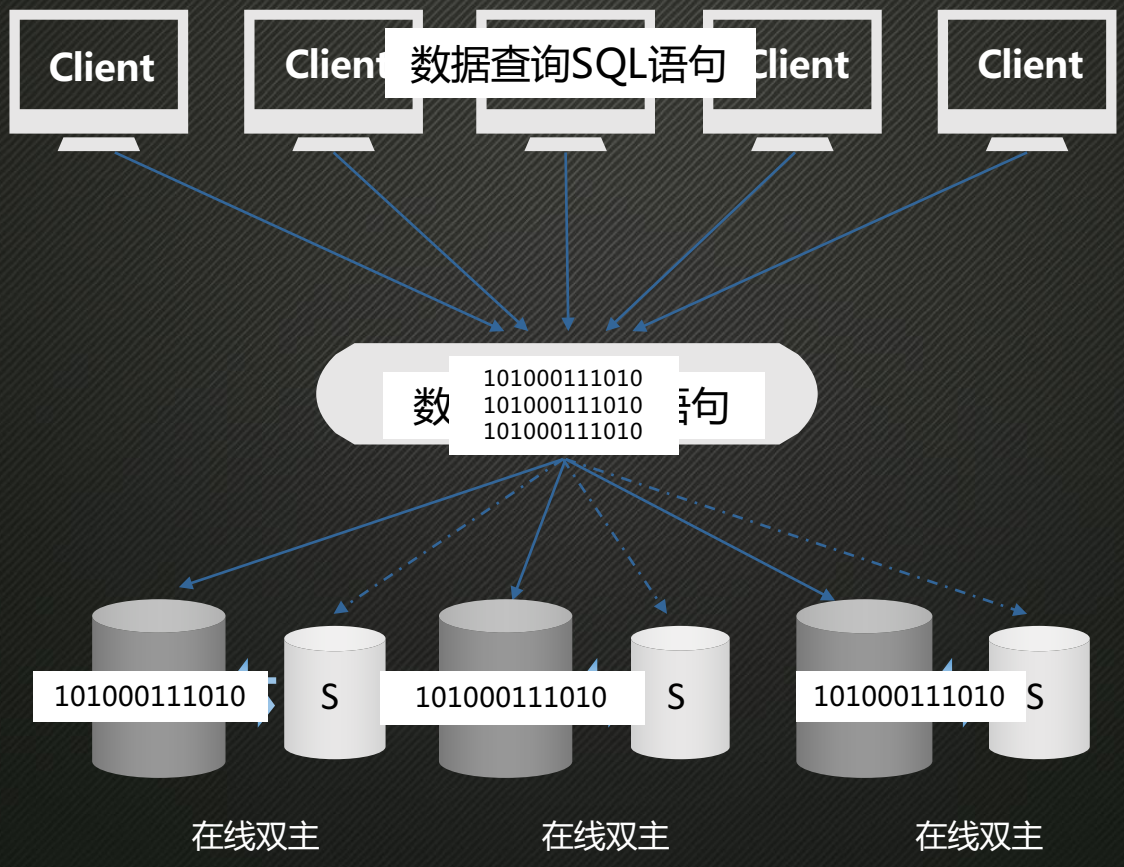


驱动模式的分布式数据库



数据服务的JDBC驱动模式

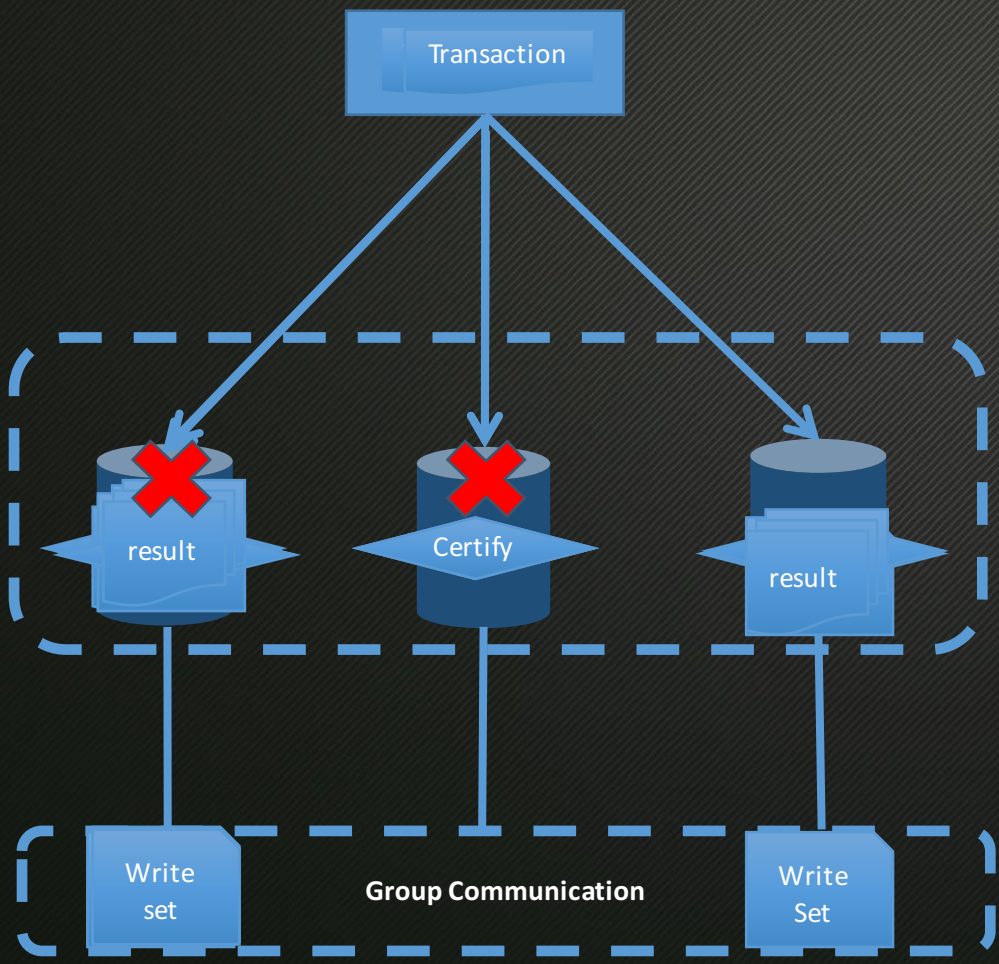
代理模式的分布式数据库



数据服务的代理模式

驱动模式 VS 代理模式

维度项	数据服务的JDBC驱动模式（TDDL为例）	数据服务的代理模式（HotDB为例）
单个数据库维度的应用连接数	单个应用服务创建的连接的数量 * 应用服务个数	HotDB服务端：单个应用服务创建的连接的数量 * 应用服务个数 数据库服务端：以MySQL数据库TPS/QPS最佳吞吐量128-256个为准
单个应用维度的数据库连接数	单个数据库连接的数量 * 数据分片数量	单个数据库连接的数量
数据服务的软件版本升级	每个应用服务的程序都须更新和重新启动，且须停机维护	只须更新2个HotDB服务程序，且无需停机维护
应用服务程序的研发语言支持	仅限JAVA语言	无限制
数据服务的高可用	无	内部机制
数据服务的功能支持	跨数据库的数据操作均不支持，须应用程序自行处理	除存储过程、自定义函数、视图和子查询外，都支持



MySQL Group Replication（实时同步）：

- 程序访问集群任意一个节点（简称：写节点）
- 事务数据发送到每个节点并进行验证
- 被访问节点（简称：写节点）返回事务执行结果
- 其余独立节点提交或回滚事务

MySQL Group Replication（实时同步）：

- 节点宕机，集群进行选举
- 异常节点退出集群
- 若异常节点为访问点（简称：写节点），则切换访问源
- 高可用，99.999%
- 强一致性，真正意义上的零数据丢失，切换时间可控制毫秒级别
- 性能损耗在可接受范围

数据分片架构设计

分布式数据库的数据分片规则

分片字段的值固定性

- 分片字段的值与行记录一同产生
- 分片字段的值在行记录生命周期内不会更新

分片字段的数据分离性

- 分片字段的值能有效控制行记录的存储位置
- 分片字段的值具有特殊业务代表性
- 业务处理操作过程中，绝大多数情况会带上分片字段
- 分片字段值与分片算法组合，能控制行记录的分布相对均匀

分片字段的伸缩性

- 分片字段与分片算法组合后，要便于数据服务的吞吐量能平滑扩容
- 分片字段与分片算法组合后，行记录的二次分片或迁移可操作性强

业务数据和账务数据等采用：

区域性银行采用机构号分片，全国或大型银行采用机构号+报文标识号分片；

小型网络支付机构采用机构号分片，大型网络机构采用机构号+报文标识号分片；

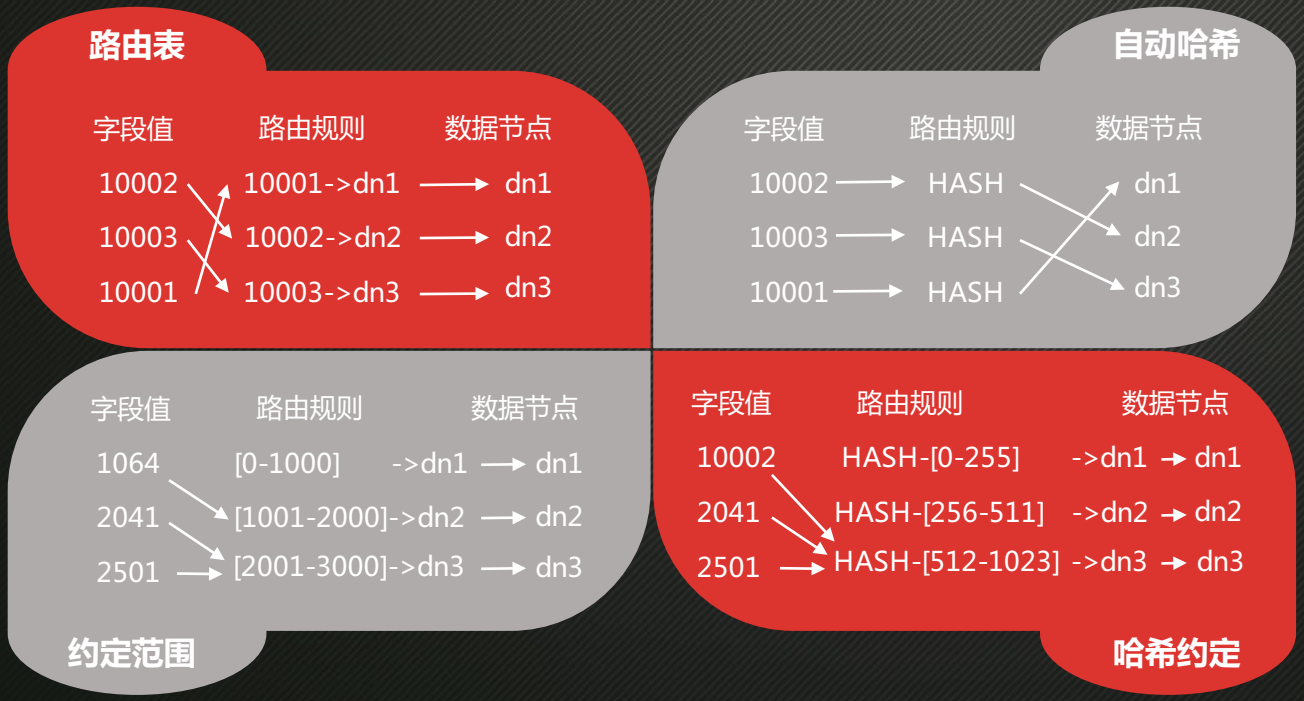
全国报文数据采用：

机构号+报文标识号；

数据分片算法：

路由算法+HASH约定算法组合；

分布式数据库的数据分片算法



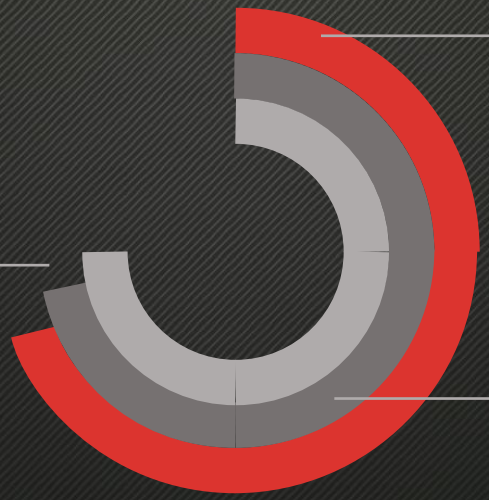
补充说明

- E-R数据分片算法是为解决主从表或称父子表的数据分片而设计支持的。
- 全局表算法减少跨库JOIN操作。
- 垂直分片表提升系统吞吐量。
- 自动哈希是按节点数量作为分母计算，哈希约定是按1024作为分母结算。

分布式数据库的数据分片算法

提供的分片函数类型

- ROUTE ：数值匹配的路由算法
- MATCH ：字符串匹配的路由算法
- RANGE ：范围算法
- HASH ：哈希算法
- AUTO ：自动哈希算法
  （路由到部分或全部的数据节点）



每种函数类型支持的数据类型

- ROUTE/RANGE ：只支持数值类型
- MATCH ：只支持字符串类型（不区分大小写）
- HASH/AUTO：支持数值类型、字符串类型、时间类型

每种函数是否需要全字匹配

- ROUTE：需要数值与HotDB配置的数值相等
- RANGE：需要数值在HotDB配置的数值范围内
- MATCH：字符串全字匹配（不区分大小写）
- HASH/AUTO：无需关注是否全字匹配，因为HotDB有内部处理机制

分布式数据库核心定位

分布式数据库核心定位

专注MySQL数据库服务的高可靠高吞吐量的分布式数据库产品，要求能在分布式数据库环境下为应用提供集中式数据库的操作体验，为海量数据、海量用户、高并发、高性能和高可用的业务系统提供强有力的支撑，同时具备强分布式透明、易扩展、无学习成本等特点。

业务场景

- 企业生产在线交易系统，即OLTP业务场景
- 经过企业级验证，成熟稳定主流开源数据库产品MySQL
- 海量数据存储
- 海量用户访问

数据分片

- 垂直拆分
- 水平拆分
- 智能分片算法

应用透明

- 单一的数据访问服务
- 屏蔽数据拆分的复杂逻辑
- 应用服务透明

服务质量

- 高可靠
- 高并发
- 高性能

分布式数据库核心定位

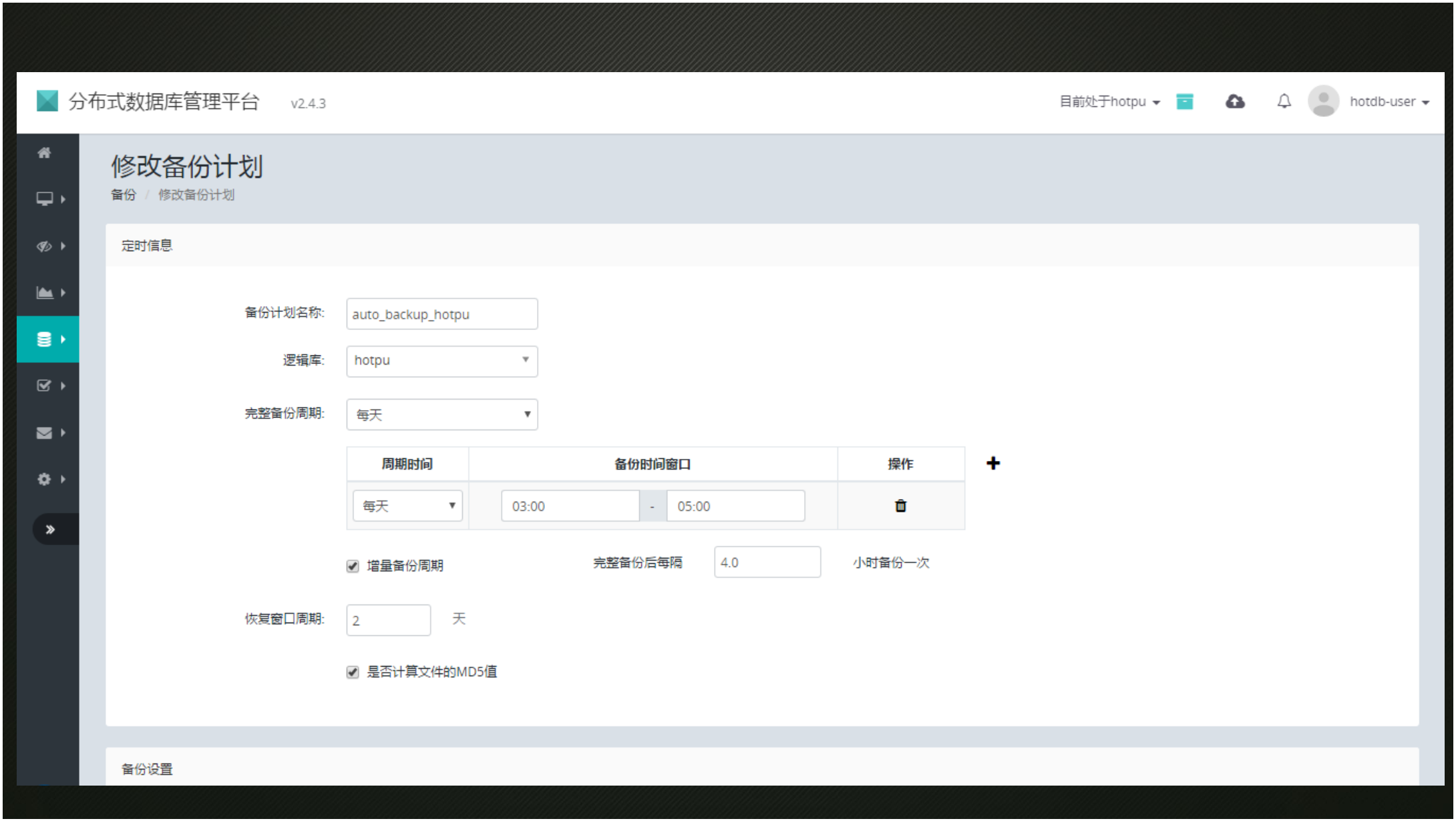
业务需求

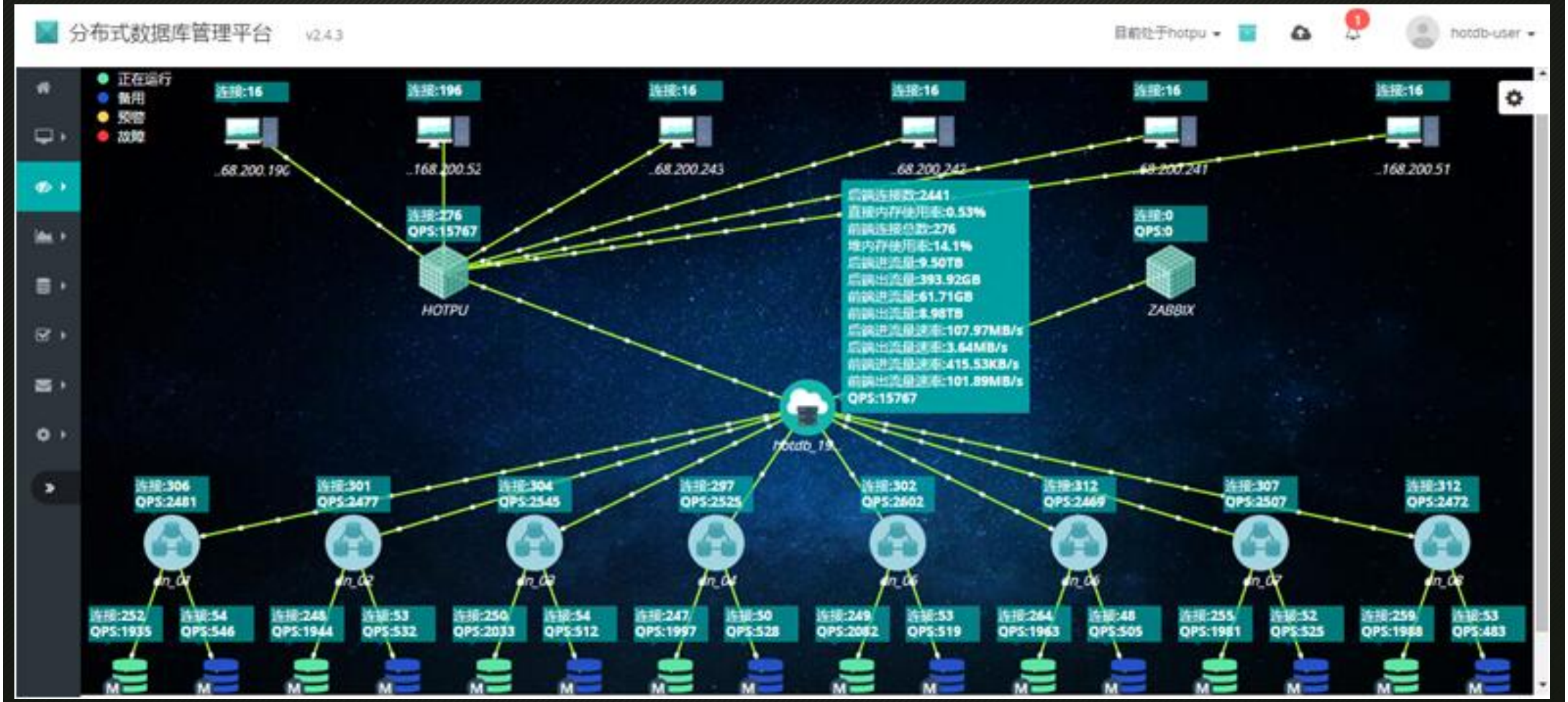
数据库管理	实例管理	节点管理	资源管理
统一访问管理	备份管理	日志管理	系统管理
业务流程	租户管理	策略管理	自动化管理
监控管理	性能管理	告警管理	安全管理

服务需求

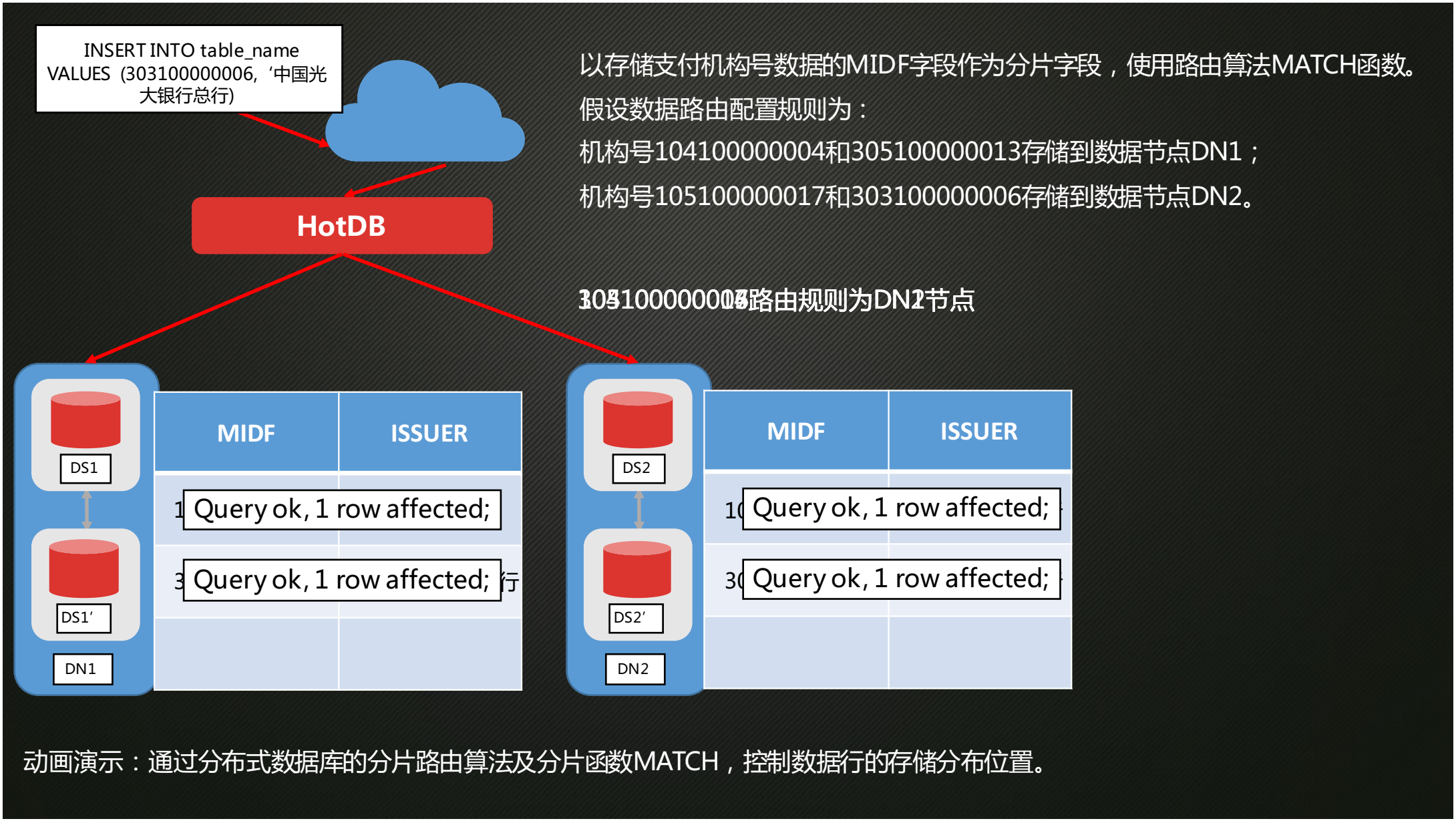
统一访问入口	数据库标准协议	多租户隔离	一键开通
服务能力全透明	资源消耗可度量	数据服务透明	数据服务扩容

分布式数据库的核心功能		
分类维度	**功能维度**	**指标维度**
数据库可用性功能	数据可靠性	99.99%
	服务可靠性	99.99%
	高可用服务	支持
	强一致备份	支持
	故障的服务恢复功能	自动
	故障对数据丢失影响	零丢失 对应节点数据库服务器不能同时损坏
	故障对应用服务影响	可用 应用服务正在做的操作报错
	主备数据库服务可用性检测算法	自动
	切换对数据同步追平判断影响	自动
	网络导致分布式数据库服务脑裂风险	存在 无影响
	网络导致数据库服务脑裂风险	不存在
	死锁算法机制	支持
数据库运维管理功能	主备数据一致无损检测	支持
	在线DDL操作对业务影响	无影响
	分布式DDL	支持
	跨数据中心的数据同步	支持
	数据库环境规范标准	支持
	图形化管理平台	支持
数据库易用性功能	全局自增序列	支持 完全透明，无须人工配置
	字符集透明	支持
	跨库JOIN	支持
	跨库判断语句支持	支持
	跨库聚合函数	支持
	跨库排序	支持
	跨库分组	支持
	跨库分组+排序	支持
	跨库排序+分页	支持
	显式分布式事务	支持
	隐式分布式事务	支持
	弱一致分布式事务	支持
	强一致分布式事务	支持
	跨库UNION/UNION ALL	支持
	读写分离	支持
	对应用的改造要求	暂不支持存储过程、视图、自定义函数

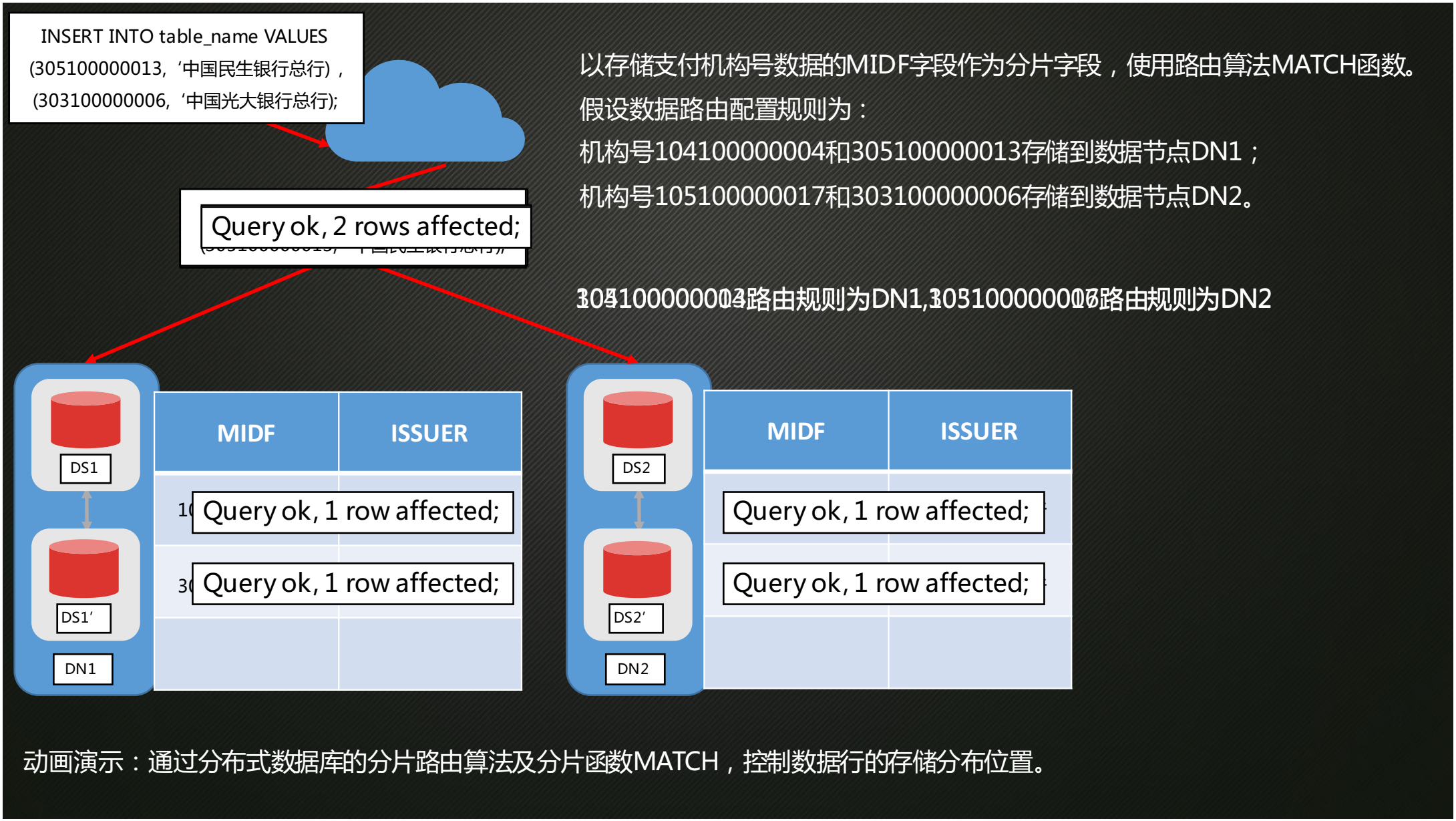

分布式数据库管理平台
v2.4.3
目前处于hotpu
hotdb-user
修改备份计划
备份 / 修改备份计划
定时信息
备份计划名称:
auto_backup_hotpu
逻辑库:
hotpu
完整备份周期:
每天
周期时间
备份时间窗口
操作
每天
03:00
-
05:00
增量备份周期
完整备份后每隔
4.0
小时备份一次
恢复窗口周期:
2
天
是否计算文件的MD5值
备份设置


分布式数据库管理平台
v2.4.3
目前处于hotpu
hotdb-user
正在运行
备用
预警
故障
连接:16
连接:196
连接:16
连接:16
连接:16
连接:16
.68.200.190
.168.200.52
.68.200.243
.68.200.242
.68.200.241
.168.200.51
连接:276
QPS:15767
HOTPU
后端连接数:2441
直接内存使用率:0.53%
前端连接总数:276
堆内存使用率:14.1%
后端进流量:9.50TB
后端出流量:393.92GB
前端进流量:61.71GB
前端出流量:8.98TB
后端进流量速率:107.97MB/s
后端出流量速率:3.64MB/s
前端进流量速率:415.53KB/s
前端出流量速率:101.89MB/s
QPS:15767
连接:0
QPS:0
ZABBIX
hotdb_19
连接:306
QPS:2481
连接:301
QPS:2477
连接:304
QPS:2545
连接:297
QPS:2525
连接:302
QPS:2602
连接:312
QPS:2469
连接:307
QPS:2507
连接:312
QPS:2472
dn_01
dn_02
dn_03
dn_04
dn_05
dn_06
dn_07
dn_08
连接:252
QPS:1935
连接:54
QPS:546
连接:248
QPS:1944
连接:53
QPS:532
连接:250
QPS:2033
连接:54
QPS:512
连接:247
QPS:1997
连接:50
QPS:528
连接:249
QPS:2082
连接:53
QPS:519
连接:264
QPS:1963
连接:48
QPS:505
连接:255
QPS:1981
连接:52
QPS:525
连接:259
QPS:1988
连接:53
QPS:483
M

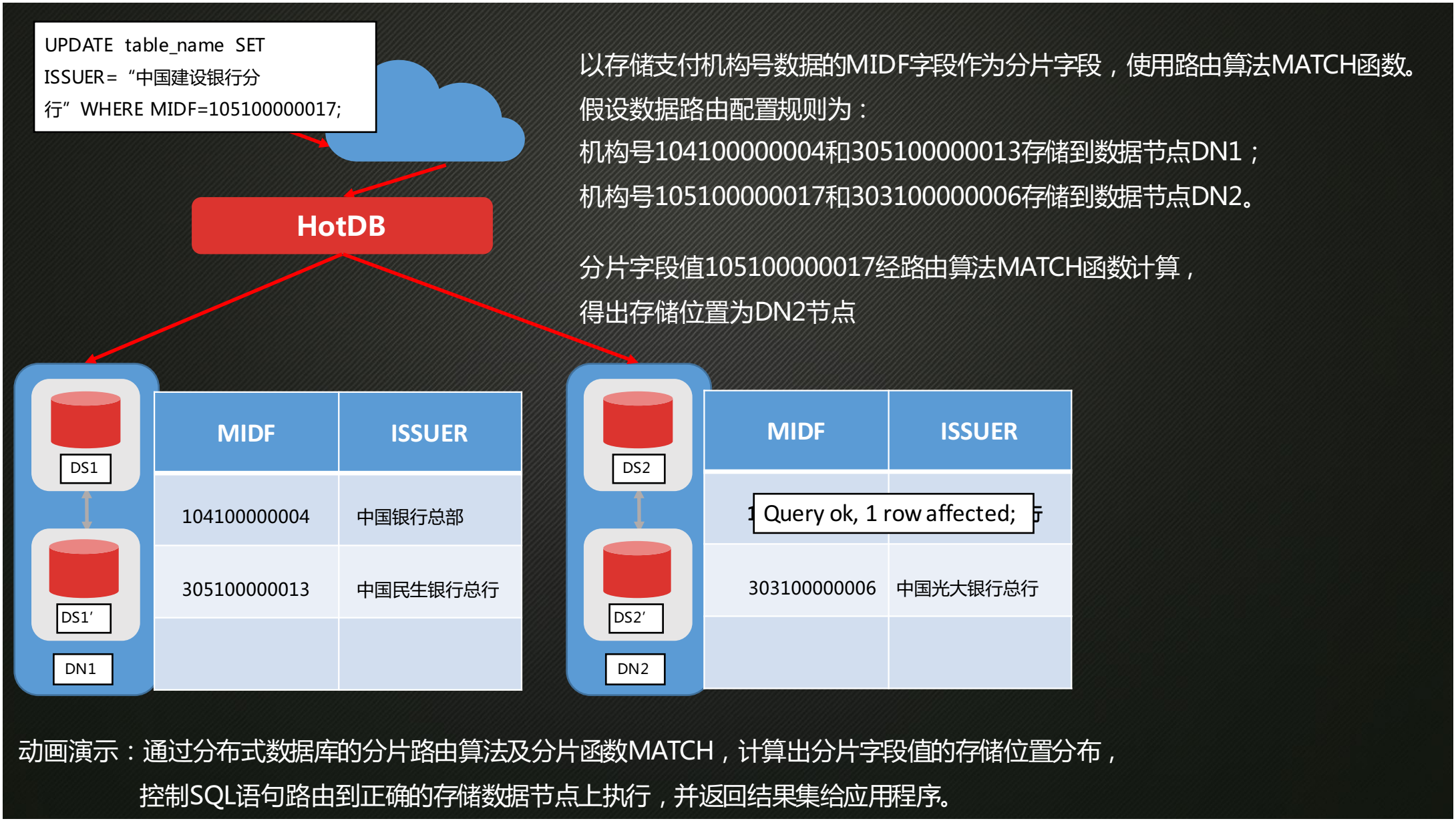

存储位置和访问位置的全透明

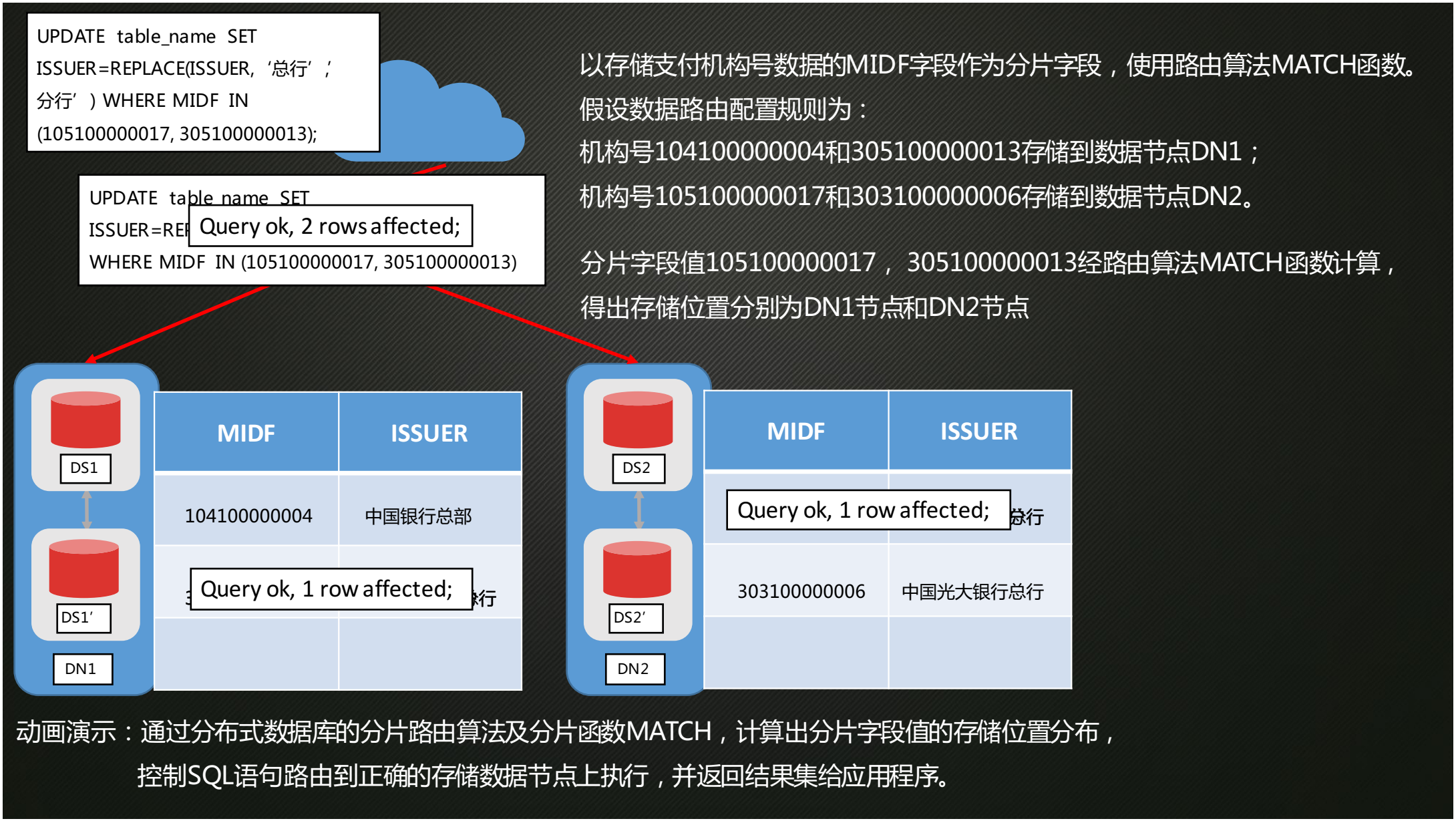


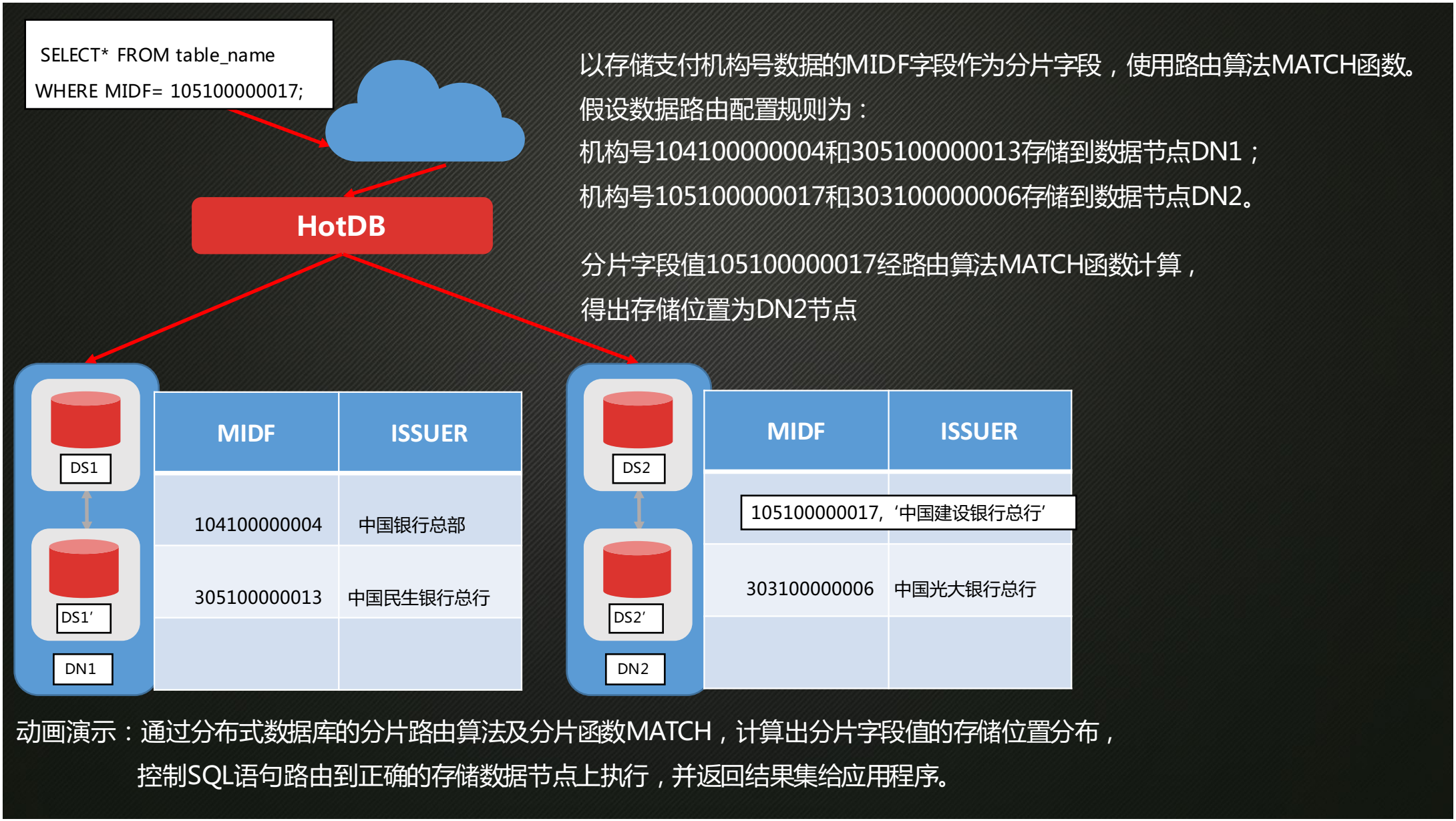
动画演示：通过分布式数据库的分片路由算法及分片函数MATCH，控制数据行的存储分布位置。



动画演示：通过分布式数据库的分片路由算法及分片函数MATCH，控制数据行的存储分布位置。


UPDATE table_name SET ISSUER=“中国建设银行分行” WHERE MIDF=105100000017;
HotDB
以存储支付机构号数据的MIDF字段作为分片字段，使用路由算法MATCH函数。
假设数据路由配置规则为：
机构号104100000004和305100000013存储到数据节点DN1；
机构号105100000017和303100000006存储到数据节点DN2。
分片字段值105100000017经路由算法MATCH函数计算，
得出存储位置为DN2节点
DS1
DS1’
DN1
MIDF
ISSUER
104100000004
中国银行总部
305100000013
中国民生银行总行
DS2
DS2’
DN2
MIDF
ISSUER
Query ok, 1 row affected;
303100000006
中国光大银行总行
动画演示：通过分布式数据库的分片路由算法及分片函数MATCH，计算出分片字段值的存储位置分布，
控制SQL语句路由到正确的存储数据节点上执行，并返回结果集给应用程序。


UPDATE table_name SET ISSUER=REPLACE(ISSUER, '总行', '分行') WHERE MIDF IN (105100000017, 305100000013);
UPDATE table_name SET ISSUER=REP
WHERE MIDF IN (105100000017, 305100000013)
Query ok, 2 rows affected;
以存储支付机构号数据的MIDF字段作为分片字段，使用路由算法MATCH函数。
假设数据路由配置规则为：
机构号104100000004和305100000013存储到数据节点DN1；
机构号105100000017和303100000006存储到数据节点DN2。
分片字段值105100000017，305100000013经路由算法MATCH函数计算，
得出存储位置分别为DN1节点和DN2节点
DS1
DS1'
DN1
MIDF
ISSUER
104100000004
中国银行总部
Query ok, 1 row affected;
DS2
DS2'
DN2
MIDF
ISSUER
Query ok, 1 row affected;
303100000006
中国光大银行总行
动画演示：通过分布式数据库的分片路由算法及分片函数MATCH，计算出分片字段值的存储位置分布，
控制SQL语句路由到正确的存储数据节点上执行，并返回结果集给应用程序。


SELECT* FROM table_name
WHERE MIDF= 105100000017;
HotDB
以存储支付机构号数据的MIDF字段作为分片字段，使用路由算法MATCH函数。
假设数据路由配置规则为：
机构号104100000004和305100000013存储到数据节点DN1；
机构号105100000017和303100000006存储到数据节点DN2。
分片字段值105100000017经路由算法MATCH函数计算，
得出存储位置为DN2节点
DS1
DS1'
DN1
MIDF
ISSUER
104100000004
中国银行总部
305100000013
中国民生银行总行
DS2
DS2'
DN2
MIDF
ISSUER
105100000017，'中国建设银行总行'
303100000006
中国光大银行总行
动画演示：通过分布式数据库的分片路由算法及分片函数MATCH，计算出分片字段值的存储位置分布，
控制SQL语句路由到正确的存储数据节点上执行，并返回结果集给应用程序。

```
SELECT O.MIDF AS A,O.ISSUER AS OISS,
O.MROOT,P.MIDF AS B,P. ISSUER AS PISS FROM
M AS O JOIN M AS P ON O.MROOT=P.MROOT
WHERE O.MROOT=104100000001;
```

支付机构号数据的MIDF字段+报文标识号MROOT字段组合作为分片字段，使用路由算法MATCH函数和哈希约定算法HASH函数。

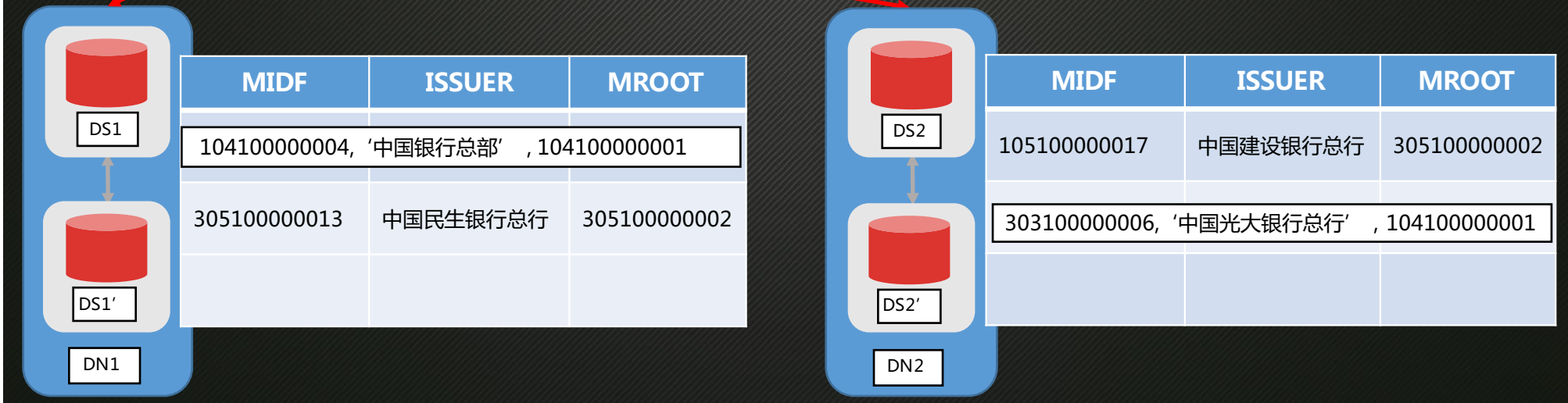
假设数据路由配置位置为：

机构号104100000004与104100000001组合、305100000013与305100000002组合存储到数据节点DN1；

机构号105100000017与305100000002组合、303100000006与104100000001组合存储到数据节点DN2。

104100000004，‘中国银行总部’,104100000001,303100000006，‘中国光大银行总行’

经过HASH函数计算，得出数据分布在DN1和DN2数据节点上，HotDB将SQL语句路由下发到DN1和DN2数据节点上并行执行，返回的结果集再经HotDB内置JOIN算法处理，最终返回应用程序期望的结果集

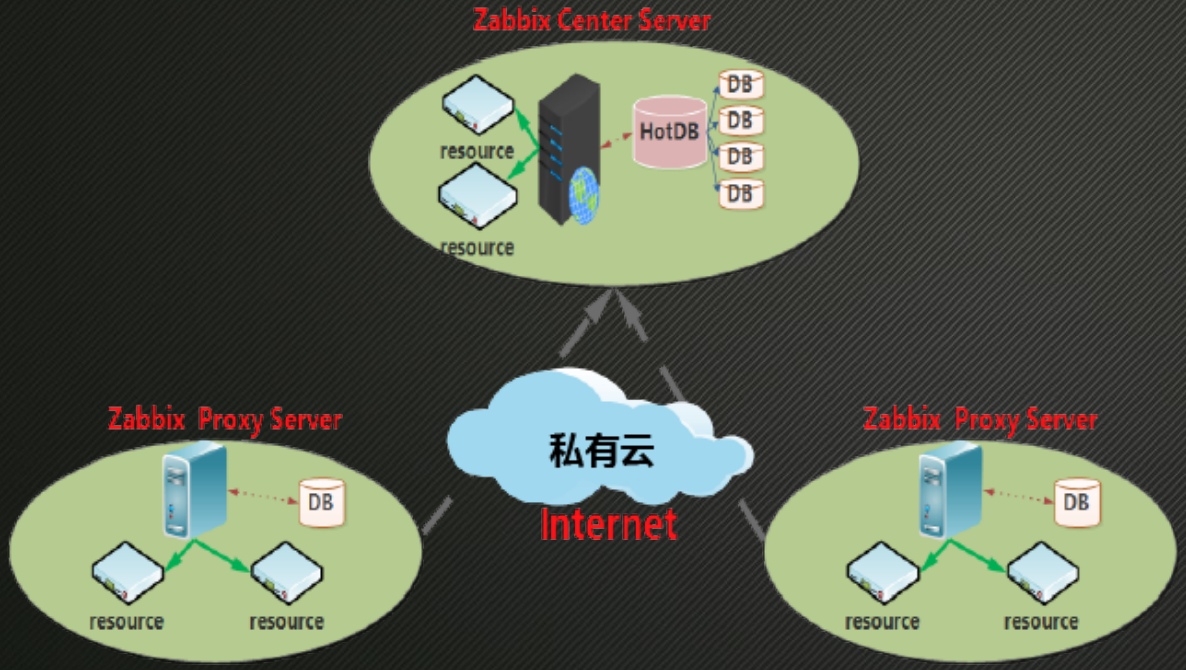


MIDF	ISSUER	MROOT
104100000004，‘中国银行总部’，104100000001		
305100000013	中国民生银行总行	305100000002

MIDF	ISSUER	MROOT
105100000017	中国建设银行总行	305100000002
303100000006，‘中国光大银行总行’，104100000001		

动画演示：通过分布式数据库的分片哈希约定算法及分片函数HASH，实现跨数据节点的JOIN及数据结果集合并处理再返回正确的结果集。

Zabbix监控软件的通用分布式解决方案



方案价值：

- 突破Zabbix监控软件NVPS的瓶颈
- 提升Zabbix监控软件的数据处理的效率和吞吐量
- 无需修改Zabbix监控软件的任何代码，做到集中式数据库变身为分布式数据库

实战案例：

- 全国 6+ 数据中心
- 物理服务器数量 8000+ 台
- 监控项数量 300万+ 个

THANKS

SequeMedia 盛拓传媒 ITPUB ChinaUnix