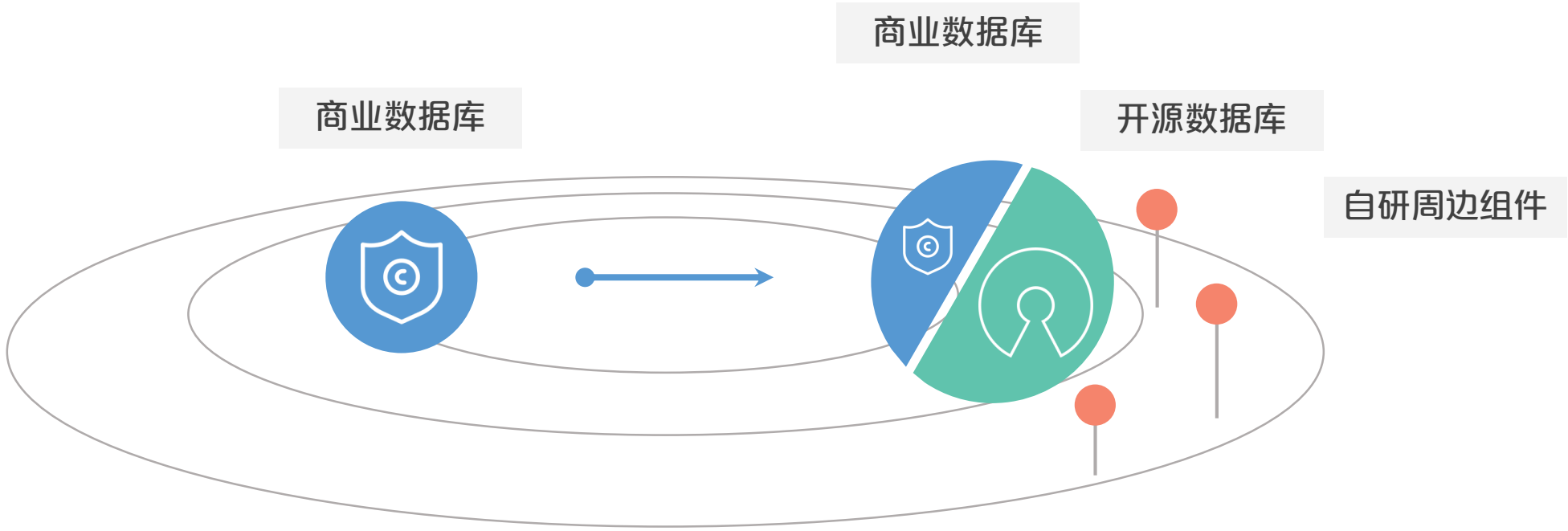
中国银联数据库变迁

2019年05月

中国银联 China UnionPay

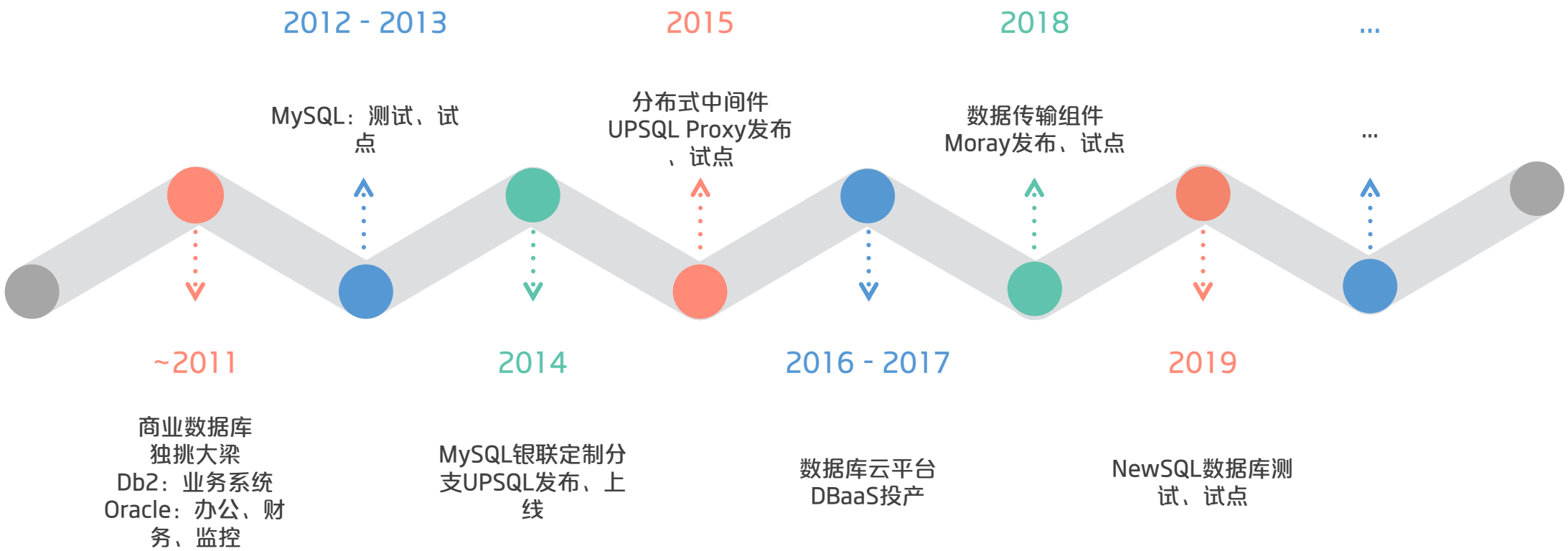
概述

从商业到开源、自研



历程回顾

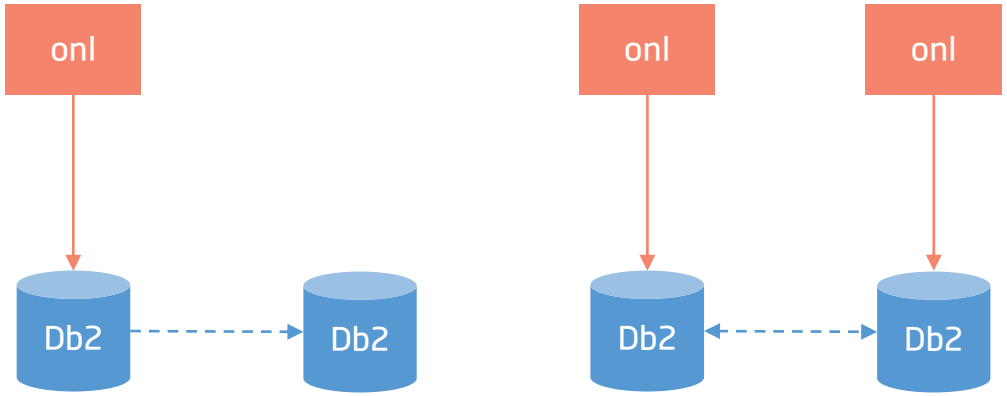
自2011年以来



重度依赖Db2

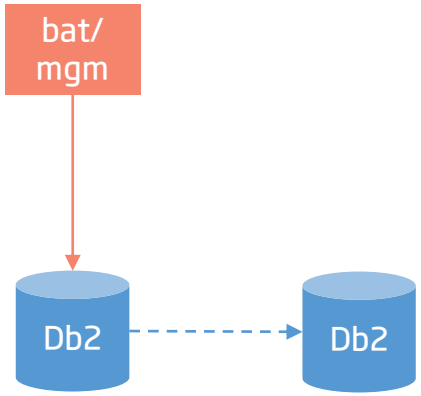
单中心内

联机类系统



dTube、GDDS同步

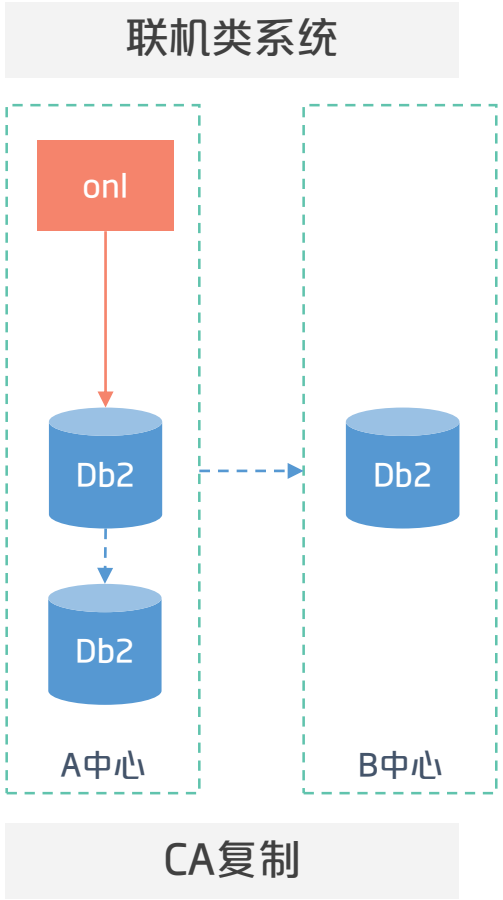
批量、管理类系统

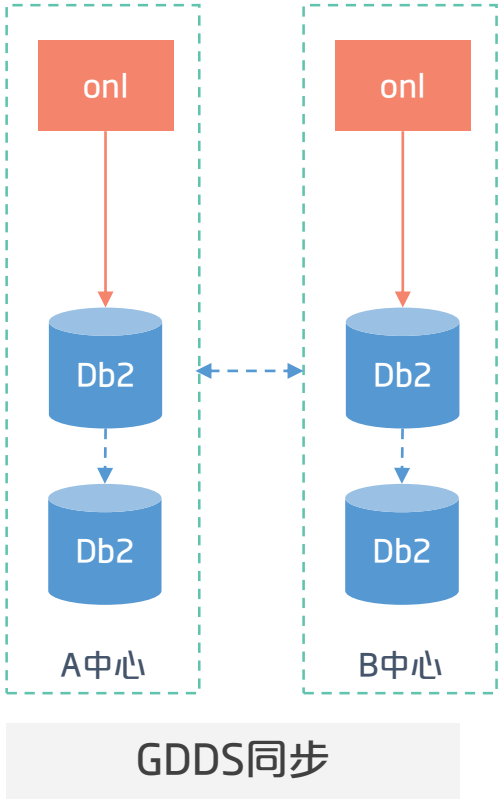


HADR同步

重度依赖Db2

异地


联机类系统
onl
Db2
Db2
Db2
A中心
B中心
CA复制


onl
onl
Db2
Db2
Db2
Db2
A中心
B中心
GDDS同步

引入MySQL

目标

扩展性

集中式架构终将遭遇性能瓶颈

掌控力

拥抱开源，逐步自主掌控

服务

更快的响应速度

降成本

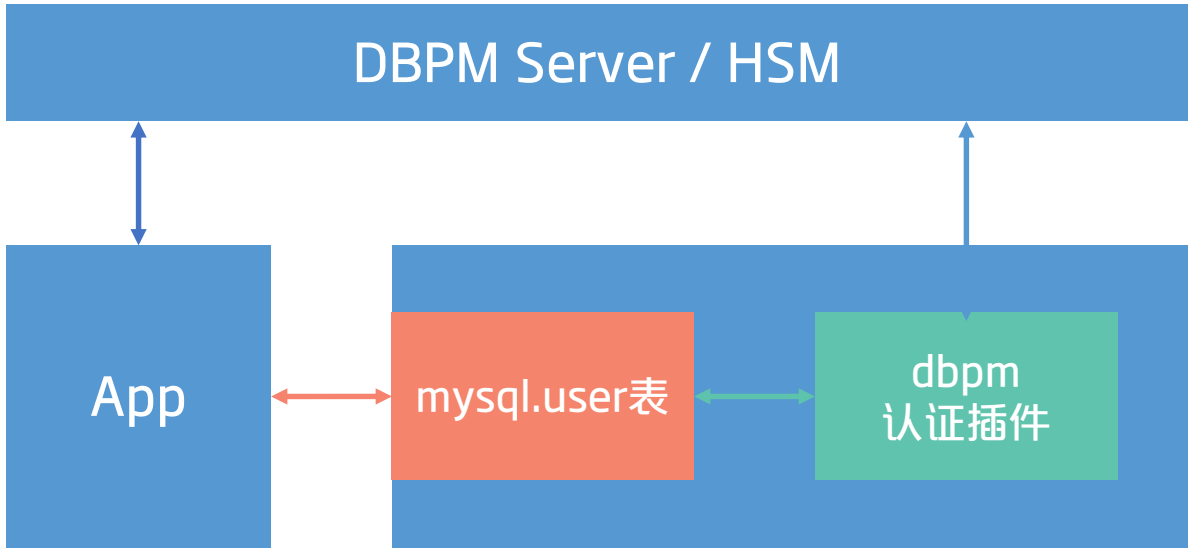
节约硬件成本、软件成本、维保费用

个性化

按自身需求定制开发

UPSQL特性

安全



- 应用程序和数据库均不保存密码
- 密码存放在DBPM服务器，统一管理，方便定期更新
- 应用程序和数据库均从DBPM服务器获取密码，由认证插件完成认证
- DBPM可使用加密机替代

UPSQL特性

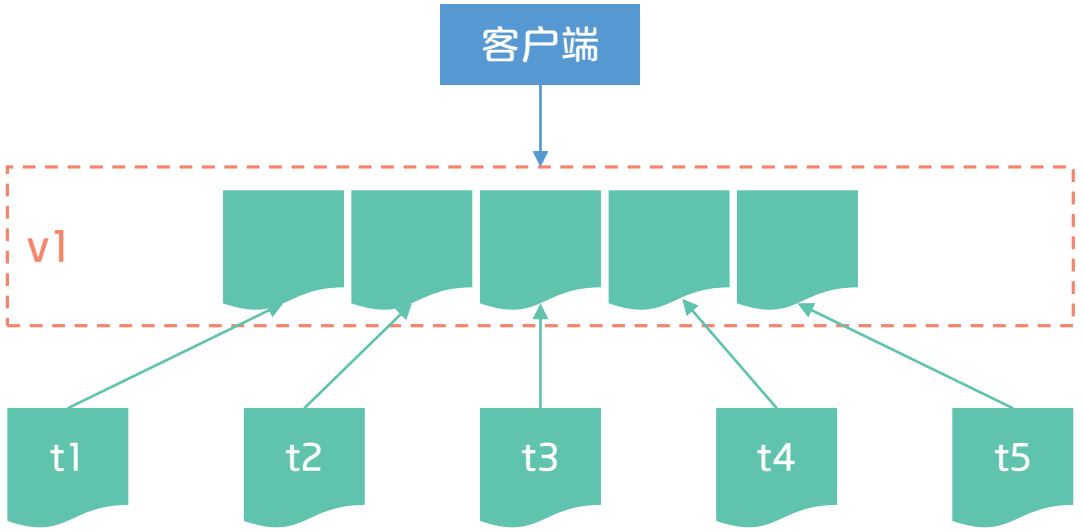
□ 性能

- 线程池
- 多队列并发控制
- 热点数据更新优化
- Union视图性能优化
- 无效链接自动清理
- ....

```
create view v1 as select * from t1 union [all]  select * from t2 union [all] ...
select * from v1 where ...
select count(*) from v1 where ...
select sum(col_x) from v1 where ...
```

MySQL先将v1的各基表全量数据汇总至临时表，然后再从临时表查询。存在以下问题：

- 无法使用基表索引，全表扫描所有基表，增加磁盘IO压力，影响整个实例性能
- 全量数据生成临时表，消耗大量内存和临时表空间



UPSQL特性

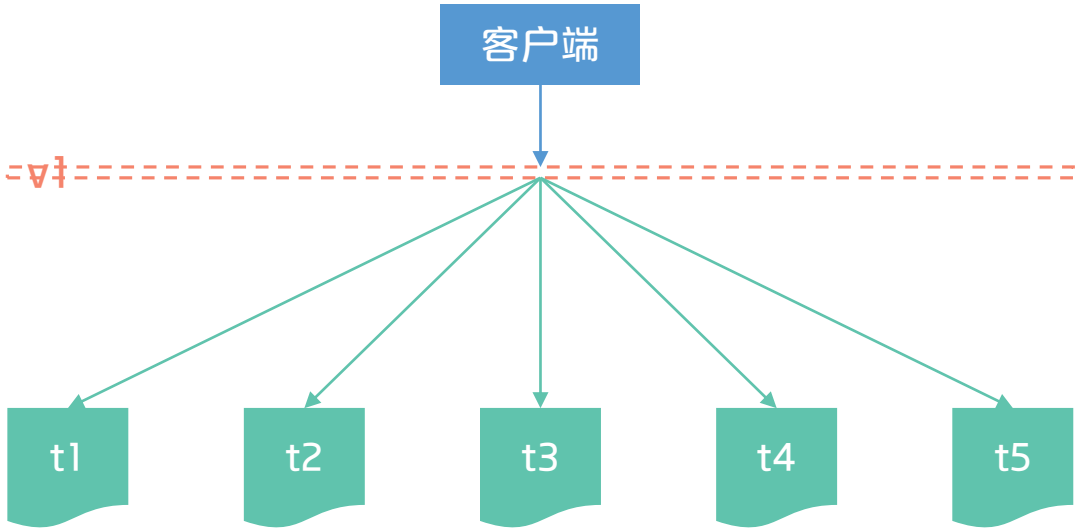
□ 性能

- 线程池
- 多队列并发控制
- 热点数据更新优化
- Union视图性能优化
- 无效链接自动清理
- ....

```
select * from (
    select * from t1 where ... union [all]
    select * from t2 where ... union [all]
    ...
) as v1;
```

UPSQL针对上述场景进行优化，将视图外层查询条件下推至基表上，实现：

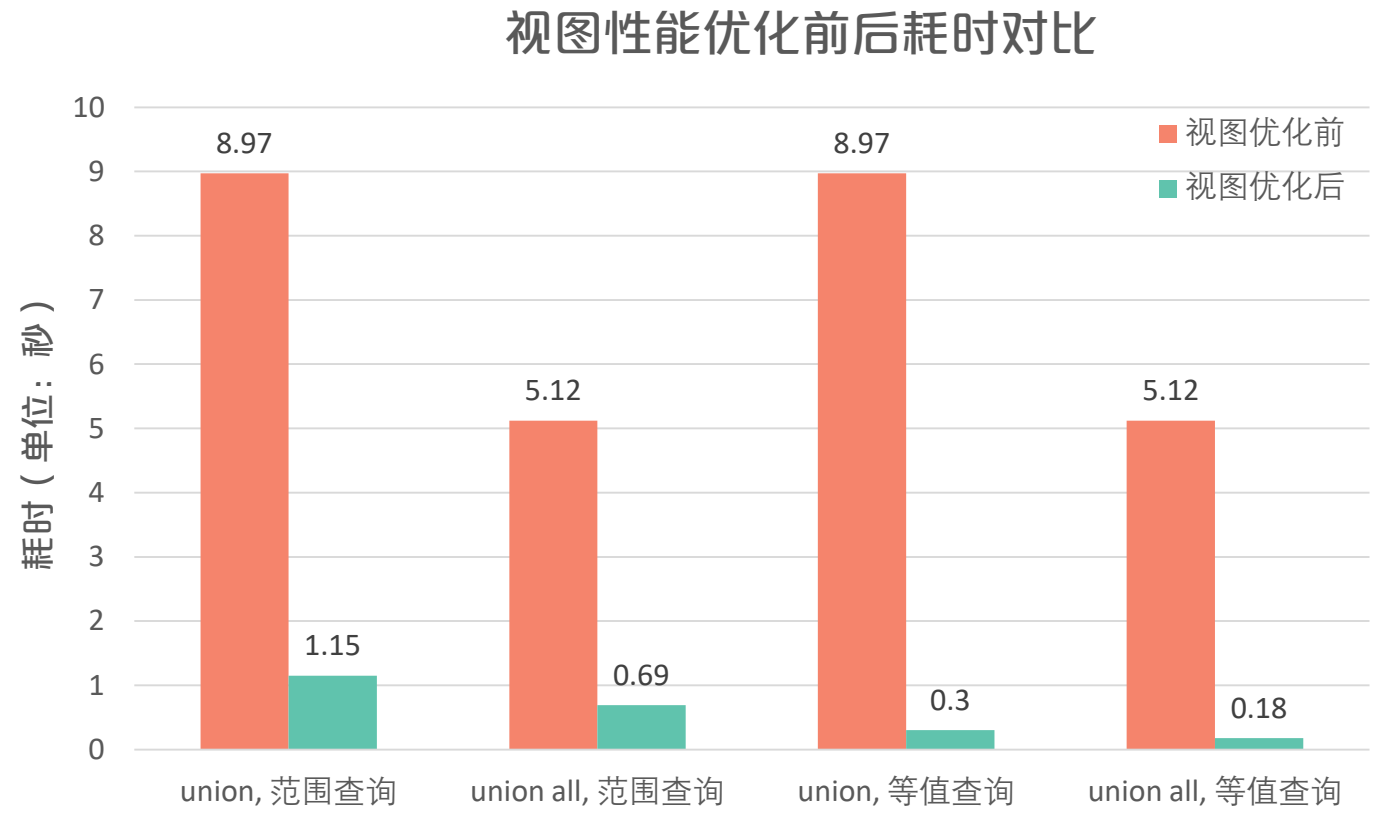
- 按条件查询基表，有可能使用索引，效率更高
- 符合条件的数据生成临时表，结果集更小，节约内存和临时表空间



UPSQL特性

□ 性能

- 线程池
- 多队列并发控制
- 热点数据更新优化
- Union视图性能优化
- 无效链接自动清理
- ....

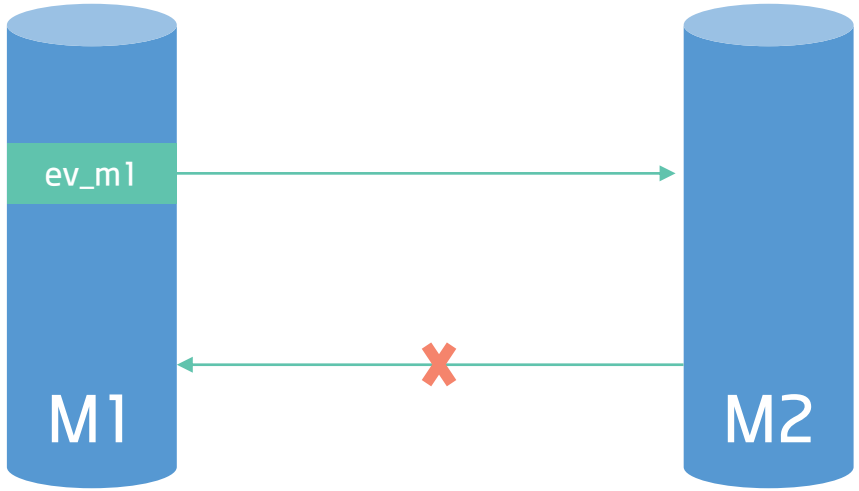


以某日志表为例，创建4张结构相同的表，并插入少量数据（每张表3万行），然后在4张表上建立视图。
将视图外层条件下推至基表，利用基表索引，查询性能提升非常明显。
仅上述12万数据总量的视图，改写后性能提升至原来的7倍~28倍。
基表数量越多，数据量越大，性能提升的效果越好。

UPSQL特性

复制优化

- relay-log读写锁优化
- 复制容错处理
- 主主复制日志过滤
- 自增列参数自动设置
- ....



- 优化前，ev_m1需回传至M1，然后被IO线程丢弃
- 优化后，在M2上即完成日志过滤，避免不必要的回传，节约带宽
- 在主主半同步复制场景减少一次ACK等待
- M2日志位点更新通过心跳事件通知M1

UPSQL特性

其他

- Binlog闪回工具
- 客户端库负载均衡支持
- 事务超时自动回滚
- 表空间碎片统计
- Informatica-ETL兼容
- 在线快速动态加字段
- 运维工具
- ……

自研组件

需求驱动


如何解决数据库高可用问题
如何实现读写分离、数据拆分
Proxy
如何迁移或同步数据
同步时如何满足个性化形变需求
Moray
如何解决数据库资源问题
如何高效运维大规模数据库
DBaaS

UPSQL-Proxy

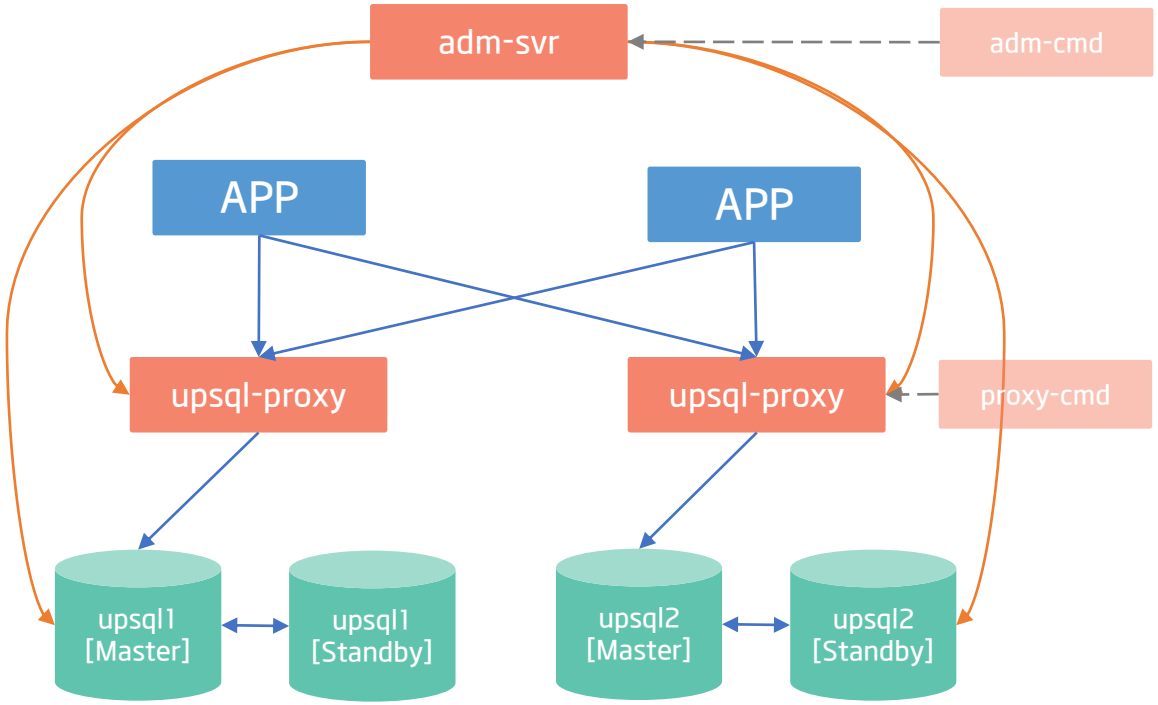
功能列表

核心功能

- 高可用
- 读写分离
- 数据拆分
- 连接池
- 分布式事务
- 支持Prepare(stmt)
- 负载均衡

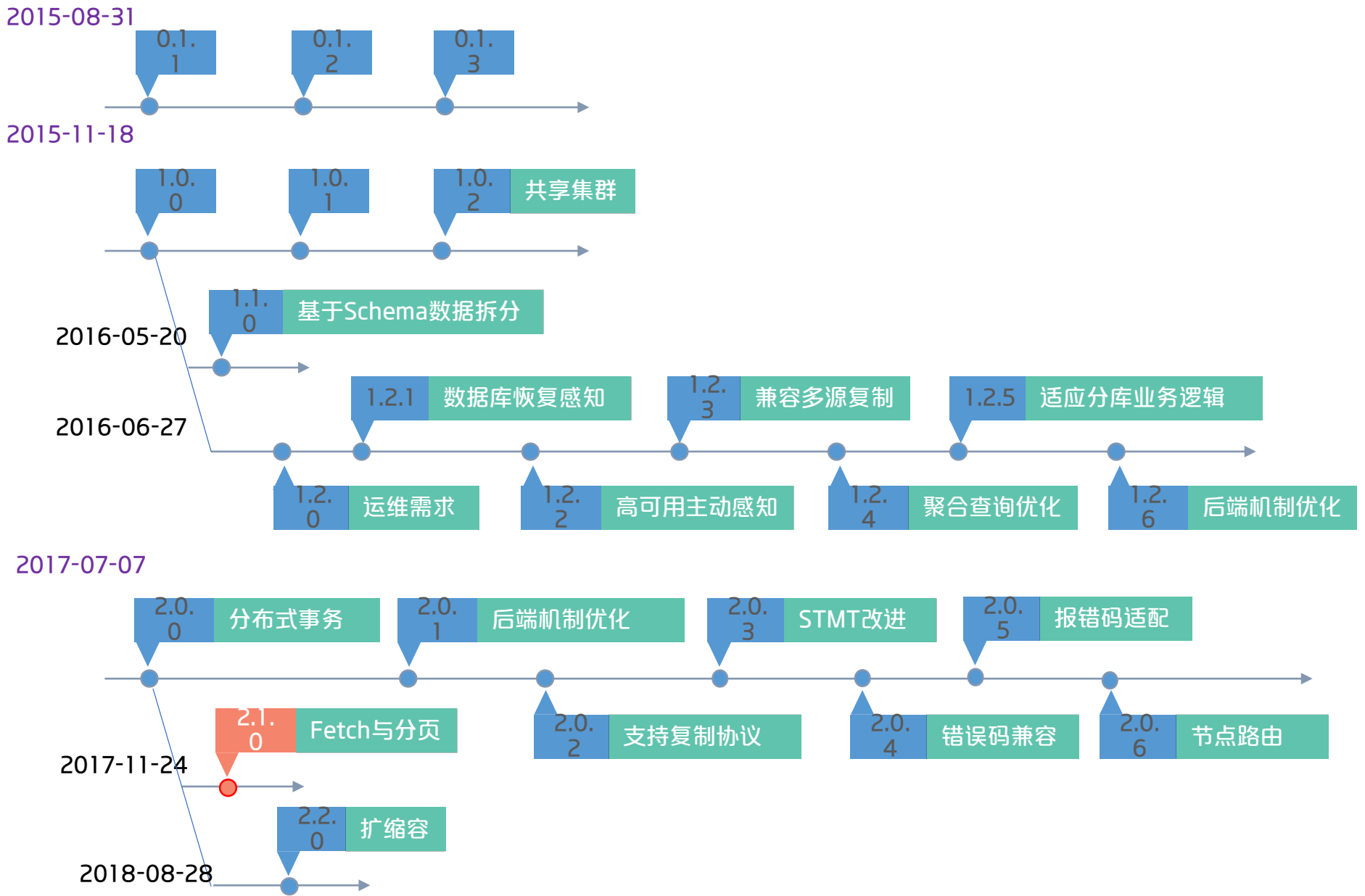
管理类功能

- 多租户
- 管理可扩展
- 日志归档
- IP控制
- 参数动态生效
- DBPM（统一密码服务器）



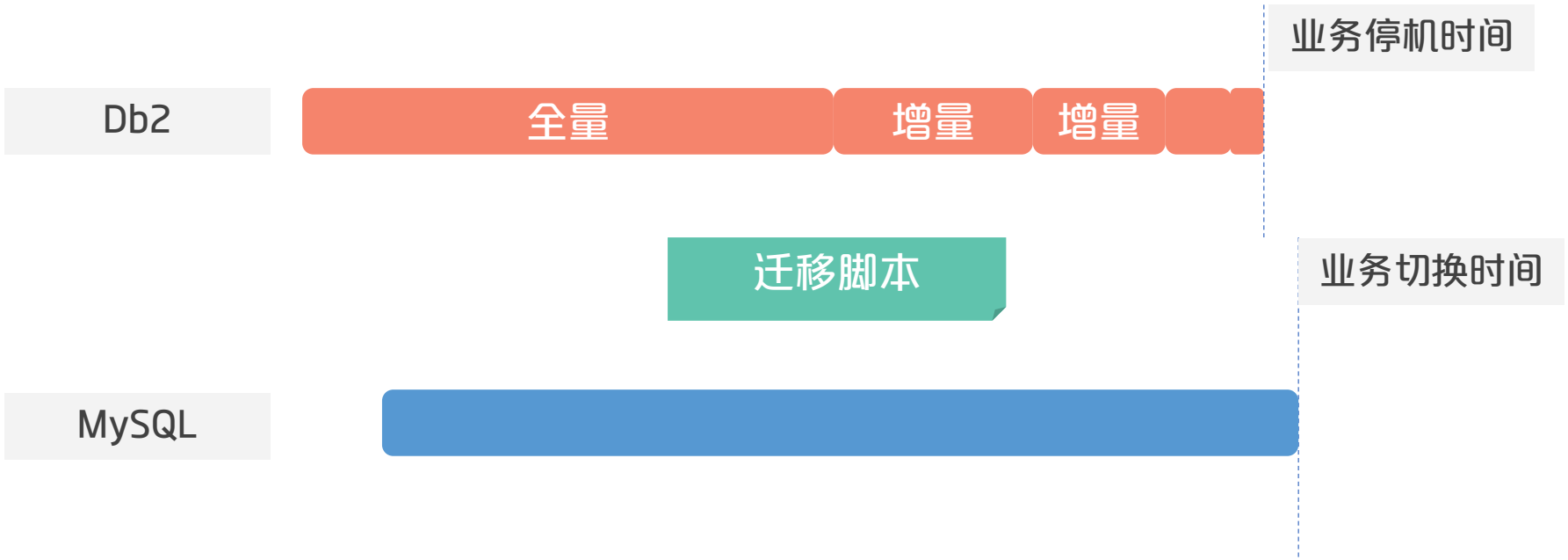
UPSQL-Proxy

版本历史



Moray前身

Db2数据迁移



一组Python脚本实现

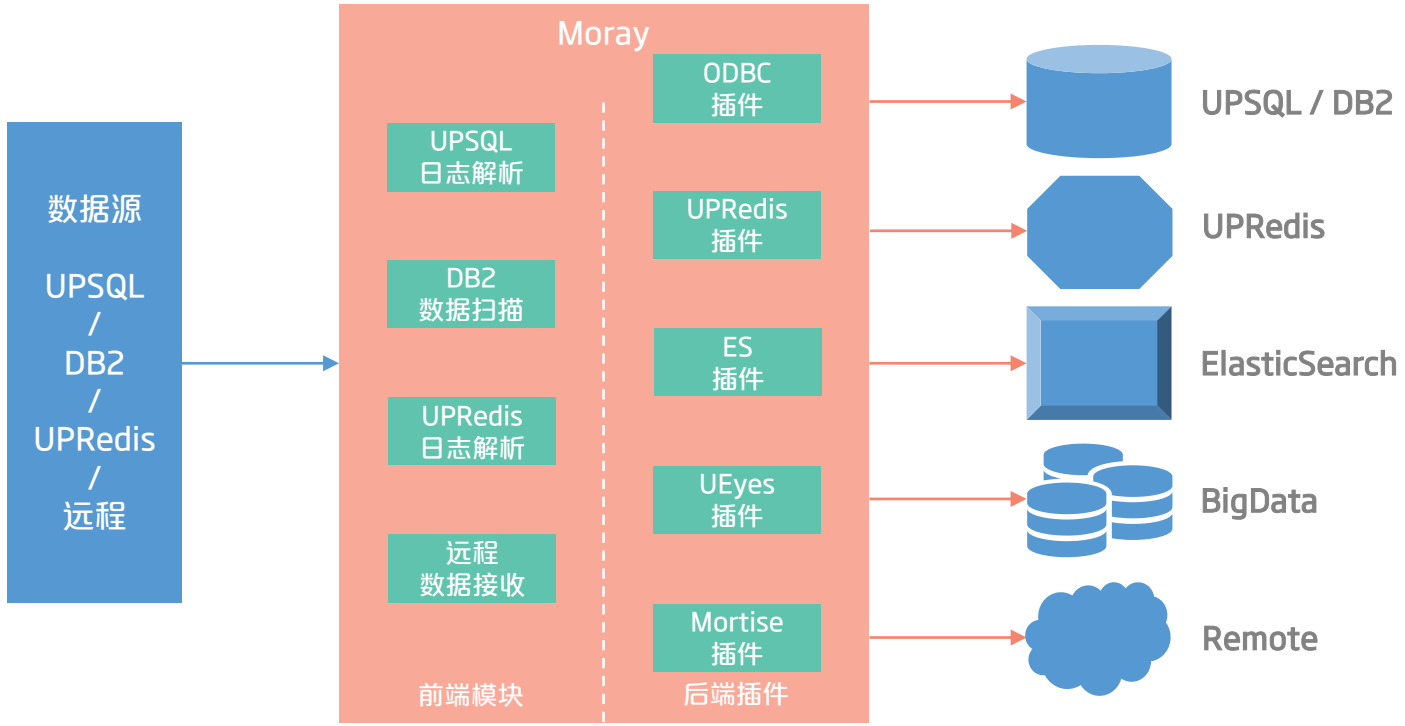
Moray前身

数据同步至大数据

UPSQL-Mover通过解析MySQL/UPSQL的binlog获取数据，并同步至大数据平台

Moray架构设计

前后端分离



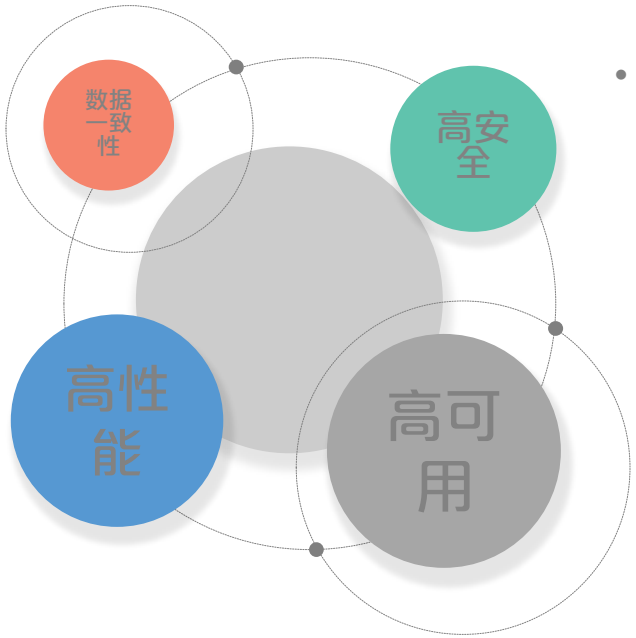
前后端模块插件化设计，支持不同数据源和目标数据库的组合。

Moray架构设计

设计要点

- 尽可能不拆分事务，保证事务原子性和数据一致性。

- 尽可能不直接读源表，不与应用竞争数据库资源；数据压缩传输，节约带宽；多线程并行回放，并优化并行算法。



- 断点续传保证数据零丢失；报文校验防数据篡改。

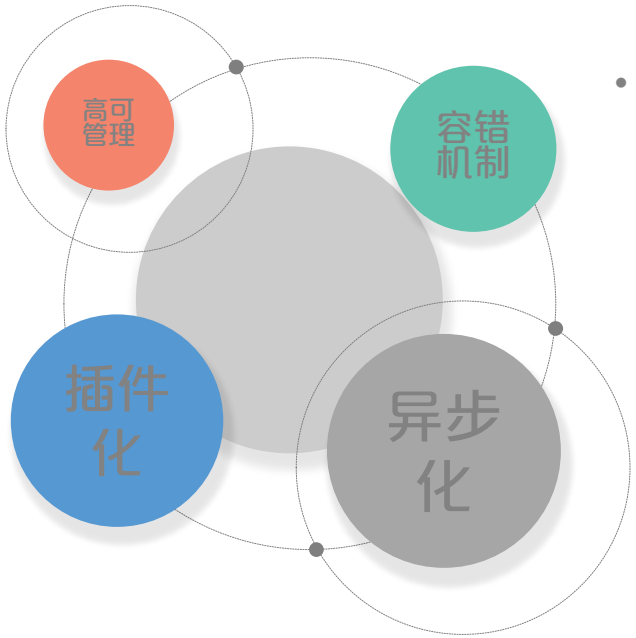
- 采用端到端同步，去中心化，各业务之间互不影响；独立运行，降低与源库的耦合度；可根据数据库架构灵活部署。

Moray架构设计

设计要点

- 管理监控平台实现节点管理和运行状态监控。如：同步进度、时延。

- 总体分为前端模块和后端插件，模块和插件均易于扩展，未来可支持更多数据源和目标库。



- 针对多种异常场景设计了容错策略，如：自动过滤重复数据；数据异常自动转换补偿等。

- 组件之间接口异步化，无阻塞，后一组件通过异步ACK给前一组件应答，提高整体性能。

数据形变

- 使用lua脚本配置数据形变功能

```lua
function moray_df()
    res = {} -- 存储行记录内容
    origin_table_name = odbc.table_name -- 源库的表名，odbc.table_name值为ORIGIN_TABLE
    col_value = odbc.ORIGIN_COL  -- 源库的ORIGIN_COL字段值
    if  (string.sub(col_value,-1) >="0" and string.sub(col_value,-1) <="4")
    then
        res.table_name = origin_table_name.."01" --  表名变换，替换为"schema.table"格式
    else
        res.table_name = origin_table_name.."02" -- 表名变换，替换为"schema.table"格式
    end
    return res -- 返回行记录
end
```

数据形变

又一例

```
function moray_df()
    res = {} -- 一行转多表：  第一张表
    res.table_name = "sbtest.test" -- 表名变换
    local column_value = to number(odbc.settle_dt)
    local skipvalue = column_value % 3
    if (skipvalue == 2) then
        res.skip = true -- 是否跳过该条记录
    end
    res.change_name = {}
    res.change_name.expire_dt = "expire_datetime" -- 列名修改
            res.col = {}
    res.col.expire_datetime = "1234" -- 列值配置值，定义row后才生效，否则不会修改列值
    res.row = {}
    res.row[1] = {} -- 一行转多行：第一行
    res.row[1].transmsn_dt_tm = "100" -- 主键修改为100
    res.row[1].expire_datetime = "5678" -- 必须定义row才可以实现列值修改，第一行的列值修改，覆盖上述的通用修改值"1234"
    res.row[2] = {} -- 一行转多行：第二行
    res.row[2].transmsn_dt_tm = "101" --  主键修改为101

    res2 = {} -- 一行转多表：  第二张表
    res2.table_name = "sbtest.test2" -- 表名表换
    res.change_name.expire_dt = "expire_datetime222" -- 列名修改

    res3 = {} -- 一行转多表：  第三张表 = 原表

    return res, res2, res3 -- 一行转多表，返回多个res
end
```

Moray适用场景

同构或异构数据库间数据同步

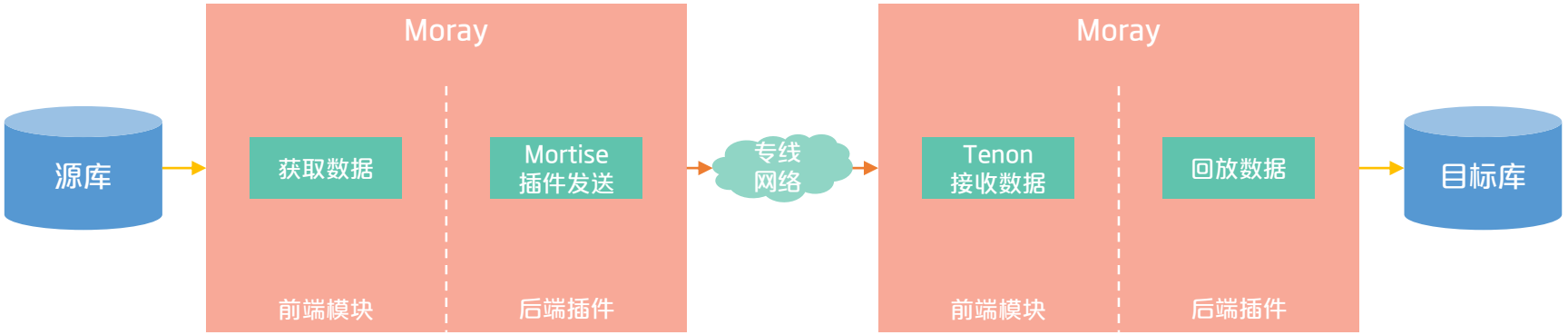
目标库 / 数据源	UPSQL	DB2	UPRedis	ElasticSearch	UEyes	Kafka
UPSQL	√	√		√	√	
DB2	√	√		√	√	
UPRedis	√		√			

根据数据源划分：

- UPSQL：解析源库binlog日志获取数据，通过ODBC插件写入UPSQL或DB2数据库，或通过ES插件写入ElasticSearch等；用于UPSQL实时数据同步。
- DB2： 扫描源库表格获取数据，通过ODBC插件写入UPSQL或DB2；用于一次性数据迁移。
- UPRedis： 解析源库AOF日志，通过redis插件写入redis，或通过ODBC插件写入UPSQL；用于UPRedis缓存数据实时同步，或持久化至数据库。

Moray适用场景

异地数据同步

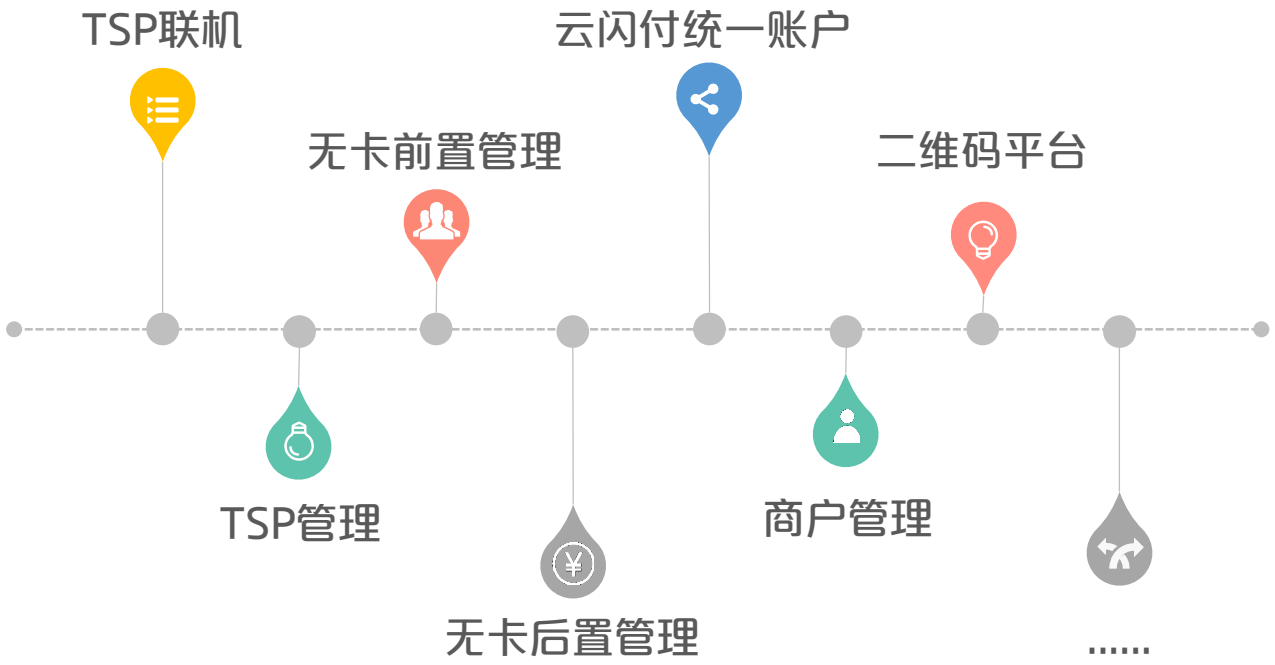


Moray通过级联实现异地数据同步；

源库端Moray获取数据，并将数据发送至异地；

目标库端Moray接收数据，并将数据回放至目标数据库。

Moray应用案例

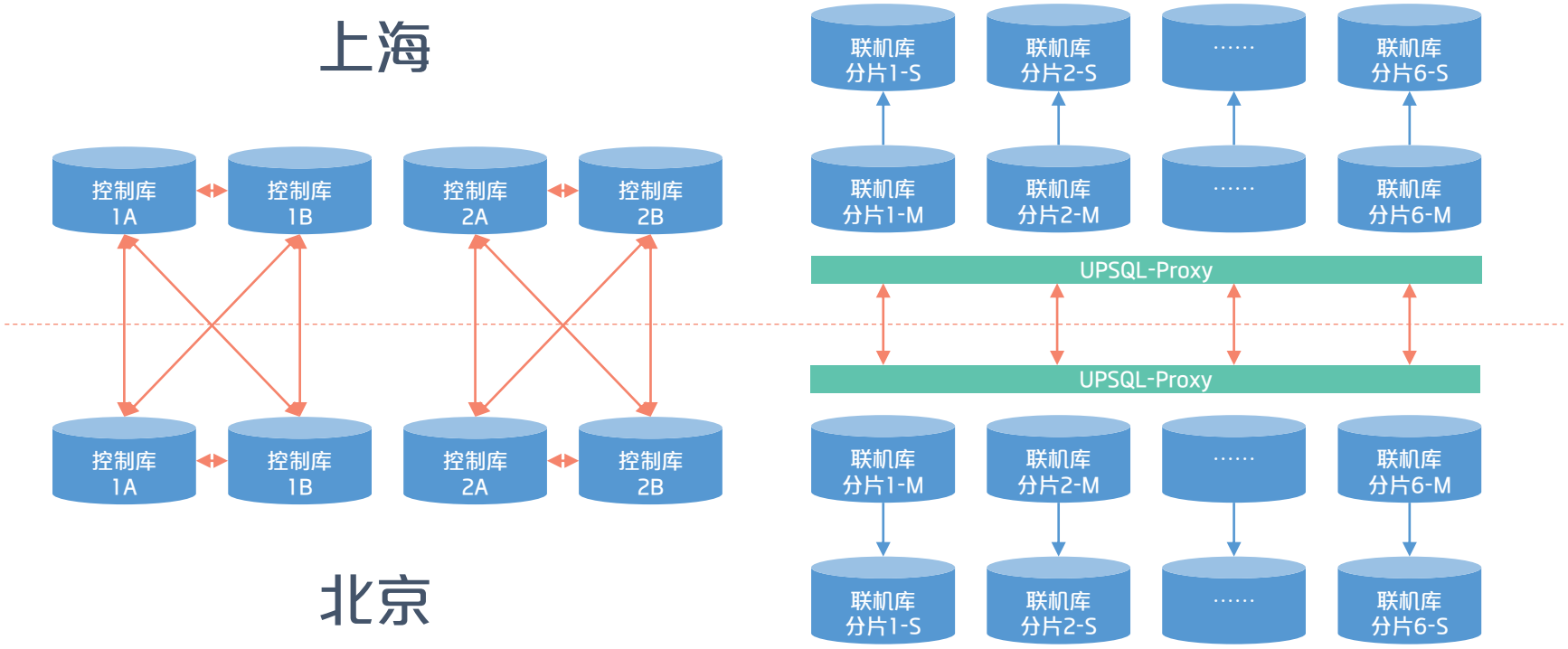


已上线

正在建设

Moray应用案例

TSP异地双活



TSP标记化服务平台全力支撑银联各类手机Pay产品

- Huawei Pay
- Mi Pay
- Apple Pay
- Samsung Pay

Moray同步

MySQL复制

DBaaS

整体架构

- 自动化部署运维
- 快速响应需求
- 资源弹性、合理配置
- 资源隔离、降低干扰
- 标准化流程、安全可靠

应用情况

银联内部270+生产系统， 2100+实例

涉及：移动支付、公缴、账务、清算、风险等

推广至18家银联分公司，2家银联子公司

与上海银行深度合作三年以上

随市场化项目走入更多机构和合作伙伴

包括：城商行、证券、航空公司、公共交通、商超等

落地境外泰国、老挝等国家转接清算中心

在银联金融行业云上为行业用户提供数据库服务

谢 谢 ！