

# 基于混搭存储引擎的融合型 分布式数据库架构

——服务型分布式计算和混搭型分布式数  
据存储助力大数据时代的数据宝藏挖掘

## DTCC

### 2015中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2015

大数据技术探索和价值发现



1

分布式的3W问题

2

服务型分布式计算

3

基于混搭存储引擎的融合型分布式数据库

4

大数据与分布式计算/分布式数据库



# 经典商业应用场景的困境（一）

## OLTP (1+1=?)



业务、数据大集中  
简单任务高并发  
响应时间敏感  
永远的痛：关系型数据库

## 经典对策

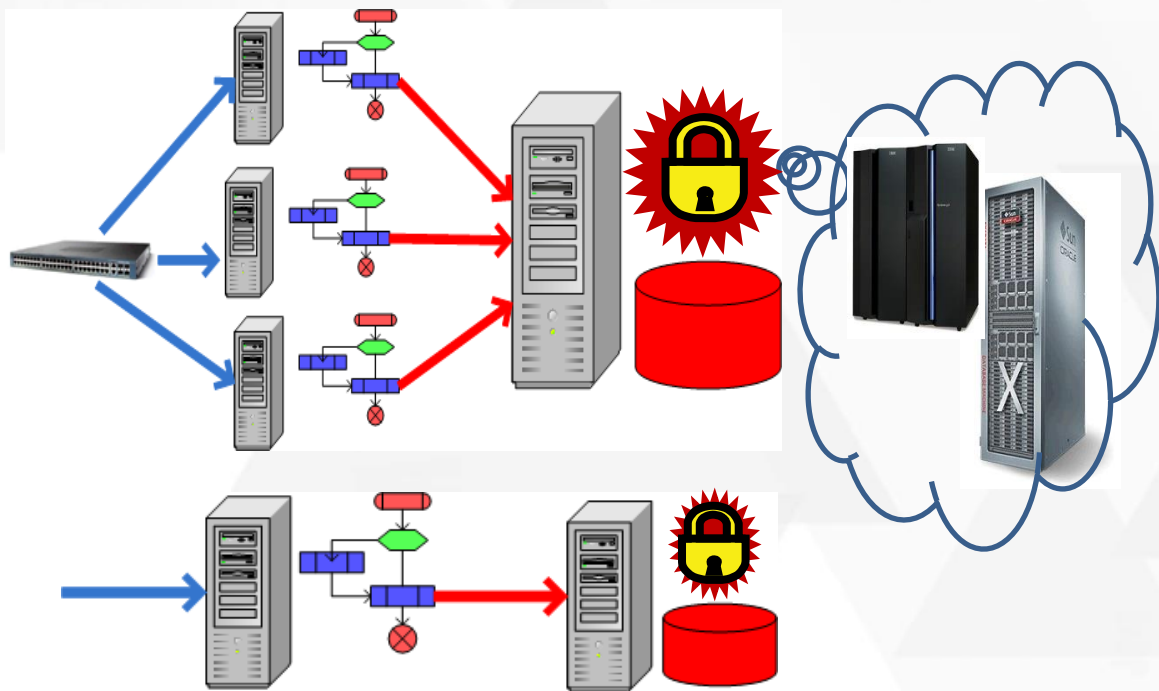


小型机  
数据库集群  
业务拆分

## 迫切需求



破除单点故障  
提升性能  
提升业务和数据规模



# 经典商业应用场景的困境（二）

## 会话型应用



灵活的通讯模式  
会话数据需长期保存  
灵活的数据形式  
数据是瓶颈，复杂业务功能瓶颈

## 经典对策

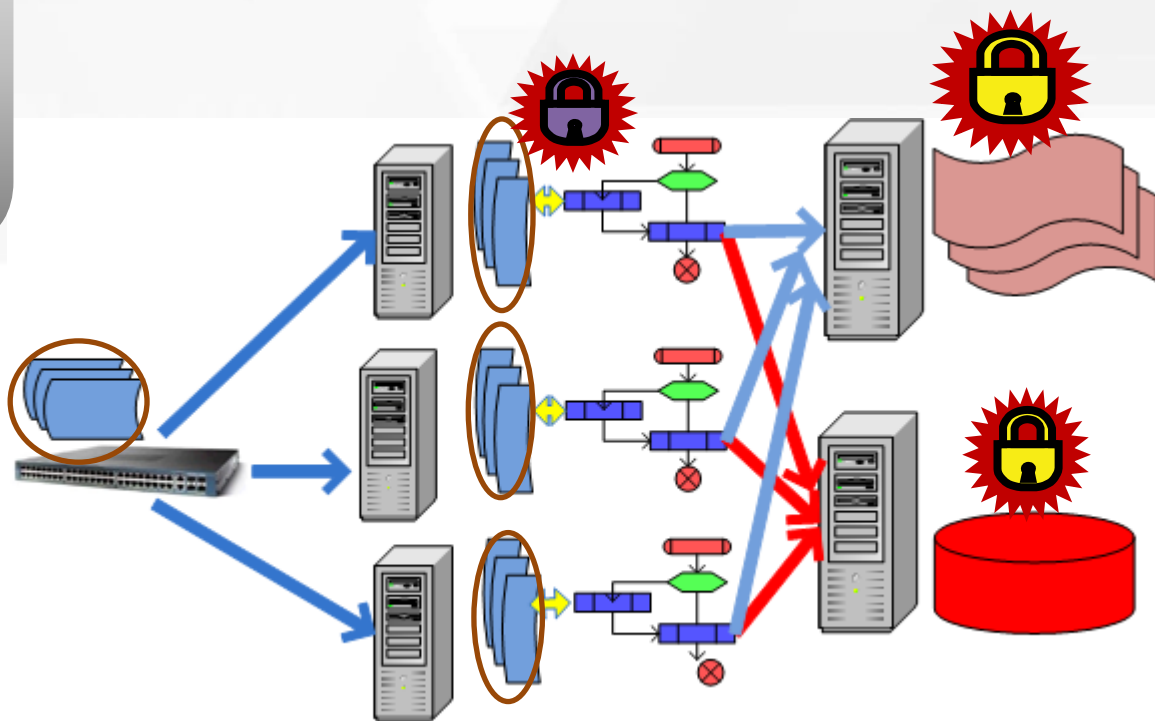


多机处理业务  
CDN  
多样性数据存储

## 迫切需求




丰富的会话语义  
丰富的数据存储形式  
弹性的业务、数据规模



# 经典商业应用场景的困境（三）


## 并行/分布式计算、大数据

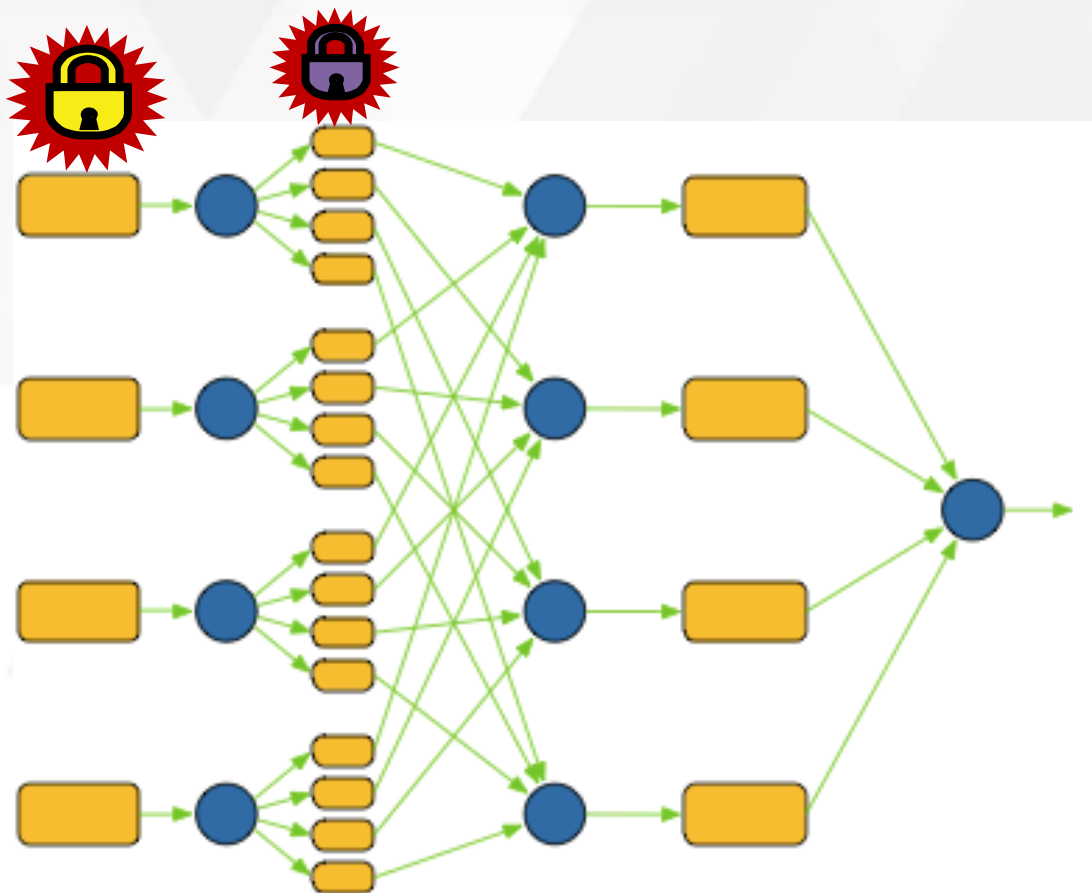
 繁重单任务  
叠加高并发雪上加霜  
切分功能瓶颈、数据处理瓶颈

## 经典对策

 多机计算  
多样性数据的多机存储  
Hadoop、Spark、Storm

## 迫切需求

 丰富的分布式语义  
大容量并行执行  
混杂海量数据的丰富检索分析  
语义、性能、伸缩性



# 困境中的答案





# 再说什么是分布式

## 分布式不是一种新技术

应用框架，设计模式

衍生的支撑技术

路由、负载均衡、任务调度、并行计算、  
资源竞争、线程间/进程间/网络通讯，  
衍生的设计需求

RASP

## 安排尽可能多的人共同执行一个任务

路由：谁能干

负载均衡：谁比较闲

## 分布式的直观分类

个体任务的步骤并行  
流水线提升吞吐量  
两者并用



## 如何分配工作？

领导分配

人人自主分配

只要有备份，就有数据不一致

## 恐怖的“时间窗口”

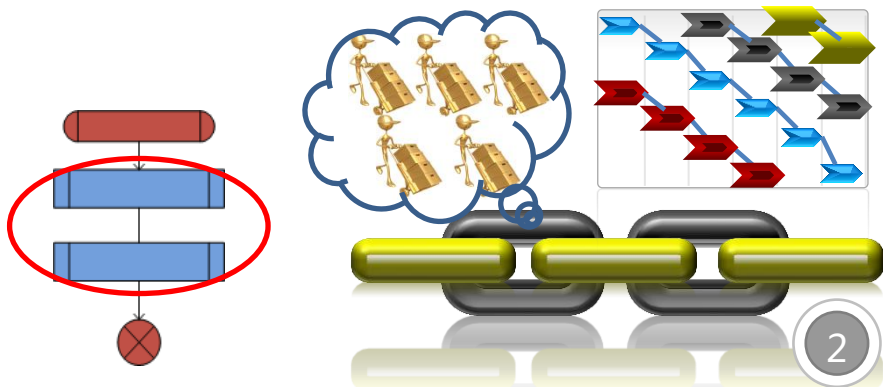
“时间窗口”发生的各种错误的自动  
识别、修复和遗留问题的清理

# 理想的分布式应用架构长什么样？

用工作流的形式来将步骤解耦并分布式

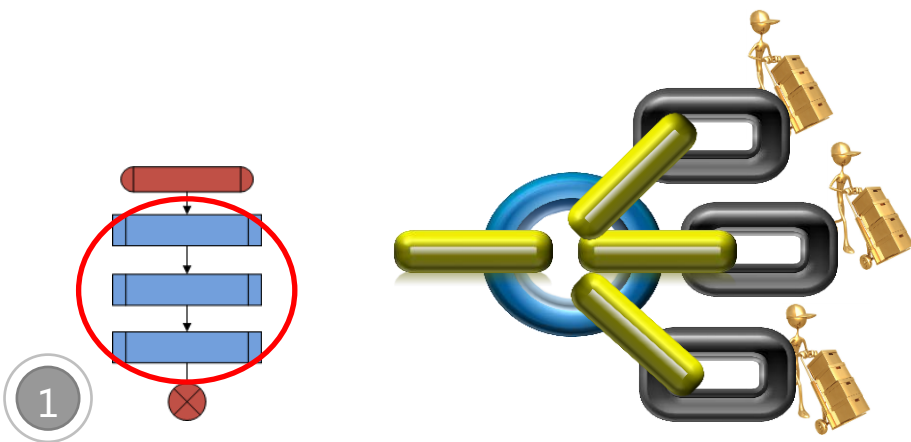
多任务串行→多任务并行

提高请求/任务吞吐量、保持响应时间



单任务串行→单任务并行

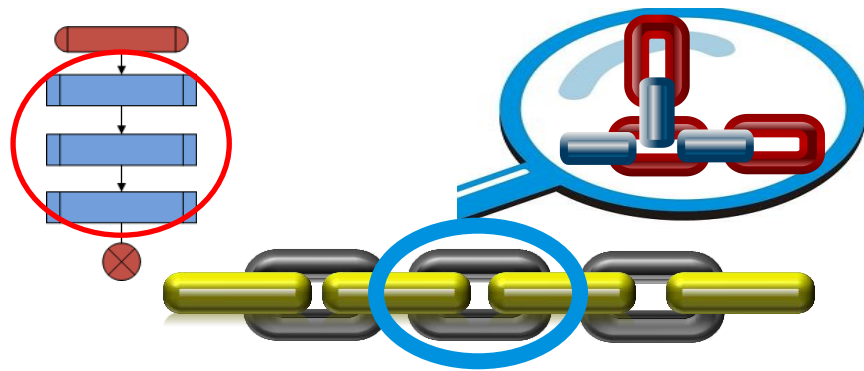
降低响应时间



分布式的粒度：子系统，模块（函数、对象）

性能与管理成本的权衡决定粒度

应用把控



理想的分布式应用设计方法论

应用角度主导的“分而治之”



# 理想的分布式计算框架/平台该做什么？

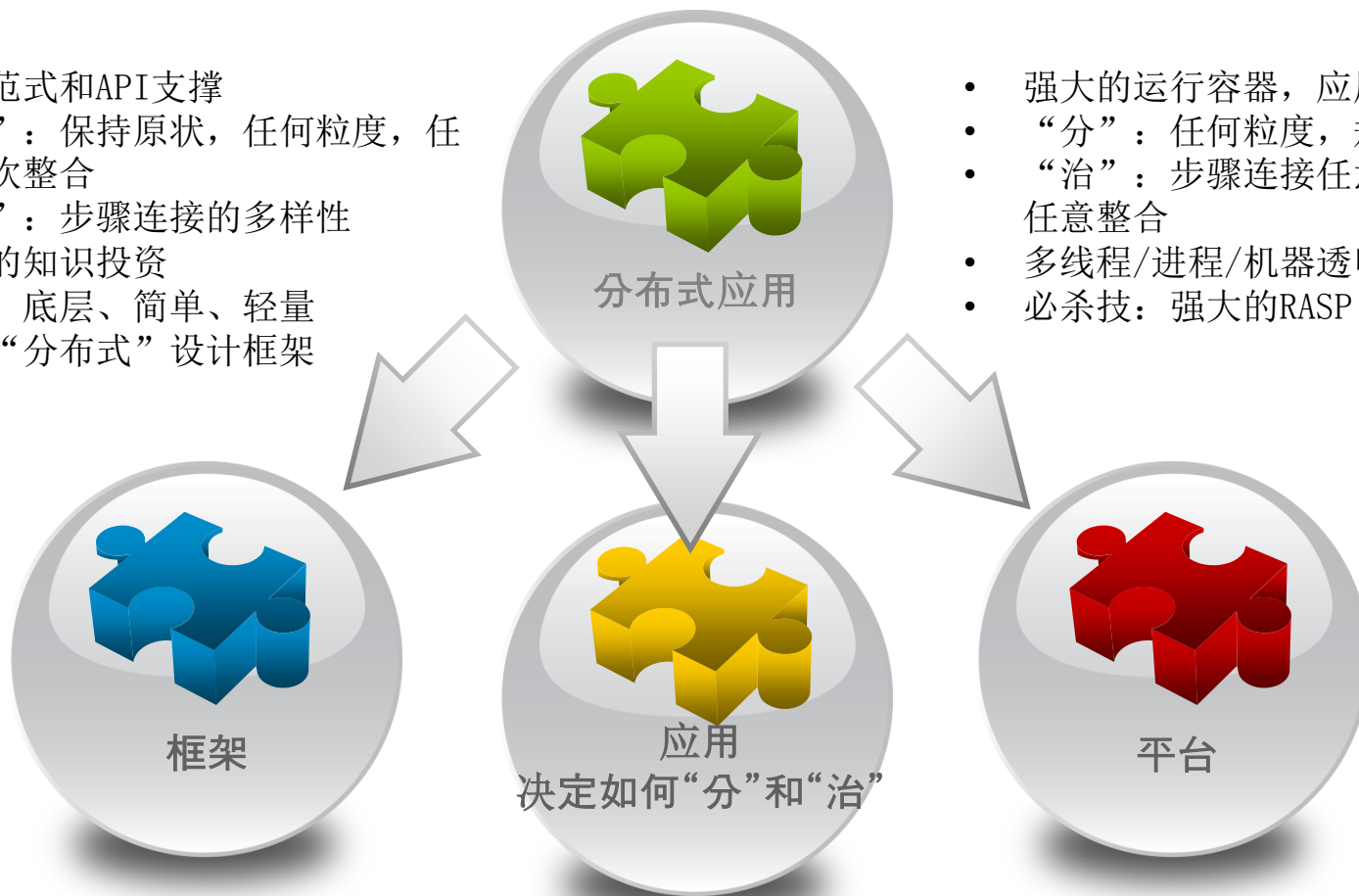
“最高深的技术是那些令人无法察觉的技术，这些技术不停地把他们自己编织进日常生活，直到你无从发现为止” ——Mark Weiser

## 开发态

- 编程范式和API支撑
- “分”：保持原状，任何粒度，任何层次整合
- “治”：步骤连接的多样性
- 最少的知识投资
- 通用、底层、简单、轻量
- 杜绝“分布式”设计框架

## 运行时

- 强大的运行容器，应用透明
- “分”：任何粒度，规模无限
- “治”：步骤连接任意顺序、任意整合
- 多线程/进程/机器透明并发
- 必杀技：强大的RASP



1

分布式的3W问题

2

服务型分布式计算

3

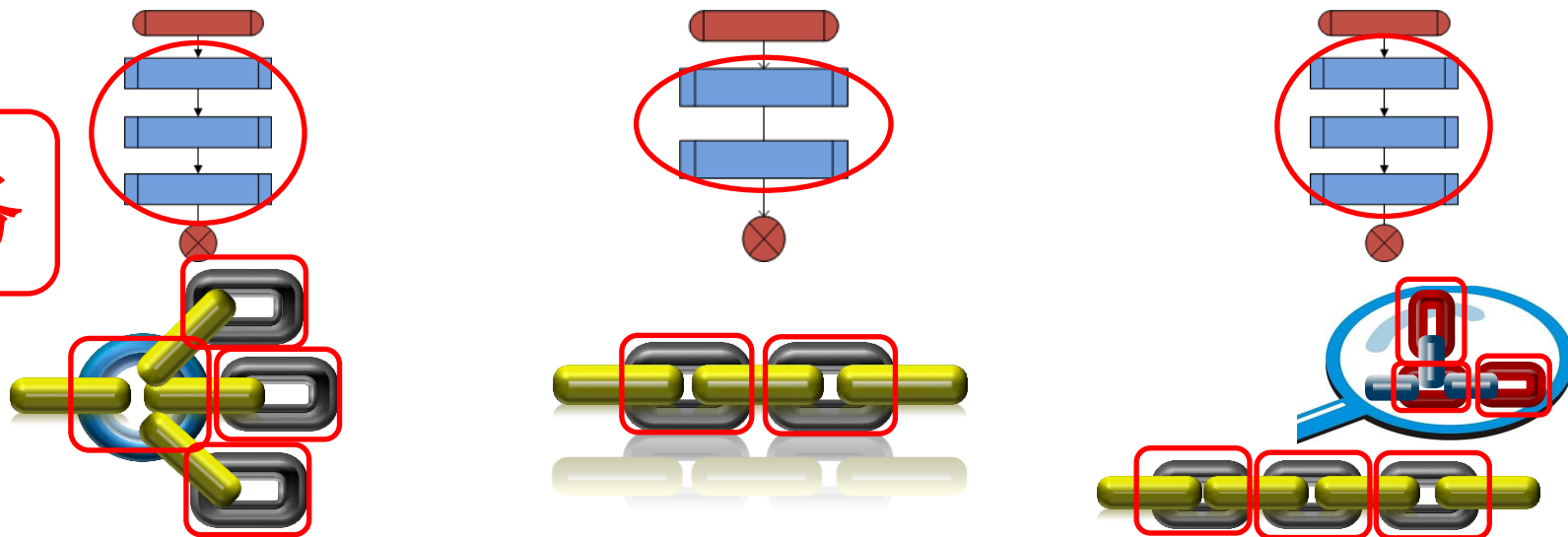
基于混搭存储引擎的融合型分布式数据库

4

大数据与分布式计算/分布式数据库

# 服务型分布式计算——分布式遇上SOA

服务



自上而下、自内而外的全SOA

## 服务化的开发方式

分布式的步骤→服务  
任何粒度的封装  
服务组装便捷  
对内对外服务统一

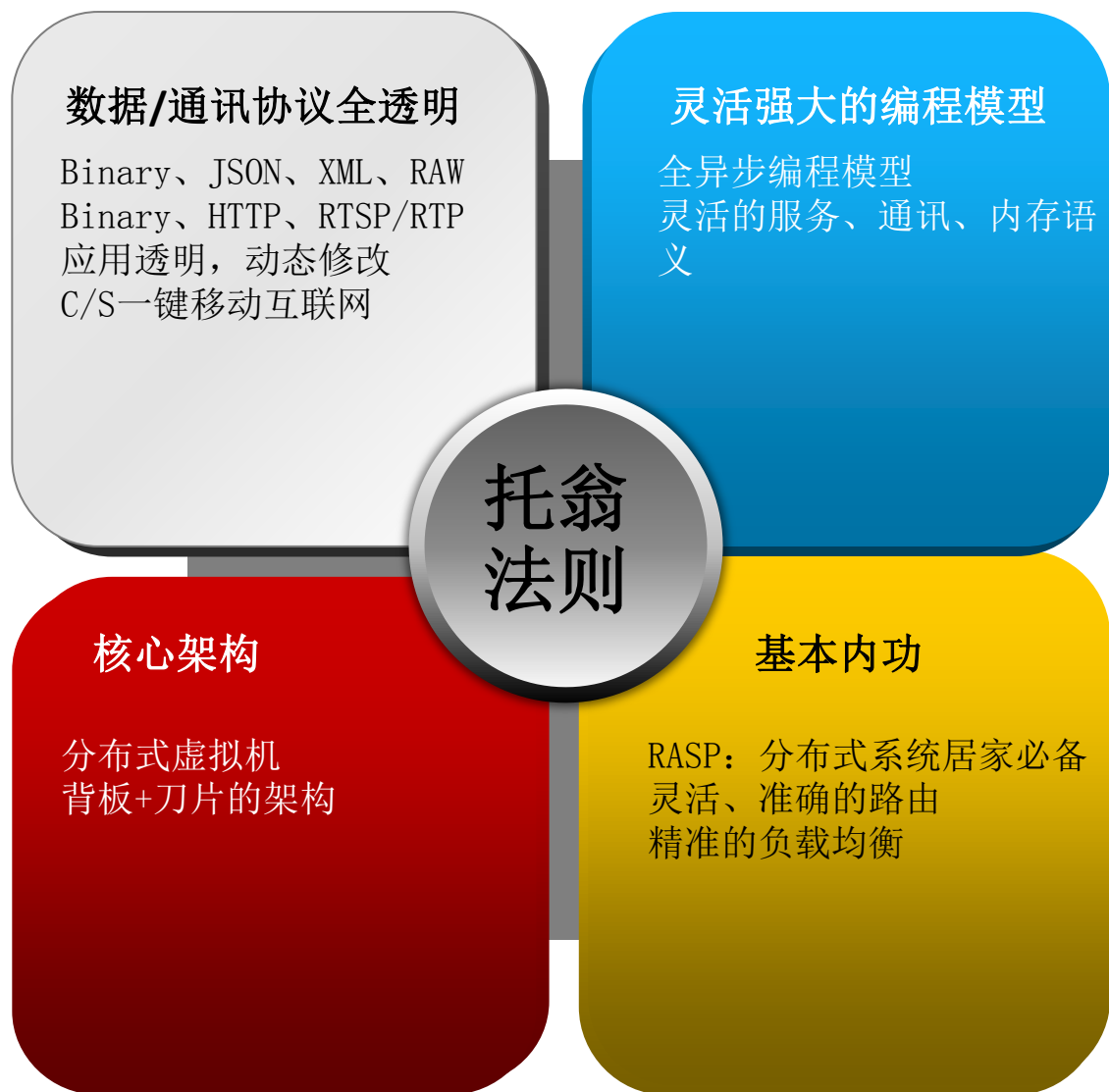
## 分布式从未如此简单

不改变业务流程和编程模型  
单机单用户思路设计分布式应用

## 分布式执行策略

服务虚拟化，计算资源虚拟化

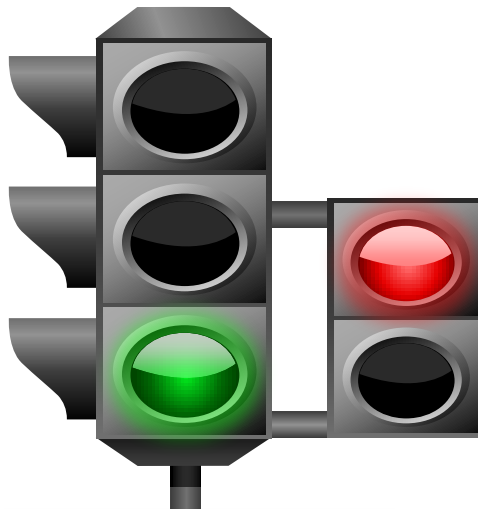
# 服务型分布式计算的核心功能



# 说说Map-Reduce

不是分布式计算、大数据领域的万金油，也不是阿司匹林！

伟大的创新  
复杂任务并行的好方案  
通用、成熟、“廉价”的  
大数据方案  
唾手可得的“免费”方案



Map-Reduce为何物

不解决基础问题，非通用方案  
应用于各种场景是错误  
低门槛带来的各种坑  
生态圈整合成本高  
开源的不断重构和发明

Google发明、Hadoop落地、Storm/Spark升华  
并行编程框架，写分布式应用的“银弹”  
合适的时间出现在合适的地方  
分布式计算和大数据的唯一方法论？

Hadoop、Spark、Storm：Map-Reduce的  
重要代言人  
分布式计算框架/平台



# 服务型分布式计算 VS Map-Reduce

## 服务型分布式计算

顺向思维

应用角度设计分布式

很少甚至不改造应用实现分布式

业务流程主导的个性化分布式模型

分布式策略通过配置而非编程

## Map-Reduce

逆向思维

所有应用需要重构

本末倒置

简单粗暴的“分而治之”

执行的架构在设计时决定



# 服务型分布式计算 vs 请求级分布式架构

## 服务型分布式计算

应用角度设计分布式  
分布式的粒度是步骤，最大程度分布式  
全异步、流水线，计算资源用到极致  
各种数据生命周期和数据共享  
请求间各种通讯  
有状态的对话

## 请求级分布式架构

针对OLTP和会话应用的经典分布式架构  
分布式的粒度是请求  
单请求成本高  
粒度太大，请求隔离差，并发弱  
数据共享弱  
请求间不能高效率通讯  
不支持对话

1

分布式的3W问题

2

服务型分布式计算

3

基于混搭存储引擎的融合型分布式数据库

4

大数据与分布式计算/分布式数据库

# 分布式存储是一种分布式计算



## DaaS

同样的方法论设计存储  
服务就是数据和数据访问



## 个性化分布式应用

先解决数据的分布 (DB sharding,  
分布式文件系统)  
数据依赖路由

分布式计算

分布式存储

分布式计算

分布式存储

分布式计算

# （分布式）数据库 VS （分布式）存储

数字化 VS 数据化



狭义数据库 VS 广义数据库



## 数据 VS 数据库 VS 存储

数据是货物

数据库将货物放进仓库有机存储管理。数据库是存储和管理数据的有效形式和方法论。

分布式数据库是分布式存储方法论和支撑技术的体现。

分布式数据库应该汲取各种存储技术的精华，在不同场景选择最合适的存储技术，而不是一味破坏式创新。

# 数据库的老问题

永远的瓶颈——数据库的容量和处理能力

数据库是永恒的焦点

I/O导致低速，竞争导致复杂  
持续的瓶颈，持续的焦点

数据库调优

应用的合理设计

合理的索引设计

存储资源换计算和I/O资源

SQL优化

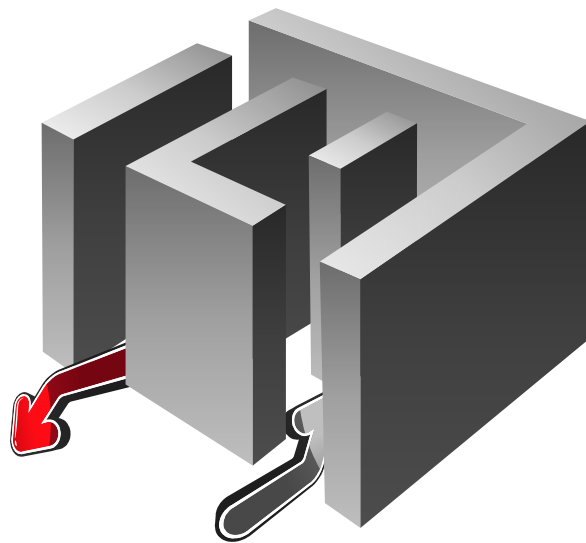
内存数据库、缓存

冷热数据分离

读写分离

数据的垂直切分

数据的水平切分（**唯一能够彻底解决数据容量、  
吞吐量、延时的全面可伸缩**）



数据集中的经典应用架构配合低速系统

永恒优化命题  
大数据时代更加严重

# 传统数据库集群方案





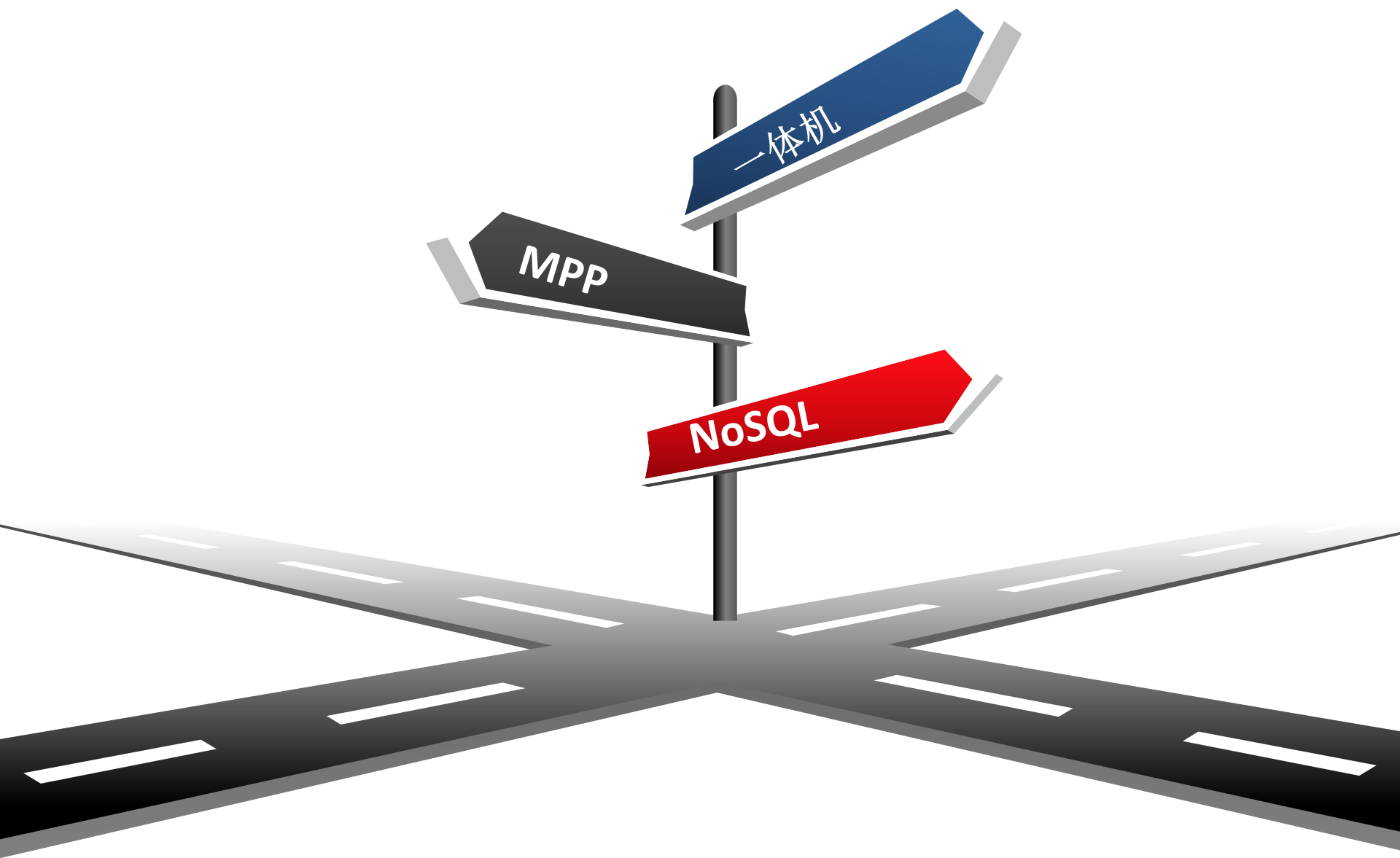
# 传统数据库面临的新烦恼

数据的维度和容量极大扩展  
数据维度频繁变化

不支持新的数据类型  
数据容量有限  
数据维度有限  
维度修改成本高企



# 常规分布式数据库方案



# 解决数据库困境的通用方案



解决老问题和新烦恼

非定制的通用解决方案

深度定制在特定场景下性能更好  
通用和专用的权衡妥协  
这个时代需要通用方案  
产品思维而非解决问题思维

轻量的可轻松定制、二次开发、优化

通用方案需要具备定制的机制  
结构化数据、非结构化数据、混合型数据  
OLTP、OLAP、混合型应用

# 基于混搭存储引擎的融合型分布式数据库

## 传统关系数据库的继承借鉴

不应完全被否定  
有限范围内的强大单机处理能力  
灵活的SQL语义

## 核心指导思想

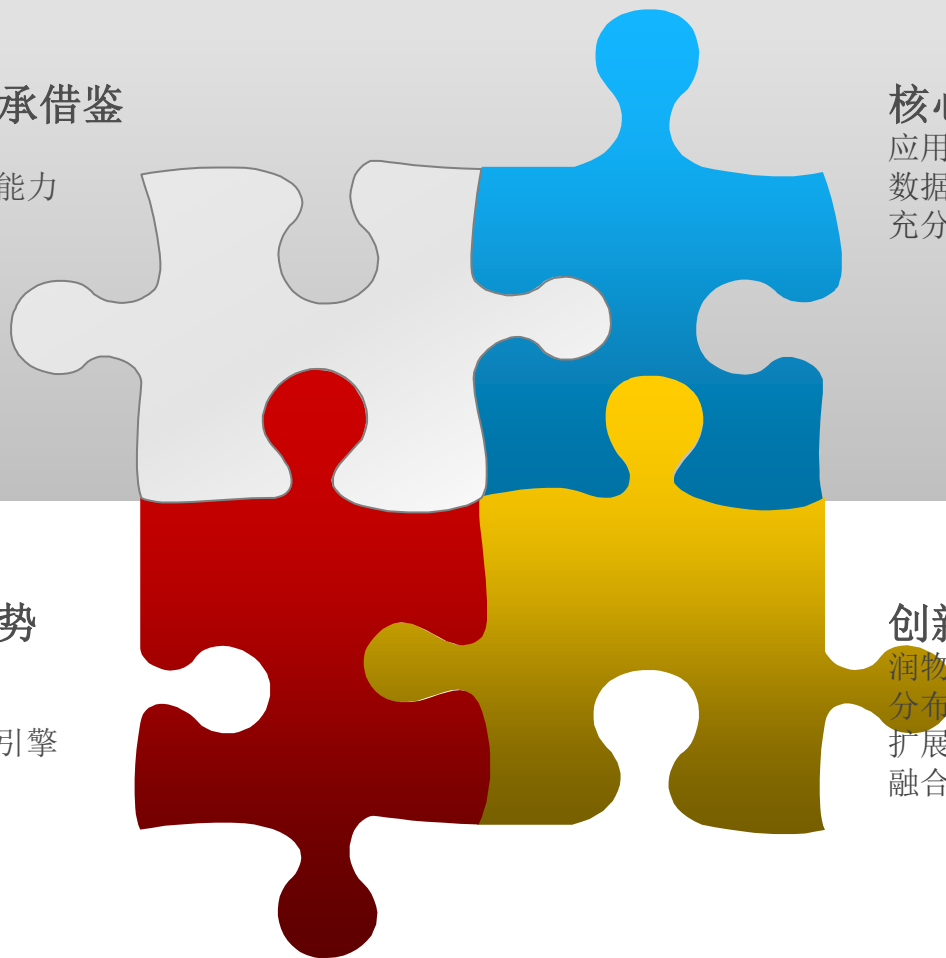
应用（数据库）角度设计分布式  
数据的分布存储（sharding）  
充分利用水平伸缩的存储和计算资源

## 融合多存储引擎的优势

将任何存储引擎分布式  
不是迁就，而是保护  
不同场景选择最合适的存储引擎  
整合多种存储引擎

## 创新

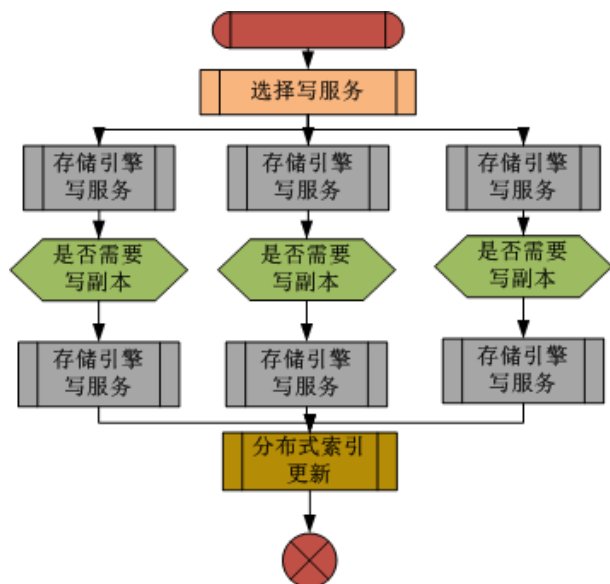
润物细无声的分布式数据库  
分布式环境下支持全SQL语义  
扩展的SQL语义  
融合混搭存储引擎的Big Table



# 融合型分布式数据库的定位



# 融合型分布式数据库的分布式架构



## 服务型分布式计算的一个实例

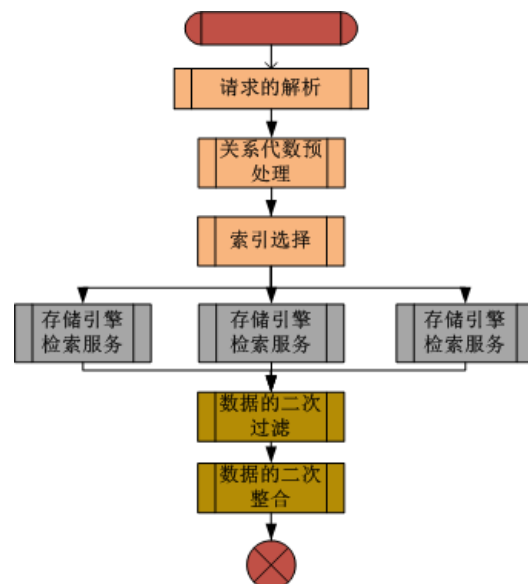
DBaaS

按照单线程数据库设计

主业务逻辑(写/读)封装成服务

多存储引擎串行读/写→异步并行调用服务

串行服务调用→异步调用流水线



## 特殊的分布式计算

数据依赖路由

根据数据分布策略选存储引擎写服务

根据分布式索引选存储引擎读服务



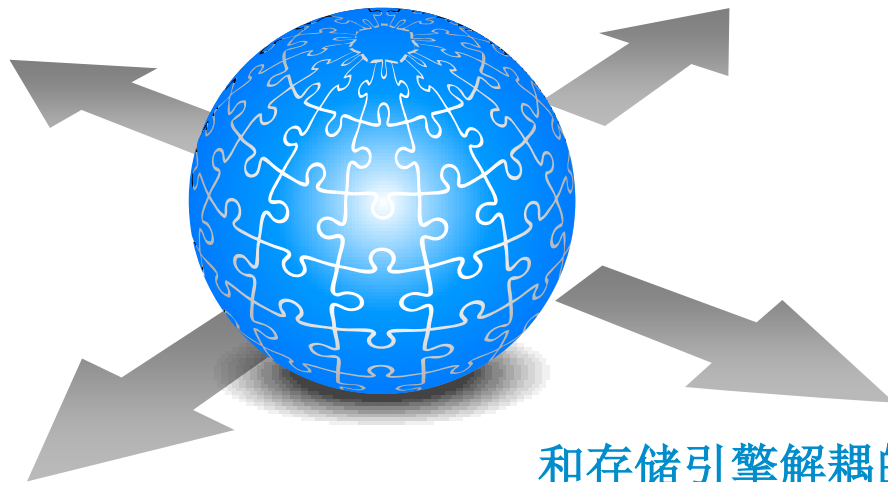
# 和传统数据库的功能区别

## 良好的数据水平分布策略

路由算法的基础，性能和伸缩性的保证  
连续性唯一标识最大程度避免数据倾斜  
物理节点虚拟化应对大量不均衡删除  
分组一致性hashing自动识别新节点，避免数据迁移  
Hashing重组进行整理

## 数据访问更复杂

数据定位  
多存储引擎的数据合并、过滤、统计、去重、排序等



## 分布式索引

高效的分布式数据访问  
Hashing的精准定位+范围查询  
最大程度发挥“库内计算”

## 和存储引擎解耦的数据库

实现K-V存储以外的所有功能

# 融合型分布式数据库的RASP

## 分布式数据库独有的RASP

### 读写高性能

数据分片  
最优执行计划

#### 读

一份数据多个拷贝  
一份拷贝多个读服务  
实时步骤优化

### 数据和服务的高可用

数据的多重备份

#### 读

数据的多个读服务

#### 写

数据的多个写服务  
应用级的数据复制保证数据的业务完整性，异构存储引擎间数据同步

### 数据规模和访问可伸缩

数据分布策略保证无数据倾斜

### 数据和服务的高可靠

可用性、稳定性  
强大的tracing和现场记录

# 服务型分布式数据库的优势

## 数据/通讯协议全透明

数据库的访问灵活  
一键变成开放平台

## 服务型分布式计算架构

完全屏蔽分布在多节点的数据和服务带来的  
复杂性，轻松的实现分布式。

服务型分布  
式计算平台  
的传承

## 服务的热插拔

规模扩展异常轻松

## 服务级的RASP

各种数据访问、二次过滤、二次整合、索引  
更新都是服务

# 我们发明车子不发明轮子

## 托翁法则

✓ 以数据水平分布为核心的分布式数据解决方案的架构都是类似的，share nothing、数据分布、数据整合、就近计算、高可用等；

✓ 一体机、集群、NoSQL提供不同的耦合程度、个性化特性以及不同的成本（共享存储、高速通信、FPGA过滤、HDFS的通信成本、Thrift的可用性等）；

✓ 融合型分布式数据库保持底层的优秀设计，更多整合创新，服务型分布式计算带来更加底层的抽象、轻设计和高性能。

# 各种数据库方案对比

	融合型分布式数据库	单机关系型数据库	主从式数据库集群	数据库一体机	水平切分的关系型数据库集群	类NoSQL数据库	MPP数据库	类HDFS的分布式文件系统
数据规模可伸缩性	可以	不可以	不可以	有限	可以	部分可以	可以	可以
数据高可用	读写	无	读写	单点故障	部分可以	部分可以	部分可以	读写
读写性能随节点数提升	线性	不能	读非线性	有限	线性	部分线性	线性	线性
支持数据类型	任意	结构化	结构化	结构化	结构化	（非）结构化	结构化	非结构化
支持业务场景	各种	OLTP	OLTP	OLTP/OLAP	OLTP	OLAP	OLAP	批处理
支持全SQL	能	能	能	能	部分	部分	能	不能
支持异构数据整合	能	不能	不能	不能	不能	不能	不能	不能
总体拥有成本（TCO）	低	低	中等	高	低	中等	低	高

# 服务型分布式计算+融合型分布式数据库

一切应用不改变业务逻辑轻松分布式  
解决各种场景下永恒的存储问题和数据库瓶颈



## OLTP

### 展现层

单一服务支持多种终端  
UI和后台服务完全隔离

### 业务层

服务虚拟化

基于配置的服务缓冲池

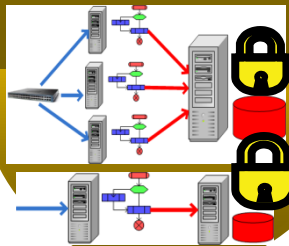
### 数据访问层

数据存储虚拟化

基于配置选择合适的存储引擎和数据部署形式

基于任何关系型数据库  
全面线性可伸缩

**服务和数据的RASP**



## 会话型应用

### 业务逻辑

丰富的语义支持

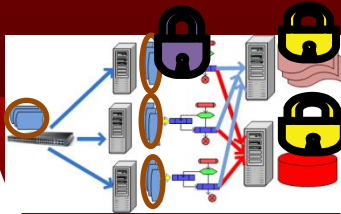
各种数据共享模式

低资源消耗的大并发连接支撑

### 数据存储

混搭的存储引擎提供各种灵活的存储形式

数据和服务容量的线性可伸缩



## 并行/分布式计算

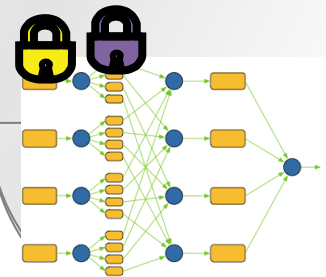
### 业务逻辑

丰富的语义支持

服务型分布式计算解决大数据时代的海量计算问题

### 数据存储

数据语义丰富，灵活存储形式，统一存储视图  
融合型分布式数据库解决大数据时代的海量数据、多样性数据、高速分析需求





1

分布式的3W问题

2

服务型分布式计算

3

基于混搭存储引擎的融合型分布式数据库

4

大数据与分布式计算/分布式数据库

# 大数据时代的量变



大数据提升了数据的价值，  
催化了数据驱动型商业模式



获取全量和全  
方位数据

隐性数据获取成为可能  
全量全方位数据

新数据带来新  
价值

全量全方位的数据中蕴含更大价值

传统数据得到  
重视

新数据的整合提升传统数据价值  
传统数据被重视  
传统数据粉墨登场大数据舞台：  
BI、物联网、工业4.0、等等

数据价值更易发  
掘

新数据提升数据处理能力  
传统数据价值发掘更加高效

数据价值得到  
升华

新数据的维度和体量爆炸性增长  
应用发生量变  
数据价值得到升华  
数据开始被重视

# 大数据时代的技术需求

大数据



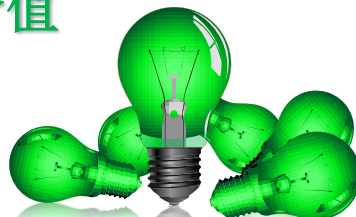
让数据说话  
和数据对话

高性价比的分布式计算和分布式数据库  
挖掘低价值密度数据中的宝藏

海量数据处理要求速度  
数据的时效性  
高效的发掘数据中的价值

服务型分布式计算  
融合型分布式数据库  
灵活的数据分析工具

大价值



全量数据、海量数据  
分布式数据库是必要条件  
分布式计算支撑海量数据处理

全方位的数据维度  
跨存储引擎的数据处理才  
可以发掘数据中的联动价值

混搭的存储引擎支撑各种类型的数据  
扩充的语义支撑混杂型数据的联动、  
连接、联合分析  
统一的数据视图



# 社会化媒体大数据案例分析

## 数据获取

分布式抓取数据  
多存储引擎支持提取数据

## 数据加工、清洗、装载

自然语言处理  
机器学习  
数据清洗、过滤、转换  
高速数据加载

## 数据的存储、整合

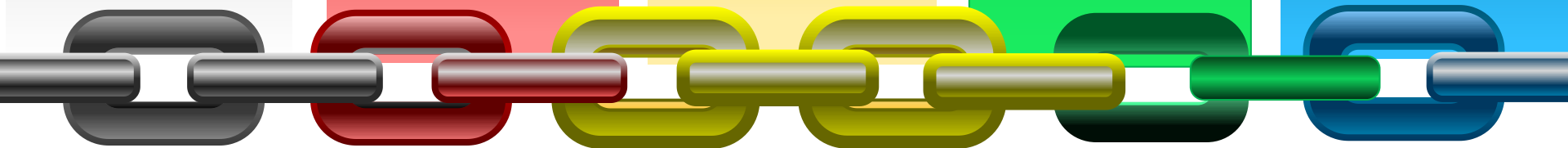
海量异构的社会化媒体数据  
结构化的内部数据  
实时的流数据  
数据整合，价值升华

## 数据分析/挖掘

各种形式、角度分析  
机器学习  
业务人员自助建模、快速探索  
高速在线分析，实时自修复

## 数据和业务的整合

数据模型以widget或者数据形式嵌入业务系统  
mashup的魅力，数据出现在该出现的地方



# 关于我们

## 我们是谁

最早的交换机中间件、内存数据库，最可靠的智能交换机  
业界第一交易中间件、消息中间件  
覆盖世界500强的3000+世界级商业应用

## 如何看世界级商业分布式计算平台、中间件和数据库

深谙前世今生、光鲜阴暗  
没有黑科技，高瞻远瞩的架构、完善缜密的设计、繁琐异常的实现、踏踏实实的积累  
各自有优势，非技术原因的发展困境  
云计算、大数据时代失去了舞台



## 我们做什么

通用的分布式计算和分布式数据库平台  
大牌互联网公司的定制系统标准化、平民化  
生逢其时

## 我们怎么做

通用系统，乐高理念，底层抽象，递进上升  
不做井底之蛙、不闭门造车、不发明车轮、不抄袭模仿  
964原则  
云计算、大数据时代，底层技术和优秀软件设计方法论从未改变

## 为什么这么做

很多大型应用没有合适的支撑产品  
我们一直在深度参与和观察  
深度参与和磨练换取的门票



董健

博晓通

email: [brucedong@inter3i.com](mailto:brucedong@inter3i.com)

Tel: +86 18611886922