



全球敏捷运维峰会

分布式数据库及数据中间件能力验证与实践



NewSQL分布式数据库实现方案

New Architecture

- TiDB
- CockroachDB

Transparent Sharding Middleware

- Apache ShardingSphere
- MyCat

Database-as-a-Service

- Amazon Aurora
- PolarDB



基于开源分布式数据库架构选型思考

1. 选择 适合的开源技术

建立正确的开源软件选型标准。
社区成熟度、软件成熟度



3. 建立 完善的保障制度

建立部署管理，配置管理，日志管理和监控告警管理等多种手段，实现对开源软件和开源架构的安全可控。



2. 使用 开源架构来支撑创新业务

通过积累最佳实践方法和经验，设计出符合业务需求和管理需求的开源架构，以满足创新业务的运行支撑要求。

4. 锻炼 具备相关技术的团队

建立开发和运维部门中的核心开源技术团队，实现开源技术领域的可持续性的发展。

充分利用自身条件，与外部开源社区建立良好沟通和互动渠道：“请进来”或“走出去”。



甜橙金融深度参与开源分布式数据技术



ShardingSphere

- 分布式数据中间件开源项目
- Apache 孵化器项目(2018/11/10)
- 9200+ Star
- 京东数科项目发起及维护
- 甜橙金融深度参与及贡献
- ShardingSphere PMC 成员

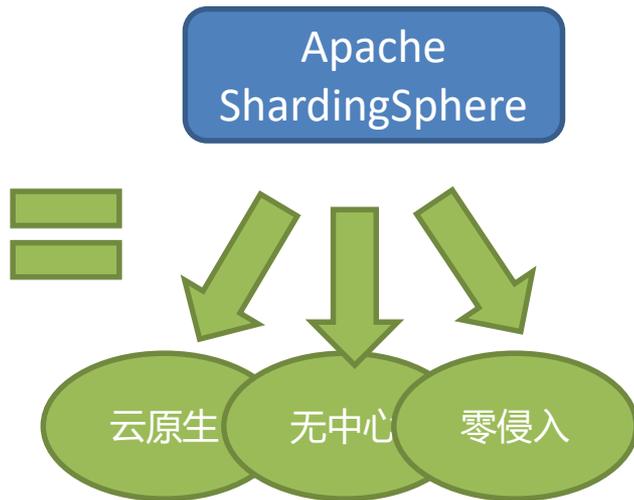
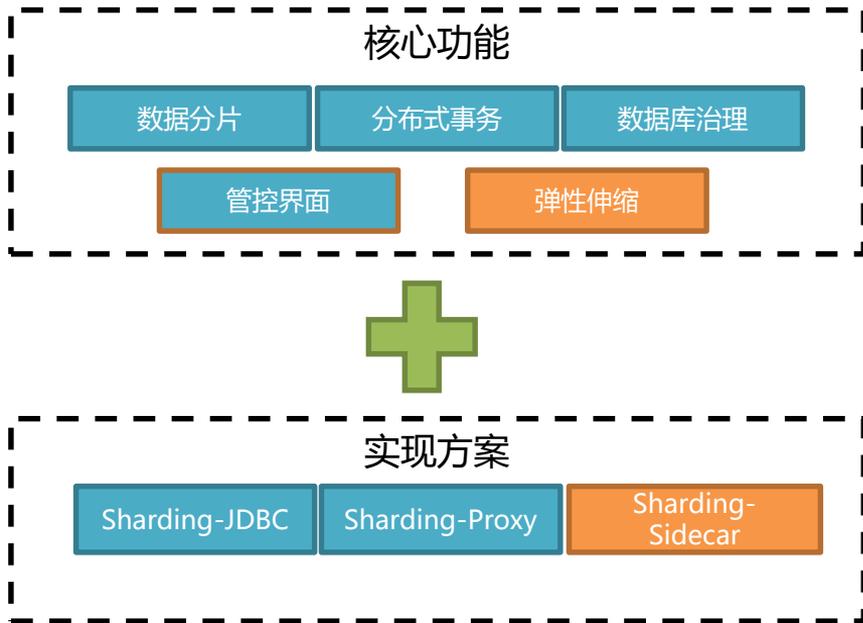


TiDB

- NewSQL 分布式数据库开源项目
- CNCF 孵化器项目(2019/5/21)
- 21000+ Star
- PingCAP 项目发起及维护
- 甜橙金融深度参与和投产实践
- 聚焦多中心多活及云原生领域能力

积极参与国内数据库及开源社区
智能运维, 云数据库管理, 数据分布式传输等领域长期研究
人工智能, 区块链, 安全, 大数据领域积极实践

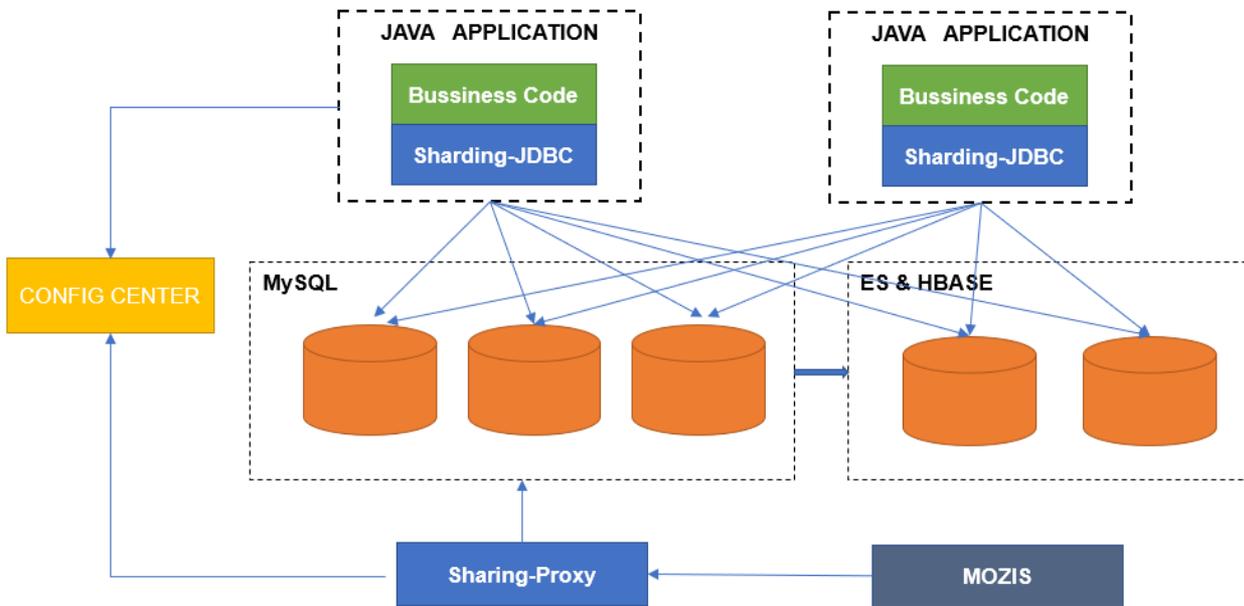
ShardingSphere 功能架构



ShardingSphere 的技术优势与特点



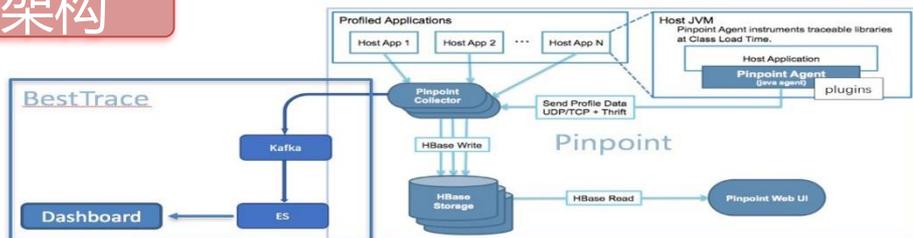
甜橙金融 ShardingSphere应用架构



- Java应用通过Sharding-JDBC接入
- 通过DTS将MySQL数据同步到ES & HBASE中，满足OLAP场景。
- MOZIS运维平台通过Sharding-Proxy接入，提供日常运维支持
- 利用配置中心对相关配置信息进行统一管理

甜橙金融 ShardingSphere应用实践 – 应用性能监控

平台架构



- 基于pinpoint进行的二次开发。
- 采用字节码增强技术
- 编写plugin完成ShardingSphere核心链路方法的监控

insert(String statement, Object parameter)	org.chengyu.sharding.mapper.OrderMapper.insert	15:52:04 193	5	1045	712	SqlSessionTemplate	MYBATIS
route(List parameters)		15:52:04 835	642	216	216	PreparedStatementRoutingEngine	SHARDING-SPHERE/ROUTE
SQL	insert into t_order (user_id, status) value						
setAutoCommit(boolean autoCommitFlag)	false	15:52:05 065	14	21	21	ConnectionImpl	MYSQL(demo_ds_1)
prepareStatement(String sql, int resultSetType, int resultSetConcurrency)		15:52:05 093	7	41	41	ConnectionImpl	MYSQL(demo_ds_1)
execute(SQLType sqlType, Collection baseStatementParameters)		15:52:05 171	37	55	13	ExecutorEngine	SHARDING-SPHERE/EXECUTE
SQL	insert into t_order_0 (user_id, status, orc						
SQL-BindValue	[[[147, SCUESS, 224195300430970800]]]						
db-instance	[ds_1]						
execute()		15:52:05 183	12	42	42	PreparedStatement	MYSQL(demo_ds_1)
SQL	insert into t_order_0 (user_id, status, orc						
SQL-BindValue	147, SCUESS, 224195300430970800						
commit()		15:52:05 240	1	28	28	ConnectionImpl	MYSQL(demo_ds_1)

甜橙金融 ShardingSphere应用实践 – 应用性能监控



Apache SkyWalking



OpenTracing



甜橙金融 ShardingSphere应用实践 – Proxy优化

Sharding-Proxy多逻辑数据源支持

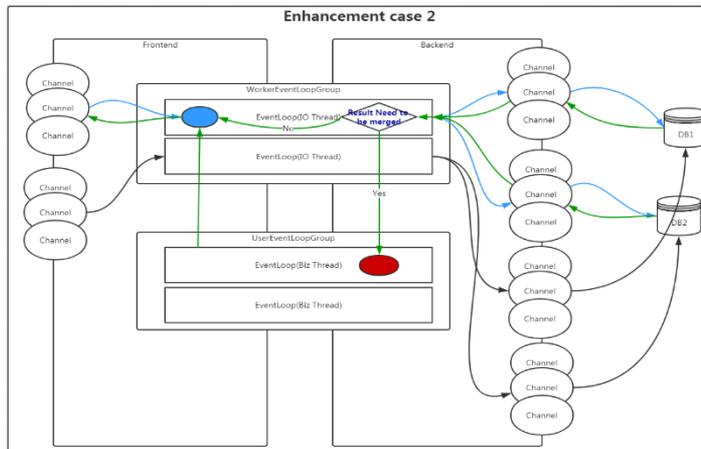
```
mysql> show databases;
+-----+
| Database |
+-----+
| sharding_db |
+-----+
1 row in set (0.02 sec)

mysql> use sharding_db;
Database changed
mysql> show tables;
+-----+
| Tables_in_sharding_db |
+-----+
| t_config |
| t_order |
| t_order_item |
| t_goods |
+-----+
4 rows in set (0.31 sec)
```

```
mysql> show databases;
+-----+
| Database |
+-----+
| master_slave_db |
| sharding_db |
| user |
| order |
+-----+
4 rows in set (0.02 sec)

mysql> use order;
Database changed
mysql> show tables;
+-----+
| Tables_in_order |
+-----+
| tbl_order |
| tbl_order_item |
+-----+
2 rows in set (0.38 sec)
```

Sharding-Proxy Backend NIO 优化



甜橙金融 ShardingSphere应用实践 – 可视化管控

Home / Data governance / Config manage

orchestration_ds admin English

schema authentication props

sharding_db master_slave_db

rule datasource rule datasource

+

Edit source here:

```
shardingTables:
- L_order_order_item
defaultDatabaseStrategy:
  inline:
    algorithmExpression: ds_${user_id % 2}
shardingColumns: user_id
tables:
  L_order:
    actualDataNodes: ds_${0..1}L_order_${0..1}
    keyGenerator:
      column: order_id
      type: SNOWFLAKE
    logicTable: L_order
    tableStrategy:
      inline:
        algorithmExpression: L_order_${order_id % 2}
    shardingColumns: order_id
  L_order_item:
    actualDataNodes: ds_${0..1}L_order_item_${0..1}
    keyGenerator:
```

Result (JS object dump):

```
{
  "shardingTables": [
    "L_order_order_item"
  ],
  "defaultDatabaseStrategy": {
    "inline": {
      "algorithmExpression": "ds_${user_id % 2}",
      "shardingColumn": "user_id"
    }
  },
  "tables": {
    "L_order": {
      "actualDataNodes": "ds_${0..1}L_order_${0..1}",
      "keyGenerator": {
        "column": "order_id",
        "type": "SNOWFLAKE"
      },
      "logicTable": "L_order",
      "tableStrategy": {
        "inline": {
          "algorithmExpression": "L_order_${order_id % 2}"
        }
      },
      "shardingColumns": "order_id"
    },
    "L_order_item": {
      "actualDataNodes": "ds_${0..1}L_order_item_${0..1}"
    }
  }
}
```

Cancel Submit

Copyright © 2016 - 2018 The Apache Software Foundation, Licensed under the Apache License, Version 2.0.

- 数据分片配置管理
- 运行实例编排管理
- 自动弹性扩缩



大并发强一致分布式数据处理场景的思考

业务场景需求	分库分表模式	NewSQL 模式
业务维度的选择和切分	适合业务中数据模型容易按照单一维度切分和查询的场景。 (比如时间+Hash)	不限维度的自由查询和数据建模（与单机数据库相同）
跨库跨节点复杂计算	数据切分后无跨节点跨库聚合及复杂关联查询的业务很友好和便利	不限维度的进行跨节点跨库的 JOIN 关联及聚合计算。(包含计算下推能力)
分布式强一致事务	数据切分后无跨节点跨库聚合及复杂关联查询的业务很友好和便利	适合联机交易业务有强一致分布式事务刚需的场景
大表占比	数据库内大表(千万到亿以上行记录规模) 数量占比较小的业务场景比较便捷	数据库内大表行记录规模和数据规模没有什么限制, 适合数据快速增长的业务
混合负载 (HTAP)	适合业务单一负载, 尤其是计算复杂度低的大并发联机交易	适合同时存在有 OLTP 联机交易和 在线实时分析的业务场景
弹性在线热扩容	对于业务中有足够变更窗口, 不需要在线弹性热扩容的场景比较轻量便利。	适合需要弹性在线热扩容, 提高计算吞吐和存储容量的场景。
计算和存储自动平衡	对于扩容后对数据没有计算重平衡及存储重平衡要求的业务比较方便。	适合业务在扩容后 需要做自动的计算重平衡和存储重平衡的场景



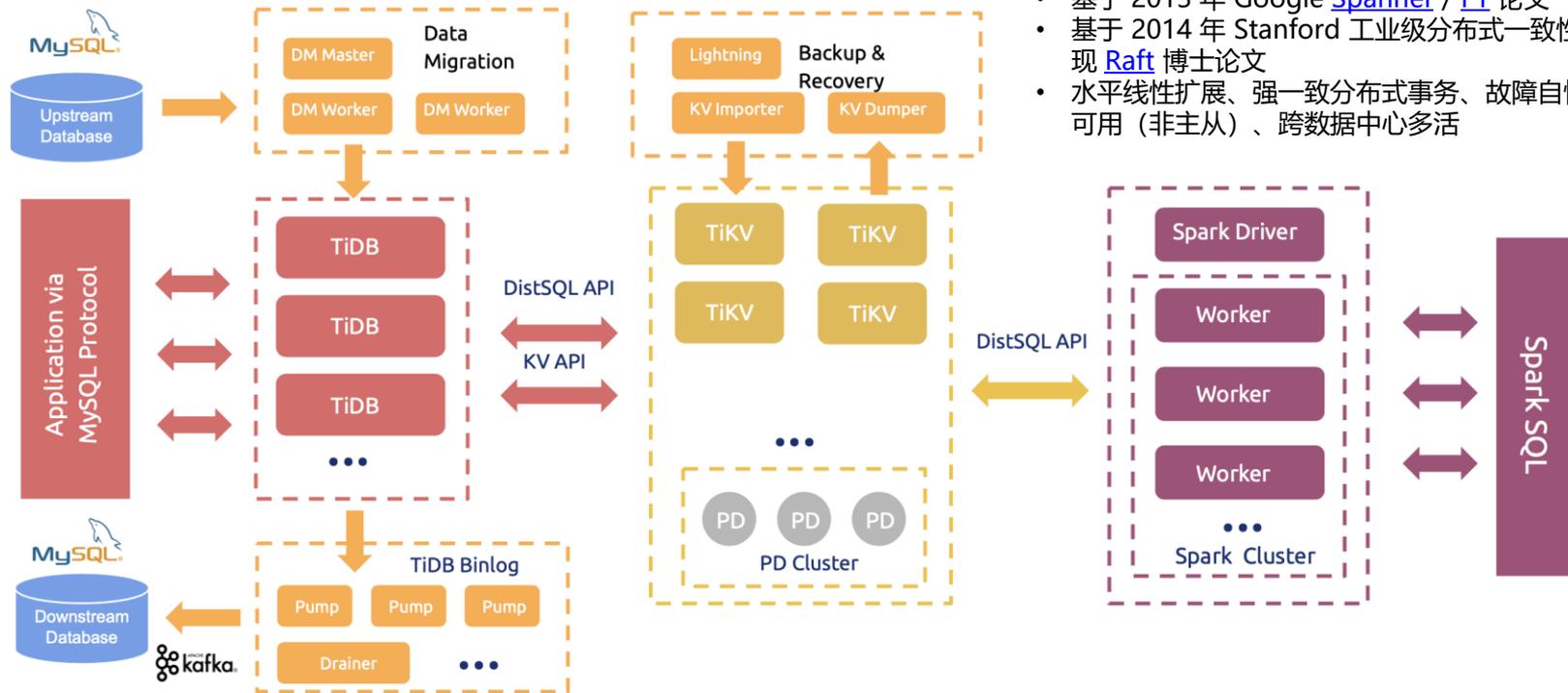
甜橙金融 数据处理技术方案选择思考

技术方向/评估维度	容量阈	性能阈	大表数量	分片规则	增长趋势	HTAP	拓扑标准
RDS 方案 (单实例)	<3T	<20000 QPS	N/A	N/A	N/A	N/A	一主两从标准架构
分布式数据中间件方案	>3T & <10T	>20000 QPS	<3	仅限日期, Hash	固定分片	不支持	两主两从标准架构
NewSQL 方案	>3T	>20000 QPS	>=3	N/A	动态扩容	支持	动态扩展调整

注：以上为甜橙金融根据自身业务特点，以及对分布式数据处理技术的实践总结的选择路径，并在未来随着技术的进行灵活调整。其他用户也需要通过自己的业务特点来有针对性的设计对应的方案路径。



典型 TiDB 应用架构



- 基于 2013 年 Google [Spanner](#) / [F1](#) 论文
- 基于 2014 年 Stanford 工业级分布式一致性协议实现 [Raft](#) 博士论文
- 水平线性扩展、强一致分布式事务、故障自恢复的高可用（非主从）、跨数据中心多活

甜橙金融 TiDB 研究与实践 – 联机交易负载场景

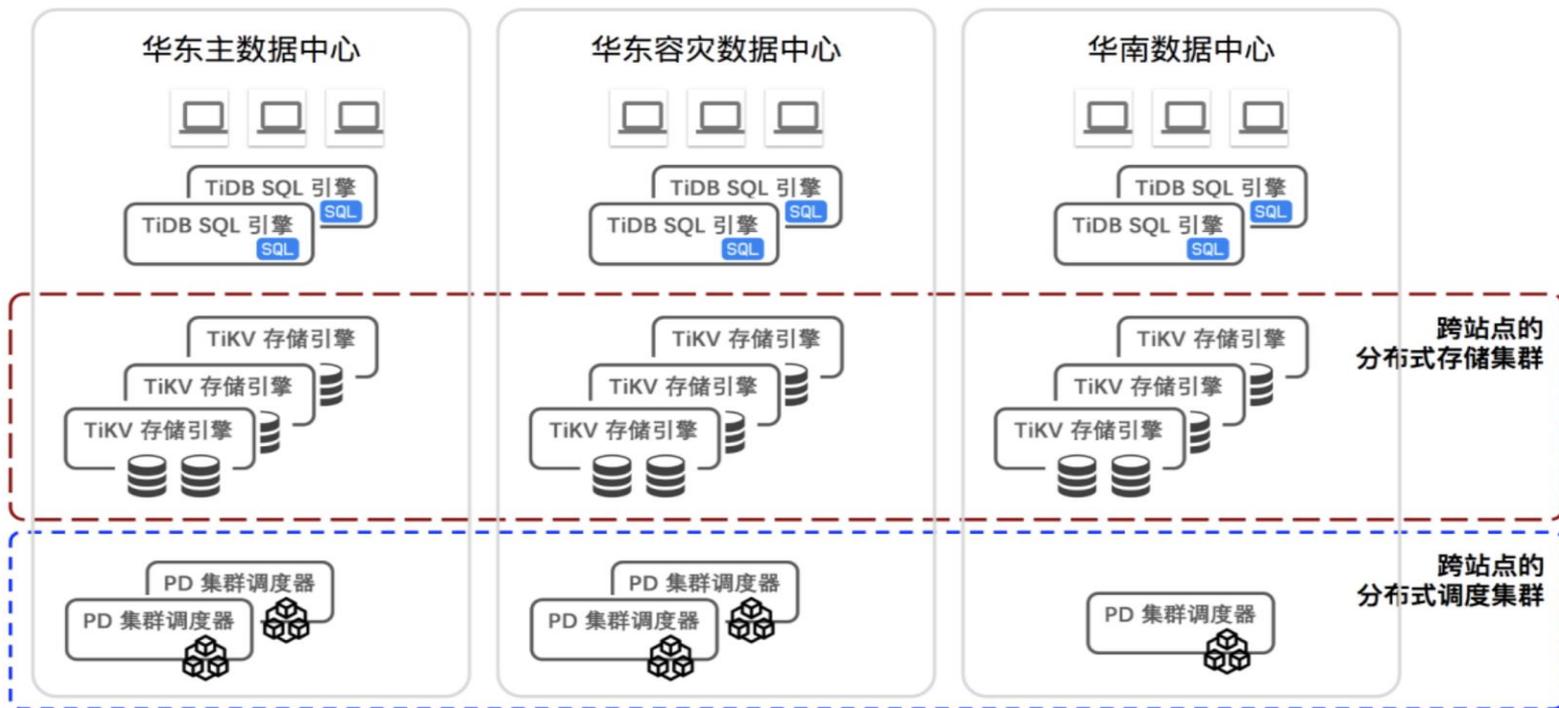
- 财务稽核, 对账, 反洗钱, 账单, 营销等重要业务
 - Multi – Raft 协议 保证节点间副本多数派强一致
 - 数据透明打散, 库内业务大表不用人工切分, 表记录到 亿/十亿及以上规模上性能无影响
 - 去中心化的 2PC (两阶段递交) 确保跨 Region 事务的强一致, 达到 ACID 标准
 - 业务不需要做分库分表改造, CRUD 行为与单机保持一致
 - 表上任意列上都可以做分布式查询计算与更新操作
 - 计算支持过滤, 聚合等下推到 数据节点分布式计算
 - MySQL JDBC 和 Mybatis ORM 直接使用
 - 直接在线扩缩容, 业务代码和数据模型不需要做调整
 - 计算层和存储层均可通过节点增加线性扩展数据库的计算吞吐和容量

投产 OLTP 交易集群典型配置:

集群角色	配置	物理节点	并行实例数量	部署拓扑
计算节点	32 Core,256GB Mem	3	3	每台启动 1个 TiDB 进程, 可在线 增加
存储节点	48 Core, 256GB Mem ,3TB SSD *3	4	12	每台启动 3 个 TiKV 进程, 可在线增加



甜橙金融 TiDB 研究与实践 – 跨中心多中心多活容灾场景



甜橙金融 TiDB 研究与实践 – 跨中心多中心多活容灾场景

dc	zone	rack	host	TiKV	
dc1	z1	r1	h208	kv208-1 kv208-2	region1
			h209	kv209-1 kv209-2	
	z2	r2	h210	kv210-1 kv210-2	region1
			h211	kv211-1 kv211-2	
dc2	z3	r3	h413	kv413-1 kv413-2	region1
			h414	kv414-1 kv414-2	
	z4	r4	h415	kv415-1 kv415-2	region1
			h416	kv416-1 kv416-2	
dc3	z5	r5	h252	kv252-1 kv252-2	
			h253	kv253-1 kv253-2	region1

- 标签系统 (Label)：TiDB 内置有一套标签系统，可以为一套集群的不同节点，按照 Site (数据中心)，Rack (同中心内不同机架)，Host (不同物理机节点) 来对应设置标签信息，从而实现将集群的跨数据中心物理拓扑和集群调度连接起来。
- 网络状态：华东两个数据中心 (主数据中心及华东容灾中心) 与华南数据中心的网络通讯条件理想的情况下，PD 可以将数据分布中的 Leader Region 动态分布到三个中心的节点上。每个中心都可以以统一视角访问和操作数据库，TiDB 引擎上的 Multi-Raft 复制机制会在不同中心的数据区域进行强一致性复制 (Raft Log based)。网络通讯条件不够理想的情况下，PD 可以将数据分布中的 Leader Region 动态分布到华东的两个数据中心中，而在异地保留 Follower Region 作为高可用保护。
- 底层数据管理：三个中心内的数据管理单位 Region 都是可以在线的做动态的分布改变。
- 在线扩容机制：动态水平扩展。扩展后的集群中的数据，会根据 PD 调度的管理，自动在后台完成数据重平衡工作，对业务连续性不产生影响。

[tikv_servers]

```
kv208-1 ansible_host=192.168.2.8 deploy_dir=/tikv1 tikv_port=20171 tikv_status_port=20181 labels="dc=dc1,zone=z1,rack=r1,host=h208"  
kv208-2 ansible_host=192.168.2.8 deploy_dir=/tikv2 tikv_port=20172 tikv_status_port=20182 labels="dc=dc1,zone=z1,rack=r1,host=h208"
```

```
kv209-1 ansible_host=192.168.2.9 deploy_dir=/tikv1 tikv_port=20171 tikv_status_port=20181 labels="dc=dc1,zone=z1,rack=r1,host=h209"  
kv209-2 ansible_host=192.168.2.9 deploy_dir=/tikv2 tikv_port=20172 tikv_status_port=20182 labels="dc=dc1,zone=z1,rack=r1,host=h209"
```

```
kv210-1 ansible_host=192.168.2.10 deploy_dir=/tikv1 tikv_port=20171 tikv_status_port=20181 labels="dc=dc1,zone=z2,rack=r2,host=h210"  
kv210-2 ansible_host=192.168.2.10 deploy_dir=/tikv2 tikv_port=20172 tikv_status_port=20182 labels="dc=dc1,zone=z2,rack=r2,host=h210"
```

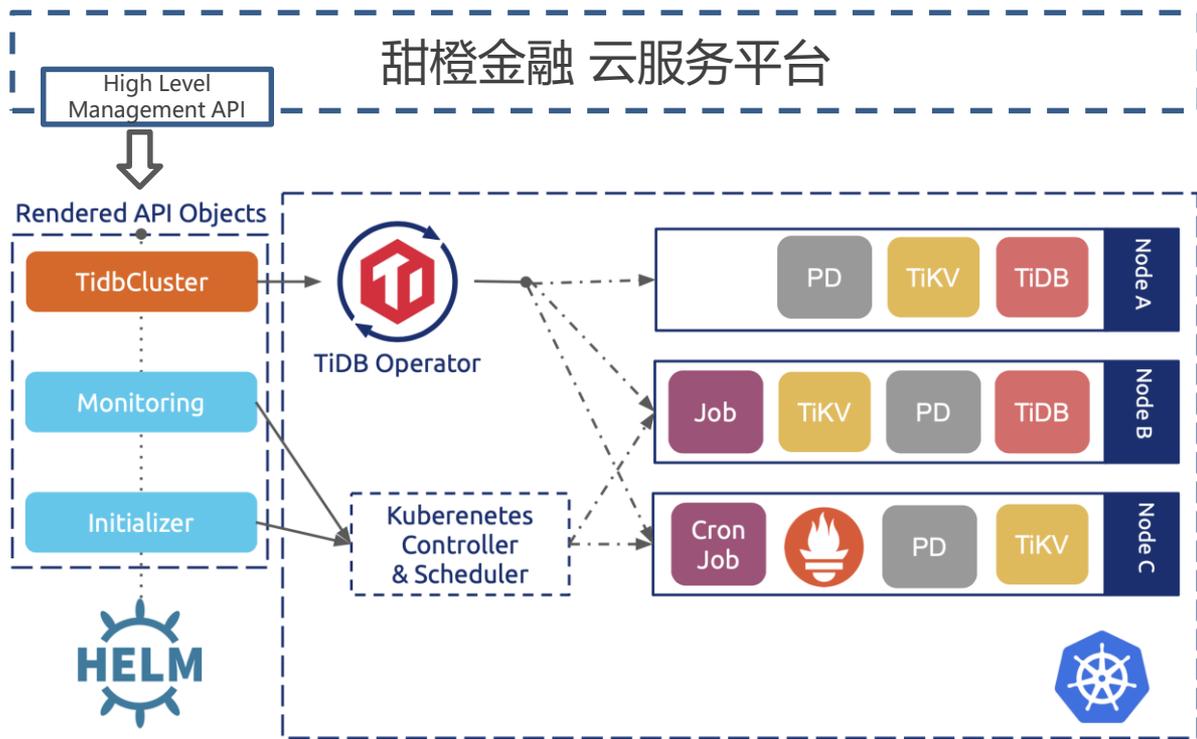
```
kv211-1 ansible_host=192.168.2.11 deploy_dir=/tikv1 tikv_port=20171 tikv_status_port=20181 labels="dc=dc1,zone=z2,rack=r2,host=h211"  
kv211-2 ansible_host=192.168.2.11 deploy_dir=/tikv2 tikv_port=20172 tikv_status_port=20182 labels="dc=dc1,zone=z2,rack=r2,host=h211"
```

根据：

Site (dc=) 映射物理中心
Zone (zone=) 映射逻辑区
Rack (rack=) 映射物理机架
Host (host=) 映射物理主机



甜橙金融 TiDB 研究与实践 – 云原生分布式数据库场景



服务器节点 (Node A Node N)

- 数据库组件实例调度在不同硬件节点
- 根据负载和调度策略动态分布
- 可定制调度策略, 改变默认的行为
- 集群实例交错在不同硬件节点
- 容器集群扩缩配合 TiDB 集群扩缩
- API 封装集成到上游统一运维管理平台
- 监控告警上报汇聚到统一监控/告警平台
- 高可用管理和自动 Failover 通过 AZ(可用区)分布到多个可用区域



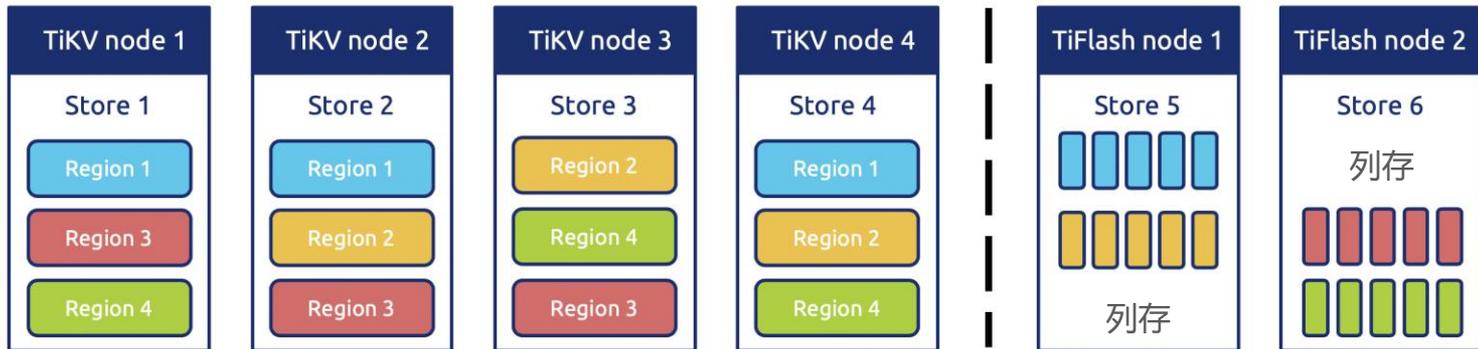
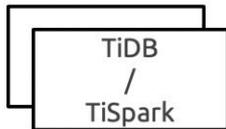
甜橙金融 TiDB 研究与实践 – HTAP 混合负载场景

- 通过 Raft Learner 独立同步一套列存
- Raft Learner 提供极低消耗的副本同步
- Raft Learner 读取协议配合 MVCC 提供**强一致**的读取
- 通过 Label 进行物理隔离，AP / TP 作业相互无影响

```
SELECT AVG(s.price) FROM  
prod p, sales s  
WHERE p.pid = s.pid  
AND p.batch_id = 'B1328';
```

IndexScan prod
(pid, batch_id = 'B1328')

TableScan sales
(price,pid)





全球敏捷运维峰会

THANK YOU!

