# ADB数据库在电信运营商系统中的应用

嘉宾：姜明俊
公司：亚信软件

ADB系统架构简介

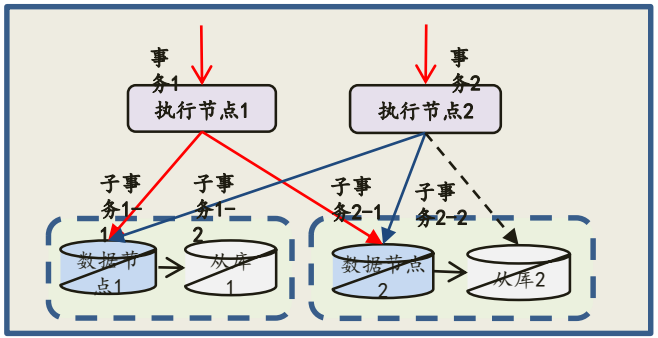
ADB实施案例介绍

ADB关键技术介绍

# 分布式数据库：两种技术架构选型

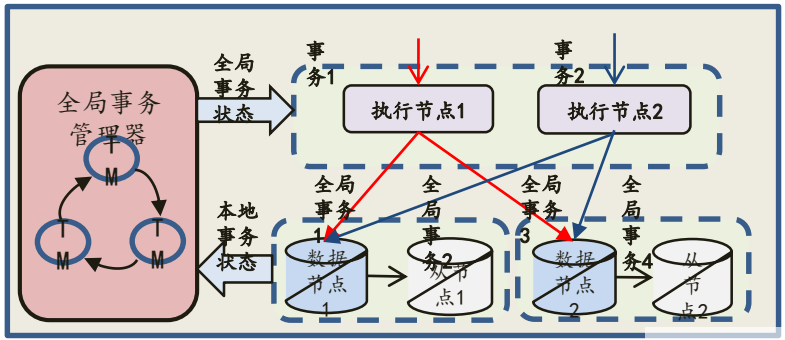
## 互联网级数据库架构

- **集群规模：**数百、上千节点存储百T以上数据，承载数百万TPS和上亿QPS，优先强调数据库的可扩展性
- **数据一致性：**弱化对数据一致性的要求，通过业务系统保证最终一致性
- **应用开发：**需要根据数据库架构进行定制开发




## 企业级数据库架构

- **集群规模：**单库规模为数十个节点，数据规模通常在10T以下，处理能力为数千TPS和上万QPS
- **数据一致性：**业务高度依赖数据库提供的ACID（原子性、一致性、隔离性、持久性）能力，必须保证数据在任何场景下的强一致性
- **应用开发：**已经过多年开发，需要支持透明、低成本的平滑迁移



# ADB产品目标和特性

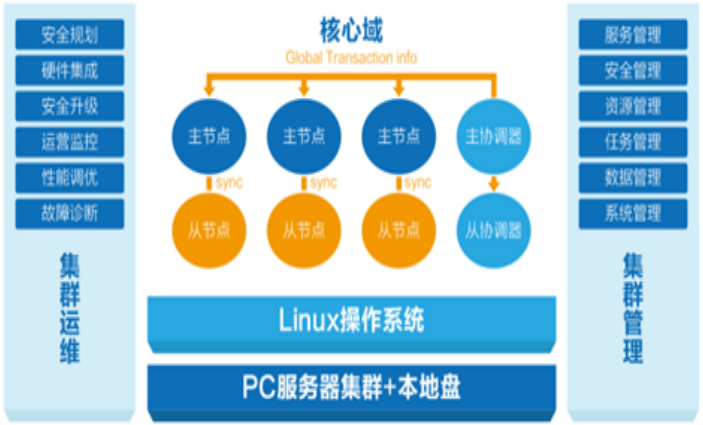
**产品目标**

企业级应用 | 安全可靠 | 可扩展 | 稳定 | 易用

**分布式**

**事务型**

**关系型**




**V2.1版本开发**
- ✓ 大量数据一致性、稳定性Bug修复
- ✓ 重构连接池提升性能
- ✓ 75%的ORACLE语法兼容，降低应用迁移难度
- ✓ 支持和ORACLE异构容灾
- ✓ 数据备份恢复，支持基于任意时间点的恢复
- ✓ 通过移动集团分布式数据库测试

**V3.1版本发布**
- ✓ 支持同步异步流复制模式自适应切换
- ✓ 分布式SQL查询引擎优化，提升复杂SQL执行效率
- ✓ 内核升级到PG9.6版本
- ✓ 秒级在线扩缩容
- ✓ 采用libpq改造pgxc的通讯方式

2014.08 2015.10 2016.12 2017.12

**参与PGXC开源版本开发**
- ✓ 日本NTT公司铃木先生2000年启动、2013年开源，2015年起处于停滞状态
- ✓ 基于PostgreSQL9.3的分布式数据库集群解决方案
- ✓ 存在问题：负载稳定性、数据一致性、应用易用性、运维支撑能力等

**V2.2版本发布**
- ✓ 自主开发全局事务管理器，实现技术架构的完全可控
- ✓ 达到85%的ORACLE语法兼容
- ✓ 增强分布式事务的处理能力
- ✓ 增强集群管理和自动化运维工具
- ✓ 增加数据存储策略，可自定义数据分片
- ✓ 支持SQL2011标准

# ADB产品系统架构

**应用可通过任何一个Coordinator进行读写，读写能力可同时扩展，所有事务具备一致的数据库视图**

## AGTM：全局事务管理器（主，从节点内）

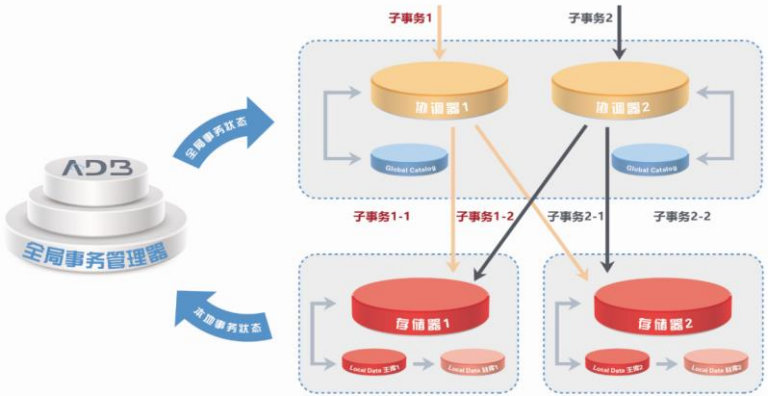
- AGTM保存事务ID和事务快照以及全局序列号、时间戳，参与2PC提供分布式MVCC能力；负载很小；可支持20个节点的集群规模，满足企业级应用需求

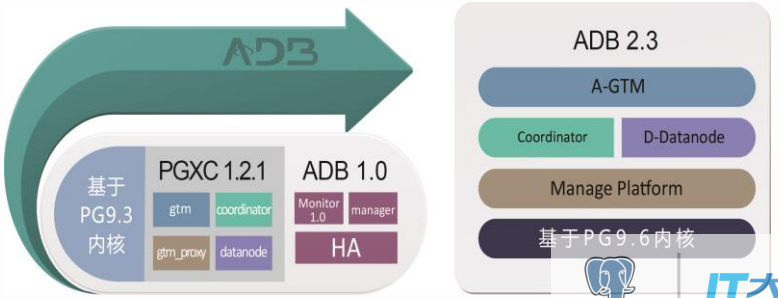
## 存储器：完成数据的存储和执行（主、从节点内）

- 支持数据节点的分组管理
- 提供应用透明的复制和分布两种存储方式
- 提供多种分片函数，支持定制开发
- 支持节点动态扩容和数据重平衡

## 协调器：处理客户端的连接请求（主、从节点内）

- 支持SQL2011标准：生成执行计划，将执行计划、全局事务ID和快照号发送给数据节点执行，对结果集归并处理
- 事务由Coordinator节点发起，并由Coordinator提交或回滚。
- 多协调节点同时接受客户端连接：提供读写可扩展性
- 对跨节点操作提供分布式事务支持
- 支持跨节点的关联、聚合类分析操作



# ADB实施案例介绍

# ADB的实施案例：某省案例介绍

## 项目背景：

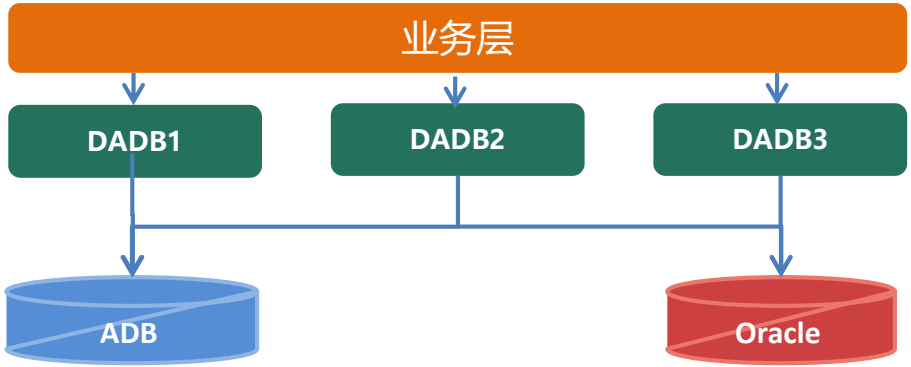
2016年，某省移动在一期渠道管理系统的基础上建设渠道中心，支撑社会渠道和自有渠道的所有日常运营管理工作。渠道中心系统尝试去‘O’，采用DADB（总线）+ ADB（数据库）的分布式部署方案

## 项目目标：

300多张业务表，数据总量500G左右，公共库cmbase，资源库channel，报表库rpt，全部迁移到ADB中

## 使用效果：

- 1200个高并发业务读写，TPS最高5600，响应延时均低于2秒，性能提升明显，集群稳定
- 异构数据库容灾，与Oracle数据库实时同步



# ADB的实施案例：某省案例介绍

## 项目背景：

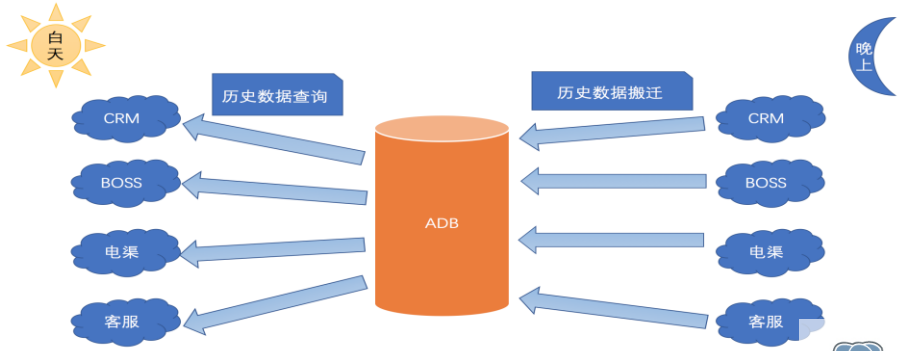
某省移动历史库去‘O’项目中，采用了ADB，存放CRM、账务、电渠、客服等系统的历史数据以及登陆日志、用户操作记录等信息。2w+张历史数据表，数据总量目前在40TB，包括原来Oracle历史库中的数据以及持续从在线库搬迁至ADB的数据，数据量年增长量20TB+。

## 项目目标：

6个月以上的业务数据通过ADB来查询，在线库只保留最近6个月的数据，以达到系统瘦身的目标。

## 使用效果：

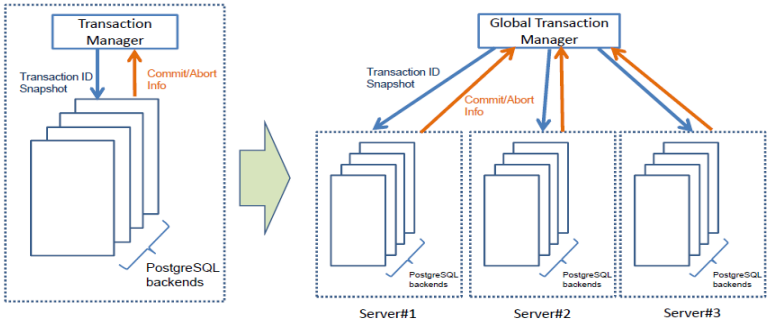
- 业务查询压测，1000并发，sql响应在200ms以内
- 晚上task数据搬迁(根据业务规则，从在线Oracle库搬迁至ADB)，一分钟100w数据量



# ADB分布式事务实现

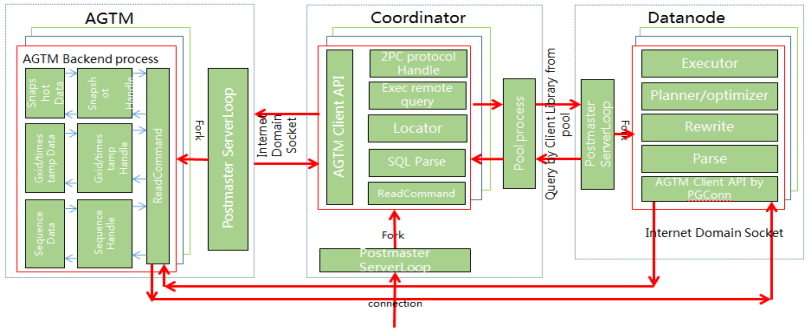
# 分布式事务处理机制

- GTM使用单进程多线程模型
- GTM只支持一主一从两副本模式
- 读写都需要分配GXID
- 事务提交分本地提交和全局提交
- GTM不参与全局2PC，会出现数据写不一致
- Snapshot不完整，会出现数据读不一致
- DataNode节点出现2PC事务状态不一致，需要再使用pgxc_clean进行手工清除遗留的两阶段事务
- 存在诸多数据异常、网络缓冲区未清除及锁不释放引起的宕机问题




PGXC分布式事务控制

- AGTM也会记录全局事务日志（WAL）参与2pc，保证全局事务状态的一致性
- 多进程多listen模型，充分利用资源，一个进程挂了，不会影响其他用户访问
- 使用PG流复制来实现HA，并能获得更好的灵活性和集群的稳定性，支持一主多从，同步+异步的混合模式
- 消息通讯协议减少70%，性能提升60%
- 不需要使用本地+全局两次提交，大大减少了事务的冲突率
- 读操作不需要消耗全局GXID资源
- ADB新增remote xact manager进程，用于自动解决分布式事务部分成功问题




ADB分布式事务控制

# 优化分布式引擎和集群通信

# 优化集群通讯

- ★ PGXC的通信接口与libpq在功能上类似，完全可以以代码质量更有保证的libpq接口实现相关功能
- ★ PGXC的通信状态DNConnectionState把握不准确，导致大量的断言错误
- ★ PGXC的通信接口效率低下，影响性能
- ★ 代码不够严谨，开发人员疲于应对相关bug

PGXC不足

- ✪ 引入PGconn作为通信句柄， 复用libpq接口大量代码
- ✪ 按需扩展libpq协议消息
- ✪ 通过libpq的copy消息实现节间数据传输，相对于PGXC效率更高
- ✪ 引入InterXactState概念，保证节间事务逻辑。

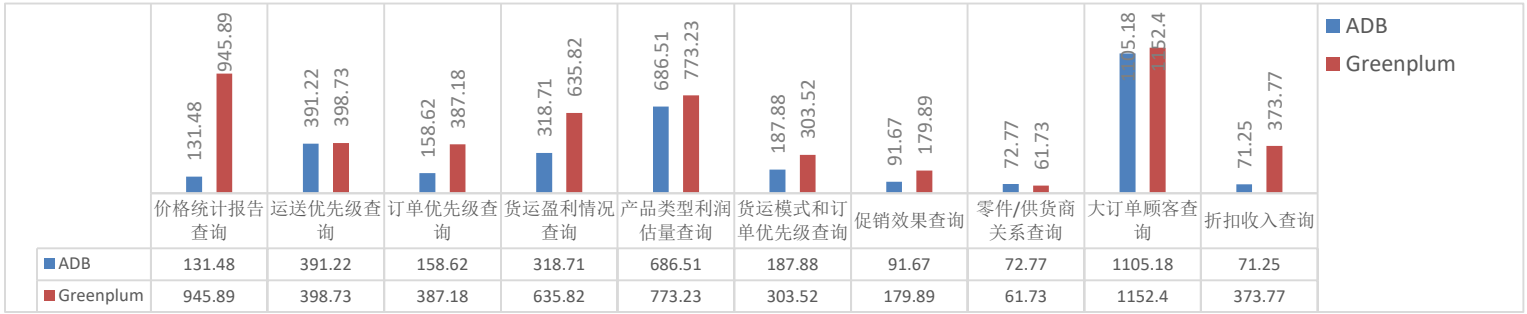
ADB优化

# 分布式执行计划改造

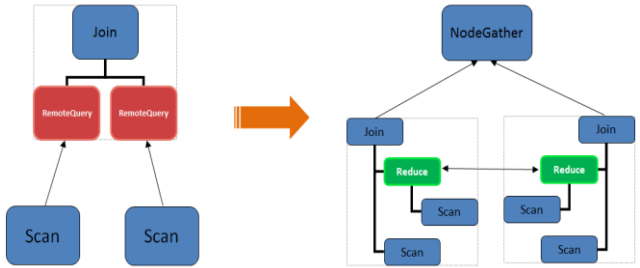
ADB相比PGXC实现了真正意义上的分布式执行，可以使聚集、排序、关联（尤其是内关联）及插入通过增加节点以倍数级提升性能

在数据传输方面性能更高、不易出错、代码简洁

**TPC-H性能对比**




| | 价格统计报告查询 | 运送优先级查询 | 订单优先级查询 | 货运盈利情况查询 | 产品类型利润估量查询 | 货运模式和订单优先级查询 | 促销效果查询 | 零件/供货商关系查询 | 大订单顾客查询 | 折扣收入查询 |
|---|---|---|---|---|---|---|---|---|---|---|
| ADB | 131.48 | 391.22 | 158.62 | 318.71 | 686.51 | 187.88 | 91.67 | 72.77 | 1105.18 | 71.25 |
| Greenplum | 945.89 | 398.73 | 387.18 | 635.82 | 773.23 | 303.52 | 179.89 | 61.73 | 1152.4 | 373.77 |




| 数据库版本 | 主机配置 | 集群 |
|---|---|---|
| ADB3.1 | CentOS 7.2（Linux version：2.6.32-642.6.1.el6.x86_64）<br>CPU：Intel(R) Xeon(R) CPU E5-2609 v3 @ 1.90GHz 12核<br>MEM：64G<br>Disk： 1.6T SCSI<br>网卡： 1000Mb<br>数据量：100GB | 2个Coordinator，3个DataNode master |
| GP5.0 | | 1个Master instance，3个primary segment |

# 语法兼容、分片、多副本、高可用

# Oracle语法兼容

- **服务器级别：通过修改数据库配置文件默认为Oracle语法**
- **会话级别： 通过设置会话级配置参数指定Oracle语法**
- **语句级别： 通过/*ora*/或/*oracle*/等前缀注释指定按Oracle语法执行SQL**

**兼容**

- ROWID 、 ROWNUM、 （ + ）、CONNECT BY 、CASE WHEN 、LIMIT 、别名等Oracle特殊语法
- 80%的常见Oracle函数，数据类型，操作符，如“||”和对空字符串的处理以及正则表达式等
- 业务层常用的隐转规则，尽管Oracle和PG都不建议这样做
- OCI接口

****运营商CRM渠道中心使用ADB完整兼容业务SQL，不需要业务专门做SQL改动**

# 数据分布策略（复制、分片）

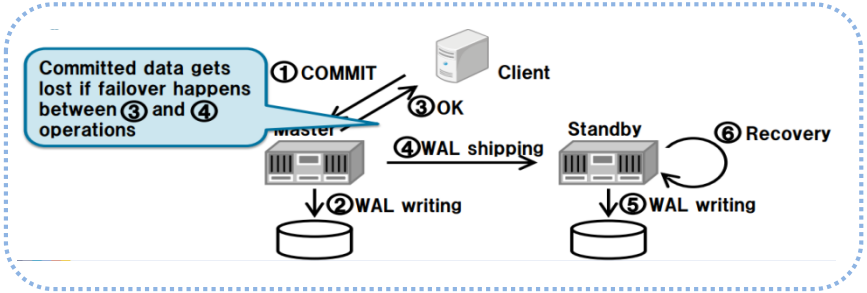
**复制表：用于静态类配置小表，每个节点上都会存储相同的一份数据；**

**分片表：数据分布在各个节点上，用于大数据量的表根据分片字段进行分片；**

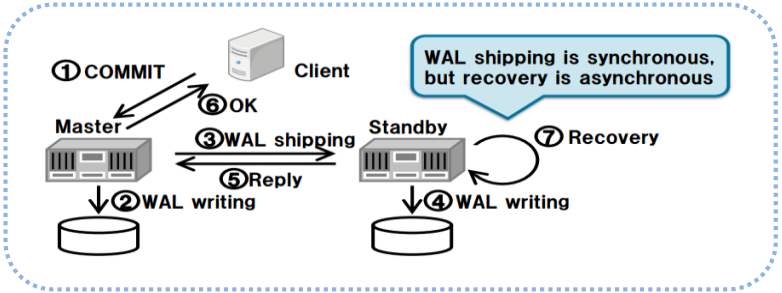
- **支持用户自定义分片**
  - 用户可根据业务场景制定更匹配的分区规则
  - 分区键可以为任意多个
  - 可预知数据的分区情况
  - 自定义分区函数的返回值需为整数
- **各个分片算法可进行互转**
- **缺省供三种分片方式** hash(column)、modulo(column)、roundrobin
  - 分区键至多1个
  - 用户无法根据业务场景制定分区规则
  - 用户无法预知数据的分布情况
  - 可能会出现数据的倾斜
  - 支持数据分组管理

# 多副本和高可用

- 基于PG的流复制实现多副本同步，对应用访问透明
- 支持同步和异步两种模式，且支持同步和异步模式的自适应切换
- 支持四种同步复制级别：on，remote_write，local，off，应对不同的数据安全和性能要求
- AGTM和DN均无单点故障 AutoFailover
  - repmgrd进程实时监控
  - 监控到master节点异常后，连接ADB-MGR执行failover操作
  - 故障节点使用pg_rewind恢复数据，并重新加入到集群中
- 秒级切换：数据无丢失




异步模式

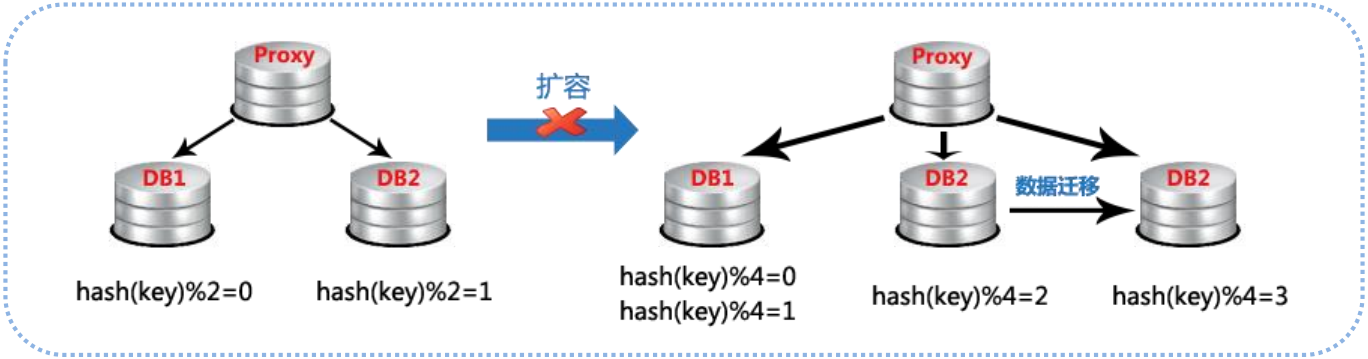



同步模式

# 扩容、迁移、备份恢复、容灾

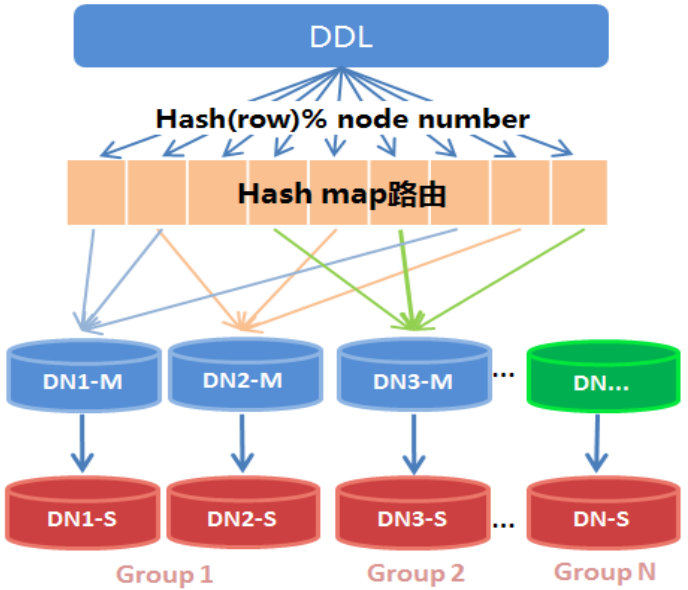
# PGXC的扩缩容问题

- 扩容过程中数据迁移时要锁集群，停止应用访问
- Rebalance操作，非常耗时，数据量大的场景无法接受，一张表2000w条记录，5个节点，完成平衡需要20-30分钟。
- 不支持单一节点扩容
- 缩容时先对数据进行备份，再删除节点，然后老数据再导入集群；
- 在线增加节点：新建表根据复制、分片策略存储到新节点；原有表数据仍然存储在老节点上；并没有解决节点数据倾斜的问题。



# ADB采用Hash+Map路由算法

- 将集群的数据划分为1024（可设定）个slot。对分片字段hash后，除1024取模，得到一个对应的slotid
- 系统表adb_slotnodemap维护数据的映射和每个slot的状态，包含如下字段
  - Slotid： 数据块ID
  - Nodeid: 节点编号
  - Status：slot的状态，包括online，moving，switch
- 数据路由时，通过slotid从映射表中找到对应的node



# 节点扩容过程

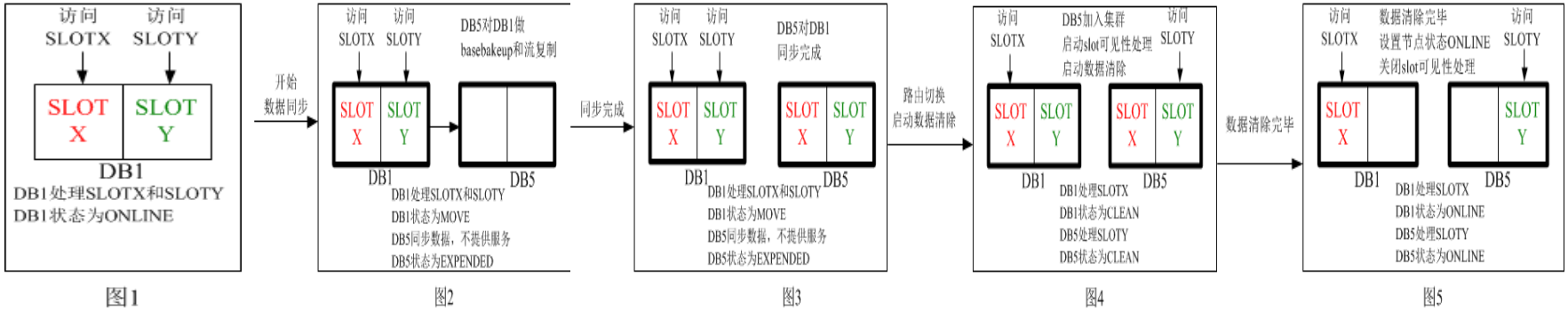



图1 图2 图3 图4 图5

**扩容前**

| SlotId | NodeID |
|---|---|
| SlotX | Node1 |
| SlotY | Node1 |

**扩容后**

| SlotId | NodeID |
|---|---|
| SlotX | Node1 |
| SlotY | Node5 |

# 某省移动扩容案例

## 项目背景：

某省移动历史库项目使用ADB做为基础数据库软件，目前存放CRM侧历史订单数据和实例数据，数据量大小40TB。前期规划局方提供了4台pc server，但由于单台pc server存储扩展有限以及后续接入系统的历史数据增长量大，局方随后追加4台pc server，年底会达到80T的总数据量。此时就面临需要对ADB集群进行扩容的操作，将40TB数据从4台分散到8台主机

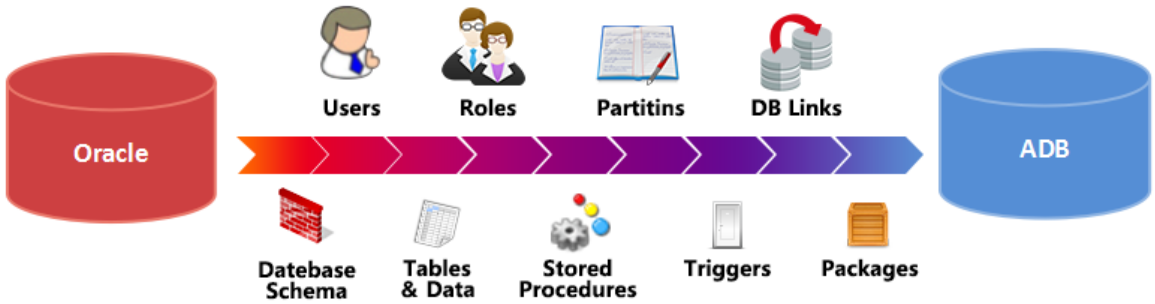
## 扩容实施：

采用ADB提供的平滑扩容技术，可以很好的解决上述问题，对业务来说，只是在路由切换步骤中短暂的操作挂起，集群锁释放后，操作继续，时间非常地短暂，真正做到了平滑扩容

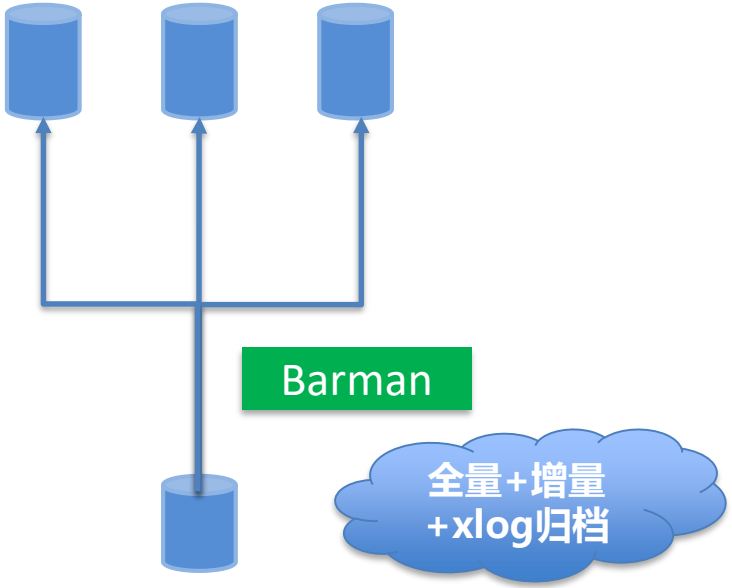
| 操作 | 用时 | 备注 |
| --- | --- | --- |
| 数据同步 -- 4节点数据同步到8节点 | 2小时5分钟 | 不影响业务，正常访问 |
| 路由切换 | 18秒 | 需要锁集群 |
| 数据清理 – 按照数据分布规则，调用vacuum进程清理各个节点的数据 | 12小时 | 不影响业务，正常访问 |

# 数据迁移

- 基于Ora2PG进行改造和优化，使其适应分布式场景，比如bug修复，distribute by语法，数据自动切片，过程跟踪，断点续抽，数据迁移完整性稽核等等
- 不分割导入100G的表到ADB，平均在50分钟，进行分割后多台并行导入，100G平均8-10分钟内完成



# 基于barrier的一致位点恢复




**基于时间点的恢复，在集群中主机时间不一致的场景下，会导致部分节点的数据恢复丢失**

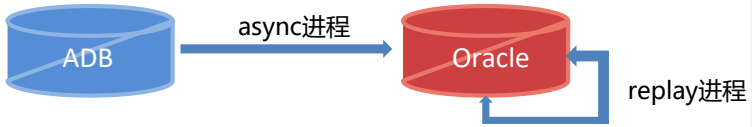
## Barrier

- 所有节点一致性位点
- 连接coordinator执行: CREATE BARRIER <barrier_name>

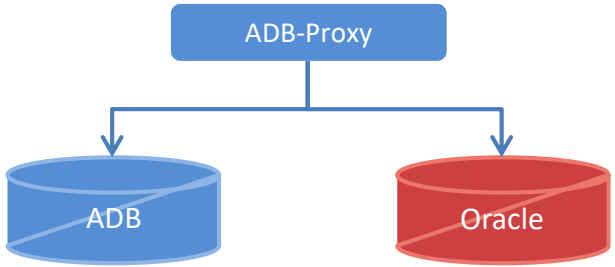
recovery.conf基于barrier的恢复，确保集群各个数据节点均恢复到一致的状态(全局barrier点)

# 数据库容灾

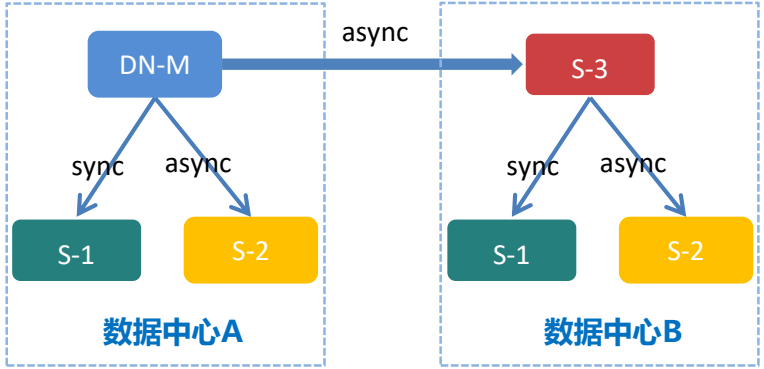
- ADB和Oracle之间的异步复制




- 通过数据库中间件实现ADB和Oracle的同步复制




异构容灾




1. 数据中心A和数据中心B之间采用异步复制
2. 每个数据中心都会部署ADB-MGR进行资源管理
2. ADB-MGR根据数据域执行failover切换
3. 数据中心A和数据中心B之间是整体切换
3. B数据中心，可以提供读的能力

异地容灾

# Thanks!