DTCC

第十一届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2020

架构革新 高效可控

IT168.com ChinaUnix.net ITPUB

北京国际会议中心 | 2020/12/21-12/23

从传统oracle数据库往云时代MySQL数据库的转型实践

兴业数金 林春

ORACLE WDP OCM讲师

DB2CHINA性能调优版版主

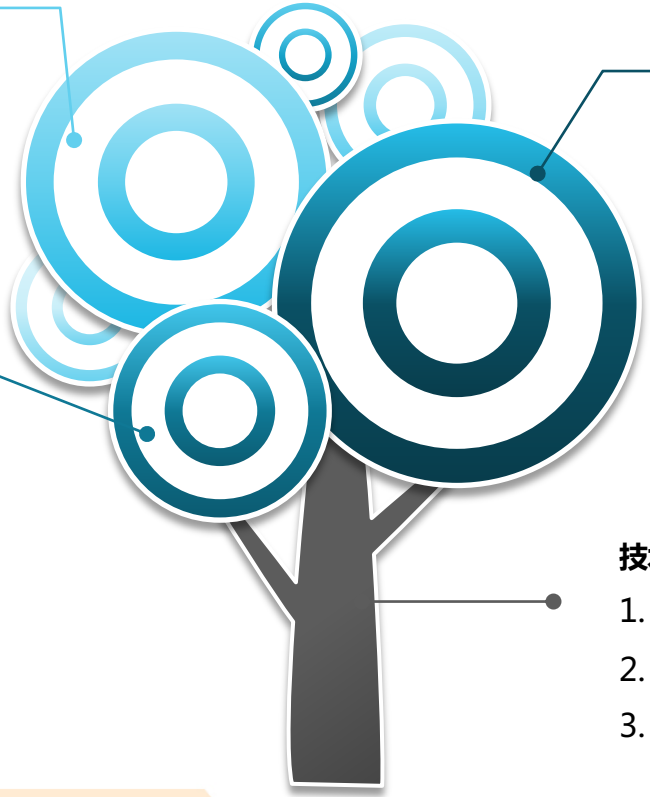
兴业数金首席数据库专家

QQ:1819442969

为什么选择MySQL

节约成本

1. 社区版免费，和企业版代码一样
2. 无须额外的昂贵的硬件
3. 运维管理工具丰富



自主可控

1. 代码完整开源开放
2. 代码更新频繁
3. 兼容国产化软硬件

架构丰富

1. 轻量级，易掌握
2. 多种架构灵活应对不同业务需求

技术生态成熟

1. 市场覆盖率高
2. 市场相关人才储备多
3. 商业资源丰富

O和M特点“大”比较

ORACLE® MySQL™

总体功能	对OLTP类型和OLAP类型支持比较全面	侧重OLTP类型，MySQL8.0引入Hash Join,、窗口函数、With公共表表达式
OLTP场景	并发量不大的情况下，Oracle的读写性能优于MySQL。在数据库连接达到较高并发量，由于Oracle连接是进程，连接时性能损耗更大	有较高并发量时，MySQL产生日志量少于Oracle，不易产生I/O瓶颈
索引功能	索引维护消耗资源少于MySQL	索引维护消耗资源较大，当MySQL的索引个数超过五个时，性能下降更为明显
表特点	Oracle的表是按照堆组织	MySQL的Innodb存储引擎表引擎表是按照主键顺序组织
水平扩展	Oracle数据库由于其极高的一致性要求，造成架构上的扩展存在限制	MySQL原生分布式架构的优势在于并发性较好，架构灵活性较高。
并发性	Oracle使用行级锁，每条记录头部锁字节记录ITL槽事务，只锁定sql需要的资源，不依赖索引	MySQL也可以使用行级锁，但这个行级锁的机制依赖于表的索引
数据持久性	Oracle的重做日志相当于逻辑日志和物理日志的结合，可以靠联机在线日志恢复客户已经提交的数据	MySQL使用InnoDB存储引擎提供redo/undo、两阶段提交特性以及binlog日志保证数据的完整性和持久性
维护性	Oracle 提供了丰富的工具，具有非常大的优势	MySQL原生工具相对简单

去O我们需要什么

MySQL运维管理平台（基础数据库环境保障）+ 去O工具（自动化流程）+ 专业团队（方法论和经验）

让企业可以完成全站数据库100%无缝的从Oracle迁移至MySQL，在漫长的去O改造更换数据库的过程中，保障业务系统全程高效、 平稳、无风险。

应用开发团队

业务流量切换

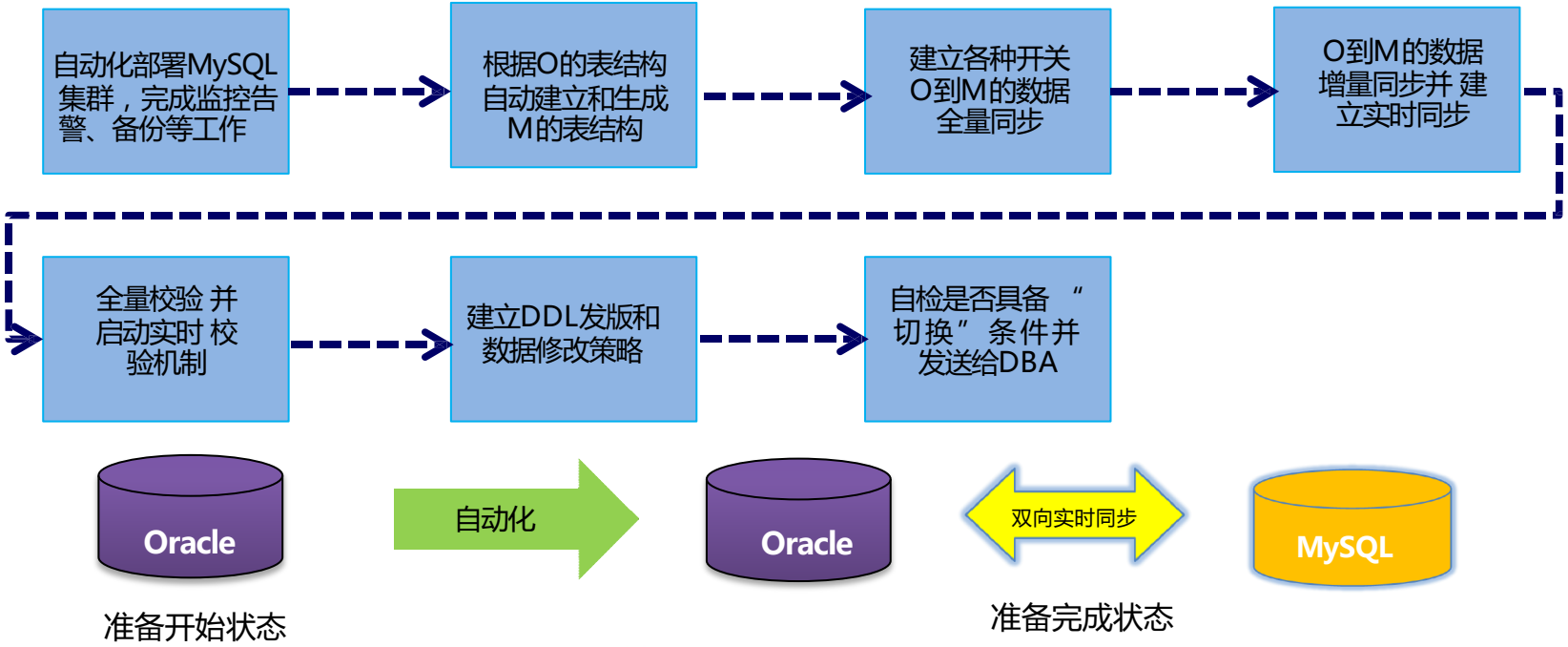
O to M数据迁移、校验和双写工具	Oracle SQL自动转MySQL SQL
业务逻辑梳理和库表单元的拆分	O to M数据字典转化、版本管理和发版
去O双写状态下兼容数据库一键切换自愈工具	Oracle存储过程智能转java代码

MySQL 自 动 化 运 维 管 理 平 台

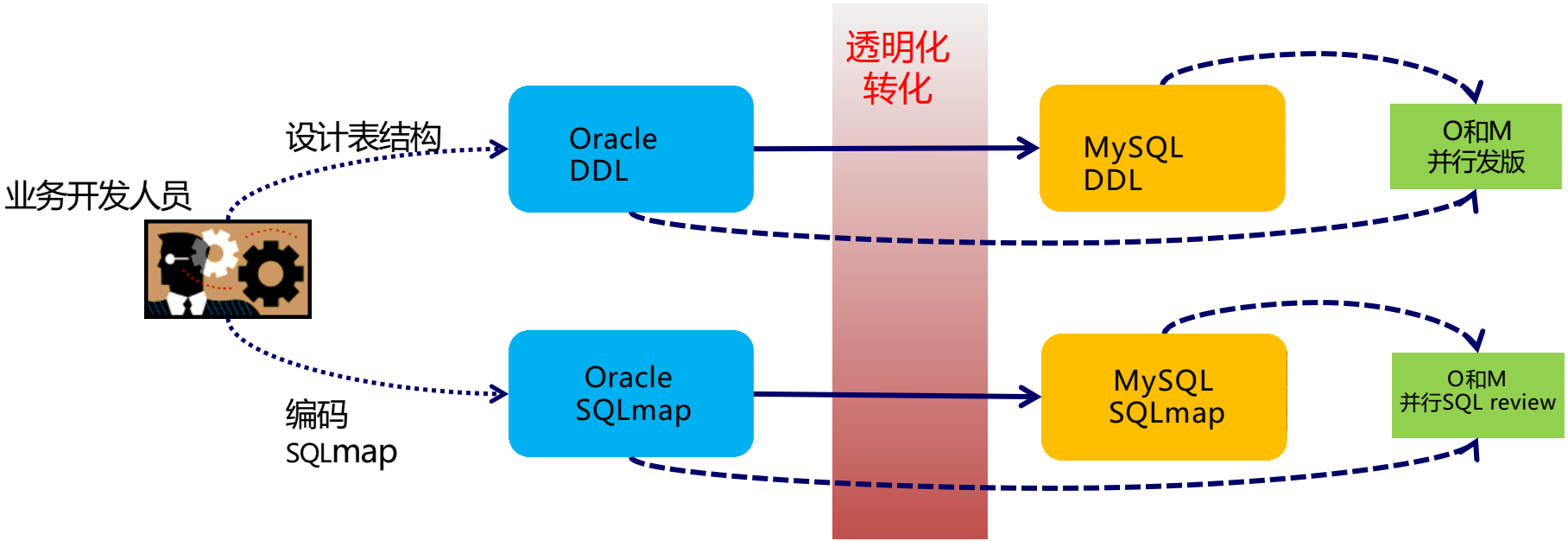
运维保障团队

去O初始化工作内容

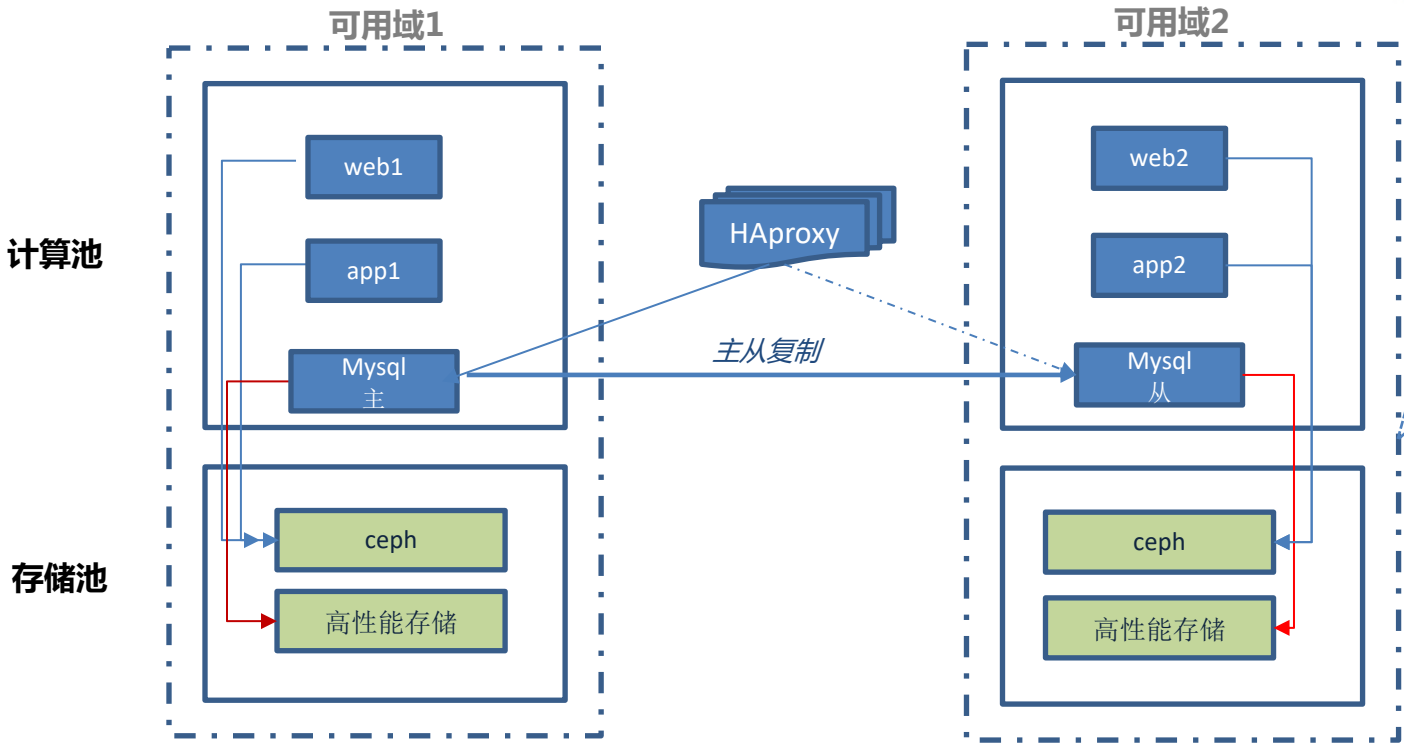
从MySQL上线到生产流量切换至MySQL期间的所有工作借助工具和脚本自动化完成。



O和M并行期间业务改造和版本发布

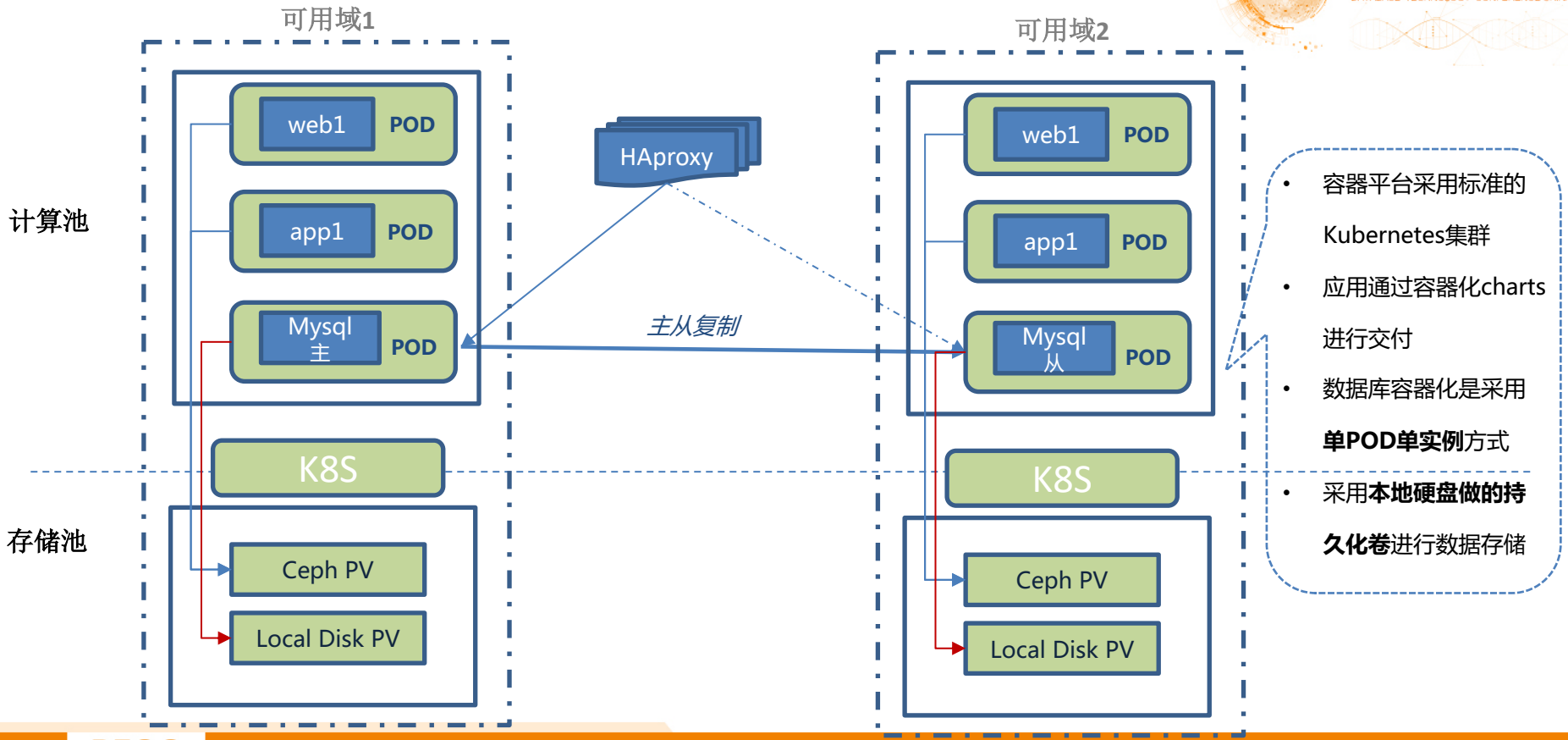


基于云平台架构演进的MySQL设计

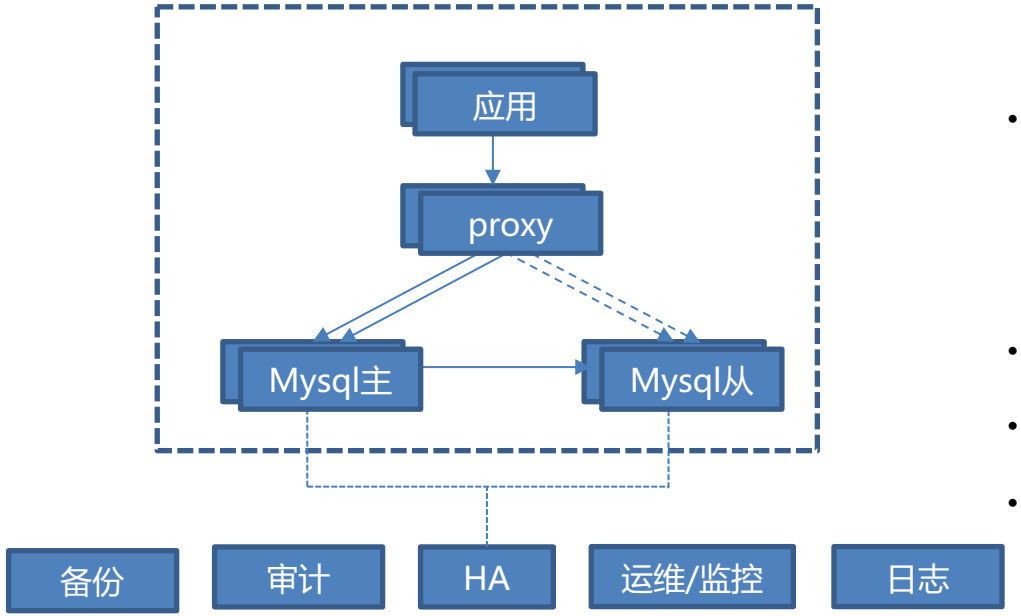


- 可用域之间是独立的资源池，**最大不相关**
- 应用跨可用域部署
- 数据库通过**主从复制**、**HA**组件实现
- 规避池级别的故障，进一步提升**高可靠性**
- **银行云100%的应用**

Mysql容器化设计

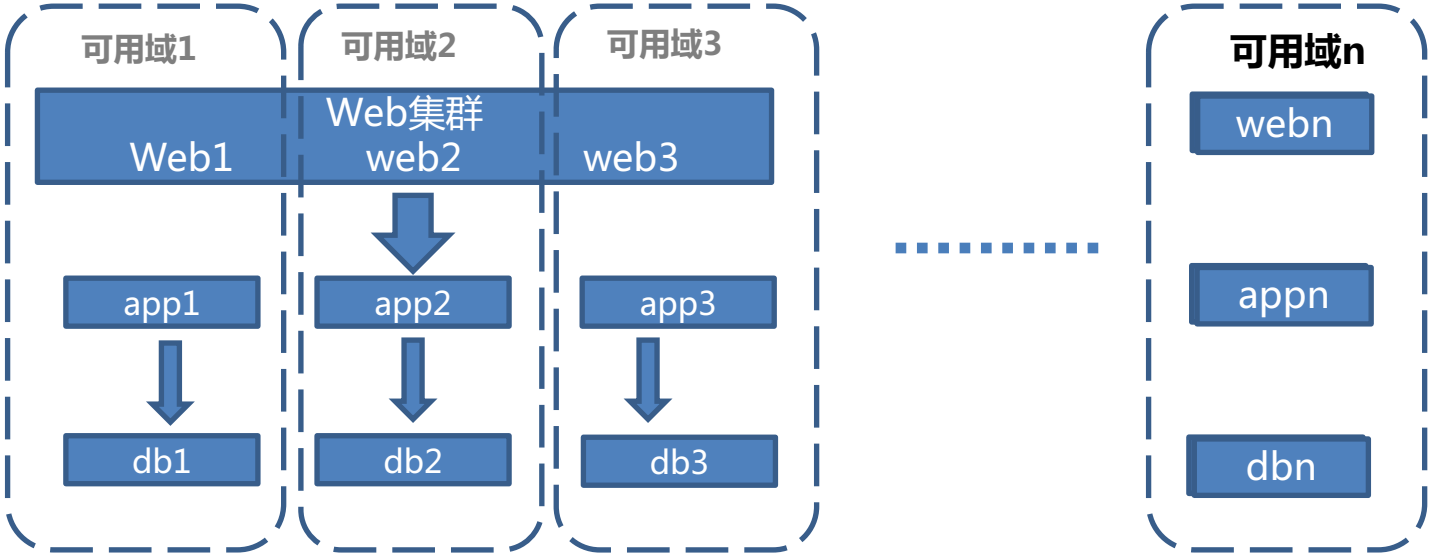

可用域1
可用域2
计算池
存储池
web1
POD
app1
POD
Mysql
主
POD
HAproxy
主从复制
web1
POD
app1
POD
Mysql
从
POD
K8S
K8S
Ceph PV
Local Disk PV
Ceph PV
Local Disk PV
• 容器平台采用标准的Kubernetes集群
• 应用通过容器化charts进行交付
• 数据库容器化是采用**单POD单实例**方式
• 采用**本地硬盘做的持久化卷**进行数据存储

MySQL HA服务云化



- **Mysql HA**服务标准化、云化，作为云服务的一个产品
- **数据库服务**方便可得，云内实现了HA切换、备份、监控等多种功能，解决了传统数据库运维中60-80%的工作量
- 对不熟悉MySQL的客户提供快速上线的可能
- **备份**：自动备份，支持全量增量等备份方式
- **运维监控**：自动化运维管理监控，大幅度提高运维效率

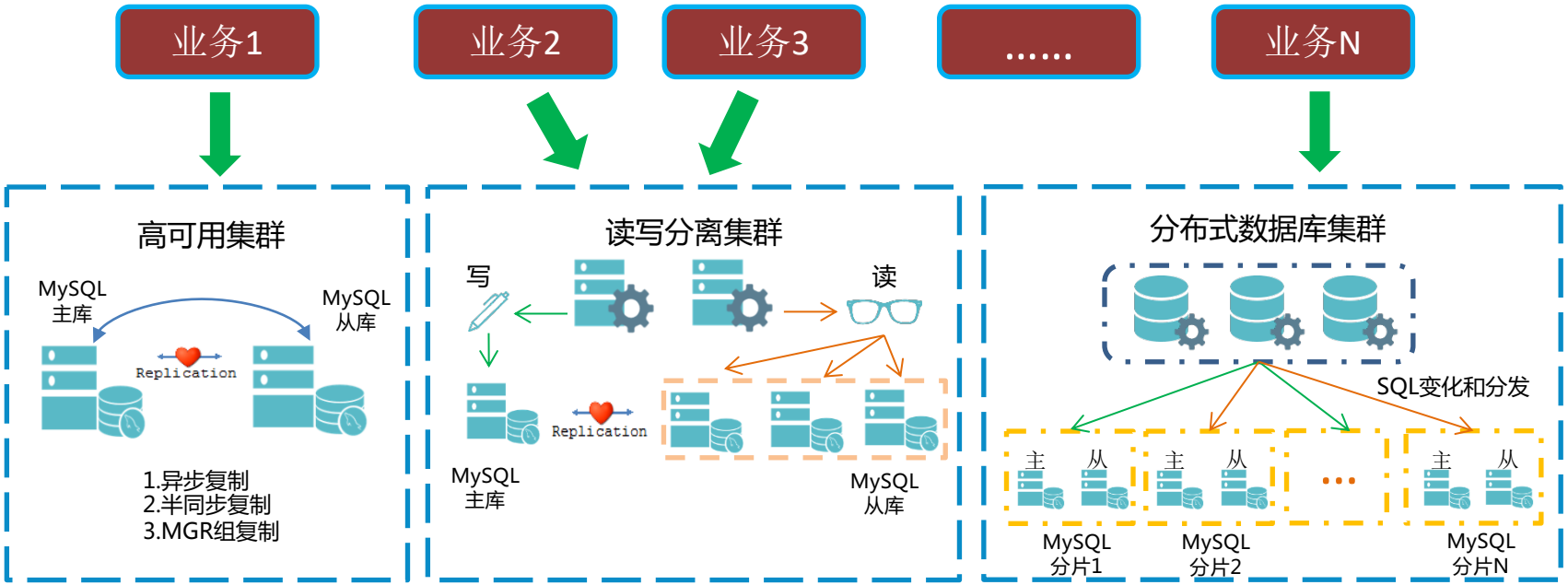
云上MySQL读写分离、分库分表



特点：MySQL天生适应云的架构，适应横向扩展

读写分离、分库分表需要人力调研改造的介入，无法云服务标准化

MySQL数据库架构选择

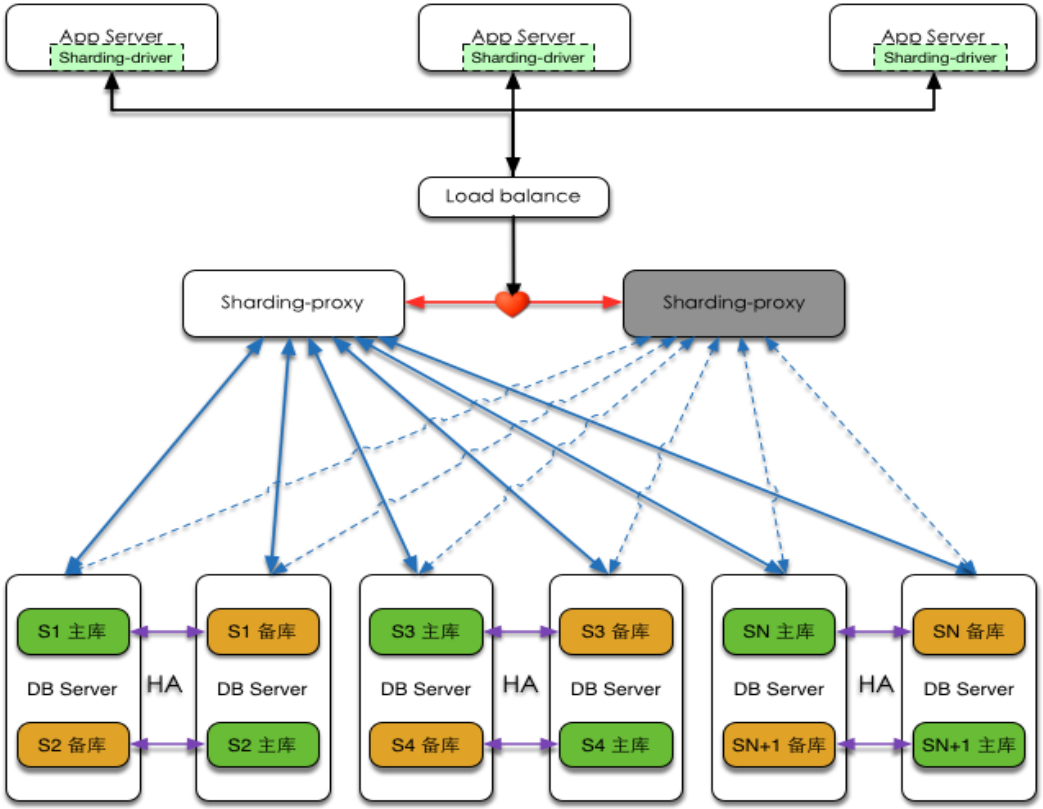


- ✓ 数据量较小，单库控制在1TB以内
- ✓ 单表记录数控制在5000W以内

- ✓ 数据量较小，单库控制在1TB以内，单表记录数控制在5000W以内
- ✓ 读写比差别大，可达到5:1以上

- ✓ 数据量较大，数据库单库数据量1TB以上，单表记录数超过1亿条
- ✓ 高并发写入，要求较高TPS

MySQL 分布式架构（分库分表）



- 数据自动水平拆分
- 集群对外提供统一接口，应用访问透明
- 支持MySQL所有SQL语法及访问接口
- 支持Hash/Range/List等多种分片算法，可扩展
- 支持跨拆分表复杂查询
- 支持事务
- 分片节点冗余高可用，故障自动切换
- MySQL作为存储节点，灵活扩展,支持主流MySQL版本

分布式是个复杂的系统工程

分布式中间件选型

数据分布情况

系统的软硬件配置

开发体系规范

合理的分片方案及拆分算法

适合的应用场景

分布式架构规划

分布式整体调优能力

SQL的分布式改造及优化

数据生命周期分析

源码级技术支持能力

完善的事务补偿机制

运维体系规范

数据分片基本原则


分片原则
1.全局表
千万以内的表不建议分片，通过索引、读写分离等方式提高性能
2.分片数量
查询SQL跨分片越多，则总体性能越差
3.数据均匀
SQL整体执行时间，取决于执行最慢节点的响应时间。分片字段及算法很关键
4.扩容
需考虑数据的增长模式、分片扩容，其中范围、一致性hash等，有利于后续扩容操作
5.应用配合
应用代码尽量规避部分跨分片操作，特殊情况下，部分需求在应用端实现

分片字段选择

- 选取的字段需符合：业务场景不会跨节点访问，数据分布相对均匀

Oracle转型MySQL常见的“坑”

a) 隔离级别与间隙锁
b) 索引设计考虑
c) 避免Truncate大表 - Bug编号68184
d) 避免使用分区表
e) 有归档数据需求的表，建议增加create_time、update_time字段分别表示写入时间和最后更新时间
f) 并发
g) 角色管理
h) insert...select语句

mysql案例1/4

故障现象：

交易连接失败

故障原因：

大量并发连接执行以下语句做绑卡查询：

```
select
ta.ID_SERIAL,ta.ID_USER,ta.CARD_NO,ta.CARD_TYPE,ta.MOB
ILE,ta.HOLDER_NAME,ta.CHANNEL,ta.SEQ,ta.IS_DEF,ta.CRE_T
IME,ta.UPD_TIME,ta.ID_CARD_AUTH_LOG,ta.DEF_CREDIT,tb.
CARD_BIN,tb.CARD_BIN_NAME from T_CARD_AUTH
ta,T_CARD_BIN tb where ta.ID_USER
='2246450297264151557' AND ta.status='1' AND
ta.CARD_NO like concat(tb.CARD_BIN,'%') order by
ta.CRE_TIME desc;
```

内存溢出

Zabbix监控

mysql案例1/4

MYSQL内存计算及优化

```
select
(@@key_buffer_size + @@query_cache_size +
@@tmp_table_size
   + @@innodb_buffer_pool_size +
@@innodb_additional_mem_pool_size
   + @@innodb_log_buffer_size
   + @@max_connections * (
      @@read_buffer_size + @@read_rnd_buffer_size
      + @@sort_buffer_size+ @@join_buffer_size
      + @@binlog_cache_size + @@thread_stack
   )
)/1024/1024/1024;
```

SQL语句的优化，避免不必要排序：

```
# User@Host: xlife[xlife] @  [10.32.4.172]  Id:
184943908
# Query_time: 5.554402  Lock_time: 3.566839
Rows_sent: 0  Rows_examined: 0
select
ta.ID_SERIAL,ta.ID_USER,ta.CARD_NO,ta.CARD_TYPE
,ta.MOBILE,ta.HOLDER_NAME,ta.CHANNEL,ta.SEQ,t
a.IS_DEF,ta.CRE_TIME,ta.UPD_TIME,ta.ID_CARD_AUT
H_LOG,ta.DEF_CREDIT,tb.CARD_BIN,tb.CARD_BIN_N
AME from T_CARD_AUTH ta,T_CARD_BIN tb where
ta.ID_USER ='2246450297264151557' AND
ta.status='1' AND ta.CARD_NO like
concat(tb.CARD_BIN,'%') order by ta.CRE_TIME desc;
```

mysql案例2/4

故障现象：

数据库连接过高，导致高可用管理用户不能够连接，因此高可用监控进程判断数据库主库状态异常，从而停止半同步复制连接，关闭应用连接，切换到主库

优化切换步骤：

先切换到备库，再关闭故障主库

优化limit子句：

Limit M,N语句不是跳过M行，而是取M+N行，然后返回放弃前M行，返回N行。当M特别大的时候，通过控制返回总行数，或者对超过阈值的页数进行SQL改写进行优化。

SQL语句优化：

```
# Query_time: 2834.274757  Lock_time: 0.000417 Rows_sent: 0  Rows_examined: 717806

select tu.USER_ID,tu.USER_NAME,case when tu.USER_STATUS='Y' then '??[7m<8C>?[7m<81>' when tu.USER_STAT
US='N' then '?7m<9C>a?[7m<8C>?[7m<81>' else '?7m<9C>a?[7m<8C>?[7m<81>' end as USER_STATUS,case when tu
.USER_SEX='M' then '?7m<94>Â·' when tu.USER_SEX='F' then '?' else '' end as USER_SEX,tu.CERT_NO,tu.MOB
LE,date_format(tu.CRE_TIME,'%Y-%m-%d %H:%i:%s') as CRE_TIME,date_format(tu.UPD_TIME,'%Y-%m-%d %H:%i:%s
') as UPD_TIME,date_format(tl.Min_CreTime,'%Y-%m-%d %H:%i:%s') as Min_CreTime
FROM T_USER_INFO tu LEFT JOIN (SELECT tcl.ID_USER,MIN(tcl.CRE_TIME) AS Min_CreTime FROM T_CARD_AUTH_LO
G tcl WHERE tcl.AUTH_TYPE = 0 GROUP BY tcl.ID_USER) tl
ON tu.USER_ID = tl.ID_USER
where 1=1 and tu.CRE_TIME >= '2018-03-19 00:00:00' and tu.CRE_TIME <= '2018-09-19 23:59:59'
limit 80000, 5000;

+----+-------------+------------+------------+-------+-------------------------+-------------------
------+---------+-----------------+--------+----------+----------------------+
| id | select_type | table      | partitions | type  | possible_keys           | key
      | key_len | ref             | rows   | filtered | Extra                |
+----+-------------+------------+------------+-------+-------------------------+-------------------
------+---------+-----------------+--------+----------+----------------------+
|  1 | PRIMARY     | tu         | NULL       | range | idx_tuserinfo_cre_time  | idx_tuserinfo_cre_t
ime    | 5       | NULL            | 494287 |   100.00 | Using index condition |
|  1 | PRIMARY     | <derived2> | NULL       | ref   | <auto_key0>             | <auto_key0>
       | 8       | xlife.tu.USER_ID |     10 |   100.00 | Using where           |
|  2 | DERIVED     | tcl        | NULL       | index | IDX_CARD_AUTH_LOG_USER_NO | IDX_CARD_AUTH_LOG_U
SER_NO | 106     | NULL            | 684008 |    10.00 | Using where           |
+----+-------------+------------+------------+-------+-------------------------+-------------------
```

mysql案例2/4

SQL语句优化：

```
# Query_time: 2834.274757  Lock_time: 0.000417 Rows_sent: 0  Rows_examined: 717806

select tu.USER_ID,tu.USER_NAME,case when tu.USER_STATUS='Y' then '??[7m<8C>?[7m<81>' when tu.USER_STAT
US='N' then '??7m<9C>a?[7m<8C>?[7m<81>' else '??7m<9C>a?[7m<8C>?[7m<81>' end as USER_STATUS,case when tu
.USER_SEX='M' then '??7m<94>Â·' when tu.USER_SEX='F' then '?' else '' end as USER_SEX,tu.CERT_NO,tu.MOB
LE,date_format(tu.CRE_TIME,'%Y-%m-%d %H:%i:%s') as CRE_TIME,date_format(tu.UPD_TIME,'%Y-%m-%d %H:%i:%s
') as UPD_TIME,date_format(tl.Min_CreTime,'%Y-%m-%d %H:%i:%s') as Min_CreTime
FROM T_USER_INFO tu LEFT JOIN (SELECT tcl.ID_USER,MIN(tcl.CRE_TIME) AS Min_CreTime FROM T_CARD_AUTH_LO
G tcl WHERE tcl.AUTH_TYPE = 0 GROUP BY tcl.ID_USER) tl
ON tu.USER_ID = tl.ID_USER
where 1=1 and tu.CRE_TIME >= '2018-03-19 00:00:00' and tu.CRE_TIME <= '2018-09-19 23:59:59'
limit 80000, 5000;

+----+-------------+------------+------------+-------+--------------------------+--------------------
-------+---------+------------------+--------+----------+-------------------------+
| id | select_type | table      | partitions | type  | possible_keys            | key
       | key_len | ref              | rows   | filtered | Extra                   |
+----+-------------+------------+------------+-------+--------------------------+--------------------
-------+---------+------------------+--------+----------+-------------------------+
|  1 | PRIMARY     | tu         | NULL       | range | idx_tuserinfo_cre_time   | idx_tuserinfo_cre_t
ime    | 5       | NULL             | 494287 |   100.00 | Using index condition |
|  1 | PRIMARY     | <derived2> | NULL       | ref   | <auto_key0>              | <auto_key0>
       | 8       | xlife.tu.USER_ID |     10 |   100.00 | Using where           |
|  2 | DERIVED     | tcl        | NULL       | index | IDX_CARD_AUTH_LOG_USER_NO | IDX_CARD_AUTH_LOG_U
SER_NO | 106     | NULL             | 684008 |    10.00 | Using where           |
+----+-------------+------------+------------+-------+--------------------------+--------------------
```

mysql案例3/4

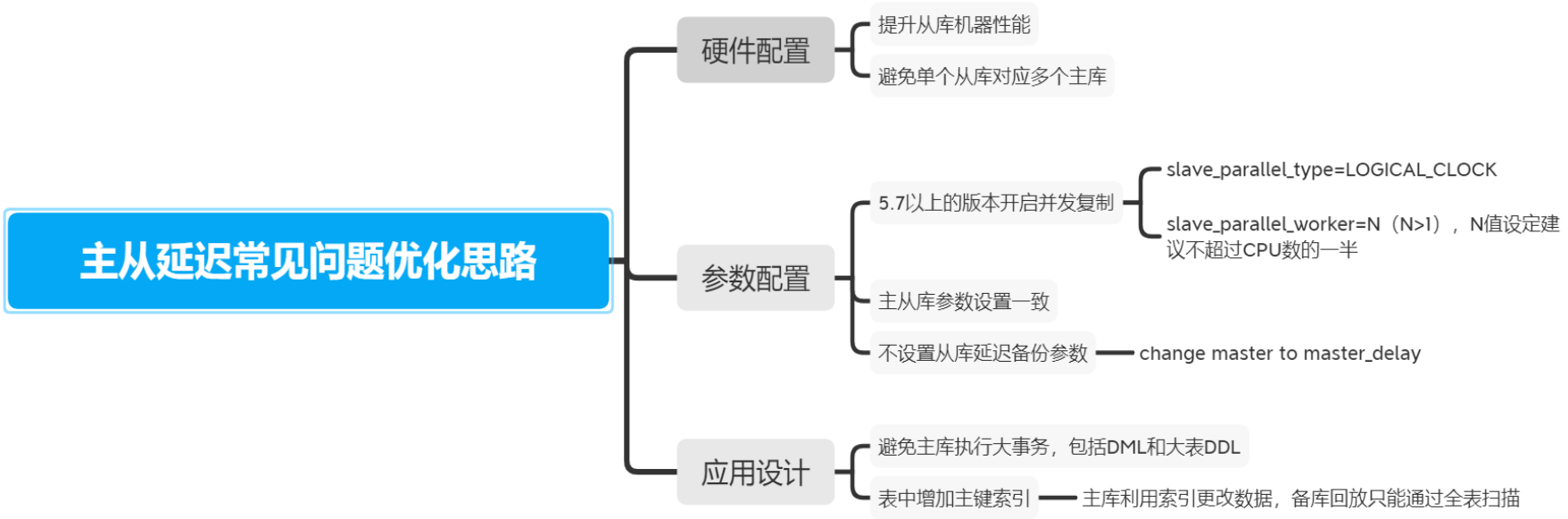
左外连接语句：

```
mysql> SELECT d.department_id, d.department_name,count(e.department_id) FROM   departments d LEFT OUTE
R JOIN employees e  ON   (e.department_id = d.department_id)  group by d.department_id, d.department_n
ame;
+---------------+---------------------+-----------------------+
| department_id | department_name     | count(e.department_id) |
+---------------+---------------------+-----------------------+
|            10 | Administration      |                     1 |
|            20 | Marketing           |                     2 |
|            30 | Purchasing          |                     6 |
|            40 | Human Resources     |                     1 |
|            50 | Shipping            |                    45 |
|            60 | IT                  |                     5 |
|            70 | Public Relations    |                     1 |
|            80 | Sales               |                    34 |
|            90 | Executive           |                     3 |
|           100 | Finance             |                     6 |
|           110 | Accounting          |                     2 |
|           120 | Treasury            |                     0 |
|           130 | Corporate Tax       |                     0 |
|           140 | Control And Credit  |                     0 |
|           150 | Shareholder Services|                     0 |
|           160 | Benefits            |                     0 |
|           170 | Manufacturing       |                     0 |
|           180 | Construction        |                     0 |
|           190 | Contracting         |                     0 |
|           200 | Operations          |                     0 |
|           210 | IT Support          |                     0 |
|           220 | NOC                 |                     0 |
```

标量子查询语句：

```
mysql> SELECT d.department_id, d.department_name,(select count(*) from employees e where e.departmen
id = d.department_id) cnt FROM   departments d;
+---------------+---------------------+------+
| department_id | department_name     | cnt  |
+---------------+---------------------+------+
|            10 | Administration      |    1 |
|            20 | Marketing           |    2 |
|            30 | Purchasing          |    6 |
|            40 | Human Resources     |    1 |
|            50 | Shipping            |   45 |
|            60 | IT                  |    5 |
|            70 | Public Relations    |    1 |
|            80 | Sales               |   34 |
|            90 | Executive           |    3 |
|           100 | Finance             |    6 |
|           110 | Accounting          |    2 |
|           120 | Treasury            |    0 |
|           130 | Corporate Tax       |    0 |
|           140 | Control And Credit  |    0 |
|           150 | Shareholder Services|    0 |
|           160 | Benefits            |    0 |
|           170 | Manufacturing       |    0 |
|           180 | Construction        |    0 |
|           190 | Contracting         |    0 |
|           200 | Operations          |    0 |
|           210 | IT Support          |    0 |
|           220 | NOC                 |    0 |
```

mysql案例4/4


主从延迟常见问题优化思路
硬件配置
提升从库机器性能
避免单个从库对应多个主库
参数配置
5.7以上的版本开启并发复制
slave_parallel_type=LOGICAL_CLOCK
slave_parallel_worker=N（N>1），N值设定建议不超过CPU数的一半
主从库参数设置一致
不设置从库延迟备份参数
change master to master_delay
应用设计
避免主库执行大事务，包括DML和大表DDL
表中增加主键索引
主库利用索引更改数据，备库回放只能通过全表扫描

mysql案例4/4

问题描述：

1、消费信贷多渠道业务系统，数据量2T以上

2、一个实例，一套数据库，不同渠道通过表名yewu1_ 、yewu2_ 类似区分

3、每天有跑批量

4、批量延迟大，过万秒

5、主备数据库

优化方法：

1、硬件优化、提升从库机器性能；

2、数据库参数调整：innodb_flush_log_at_trx_commit、sync_binlog；

3、业务优化，监管数据到大数据平台；

4、分表分库，不同业务分库处理；

THANKS