阿里云
专有云企业版

技术白皮书

产品版本：V3.6.0

文档版本：20191028

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的”现状“、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表本文档中的内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含”阿里云”、Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止**： 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告**： 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于警示信息、补充说明等，是用户必须了解的内容。	 **注意**： 导出的数据中包含敏感信息，请妥善保存。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明**： 您也可以通过按**Ctrl** + **A**选中全部文件。
>	多级菜单递进。	**设置** > **网络** > **设置网络类型**
粗体	表示按键、菜单、页面名称等UI元素。	单击 **确定**。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *`Instance_ID`*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *`[-all|-t]`*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *`{stand | slave}`*

目录

1 云服务器ECS

1.1 什么是云服务器ECS

云服务器ECS（Elastic Compute Service）提供一种处理能力可弹性伸缩的计算服务，它的管理方式比物理服务器更简单高效。根据业务需要，您可以随时创建实例、扩容磁盘或批量删除多台云服务器实例。

云服务器ECS（以下简称ECS实例）是一个虚拟的计算环境，包含CPU、内存等最基础的计算组件，是云服务器呈献给每个用户的实际操作实体。ECS实例是云服务器最为核心的概念，您可以通过ECS管理控制台完成对ECS实例的一系列操作。其他资源，包括块存储、镜像、快照等，只有与ECS实例结合后才能使用，详见图 1-1: ECS示意图。

图 1-1: ECS示意图

1.2 产品架构

云服务器ECS系统由虚拟化平台与分布式存储、控制系统、运维及监控系统组成。

1.2.1 虚拟化平台与分布式存储

虚拟化是ECS的基础。阿里云采用KVM虚拟化技术，将物理资源进行虚拟化，通过虚拟化后的虚拟资源，对外提供弹性计算服务。

ECS包括两个重要的模块：计算资源模块和存储资源模块。

- 计算资源指CPU、内存、带宽等资源，通过将物理服务器上的计算资源虚拟化再分配给ECS使用。一台ECS的计算资源只能位于一台物理服务器上。当一台物理服务器上资源耗尽时，只能在另外的物理服务器上创建ECS。通过资源的QoS，保证同一台物理服务器上不同ECS互不影响。
- 存储采用了大规模分布式存储系统，将整个集群中的存储资源虚拟化后，整合在一起对外提供服务。同一台ECS的数据，保存在整个集群中。在分布式存储系统中，每份数据都提供三份副本，当单份数据损坏后，可实现数据的自动拷贝。

关于虚拟化平台与分布式存储的原理，具体如下各图所示。

图 1-2: 三副本存储

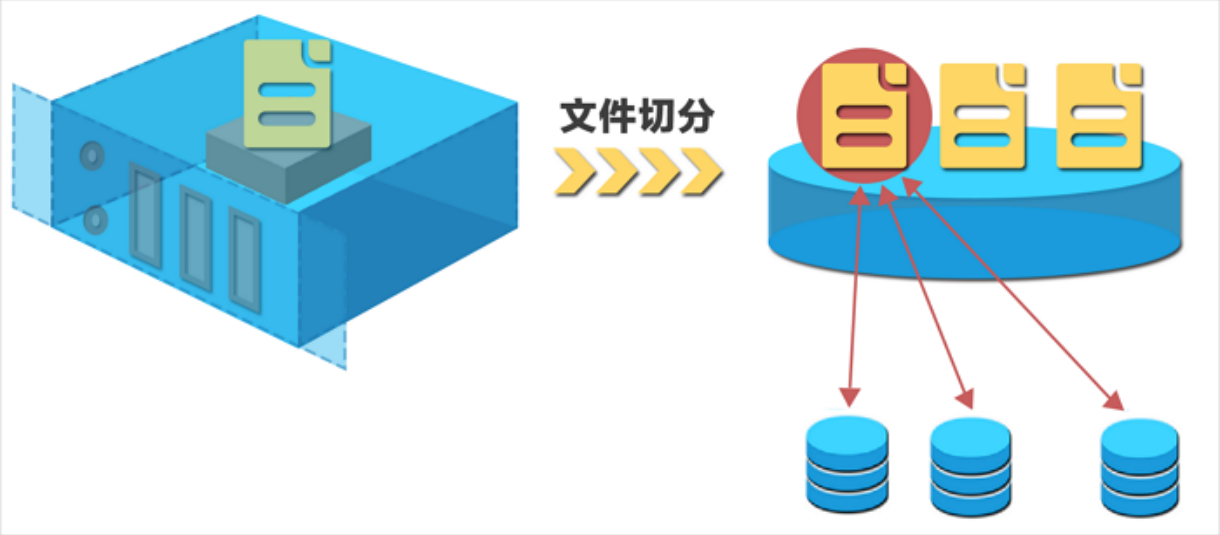


图 1-3: 数据的自动拷贝

1.2.2 控制系统

控制系统是弹性计算平台的核心，它决定着ECS启动在哪一台物理服务器上且ECS的所有功能及信息都需要通过控制中心统一处理与维护。

控制系统由以下模块组成：

- **数据采集**

  负责整个虚拟化平台的数据采集，包括计算资源、存储资源、网络资源等使用情况。通过数据采集可以对集群的资源使用情况进行统一的监控管理，并作为资源调度的一个重要依据。

- **资源调度系统**

  决定ECS启动的位置，在创建ECS时，会根据物理服务器的资源负载情况，合理地调度ECS。且在ECS发生故障时，决定ECS再次启动的位置。

- **ECS管理模块**

  管理及控制ECS，例如启动、关闭、重启云服务。

- **安全控制模块**

  进行整个集群的网络安全监控与管理。

1.3 功能特性

ECS是弹性计算产品的核心部分，主要为用户提供计算服务。创建并启动一台ECS只需数分钟，且ECS一经创建即有特定的系统配置。与传统服务器相比，大大提升了用户业务开展的效率。

使用ECS与传统托管物理服务器的使用方法完全相同，用户对ECS有完全控制权，可通过远程的方式或API的方式（控制面板）对ECS进行一系列的基本操作。

ECS的计算能力可用虚拟CPU、虚拟MEM来表示；磁盘存储能力可用云磁盘容量来衡量。区别于传统服务器，ECS有较为灵活的机器配置。用户可以根据需求灵活配置ECS，在服务器运行过程中，如果现有服务器配置不能满足业务需求，可随时调整服务器配置。

ECS的生命周期从ECS创建到ECS释放。当ECS释放后，所有的数据将彻底删除，不可找回。

阿里云专有云云服务器ECS控制台管理界面一般会包括：

- **概览**

  提供了已创建的和运行中的实例数量，以及各可用区ECS资源的数量分布情况。

- **实例**

  查看和管理已创建的实例；提供在线重启、停止、启动、删除、登录VNC、更换系统盘、修改密码、变配等操作；提供查看实例的基本信息和配置信息。

- **磁盘**

  查看和管理已创建的磁盘；提供在线重新初始化磁盘、创建快照、设置自动快照策略、删除磁盘、挂载/卸载磁盘的功能；提供查看磁盘基本信息以及挂载信息。

- **镜像**

  查看和管理已创建或被分享的镜像；提供镜像复制、镜像共享、镜像删除等功能。

- **快照**

  查看和管理已创建的快照；提供在线回滚磁盘、创建自定义镜像、删除快照的功能。

- **自动快照策略**

  查看和管理已创建的自动快照策略；提供自动快照策略批量设置、修改自动快照策略信息以及自动快照策略的删除等功能。

- **安全组**

  查看和管理已创建的安全组；提供安全组的创建、修改、删除以及批量删除等功能；提供安全组内的实例以及安全组内规则的查看功能。

- **弹性网卡**

  查看和管理已创建的弹性网卡；提供弹性网卡的创建、修改、删除以及实例绑定和解绑等功能。

- **部署集**

  查询和管理已创建的部署集；提供部署集的创建、修改以及删除等功能；提供查看部署集的基本信息功能。

2 容器服务

2.1 什么是容器服务

容器服务（Container Service）是一种高性能可伸缩的容器管理服务，支持企业级Kubernetes容器化应用的生命周期管理。

容器服务简化集群的搭建和扩容等运维工作，整合阿里云虚拟化、存储、网络和安全能力，打造云端最佳的 Kubernetes 容器化应用运行环境。容器服务是Kubernetes认证服务供应商，全球首批通过Kubernetes一致性认证的平台服务，为您提供专业的支持和服务。

2.1.1 容器技术

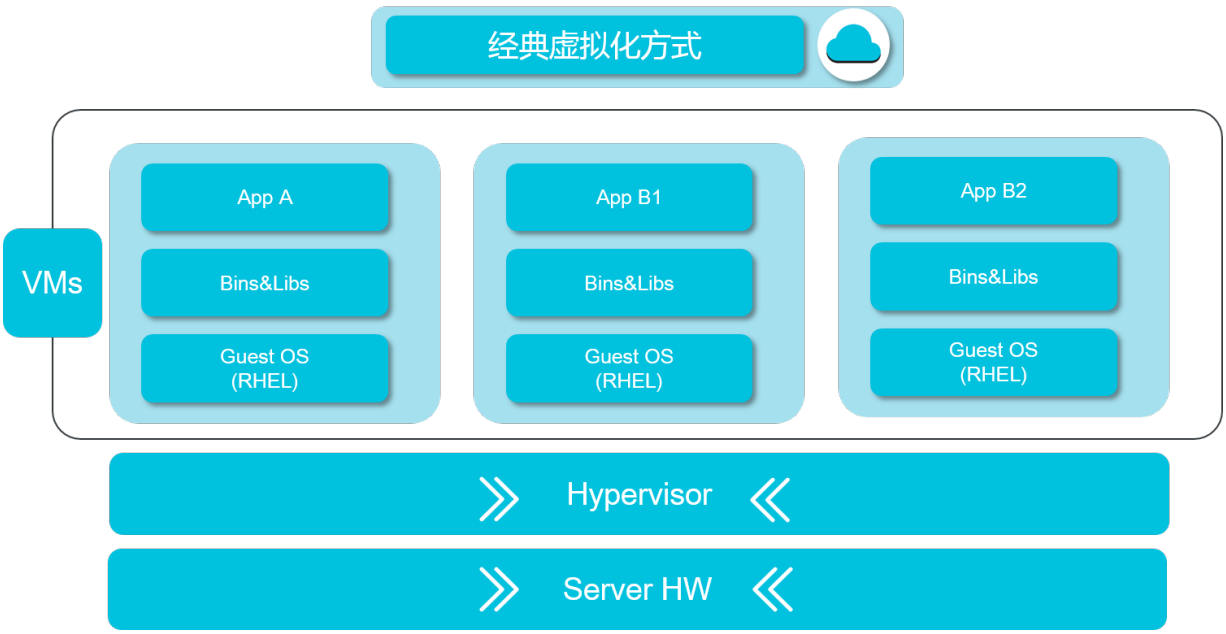
容器技术是一种轻量级的操作系统级虚拟化技术。用户通过容器镜像来交付应用，其中包含了应用程序及所需的运行时依赖。容器镜像具有良好的可移植性，可以在不同环境下保证部署的一致性。容器之间运行时相互隔离，具有相当好的安全性。

容器技术避免了不同应用在同一个环境中可能存在的版本冲突，以及同一个软件在不同环境中可能存在的运行环境不一致的问题。所有容器共享宿主机的操作系统内核，这使得容器比虚拟机更轻量级，可以快速启动，并进行细粒度的资源控制。

容器技术与虚拟化

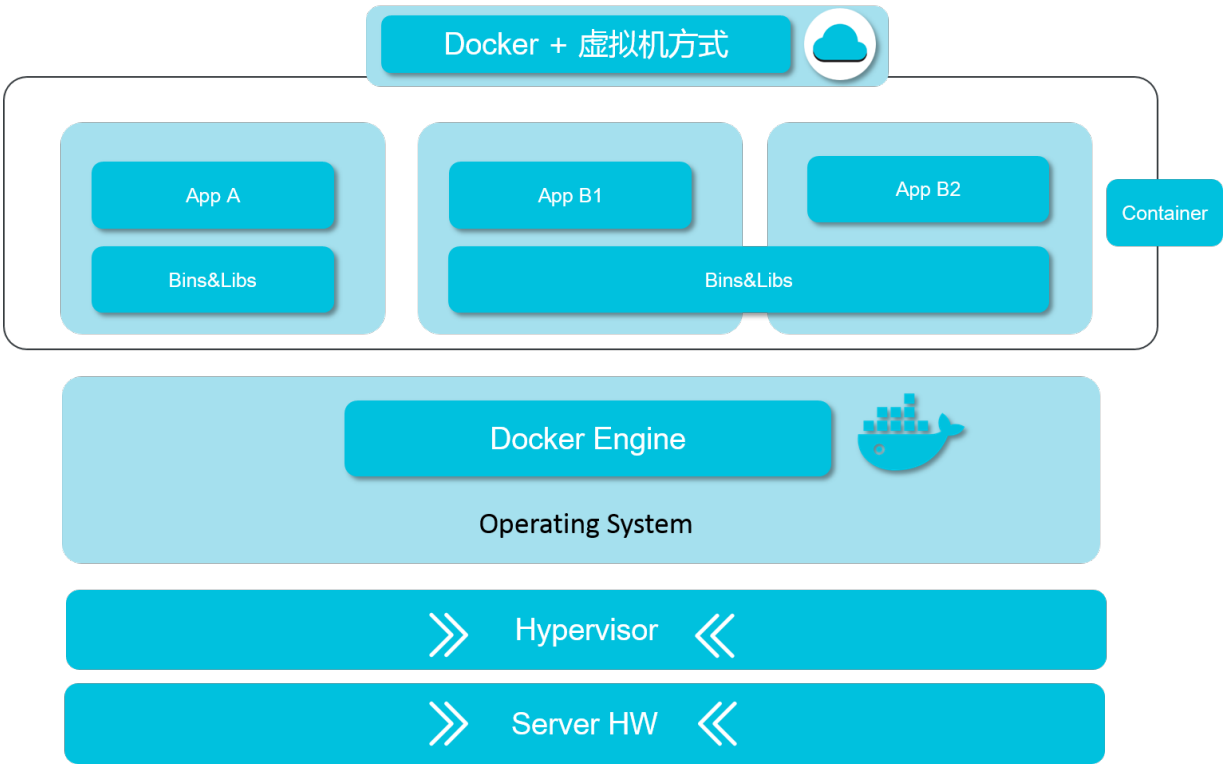
容器技术和传统的虚拟化并不冲突。传统的虚拟化方式是将操作系统到应用的所有要素全部包含在一起，如下图所示。

图 2-1: 经典虚拟化



容器只将应用的代码和运行环境打包，镜像可以进行复用，非常简单。

图 2-2: Docker+虚拟化



结合容器和虚拟化技术，可以利用虚拟机提供弹性基础架构，提供更好的安全隔离，动态热迁移能力；同时还可以利用容器技术实现简化应用部署、运维，实现弹性应用架构。

技术特点

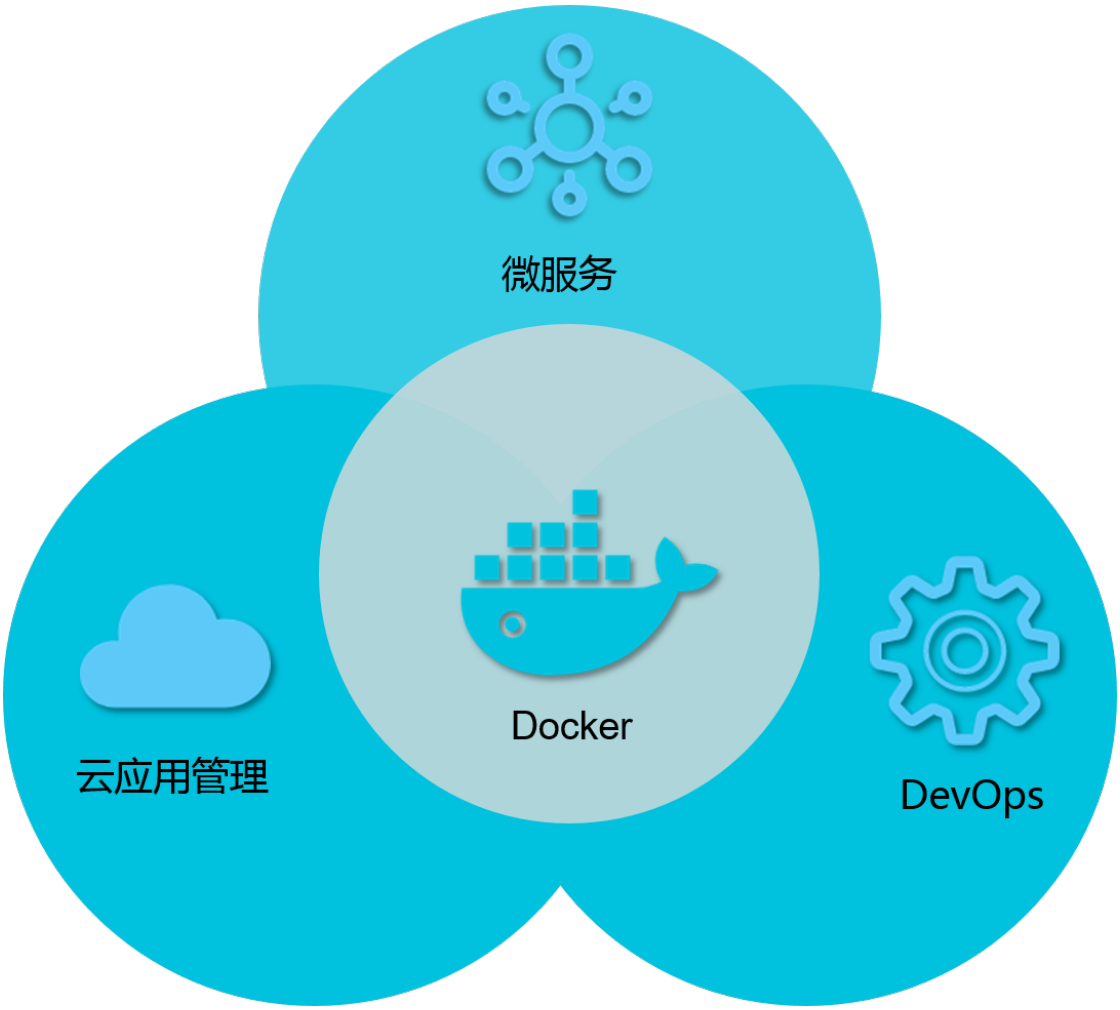
容器技术的特点是：敏捷、可移植、可控。

- **敏捷**：简单快速是容器技术吸引开发人员的重要特性，一致性的交付能力使得软件的交付更快，企业的开发效率更敏捷。
- **可移植性**：开发人员可以把容器化的应用从开发环境转移到测试环境，最终到生产环境。在这个过程中同样的镜像运行结构一致。这意味着计算能力可以跨越数据中心边界进行部署，从而让混合云中的计算能力迁移真正可行。
- **可控**：生产环境的应用需要保证服务等级协议（SLA），要有完善的管理能力、安全和监控能力。容器技术使应用环境标准化，开发人员可以利用自动化工具来管理基础架构和应用，保证了所有操作自动化、可控、可回溯。

应用场景

容器技术可以应用在非常多的场景。热门的场景包括： DevOps、云应用管理和微服务，这三个场景对容器技术要求较高，讨论得比较多，研究也比较充分。

图 2-3: 热点场景



2.2 产品架构

阿里云容器服务完全兼容Kubernetes yaml编排语法，兼容Kubernetes集群管理，并在此基础上做了大量的扩展和针对阿里云的深度优化；支持通过图形化界面和Open API管理集群和容器应用。

在底层架构上，您拥有并独占云服务器或物理机，保证底层环境安全可控，支持定制安全组和专有网络 VPC 安全规则。

为了使您的应用低成本上云，容器服务实现了兼容标准Docker API的程序接口，兼容所有的Docker镜像，提供Kubernetes yaml编排示例模板，支持应用无缝迁云，同时也为第三方提供了灵活可定制的扩展机制。

容器服务的系统架构如下所示。

图 2-4: 系统架构

容器服务基于原生Kubernetes进行适配和增强，简化集群的搭建和扩容等工作，整合阿里云虚拟化、存储、网络和安全能力，打造云端最佳的 Kubernetes 容器化应用运行环境。

产品特性	特性说明
Dedicated集群形态	融合阿里云虚拟化技术，可使用ECS、EGS、神龙服务器作为集群节点，实例规格灵活配置，支持丰富的插件。
阿里云Kubernetes集群管控服务	支持强大的网络、存储、集群管理、水平扩容、应用扩展等特性。
阿里云Kubernetes管理服务	支持安全镜像，与阿里云RAM、KMS、日志、监控等产品高度集成，提供一个安全合规的Kubernetes解决方案。
便捷、高效的使用方式	容器服务Kubernetes版提供Web Console 和API&SDK。

容器服务的能力栈如下图所示。容器服务构建在云基础设施之上，和阿里云能力深度整合并支持第三方扩展，可以支持不同的应用类型。

图 2-5: 功能架构

2.3 功能特性

产品功能

集群管理

- 通过控制台10分钟一键创建经典Dedicated Kubernetes集群，支持GPU服务器。
- 提供容器优化的OS镜像，提供**稳定测试和安全加固**的Kuberntes和Docker版本。
- 支持多集群管理，支持集群升级和伸缩。

一站式容器生命周期管理

- **网络**

  提供阿里云优化的高性能VPC/ENI网络插件，性能优于普通网络方案20%。

  支持容器访问策略和流控限制。

- **存储**

  支持阿里云云盘、对象存储OSS，提供标准的FlexVolume驱动。

  支持存储卷动态创建，迁移。

- **日志**

支持高性能日志自动采集和阿里云日志服务集成。

支持和第三方开源日志解决方案集成。

- **监控**

  支持容器级别和VM级别的监控。您还可以和第三方开源监控解决方案进行集成。

- **权限**

  支持集群级别的RAM授权管理。

  支持应用级别的权限配置管理。

- **应用管理**

  支持灰度发布，支持蓝绿发布。

  支持应用监控，应用弹性伸缩。

高可用调度策略，轻松打通上下游交付流程

- 支持服务级别的亲和性策略和横向扩展。
- 支持跨可用区高可用和灾难恢复。
- 支持集群和应用管理的 OpenAPI，轻松对接持续集成和私有部署系统。

3 弹性伸缩ESS

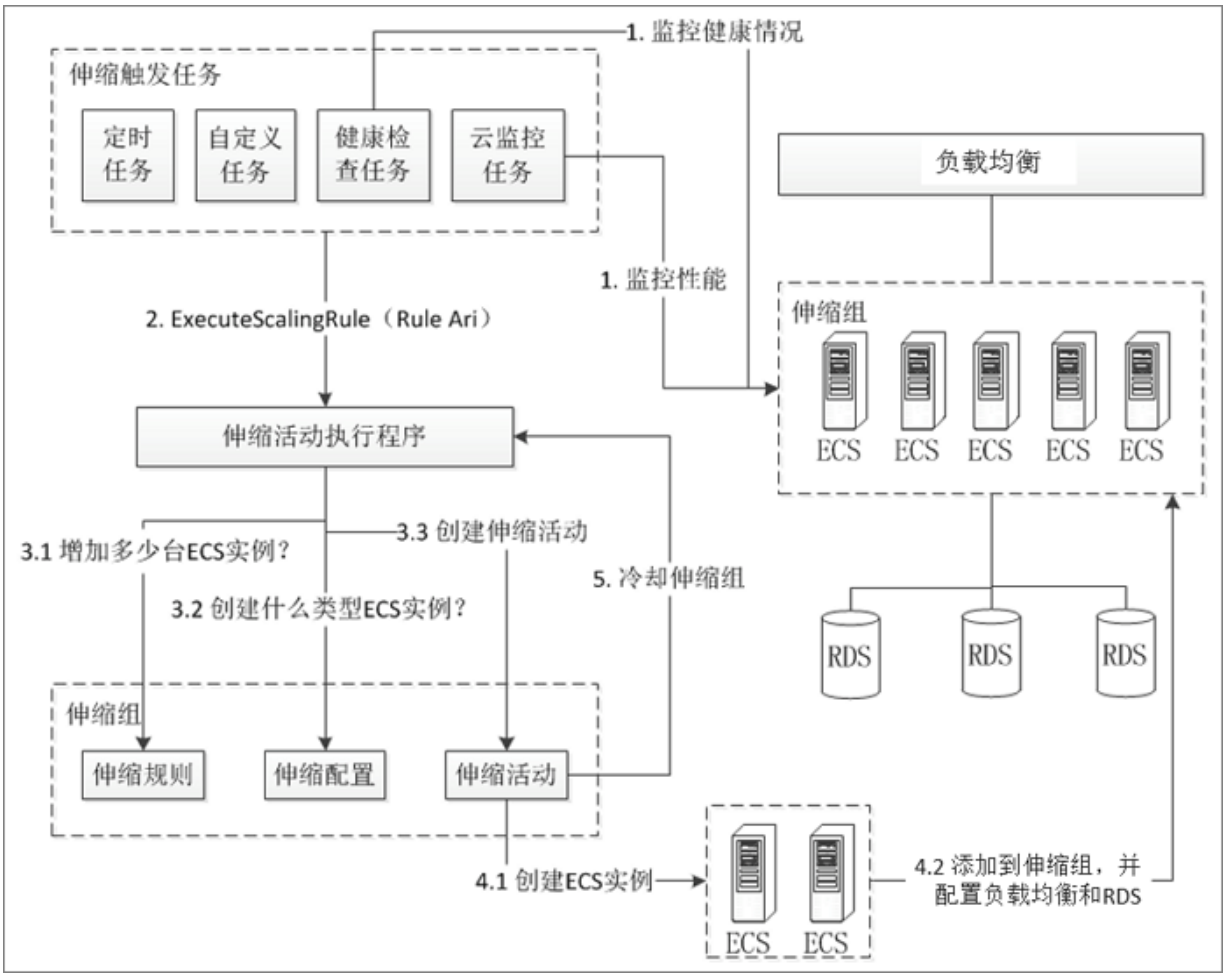
3.1 什么是弹性伸缩ESS

弹性伸缩服务（Elastic Scaling Service）是根据用户的业务需求和策略，经济地自动调整弹性计算资源的管理服务。弹性伸缩不仅适合业务量不断波动的应用程序， 同时也适合业务量稳定的应用程序。

弹性伸缩根据用户的策略定义和业务需求变化，动态地调整弹性计算资源，在有效支撑业务负载变化的同时保持最合理经济的基础设施费用开支。它可以根据用户设置的伸缩策略和模式，在业务需求增长时自动增加ECS实例以保证计算能力，在业务需求下降时自动减少ECS实例以节约成本，还可以自动替换不健康的ECS实例使业务始终保持正常的负载，为业务保驾护航。

在此基础上，弹性伸缩与负载均衡SLB和云数据库RDS无缝集成，可以自动向负载均衡的后端服务器组中添加或移除相应的ECS实例，以及自动向RDS访问白名单中添加或移除ECS实例的IP，无需人工干预，即可应对各种复杂场景，真正实现对业务负载的弹性处理能力。详见图 *3-1:* 弹性伸缩示意图。

图 3-1: 弹性伸缩示意图


1. 监控健康情况
伸缩触发任务
定时任务
自定义任务
健康检查任务
云监控任务
负载均衡
1. 监控性能
伸缩组
2. ExecuteScalingRule（Rule Ari）
ECS ECS ECS ECS ECS
伸缩活动执行程序
3.1 增加多少台ECS实例？
3.3 创建伸缩活动
5. 冷却伸缩组
3.2 创建什么类型ECS实例？
RDS RDS RDS
伸缩组
伸缩规则
伸缩配置
伸缩活动
4.1 创建ECS实例
ECS ECS
4.2 添加到伸缩组，并配置负载均衡和RDS

3.2 产品架构

弹性伸缩是对ECS实例做服务编排的系统，依赖ECS等基础组件提供服务。弹性伸缩系统由任务调度、任务处理、数据库和中间件服务组成。

图 3-2: 架构示意图

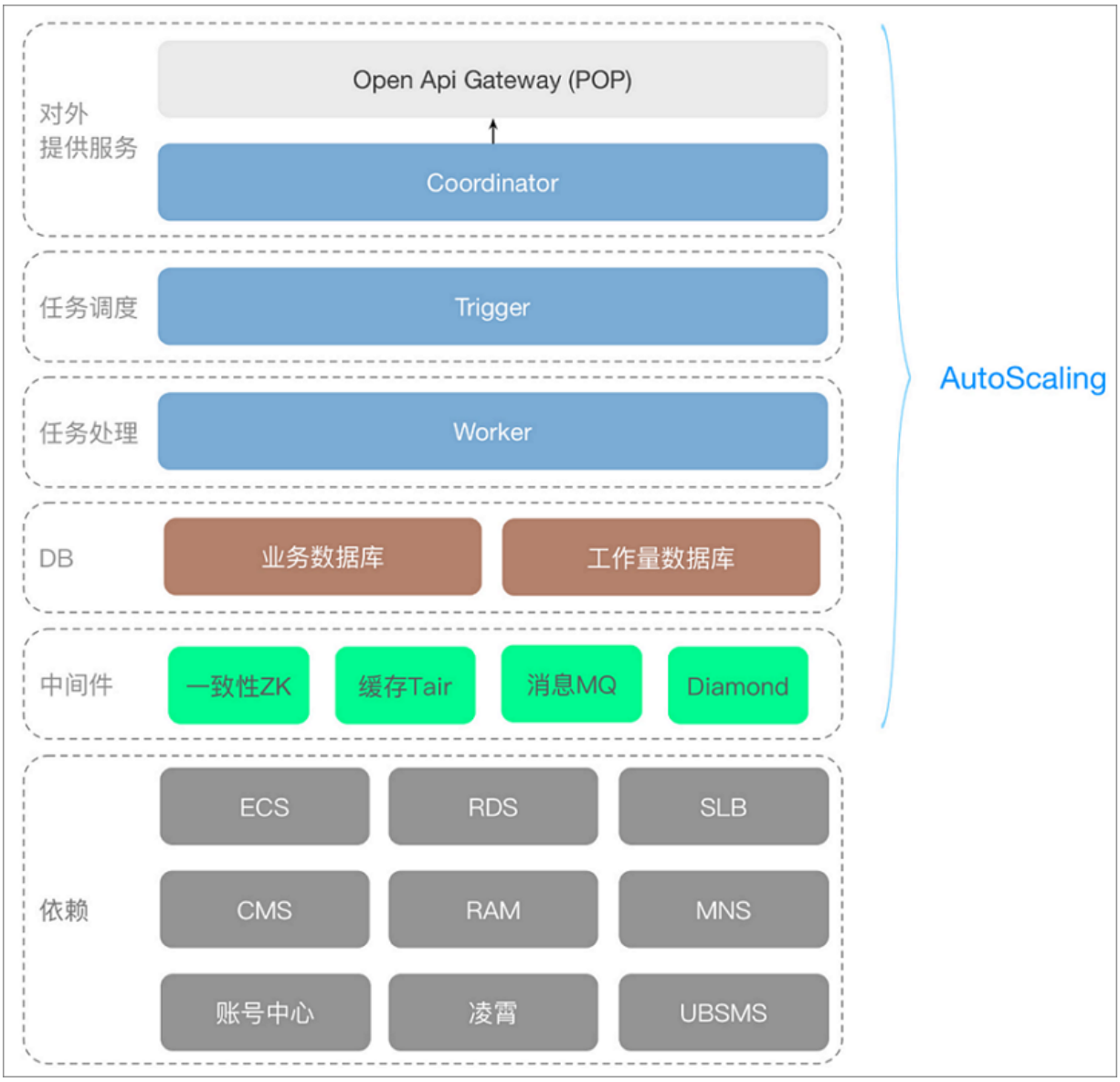


表 3-1: 架构说明

层级名称	说明
中间件层	一致性中间件Zookeeper 给后裔管控提供分布式锁的实现。
	缓存中间件Tair 给后裔管控提供缓存服务。

层级名称	说明
	消息中间件MQ 虚拟机状态消息队列服务 。
	Diamond 管理持久配置服务。
数据库层，包括业务和工作量数据库。	任务处理，Worker 整个系统的核心部分，接收任务，并对任务进行分解、执行、反馈结果等流程。
	任务调度，Trigger 负责实例和伸缩组健康检查、定时任务以及与云监控的交互，转化为调度任务。
对外提供服务	Coordinator 整体产品的入口，对外提供服务的管控层，处理各个接口调用、任务触发等。
	Open API Gateway，提供鉴权和参数透传等基础服务。

3.3 功能特性

3.3.1 典型场景

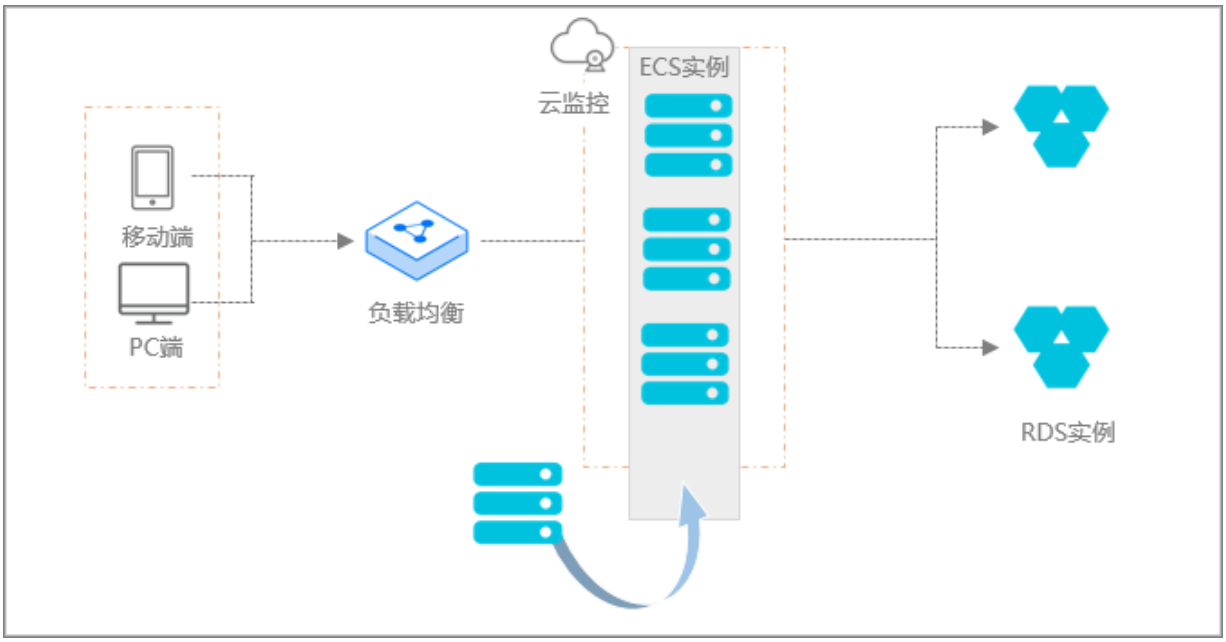
弹性伸缩自动为您调整弹性计算资源大小，以满足您业务需求的变化。弹性伸缩根据您设置的伸缩规则，在业务需求增长时自动为您增加ECS实例以保证计算能力，在业务需求下降时自动减少ECS实例以节约成本。

3.3.1.1 弹性扩张

当您的业务升级时，弹性伸缩为您自动完成底层资源升级，避免访问延时和资源超负荷运行。

您可以设置定时任务来执行弹性扩张，或配置云监控实时关注您的ECS实例使用情况。例如，当云监控检测到伸缩组内的ECS实例vCPU使用率突破80%时，弹性伸缩根据您配置的伸缩规则弹性扩张ECS资源，自动创建合适数量的ECS实例，并自动添加ECS实例到负载均衡实例和RDS实例的访问白名单中。

图 3-3: 弹性扩张

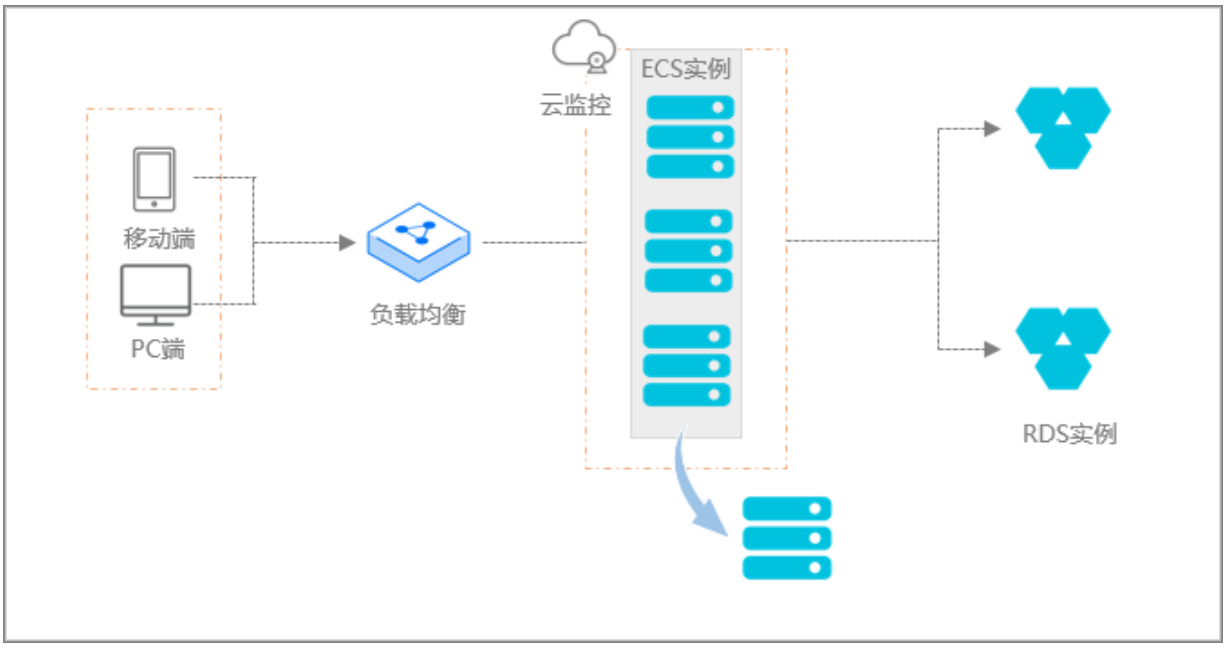


3.3.1.2 弹性收缩

当您的业务需求下降时，弹性伸缩为您自动完成底层资源释放，避免资源浪费。

您可以设置定时任务来执行弹性扩张，或配置云监控实时关注您的ECS实例使用情况。例如，当云监控检测到伸缩组内的ECS实例vCPU使用率低于30%时，弹性伸缩根据您配置的伸缩规则弹性收缩ECS资源，自动释放合适数量的ECS实例，并自动从负载均衡实例和RDS实例的访问白名单中移除ECS实例。

图 3-4: 弹性收缩

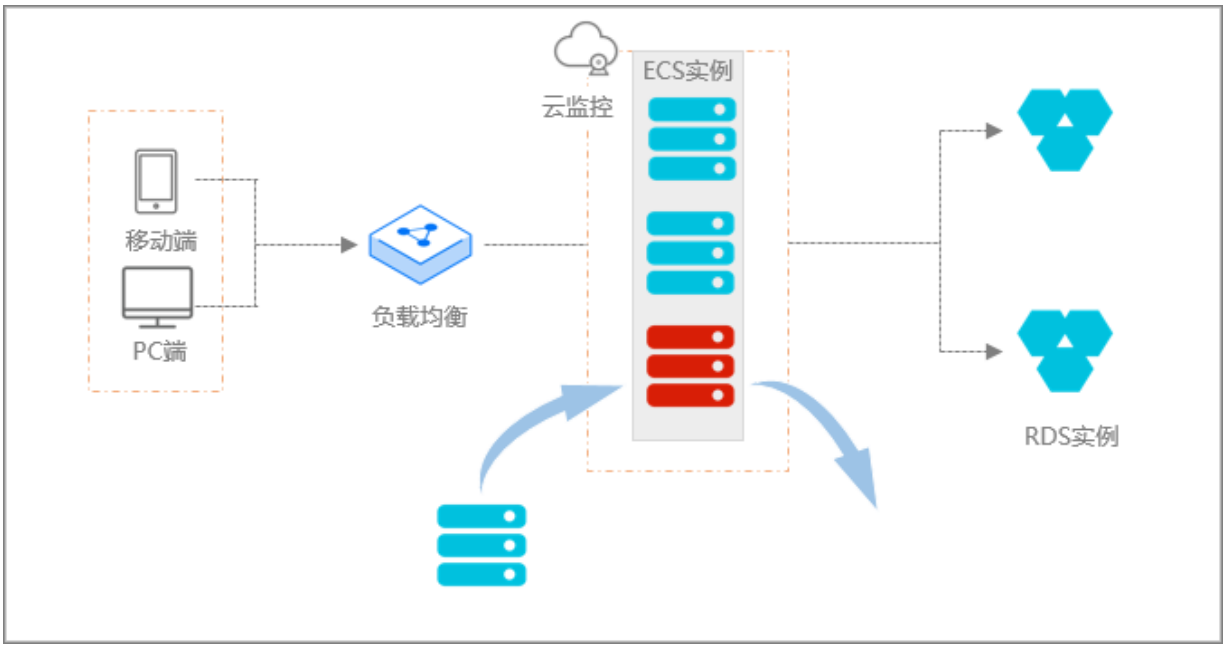


3.3.1.3 弹性自愈

弹性伸缩提供健康检查功能，自动监控伸缩组内的ECS实例的健康状态，避免伸缩组内健康ECS实例低于您设置的最小值。

当检测到某台ECS实例处于不健康状态时。弹性伸缩自动释放不健康ECS实例并创建新的ECS实例，自动添加新ECS实例到负载均衡实例和RDS实例的访问白名单中。

图 3-5: 弹性自愈



3.3.2 功能组件

创建完整的弹性伸缩方案，您需要创建伸缩组、伸缩配置、伸缩规则、并设置定时任务来实现弹性扩张、收缩等场景。

弹性伸缩的创建流程图如下所示：

伸缩组

伸缩组（Scaling Group）是具有相同应用场景的 ECS 实例的集合。伸缩组定义了组内 ECS 实例数的最大值、最小值及其相关联的负载均衡实例和 RDS 实例等属性。

伸缩配置

伸缩配置（Scaling Configuration）是弹性伸缩用于创建 ECS 实例的模板。在创建伸缩配置时，您可以指定 ECS 实例的信息，例如实例规格、镜像类型、存储大小、登录实例用的密钥对等。您也可以修改现有伸缩配置，快速满足新的业务需求。

伸缩规则

伸缩规则（Scaling Rule）指弹性伸缩服务弹性扩张或收缩 ECS 资源时所依据的规则，目前支持以下三种规则：

- 调整至 N 台：执行伸缩规则后，服务中的实例数将被调整至 N 台。
- 增加 N 台：执行伸缩规则后，服务中的实例数在当前数量基础上增加 N 台。
- 减少 N 台：执行伸缩规则后，服务中的实例数在当前数量基础上减少 N 台。

定时任务

定时任务是一种按照指定执行计划管理伸缩组的方式，可以在您指定的时间触发您指定的伸缩规则来执行伸缩活动，达到调整伸缩组内实例个数的目的。

4 对象存储OSS

4.1 什么是对象存储OSS

阿里云对象存储服务（Object Storage Service，简称 OSS）是阿里云提供的海量、安全、低成本、高可靠的云存储服务。

OSS可以被理解成一个即开即用，无限大空间的存储集群。相比传统自建服务器存储，OSS 在可靠性、安全性、成本和数据处理能力方面都有着突出的优势。使用 OSS，您可以通过网络随时存储和调用包括文本、图片、音频和视频等在内的各种非结构化数据文件。

OSS 将数据文件以对象/文件（Object）的形式上传到存储空间（Bucket）中。OSS 提供的是一个 Key-Value 键值对形式的对象存储服务。用户可以根据Object的名称（Key）唯一地获取该Object的内容。

您可以进行以下 OSS 相关的操作：

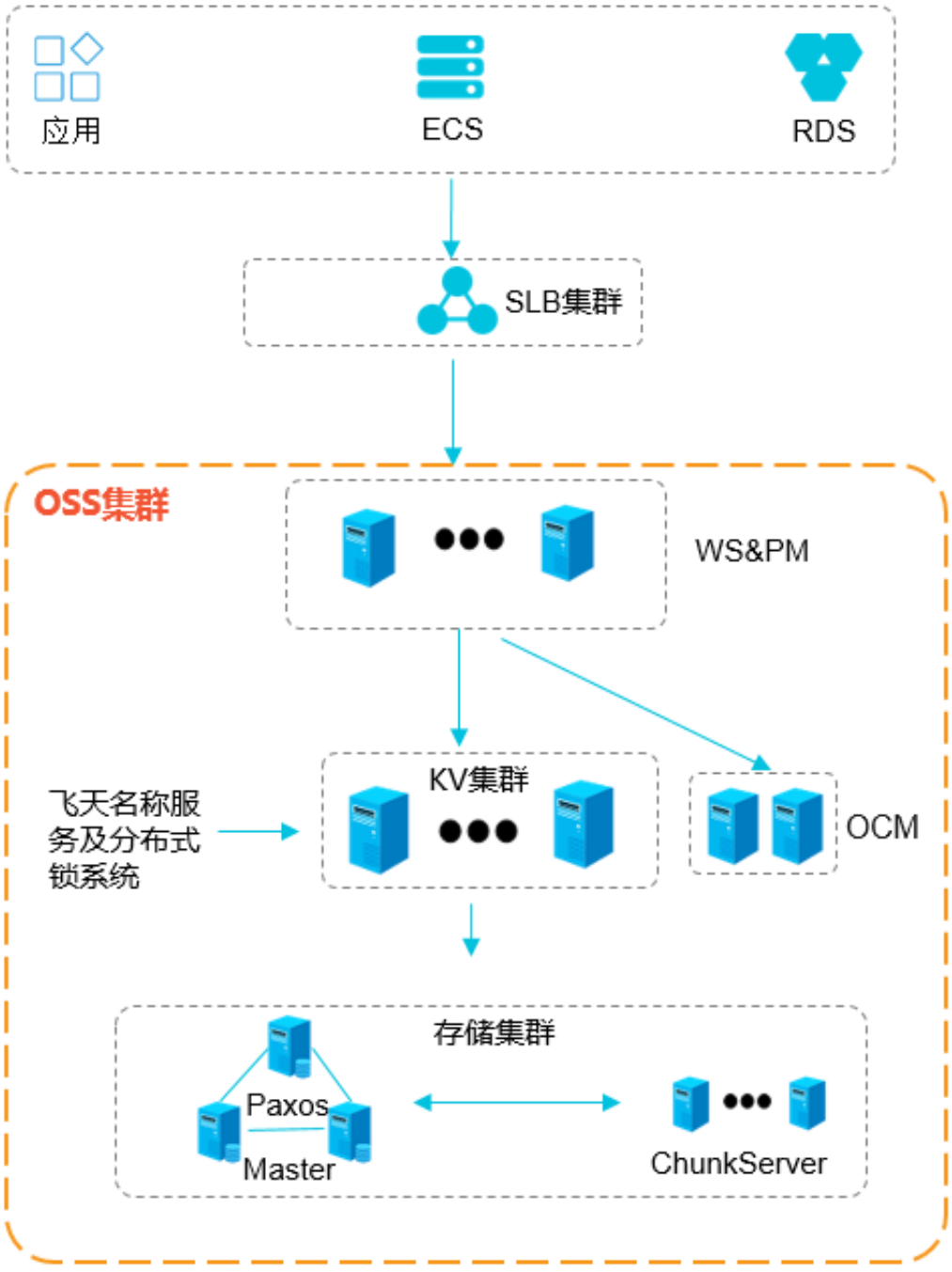
- 创建存储空间，并向存储空间中上传文件。
- 获取已上传文件的地址，进行文件的分享和下载。
- 修改存储空间或文件的属性或元信息，为其设置相应的权限。
- 在对象存储 OSS 控制台进行基础和高级 OSS 操作。
- 通过开发工具包 SDK 或直接在应用程序中调用 RESTful API，进行基础和高级 OSS 操作。

4.2 产品架构

对象存储OSS是构建在阿里云飞天平台上的一种存储解决方案。其基础是飞天平台的分布式文件系统、分布式任务调度等基础设施。这些基础设施提供了OSS以及其他阿里云服务所需的分布式调度、高速网络、分布式存储等重要特性。

OSS的架构如下图所示。

图 4-1: OSS架构图



数据流经负载均衡分发至OSS后的处理如下：

1. 最上层是协议接入层，负责接收用户使用RESTful协议发来的请求，进行安全认证。如果认证通过，用户的请求将被转发到Key-Value引擎继续处理；如果认证失败，直接返回错误信息给用户。

2. 数据访问层负责数据结构化处理，即按照Key来查找或存储数据，支持大规模并发的请求。当协调服务集群变更导致服务被迫改变运行物理位置时，可以快速协调找到接入点。
3. 最底层是持久存储层，即大规模分布式文件系统。元数据存储在Master上，Master之间采用分布式消息一致性协议（Paxos）保证元数据的一致性。从而实现高效的数据分布式存储和访问，保证数据在系统中有3个备份以及在软硬件错误发生以后的故障恢复。

4.3 功能特性

存储空间概览

展示请求者所拥有的所有Bucket，在通过HTTP访问OSS服务地址时将默认展示您所拥有的所有Bucket。

设置并查询Bucket访问权限

用户可以设置和查看Bucket的访问权限。Bucket的访问权限可以为以下任意一种：

- Private（私有权限）：只有该存储空间的创建者或者授权对象可以对该存储空间内的文件进行读写操作，其他人在未经授权的情况下无法访问该存储空间内的文件。
- Public-read（公共读，私有写）：只有该存储空间的创建者可以对该存储空间内的文件进行写操作，任何人（包括匿名访问）可以对该存储空间中的文件进行读操作。
- Public-read-write（公共读写）：任何人（包括匿名访问）都可以对该存储空间中的文件进行读写操作。

创建/删除 Bucket

一个用户默认最多创建 10 个 Bucket。新创建的 Bucket 命名要符合Bucket命名规范。

创建Bucket可能存在以下情况：

- 如果创建的 Bucket 不存在，系统按照 Bucket 名称创建 Bucket，并返回成功标志。
- 如果要创建的Bucket已存在，且请求者是所有者，则保留原来 Bucket，并返回成功标志。
- 如果要创建的 Bucket 已存在，且请求者不是所有者，则返回失败标志。

删除Bucket的操作需要满足以下条件：

- Bucket 存在。
- 访问者对 Bucket 有删除权限。
- Bucket 为空。

列出 Bucket 中的所有 Object

根据 Bucket 名称列出此 Bucket 下的所有 Object 信息，访问者必须具有对相应 Bucket 的操作权限，当访问的 Bucket 不存在时返回错误信息。

OSS 支持前缀查询，可以设置一次最大返回的文件数量（最大支持设置1000）。

上传/删除 Object 文件

上传 Object 到指定的 Bucket 空间下。在满足如下条件下 Object 会上传成功：Bucket 存在、访问者拥有对 Bucket 相应的操作权限。当 Bucket 中存在同名 Object 文件时将会覆盖掉原来的 Object 文件。根据 Object 名称删除某个特定 Object，访问者必须有此 Object 相应的操作权限。

获取 Object 文件或元信息

取得 Object 文件内容信息或者元信息，访问者需要对 Object 有相应的操作权限。

访问 Object

OSS 支持通过 URL 方式访问某文件。

日志及监控操作

用户可以选择开启 Bucket 的日志记录功能，一旦开启，OSS 会按照小时粒度推送日志。用户可以在Apsara Stack控制台OSS首页查询存储空间、流量、请求等信息。

OSS VPC访问控制

用户可以创建OSS和VPC之间的通道（Single Tunnel），实现从VPC访问OSS资源。

5 表格存储TableStore

5.1 什么是表格存储

表格存储（Table Store）是构建在阿里云飞天分布式系统之上的 NoSQL 数据存储服务，提供海量结构化数据的存储和实时访问。

5.1.1 技术背景

DT 时代下的数据特点

随着移动互联网的普及并深入到各个行业和领域中，互联网应用呈现出如下几个非常显著的特点和趋势：

- 应用需要存储和处理的数据量接近指数级增长，比如微博、社交事件、图片、访问日志等。
- 诸如手机等移动设备的普及以及物联网设备的增加，结构化数据的存储将面临越来越高的写入并发。
- 需要处理的数据没有严格的 schema，更趋向于半结构化，数据的字段会动态的变化。
- 用户的访问存在明显的热点和高峰，比如各种大促期间，应用的用户访问量会在瞬间达到非常高的值。
- 由于移动互联网无时无刻都在接入用户，用户对互联网应用的可用性要求也非常高，很难接受故障导致的服务不稳定甚至是计划中的服务停机。
- 大量的数据信息对计算分析提出了更高的要求。

传统 IT 软件解决方案的挑战

使用传统 IT 软件解决方案很难面对这些新的趋势和挑战，主要体现在如下几个方面：

- 规模可扩展

  传统的软件如关系型数据库很难处理这样快速增长的数据量，不管是在数据写入的吞吐率还是在数据量变大之后的访问效率上，都存在巨大的瓶颈。于是使用传统数据库的方案不得不进行手工和静态的分库分表策略，而这个策略也意味着很大的系统维护代价，特别是在增加节点进行扩容的时候，需要对已有的数据进行重新的切分和迁移，在这个过程中服务的性能、稳定性和可用性都很难得到很好的保证，而且整个过程是非常复杂的。
- 数据模型变更

传统数据库处理的数据都具有严格的 schema，数据中包含的列数通常是固定的，很少去修改。频繁修改表 schema 和列数的设置，会对服务的可用性产生较大的影响，因此传统方案在面对结构越来越松散的互联网应用的数据时显得很力不从心。

- 快速伸缩

  传统的解决方案中，业务的访问压力是比较平稳的，系统不会经常面临资源需要快速调整（扩容和减容）的情况，因此一旦发生这种情况，就需要很大的代价，比如对数据进行重新切分，对切分之后的数据进行迁移，一旦业务压力下降之后，为了避免资源利用率低的问题，又要对多余的机器进行下线处理，又会经历数据的再一次搬迁，整个过程极其复杂且效率低下。

- 运维保障

  使用传统的软件方案需要专门来处理机器硬件（网络、磁盘）等设备发生故障时的服务恢复，需要处理硬件的更换，需要处理软件的版本升级，配置调优和更新，要让这些过程对应用透明，不影响服务的可用性，需要有专门的运维保障和系统工程师团队才能达到。不管是从人才招聘还是从成本投入上来说，这些工作对于快速发展的企业都是巨大的挑战。

- 计算瓶颈

  在现有的业务系统中，我们通常使用的是 OLTP （OnLine Transaction Processing，联机事务处理）系统来对数据进行处理和分析，如 MySQL、Microsoft SQL Server 等关系数据库系统。这些关系数据库系统擅长事务处理，在数据操作中保持着严格的一致性和原子性，能够很好支持频繁的数据插入和修改，但是，一旦需要进行查询或计算的数据量过大，达到数千万甚至数十亿条，或需要进行的计算非常复杂，OLTP 类数据库系统便力不从心了。

5.1.2 表格存储技术

表格存储是构建在阿里云分布式操作系统飞天之上的 NoSQL 数据存储服务，通过将数据表进行分区并且将数据分区调度到不同的节点上进行服务，从而提供可扩展的能力。在单机的硬件出问题时，表格存储服务通过心跳机制快速发现有问题的节点，并且把该节点上数据分区快速迁移到健康的节点上继续服务，从而达到服务的快速恢复能力。

数据分区和负载均衡

表中每一行主键的第一列称为数据分片键（Partition Key），系统根据数据分片键的范围将表切分为多个分区，这些分区被系统均匀地调度到不同的存储节点上。当单个数据分区内的数据不断增加到一定程度时，分区会自动分裂成两个更小的分区，从而将数据和访问 load 分散到两个分区上，这两个分区会被调度到不同的节点上，从而将访问 load 分散到不同的节点上，最终达到了单表数据规模和访问压力的线性扩展。

技术指标：表格存储支持的单表最大能够到 PB 级别，最多能够提供百万级别的并发读写能力。

单机故障自动恢复

在表格存储的存储引擎中，每个节点都会服务一批不同表的数据分区，这些分区的分布和调度信息由一个 Master 节点来负责管理，并且这个 Master 节点也会监控每个服务节点的健康状态。当发现某个服务节点出现问题时， Master 就会将原先分配给这个服务节点的数据分区调度到其他健康的节点上。由于数据分区的迁移只是逻辑上的迁移，不涉及实际数据的移动，所以当出现单机故障时，服务能够在很短的时间内恢复。

技术指标：单个机器节点的故障只影响部分数据分区的服务，并且能够在分钟级别恢复。

同城及异地灾备

为满足业务对安全性和可用性的需求，表格存储提供了同城及异地双集群主备容灾，容灾粒度为实例级别。容灾实例下的数据表上的数据插入、更新、删除操作会异步复制到备实例下的同名表，主备实例的数据保持一致的时间取决于主备集群的网络环境，在理想的网络环境下，为毫秒级别延时。所以在进行手动切换时，需要先停止对主集群的读写，并等待所有数据备份完成，再切换服务到备集群。主备切换之后，1 个小时内不能再进行主备切换，且需要清理原集群数据并重新设置备集群信息。

同城双集群主备容灾中，应用访问主备集群的表格存储的服务域名不变，即发生切换之后，应用程序无需更改。异地双集群主备容灾中，主备集群的服务域名会不同，在发生切换情况下，应用程序需要更改访问表格存储的域名。

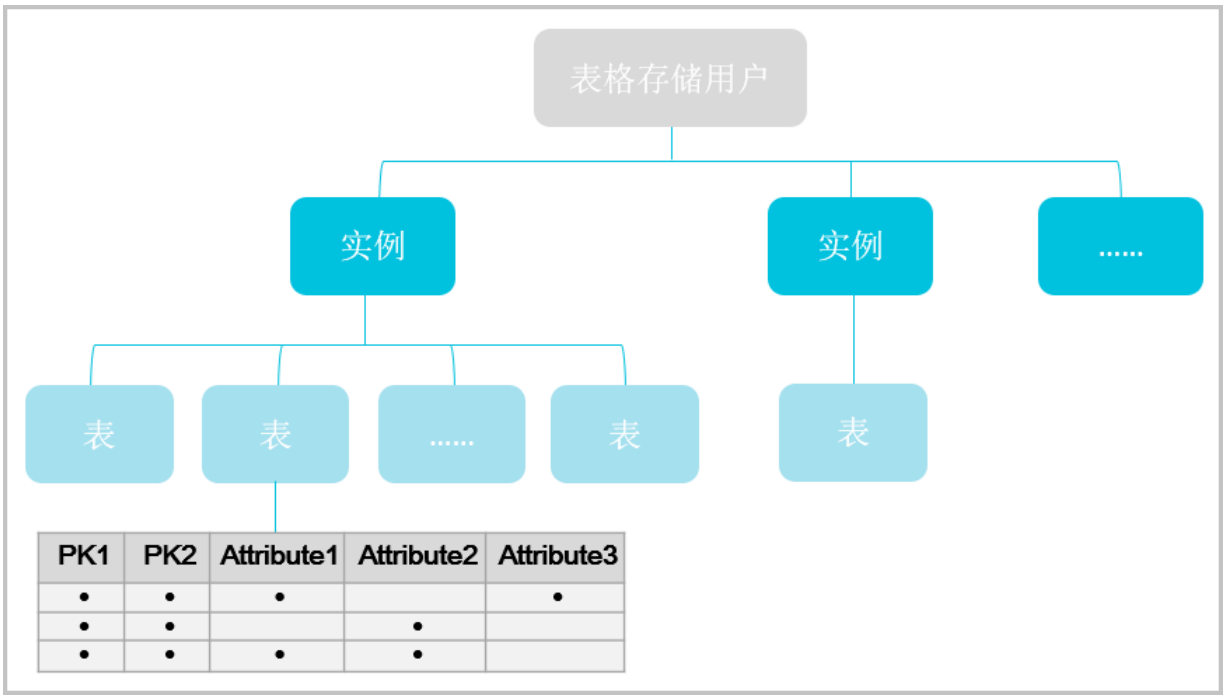
技术指标：RTO 小于 2 分钟，RPO 小于 5 分钟，RCO 为 1。

5.2 功能特性

5.2.1 用户和实例

用户和实例架构图如*Figure 5-1: 用户和实例架构图*所示。

Figure 5-1: 用户和实例架构图

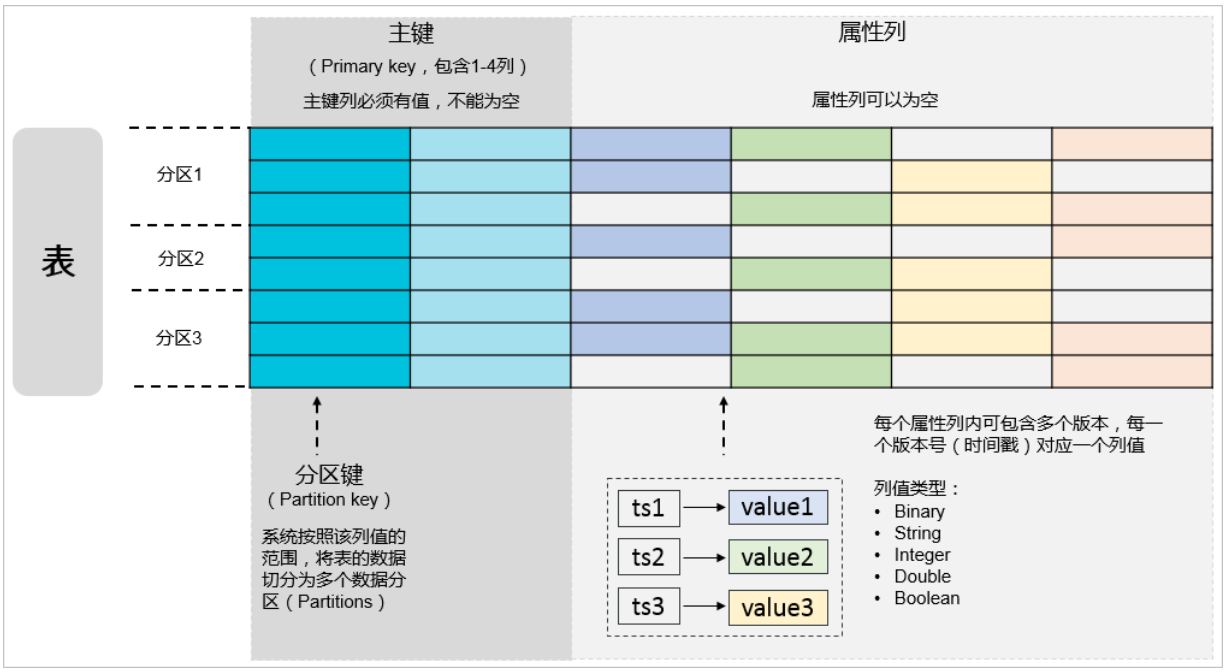


- 通过云账号进行登录。
- 用户的操作均可被细粒度审计。
- 用户通过实例来组织资源，一个用户可以创建多个实例，每个实例可以创建、管理多张数据表。
- 实例是多租户隔离的基本单位。
- 不同的用户可以授予不同的权限。

5.2.2 数据表

数据表结构图如数据表结构图所示。

图 5-2: 数据表结构图



- 数据表是资源分配的最小单元。
- 表是行的集合，行由主键和属性组成。
- 表根据第一个主键列大小对数据进行分片。
- 表中的所有行都必须包含相同数目和名称的主键列。
- 每行包含的属性列的数目、名字和数据类型也可以不同。
- 一行中属性列的个数没有限制，但一次请求写入的属性列不能够超过1024列。
- 单表可支持千亿行甚至更多数据。
- 单表数据规模可达到 PB 级别。

5.2.3 数据分片

- 表根据第一个主键列大小对数据进行分片。
- 第一个主键列值在同一个分片范围内的行会被分配到同一个数据分片。
- 表格存储服务会根据特定的规则对分片进行分裂和合并，以达到更好的负载均衡。
- 同一个分片键下的数据建议不超过 10 GB。

5.2.4 表的常用命令与函数

操作表的常用命令

- ListTable：列出实例下所有的表。

- CreateTable：创建数据表。
- DeleteTable：删除表。
- DescribeTable：获取表的属性信息。
- UpdateTable ：更新表的预留读/写吞吐量配置。
- ComputeSplitPointsBySize：将全表的数据在逻辑上划分成接近指定大小的若干分片，返回这些分片之间的分割点以及分片所在机器的提示。

操作表中数据的常用函数

- GetRow：读取单行数据。
- PutRow：新插入一行数据。
- UpdateRow：更新一行数据。
- DeleteRow：删除一行数据。
- BatchGetRow：批量读取一张或者多张表的多行数据。
- BatchWriteRow：批量插入、更新、删除一张表或者多张表的多行数据。
- GetRange：读取表中一个范围内的数据。

5.2.5 授权与权限控制

表格存储的权限

结合访问控制服务和专有网络，表格存储支持如下的权限控制：

- 表级别的授权操作。
- API 粒度的权限控制。
- 支持 IP 限制、https、MFA（多因素认证）、访问时间限制等多种鉴权条件。
- 临时授权访问（STS）。
- 支持专有网络（VPC）访问控制。

云控制台

- 支持云平台账号登录与鉴权。
- 提供图形化的实例创建、管理和删除的功能。
- 提供图形化的数据表的创建、管理、调整预留读写吞吐量和删除的功能。
- 提供表级别的监控信息展示。

5.3 产品优势

表格存储（Table Store）是构建在阿里云飞天分布式系统之上的 NoSQL 数据存储服务，提供海量结构化数据的存储和实时访问。表格存储以实例和表的形式组织数据，通过数据分片和负载均衡技术，达到规模的无缝扩展。表格存储向应用程序屏蔽底层硬件平台的故障和错误，能自动从各类错误中快速恢复，提供非常高的服务可用性。表格存储管理的数据全部存储在 SSD 中并具有多个备份，提供了快速的访问性能和极高的数据可靠性。用户在使用表格存储服务时，只需要按照预留和实际使用的资源进行付费，无需关心数据库的软硬件升级维护、集群缩容扩容等复杂问题。

表格存储有如下特点：

- **扩展性**

  表格存储的表数据量没有上限，随着表数据量的不断增大，表格存储会进行数据分区调整从而为该表配置更多的存储并提供更高的并发访问能力。

- **数据可靠性**

  表格存储通过存储多个数据备份及备份失效时的快速恢复，提供极高的数据可靠性。

- **高可用性**

  通过自动的故障检测和数据迁移，表格存储对应用屏蔽了机器和网络的硬件故障，提供高可用性。

- **管理便捷**

  应用程序无需关心数据分区的管理、软硬件升级、配置更新、集群扩容等繁琐运维任务。

- **访问安全性**

  表格存储提供多种权限管理机制，并对应用的每一次请求都进行身份认证和鉴权，以防止未授权的数据访问，确保数据访问的安全性。

- **强一致性**

  表格存储保证数据写入强一致，写操作一旦返回成功，应用就能立即读到最新的数据。

- **灵活的数据模型**

  表格存储的表无固定格式要求，每行的列数可以不相同，支持多种数据类型，如 Integer、Boolean、Double、String、Binary。

6 文件存储NAS

6.1 什么是NAS

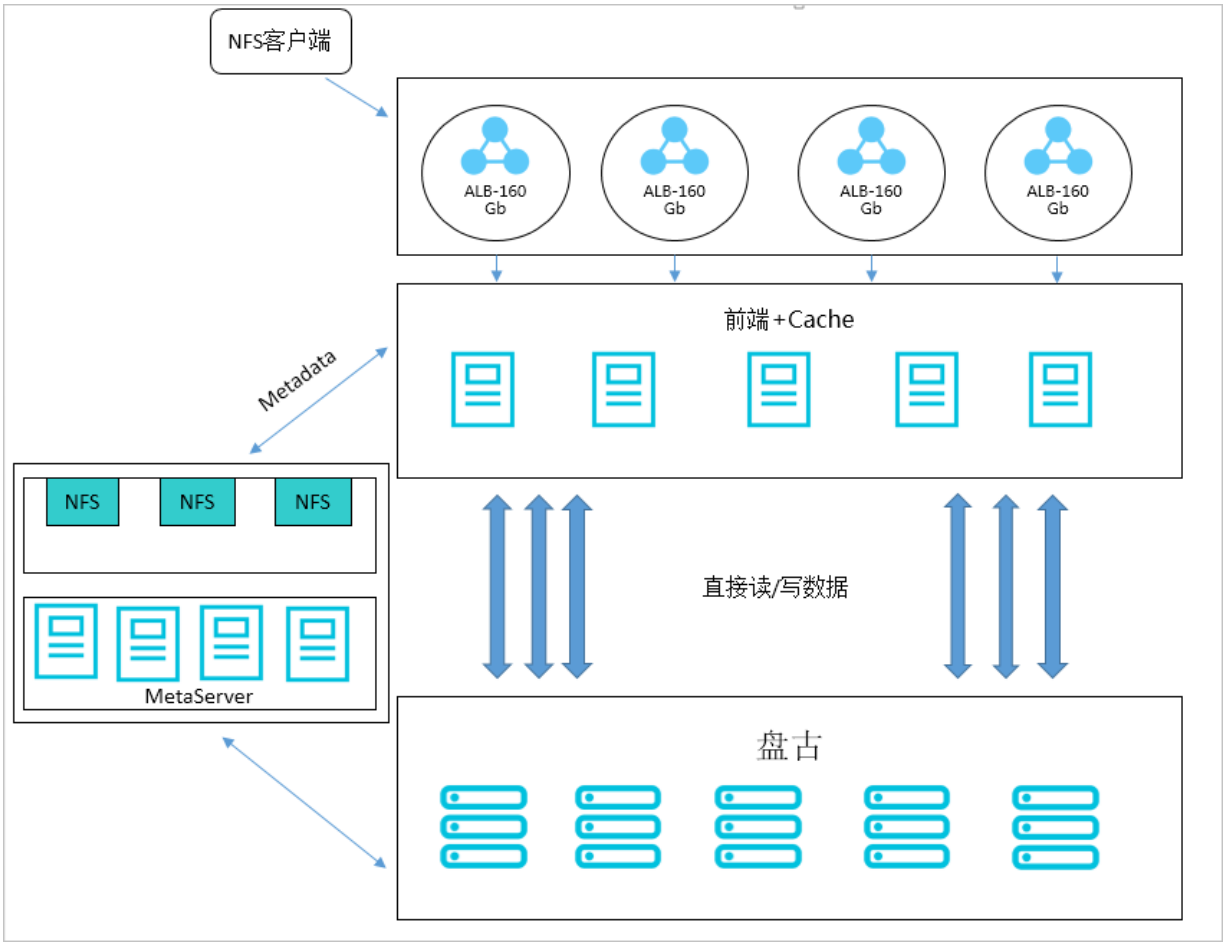
阿里云文件存储（Network Attached Storage，简称NAS）是面向阿里云ECS实例、E-HPC和容器服务的文件存储服务，提供标准的文件访问协议，用户无需对现有应用做任何修改，即可使用具备无限容量及性能扩展、单一命名空间、多共享、高可靠和高可用等特性的分布式文件系统。

6.2 系统架构

NAS后端基于阿里云盘古分布式存储，数据3副本分布存储于多台盘古节点上。前端访问节点接受NFS客户端的连接请求和提供Cache功能，自身是无状态和分布式部署，保证前端的高可用。NAS的Metadata数据保存在MetaServer上，前端机的I/O请求在用MetaServer获得NAS的Metadata后，User Data的读写直接到后端盘古数据节点。

架构上前后端可以单独弹性扩展，在保证高可用的前提下，做到吞吐的高并发和低时延。

图 6-1: 系统架构



6.3 基本功能

NAS支持NFSv3和NFSv4协议，应用无需修改即可以使用。它们都可以满足用户业务对文件存储的需求，例如业务数据文件共享、OA系统后端文件存储、企业数据库的备份存储、业务系统日志的存储和分析、Web站点数据的存储和分发、业务系统开发和测试数据的存储等场景。

图 6-2: 基本功能

6.4 产品优势

- 多共享

  支持1万个Client同时通过NFS v3.0/4.0协议挂载同一个文件系统，实现数据共享。

- 高性能

  集群的最大吞吐可达到20 GBps，IOPS可以达到2万以上。

- 弹性扩展

  业务数据弹性增长。最大单文件系统可达10 PB的存储空间，每个文件系统支持最多10亿文件，单文件最大32 TB。

- 高可靠

基于3副本的盘古分布式系统，提供高数据可靠性，保护用户数据安全。

- 安全

  支持VPC、安全组、ACL、主子账号等安全特性，保障用户数据隔离。

- 全局命名空间

  文件系统数据分布在整个NAS集群上，提供单一命名空间。

7 分布式文件系统DFS

7.1 什么是DFS

阿里云分布式文件系统（Distributed File System，简称DFS）是面向阿里云ECS实例及容器服务等计算资源的文件存储服务，提供标准的HDFS访问协议，用户无需对现有大数据分析应用做任何修改，即可使用具备无限容量及性能扩展、单一命名空间、多共享、高可靠和高可用等特性的分布式文件系统。

用户创建DFS文件系统实例后，即可在ECS及容器服务等计算资源内通过标准的HDFS协议接口访问文件系统。多个计算节点可以同时访问同一个文件系统，共享文件和目录。

7.2 设计理念

背景

大数据应用及 Hadoop 技术的发展对分布式文件系统提出了越来越高的要求。相对于传统的 Hadoop 分布式文件系统（HDFS），用户需要文件系统具有以下的特性：

- 云端数据互通
- 存储数据可以随时计算
- 支持容灾
- 高性能、低成本

针对这些需求，阿里巴巴自主研发了分布式文件系统 DFS。这是一种兼容 Hadoop 的高性能云化分布式文件系统。

设计理念

DFS 基于以下理念设计而成：

- 兼容 Hadoop 生态，Hadoop 应用能够无缝接入云化的海量数据。
- 对软硬件进行一体化设计，提供端到端的极致性能和低成本。
- 与阿里现有的存储产品（OSS/NAS）数据实现内部打通，使数据能够随时接入阿里云的 ECS 计算资源和 MaxCompute 大数据计算服务。
- 提供智能化的管理及运维能力，为用户提供优秀的使用体验。

核心指标

DFS 在设计时的核心指标如下：

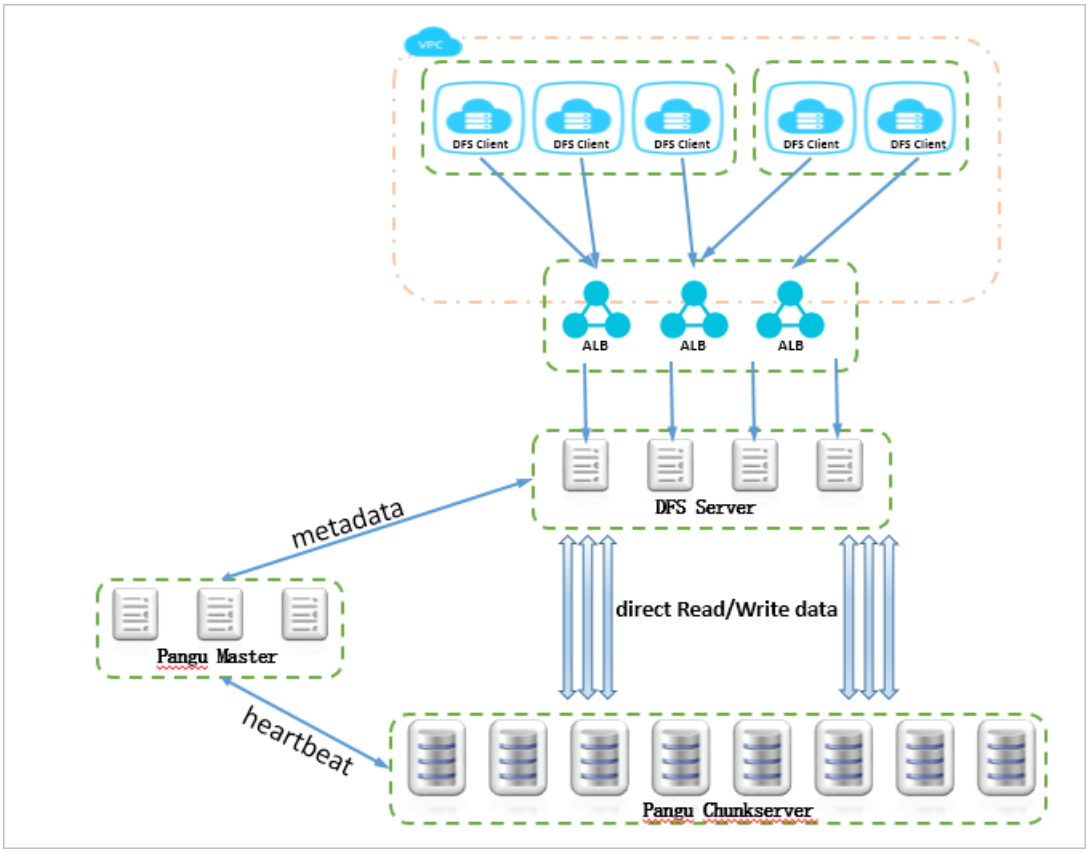
- 兼容 HadoopFS 文件系统接口。
- 支持 3 可用区（AZ）容灾备份，提供高可用性。
- 单存储节点吞吐 20 Gb，并支持线性扩展。
- 单集群规模可以达到 10,000 台以上。
- 支持自动化日常运维及管理。

7.3 系统架构

DFS 分布式文件系统主要分为后端和前端两部分。

DFS 的后端基于阿里云盘古分布式存储系统。数据以多副本的形式分布存储于多台盘古节点上。DFS 的前端访问节点接受 ECS （如 MapReduce 及 Spark 等 Hadoop 计算应用）或容器服务实例的连接请求并提供缓存功能。盘古还负责管理文件系统的 Metadata 和 data 数据。

DFS 的整体架构图如下：



7.4 产品定位

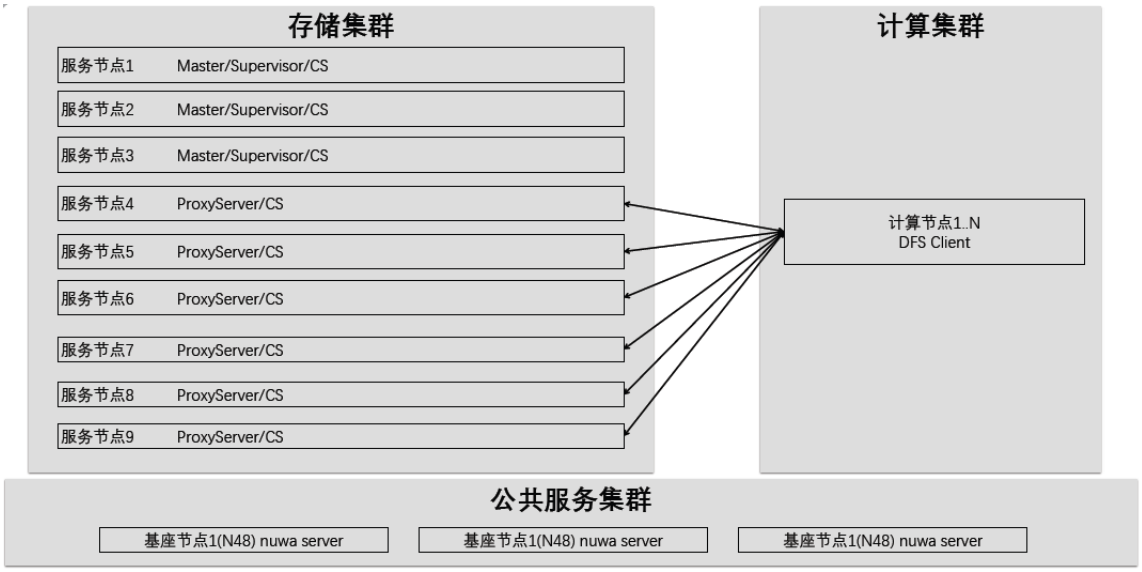
与传统 HDFS 的对比

传统的 HDFS 系统被设计为能够提供高吞吐量的数据访问，以适合大规模数据集上的应用。与其相比，DFS 在系统弹性、小文件存储性能及多租户方面具有自身的优势。

系统弹性

传统 HDFS 的 DataNode 同时也要提供计算能力，因此在规划和容量伸缩时不够灵活。

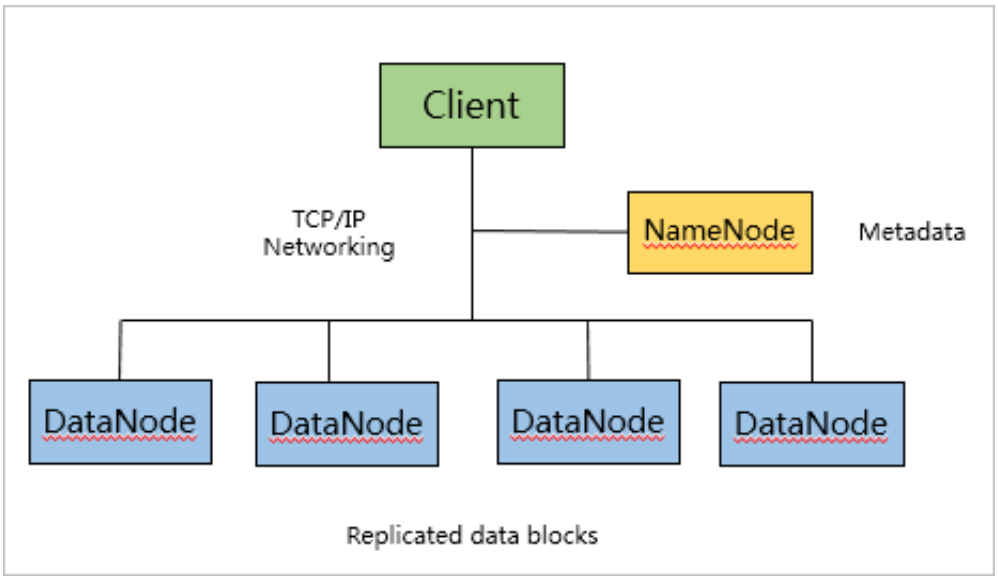
DFS 系统的部署架构如下图所示：



由图中可见，DFS 分布式文件系统在部署时分为两个集群：存储集群与公共服务集群。存储集群和计算集群相互独立，可以分别进行规划和伸缩。计算运行完毕后，资源即可释放，而存储的数据能够继续保留，从而为 DFS 提供了更高的系统弹性。

小文件存储性能

传统 HDFS 系统架构图如下：



如图所示，数据在 HDFS 系统中被分块复制在多个 DataNode 中，各文件块的 Metadata 保存和维护在唯一的 NameNode 中。Client 对文件的所有 I/O 请求都要先从 NameNode 上获取 Metadata，然后采用链式的方式将数据写入每个Datanode。同时，HDFS诞生于HDD时代，无法适应新型闪存型介质，如：SSD、Nvme等。

这种架构适合以流式的方法读取较大的数据集合，而在处理小文件时具有明显的缺点，因为大量的小文件会产生海量的 Metadata 信息，容易耗尽 NameNode 的资源。而且所有文件的操作请求都要经过 NameNode 进行，提高了数据访问的时延。

而 DFS 系统的架构从初始就考虑了小文件的存储需求，通过支持分层存储、优化元数据、优化写入流等手段大幅提高了小文件的吞吐和存储效率。

多租户

传统 HDFS 是为非云计算环境单集群单租户设计。 DataNode 强调数据的本地性，同时提供存储和计算能力，因此计算系统和存储系统是紧密绑定的。

与此相对，DFS 源自云计算环境，具备对云计算的原生支持。作为云存储系统，支持在存储系统中为多个租户建立多个文件系统实例，同时每个实例都支持利用多个计算集群进行运算。

7.5 产品价值

DFS 在便捷性、性能及成本方面具有以下价值：

便捷接入

- 兼容 HadoopFS 文件系统接口，现有 Hadoop 生态应用可零成本接入。
- 与阿里云存储产品数据互通，实现数据池化。

- 与 MaxCompute 大数据计算服务方便接入。

极致性能

- 针对小文件场景进行优化，大幅提高了小文件吞吐性能。
- 充分利用新一代闪存的硬件优势。

节省成本

- 实现存储云化，能够进行弹性伸缩，节省存储成本。
- 与现有存储数据互通，能够即刻连接计算能力，节省时间成本。
- 满足海量小文件存储需求，降低数据管理成本。

7.6 应用场景

场景一：共享存储和高可用

如果用户有以下业务需求，推荐使用 DFS 文件系统存储文件：

- 需要对文件进行共享访问。
- 对文件的可用性要求较高。

应用方式：DFS 文件系统提供标准的 HDFS 协议，因此用户可以使用标准 HDFS 接口将文件实时或者批量存入 DFS 文件系统。

场景二：大数据分析与机器学习

在大数据分析与机器学习场景中，应用对数据访问的吞吐性能和延迟有较高要求。而DFS 能够提供高吞吐量和低延迟的访问能力，无需将数据迁移到计算资源本地，因此在该场景下推荐使用 DFS 系统存储数据。

应用方式：将数据通存入 DFS 系统，ECS 实例或其他计算资源即可直接访问这些数据。将 Hadoop 或其他机器学习应用部署在多个计算资源上，这样应用可以直接通过 HDFS 接口访问数据，进行离线或在线计算，也可以直接将计算结果输出到 DFS 上做永久保存。

8 云数据库RDS版

8.1 什么是关系型数据库RDS

阿里云关系型数据库RDS（Relational Database Service）版包含MySQL、SQLServer、PostgreSQL和PPAS 四种存储引擎，您可以方便、快捷地创建出适合自己应用场景的数据库实例。

阿里云关系型数据库RDS是一种稳定可靠、可弹性伸缩的在线数据库服务。基于阿里云分布式文件系统和高性能存储，提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案，彻底解决数据库运维的烦恼。

云数据库MySQL版

云数据库MySQL版基于阿里巴巴的MySQL源码分支，经过双十一高并发、大数据量的考验，拥有优良的性能和吞吐量。云数据库MySQL版支持实例管理、帐号管理、数据库管理、设置实例白名单、备份恢复、数据透明加密以及迁移数据等基本功能。除此之外还提供如下高级功能：

- **只读实例**：在对数据库有少量写请求，但有大量读请求的应用场景下，单个实例可能无法抵抗读取压力，甚至对主业务产生影响。为了实现读取能力的弹性扩展，分担数据库压力，云数据库MySQL 5.6版的实例支持只读实例，利用只读实例满足大量的数据库读取需求，以此增加应用的吞吐量。
- **读写分离**：读写分离功能是在只读实例的基础上，额外提供了一个读写分离地址，联动主实例及其下的所有只读实例，实现了自动的读写分离链路。应用程序只需连接同一个读写分离地址进行数据读取及写入操作，读写分离程序会自动将写入请求发往主实例，而将读取请求按照用户设置的权重发往各个只读实例。用户只需通过添加只读实例的个数，即可不断扩展系统的处理能力，应用程序上无需做任何修改。
- **CloudDBA数据库性能优化**：针对SQL语句的性能、CPU使用率、IOPS使用率、内存使用率、磁盘空间使用率、连接数、锁信息、热点表等，CloudDBA提供了智能的诊断及优化功能，能最大限度发现数据库存在的或潜在的健康问题。CloudDBA的诊断基于单个实例，该诊断会提供问题详情及相应的解决方案，可为您管理实例运行状况带来极大的便利。
- **数据压缩**：云数据库MySQL 5.6版支持通过TokuDB存储引擎压缩数据。经过大量测试表明，数据表从InnoDB存储引擎转到TokuDB存储引擎后，数据量可以减少80%到90%，即2T的数据量能压缩到400G甚至更低。除了数据压缩外，TokuDB存储引擎还支持事务和在线DDL操作，可以很好兼容运行于MyISAM或InnoDB存储引擎上的应用。

云数据库SQL Server版

阿里云数据库SQL Server版不仅拥有高可用架构和任意时间点的数据恢复功能，强力支撑各种企业应用，同时也包含了微软的License费用，您无需再额外支出License费用。

云数据库SQL Server版支持实例管理、帐号管理、数据库管理、设置实例白名单、备份恢复、数据透明加密以及迁移数据等基本功能。

云数据库PostgreSQL版

PostgreSQL是全球最先进的开源数据库，它的优点主要集中在对SQL规范的完整实现以及丰富多样的数据类型支持，包括JSON数据、IP数据和几何数据等。除了完美支持事务、子查询、多版本控制（MVCC）、数据完整性检查等特性外，阿里云数据库PostgreSQL版还集成了高可用和备份恢复等重要功能，减轻您的运维压力。

云数据库PostgreSQL版支持实例管理、帐号管理、数据库管理、设置实例白名单、备份恢复以及迁移数据等基本功能。

云数据库PPAS版

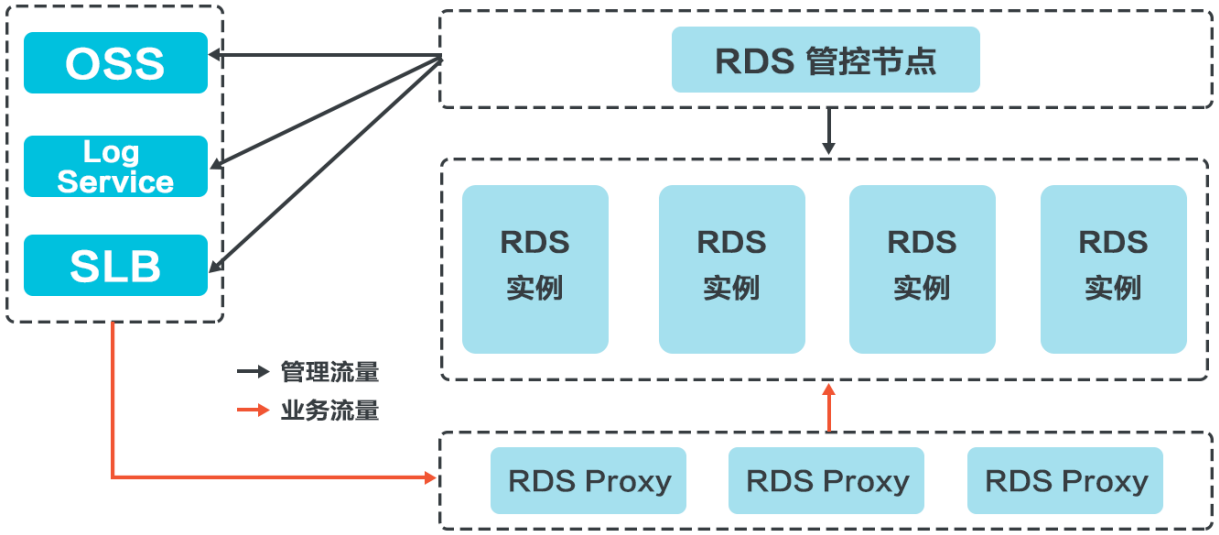
PPAS（Postgres Plus Advanced Server）是一个稳定、安全且可扩展的企业级关系型数据库，基于全球最先进的开源数据库PostgreSQL，并在性能、应用方案和兼容性等方面进行了增强，提供直接运行Oracle应用的能力。您可以在PPAS上稳定地运行各种企业应用，同时得到更高性价比的服务。

阿里云数据库PPAS版集成了实例管理、帐号管理、数据库管理、设置实例白名单、备份恢复以及迁移数据等基本功能。

8.2 产品架构

云数据库RDS版的系统架构如下。

图 8-1: RDS系统架构



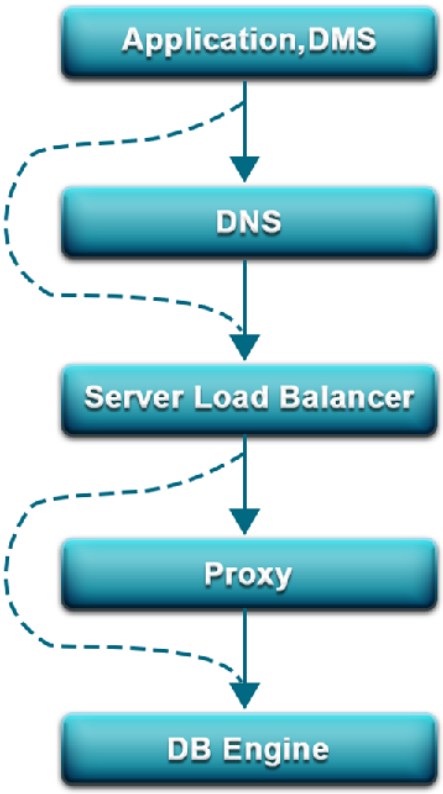
8.3 功能特性

高可用服务RDS主要包括6大核心服务：数据链路服务、调度服务、备份服务、高可用服务、监控服务、迁移服务。

8.3.1 数据链路服务

数据链路服务主要提供数据操作，包括表结构和数据的增删改查。

图 8-2: RDS 数据链路服务



8.3.1.1 DNS

DNS模块提供域名到IP的动态解析功能，以便屏蔽RDS实例IP地址变化带来的影响。例如：

某RDS实例的域名为test.rds.aliyun.com，而这个域名对应的IP地址为10.1.1.1。某程序连接池中配置为test.rds.aliyun.com或10.1.1.1，都可以正常访问RDS实例。

当该RDS实例发生了实例迁移或者版本升级后，IP地址就可能变为10.1.1.2。如果程序连接池中配置的是test.rds.aliyun.com，仍然可以正常访问RDS实例。如果程序连接池中配置的是10.1.1.1，就无法访问RDS实例了。

8.3.1.2 SLB

SLB模块提供实例IP地址（包括内网和外网IP），以便屏蔽物理服务器变化带来的影响。例如：

某RDS实例的内网IP地址为10.1.1.1，对应的Proxy或者DB Engine运行在192.168.0.1上。正常情况下，SLB模块会将访问10.1.1.1的流量重定向到192.168.0.1上。当192.168.0.1发生了故障，处于热备状态的192.168.0.2接替了192.168.0.1的工作。此时SLB模块会将访问10.1.1.1的流量重定向到192.168.0.2上，RDS实例仍旧正常提供服务。

8.3.1.3 Proxy

Proxy模块提供数据路由、流量探测和会话保持等功能，包括以下功能：

- 数据路由：支持大数据场景下的分布式复杂查询聚合和相应的容量管理。
- 流量探测：降低SQL注入的风险，在必要情况下支持SQL日志的回溯。
- 会话保持：解决故障场景下的数据库连接中断问题。

8.3.1.4 DB Engine

RDS全面支持主流的数据库协议，具体情况如下表所示：

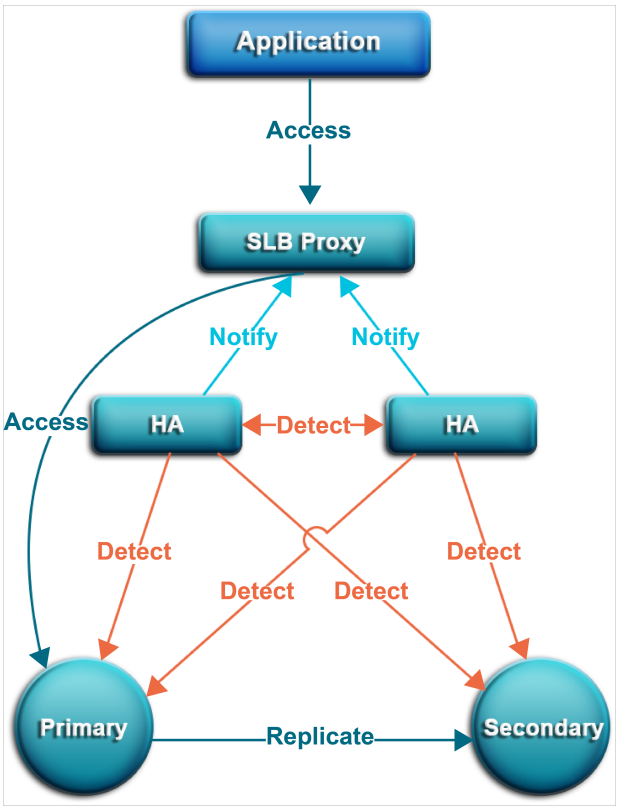
表 8-1: RDS支持的数据库协议

RDBMS	Version
My SQL	5.6（含只读实例）
SQL Server	2008 R2
Postgre SQL	9.4
PPAS	9.3/9.6

8.3.2 高可用服务

高可用服务用于保障数据链路服务的可用性，以及负责处理数据库内部的异常。高可用服务由多个HA 节点提供，本身具有高可用的特点。

图 8-3: RDS 高可用服务



8.3.2.1 Detection

Detection模块负责检测DB Engine的主节点和备节点是否提供了正常的服务。

通过间隔为8-10秒的心跳信息，HA节点可以轻易获得主节点的健康情况，再结合备节点的健康情况和其他HA节点的心跳信息，Detection模块可以排除网络抖动等异常引入的误判风险，在30秒内完成异常切换操作。

8.3.2.2 Repair

Repair模块负责维护DB Engine的主节点和备节点之间的复制关系，同时修复主节点或者备节点在日常运行中出现的错误，例如：

- 主备复制异常断开的自动修复；
- 主备节点表级别损坏的自动修复；
- 主备节点Crash的现场保存和自动修复。

8.3.2.3 Notice

Notice模块负责将主备节点的状态变动通知到SLB或者Proxy，保证您访问正确的节点。例如，

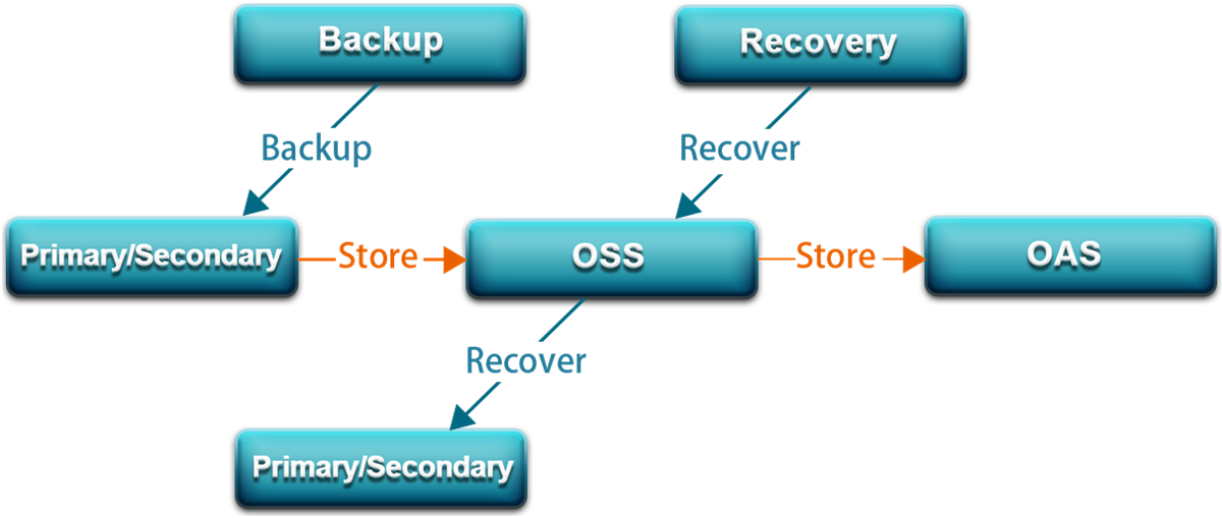
Detection模块发现主节点异常，通知Repair模块进行修复。Repair模块尝试后无法修复主节点，通知Notice进行流量切换。Notice模块将切换请求转发至SLB或者Proxy，此时您的流量全部指向备节点。

与此同时，Repair在别的物理服务器上重建新的备节点，并将变动同步给Detection模块。Detection模块开始重新检测实例的健康状态，并通过。

8.3.3 备份服务

备份服务主要提供数据的离线备份、转储和恢复。

图 8-4: RDS备份服务



8.3.3.1 Backup

Backup模块负责将主备节点上面的数据和日志压缩和上传。RDS默认将备份上传到OSS中，在特定场景下还支持将备份文件转储到更加廉价和持久的归档存储上。在备节点正常运作的情况下，备份总是在备节点上面发起，以避免对主节点提供的服务带来冲击。在备节点不可用或者损坏的情况下，备份模块会通过主节点创建备份。

8.3.3.2 Recovery

Recovery模块负责将OSS上的备份文件恢复到目标节点上，包括以下功能：

- 回滚主节点：您发起数据相关的误操作后可以通过回滚功能按时间点恢复数据。
- 修复备节点：在备节点出现不可修复的故障时自动新建备节点来降低风险。
- 创建只读实例：通过备份来创建只读实例。

8.3.3.3 Storage

Storage模块负责备份文件的上传、转储和下载。

目前备份数据全部上传至OSS进行存储，您可以根据需要获取临时链接来下载。

说明：

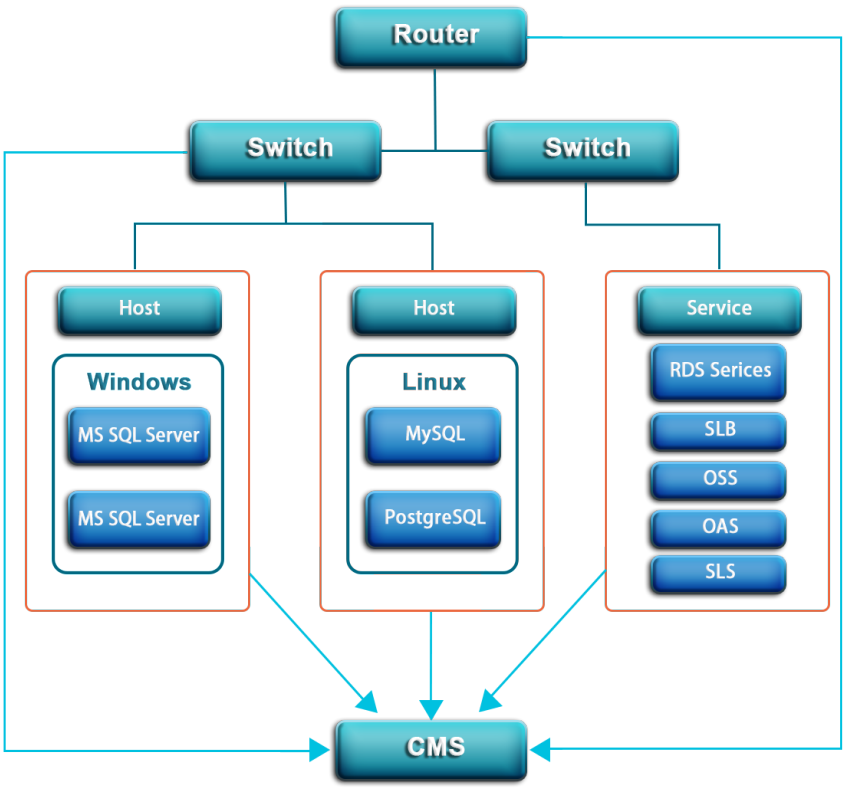
PPAS暂时不提供下载备份文件功能。

在某些特定场景下，Storage模块支持将OSS上面的备份文件转储至归档存储来提供更长时间和更低费用的离线存储。

8.3.4 监控服务

监控服务提供物理层、网络层、应用层等多方位的监控服务，保证业务可用性。

图 8-5: RDS监控服务



8.3.4.1 Service

Service模块负责服务级别的状态跟踪，监控负载均衡、OSS、归档存储和日志服务等RDS依赖的其他云产品是否正常，包括功能和响应时间等。对RDS内部的服务，Service也会通过日志来判定是否正常运作。

8.3.4.2 Network

Network模块负责网络层面的状态跟踪。包括：

- ECS与RDS之间的连通性监控；
- RDS物理机之间的连通性监控；
- 路由器和交换机的丢包率监控。

8.3.4.3 OS

OS模块负责硬件和OS内核层面的状态跟踪。包括：

- 硬件检修：OS模块会不断检测CPU、内存、主板、存储等设备的工作状态，预判是否会发生故障，提前进行自动报修。
- OS内核监控：OS模块会跟踪数据库的所有调用，从内核态分析调用缓慢或者出错的原因。

8.3.4.4 Instance

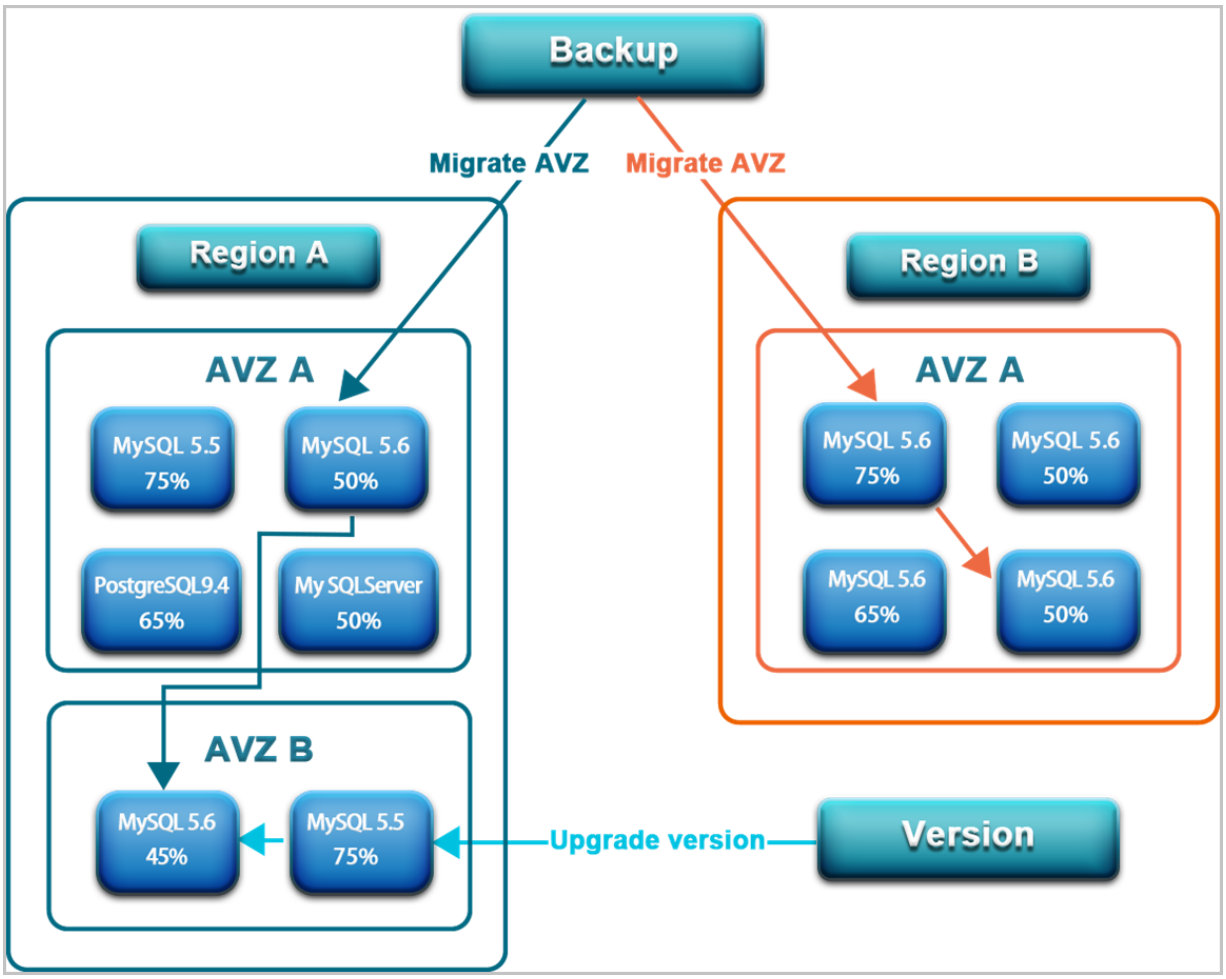
Instance模块负责RDS实例级别的信息采集。包括：

- 实例的可用信息；
- 实例的容量和性能指标；
- 实例的SQL执行记录。

8.3.5 调度服务

调度服务主要提供资源调配和实例版本管理。

图 8-6: RDS调度服务



8.3.5.1 Resource

调度服务由Resource模块完成，主要提供资源调配服务。

Resource模块主要负责RDS底层资源的分配和整合，对您来说就是实例的开通和迁移。例如，

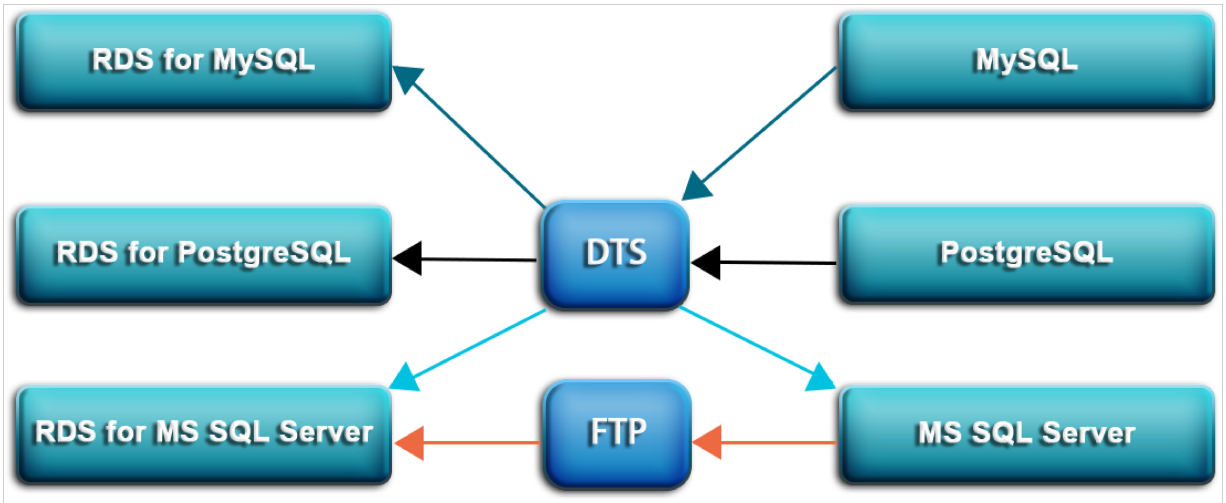
您通过控制台或者Open API创建实例，Resource模块会计算出最适合的物理服务器来承载流量。RDS实例迁移的情况类似。

在经过长时间的实例创建、删除和迁移后，Resource模块会计算资源碎片化程度，并定期发起资源整合提高服务承载量。

8.3.6 迁移服务

迁移服务主要帮助您把数据从自建数据库迁移到RDS上。

图 8-7: RDS 迁移服务



8.3.6.1 FTP

FTP模块主要负责RDS for SQL Server的全量迁移上云。

您在自建的 SQL Server数据库上进行一次备份后，通过FTP客户端将备份文件上传至RDS提供的专用FTP上，FTP模块再将备份文件还原到指定的RDS实例上。

9 云数据库KVStore for Redis

9.1 什么是云数据库KVStore for Redis

阿里云数据库KVStore for Redis（KVStore for Redis）是兼容开源Redis协议的在线Key-Value存储服务。它支持字符串（String）、链表（List）、集合（Set）、有序集合（SortedSet）、哈希表（Hash）等多种数据类型；它还支持事务（Transactions）、消息订阅与发布（Sub/Pub）等高级功能。通过内存加硬盘的存储方式，云数据库KVStore for Redis在提供高速数据读写能力的同时满足数据持久化需求。

除此之外，云数据库KVStore for Redis作为云计算服务，其硬件和数据部署在云端，有完善的基础设施、网络安全保障和系统维护服务。

9.2 功能特性

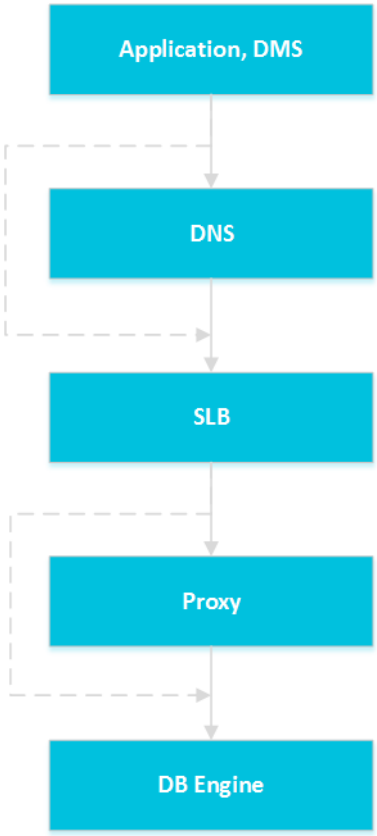
高可用服务Redis主要包括四大核心服务：

- 数据链路服务
- 高可用服务
- 监控服务
- 调度服务

9.2.1 数据链路服务

数据链路服务主要提供数据操作，包括数据的增删改查。

您可以通过应用程序连接Redis服务。



9.2.1.1 DNS

DNS模块提供域名到IP的动态解析功能，以便屏蔽Redis实例IP地址变化带来的影响。

例如，某Redis实例的域名为**test.kvstore.aliyun.com**，而这个域名对应的IP地址为**10.1.1.1**。某程序连接池中配置为**test.kvstore.aliyun.com** 或**10.1.1.1**，都可以正常访问云数据库KVStore for Redis实例。当该云数据库KVStore for Redis实例发生了跨机故障迁移或者版本升级后，IP地址可能变为**10.1.1.2**。如果程序连接池中配置的是**test.kvstore.aliyun.com**，那么还可以正常访问云数据库KVStore for Redis实例；如果程序连接池中配置的是**10.1.1.1**，就无法访问实例了。

9.2.1.2 SLB

SLB模块提供实例IP地址重定向的功能，以便屏蔽物理服务器变化带来的影响。

例如，某云数据库KVStore for Redis实例的内网IP地址为10.1.1.1，对应的Proxy或者DB Engine运行在192.168.0.1上。在正常情况下，SLB模块会将访问10.1.1.1的流量重定向到192.168.0.1上。当192.168.0.1发生了故障，处于热备状态的192.168.0.2接替了192.168.0.1的工作。此时SLB模块会将访问10.1.1.1的流量重定向到192.168.0.2上，Redis实例仍旧正常提供服务。

9.2.1.3 Proxy

Proxy模块提供数据路由、流量探测和会话保持等功能。

- 数据路由功能：Redis支持集群版架构，可进行分布式路由复杂查询和分区策略。
- 流量探测功能：降低利用Redis漏洞进行网络攻击的风险。
- 会话保持功能：解决故障场景下的数据库连接中断问题。

9.2.1.4 DB Engine

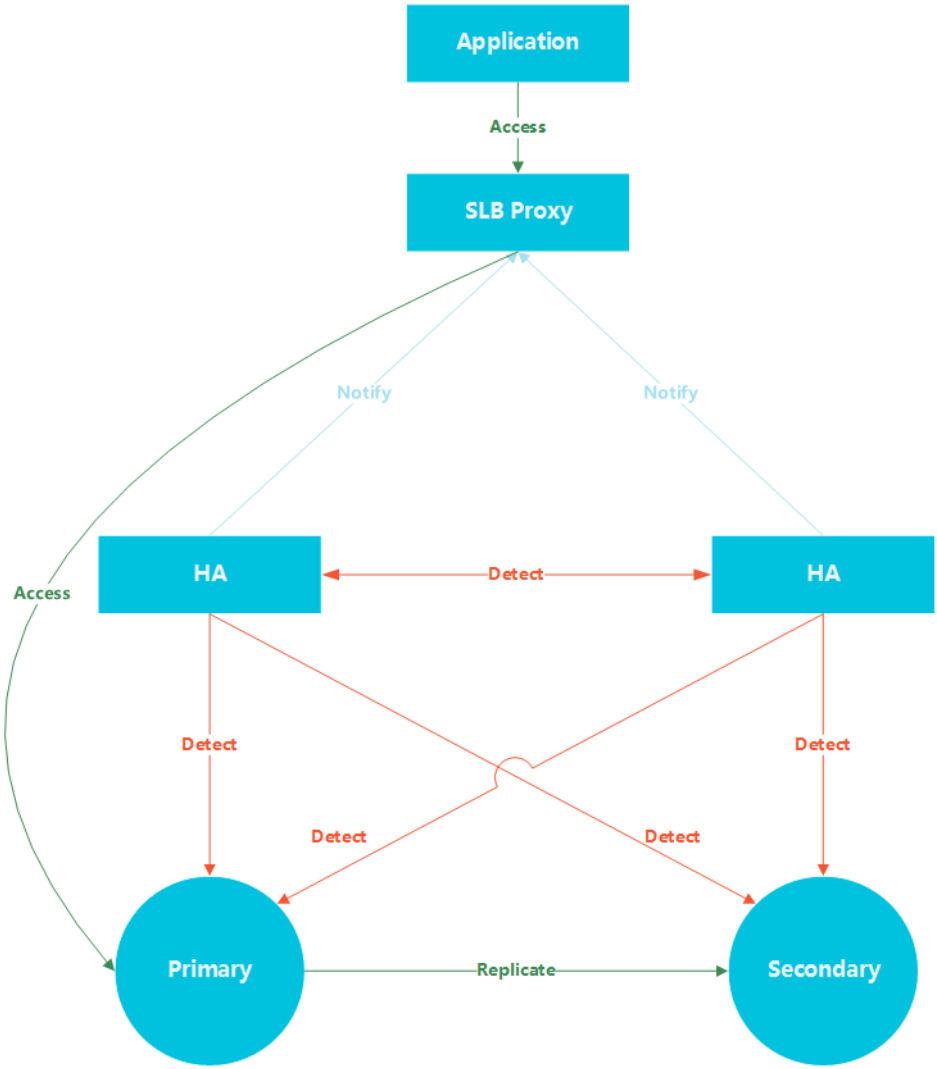
阿里云云数据库KVStore for Redis支持标准协议：

引擎	Version
Redis	兼容 2.8 和 3.0 GEO

9.2.2 高可用服务

高可用服务主要保障数据链路服务的可用性，除此之外还负责处理数据库内部的异常。

另外，高可用服务由多个HA节点提供，本身具有高可用的特点。



9.2.2.1 Detection

Detection模块负责检测DB Engine的主节点和备节点是否正常运行。

通过间隔为8秒～10秒的心跳信息，HA节点可以轻易获得主节点的健康情况。再结合备节点的健康情况和其他HA节点的心跳信息，Detection模块可以排除网络抖动等异常引入的误判风险，在30秒内完成故障转移操作。

9.2.2.2 Repair

Repair模块负责维护DB Engine的主节点和备节点之间的复制功能，还会修复主节点或者备节点在日常运行中出现的错误，如下：

- 主备复制异常断开的自动修复
- 主备节点表级别损坏的自动修复
- 主备节点Crash的现场保存和自动修复

9.2.2.3 Notice

Notice模块负责将主备节点的状态变动通知到SLB或者Proxy，保证用户访问正确的节点。

例如，Detection模块发现主节点异常，并通知Repair模块进行修复。Repair模块进行了尝试后无法修复主节点，就通知Notice模块进行流量切换。Notice模块将切换请求转发至SLB或者Proxy，此时用户流量全部指向备节点。与此同时，Repair在其他物理服务器上重建了新的备节点，并将变动同步给Detection模块。Detection模块开始重新检测实例的健康状态，并通过检测。

9.2.3 监控服务

监控服务主要提供服务、网络、操作系统和实例层面的状态跟踪。

9.2.3.1 服务层面监控

单独的Service模块负责服务层面的监控。Service模块会监控云数据库KVStore for Redis所依赖的SLB等其他云产品是否正常运行，包括功能和响应时间等。

9.2.3.2 网络层面监控

网络层面的Network模块负责网络层面的状态跟踪，监控的内容包括：

- ECS与云数据库KVStore for Redis之间的连通性监控。
- 云数据库KVStore for Redis物理机之间的连通性监控。
- 路由器和交换机的丢包率监控。

9.2.3.3 操作系统层面监控

操作系统OS模块负责硬件和OS内核层面的状态跟踪。监控的内容包括：

- 硬件检修：OS模块会不断检测CPU、内存、主板、存储等设备的工作状态，并预判是否会发生故障，并提前进行自动报修。
- OS内核监控：OS模块会跟踪数据库的所有调用，并从内核状态分析系统调用缓慢或者出错的原因。

9.2.3.4 实例层面监控

实例层面监控的Instance模块负责云数据库KVStore for Redis实例级别的信息采集。监控的内容包括：

- 实例的可用信息。
- 实例的容量。

9.2.4 调度服务

调度服务主要提供资源调配，主要负责云数据库KVStore for Redis底层资源的分配和整合，对用户而言就是实例的开通和迁移。

例如，用户通过控制台创建实例，调度服务会计算出最合适的物理服务器来承载流量。

在经过长时间的实例创建、删除和迁移后，机房会产生资源碎片。调度服务会计算机房内的资源碎片化程度，并定期发起资源整合以提高机房的服务承载量。

10 云数据库MongoDB版

10.1 什么是云数据库MongoDB

云数据库MongoDB版完全兼容MongoDB协议，提供稳定可靠、弹性伸缩的数据库服务。为您提供容灾、备份、恢复、监控、报警等方面的全套数据库解决方案。

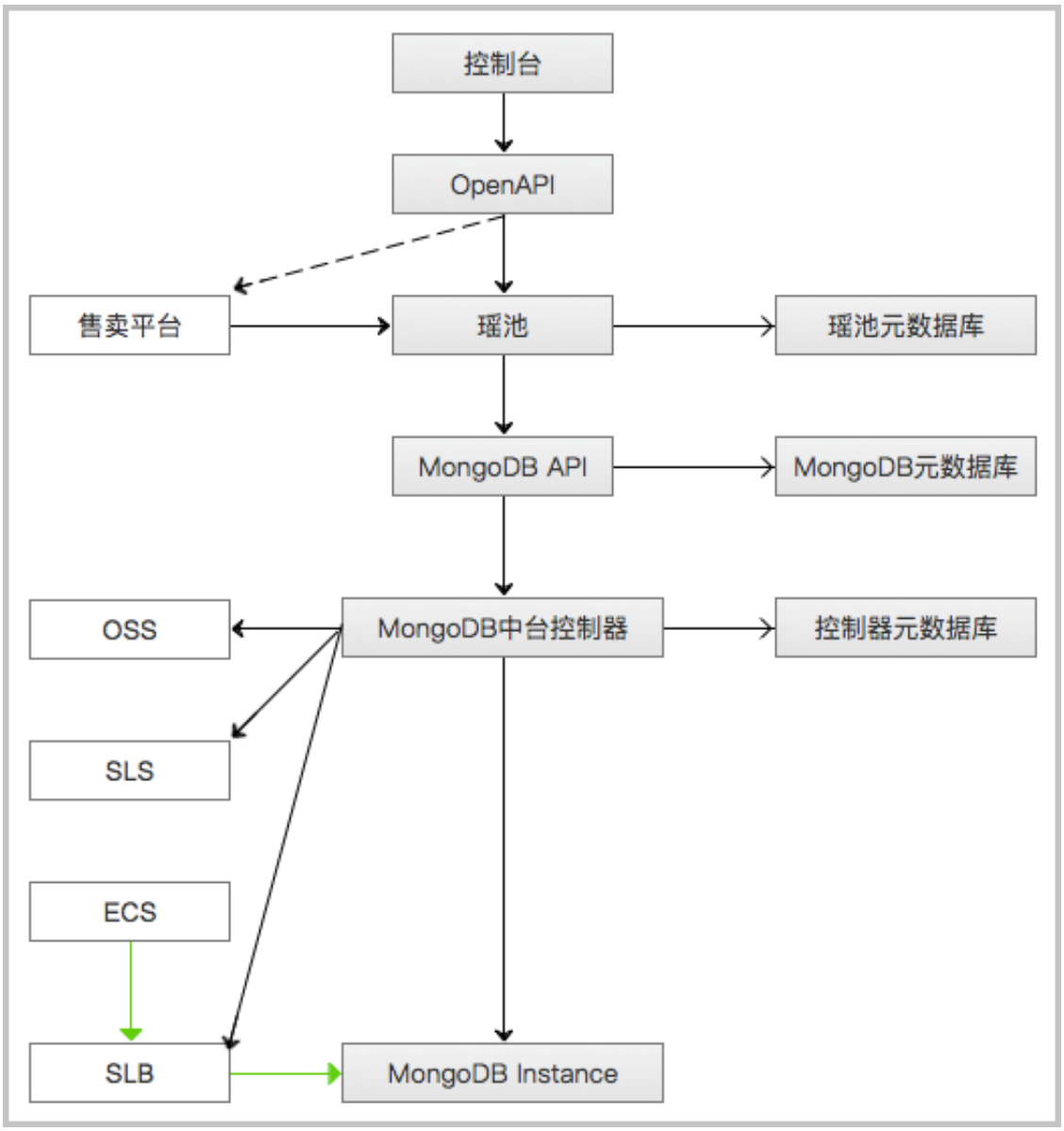
云数据库MongoDB版默认部署三节点副本集架构，主节点（Primary）支持读写访问，其中一个备节点（Secondary）提供日常只读操作，另外一个备节点（Hidden）用于高可用保障。

云数据MongoDB版从多个角度提供方案来保障服务安全可靠，包括但不限于：

- 访问控制：数据库账号密码、IP白名单；
- 网络隔离；
- 数据备份；
- 版本维护；
- 服务授权。

10.2 系统架构

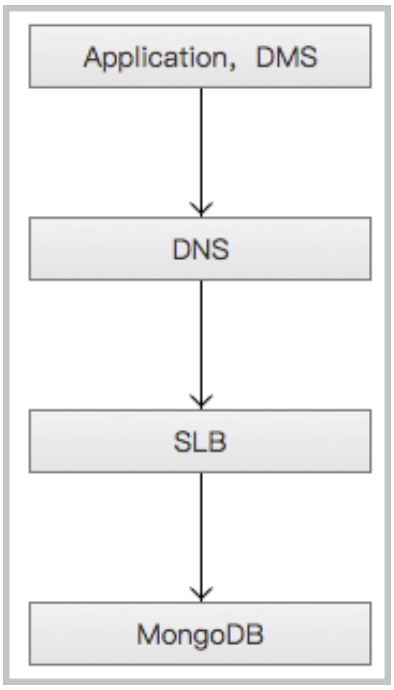
云数据库MongoDB主要包含以下六大核心服务: 数据链路服务、调度服务、备份服务、高可用服务、监控服务、迁移服务。



10.3 功能模块

10.3.1 数据链路服务

数据链路服务主要提供数据操作支持。



DNS

例如，某个MongoDB实例的某个节点域名为mongodb.aliyun.com，而这个域名对应的 IP 地址为10.1.1.1。 某程序连接池中配置为mongodb.aliyun.com 或 10.1.1.1，都可以正常访问MONGODB实例。

当该MongoDB实例发生了规格升级或者实例迁移后，IP地址就可能变为10.1.1.2。如 果程序连接池中配置的是mongodb.aliyun.com，仍然可以正常访问MongoDB实例。如果程 序连接池中配置的是10.1.1.1，就无法访问MongoDB实例了。

SLB

SLB模块提供实例IP地址(包括内网和外网 IP)，以便屏蔽物理服务器变化带来的影响。

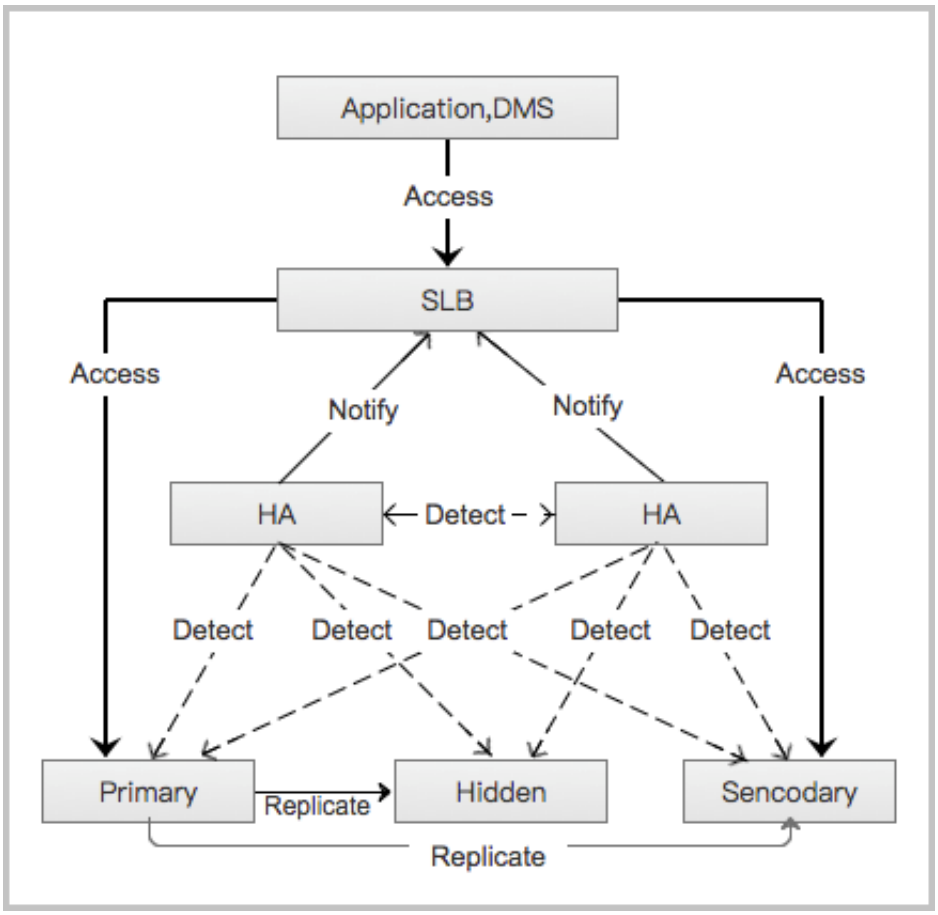
例如，某个MongoDB实例的某个节点的内网IP地址为10.1.1.1，对应的 MongoDB实例运行在192.168.0.1 上。在正常情况下，SLB模块会将访问10.1.1.1的流量重定向到192.168.0.1上。

当 192.168.0.1发生了故障，处于热备状态的192.168.0.2 接替了192.168.0.1 的工 作。此时SLB模块会将访问10.1.1.1.的流量重定向到192.168.0.2 上，MONGODB实例仍旧正常提供服务。

10.3.2 高可用服务

高可用服务主要保障数据链路服务的可用性，除此之外还负责处理数据库内部的异常。

另外，高可用服务由多个HA节点提供，本身具有高可用的特点。



Detection

Detection 模块负责检测 MongoDB的Primary节点、Secondary节点和Hidden节点是否提供了正常的服务。通过间隔为8-10秒的心跳信息，HA节点可以轻易获得Primary节点的健康情况。再结合Secondary节点和Hidden节点的心跳信息，Detection模块可以排除网络抖动等异常 引入的误判风险，在30秒内完成异常切换操作。

Repair

Repair模块负责维护MongoDB 的Primary节点和Secondary节点、Hidden节点之间的复制关系，并负责修复或重建产生错误的节点。

Notice

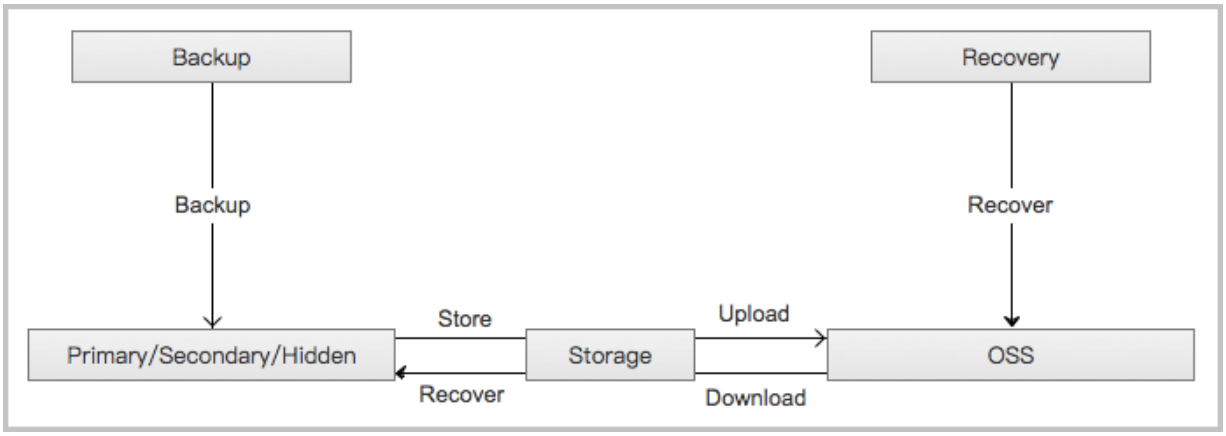
Notice模块负责将节点的状态变动通知到SLB ，保证您访问正确的 节点。

例如，Detection模块发现Primary节点异常，通知Notice进行流量切换。Notice模块将切换请求转发至SLB，此时您之前指向Primary节点的流量将全部指向Secondary节点，Secondary节点转变为Primayr节点；之前指向Secondary节点的流量将全部指向Hidden节点，Hidden节点转变为Secondary节点。

与此同时，Repair尝试将原Primary节点修复为新的Hidden节点，若修复无效则在别的物理服务器上重建新的Hidden节点，并将变动同步给Detection模块，Detection模块开始重新检测实例的健康状态。

10.3.3 备份服务

备份服务主要提供数据的离线备份、转储和恢复。



Backup

Backup模块负责将实例上面的数据和日志备份、压缩并上传到OSS中。MongoDB的备份总是在Hidden节点上面发起，以避免对Primary和Secondary节点的服务带来冲击。

Recovery

Recovery模块负责将OSS中的备份文件恢复到目标节点上。

回滚Primary节点功能：您发起数据相关的误操作后可以通过回滚功能按时间点恢复数据。

修复Secondary\Hidden节点功能：在备节点出现不可修复的故障时自动新建备节点来降低风险。

Storage

Storage模块负责备份文件的上传、转储和下载。目前备份数据全部上传至OSS进行存储，您可以根据需要获取临时链接来下载。

10.3.4 监控服务

监控服务主要提供服务、网络、操作系统和实例层面的状态跟踪。

Service

Service模块负责服务级别的状态跟踪。例如，Service模块会监控SLB、OSS和SLS等MongoDB 依赖的其他云产品是否正常，包括功能和响应时间等。另外对MongoDB内部的服务，Service也会通过日志来判定是否正常运作。

Network

Network模块负责网络层面的状态跟踪。 例如，ECS与MongoDB之间的连通性监控，MongoDB物理机之间的连通性监控，路由器和交换机的丢包率监控。

OS

OS模块负责硬件和OS内核层面的状态跟踪。

例如：

- 硬件检修：OS模块会不断检测CPU、内存、主板、存储等设备的工作状态，并预判是否会发生故障，并提前进行自动报修。
- OS内核监控：OS模块会跟踪数据库的所有调用，并从内核态分析调用缓慢或者出错的原因。

Instance

Instance模块负责：

- MongoDB实例级别的信息采集；
- 实例的可用性信息；
- 实例的容量和性能指标；
- 实例的语句执行记录。

10.3.5 调度服务

调度服务主要提供资源调配和实例版本管理。

Resource

Resource模块主要负责MongoDB底层资源的分配和整合，对您而言就是实例的开通和变更配置。

例如，您通过控制台或者Open API创建实例，Resource模块会计算出最适合的物理服务器来承载流量。在经过长时间的实例创建、删除和迁移后，Resource模块会计算可用区内的资源碎片化程度，并定期发起资源整合以提高可用区的服务承载量。

Version

Version模块主要负责MongoDB实例的版本升级。例如，MongoDB大版本升级，MongoDB 3.2升级至3.4。MongoDB小版本升级，MongoDB源码存在的bug或MongoDB内核定制化优化。

10.3.6 迁移服务

迁移服务主要帮助您把数据从自建数据库迁移到MongoDB中。

数据传输（Data Transmission）服务DTS是阿里云提供的一种数据源之间数据交互的数据流服务，当前支持对MongoDB数据的全量和增量迁移。

- 全量数据迁移：数据传输DTS将源数据库迁移对象的存量数据全部迁移到目标实例。
- 增量数据迁移：增量数据迁移将迁移过程中，本地MongoDB实例的增量更新数据同步到云数据库MongoDB，最终本地MongoDB同云数据库MongoDB进入动态数据同步的过程。使用增量数据迁移，可以实现在本地MongoDB正常提供服务的时候，平滑完成本地MongoDB到云数据库MongoDB的数据迁移。

11 云数据库KVStore for Memcache

11.1 产品概述

云数据库KVStore for Memcache（KVStore for Memcache）是一种高性能、高可靠、可平滑扩容的分布式内存数据库服务。云数据库KVStore for Memcache基于飞天分布式系统及高性能存储进行开发，提供了双机热备、故障恢复、业务监控、数据迁移等功能。

除此之外，云数据库KVStore for Memcache作为云数据库缓存服务，其硬件和数据部署在云端，有完善的基础设施、网络安全保障和统维护服务。

11.2 功能特性

Memcache主要包括四大核心服务：

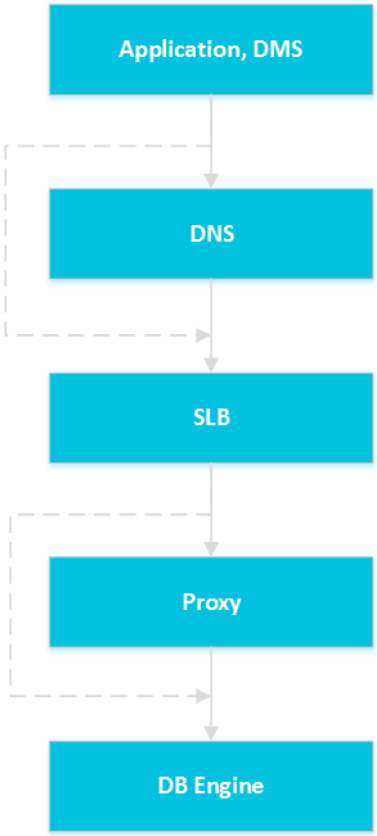
- 数据链路服务
- 高可用服务
- 监控服务
- 调度服务

11.2.1 数据链路服务

数据链路服务主要提供数据操作，包括数据的增删改查。

您可以通过应用程序连接Memcache服务，也可以通过阿里云提供的数据管理工具（DMS）进行图形化数据管理。

图 11-1: 数据链路服务



11.2.1.1 DNS

DNS模块提供域名到IP的动态解析功能，以便屏蔽Memcache实例IP地址变化带来的影响。

例如，某个Memcache实例的域名为**test.kvstore.aliyun.com**，而这个域名对应的IP地址为10.1.1.1。某程序连接池中配置为**test.kvstore.aliyun.com** 或10.1.1.1，都可以正常访问云数据库KVStore for Memcache实例。当该云数据库KVStore for Memcache实例发生了跨机故障迁移或者版本升级后，IP地址可能变为10.1.1.2。如果程序连接池中配置的是**test.kvstore.aliyun.com**，那么还可以正常访问云数据库KVStore for Memcache实例；如果程序连接池中配置的是10.1.1.1，就无法访问实例了。

11.2.1.2 SLB

SLB模块提供实例IP地址，以便屏蔽物理服务器变化带来的影响。

例如，某云数据库KVStore for Memcache实例的内网IP地址为10.1.1.1，对应的Proxy或者DB Engine运行在192.168.0.1上。在正常情况下，SLB模块会将访问10.1.1.1的流量重定向到192.168.0.1上。

当192.168.0.1发生了故障，处于热备状态的192.168.0.2接替了192.168.0.1的工作。此时SLB模块会将访问10.1.1.1的流量重定向到192.168.0.2上，Memcache实例仍旧正常提供服务。

11.2.1.3 Proxy

Proxy模块提供数据路由、流量探测和会话保持等功能。

- 数据路由功能：Memcache支持集群版架构，可进行分布式路由复杂查询，分区策略。
- 流量探测功能：降低利用Memcache漏洞进行网络攻击的风险。
- 会话保持功能：解决故障场景下的数据库连接中断问题。

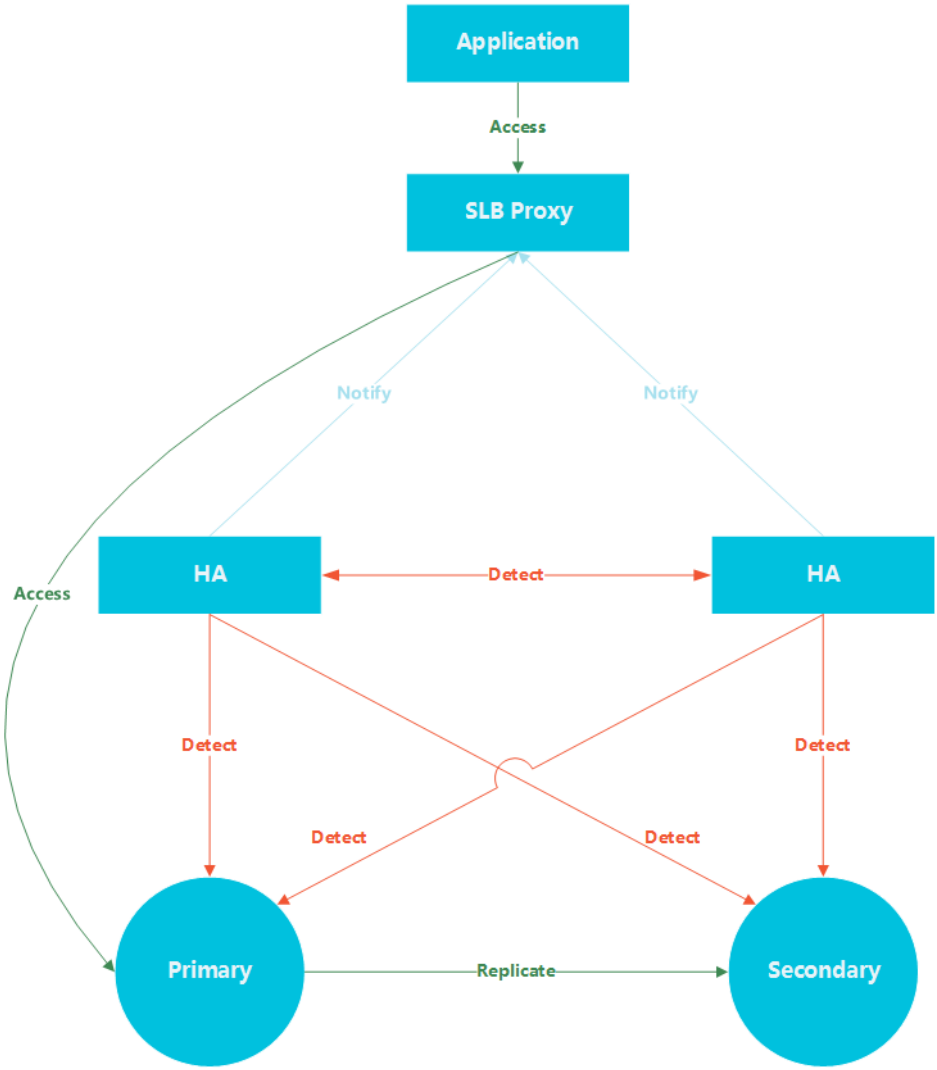
11.2.1.4 DB Engine

云数据库KVStore for Memcache支持标准的Memcached文本协议和二进制协议，支持各种客户端直接连接。

11.2.2 高可用服务

高可用服务主要保障数据链路服务的可用性，除此之外还负责处理数据库内部的异常。 另外，高可用服务由多个HA节点提供，本身具有高可用的特点。

图 11-2: 高可用服务



11.2.2.1 Detection

Detection模块负责检测DB Engine的主节点和备节点是否正常运行。

通过间隔为8秒至10秒的心跳信息，HA节点可以轻易获得主节点的健康情况。再结合备节点的健康情况和其他HA节点的心跳信息，Detection模块可以排除网络抖动等异常引入的误判风险，在30秒内完成异常切换操作。

11.2.2.2 Repair

Repair模块负责维护DB Engine的主节点和备节点之间的复制功能，还会修复主节点或者备节点在日常运行中出现的错误。

- 主备复制异常断开的自动修复
- 主备节点表级别损坏的自动修复

- 保存主备节点Crash的现场，并自动进行修复

11.2.2.3 Notice

Notice模块负责将主备节点的状态变动通知到SLB或者Proxy，保证用户访问正确的节点。

例如，Detection模块发现主节点异常，并通知Repair模块进行修复。Repair模块进行了尝试后无法修复主节点，通知Notice进行流量切换。Notice模块将切换请求转发至SLB或者Proxy，此时用户流量全部指向备节点。与此同时，Repair在别的物理服务器上重建了新的备节点，并将变动同步给Detection模块。Detection模块开始重新检测实例的健康状态并通过。

11.2.3 监控服务

监控服务主要提供服务、网络、操作系统和实例层面的状态跟踪。

11.2.3.1 服务层面监控

单独的Service模块负责服务层面的监控。例如，Service模块会监控Mencache所依赖的SLB等相关云产品是否正常运行。

11.2.3.2 网络层面监控

网络层面的Network模块负责网络层面的状态跟踪。

例如，ECS与云数据库KVStore for Memcache之间的连通性监控、云数据库KVStore for Memcache物理机之间的连通性监控、路由器和交换机的丢包率监控。

11.2.3.3 操作系统层面监控

操作系统OS模块负责硬件和OS内核层面的状态跟踪。监控的内容包括：

- 硬件检修：OS模块会不断检测CPU、内存、主板、存储等设备的工作状态，并预判是否会发生故障，并提前进行自动报修。
- OS内核监控：OS模块会跟踪数据库的所有调用，并从内核状态分析调用缓慢或者出错的原因。

11.2.3.4 实例层面监控

实例层面监控的Instance模块负责云数据库KVStore for Memcache实例级别的信息采集。例如，实例可用信息和实例容量等信息的采集。

11.2.4 调度服务

调度服务主要提供资源调配，主要负责云数据库KVStore for Memcache底层资源的分配和整合，对用户而言就是实例的开通和迁移。

例如，用户通过控制台创建实例，调度服务会计算出最合适的物理服务器来承载流量。

在经过长时间的实例创建、删除和迁移后，机房会产生资源碎片。调度服务会计算机房内的资源碎片化程度，并定期发起资源整合以提高机房的服务承载量。

12 HybridDB for MySQL

12.1 概述

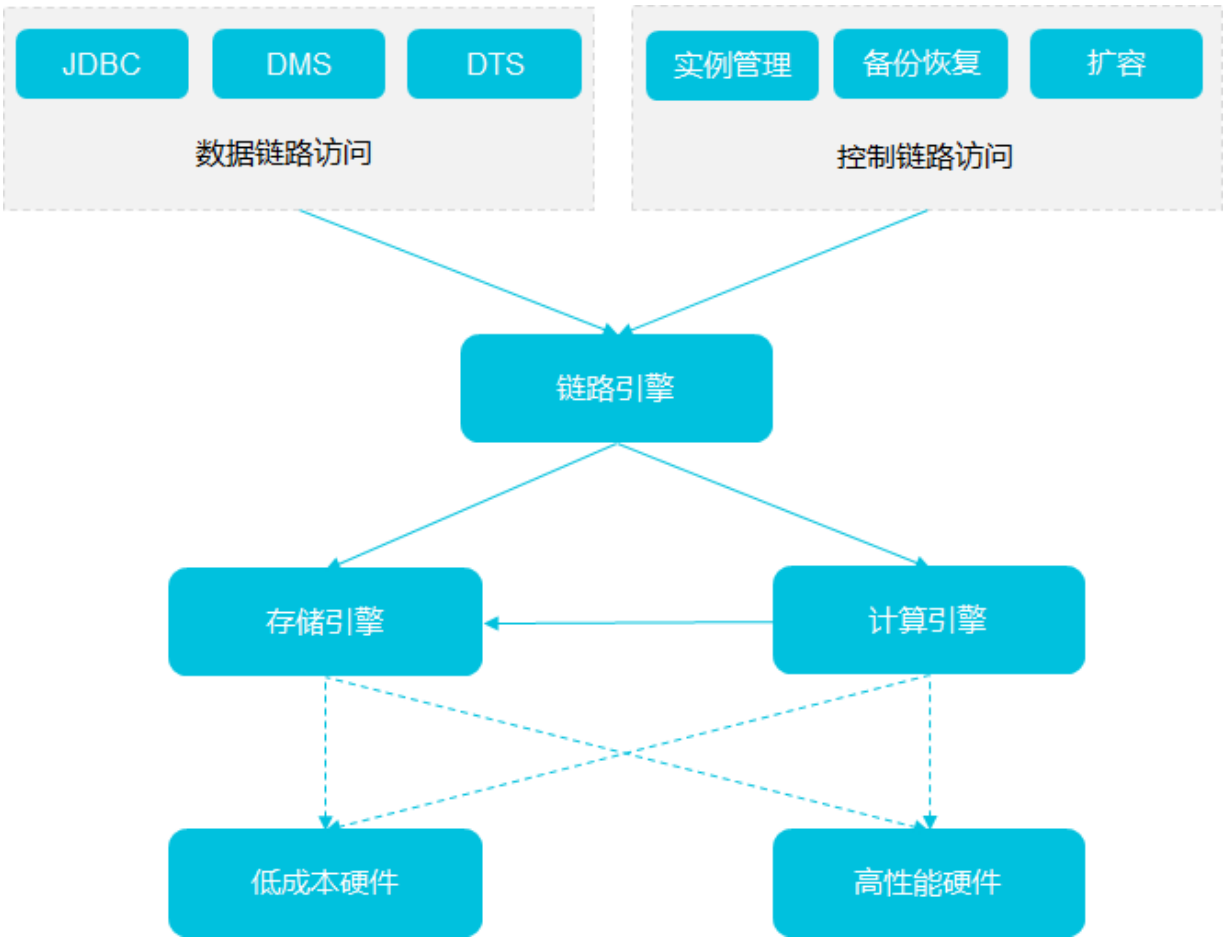
云数据库HybridDB for MySQL （原名PetaData）是阿里云自研的关系型HTAP型数据库产品。可在海量数据上同时支持在线事务（OLTP）和在线分析（OLAP）。HTAP（Hybrid Transaction/Analytical Processing）意为将数据的事务处理（TP）与分析（AP）混合处理，从而实现对数据的实时处理分析。

HybridDB for MySQL采用一份数据存储来进行OLTP和OLAP处理，解决了以往需要把一份数据进行多次复制来分别进行业务交易和数据分析的问题，极大地降低了数据存储的成本。因为采用一份数据，HybridDB for MySQL免去了以往在线数据库（Operational Database）和数据仓库（Data Warehouse）之间的海量数据加载过程，极大地缩短了数据分析的延迟，使得实时分析决策系统成为可能。

12.2 技术架构

阿里云HybridDB for MySQL的主要设计思想就是链路、计算和存储分离，实现松耦合分布式架构的HTAP数据库云服务，其核心技术架构如图 *12-1:* 技术架构所示。

图 12-1: 技术架构



数据链路访问

数据链路访问即应用程序访问数据库的路径，与RDS数据链路访问一致。提供了基于SSL（即安全套接层）协议的可信传输，以确保用户与数据库服务器间的安全数据交换。

HybridDB for MySQL完全兼容MySQL协议、语法，完全继承MySQL生态下的应用和工具，用户无需改变使用习惯，采用全自研的链路存储计算分离架构，可以满足不同业务规模的企业级应用需求，并与之共同成长。

用户可通过DMS对数据库进行管理和操作，可通过DTS完成数据上云、迁移、和增量订阅。

控制链路访问

控制链路访问层依赖RDS的管控功能，即包括调度系统、监控系统、访问控制系统等，以及HybridDB for MySQL自身的实例管理、备份恢复、系统扩容等。

实例管理即对HybridDB for MySQL数据库进行日常管理，包括新建用户、容量监控等。

链路引擎

链路引擎是HybridDB for MySQL的创新之一。用户来自数据链路层的SQL经过链路引擎解析、优化之后，生成相应的执行计划：简单查询场景例如简单聚合的查询场景直接将计算推至存储节点执行，若查询较复杂则直接由计算引擎生成相关计划树并执行。同时将数据按照内建规律进行分库分表操作，对用户屏蔽。

链路引擎的功能如下：

- SQL解析和分库分表。
- 连接防闪断，故障切换等高可用功能。
- 分布式环境通过请求级连接池支持高并发访问。
- 通过事务协调器和存储节点的二阶段提交实现事务一致。
- 全局唯一ID生成者。

计算引擎

HybridDB for MySQL的计算引擎是自主研发的高性能综合性计算引擎，以SQL为接口，用一套框架统一支持ad-hoc计算、流式计算、图和迭代计算。计算引擎采用算子层加表示层的组合模式，底层采用一套算子模型，可以完备表示出ad-hoc 查询、流式计算，图和机器学习等需求，可以轻松支持各种UDF扩展。

- 全面兼容MySQL全部查询语法和函数。
- 支持Oracle开窗、排名、集合、多维分析等高频分析操作。
- 100%支持TPC-H、TPC-DS大数据工业标准。

存储引擎

存储引擎中存储实际数据库的数据，HybridDB for MySQL的存储引擎除实现数据存储管理、封锁、并发控制、事务管理、缓存管理、空间管理等常规功能外，还提供数据压缩和分区、列存储索引等性能优化功能。

另外，HybridDB for MySQL的存储引擎可插拔，可替换为InnoDB等其他存储引擎。

12.3 产品优势

“事务与分析”一体

HybridDB for MySQL通过技术创新将链路、计算和存储分离架构，系统在保留事务处理能力的同时，增加计算引擎引入强大的OLAP能力，于此同时获得了分布式线性扩展的能力。

在支持事务型场景的同时，引入了自研的计算引擎，实现多节点并行大幅提升计算效率，可在支持事务处理场景的同时支持分析型场景。

因此HybridDB for MySQL可以基于一份数据进行事务（OLTP）与分析（OLAP）混合处理，免去了以往在线数据库（Operational Database）和离线数据仓库（Data Warehouse）之间的海量数据加载过程，避免了数据的多次复制、传输和存储，降低存储成本的同时极大地缩短了数据分析的延迟，使得即席分析决策系统成为可能。

系统可用性与扩展性

HybridDB for MySQL全链路均有高可用设计，链路引擎、计算引擎为无状态设计，存储引擎为一主多备，底层可维护多个数据库节点，一主多备的复制拓扑结构意味着每个节点都是全量的数据。

HybridDB for MySQL本身也支持实时备份，并支持按备份集恢复。结合链路引擎的HA和防闪断功能，最大限度地保证用户的数据可靠性和服务可用性。

HybridDB for MySQL在节点存储不足时需要进行系统扩容。扩容任务会生成一个迁移计划，记录了每个存储分区的迁移方向。全量数据的物理迁移过程，通过在分区备库上生成存储分区的全量文件快照，然后通过文件拷贝传递到目的分区的主备库上，最后进行物理加载。增量数据的多主复制过程是通过在一个目的实例上建立多个复制通道，将多个源实例的数据复制过来。链路引擎进行元数据变更并始终保持与用户的连接，整个升级过程无需断开用户连接，仅会发生闪断。

SQL兼容高易于上手

HybridDB for MySQL完全兼容MySQL的语法和生态，并引入了大量的Oracle聚合函数。兼容主流SQL标准，100%支持TPC-H、TPC-DS，以及Oracle分析函数，无论是MySQL还是Oracle用户，都可以轻松掌握基于HybridDB for MySQL的数据开发。

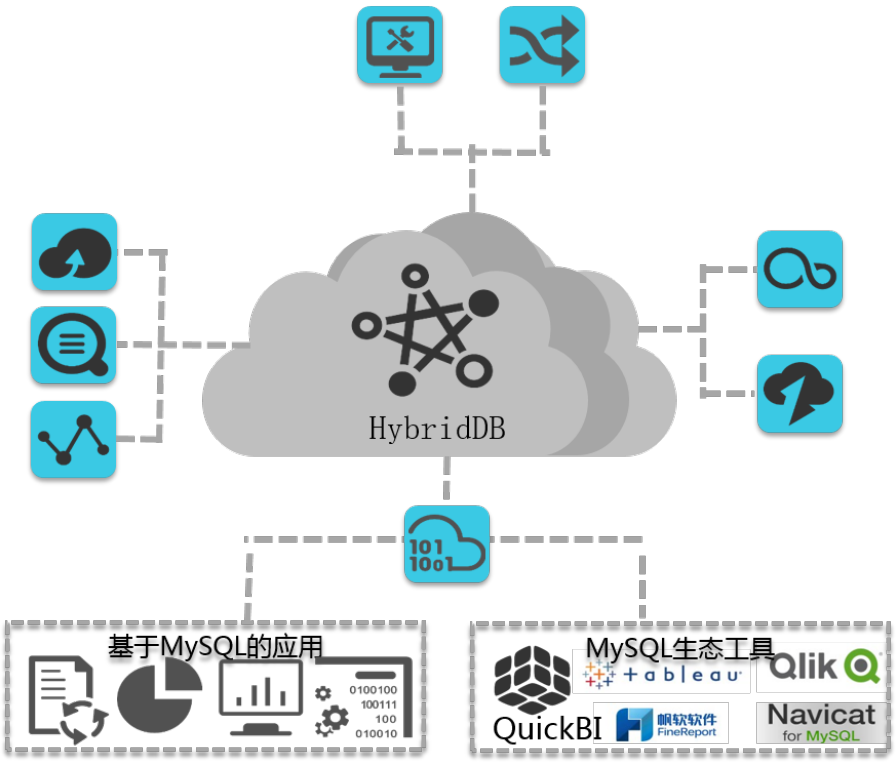
支持原生接口如C/C++、JDBC、ODBC、.NET、Python、PHP、Ruby等。

产品体系化生态完整

HybridDB for MySQL与云监控（CloudMonitor）、数据传输（DTS）、数据集成（Data Integration）、数据管理（DMS）等阿里云产品无缝集成，实现从数据上云到系统监控、数据管理的完整体系。通过数据集成、数据传输等工具打通ADS、MaxCompute等近线/离线产品，以及SLS、OSS等在线存储产品。

由于HybridDB for MySQL完全兼容MySQL语法和生态，支持流行的开发框架如Hibernate、Mybatis、ADO、EF、Zend，RoR等，并在不断完善更新支持列表；支持主流的报表产品如QuickBI、QlikView、Tableau等。

图 12-2: 完整生态



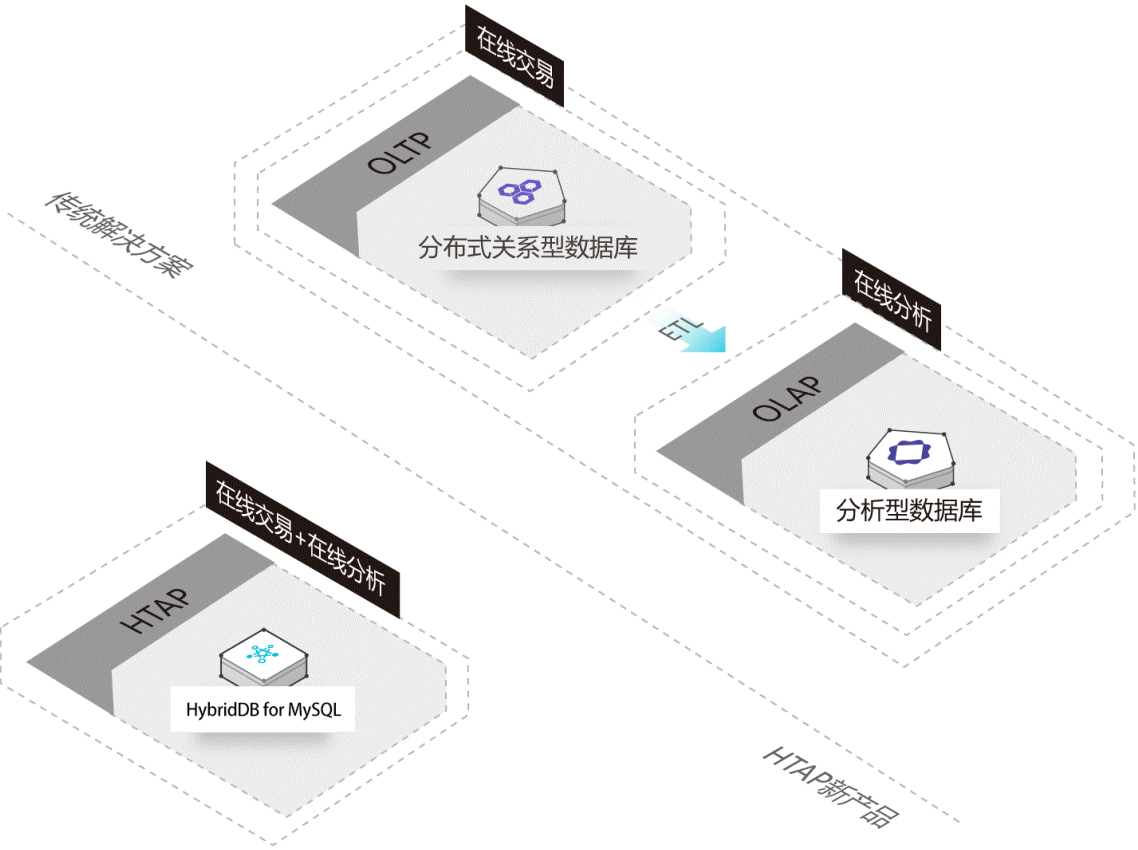
12.4 适用场景

大数据存储与分析

传统的数据分析场景下，用户需要把数据从在线数据库（Operational Database）复制到离线数据仓库（Data Warehouse）之后再进行数据分析，需要对海量数据进行多次复制、传输、加载和存储等多方面工作。

HybridDB for MySQL可以基于一份数据进行事务（OLTP）与分析（OLAP）混合处理，免去了在线数据库和离线数据仓库之间海量数据的复制、传输、加载和存储，降低存储成本的同时极大地缩短了数据分析的延迟，使得实时分析决策系统成为可能。

图 12-3: 大数据存储与分析

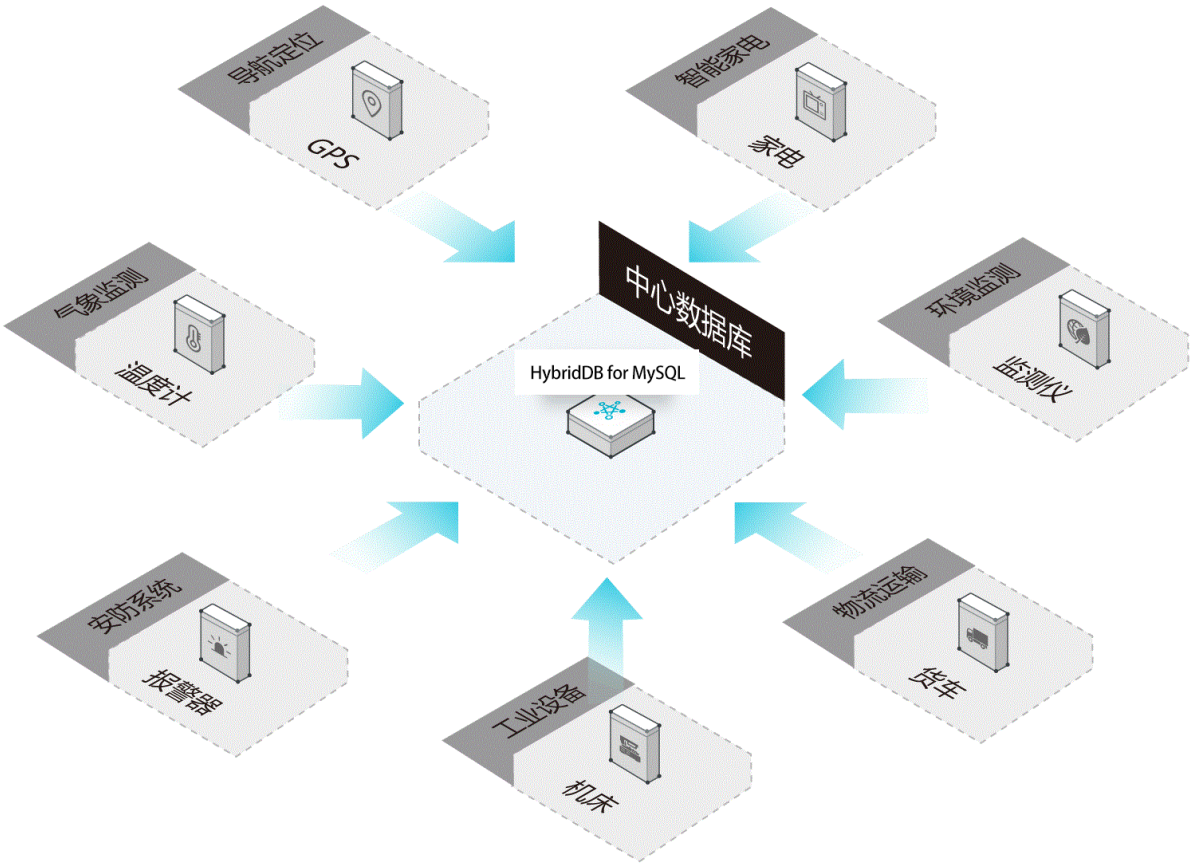


物联网

物联网有着数据采集点多、数据总量大、并发高的应用特点。在基于 MySQL 数据库的系统架构下，存放着设备信息的库表，往往需要使用分库分表和相应的数据分发技术才能承担业务流量。分库分表的设计不仅增加了数据库运维人员的管理难度，也让系统架构的扩展受到了很大的限制。

HybridDB for MySQL的分布式架构屏蔽了分库分表的细节，只对用户提供一个数据库连接地址和相应的逻辑库表，让用户的开发和运维成本降低到最低。当面对业务量暴涨的情况下，用户只要简单地增加存储节点就能将数据拆分到更多的服务器上，而这一切细节都不需要用户去深入了解。

图 12-4: 物联网



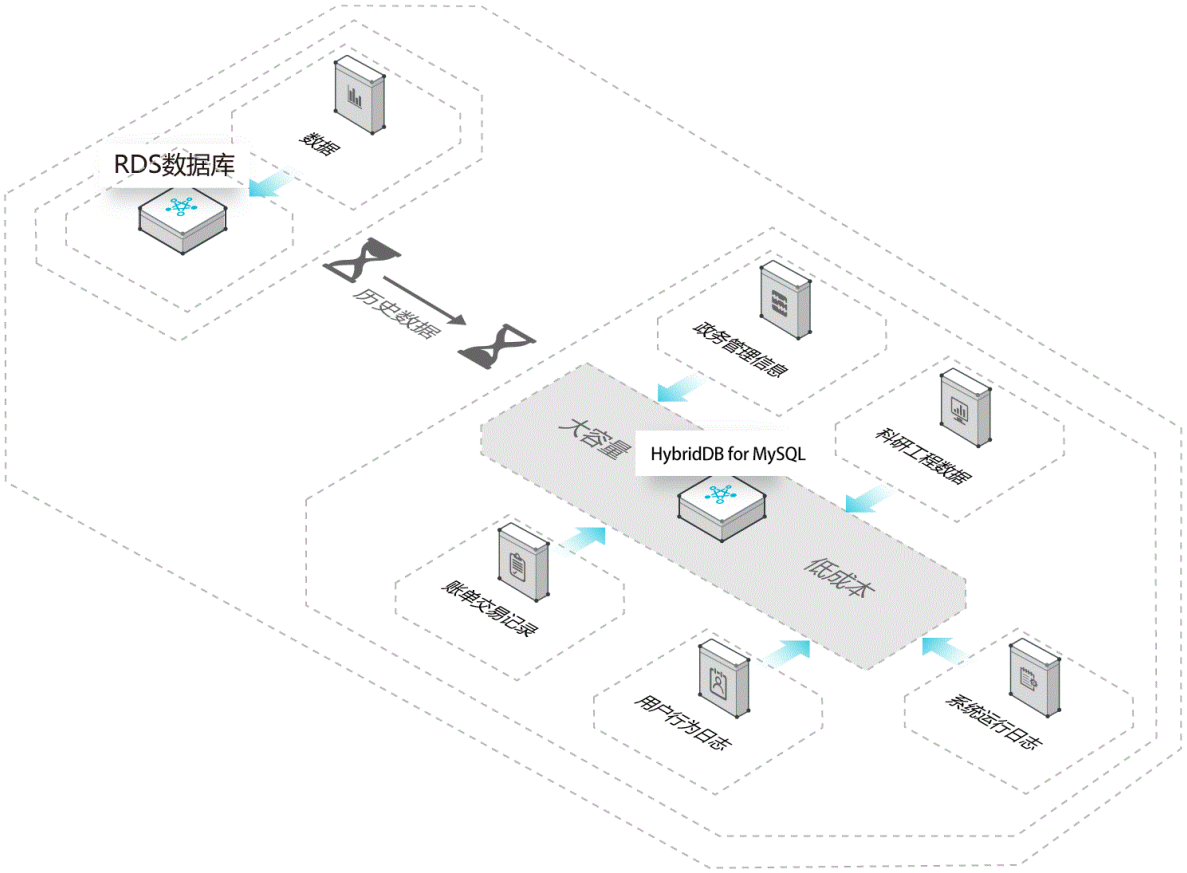
历史日志

很多用户为了保障在线数据库的性能和容量，降低总体数据存储成本，会将线上业务系统早期生成的数据转移到历史库进行保存。同时，这些海量的商业历史数据对于过去业务的分析和未来业务的规划展望又具有非常重要的价值，需要不定期地进行数据分析。

HybridDB for MySQL可存储海量的历史数据（最高可达到PB级），并且可以通过数据压缩来进一步节省存储空间。另有价格低廉的普通HDD硬盘存储供用户选择，极大地减少了数据存储的成本。

同时，HybridDB for MySQL作为新型HTAP数据库，用户可以随时直接对这些历史数据进行多维度的OLAP数据分析，而无需再将这些数据重新导入到商业BI系统中去。

图 12-5: 历史日志


RDS数据库
数据
历史数据
政务管理信息
科研工程数据
大容量
HybridDB for MySQL
低成本
账单交易记录
用户行为日志
系统运行日志

13 HybridDB for PostgreSQL

13.1 什么是HybridDB for PostgreSQL

云数据库 HybridDB for PostgreSQL（ApsaraDB HybridDB for PostgreSQL）是一种在线分布式云数据库，由多个计算组组成，可提供大规模并行处理（MPP）数据仓库服务。

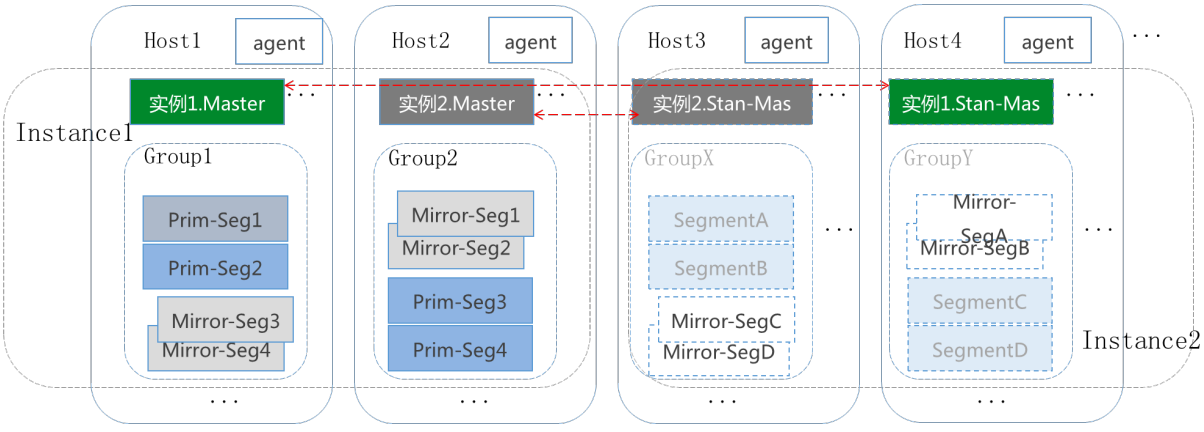
HybridDB for PostgreSQL 基于 Greenplum Database 开源数据库项目开发，由阿里云深度扩展后，具备如下特性：

- 兼容 Greenplum，用户可以直接使用所有支持 Greenplum 的工具。
- 支持 OSS 存储、JSON 数据类型、HyperLogLog 预估分析等功能特性。
- 支持 SQL 2008 标准查询语法及 OLAP 分析聚合函数，能够提供灵活的混合分析能力。
- 支持行存储和列存储混合模式，分析性能优越。
- 支持数据压缩技术，存储成本低廉。
- 提供在线扩容、性能监测等服务，用户无需再进行复杂的大规模 MPP 集群的运维管理工作，使 DBA、开发人员及数据分析师只需专注于如何通过 SQL 提高企业的生产力、创造核心价值。

13.2 系统架构

HybridDB for PostgreSQL的系统架构如下：

图 13-1: 系统架构图



HybridDB for PostgreSQL由两大组件组成：Master节点和Segment节点。

Master节点是HybridDB forPostgreSQL数据库系统的应用接入点，它负责接收客户端的连接和SQL查询请求，并且将工作分配给Segment节点执行。系统同时在另一台物理机上部署Master的备节

点，采用Replication方式同步主备节点以支持故障转移。备节点不能连接Segment节点，也不接受外部链接。

Segment节点是独立的HybridDB for PostgreSQL数据实例节点，每个Segment节点都会存储一部分的数据，并行地执行计算。每个Segment节点均采用Primary/Mirror主备架构，以支持故障转移。

13.3 功能特性

分布式

- **MPP架构**

  基于分布式大规模并行处理，随计算单元的添加线性扩展存储及计算能力，充分发挥每个计算单元的OLAP计算效能。

- **分布式事务**

  支持分布式的SQL OLAP统计及窗口函数，支持分布式PL/pgSQL存储过程、触发器，实现数据库端分布式计算过程开发。

学习分析

- **MADlib机器学习**

  为数据科学用户提供基于SQL的海量数据机器学习工具，支持50多种数据库内置学习算法。

- **GIS地理分析**

  符合国际OpenGIS标准的地理数据混合分析，通过单条SQL即可从海量数据中进行地理信息的分析，如：人口流量、面积统计、行踪等分析。

数据互通

- **异构数据导入**

  MySQL数据库可以通过mysql2pgsql进行高性能数据导入，目前业界流行的ETL工具均支持以HybridDB为目标的ETL数据导入。

- **OSS异构存储**

  可将存储于OSS中的格式化文件作为数据源，通过外部表模式进行实时操作，使用标准SQL语法实现数据查询。

- **透明数据复制**

  支持数据从PostgreSQL/PPAS透明流入，持续增量无需编程处理，简化维护工作，数据入库后可再进行高性能内部数据建模及数据清洗。

安全性

- **IP白名单配置**

  最多支持配置1000个允许连接HybridDB for PostgreSQL实例的服务器IP地址，从访问源进行直接的风险控制。

- **DDoS防护**

  在网络入口实时监测，当发现超大流量攻击时，对源IP进行清洗，清洗无效情况下可以直接拉进黑洞。

功能限制

- Geenplum Database 核心功能的限制参见*Greenplum文档*。
- 权限限制：HybridDB for PostgreSQL 的初始用户（称为“根用户”）有创建数据库（CREATEDB）、创建用户（CREATEROLE）的权限。但如果没有超级用户（SUPERUSER）权限，就无法执行要求超级用户权限的操作，例如，无法执行 pg_ls_dir 等文件操作函数。超级用户有权限查看和修改所有其他非超级用户的数据，终止（Kill）其他非超级用户的连接等。
- 不支持 PL/R 和 PL/Java 插件。
- 支持 PL/Python 插件创建，但不支持使用 PL/Python 语言创建函数。
- 不支持 gpfdist 工具。
- 不支持 MapReduce 接口、gphdfs 存储接口以及本地外部表。
- 暂不支持自动备份和恢复功能。HybridDB 会保存两份数据，用户也可以使用 pg_dump 工具自行备份。

13.4 产品优势

实时分析	支持SQL语法进行分布式GIS地理信息数据类型实时分析，协助物联网、互联网实现LBS位置服务统计。 支持SQL语法进行分布式JSON、XML、模糊字符串等数据实时分析，帮助金融、政企行业实现报文数据处理及模糊文本统计。
稳定可靠	支持分布式ACID数据一致性，实现跨节点事务一致，所有数据双节点同步冗余。 分布式部署，计算单元、服务器、机柜三重防护，提高重要数据基础设施保障。

简单易用	丰富的OLAP SQL语法及函数支持，众多Oracle函数支持，业界流行的BI软件可直接联机使用。 可与云数据库RDS（PostgreSQL/PPAS）实现数据通讯，实现OLTP+OLAP（HTAP）混合事务分析解决方案。
性能卓越	支持行列混合存储。在进行OLAP分析时，列存储的性能比行存储的性能要高得多。 支持高性能OSS并行数据导入，避免单通道导入的性能瓶颈。
灵活扩展	按需等比扩展计算单元，CPU、内存、存储空间，从而提高OLAP性能。 支持透明的OSS数据操作，非在线分析的冷数据可灵活转存到OSS对象存储，数据存储容量无限扩展。

14 数据管理DMS

14.1 什么是数据管理服务

数据管理服务（Data Management Service，简称DMS）提供关系型数据库和OLAP数据库的统一管理。它源自阿里数据库服务平台iDB，为数万研发人员提供数据库研发支撑，已在线上运行8年。您可以使用数据管理DMS轻松构建企业独有的数据库DevOps，促进数据库研发自助化，提升研发效率，同时保证员工数据库访问安全及数据库高性能。

DMS主要用于管理MySQL、SQL Server、PostgreSQL、DRDS等关系型数据库及ADS这种OLAP数据库类型。它是一种集数据管理、结构管理于一体的数据管理服务。

14.2 产品架构

阿里云DMS包含三层结构：业务层、调度层、连接层，用于对关系型数据库的实时数据访问和后台数据任务的调度。

业务层

- DMS业务层为您提供在线图形化的数据库操作，业务层可线性扩展，提升DMS整体服务能力。
- 宕机无状态切换，确保7×24小时服务。

调度层

- 调度层为您提供的调度主要包含：导出、导入、表结构对比。其后台主要通过线程池进行调度，分为实时调度和后台定时调度两类。
- 实时调度提供前端点击后立即调度并一次性任务处理，用户提交任务后无需等待结果，DMS后台自动完成所有工作，结束后用户下载或查看结果。
- 后台定时调度任务为用于定时获取用户指定的数据（如数据趋势），DMS后台将统一定义调度各项任务对各项业务数据进行采集，提供查阅和分析。

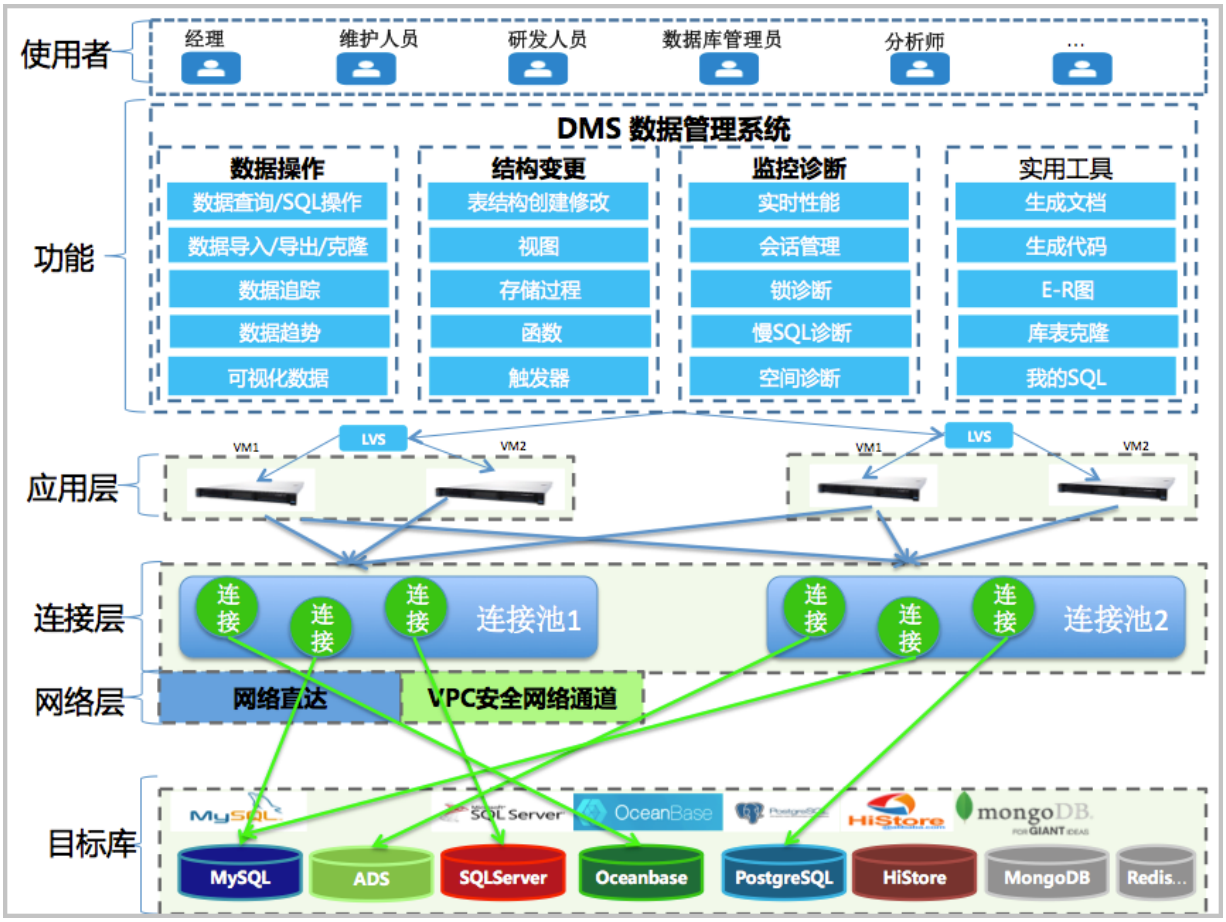
连接层

连接层为DMS的访问数据的核心部件，主要包含如下几点：

- 兼容MySQL、SQLServer、PostgreSQL、DRDS、AnalyticDB的请求。
- 会话隔离及保持，即通过DMS打开的多个SQL窗口的会话相互隔离，并尽可能保持各SQL窗口内的会话状态，接近客户端的体验。
- 实例会话数量控制，防止对单个实例建立大量的连接数。

- 对不同功能采用不同的连接回收策略，确保了功能体验的差异性，同时也减少了数据库的连接数。

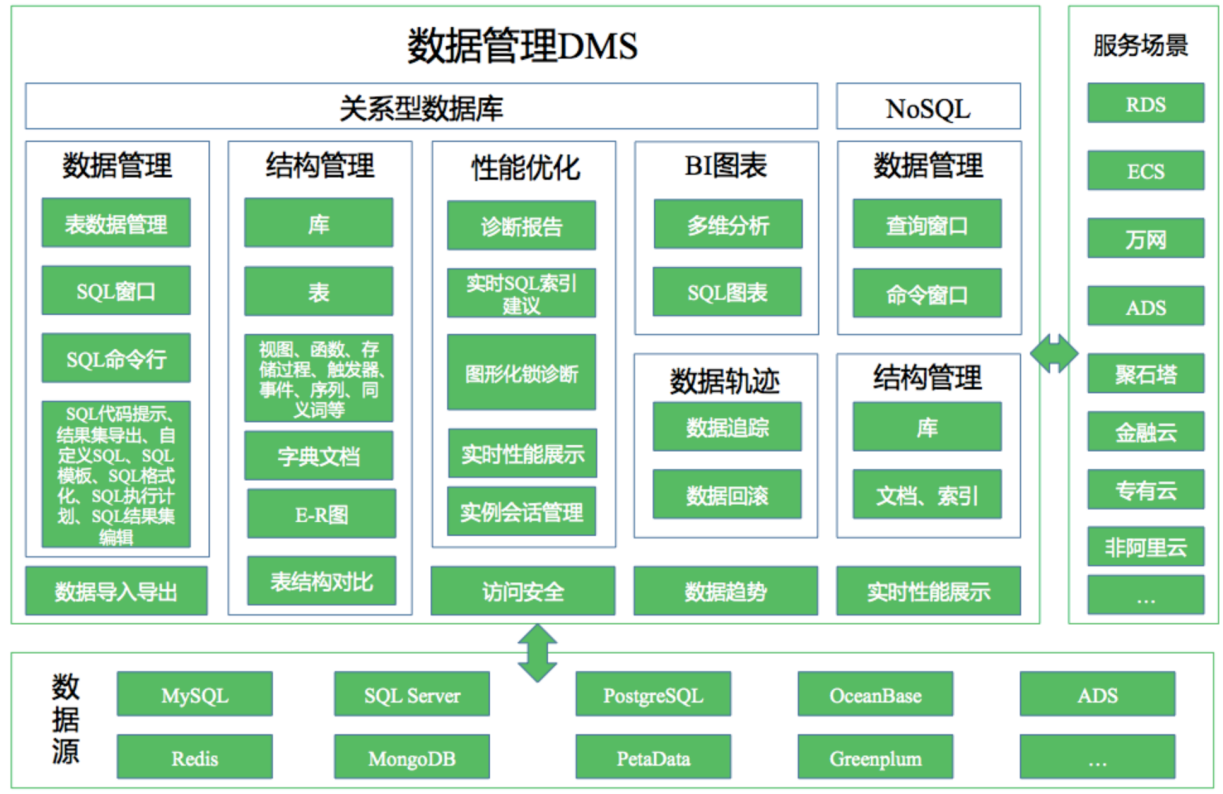
图 14-1: DMS系统架构图



14.3 功能模块

DMS功能如下图所示：

图 14-2: 功能模块图



- **关系型数据库功能**

  # 数据管理：提供SQL窗口、SQL命令行、表数据管理、SQL智能提示、SQL格式化、自定义SQL、SQL模板、SQL执行计划、导入导出等管理功能。

  # 结构管理：提供库、表、视图、函数、存储过程、触发器、事件、序列、同义词等对象管理以及表结构对比功能。

  # 性能优化：提供实时性能展示、实时SQL索引建议、图形化锁管理、实例会话管理、诊断报告等功能。

  # 访问安全：提供4层认证体系，提供登录/操作审计，提供云帐号授权、访问地址授权、功能开关等细粒度授权功能。

- **NoSQL功能**

  # 数据管理：提供查询窗口和命令窗口功能。

  # 结构管理：提供库、文档和索引等对象管理功能。

  # 实时性能展示：提供核心性能指标的实时展示。

14.4 产品优势

极大提升研发效率

- 表结构对比。
- SQL智能提示。
- 自定义SQL/SQL模板的复用。
- 工作环境自动恢复。
- 字典文档导出。

实时优化数据库性能

- 实用的会话管理。
- 核心指标的秒级监控。
- 图形化锁管理。
- SQL索引的实时建议。
- 整体性能的诊断报告。

全面的访问安全保护

- 4层认证体系。
- 细粒度授权。
- 登录/操作审计。

丰富的数据源支持

- 关系型数据库：MySQL、SQL Server、PostgreSQL、PPAS、OceanBase、PetaData等。
- NoSQL：Redis、MongoDB、Memcache等。

14.5 产品价值

数据管理旨在为用户提供一个便捷、易用、安全的数据库访问和管理平台，通过可视化的数据操作服务，让用户通过浏览器来使用他们的数据库，减少了用户安装种类繁杂的数据库客户端的烦恼。通过在线编辑数据，可轻松的完成表数据的操作，表结构的变更，而不需要编写复杂的SQL语句。DMS通过自研的高级功能，为用户提供普通客户端无法完成的功能，如表结构同步、数据库克隆、结果集图表展示、实时监控等。

用户使用DMS时需要先登录Apsara Stack控制台，再使用自己的数据库账号密码登录，通过两层的账号认证，防止数据库账号密码被盗用的情况。DMS支持HTTPS/SSL传输，避免数据在传输过程中被监听或者篡改。

DMS也支持RAM和STS Token做权限验证，避免权限盗用。

DMS支持VPC实例访问，在保证数据库实例网络安全的情况下，为用户提供访问数据的接口，这是普通客户端无法做到的。

在功能上，DMS给用户带来了如下的价值：

- 便捷的数据操作
  - # 现存问题：用户需要通过一款便捷而功能全面的产品来完成SQL操作，保存常用操作，并把常用操作应用到具体的业务中。
  - # 解决方案：通过DMS打开表功能，用户可以通过类似Excel的方式操作表数据，不懂SQL也能对表数据进行增删改查以及统计分析。通过DMS自定义SQL，用户可以保存常用业务SQL，并在管理其他数据库/实例时直接应用这些SQL。
- 库表结构的可视化展示
  - # 现存问题：用户在设计新业务表或梳理已有业务表时，往往要整体了解数据库中所有表的结构。用户可以通过客户端逐一执行SQL命令来展示表结构，但不直观、不方便。
  - # 解决方案：用户通过DMS生成文档功能，可以一键生成整个数据库的表结构，可以在线浏览，也可以导出为Word、Excel、PDF等格式。
- 实时优化数据库性能
  - # 现存问题：数据库的性能优化需要以长时间详细的监控记录为基础，并进行细致分析和异常定位，才能更好地有效优化并提升其性能。
  - # 解决方案：DMS提供了数据库性能秒级监控，包含select/insert/update/delete、活跃连接数以及网络流量等指标，让用户不错过任何性能波动。DMS提供了数据库会话查看和中止功能。
- 绘制SQL结果集的图表
  - # 现存问题：过去，用户通常通过SQL来查出数据，然后再将数据导入到Excel中，制作折线图、饼图等静态图表，过程相对繁琐。
  - # 解决方案：用户可以通过DMS直接基于SQL结果集来绘制图表，还可以制作很多高级图表，如动态图表、环比、个性化Tooltip等，让用户高质量完成工作。
- SQL复用

现存问题：用户访问数据库，几乎都会执行SQL，简单查询SQL比较容易上手，但对于复杂分析或带有业务逻辑的查询，每次重新编写成本太高，而保存到文本中，则需要持续维护对应关系且无法随时随地使用。

解决方案：用户通过DMS的“我的SQL”功能可以将常用SQL保存到DMS，SQL复用不受本地保存的限制，复用范围灵活，无论是当前数据库、当前实例还是全部实例都可以复用。

- 表数据量变化监控

  一谈到数据分析，总离不开大数据，但要发挥大数据分析的真正价值并没有想象的那么容易，DMS倡导：不要等到把数据搞大了才分析。

  数据管理DMS提供表数据量变化监控功能，通过对RDS内核定制，高性能采集每个实例、数据库及表的行数变化，并通过专业的数据分析交互，提供实时监控、历史趋势及明细数据等分析方式

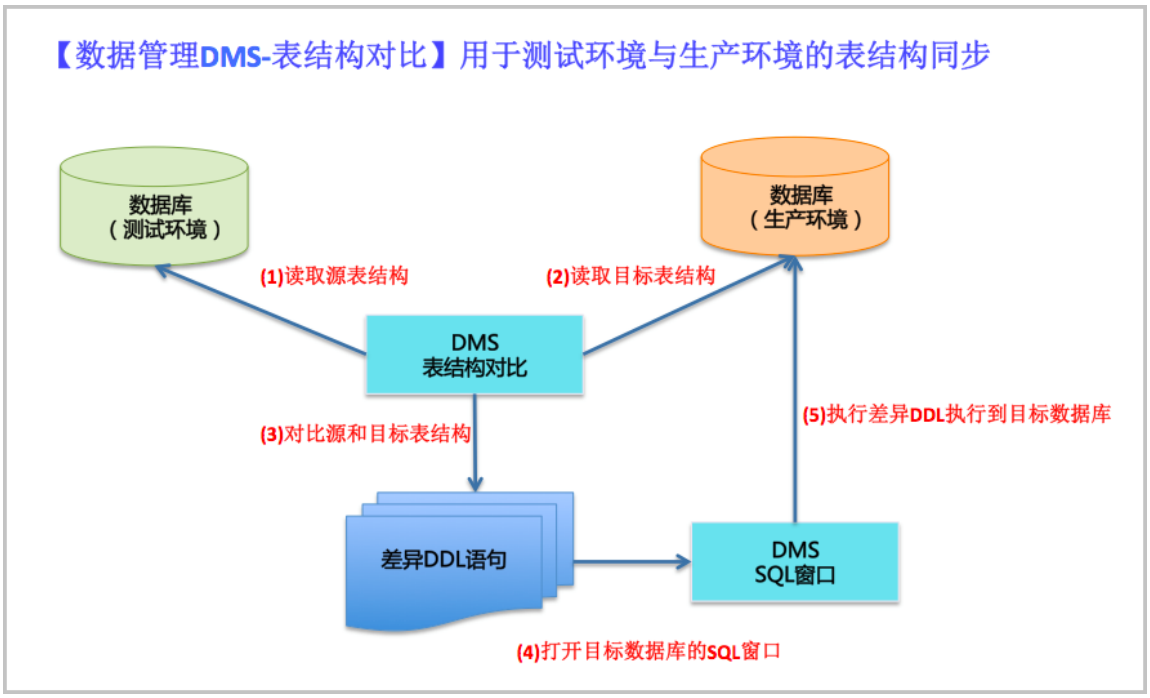
- 表结构同步

  # 现存问题：在企业中，数据库环境都会区分生产环境和测试环境。在测试环境验证后，就会在生产环境发布。一旦测试环境中某些表结构没有同步到生产环境，发布就会出现大故障。

  # 解决方案：企业用户通过DMS的结构对比功能，可以识别出生产环境和测试环境数据库表结构不一致，并得到表结构订正的DDL语句，确保生产环境和测试环境的表结构完全一致。

  关于如何使用表结构对比功能来同步表结构，如图图 *14-3:* 数据管理*DMS*-表结构对比-表结构同步所示。

图 14-3: 数据管理DMS-表结构对比-表结构同步



15 云数据库OceanBase版

15.1 什么是云数据库OceanBase

云数据库OceanBase是阿里巴巴/蚂蚁金服集团独立自主研发的一款分布式关系数据库产品，融合传统关系数据库和分布式系统的优势，具备高可用、高性能和高可扩展性，在功能上兼容MySQL等特点，在通用硬件上提供金融级高可用的数据库服务。

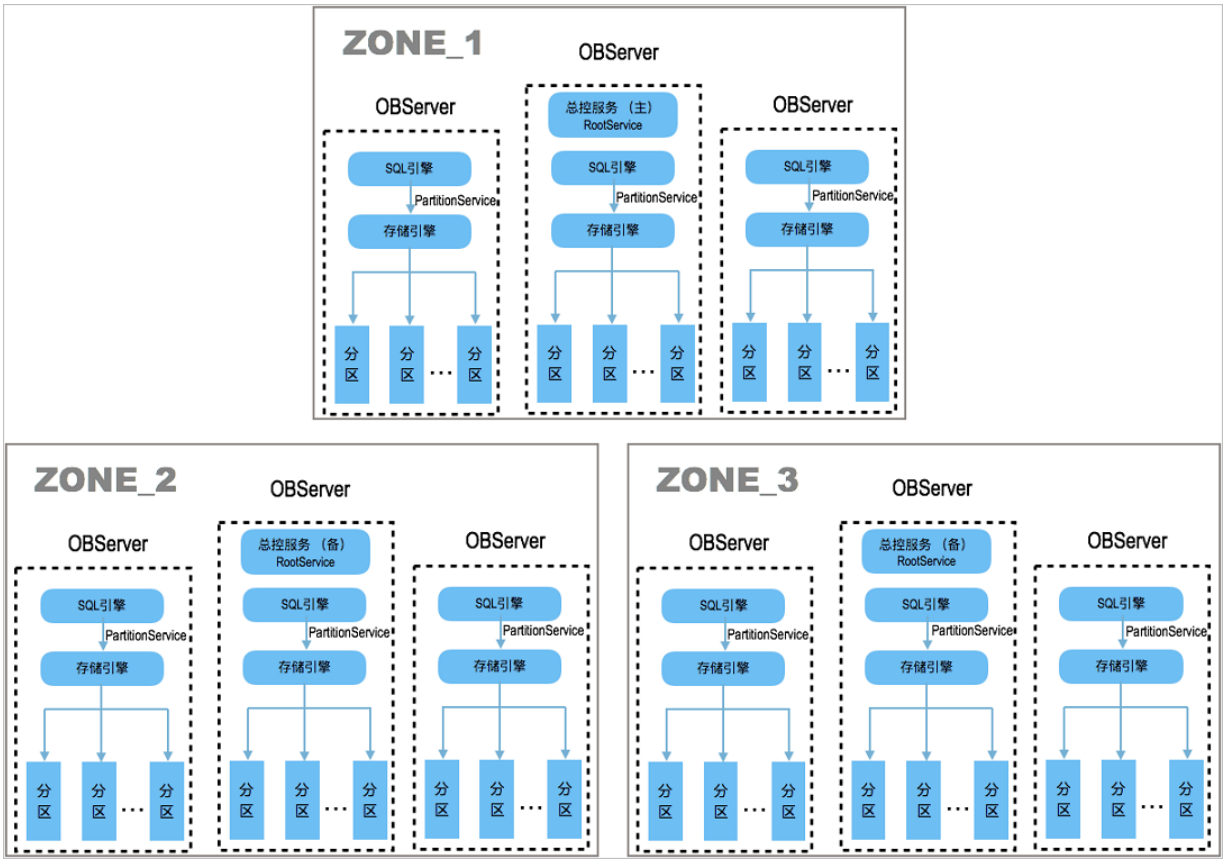
云数据库OceanBase的数据主要可以分为基准数据和增量数据。基准数据是只读数据；增量数据是增、删、改的数据。云数据库OceanBase内部通过合并操作定期将增量数据融合到基准数据中。

云数据库OceanBase是真正的云数据库，采用**单实例多租户**管理模式。根据用户的需求，分配给用户指定规格的**租户**空间，租户可定义的规格包括 CPU 核数、内存、 存储空间等。

15.2 产品架构

云数据库OceanBase的整体架构如图 *15-1:* 云数据库*OceanBase*整体机构所示。

图 15-1: 云数据库OceanBase整体机构



云数据库OceanBase一般部署为三个副本（Zone），每个副本（Zone）由多台物理机器（OBServer）组成。每个Zone包含两种角色：总控服务RootService以及分区服务PartitionService。每个分区服务(PartitionService)管理多个分区，结构上分为两层，SQL引擎以及存储引擎。主RootService和所有OBServer之间维持租约，当OBServer出现故障时，主RootService能够检测到并执行故障恢复操作。RootService和PartitionService 均部署在OBServer上。一台OBServer上可以同时部署RootService 和 PartitionService。RootService选主是通过系统自动选举完成，部署的时候需要指定 RootService列表信息。

15.3 功能特性

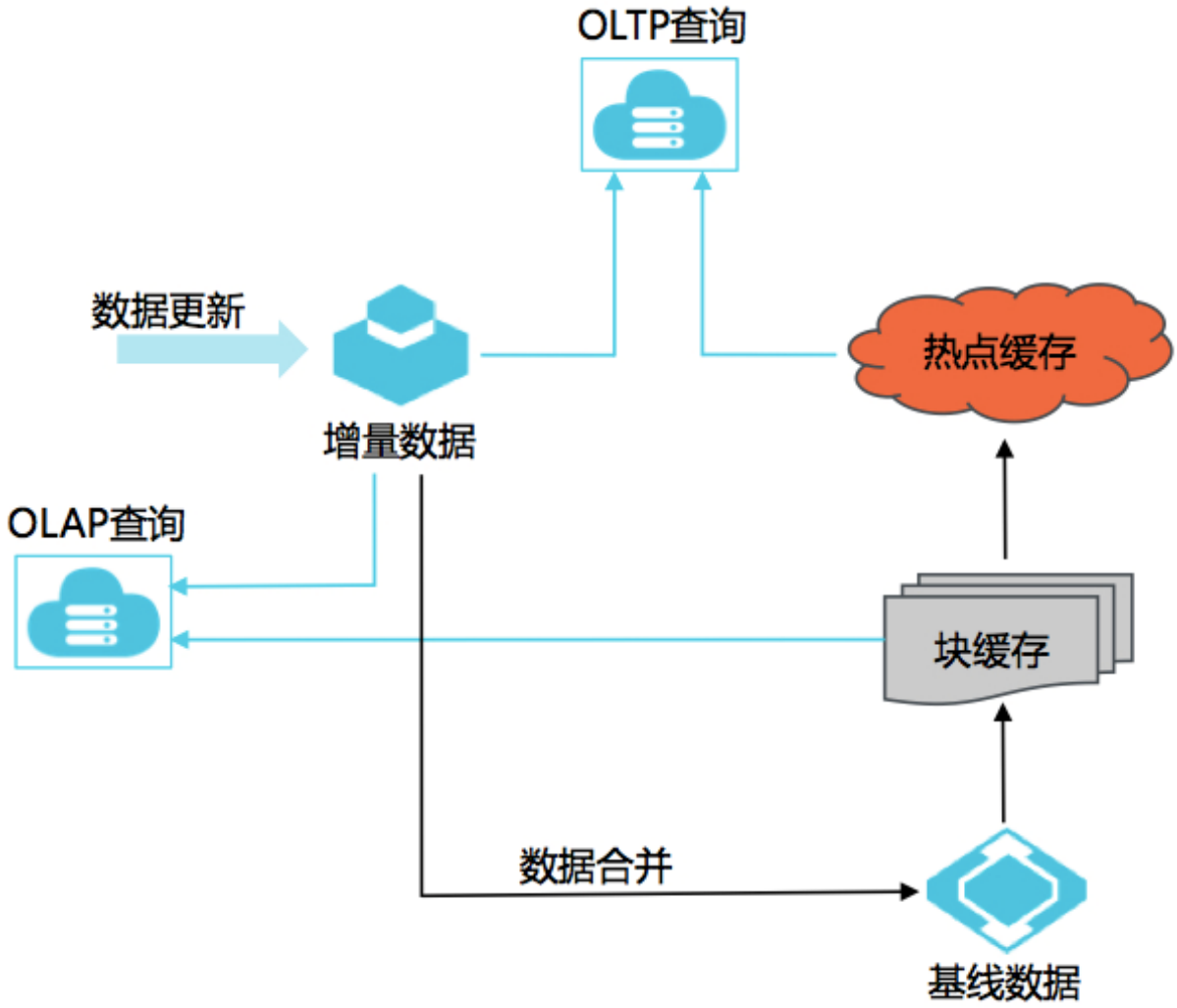
15.3.1 高效的存储引擎

云数据库OceanBase采用的是share-nothing的分布式架构，每个OBServer都是对等的，管理不同的数据分区。

单机的存储引擎采取读写分离的架构，将当前更新的动态数据存入内存称为MemTable，存量的基线数据存在磁盘，称为SSTable。

一个Partition的所有数据（基线数据、增量数据和事务日志）都放在一个OBServer中，因此，针对一个Partition的读写操作不会有跨机的操作，数据的写入也分布到多点并行执行。

图 15-2: OceanBase的高效存储引擎



读写分离的存储结构带来很多好处，因为有大量的静态基线数据，可以很方便对其进行压缩，减少存储成本；另外做行级缓存也不用担心写入带来的缓存失效问题。缺点是读的路径变长，数据需要实时合并可能带来性能问题，OceanBase采用了很多的优化手段，比如 bloom filter cache 对不存在的行做过滤（insert row判断行是否存在无需IO操作），尽量优先读取更新的内容（Active MemTable），如果发现用户读取的列已经读取到，则无需进一步读取基线数据进行合并。

由于增量数据写在内存中，所以内存写到一定量后需要和基线数据合并而生成新的基线，这个过程我们称为合并， 合并会造成一些额外的压力，可能对客户的请求有影响，所以我们采取轮转合并的策略， 即将OceanBase 的多个IDC中的其中一个IDC的流量切走，进行合并，待合并完成后再将流量切到已合并的IDC上，这样可以避免对业务的影响，同时这种策略也可以用在升级维护中，当需要进行版本升级的时候， 可以将其中一台ObServer 的流量切走进行升级， 升级完成后逐步灰度切流， 一旦出现问题即可快速无损回滚。

15.3.2 可扩展性

云数据库OceanBase引入了传统关系型数据库中的数据分区表的方法，语法上也兼容传统数据库创建分区表的方法，这种设计兼具分布式系统的扩展性和关系数据库的易用性和灵活性，符合DBA的使用习惯。

云数据库OceanBase持在线线性扩展，当集群存储容量或处理能力不足时，可以随时加入新的OBServer，系统会自动进行数据迁移，根据机器的处理能力，将合适的数据分区迁移到新加入的机器上；同样在系统容量充足和处理能力富余时，也可以将机器下线，降低成本；在类似于双11大促之类的活动中，可以提供良好的弹性伸缩能力。

对一个数据分区所有读写操作都在其所在的OBServer完成，涉及多个数据分区的事务，会采用两阶段提交的方式在多个OBServer上执行。

在云数据库OceanBase中，事务分为几种类型：

- 单分区事务：和传统的关系数据库一样，属于单机事务，不需要走两阶段提交协议。
- 单机多分区事务：需要走两阶段提交协议，且针对单机做了专门的优化。
- 多机多分区事务：需要走完整的两阶段提交协议。

标准的两阶段协议分为两个阶段：

1. Prepare 阶段：协调者（Coordinator）给所有参与者（Participant）发送prepare请求，参与者将重做日志（Redo）以及prepare日志持久化成功后应答协调者。

   说明：
   这里的持久化成功要求满足多数派原则，即超过一半以上的副本都同步完成且写盘成功。

2. Commit/Abort 阶段：如果所有的参与者全部prepare成功，则协调者给每个参与者发送commit请求；否则，发送abort请求。参与者将commit或者abort日志持久化成功后应答协调者。

Commit/Abort阶段执行完成后，分布式事务的状态才能最终确定。最后，两阶段提交协议还有一个Clear阶段。这个阶段后台异步执行，用于回收前面两个阶段申请的资源。

对于多机多分区事务，需要等到第2步Commit/Abort 执行完成后才能应答客户端事务的最终结果。

对于单机多分区事务，只需要等到第1步Prepare执行完成后即可应答客户端事务的最终结果，从而降低单机多分区事务的延时。

云数据库OceanBase采用多版本并发技术（MVCC）实现并发控制，读事务和写事务之间互不影响。如果读请求只涉及一个分区或者单台OBServer的多个分区，则只需要读取该OBServer上某个时间点的快照即可；如果读请求涉及到多台 OBServer的多个分区，则需要执行分布式快照读。

15.3.3 基于Paxos的日志同步

云数据库OceanBase系统中的每个分区都维护了多个副本，一般为三个，且部署到三个不同的数据中心（Zone）。

整个系统中有成百上千万个分区，这些分区的多个副本之间通过Paxos协议进行日志同步。

每个分区和它的副本构成一个独立的Paxos复制组（Paxos Replication Group），其中一个副本为主（Leader），其它副本为备（Follower）。

每台ObServer服务的一部分分区为Leader，一部分分区为Follower。当ObServer出现故障时，Follower分区不受影响，Leader分区的写服务短时间内会受到影响，直到通过Paxos协议将该分区的某个Follower选为新的Leader为止，整个过程大约20～30秒左右。

OceanBase复制协议分为两个部分：Paxos 分布式选举以及日志同步。

Paxos 分布式选举确保每个分区总是能够选出唯一的Leader，由Leader将日志同步到Follower。只有日志同步到多数派并且写盘成功，才认为事务执行成功。

假设总共有三个副本（N = 3），那么，需要确保 Leader 写盘成功以及其中某个 Follower同步完成且写盘成功。如果某个Follower出现故障，Leader 和剩余的一个Follower还能够构成多数派，系统继续提供服务。如果Leader出现故障，则Leader 最后执行的少量事务可能没有同步到多数派，这些事务的状态不确定，称为未决事务，需要等到某个 Follower接替为新的Leader后才能最终确定。Follower接替为Leader后的第一步操作就是明确前一任 Leader产生的未决事务的状态，这个过程称为重新确认（Reconfirm）。等到 Reconfirm过程全部完成后，新的Leader才开始正式提供服务，系统恢复正常。

当分区发生复制或者迁移时，该分区对应的Paxos复制组发生了成员变更，需要执行成员变更操作。可以想象，成员变更操作也是一个投票，需要得到多数派的认可，且这个多数派包括成员变更前的多数派以及成员变更后的多数派。

15.3.4 多租户

云数据库OceanBase的多租户特性，主要包含如下几点：

- 将多个数据库实例管理和运维的成本和复杂性降低到和一个数据库实例相当。

OceanBase的其中一个优势就是大集群的规模优势，取得这个优势的一个重要原因就是原先的多个数据库实例，在OceanBase版本中管理的成本相当于一个实例。

- 充分利用系统资源，使得同样的资源可以服务更多的业务。通

  过将波峰、波谷期不同的业务系统部署到一个集群，以实现对系统资源最大限度的使用。
- 租户之间的隔离性。

  在数据安全方面，不允许跨租户的数据访问，确保用户的数据资产没有泄露的风险。

  在资源使用方面表现为租户**独占**其资源配额，该租户对应的前端应用，无论是响应时间还是TPS/QPS都比较平稳，不会受到其他租户负载轻重的影响。
- ObServer 的工作线程执行SQL请求时，只要超过一个时间片（10ms左右），就会主动陷入调度器。调度器会根据当时的负载情况选择继续执行该请求，还是将该请求所在的线程调度出去，并调入其它线程继续运行。

15.3.5 MySQL兼容

云数据库OceanBase采取与MySQL兼容的方式，让基于MySQL的应用程序可以不经修改就能运行在OceanBase之上。

云数据库OceanBase在MySQL兼容性方面表现为：

- 接口层面

  OceanBase广泛使用的接口主要是JDBC。通过不断增强在前后台协议上和MySQL的兼容性，目前，使用MySQL的驱动即可无障碍地访问OceanBase。
- 数据模式层面

  完整地支持了数据库（database）、表（table）、视图（view）、自增列（auto increment）等SQL标准的以及MySQL专有的数据模式，并且在数据库系统中实现了多租户（multi-tenant）。

  原先基于不同MySQL实例的业务可以无缝地迁移到一个OceanBase集群。
- 语句/数据类型层面

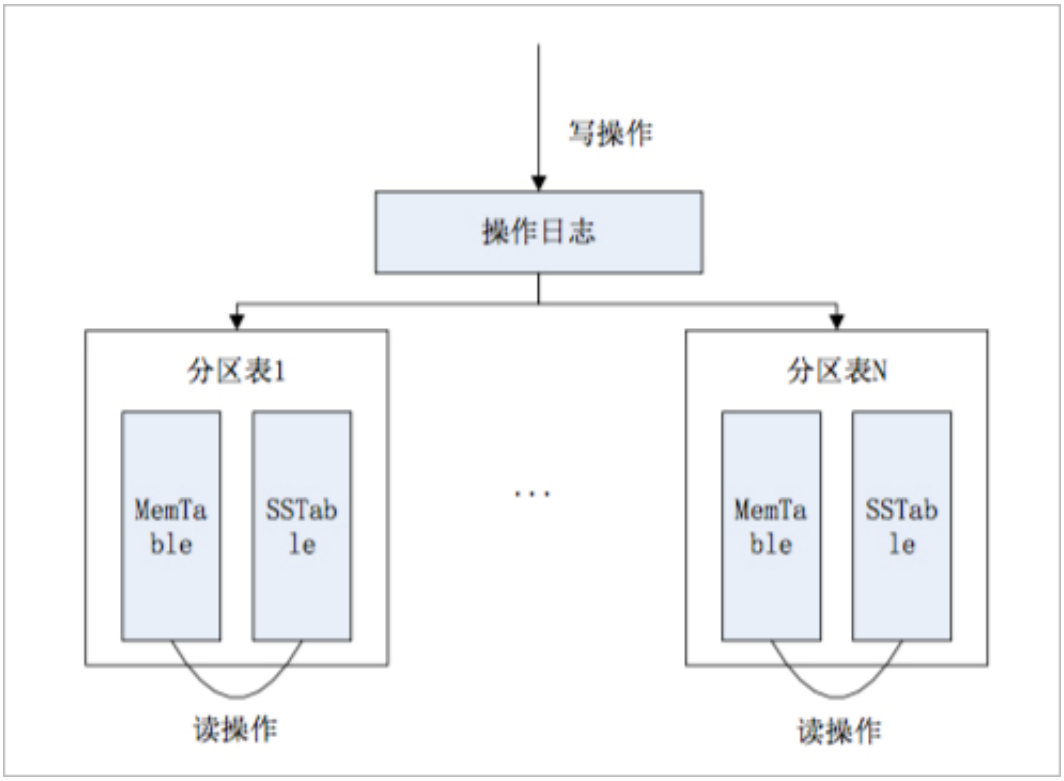
  目前的主流产品在SQL语句层面大多遵守ISO/IEC 9075 标准定义的一系列规范，最新的版本是SQL 2011。
- 事务层面

  系统支持的事务隔离级别以及并发控制方面的表现。

  OceanBase采用的是多版本的并发控制协议，读写不等待，支持Read Committed隔离级别。

15.3.6 单机引擎

图 15-3: 单击引擎原理图



云数据库OceanBase单机引擎的原理如图 *15-3:* 单击引擎原理图所示，OceanBase的每个分区有一个MemTable用于存储修改增量。同时，OceanBase的复制迁移不仅涉及基线数据SSTable，还涉及MemTable。

写操作需要首先记录Redo日志并通过Paxos协议复制到多数派，接着再修改内存。为了提高系统的吞吐量，记录Redo日志时使用了Group Commit技术。另外，由于OceanBase的定位是内存数据库，写日志也需要实现成异步，即SQL线程提交写日志任务后不需要等待，就可以立即处理下一个任务，等到写日志完成后再异步应答客户端，从而做到最优提交。

当内存中的修改增量超过某个阈值时，将触发转储或合并操作。其中，转储操作将内存中的数据写入磁盘生成转储SSTable，而合并操作需要融合基线 SSTable、转储SSTable以及MemTable，生成新的基线SSTable。为了避免读取太多SSTable，转储操作需要控制SSTable的数量，OceanBase的策略是只保留一个有效的转储SSTable，如果已经有一个有效的转储 SSTable，那么，下次执行转储操作时会直接融合该转储SSTable和MemTable。

由于每天的修改增量相对基线数据往往比较小，因此，转储或者合并操作发生的概率很低，事务涉及的操作基本都发生在内存中，每个分区都相当于是一个准内存数据库引擎。

15.3.7 内存事务引擎

内存事务引擎主要包括两个部分：内存索引结构以及多版本并发控制引擎。

OceanBase MemTable采用双索引结构：B+树索引以及哈希索引，B+树索引能够更好地支持范围查找；哈希索引是针对单行查找的一种优化。

每次事务执行时，MemTable会自动维护B+树索引与哈希索引之间的一致性。

OceanBase的内存B+树实现得非常高效，基本没有并发冲突，测试性能优于其它内存数据库使用的替代结构，例如MemSQL中的无锁SkipList。

MemTable在内存中维护了历史版本的事务，每一行将历史事务针对该行的操作按照时间顺序组织成行操作链，新事务提交时会往行操作链尾部追加新的行操作。如果行操作链保存的历史事务过多，将影响读取性能，此时需要触发compaction操作，融合这些历史事务以生成新的行操作链。

例如，某一行有3个行操作：`{(c1, 1), (c2, ‘a’)}`、`{(c1, 2), (c3, true)}` 以及 `{(c1, 3), (c2, “b”)}`，那么，融合后新的行操作链只有1个行操作：`{(c1, 3), (c2, “b”), (c3, true)}`。Compaction操作并不会删除老的行操作链，新的行操作链还会有一个反向指针指向老的行操作链。如果要读取更老的历史快照，只需要顺着内存中的反向指针往前回溯即可，相当于在内存中执行数据库Undo操作。

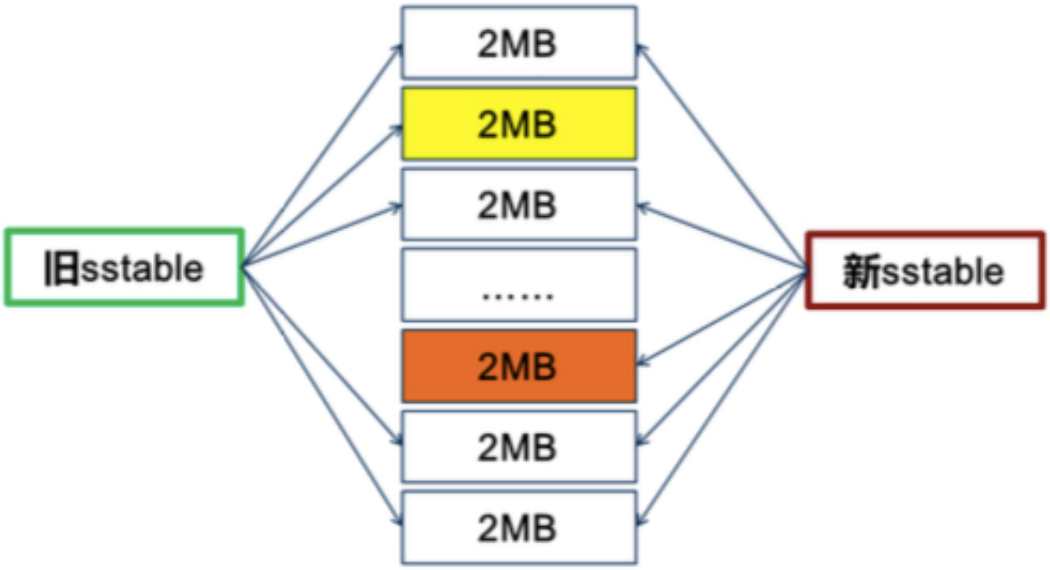
Follower需要接受Leader以Paxos协议发送的Redo日志并回放到MemTable。

MemTable支持多线程并发回放，且Follower回放的开销远小于Leader执行事务的开销，从而避免出现高峰期Follower赶不上Leader的情况。

15.3.8 基线存储

基线存储数据结构称为SSTable（Sorted String Table，有序字符串表），这个名称最初来源于Google Bigtable系统。

图 15-4: 基线存储



如图 *15-4:* 基线存储所示，为了减少SSTable合并的开销，SSTable内部划分为2MB的宏块（Macro Block）。每次旧的SSTable和MemTable合并生成新的SSTable时，如果宏块没有修改，那么，宏块的内容不需要重写，新的SSTable只需要指向老的宏块即可；如果宏块发生修改，那么，需要融合宏块和MemTable 中对应的内容生成新的宏块。可以想象，大部分业务发生修改的宏块占比都比较低，因此，大大降低了合并开销，并节省了磁盘空间。

每个宏块内部又划分为大小在4kb～64kb的微块（Micro Block），微块内部的数据行按照主键有序存储。SSTable保存了宏块的索引结构（有序数组），而宏块内部又保存了微块的索引结构（有序数组）。微块类似于关系数据库中的页面，是读取时块缓存（Block Cache）的基本单位。

除了Block Cache，基线存储还会维护行缓存（Row Cache）、块索引缓存（Block Index Cache）以及布隆过滤器缓存（Bloom Filter Cache）。

对于单行操作，如果行存在，则会命中Row Cache；如果行不存在，则命中Bloom Filter Cache。因此，绝大部分单行操作在基线存储中只需要一次缓存查找，没有额外开销。

15.3.9 管控服务（RootService）

云数据库OceanBase的RootService和PartitionService一样，是OBServer进程的一个功能模块。

OceanBase系统中有一个初始分区（__all_core_table），所有的其它分区都能由该分区逐级索引到。初始分区的Leader所在的OBServer自动提供RootService服务。

RootService的功能主要包括：服务器与 Zone管理、分区管理、每日合并控制、系统自举和DDL操作等。

服务器与Zone管理

RootService与所有OBServer维持租约，管理集群中的所有OBServer，处理OBServer上下线和Zone上下线。

分区管理

RootService管理集群中分区数据分布，发起分区复制、迁移以及分裂、合并等操作。

当OBServer发生故障时，它上面服务的 Leader 分区会通过Paxos协议切到其它OBServer。

如果这台OBServer很快重新上线，那么，它上面的分区副本会重新加入到Paxos复制组；如果这台ObServer很久未上线，那么，它上面的分区副本将被永久删除。接着，RootService会发起分区复制操作，在其它OBServer上增加新的分区副本。

RootService执行分区管理操作时需要确保负载均衡，考虑的因素包括每台 OBServer上的CPU、磁盘使用量、内存使用量和IOPS使用情况等，避免同一张表格的分区全部落到少数几台OBServer上。

每日合并控制

OceanBase中的动态数据和静态数据在每天的指定时间（通常为应用的访问低峰期间）进行数据合并操作，此过程称为每日合并。

每日合并由RootService整体协调，整个集群中的所有OBServer同时进行。

为了尽可能降低每日合并对正常业务的影响，OceanBase中实现了按zone错峰合并策略，即合并之前先将本zone全部分区的Leader切到其它zone，等到合并完成后再切回来。分区 Leader 切换后，读写流量也会跟着自动切换，确保正在执行合并的zone基本没有外部流量。

DDL执行

RootService负责执行DDL操作，包括创建表格、删除表格、增加列、删除列、修改列的属性以及增加索引等。

系统自举（BootStrap）

BootStrap是系统的初始化安装过程，主要用于创建系统内部表，并初始化系统配置。

15.3.10 ObProxy

云数据库OceanBase的应用可以通过ObProxy访问OceanBase服务。应用与ObProxy之间采用标准的MySQL协议，因此，从应用角度看，OceanBase相当于是一个更大、更快、更好的MySQL。

ObProxy实现了一个高性能且易于运维的反向代理服务器。它包含如下几个主要的功能：：

- 反向代理功能

  将请求转发到数据所在的 OBServer，并将OBServer 的响应结果转发给客户端；同时还要防连接闪断，保证 OBServer 宕机或升级不影响客户端正常请求；以及兼容所有 MySQL 客户端。可以想象，ObProxy 内部需要解析用户的SQL请求，并根据请求的分区定位OBServer，处理 OBServer 故障、OceanBase集群错峰合并等内部事件。

- 性能需求

  ObProxy服务器设计数量为OBServer的十分之一， 在性能上吞吐量为100万 QPS，额外增加的响应延迟小于 0.5ms。同时还要支持10万量级的并发连接。为了达到这个要求，ObProxy底层使用了全异步模型及高性能网络框架。

- 运维需求

  主要包括：

  # ObProxy支持热升级；
  # 能够自动配置更新；
  # 支持安全性需求，如配置IP 白名单，防SQL注入，流量控制等；
  # 支持部署在独立的服务器或者直接部署在客户端。

15.4 OceanBase容灾与部署方案

云数据库OceanBase底层通过Paxos协议在保证强一致性的前提下实现了高可用。

Paxos是一种强同步协议，理论上无法规避网络延时问题。因此，在设计容灾方案时，需要考虑机房架构、网络架构对部署的影响。

另外，传统数据库采用主备模式只需要两个副本，而OceanBase使用Paxos协议，至少需要三个副本，甚至五个副本，相应地，OceanBase也需要避免大量副本占用过多资源。

15.4.1 容灾能力

云数据库OceanBase可以做到RPO为0，RTO在1分钟以内的自动无损容灾，并将容灾能力划分为三个等级：

- 服务器（Server）级无损容灾：能够容忍单台服务器不可用，自动无损切换。
- 机房（Zone）级无损容灾：能够容忍单个机房不可用，自动无损切换。
- 地区（Region）级无损容灾：能够容忍某个城市整体不可用，自动无损切换。

15.4.2 部署方式

15.4.2.1 异地三机房

理想的容灾方案是地区级容灾。

以Google Spanner系统为例，全球部署一套生产系统，每次写入都会通过Paxos协议跨地区同步到多个城市。这种方式的问题在于写入延时太长，Google Spanner系统每次写入延时在几十毫秒到几百毫秒之间，取决于不同副本所在城市之间的物理距离。

如果采用异地三机房部署，每次数据库事务提交都会增加一次异地同步延时。

一般来讲，国内不同地区之间的机房延时(例如上海到深圳)在几十毫秒左右，而每一笔业务往往包含多个数据库事务。因此，可能需要对业务进行针对性改造，减少每一笔业务包含的数据库事务个数。

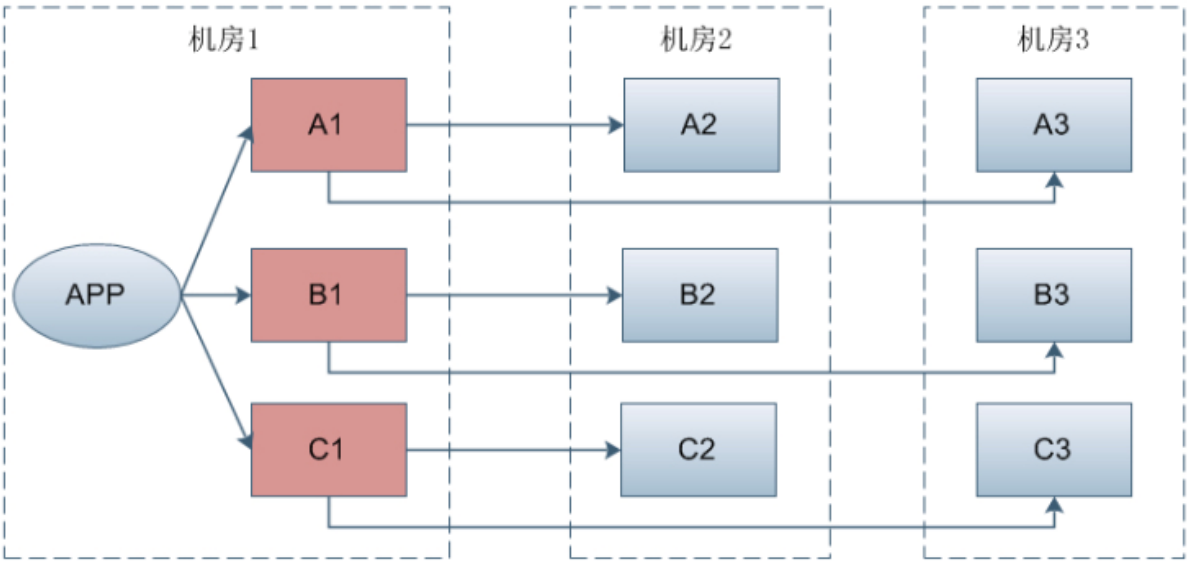
15.4.2.2 同城三机房

折衷的方式是只支持机房级容灾。

通过Paxos协议的原理可以知道，为了支持机房级容灾，至少需要有三个机房；否则，当一个机房出现故障时，另外一个机房无法单独构成多数派，也就无法实现无损切换。

三个机房中，一个机房为主库，另外两个机房为备库。APP和主库部署在同一个机房，如图 *15-5: 同城三机房*所示。

图 15-5: 同城三机房



正常情况下，业务端访问部署在同机房的主分区，即业务端APP访问机房1的A1、B1、C1。

当机房1的OceanBase出现故障时，它上面服务的主分区可能切到机房2或者机房3，例如A1、B1、C1分别切到A2、B3、C2。接下来，部署在机房 1 的 APP 会跨机房访问新的主 分区 A2、B3、C2。等到机房 1 从故障中恢复过来以后，OceanBase又会把主分区重新切回机房1的A1、B1、C1，避免APP跨机房访问。在这种情况下，机房1是OceanBase的优先机房，内部通过primary zone 来标识。当然，如果机房1整体故障，里面的APP往往也不可用，应用系统有一套针对APP的容灾方案。APP容灾不是我们的讨论重点，本文的讨论均假设 APP 是持续可用的。

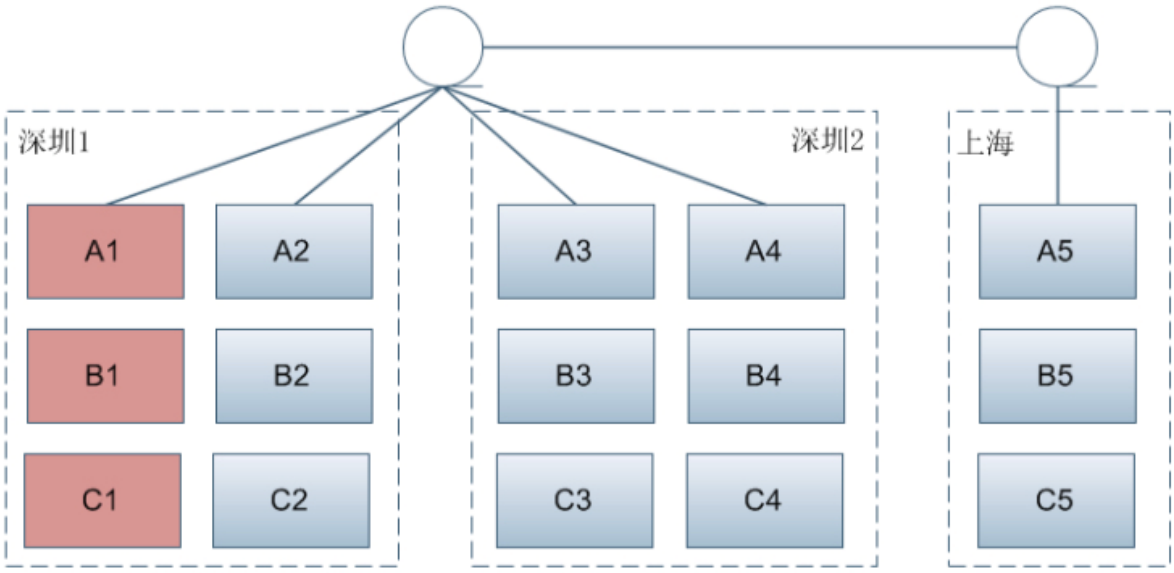
对于同城三机房部署，每次事务提交都需要至少强同步到另外一个机房的某个备副本。一般来说，同城网络延时在2毫秒，因此，每次事务提交增加的延时在2毫秒左右，以支付宝核心业务为例，一次数据库事务往往包含6~15条SQL语句，整个事务的执行时间在15毫秒到50毫秒左右，2毫秒的额外延时可以接受。

15.4.2.3 两地三中心

考虑到机房和网络建设成本，部分地区可能总共只有两个机房。软件架构应该尽可能普适，因此，我们需要对这种场景做专门的设计。

由于机房级无损容灾至少需要三个机房，因此，除了同一个城市的两个机房外，还需要在另外一个城市找第三个机房，一起组成两地三数据中心。

图 15-6: 两地三中心



如图 *15-6:* 两地三中心所示，假设深圳有两个机房，上海一个机房。对于每个分区，深圳机房分别部署两个副本，上海机房只部署一个副本，形成**2 + 2 + 1**部署。每个分区总共包含五个副本，正常情况下，只需要强同步深圳的三个副本即可，每次事务提交增加的网络延时在2毫秒之内。

当上海机房出现故障时，仍然只需要强同步深圳的三个副本，系统无影响。当深圳机房 出现故障时，例如深圳2机房故障，此时，系统总共只有三个副本。如果不做处理，每个事务提交时都需要做一次深圳到上海的跨地区同步，网络延时不可接受。OceanBase的做法是利用Paxos协议做一次成员变更操作，将深圳2机房的副本从Paxos选举组中剔除，副本总数由五个变成三个。这样，以后只需要同步三个副本中的两个即可，避免了跨地区同步，这是因为成员变更也是Paxos协议的一部分，只要同时得到变更前与变更后的多数派承认，成员变更即可生效。

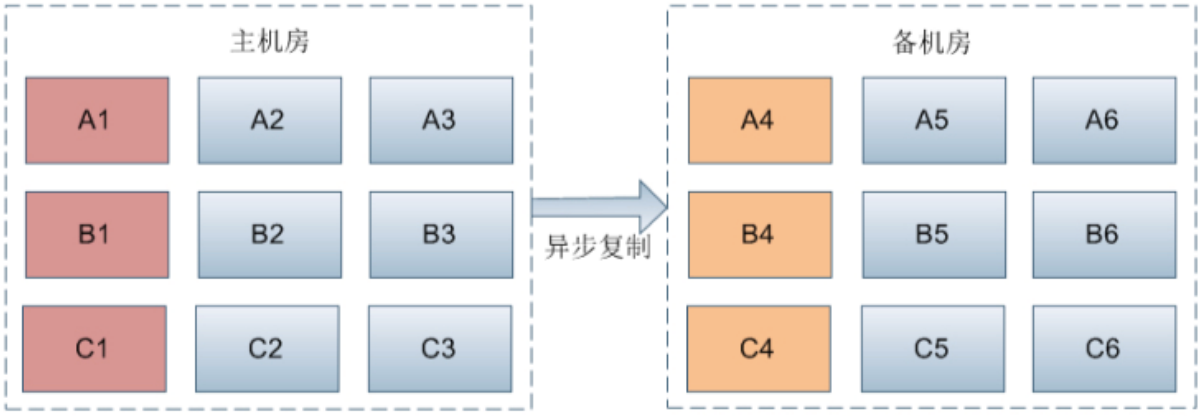
相比较传统关系数据库在银行往往也部署成**两地三中心**，同城两个机房，一个主库，一个热备；异地一个冷备。当主库出现故障时，无论热备还是冷备，都没有包含最新的修改，因此，除非等待主库恢复才提供服务，否则，强制将备库切为主库必然导致数据丢失。这就意味着，传统关系数据库的“两地三中心”方案要么选择高可用，要么选择强一致。 如果选择高可用，RTO可以做到很小，但是RPO总是大于0;如果选择强一致，RPO等于0，但是RTO不可控。而OceanBase的**2 + 2 + 1**方案虽然也部署了两地三中心，但是，底层通过Paxos协议实现了机房级无损容灾。当某个机房出现故障时，RPO为0且RTO在1分钟以内。

15.4.2.4 双机房热备

部分业务可能总共只有两个机房，这种情况下，理论上无法做到机房级无损容灾。

此时，OceanBase可以提供双机房热备容灾方案。

图 15-7: 双机房热备



如图 *15-7:* 双机房热备所示，主机房和备机房分别独立部署了OceanBase集群，主机房的集群称为主集群，备机房的集群称为备集群。主集群的三个分区A1、B1、C1是主分区，备集群的三个分

区 A4、B4、C4是主分区。主集群的修改操作会异步复制到备集群。如果单台OceanBase服务器故障，能够自动无损切换，RTO在1分钟以内，RPO为0；如果主集群整体故障，可以将备集群强制切换为主集群继续提供服务，RTO在5分钟以内，RPO在秒级。这就意味着，双机房热备方案支持服务器级无损容灾，不支持机房级无损容灾。

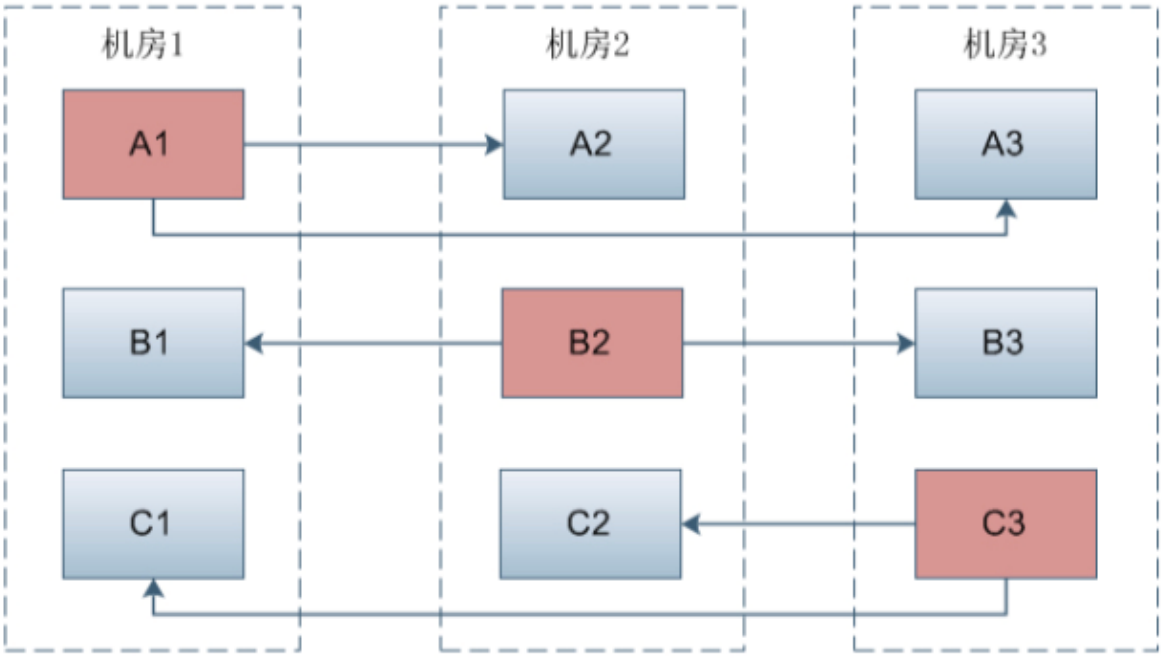
15.4.3 部署与成本

15.4.3.1 交叉部署

如果OceanBase 部署了三个机房，机房1为主机房，机房2和机房3为备机房，无法提供服务，相当浪费。这种情况下，可以做成交叉部署。

如图 *15-8:* 交叉部署所示，A1、B2、C3为主分区，分别部署在机房1、机房2和机房3，从而把这三个机房的OceanBase机器资源都给用上，避免浪费。

图 15-8: 交叉部署



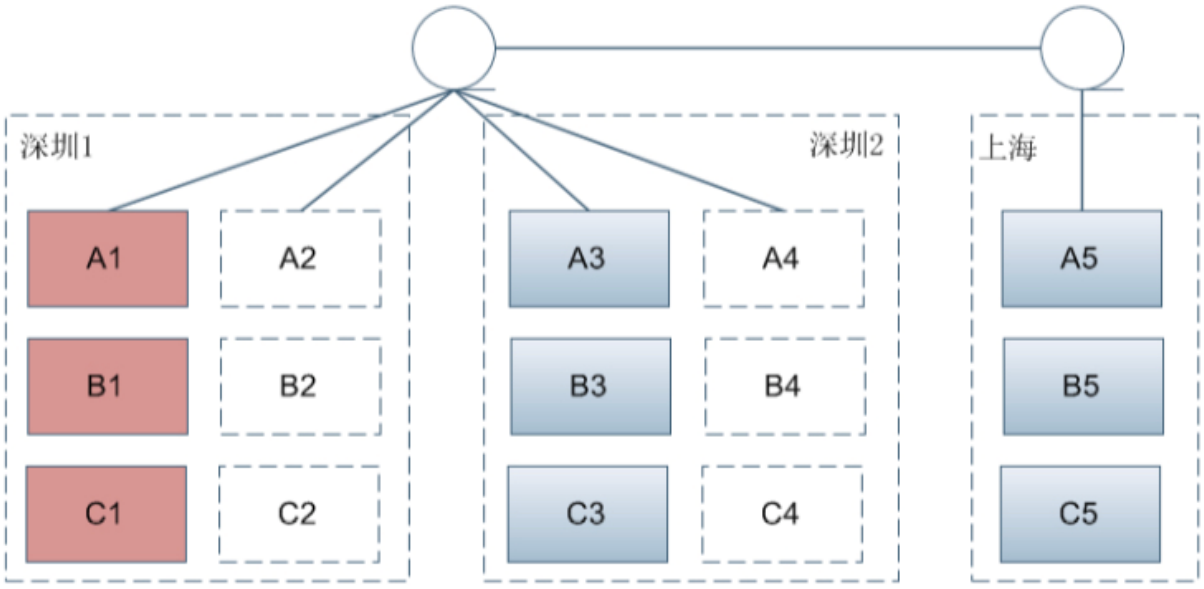
15.4.3.2 日志副本

即使选择交叉部署，每个副本仍然需要保存完整的SSTable和MemTable，占用存储空间和内存。考虑到很多副本只是为了高可用，因此，可以只写重做日志(redo log)，这些副本称为**日志副本**。

以**2 + 2 + 1**部署为例，可以将深圳1和深圳2机房中各一个副本改造为日志副本。

如图 *15-9:* 日志副本所示，副本A2、A4、B2、B4、C2、C4为日志副本，只写日志，不保存SSTable和MemTable。另外，再通过将日志副本和正常副本交叉部署，可以做到日志副本基本不占用额外的机器资源。

图 15-9: 日志副本



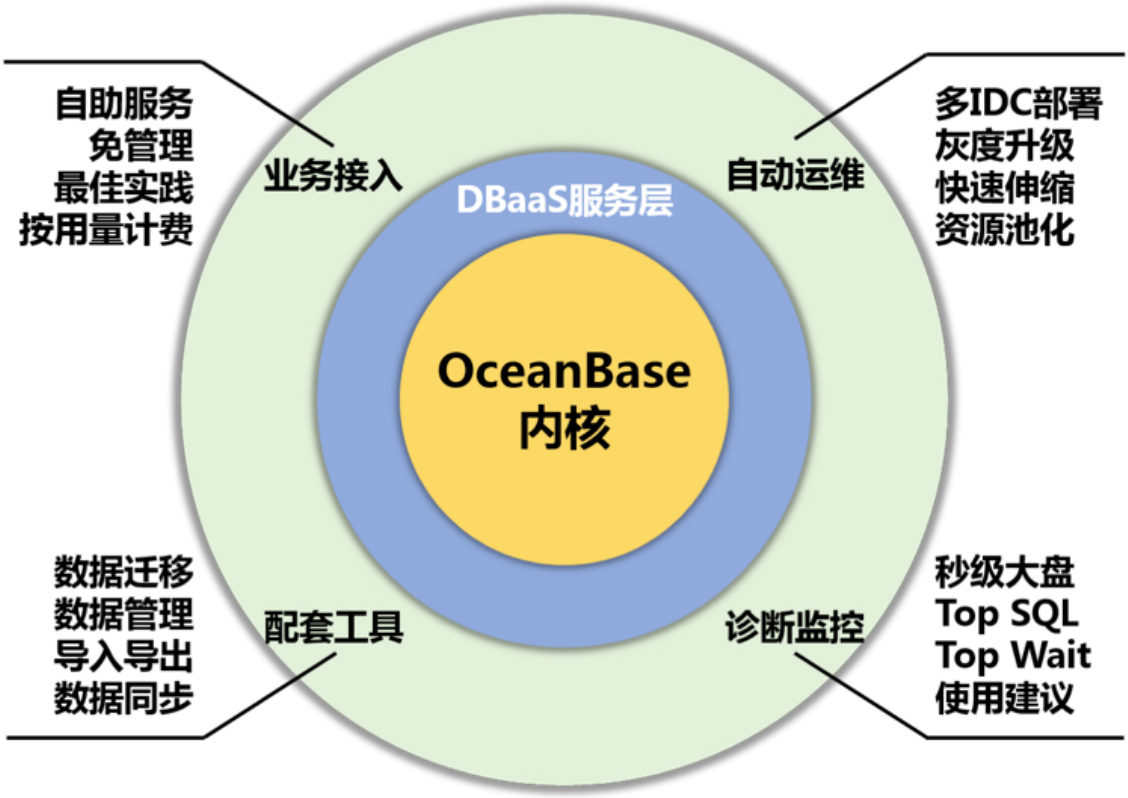
15.5 OceanBase云平台

如图 *15-10: OceanBase*云平台所示，OceanBase系统本身只提供数据库内核功能，所有的业务接入都需要通过OceanBase云平台（OCP）。OceanBase 与OCP一起提供DBaaS服务。

用户可以从OCP平台申请租户实例，并设置实例的资源限制（CPU、Memory、IOPS），使用一键上云组件将原有的MySQL业务一键迁移到OceanBase，在云平台上动态增加或减少租户的资源，执行备份恢复。

另外，OCP平台还提供诊断监控、运行报表、智能告警以及资源调度等服务。即使用户写的SQL语句不太合理，OCP平台也能够自动发现并且提供优化建议。

图 15-10: OceanBase云平台



其中，DBaaS服务层还提供了丰富的API，可以根据自身的业务流程进行定制。

16 数据传输服务DTS

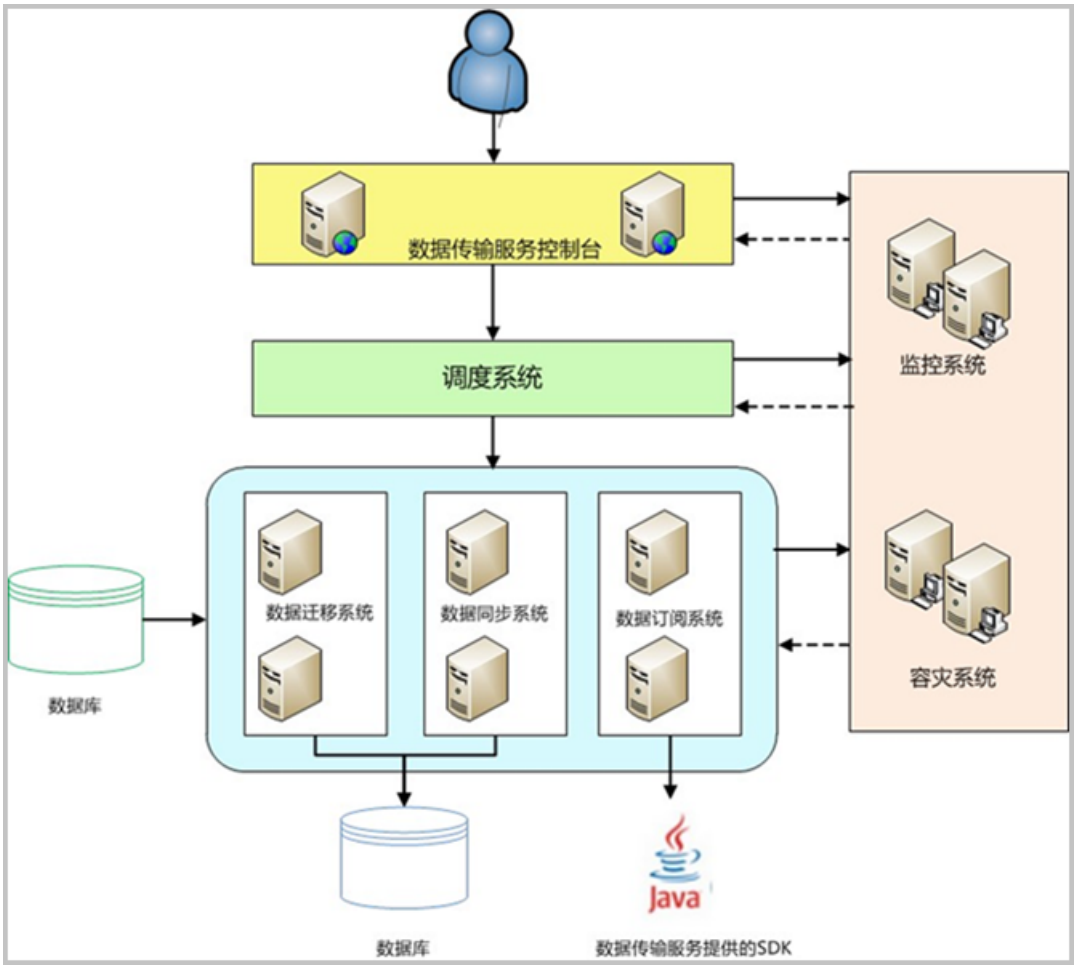
16.1 什么是DTS

数据传输服务（Data Transmission Service，简称DTS）是阿里云提供的一种数据服务，它支持关系型数据库及大数据等多种类型的数据库之间的数据交互。它提供了数据迁移、实时数据订阅及数据实时同步等多种数据传输能力。通过数据传输可实现不停服数据迁移、数据异地灾备、缓存更新策略、离线在线数据实时同步，以及异步消息通知。

多种业务应用场景，助您构建安全、可扩展、高可用的数据架构。

16.2 系统架构

DTS系统架构如下图所示。



16.3 环境说明

DTS支持的机型列表如下：

- PF51.*
- PV52P2M1.*
- DTS_E.*
- PF61.*
- PF61P1.*
- PV62P2M1.*
- PV52P1.*
- Q5F53M1.*
- PF52M2.*
- Q41.*
- Q5N1.22
- Q5N1.2B
- Q46.22
- Q46.2B
- W41.22
- W41.2B
- W1.22
- W1.2B
- W1.2C
- D13.12

操作系统列表如下：

AliOS7U2-x86-64

注意：

- 非列表里面的机型，请勿使用DTS。
- DTS所使用的目录为*/apsara*，只会使用到一块硬盘，请确保该硬盘的空间大于2T。如果*/apsara*目录空间小于2T，会影响任务的正常运行、导致数据报错；如果任务失败，会影响任务的恢复和数据的回拉。

16.4 基本功能

16.4.1 数据迁移

16.4.1.1 功能简介

数据迁移功能旨在帮助用户方便、快速地实现各种数据源之间的数据迁移。实现数据迁移上云、阿里云内部跨实例数据迁移、数据库拆分扩容等业务场景。数据传输服务提供的数据迁移功能能够支持同异构数据源之间的数据迁移，同时提供了库表列三级映射、数据过滤多种ETL特性。

16.4.1.2 数据源

数据迁移支持多种数据源之间的数据迁移，不同数据源的支持情况如下表：

表 16-1: 不同数据源的支持情况

数据源	结构迁移	全量迁移	增量迁移
MySQL > RDS for MySQL	支持	支持	支持
MySQL > DRDS	不支持	支持	支持
MySQL > HybridDB for MySQL	不支持	支持	支持
Oracle > DRDS	不支持	支持	支持

16.4.1.3 在线迁移

数据传输服务提供的数据迁移方式是在线迁移，在线迁移只要用户配置迁移的源、目标实例及迁移对象即可，DTS会自动完成整个数据迁移过程。在线迁移支持数据不停服迁移，但要求DTS服务器能够同时与源实例、目标实例连通。

16.4.1.4 迁移模式

数据迁移支持结构迁移、全量数据迁移及增量数据迁移等迁移模式。其中：

- 结构迁移：将源实例中的结构对象定义一键迁移至目标实例。
- 全量数据迁移：将源实例中的历史存量数据迁移至目标实例。
- 增量数据迁移：将迁移过程中源实例产生的增量数据实时同步到目标实例。结构迁移 + 全量数据迁移 + 增量数据迁移可以简单实现业务不停服迁移。

16.4.1.5 ETL特性

数据迁移支持多种ETL特性，主要包括：

- 支持库表列三级对象名映射。库表列三级对象名映射是指可以实现源实例和目标实例的库名或表名，甚至列名不同的两个对象之间的数据迁移。
- 支持迁移数据过滤，用户可以对需要迁移的表配置SQL过滤条件，过滤数据。例如用户可以配置时间条件，只迁移最新的数据。

16.4.1.6 迁移任务

迁移任务是DTS进行数据迁移的基本单元。如果需要进行数据迁移，必须在DTS控制台创建一个迁移任务。当创建迁移任务时，需要配置待迁移源实例的连接方式、目标实例的连接方式、迁移对象及迁移类型等信息。用户可以在DTS控制台进行迁移任务的创建、管理、停止及删除等操作。

16.4.2 数据订阅

16.4.2.1 功能简介

实时数据订阅功能旨在帮助用户获取RDS/DRDS的实时增量数据，用户能够根据自身业务需求自由消费增量数据，例如实现缓存更新策略、业务异步解耦、异构数据源数据实时同步及含复杂ETL的数据实时同 步等多种业务场景。

功能列表：支持经典网络、专有网络下RDS For MySQL实例的数据订阅。

实时数据订阅支持的数据源类型包括：

- RDS For MySQL
- DRDS

其中，DRDS 不记录事务日志。如果需要订阅DRDS的实时增量数据，那么需要通过订阅DRDS底层挂载的RDS实例的增量日志来实现。

16.4.2.2 订阅对象及通道

订阅通道

订阅通道是进行增量数据订阅与消费的基本单元。如果要订阅RDS的增量数据，必须在数据传输控制台创建一个针对这个RDS实例的订阅通道。订阅通道会实时拉取RDS的增量数据，并将最新一段时间的增量数据保存在订阅通道中，用户可以使用数据传输提供的SDK从这个订阅通道中订阅增量数据，并进行相应的消费。同时，用户可以在数据传输控制台进行订阅通道的创建、管理及删除等操作。

一个订阅通道同时只能被一个下游SDK订阅消费，如果用户有多个下游需要订阅同一个RDS实例时，需要创建多个订阅通道。这些订阅通道订阅的RDS实例均为同一个实例ID。

订阅通道在创建及运行过程中，不同阶段会处于不同的状态，具体如表 *16-2:* 通道状态及说明所示。

表 16-2: 通道状态及说明

通道状态	状态说明	可进行操作
预检中	订阅通道已经完成任务配置，正在进行启动之前的简单预检查。	删除订阅
未启动	迁移任务已经通过迁移之前的预检查，但是还没有启动订阅。	• 开始订阅 • 删除订阅
初始化	订阅通道正在进行启动初始化，一般需要1分钟左右。	删除订阅
正常	订阅通道正在正常拉取RDS实例的增量数据。	• 查看示例代码 • 查看订阅数据 • 删除订阅
异常	订阅通道拉取RDS实例增量数据异常。	• 查看示例代码 • 删除订阅

订阅对象

数据订阅的订阅对象可以为：库、表。用户可以根据需要订阅某几个表的增量数据。

数据订阅将增量数据细分为数据变更和数据结构变更，配置数据订阅时，可以选择需要订阅的具体数据变更类型。

16.4.2.3 高级特性

数据订阅支持多种特性，有效降低用户使用门槛，主要包括：

- 动态增减订阅对象

  在数据订阅过程中，用户可以随时增加或减少需要订阅的对象。

- 在线查看订阅数据

  数据传输DTS控制台支持在线查看订阅通道中的增量数据。

- 修改消费时间点

  数据订阅支持用户随时修改需要消费数据对应的时间点。

16.4.3 数据同步

16.4.3.1 功能简介

数据实时同步功能旨在帮助用户实现两个数据源之间的数据实时同步。通过数据实时同步功能可实现数据异地灾备、本地数据灾备及在线离线数据打通（OLTP到OLAP）数据同步等多种业务场景。

功能列表：

- 支持任何两个RDS For MySQL实例间的数据实时同步。
- 支持RDS for MySQL实例和分析型数据库AnalyticDB实例间的数据实时同步。
- 支持RDS for MySQL实例和MaxCompute实例间的数据实时同步。

16.4.3.2 同步作业

同步作业是数据实时同步的基本单元。如果要进行两个实例间的数据同步，必须在数据传输控制台创建同步作业。

同步作业在创建及运行过程中，不同阶段会处于不同的状态，具体如表 *16-3:* 作业状态及说明所示。

表 16-3: 作业状态及说明

作业状态	状态说明	可进行操作
预检中	同步作业正在进行启动前的预检查。	• 查看同步配置 • 删除同步 • 复制同步配置 • 配置监控报警
预检查失败	同步作业预检查没有通过。	• 预检查 • 查看同步配置 • 修改同步对象 • 修改同步速度 • 删除同步 • 复制同步配置 • 配置监控报警
未启动	迁移任务已经通过迁移之前的预检查，但是还没有启动。	• 预检查 • 开始同步 • 修改同步对象

作业状态	状态说明	可进行操作
		• 修改同步速度 • 删除同步 • 复制同步配置 • 配置监控报警
同步初始化中	同步作业正在进行同步初始化。	• 查看同步配置 • 删除同步 • 复制同步配置 • 配置监控报警
同步初始化失败	同步作业在初始化过程中，迁移失败。	• 查看同步配置 • 修改同步对象 • 修改同步速度 • 删除同步 • 复制同步配置 • 配置监控报警
同步中	同步作业正常同步中。	• 查看同步配置 • 修改同步对象 • 修改同步速度 • 暂停同步 • 删除同步 • 复制同步配置 • 配置监控报警
同步失败	同步作业同步异常。	• 查看同步配置 • 修改同步对象 • 修改同步速度 • 启动同步 • 删除同步 • 复制同步配置 • 配置监控报警
暂停中	同步作业执行了暂停，处于暂停状态。	• 查看同步配置 • 修改同步对象 • 修改同步速度 • 启动同步 • 删除同步

作业状态	状态说明	可进行操作
		• 复制同步配置 • 配置监控报警

16.4.3.3 同步对象

- 数据同步的同步对象的选择粒度可以为：库、表、列。用户可以根据需要同步某几个表的数据。
- 数据同步支持库、表、列名映射，即支持不同名称的数据库、表以及列之间的同步。
- 用户还可以根据业务需求，只同步表中的某几列数据。

16.4.3.4 高级特性

数据同步支持多种特性，有效降低用户使用门槛，主要包括：

- 动态增减同步对象

  在数据同步过程中，用户可以随时增加或减少需要同步的对象。

- 完善性能查询体系

  数据同步提供同步延迟、同步性能（RPS、流量）趋势图，用户可以方便查看同步链路的性能趋势。

16.5 产品优势

数据传输（Data Transmission）服务支持关系型数据库、OLAP等数据源间的数据传输。它提供了数据迁移、数据订阅及数据同步等多种数据传输方式。相对于第三方迁移同步工具，数据传输服务提供更丰富多样、高性能、高安全可靠的传输链路，同时它提供了诸多便利功能，极大地方便了传输链路的创建及管理。

丰富多样

数据传输服务能够支持多种同异构数据源之间的迁移同步。对于异构数据源之间的迁移，数据传输服务支持结构对象定义的转化。

数据传输服务支持多种传输方式，数据迁移、数据订阅及数据同步。其中数据订阅及数据同步均为实时数据传输方式。

为了降低数据迁移对应用的影响，数据迁移功能支持不停服迁移方式。不停服迁移，可实现在数据迁移过程中，应用停机时间降低到秒级别。

高性能

数据传输服务使用高规格服务器来保证每条迁移同步链路都能拥有良好的传输性能。

对于数据迁移，数据传输服务底层使用了多种性能优化措施。

相对于传统的数据同步工具，数据传输服务的实时同步功能能够将并发粒度缩小到事务级别，能够并发同步同张表的更新数据，从而极大得提升同步性能。

安全可靠

数据传输服务底层为服务集群，集群内任何一个节点宕机或发生故障，控制中心都能够将这个节点上的所有任务快速切换到其他节点上。

数据传输服务各模块间采用安全传输协议及安全token认证，有效得保证数据传输可靠性。

简单易用

数据传输服务提供可视化管理界面，提供向导式的链路创建流程，用户可以在其控制台简单轻松地创建自己的传输链路。

数据传输服务控制台展示了链路的传输状态、传输进度和传输性能等信息，用户可以方便管理自己的传输链路。

为了解决网络或系统异常等导致的链路中断问题，数据传输服务提供链路断点续传功能，且定期监测所有链路的状态，一旦发现链路异常，先尝试自动修复重启，如果链路需要用户介入修复，那么用户可以直接在控制台修复后触发链路重启。

17 负载均衡SLB

17.1 什么是负载均衡

负载均衡（Server Load Balancer）是将访问流量根据转发策略分发到后端多台云服务器（ECS实例）的流量分发控制服务。负载均衡扩展了应用的服务能力，增强了应用的可用性。

负载均衡通过设置虚拟服务地址，将添加的ECS实例虚拟成一个高性能、高可用的应用服务池，并根据转发规则，将来自客户端的请求分发给云服务器池中的ECS实例。
负载均衡默认检查服务器池中的ECS实例的健康状态，自动隔离异常状态的ECS实例，消除了单台ECS实例的单点故障，提高了应用的整体服务能力。此外，负载均衡还具备抗DDoS攻击的能力，增强了应用服务的防护能力。

负载均衡由以下三个部分组成：

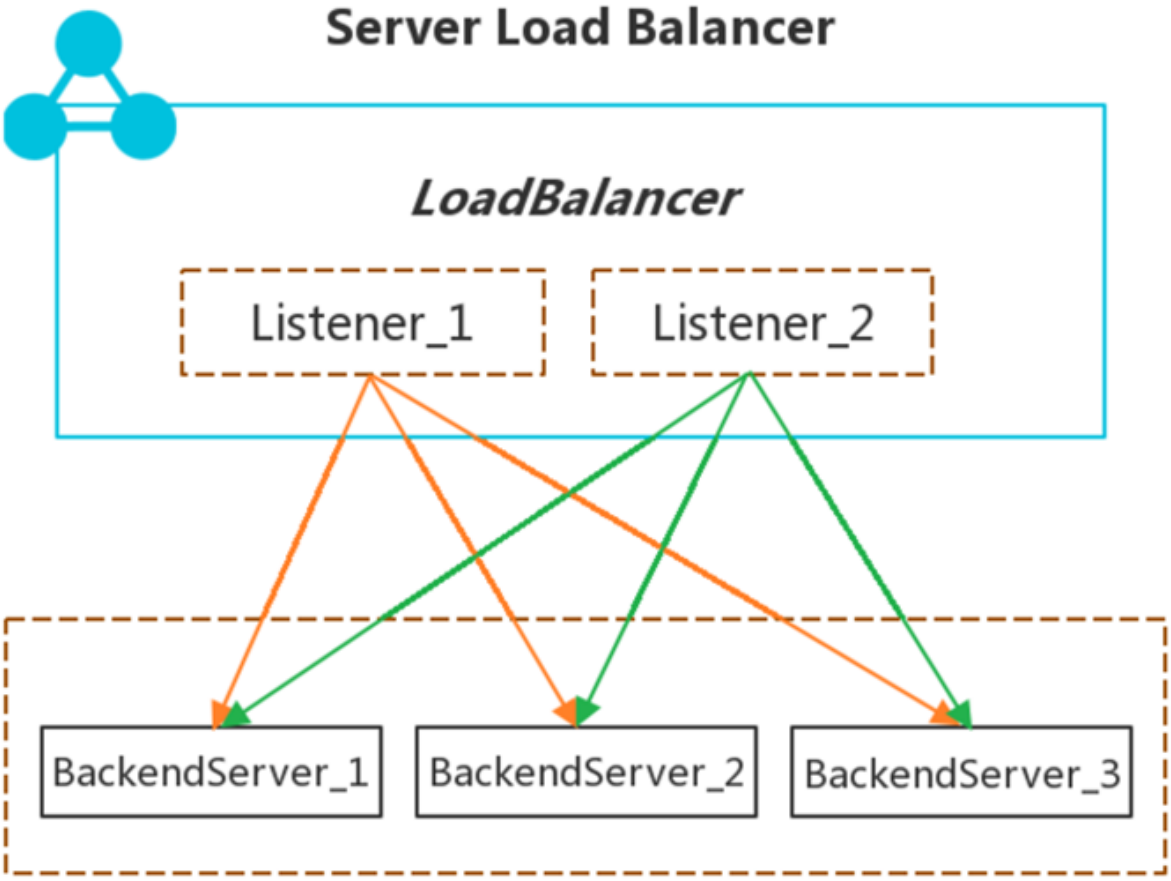
- 负载均衡实例（Server Load Balancer instances）：一个负载均衡实例是一个运行的负载均衡服务，用来接收流量并将其分配给后端服务器。

  要使用服负载均衡服务，您必须创建一个负载均衡实例，并至少添加一个监听和两台ECS实例。
- 监听（Listeners）：监听用来检查客户端请求并将请求转发给后端服务器。监听也会对后端服务器进行健康检查。

  您可以根据需要创建七层（HTTP/HTTPS协议）或四层（TPC/UDP）监听。对于七层监听，您可以创建基于域名和URL的转发规则。
- 后端服务器（Backend servers）：后端服务器是添加到负载均衡实例中用来接收和处理转发的请求的ECS实例。您可以通过创建服务器组对运行不同应用或承载不同作用的ECS实例进行分类。

  如下图所示，来自客户端的请求经过负载均衡实例后，负载均衡监听根据配置的监听规则，将请求转发给对应的后端ECS实例。

图 17-1: 负载均衡构成

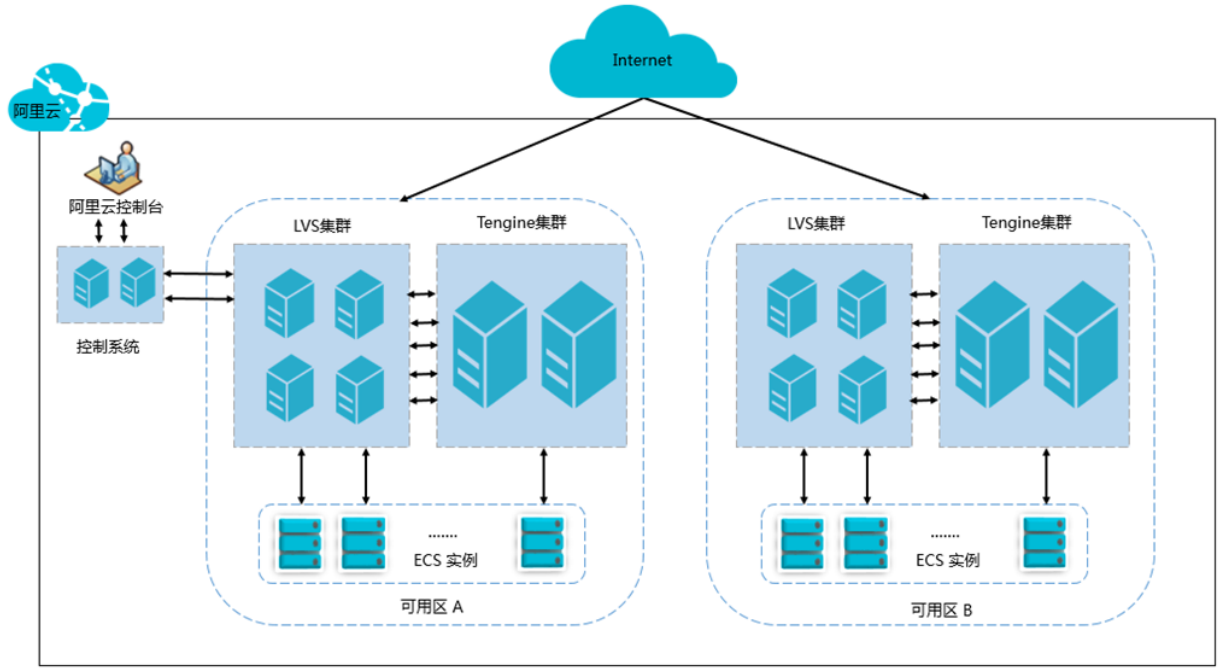


17.2 产品架构

负载均衡采用集群部署，可实现会话同步，以消除服务器单点，提升冗余，保证服务的稳定性。专有云当前提供四层（TCP协议和UDP协议）和七层（HTTP和HTTPS协议）的负载均衡服务。

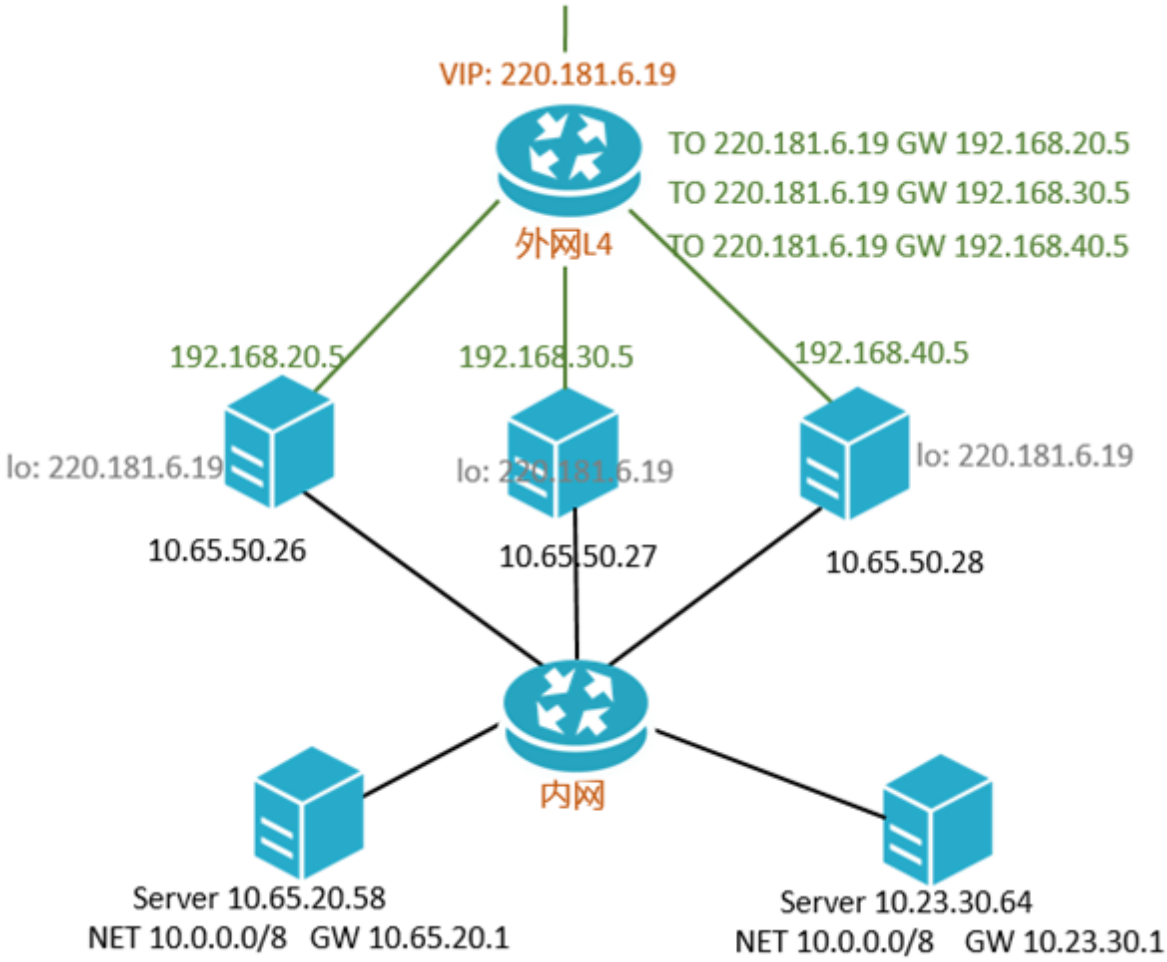
- 四层采用开源软件 LVS（Linux Virtual Server）+ keepalived的方式实现负载均衡，并根据云计算需求对其进行了定制化优化。
- 七层采用Tengine实现负载均衡。Tengine是由淘宝网发起的Web服务器项目，它在Nginx的基础上，针对大访问量网站的需求，添加了很多高级功能和特性。

图 17-2: 负载均衡架构



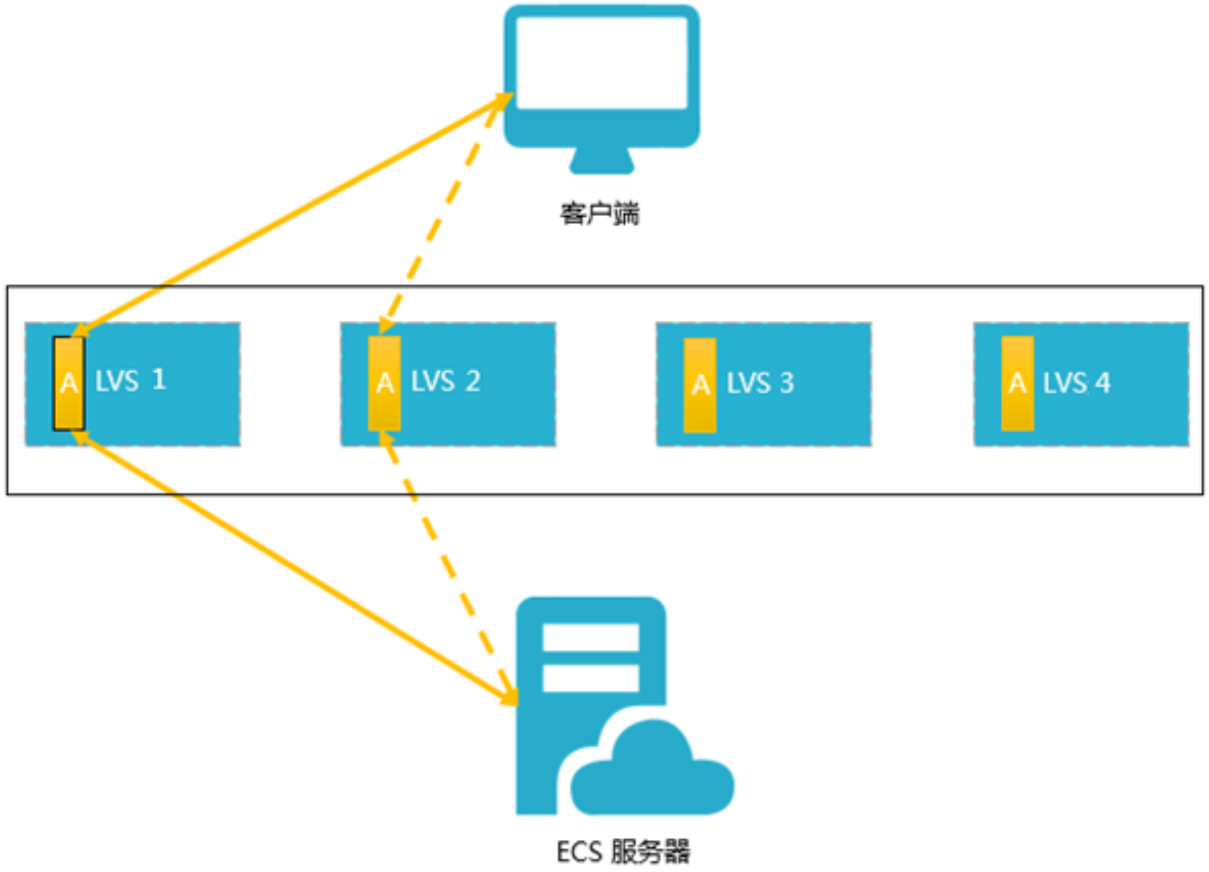
如下图所示，四层负载均衡实际上是由多台LVS机器部署成一个LVS集群来运行的。采用集群部署模式极大地保证了异常情况下负载均衡服务的可用性、稳定性与可扩展性。

图 17-3: 集群部署



LVS集群内的每台LVS都会进行会话，通过组播报文同步到该集群内的其它LVS机器上，从而实现LVS集群内各台机器间的会话同步。如下图所示，当客户端向服务端传输三个数据包后，在LVS1上建立的会话A开始同步到其它LVS机器上。图中实线表示现有的连接，图中虚线表示当LVS1出现故障或进行维护时，这部分流量会转发到一台可以正常运行的机器LVS2上。因而负载均衡集群支持热升级，并且在机器故障和集群维护时最大程度对用户透明，不影响用户业务。

图 17-4: 会话同步



17.3 四层负载均衡LVS技术特点

官方LVS存在的问题

LVS是全球最流行的四层负载均衡开源软件，由章文嵩博士在1998年5月创立，可以实现LINUX平台下的负载均衡。LVS是基于linux netfilter框架实现（同iptables）的一个内核模块，名称为IPVS (IP Virtual Server)，其钩子函数分别HOOK在LOCAL_IN和FORWARD两个HOOK点。

在云计算大规模网络环境下，官方LVS存在如下问题：

- 问题1：LVS支持NAT/DR/TUNNEL三种转发模式。上述模式在多vlan网络环境下部署时，存在网络拓扑复杂，运维成本高的问题。
- 问题2：和商用负载均衡设备（如F5）相比，LVS缺少DDoS攻击防御功能。
- 问题3：LVS采用PC服务器，使用常用软件keepalived的VRRP心跳协议进行主备部署，其性能无法扩展。
- 问题4：LVS常用管理软件keepalived的配置和健康检查性能不足。

四层负载均衡LVS定制化功能

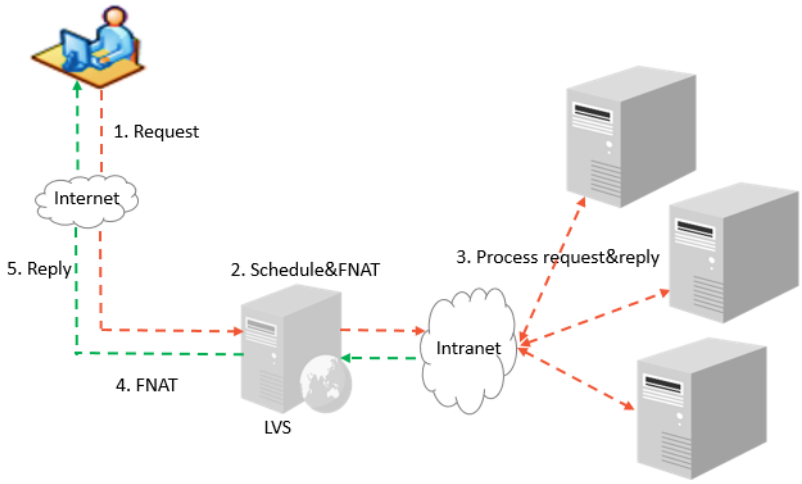
为了解决上述问题，阿里巴巴在官方LVS基础上进行了定制化。 Ali-LVS开源地址*https://github.com/alibaba/LVS*。

- 定制1：新增转发模式FULLNAT，实现LVS和Real Server间跨vlan通讯。
- 定制2：新增synproxy等攻击TCP标志位DDoS攻击防御功能。
- 定制3：采用LVS集群部署方式。
- 定制4：优化keepalived性能。

定制1：FULLNAT技术

- FULLNAT实现主要思想：引入local address（内网IP地址），cip-VIP转换为lip-rip，而lip和rip均为IDC内网IP，可以跨vlan通讯。
- IN/OUT的数据流全部经过LVS。为了保证带宽，采用万兆（10G）网卡。
- FULLNAT转发模式，当前仅支持TCP协议。

图 17-5: FULLNAT转发



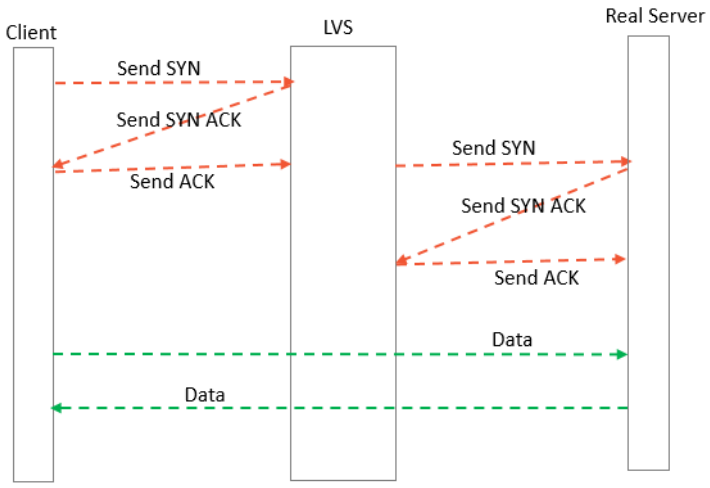
定制2：SYNPROXY技术

LVS针对TCP标志位DDoS攻击和Synflood攻击，利用synproxy模块进行防御。实现主要思想：参照Linux TCP协议栈中syncookies的思想，LVS代理TCP三次握手。

代理过程如下：

1. Client发送syn包给LVS。
2. LVS构造特殊seq的synack包给client，client回复ack给LVS。
3. LVS验证ack包中ack_seq是否合法；如果合法，则LVS再和Realserver建立3次握手。

图 17-6: LVS代理TCP三次握手



针对ACK、 FIN和RST Flood攻击，LVS查找连接表，如果不存在，则直接丢弃。

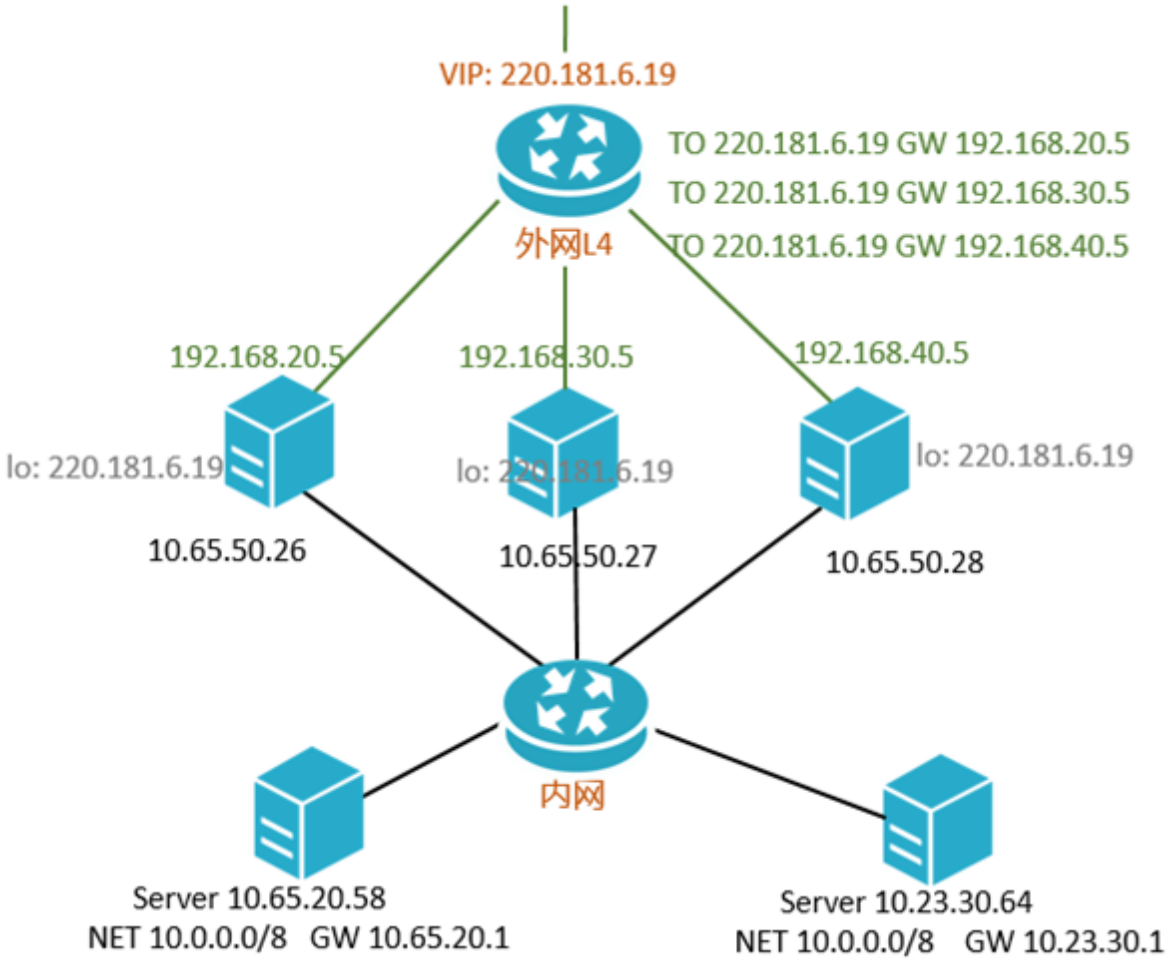
定制3：集群部署方式

LVS集群部署方式实现的主要思想：LVS和上联交换机间运行OSPF协议，上联交换机通过ECMP等价路由，将数据流分发给LVS集群，LVS集群再转发给业务服务器。

集群部署方式极大地保证了异常情况下负载均衡服务的稳定性。

- 健壮性：LVS和交换机间运行OSPF心跳。一个VIP配置在集群的所有LVS上，当一台LVS不可用时，交换机会自动发现并将其从ECMP等价路由中剔除。
- 可扩展：如果当前LVS集群无法支撑某个VIP的流量，LVS集群可以进行水平扩容。

图 17-7: 集群部署



定制4：keepalived优化

对LVS管理软件keepalived进行了全面优化，包括：

- 优化了网络异步模型，select改为epoll方式。
- 优化了reload过程。

四层负载均衡的特点

综上所述，四层负载均衡有如下特点：

- 高可用：LVS集群保证了冗余性，无单点。
- 安全：LVS自生攻击防御+云盾，提供了近实时防御能力。
- 健康检查：对后端ECS进行健康检查，自动屏蔽异常状态的ECS，待该ECS恢复正常后自动解除屏蔽。

17.4 七层负载均衡Tengine技术特点

Tengine是阿里巴巴发起的Web服务器项目，其在Nginx的基础上，针对大访问量网站的需求，添加了很多高级功能和特性。Nginx是当前最流行的7层负载均衡开源软件之一。Tengine开源地址*http://tengine.taobao.org/*。

Tengine定制化功能

针对云计算场景，Tengine定制的主要特性如下：

- 继承Nginx-1.4.6的所有特性，100%兼容Nginx的配置。
- 动态模块加载（DSO）支持。加入一个模块不再需要重新编译整个Tengine。
- 更加强大的负载均衡能力，包括一致性hash模块、会话保持模块，还可以对后端的服务器进行主动健康查，根据服务器状态自动上线下线。
- 监控系统的负载和资源占用从而对系统进行保护。
- 显示对运维人员更友好的出错信息，便于定位出错机器。
- 更强大的防攻击（访问速度限制）模块。

七层负载均衡特点

采用Tengine作为负载均衡的基础模块，阿里七层负载均衡产品有如下特点：

- 高可用：Tengine集群保证了冗余性，无单点。
- 安全：多维度的CC攻击防御能力。
- 健康检查：对后端ECS进行健康检查，自动屏蔽异常状态的ECS，待该ECS恢复正常后自动解除屏蔽。
- 支持7层会话保持功能。
- 支持一致性hash调度。

18 专有网络VPC

18.1 什么是专有网络VPC

专有网络VPC（Virtual Private Cloud）是一个隔离的网络环境，专有网络之间逻辑上彻底隔离。

您可以完全掌控自己的专有网络，例如选择IP地址范围、配置路由表和网关等，您可以在自己定义的专有网络中使用阿里云资源如ECS、RDS、SLB等。您可以将专有网络连接到其他专有网络或本地网络，形成一个按需定制的网络环境，实现应用的平滑迁移上云和对数据中心的扩展。

组成部分

每个VPC都由一个私网网段、一个路由器和至少一个交换机组成。

- 私网网段

  网段是专有网络的私网地址范围，所有部署在该VPC内的云资源的IP地址都在指定的私网网段内。在创建专有网络和交换机时，您需要以CIDR地址块的形式指定专有网络使用的私网网段。

  您可以使用下表中标准的私网网段及其子网作为VPC的私网地址。

网段	可用私网IP数量 （不包括系统保留地址）
192.168.0.0/16	65,532
172.16.0.0/12	1,048,572
10.0.0.0/8	16,777,212

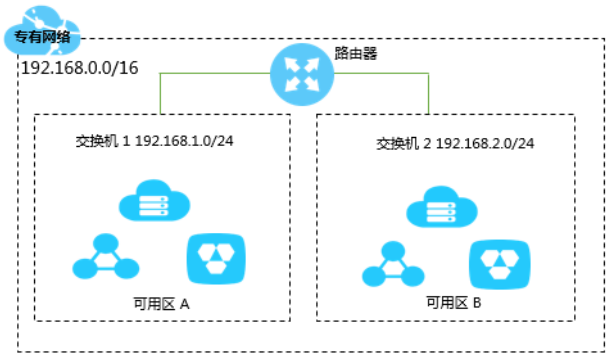
- 路由器

  路由器（VRouter）是专有网络的枢纽。作为专有网络中重要的功能组件，它可以连接VPC内的各个交换机，同时也是连接VPC和其他网络的网关设备。每个专有网络创建成功后，系统会自动创建一个路由器。每个路由器关联一张路由表。

- 交换机

  交换机（VSwitch）是组成专有网络的基础网络设备，用来连接不同的云产品实例。创建专有网络之后，您可以通过创建交换机为专有网络划分一个或多个子网。同一专有网络内的不同交换机之间内网互通。您可以将应用部署在不同可用区的交换机内，提高应用的可用性。

图 18-1: 专有网络



18.2 产品架构

基于目前主流的隧道技术，专有网络（Virtual Private Cloud，简称VPC）隔离了虚拟网络。每个VPC都有一个独立的隧道号，一个隧道号对应着一个虚拟化网络。

背景信息

随着云计算的不断发展，对虚拟化网络的要求越来越高，比如弹性（scalability）、安全性（security）、可靠性（reliability）和私密性（privacy），并且还有极高的互联性能（performance）需求，因此催生了多种多样的网络虚拟化技术。

比较早的解决方案是将虚拟机的网络和物理网络融合在一起，形成一个扁平的网络架构，例如大二层网络。随着虚拟化网络规模的扩大，这种方案中的ARP欺骗、广播风暴、主机扫描等问题会越来越严重。为了解决这些问题，出现了各种网络隔离技术，把物理网络和虚拟网络彻底隔开。其中一种技术是用户之间用VLAN进行隔离，但是VLAN的数量最大只能支持到4096个，无法支撑公共云的巨大用户量。

原理描述

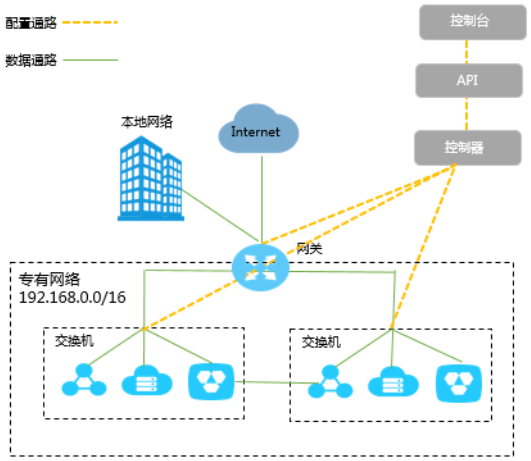
基于目前主流的隧道技术，专有网络（Virtual Private Cloud，简称VPC）隔离了虚拟网络。每个VPC都有一个独立的隧道号，一个隧道号对应着一张虚拟化网络。一个VPC内的ECS之间的传输数据包都会加上隧道封装，带有唯一的隧道ID标识，然后送到物理网络上进行传输。不同VPC内的ECS因为所在的隧道ID不同，本身处于两个不同的路由平面，所以不同VPC内的ECS无法进行通信，天然地进行了隔离。

基于隧道技术，阿里云的研发团队自研了交换机，软件自定义网络（Software Defined Network，简称SDN）技术和硬件网关，在此基础上实现了VPC产品。

逻辑架构

如下图所示，VPC包含交换机、网关和控制器三个重要的组件。交换机和网关组成了数据通路的关键路径，控制器使用自研的协议下发转发表到网关和交换机，完成了配置通路的关键路径。整体架构里面，配置通路和数据通路互相分离。交换机是分布式的结点，网关和控制器都是集群部署并且是多机房互备的，并且所有链路上都有冗余容灾，提升了VPC产品的整体可用性。

图 18-2: 专有网络架构



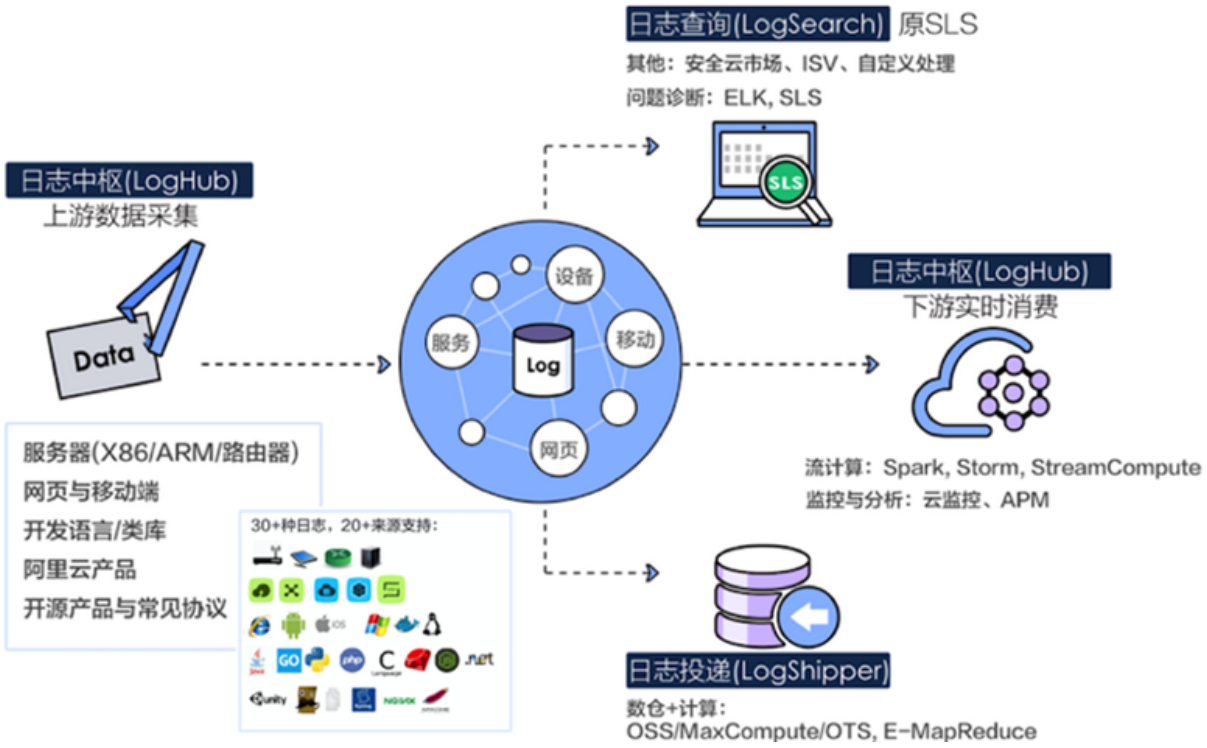
19 日志服务

19.1 什么是日志服务

日志服务LOG（Log Service）是针对日志类数据场景的一站式解决方案，解决海量日志数据采集与订阅、日志投递与查询功能。

- 实时采集与订阅：通过客户端、API、Tracking JS、Library 等手段实时采集来自多个渠道的日志数据。数据写入后，可以进行实时订阅读取。例如通过 Spark Streaming、Storm、Consumer Library 等接口对数据进行实时处理。
- 日志查询：实时索引日志数据，并提供实时、海量存储查询引擎。可以基于时间、关键词、上下文等任意维度进行日志查询。

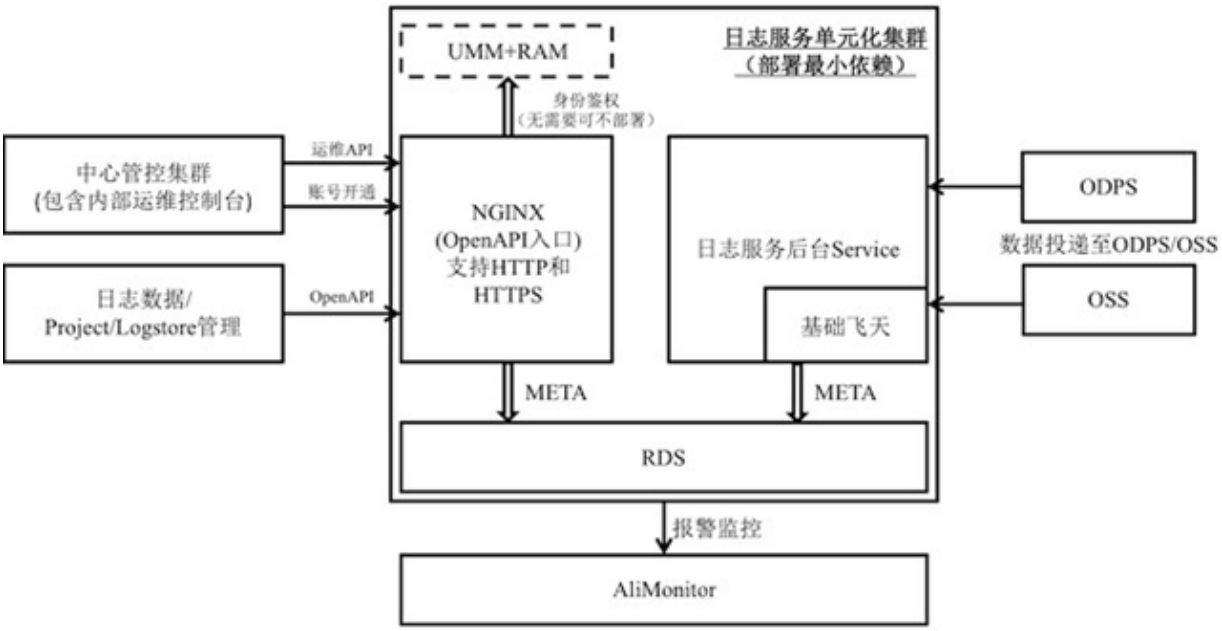
日志服务提供弹性伸缩、自动扩容等能力，可以水平支持 PB 级数据。



19.2 系统架构

日志服务的架构如下图所示。

图 19-1: 产品架构



- 左侧为控制台、OpenAPI 等，用以和外部交互。
- 中间为系统核心模块，主要为：
  - # UMM-RAM 账号
  - # RDS 存储元数据
  - # Nginx 为前端服务器
  - # 日志服务后台是后端业务服务器

19.3 产品组件

Logtail

帮助您快速收集日志的Agent。其特点如下所示：

- 基于日志文件、无侵入式的收集日志
  - # 只读取文件。
  - # 采集过程无侵入。
- 安全、可靠
  - # 支持文件轮转不丢失数据。
  - # 支持本地缓存。
  - # 网络异常重试。

- 方便管理

  # 支持Web端。

  # 支持可视化配置。

- 完善的自我保护

  # 实时监控进程对CPU、内存资源的消耗。

  # 限制进程的资源使用上限。

前端服务器

采用LVS + Nginx构建的前端机器。其特点如下所示：

- HTTP、REST协议
- 水平扩展

  流量上涨时可快速通过增加前端机来提高处理能力。

- 高吞吐、低延时

  # 纯异步处理，单个请求异常不会影响其他请求。

  # 内部采用专门针对日志的Lz4压缩，提高单机处理能力，降低对网络带宽要求。

后端服务器

后端是分布式的进程，部署在多个机器上，完成实时对Logstore数据的存储、索引、查询整体后端服务的特点如下所示：

- 数据高安全性

  # 您写入的每条日志，都会被保存3份。

  # 任意磁盘损坏、机器宕机情况下，自动复制数据、修复磁盘。

- 稳定服务

  # 进程崩溃和机器宕机时，Logstore会自动迁移。

  # 自动负载均衡，确保无单机热点。

  # 严格的Quota限制，防止单个用户行为异常对其他用户产生影响。

- 水平扩展

  # 以分区（Shard）为单位进行水平扩展。

  # 用户可以按需动态增加分区来增加吞吐量。

19.4 功能特性

日志实时采集（LogHub）

实时采集。通过多种方式实时采集海量数据、下游实时消费。

- 使用 Logtail 采集日志：稳定可靠、安全、全平台（Linux、Windows、Docker）、高性能、低资源占用。
- 通过 API/SDK 采集日志：灵活方便，可扩展，支持移动端和多种开发语言。
- 云产品日志采集：支持云服务器（Elastic Compute Service，ECS）等云产品的日志接入。一键打通，便捷高效。
- 其他日志采集方式：Syslog、Unity3D、Logstash、Log4j、Nginx 等。

日志实时消费（LogHub）

流计算、协同消费库、多语言支持。

- 功能完善：覆盖 Kafka 100% 功能，并提供保序、弹性伸缩、根据时间段跳转等功能。
- 稳定可靠：写入即可消费；数据多份拷贝；快速弹性伸缩；低成本。
- 使用便捷：支持 Spark Streaming、Storm、Consumer Library（一种自动负载均衡的编程模式）、SDK 订阅等。

日志查询（LogSearch）

实时索引、查询数据。对日志数据创建索引，提供基于时间、关键词进行检索。

- 大规模：PB 级实时索引（写入 1 秒内即可查）；日志查询量每秒可达十亿级以上。
- 查询灵活：支持关键词、模糊、跨 Topic 查询、上下文查询。

19.5 产品价值

帮助您快速构建面向海量日志数据的解决方案。

解决了以下场景中的问题：数据采集、实时计算、数仓与离线分析、产品运营与分析、运维与管理等场合。

- 数据收集与消费（Data Collection & Consumption）
- 数据清洗与流计算（ETL/Stream Processing）
- 事件溯源（Event Sourcing/Tracing）
- 日志管理（LogManagement）

20 资源编排

20.1 什么是资源编排服务

资源编排服务ROS（Resource Orchestration Service）可以帮助用户简化云计算资源管理和自动化运维的服务。用户遵循资源编排定义的模板规范，编写资源栈模板。在模板中，您定义所需的云计算资源、资源间的依赖关系、资源配置（如 ECS 实例、 RDS 数据库实例）等，资源编排将根据模板为您创建和配置这些资源。

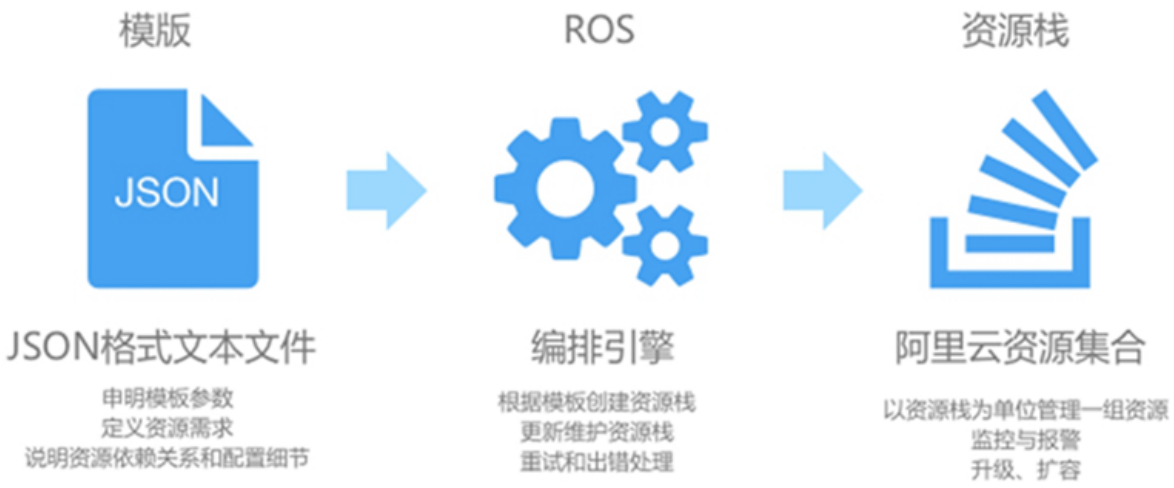
资源编排通过编排引擎自动完成所有资源的创建和配置，以达到自动化部署、运维的目的。资源编排模板是一种用户可读、易于编写的文本文件。您可以直接编辑 JSON 格式文本，也可以使用资源编排控制台提供的可视化编辑器，更为直观地编辑模板。您可以随时编辑修改模板。通过 SVN、Git 等版本控制工具可以控制模板的版本，以达到控制基础设施版本的目的。也可以通过 API、SDK 等方式把资源编排的编排能力与自己的应用整合，做到基础设施即代码（Infrastructure as Code）。

资源编排模板也是一种标准化的资源和应用交付方式。如果您是独立软件供应商（ ISV），您可以通过资源编排模板交付包含云资源和应用的整体系统和解决方案。ISV 可以通过这种交付方式，整合云的资源和 ISV 的软件系统，达到统一交付的目的。

资源编排服务是通过资源栈 （Stack）这种逻辑集合来统一管理一组云资源（一个资源栈即为一组云资源），所以，对于云资源的创建、删除、克隆等操作，都可以以资源栈为单位来完成。在 DevOps 实践中，可以轻松地克隆开发、测试、线上环境。同时，也可以更容易实现应用的整体迁移和扩容。

综上所述，通过使用资源编排模板，创建资源栈的流程如下图。

图 20-1: 创建资源栈流程



20.2 功能特性

ROS是云计算中很重要的一个服务，您的整个基础架构全部包含在编辑好的ROS模板中，无论何时何地通过ROS就能在云上构建出自己的基础架构，正真做到基础设施即代码（Infrastructure is Code）。与直接调用各产品的OpenAPI相比，大大提高了您展开业务的效率。

ROS层面所看到都是堆栈，在堆栈之下是资源，这些资源也可以通过各产品的控制台访问。通过ROS的控制台可以操作ROS的堆栈和堆栈的一组资源。

ROS专有云控制台一般会包括：

- **资源栈管理**

  提供您已经创建的资源栈的概览信息，可以查看堆栈的详细信息，浏览堆栈的资源信息、事件信息以及原始模板。以及可适用于某一个堆栈的操作，现在支持的操作有：重新创建、健康检查、更新堆栈。重新创建是指用该堆栈的原始模板重新创建一个堆栈，可以指定不同的参数。健康检查是指检查该栈中的资源状态是否可用；更新堆栈是指通过修改原始模板，更新堆栈中的资源。

- **新建资源栈**

  通过ROS模板创建一个全新的堆栈。

- **资源类型**

  列出ROS当前所支持的所有的资源类型。

- **ECS实例相关信息**

当前各可用区中ECS所支持的规格、镜像，通过单击ECS规格行中的**创建**按钮，ROS可以很快的创建出您想要的ECS资源。

20.3 产品价值

ROS使用灵活、简便、不需要关注个产品的调用逻辑，可为您节省很高的开发运维成本。

- 模板简单易懂；
- 实现资源关系的编排；
- 实现基础架构一键交付；
- 资源安组运维成本小；
- 资源更新操作简便快捷。

下面表格列出了ROS和传统OpenAPI相比所具有的优势。

表 20-1: ROS与传统OpenAPI对比表

对比项	ROS	openAPI
部署交付周期	快速编辑模板	开发调试数天
配置动态伸缩	修改模板实现伸缩	开发调试数天
底层API升级	修改模板实现	开发调试数天
组资源更新	修改模板实现	开发调试数天
可迁移性	相同模板直接创建	开发调试

ROS具有灵活性、方便性且低成本的特点。使您有更多时间关注自己的核心业务，做到基础设施即代码（Infrastructure is Code）。在DevOps实践中，可以很轻松的克隆开发、测试、线上环境，同时，也可以更容易做到应用的整体迁移和扩容。

21 云盾

21.1 什么是云盾

专有云云盾是从网络安全、服务器安全、应用安全、数据安全、安全管理和安全服务等多维度防护云上安全的一套专有云安全解决方案。

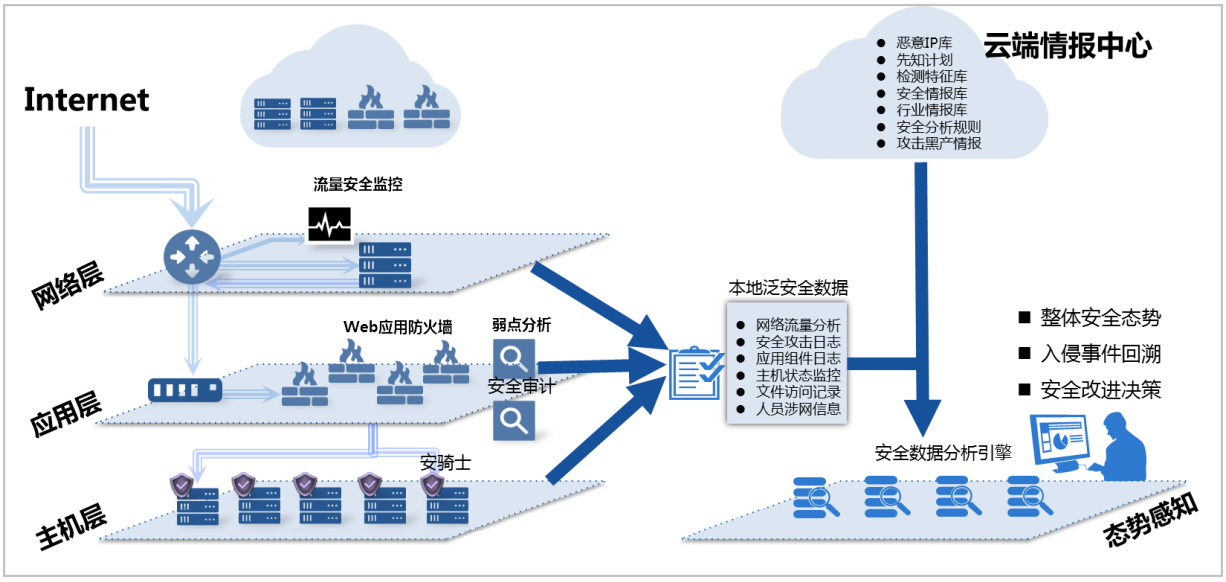
在云计算环境下，随着技术的发展，传统以检测技术为主的边界安全防护不能充分保障云上业务的安全。专有云云盾结合云计算平台强大的数据分析能力以及专业的安全运营团队，从网络层、应用层、主机层等多个层面为用户提供一体化的安全防护服务。

专有云云盾是一款适用于核心业务应用对外的互联网化防护体系，能够为您提供DDoS检测/防御、Web层攻击检测/防御、Web漏洞发现/修复、主机漏洞发现/修复、主机防入侵的实时防护能力。通过现网获取的丰富本地泛安全数据与云端情报，统一在安全数据分析引擎集群中进行安全大数据分析，为安全管理员呈现整体安全态势、入侵事件回溯，包括针对性攻击发现、人员情报泄漏预警、入侵原因分析等。通过这些核心安全信息的分析展现，安全管理员不仅能够了解安全状况，还可以借助安全数据分析引擎开放的自定义分析界面对已有安全数据进行场景化分析，实现安全分析能力的灵活定制。

21.2 产品架构

云盾标准版的产品结构如图 *21-1:* 云盾标准版产品结构图所示。

图 21-1: 云盾标准版产品结构图



- **流量安全监控**： 在专有云的网络边界，通过流量镜像的方式对出入专有云的所有网络流量进行逐包检测分析。同时，分析结果将作为云盾其他防护模块的参考依据。
- **主机入侵检测**： 通过在物理服务器上部署客户端进行信息搜集和检测，实时监测专有云环境中所有物理服务器主机的安全状态，及时发现文件篡改、异常进程、异常网络连接、可疑端口监听等异常行为，帮助用户及时发现服务器安全隐患。
- **安骑士**： 通过日志监控、文件分析、特征扫描等手段，为云服务器（ECS）提供漏洞管理、基线检查、入侵检测、资产管理等安全功能。
- **安全审计**： 收集专有云平台内的数据库日志、主机日志，用户侧控制台操作日志、运维侧控制台操作日志，和网络设备日志，具备完整的日志收集、存储、分析、报警等功能。
- **Web应用防火墙（WAF）**： 保护网站的应用程序避免遭受常见Web漏洞的攻击，防御SQL注入、XSS跨站脚本、常见Web服务器插件漏洞、木马上传、非授权访问等OWASP常见攻击，过滤海量恶意访问，避免网站数据泄露，保障网站的安全性与可用性。
- **态势感知**：汇集网络流量、以及主机端信息，通过机器学习和数据建模发现潜在的入侵和攻击威胁，从攻击者的角度有效捕捉高级攻击者发起的漏洞攻击、新型病毒攻击事件，并有效展示正在发生的安全攻击行为，实现业务安全可视和可感知。

除上述安全产品外，云盾标准版还包含安全运营驻场服务。专有云安全运营驻场服务帮助用户更好地利用专有云产品及云盾产品的安全特性，保障租户层应用安全。安全运营服务包括上线前安全评估、安全访问控制策略管理、云盾产品接入配置、周期性安全评估、日常安全巡检、安全应急等一系列服务内容，全面覆盖专有云平台租户业务的云上安全生命周期。通过安全运营驻场服务，帮助用户梳理并建立云上安全运营体系，全面提升应用系统安全性，保障用户业务的安全和稳定运行。

同时，专有云云盾提供以下可选安全产品供用户根据自身安全需求进行选择，进一步完善专有云安全体系：

- **DDoS流量清洗**： 对DDoS攻击流量进行检测及过滤，防御DDoS流量攻击。
- **云防火墙**：基于业务可视化的结果进行业务梳理和业务隔离，实现专有云环境中东西向流量的安全访问控制。
- **堡垒机**：通过身份管理、授权管理、双因子认证、实时会话监控与切断、审计录像回放、高危指令查询等功能，为专有云平台的物理服务器或云服务器（ECS）的运维提供完整的审计回放和权限控制服务。
- **数据库审计**：实现对云端自建数据库、RDS数据库访问的精确审计，以及准确的应用用户关联审计，并具备风险状况、运行状况、性能状况、语句分布的实时监控能力。

- **数据发现与脱敏**：通过扫描IP段设备流量信息检测数据资产，发现数据库分布；通过系统内置发现规则发现敏感数据，并对敏感数据分级分类；通过对资产SQL语句量和会话并发量判断资产使用热度，根据流量统计发现静默资产；通过数据库授权发现资产权限分布及权限详情；通过对敏感数据进行数据抽取、数据漂白、和动态掩码的脱敏处理，满足生产数据面向测试、开发、培训和数据共享场景的数据安全需求，实现“用”、“护”结合。

21.3 功能特性

21.3.1 云盾标准版

21.3.1.1 流量安全监控

流量安全监控模块是阿里云安全团队自主研发的毫秒（ms）级攻击监控产品。通过对专有云入口镜像流量包的深度解析，实时地检测出各种攻击和异常行为，并与其他防护模块联动防护。同时，流量安全监控模块在整个云盾防御体系中，提供了丰富的信息输出与基础的数据支持。

功能模块

流量安全监控模块提供以下功能：

功能项	功能说明
流量统计分析	通过流量镜像方式旁路对进出互联交换机（ISW）的流量进行统计，生成流量图。
异常流量检测	通过流量镜像方式旁路检测超过阈值的异常流量，并牵引到DDoS流量清洗产品进行清洗。阈值支持设置流速（Mbps）、包速（PPS）、HTTP请求速率（QPS）、新建连接数等参数。
恶意主机识别	针对专有云内部的恶意主机对外发起的攻击行为进行检测，发现内部已经被控制的云服务器。
Web应用攻击防护	根据内嵌的Web应用攻击检测规则，对常见的Web应用攻击进行网络层拦截旁路阻断，防护包括SQL注入、代码执行、命令执行、脚本木马、文件包含、上传漏洞利用、常见CMS漏洞利用等Web攻击。
异常TCP连接阻断	通过旁路向服务端和客户端发送TCP Reset报文，阻断四层TCP连接。
网络日志记录	记录四层UDP、TCP流量日志，HTTP请求的Request和Response日志，供态势感知模块进行大数据模型分析。

工作原理

流量安全监控模块对流量进行深度包分析，实时检测各种攻击和异常行为，同时将安全事件上报到专有云云安全中心，与其他防护系统产生联动。在整个专有云云盾中，流量安全监控模块提供丰富的信息输出与基础的数据支撑。

流量安全监控模块的数据处理分为收集、汇总、输出三个步骤，处理过程之间使用Socket协议进行数据交换。

- 收集：由多台安装双端口万兆网卡的高性能 PC完成流量数据的收集。
- 汇总：一个 IP 的流量可能经过多个收集器，因此必须汇总后才能输出有效信息。
- 输出：将汇总后的流量数据存储并完成信息输出。

21.3.1.2 主机入侵检测

主机入侵检测模块通过在物理服务器主机上部署的客户端进行信息搜集和检测，实时检测专有云环境中所有物理服务器主机，及时发现文件篡改、异常进程、异常网络连接、可疑端口监听等行为，帮助用户及时发现主机安全隐患。

功能模块

主机入侵检测模块提供以下功能：

功能项	功能说明
关键目录完整性检测	监控主机系统特定目录（*/etc/init.d*）中文件的完整性，及时发现篡改行为并进行告警。
异常进程告警	及时发现异常进程启动并进行告警，支持对XOR DDoS木马、Bill Gates DDoS木马、Minred挖矿程序等异常进程的检测。
异常端口告警	及时发现新建的端口监听并进行告警。
异常网络连接告警	及时发现主动外连公网的网络连接并进行告警。

工作原理

部署在物理服务器上的客户端Agent实时收集物理服务器上的关键目录、进程、端口开放、网络连接等信息，通过规则、特征匹配的方式发现服务器中的异常行为，上报相关信息并对异常事件进行告警。

21.3.1.3 安骑士

安骑士模块通过日志监控、文件分析、特征扫描等手段，为云服务器ECS提供漏洞管理、基线检查、入侵检测、资产管理等安全防护措施。安骑士模块包含客户端和服务器端，客户端配合服务器端监测针对主机系统层和应用层的攻击行为及漏洞信息，实时防护主机安全。

功能模块

安骑士模块提供以下功能：

功能分类	功能项	功能说明
漏洞管理	Linux软件漏洞检测	通过检测云服务器上安装软件的版本信息，与CVE官方的漏洞库进行匹配，检测出存在漏洞的软件（包括SSH、OpenSSL、MySQL等软件漏洞），并推送漏洞信息，提供修复建议。
	Windows漏洞检测与修复	通过订阅微软官方更新源，检测云服务器中存在的未修复的高危官方漏洞，并推送微软官方补丁进行修复（如“SMB远程执行漏洞”）。 **说明：** 默认系统只推送高危漏洞，安全更新和低危漏洞支持手动更新。
	Web-CMS漏洞检测与修复	同步阿里云安全情报源，通过目录及文件的检测方案检测Web-CMS软件漏洞，提供云盾自研补丁（修复如Wordpress、Discuz等软件漏洞），且支持漏洞修复及回滚操作。
	配置型、组件型的漏洞检测	检测无法通过版本匹配和文件判断方式发现的漏洞，精准识别软件高危配置漏洞，例如Redis未授权访问、ImageMagick等配置型、组件型漏洞。
基线检查	账号安全基线检查	• 检测SSH、RDP、FTP、MySQL、PostgreSQL、SQLServer服务中存在的弱口令账号。 • 检测云服务器中存在的可疑的隐藏账号、克隆账号等风险账号。 • 检测Linux系统服务器中的密码策略合规性。 • 检测云服务器中存在的空密码账户。
	系统配置检测	对系统组策略、登录基线策略、注册表配置风险进行检测，包括：

功能分类	功能项	功能说明
		• 检测Linux系统服务器的定时任务中是否包含可疑的自启动项目。 • 检测Windows系统服务器中的自启动项。 • 检测系统共享配置。 • 检测Linux系统服务器的SSH登录安全策略配置。 • 检测Windows系统服务器中账号相关的安全策略。
	数据库安全基线检查	检测服务器上的Redis服务是否对公网开放、是否存在未授权访问漏洞并向系统关键文件写入异常数据的情况。
	合规对标检测	按照CIS-Linux Centos7最新基线标准进行系统层面基线合规检测。
入侵检测-异常登录	异地登录告警	自动记录所有登录记录，通过分析和记录用户常用登录位置，识别常用的登录区域（精确到地市级），对疑似的非常用地登录行为进行告警，且支持自定义常用登录地配置。
	非白名单IP登录告警	配置登录白名单IP后，对来自非白名单IP的登录事件进行告警。
	非法时间登录告警	配置合法登录时间后，对非合法时间段内的登录事件进行告警。
	非法账号登录告警	配置合法登录账号后，对非合法账号的登录事件进行告警。
	暴力破解登录拦截	对非法暴力破解密码的异常登录行为进行识别，实时检测并拦截暴力破解攻击，避免被黑客多次猜解密码而导致入侵。支持对SSH和RDP服务的暴力破解行为进行监控。
入侵检测-网站后门查杀	网站后门（Webshell）查杀	通过自研网站后门查杀引擎对云服务器中存在的脚本后门进行精准查杀（支持配置定时查杀和实时防护扫描策略），识别包括以ASP、PHP、JSP编写的脚本后门文件，并支持手动对脚本后门文件进行隔离。
入侵检测-主机异常进程	进程异常行为检测	检测诸如反弹Shell、JAVA进程执行CMD命令、Bash异常文件下载等进程异常行为。

功能分类	功能项	功能说明
资产管理	资产分组	支持对云服务器进行最多四级分组，并可按照地域、在线状态等信息进行筛选，且支持资产标签管理功能。
	资产指纹	• **端口监听**：收集、展示端口监听信息，对变动进行记录，便于清点端口开放情况 • **账号管理**：收集账户及对应权限信息，清点特权账户、发现提权行为 • **进程管理**：通过进程快照信息的收集和呈现，清点合法进程、发现异常进程 • **软件管理**：清点软件安装信息，在高危漏洞爆发时快速定位受影响资产
主机日志	日志检索	• **进程相关日志**： # 进程启动：进程一旦启动，记录该启动事件的详细信息 # 进程快照：记录某一时刻的进程全量日志信息 • **网络相关日志**： # 五元组日志：从主机端和网络端同时采集的连接五元组信息 # Web日志：从网络上抓取的HTTP访问日志（暂不支持HTTPS） # DNS日志：对外请求的DNS解析日志（暂不支持内网DNS） • **其它日志**： # 系统登录：通过SSH、RDP方式的系登录流水日志 # 端口监听快照：某一时刻的所有对外监听端口的快照数据 # 账号快照：某一时刻的所有账号信息的快照数据

工作原理

安骑士模块采用Client-Server架构，安骑士客户端Agent安装在云服务器上。客户端Agent与安骑士服务器端通过TCP长连接进行通信，通过HTTP方式从服务器端获取需要的脚本、规则、安装包等文件。

安骑士客户端Agent分为Windows版本和Linux版本，客户端Agent自动连接到安骑士服务器端进行在线升级。

安骑士模块核心功能的工作原理如下：

- **漏洞管理**：安骑士客户端Agent通过收集云服务器的相关信息（如组件信息、软件版本、文件信息、注册表信息等），与由安骑士服务器端下发的漏洞检测规则进行匹配，一旦命中漏洞检测规则，则将相关信息上报至安骑士服务器端进行进一步匹配分析。检测发现的漏洞将通过控制台展示，用户通过控制台或者调用OpenAPI触发漏洞修复，安骑士服务器端将漏洞补丁下发至对应云服务器上的客户端Agent并由客户端Agent自动执行漏洞修复，并同步漏洞修复状态至安骑士服务器端。
- **基线检查**：安骑士服务器端根据用户手动触发或所设定的周期性检测策略，向云服务器上的客户端Agent下发基线检查请求，客户端Agent根据相应的检测策略收集主机信息并与安全基线进行比对，一旦发现不符合安全基线的检查项则标记为风险项并上报至安骑士服务器端。
- **异常登录**：安骑士客户端Agent实时监控云服务器主机系统的登录日志，在Linux系统主机中则监控*/var/log/secure*、*/var/log/auth.log*文件，记录并统计所有登录成功、失败事件。一旦发现符合非法登录或暴力破解行为的异常登录事件，上报至安骑士服务器端。
- **网站后门**：通过自研的网站后门动态识别引擎，安骑士客户端Agent有效检测各类复杂变形的Webshell，通过将其还原到可识别状态，分析其更隐蔽的后门行为，避免因只使用静态规则而导致被轻易绕过的现象。
- **主机异常进程**：安骑士服务端的数据分析规则引擎对安骑士客户端Agent收集到的云服务器上的主机进程数据进行分析，发现主机中的异常进程，包括反弹Shell、挖矿进程、DDoS木马、蠕虫病毒、黑客工具等恶意程序。
- **日志采集**：通过安骑士客户端Agent，收集云服务器的各类进程、网络日志等信息。

应用场景

安骑士模块适用于下列场景的主机安全防护：

- **使用通用软件建站的场景**

在这种场景下，极可能因通用软件的漏洞被利用而导致被黑客入侵，使用安骑士高级版进行漏洞检测，一旦发现漏洞可快速进行一键修复。

- **使用Web应用服务的场景**

  无论是内部Web服务还是外部Web服务，黑客都可能通过Web服务窃取网站的核心数据，使用安骑士高级版可有效阻止黑客从外部的攻击和内部的渗透。

21.3.1.4 安全审计

安全审计模块提供基于云计算平台的一体化审计解决方案。对标信息系统安全等级保护基本要求，安全审计模块从物理服务器层面、网络设备层面、云计算平台应用层面分别进行审计，实现行为日志的收集、存储、分析、报警等功能。

通过采集各个云产品、网络设备、主机、数据库等数据源的相关日志，并通过所设置的审计规则结合审计规则引擎完成对整个专有云的安全审计工作，对于触发规则的操作进行报警提示。同时，安全审计提供日志查询与规则配置功能，全面满足用户的等保需求。

功能模块

安全审计模块提供以下功能：

功能项	功能说明
审计一览	支持根据时间、数据库、网络、主机、用户操作、运维操作等多种维度查询并生成审计报表。通过报表反馈原始日志、审计事件、审计风险、日志用量、存储用量等审计系统运营情况。
原始日志	根据审计对象、审计类型、风险级别、时间、关键字等多种维度查询7天内的所有原始日志。
审计查询	根据审计对象、审计类型、风险级别、时间、关键字等多种维度查询触发审计规则的30天内的审计日志。
策略设置	• **审计策略**：查询已接入的云产品、主机、网络设备、数据库配置的审计规则，并支持审计规则的添加、修改、删除。 • **类型设置**：查询或添加新的审计类型。 • **告警设置**：根据审计规则、审计风险设定报警接收人。 • **存档管理**：查询、下载185天内的所有原始日志文件。 • **导出管理**：查询、管理日志导出任务。 • **系统设置**：配置审计系统的全局参数，包括每天报警次数、全天日志审计量、主机日志量、网络设备日志量、用户操作与运维操作日志量等参数的配置。

工作原理

安全审计模块收集各个数据源提供的日志并存储到日志服务。安全审计服务定时对存储在日志服务中的日志按照产品、日志类型进行消费，并通过审计规则引擎对审计日志进行判定，触发审计规则的操作将通过报警模块提示相关责任人并存档，而未触发规则的日志将直接进行持久化存档保存，便于将来随时查询使用。

功能优势

安全审计模块具有以下特点和优势：

- **行为日志全面无死角**

  覆盖专有云平台的多个业务和物理宿主机，从各个角度对行为进行收集，确保了不会因为覆盖面不够导致的审计缺失。日志收集中心集中、准实时、同步回收行为日志。

- **日志存储可靠**

  审计日志的存储基于云计算存储服务，通过集群化三备份保障存储安全稳定性，存储空间支持快速扩充。

- **海量数据实时查询**

  通过对海量日志数据构建全文索引，具备大量数据的快速检索查询能力。

21.3.1.5 Web应用防火墙

Web应用防火墙（简称WAF），是面向云上租户Web应用的安全防护系统，有效防御常见Web攻击。这类攻击既有诸如SQL注入、XSS跨站脚本等常见Web应用攻击，也有CC攻击这种影响网站可用性的资源消耗型攻击。同时，WAF模块支持根据网站实际业务制定精准的防护策略，过滤对用户网站有恶意针对性的Web请求。

WAF防护的流量定位在HTTP/HTTPS的网站业务上，支持用户在WAF的管理控制台中自主导入证书与私钥，从而实现业务的全链路加密，避免数据在链路中被监听的可能，从而满足对于HTTPS业务的安全防护需求。

功能模块

Web应用防火墙模块提供以下功能：

功能项	说明
Web常见攻击防护	防护SQL注入、XSS跨站脚本、文件上传、文件包含、常见目录遍历、常见CMS漏洞、代码执行注入、脚本后门攻击、扫描器攻击等常见Web攻击类型。

功能项	说明
	针对Web攻击提供观察和阻断两种模式： • 观察模式对攻击告警但不立刻阻断，便于评估误报现象。 • 阻断模式则直接拦截阻断带有攻击的请求。
缓解CC攻击	支持对请求URL的频率及访问地址分布、异常响应码等信息进行统计，拦截异常行为。 针对CC攻击提供正常模式和攻击紧急模式两种CC防护策略。在CC攻击导致网站不可访问时，可启用攻击紧急模式加强CC攻击防护力度，缓解CC攻击。 CC安全防护除了系统规则防护，还支持对单一源IP的URL访问进行自定义规则设置。
精准访问控制	提供友好的配置界面，支持IP、URL、Referer、User-Agent等HTTP常见字段的条件组合，打造强大的精准访问控制策略，并支持盗链防护、网站后台保护等防护场景。
恶意IP自动封禁	当某IP持续对域名发起Web攻击时，可自动封禁该IP一段时间。
地区封禁	提供基于地理位置的区域封禁能力，可对指定的省份或者海外地区来源IP进行一键封禁。

工作原理

WAF通过域名转发配置的方式，让网站用户的访问请求先经过WAF集群，由WAF对访问请求进行检测、过滤、清洗后，通过反向代理的方式将正常请求转发回源站应用服务器，实现对源站服务器的应用层安全防护。

应用场景

WAF的使用场景包括政府、金融、保险、电商、O2O、互联网+、游戏等各类网站的Web应用安全防护，主要用于解决以下问题：

- 防数据泄密，避免因黑客的注入攻击导致网站核心数据被拖库泄露。
- 防CC攻击，通过阻断海量的恶意请求，保障网站可用性。
- 阻止木马上传，防止网页被篡改，保障网站的公信力。
- 提供虚拟补丁，针对网站被曝光的最新漏洞，最大可能提供快速修复的规则。

21.3.1.6 态势感知

态势感知模块是一款由阿里云安全团队自主研发的大数据安全分析系统。通过机器学习和数据建模对专有云环境中主机流量和网络流量进行深度解析，检测各种威胁、攻击、访问等异常行为，从攻

击者的角度有效捕捉高级攻击者使用的漏洞攻击、新型病毒攻击事件，有效展示正在发生的安全攻击行为，实现业务安全可视和可感知。

功能模块

态势感知模块提供以下功能：

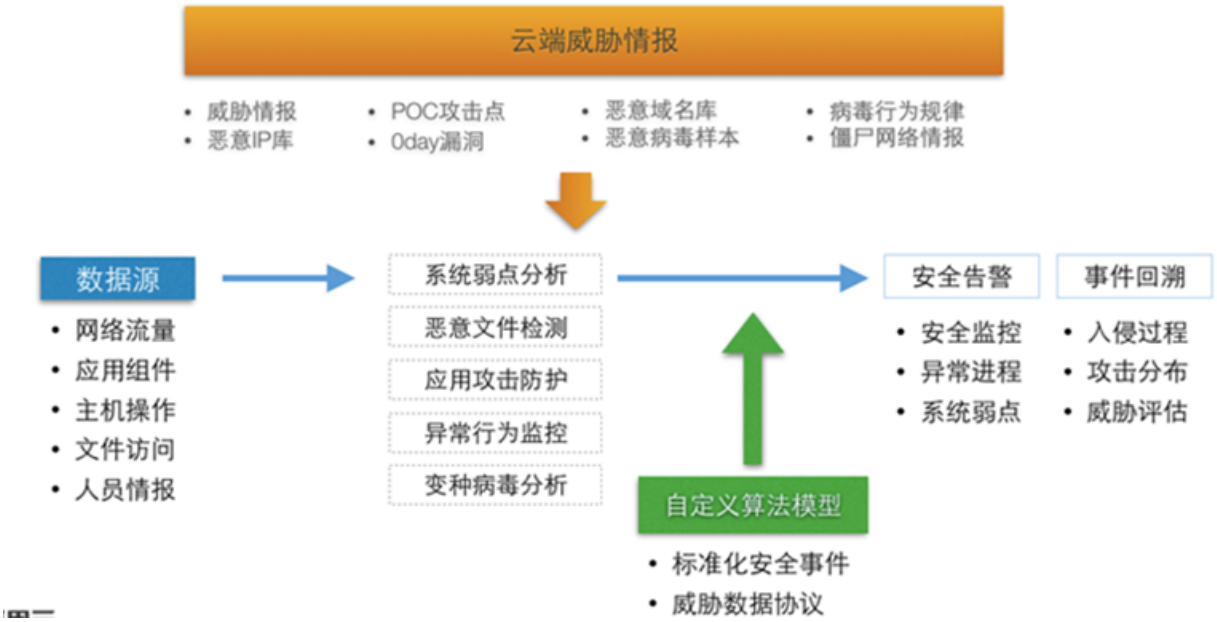
功能项	功能说明
安全态势总览	提供安全整体态势信息，包括紧急事件数量、今日攻击、今日弱点、安全攻击趋势、最新威胁分析展示、最新情报展示、防护资产情况等。
访问分析	全量分析防护范围内Web业务的外部访问情况，包括TOP10被访问业务、正常访问IP数、恶意访问IP数、爬虫访问IP数以及部分访问细节样例。
可视化大屏	基于态势感知的大屏展示功能，提供包括基于地图的流量安全监控大屏及展示主机业务安全状态的安全态势大屏。
安全事件分析	基于大数据算法模型，识别检测云环境中以下安全事件： • 肉鸡行为：主机被黑客控制沦为肉鸡并对外发起DDoS攻击等行为。 • 暴力破解成功：主机遭受暴力破解攻击且被恶意攻击者成功登录。 • 后门：主机中存在WannaCry勒索软件、MySQL服务异常SQL脚本、Webshell后门等。 • DDoS攻击：主机遭受DDoS攻击。 • 黑客工具：主机登录凭证遭窃取，发现主机上残留的黑客工具及黑客攻击行为。 • 异常网络连接：利用PowerShell下载可疑文件、执行VBScript异常命令、Linux系统主机下载恶意文件或脚本、反射Shell行为。 • 网络流量异常：挖矿程序运行。
流量统计分析	针对监控范围内的流量进行统计，包含今日流量、30天维度、90天维度的统计展现可针对某单一IP进行展现，并能够对访问QPS进行频次统计。
恶意主机识别	对内部恶意主机对外的攻击行为进行监测，发现内部已被控制的主机，包括CC攻击发现、DDoS攻击指令发现。
Web攻击检测	检测Web漏洞利用、黑客扫描工具、上传/连接webshell、SQL注入、XSS攻击、本地/远程文件包含、代码/命令执行等攻击。
服务器漏洞利用	基于报文特征，将特征字符转换成二进制进行匹配，发现如Redis服务漏洞被利用等安全事件。

功能项	功能说明
应用漏洞分析	针对Web应用层漏洞进行扫描，并提供扫描结果的验证手段与相应的修复建议。周期性结合NAT资产、主机资产自动进行扫描，对已经发现的应用漏洞进行统一验证，更新应用漏洞状态。
弱口令分析	针对Web、SSH、FTP、MySQL和SQLServer等通用系统的账号进行弱口令扫描，并支持添加自定义弱口令，每天定时结合NAT资产、主机资产自动进行扫描，对已经发现的弱口令进行统一验证，更新弱口令发现时间。
配置项检测	根据对外业务页面的访问情况进行扫描发现，对Web页面配置项的泄露进行告警，并在每天定时对已经发现的配置项泄露情况进行验证，更新发现时间。

工作原理

态势感知模块功能实现原理如图 *21-2: 态势感知功能原理图*所示。

图 21-2: 态势感知功能原理图



- **大数据安全分析平台**

  # **网络侧**：态势感知将流量安全监控模块采集到的HTTP数据（Request和Response）整合成完整的HTTP日志，通过大数据模型进行计算分析，得到事件和威胁结果。

  # **主机侧**：态势感知通过规则引擎将安骑士模块采集到主机进程信息进行比对分析，得到事件和威胁结果。

- **安全事件**：态势感知综合以下三方面的安全事件，为用户集中展示所有安全事件信息。

  # 安骑士模块上报的安全事件

  # 规则引擎分析主机进程信息所得到的主机侧安全事件

  # 大数据模型分析网络HTTP日志所得到的网络侧安全事件

- **弱点分析**：弱点分析功能涉及Cactus-batch和Cactus-keeper两个模块。

  # Cactus-batch模块负责数据的处理，将需要扫描的URL处理后通过消息队列发送给Cactus-keeper模块。

  # Cactus-keeper模块集成扫描引擎，基于阿里云多年积累沉淀的扫描规则和插件库，对系统的漏洞、弱口令和配置项进行扫描，及时发现并上报系统中存在的弱点，让用户对自身系统的缺陷不足有充分的了解，以便采取措施应对这些缺陷。

功能优势

态势感知具有以下特点和优势：

- **弱点扫描速度快，漏洞检测覆盖全**

  弱点分析功能采用无状态扫描技术，可在5 MB带宽条件下，并发每秒扫描10,000个IP地址。同时，拥有对第三方专属漏洞扫描的专利技术，通过指纹识别的方式对第三方CMS系统进行快速扫描。

  弱点分析功能通过与流量安全监控模块联动，实时监控网络上的URL请求、对所有目标URL和接口进行全覆盖扫描，并支持通过代理、HTTPS、DNS绑定的方式进行扫描。

  弱点分析检测的覆盖范围包括：

  # 支持对30多种通用的Web漏洞及150多种流行的专属Web应用漏洞的扫描，漏洞类型覆盖OWASP、WASC、CNVD的漏洞分类。

  # 支持对常见系统和数据库的弱口令扫描。

  # 支持对Web2.0、AJAX各种脚本语言、PHP、ASP、.NET和Java等环境的恶意篡改检测。

  # 支持对复杂字符编码的恶意篡改检测。

  # 支持对Chunk、Gzip、Deflate等压缩方式的恶意篡改检测。

  # 支持对Basic、NTLM、Cookie、SSL等认证方式的恶意篡改检测。

  同时，借助阿里云大数据计算能力，云盾每天对阿里云平台上海量的攻击行为进行数据挖掘和分析，实时获取最新的攻击行为样本和情报从而及时发现0Day漏洞，形成安全漏洞库并应用于弱点分析功能，进一步保障弱点分析功能的及时性和全面性。

- **大数据威胁分析**

  态势感知模块不仅拥有PB级别的大数据分析和计算能力，汇集全网安全数据和威胁情报，而且通过机器学习建立完整智能的安全威胁模型，并作用于百万用户的实际业务场景中。

  态势感知模块聚焦数据中心云计算用户面临的定向Web应用攻击、面向系统的暴力破解、黑客入侵行为、应用层主机层漏洞等多个方面的新威胁和新的安全趋势，帮助用户抵御来自各个维度和领域的新型安全威胁。

- **大屏展现**

  基于互联网可视化技术，将大数据威胁分析成果以直观的图形呈现于可视化大屏，作为用户专有云平台安全决策的重要支撑工具。

21.3.1.7 安全运营驻场服务

为打造稳定、可靠、安全、合规的专有云平台，云盾标准版包含一系列安全产品及安全运营驻场服务来保障用户的系统及数据的可用性、机密性和完整性。其中，安全服务是整个安全保障体系中不可或缺的部分，通过“产品+服务“的方式充分发挥专有云产品及云盾安全产品的安全特性，从技术和管理的角度提升专有云环境的安全性，切实保障用户的专有云平台。

专有云安全运营驻场服务的目标是帮助用户更好地利用专有云产品及云盾安全产品的安全特性，管理租户层应用安全的一项服务。安全运营服务包括上线前安全评估、安全访问控制策略优化、周期性安全评估、安全巡检、安全应急等一系列服务内容，全面覆盖专有云平台租户业务的完整安全生命周期。通过安全运营驻场服务，帮助用户梳理并建立云上安全运营体系，全面提升应用系统安全性，保障用户业务的安全和稳定运行。

服务内容

专有云安全运营驻场服务包含以下内容：

表 21-1: 安全运营驻场服务内容列表

服务类别	服务项	服务内容
租户安全运营	租户资产调研	在租户授权的前提下，定期调研、整理云上租户业务清单，包括业务系统名称、ECS、RDS、IP地址、域名、责任人等信息。
	租户准入安全检测	• 在租户新业务迁移上云前，通过自动化工具及人工方式对目标业务系统的系统漏洞、应用漏洞进行检测。

服务类别	服务项	服务内容
		• 提供漏洞修复指导及漏洞修复后的验证服务。
	周期性租户业务安全检测	• 周期性通过自动化工具对租户已上线业务的系统漏洞、应用层漏洞及安全风险进行评估，确定存在的风险和漏洞。 • 针对安全检测结果提供风险处置建议，包括但不限于安全策略的设置、安全补丁更新、应用层漏洞改进建议等。
	准入访问控制管理	租户新业务上云时，提供网络访问控制策略实施检测及指导。
	访问控制巡检	周期性对租户业务的访问控制风险进行巡检。
	租户日常安全风险巡检	监控、巡检专有云云盾中出现的安全事件，核实安全事件后，通知并指导用户进行修复。
云盾安全运营	云盾规则更新	负责定期更新云盾产品规则库。
	云盾接入服务	• 提供用户应用系统接入云盾产品的技术支持。 • 协助用户定制、优化云盾安全策略。
应急响应	安全通告	同步阿里云的安全应急通告，并指导用户进行相关修复工作。
	安全事件处理	发生类似于黑客入侵等紧急安全事件时，及时提供安全应急响应服务。

服务输出

安全运营驻场服务提供以下文档输出：

- 《XXX服务周报》、《XXX服务月报》、《XXX服务年报》
- 《XXX资产清单》
- 《XXX系统安全检测报告》

服务等级协议

安全运营驻场服务包含以下SLA条款：

- **资产管理**：每月更新一次资产清单。
- **安全事件响应时间**：正常工作时间，30分钟内响应。

- **安全检测**：

  # 上线前安全检测在两个工作日内完成。

  # 周期性安全检测每季度执行一次。

职责分工

安全运营驻场服务由阿里云授权的合作伙伴提供，阿里云提供服务质量管理、服务技术支持工作。

项目方	职责分工
阿里云	• 服务提供商及驻场人员的任务分配管理。 • 对服务提供商及驻场人员进行服务质量考核。 • 对服务提供商及驻场人员进行培训和技术支持。 • 安全服务项目的整体沟通、过程及质量管理工作。
服务提供商	• 对租户侧系统进行安全检测、巡检。 • 对发现的漏洞提供安全漏洞整改建议。 • 租户侧准入访问控制策略维护。 • 云盾系统安全规则、安全策略的更新维护。 • 出现安全事件时，提供应急响应服务。 • 对租户进行安全技术指导。
专有云用户	• 提供必要的授权，协助服务人员开展日常运营工作。 • 根据安全建议结合自身业务执行安全整改计划。 • 在内部推动安全整改计划的落地，改进安全体系。

风险控制

安全运营驻场服务通过以下管理办法实行风险控制：

类别	风险控制项	风险管理办法
人员组织安全	组织要求	由阿里云协同已授权的安全企业提供服务。
	人员要求	所有服务人员均经过阿里云安全团队能力评估且经过标准培训。
保密要求	保密协议	提供服务的企业和个人均需签署保密协议。
服务工具安全	工具选择	服务过程只能使用阿里云指定的安全工具。
	工具使用	对工具的使用需进行标准化配置，避免工具使用风险。

类别	风险控制项	风险管理办法
操作安全	操作流程	对扫描等可能带来风险的行为，采用分层、分批的方式执行。
	风险提醒	对操作过程中可能存在的高危风险，需提前对用户进行充分说明，并提供风险规避或控制的方法，只有在征得用户同意后方可进行操作。

21.3.2 可选安全产品

除云盾标准版中的安全产品外，专有云云盾还提供多款可选安全产品满足各种安全需求。建议您在云盾标准版的基础上根据您专有云环境中的实际业务场景，选择所需的安全产品，满足您的专有云安全防护需求。

21.3.2.1 DDoS流量清洗

DDoS流量清洗模块是基于阿里云自主开发的大型分布式操作系统和十余年安全攻防的经验的成果，为云平台用户提供基于云计算架构设计和开发的海量DDoS攻击防御模块。

功能模块

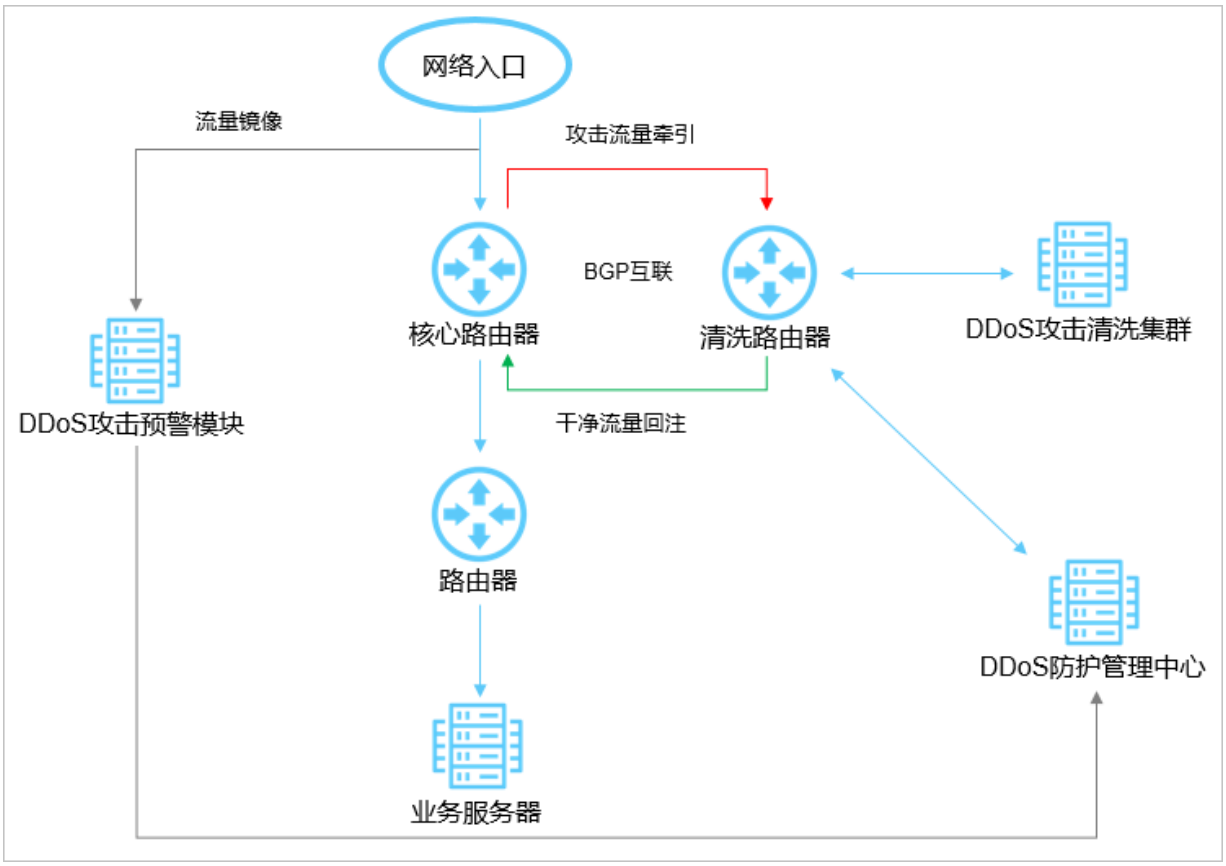
DDoS流量清洗模块提供以下功能：

功能项	功能说明
DDoS攻击清洗能力	检测并防御SYN Flood、ACK Flood、ICMP Flood、UDP Flood、NTP Flood、DNS Flood、HTTP Flood等攻击。
DDoS攻击查看	支持在界面查看DDoS攻击事件，可通过IP地址、状态、事件信息搜索到对应的DDoS攻击事件。
DDoS流量分析	支持针对某DDoS攻击进行流量分析，查看DDoS攻击的流量协议，并展示发起该攻击事件的Top10主机IP。

工作原理

通过流量安全监控模块的检测和调度，DDoS流量清洗模块对流量进行牵引、清洗和回注，如图 *21-3:* 流量清洗过程示意图所示，提供对DDoS攻击的防御，保证业务正常进行。

图 21-3: 流量清洗过程示意图



DDoS流量清洗模块与流量安全监控模块进行通信，当流量安全监控模块检测到DDoS攻击时，通知DDoS清洗系统进行引流清洗。DDoS流量清洗模块与出口网关设备直连，当发生DDoS攻击时，向出口网关发布BGP路由，将攻击流量引入DDoS流量清洗系统进行清洗，DDoS流量清洗模块根据所配置的清洗策略执行流量清洗，过滤异常流量并将正常访问流量回注至出口网关设备。

功能优势

DDoS流量清洗模块具有以下特点和优势:

- **全面覆盖常见DDoS攻击类型**

  DDoS流量清洗模块帮助用户抵御各类基于网络层、传输层及应用层的各种DDoS攻击（包括CC、SYN Flood、UDP Flood、UDP DNS Query Flood、(M)Stream Flood、ICMP Flood、HTTP Get Flood等所有 DDoS 攻击方式），并通过短信实时通知用户网络防御状态。

- **快速自动响应，一秒内进入防护状态**

  DDoS流量清洗模块采用全球领先的检测和防护技术，能在一秒内完成攻击发现、流量牵引和流量清洗全部动作。在防护触发条件上，不仅依赖流量阈值，还对网络行为进行统计判断，精准识

别DDoS攻击，从而大幅减少网络抖动现象，全面保障用户业务在遇到DDoS攻击时业务的持续性。

- **高弹性、高冗余的DDoS防御能力**

  云计算架构具有高弹性和大冗余特点，DDoS攻击防御模块可在云环境内部中无缝扩容，实现DDoS攻击防御能力的高弹性。

- **双向防护，避免云资源被滥用**

  DDoS流量清洗模块不仅防护来自于云外的DDoS攻击，还能及时发现云环境内资源被滥用的非法行为。一旦发现专有云环境内有服务器被利用向外发起DDoS攻击的行为，流量安全监控模块将与安骑士模块联动，限制被滥用的云服务器的网络访问行为并产生告警，实现对内部主机的有效管控。

21.3.2.2 云防火墙

云防火墙模块是一款针对云环境的防火墙安全产品，主要解决云上业务快速变化带来的安全边界模糊甚至无法定义的问题。云防火墙首创性地采用“基于业务可视化的结果进行业务梳理和业务隔离”的技术，实现专有云环境中东西向流量的安全访问控制。

- **零配置的业务可视**：在不影响业务运行的前提下，自动根据实际业务进行聚类分区、分组，让管理人员可以清晰看见业务结构。
- **保持业务有序性**：在云服务器快速增长的情况下，通过可视化技术和拓扑式图形界面，帮助管理人员将业务合理分区、分组，保持其有序性。
- **微隔离**：实现专有云环境中，面向具体服务器应用的安全微隔离部署。

功能模块

云防火墙模块提供以下功能：

功能项	功能说明
智能分组	基于阿里云机器学习的技术，通过根据服务器属性的传播聚类和加权快速分割，自动将业务进行分区和分组。
拓扑化	基于拓扑化的形式，展示专有云环境中所有云服务器的各种信息以及服务器之间的访问关系，通过鼠标单击、悬浮、信息高亮等呈现手段，让用户直观地了解全局业务。
关联展示	在云服务器角色管理过程中，将服务器信息与访问关系进行关联，并通过多种可视化手段进行呈现，将访问片段形成一个完整访问关系。

功能项	功能说明
角色管理	针对云服务器进行角色定义。在业务无序的情况下，有些云服务器用途已经模糊，通过云防火墙提供的快捷操作方式，帮助用户快速找到云服务器的用途和访问关系。
智能搜索	通过可叠加的搜索条件和多样的规则动作，让管理员在无序的业务拓扑中，搜索到自己需要的服务器或流量。
业务分区/分组	支持基于业务将云服务器分区分组。
基于流量线条的策略定义	通过业务流量可视化，判断流量的合法性，直接单击流量线条即可完成策略的下发。
云服务器隔离	在业务拓扑可视的前提下，为外部到云服务器之间、云服务器之间的访问，部署基于角色的安全策略。
访问关系授权	业务正常访问中，存在不定期的策略开通、变更的情况。通过业务流量可视化，让用户轻松准确地开通、变更白名单策略。

工作原理

云防火墙通过安骑士客户端Agent收集专有云环境中云服务器ECS的网络、进程信息，如实还原业务流量（服务器信息、服务器之间的访问关系），并通过一系列基于算法的数据分析，帮助用户快速找到想要看见的信息（服务器、访问连接）。

云防火墙模块与安骑士模块共用在云服务器ECS中部署的客户端Agent，收集ECS实例基本信息（主要用于实现智能分区、分组）及ECS主机中的进程、端口、协议、源IP、目标IP等信息。由云防火墙服务端进行汇总分析，通过云防火墙拓扑图展示专有云环境中各服务器之间的访问关系。同时，借助安全组的访问控制规则将基于业务可视化设置的访问控制规则真实下发，实现东西向流量的安全访问控制。

通过对业务流量进行学习和展示，引导用户快速完成业务服务器的分区、分组。同时，引导管理员对业务流量进行逐一排查和确认，最终将原本无序的业务分区、分组、并部署安全策略。同时，云防火墙持续进行业务流量的主动学习和全局展示，保持业务的有序。

应用场景

云防火墙模块适用于下列场景：

- **实现微隔离**：通过云防火墙实现精细化的微隔离。通过业务分区、角色分组将原来为了防止业务出现中断而不得不开放的端口进行更精细的管理，缩小受攻击面，降低安全隐患。

- **协助甄别流量是否安全**：在云防火墙的流量视图中，查看流量是否安全。例如，判断HTTP流量是否都已切换为HTTPS的流量、连接TCP 3306端口（MySQL的业务端口）的流量是否有来自互联网的流量。
- **判断服务器变更是否对业务造成影响**：服务器需要迁移和下线时，可以通过云防火墙先观察是否还存在相关流量，从而判断变更是否会对业务产生影响。
- **帮助业务快速扩容**：云防火墙采用去IP化的策略定义方式，能够在业务快速增长时，保持策略不发生频繁变更。例如，业务高峰期间，只需要赋予一批新添加的服务器相同的角色，无需修改任何策略，即可完成扩容。
- **流量可视，快速运维**：传统方式下，想要了解出入服务器的连接情况，只能通过抓包软件或`tcpdump`命令。而云防火墙通过流量线条的方式，将相关信息进行可视化的呈现。
- **发现是否存在端口滥用**：不同的业务开发部门，可能导致服务器提供同一服务（相同应用和进程）却使用了不同的业务端口。既浪费端口资源，也难以运维。通过流量的可视化，云防火墙可以清晰的甄别出此种端口滥用情况。

功能优势

相比传统防火墙，云防火墙模块具有以下特点和优势：

传统防火墙	云防火墙
不关注用户业务。	提供业务可视化的价值，帮助用户先看见业务，再进行策略部署。
不关注策略正确性。	通过流量的可视化，最大程度保障策略的正确性。
不关注后续策略运维的复杂性。	通过拓扑化将资产信息、资产的访问关系一一呈现，让运维变得更加简单。

21.3.2.3 堡垒机

堡垒机模块为云服务器的运维提供完整的审计回放和权限控制服务。基于账号（Account）、认证（Authentication）、授权（Authorization）、审计（Audit）的AAAA统一管理方案，通过身份管理、授权管理、双因子认证、实时会话监控与切断、审计录像回放、高危指令查询等功能，增强运维管理的安全性。

堡垒机符合各类法令法规的要求，包括等级保护、银监会、证监会、PCI 安全标准委员会、企业内控管理等相关政策规定要求。

功能模块

堡垒机模块提供以下功能：

功能项	功能说明
登录认证机制	支持本地认证方式，同时支持手机App动态口令、短信口令等双因子认证方式。
凭据托管、单点登录	支持托管云服务器ECS的账户和密码（或SSH密钥），运维人员只需要登录到云盾堡垒机后即可直接登录ECS云服务器，无需使用ECS云服务器的账户密码信息。
系统运维	支持Web端调用本地工具实现单点登录；支持“本地客户端登录堡垒机，再选择服务器”的方式进行运维。
运维监控及阻断	针对运维人员的操作过程进行实时监控，并支持以切断操作会话的方式阻断违规操作等异常行为。
日志回放、事后追溯	• 提供录像式日志回放功能，并且可通过关键信息进行定位回放。 • 提供指令记录功能，并且可基于关键指令进行过滤检索。 • 提供图像记录功能，并且可基于关键文字进行过滤检索。 • 提供文件审计功能，并且可详细记录上传下载的文件名等信息。

应用场景

随着政府部门、金融机构、企事业单位不断将自身业务迁移到专有云平台，专有云平台的安全性就越来越受到关注，而运维安全就是专有云平台用户最为担心的问题之一。传统的运维方式主要有以下两种方式：

- 设置一台跳板云服务器，仅开放跳板云服务器的SSH（TCP 22端口）、远程桌面RDP（TCP 3389端口）远程登录方式，所有运维人员先登录跳板云服务器，再从跳板云服务器访问其他云服务器进行运维操作。
- 通过搭建VPN系统，运维人员只有登录VPN后才能够和云平台内部的云服务器进行互通、运维。

上述两种方式在一定程度解决了运维过程的安全问题，但是仍然存在不少安全隐患：

- 将云服务器的SSH（TCP 22端口）、远程桌面RDP（TCP 3389端口）远程登录方式直接暴露在公网环境中，导致RDP、SSH等协议漏洞被利用、账号密码被暴力破解的安全风险较高，一旦账号密码被破解，企业的核心机密数据、用户数据就可能外泄，给企业带来巨大的损失。
- 运维、开发、第三方维护人员共用root、administrator等高权限账号，容易出现越权操作、敏感数据泄漏，而且发生安全事件后也难以定位到责任人。
- 开发和运维过程不透明，一旦出现异常操作，事后的分析难度较大。
- 金融云、政务云等用户面临着银监会、公安部、等级保护等法律法规监管要求，而上述两种运维方式都无法符合这些监管的要求。

通过部署云盾堡垒机进行统一的运维管理，则可以解决以上安全隐患，增强运维管理的安全性。

堡垒机适用于以下场景：

- **审计合规要求严格的场景**

  例如，金融保险行业，面临行业高规格的安全监管要求，需要建立完善的审计机制。通过云盾堡垒机可以实现：

  # 部门权限隔离：基于部门隔离功能，实现各部门有效管理和审计。
  # 统一运维入口：为操作人员提供了统一的运维入口，解决分散登录的问题。
  # 满足合格审核：建立健全的云上运维审计机制，满足行业监管要求。

- **高效稳定的运维管理场景**

  例如，互联网行业企业，发展快速、人员与系统增长迅速，需要高效、稳定的操作审计系统。通过云盾堡垒机可以实现：

  # 高并发会话：支撑千人级别的并发会话。
  # 稳定运行：有高稳定性的SLA保障。
  # 运维故障回溯：运维人员难免发生误操作，通过回溯操作内容，建立运维红线。

功能优势

堡垒机具有以下特点和优势：

- **多协议支持**：支持SSH、Telnet、RDP、VNC、SFTP、FTP、Rlogin等多种协议。
- **授权规则策略管控**：

  # 支持命令审批、命令拦截、命令阻断。
  # 支持文件传输控制。
  # 支持基于来源IP、时间等因素的访问控制。

- **审计合规**：满足金融行业监管单位、信息安全等级保护的审计要求。
- **稳定可靠、安全可信**：

  # 依托于阿里云IaaS自身健壮性，堡垒机稳定可靠。
  # 通过“手机+静态密码+短信动态口令（手机App口令）”的方式实现双因子认证，确保访问身份的安全可靠。

性能指标

堡垒机模块的性能指标如下：

- **平台侧堡垒机**：用于专有云平台物理服务器的运维管理。

  # 500台物理服务器资产接入

  # 500个并发会话

- **租户侧堡垒机**：用于专有云平台上云服务器ECS的运维管理。

  # 200台云服务器资产接入

  # 200个并发会话

21.3.2.4 数据库审计

数据库审计模块是随着用户应用上云、云端数据安全面临挑战而推出的适用于专有云环境中数据库安全审计的产品。基于阿里巴巴集团多年数据库安全技术积累，数据库审计系统将传统产品与云端相结合，在专有云环境中形成一套为数据库运维和安全管理人员提供安全、诊断与维护能力为一体的安全管理工具。

数据库审计系统实现了对云端自建数据库、RDS数据库访问的精确审计及准确的应用用户关联审计，并具备风险状况、运行状况、性能状况、语句分布的实时监控能力。

数据库审计系统通过数据库化的界面语言、智能化的协议识别、可视化的运行状况呈现、可交互可下钻的风险追踪能力，完美实现快速部署、方便维护的云数据库审计。

功能模块

数据库审计模块提供以下功能：

功能项	功能说明
审计内容	数据库审计的审计内容包括： • 会话的终端信息：IP、MAC地址、端口、客户端工具名(程序名)、数据库用户名 • 会话的主机信息：IP、MAC地址、端口、数据库名（实例名） • 会话的其它信息：登录时间、会话时长 • 操作信息：操作类型（DDL、DML、DCL等）、操作时间、执行时长、操作成功与失败、受影响行数、操作对象（表、列、存储过程名称）、SQL语句
审计过滤规则	数据库审计的审计过滤规则包含以下审计策略要素： • who：数据库用户名、应用用户、主机名称、操作系统账号 • what：表、字段、包、存储过程、函数、视图 • where：IP地址、用户名、端口号、数据库类型 • when：操作时间、登录时间等

功能项	功能说明
	• how：客户端工具 • Range：修改、删除或查询的行数 • ResultSet：返回结果集
审计信息展示	支持从会话维度、语句类型纬度、风险纬度三个纬度进行导航展示。 审计信息包括告警级别、事件发生时间、客户端IP、目标数据库IP、操作类型、客户端MAC地址、目标数据库MAC地址、客户端端口、操作信息大小、返回状态、结果信息、客户端执行命令等详细信息内容。
审计查询	支持基于时间、客户端IP、数据库服务器IP、用户名、数据库操作命令、数据库表名/字段名等多种丰富的查询检索条件。
统计分析	• 事件分析：依据安全策略，以时间为基线，统计异常事件的发生数量和趋势。 • 告警分析：依据安全策略，以时间为基线，统计严重告警事件的发生数量和趋势。 • 综合分析：以数据库操作类型为基线统计各类操作状况。 • 会话统计：以时间和IP为基线统计会话的数量、流量、操作的语句数量。
审计报表	• 支持（系统级）多数据库聚合报表展现和单数据库综合性报表展现。 • 基于总体概况、性能、会话、语句、风险多层面展现报表。 • 支持图表结合展现，支持柱形图、饼状图、条形图，双轴折线图等多种统计图展现形式。 • 支持报表定时推动功能，自定义推送周期以邮件形式推送报表文档。 • 支持日、周、月 等综合性报表和自定义分析型报表功能。
业务化语言	支持业务化语言展现，可自定义SQL语句转换业务化语言展现方式。

应用场景

数据库审计模块适用于以下场景：

- **安全事件追查**：数据库审计系统提供语句、会话、IP、数据库用户、业务用户、响应时间、受影响行数等多种维度的数据库操作记录和事后分析能力，是安全事件发生后最为可靠的追查依据和来源。通过SQL行为与业务用户的准确关联，使数据库访问行为有效定位到业务工作人员，实现有效追责、定责。
- **数据库性能诊断**：数据库审计系统实时显示数据库的运行状况、数据库访问流量、并发吞吐量、SQL语句的响应速度；同时，提供最慢语句、访问量最大语句的分析，帮助运维人员进行性能诊断。

- **发现程序后门**：数据库审计系统提供SQL学习和SQL白名单能力，实现对业务系统的SQL建模；通过合法系统行为的建模，使隐藏在软件系统中的后门程序在启动时即被发现，并提供实时的告警能力，降低数据泄漏损失。
- **数据库攻击响应**：数据库审计系统提供数据库风险告警能力，对于SQL注入、数据库漏洞攻击、过量数据下载、危险SQL语句（如No where delete)等风险行为的策略制定能力，提供实时告警能力。

功能优势

数据库审计模块具有以下特点和优势：

- **满足合规要求**：通过数据库审计系统可以监控每个用户的行为、各种可疑操作并进行告警通知，并对操作记录进行全面的分析。同时，支持对于自身审计进程的监控，具备自动归档能力，防止审计记录被恶意删除。
- **精准应用关联审计**：数据库审计系统的应用关联审计通过在Web服务器部署JDBC TAP插件的方式实现，通过配置该插件来“代理”数据库原生的驱动和数据库的会话，连接时自动采集Web客户端的信息。应用关联审计能够将Web客户端的应用请求和数据库处理语句完美匹配，不会影响数据库和应用业务的运行，同时可以经受任何并发压力。
- **全面审计**：数据库审计系统在网络镜像技术基础上，通过可配置的主机探针技术实现了数据库主机的流量捕获，从而实现了对数据库访问流量的全捕获。
- **准确审计**：基于精确协议分析、完全SQL解析、参数化匹配和100%应用关联技术，创造了业界最为准确的数据库审计系统，为可信审计追踪提供了坚实的基础。
- **高效审计**：通过SQL语句根据规则进行分类的技术，实现高效SQL解析；通过分级存储和批量入库技术实现审计数据高效入库，入库性能达到万条/秒。
- **高效分析**：通过三级存储机制，实现了海量数据存储；通过列存技术，实现了数据的快速分析与检索，亿级数据秒级响应；通过多维分析线索，实现了灵活的信息挖掘。
- **便捷使用**：以业务数据库为界面组织中心，按照语句、会话、风险为线索实现数据库审计信息进行组织和导航；提供业务翻译能力，将枯燥的SQL语句翻译成中文的业务操作描述。

21.3.2.5 数据发现与脱敏

在专有云业务场景中，随着数据资产的不断增加，管理员往往很难直观的了解云上自己的数据库有多少台、如何分布、是否启用。通过数据库发现功能，能够扫描云数据库的分布，自动发现专有云内的所有数据库。同时，通过数据脱敏功能有效防止企业内部对隐私数据的滥用，防止隐私数据在

未经脱敏的情况下从企业流出。满足专有云环境中，既要保护隐私数据，又要满足开发、测试、模型训练等业务对数据的需求；同时也保持监管合规，满足企业合规性。

- **数据发现模块**：通过扫描IP段设备流量信息检测数据资产，发现数据库分布；通过系统内置发现规则发现敏感数据，对其敏感数据分级分类，呈现可视化敏感数据分布；通过对资产SQL语句量和会话并发量判断资产使用热度，根据流量统计发现静默资产；通过数据库授权发现资产权限分布及权限详情。
- **数据脱敏模块**：通过对敏感数据进行数据抽取、数据漂白、和动态掩码的脱敏处理，满足生产数据面向测试、开发、培训和数据共享场景的数据安全需求，实现“用”、“护”结合。

功能模块

数据发现与脱敏模块提供以下功能：

功能项	功能说明
云数据库自动嗅探	提供自动搜索专有云环境中数据库的功能，也支持指定IP段和端口的范围进行搜索。自动发现数据库的端口号、数据库类型、数据库实例名、数据库服务器IP地址等基本信息。
自动识别敏感数据	提供敏感信息的自动发现能力，通过内置的敏感数据特征库，对常见的如姓名、证件号、银行账户、金额、日期、住址、电话号码、Email地址、车牌号、车架号、企业名称、工商注册号、组织机构代码、纳税人识别号等敏感数据自动识别。
敏感数据分类分级	根据不同数据特征，对常见的敏感数据进行分类。根据不同的数据类型指定的不同敏感级别，自动对包含敏感数据的表、模式、库进行敏感度评分。
权限梳理	对数据库中不同用户、不同对象的权限进行梳理并监控权限变化。权限梳理主要从以下两个维度展开： • 监控数据库中的用户的启用状态、权限划分、角色归属等基本信息。 • 对数据库中的对象可被哪些用户访问的情况进行归纳总结，特别是对包含了敏感列的表或者敏感度评分较高的对象，着重监测其访问权限划分情况。
敏感数据管理	**敏感数据发现**：根据指定的部分敏感数据特征或预定义的敏感数据特征，在执行任务过程中对抽取的数据进行自动的识别，发现敏感数据，并自动根据规则对发现的敏感数据进行脱敏处理。 **敏感数据字典管理**：对敏感字段进行分类管理，对同类敏感数据实施统一的脱敏算法和策略，保证同一组织内跨系统、跨库之间的脱敏一致性；并支持敏感数据字典导入、导出等功能。

功能项	功能说明
脱敏算法	根据不同数据特征，内置了丰富高效的脱敏算法，可对常见的姓名、证件号、银行账户、金额、日期、住址、电话号码、Email地址、车牌号、车架号、企业名称、工商注册号、组织机构代码、纳税人识别号等敏感数据进行脱敏，内置脱敏算法具有如下特性： • 同义替换：使用相同含义的数据替换原有的敏感数据。例如，姓名脱敏后仍然为有意义的姓名，住址脱敏后仍然为住址。 • 部分数据遮蔽：将原数据中部分或全部内容，用“*”或“#”等字符进行替换，遮盖部分或全部原文。 • 混合屏蔽：将相关的列作为一个组进行屏蔽，以保证这些相关列中被屏蔽的数据保持同样的关系。例如，城市、省、邮编在屏蔽后保持一致。 • 确定性屏蔽：确保在运行屏蔽后生成可重复的屏蔽值。可确保特定的值（如客户号、身份证号码、银行卡号）在所有数据库中屏蔽为同一个值。 • 可逆脱敏：确保脱敏后的数据可还原，便于将第三方分析机构和内部经分团队基于脱敏后数据的分析结果还原为业务数据。 支持的脱敏算法包括屏蔽、变形、替换、随机、格式保留加密（FPE）和强加密算法（如AES加密算法）。
数据子集管理	支持对目标数据库中一部分数据进行脱敏。提供多种数据子集抽取方式，包括基于事实表发起的百分比抽取方式；基于基础表中的身份、商品信息发起的向下延展的数据抽取方式；针对多表的灵活条件设定的抽取方式。根据过滤条件，对数据来源进行过滤筛选形成数据子集，以适应不同场景下脱敏需求。
脱敏策略和方案管理	针对不同脱敏项目，可以配置定制化的脱敏策略，或实现脱敏算法的扩展；支持脱敏策略的导入导出，以实现策略复用。对于同一类应用场景，可将若干脱敏策略组合成为适用于该场景的脱敏方案。脱敏方案制定后，可被重复利用于该场景下不同批次数据的脱敏需求。
脱敏任务管理	可对脱敏任务进行停止、启动、重启、暂停、继续，并且支持任务并发，充分利用系统资源，提高脱敏效率。同时，脱敏任务可兼容执行过程中遇到的异常情况，支持跳过异常数据继续执行任务。
脱敏数据验证	支持对脱敏后的数据进行“验证”，确定哪些数据是“漏网”的真实数据，从而帮助用户在使用这些数据前能够及时的发现并相应的弥补脱敏脚本的不足。

应用场景

数据发现与脱敏模块适用于以下场景：

- **自动发现专有云中的数据资产**：一般专有云大规模应用的后台数据库都有上百个表和上千个列。保护核心数据资产，首先要了解哪些数据是敏感数据，通过敏感数据发现功能对表和列进行扫

描，发现密码、个人标识信息、信用卡账户等敏感数据，同时也可以通过自定义敏感对象搜索关键字的方式来发现用户特定的敏感数据。

- **协助完善数据分级分类机制**：在数据的使用中，不同的数据拥有不同的敏感级别和密级，不同级别的数据所需的安全策略和加固方式都是不同的。数据梳理模块对敏感数据支持完善的定级机制，能够协助管理员判断数据库中表、模式和整库的敏感级别，便于进一步制定专有云数据安全策略或采取其他数据加固措施。
- **监控数据权限变化**：让管理员实时了解数据资产的权限变化，支持按照用户维度、对象维度对数据库的权限进行分析整理，并且能够对比不同轮次扫描之间的权限变化情况。实时查看，展现数据库实时的安全视图；定期扫描，反映当前安全状况相对于基线的变化，并可查看详细的变化状况。
- **保护隐私数据，满足合规性**：数据脱敏模块提供了丰富的内置脱敏算法和灵活的、流程化的策略和方案管理能力，支持对多种数据源进行脱敏处理，帮助在专有云环境中不改变业务流程的前提下快速部署实施，有效的降低脱敏的复杂度和风险，控制脱敏成本。
- **敏感数据自动发现**：内置了大量的敏感数据发现算法，能够通过对数据的采样分析，自动发现系统中的敏感数据，包括姓名、证件号、银行账户、金额、日期、住址、电话号码、Email地址、车牌号、车架号、企业名称、工商注册号、组织机构代码、纳税人识别号等；同时提供了用户自定义敏感数据特征的扩充能力。通过敏感数据自动发现功能，不仅可以避免人工定义敏感数据带来的大量工作，同时可确保不会遗漏隐私信息，更能够持续发现新的敏感数据字段。
- **敏感数据字典管理**：以敏感数据为中心，进行分类管理数据库字段。例如，将“身份证号”作为一类敏感数据，管理所有数据库中的身份证号数据字段，实施统一的脱敏算法，并且支持敏感数据字典的导入、导出等管理功能。

功能优势

数据发现与脱敏模块具有以下特点和优势：

- **数据安全始于资产梳理**

  帮助企业快速精准资产梳理，自动发现数据资产分布，并对数据资产进行分级分类、权限梳理。

  通过自动发现技术、静+动态梳理有效地解决了企业对专有云环境中数据库资产安全状况摸底及资产管理工作；改善了以往传统方式企业在资产管理和梳理的工作模式，提高了工作效率，保证了资产梳理工作质量。合规合理的梳理方案，能做到对风险预估和异常行为评测，极大程度地避免了核心数据遭破坏或泄露的安全事件。

- **数据存储与分布梳理**

自动帮助用户明确自身数据如何被存储，在被哪些部门、系统、人员使用，以及如何使用。

通过对数据资产的梳理，明确了在数据安全治理中不同受众的分工、权利和职责。

- **敏感数据梳理，核心数据安全管控**

  敏感数据在数据库中的分布，是实现管控的关键。只有梳理清楚敏感数据在数据库中分布，才能针对数据库实现安全管控策略；对该数据库的运维人员实现安全管控措施；对该数据库的数据导出，实现具体的模糊化策略；对该数据库数据的存储实现加密安全方案。

- **防止生产库中敏感数据泄露**

  通过对生产库中的身份信息、地址信息、银行卡号信息、电话号码信息等敏感数据进行混淆、扰乱后再提供给第三方使用，防止生产库中的敏感数据泄露。

- **提升测试、开发和培训数据质量**

  通过内置的脱敏策略和算法保证脱敏后数据的有效性（保持原有数据类型和业务格式要求）、完整性（保证长度不变化、数据内涵不丢失）、关系性（保持表间数据关联关系、表内数据关联关系），提升数据质量。

21.4 产品价值

自网络安全法正式实施以来，关键信息基础设施安全保护条例和云等保2.0标准接连发布，通过等级保护测评已成为专有云平台的最基本的安全需要。同时，随着云上业务面临黑客入侵、加密勒索等安全风险的增加，对于安全防护和检测能力的需求越来越迫切。

云盾在阿里云专有云出口，通过流量安全监控系统在网络层对恶意的攻击行为进行识别，实时地阻断网络攻击行为。在主机层对Web木马和恶意文件进行实时查杀，避免服务器被攻击者利用。实时拦截暴力破解行为，并对异常的登录行为进行告警，避免攻击者利用弱口令登录系统窃取或者破坏用户业务数据。

纵深防御的安全体系架构

云盾由多个功能模块组成，在专有云网络出口、专有云网络中、专有云服务器上实现纵深防御，多点联动。为了方便用户集中管理和实时掌握云平台安全风险，云盾提供了统一的管理视图，用户可以在云盾集中管控系统上对所有安全防护模块中的安全策略进行统一管理，同时还可以在集中管控系统上对日志进行关联分析。

云盾由涵盖网络安全、主机安全、应用安全、弱点分析等多层次的安全防护模块组成，在云边界、云网络中、云服务器上形成一套纵深的防御体系，通过集中管控的管理中心协调调度，综合各模块

提供的安全信息，做出最准确的判断，并且可以在最合适的位置检测和阻断恶意的攻击行为，有效地保护了云环境不受外界攻击者的侵扰，保障用户业务系统的安全。

跟云平台深度耦合的安全方案

十年攻防，一朝成盾。在经历了阿里巴巴集团自身业务十年来的安全护航以及阿里云六年安全运营保障，阿里巴巴积累了大量的安全研究成果、安全数据和安全运营方法，形成了一支专业的云安全专家团队。云盾是集合这些安全专家多年攻防经验开发出来的一套专门面向云平台的攻击防护产品，可有效地保护专有云上用户的云网络环境和业务系统的安全。

云盾的各组件是软件虚拟化，具有较为广泛的硬件兼容性，可以快速的部署扩容和投入使用，适应云计算弹性的特点；云边界、云网络上防御模块采用的是旁路的架构，贴合云的业务，对云平台业务影响最小化；云服务器上的防御模块是虚拟化，适应虚拟机灵活的特点。

提供租户维度的安全自服务感知

云平台是面向租户维度的，租户可以通过云安全中心控制台查看租户侧自身的安全防护情况，生成简单的报表，并且通过外部资源配置可以实现短信和邮件方式的告警通知。

安全能力输出

云盾的防护策略和数据源自于多年的积累，百万级的用户每天面临多达几十万次的各种攻击，阿里充分利用了这些安全攻防的数据积累，每天对10多TB的安全数据进行分析。分析结果形成恶意IP库、恶意行为库、恶意样本库、安全漏洞库等基础安全能力，并及时应用到云盾的各个防护模块中，提升云盾防护能力，为用户带来更好的安全保障。

22 密钥管理服务KMS

22.1 产品概述

密钥管理服务（Key Management Service，简称KMS）是阿里云提供的一款安全、易用的管理类服务。用户无需花费大量成本来保护密钥的保密性、完整性和可用性。

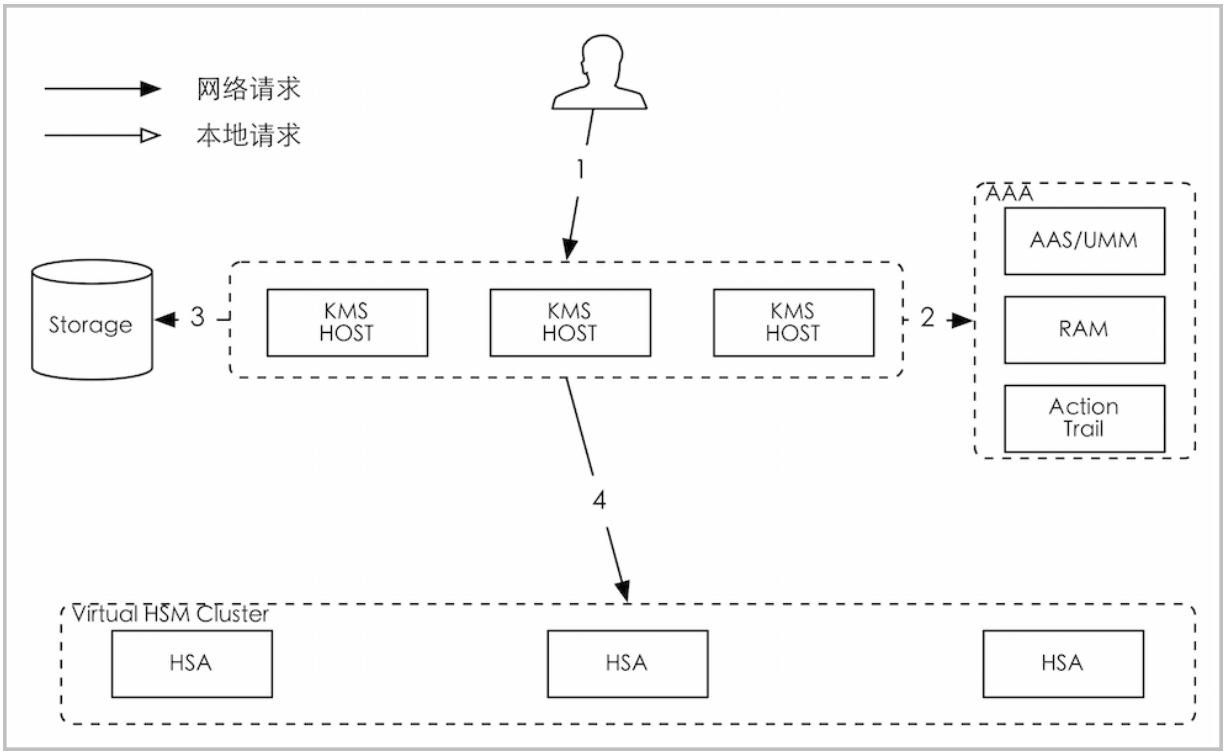
借助密钥管理服务，用户可以安全、便捷的使用密钥，专注于开发加解密功能场景。

22.2 产品架构

KMS采用Region化方式部署，每个Region之间功能完成相同，但数据互相独立；单个Region内采用分布式架构，由多个等价的节点共同组成，同Region下任意一个节点都具有相同的可用性，同时可以根据实际的访问需求进行扩容、缩容。

KMS的架构如图 *22-1:* 产品架构图所示。

图 22-1: 产品架构图



KMS分为如下四个模块：

- 存储模块Storage

  负责EKT（Exported Key Token 加密后的用户主密钥）以及其他元数据的存储。

- AAA

  认证、鉴权、审计模块。

- KMSHOST

  负责处理用户API请求。

- HSA（Hardware Security Appliance）

  模拟硬件加密机（Hardware Security Module，简称HSM）模块，负责处理KMS密码学相关逻辑，采用raft协议的分布式存储与可信计算相关技术。

22.3 功能特性

22.3.1 方便的密钥管理

用户可以通过KMS提供的API或KMS控制台便捷地进行密钥管理，包括：

- 可随时禁用、启用用户密钥，当密钥被禁用后，使用该密钥加密的数据将无法被解密。
- 密钥删除采用预删除的策略，可随时取消对密钥的预删除，降低误操作可能带来的影响。
- 可通过访问控制（RAM）对密钥权限进行管理，让密钥的加密、解密权限进行分离。
- 可通过EncryptionContext实现对密钥、密文的进一步详细控制。

22.3.2 信封加密技术

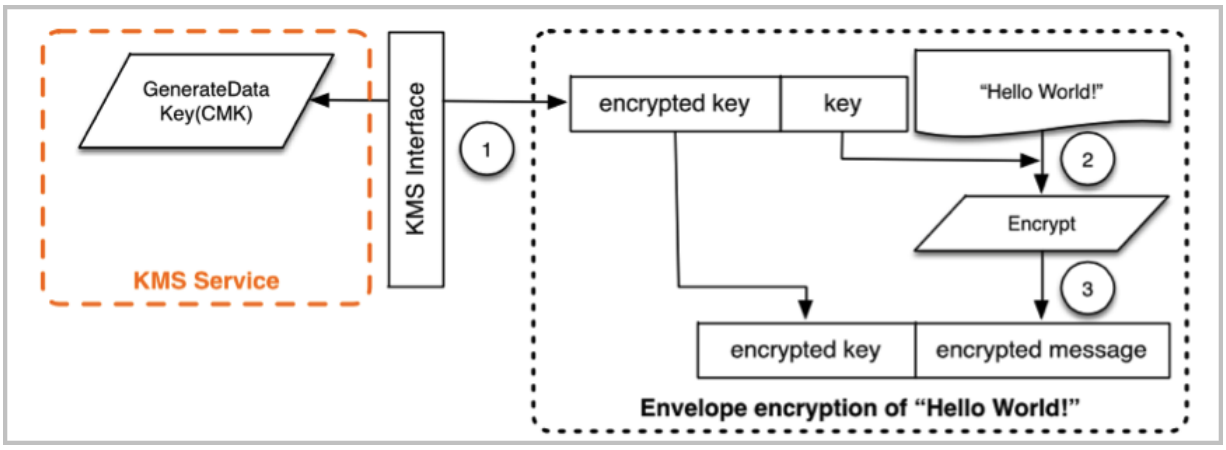
KMS虽然提供加密（Encrypt）API，但是实际上，KMS并不承担直接的数据加密工作，KMS提供的是主密钥的管理服务和数据密钥的加解密服务，需要用户自己通过数据密钥加密数据。

用户可以使用自己的数据密钥加密数据，然后通过Encrypt接口保护自己的数据密钥，也可以直接使用KMS产生数据密钥的API（GenerateDataKey）获取数据密钥。

加密流程

信封加密技术的加密流程如图 *22-2:* 加密流程图所示。

图 22-2: 加密流程图



从图 *22-2:* 加密流程图可知，加密流程如下：

1. 通过指定的MasterKey产生一个DataKey，得到key和encrypted key。

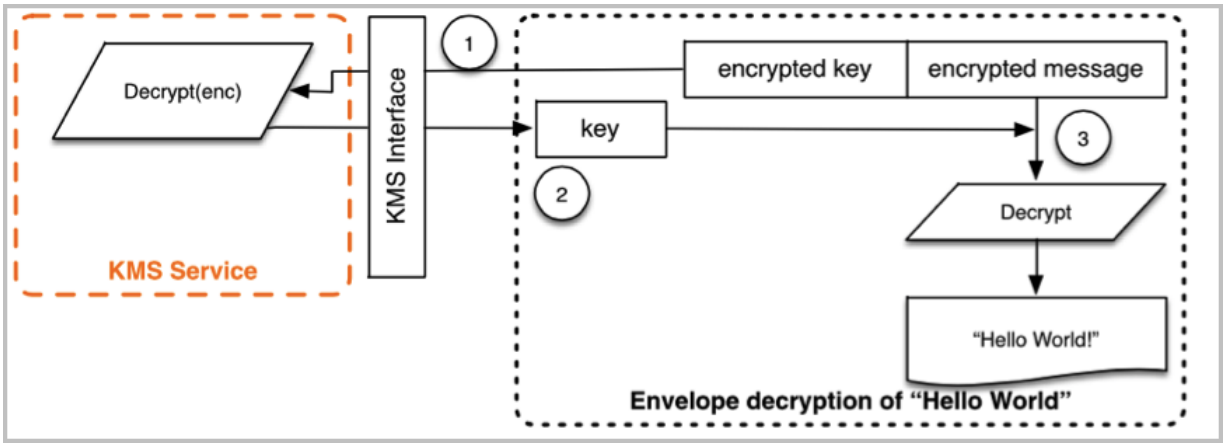
   或者用户自己产生DataKey，通过Encrypt接口获取encrypted key。

2. 使用DataKey进行数据加密，得到密文。
3. 将encrypted key和密文一起保存。

解密流程

信封加密的解密流程如图 *22-3:* 解密流程图所示。

图 22-3: 解密流程图



如图 *22-3:* 解密流程图可知，解密流程如下：

1. 将encrypted key通过KMS进行解密。
2. 得到明文的key。

3. 使用明文的key解密密文，得到明文。

22.3.3 安全的密钥存储

KMS通过以下方式保证密钥存储的安全性：

- 用户主密钥明文只会出现在HSA模块的内存中，KMS的Storage模块只会存储用户主密钥的密文。
- 用户主密钥通过HSA模块的DomainKey进行加密，DomainKey每天轮转一次。
- DomainKey通过可信计算技术进行加密存储，基于分布式存储协议的存储方式，保证DomainKey的高可靠性。

23 云解析DNS

23.1 什么是专有云DNS

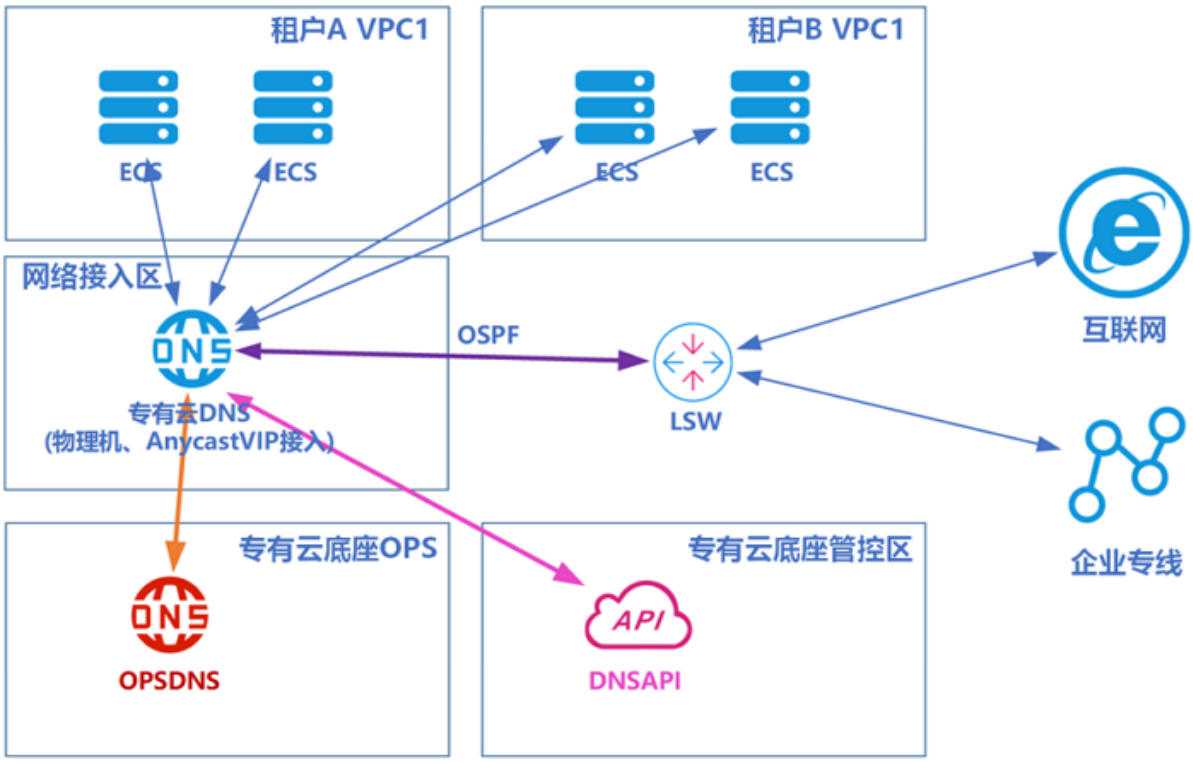
专有云DNS是运行在阿里云专有云环境的一套DNS产品，为专有云环境提供域名解析服务。专有云DNS通过设置域名与IP地址的对应规则和策略，可以将来自客户端的域名访问请求重定向到云平台中的云资源上、企业内网的业务系统上、互联网服务提供商的服务资源上等。

专有云DNS为VPC网络环境提供基础DNS解析调度服务。通过专有云DNS，您在自己的VPC内可以：

- 访问VPC内的其他ECS主机。
- 访问云平台提供的云服务实例资源。
- 访问客户自定义的企业级业务系统。
- 访问互联网上的业务和服务。
- 通过企业专线，与企业自建DNS实现互通。

23.2 系统架构

图 23-1: 专有云DNS架构图



专有云DNS整体产品的部署架构

- 标准采用两台物理服务器放在网络接入区，并支持横向扩展。
- 两个管控接口做bond，上联管控ASW，网关指向默认内网网关。
- 两个服务接口上联LSW（支持ECMP），接口上运行OSPF路由协议发布anycast vip路由，同时允许通过两个服务接口外联公网。
- 集群支持单AZ/多AZ/多Region部署。
- 管控系统部署在管控区容器中。

23.3 基本功能

内网域名管理

专有云DNS提供了内网域名的数据管理功能，可以实现内网域名的注册、搜索、功能备注和删除操作。另外，还可以进行域名主机记录的添加、删除和修改操作。支持的记录类型包括A、AAAA、CNAME、NS、MX、TXT、SRV、PTR。

内网域名管理功能提供了内网域名的DNS解析功能，为VPC网络内的服务器提供域名解析服务，DNS服务地址采用anycast部署，提供在机房级容灾场景下的服务无缝切换。

转发域名管理

专有云DNS提供域名级别的转发操作，将特定域名的解析操作转发到其他DNS服务器上进行解析。

转发域名功能提供两种转发操作模式，强制转发模式和优先转发模式。

- 在强制转发模式下，设置只使用转发目的DNS服务器做域名解析，如果解析不到（解析超时）则返回DNS客户端提示查询失败。
- 在优先转发模式下，设置优先使用转发目的DNS服务器做域名解析，如果查询不到再使用本地DNS服务器做域名解析。

递归解析管理

DNS支持递归解析功能，提供互联网域名的递归解析操作，满足企业主机访问互联网服务的需求。

选项配置

专有云DNS提供打开、修改和关闭全局默认转发配置功能。

23.4 产品优势

企业域的域名管理

专有云DNS提供企业域的域名管理操作和解析操作。

- 支持云资源实例域名（包括ECS主机实例域名）的正解和反解。
- 支持企业内网业务域名的正解和反解。
- 支持常见的域名解析记录类型(A、AAAA、CNAME、NS、MX、TXT、SRV、PTR)的添加、修改和删除操作。
- 支持域名的一条主机记录添加多个解析记录(A、AAAA、PTR)，解析响应默认应答所有记录，并支持随机轮转，实现简单负载均衡。

灵活组网和并网

专有云DNS提供企业域的域名转发操作，实现灵活组网和并网操作。

- 支持全局域名转发。
- 支持按域名转发。

企业主机访问互联网

在开通对外访问公网的情况下，专有云DNS支持公网域名递归解析功能，提供互联网域名的递归解析操作，满足企业主机访问互联网服务的需求。

统一管控平台

专有云DNS的管控系统集成在专有云控制台统一管控平台中，统一账号统一管理。专有云DNS具有以下优势：

- 数据管理和服务管理支持Web操作，操作简单容易理解。
- 专有云DNS产品采用集群化部署，扩展性强，支持横向扩容。
- 专有云DNS产品支持多可用区部署，支持同城双活、同城容灾和异地容灾功能。
- 专有云DNS产品采用Anycast部署，提供平滑的服务高可用切换和容灾切换。

23.5 产品价值

专有云DNS作为基础核心网络服务，掌管着企业专有云流量入口大门，除了提供基础域名解析服务，还为应用提供多种流量负载和调度的服务，实现专有云和自建IDC的互联互通。专有云DNS将伴随企业客户共生共赢，提供贴身的基础服务保障，为企业云环境建设、机房高可用、流量负载均衡均衡、容灾切换等多种IT架构提供丰富的解决方案，为企业客户的IT业务提供保驾护航服务。

24 企业级分布式应用服务EDAS

24.1 什么是企业级分布式应用服务EDAS

企业级分布式应用服务 EDAS（Enterprise Distributed Application Service）是阿里巴巴中间件团队研发的 PaaS 平台，为企业提供高可用和分布式的互联网架构解决方案。

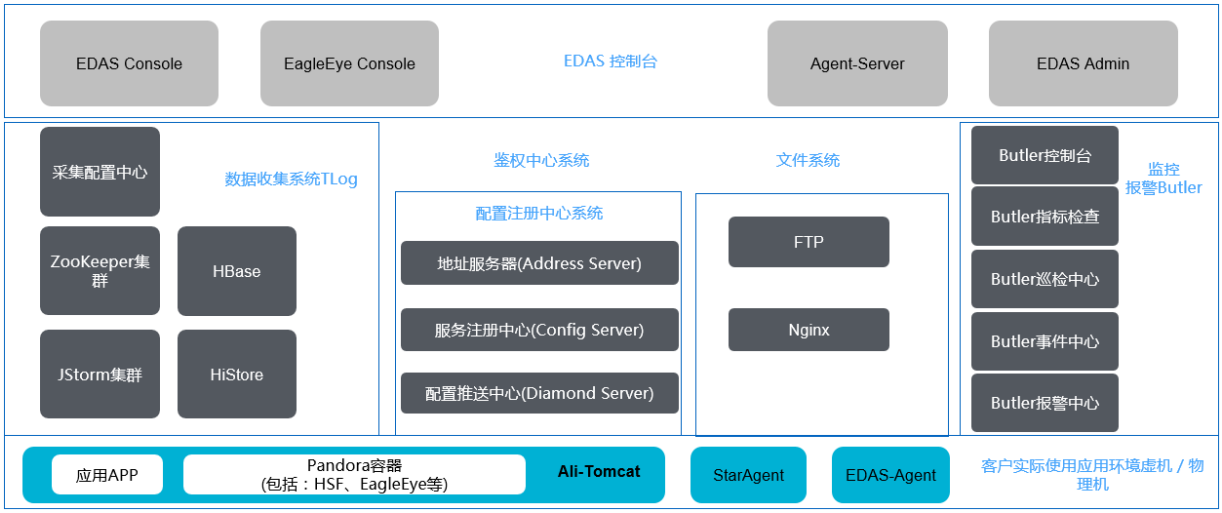
EDAS 充分利用阿里云的资源管理和服务管理系统，并在此基础上提供了一系列强大的功能，包括分布式服务框架、服务治理、统一配置管理、分布式链路跟踪、高可用及数据化运行等。这些功能在阿里巴巴集团内部经过核心电商平台多年严苛的测试。

通过 EDAS，您可以轻松构建微服务架构，为您的企业建设大规模分布式系统，用于发布和管理应用。EDAS 协助您进行 IT 系统转型，以满足不断增长的业务需求。

24.2 产品架构

企业级分布式应用服务EDAS由控制台、数据采集系统、配置注册中心和鉴权中心等系统组成，整体系统架构如图 *24-1: EDAS系统架构*所示。

图 24-1: EDAS系统架构



- EDAS控制台

  用户使用EDAS系统功能的操作界面，是唯一可以让客户直接使用的系统。用户通过控制台可以实现资源管理、应用生命周期管理、运维管控及服务治理、立体化监控及数字化运营等。

- 数据收集系统

实时收集EDAS集群及所有客户应用机器的系统运行状态，调用链日志等，并进行实时汇总计算和存储。

- 配置注册中心系统

  HSF（RPC框架）服务发布及订阅的中心服务器，也是分布式配置配置推送的中心服务器。

- 鉴权中心系统

  对用户的数据进行权限控制，以保证各个用户的数据安全。

- 运维系统

  EDAS对外输出的主要日常监控及报警工具，提供EDAS所有组件的日常巡检及报警等工作。

- 命令通道系统

  远程发送相关指令到应用机器执行的控制中心。

- 文件系统

  用于存放用户上传的WAR包及JDK、Ali-Tomcat等必须组件。

24.3 组件及作用

24.3.1 EDAS控制台

EDAS控制台是供用户使用EDAS系统功能的操作界面，是唯一可以让客户直接使用的系统。用户通过控制台可以实现资源管理、应用生命周期管理、运维管控及服务治理、立体化监控及数字化运营等。

EDAS控制台包含2个组件，EDAS Console和EDAS Admin。

- Console是真正供客户访问的操作界面。
- Admin组件主要用来执行后台定时任务，例如：定时同步ECS数据、定时自动扩缩容等。

24.3.2 数据收集系统

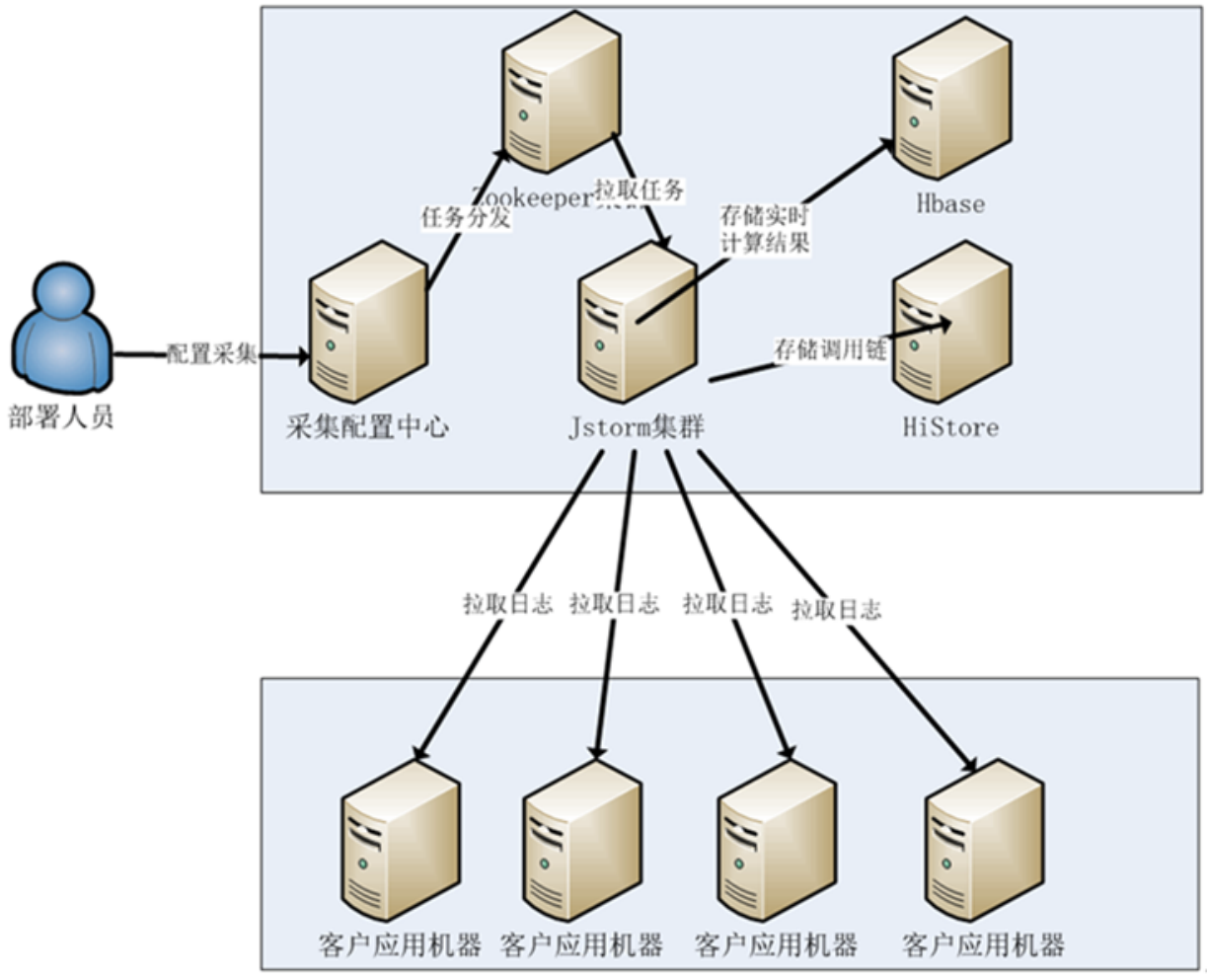
数据收集系统负责实时收集EDAS集群及所有客户应用机器的系统运行状态，调用链日志等，并进行实时汇总计算存储到HBase及HiStore中。HBase存储实时计算的结果，HiStore存储调用链的详细数据，存储的数据作为监控报警及调用链查看的基础数据。

数据收集系统包含采集配置中心、JStorm实时采集节点、HBase和 HiStore。

- 采集配置中心，主要是用来配置采集规则、采集的切分规则、采集的目标节点等。配置完成之后生成采集任务推送到ZooKeeper。

- JStorm实时采集节点，是任务真正执行的节点，任务会分发到每个采集节点执行，采集时会最主动访问客户应用机器的8182端口进行日志数据的拉取，日志拉取后会进行实时的分析和计算。
- HBase用于存储实时计算后的各种数据，目前默认保留2000小时，对外输出时根据情况进行调整。
- HiStore用于存储调用链的详情日志，用于调用链查询，目前默认保留7天时间。

图 24-2: 数据收集流程图



24.3.3 运维（Butler）系统

Butler系统是EDAS对外输出的主要日常监控及报警工具，提供EDAS所有组件的日常巡检及报警等工作。在基础硬件及网络完善的情况下，可以实时监控EDAS系统各个组件的运行状态，如果发现异常可以触发报警通知运维人员进行及时故障排查。

24.3.4 配置注册中心系统

配置注册中心是HSF（RPC框架）服务发布及订阅的中心服务器，也是分布式配置配置推送的中心服务器。

配置注册中心包含地址服务器（Address Server）、服务注册中心（ConfigServer）、配置推送中心（Diamond Server）：

- 地址服务器，底层是Tengine，配置了服务注册中心和配置推送中心的地址列表。HSF或配置推送时，首先需要连接地址服务器获取对应的地址列表后，才能通信。
- 服务注册中心，HSF服务提供者发布的服务注册到本中心，HSF消费者从本中心订阅对应的服务，获取提供服务的IP列表，进行服务调用。
- 配置推送中心，提供了配置信息管理功能，通过配置客户端发布和订阅相应信息，确保配置信息在多个系统保持实时同步。

24.3.5 鉴权中心系统

为保证各个用户的数据安全，使用鉴权系统对用户的数据进行权限控制。用户登录也使用鉴权系统的单点登录系统进行登录。

鉴权系统需要依赖地址服务器、配置推送中心，目前可以和配置注册中心共用这2个组件。

24.3.6 命令通道系统

命令通道系统是远程发送相关指令到应用机器执行的控制中心。

命令通道系统包含操作控制台、命令通道管理节点、命令通道服务器：

- **操作控制台**是供运维人员进行客户应用机器状态查询或者管理的操作界面。
- **命令通道管理节点**提供链接管理分配功能，客户端首先访问管理节点获取可以长链接的服务器，然后再与服务器建立长连接。
- **命令通道服务器**是真正与客户端建立长链接并进行命令下发的节点。

24.3.7 文件系统

文件系统用于存放用户上传的WAR包及JDK、Ali-Tomcat等必须组件。

专有云部署时，使用Nginx+FTP搭建的文件系统。文件系统内上传的WAR包每个应用只会保留7个最新的。

24.4 功能特性

24.4.1 全面兼容Apache Tomcat容器

作为EDAS平台应用运行的基础容器，EDAS Container集成了阿里巴巴中间件技术栈，在容器启动、容器监控、稳定性及性能上得到了极大的提升。同时，EDAS Container全面兼容Apache Tomcat。

24.4.2 以应用为中心的中间件PaaS平台

应用基本管理和运维

在EDAS平台可视化的管控界面上，您可以一站式完成应用生命周期的管控，包括创建、部署、启动、停止、扩容、缩容和应用下线等，实现对应用的全流程管理。依托阿里巴巴平台超大规模集群运维管理经验，轻松运维上千个实例的应用。

弹性伸缩

EDAS支持手动和自动两种方式来实现应用的扩容与缩容；通过对CPU、内存和负载的实时监控来实现对应用的秒级扩容和缩容。

主子账户体系

EDAS独创主子体系，你可以根据自己企业的部门划分、团队划分和项目划分在EDAS平台上建立对应的主子账号关系；同时，ECS资源也以主子账号关系进行划分，以便进行资源的分配。

角色与权限控制

应用的运维通常涉及应用研发负责人、应用运维负责人和底层机器资源负责人。不同的角色对于一个应用的管理操作各不一致，因此EDAS提供了角色和权限控制机制，方便您为不同的账号定义各自的角色，并分配相应的权限。

24.4.3 丰富的分布式服务

分布式服务框架

自2007年，伴随着阿里巴巴电商平台大规模分布式改造的持续进行，自主研发的分布式服务框架HSF（High Speed Framework）和Dubbo应运而生。HSF是一款面向企业级互联网架构的分布式服

务框架，以高性能网络通信框架为基础，提供了诸如服务发布与注册、服务调用、服务路由、服务鉴权、服务限流、服务降级和服务调用链路跟踪等一系列久经考验的功能特性。

分布式配置管理

集中式系统变成分布式系统后，如何有效的对分布式系统中每一个机器上的配置信息进行有效的实时管理成了一个难题。EDAS提供高效的分布式配置管理，能够将分布式系统的配置信息在EDAS控制台上集中管理起来，做到一处配置，处处使用。更重要的是，EDAS允许您在控制台上对配置信息进行修改，在秒级时间内就能够实时通知到所有的机器。

分布式任务调度

任务调度服务允许您配置任意周期性调度的单机或者分布式任务，并能对任务运行周期进行管理，同时提供对任务的历史执行记录进行查询。适用于诸如每天凌晨2点定时迁移历史数据，每隔5分钟进行任务触发，每个月的第一天发送系统月报等任务调度场景。

24.4.4 运维管控与服务治理

服务鉴权

HSF服务框架致力于保证每一次分布式调用的稳定与安全。在服务注册、服务订阅以及服务调用等每一个环节，都进行严格的服务鉴权。

服务限流

EDAS可以对每一个应用提供的众多服务配置限流规则，以实现对服务的流控，确保服务能够稳定运行。限流规则可以从QPS和线程两个维度进行配置，确保系统再应对流量高峰时能以最大的支撑能力平稳运行。

服务降级

与服务限流相反，每一个应用会调用许多外部服务，对于这些服务配置降级规则可以实现对劣质服务的精准屏蔽，确保应用自身能够稳定运行，防止劣质的服务依赖影响应用自身的服务能力。EDAS从响应时间维度对降级规则进行配置，在应对流量高峰时合理地屏蔽劣质依赖。

24.4.5 立体化监控与数字化运营

分布式链路跟踪

EDAS鹰眼监控系统能够分析分布式系统的每一次系统调用、消息发送和数据库访问，从而精准发现系统的瓶颈和隐患。

服务调用监控

EDAS能够针对应用的服务调用情况，对服务的QPS、响应时间和出错率进行全方面的监控。

IaaS基础监控

EDAS能够针对应用的运行状态，对机器的CPU、内存、负载、网络和磁盘等基础指标进行详细的监控。

24.5 性能指标

指标项	规格要求
访问处理性能	在简单调用场景下，不考虑不确定的服务提供方响应时间，1KB单个服务请求消息大小，RPC服务调用每核CPU的QPS为4000。可线性扩展，单个注册中心支持20000个。
基本功能	提供集成多款互联网中间件的Java容器，全面兼容Apache Tomcat。
	提供应用生命周期管理，包括应用发布、启动、停止、扩容等。
	提供对硬件基础指标的监控。
	提供对Java容器的监控。
	提供完善的服务鉴权机制。
	提供分布式RPC调用，消息和多种数据源的事务处理。
	提供完整的针对服务链路和系统指标的日志、巡检、监控和链路跟踪。
	提供分布式系统配置推送。
可靠性	数据系统采用多级缓存和主备存储方案。
扩展性	支持服务节点的无间断扩容与缩容。
技术成熟	基于阿里内部长期使用与沉淀的高可用高性能分布式集群技术产品构建，团队成员具有处理这一领域问题的丰富经验。

25 分布式关系型数据库DRDS

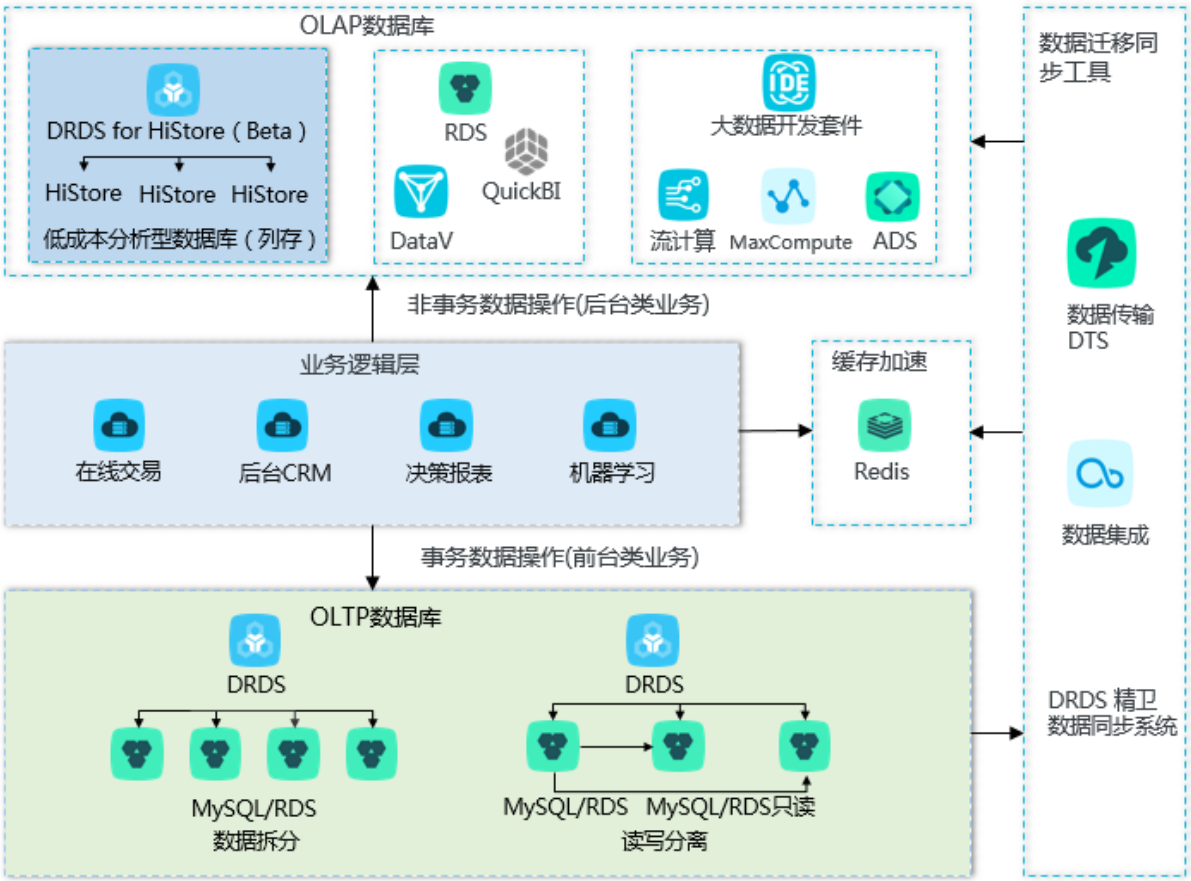
25.1 产品概述

分布式关系型数据库服务（Distributed Relational Database Service，简称DRDS）是阿里巴巴集团自主研发的中间件产品，专注于解决单机关系型数据库的扩展问题。

DRDS是阿里巴巴集团业务接入关系型数据库的标准，与淘宝TDDL（Taobao Distributed Data Layer）产品共享一份数据库拆分逻辑及团队。DRDS兼容MySQL交互协议，支持大部分的MySQL DML和DDL语法，具备分库分表、平滑扩容、服务升降配、透明读写分离等分布式数据库核心能力。产品具备轻量（无状态）、灵活、稳定和高效等特点，提供了分布式数据库全生命周期的运维管控能力。

DRDS主要应用场景在大规模在线数据操作上，偏数据落库的前台业务。通过贴合业务的拆分方式，将操作效率提升到极致，有效满足用户高并发低延迟的数据库操作需求。

图 25-1: DRDS结构示意图



DRDS主要解决了以下问题：

- 单机数据库容量瓶颈：随着数据量和访问量的增长，硬件升级并不能完全解决问题。DRDS分布式方案利用多个机器共同服务，解决了数据存储容量和访问量的瓶颈问题。
- 关系型数据库数据库扩展困难：因为分布式数据库天然属性，DRDS可以通过数据平滑挪动技术，改变数据分片存放节点，从而达到关系型数据库动态扩展问题。

25.2 产品架构

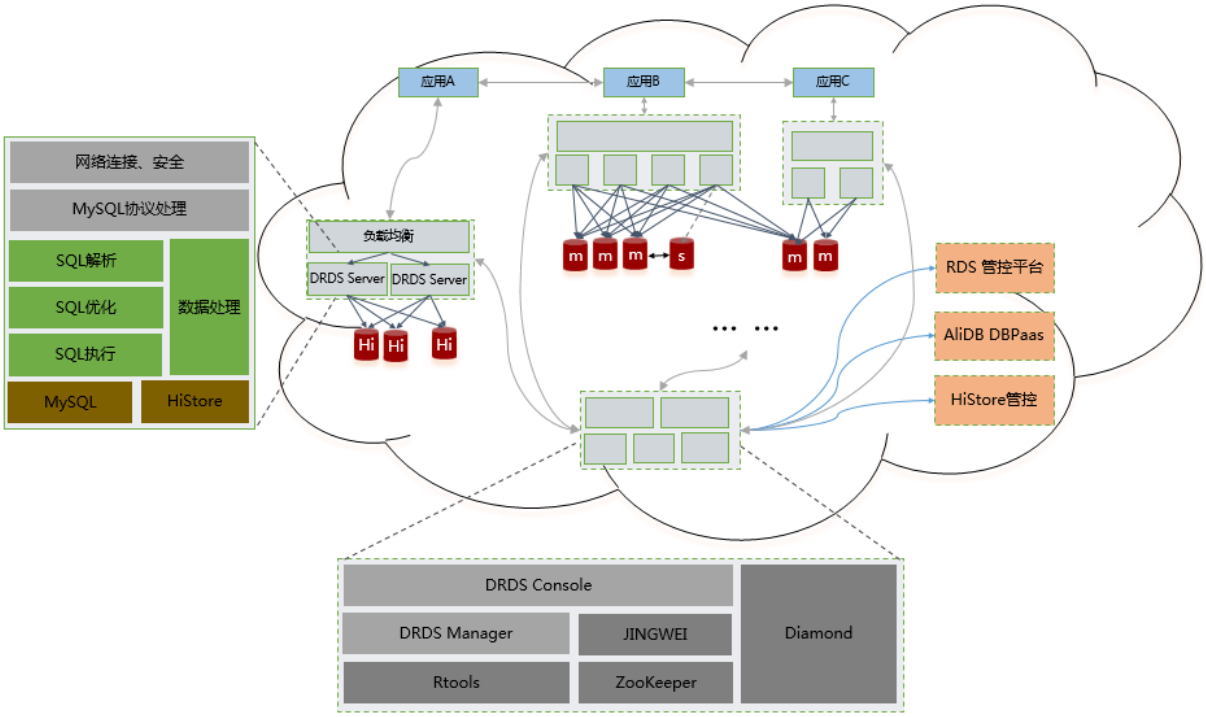
DRDS输出方式有两种：阿里云整体输出和阿里中间件单独输出。两种输出方式，在DRDS 依赖的组件上会有所不同，特性上也有部分差异。

下表简要说明了两者的区别：

差异项	阿里云整体输出	阿里中间件单独输出
MySQL	RDS for MySQL	阿里巴巴集团数据库系统（DBPaas）
负载均衡	集中式负载均衡（SLB）	客户端负载均衡（VIPServer）
特色存储支持	无	高压缩比列式存储（HiStore）

DRDS的系统架构如图 *25-2: DRDS系统架构图*所示。

图 25-2: DRDS系统架构图



DRDS Server

DRDS Server是DRDS的服务层，由多个服务节点组成服务集群，提供分布式数据库服务，包括读写分离、SQL路由执行、结果合并、动态数据库配置、全局唯一ID服务等功能。

RDS for MySQL（图中m，s表示）

RDS for MySQL主要是负责在线数据数据存储和操作，通过MySQL主备复制实现高可用，配合RDS的主备切换系统实现动态数据库故障转移。

RDS for MySQL的管理控制平台实现了RDS生命周期内的管理、监控、告警、资源管理等工作。

HiStore

当DRDS单独输出时（非阿里云整体输出），DRDS支持HiStore作为物理存储。HiStore是阿里巴巴自研的低成本高性能列存数据库。通过列式存储、知识网格、多核利用，做到相对于行存（如MySQL）更低的成本，更高的数据聚合和即席查询能力。

HiStore的管理控制平台实现了HiStore生命周期内的管理、监控、告警、资源管理等工作。

DBPaaS

当DRDS单独输出时（非阿里云整体输出），自带MySQL数据库运维平台DBPaas，实现MySQL生命周期内的管理、监控、告警、资源管理等工作。

SLB（集中式负载均衡）

无需在用户侧安装客户端，用户的请求通过集中式负载均衡进行流量分配。当某个实例节点出现故障或者新增服务节点时，都能够通过负载均衡保证底层节点的流量相对均衡分配。

VIPServer（客户端负载均衡）

需要在用户侧安装客户端，弱依赖于中心管控服务（只有在发生负载配置变更时才会进行交互），用户的请求通过该模块进行流量分配，当某个实例节点出现故障或者新增服务节点时，都能够通过均衡负载保证底层节点的流量相对均衡分配。

Diamond配置中心

Diamond（配置中心）是负责DRDS配置存储和管理的系统，提供配置存储、查询、通知功能。在DRDS中主要存储数据库源数据、拆分规则、DRDS开关等配置。

精卫（JINGWEI）

DRDS数据迁移同步主要由精卫系统负责，核心能力包含全量数据迁移和增量数据同步，衍生出来的功能包括数据平滑导入、平滑扩容、全局二级索引能力。精卫数据迁移同步系统需要ZooKeeper和DRDS Rtools提供支撑。

DRDS Console（用户控制台）

DRDS 控制台主要面向业务DBA，按照用户隔离资源和操作，提供实例管理、库表管理、读写分离配置、平滑扩容、监控展现和IP白名单等功能。

DRDS Manager（运维控制台）

DRDS Manager主要面向全局运维和DBA，提供所有DRDS资源管理和系统监控，主要功能包括两个方面：

- RDS实例依赖的所有资源管理，包括虚拟机、负载均衡、域名等资源；
- DRDS实例状态监控，包括QPS、活跃线程、连接数、各节点网络I/O、各节点CPU占用等指标。

Rtools

Rtools是DRDS的运维支撑系统，支持进行数据库配置、读写权重、连接参数等管理以及库表拓扑、拆分规则等管理。

25.3 功能特性

25.3.1 分库分表

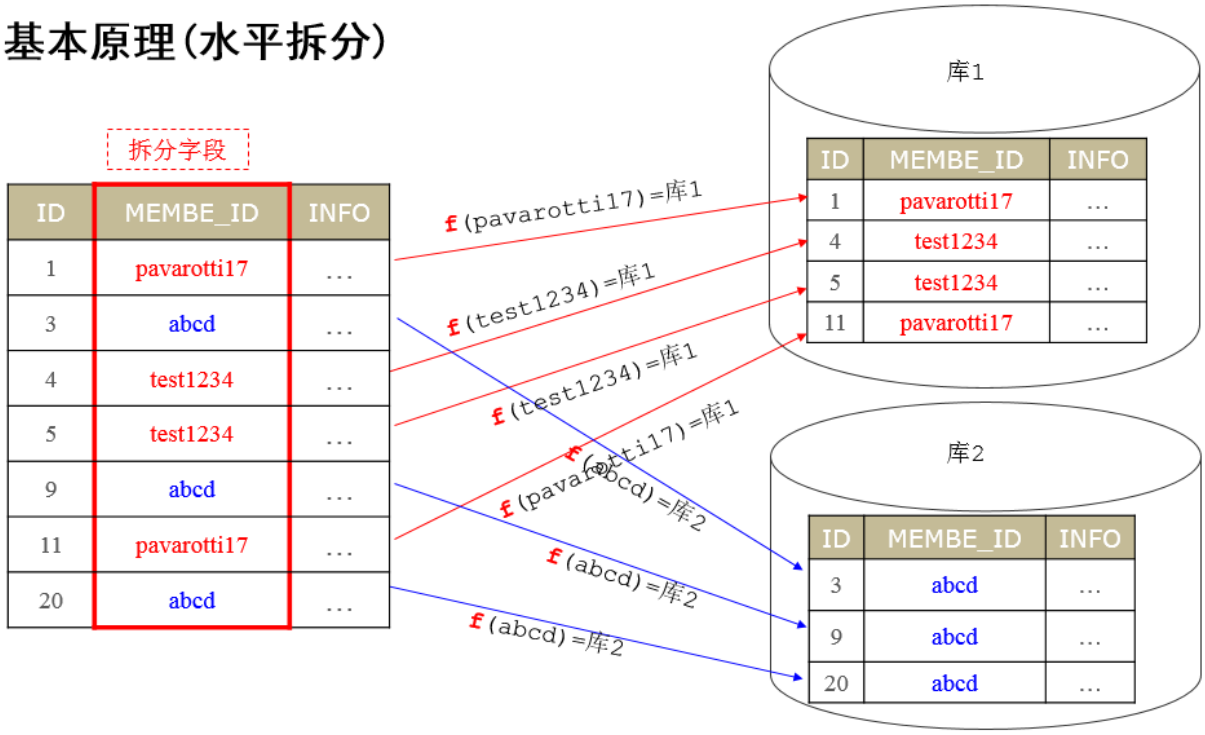
DRDS核心原理是数据水平拆分， 将数据库数据按某种规则分散存储到多个稳定的MySQL数据库上。这些MySQL数据库可分布于多台机器乃至跨机房，对外服务（增删改查）尽可能保证如同单MySQL数据库体验。拆分后，在MySQL上物理存在的数据库称为分库，物理的表称为分表（每个分表数据是完整数据的一部分）。DRDS 通过在不同MySQL实例上挪动分库，实现数据库扩容，提升DRDS数据库总体访问量和存储容量。

DRDS暴露了拆分规则，让您有机会选择贴合业务数据特点的拆分策略，让在线事务型数据库操作尽可能在高并发的场景下保持低延迟。所以使用DRDS时，如何选择拆分字段成为数据库表结构设计的重要步骤之一。总体有几条原则如下：

- DRDS最擅长前台落数据业务。这类业务大部分操作围绕某一个数据库主体展开，比如互联网业务典型的按照用户展开业务操作，物联网围绕设备、车辆等展开业务操作，银行政府机构柜面类业务围绕客户展开业务操作，电商ISV、餐饮ISV围绕商家展开业务操作等。这类业务数据可以按围绕的数据库主体进行拆分，配合全局二级索引和最终一致事务，能够很好地解决业务超大数据量、高并发、低延迟数据库使用需求 。
- 后台类业务中，按条件组合过滤出一批数据分页展示，并且处理数据后写回数据库是DRDS能够部分解决的业务场景。这时可能存在大量单表和多表关联，多种过滤条件组合删选，多表事务处理等。这类场景首要拆分方式还是推荐数据主体拆分。如果数据库处理和时间紧相关，也可以按时间拆分。

数据拆分原理如图 *25-3:* 数据拆分示意图所示。

图 25-3: 数据拆分示意图



25.3.2 多种水平拆分方式

DRDS支持多种贴合业务的拆分方式，各有特点。拆分方式有2个要素：拆分字段和拆分函数，前者决定计算参数，后者决定计算方法。如果不指定拆分字段，则为单表，即不拆分，默认放在_0000库。

拆分字段包含分库字段和分表字段，可设置成一样，也可以不一样，目前支持的组合是：

分库字段	分表字段	分库和分表字段相同
无	无	无
数字、字符串	无	无
数字、字符串	数字、字符串	相同
数字、字符串	数字、字符串	不同
数字、字符串	日期	不同

拆分函数可以用于分库，也可以用于分表，说明和约束如下表所示：

拆分函数	描述	是否支持用于分库	是否支持用于分表
HASH	简单取模	是	是

拆分函数	描述	是否支持用于分库	是否支持用于分表
UNI HASH	简单取模	是	是
RIGHT_SHIFT	数值向右移	是	是
RANGE_HASH	双拆分列哈希	是	是
MM	按月份哈希	否	是
DD	按日期哈希	否	是
WEEK	按周哈希	否	是
MMDD	按月日哈希	否	是
YYYYMM	按年月哈希	是	是
YYYYWEEK	按年周哈希	是	是
YYYYDD	按年日哈希	是	是
YYYYMM_OPT	按年月哈希，改进型	是	是
YYYYWEEK_OPT	按年周哈希，改进型	是	是
YYYYDD_OPT	按年日哈希，改进型	是	是

25.3.3 平滑扩容

DRDS扩容通过增加RDS/MySQL实例数，将原有的分库迁移到新的RDS/MySQL实例上，达到扩容的目标。

DRDS扩容原理

步骤如下：

1. 创建扩容计划

   选择新增加RDS/MySQL，并选定需要迁移到新RDS/MySQL实例上的分库，提交任务后系统自动在目标RDS/MySQL上创建数据库和账号，并提交任务进行数据迁移同步。

2. 全量迁移

   系统选择当前时间之前的一个时间点，将这个时间点之前的数据进行全量的数据复制迁移。

3. 增量数据同步

   完成全量迁移后，基于全量迁移开始之前时间点的增量变更日志进行增量同步，最终原分库和目标分库数据实时同步。

4. 数据校验

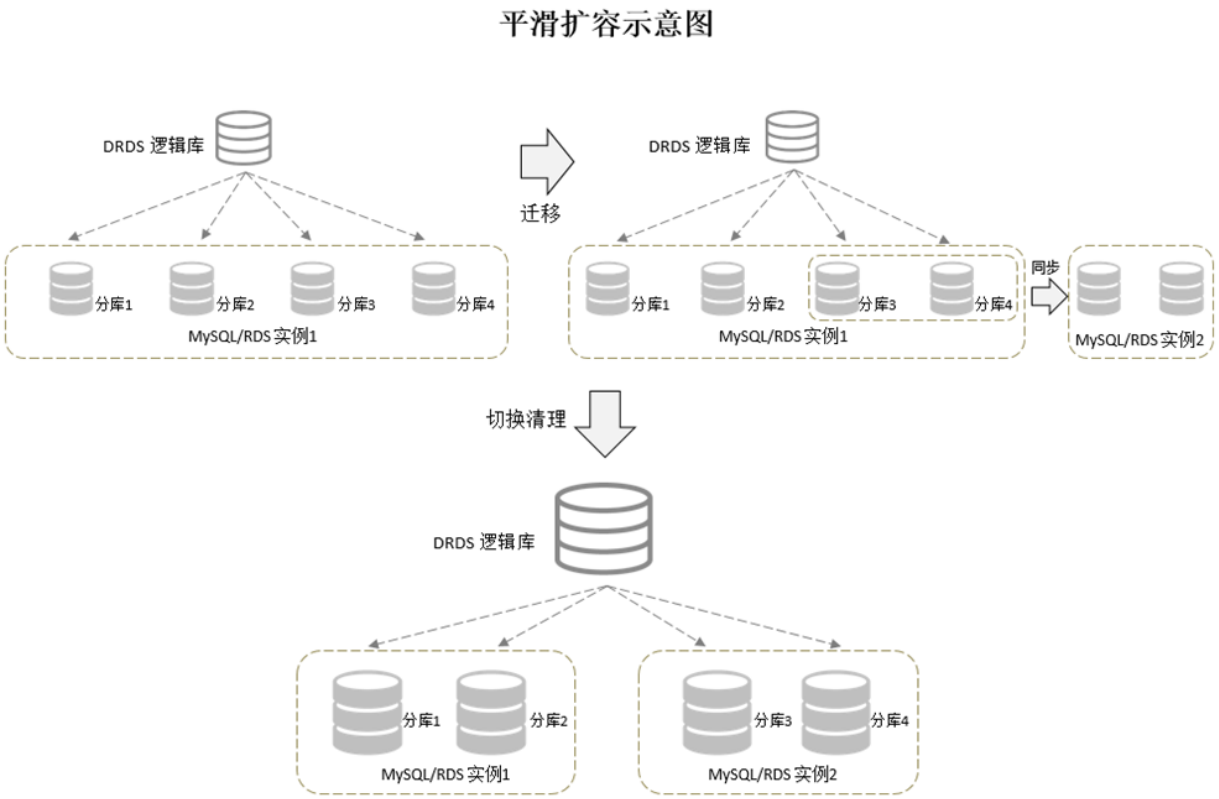
增量达到准实时同步后，系统自动做全数据校验，并且订正因为同步延迟造成的不一致数据。

5. 应用停写和路由切换

校验完成后，并且增量依然维持准实时同步，业务选定时间进行切换，为确保数据严格一致，建议应用停服（也可以不停，但可能面临同一条数据高并发写入覆盖问题），引擎层进行分库规则的路由切换，将后续流量转向新库，切换过程秒级完成。

分库迁移示意图如图 *25-4:* 扩容示意图所示。

图 25-4: 扩容示意图



为了保证数据本身的安全，便于扩容回滚，在路由规则切换完成后，数据同步依然会运行，直到数据运维人员确认服务正常后在控制台主动发起旧分库数据的清理。

整个扩容过程对上层的业务正常服务几乎没有影响（如果RDS/MySQL实例规格过小或者压力过大则可能造成部分影响），切换时如果应用不停服，建议操作选择在数据库访问低谷期进行，降低同一条数据并发更新覆盖的概率。

25.3.4 读写分离

DRDS的读写分离功能是基于RDS/MySQL只读实例所做的一种相对透明读流量切换策略。

业务应用在能够忍受只读实例相对于主实例数据同步延迟的前提下，不需要修改代码，即可在DRDS控制台中增加RDS/MySQL只读实例和调整读权重，将读流量按照需要的比例在RDS/MySQL主实例与多个RDS/MySQL只读实例之间调整，写操作和事务操作则统一走RDS/MySQL主实例。

需要注意的是，主RDS/MySQL实例和只读RDS/MySQL存在数据同步延迟，并且在发生大的DDL或者数据订正时，有可能导致分钟级别以上的延迟，所以需要业务忍受该情况所带来的影响。

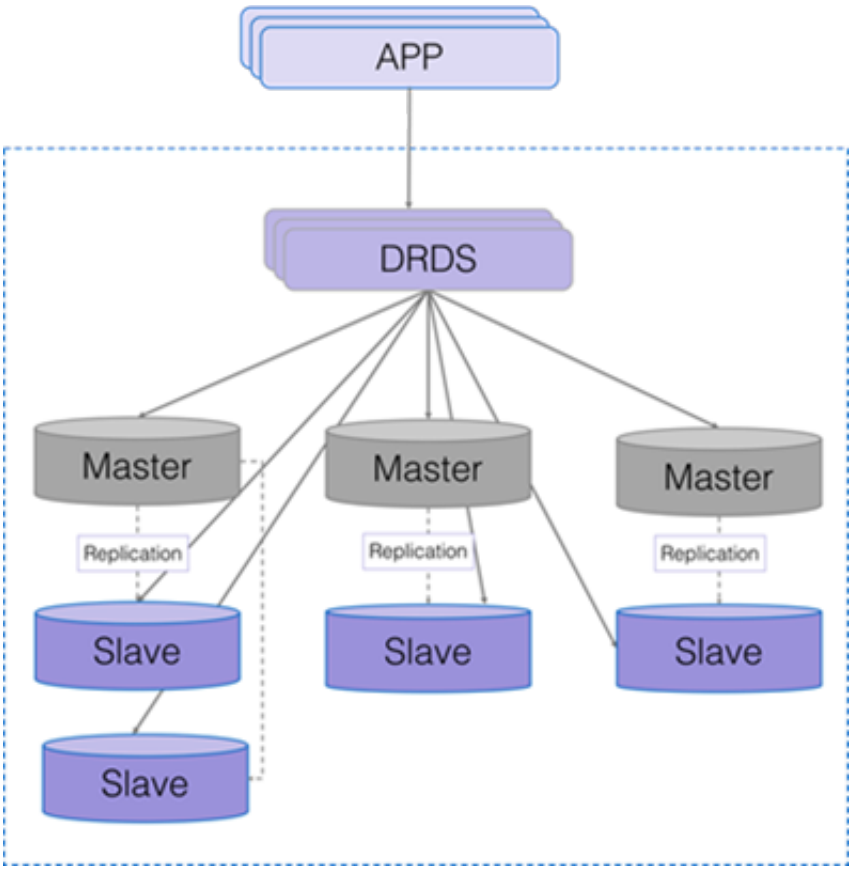
添加只读实例可以使读性能线性提升。例如在初始有一个只读实例的情况下，挂载一个只读实例，读性能提升至原来两倍，挂载2个只读实例，则读性能为单个主库读性能的三倍。

读写分离流量分配与扩展

DRDS读写分离应用层不需要修改任何代码，只需要在DRDS控制台中增加只读实例和调整读权重，即可将读流量按照需要的比例在主实例与多个只读实例之间调整，写操作则统一走主实例。

添加只读实例可以使读性能线性提升。例如在初始有一个只读实例的情况下，挂载一个只读实例，读性能提升至原来2倍，挂载2个只读实例为单个主库的3倍。如图 *25-5:* 读写分离流量分配与扩展示意图所示。

图 25-5: 读写分离流量分配与扩展示意图



读实例上读所操作的数据都是从主实例上异步同步的，存在毫秒级别延迟，对于实时性要求特别强的SQL可以通过DRDS Hint指定主库执行，如下所示：

```
/*TDDL:MASTER/select * from tddl5_users;
```

DRDS支持通过SHOW NODE指令查看实际读流量分布。如图 *25-6: SHOW NODE*指令查看实际读流量分布所示。

图 25-6: SHOW NODE指令查看实际读流量分布

```
mysql> show node;
+------+------------------+------------------+------------------+--------------------+--------------------+
| ID   | NAME             | MASTER_READ_COUNT | SLAVE_READ_COUNT | MASTER_READ_PERCENT | SLAVE_READ_PERCENT |
+------+------------------+------------------+------------------+--------------------+--------------------+
| 0    | USERDATABASE_RDS |               10 |                2 |                83% |                17% |
+------+------------------+------------------+------------------+--------------------+--------------------+
1 row in set (0.00 sec)
```

非拆分模式的读写分离

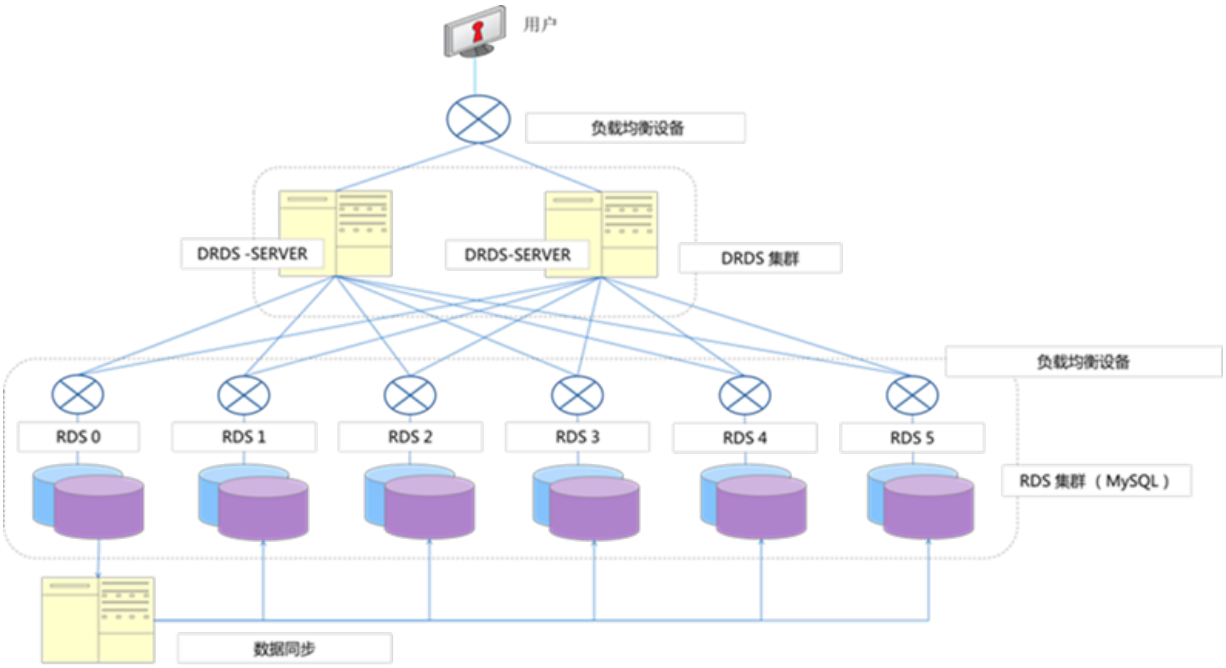
DRDS的读写分离可以在非拆分模式下独立使用。

DRDS控制台上创建DRDS数据库时，在选定一个数据库实例的情况下，可以选择将底层数据库实例下的一个逻辑数据库直接引入DRDS做读写分离，不需要做数据迁移。

25.3.5 服务升降配

DRDS Server层通过集群方式部署，由多个服务节点构成一个服务实例，通过负载均衡与DNS对外提供服务。DRDS的多个服务节点之间无状态同步，均衡处理外部请求。当服务集群的处理能力不足的时候，支持实时增加服务节点，扩展服务能力。DRDS服务层资源都利用率比较低的情况下，支持降低集群规模，降低服务层服务能力，做到服务能力弹性扩展。如图 25-7: 服务升降配示意图所示。

图 25-7: 服务升降配示意图



25.3.6 账号和权限系统

DRDS账号权限系统用法和MySQL一致，但是不支持跨多个数据库的授权，并且权限丰富度上也相较MySQL有一定的精简，支持 GRANT、REVOKE、SHOW GRANTS、CREATE USER、DROP USER、SET PASSWORD 等语句，目前支持库级和表级权限的授予，全局级别和列级别权限暂时不支持。

账号规则中：

- 控制台上创建的管理员账号具有所有权限；

- 只有管理员账号具有创建账号和授权功能，其它账号只能由管理员账号创建，其权限只能由管理员账号授予；
- 管理员账号是跟数据库绑定的，没有其它数据库的权限，连接的时候只能连接自己对应的那个数据库，不能授予其它数据库的权限给某个账号。例如 EasyDB这个管理员账号只能连接 EasyDB 数据库，只能授予 EasyDB 数据库权限或者 EasyDB 数据库中表的权限给某个账号。

权限目前支持和表相关联的8个基本权限项：
CREATE、DROP、ALTER、INDEX、INSERT、DELETE、UPDATE、SELECT。其中：

- TRUNCATE 操作需要有表上的 DROP 权限；
- REPLACE 操作需要有表上的 INSERT 和 DELETE 权限；
- CREATE INDEX 和 DROP INDEX 操作需要有表上的 INDEX 权限；
- CREATE SEQUENCE 需要有数据库级的创建表（CREATE）权限；
- DROP SEQUENCE 需要有数据库级的删除表（DROP）权限；
- ALTER SEQUENCE 需要有数据库级的更改表（ALTER）权限；
- INSERT ON DUPLICATE UPDATE 语句需要有表上的 INSERT 和 UPDATE 权限。

25.3.7 全局唯一数字序列

DRDS全局唯一数字序列（64位数字，对应MySQL中Signed BIGINT类型）的主要目标是为了生成全局唯一的数字序列（不保证递增），通常用于主键列、唯一键等值的生成。

DRDS 全局唯一数字序列能够被隐式使用，在表被创建为拆分表，并且主键为auto_increment，业务插入数据，并且没有指定主键的情况下，自动填充全局唯一主键，如同单机MySQL体验。

DRDS 全局唯一数字序列也可以被显式使用，您可以通过`select xxx_seq.nextval from dual where count= ?`单个或批量获取全局唯一数字序列，在应用中另作他用。

25.3.8 秒级监控

DRDS支持通过指令 **`SHOW FULL STATS`**支持秒级监控，配合业务自身监控系统，或者对接第三方开源监控软件，达到比较好的监控告警效果。

指令支持的指标和说明如下表所示：

指标	说明
QPS	应用到DRDS的QPS, 通常称为逻辑QPS
RDS_QPS	DRDS到RDS的QPS，通常称为物理QPS

指标	说明
ERROR_PER_SECOND	每秒错误数量，包含SQL语法错误、主键冲突、系统错误、连通性错误等各类错误总和
VIOLATION_PER_SECOND	每秒主键或者唯一键冲突数量
MERGE_QUERY_PER_SECOND	通过分库分表，从多表中进行的查询
ACTIVE_CONNECTIONS	正在使用的连接数
CONNECTION_CREATE_PER_SECOND	每秒创建的连接数
RT(MS)	应用到DRDS的响应时间，通常称为逻辑RT
RDS_RT(MS)	DRDS到RDS/MySQL的响应时间，通常称为物理RT
NET_IN(KB/S)	DRDS收到的网络流量
NET_OUT(KB/S)	DRDS输出的网络流量
THREAD_RUNNING	正在运行的线程数
HINT_USED_PER_SECOND	每秒带HINT的查询总数
HINT_USED_COUNT	从启动到现在带HINT的查询总数
AGGREGATE_QUERY_PER_SECOND	每秒聚合查询数量
AGGREGATE_QUERY_COUNT	聚合查询总数（历史计数）
TEMP_TABLE_CREATE_PER_SECOND	每秒创建的临时表数量
TEMP_TABLE_CREATE_COUNT	从启动到现在创建的临时表总数量
MULTI_DB_JOIN_PER_SECOND	每秒跨库JOIN数量
MULTI_DB_JOIN_COUNT	从启动到现在为止跨库JOIN总量

25.3.9 分布式SQL引擎

DRDS分布式SQL引擎目标是实现与单机MySQL数据库的高度兼容，实现分布式SQL操作下推（offload）。DRDS层面对SQL的操作包括SQL分析、SQL优化、SQL路由和数据聚合计算等步骤。

分布式SQL下推核心原则有如下几个：

- 尽可能贴近数据做数据处理；
- 减少数据的网络传输；
- 减少DRDS层面的计算量，尽量将计算下推到下层的数据节点上；
- 充分发挥数据库存储的特性和能力。

25.3.10 高可用架构

DRDS Server自动切流

DRDS实例由多个DRDS Server（服务进程）构成，通过负载均衡设备以单一连接串的方式提供服务。当一个DRDS Server出现故障时，流量以秒级切换到其他DRDS Server上。整个切换过程对用户透明，应用代码无需变更，应用进程无需重启。

只读实例自动切流

DRDS支持读写分离功能，可以在控制台的 **DRDS数据库 > 读写分离**页面进行配置。将部分读流量分配到备实例上。DRDS将识别出读SQL命令请求，并按照配置的比例下发到主、备RDS/MySQL实例执行，达到读写分离的目的。分配了读流量的备实例称为只读实例。

如果配置了多个只读实例，当其中一个只读实例出现故障（具体表现为无法连接）时，DRDS会自动收回故障实例上的读流量，并按照剩余正常只读实例的读流量比例重新分配执行。

只读实例自动切流过程对用户透明，应用无需重启。当所有只读实例都不可用时，为了防止主实例压力过大，将仍然按比例分配读SQL命令请求到只读实例和主实例上，并对分配到只读实例上的读SQL命令请求快速报错。

说明：

所有写SQL命令请求和事务均自动下发到主实例上执行，与只读实例的可用性无关。

25.3.11 软件升级

- DRDS定期提供数据库软件的新版本。
- 版本升级是非强制性的，只有您主动要求，才会升级到指定版本。
- 当DRDS评估您的版本存在重大安全隐患时，会主动通知您安排时间进行升级。DRDS团队将会全程支持升级过程。
- DRDS升级过程通常在五分钟以内完成，升级期间可能有数次数据库连接闪断。在应用程序正确配置了数据库连接重连（或连接池）的情况下，不会对应用程序造成明显的影响。

25.3.12 SQL兼容性

DRDS兼容MySQL交互协议，支持大部分MySQL查询语法，支持常用的DML语法和DDL语法。但由于分布式数据库和单机数据库存在较大的架构差异，存在SQL使用限制，相关兼容性和SQL限制描述如下。

说明：

因为MySQL历史版本众多，语法也在革新，DRDS版本也在不断革新，故此文档兼容性只做参考，具体是否完全匹配业务请以实际测试为准。

DRDS SQL限制

SQL大类限制

- 暂不支持用户自定义数据类型和自定义函数。
- 暂不支持视图、存储过程、触发器和游标。
- 暂不支持`BEGIN...END`、`LOOP...END LOOP`、`REPEAT...UNTIL...END REPEAT`、`WHILE...DO...END WHILE`等复合语句。
- 暂不支类似IF，WHILE等流程控制类语句。

小语法限制

DDL

- CREATE TABLE tbl_name LIKE old_tbl_name不支持拆分表。
- CREATE TABLE tbl_name SELECT statement不支持拆分表。

DML

- 暂不支持SELECT INTO OUTFILE/INTO DUMPFILE/INTO var_name。
- 暂不支持INSERT DELAYED语法。
- 暂不支持非WHERE 条件的相关子查询。
- 暂不支持SQL中带聚合条件的相关子查询。
- 暂不支持SQL中对于变量的引用和操作，比如`SET @c=1, @d=@c+1; SELECT @c, @d`。

数据库管理

- SHOW WARNINGS语法不支持LIMIT/COUNT的组合。
- SHOW ERRORS语法不支持LIMIT/COUNT的组合。

DRDS SQL兼容

协议兼容

DRDS支持MySQL Workbench，Navicat For MySQL，SQLyog等主流客户端。

说明：

DRDS支持支持通用的数据库增删改查操作，而各种额外的特色功能（比如导入、诊断等）没有进行过精确测试。

DDL语法兼容

- `CREATE TABLE` 语法
- `CREATE INDEX` 语法
- `DROP TABLE` 语法
- `DROP INDEX` 语法
- `ALTER TABLE` 语法
- `TRUNCATE TABLE` 语法

DML语法兼容

- INSERT 语法
- REPLACE 语法
- UPDATE 语法
- DELETE 语法
- Subquery 语法
- Scalar 子查询
- Comparisons 子查询
- 带ANY、IN或者SOME的子查询
- 带的ALL的子查询
- 列子查询
- 带EXISTS或者NOT EXISTS的子查询
- FROM子句中的子查询
- SELECT语法

Prepare语法兼容

- PREPARE 语法
- EXECUTE 语法
- DEALLOCATE PREPARE 语法

数据库管理语法兼容

- SET
- SHOW
- KILL 'PROCESS_ID' （DRDS不支持KILL QUERY指令，只支持KILL 'PROCESS_ID’）
- SHOW COLUMNS
- SHOW CREATE TABLE
- SHOW INDEX
- SHOW TABLES
- SHOW TABLE STATUS
- SHOW TABLES
- SHOW VARIABLES
- SHOW WARNINGS
- SHOW ERRORS

注意：

其他SHOW指令会默认下发到DB处理，返回来的结果数据没有进行分库数据合并。

数据库工具指令

- DESCRIBE 语法
- EXPLAIN 语法
- USE 语法

DRDS自定义指令

- SHOW SEQUENCES / CREATE SEQUENCE / ALTER SEQUENCE / DROP
- SEQUENCE 【 DRDS全局唯一数字序列管理】
- SHOW PARTITIONS FROM TABLE 【查询表的拆分字段】
- SHOW TOPOLOGY FROM TABLE 【查询表的物理拓扑】
- SHOW BROADCASTS 【查询所有广播表】
- SHOW RULE [FROM TABLE] 【查询表拆分定义】

- SHOW DATASOURCES 【查询后端DB连接池定义】
- SHOW DBLOCK / RELEASE DBLOCK 【分布式LOCK定义】
- SHOW NODE 【查询读写库流量】
- SHOW SLOW 【查询慢SQL列表】
- SHOW PHYSICAL_SLOW 【查询物理DB执行慢SQL列表】
- TRACE SQL_STATEMENT / SHOW TRACE 【跟踪SQL整个执行过程】
- EXPLAIN [DETAIL/EXECUTE] SQL_STATEMENT 【分析DRDS执行计划和物理DB上的执行计划】
- RELOAD USERS 【同步DRDS控制台用户信息到DRDS SERVER】
- RELOAD SCHEMA 【清理DRDS对应DB库数据缓存，比如SQL解析/语法树/表结构缓存】
- RELOAD DATASOURCES 【重建后端与所有DB的连接池】

数据库函数

- 带拆分键的SQL，所有MySQL函数支持
- 不带拆分键的SQL，部分函数支持。
- 操作符函数

函数	描述
AND，&&	逻辑与
=	赋值（作为SET语句的一部分，或者作为UPDATE语句中SET子句的一部分）
BETWEEN... AND...	确定一个值在某个范围
BINARY	将字符串转换成二进制字符串
&	按位与
~	按位取反
^	按位异或
DIV	整数除
/	除操作符
<=>	NULL-safe 等号
=	等号
>=	大于等于号

函数	描述
>	大于号
IS NOTNULL	非NULL值检测
ISNOT	布尔非
ISNULL	NULL值检测
IS	布尔检测
<<	按位左移
<=	小于等于号
<	小于号
LIKE	简单模式匹配
-	减号
%,	取模
NOTBETWEEN... AND...	确定一个值不在某个范围内
!=，<>	不等于
NOTLIKE	简单模式匹配非
NOTREGEXP	负的REGEXP.
NOT，！	非
OR	逻辑或
+	加号
REGEXP	使用正则表达式的模式匹配
>>	按位右移
RLIKE	等同于REGEXP.
*	乘号
-	改变参数的符号，取反。
XOR	逻辑异或
Coalesce	返回第一个非NULL参数
GREATEST	返回最大的参数
LEAST	返回最小的参数

函数	描述
STRCMP	比较两个字符串

- 流程控制函数

函数	描述
CASE	Case 操作符
IF()	If/else 结构.
IFNULL()	Null if/else 结构
NULLIF()	如果expr1 =expr2，返回NULL

- 数学函数

函数	描述
ABS()	返回绝对值
ACOS()	返回反余弦值
ASIN()	返回反正弦值
ATAN2()	返回两个参数的反正切值
ATAN()	返回参数的反正切值
CEIL()	向上取整
CEILIG()	向上取整
CONV()	在不同的数字基数之间对数字进行转换。
COS()	返回余弦值
COT()	返回余切值
CRC32()	计算CRC（循环冗余校验）值
DEGREES()	将弧度转换成度数
DIV	整数除
EXP()	以e为底的指数函数
FLOOR()	向下取整
LN()	返回参数的自然对数
LOG10()	返回参数的底数为10的对数
LOG2()	返回参数的底数为2的对数

函数	描述
LOG()	返回第一参数的自然对数
MOD()	取余
%,MOD	取模
PI()	返回PI值
POW()	返回第一参数的N次幂，其中N为第二个参数
POWER()	返回第一参数的N次幂，其中N为第二个参数
RADIANS()	将参数转换成弧度
RAND()	返回随机浮点数
ROUND()	取整
SIGN()	返回参数的符号
SIN()	返回参数的正弦值
SQRT()	返回参数的平方根
TAN()	返回参数的正切值
TRUNCATE(	截断到指定的小数位数

- 字符串函数

函数	描述
ASCII()	返回字符的ASCII码值
BIN()	返回字符的二进制值
BIT_LENGTH()	返回字符串的比特长度
CHAR_LENGTH()	返回字符串的长度，按字符个数计
CHAR()	将传入的整数转换成字符
CHARACTER_LENGTH()	等同于 CHAR_LENGTH()
CONCAT_WS()	用指定分隔符将传入的参数连接起来
CONCAT()	返回连接字符串
ELT()	返回索引号处的字符串
EXPORT_SET()	
FIELD()	返回首个参数在后续参数中的索引位置

函数	描述
FIND_IN_SET()	返回首个参数在第二个参数中的索引位置
FORMAT()	返回指定小数点后位数的格式化的数字
HEX()	将传入的十进制数或字符串转换成十六进制表示方式
INSERT()	在指定的位置插入指定字符数的子字符串
INSTR()	返回子字符串首次出现的索引位置
LCASE()	等同于LOWER()
LEFT()	返回最左侧指定数目的字符
LENGTH()	返回字符串的长度，以字节为计数
LIKE	简单模式匹配
LOCATE()	返回子字符串首次出现的位置
LOWER()	转换成小写字母
LPAD()	返回字符串参数，用指定的字符串左填充
LTRIM()	删除头部的空格
MAKE_SET()	返回一个用逗号隔开的，包含位集合中对应位的字符串的集合
MID()	返回从指定位置开始的子字符串
NOTLIKE	简单模式匹配非
NOTREGEXP	正则表达式非
OCT()	返回一个字符串，将一个数字转换为八进制表示方式
OCTET_LENGTH()	等同于 LENGTH()
ORD()	返回参数最左边字符的字符编码
POSITION()	等同于 LOCATE()
QUOTE()	将参数转义以便用于SQL语句
REPEAT()	按指定次数重复一个字符串
REPLACE()	替换指定字符串出现的所有地方
REVERSE()	反转字符串中的字符
RIGHT()	返回最右侧指定数目的字符
RPAD()	按照指定次数追加字符串
RTRIM()	删除尾部的空格

函数	描述
SPACE()	返回由指定空格数组成的字符串
STRCMP()	比较两个字符串
SUBSTR()	返回指定的子字符串。
SUBSTRING_INDEX()	返回字符串中指定出现次数的分隔符之前的子字符串
SUBSTRING()	返回指定的子字符串
TRIM()	删除头部和尾部的空格
UCASE()	等同于 UPPER()
UNHEX()	返回一个字符串，为一个数字的十六进制表示
UPPER()	转换为大写字母

- 时间函数

函数	描述
ADDDATE()	将一个时间值（间隔）加到一个日期上
ADDTIME()	加上时间
CURDATE()	返回当前日期
CURRENT_DATE()	CURRENT_DATE 等同于 CURDATE()
CURRENT_TIME()	CURRENT_TIME 等同于 CURTIME()
CURRENT_TIMESTAMP()	CURRENT_TIMESTAMP 等同于 NOW()
CURTIME()	返回当前时间
DATE_ADD()	将一个时间值（间隔）加到一个日期上
DATE_FORMAT()	根据指定格式化日期
DATE_SUB()	从一个日期上减去指定的时间值（间隔）
DATE()	从一个日期表达式或者日期时间表达式中提取日期的部分
DATEDIFF()	两个日期相减
DAY()	等同于 DAYOFMONTH()
DAYNAME()	返回日期的周名称
DAYOFMONTH()	返回日期是月份的第几天（0-31）
DAYOFWEEK()	返回日期是周的第几天（其中周日为1，以此类推，周六为7）.

函数	描述
DAYOFYEAR()	返回日期是年的第几天（1-366）
EXTRACT()	从日期中提取某一部分
FROM_DAYS()	将天数转换成日期
FROM_UNIXTIME()	将UNIX时间戳格式化成日期
GET_FORMAT()	返回日期格式字符串
HOUR()	从传入的时间参数中提取小时的数据
LAST_DAY()	返回参数所在月份的最后一天
LOCALTIME()	LOCALTIME 等同于 NOW()
LOCALTIMESTAMP, LOCALTIMESTAMP()	等同于 NOW()
MAKEDATE()	提供年份和天数，返回日期
MAKETIME()	提供小时、分钟、秒钟，构建一个时间
MICROSECOND()	返回参数的微秒
MINUTE()	返回参数的分钟
MONTH()	返回传入日期的月份
MONTHNAME()	返回月份的名称
NOW()	返回当前日期和时间
PERIOD_ADD()	将一段时间加到一个年月格式的日期上
PERIOD_DIFF()	返回两个时期之间的月数
QUARTER()	返回日期参数所处的季度
SEC_TO_TIME()	将秒转换成'HH:MM:SS'格式
SECOND()	返回时间秒部分的数据（0-59）
STR_TO_DATE()	将字符串转换成日期
SUBDATE()	当调用三个参数时，等同于DATE_SUB()
SUBTIME()	时间相减
SYSDATE()	返回函数执行的时间
TIME_FORMAT()	格式化时间
TIME_TO_SEC()	将参数转换成秒

函数	描述
TIME()	提取传入参数的时间部分
TIMEDIFF()	时间相减
TIMESTAMP()	当有一个参数时，该函数返回日期或者日期时间表达式，当有两个参数时，返回参数的和
TIMESTAMPADD()	在日期时间表达式上加上一段时间间隔
TIMESTAMPDIFF()	在日期时间表达式上减去一段时间间隔
UNIX_TIMESTAMP()	返回UNIX时间戳
UTC_DATE()	返回当前的UTC日期
UTC_TIME()	返回当前的UTC时间
UTC_TIMESTAMP()	返回当前的UTC日期和时间
WEEKDAY()	返回周索引，其中周日为1，以此类推，周六为7
WEEKOFYEAR()	返回日期的日历周数
YEAR()	返回年份

- 类型转换函数

函数	描述
BINARY	将字符串转换成二进制字符串
CAST()	将一个值转换成某种类型
CONVERT()	将一个值转换成某种类型

25.3.13 表单拆分

DRDS为用户提供了便捷的表单拆分和变更功能，支持单表转分表、分表转单表、分表转分表三种变更方式，拆分变更灵活自主。

25.3.14 多可用区实例

DRDS 支持选择跨多可用区的 DRDS 实例，在单一区域出现不可用的情况下，可确保 DRDS 实例级别的可用性。

25.3.15 可用区迁移

DRDS 提供可用区迁移功能，支持单可用区与双可用区之间的双向迁移。用户可在 DRDS 可用区选择错误或RDS 可用区库存不足的情况下进行 DRDS 可用区迁移。

26 消息队列MQ（铂金版）

26.1 什么是消息队列MQ

本节主要介绍消息队列MQ的定义。

消息队列MQ（Message Queue）是阿里巴巴集团自主研发的专业消息中间件。 产品基于高可用分布式集群技术，提供消息订阅和发布、消息轨迹查询、定时（延时）消息、资源统计、监控报警等一系列消息云服务，是企业级互联网架构的核心产品。 MQ历史超过9年，为分布式应用系统提供异步解耦、削峰填谷的能力，同时具备海量消息堆积、高吞吐、可靠重试等互联网应用所需的特性，是阿里巴巴双11使用的核心产品。

MQ是阿里云正式商用的产品，支持在多个地域（Region）部署高可用消息云服务，单个域内支持多机房部署，可用性极高，即使整个机房都不可用，仍然可以为应用提供消息发布服务，产品稳定性及可用性完全按照阿里巴巴内部标准来实施，无单点。

MQ目前提供TCP和MQTT两种协议层面的接入方式，支持Java、C++以及.NET不同语言，方便不同编程语言开发的应用快速接入MQ消息云服务。 用户可以将应用部署在阿里云ECS、企业自建云，或者嵌入到移动端、物联网设备中与MQ建立连接进行消息收发，同时本地开发者也可以通过公网接入MQ服务进行消息收发。

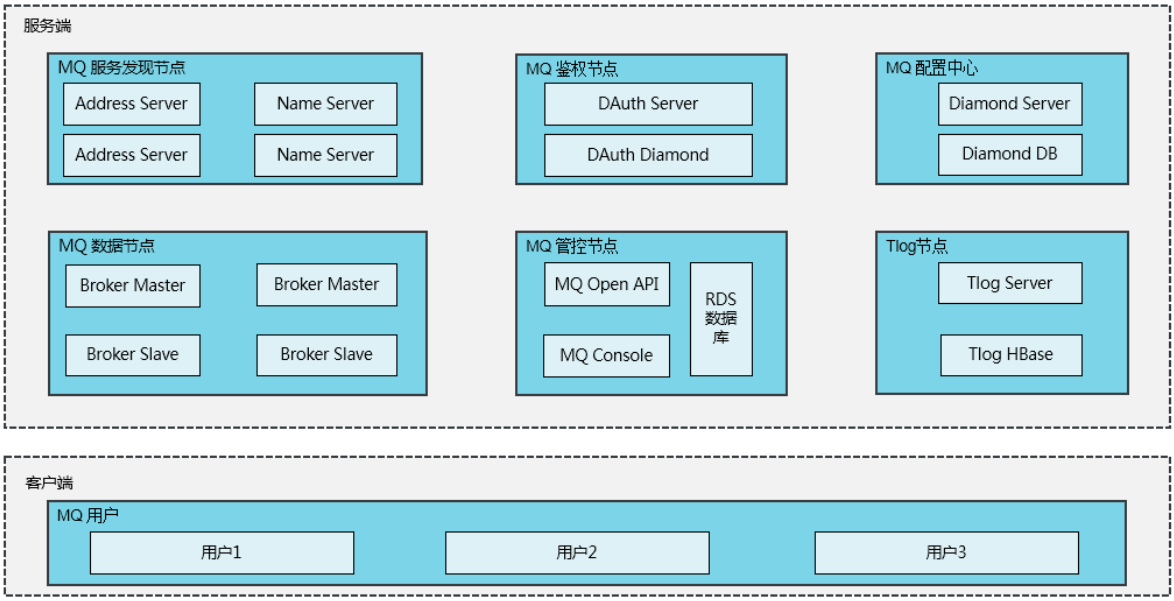
26.2 产品架构

本文向您介绍MQ的系统架构。

系统架构

MQ主要由MQ服务发现节点、MQ Broker、MQ Console、MQ Open API、MQ鉴权节点、MQ配置中心以及Tlog等系统模块组成。

图 26-1: MQ系统架构图



MQ服务发现节点

MQ服务发现节点由Name Server和Address Server组成。Name Server负责消息队列服务的注册与查找，是实现消息队列服务弹性部署和线性扩展的核心。

Name Server几乎无状态节0点，节点之间不需要进行数据同步，可集群部署横向扩展。

Address Server，又称Cai，是消息队列MQ的地址服务器，主要负责Name Server的域名注册和发现。

MQBroker

消息队列服务器（MQ Broker）是消息队列的核心处理模块，由多个服务节点组成服务集群，负责消息的收发以及消息的存储。MQBroker支持集群化、多副本部署，主备复制实现高可用。提供高可用、稳定高效、可线性扩容的消息服务能力。MQ Broker将消息队列将Topic信息、订阅信息等注册到Name Server上，提供服务。

MQ Console

MQ Console（控制台）为用户提供消息的Topic管理、生产者管理、消费者管理、消息查询、消息轨迹查询、资源报表、以及监控告警等一整套完备的运维功能。用户可以通过MQ Console，快速接入消息队列、对用户资源进行管理、问题排查以及通过监控告警等服务帮助用户及时的发现问题。

MQ Open API

为方便用户接入、自主运维，MQ Open API通过HTTP/HTTPS接口的形式提供一整套API服务，用户可以通过Open API创建Topic、生产者信息、消费者信息、消息查询以及消费者状态查询等。

MQ鉴权节点

MQ鉴权节点（DAuth）为消息队列提供统一登录服务以及安全访问控制。包括资源的权限控制、跨账号与子账号访问控制、资源授权等功能。

MQ配置中心

MQ配置中心（Diamond）主要负责存储消息队列的配置信息，如VIP转换规则、资源权限信息等数据。

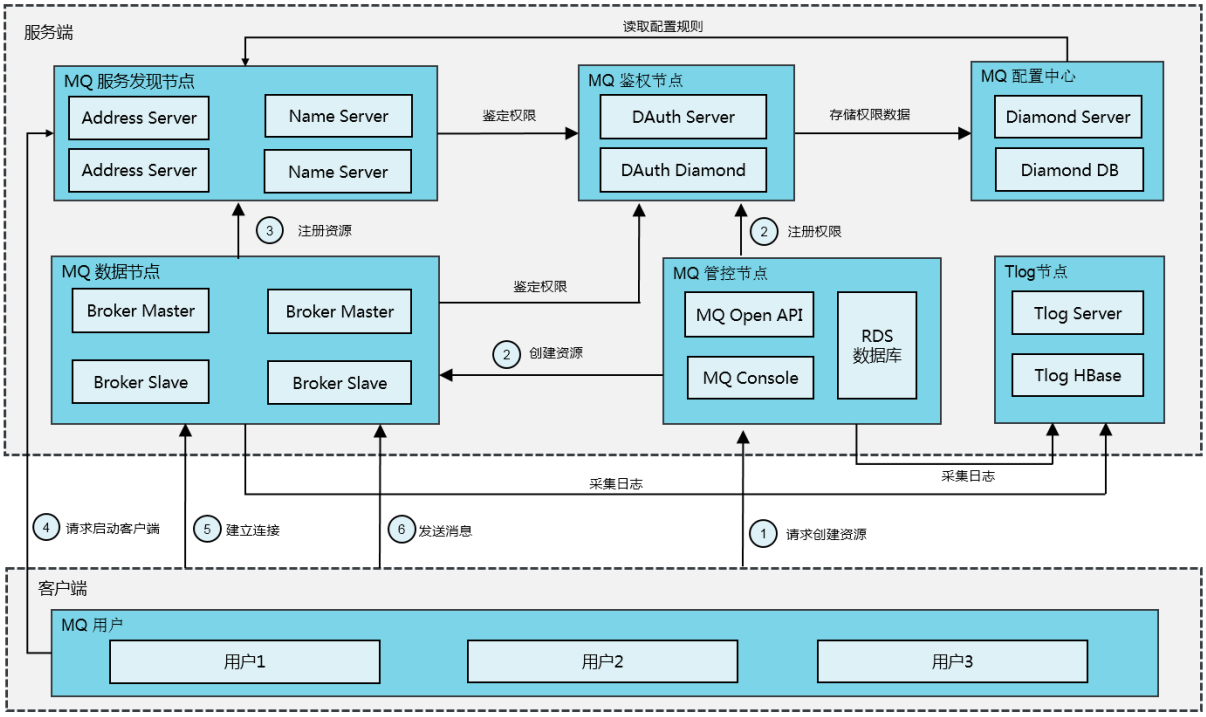
Tlog

Tlog收集消息队列的资源报表数据及其他关键日志。

26.3 数据访问流程

下面以“创建资源>启动客户端>发送消息”的流程场景为例，向您展示MQ客户端和服务端之间的数据访问流程。

图 26-2: MQ数据访问流程图



创建资源>启动客户端>发送消息

1. 您向MQ管控节点发起创建资源（Topic、ProducerID和ConsumerID）的请求。您可以直接通过MQOpen API直接创建资源，也可以通过MQ控制台发起创建资源的请求，控制台随即通过MQOpen API创建资源。
2. MQ Open API将资源创建到Broker，将资源权限注册到DAuth。资源创建属于管控操作，此类管控数据会存储到RDS数据库。
3. MQ Broker将资源信息注册到MQ服务发现节点。
4. MQ客户端向MQ服务发现节点查询提供Topic服务的Broker地址列表，查询到后将Broker信息返回给MQ客户端。
5. MQ客户端和这些Broker建立连接。
6. MQ客户端向Broker发送消息。

说明：

- 步骤4、5、6中均涉及向鉴权节点DAuth申请权限鉴定。
- DAuth处理的鉴权数据存储到MQ配置中心，MQ服务发现节点从MQ配置中心读取配置规则。
- 接收消息的数据访问流程和发送消息的一致。
- Tlog会从Broker和管控节点搜集日志和统计数据。

26.4 高可用部署架构

Broker集群部署

为保证服务的可用性，消息队列支持多节点集群化部署。

- 同一个Region内支持跨机房、跨地域部署，提高网络问题的容灾能力。
- 支持集群化部署方式，提高部分节点不可用时的容灾能力。
- 消息重发机制。当某个节点服务不可用时，迅速切换到其他节点，提高集群的服务能力。

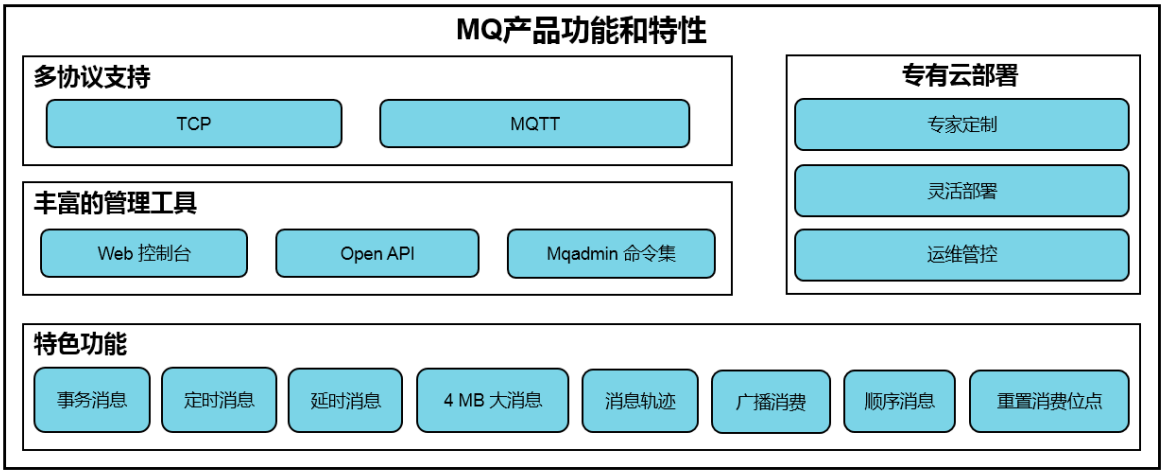
Broker主备部署

- 为保证数据的可靠性，消息队列支持一主多备部署方式。主备复制模式包括同步复制和异步复制。
- MQ Broker同时支持主备间自动切换。一旦主机出现不可用时，根据主备切换策略进行切换，迅速恢复服务。

26.5 功能特性

MQ提供了多种协议和开发语言的接入方式以及多维度的管理工具，同时针对不同的应用场景提供了一系列的特色功能。

图 26-3: MQ功能概览图



26.5.1 多协议支持

MQ提供了多种协议和开发语言的接入方式，包括：

- 支持TCP协议：提供更为专业、可靠、稳定的TCP协议的SDK接入。
- 支持MQTT协议：支持主动推送模型，多级Topic模型支持一次触达1000万+ 终端，可广泛应用于物联网和社交即时通信场景。

26.5.1.1 支持TCP协议

TCP接入主要有以下几大优势：

- 长连接方式提供更高的服务性能；
- 长轮询的实现方式，使得消息收发具有更高的实时性；
- 官方提供JAVA、C++、.NET、PHP四种高可靠SDK接入；
- 支持3种消息发送方式，消息场景全覆盖：可靠同步、可靠异步、oneway方式；
- 支持事务消息、定时/延时消息、顺序消息；
- 支持集群方式与广播方式订阅；
- 更为完整的运维配套支持，包括消息堆积、消息轨迹、消费者运行状态信息等。

消息发送

TCP协议支持三种消息发送方式：可靠同步发送、可靠异步发送、单向 (Oneway) 发送。本文介绍了每种实现的原理、使用场景以及三种实现的异同，同时提供了代码示例以供参考。

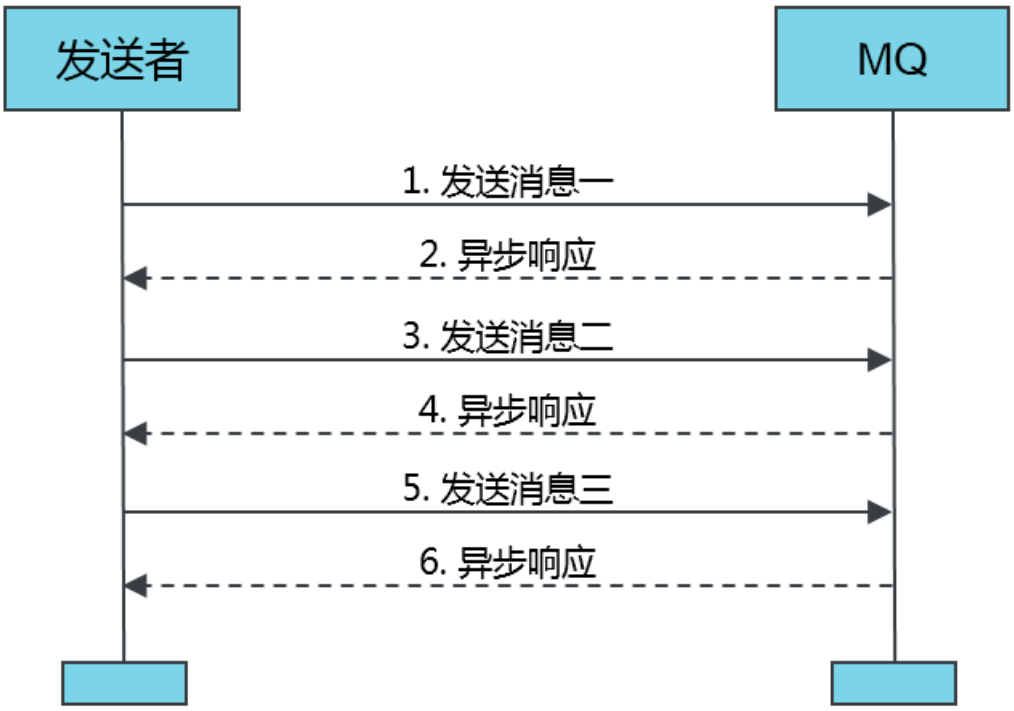
- # **可靠同步发送**

  原理：同步发送是指消息发送方发出数据后，会在收到接收方发回响应之后才发下一个数据包的通讯方式。

  应用场景：此种方式应用场景非常广泛，例如重要通知邮件、报名短信通知、营销短信系统等。

- **可靠异步发送**

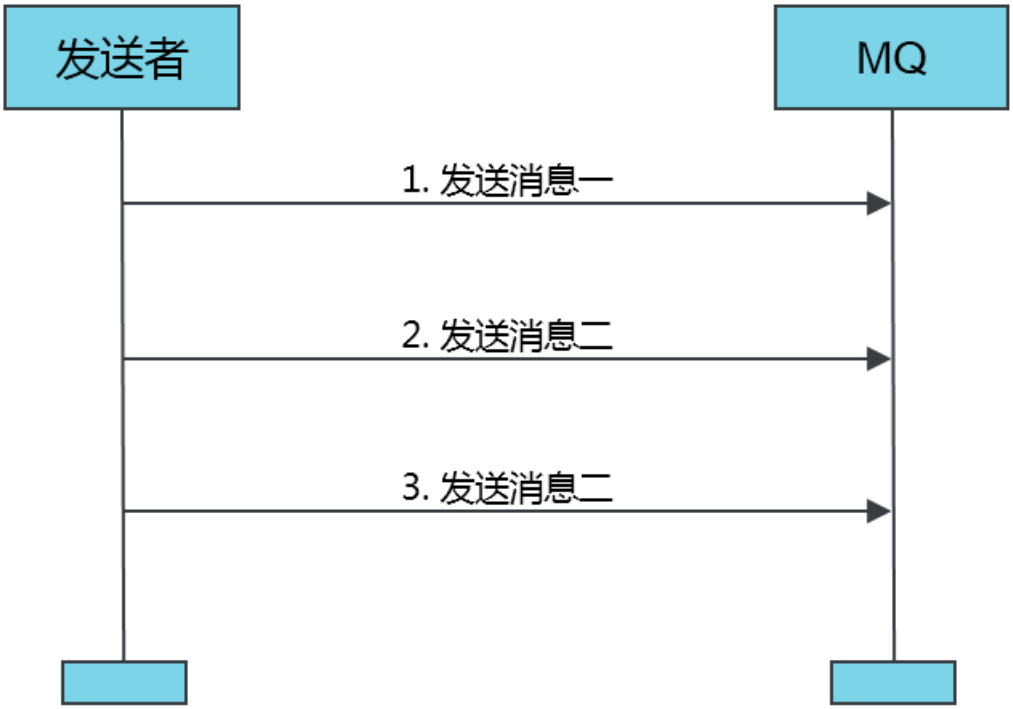
  原理：异步发送是指发送方发出数据后，不等接收方发回响应，接着发送下个数据包的通讯方式。MQ的异步发送，需要用户实现异步发送回调接口（SendCallback），在执行消息的异步发送时，应用不需要等待服务器响应即可直接返回，通过回调接口接收务器响应，并对服务器的响应结果进行处理。



应用场景：异步发送一般用于链路耗时较长，对RT响应时间较为敏感的业务场景，例如用户视频上传后通知启动转码服务，转码完成后通知推送转码结果等。

- **单向（Oneway）发送**

  原理：单向（Oneway）发送特点为只负责发送消息，不等待服务器回应且没有回调函数触发，即只发送请求不等待应答。此方式发送消息的过程耗时非常短，一般在微秒级别。



应用场景：适用于某些耗时非常短，但对可靠性要求并不高的场景，例如日志收集。

下表概括了三者的特点和主要区别。

发送模式	发送TPS	发送结果反馈	可靠性
同步发送	快	有	不丢失
异步发送	快	有	不丢失
单向发送	最快	无	可能丢失

消息订阅

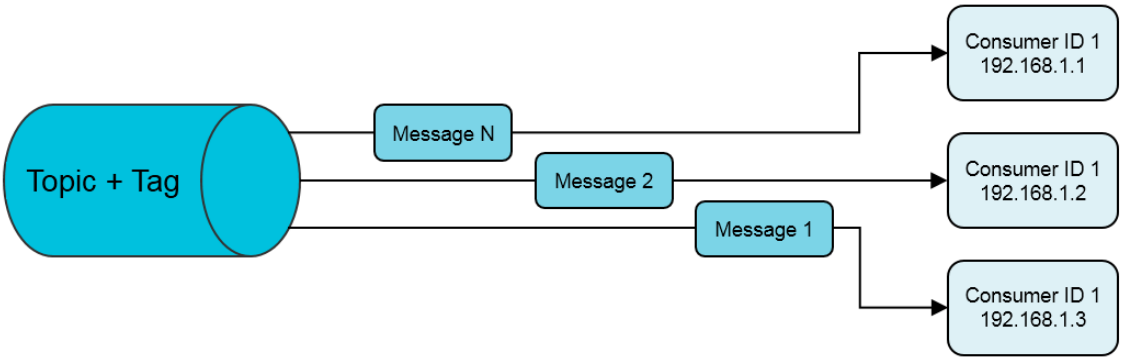
MQ是基于发布订阅模型的消息系统。在MQ消息系统中消息的订阅方订阅关注的Topic，以获取并消费消息。由于订阅方应用一般是分布式系统，以集群方式部署有多台机器。因此MQ约定以下概念。

- 集群：MQ约定使用相同Consumer ID的订阅者属于同一个集群，同一个集群下的订阅者消费逻辑必须完全一致（包括Tag的使用），这些订阅者在逻辑上可以认为是一个消费节点。
- 集群消费：当使用集群消费模式时，MQ认为任意一条消息只需要被集群内的任意一个消费者处理即可。

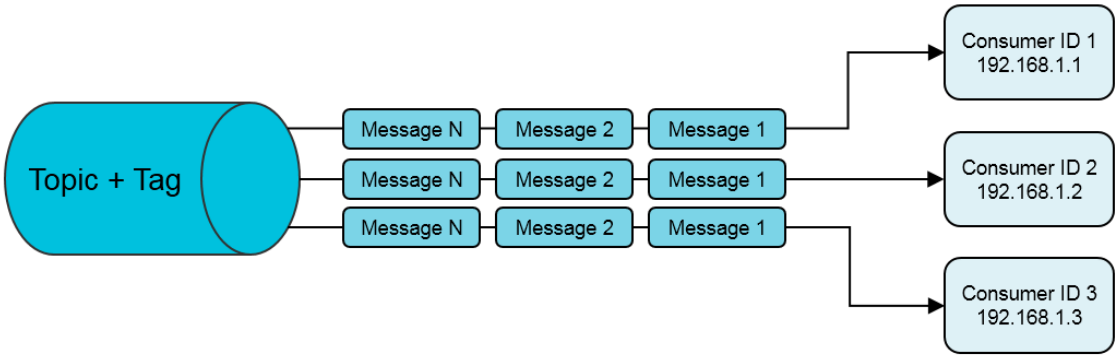
- 广播消费：当使用广播消费模式时，MQ会将每条消息推送给集群内所有注册过的客户端，保证消息至少被每台机器消费一次。

两种消费模式对比如下：

- 集群消费模式



- 广播消费模式



26.5.2 特色功能

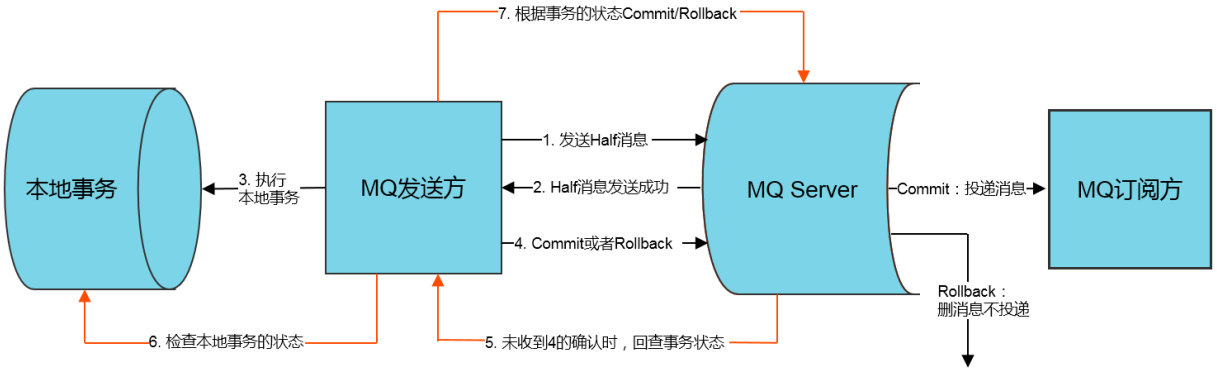
- 事务消息：实现类似X/Open XA的分布事务功能，以达到事务最终一致性状态；
- 定时和延时消息：允许消息生产者指定消息进行定时或延时投递，最长支持40天；
- 顺序消息：支持消息的全局顺序与局部顺序；
- 消息过滤：支持消费者根据Tag在MQ服务端完成消息过滤。
- 消息轨迹：通过消息轨迹，用户能清晰定位消息从发布者发出，经由MQ服务端，投递给消息订阅者的完整链路，方便定位排查问题。
- 广播消费：允许一个Consumer ID所标识的所有Consumer都会各自消费某条消息一次。

26.5.2.1 事务消息

MQ提供类似X/Open XA的分布事务功能，通过MQ事务消息能达到分布式事务的最终一致。

MQ事务消息交互流程如图 *26-4: MQ事务消息交互流程图*所示：

图 26-4: MQ事务消息交互流程图



其中：

- 发送方向MQ服务端发送消息。
- MQ Server将消息持久化成功之后，向发送方ACK确认消息已经发送成功，此时消息为半消息。
- 发送方开始执行本地事务逻辑。
- 发送方根据本地事务执行结果向MQ Server提交二次确认（Commit或是Rollback），MQ Server收到Commit状态则将半消息标记为可投递，订阅方最终将收到该消息；MQ Server收到Rollback状态则删除半消息，订阅方将不会接受该消息。
- 在断网或者是应用重启的特殊情况下，上述步骤4提交的二次确认最终未到达MQ Server，经过固定时间后MQ Server将对该消息发起消息回查。
- 发送方收到消息回查后，需要检查对应消息的本地事务执行的最终结果。
- 发送方根据检查得到的本地事务的最终状态再次提交二次确认，MQ Server仍按照步骤4对半消息进行操作。

26.5.2.2 定时和延时消息

- **定时消息**：Producer将消息发送到MQ服务端，但并不期望这条消息立马投递，而是推迟到在当前时间点之后的某一个时间投递到Consumer进行消费，该消息即定时消息。
- **延时消息**：Producer将消息发送到MQ服务端，但并不期望这条消息立马投递，而是延迟一定时间后才投递到Consumer进行消费，该消息即延时消息。

定时（延时）消息适用于如下一些场景：

- 消息生产和消费有时间窗口要求：比如在电商交易中超时未支付关闭订单的场景，在订单创建时会发送一条MQ延时消息，这条消息将会在30分钟以后投递给消费者，消费者收到此消息后需要判断对应的订单是否已完成支付；如支付未完成，则关闭订单，如已完成支付则忽略。
- 通过消息触发一些定时任务，比如在某一固定时间点向用户发送提醒消息。

定时消息和延时消息的使用在代码编写上存在略微的区别：

- 发送定时消息需要明确指定消息发送时间点之后的某一时间点作为消息投递的时间点。
- 发送延时消息时需要设定一个延时时间长度，消息将从当前发送时间点开始延迟固定时间之后才开始投递。

26.5.2.3 顺序消息

消息队列MQ的支持顺序消息，消息的生产与消费保持有序，其中包括：全局有序和分块有序。

- **全局有序**：所有的消息以消息达到消息队列服务器时的顺序进行消息的消费。
- **分块有序**：根据业务指定的sharding_key进行分块，同一块内消息以消息达到消息队列服务器时的顺序进行消息的消费，不同块之间无顺序关系。

两种有序消息的特点：

- **全局有序**：无论是消息的生产，或是消息的消费，都必须是单实例单线程运行，无法横向扩展，性能相对较弱。
- **分块有序**：根据sharding_key进行分块，块内部单并发运行，块与块之间并发运行，可以横向扩展，性能相对较高。

26.5.2.4 消息过滤

消息队列MQ支持消费者根据Tag在MQ服务端完成消息过滤，从而满足用户对于订阅不同消息类型的需求。

所谓Tag，即消息标签、消息类型，用来区分某个MQ的Topic下的消息分类。MQ允许消费者按照Tag对消息进行过滤，确保消费者最终只消费到他关心的消息类型。

26.5.2.5 消息轨迹

本节介绍MQ消息轨迹的基本原理、使用场景、以及使用案例。

说明：

目前MQ支持TCP协议下的消息轨迹查询。

基本原理

定义：消息轨迹指的是一条消息从生产方发出到消费方消费处理，整个过程中的各个相关节点的时间地点等数据汇聚而成的完整链路信息。

原理：MQ系统中，一条消息的完整链路包含生产方、服务方、消费方三个角色，每个角色处理消息的过程中都会在轨迹链路中增加相关的信息，将这些信息汇聚即可获取任意消息当前的状态，从而为生产环境中的问题排查提供强有力的数据支持。

消息轨迹数据：

表 26-1: 消息轨迹数据

生产方信息	消费方信息	服务方信息
生产客户端信息	消费客户端信息	消息所属的Topic
发送时间	投递时间，投递轮次	消息存储的地域位置
发送成功与否	消费成功与否	消息的Key
发送耗时	消费耗时	消息的Tag

消息轨迹查询规则：

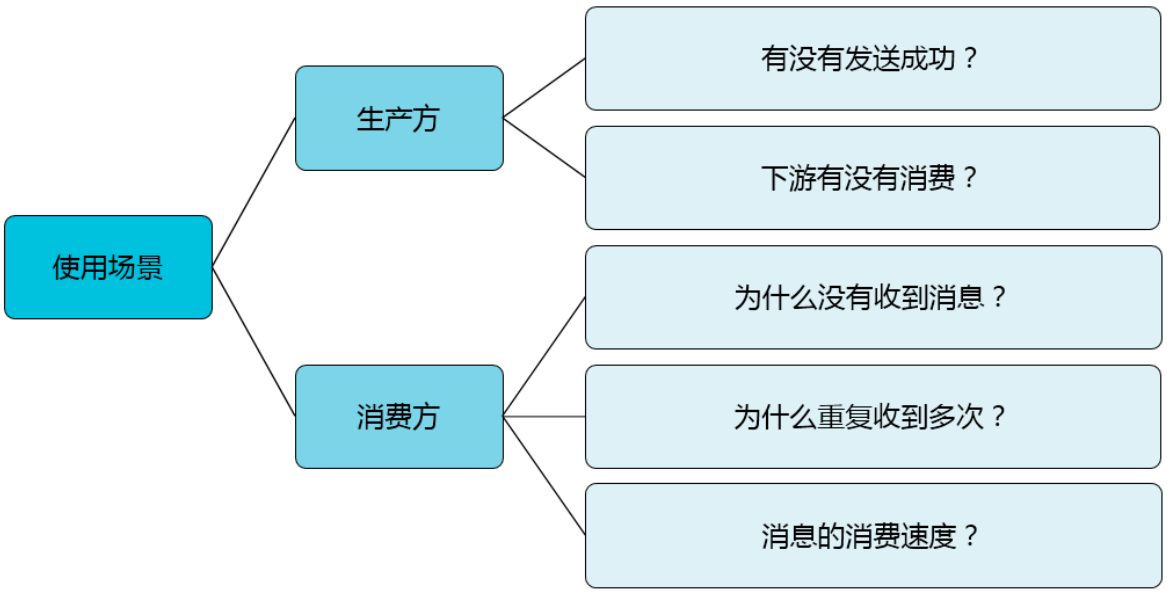
表 26-2: 消息轨迹查询规则

消息类型	可以查询的时间	查询说明
普通消息	消息发送后	消息发送之后有发送轨迹，没消费前显示“尚未消费”。消费后会展示投递和消费信息。
顺序消息	消息发送后	消息发送之后有发送轨迹，没消费前显示“尚未消费”。消费后会展示投递和消费信息。
定时/延时消息	当前系统时间到达消息指定消费的时间后	当前系统时间没有到达指定消费的时间，轨迹可以查询到，但是消息查询不到。
事务消息	消息发送后	消息发送之后有发送轨迹。 事务未提交之前，轨迹可以查询到，但是消息查询不到。

使用场景

在生产环境的消息收发不符合预期时可以使用消息轨迹工具排查问题。通过消息的属性（Message ID、Message Key、Topic）搜索相关的消息轨迹，找到消息的实际收发状态，帮助诊断问题。

图 26-5: 消息轨迹



案例

假设您根据业务日志里的信息判断某条消息一直没有没有收到，请参考以下步骤，利用消息轨迹来排查MQ问题。

1. 收集怀疑的消息的信息，包括Message ID，Message Key，Topic以及大概的发送时间范围。
2. 进入MQ控制台，根据已有的信息新建查询任务，查询相关的消息的轨迹。
3. 查看结果，并分析判断原因。如果轨迹显示尚未消费，则可以去**消费者管理**页面查询，确认是否有堆积导致消息尚未消费。
4. 如果发现已经消费，请根据消费端的信息，找到对应的客户端机器和时间，登录查看相关日志。

26.5.2.6 广播消费

本节主要介绍MQ的广播消费的基本概念，适用场景以及注意事项。

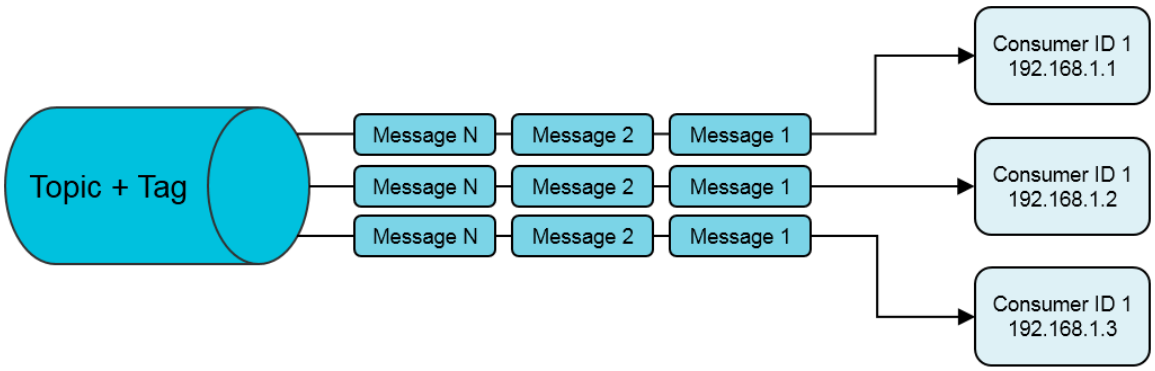
基本概念

MQ是基于发布订阅模型的消息系统。在MQ消息系统中消息的订阅方订阅关注的Topic，以获取并消费消息。由于订阅方应用一般是分布式系统，以集群方式部署有多台机器。因此MQ约定以下概念。

- **集群**：MQ约定使用相同Consumer ID的订阅者属于同一个集群，同一个集群下的订阅者消费逻辑必须完全一致（包括Tag的使用），这些订阅者在逻辑上可以认为是一个消费节点。
- **广播消费**：使用广播消费模式时，MQ会将每条消息推送给集群内所有注册过的客户端，保证消息至少被每台机器消费一次。

应用场景

图 26-6: 广播消费模式



适用场景及注意事项：

- 每条消息都需要被相同逻辑的多台机器处理。
- 消费进度在客户端维护，出现重复的概率稍大于集群模式。
- 广播模式下，MQ 保证每条消息至少被每台客户端消费一次，但是并不会对消费失败的消息进行失败重投，因此业务方需要关注消费失败的情况。
- 广播模式下，第一次启动时默认从最新消息消费，客户端的消费进度是被持久化在客户端本地的隐藏文件中，因此不建议删除该隐藏文件，否则会丢失部分消息。
- 广播模式下，每条消息都会被大量的客户端重复处理，因此推荐尽可能使用集群模式。
- 目前仅 Java 客户端支持广播模式。

- 广播模式下服务端不维护消费进度，所以服务端不提供堆积查询和报警功能。

26.5.3 MQ应用场景

MQ可应用在多个领域，包括异步通信解耦、企业解决方案、金融支付、电信、电子商务、快递物流、广告营销、社交、即时通信、手游、视频、物联网、车联网等。

MQ可以应用但不局限于以下业务场景：

- 一对多，多对多异步解耦，基于发布订阅模型，对分布式应用进行异步解耦，增加应用的水平扩展能力。
- 削峰填谷，大促等流量洪流突然来袭时，MQ可以缓冲突发流量，避免下游订阅系统因突发流量崩溃。
- 日志监控，作为重要日志的监控通信管道，将应用日志监控对系统性能影响降到最低。
- 消息推送，为社交应用和物联网应用提供点对点推送，一对多广播式推送的能力。
- 金融报文，发送金融报文，实现金融准实时的报文传输，可靠安全。
- 电信信令，将电信信令封装成消息，传递到各个控制终端，实现准实时控制和信息传递。

26.6 软件升级

升级采取兼容性灰度升级策略，总体上分为MQ客户端升级、MQ Broker升级和MQ Name Server升级。

- 消息队列定期提供客户端的最新版本，每个版本都有详细说明。客户端升级为非强制性升级，只有您主动要求，才会升级到指定版本。
- MQ Name Server采取灰度发布的策略，采用分批升级的策略，升级过程对用户透明。同时，Name Server升级会保持与客户端以及Broker的兼容。
- MQ Broker升级采取灰度发布的策略，采用分批升级的策略，升级过程对用户透明。同时，Broker升级会保持与客户端的兼容。

27 消息队列MQ（专业版）

27.1 什么是消息队列MQ

本节主要介绍消息队列MQ的定义。

消息队列MQ（Message Queue）是阿里巴巴集团自主研发的专业消息中间件。 产品基于高可用分布式集群技术，提供消息订阅和发布、消息轨迹查询、定时（延时）消息、资源统计、监控报警等一系列消息云服务，是企业级互联网架构的核心产品。 MQ历史超过9年，为分布式应用系统提供异步解耦、削峰填谷的能力，同时具备海量消息堆积、高吞吐、可靠重试等互联网应用所需的特性，是阿里巴巴双11使用的核心产品。

MQ是阿里云正式商用的产品，支持在多个地域（Region）部署高可用消息云服务，单个域内支持多机房部署，可用性极高，即使整个机房都不可用，仍然可以为应用提供消息发布服务，产品稳定性及可用性完全按照阿里巴巴内部标准来实施，无单点。

MQ目前提供TCP和MQTT两种协议层面的接入方式，支持Java、C++以及.NET不同语言，方便不同编程语言开发的应用快速接入MQ消息云服务。 用户可以将应用部署在阿里云ECS、企业自建云，或者嵌入到移动端、物联网设备中与MQ建立连接进行消息收发，同时本地开发者也可以通过公网接入MQ服务进行消息收发。

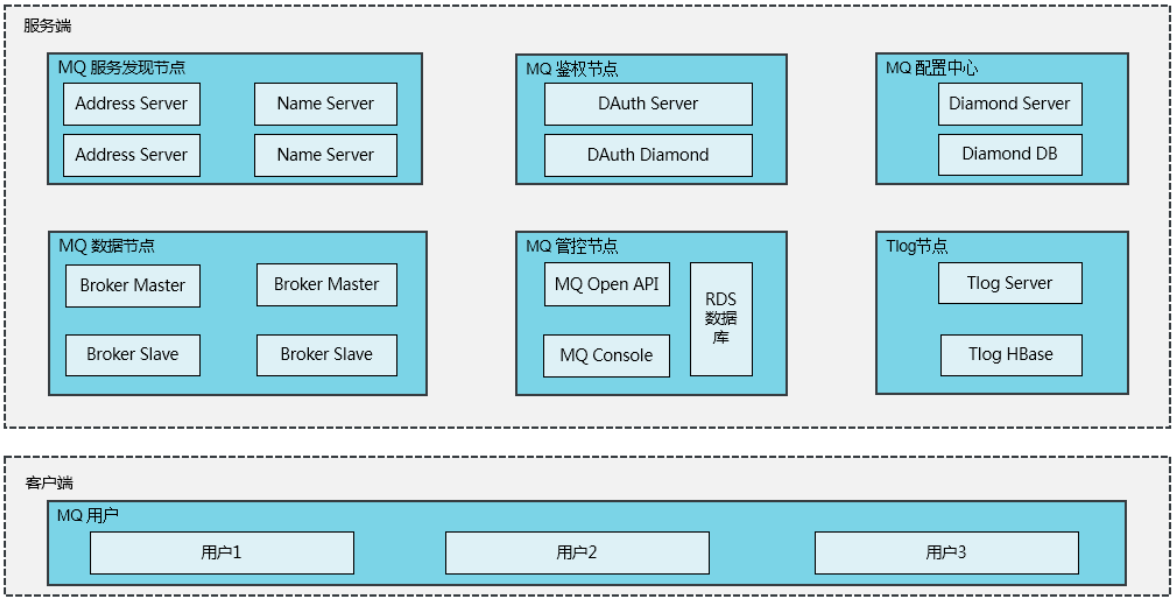
27.2 产品架构

本文向您介绍MQ的系统架构。

系统架构

MQ主要由MQ服务发现节点、MQ Broker、MQ Console、MQ Open API、MQ鉴权节点、MQ配置中心以及Tlog等系统模块组成。

图 27-1: MQ系统架构图



MQ服务发现节点

MQ服务发现节点由Name Server和Address Server组成。Name Server负责消息队列服务的注册与查找，是实现消息队列服务弹性部署和线性扩展的核心。

Name Server几乎无状态节0点，节点之间不需要进行数据同步，可集群部署横向扩展。

Address Server，又称Cai，是消息队列MQ的地址服务器，主要负责Name Server的域名注册和发现。

MQBroker

消息队列服务器（MQ Broker）是消息队列的核心处理模块，由多个服务节点组成服务集群，负责消息的收发以及消息的存储。MQBroker支持集群化、多副本部署，主备复制实现高可用。提供高可用、稳定高效、可线性扩容的消息服务能力。MQ Broker将消息队列将Topic信息、订阅信息等注册到Name Server上，提供服务。

MQ Console

MQ Console（控制台）为用户提供消息的Topic管理、生产者管理、消费者管理、消息查询、消息轨迹查询、资源报表、以及监控告警等一整套完备的运维功能。用户可以通过MQ Console，快速接入消息队列、对用户资源进行管理、问题排查以及通过监控告警等服务帮助用户及时的发现问题。

MQ Open API

为方便用户接入、自主运维，MQ Open API通过HTTP/HTTPS接口的形式提供一整套API服务，用户可以通过Open API创建Topic、生产者信息、消费者信息、消息查询以及消费者状态查询等。

MQ鉴权节点

MQ鉴权节点（DAuth）为消息队列提供统一登录服务以及安全访问控制。包括资源的权限控制、跨账号与子账号访问控制、资源授权等功能。

MQ配置中心

MQ配置中心（Diamond）主要负责存储消息队列的配置信息，如VIP转换规则、资源权限信息等数据。

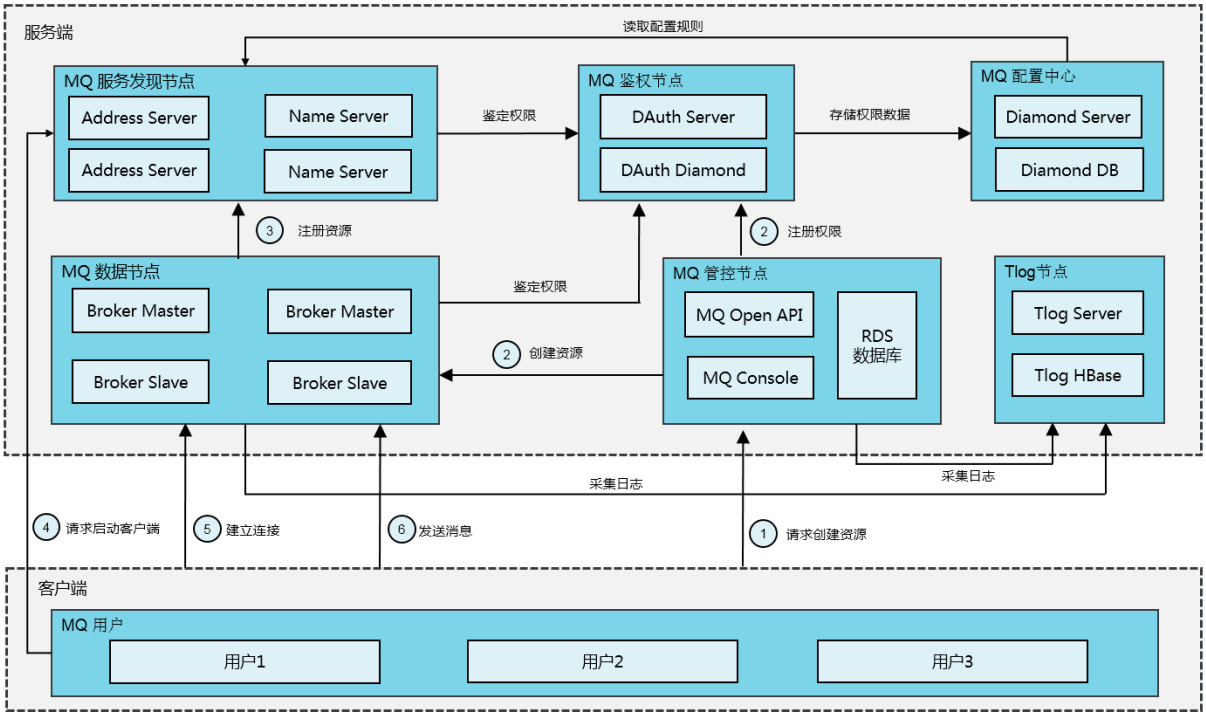
Tlog

Tlog收集消息队列的资源报表数据及其他关键日志。

27.3 数据访问流程

下面以“创建资源>启动客户端>发送消息”的流程场景为例，向您展示MQ客户端和服务端之间的数据访问流程。

图 27-2: MQ数据访问流程图



创建资源>启动客户端>发送消息

1. 您向MQ管控节点发起创建资源（Topic、ProducerID和ConsumerID）的请求。您可以直接通过MQOpen API直接创建资源，也可以通过MQ控制台发起创建资源的请求，控制台随即通过MQOpen API创建资源。
2. MQ Open API将资源创建到Broker，将资源权限注册到DAuth。资源创建属于管控操作，此类管控数据会存储到RDS数据库。
3. MQ Broker将资源信息注册到MQ服务发现节点。
4. MQ客户端向MQ服务发现节点查询提供Topic服务的Broker地址列表，查询到后将Broker信息返回给MQ客户端。
5. MQ客户端和这些Broker建立连接。
6. MQ客户端向Broker发送消息。

说明：

- 步骤4、5、6中均涉及向鉴权节点DAuth申请权限鉴定。
- DAuth处理的鉴权数据存储到MQ配置中心，MQ服务发现节点从MQ配置中心读取配置规则。
- 接收消息的数据访问流程和发送消息的一致。
- Tlog会从Broker和管控节点搜集日志和统计数据。

27.4 高可用部署架构

Broker集群部署

为保证服务的可用性，消息队列支持多节点集群化部署。

- 同一个Region内支持跨机房、跨地域部署，提高网络问题的容灾能力。
- 支持集群化部署方式，提高部分节点不可用时的容灾能力。
- 消息重发机制。当某个节点服务不可用时，迅速切换到其他节点，提高集群的服务能力。

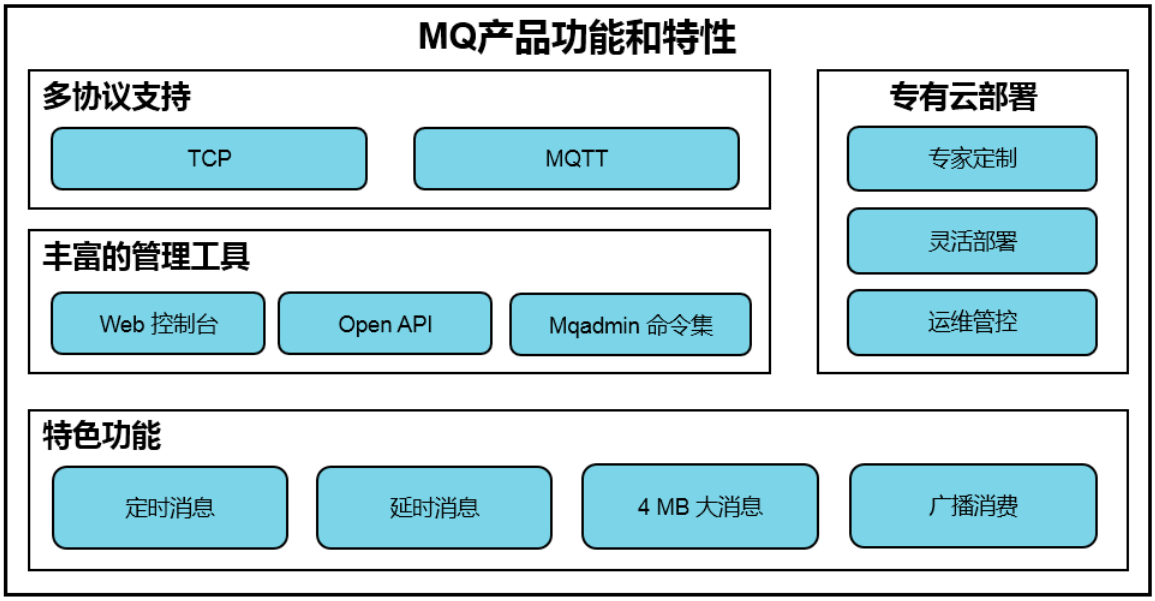
Broker主备部署

- 为保证数据的可靠性，消息队列支持一主多备部署方式。主备复制模式包括同步复制和异步复制。
- MQ Broker同时支持主备间自动切换。一旦主机出现不可用时，根据主备切换策略进行切换，迅速恢复服务。

27.5 功能特性

MQ提供了多种协议和开发语言的接入方式以及多维度的管理工具，同时针对不同的应用场景提供了一系列的特色功能。

图 27-3: MQ功能概览图



27.5.1 多协议支持

MQ提供了多种协议和开发语言的接入方式，包括：

- 支持TCP协议：提供更为专业、可靠、稳定的TCP协议的SDK接入。
- 支持MQTT协议：支持主动推送模型，多级Topic模型支持一次触达1000万+ 终端，可广泛应用于物联网和社交即时通信场景。

27.5.1.1 支持TCP协议

TCP接入主要有以下几大优势：

- 长连接方式提供更高的服务性能；
- 长轮询的实现方式，使得消息收发具有更高的实时性；
- 官方提供JAVA、C++、.NET、PHP四种高可靠SDK接入；
- 支持3种消息发送方式，消息场景全覆盖：可靠同步、可靠异步、oneway方式；
- 支持定时/延时消息；

- 支持集群方式与广播方式订阅；
- 更为完整的运维配套支持，包括消息堆积、消费者运行状态信息等。

消息发送

TCP协议支持三种消息发送方式：可靠同步发送、可靠异步发送、单向 (Oneway) 发送。本文介绍了每种实现的原理、使用场景以及三种实现的异同，同时提供了代码示例以供参考。

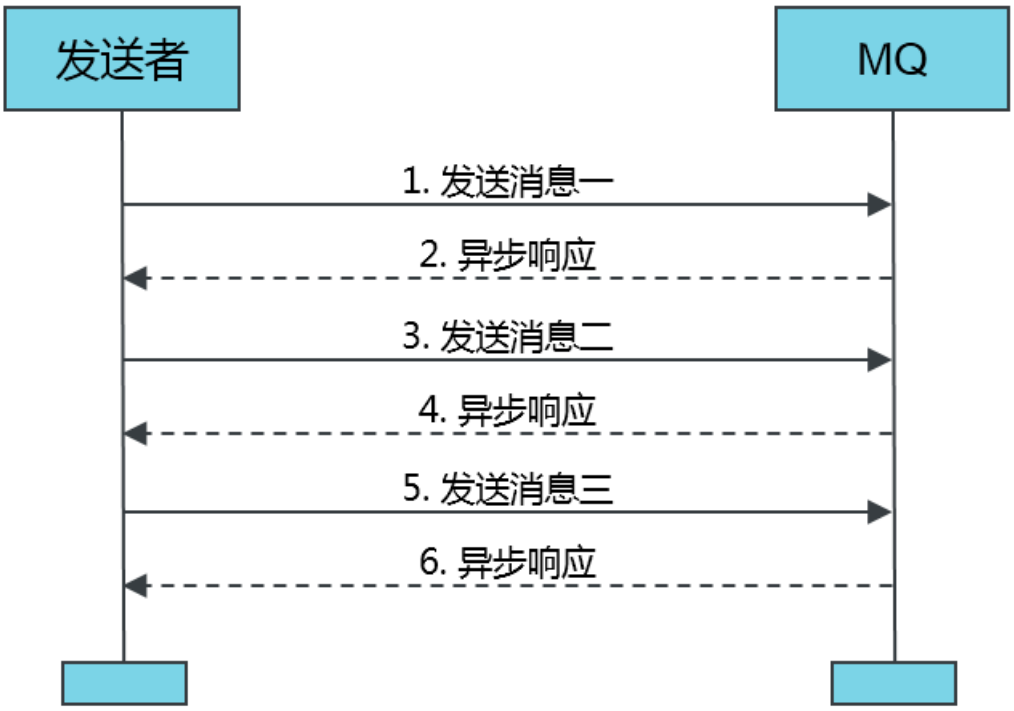
- # **可靠同步发送**

  原理：同步发送是指消息发送方发出数据后，会在收到接收方发回响应之后才发下一个数据包的通讯方式。

  应用场景：此种方式应用场景非常广泛，例如重要通知邮件、报名短信通知、营销短信系统等。

- **可靠异步发送**

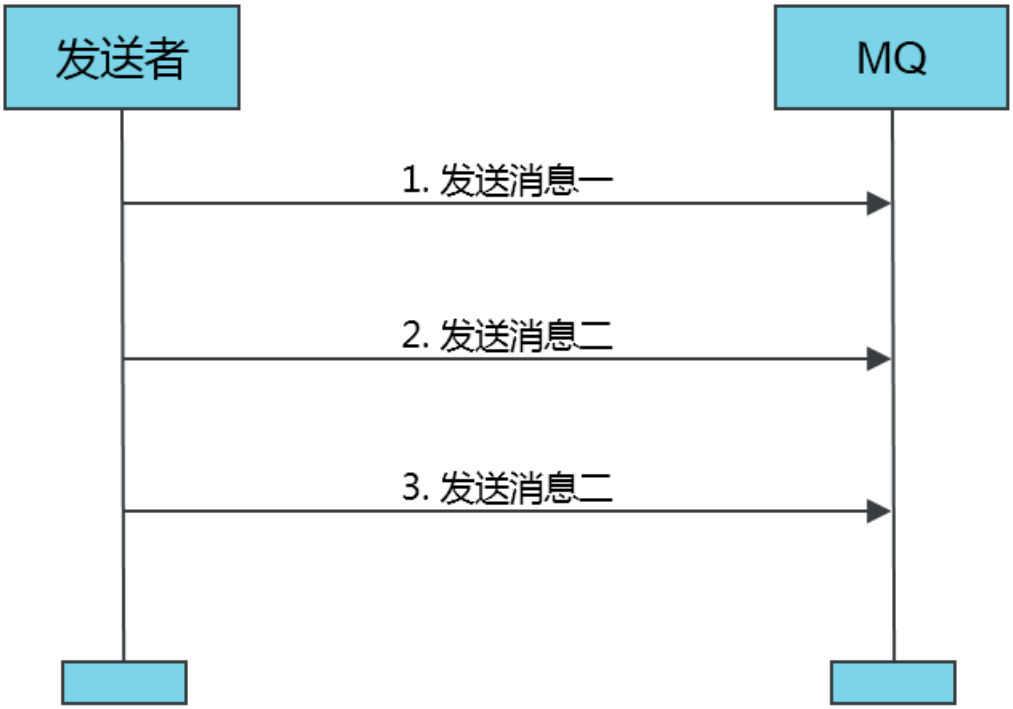
  原理：异步发送是指发送方发出数据后，不等接收方发回响应，接着发送下个数据包的通讯方式。MQ的异步发送，需要用户实现异步发送回调接口（SendCallback），在执行消息的异步发送时，应用不需要等待服务器响应即可直接返回，通过回调接口接收务器响应，并对服务器的响应结果进行处理。



应用场景：异步发送一般用于链路耗时较长，对RT响应时间较为敏感的业务场景，例如用户视频上传后通知启动转码服务，转码完成后通知推送转码结果等。

- **单向（Oneway）发送**

  原理：单向（Oneway）发送特点为只负责发送消息，不等待服务器回应且没有回调函数触发，即只发送请求不等待应答。此方式发送消息的过程耗时非常短，一般在微秒级别。



应用场景：适用于某些耗时非常短，但对可靠性要求并不高的场景，例如日志收集。

下表概括了三者的特点和主要区别。

发送模式	发送TPS	发送结果反馈	可靠性
同步发送	快	有	不丢失
异步发送	快	有	不丢失
单向发送	最快	无	可能丢失

消息订阅

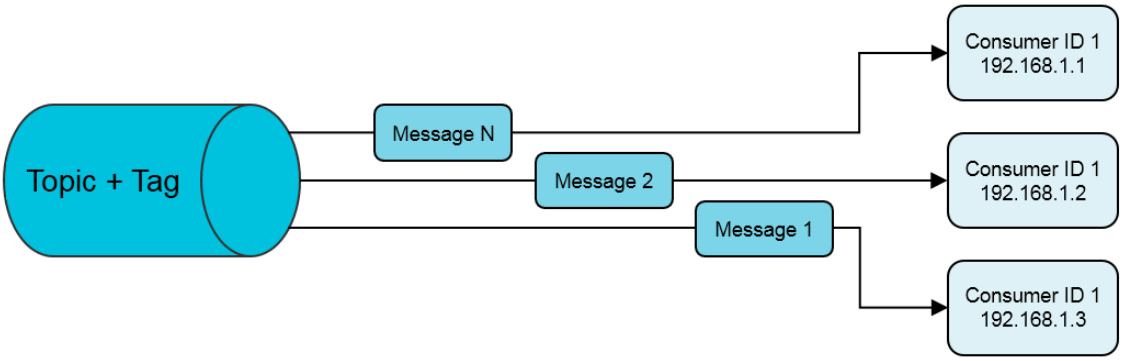
MQ是基于发布订阅模型的消息系统。在MQ消息系统中消息的订阅方订阅关注的Topic，以获取并消费消息。由于订阅方应用一般是分布式系统，以集群方式部署有多台机器。因此MQ约定以下概念。

- 集群：MQ约定使用相同Consumer ID的订阅者属于同一个集群，同一个集群下的订阅者消费逻辑必须完全一致（包括Tag的使用），这些订阅者在逻辑上可以认为是一个消费节点。
- 集群消费：当使用集群消费模式时，MQ认为任意一条消息只需要被集群内的任意一个消费者处理即可。

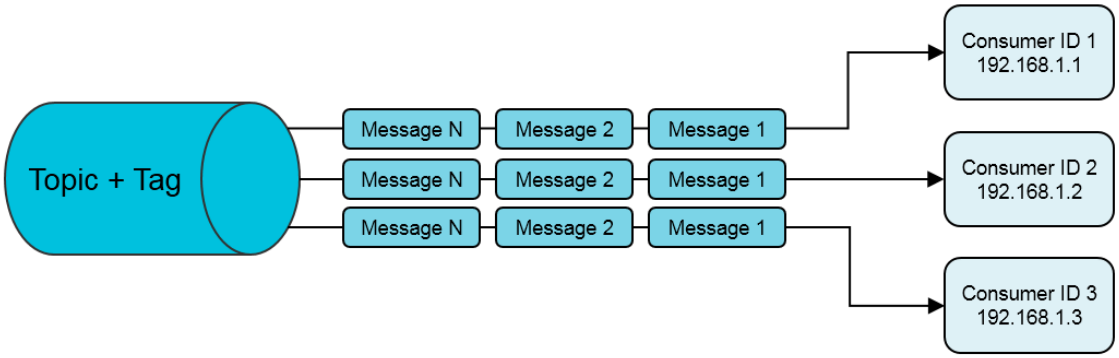
- 广播消费：当使用广播消费模式时，MQ会将每条消息推送给集群内所有注册过的客户端，保证消息至少被每台机器消费一次。

两种消费模式对比如下：

- 集群消费模式



- 广播消费模式



27.5.2 特色功能

- 定时和延时消息：允许消息生产者指定消息进行定时或延时投递，最长支持40天；
- 消息过滤：支持消费者根据Tag在MQ服务端完成消息过滤。
- 消息轨迹：通过消息轨迹，用户能清晰定位消息从发布者发出，经由MQ服务端，投递给消息订阅者的完整链路，方便定位排查问题。
- 广播消费：允许一个Consumer ID所标识的所有Consumer都会各自消费某条消息一次。

27.5.2.1 定时和延时消息

- **定时消息**：Producer将消息发送到MQ服务端，但并不期望这条消息立马投递，而是推迟到在当前时间点之后的某一个时间投递到Consumer进行消费，该消息即定时消息。
- **延时消息**：Producer将消息发送到MQ服务端，但并不期望这条消息立马投递，而是延迟一定时间后才投递到Consumer进行消费，该消息即延时消息。

定时（延时）消息适用于如下一些场景：

- 消息生产和消费有时间窗口要求：比如在电商交易中超时未支付关闭订单的场景，在订单创建时会发送一条MQ延时消息，这条消息将会在30分钟以后投递给消费者，消费者收到此消息后需要判断对应的订单是否已完成支付；如支付未完成，则关闭订单，如已完成支付则忽略。
- 通过消息触发一些定时任务，比如在某一固定时间点向用户发送提醒消息。

定时消息和延时消息的使用在代码编写上存在略微的区别：

- 发送定时消息需要明确指定消息发送时间点之后的某一时间点作为消息投递的时间点。
- 发送延时消息时需要设定一个延时时间长度，消息将从当前发送时间点开始延迟固定时间之后才开始投递。

27.5.2.2 消息过滤

消息队列MQ支持消费者根据Tag在MQ服务端完成消息过滤，从而满足用户对于订阅不同消息类型的需求。

所谓Tag，即消息标签、消息类型，用来区分某个MQ的Topic下的消息分类。MQ允许消费者按照Tag对消息进行过滤，确保消费者最终只消费到他关心的消息类型。

27.5.2.3 消息轨迹

本节介绍MQ消息轨迹的基本原理、使用场景、以及使用案例。

> **说明**：
> 目前MQ支持TCP协议下的消息轨迹查询。

基本原理

定义：消息轨迹指的是一条消息从生产方发出到消费方消费处理，整个过程中的各个相关节点的时间地点等数据汇聚而成的完整链路信息。

原理：MQ系统中，一条消息的完整链路包含生产方、服务方、消费方三个角色，每个角色处理消息的过程中都会在轨迹链路中增加相关的信息，将这些信息汇聚即可获取任意消息当前的状态，从而为生产环境中的问题排查提供强有力的数据支持。

消息轨迹数据：

表 27-1: 消息轨迹数据

生产方信息	消费方信息	服务方信息
生产客户端信息	消费客户端信息	消息所属的Topic
发送时间	投递时间，投递轮次	消息存储的地域位置
发送成功与否	消费成功与否	消息的Key
发送耗时	消费耗时	消息的Tag

消息轨迹查询规则：

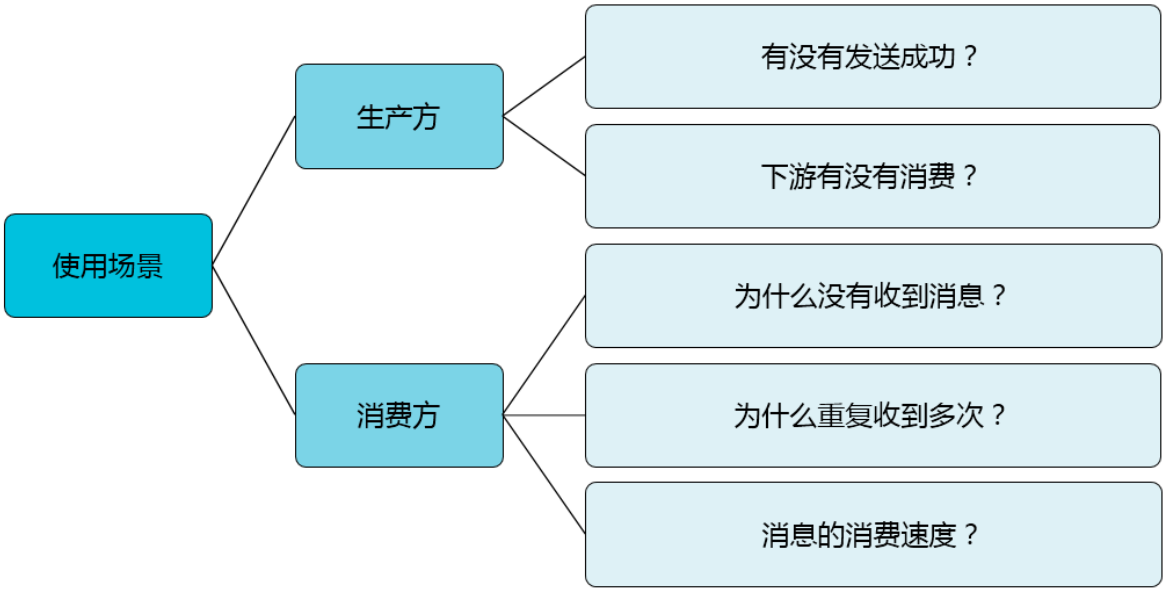
表 27-2: 消息轨迹查询规则

消息类型	可以查询的时间	查询说明
普通消息	消息发送后	消息发送之后有发送轨迹，没消费前显示“尚未消费”。消费后会展示投递和消费信息。
顺序消息	消息发送后	消息发送之后有发送轨迹，没消费前显示“尚未消费”。消费后会展示投递和消费信息。
定时/延时消息	当前系统时间到达消息指定消费的时间后	当前系统时间没有到达指定消费的时间，轨迹可以查询到，但是消息查询不到。
事务消息	消息发送后	消息发送之后有发送轨迹。 事务未提交之前，轨迹可以查询到，但是消息查询不到。

使用场景

在生产环境的消息收发不符合预期时可以使用消息轨迹工具排查问题。通过消息的属性（Message ID、Message Key、Topic）搜索相关的消息轨迹，找到消息的实际收发状态，帮助诊断问题。

图 27-4: 消息轨迹



案例

假设您根据业务日志里的信息判断某条消息一直没有没有收到，请参考以下步骤，利用消息轨迹来排查MQ问题。

1. 收集怀疑的消息的信息，包括Message ID，Message Key，Topic以及大概的发送时间范围。
2. 进入MQ控制台，根据已有的信息新建查询任务，查询相关的消息的轨迹。
3. 查看结果，并分析判断原因。如果轨迹显示尚未消费，则可以去**消费者管理**页面查询，确认是否有堆积导致消息尚未消费。
4. 如果发现已经消费，请根据消费端的信息，找到对应的客户端机器和时间，登录查看相关日志。

27.5.2.4 广播消费

本节主要介绍MQ的广播消费的基本概念，适用场景以及注意事项。

基本概念

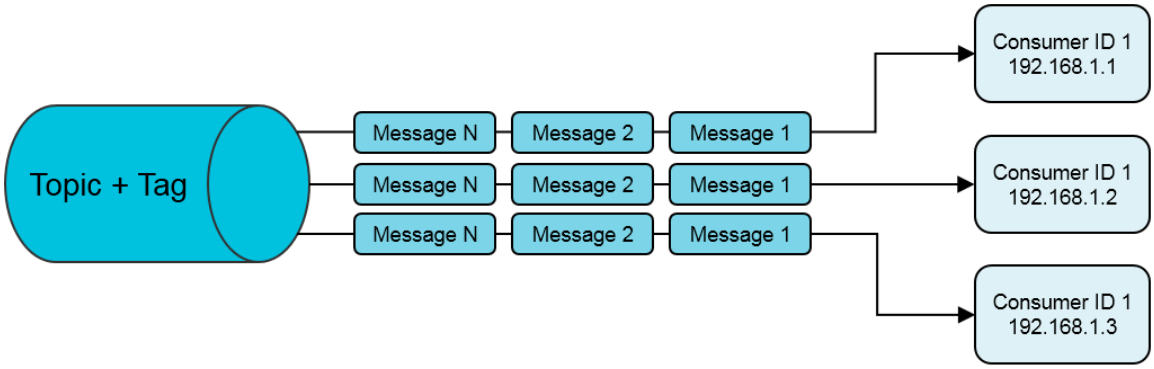
MQ是基于发布订阅模型的消息系统。在MQ消息系统中消息的订阅方订阅关注的Topic，以获取并消费消息。由于订阅方应用一般是分布式系统，以集群方式部署有多台机器。因此MQ约定以下概念。

- **集群**：MQ约定使用相同Consumer ID的订阅者属于同一个集群，同一个集群下的订阅者消费逻辑必须完全一致（包括Tag的使用），这些订阅者在逻辑上可以认为是一个消费节点。

- **广播消费**：使用广播消费模式时，MQ会将每条消息推送给集群内所有注册过的客户端，保证消息至少被每台机器消费一次。

应用场景

图 27-5: 广播消费模式



适用场景及注意事项：

- 每条消息都需要被相同逻辑的多台机器处理。
- 消费进度在客户端维护，出现重复的概率稍大于集群模式。
- 广播模式下，MQ 保证每条消息至少被每台客户端消费一次，但是并不会对消费失败的消息进行失败重投，因此业务方需要关注消费失败的情况。
- 广播模式下，第一次启动时默认从最新消息消费，客户端的消费进度是被持久化在客户端本地的隐藏文件中，因此不建议删除该隐藏文件，否则会丢失部分消息。
- 广播模式下，每条消息都会被大量的客户端重复处理，因此推荐尽可能使用集群模式。
- 目前仅 Java 客户端支持广播模式。
- 广播模式下服务端不维护消费进度，所以服务端不提供堆积查询和报警功能。

27.5.3 MQ应用场景

MQ可应用在多个领域，包括异步通信解耦、企业解决方案、金融支付、电信、电子商务、快递物流、广告营销、社交、即时通信、手游、视频、物联网、车联网等。

MQ可以应用但不局限于以下业务场景：

- 一对多，多对多异步解耦，基于发布订阅模型，对分布式应用进行异步解耦，增加应用的水平扩展能力。

- 削峰填谷，大促等流量洪流突然来袭时，MQ可以缓冲突发流量，避免下游订阅系统因突发流量崩溃。
- 日志监控，作为重要日志的监控通信管道，将应用日志监控对系统性能影响降到最低。
- 消息推送，为社交应用和物联网应用提供点对点推送，一对多广播式推送的能力。
- 金融报文，发送金融报文，实现金融准实时的报文传输，可靠安全。
- 电信信令，将电信信令封装成消息，传递到各个控制终端，实现准实时控制和信息传递。

27.6 软件升级

升级采取兼容性灰度升级策略，总体上分为MQ客户端升级、MQ Broker升级和MQ Name Server升级。

- 消息队列定期提供客户端的最新版本，每个版本都有详细说明。客户端升级为非强制性升级，只有您主动要求，才会升级到指定版本。
- MQ Name Server采取灰度发布的策略，采用分批升级的策略，升级过程对用户透明。同时，Name Server升级会保持与客户端以及Broker的兼容。
- MQ Broker升级采取灰度发布的策略，采用分批升级的策略，升级过程对用户透明。同时，Broker升级会保持与客户端的兼容。

28 业务实时监控服务ARMS

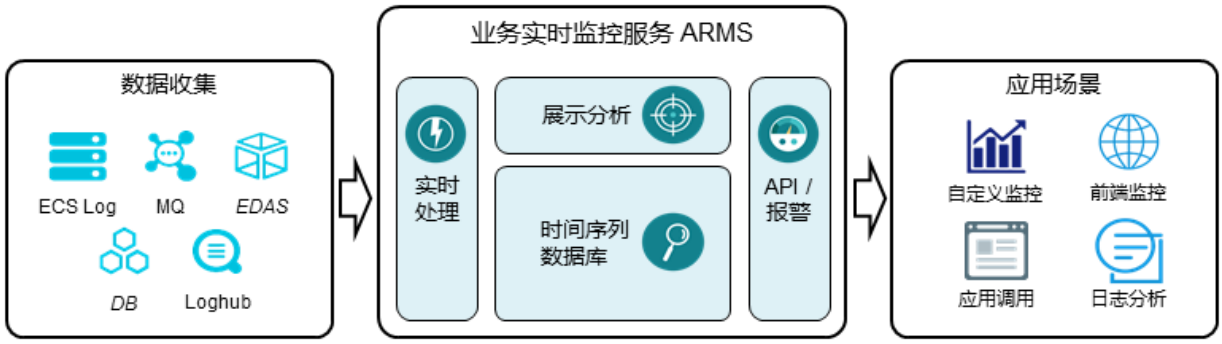
28.1 什么是业务实时监控 ARMS

业务实时监控服务 ARMS（Application Real-Time Monitoring Service）是一款阿里云应用性能管理（APM）类监控产品。借助本产品，您可以基于前端、应用、业务自定义等维度，迅速便捷地为企业构建秒级响应的业务监控能力。

工作流程

ARMS 工作流程如下图所示：

图 28-1: ARMS 工作流程



- 数据收集：ARMS 支持通过配置从 ECS Log、MQ 和 Loghub 上抓取日志。
- 任务定义：
  - # 通过任务配置来定义实时处理、数据存储、展示分析、数据 API 和报警等任务，从而定义出自己的应用场景。
  - # 通过前端监控、应用监控等预设场景直接进行业务监控。
- 应用场景：如上所述，除了自定义监控以外，ARMS 还有可直接使用的预设监控场景，包括前端监控、应用监控等。

适用场景

- **业务深度定制监控**：可按需深度定制具备业务属性的实时监控报警和大盘。业务场景包括电商场景、物流场景、航旅场景等。
- **前端体验监控**：按地域、渠道、链接等维度实时反映用户页面浏览的性能和错误情况。
- **应用性能和异常监控**：对分布式应用进行性能异常监控和调用链查询的应用性能管理（APM）能力。

- **统一报警和报表平台**：集自定义监控、前端监控和应用监控为一体的统一报警和报表平台。

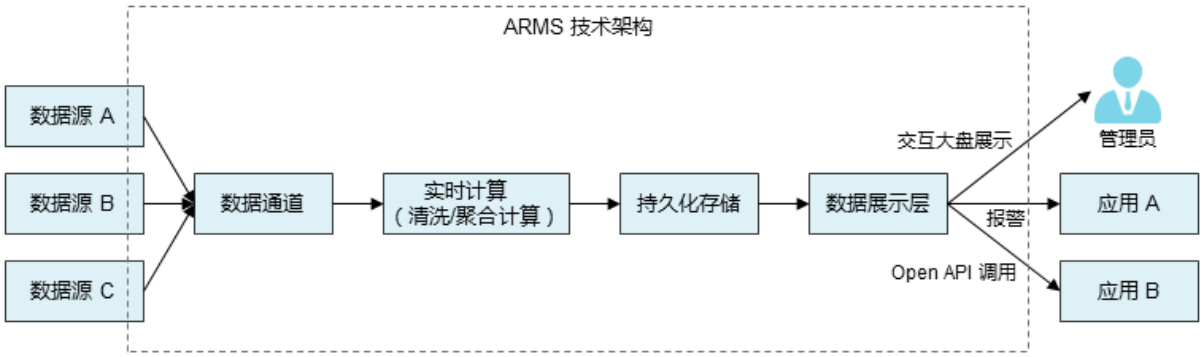
图 28-2: ARMS 适用场景示意图

凭借 ARMS，IT 人员能够在数分钟内搭建和启动基于大数据平台的业务实时监控系统，在充分发挥数据监控时效性的同时提升 IT 人员效率。

28.2 产品架构

本文介绍了 ARMS 的产品架构。

图 28-3: ARMS 技术架构



在一个完整的监控任务中，数据流依次经过以下环节：

1. 从数据源流入数据通道进行统一管理和缓存。
2. 从数据通道流入实时计算引擎进行实时计算。
3. 实时计算的结果流入持久化存储进行统一管理。
4. 通过数据展示层对数据进行各类导出，包括 Open API 直接读取、报表展示、报警通知等。

28.2.1 数据源

在ARMS中，数据源负责为ARMS提供数据输入。数据源包括以下几种方式：

- StarAgent： 通过StarAgent完成在服务器上的日志的增量拉取，例如日志文件。适用所有可以网络直连互通的网络环境。
- MQ ：通过配置MQ相应Topic的接收端，将指定的数据以消息方式传输到ARMS计算节点。

ARMS在收集数据时，需要在已定义的数据源中通过建立数据采集规则来实现数据采集。以采集服务器上的日志为例，采集规则包括数据源的选择、数据的文件路径等。

28.2.2 数据通道

从数据源中流出的数据首先进入数据通道。数据通道在ARMS中相当于一个数据队列。它主要起到以下作用：

保证从数据源中流出的数据能立即被ARMS接收到，为下游的实时计算层充当缓冲层。

当计算节点出现任何异常时，相应时间点数据能统一从日志通道重新发送给计算层，以保证所有数据至少被实时计算层处理过一次。

日志通道在ARMS上对于用户是透明不可见的。您只需控制数据源和实时计算层的计算逻辑，日志通道的各类管理由ARMS自动完成。

28.2.3 实时计算

在数据通道中的数据会被ARMS计算层节点实时读取。ARMS实时计算层的实时计算能力由基于阿里巴巴内部开发且开源的实时计算引擎JStorm提供，实现毫秒级的流式计算处理能力。

ARMS 的实时计算层并不要求您基于实时计算引擎编写流式计算程序，而是通过拖拽方式方便地定义出实时计算任务。在 ARMS 中，您只需要基于交互界面做以下事情：

- 定义数据的清洗逻辑：例如一行日志数据是以“|”或是“；”符号进行切分(清洗的一种逻辑)，切分后的 Key-Value 设定等。
- 定义数据集的聚合计算逻辑：例如数据以何种维度进行聚合（Group By），聚合的计算方式（Sum，Max）等。

在监控任务中，通过定义清洗和聚合计算逻辑来产出计算结果，最终的计算结果会形成一个个特定的数据集，进行持久化的存储。

28.2.4 持久化存储

作为一款致力于提供端到端实时监控解决方案的产品，ARMS除了提供实时计算能力以外，同时提供了将计算结果持久化以供下游使用的能力。

在 ARMS 中，持久化存储通过定制时间序列数据库实现，以达到高吞吐和高扩展的性能要求。在实时计算中产生的数据集以ARMS特定的数据结构进行存储。存储具体结构对外部透明，不需要用户干预。您只需要在实时计算中通过控制产生的数据集和相关属性（索引键、保存周期等）来控制哪些数据需要进行持久化存储，并管理对应的存储空间。

ARMS 持久化到后端时间序列数据库的数据结构对外部透明。您可以通过访问数据展示层来间接访问时间序列数据库的数据结果。

28.2.5 数据展示层

ARMS的监控数据结果一般可通过以下三种方式进行利用。

- RESTful API：最直接的方式。通过ARMS提供的RESTful API，基于数据集定义的各类查询Key对数据结果进行访问。
- 交互式大盘：ARMS中用户基于数据集自定义的一组交互式报表。一个交互式大盘可使用不同图表形式展示多个数据集。查询时间跨度可自定义。

- 报警通知：通过定义报警策略，定期对数据集结果进行抓取和匹配，对符合特征的事件以电子邮件、短信等方式报警。

28.3 可靠性方案

本章说明了 ARMS 的可靠性方案。

28.3.1 QoS保障策略

ARMS提供数据计算、存储写入、API读取等多方面端到端的服务质量（Quality of Service，简称QoS）控制。

数据计算策略

- Topology配置限制策略

  ARMS针对每个Topology上的数据流入模块（Storm Spout）设置了数据计算的最大内存值和最大CPU核数。这样可以有效控制数据量突发造成的影响，避免影响整个ARMS计算集群。

  如果您需要应对可预估的数据量暴涨情况，则建议在突发情况下增加Topology配 置，增大相应的任务并发数以及内存数，以解决计算规模问题。

- Topology公共任务保护策略

  对于公共任务，ARMS把多个任务放置在一个Topology进行计算。为了防止一个任务拖垮所有Topology，ARMS设置了流量和维度限制上限，防止一个任务过量计算而拖垮整个集群。

存储写入策略

ARMS对用户的存储写入做出了限制，限制力度与Topology的数量和大小成正比。

这样可以防止当数据量暴涨时，某个用户将整个存储写挂。列式存储的写入限制不会造成用户的数据丢失。当写入实际速度小于数据实际的产生速度时，计算任务会相应地被阻塞，也就是实时监控数据会有相应的延时产生。

如果您需要解决可预见的列式存储写入瓶颈，只需要增加Topology的数量和大小，以提高存储的写入QoS限制。

API读取策略

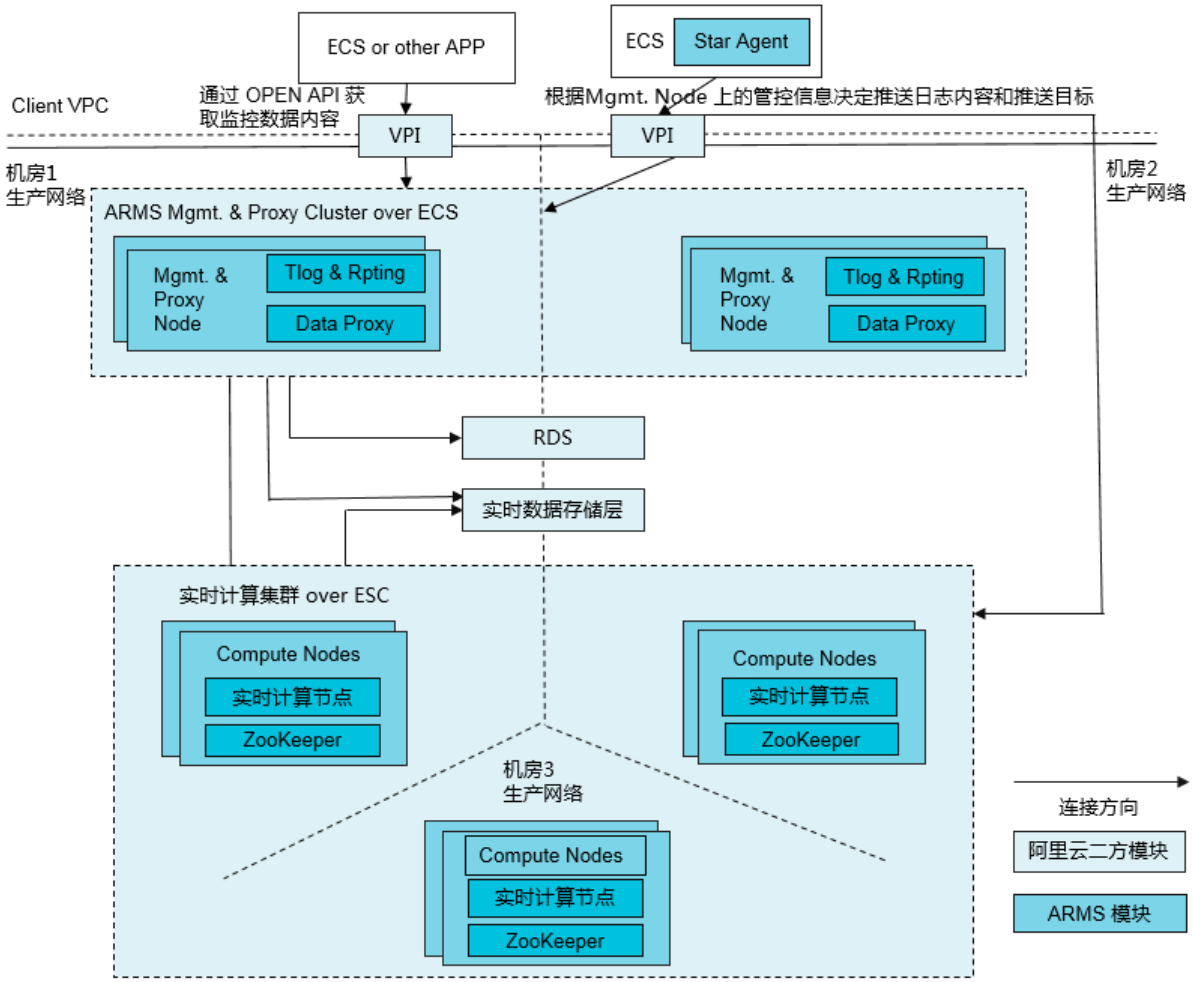
ARMS上的存储数据结果可以通过API被用户或者其他应用读取。为了防止应用将ARMS的API代理节点（ARMS Console）或存储读挂，ARMS 对 API 读取速度做了限制。默认每个用户的API读取速度限制是100。

如果有大量的API读取峰值需求，可通过调整相应QoS策略来解决。

28.3.2 高可用和容灾保障

ARMS的系统架构达到总体无单点，支持双站点双活式容灾。

图 28-4: ARMS系统架构图



站点内高可用

ARMS系统架构无论在数据输入层、计算层、存储层，还是API调用层，都采用高可用架构，且理论上支持无限横向扩张。

- 数据输入层在专有云上可对接简单日志服务（SLS）及消息队列服务（MQ），在可用性和性能上均无单点。
- 计算层采用JStorm集群。当计算节点宕机，JStorm主控节点（Nimbus）将根据Zookeeper信息自动在健康的Supervisor上重新启动计算任务。
- 存储层采用列式存储、两份备份。即使两台节点同时宕机，数据仍可用。

- API调用代理层均为无状态节点，通过DNS/VIP将负载均衡分散到各代理节点，整个代理层可横向扩展。

站点间容灾策略

ARMS机房发生整体宕机时，理论上有分钟级的恢复时间目标（Recovery Time Objective，RTO）和接近于零的恢复点目标（Recovery Point Objective，RPO）。

- 数据接入层、计算层、数据代理层需要故障切换到容灾站点时，理论上会存在些许RTO。
- 列式存储通过多活部署，集群直接跨站点。灾难时，理论RTO为零。数据通过集群内数据复制功能，理论上能达到接近于零的RPO 。

28.3.3 软件升级

- 升级采取兼容性灰度升级策略。
- 升级总体上就是ARMS Console升级。列式存储、JStorm集群以及数据接入层几乎无需或很少需要升级。
- ARMS Console升级采取灰度发布的策略，分批进行升级。

28.4 功能解析

ARMS 的重要功能涉及的概念包括：

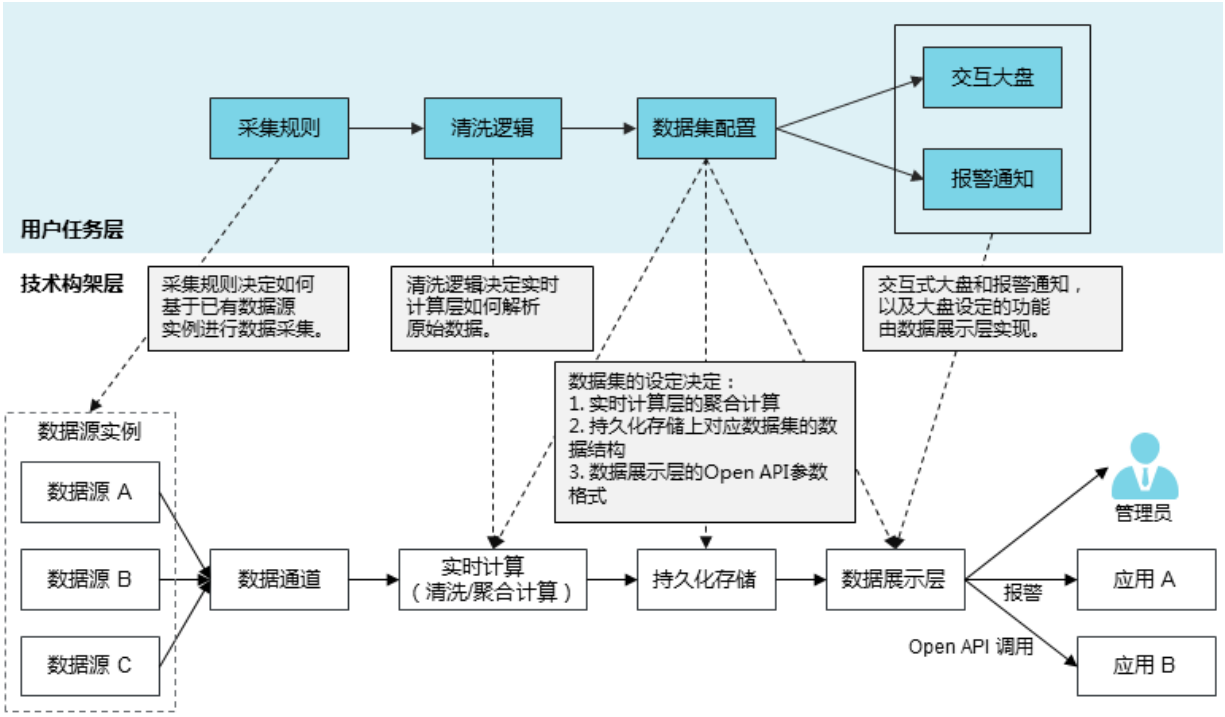
- 数据源：ARMS 获取数据的地方，目前主要支持 StarAgent 数据源采集增量数据、MQ 数据源采集等。
- 监控任务：一个任务代表 ARMS 从数据抓取、数据处理、数据存储到结果展示和导出的一个实例。
- 数据集：代表一个监控业务的数据结果，其结果可以用 Open API 导出。数据集直接被报表控件和报警规则依赖。
- 交互大盘：基于数据集自定义的一组交互式报表。
- 报警规则：定义如何从现有数据集中生成报警。

28.4.1 关键流程

在 ARMS 中，最关键的流程是定义监控任务。通过定义一个监控任务，可以利用数据源产生一系列的监控结果，包括生成数据集、报表控件和报警规则。

如图 *28-5: ARMS创建任务流程图*所示，创建一个任务所需要的大致流程和组件，以及组件在 ARMS 技术架构中的依赖方式如下。

图 28-5: ARMS创建任务流程图



下文对任务组件作简要说明。

- 采集规则(必选)：定义数据如何从不同的数据源实例进行采集。
- 清洗逻辑(必选)：定义如何解析采集到的数据。
- 数据集配置(必选)：通过配置数据集来定义任务如何基于采集到的数据做聚合计算，持久化存储，以及 RESTful API 访问输出。
- 报警通知(可选)：基于数据集，通过定义报警含义和通知方式，提供报警能力。
- 交互大盘(可选)：通过集成数据集和报警通知数据提供可视化大盘能力。

28.4.2 监控任务

在ARMS中，监控任务主要分为两大类。

- **预定义任务**：如异常堆栈监控，商品销售量统计等。通过创建这类任务，您可以直接使用预定义的清洗逻辑、数据集、报表控件的组件，快速组装出一个针对特定场景的监控任务。
- **定制任务**：根据提示步骤，一步步手动定制任务的各类组件，组装出一个完整监控任务。

创建了监控任务后可在相应的任务管理界面进行管理。除了查看、删除以外，还可以针对监控任务进行起停操作。任务只有被启动的时候ARMS才会进行数据采集、计算和存储数据。当任务被停止的时候，以上工作也会被停止。

28.4.3 采集规则定义

采集规则定义了数据如何从数据源实例中进行采集。您可以基于已定义的数据源建立采集规则。

- StarAgent数据源：直接选取填写目标IP列表即可。
- MQ数据源：直接填写您的Topic和Topic所在的Region即可。

28.4.4 数据清洗定义

每个监控任务对应一个清洗逻辑。清洗逻辑定义如何解析采集到的数据。对于文本类数据，ARMS支持多种数据清洗方式。例如：

- 通过特定分隔符如“|” “，” “=” 对数据进行清洗，从而清洗出不同的 Key-Value (KV) 。一个极简的例子包括：itemID=abc|amount=100 的数据日志会被清洗成 itemID 为 String abc，amount 为 int 100，一共两组 Key-Value 。
- 支持基于JSON格式的数据，通过解析 Hash 数据结构清洗出不同的 KV 。
- 支持用户自定义的清洗逻辑，如基于不同清洗符的清洗嵌套等。

28.4.5 数据集配置管理

数据集是ARMS中数据计算和数据持久化的重要概念。一个监控任务可对应一个或多个数据集。

创建数据集

在定义了清洗逻辑以后，通过以下方法定义数据集：

- 直接创建：创建一个数据集，并定义其维度（RESTful API 查询 Key），统计值（Value）。
- 间接创建：通过创建一个报表控件并定义控件要展示的值和维度，或通过创建一个报警通知并定义要监控的值和维度，来间接创建一个数据集。

数据集与实时计算逻辑和数据导出格式

无论使用直接创建还是间接创建，当创建了一个数据集以后，定义的维度（Key）和统计值（Value）将直接决定数据在ARMS中如何进行实时计算，以及其RESTful API的查询参数组合和返回值方式。

一个极简单的例子，例如某电商想统计各类商品的各个时刻的实时销售额，用于实时展示和事后统计。其设计的统计维度和统计值为：

- 查询维度为时间（TimeStamp）和商品类目ID（String）。
- 统计值为销售额（Sum(Int)）。

那么：

- 首先，在实时计算中，ARMS在后面的JStorm引擎中会对大量的输入数据基于时间和商品类目ID做类似于Reduce的计算，在计算中对销售额做Sum操作。
- 计算后的结果，根据聚合粒度实时在存储层中持久化。其对应的查询Open API中的必选查询Key为时间和销售类目ID，返回的Int值表示指定时刻和商品类目的销售总额。

数据集的聚合粒度和保存周期

在进行数据集配置时，您可以定义聚合的粒度，例如1分钟聚合一次还是1小时聚合一次，以及响应的数据的保存周期。一个数据集的聚合粒度和保存周期设置将直接影响其在ARMS的持久化存储层的存储容量。

对于大多数场景，您可能想定义以下聚合和保存的方式组合。这样既可以保证最近时刻的数据精确性，又可以满足长期的统计工作需求，而且还最大限度利用了空间。

- 1分钟的数据聚合频率，保存7天。
- 1小时的数据聚合频率，保存30天。
- 1天的数据聚合频率，保存3年。

配置了数据集以后，可以通过数据集管理界面对数据集进行管理，包括启动/停止操作。

- 数据集启动操作将保证在对应任务启动时，对应的计算将被执行且结果持久化到存储层。
- 数据集停止时，即便其对应的任务在启动状态，数据集对应的计算也不会被执行。数据集停止期间未被处理的数据流亦无法被回溯执行。

28.4.6 报警规则管理

您可以直接创建一个报警规则，或者基于一个现有的数据集创建一个报警规则。

报警规则是对一个现有数据集的处理定义，包括：

- 需要判断的指标阈值。
- 超过阈值后的处理规则。

28.5 技术规格

表 28-1: 功能列表

指标项	规格要求
基本功能	接入端支持 MQ 、文件日志。

指标项	规格要求
	基于任意日志或数据格式的清洗（切分）功能，能切分的格式包括 KV、Json、Exception 和其他各类特定日志，如 Nginx、Apache HTTP 和其他用户自定义日志。
	支持的数据清洗结果监控计算包括 Sum、Count、Max、Min、抽样、TopN、Count Distinct，以及“+”、“-”、“*”、“/”、同比、环比等基本计算。监控结果支持多维度查询。
	监控结果（数据集）支持 API 查询，支持报表展示，支持报警检测。
	支持各类在线图表格式，包括饼图、柱状图、翻牌器、折线图、面积图等。支持交互式图表，时间和关键输出指标可交互输入。在线图表可自动刷新。
	数据集支持上钻和下钻操作。
	计算和存储支持在线扩容。
	报警支持短信、邮件和其他定制命令接口格式；报警级别定制；报警内容可基于数据集结果任意定制。
	报警可过滤，防止报警风暴。
	支持数据的生命周期管理，可区分时间粒度，保存周期可定制。例如按分钟聚合的结果保存 7 天，小时粒度保存 1 个月，天粒度保存 1 年。
开放性	运维管理平台可以支持两种类型的 API： • 数据 API，支持对数据结果的查询。 • 管理 API，支持对系统管理和运维的自动化集成。
安全隔离性	有租户概念，不同租户之间数据和任务不可见。
	不同租户之间有性能 QoS，单个租户的并发不影响其他租户的性能。
	不同租户之间的数据生命周期管理互不影响。
技术成熟	在阿里巴巴内部经过 5 年实践考验，包括基础架构监控、电商监控、物流监控等场景。

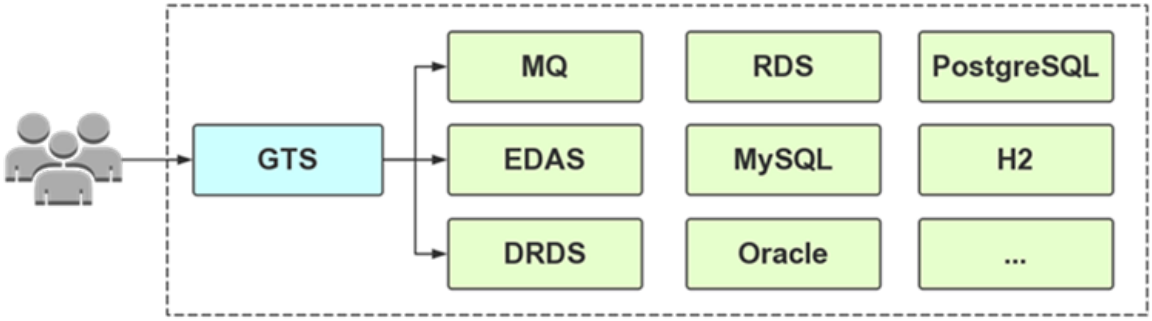
29 全局事务服务GTS

29.1 产品概述

全局事务服务GTS（Global Transaction Service）是一款高性能、高可靠、接入简单的分布式事务中间件，用于解决分布式环境下的事务一致性问题。

传统的事务主要是指单机数据库的ACID特性，GTS在支持分布式数据库事务的基础上，将事务的范围拓展到了多种资源，让分布式环境下的多个资源的操作加入事务的范畴，赋予了分布式资源操作ACID特性。

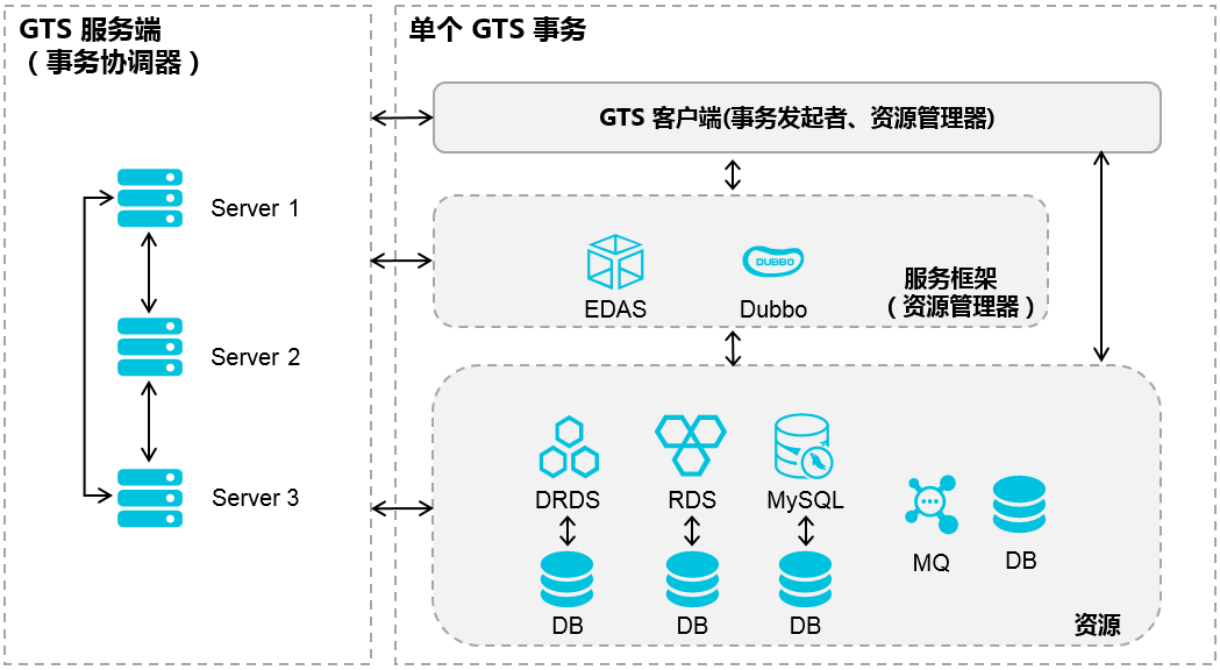
该产品支持DRDS、RDS、Oracle、MySQL、PostgreSQL、OceanBase、Petadata等多种数据源，并可以配合使用EDAS、Dubbo、Spring Cloud及多种私有RPC框架，同时还兼容 MQ 等中间件产品，能够轻松实现分布式数据库事务、多库事务、消息事务、服务链路级事务等多种业务需求。



29.2 产品架构

全局事务服务GTS主要由GTS服务端、GTS客户端和GTS资源端三大部分组成，整体的系统架构如图 *29-1:* 系统架构所示。

图 29-1: 系统架构



GTS服务端

GTS服务端是分布式事务的协调者，提供高可用、高可靠、稳定高效的事务协调能力。

主要负责分布式事务的推进，包括：

- 为GTS客户端发起的分布式事务请求分配全局唯一的事务ID。
- 处理GTS资源端提交的事务分支注册及状态。
- 负责全局事务的提交或回滚。

GTS客户端

GTS客户端是分布式事务的发起方，部署在您的代码中，是您使用GTS的接口。主要负责：

- 发起事务，界定事务边界。
- 通知GTS服务端开始或者提交一个分布式事务。
- 从配置中心获取GTS服务端列表。
- 控制GTS资源端执行业务的数据操作。

GTS资源端

GTS资源端（包括数据库、消息系统等）负责具体的资源操作，在操作过程中，记录必要的事务日志并将执行状态汇报给GTS服务端。

29.3 组件与作用

GTS与所依赖的各个组件功能如表 *29-1:* 组件作用 所示。

表 29-1: 组件作用

组件分组	组件	功能简单说明
GTS server	GTS server	负责分布式事务的推进，为GTS客户端发起的分布式事务请求分配全局唯一的事务ID，并记录资源管理器提交的事务分支的状态，最终负责全局事务的提交或回滚。
GTS client	GTS client	GTS客户端部署在用户代码中，是用户使用GTS的接口。
	GTS-DRDS client	GTS在DRDS server端集成了一个GTS client，帮助用户在DRDS上使用GTS时，无需再依赖GTS client。
公共基础组件	Diamond	负责提供系统的配置管理能力，是GTS的配置中心。
	Butler	负责提供系统监控报警能力。部署上完全独立的中间件监控组件，通过配置方式实现对目标的系统级监控。

29.4 功能特性

GTS 主要能够解决以下三方面的问题：

跨数据库的分布式事务

业务初始阶段往往规模比较小，大多情况下，单库就可以满足需求。经过一段时间后，业务规模也会随之变得大而复杂，会出现分库的情况，这时原有的单机事务往往会变成分布式事务。使用 GTS 可以让您像使用传统单机数据库事务一样，轻松接入分布式事务，且代码改动成本极低。

跨消息和数据库的分布式事务

在某些业务场景中，需要进行多个 DB 操作的同时，还会调用消息系统，DB 操作成功、消息发送失败或者反过来都会造成业务的不完整。GTS 可以让您轻松解决消息系统和数据库的事务，将各个资源加入事务范畴。

跨服务的分布式事务

业务完成服务化后，资源与客户端调用解耦，同时又要保证多个服务调用间资源的变化保持强一致，否则会造成业务数据的不完整。GTS 支持跨服务的事务，无论您使用的是 Dubbo、Spring Cloud，还是 EDAS，都可以接入事务，让您的服务操作没有事务的后顾之忧。

29.5 技术规格

表 29-2: 技术特性参数列表

指标项	规格要求
基本功能	支持DRDS、RDS、Oracle、MySQL、PostgreSQL、OceanBase、Petadata等多种数据库的跨数据库事务。
	支持 EDAS、Dubbo、Spring Cloud 及多种私有RPC框架的跨服务事务，并深度兼容了EDAS服务框架。
	支持MQ消息队列的事务消息。
可靠性	中间状态多份落盘存储，经过严格断电测试，严格保证数据一致性。
高可用性	GTS具有同区域高可用特性，即使突发事件造成集群中某一台机器挂掉，GTS仍然能够提供原本一半的服务能力。
技术成熟	基于阿里内部长期使用与沉淀的高可用高性能分布式集群技术产品构建，团队成员具有处理这一领域问题的丰富经验。

30 云服务总线CSB

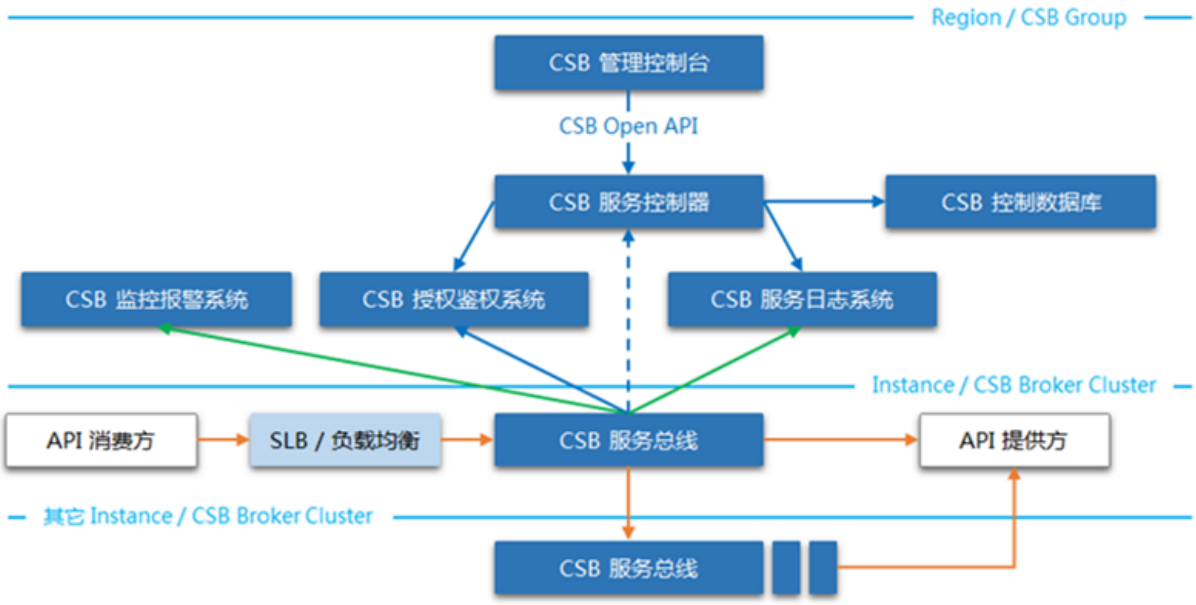
30.1 产品概述

云服务总线CSB (Cloud Service Bus) 是处理系统间服务互通和管理的计算服务，面向专有云和专有域，帮助企业在自己的多个系统之间，或者与合作伙伴以及第三方的系统之间实现跨系统跨协议的服务能力互通。各个系统以发布、订阅服务API的形式相互开放，并对服务API进行统一管理和组织，围绕API互动，实现企业内部各部门之间，以及企业与合作伙伴或者第三方开发者之间业务能力的融合、重塑、和创新。

云服务总线CSB主要针对需要对各系统间服务访问和对外开放进行管理和控制的场景，比如安全授权、流量限制等。

30.2 产品架构

云服务总线CSB的整体架构如下图所示。



针对不同场景，CSB逻辑上包含服务总线系统、管控系统和运维监控系统。

- 服务总线系统
  - # 服务总线系统提供高可用、稳定高效、可线性扩容的服务能力和服务访问控制。
  - # 服务总线系统从服务控制器获得服务定义、服务控制、以及服务授权信息，用以处理API消费方的服务调用请求，包括认证鉴权、协议转换、流量控制等服务控制能力，以及和其它服务总线实例协调处理跨多个实例的级联式服务调用。
  - # 在服务处理过程中，服务总线系统由授权鉴权系统支持完成对服务调用的认证鉴权，服务处理日志会推送到服务日志处理系统，服务总线系统的状况也会实时推送到监控报警系统。
- 服务管控系统
  - # 管控系统提供了灵活的服务管理和组织功能。
  - # 用户可以使用控制台来进行服务发布和订阅，以及对已发布、已订阅的服务进行管理和组织。和管理控制台一样，得到适当授权的用户或者系统也可以直接调用由服务控制器提供的CSB Open API，进行服务和平台的管理。
- 运维监控系统
  - # 运维监控系统提供及时详尽的系统状态信息以及服务调用状况，方便进行系统维护和问题排查。
  - # 服务总线系统向监控报警系统以及服务日志系统提供系统和服务状况信息，运维监控系统对这些信息进行处理以支持系统与服务的监控报警和分析管理。

30.3 组件与作用

CSB与所依赖的各个组件功能如表 *30-1: CSB*组件列表所示。

表 30-1: CSB组件列表

组件分组	组件	功能简单说明
CSB服务总线	CSB Broker	负责各自服务域的服务接入和开放控制。
CSB管控中心	CSB Console	负责服务、用户、系统的维护、管理的控制台。
	CSB Admin	负责支持CSB控制台并向CSB服务实例提供管控服务。
	CSB Cache	负责服务定义、授权、系统配置、消费状态等信息的高速缓存。
	CSB Store	负责保存实例、服务元数据、用户和订购信息。

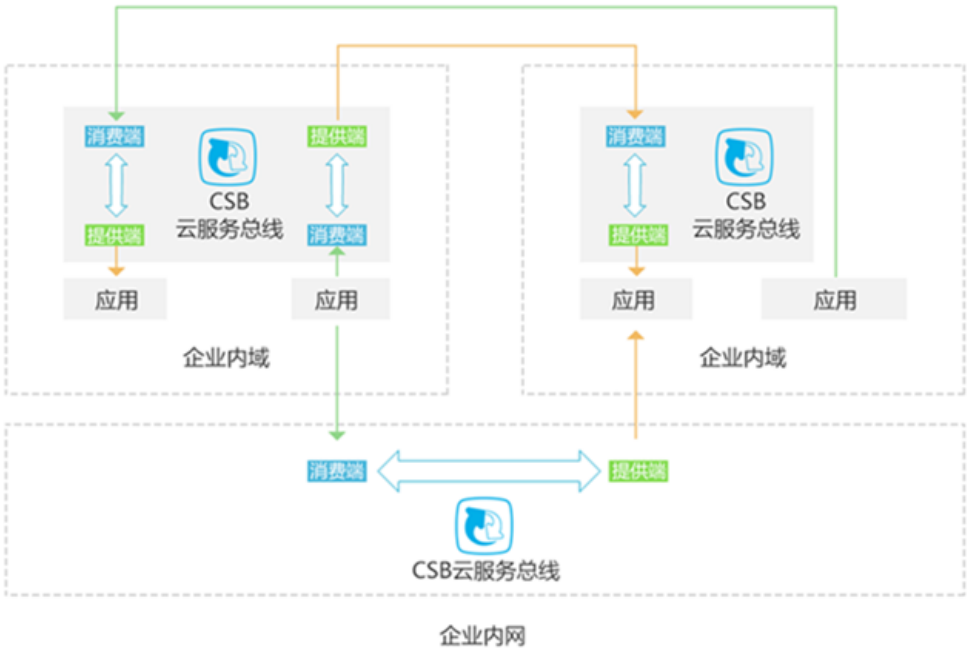
组件分组	组件	功能简单说明
公共基础组件	TLog	负责提供服务日志采集统一配置和管控。
	DAuth	负责提供用户账号系统接入和服务访问的认证、授权、鉴权。
	HBase	负责海量服务数据的分布式存储。
	JStorm	负责服务、数据的实时计算。
	Butler	负责提供系统监控报警能力。部署上完全独立的中间件监控组件，实现系统级监控。

30.4 功能特性

30.4.1 服务能力开放场景

面向企业专有云，其典型场景为内部（各个部门、地区、业务系统之间）互通以及内外（与合作伙伴、第三方开发组织等）互通。下图所示就是一个典型的用CSB支持内部互通混合内外互通场景：

- 一个域的应用通过另一个域的CSB实例访问其内部应用服务。
- 两个域的CSB实例构成的桥接通道实现互通。
- 各个域之间通过企业统一的CSB实例实现互通。



实际上，云服务总线CSB产品还支持更复杂的多实例，甚至多群组的联动场景。对应分级、分层，或者有特殊的系统间隔离要求，以及更复杂的多个环境如企业内网和公共云的混合场景。

30.4.2 API服务总线

可以在在服务能力开放平台建设中提供基本的实时服务控制能力，包括：

- 协议转换：支持常用协议服务的接入和开放，处理协议转换和接口映射。
- 认证鉴权：判断是否是合法用户，是否已授权访问当前服务。
- 服务路由：根据请求参数，将请求路由到相应的后端服务端点。
- 服务控制：支持调用者的服务访问流量、实例级的总体访问流量，及基于黑白名单的实例级和服务级的访问流量的限制和检查。

关于服务控制，还有其他特色功能，例如请求校验、响应缓存以及定制化报文转换等功能，正在逐步产品化整理推出。

其中请求校验可以按照用户指定的规则来检查请求内容，判断是否是合理请求，检查入参的格式、取值方位、组合等等，来过滤无效请求，降低对后端服务提供方的压力；而响应缓存可以按用户指定的规则，对指定的API服务，在一段时间内直接缓存回复相同内容的请求，在大量消费者的情况下，可以极大降低对后方提供者的压力。

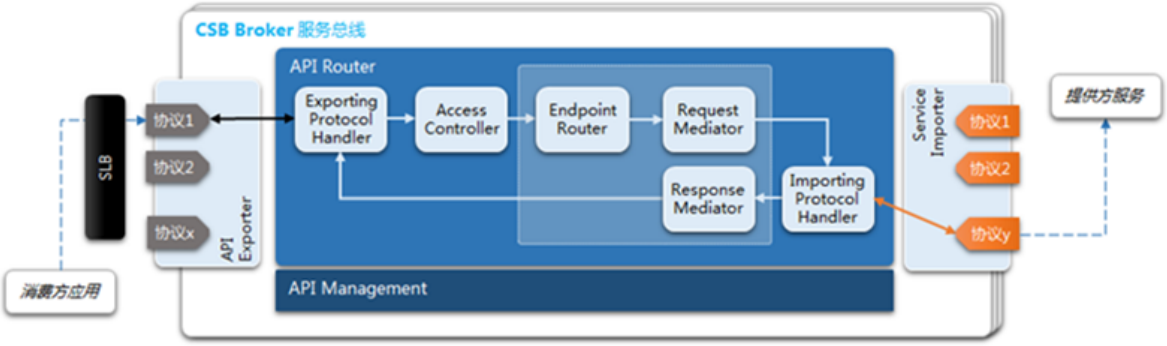
30.4.2.1 协议转换

需要连通的各方系统很可能是基于不同的技术架构构建，提供的对接接口使用的协议各有不同，有的可能是HTTP OpenAPI ，有的可能是Web Service ，有的可能是应用互联网架构新建的系统，例如Dubbo或者EDAS的HSF服务，甚至可能是完全私有定制化的协议。要实现互通，就必须解决多个不同的协议之间如何转换的问题；同时对服务的提供方而言，所提供的现存服务的协议是确定的，但开放出去，很可能面临着完全不同的消费偏好，需要支持一个服务用不同的多种协议同时进行开放。

协议转换包括两个部分：

- 将一种协议的请求转换成另一种协议的请求。

  例如HSF服务开放成了HTTP OpenAPI，CSB接收到HTTP OpenAPI请求时，需要能把请求内容转换成对指定HSF服务的访问请求，在收到HSF的服务回复后，再转换成HTTP OpenAPI的响应结果。



还要考虑特殊情况，例如接入的服务使用了特殊的协议，或者需要遵循服务提供方的安全处理，这时就需要产品提供定制化协议的支持。

在CSB上发布服务的时候，在指定了接入的服务协议后，可以选择用哪些协议开放出去。对应不同的接入类型，即服务提供方的原有协议类型，需要提供不同的接入信息让CSB知道如何能访问这个服务。如果需要，CSB可以提供定制化服务支持特殊的协议的接入和开放。

- 将按消费方根据开放定义发出的请求，映射到接入方（提供方）的接口

  服务开放出来的接口，有可能和服务自己实际的接口是不一致的。例如有些入参或者出参并不想暴露，或者使用不同名字，等等。这时就需要在服务自己的接口（接入接口）和开放接口间做映射。CSB提供的基本方式是以服务自己的接口作为基础来指定映射。

30.4.2.2 认证鉴权

服务的开放要安全可控。云服务总线CSB的授权鉴权组件支持对接企业自己的账号系统，CSB产品提供身份认证和访问鉴权，检查服务调用者是否是合法用户，以及是否已授权可以消费该服务。

30.4.2.3 服务控制

除了基于IP的黑白名单，以及其他正在产品化的请求校验、响应缓存等功能，CSB支持的典型的服务控制就是流量控制，可指定访问频度限制，即每秒最多调用次数。首先要支持的，是用户的单个消费凭证对单个API的访问频度限制，其次还要考虑系统级的保护。所谓系统级的保护就是防止整体服务消费频度过高，在压力大时进行调整限流，回绝部分访问请求，保持系统高效的处理效率。

30.4.3 API管理组织

API服务的管理针对两个不同的角色：发布者和消费者。从发布者角度出发，希望能对自己发布的服务有很好组织，及时掌握所发布服务的订阅、消费情况，可以方便地管理对所发布服务的订阅授权；从消费者的角度出发，希望能方便地定位找到自己想要的服务，选择质量或者服务水平更适合自己的服务，方便地管理自己的订阅，及时掌握所订阅服务的消费情况等等。

服务管理除了服务本身的生命周期管理，如注册发布、启用停用、下线注销等，还要支持服务的订阅审批、授权鉴权，以及服务目录的管理。主要的角色和操作有：

- 服务发布者：服务组管理、发布服务、发布管理。
- 服务订阅者：消费凭证管理、订阅服务、订阅管理、订购审批。
- 系统管理员：实例管理、用户管理。

30.4.3.1 服务发布

发布API有如下几个关键概念。

- 接入服务：提供API对应的后端服务信息，让CSB能访问到这个已有的服务。
- 开放服务：指明API开放的协议，以及开放的接口和后端服务接口如何对应。
- 访问控制：指明API开放的策略，是否限流，对谁可见，访问是否需要授权。

目前，CSB用服务分组实现了对于用户所发布服务的原子分类，至于更复杂的服务目录结构的组织方式，往往具体的业务需求有很大差异，CSB建议业务方视具体场景单独构建。

30.4.3.2 服务授权

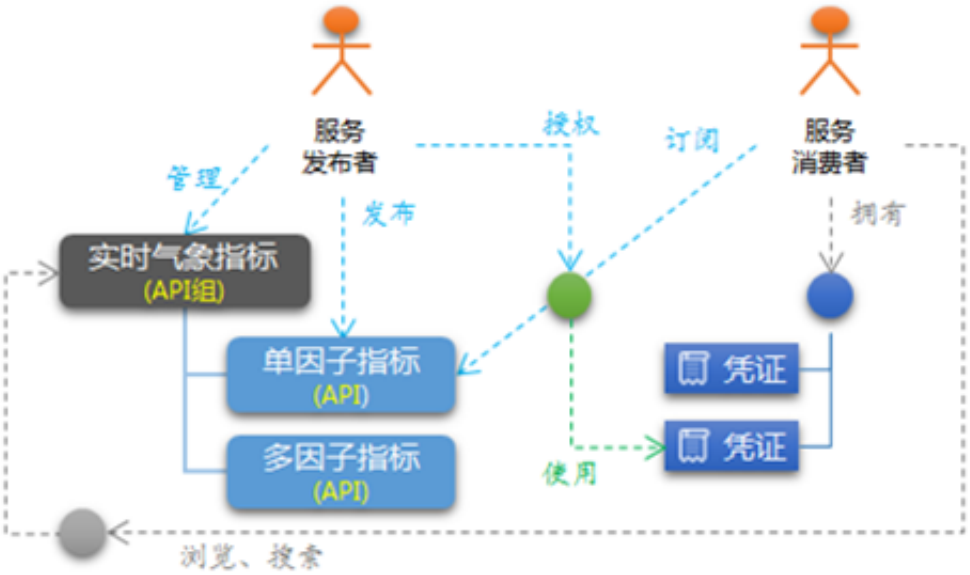
在云服务总线CSB中，用户是对等的，没有从属概念，只有授权关系。用户可以拥有属于自己的一个或多个CSB实例，具有这些CSB实例的管理员权限。可以定义用户分组，控制其他用户对这些CSB实例的使用权限，即其他用户是否能在某个实例上发布和订阅服务。实际在企业中，往往只有少量用户会拥有CSB实例，大多数用户只是被授权访问使用CSB实例。

用户在取得某个CSB实例的访问权限后，可以在该实例上发布或者订阅服务。服务的发布者就是该服务的拥有者，可以审批授权其他用户的订阅申请。类似的，CSB用户在这里同样是对等的，CSB只关心两个用户之间针对具体服务的授权关系，并不关心用户在企业组织结构中的实际层级和隶属关系。

订阅API有如下几个关键概念。

- 消费凭证：API消费方应用在调用API时用的凭证，一个用户可有多个凭证。
- 订阅服务：API消费者使用自己的某个消费凭证来订阅API 。

- 授权服务：API发布者对API订阅授权，包括允许、禁止、流量控制等。
- 搜索服务：API消费者通过服务名、服务组名来搜索和浏览可订阅的API 。



云服务总线CSB还有群组的概念，对应于相对隔离的管理环境。例如企业的内部数据中心和阿里云的某个地域（Region）即是不同的群组。相应地，也有CSB群组管理员的角色，与CSB实例管理员不同，只有群组管理员可以应用户请求创建CSB实例，以及设置跨实例的服务链路。

30.4.3.3 服务消费

云服务总线CSB提供服务消费的计量统计，可以让服务的发布者和消费者进行查看，了解服务消费量的实时统计，以及服务消费质量信息。CSB为服务发布者提供服务被调用的详情，包括整体的服务调用量，各种类型的错误占比，近一段时间的消费曲线，以及各个消费者的消费统计等等；类似地，CSB也为服务消费者提供调用服务的详情。

服务消费计量也可以用来做服务计费。目前CSB没有计划提供服务调用的定价计费功能。

API消费方应用调用API有多种方式，例如HTTP API、HSF API、Web Service API等。其中HSF API的消费调用沿用HSF原有的服务调用方式，无需任何专用SDK ，如有必要可以指定CSB的服务IP进行HSF服务调用。而调用HTTP API则需要使用CSB Client SDK ，Web Service则需要使用CSB WS SDK；目前提供了Java、PHP版本的HTTP Client SDK和WS SDK。

30.4.3.4 API运维监控

日志监控

提供完整的服务和系统的日志、巡检和监控，包括对CSB所有组件系统指标的监控以及运行环境指标的监控，例如：单机和集群的负载、内存和CPU的使用率，JVM的线程、内存以及GC情况等等。还有自身服务处理的工作情况，包括各种异常情况的记录。还可以配置巡检规则，对指定的指标定期巡检并制定报警规则。此外，服务链路的分析信息也很重要，有助于快速定位是哪个环节什么原因出现问题。

系统管理

包括实例及实例上发布规则的管理，以及用户访问授权、用户组定义管理等等。

注意：
有些管理功能是只有群组管理员才有权限的，例如跨实例调用链路的定义，对用户发起的实例创建请求的审批等。

30.5 技术规范

本章介绍CSB的技术规范，包括：服务开放规范、服务管理规范、服务安全规范、服务消费标准以及企业自有账号系统接入DAuth标准。

30.5.1 服务开放规范

介绍服务开放过程中，命名、协议转换、服务路由、协议接入与开放及安全控制等方面的规范。

30.5.1.1 命名规范

介绍API相关的命名规范。

API分组

API分组的命名长度为1~64个ASCII字符，允许大小写英文字母、数字0~9、以及减号“-”。

CSB目前并不限制一个用户下API分组的个数，为良好的管理考虑，建议有意义且规范的命名。API分组的本意是对用户拥有（所发布）的API做基本的分类管理。而分类可能是多层次的，建议采用“AAA-BBB-CCC”这样的模式，表示该分组是AAA类型下BBB子类型下的CCC子类型。其中AAA、BBB、CCC可以是有意义的单词或者编码，全部小写、全部大写、或者大写字母开头混合数字。可根据业务方需要按照命名规则自定义分组名称。

API命名

API的命名与API分组的命名类似，长度为1~128个ASCII字符，允许大小写英文字母、数字0~9、以及“.”。API的命名是群组唯一且无法更改的，因此需要慎重。建议采用“aaa.bbb.ccc.ddd”的模式来命名，用“.”分成多节。各节以字母开头全部小写，除非必要或者固有名词（如DB2），不要混入数字或大写字母。可根据业务方需要按照命名规则自定义分组名称。例如“taobao.lbs.user.distance.get”。

CSB提供了单独的API版本号属性，因此不建议直接在API命名上写明API版本等信息，除非要体现接入的已有服务提供方的版本标号等信息。

API参数

API的参数设置是以提供方即接入方接口为基础的，接入参数名和接入的已有服务的参数一致，但开放出去的参数要制定统一规范。一种可参考的方式是：开放参数采用lowerCamelCase方式，由多个单词组合成一个单词串，且除了第一个词以外每个词的第一个字母大写，除非特别必要或者固有名词（如DB2），不要混入数字。

API错误代码

发布服务时，用户可以填写该服务的错误代码列表，该信息只是用于给API消费者提供信息，以便在发生调用错误时定位和理解错误原因，CSB并不会根据该列表的内容做任何处理。服务发布者应该按所接入的服务本身的错误信息提供，具体给出解决错误的方法，以及出现这个错误的可能的原因。

30.5.1.2 协议转换支持

开箱支持常用协议服务的接入和开放(HTTP/HSF/Dubbo/Web Service)。

两个概念：

- 服务接入– 在CSB上注册这个服务并且提供足够的信息让CSB可以访问这个已有的服务。
- 服务开放 – 把一个已接入的服务在某个CSB上提供对应的不同协议的API调用入口。

具体协议转换支持如下：

- 缺省支持的服务接入、开放协议如下表所示：

支持的接入协议类型	对应支持的协议开放类型
HTTP Open API 包括Restful API	HTTP Open API, Web Service
HSF	HTTP Open API, Web Service, HSF级联

Dubbo	HTTP Open API, Web Service
Web Service	HTTP Open API

- CSB支持数组、列表、集合类型的服务参数，也可以定义复杂的多级参数结构。
- CSB支持接入接口和开放接口的参数映射，可以设置缺省值以及在开放接口上是否可见。
- CSB暂不支持OAuth 2.0开放标准。
- CSB接入规范/限制如下表所示：

开放 > 接入协议对	**接入规范/限制明细**
HTTP > HTTP	**1.** 支持GET/PUT/POST/DELETE **2.** 参数传递 **a.** Query：Key-Value 形式 **b.** Body：Key-Value/JSON/byte[] 形式 **3.** 支持RESTful风格的接入，如： http://localhost:9090/aaa/{userId}/test **4.** 支持HTTP返回结果透传，不做任何变换加工
HTTP > Dubbo	**1.** 支持指定注册中心或使用默认注册中心 **2.** 支持开放一个接口的某个指定方法或全部方法（方法名用“*”） **3.** 支持Dubbo和Hessian序列化 **4.** 支持简单数据类型泛型的HashMap **5.** 返回结果为JSON格式
HTTP > Web Service	**1.** 支持两种格式的用户名密码接入SOAP Header **a.** Authorization类型 **b.** WSSE类型 **2.** 不支持WSS和Policy等相关的安全和加密处理 **3.** 不支持MTOM等附件数据传输方式的调用 **4.** 对于anytype参数类型，参数定义时要经过特殊定义为“ws:anyType” **5.** 接入后端WS服务方法必须是同步的，不支持异步 **6.** 支持的bindType是Document-Literal (Wrapper or Unwrapper) **7.** 返回结果是将SOAP XML转化为JSON格式
HTTP > HSF	1. 允许HTTP API参数进入HSF RPC上下文 2. 允许指定后端HSF RPC租户信息、超时时间 3. 开放的HTTP API支持RESTful风格

Web Service > HTTP/HSF/ Web Service	1. 相应服务会以SOAP 1.1的格式发布成一个Web Service并提供标准的WSDL 2. 需要使用CSB提供的WS Client SDK（目前有Java和PHP版本）传递AccessKey和signature等相关Header信息 3. 暂不支持入参的可选和默认值设置 4. 支持Web Service透传，在此模式下支持MTOM等附件数据传输方式
Web Service > Dubbo	1. 支持指定注册中心或使用默认注册中心 2. 支持开放一个接口的某个指定方法 3. 支持Dubbo和Hessian序列化 4. 返回结果为JSON格式

30.5.1.3 服务路由

支持参数路由设置，在服务发布的时候，指定对应的参数和相应的表达式，在服务调用的时候，根据不同的参数值将请求转发到相应的服务节点进行处理。

30.5.1.4 协议接入与开放建议

要在CSB上接入一个已有服务，需要看这个服务的实现协议是什么，如果不在上述1.2 章节协议转换支持列表中的话，就需要业务方单独开发，对该服务进行包装和封装之后，再接入CSB。

对于服务的开放协议，如果消费方没有特别的要求，建议使用HTTP Open API。

30.5.1.5 安全控制

关于安全控制的完整详细说明，请参考云服务总线CSB的安全技术白皮书，这里仅介绍在服务发布时相关的安全控制。

30.5.1.5.1 允许未授权访问与黑白名单

服务在发布时可以指定是否允许未经授权访问。如果开启未授权访问，对该服务API的访问不会做授权检查，但是签名检查还会进行，即还是会检查当前消费者是否是合法用户，当前请求是否是伪造请求。

未授权访问开关可以和黑白名单控制配合，形成期望的访问控制效果。例如可以开启未授权访问，同时配置黑白名单，使得固定来源的服务请求方不需要经过授权即可访问。但是注意，目前未授权访问开关是API整体粒度的，暂不支持设置一个API，对某些访问来源不做授权检查，而对其它来源做授权检查。

黑白名单并不是在服务发布时就指定的，而是在发布以后另行设置。

云服务总线CSB提供黑白名单功能，通过IP对服务调用的来源进行过滤。基本策略是黑名单优先，就是说首先会检查请求来源IP是否在黑名单内，如果是的话直接拒绝，然后看是否在白名单内，是的话就通过，否则依据系统的缺省策略（通过/拒绝）来处理。在经过黑白名单的过滤后，才会进入验签和授权检查阶段。如果目标服务开启了未授权访问，则只会做验签，之后的授权检查会直接通过。

30.5.1.5.2 支持RBAC

CSB Console实现统一权限管理，可自定义角色及其对各种资源的权限设置。角色、角色对应的权限、角色的授权规则可以定制修改；资源类型、操作这些是系统预置确定的不可以定制。目前暂以产品API方式提供数据配置，未提供配置操作界面。

30.5.2 服务管理规范

介绍API组织、用户、实例、群组以及凭证相关的管理规范。

30.5.2.1 API组织

API组织首要的就是API分类，在上一章的API分组命名规范中已经提到，API分组提供了简单的一级API分类。

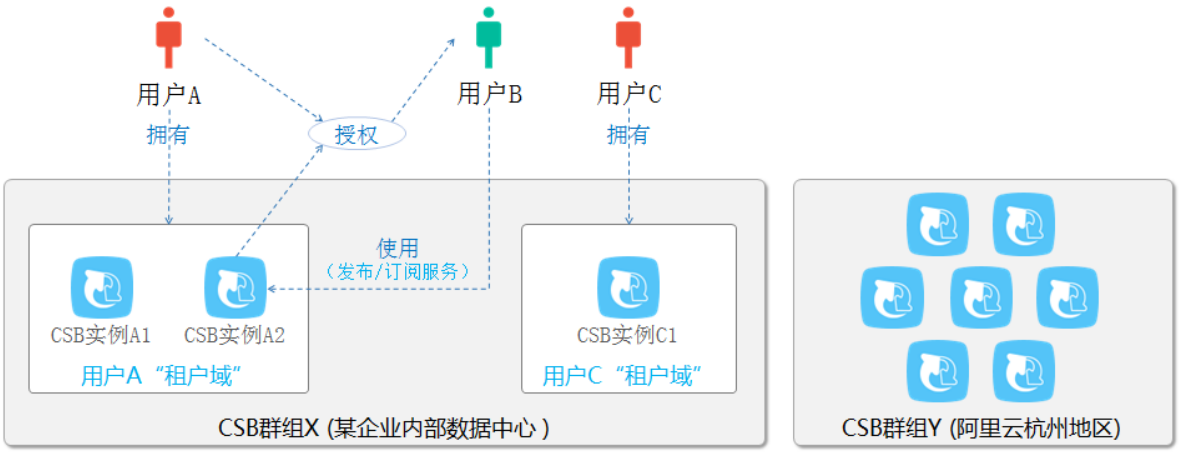
就分类来讲，不同的业务场景需求千变万化，可能是简单的一级分类就够了，例如淘宝开放平台的“用户/类目/商品/交易/...”一级分类方式；也可能是需要复杂的多级树形目录结构，例如常见的各种APP Store的分类方式。CSB目前是不支持复杂的树形目录结构的，实际上复杂类目的定义和管理也是一个业务定制化的功能，需要业务方自行实现单独的分类管理体系，再将具体的API与一个或多个分类节点做关联。

在CSB中良好API分组的命名有助于关联，业务方可能有多个不同的分类定义（国标、按组织结构、按业务类型等等），这里建议确定一个单一的基本标准分类，并按此标准对API分组命名。

30.5.2.2 用户、实例、群组

介绍用户、实例和群组的管理规范。

图 30-1: 用户、实例和群组关系示意图



用户

- 用户关系、服务订阅-审批

  在云服务总线CSB中，用户是对等的，没有从属概念，只有授权关系。业务方组织结构和复杂的人员隶属关系在CSB并不体现。如果业务方有特别流程上的控制，例如服务的发布需要发布者所属领导审批，这种流程需要在CSB之外通过工作流产品或者流程引擎实现，CSB提供产品API支持，例如在审批流程完成后实现API的最终发布生效。

  同样，两个用户之间发生的服务API订阅-审批也是如此，CSB不约束任何两个用户之间的API订阅-审批关系。如果业务上需要限制某个用户的API不可以被某个特定用户订阅，需要独立于CSB做额外管理和控制，CSB本身控制台不能做这样的约束，服务可见域的功能还未正式推出。

- 用户与实例，用户账号系统

  每个用户都可以拥有属于自己的一个或多个CSB实例，具有这些CSB实例的管理员权限。可以授权控制其他用户对这些CSB实例的访问使用权限，即是否允许其他用户发布服务、订阅服务。每个用户和他所拥有的所有CSB实例，即构成该用户的CSB租户域。每个实例有且只有一个拥有者。

  在企业当中需要的CSB实例，即服务开放平台应该很少，大多数企业只需要一个就够了。因此相应的真正拥有实例的用户也应该很少，大多数用户只是使用别人的实例，进行服务发布和订阅。对应到具体的企业业务场景，需要明确以下两个问题中关注的人群，并为之开通CSB用户账号。

1. 需要使用CSB实例，即进行发布和订阅操作的人有哪些？
2. 需要拥有CSB实例，即管理某个服务开放平台的人是谁？

CSB可以通过其DAuth组件对接企业自己的用户账号系统，也可以直接由DAuth提供这一功能。注意DAuth的主要职责是提供授权鉴权功能，可以对接一个用户已有的账号系统，不建议直接使用它来做账号分发管理。

关于DAuth的接入细节，请参考附录中的附录-企业自有账号系统接入*DAuth*标准一节。注意，DAuth仅提供接入标准，业务方需要按此接入标准，提供DAuth需要的登录页面实现，以及给DAuth调用以获取用户信息的接口。

实例

在云服务总线CSB中，一个实例就是一个开放平台，大家可以在这个平台上发布和订阅服务，每个实例都是由一组CSB服务总线节点（Broker）构成的集群，通常负责一个业务域内能力的对外开放，也可以把外部的能力发布开放给业务域的内部。
实际上，用户通过CSB控制进行实例创建的操作仅仅是发起一个实例创建的申请，用户在这个操作中只需要提供实例名称，实例的创建还需要提供机器资源并进行部署和激活。

群组

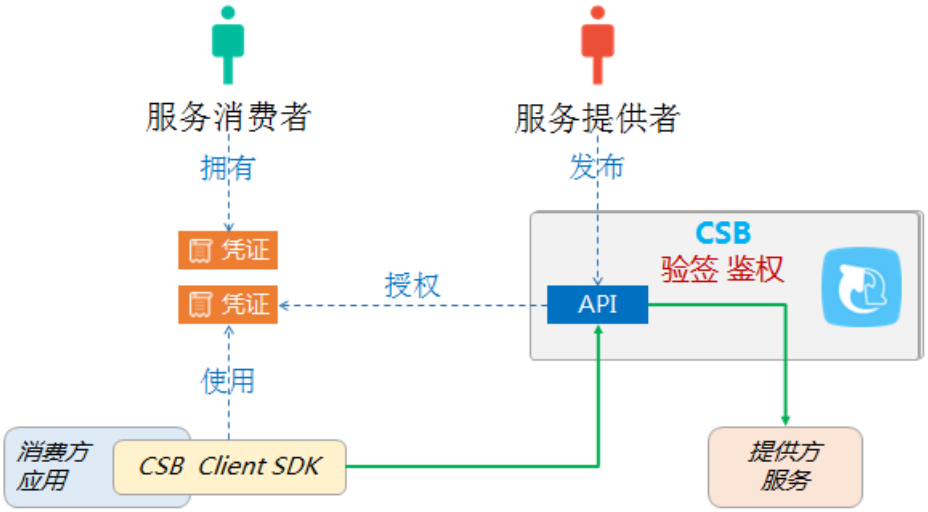
云服务总线CSB有群组的概念，对应于相对隔离的管理环境。例如企业的内部数据中心和阿里云公共云的某个地域（region）即是不同的群组。相应地，也有CSB群组管理员的角色，与CSB实例管理员不同，只有群组管理员可以应用户请求创建CSB实例。例如在阿里云公共云环境中，CSB产品运维团队即是该群组的管理员。

企业一般不会有多个群组，除非企业存在多个不同的物理上或者逻辑上隔离的业务域，甚至连账号体系都不是一个，才需要用多个群组来各自管理并实现互通。要注意的是，每个群组都要有自己的群组管理员，负责管理群组内的用户，以及制定群组内各个实例之间的互通规则。

30.5.2.3 凭证

云服务总线CSB的凭证即服务API消费凭证。每个用户可以创建多个消费凭证，用以订阅服务。注意用户是使用凭证订阅服务API，即与服务API构成订阅关系的是凭证，而不是用户账号，真正在运行时调用服务的时候，使用的也是消费凭证。

图 30-2: 服务凭证示意图



服务消费者的应用使用CSB Client SDK对服务API调用进行签名，该调用请求到达CSB后，CSB会：

- 先进行签名验证（验签），确认请求方的身份，以及消息没有被篡改重放；
- 再进行鉴权，检查该调用所使用的消费凭证是否已经授权允许调用该服务API。

用户创建凭证，只需要提供凭证名称，该名称是当前用户唯一的。通常，一个凭证对应该用户所需API的一个或一组现实应用，因此对凭证的命名也应该体现这一对应关系。具体规范可根据业务场景自行定义。

有一种特殊的情况是，用户创建凭证是给别人用的，因为其他人只要知道该凭证的信息就可以调用CSB上订阅的服务，不需要成为CSB的正式用户，这种场景下，凭证的命名可以标识对应的使用方。

30.5.3 服务安全规范

请参考《*CSB* 安全白皮书》。

30.5.4 服务消费规范

请参考 《*CSB SDK* 参考》。

30.5.5 附录-企业自有账号系统接入DAuth标准

介绍DAuth及其实现细节。

概述

客户账号中心以类OAuth协议方式接入DAuth单点登录体系。要求客户实现登录页面并在浏览器端保存登录凭证（Cookie），提供通过登录凭证换取登录用户信息的后台接口（HTTP/HTTPS）。

实现细节

- 实现登录页面

  通常在客户原有登录页面基础上进行简单改造，除基本的验证用户密码功能外，需满足两点需求：

  # 登录成功后跳转

    通过登录页面的URL参数（参数名自定义，如”back_url”）指定登录后返回页面地址，该参数需经过URL_ENCODE（编码方式可指定，默认UTF-8）。当验证用户密码通过后，服务端将控制浏览器重定向到该参数指定URL。

  # 浏览器端保存凭证

    当验证用户密码通过后，服务端在返回重定向的同时向浏览器写入Cookie，保存登录凭证（Cookie名自定义，如”sso_token”）。Cookie要求写入登录应用的上级域，以确保同域的应用能够获取。凭证内容需要能够唯一定位登录用户身份，如包含用户唯一标识。出于安全考虑，通常进行加密存储（算法无要求，服务端可以自己解密即可），并加入过期时间戳等信息。

- 实现获取用户接口

  通过登录时写入浏览器端的登录凭证，调用后端接口获取用户信息

  # URl: 自定义，如/sso/getLoginUser

  # 方法: GET

  # 协议: HTTP/HTTPS

  # 参数：

    token - 字符串，从cookie中获取的登录凭证

  # 返回为JSON格式，包含字段：

    ■ code - 整型，0表示成功，其他失败

  - message - 字符串，失败时的错误信息
  - loginId - 字符串，账号的登录名，全局唯一
  - uid - 字符串，账号的唯一ID，可以与loginId相同
- 安全：

  若需防止数据被旁路监听，可以采用HTTPS协议;

  若需控制对接口的调用，可以引入签名机制：服务端对授权的调用方颁发一对秘钥appKey和appSecret，调用方请求参数中携带appKey和用appSecret对token生成的签名signature（HmacSHA1算法）。服务端收到请求先根据appKey找到对应appSecret，对token重新生成签名，与请求中的signature进行比对，达到验证身份的目的。

实现查询用户接口（可选）

根据登录名查找账号唯一ID。若二者相同，可以不提供。

- URI: 自定义，如/sso/findUser
- 方法: GET
- 协议: HTTP/HTTPS
- 参数：

  loginId - 字符串，账号登录名
- 返回为JSON格式，包含字段：
- code - 整形，0表示成功，其他失败
- message - 字符串，失败时的错误信息
- uid - 字符串，账号的唯一ID
- 安全：

  若需防止数据被旁路监听，可以采用HTTPS协议;

  若需控制对接口的调用，可以引入签名机制，参考获取用户接口，其中签名是对loginId计算生成。

30.6 技术规格

指标项	规格要求
基本功能	多协议服务互联，支持HTTP OpenAPI、HSF、Dubbo、Web Service等常用协议。

指标项	规格要求
	提供完备的服务安全调用，包括加密，鉴权，流量控制保护，服务路由、黑白名单等访问控制。
	提供多个服务节点集群之间的级联API发布管理。
	支持从发布、订购、消费到注销的API全生命周期管理。
	提供完善的服务访问授权管理机制。
	提供及时详细的服务消费统计与质量报告。
	提供完整的针对服务链路和系统指标的日志、巡检和监控。
	提供系统用户管理，以及跨实例链路的规则配置管理。
可靠性	数据系统采用多级缓存和主备存储方案。
	API服务系统采用无状态的集群结构，采用多级缓存系统，后端的数据库宕机，不影响开放平台提供服务。
扩展性	支持服务节点的无间断扩容。
技术成熟	基于阿里内部长期使用与沉淀的高可用高性能分布式集群技术产品构建，团队成员具有处理这一领域问题的丰富经验。

31 分布式调度任务

31.1 什么是分布式任务调度SchedulerX

分布式任务调度 SchedulerX 为客户提供各种各样精确到秒级的高可用的任务调度服务。

SchedulerX 主要有以下作用：

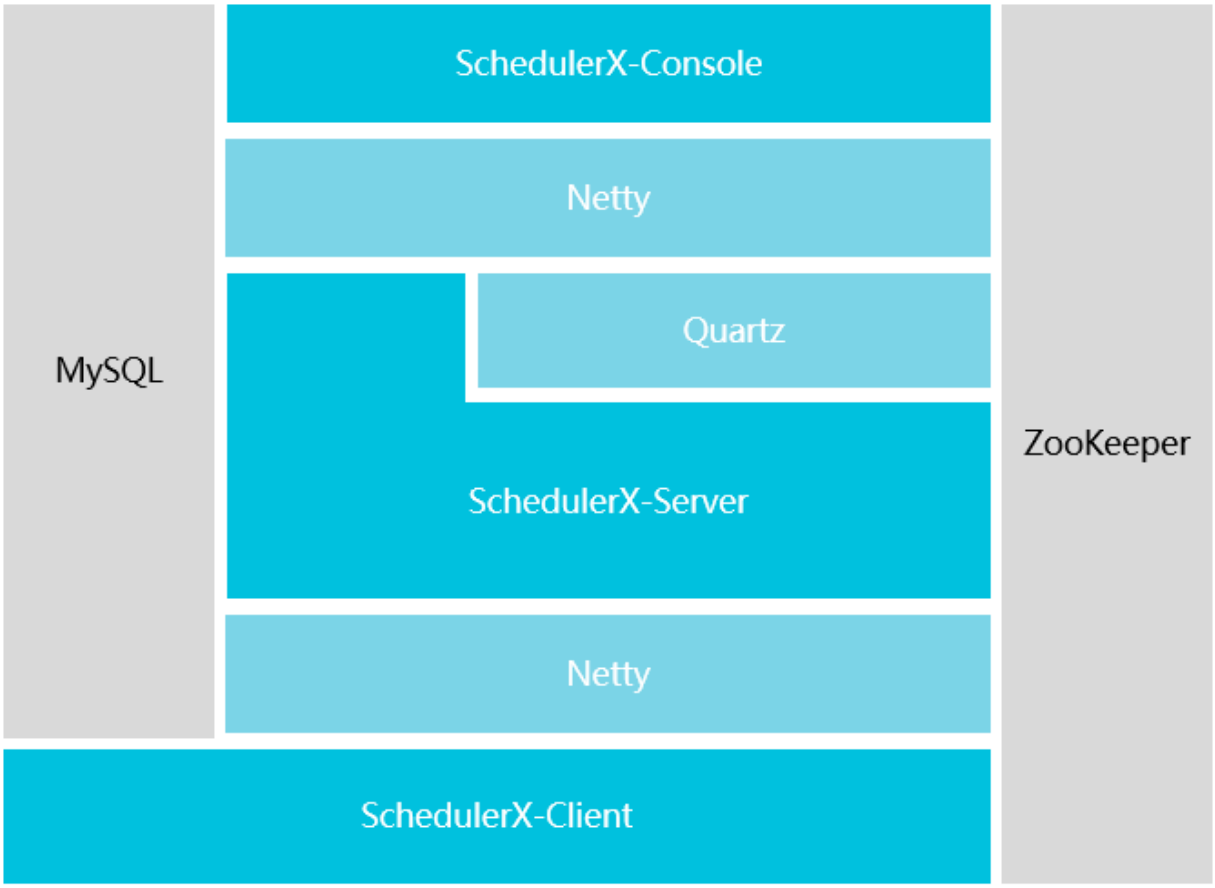
- 作为定时、即时任务调度系统，帮助用户按计划批量处理任务。
- 其分布式特性保证了系统的高可用，同时提高了系统的整体性能。
- 支持多种任务类型的管控，满足用户各种使用场景的需求。

31.2 产品架构

本文档介绍SchedulerX的产品架构，以及各模块的作用。

SchedulerX的架构如图：

图 31-1: SchedulerX产品架构



SchedulerX各模块的作用如下：

- **SchedulerX-Server**：SchedulerX 的调度中枢，负责 Job 的触发、状态统计等。
- **ScheudlerX-Console**：SchedulerX 的 Web 控制台，是人机交互的主要入口。用户登录 SchedulerX-Console，可以进行 Job 的日常管控，包括 Job 维护，执行状态查看等一系列操作。同时，客户端通过 SchedulerX-Console，可以发现 SchedulerX-Server 的 IP 列表，客户端基于此找到需要连接的 Server 并建立连接。
- **SchedulerX-Client**：作为客户端嵌入到用户自己的应用中，是 Job 真正运行的地方。Server 发起触发命令，Client 收到命令后，执行用户自定义的业务逻辑，并上报执行结果给 Server。
- **ZooKeeper**：作为 Server 列表和 Console 列表的注册中心，同时作为多 Server Job 触发的协调中心。
- **MySQL**：作为 Job 相关元数据配置的存储中心，同时保存着中间运行时状态，包括 Job 触发实例、子任务等数据。
- **Netty**：Netty 是各模块之间通信的网络底层框架。

31.3 产品功能

本文档介绍SchedulerX的功能。

- **简单 Job 单机版**

  简单 Job 单机版是最常见的应用场景，每次触发会随机选择一台客户端机器去执行任务。

- **简单 Job 多机版**

  同一分组下的机器同时运行一个相同的任务。

- **网格 Job**

  将大 Job 拆分为多个子任务，并分发到多台机器上执行。能极大提高任务的整体执行速度，支持多级分发、一级分发限流和子任务失败重试等特性。

- **并行 Job**

  和网格 Job 类似，但是不支持拆分成大量的子任务，拆分子任务数上限为1000，不支持失败重试，不支持限流。

- **顺序流的依赖 Job**

  可以通过配置依赖 Job，让一组任务按顺序执行。

- **常驻 Job**

常驻 Job 的特点是，一次触发，一直运行。支持负载均衡，客户端机器扩缩容后，重新负载分片。

- **Web 控制台管控**

  完善的 Web 控制台，拥有众多的 Job 管控入口，包括修改、触发、启动、停止、触发历史查看等特性，是用户管控任务非常方便的工具。

31.4 产品优势

介绍SchedulerX的优势。

系统可靠

- 分布式的系统架构，保证服务端同时有多台 Server 管控 Job，保证任务的正常触发和执行。
- 客户端也是分布式的架构，一个客户端任务触发失败，会有另一个客户端来执行任务。
- Job 元数据信息在服务端保存，同时可以通过备份方式来保证数据的安全。

数据安全

通过分组隔离应用之间的 Job 相关数据，同时通过权限控制来约束用户只能管控自己的 Job，保证了数据的安全性。

性能卓越

分布式的系统架构，能大大提升系统的处理能力。几乎所有性能都取决于客户端机器的处理能力，客户端机器能线性扩容来提升性能。

简单易用

- 接入简单：客户端只需要引入一个 JAR 包即可，同时支持 Spring 方式和编码方式来启动应用。
- 编程模型简单：只需要实现一个接口，并覆盖其中的方法即可。
- Web 控制台拥有丰富的管控功能，包括任务的查看和修改，以及执行进度和执行结果的查看。

31.5 应用场景

介绍SchedulerX的应用场景。

- **固定时间点触发的任务**

  例如：2016年11月11日0点执行的一次任务。

- **周期性触发的任务**

  例如：每秒钟（或者每小时、每天、每星期、每月等）执行一次的任务。

- **通过控制台手动触发的任务**

  例如：可以通过控制台手工触发任务的调度执行。任务触发执行后，由用户实现的 Job 处理器接口中的代码决定具体要完成的业务逻辑功能（例如扫表、触发 RPC 调用、入库、执行本地脚本等）。

32 MaxCompute

32.1 什么是MaxCompute

大数据计算服务（MaxCompute）是基于飞天操作系统分布式平台，由阿里云自主研发的海量数据离线处理服务。MaxCompute提供针对TB/PB级别数据、实时性要求不高的批量处理能力，主要应用于日志分析、机器学习、数据仓库、数据挖掘、商业智能等领域。

32.2 产品特性和核心优势

产品特点

- MaxCompute是面向大数据处理的分布式系统，主要提供结构化数据的存储和计算，是阿里巴巴云计算整体解决方案中最核心的主力产品之一，是阿里巴巴大数据平台的基础计算平台。MaxCompute中的多租户、数据安全、水平扩展等特性是MaxCompute的核心设计目标，采用抽象的作业处理框架为不同用户对各种数据处理任务提供统一的编程接口和界面。
- 采用分布式架构，规模可以根据需要平行扩展。
- 自动存储容错机制，保障数据高可靠性。
- 所有计算在沙箱中运行，保障数据高安全性。
- 以RESTful API的方式提供服务。
- 支持高并发、高吞吐量的数据上传下载。
- 支持离线计算、机器学习两类模型及计算服务。
- 支持基于SQL、MapReduce、Graph、MPI等多种编程模型的数据处理方式。
- 支持多租户，多个用户可以协同分析数据。
- 支持基于ACL和policy的用户权限管理，可以配置灵活的数据访问控制策略，防止数据越权访问。
- 支持ElasticSearch的增强应用，即ElasticSearch on MaxCompute。
- 支持Spark的增强应用，即Spark on MaxCompute。

产品优势

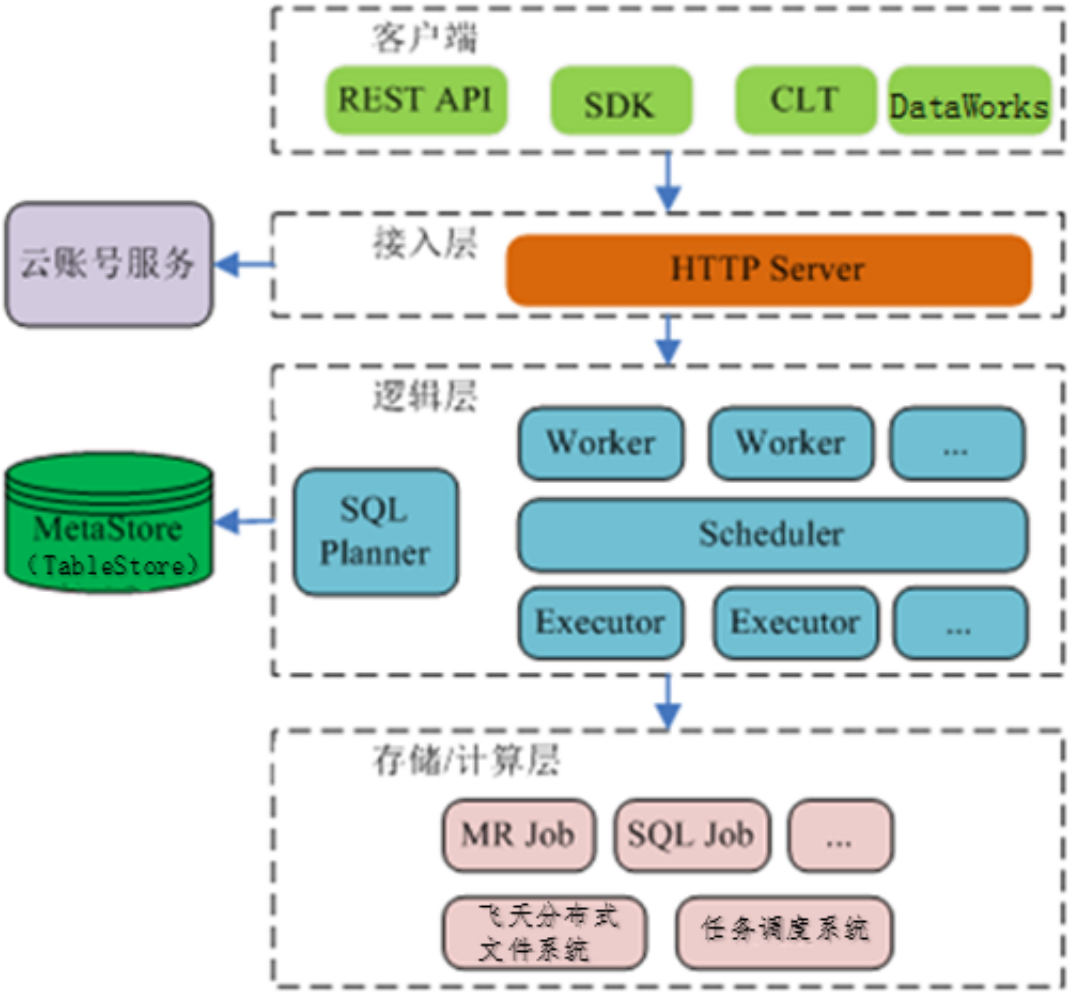
- **国内唯一的大数据云服务平台，真正的数据分享平台**：能够同时做到数据仓库、数据挖掘、数据分析、数据分享；同时，还是阿里集团内部使用的统一数据处理平台，支持阿里贷款、数据魔方、DMP（阿里巴巴广告联盟）、余额宝等多款产品。

- **支持的集群及用户规模极大，同时能够支持极高的作业并发数**：单一集群规模可以达到10000+服务器（保持80%线性扩展）；单个MaxCompute可以多集群部署100万服务器以上，无限制（线性扩展略差），支持同城多数据中心模式；10000+用户数，1000+项目应用，100+部门（多租户）；100万以上作业（目前单日平均提交任务），20000以上并发作业。
- **海量运算触手可得**：用户不必关心数据规模增长带来的存储困难、运算时间延长等烦恼，根据用户的数据规模自动扩展集群的存储和计算能力，使用户专心于数据分析和挖掘，最大化发挥数据的价值。
- **服务“开箱即用”**：用户不必关心集群的搭建、配置和运维工作，仅需简单的几步操作，用户便在MaxCompute中上传数据、分析数据并得到分析结果。
- **数据存储安全可靠**：采用多副本技术、读写请求鉴权、应用沙箱、系统沙箱等多层次数据存储和访问安全机制来保护用户的数据，使其不丢失、不泄露、不被窃取。
- **多用户协作的多租户机制**：通过配置不同的数据访问策略，用户可以让组织中的多名数据分析师协同工作，并且每人仅能访问自己权限许可内的数据，在保障数据安全的前提下最大化提高工作效率。
- **支持多Region部署**：能够指定计算集群，可以更合理的利用计算资源；同时，由于集群间数据交互在MaxCompute服务内部进行，集群间数据复制与同步按照配置管理，不再涉及跨域操作，大幅度节省数据处理等待时间。

32.3 产品架构

MaxCompute的功能架构如图 *32-1: MaxCompute功能架构图*所示：

图 32-1: MaxCompute功能架构图



MaxCompute由四部分组成，分别是**客户端**、**接入层**、**逻辑层**及**计算层**，每一层均可平行扩展。

MaxCompute的客户端有以下几种形式：

- **API**：以RESTful API的方式提供离线数据处理服务。
- **SDK**：对RESTful API的封装，目前有Java等版本的实现。
- **CLT**（**Command Line Tool**）：运行在Window/Linux下的客户端工具，通过CLT可以提交命令完成Project管理、DDL、DML等操作。
- **DataWorks**：提供了上层可视化ETL/BI工具，用户可以基于DataWorks完成数据同步、任务调度、报表生成等常见操作。

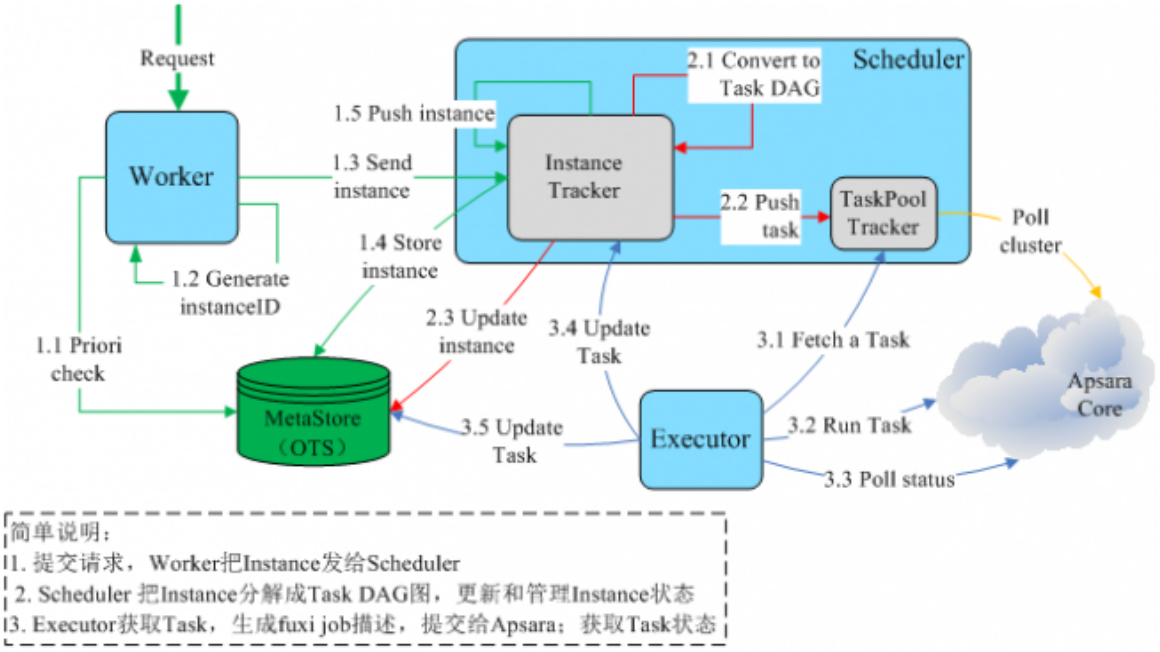
MaxCompute接入层提供HTTP（HTTPS）服务、Load Balance、用户认证和服务层面的访问控制。

MaxCompute逻辑层是核心部分，实现用户空间和对象的管理、命令的解析与执行逻辑、数据对象的访问控制与授权等功能。逻辑层包括两个集群：调度集群与计算集群。调度集群主要负责用户空间和对象的管理、Query和命令的解析与启动、数据对象的访问控制与授权等功能；计算集群主要负责task的执行。控制集群和计算集群均可根据规模平行扩展。在调度集群中有Worker、Scheduler和Executor三个角色，其中：

- **Worker处理所有RESTful请求**：包括用户空间（project）管理操作、资源（resource）管理操作、作业管理等，对于SQL、MapReduce、Graph等启动Fuxi任务的作业，会提交Scheduler进一步处理。
- **Scheduler负责instance的调度**：包括将instance分解为task、对等待提交的task进行排序、以及向计算集群的Fuxi master询问资源占用情况以进行流控（Fuxi slot满的时候，停止响应Executor的task申请）。
- **Executor负责启动SQL/ MR task**：向计算集群的Fuxi master提交Fuxi任务，并监控这些任务的运行。

简单的说，当用户提交一个作业请求时，接入层的Web服务器查询获取已注册的Worker的IP地址，并随机选择某些Worker发送API请求。Worker将请求发送给Scheduler，由其负责调度和流控。Executor会主动轮询Scheduler的队列，若资源满足条件，则开始执行任务，并将任务执行状态反馈给Scheduler。其对应的业务执行逻辑图如下图所示：

图 32-2: 业务执行逻辑图



MaxCompute存储与计算层为阿里云自主知识产权的云计算平台的核心构件，是飞天操作系统内核，运行在和控制集群独立的计算集群上。上面的MaxCompute架构图中仅列出了若干飞天内核主要模块，例如：飞天分布式文件系统、任务调度系统等。

其中，**飞天分布式文件系统**是一个分布式文件系统，其设计目标是将大量通用机器的存储资源聚合在一起，为用户提供大规模和可靠的分布式存储服务，是飞天操作系统内核的重要组成部分。

飞天分布式文件系统目前包括三台master和多台chunkserver，前者负责文件meta信息的存储和管理，后者负责数据存储。为了保证可靠性，同一个数据块会被存储在多个chunkserver上，一般情况下飞天分布式文件系统保存的数据会有3份副本，所有的MaxCompute数据文件都保存在飞天分布式文件系统上，在 /product/aliyun/odps飞天分布式文件系统目录下可以找到。需要注意，其中master是热备，同一时间只有一台在工作。

- **master**：
  1. PanguMaster维护了整个文件系统的元数据，其中包括命名空间、文件到数据块的映射及数据块的存储地址等。
  2. 作为整个分布式文件存储系统的大脑，PanguMaster控制着系统层面的活动如孤立数据块的垃圾回收、chunckserver间的数据合并、判断chunkserver是否健康、恢复由于chunkserver宕机所造成的数据块丢失等。
  3. PanguMaster同时还负责调节同一时间片中多台客户端对数据的访问来维护同一集群中数据的完整性。
  4. PanguMaster只对客户端提供元数据相关的操作，而数据传输相关的通讯都直接在chunkservers间进行。
- **chunkserver**：飞天分布式文件系统上的文件会被切成多个固定大小的存储单元，每个存储单元都叫一个chunk。存储数据块chunk的机器称为chunkserver，对每个数据块，创建之初，pangumaster都会分配一个128位的ID，飞天分布式文件系统客户端根据ID来读取存储在磁盘上的数据块。

同时，为了更好地支持MaxCompute表中的结构化数据，MaxCompute使用统一的数据文件格式，实现了一种特殊格式的飞天分布式文件系统文件，称为CFile。

- CFile是一种基于列存储的文件格式，其主要目的是为了降低离线数据处理过程中的无效磁盘读取操作。文件中的数据以列为单位聚簇组织（聚簇被称为Block），并在存储到文件系统之前进行压缩，减少了存储空间的占用。在离线数据处理的场景中，用户只需要读取待处理的数据，避免了无用的磁盘操作，提高读取的磁盘效率，同时减少了网络带宽的使用。

- CFile文件的存储结构逻辑上可以分为三个区域，分别为数据区（Data区）、索引区（Index区）以及元信息区（Header区）。其中，数据区存储的是按照列划分，以Block为组织单位的用户数据；索引区存储的是每列的数据Block对应的索引，其中包含每个Block在文件中的起始位置、Block压缩后的长度以及Block内的数据个数（对非定长的数据类型如string而言）。元信息区存储了该文件中每一列的元信息，例如该列的索引在文件中的起始位置以及索引长度、该列的类型信息、压缩方法等，以及文件中用户数据的行数以及版本号等。
- 目前支持如下五种原始的数据类型（即MaxCompute支持该五种类型）：
  # BIGINT，8字节有符号整型。
  # BOOLEAN，布尔型，包括TRUE/FALSE。
  # DOUBLE，8字节双精度浮点数。
  # STRING，字符串，需要注意在MaxCompute中的函数会假设STRING中存的是UTF8编码的字符串，其他编码格式可能会导致异常。
  # DATETIME日期类型，格式为**YYYY-MM-DD HH:mm:SS**，例如：2012-01-02 10:09:25。

而**任务调度系统**则是飞天操作系统平台内核中负责资源管理和任务调度的模块，也为应用开发提供了一套编程基础框架。其主要功能是充分利用整个集群的硬件资源来服务用户和系统的计算需求。任务调度系统同时支持两种应用类型的计算：低延迟的在线服务和高吞吐的离线处理，分别称为Fuxi Service和Fuxi Job，可以对应Hadoop中的Yarn。

- **Fuxi Service**：Service是任务调度系统上的常驻进程，由用户申请创建及销毁，任务调度系统不会主动销毁Service进程。
- **Fuxi Job**：Job是任务调度系统上的临时任务，一旦任务结束，资源就会被释放，由任务调度系统回收。

在资源管理上，任务调度系统负责调度和分配集群的存储、计算等资源给上层应用，支持计算资源额度、访问控制和作业优先级，保证有效的资源共享。在任务调度上，任务调度系统提供了数据驱动的多级流水线并行计算框架，类似于MapReduce编程模式，适用于海量数据处理和大规模计算等复杂应用。

任务调度系统目前包括两台master和多台tubo，其中master是冷备，同时只有一台在工作。而每台计算节点上都启动了一个tubo进程，用来管理单机资源，例如汇总整机的可用的CPU、内存、硬盘、网络，同时记录每台机器已被占用的资源，每个机器上的tubo进程会将这些资源汇报给FuxiMaster，由FuxiMaster统一管理及调度。

32.4 产品价值

MaxCompute与传统数据库相比，具有如下的价值点：

表 32-1: 价值点对比

价值点	传统数据库	MaxCompute
系统扩展能力	共享磁盘技术，难以超过100节点，分库分表技术将导致应用数据碰撞、比对等大规模计算带来海量数据的数据交换开销，极大限制应用的分析能力。	超过10000节点，支持超过1.5EB数据量。例如阿里双11活动中，6小时内MaxCompute处理数据量超过300PB。
数据类型支持	不适用于非结构化计算。	同时支持结构化、非结构化数据处理。
高可用性	无冗余存储，传统备份恢复对PB级数据量不可用，单点磁盘故障会导致全库不可用。	无共享的多副本技术，无单点故障。
复杂计算能力	不支持迭代计算，不支持图计算。共享磁盘技术，复杂计算导致节点间大量数据交换，带宽难以支持。	分布式存储，支持MR、SQL、迭代计算、MPI、图计算等多种计算框架。
并发能力	一个大规模计算任务可能会消耗掉全部系统资源（如索引计算），存在网络、热点盘（数据字典）等多种瓶颈，无法支持高并发访问。	具有完善的多租户资源隔离和资源管理工具，用户对集群资源一目了然，并且很方便的管理每个业务使用的资源量，可以支持超过1万的并发访问。
性能支持	索引机制导致实时入库数据难以支持分析型应用，海量数据碰撞比对、分析预测计算时间可能超过24小时，性能无法满足需求。	关注与海量数据的并行计算能力，支持数据实时入库可用、具备高性能大规模离线计算、海量数据实时多维分析、流计算等多种高性能计算能力。

32.5 功能描述

32.5.1 Tunnel

Tunnel是MaxCompute提供的数据通道服务，各种异构数据源都可通过Tunnel服务导入MaxCompute或从MaxCompute导出。它是MaxCompute数据对外的统一通道，提供高吞吐、持续稳定的服务。

Tunnel提供了Restful API接口，提供了Java SDK，可以方便用户编程。

32.5.2 SQL

MaxCompute SQL是一种结构化查询语言，语法和Oracle/MySQL/Hive SQL类似，可以看作是标准SQL的子集，但不能因此简单的把MaxCompute SQL等价成一个数据库，它在很多方面并不具备数据库的特征，如事务、主键约束、索引等。

MaxCompute SQL适用于海量数据（TB级别），实时性要求不高的场合，它的每个作业的准备、提交等阶段要花费较长时间，因此要求每秒处理几千至数万笔事务的业务是不能用MaxCompute SQL完成的。

32.5.3 MapReduce

MapReduce是一种编程模型，基本等同Hadoop中的MapReduce。用于大规模数据集（TB级别）的并行运算MaxCompute。

用户可以使用MapReduce提供的接口（Java API）编写MapReduce程序处理MaxCompute中的数据。概念“Map（映射）”和“Reduce（归约）”和它们的主要思想，都是从函数式编程语言和矢量编程语言中借鉴来的特性。它极大地方便了编程人员在不会分布式并行编程的情况下，将自己的程序运行在分布式系统上。

当前的软件实现是指定一个Map（映射）函数，用来把一组键值对映射成一组新的键值对，指定并发的Reduce（归约）函数，用来保证所有映射的键值对中的每一个共享相同的键组。

MaxCompute MapReduce特性：

- Hadoop-style，针对MaxCompute场景设计（用于处理Table和Volume）。
- 输入输出仅支持MaxCompute内置类型。
- 可以输入多表，输出多表或到不同分区。
- 可以读资源（Resource）。
- 不支持输入view。
- 受限的沙箱安全环境。

32.5.4 Graph

Graph是MaxCompute提供的面向迭代的图计算处理框架，为用户提供类似Pregel的编程接口，用户可以基于Graph框架开发高效的机器学习或数据挖掘算法。

在互联网环境下，存在很多海量图结构的数据，例如社交网络、物流信息等，这类图计算模型的典型特点是迭代，整个计算过程是通过一轮一轮反复迭代求解，最后达到一个收敛状态。例如对于需要迭代学习模型参数的机器学习算法而言，图计算模型比MapReduce有天然优势。在实际应用中，用户将问题抽象成图，然后以顶点为中心，通过超步进行迭代更新。

MaxCompute Graph目前提供两种模式：

- 离线模式：适用于计算规模较大的场景，类似于MapReduce作业，每次运行完成加载和计算两个过程。
- 交互模式：适用于计算规模较小的场景，用户实现UDF，然后通过命令行方式交互。

在离线模式下，加载和计算是两个独立的步骤，数据加载后会常驻内存，用户可以对数据执行不同的计算逻辑。例如风控部门每天会加载一次数据，运营人员会对这份数据执行不同的查询逻辑，查看数据之间的关系。

在阿里巴巴内部，MaxCompute Graph已经有很多应用，例如实现带权重的PageRank算法计算支付宝用户身边影响力指数；实现变分贝叶斯EM模型，基于用户购买的商品属性信息，推测用户的汽车品牌分布等。

32.5.5 非结构化数据的访问及处理

MaxCompute团队依托MaxCompute系统架构，引入非结构化数据处理框架，解决MaxCompute SQL面对MaxCompute表外的各种用户数据时（例如：OSS上的非结构化数据或者来自TableStore的非结构化数据），需要首先通过各种工具导入MaxCompute表，才能在其上面进行计算，无法直接处理的问题。

用户只需要通过一条简单的DDL语句，在MaxCompute上创建一张外部表，建立MaxCompute表与外部数据源的关联，即可以为各种数据在MaxCompute上的计算处理提供入口。并且创建好的外部表可以像普通的MaxCompute表一样使用，充分利用MaxCompute SQL的强大计算功能。

32.5.6 增强功能

32.5.6.1 Spark on MaxCompute

32.5.6.1.1 开源平台——Cupid

MaxCompute是阿里云自主研发的大数据解决方案，其规模和稳定性在业界都是领先的。大数据开源社区当前也非常活跃，应对各类需求的系统快速出现并发展。为了更好服务用户，同时也为了丰

富MaxCompute的生态，MaxCompute团队研发了Cupid平台，把MaxCompute跟开源社区连接到一起，兼顾开源社区的多样性和飞天操作系统的规模和稳定性。

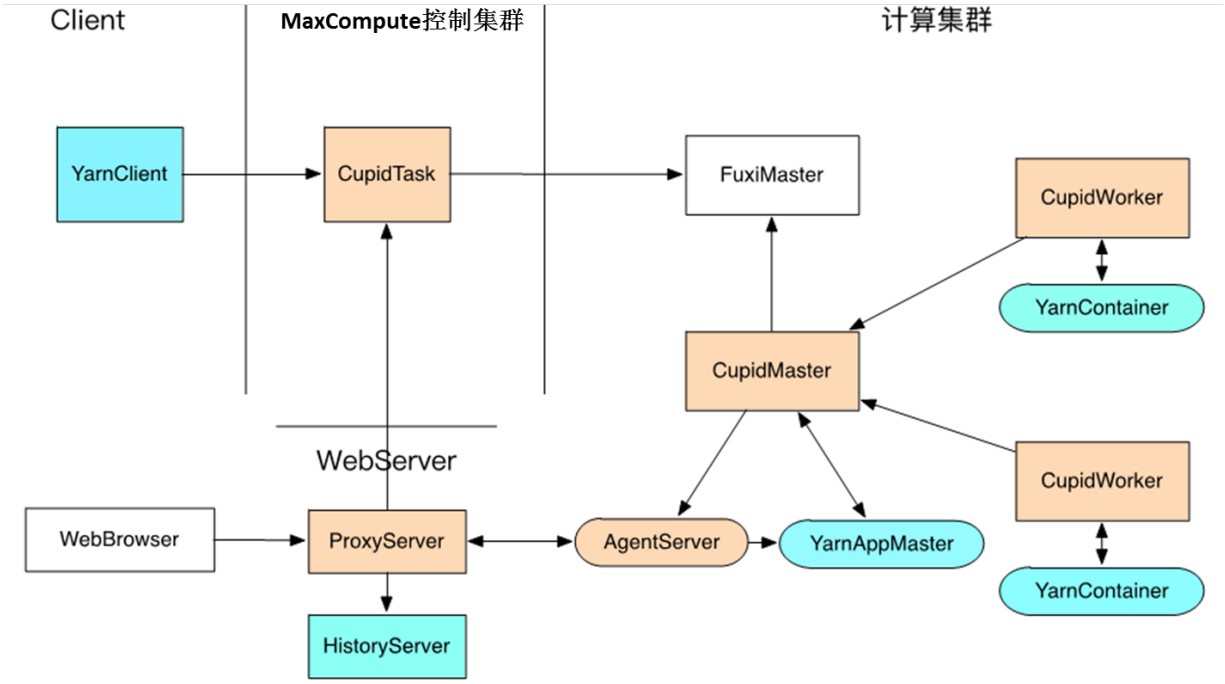
开源社区和飞天操作系统的软件栈比较相似，但也有不同之处。

对于分布式文件系统，开源社区大多使用HDFS，飞天操作系统里面则是使用飞天分布式文件系统；对于分布式调度系统，开源社区用的比较多的是Yarn，飞天操作系统里面则是使用任务调度系统；在此之上是应对各类场景的计算引擎。为了能够把开源应用（例如Spark）运行在MaxCompute系统中，Cupid首先要做兼容。

32.5.6.1.1.1 兼容Yarn

Yarn面向应用的接口主要有三个：YarnClient、AMRMClient和NMClient。YarnClient用于客户在应用端提交作业到集群；AMRMClient用于AppMaster向ResourceManager申请和释放资源；NMClient用于启动和停止应用的Container。

图 32-3: Yarn on MaxCompute



一个Yarn上App执行到MaxCompute平台的流程如上图所示，其中黄色部分是Cupid平台的组件，浅蓝色部分是开源组件。具体流程如下：

1. 用户通过YarnClient访问MaxCompute控制集群，提交作业给FuxiMaster。
2. FuxiMaster启动一个CupidMaster，CupidMaster按照Yarn的协议把YarnAppMaster启动起来。
3. YarnAppMaster通过CupidMaster跟FuxiMaster交互，申请和释放资源。

4. 启动新的Container时，任务调度系统的tubo会先启动一个CupidWorker，CupidWork根据Yarn的协议启动Container。

> **说明：**
> YarnAppMaster中一般都会有UI，Cupid通过代理机制实现。

32.5.6.1.1.2 兼容FileSystem

开源社区里面大多使用HDFS作为分布式存储，Hadoop社区提供了FileSystem接口，目前阿里云的OSS，亚马逊的s3都兼容了这个接口。MaxCompute团队同样实现了飞天分布式文件系统兼容FileSystem接口，提交到MaxCompute集群的开源job同样可以原封不动运行起来，并且获得原生飞天分布式文件系统的性能。

> **说明：**
> 需要注意，飞天分布式文件系统不直接对外提供服务，飞天分布式文件系统上面的数据只能作为作业中间数据，例如checkpoint之类的。如果用户想要使存入的数据在其他环境同样能够访问，目前可以选择使用OSS。

32.5.6.1.1.3 DiskDrive

开源应用大多会使用本地文件系统来进行数据处理，例如spark shuffle、storage等。在大集群环境下，磁盘是一种非常重要的系统资源，应该被统一管理，才能保证高可用、高性能以及安全性。在飞天操作系统中，磁盘都是被飞天分布式文件系统统一管理。为了在飞天分布式文件系统的基础上提供本地文件系统接口，Cupid把网盘引入到MaxCompute，设计并实现了DiskDriverService系统。

32.5.6.1.2 功能扩展

由于MaxCompute提供了Cupid框架来支持开源应用，因此，Spark可以在MaxCompute上面运行起来。为了更好的方便用户使用，也为了更好地跟MaxCompute融合，Spark on MaxCompute需要对Spark进行功能扩展，主要包括：保证Spark开源程序的安全隔离；跟MaxCompute数据打通；在多租户的集群中提供交互式。

32.5.6.1.2.1 安全隔离

Spark作业提交到MaxCompute计算集群中，为防止用户恶意代码对系统进行攻击，Spark作业需要运行在安全沙箱里面。整体使用父子进程架构，Cupid框架代码运行在父进程中，Spark代码运行在子进程中。Spark需要访问系统服务时，可以通过父子进程通信的方式，由父进程代理访问。

32.5.6.1.2.2 数据互通

运行在MaxCompute中的Spark的一个优势是，Spark作业和MaxCompute作业在整个集群资源上是统一的，可以做到相互之间数据直接访问，而不需要跨级群拖数据。

数据互通包括两个方面：元数据和存储数据，并且Spark存储MaxCompute数据必须通过MaxCompute账号体系进行鉴权。**Spark on MaxCompute**实现了OdpsRDD、`OdpsDataFrame`，可以满足用户对Spark这两种API的需求。同时Spark SQL可以直接访问MaxCompute元数据进行SQL优化，并且在物理层直接存取MaxCompute数据格式。

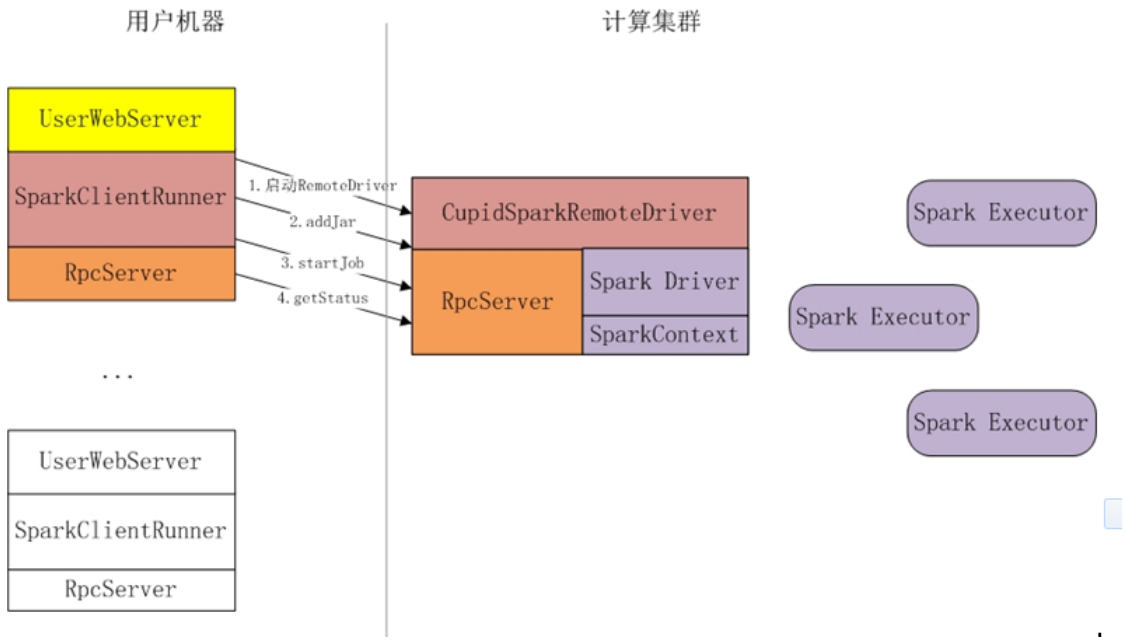
32.5.6.1.2.3 Client模式

社区spark生产主要使用**`yarn-cluster`**、**`yarn-client`**两种模式。**`yarn-cluster`**模式将spark作业提交到集群运行，运行完毕后客户端打印状态日志。这种模式无法向一个Spark Session动态多次提交作业，且客户端无法获取每个job的状态及结果；而**`yarn-client`**模式，主要解决spark交互式场景问题，需要在客户端机器启动`Driver`，无法将Spark Session作为一个服务。因此MaxCompute团队基于**Spark on MaxCompute**开发了**`Client`**模式来解决上述的问题。该模式具有以下特点：

1. 客户端轻量级，不用再启动spark的Driver。
2. 客户端有一套API向MaxCompute集群的同一个Spark Session动态提交作业，并监控状态。
3. 客户端可以通过监控作业状态及结果，构建作业之间的依赖关系。
4. 用户可以动态编译应用程序jar，通过客户端提交到原有的Spark Session运行。
5. 客户端可以集成在业务的WebServer中，并且可以进行水平扩展。

`Client`模式会先通过CupidSparkClientRunner在MaxCompute集群启动一个Spark Session。后续通过CupidSparkClientRunner在客户端提交作业、获取作业状态及运行结果，同时各个作业之间可以共享cache的数据等；也可以构建多个CupidSparkClientRunner与同一个Spark Session交互，运行时的框架图如下图所示。

图 32-4: Spark Client模式



具体使用Spark Client模式的执行流程如下所示：

1. 向MaxCompute集群提交启动CupidSparkRemoteDriver，并获取SparkClientRunner对象。
2. 通过SparkClientRunner添加要执行作业的jar包到RemoteDriver。
3. SparkClientRunner使用job的classname向RemoteDriver提交运行。
4. SparkClientRunner通过提交作业返回的job id来监控作业的状态。

32.5.6.1.2.4 Spark生态支持

Spark拥有丰富的生态：Mllib、Streaming、PySpark、SparkR、Graphx、SQL。**Spark on MaxCompute**对Spark的完整生态提供支持，使用方式跟开源社区保持一致。此外，**Spark on MaxCompute**也支持Spark UI以及历史日志的访问。

32.5.6.2 ElasticSearch on MaxCompute

32.5.6.2.1 概念简介

索引词（**term**）：索引词（term）是一个能够被索引的精确值，可以通过term查询进行准确的搜索。

文本（**text**）：文本是一段普通的非结构化文字。通常，文本会被分析成一个个的索引词，存储在ElasticSearch的索引库中。

集群（**cluster**）：集群由一个或者多个节点组成，对外提供索引和搜索服务。ElasticSearch部署在MaxCompute的飞天集群中，ElasticSearch集群是飞天集群的一部分。

节点（**Node**）：一个节点是一个逻辑上的堵路服务，是ElasticSearch集群的一部分，可以存储数据并参与集群的索引和搜索功能。

分片（**shard**）：分片是单个Lucene实例，是Elasticsearch管理的比较底层的功能。ElasticSearch集群会自动管理集群中的所有分片，在某一个节点发生故障的时候，ElasticSearch会把分片移动到不同的节点或者添加新的节点。

复制（**Replica**）：复制（replica）是Elasticsearch的备份概念，ElasticSearch on MaxCompute已经提供了多副本功能，有效提高了系统级别可用性，因此复制（replica）在本产品中建议默认为1。

索引（**Index**）：索引是具有相同结构的文档集合。例如，可以有一个客户信息的索引，包括一个产品目录的索引、一个订单数据的索引。在系统上索引的名字全部小写，通过这个索引可以执行索引、搜索、更新和删除操作。在单ElasticSearch集群中，可以定义多个索引。

类型（**type**）：在索引中可以定一个或者多个类型，类型是索引的逻辑分区。一般情况下，一个类型定义为具有一组公共字段的文档。

映射（**Mapping**）：映射就像关系型数据中的表结构，每个索引都有一个映射，它定义了索引中每一个字段类型，以及一个索引范围内的设置。一个映射可以事先被定义，或者在第一次存储文档的时候自动识别。

文档（**document**）：文档是存储在ElasticSearch中的一个Json格式的字符串，就像关系数据库中标的一行。每个文档都有一个类型和一个ID，每个文档就是一个Json对象，存储了零个或者多个字段，即键值对。

字段：文档中包含零个或者多个字段，字段可以是一个简单的值，也可以是一个复杂的对象嵌套结构。字段类似于关系数据库表中的列，每个字段都有一个字段类型。

32.5.6.2.2 分布式架构

32.5.6.2.2.1 基本原理

ElasticSearch集群由多个节点组成， MaxCompute平台统一控制ElasticSearch节点及整个服务的启动、停止以及对计算资源进行分配，采用统一的调度机制进行failover来提供高可用性。

数据以多副本的形式存储在飞天分布式文件系统上，在部分机器宕机的情况下，飞天分布式文件系统能够做到数据高可用性，不会丢失数据。

索引被拆分成多个分片，每个分片均匀分布在集群的多个节点上。因此，通过多个节点上的分片并行进行搜索，提高了搜索性能。

32.5.6.2.2.2 功能特点

分布式架构的功能特点如下。

- 分布式架构，可弹性伸缩。
- 分布式搜索，提高搜索性能。
- 数据多副本存储，提高可靠性。
- 完整的计算和存储资源管理机制。
- 从节点到集群服务多层级failover机制。

32.5.6.2.3 全文检索

32.5.6.2.3.1 基本原理

全文检索，是在大量的文本内容中检索到包含指定的内容的记录。为了快速检索到指定记录，需要将文本数据分词后，以词->文档的方式建立倒排索引，以达到快速以词查找文档的效果。

32.5.6.2.3.2 功能特点

全文检索的功能特点如下。

- 一次索引，多次使用。
- 可灵活配置搜索规则，按字段控制索引方式和分词方式。
- 可动态修改索引结构，通过索引重建快速完成数据的重新索引和分布。

32.5.6.2.4 权鉴控制

32.5.6.2.4.1 基本原理

在对外服务的入口处，增加鉴权控制，检查用户有没有索引库的访问和使用权限。

32.5.6.2.4.2 功能特点

启用鉴权控制机制，可以保证数据的安全，以避免出现人为恶意造成数据危险操作。

32.5.7 MaxCompute多Region部署

MaxCompute支持多Region部署。其中，控制集群中心化部署，主要是进行资源的配置，计算任务的管理；计算集群每个Region独立部署，主要实现项目创建及计算任务的下发。

MaxCompute多Region部署的特点如下：

- 同一个MaxCompute服务实例可以管理异地不同的集群。
- 集群间数据交互在MaxCompute服务内部进行，集群间数据复制与同步按照配置管理。
- 元数据统一存放，对不同机房间网络等基础设施条件要求较高。
- 账号系统统一。
- 大数据应用开发系统，如Dataworks统一接入。
- 需要改造目前MaxCompute集群为多集群模式，才能进行多Region部署。

  说明：

  改造条件及限制需满足以下要求。

  # 网络带宽需要满足region间数据同步要求、链路冗余。
  # 中心region控制集群对于基础服务，例如DNS、TableStore的延迟比较高，最好在同一个机房内，网络延迟在5ms以内。
  # 中心region控制集群与其他region计算集群间，要求网络延迟在20ms以内。
  # 两地集群间及集群内机器间，要求时钟一致。
  # 网络带宽需要满足集群间数据复制要求。
  # 需要一套DNS服务。
  # 需要集群间机器网络打通，两地集群的网络条件基本一致（千兆/万兆）。
- 运维、部署及升级的相关支持上与单集群方式不同，对现场运维能力要求较高。

32.6 应用场景

MaxCompute主要面向三类大数据处理场景：

- 基于SQL构建大规模数据仓库和企业BI系统。
- 基于MapReduce和MPI的分布式编程模型开发大数据应用。
- 基于统计和机器学习算法，开发大数据统计模型和数据挖掘。

32.6.1 搭建数据仓库

图 32-5: 搭建数仓

使用MaxCompute可以轻松打造一个云端数据仓库，借助MaxCompute的分区、数据表统计、表的生命周期等功能，用户可以很方便的实现数据仓库的历史信息增强存储、冷热表区分、数据质量控制等场景。

阿里金融数仓团队基于MaxCompute构建了一个完善复杂、功能强大的数据仓库体系，包含六个层次：源数据层、ODS层、企业数据仓库层、通用维度模型层、应用集市层和展现层。

- 源数据层处理各个来源数据，包括淘宝、支付宝、B2B、外部数据等。
- ODS作为数据导入的临时存储层。
- 企业数据仓库层采用3NF建模方式，按主题（如商品、店铺）进行划分，包括完整的历史数据。
- 通用维度模型以维度建模方式构建面向通用业务应用的模型层，不以满足特定的应用为目的。通用维度模型层的目的是屏蔽业务需求变化，以一致性维度和事实的方式为上层提供数据。
- 应用集市层是面向需求，构建满足某一应用需求的数据集市。
- 展现层提供一些数据门户（Portal）和服务等，供应用访问。

在这个体系架构中，不可避免会涉及元数据管理等其他方面。

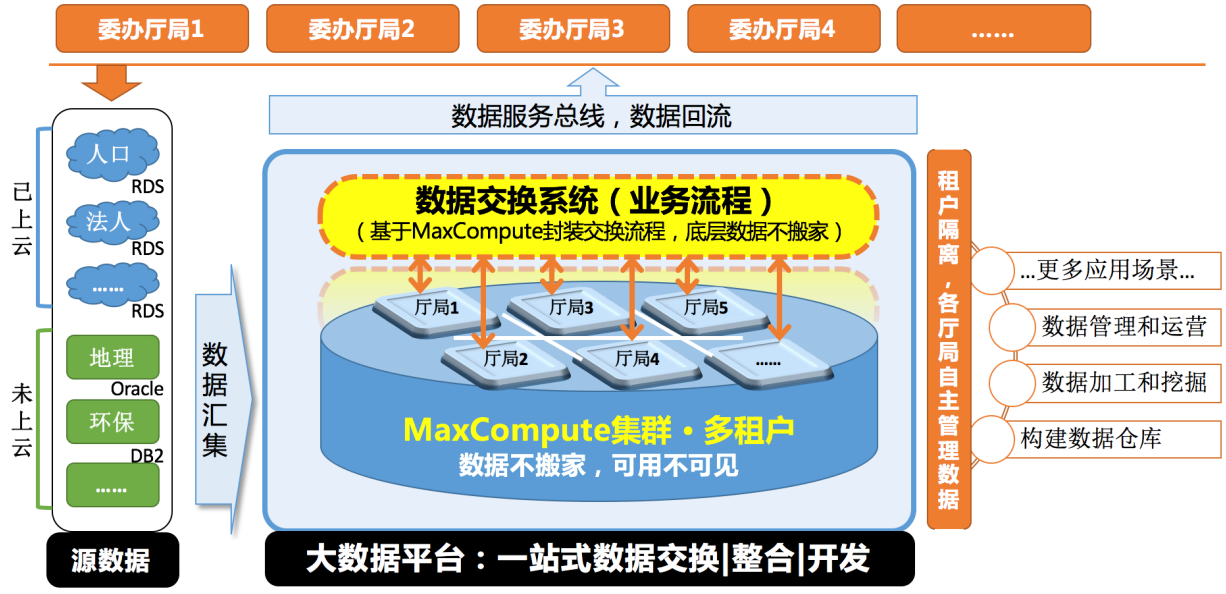
金融的数据仓库主要是基于MaxCompute SQL完成离线计算，并通过一系列指标规则和算法完成离线决策，输出结果给在线决策使用。

跟传统数据库相比，基于MaxCompute搭建数仓，有以下不同之处：

- **历史数据存储**：MaxCompute天生支持大数据，不必像传统数据库那样将历史数据转储到廉价存储媒介。
- **分区方式**：传统数据库支持的分区方式更丰富一些（例如支持范围分区），但在数仓场景下，MaxCompute目前支持的分区方式基本够用。不管用哪个数据库搭建数仓，表分区的设计理念和原则一致。
- **大宽表**：MaxCompute按字段存储，建大宽表有天然优势。
- **数据整合**：传统数据库都用存储过程来加工、整合数据，MaxCompute则需要将逻辑拆分成一段段SQL，虽然实现途径不同，但算法是一致的。经过几年的使用比较，采用分段SQL的方式更清晰和高效，而存储过程的方式更灵活且具备处理复杂逻辑的能力。

32.6.2 大数据共享及交换

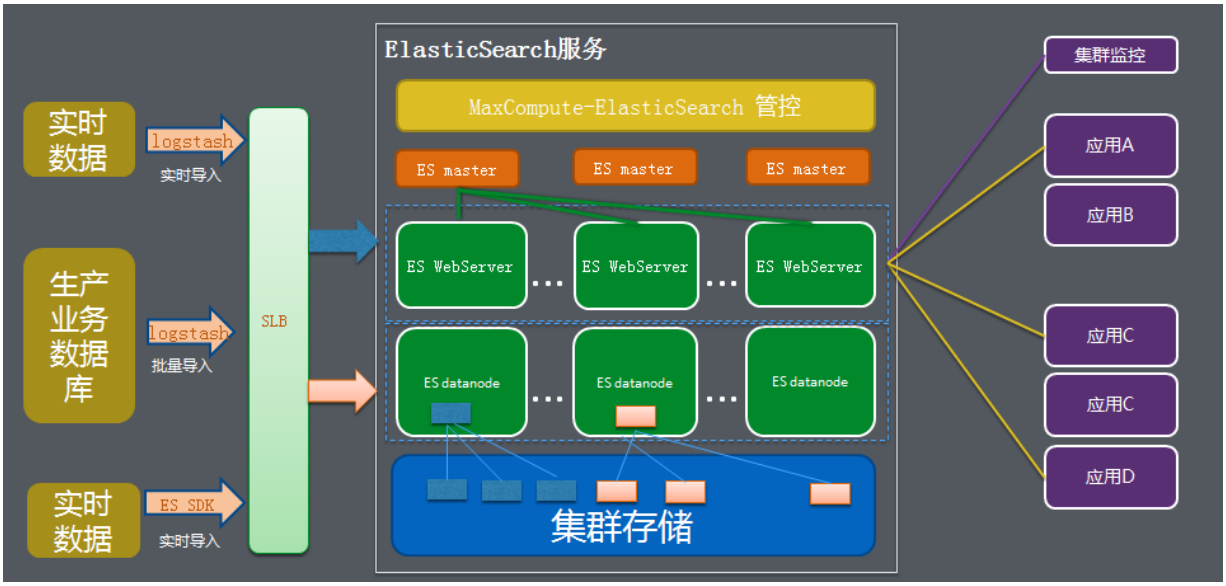
图 32-6: 大数据共享及交换



MaxCompute具有丰富的多种权限管理方式、灵活数据访问控制策略。MaxCompute提供丰富的授权管理手段，包括ACL、角色授权、 Policy授权、跨Project授权以及Label机制，可以提供精确到列级别的安全方案，满足一个组织或者跨组织间的授权需求。安全要求较高的项目，可以提供项目保护机制，防止数据泄露，而且对用户的任何操作都记录日志便于事后追溯审计。

32.6.3 ElasticSearch on MaxCompute典型实践

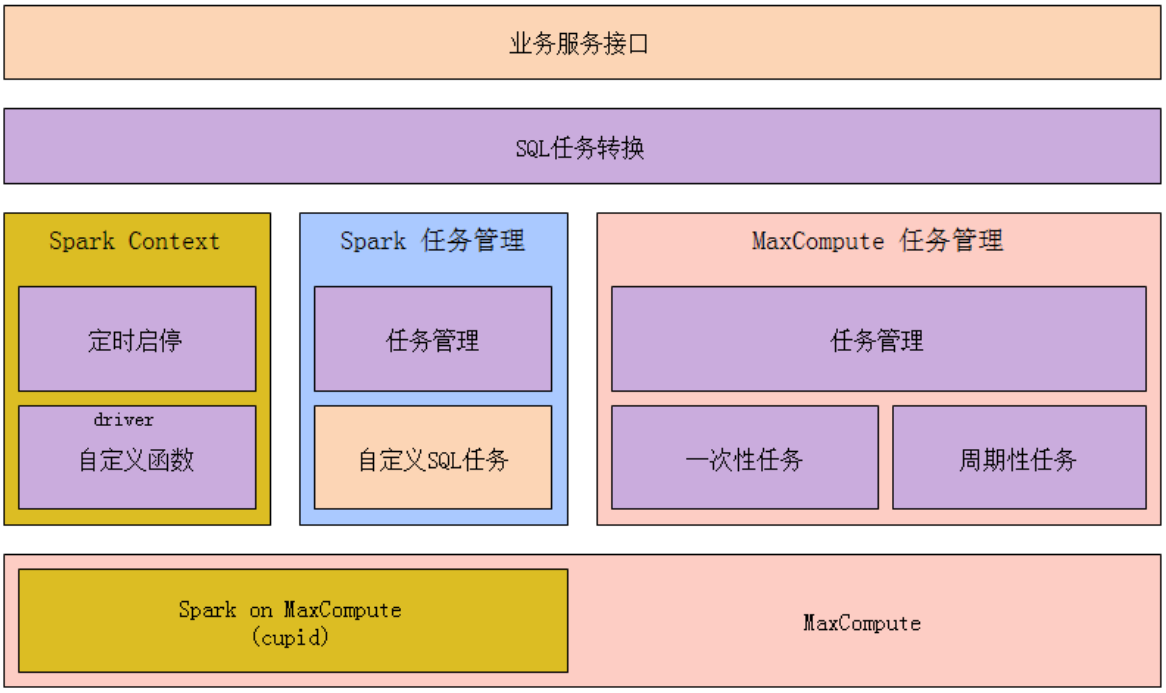
图 32-7: 典型实践



ElasticSearch on MaxCompute支持用户在MaxCompute集群上用提交作业的方式启动一套ElasticSearch服务。项目不修改原生ElasticSearch代码，且**ElasticSearch on MaxCompute**的运行模式与原生ElasticSearch集群一致。

32.6.4 Spark on MaxCompute典型实践

图 32-8: 典型实践



Spark on MaxCompute支持Client Mode的业务计算平台应用，应用框架如上图所示。

33 数据工场DataWorks

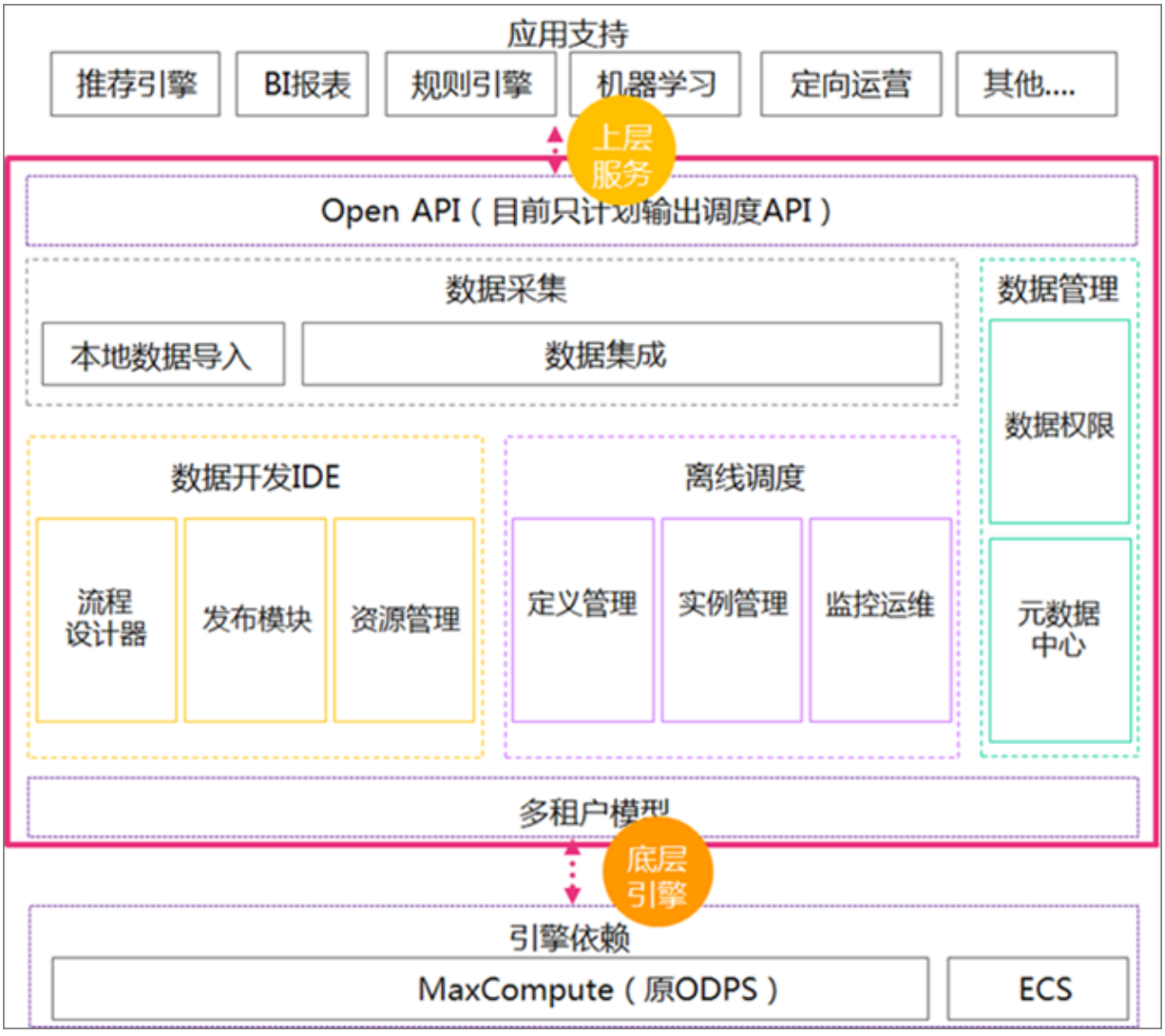
33.1 产品概述

DataWorks是阿里云推出的一款大数据领域平台级产品。面向企业和个人用户提供端到端的一站式大数据开发、管理、离线调度和应用解决方案。

DataWorks以让更多人使用更多数据为使命，提供DT时代的大数据基础服务。

- 让大型企业构建PB、EB级别的数据仓库，实现超大规模数据集成，对数据进行资产化管理，通过对数据价值的深度挖掘实现业务的数据化运营。
- 让中小企业和个人用户快速构建数据应用，助力中小企业的数据业务创新。

图 33-1: 产品组成



DataWorks产品由数据开发IDE、离线调度系统、数据集成工具和数据管理四大部分组成。

- **数据开发IDE**：提供开箱即用的一站式数据开发工具，满足在线SQL、MR、Shell的编码工作，并提供多人协同开发和代码版本管理功能。通过可视化的流程设计工具可以满足快速构建数仓调度的依赖关系。
- **离线调度系统**：提供百万级的离线任务调度能力，以及在线运维、在线日志查询、调度状态监控报警等一系列功能。
- **数据集成工具**：提供海量异构数据源的数据集成能力，打通阿里云80%的数据库及存储设备的数据链路，以及常用关系型数据库、FTP、HDFS等多种数据链路，并且提供周期性定时集成的能力。
- **数据管理系统**，提供对MaxCompute（原ODPS）中以公司为单位的全量数据的管理能力，并提供权限管理、数据血缘和元数据查看等功能。

33.2 产品特性和核心优势

超大规模数据处理能力

DataWorks与大数据计算服务MaxCompute（原ODPS）天然集成，单个集群的规模可达5000台，并且具备跨机房的线性扩展能力，轻松处理海量数据。离线调度支持百万级任务量，实时监控告警。

核心指标

- 万亿级数据JOIN，百万级并发job，作业I/O可达PB级/天。
- 具备跨集群（机房）数据共享能力，支持万级别的集群数，扩容不受限制。
- 提供功能强大易用的SQL、MR引擎，兼容大部分标准SQL语法。
- MaxCompute（原ODPS）采用三重备份、读写请求鉴权、应用沙箱、系统沙箱等多层次数据存储和访问安全机制来保护用户的数据，使其不丢失、不泄露、不被窃取。

一站式的数据开发环境

数据开发、离线调度、调度运维、监控告警、数据管理全流程串通。

核心指标

- 提供全流程所有功能。
- 提供可视化工作流程设计器功能，类似Kettle的工具，支持用户对流程进行设计并编辑。
- 多人协同作业机制，分角色进行任务开发、线上调度、运维、数据权限管理等功能，数据及任务无需落地即可完成复杂的操作流程。

海量异构数据源快速集成能力

提供11种异构数据源的数据读取能力，12种异构数据源的数据写入能力。并且提供脏数据过滤，流量控制等功能。

核心指标

- 提供mysql、oracle、sqlserver、postgresql、rds、drds、MaxCompute、ftp、oss、hdfs、dm、sysbase的数据读取能力。
- 提供mysql、oracle、sqlserver、postgresql、rds、drds、MaxCompute、ads、ocs、oss、hdfs、dm、sysbase的数据写入能力。
- 提供脏数据过滤、流量控制能功能。
- 可以周期性调度，周期性数据集成能力。

Web化的软件服务

可在互联网/内部网络环境下直接使用，无需安装部署，拎包入驻，开箱即用。

多租户权限模型

多租户模型确保用户的数据被安全隔离，以租户为单位进行统一的权限管控、数据管理、调度资源管理和成员管理工作。

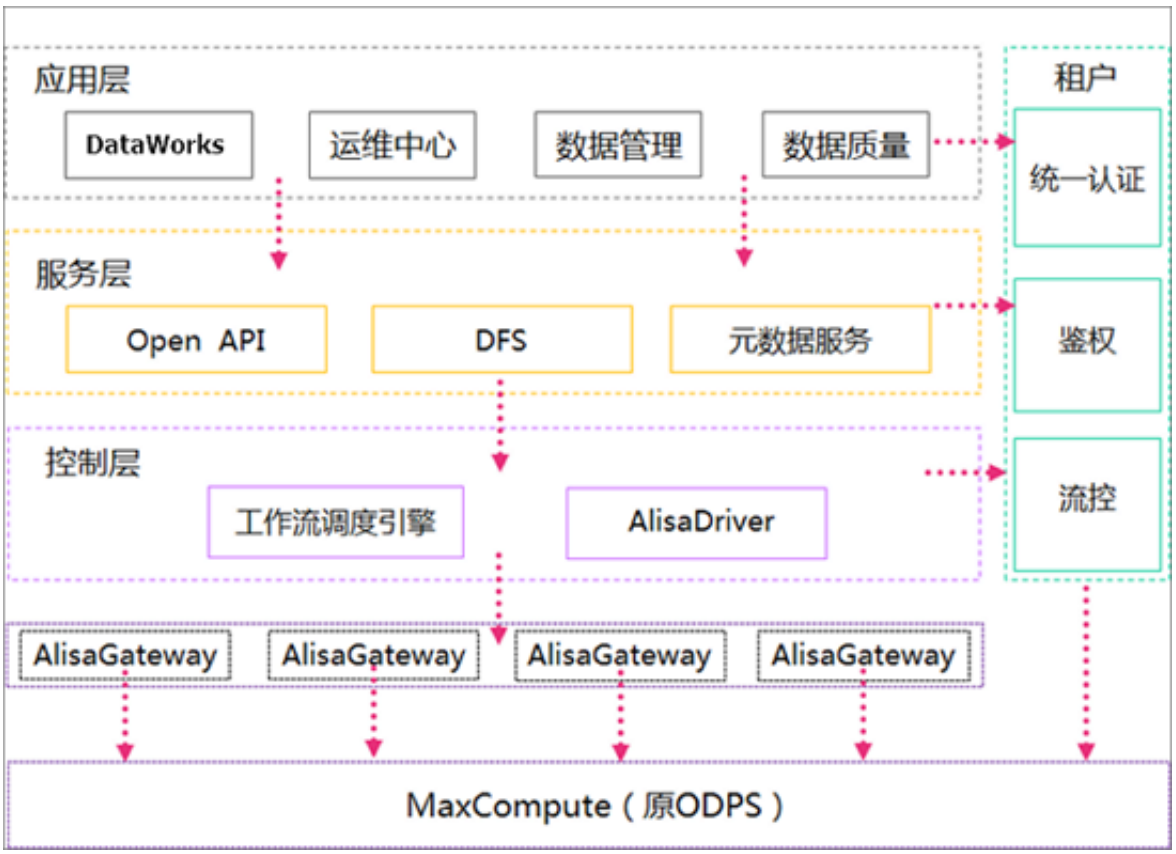
开放的平台

所有模块已实现组件化、服务化，您可基于DataWorks的Open API来定制开发扩展功能。

33.3 系统架构

系统开放性架构

图 33-2: 系统架构图



DataWorks采用组件化、服务化设计理念，分为以下三层。

- **控制层**：DataWorks离线加工的核心，工作流调度引擎承接着整个DataWorks的调度，包括：工作流的转实例、工作流调度，AlisaDriver主要协调、控制所有任务的执行。
- **服务层**：为应用层或外部其他应用提供服务。
- **应用层**：基于底层服务直接和用户进行交互，为您提供可视化操作的界面。

安全架构

DataWorks的安全架构，由平台自身的安全实现层、平台内置的安全服务层、租户可选的安全产品层构成。

- **平台自身的安全实现层**：保障平台在代码实现和部署配置时产品自身的安全性。
- **平台内置的安全服务层**：为租户和其用户提供平台基础性的安全服务能力，如租户资源隔离、身份认证、权限鉴别和日志合规审计等。

- **租户可选的安全产品层**：为租户和其用户提供可选的、已集成的安全产品或工具，帮助租户根据其自行定义的安全策略对其拥有的系统、数据进行安全防护和运维管理。

多租户模型

DataWorks拥有自己的多租户权限模型。

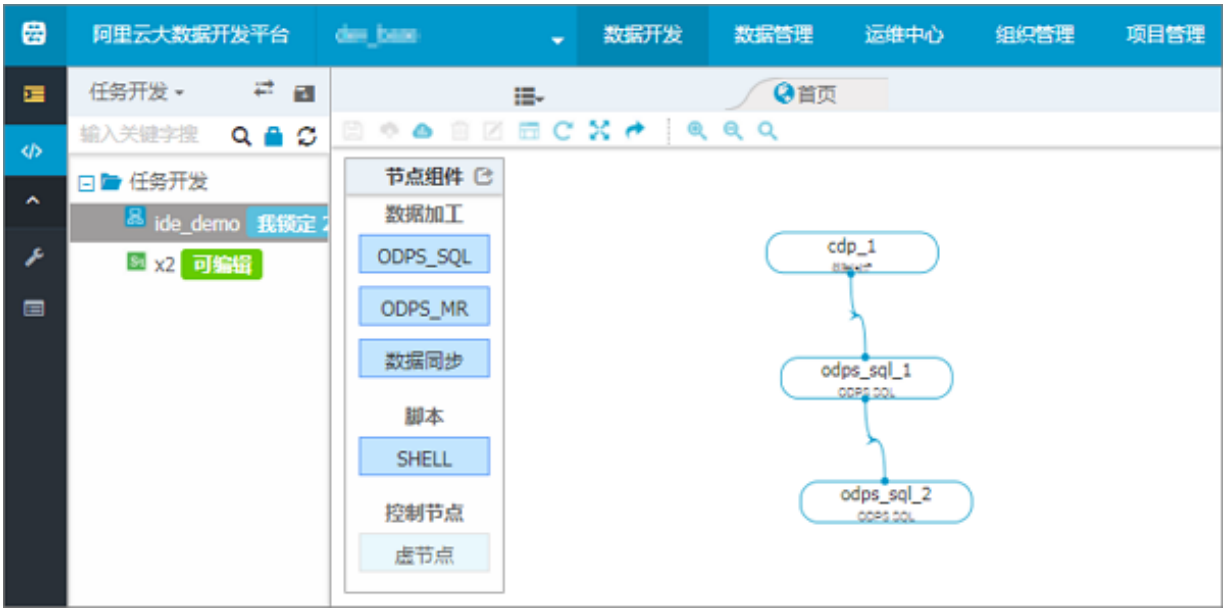
- 弹性的存储和计算资源：租户可按需申请资源配额，独立管理自己的资源。
- 租户独立管理自有的数据、权限、用户、角色，彼此隔离，以确保数据安全。

33.4 功能描述

33.4.1 数据开发IDE

DataWorks的数据开发IDE模块，提供一站式的集成开发环境，可满足大数据环境下的快速数仓建模、数据查询、ETL开发、算法开发等需求，并提供多人在线协同开发、文件版本控制等功能。

图 33-3: 数据开发



功能特性

- 提供可视化工作流程设计器功能，类似Kettle的工具，支持您对流程进行设计并编辑，对流程中的每一个任务节点进行相应的开发工作。
- 提供本地数据上传功能，支持本地文本数据快速上云。
- 提供海量异构数据源的数据快速集成能力。

> **说明：**
>
> 目前数据同步任务支持的数据源类型如下所示。
>
> # 取数据支持的数据源：mysql、oracle、sqlserver、postgresql、rds、drds、maxcompute、ftp、oss、hdfs、dm、sysbase。
> # 写数据支持的数据源：mysql、oracle、sqlserver、postgresql、rds、drds、maxcompute、ads、ocs、oss、hdfs、dm、sysbase。

- 提供Web IDE编程和调试环境，支持SQL、MR、SHELL（有限支持）、数据同步等多种程序类型。
- 跨项目发布：快速将任务及代码部署到其他项目的调度系统。
- 协同开发：代码版本管理，多人协同模式下的代码锁管理和冲突检测机制。
- 提供MaxCompute（原ODPS）表搜索、资源搜索引用、自定义函数搜索引用、数据查询功能，您可轻松索引数据。

33.4.2 数据管理

数据管理为您提供租户范围内数据表搜索、数据表详情查看、数据表权限管理、收藏数据表等功能。详细操作请参见《*DataWorks*用户指南》中的数据管理。

33.4.3 调度系统

离线调度系统为您提供百万量级任务的离线调度服务，并提供可视化运维页面、在线日志查询、监控告警等功能。详细操作请参见《*DataWorks*用户指南》。

功能特性：

- 调度系统可支撑的job数量达到百万级。
- 执行框架采用分布式架构，并发作业数可线性扩展。
- 支持多时间粒度的调度周期：分钟、小时、日、周、月、年。
- 支持节点空跑、暂停、一次性运行等特殊状态控制。
- 可视化展示调度任务DAG图，极大地方便用户对线上任务进行运维管理。
- 支持实时任务运行状态监控告警功能，短信、邮件的告警方式。
- 支持单任务重跑、多任务重跑、结束进程、置成功、暂停等线上运维操作功能。
- 支持补数据（串行执行多周期实例）。

- 提供全局的任务统计信息汇总界面，任务统计内容包括：总调度任务数、出错调度任务数、运行调度任务数、计算资源消耗Top10调度任务、计算时间消耗Top10调度任务、任务类型分布等信息。

33.4.4 数据集成

提供多种异构数据源的快速集成服务，为跨平台的异构数据提供快速数据整合的能力。

功能特性

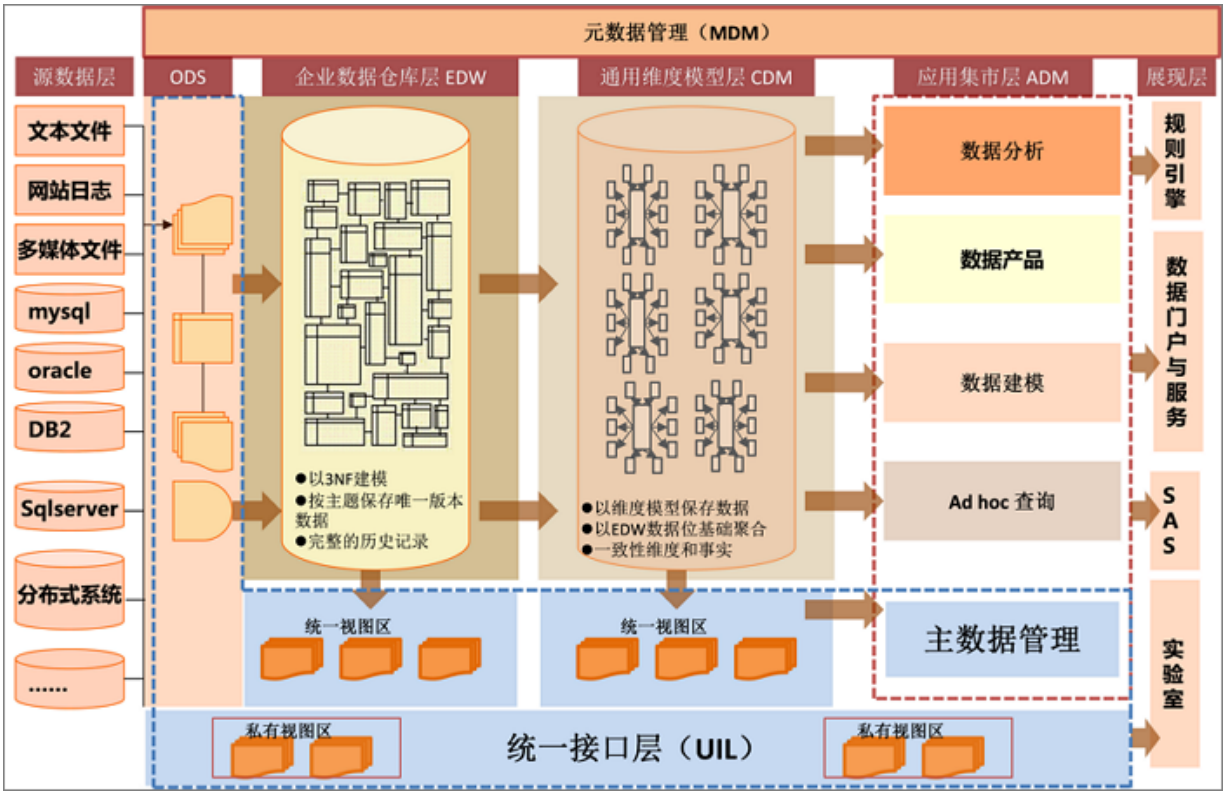
- **支持以多种数据通道**
  - # 取数据支持的数据源：mysql、oracle、sqlserver、postgresql、rds、drds、MaxCompute、ftp、oss、hdfs、dm、sysbase。
  - # 写数据支持的数据源：mysql、oracle、sqlserver、postgresql、rds、drds、MaxCompute、ads、ocs、oss、hdfs、dm、sysbase。
- **可靠的数据质量**
  - # 支持各种数据类型间的转换。
  - # 精确识别脏数据，进行过滤、采集、展示，为您提供可靠的脏数据处理功能，让您准确把控数据质量。
  - # 支持作业的执行情况汇报，让您能够及时了解任务的状态，如查看同步的数据量、脏数据等信息。
- **强劲的传输速度**
  - # 拥有单通道插件性能，单进程能够使单机网卡（200MB/s）满负荷。
  - # 全新的分布式模型，吞吐量无限水平扩展，能够为您提供GB级、乃至TB级数据流量。
- **友好的控制体验**
  - # 精确且强大的流控保证，支持通道、记录流、字节流三种流控模式。
  - # 完备且健全的容错处理，能够做到线程级别、进程级别、作业级别多层次局部/全局的重试。
- **清晰的内核设计**
  - # 拥有专业的框架设计和强大的执行引擎，引擎解决了通用的插件需求，规范了制定流程，同时能够自动发现新增插件。
  - # 更加清晰易用的插件接口，让插件开发人员专注于业务开发，而不再关注框架细节。

33.5 应用场景

33.5.1 大型数据仓库搭建

大型企业可在专有云环境下使用DataWorks来构建超大型的数据仓库。

图 33-4: 构建数据仓库



DataWorks为这类客户提供卓越的海量数据集成能力。

- 海量存储：可支持PB、EB级别的数据仓库，存储规模可线性扩展。
- 数据集成：支持多种异构数据源的数据同步和整合，消除数据孤岛。
- 数据开发：基于MaxCompute（原ODPS）的大数据开发，支持SQL、MR等编程框架，以及贴近业务场景的白屏化工作流设计器。
- 数据管理：基于统一的元数据服务来提供数据资源管理视图，以及数据权限审批流程。
- 离线调度：可以提供多时间维度的周期性调度能力，支持每天百万级的调度并发，并对任务调度实时监控，对错误及时告警。

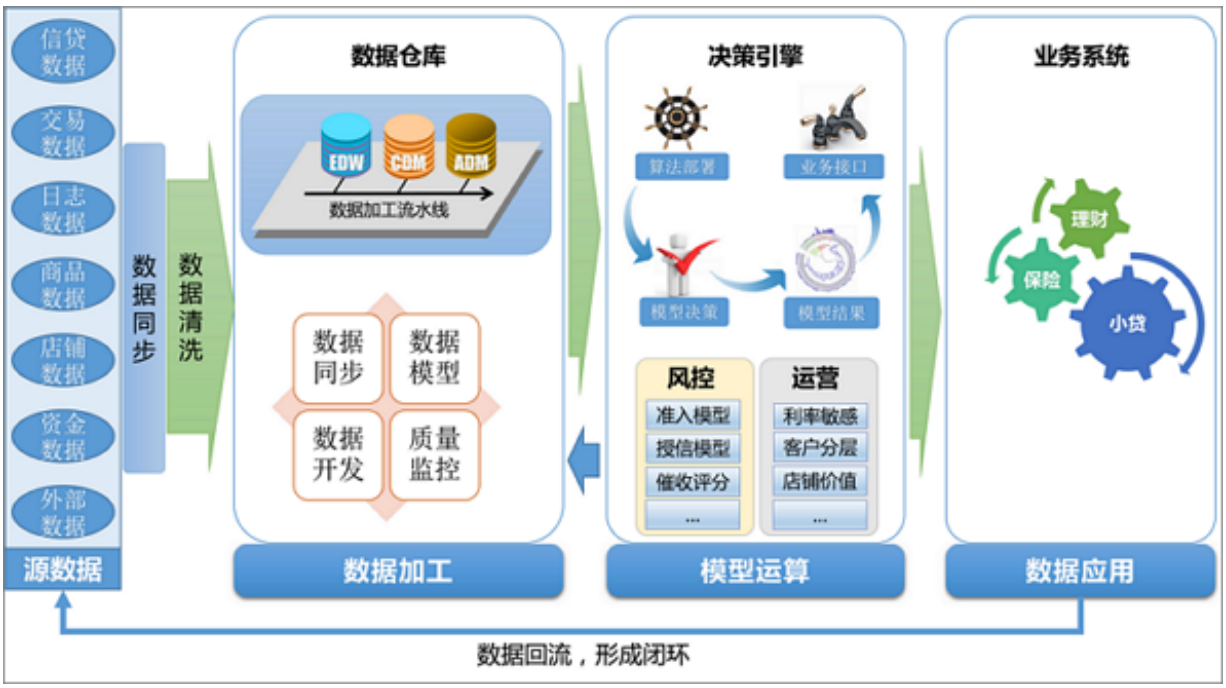
33.5.2 数据化运营

- 创新业务：通过数据挖掘建模和实时决策系统，将大数据加工结果直接应用于业务系统。

- 中小企业：基于DataWorks平台快速使用和分析数据，助力企业的经营决策。

阿里小贷的数据业务运营模式如下所示。

图 33-5: 数据化运营



34 分析型数据库AnalyticDB

34.1 什么是分析型数据库

分析型数据库AnalyticDB（原名 ADS）是阿里巴巴针对海量数据分析自主研发的实时高并发在线分析RT-OLAP（Realtime OLAP）云计算服务，支持对千亿级数据进行即时的（毫秒级）多维分析透视和业务探索。

说明：

联机分析处理OLAP（Online Analytical Processing）系统是相对联机事务处理OLTP（Online Transaction Processing）系统而言的，它擅长对已有的海量数据进行多维度的、复杂的查询和分析，适用于分析型数据库系统。OLTP系统擅长事务处理，数据操作保持着严格的一致性和原子性，支持频繁的数据插入和修改，通常用于MySQL、Microsoft SQL Server等关系型数据库系统。

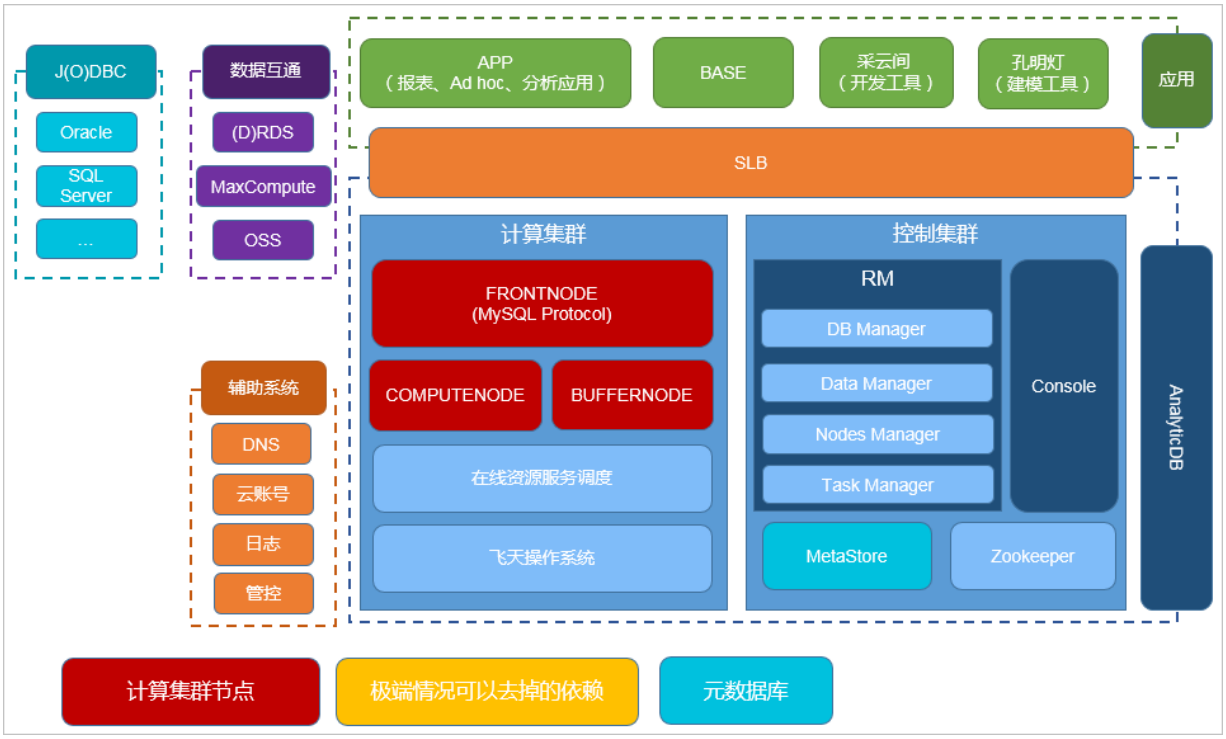
AnalyticDB是一套RT-OLAP系统，具有以下特点：

- 兼容 MySQL、现有的商业智能 BI （Business Intelligence）工具和ETL（Extract-Transform-Load）工具，可以经济、高效、轻松地分析与集成您的所有数据。
- 采用关系模型存储，可以使用SQL进行自由灵活的计算分析，无需预先建模。
- 采用分布式计算技术，具有强大的实时计算能力。在处理百亿条甚至更多量级的数据上，AnalyticDB的性能可以达到甚至超越MOLAP类系统。AnalyticDB在数百毫秒内可以完成百亿级的数据计算，使用者可以根据自己的想法在海量数据中进行自由的探索，而不是根据预先设定好的逻辑查看已有的数据报表。
- 兼具实时和自由的海量数据计算能力，拥有快速处理千亿级别的海量数据的能力。在AnalyticDB系统中，数据分析使用的数据是业务系统中产生的全量数据而不再是抽样的，这使得数据分析结果具有最大的代表性。
- 能够支撑较高并发查询量，同时通过动态的多副本数据存储计算技术也保证了较高的系统可用性，所以AnalyticDB能够直接作为面向最终用户（End User）的产品（包括互联网产品和企业内部的分析产品）的后端系统。当前AnalyticDB已普遍应用于拥有数十万至上千万最终用户的互联网业务系统中，例如淘宝数据魔方、淘宝指数、快的打车、阿里妈妈达摩盘（DMP）、淘宝美食频道等。

作为海量数据下的实时计算系统，AnalyticDB给使用者带来了极速的、自由的大数据在线分析计算体验。

34.2 产品架构

AnalyticDB是基于MPP架构并融合了分布式检索技术的分布式实时计算系统，构建在飞天操作系统之上。AnalyticDB的主体部分主要由底层依赖、计算集群、控制集群和外围模块组成，具体如下图所示。



底层依赖

底层依赖包括：

- 飞天操作系统：用于资源虚拟化隔离、数据持久化存储、构建数据结构和索引。
- MetaStore：阿里云RDS关系数据库或阿里云表格存储，用于存储分析型数据库的各类元数据（注意并不是实际参与计算用的数据）。
- 开源Apache ZooKeeper模块：用于对各个组件进行分布式协调。

计算集群

计算集群是计算资源实际包括的内容，均可进行横向扩展。计算集群运行在飞天操作系统上，通过在线资源调度模块来调度计算资源。计算集群包括：

- FRONTNODE：用于处理用户连接接入认证、鉴权，查询路由和分发路由，以及提供元数据查询管理服务。
- COMPUTENODE：用于进行实际的数据存储与计算的 。
- BUFFERNODE：用于处理数据实时更新、数据缓冲和实时数据写入版本控制。

控制集群

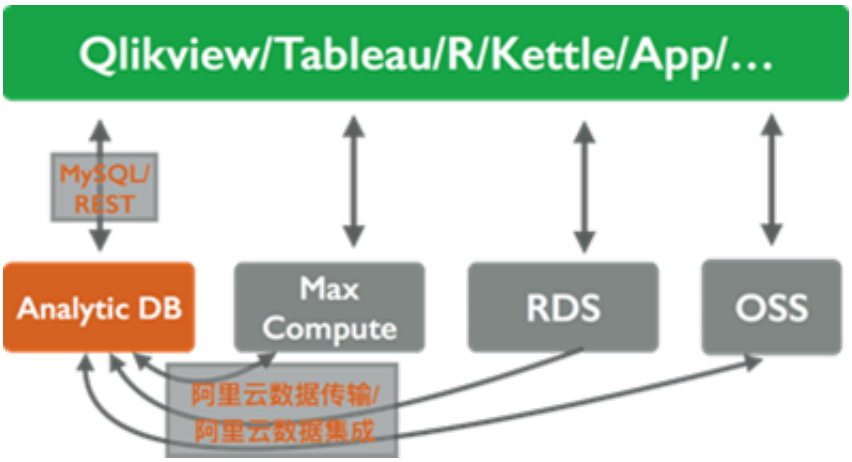
控制集群（即资源管理器RM）用于控制计算集群中数据库资源分配、数据库内数据和计算资源的分布、飞天集群上的计算节点管理、数据库后台运行的任务管理等。控制集群实际上由多个模块组成，一个控制集群可以同时管理部署于不同机房的多套计算集群。

外围模块

外围模块主要包括：

- 阿里云负载均衡：用于管理Front Node的分组和负载均衡。
- 阿里云DNS系统：用于发布数据库域名。
- 阿里云账号系统。
- AnalyticDB控制台（Admin Console）。
- 用户控制台（DMS for Analytic DB）。

外围模块与外部系统交互如下图所示。



- 支持从MaxCompute批量导入数据，也支持快速批量导出海量数据到MaxCompute。
- 支持实时的将(D)RDS的数据同步到分析型数据库中（需借助外部同步工具）。
- 支持从OSS批量导入数据，也支持快速批量导出海量数据到OSS。

AnalyticDB主要支持的客户端、驱动、编程语言和中间件如下：

- 客户端和驱动：支持MySQL 5.1/5.5/5.6系列协议的客户端和驱动，如MySQL 5.1.x jdbc driver、MySQL 5.3.x odbc connector(driver)、MySQL 5.1.x/5.5.x/5.6.x 客户端。
- 编程语言：JAVA、Python、C/C++、Node.js、PHP、R（RMySQL）。
- 中间件：Websphere Application Server 8.5、Apache Tomcat、JBoss。

34.3 功能特性

34.3.1 实体

34.3.1.1 用户

用户是AnalyticDB数据库的使用者，可通过云账号进行登录数据库。在AnayticDB数据库中，不同用户可以被授予不同的权限，用户的操作也均可以被细粒度审计。

34.3.1.2 数据库

数据库是组织、存储和管理数据的仓库。在AnalyticDB中，数据库是租户隔离的基本单位，是用户所关心的最大单元，也是用户和分析型数据库系统管理员的管理职权的分界点。

分析型数据库系统管理员最小可管理数据库粒度的参数，未经用户授权，无法查看和管理数据库的内部结构和信息。用户只能查看大多数数据库级别的参数，但不能修改。

在AnalyticDB中，一个数据库对应一个用于访问的域名和端口号，并且有且只有一个Owner（即创建者）。

在AnalyticDB中，用户的宏观资源配置是以数据库为粒度进行的，所以在创建数据库时分析型数据库通过业务预估的QPS、数据量、Query类型等信息智能的分配数据库初始资源。

34.3.1.3 表组

表组是一系列可发生关联的表的集合，是AnalyticDB为方便管理关联数据和资源配置而引入的概念。在AnalyticDB中，表组可分为普通表组和维度表组，说明如下：

- 普通表组：
  # 普通表组是数据物理分配的最小单元，数据的物理分布情况通常无需用户关心。
  # 一个普通表组最大支持创建256个普通表。
  # 数据库中数据的副本数（指数据在AnalyticDB中同时存在的份数）必须在表组上进行设定，同一个表组的所有表的副本数一致。
  # 只有同一个表组的表才支持快速HASH JOIN。
  # 同一个表组内的表可以共享一些配置项（例如：查询超时时间）。如果表组中的单表对这些配置项进行了个性化配置，那么在进行表关联时会用表组级别的配置进行覆盖单表的个性化配置。
  # 同一个表组的所有表的一级分区（即HASH分区）的分区数建议一致。
- 维度表组：

用于存放维度表（一种数据量较小，但能和任何表进行关联的表）。

维度表组是创建数据库时自动创建的，一个数据库有且只有一个，不可修改和删除。

34.3.1.4 表

AnalyticDB支持标准的关系表模型。在AnalyticDB中，表通常分为普通表（也称事实表）和维度表，说明如下：

- 普通表：创建时必须至少指定一级分区（即HASH分区）列、分区相关信息，同时还需要指定该普通表所属的表组。普通表支持根据若干列进行数据聚集（聚集列），以实现高性能查询优化。单表最大支持1024个列，可支持数千亿行甚至更多的数据。
- 维度表：创建时不需配置分区信息，但会消耗更多的存储资源，单表数据量大小受限。维度表最大可支持千万级的数据条数，可以和任意表组的任意表进行关联。

普通表最多支持两级分区，一级分区是 HASH分区，二级分区是LIST 分区。HASH 分区和LIST分区说明如下：

- HASH 分区：

  # 根据导入操作时已有的一列内容进行散列后进行分区。

  # 一个普通表至少有一级HASH分区，分区数最小支持8个，最大支持256个。

  # 多张普通表进行快速 HASH JOIN ，JOIN KEY必须包含分区列，并且这些表的HASH分区数必须一致。

  # 数据装载时，仅包含HASH分区的数据表会全量覆盖历史数据。

- LIST分区：

  # 根据导入操作时所填写的分区列值来进行分区。同一次导入的数据会进入同一个LIST分区，因此LIST分区支持增量的数据导入。

  # 一个普通表默认最大支持365*3个二级分区。

在AnalyticDB中，两级分区表同时支持HASH JOIN和增量数据导入。对于任一分区形态的表，查询操作均不强制要求指定分区列，但查询操作指定分区列或分区列范围时可能提高查询性能。

在AnalyticDB中，根据表的数据更新方式，表分为批量更新表和实时更新表，说明如下：

- 批量更新表：适合将离线系统（如MaxCompute）产生的数据批量导入到分析型数据库，供在线系统使用。
- 实时更新表：支持通过INSERT、DELETE语句增加和删除单条数据，适合从业务系统直接写入数据。

注意：

- 分析型数据库不支持读写事务。
- 数据实时更新后，一分钟左右才可查询。
- 分析型数据库遵循最终一致性。

34.3.1.5 维度表

- 支持和任意表组的任意表以任意列进行Join加速。
- 最大可支持千万级的数据条数。
- 无需指定分区方式。

34.3.1.6 列

- 支持boolean、tinyint、smallint、int、bigint、float、double、varchar、date、timestamp等多种MySQL标准数据类型。
- 支持多值列multivalue（AnalyticDB特有的数据类型列），高性能存储和查询一个列中的多种属性值信息。
- 支持删除列的自动化索引，无需手动追加建立HashMap索引。

34.3.1.7 ECU

ECU是弹性计算单元，是Analytic DB的资源调度和计量的基本单位。

ECU可配置不同的型号，每种型号的ECU可配置CPU核数（最大、最小）、内存空间、磁盘空间、网络带宽等多种资源隔离指标。Front Node、Compute Node、Buffer Node均由ECU进行资源隔离。Compute Node的ECU数量和型号需由用户配置（弹性扩容/缩容），Front Node、Buffer Node的数量和型号系统自动根据ComputeNode的情况换算和配置。Analytic DB出厂时已根据机型配置预设了多种最佳的ECU型号。

34.3.2 DDL

数据库管理：

- 通过DDL创建数据库。
- 通过DDL删除数据库。
- 查看全部有权限的数据库列表（show databases）。
- 查看和管理每个数据库的访问信息（域名、端口等信息）。

- 通过DDL对数据库使用的ECU资源进行扩容、缩容。
- 通过DDL创建表组和修改表组属性。
- 通过DDL创建表。
- 通过DDL在已创建的表中增加列。
- 通过DDL修改表属性。
- 通过DDL修改索引。
- 支持Create-table-as-Select创建临时表。

34.3.3 DML

34.3.3.1 SELECT

- 和标准MySQL的Query兼容达90%。
- 支持表达式、函数、别名、列名、case when等列投射形式。
- 支持From表名as别名，Join表名as别名。
- 支持事实表之间的Join（若需加速Join则有限定条件）和事实表与维度表的Join（几乎无限制）。
- 支持多个on条件的Join（若需加速Join则其中必须包含一个一级分区列）。
- 过滤条件（where）中，支持and和or表达式组合、支持函数表达式、支持between、is等多种逻辑判断和条件组合。
- 支持多列group by，并且支持case when等列投射表达式产生的别名进行group by，支持常见的聚合函数。
- 支持order by表达式、列，并支持正序和倒序。
- 支持having。
- 支持子查询（建议不超过3层），支持在特定条件下的两个子查询的Join，支持过滤条件中的in中使用数据全部源自维度表的子查询，通过Full MPP Mode支持任意的in子查询。
- 支持带有一级分区列的多列的[count]distinct，在Full MPP Mode下支持任意列的[count]distinct。
- 支持常数列。
- 支持union/union all，有限定条件的支持minus/intersect。

34.3.3.2 INSERT/DELETE

- 支持对已定义主键的实时写入表进行INSERT、DELETE操作。
- 在资源足够的情况下，单表可支撑5万次每秒以上的INSERT操作，数据插入后数分钟内生效。

- 多种机制保障写入成功的数据不会丢失，insert支持overwrite、ignore两种模式。
- 支持insert into...select from。

34.3.4 存储模式

AnalyticDB支持两种存储模式：

存储模式	描述	优点	缺点	是否支持模式切换
高性能存储模式	采用全SSD（或Flash卡）作为计算用数据存储，采用内存作为数据和计算的动态缓存实例，可运行于双千兆或双万兆的网络服务器上。	计算性能好、查询并发能力强。	存储成本较高。	不支持。
大存储模式	采用分布式存储的SATA磁盘作为计算用数据存储，采用SSD和内存两级作为数据和计算的动态缓存实例。必须运行于双万兆的网络服务器上。	存储成本低。	查询并发能力相对较弱，一次性计算较多行列时性能较差。	支持切换到高性能存储模式。

34.3.5 计算引擎

AnalyticDB支持两套计算引擎：

计算引擎	描述	优点	缺点
COMPUTENode Local /Merge（简称LM）	原有计算引擎。	计算性能好、并发能力强。	对部分跨一级分区列的计算支持较差。
Full MPP Mode（简称MPP）	新增的计算引擎。优点是计算功能全面，支持跨一级分区列的计算	计算功能全面，支持跨一级分区列的计算，可以通过全部TPC-H查询测试用例（22个）和60%以上的TPC-DS查询测试用例。	计算性能相对LM引擎较差，计算并发能力相对LM引擎很差。

当开启MPP引擎时，AnalyticDB自动对查询（Query）进行路由，并将LM引擎不支持的Query路由到MPP引擎，以兼顾AnalyticDB的性能和通用性。用户也可以通过Hint来指定某个Query 使用的计算引擎。

34.3.6 系统资源管理

AnalyticDB通过ECU（弹性计算单元）进行资源管理。通过操作系统底层技术和飞天操作系统提供的分布式资源调度能力，AnalyticDB为每个数据库实例创建完全独立的FRONTNODE、COMPUTENODE、BUFFERNODE进程。每个数据库至少拥有FRONTNODE、COMPUTENODE、BUFFERNODE进程各两个（双副本双活）。

您可以通过控制ECU型号来控制FRONTNODE、COMPUTENODE、BUFFERNODE进程的配置。通过ECU型号可以区分的资源包括CPU 核数（支持独占和共享）、内存大小（独占）、SSD大小（独占）、网络带宽（独占）、SATA数据逻辑大小（仅大存储实例的COMPUTENODE可选）。

您可以通过ECU的数量来控制一个数据库实例所要启用的COMPUTENODE数量；而通过ECU上配置的COMPUTENODE与FRONTNODE和BUFFERNODE的比例，系统会自动启用相应数量的FRONTNODE和BUFFERNODE。通过这种方式可以达到容量水平伸缩的目的。

FRONTNODE、COMPUTENODE、BUFFERNODE进程默认混部在同一批物理机上，您也可以通过参数配置，强制让不同的角色运行于不同的物理机上。

除此之外，AnalyticDB的后台任务、数据库AM等也会占用一定量的系统资源。

34.3.7 权限与授权

授权模型：

- 支持标准MySQL模式的权限模型。
- 支持对数据库、表组、表、列四个级别进行ACL授权。
- 支持数据库Owner授权给任意合法账号。
- 支持单独控制或全局控制数据库创建权限。
- 支持超级管理员、系统管理员等角色。
- 支持每个级别授予不同的权限。
- 支持ADD USER/REMOVE USER语句添加和删除用户。
- 支持GRANT语句进行授权。
- 支持REVOKE语句进行权限回收。
- 支持SHOW GRANTS ON语句查看各级对象上的用户权限。
- 支持LITS USERS语句查看全部有权限的用户。
- 超级管理员：集群初始建立时指定的账号，具有任命系统管理员和数据库管理员的权限，无其他权限。

- 系统管理员：由超级管理员任命，具有查看和操作SYSDB的权限。
- 数据库管理员：由超级管理员任命，具有为其他用户创建数据库和删除其他用户的数据库的权限。
- 数据库Owner：数据库的所有者，具有一个数据库的全部权限，并可以授权一般用户访问自己的数据库。

34.3.8 数据导入导出

支持快速导入导出海量数据：

- 支持任何SELECT语句的查询输出。
- DUMP DATA语句支持将大量数据快速导出到OSS等DFS中以及MaxCompute。
- 支持类BULKLOAD模式导入MaxCompute、OSS、RDS中用户存放的数据。
- 内置支持使用LOAD DATA语句进行导入。
- 内置支持导入数据Owner校验，保证导入安全。

34.3.9 元数据

34.3.9.1 information_schema

- 最大限度兼容MySQL标准的数据库、表、列等信息，元数据完全可被您使用并可进行交互。
- 数据导入的记录和进度均可在元数据库进行查询。
- 提供ECU运行状态，以及ECU扩容、缩容记录表。
- 元数据按照数据库进行隔离，您无法访问无权使用的元数据。

34.3.9.2 performance_schema

- 提供实时元仓，可进行SQL粒度的查询审计以及分钟粒度的插入性能统计。
- 提供分钟级别更新的QPS、RT、请求数、数据量大小等实时性能监测。
- 元数据按照DB进行隔离。

34.3.9.3 sysdb

- 面向系统管理员和运维人员的元数据库。
- 支持查看AnalyticDB全部模块的运行状态、运行历史记录等，拥有数十张各个主题的系统元数据表。
- 支持查看系统的运行状态，并在有需要时可以进行修改。
- 支持查看或修改系统各组件的参数，运行计算集群升级、降级、扩容、缩容、挂起命令。

34.3.10 管理控制台

34.3.10.1 用户控制台（DMS for AnalyticDB）

- 支持阿里云账号登录与鉴权。
- 提供图形化的数据库创建与管理、表组/表的创建与管理功能。
- 提供友好的SQL查询调试功能，并提供图形化的执行计划展示。
- 提供友好的用户与权限查看、管理功能。
- 提供友好的数据导入导出、导入导出状态查询和导入导出进度查询界面。
- 提供两分钟内实时系统性能报告的展示以及最近七天内小时粒度详细的离线性能报告展示。
- 提供数据库资源可视化。
- 提供扩容、缩容、查看扩容/缩容状态和历史的功能。

34.3.10.2 运维管理控制台（大数据管家）

- 用于系统后台运维人员管理系统资源、监控系统运行状态和修改系统参数。
- 提供图形化的界面展示各个数据库的存储计算资源，以及在物理节点上的占用、分布。
- 提供图形化的查看和管理系统参数功能。
- 提供图形化的查看和管理数据导入全链路状态功能。
- 提供图形化的分布式日志提取和查看工具。

34.3.11 特色功能

34.3.11.1 特色函数

- 支持高性能的根据地理坐标范围筛选数据（方形和圆形圈选、点距离计算等）。
- 支持智能分段统计函数（指标的自动分段）。
- 支持快速多列聚合函数。
- 根据列的数据分布智能为全部列建立索引，无需您进行任何操作。
- 对完全不需进行检索的列，您可手动关闭智能索引。

34.3.11.2 智能缓存和CBO优化

- 拥有多层智能缓存，最大限度的利用内存加速计算，但是可计算的数据量大小不受内存大小限制。
- 拥有智能的CBO优化器，可以根据数据分布情况和您的SQL执行情况动态优化计算执行计划，让您从SQL写法优化中解脱。

- 拥有智能的分布式长尾处理技术，大幅度降低分布式系统中单节点繁忙以及网络等不确定性因素对响应时间的影响。

34.3.11.3 Quota控制

- 支持每次导入数据量、单表导入次数、并发任务数、DB导入次数、DB导入总数据量等控制。
- 支持ECU总数据量控制、ECU库存控制、单次扩容ECU数量控制、每天扩容次数控制。
- 支持DB的总表数、总实时表数、总表组数、单表最大列数、单表分区数（一级、二级）控制。
- 支持系统元数据库（SYSDB、ADMIN DB）流控和风控。
- 支持大存储模式实例，使用SATA进行计算数据存储，使用SSD和内存作为缓存加速热点数据查询。

34.3.11.4 Hint和小表广播

- 支持通过Hint干预执行计划，如计算引擎的选定和索引使用的控制。
- 支持小表广播模式Join，在小表（物理表或虚表）Join大表时通过 Hint指定小表广播，在不符合加速Join条件时亦可获得比较好的Join性能。

34.4 产品优势

优势	描述
海量数据计算能力	最高支持计算单表万亿记录、PB级别的数据。
全量数据分析	数据分析中使用的数据不再是抽样的，而是全量数据，分析结果具有最大的代表性。
查询响应极速	支持在毫秒级内对百亿级数据进行多维透视。
高并发、高可用	支持高并发查询量，并且通过动态的多副本数据存储计算技术来保证系统的高可用性，能够直接作为面向最终用户（End User）产品（包括互联网产品和企业内部的分析产品）的后端系统。
查询形式自由灵活	支持通过SQL灵活的对海量数据进行多维分析、数据透视、数据筛选。
数据导入多通道并行	支持离线通道、在线通道双模式并行数据导入，导入性能随集群规模线性扩展。
安全机制精细	支持精确到列级别的权限管理和超细粒度的用户操作审计，通过公私钥机制保护数据安全。

优势	描述
兼容性良好	全面兼容MySQL协议（包括数据元信息）、兼容商业分析工具和应用、内置支持多种数据源数据快速接入，大幅度降低业务系统和商业软件的接入成本。

35 流计算StreamCompute

35.1 产品历程

阿里云流计算源自于阿里集团内部双十一实时大屏业务，从最开始支持双十一大屏展现和部分实时报表业务的实时数据业务团队，历经约5年的长期摸索和发展，到最终成长为一个独立稳定的云计算产品团队。阿里云流计算期望将阿里集团本身沉淀多年的流计算产品、架构、业务能够以云产品的方式对外提供服务，助力更多的企业实时化自身大数据业务。

最初阿里集团支撑双十一大屏等业务同样采用的是开源的Storm作为基础系统支持，并在上面开发相关Storm代码。这个时期的实时业务处于萌芽阶段，规模尚小。数据开发人员使用Storm原生API开发流式作业，开发门槛高，系统调试难，存在大量重复的人工工作。

阿里集团的工程师针对这类大量重复工作，开始考虑进行业务封装和抽象。首先我们希望有个流计算和批处理的一体化处理方案。Spark和Flink都具有流和批处理能力，但是他们的做法是相反的。Spark Streaming是把流转化成一个个小的批来处理，这种方案的一个问题是我们需要的延迟越低，额外开销占的比例就会越大，这导致了Spark Streaming很难做到秒级甚至亚秒级的延迟。Flink是把批当作一种有限的流，这种做法的一个特点是在流和批共享大部分代码的同时还能够保留批处理特有的一系列的优化。因为这个原因，如果要用一套引擎来解决流和批处理，那就必须以流处理为基础，所以我们决定先选择一个优秀的流处理引擎。从功能上流处理可以分为无状态的和有状态两种。在流处理的框架里引入状态管理大大提升了系统的表达能力，让您能够很方便地实现复杂的处理逻辑，是流处理在功能上的一个飞跃。

任何技术的发展一定遵循小众/创新到大众/普及的成长轨迹，而从小众到大众、从创新到普及的转折点一定在于技术的功能成熟和成本降低。阿里工程师开始思考如何更大程度降低数据分析产品门槛从而普及到更多的用户。Flink在架构上有很多创新，是非常领先的，但是在工程上的实现有一些不足支出，不同的Job的任务可能会运行在同一个进程里，这样大大降低了系统运算的性能，阿里的工程师为了解决这个问题，重新实行了Yarn的结合，完全解决了这些问题，另外Flink是通过checkpoint的机制来保证一致性的，但原有的机制效率比较低，导致在状态（增量计算保存的状态）较大的时候不可用，Blink大大优化了checkpoint，能够高效地处理很大的状态。稳定性和可扩展性在生产上都是至关重要的，通过在大集群上的锤炼，Blink解决了一系列这方面的问题和瓶颈，已经成为一个能够支撑核心业务的计算引擎。同时我们扩展了Flink的 SQL层，使得它能够比较完备地支持较复杂的业务，正是不断的去发现、去创新，才有了今天阿里云流计算的核心计算引擎（Blink）。

35.2 产品特点

StreamCompute的底层引擎Blink是基于Flink开发的，继承了Flink的所有优点并改进了Table API，使其更完整，因此您可以使用相同的SQL进行批处理和流式处理。 更强大的YARN模式，仍然100%兼容Flink的API和更广泛的生态系统。

图 35-1: 流计算技术栈及StreamCompute和竞品的关系

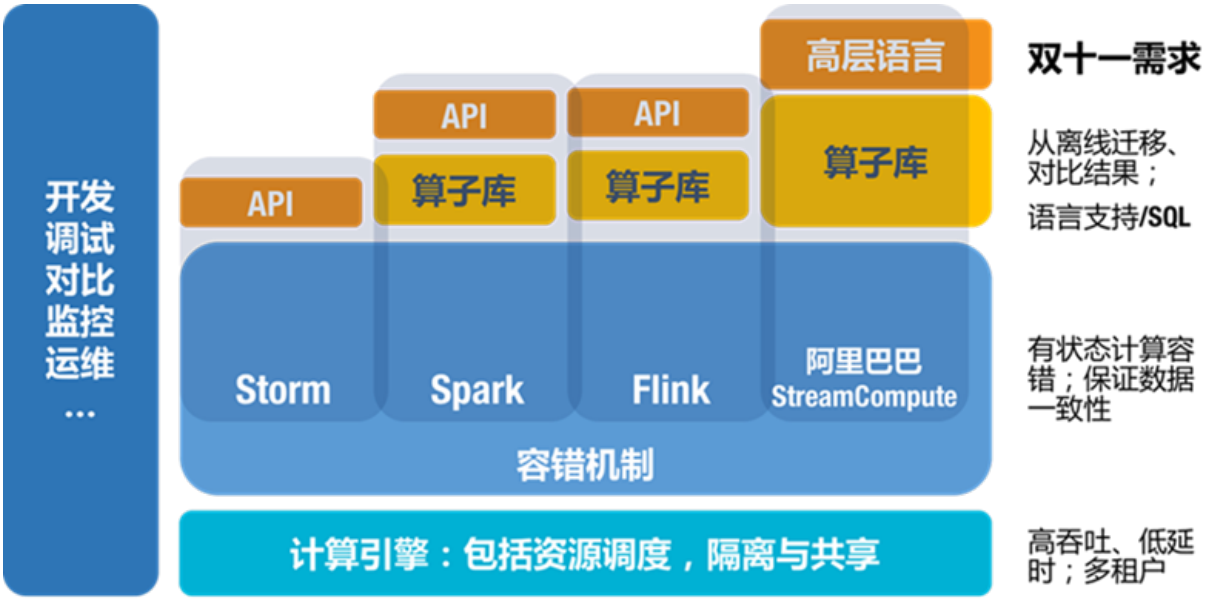


图 *35-1: 流计算技术栈及StreamCompute和竞品的关系*展示了流计算产品的技术栈，阿里云StreamCompute在世界级挑战的业务场景锤炼中，逐步形成了如下竞争优势。

- **功能强大**

  不同于其他开源流计算中间件只提供粗陋的计算框架，大量的流计算细节需要业务人员造轮子重新实现。阿里云流计算集成诸多全链路功能，方便您进行全链路流计算开发，包括：

  # 强大的流计算引擎。
    - 提供流式计算的标准StreamSQL，支持各类Fail场景的自动恢复，保证故障情况下数据处理的准确性。
    - 支持多种内建的字符串处理、时间、统计等类型函数。
    - 精确的计算资源控制，彻底保证多租户之间作业的隔离性。
  # 关键性能指标超越开源Flink的3到4倍，数据计算延迟优化到秒级乃至亚秒级，单个作业吞吐量可做到百万(记录/秒)级别，单集群规模在数千台。

深度整合各类云数据存储，包括MaxCompute、DataHub、Log Service、RDS、Table Store、AnalyticDB等各类数据存储系统，无需额外的数据集成工作，阿里云流计算可以直接读写上述产品数据。

- **托管的实时计算服务**

  不同于开源或者自建的流式处理服务，阿里云流计算是完全托管的流式计算引擎，阿里云可针对流数据运行查询，无需预置或管理任何基础设施。在阿里云流计算，您可以享受一键启用的流式数据服务能力。阿里云流计算天然集成数据开发、数据运维、监控告警等服务，方便您以最小成本试用和迁移流式计算。

  提供完全租户隔离的托管运行服务，从最上层工作空间到最底层执行机器，提供最有效的隔离和全面防护，让您放心使用流计算。

- **良好的流式开发体验**

  支持标准SQL(产品名称为：BlinkSQL)，提供内建的字符串处理、时间、统计等各类计算函数，替换业界低效且复杂的Flink开发，让更多的BI人员、运营人员通过简单的BlinkSQL可以完成实时化大数据分析和处理，让实时大数据处理普适化、平民化。

  提供全流程的流式数据处理方案，针对全链路流计算提供包括数据开发、数据运维、监控预警等不同阶段辅助套件，让数据开发仅需三步，即可完成流式计算作业上线。

- **成本低廉**

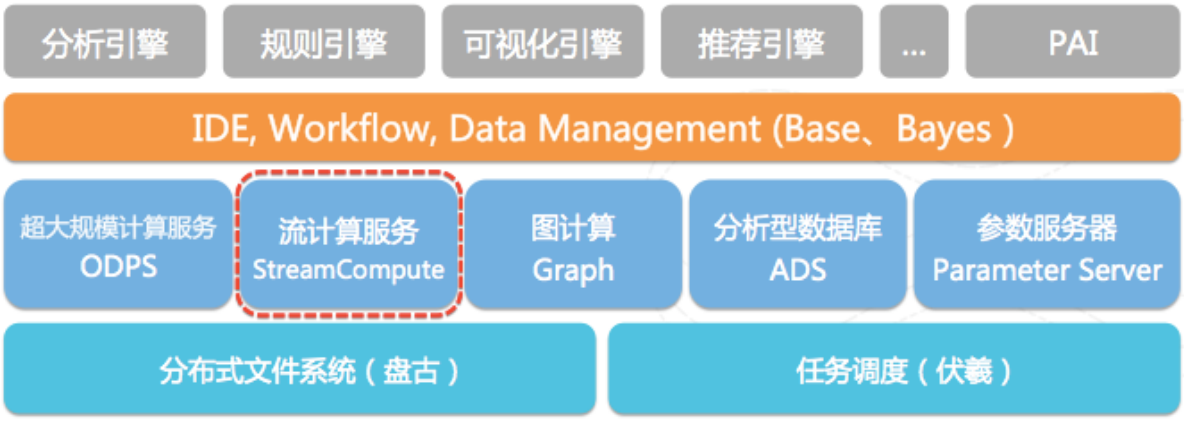
  大量优化的SQL执行引擎，会产生比手写原生Flink作业更高效且更廉价的计算作业，无论开发成本和运行成本，阿里云流计算均要远超开源流式框架。

35.3 产品战略地位及发展路线

阿里云流计算在阿里集团内部有数千台规模的集群，服务20多个BU的数百个实时应用，日均消息处理数千亿，流量近PB级别，成为阿里集团最核心的分布式计算服务之一。

StreamCompute 在阿里云计算平台中的位置如图 *35-2:* 流计算战略地位所示。

图 35-2: 流计算战略地位



阿里云流计算后续主要在以下几个方面重点加强。

- 计算引擎：重点针对性能提升、多种消息处理语义支持等方面进行优化。
- 编程接口：提供更丰富的API支持，支持多语言，兼容开源系统API，比如Storm API、Beam API等。
- 语言：丰富Streaming场景的SQL表达能力，增加对Temporal、CEP等语法和语义的支持。
- 产品：对StreamCompute的可调试性、一键部署、热升级、培训体系等方面持续进行完善。

35.4 产品架构

35.4.1 业务架构

阿里云流计算是指一套轻量级的提供SQL表达能力的流式数据加工处理引擎，如图 *35-3:* 业务架构所示。

图 35-3: 业务架构

- **数据产生**

  生产数据发生源，通常在服务器日志、数据库日志、传感器、第三方数据均是数据产生方，这份流式数据将作为流计算的驱动源进入数据集成模块。

- **数据集成**

  提供流式数据集成的用以进行数据发布和订阅的数据总线，包括可以集成大数据计算的DataHub、连接物联网信息的 IoTHub、和对接 ECS 日志的 Log Service。

- **数据计算**

  阿里云流计算通过订阅数据集成提供的流式数据，驱动流计算的运行。

- **数据存储**

  流计算本身不带有任何存储，流计算将流式加工计算的结果写入数据存储，包括关系型数据库、NoSQL数据库、OLAP 系统等等。

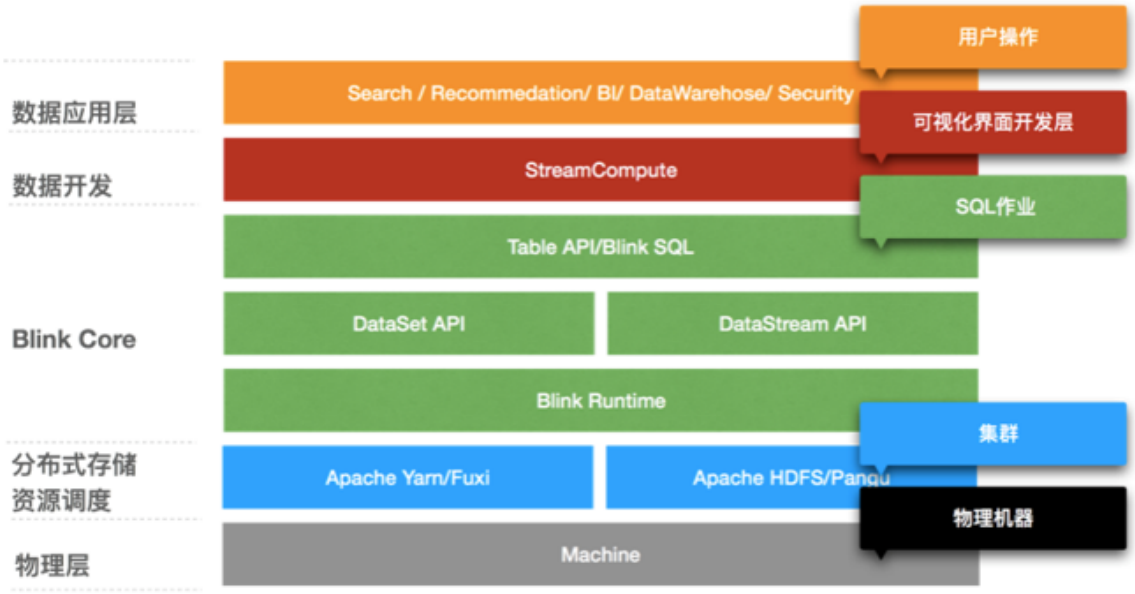
- **数据消费**

  不同的数据存储可以进行多样化的数据消费。提供消息队列的数据存储可以用作报警，提供关系型数据库的可以提供在线业务支持等等。

35.4.2 技术架构

流计算是一个实时的增量计算平台，其能提供类似SQL的语言，通过MapReduceMerge计算模型（简称MRM）完成增量式计算。流计算具有比较完善的FailOver机制，能保证在各种异常情况下数据的精确性，如技术架构图所示。

图 35-4: 技术架构图



流计算主要由5部分组成。

- **数据应用层**

  主要提供了开发平台，便于新业务的开发和作业提交，系统提供了完善的监控告警系统，在作业出现延迟时及时通知到业务方，同时您可以通过Blink UI等系统了解线上作业的运行情况和性能瓶颈，从而能够及时、更好的优化作业。

- **数据开发**

  该层主要负责Blink SQL的解析和逻辑及物理执行计划的生成，并最终将执行计划转化成可执行的DAG。该层会依据SQL得到的DAG生成由不同Model组成的有向图，用以处理具体的业务逻辑，通常一个Model会包含以下三部分。

  # Map：进行数据过滤、分发（group）或Join（MapJoin）等操作。
  # Reduce：完成一个batch内的聚合计算（流计算将流数据打包成一个个批任务来进行处理，每个批任务内会有多条数据记录）。
  # Merge：将该批任务内的计算结果与以前的结果state进行merge操作得到新的state，在n个批任务处理完成后进行checkpoint操作（n值可配置），从而将该state持久化化到State系统中（如HBase、Tair等）。

- **Blink Core**

提供多种计算模型、Table API和Blink SQL。下层支持DataStream API和DataSet API。最下层需要有负责资源调度的Blink Runtime来保证作业运行稳定。

- **分布式资源调度**

  整个流计算集群是构建在Gallardo调度系统之上，其本身也是流计算能够有效运行和出错恢复的重要保证。

- **物理层**

  物理层是指阿里云给与的强大的硬件机器的集群支持。

36 大数据应用加速器DTBoost

36.1 前言

随着近五年互联网和大数据技术的蓬勃发展，各类数据产品应运而生。从阿里大数据的应用发展来看，阿里依然面临很多挑战。

- 为应对数据量高速的增长，衍生出多样的分布式数据计算与存储技术，以解决各类应用场景下的难题，而非传统IT架构当中只需要单一数据库即可支撑整个企业的数据分析报表问题。如何对各类数据的积累进行有效的整合与管理，各个业务库的数据之间如何打通在多个计算存储资源上合理的分布管理也成为一大难题。
- 大数据在各个行业当中的应用，如数字广告、互联网金融、电子商务、在线风控等场景当中，一个数据应用需要包括报表分析、行为预测、实时监控、信用评分、个性化推荐、文本挖掘、时空数据等各类大数据技术方法的综合运用，而不仅仅是做企业经营的报表统计。
- 当下对运用数据的用户不再局限于专业的数据分析师、数据仓库工程师，更多的是能够让非技术背景的业务人员以他能够理解的方式灵活的探查数据。
- 如果想运用好大数据，就需要对企业的IT架构、技术人员的综合能力提出更高的要求。

  # 需要了解各个专业分布式计算和存储资源的特性。

  # 需要深入了解业务人员使用数据的场景，并制作出面向业务的数据产品。

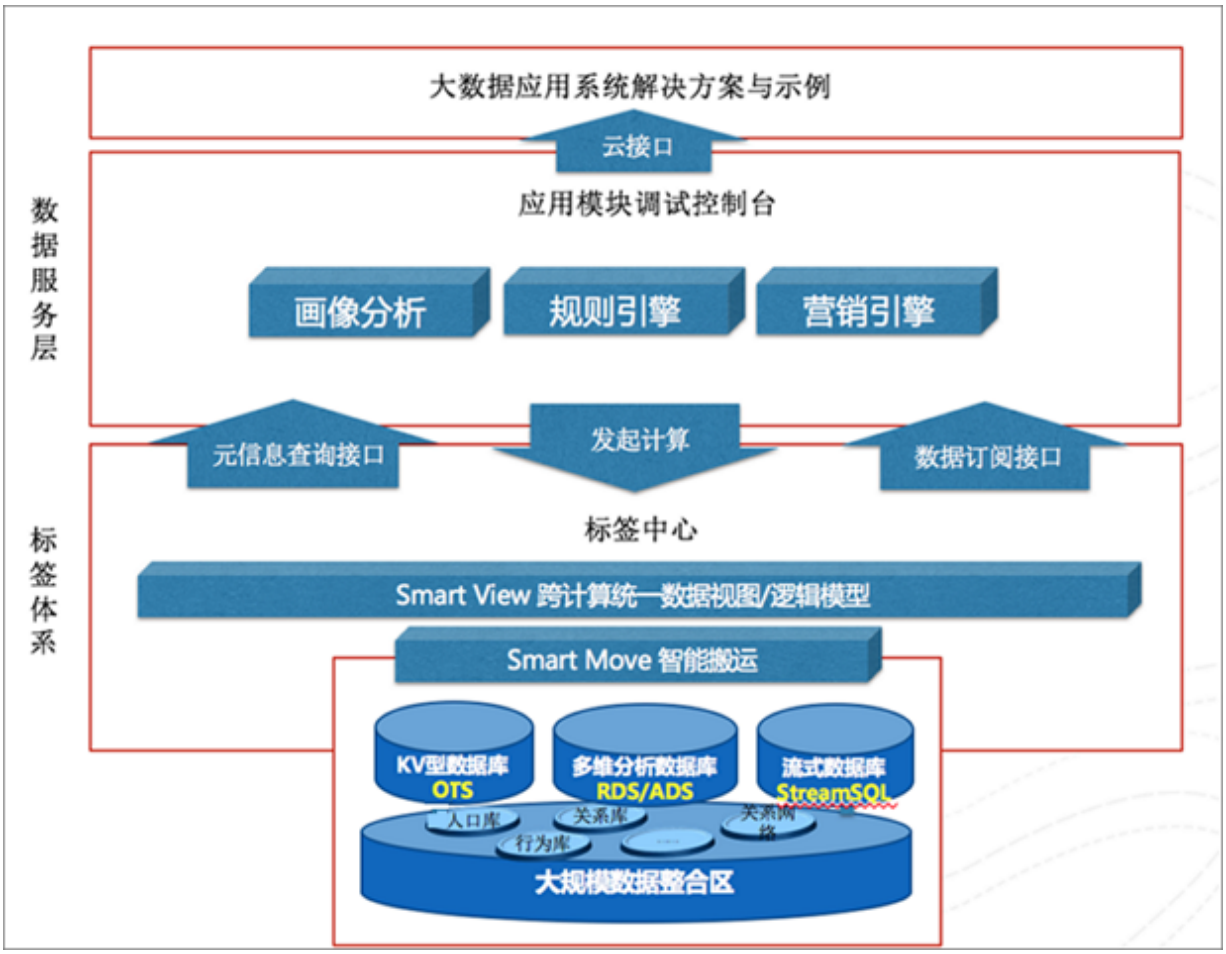
  # 需要将计算和存储资源针对数据分析、算法服务等多种应用场景进行合理的架构。

阿里云DTBoost从大数据应用落地点出发，提供了一套大数据应用开发套件，能够帮助开发者从业务需求的角度有效的整合阿里云各个大数据产品，大大降低搭建大数据应用系统当中绝大部分的系统工程工作，在相应行业应用解决方案的结合下，让不熟悉大数据应用系统开发的程序员也可快速为企业搭建大数据应用，从而实现大数据价值的快速落地。

36.2 产品概述

DTBoost的产品组件如图 *36-1: DTBoost产品组件*所示。

图 36-1: DTBoost产品组件



概括来讲，DTBoost是以标签中心为基础，建立跨多个云计算资源之上的统一逻辑模型，开发者可以在标签逻辑模型视图上结合画像分析、规则引擎、营销引擎等多个业务场景的数据服务模块，通过接口的方式进行快速的应用搭建。

这种方式的好处如下所示：

- 应用开发人员不需对下层多个计算存储资源进行深入理解，减少与复杂的系统对接工作。
- 通过数据服务的形式，有助于IT部门对数据使用进行管理，避免资源的重复和冗余。

因为大数据计算能力的增强，开发者只需要把需要使用的数据在模型当中进行管理后，即可通过API方式进行相应的计算，并对接到产品界面端。或通过提供的界面配置功能直接生成可以独立部署的代码，以快速搭建相应的大数据产品。

整个产品系列包括以下几个模块。

- **标签中心**

  # 云计算资源管理

模型探索

模型管理

- **画像分析**

分析服务

界面配置

- **规则引擎**

服务层

控制层

执行层

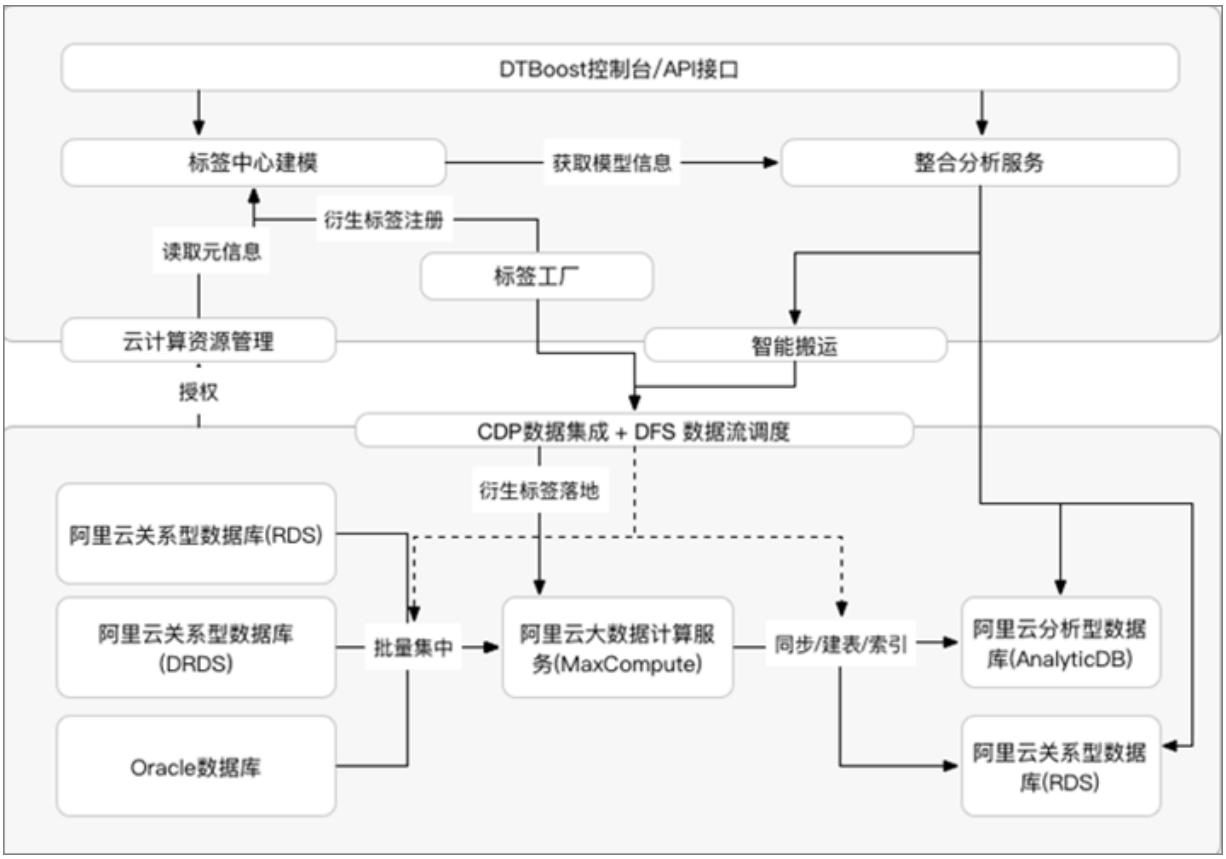
36.3 产品架构

在大数据环境下，一个数据应用往往需要通过多个计算资源来配合完成，最简单来说，一般数据需要先在离线环境当中进行离线加工处理，再同步至在线数据库当中进行在线分析查询。那么标签中心所能够做的就是与多个数据库进行通信，获取多个计算存储资源的数据元信息后进行逻辑建模，并把各个数据服务模块接口传入的指令解析后将真实的计算命令传给每一个计算资源。

下文将以画像分析为例，为您介绍DTBoost的总体架构。

以最常见的OLAP分析场景来看，一般需要从业务库当中将数据进行抽取，加载到大数据（离线）计算服务中进行集中，进行相应的加工、衍生后，再把所需要分析的数据同步到在线分析库（在大数据量下通常会使用分析型数据库AnalyticDB）中。

图 36-2: 技术架构图



您从DTBoost控制台或API进入，通过把自己的云计算资源授权给DTBoost后，即可通过DTBoost读取各个云计算资源中的数据元信息。经过建模配置后，在相关的数据服务模块中可以进行手工/自动触发标签中心的智能搬运模块，通过把相关的数据同步调度任务发送给数据流服务DFS（Data Flow Service）和数据集成，来对所需要整合的数据以标签粒度来进行业务库到离线数据仓库的批量集中，以及到在线分析数据库的同步、建表、索引工作。在数据准备完成之后，就可以通过相关的数据服务API接口或者在控制台上基于标签模型视图之上进行相关的计算。对于当中需要离线计算加工的部分，一些常用的加工可以通过标签工厂来对标签进行批量的衍生（如常见的聚合、筛选组合等）落地到MaxCompute中。

整个过程可以看作DTBoost在大数据平台对各个计算资源之间满足常见业务场景的架构方案进行了系统集成，简化了各个系统之间手工对接等过程。

36.4 功能模块

36.4.1 标签中心

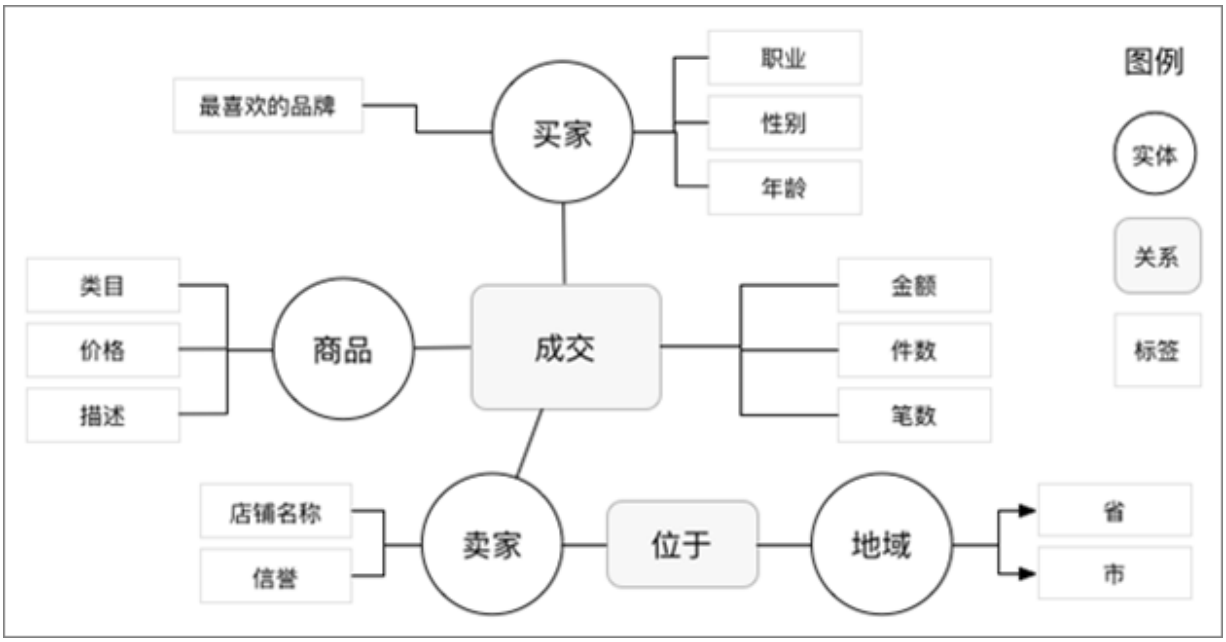
36.4.1.1 概念说明

标签中心的作用是在现有的数据表之上构建跨计算存储的逻辑模型，直接让您在视图层上对数据进行管理、加工、查询，屏蔽下层的多个大数据计算存储资源，简化数据的使用。当整个数据架构越复杂，越是需要多个计算存储资源组合使用的场景下，标签中心的价值就越为明显。

标签建模的方法来源于阿里巴巴用户画像体系，广泛应用于精准营销、个性化推荐、用户画像、信用评分等需要基于明细数据进行计算的大数据应用当中。所谓标签就是对用户这一对象的一个最小描述单元，代表着所描述对象某一个具体的客观事实的抽象表达，如属性（性别：标签值男/女，年龄：标签值实际年龄），行为（成交金额、收藏次数、位置定位），或者是兴趣（对于多个关键词的偏好度），是一种以业务视角出发的数据建模方法，标签既可能是数值、也可能是枚举值，也可以是多个Key-Value组织的列，还可能是多字段组成的事实表（如对象、时间、谓语、宾语）。从概念模型上讲，标签体系是指围绕多个实体对象（如买家、卖家、商品、企业、设备），以及实体之间的关系（如成交、检修、位于等），建立标签化描述的方法。

标签建模如图 *36-3:* 标签建模所示。

图 36-3: 标签建模



这种建模方式看起来可能类似于角模型（Anchor）或者是图模型（Graph），其实并不一样。传统的建模过程是根据业务需求设计概念和逻辑模型，再根据逻辑模型对物理数据表进行加工和规整。

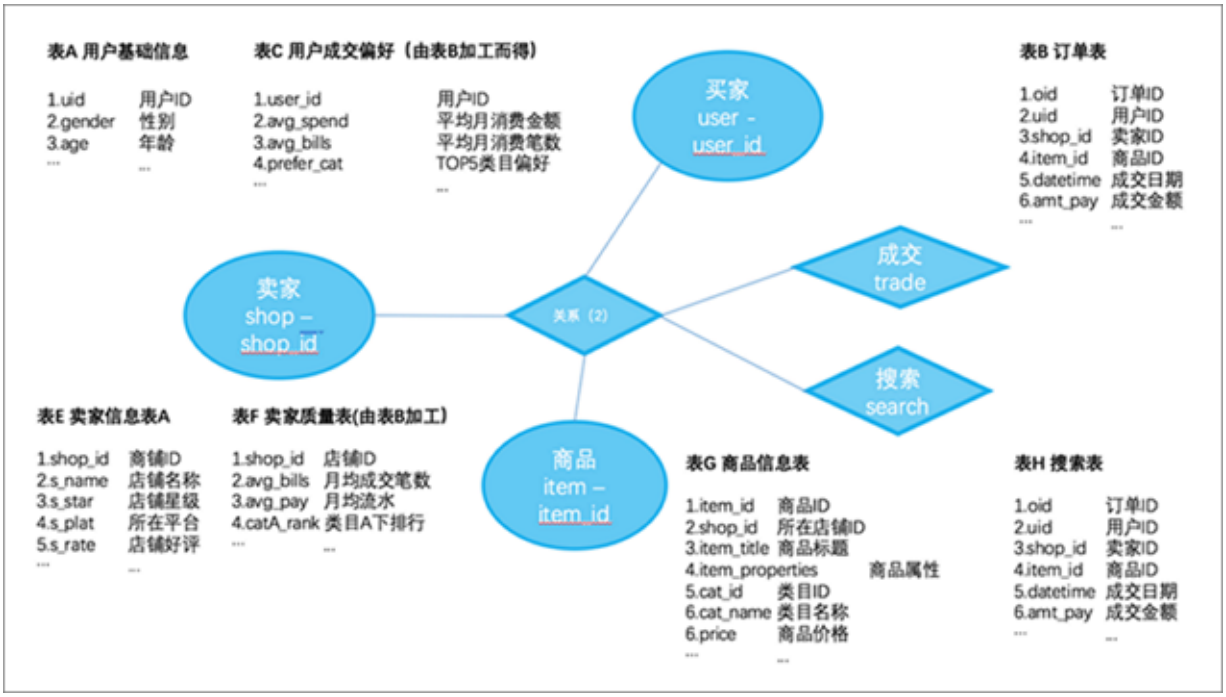
而标签建模是在已有的物理数据/模型之上直接建立逻辑模型，通过各个数据服务的代理解析，让用户可以在视图上直接进行各类的计算，不需要预先对物理数据进行大规模的加工处理，即用即算。

但需要明确的是，总体来说，标签仍然是建立在物化数据之上，因为在跨计算的语境之下可能会面临多个计算的查询语言和性能的差异，建立在逻辑请求上的标签很可能会无法执行，所以总体来讲定义的每一个标签还是需要对应到落地的物理表上。但在DTBoost当中，可以在相应的数据服务当中以某一个计算查询逻辑定义为一个临时标签使用，但关系到跨计算之时还是需要将之物化，避免错误发生的可能。

标签模型是围绕实体（Entity）、关系（Link）、标签（Tag）三大元素对分布在不同数据库中的数据进行网络化的建模方式。实体用于描述某个客观的对象，如设备-人员-地址等，对应到物理数据表上一般就是属性表，有一个主键来代表每一个对象，剩下的每一列就是标签即描述对象的属性。那么关系是表示对象和对象之间的联系、事件、行为，一般对应到物理数据表上一般就是事实流水表，如成交-检修-乘车等。

实体关系建模如图 *36-4:* 实体关系建模所示。

图 36-4: 实体关系建模



相比于指标-维度体系，这种建模方式更适用于对于明细数据描述和表达。明细数据大部分都是事实表，引入关系的概念对应到流水事实表上，把多个实体之间的关系很好的呈现表达，既有利于管理也方便分析时的表达，在对业务端呈现上也更接近于概念模型的设计，方便您理解。

在经过建模转化之后，可以将上表中的模型逻辑关系转化为图 *36-5:* 实体关系管理所示。成交表对应到关系结点上，金额和时间是关系上的标签，用户表和商品表对应到买家和商品两个实体上，性别、年龄是买家的标签。这种建模方式非常便于各类基于明细行为、关系数据进行分析的场景。

图 36-5: 实体关系管理

您可以在标签中心页面下看到标签中心的几大功能，包括**云计算资源、实体关系标签、模型探索**和**标签同步**。

36.4.1.2 适用场景

标签中心是跨计算存储、可在物理模型之上逻辑动态建模、与数据服务结合面向大数据应用开发的数据建模、数据管理工具，并能够通过可视化的方法清晰的展现企业的数据模型视图。

标签中心适用于以下场景。

- **数据模型探索管理**

  标签中心提供一种业务视角的数据发现、模型探索的工具，便于业务人员、开发人员、数据管理人员透视企业的数据资产。

- **为数据服务提供视图支撑**

  为多个计算引擎上的数据提供一个统一的数据视图，结合数据服务能够方便地进行业务逻辑计算操作。

- **数据权限管理**

  可以通过逻辑层对数据访问权限进行有效控制，比物理表的访问管理更加安全有效。

36.4.1.3 功能组件

36.4.1.3.1 云计算资源管理

云计算资源管理能够支持与多个计算存储资源通信，并可获取元信息。

目前DTBoost支持对以下计算存储资源进行管理。

- Oracle数据库
- 阿里云关系型数据库（RDS）
- 阿里云大数据计算（MaxCompute）
- 阿里云分析型数据库（AnalyticDB）
- 阿里云表格存储（Table Store）
- 阿里云流计算（StreamCompute）

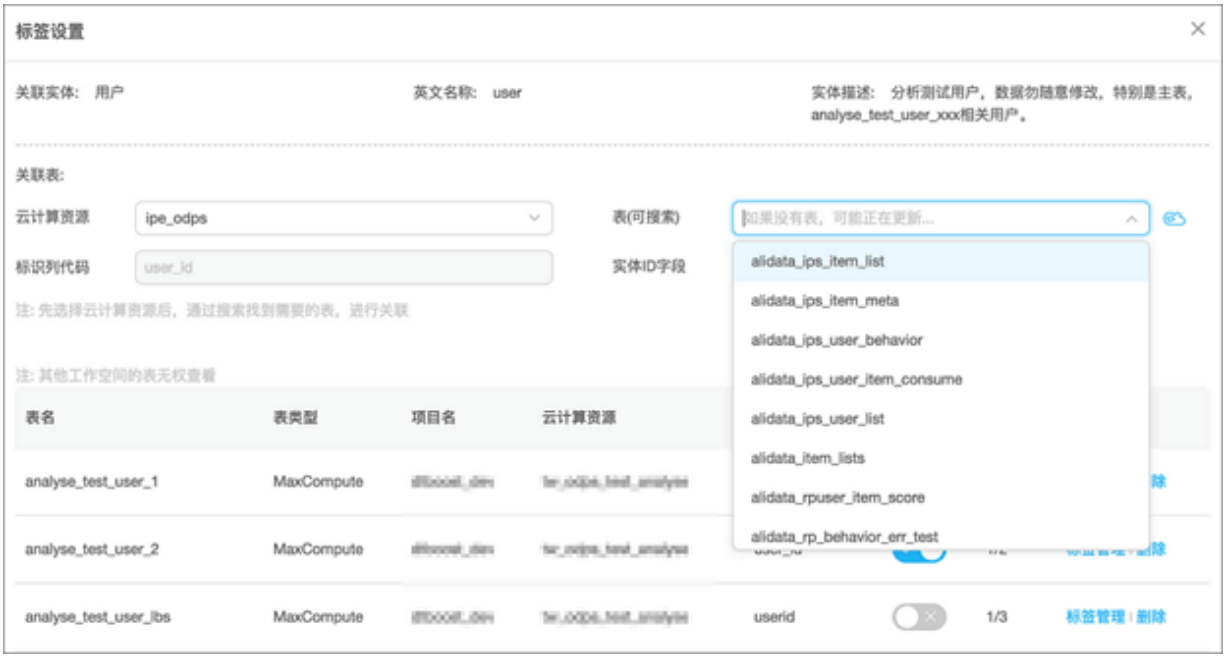
36.4.1.3.2 模型管理

实体关系管理

实体/关系管理是标签中心当中对逻辑模型进行配置的主要功能，能够读取不同来源的数据库的元信息，整合为实体或者关系。

描述同一个实体（主键）的多张表可以在逻辑层上聚合在一个实体下，形成一张大宽表（即原始表），如图 *36-6:* 实体关系建模示意所示。

图 36-6: 实体关系建模示意



关系的建立则是可以把联合主键表看作为关系，将多个实体关联起来。其余的描述字段则根据相应的情况定义为标签，如图 *36-7:* 关系定义所示。

图 36-7: 关系定义

标签管理

标签管理模块能够对所有的标签进行查看、检索和修改，如图 *36-8:* 标签管理和图 *36-9:* 设置标签所示。

图 36-8: 标签管理

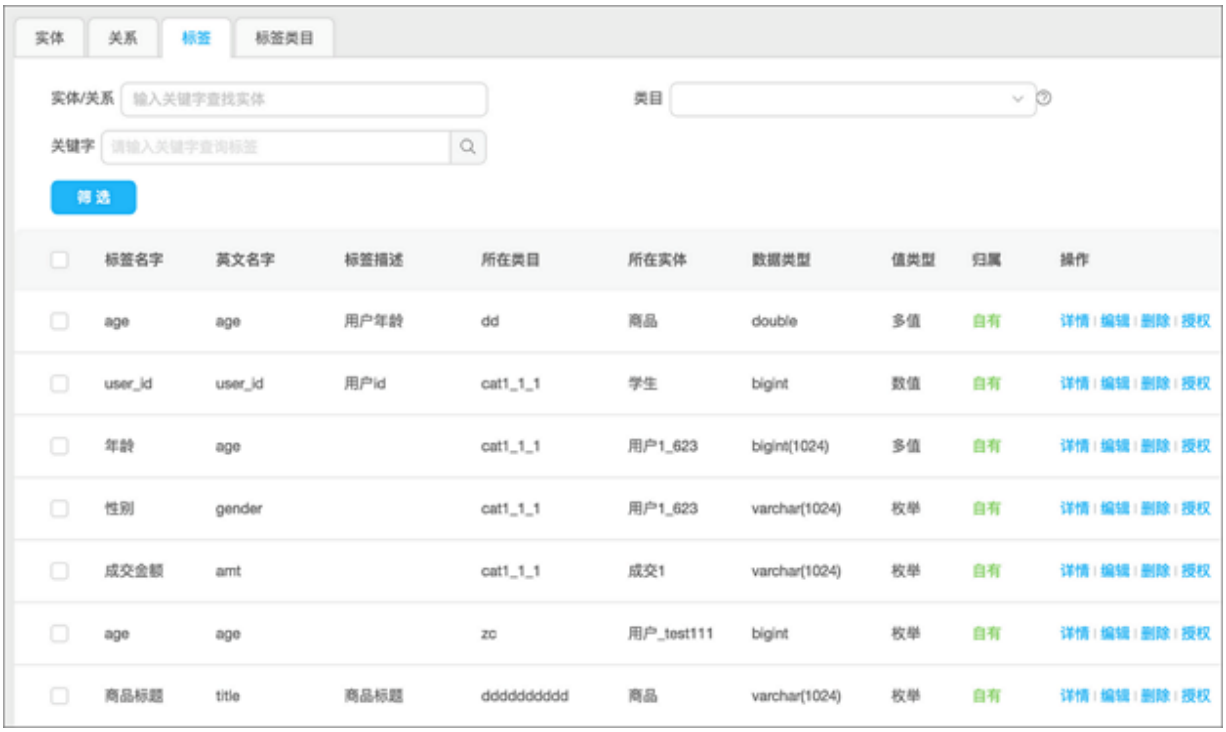


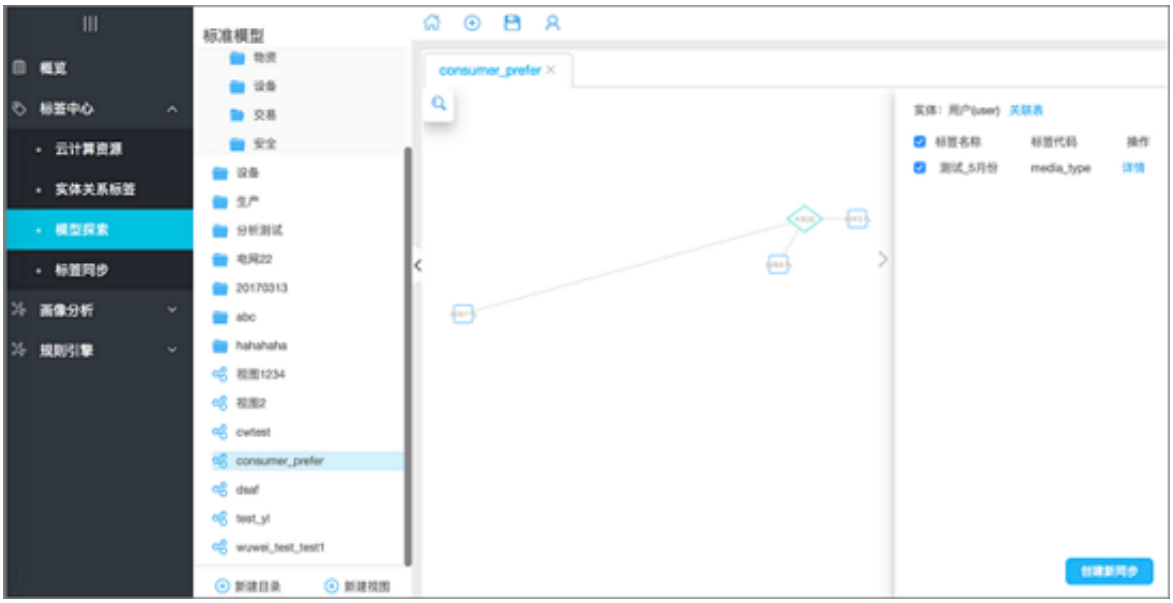
图 36-9: 设置标签

36.4.1.3.3 模型探索与数字订阅

模型探索部分可以通过关系图的方式查看所有的实体，实体与实体之间的联通关系及其属性，以及实体/关系下关联的标签情况。

如图 *36-10:* 实体关系模型探索所示，通过模型探索可以对整个标签模型进行全局的分析查看。

图 36-10: 实体关系模型探索



标签数据订阅是DTBoost处理跨计算数据流转的重要功能之一。在相应的数据服务需要使用到数据的时候，标签中心提供了将分散在多个存储当中的数据订阅至数据服务需要计算的位置的功能。对于同步且相应时间要求高的场景来说，需要您在相应的数据服务当中提前进行手工订阅操作。对于异步或者请求相应要求不高的同步计算场景来说，这个订阅过程是透明的。

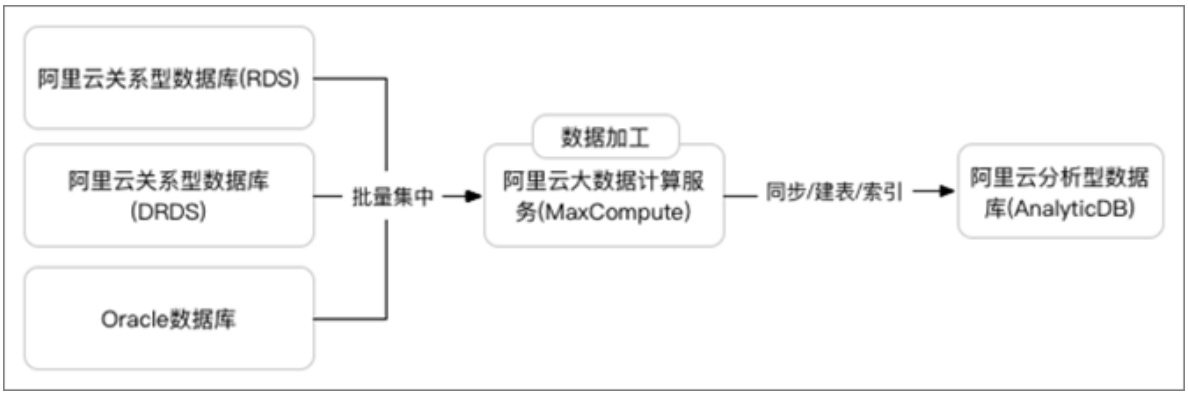
标签中心的技术架构如图 *36-11:* 标签中心技术架构所示。

图 36-11: 标签中心技术架构

智能搬运内置了针对几套典型的架构路径。

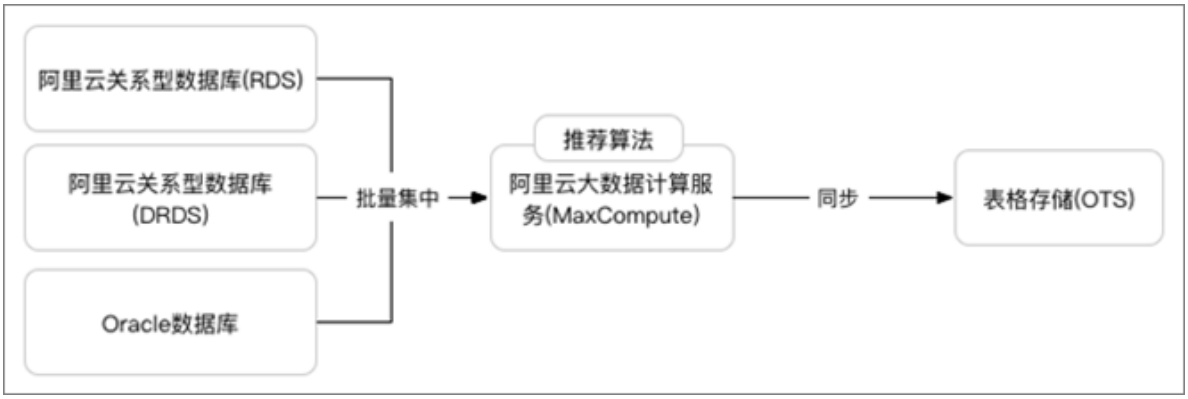
- 批量大数据在线分析，如图 *36-12:* 批量大数据在线分析所示。

图 36-12: 批量大数据在线分析



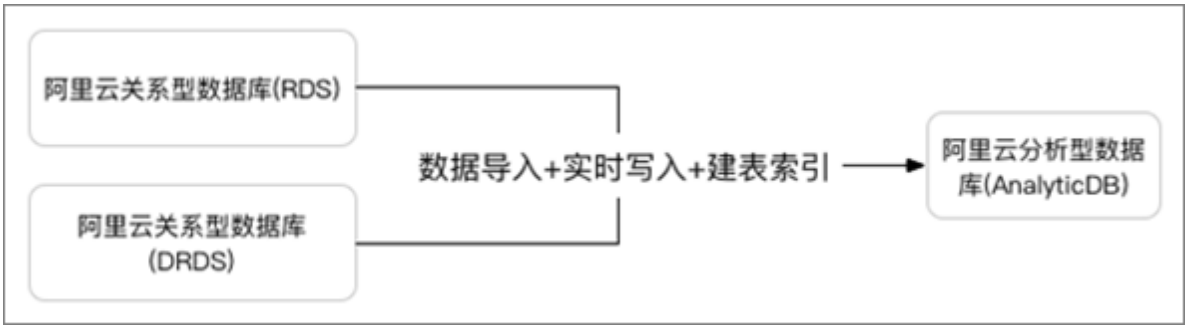
- 批量数据算法计算在线查询，如图 *36-13:* 批量数据算法计算在线查询所示。

图 36-13: 批量数据算法计算在线查询



- 实时大数据在线分析，如图 *36-14:* 实时大数据在线分析所示。

图 36-14: 实时大数据在线分析



对于整合分析这类OLAP/ADHOC场景来说，提供了将Oracle、关系型数据库（MySQL）等业务库中的数据同步至MaxCompute中，再订阅到所使用的分析库中，如阿里云分析型数据库（AnalyticDB）、关系型数据库（RDS）等。

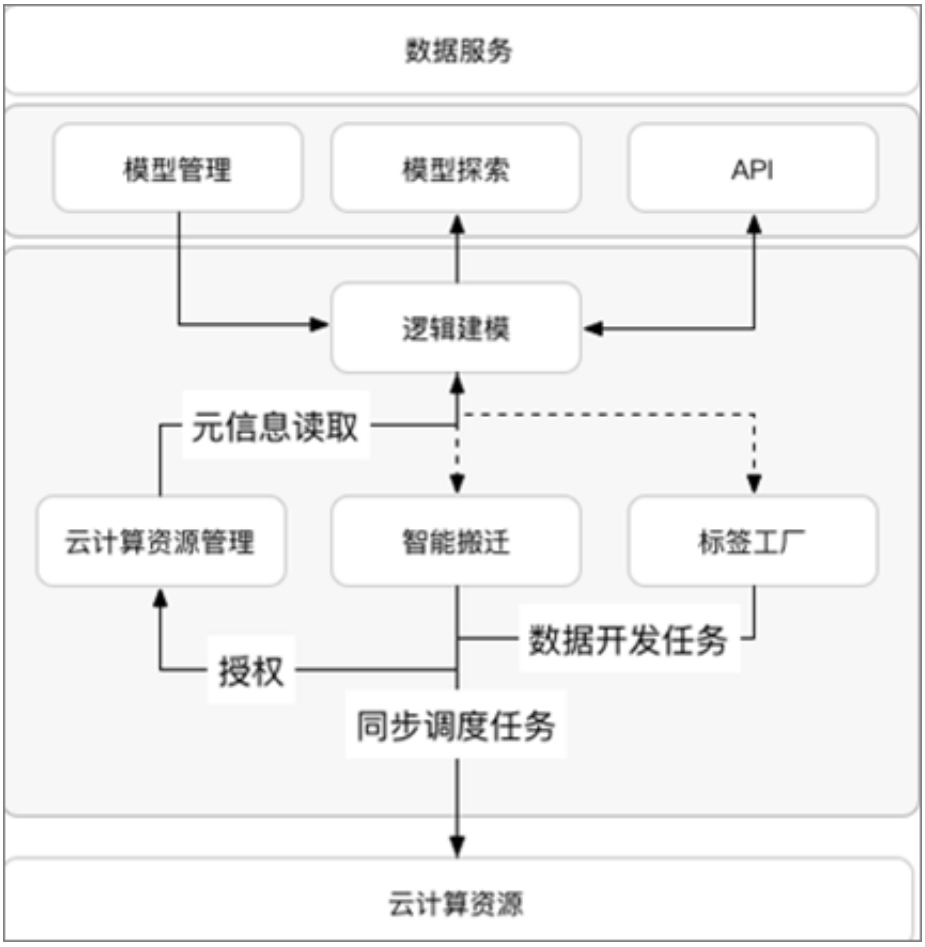
对于规则引擎这类流式计算的场景来说，提供了将离线数据、流式数据进行归并，将规则所需要的离线历史数据订阅至阿里云表格存储当中，并根据规则计算结果订阅至所需要的存储计算资源（MySQL/MaxCompute/AnalyticDB等）中。

对于目前尚未以标准方式提供的订阅路径，可以进行相应的定制。

36.4.1.4 技术架构

技术架构如图 *36-15:* 标签中心组件架构所示。

图 36-15: 标签中心组件架构



36.4.1.5 产品特性

对于这种数据体系的规划上来说，往往是由业务驱动的，是累积增加的，随着不同的业务板块的开展会逐渐纳入更多的数据源。如果按照传统数据仓库的做法会面临以下问题。

- 需要不断地在物理层进行数据表的归并，下层表的频繁变化可能会造成数据使用的不稳定。

- 当标签的需求越来越多，因为不可能无限制的在物理层将数据拼在一张原始表中，那分散的数据表也会越来越多，会造成检索和管理的困难。
- 在不同的应用当中不可能是整表使用，往往是需要多张表中的某几列，那多个应用不断的抽取再整合也会造成管理和检索的困难。
- 标签可能是实时数据、也有可能是离线数据，数据存储方式不同同样造成管理使用的困难。

标签体系建模和传统的BI分析建模相比，具有以下特性。

- **业务视角管理**

  围绕实体>关系>标签这三个元素进行建模，是从业务的角度出发对数据进行组织管理，而不是从表的概念出发进行建模，便于应用层对数据运用和管理的理解、操作，以近似于概念模型的形态透出，让人人都能看得懂。

- **跨计算的统一逻辑模型**

  传统建模的数据来源和模型的使用一般在同一数据库当中，而大数据环境下因为数据采集类型的多样性，和数据计算的多样性使得来源和使用分散在不同的计算存储资源当中，数据产生与加工首先就可能分布在不同的数据库当中，其次同一份数据需要进行跨流式、Adhoc类多维分析、离线算法加工等多种方式的计算，数据需要能在多个存储和计算资源当中自由流转。

  所以标签体系是把多个计算当中的拷贝在逻辑视图上进行唯一映射，即一个标签对应到多个计算当中的物理字段。

- **灵活拓展性**

  表/标签之间的逻辑关系的建立也是在逻辑层上完成的，这就使得模型的维护是可以动态建设的，便于模型的维护和管理，而无需在物理层将数据进行归并后再使用。每一个标签之间可以独立使用，这种离散的列化操作方式也使的数据的使用上更为灵活。

从另一方面来说，计算能力的增强和数据使用场景的丰富，更多的数据计算是需要直接作用在明细的行为数据上，而非只是对指标的多维统计。传统数据集市建模的**指标>维度**体系就略显狭窄。标签的定义上涵盖了多种数值类型，既可以是单列，也可以是维度＋标签组成的复合标签（这种方式通常用于描述某种行为），赋予应用操作上更大的灵活度。

36.4.2 画像分析

36.4.2.1 适用场景

数据服务是架设在标签视图之上的业务功能模块，可以通过界面化的配置或者API的操作能够以标签为粒度对跨计算资源的数据进行统一的业务计算操作。透过数据服务+逻辑建模的组合，既节省

工作量又有很好的扩展性。特别是对于大数据环境需要整合多个系统数据的前提下，很难一次把所有数据需求全部规划完整，那么这种动态逻辑建模的方式就有非常好的扩展性。

从应用的角度来说，由标签模型视图层隔开明细数据复杂的数据结构，能够在相对扁平的标签体系之上进行明细数据上计算和查询，对于数据开发与应用开发的分工流程上来说也更为合理。

图 36-16: 画像分析

画像分析作为DTBoost数据服务的其中之一，适用的场景主要是结合阿里云分析型数据库（Analytics DataBase），将您分布在多个存储资源的数据整合起来，在标签模型上构建大数据画像类的交互式分析应用，让您的业务人员可以自由灵活的分析这些对象各种属性与行为之间的关联性。可以广泛应用于工业设备画像分析、企业经营画像分析、用户行为画像分析等多个场景当中。

大数据画像类分析应用的特性如下所示。

- **基于行为等明细数据的分析**

  在过去以各项KPI指标计算为主要分析目的背景下，很容易把所有的指标计算提前构建。随着数据采集和使用场景的丰富，业务人员希望能够自由地分析各类行为明细数据，如查看不同客户属性在各个商品类目下消费的偏好和关联购买的情况，或者不同时间采购的不同类型、属性、地域设备的故障率与检修情况，还能够把多个维度细分下的具体客户/设备清单进行查看。业务人员进行的分析可能是任意维度之间的交叉关系，就很难进行预先的计算。

- **从半结构化数据中抽取特征**

从另一点来说，灵活分析还意味着能够与预测、评分、文本特征提取等算法技术相结合，进行广度与深度兼备的分析。往往很多的画像特征如抽象的兴趣，如喜欢动漫、爱美一族等风格兴趣偏好类的特征，通常需要通过算法从用户的点击、收藏、购买行为与相关物品的文本描述当中进行特征抽取。这就需要能够借助一些偏好计算、文本挖掘类的算法能够从这些半结构化的数据当中对用户互相的特征进行深度的挖掘。

- **交互式的查询分析**

  业务人员希望得到的分析是在数据当中探索有用的信息，如发现影响消费者购买的可能因素，或者故障设备的关联因素，这就需要能够根据不断调整的筛选条件、维度组合、下钻上聚能够快速返回结果，直到获取到足够多的信息。这就对查询速度的高响应提出了要求。

  在这种交互式的分析场景下也对整体界面的组织提出了要求，业务人员关心的是在不断探索中获得的数据洞察，如果还需要用户进行复杂的报表配置或者是数据结构/技术上的学习理解，就会大大影响数据探索发现的过程。各种数据的分析还需要与各种类型的可视化形态结合，除了常规的图表外，可能还需要各种尺度特别是城市内尺度的地图图表，表达拓扑关系的关系网络图表，以及能展示文本特征的图表。

从以上几点来看，交互式数据分析产品的开发变得非常有挑战性。

- 需要充分理解数据结构，才能把跨表查询的逻辑与界面交互进行有机的组织。
- 需要了解多个专业存储与计算资源的特性，把不同计算产出的结果组织到同一个分析界面中。
- 需要熟悉各类的可视化图表与分析控件的使用方法，结合到不同类型的分析中。

36.4.2.2 功能组件

画像分析提供了两大块的功能，分析服务层部分用户可以在控制台中完成。

36.4.2.2.1 接口调试

分析服务接口模块可以让您在此进行分析语句的调试和自助化封装数据分析接口。画像分析的查询表达都是建构在实体关系模型之上的。

您也可以对接口进行调试，查看查询的结果/执行错误、语法每一步解析所耗费的时间以及所解析的真实SQL语句，来帮助您调试分析接口。

36.4.2.2.2 界面配置

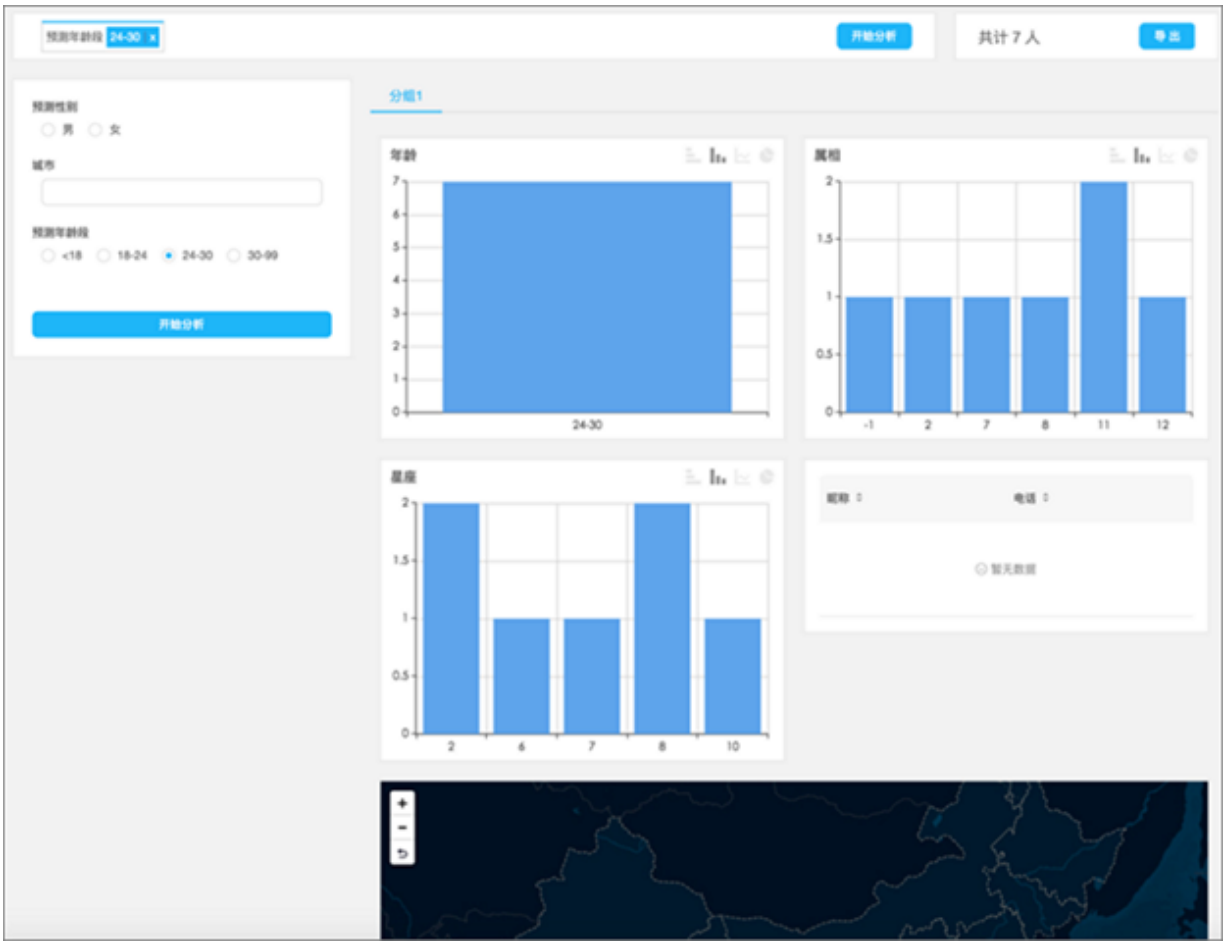
如图 *36-17:* 筛选条件配置所示，通过画像分析界面搭建工具，灵活配置交互式画像分析界面。对筛选出来的特定的分析对象进行多维透视，并进一步钻取分析，并可以将分析筛选出来的对象导出到

其他系统当中，如结合广告投放系统进行精准营销。整个界面的代码是完全开放的，可以与您的现有系统进行整合。

图 36-17: 筛选条件配置

如图 *36-18: 分析应用*所示，上层提供的交互分析应用框架是以源代码的方式提供，您只需把整个应用运行，会根据配置文件的填写自动渲染出一个可进行交互式分析的界面。您可以进行代码的修改进行样式的改造。多个配置文件可以通过配置从不同的URL进行路由，让不同的用户可以看到不同的分析应用。

图 36-18: 分析应用



36.4.2.3 典型应用

36.4.2.3.1 用户全景画像

在受众分析、CRM、用户行为和人口分析等场景下，通常需要对这些人群的明细行为数据进行分析。

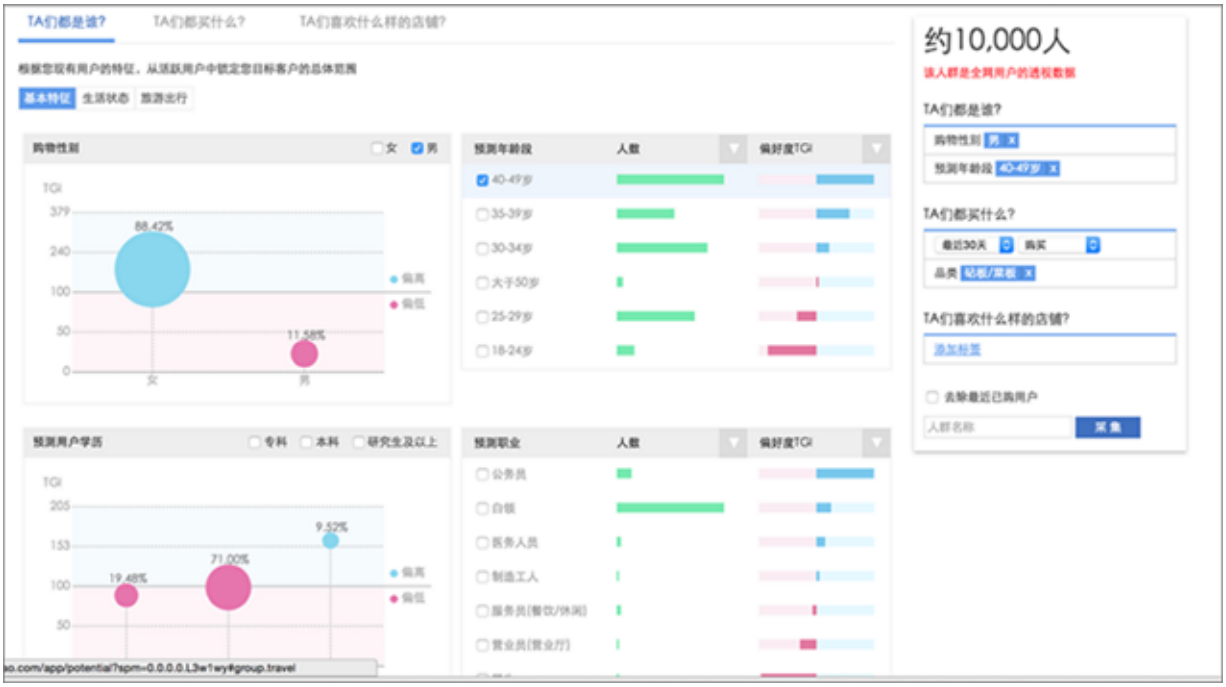
分析人员在进行分析时，不同的用户属性和用户行为之间会形成多种组合，所以无法按传统数据分析的建模方法提前预算好所有组合。

例如用户根据消费者行为选取想要分析的目标群体，如图 *36-19:* 选择目标群体所示。

图 36-19: 选择目标群体

对于选取的目标群体，分析其各项特征，发现分布密集的特征，如图 *36-20: 分析各项特征*所示。

图 36-20: 分析各项特征

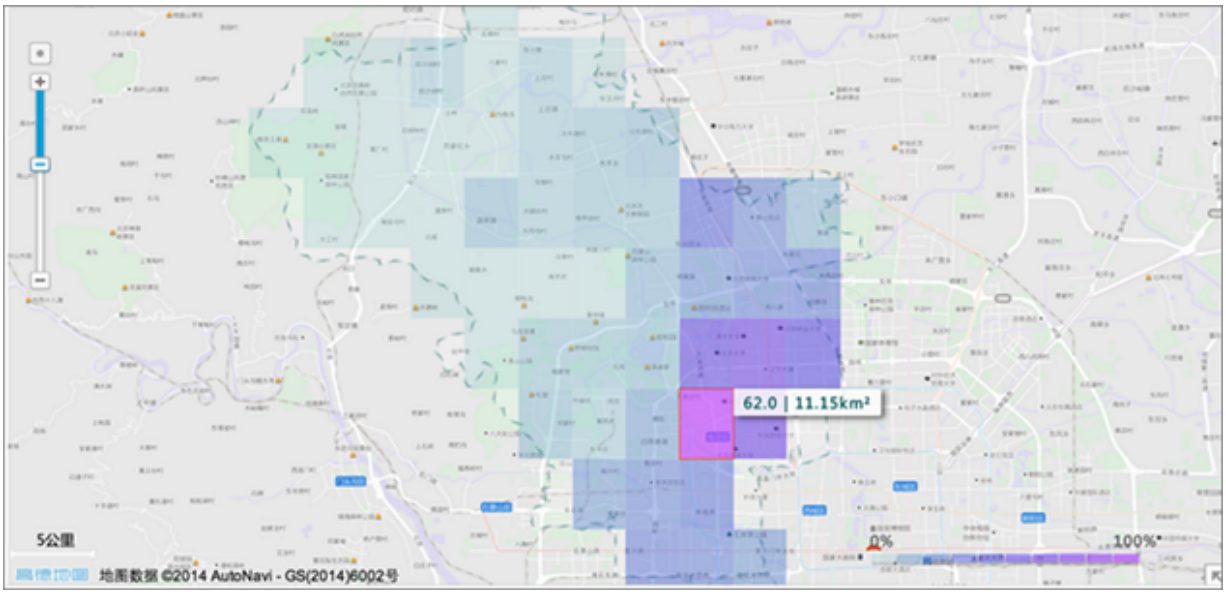


再根据这些典型特征，找到还未购买的群体，如图 *36-21: 找到未购买的群体*所示。

图 36-21: 找到未购买的群体

针对这类群体，对接下游系统，如图 *36-22:* 对接下游系统所示。

图 36-22: 对接下游系统



36.4.2.3.2 设备全履历

设备全履历如图 *36-23:* 电压等级、图 *36-24:* 地级供电公司和图 *36-25:* 设备明细所示。

图 36-23: 电压等级

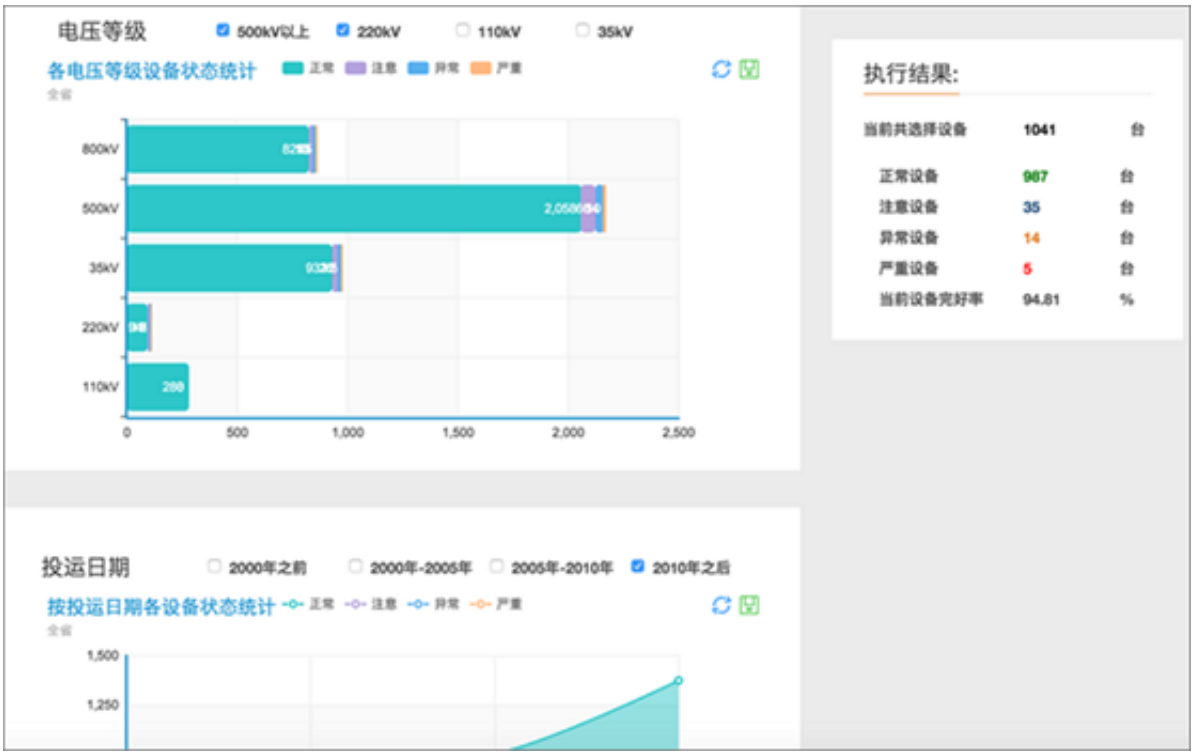


图 36-24: 地级供电公司

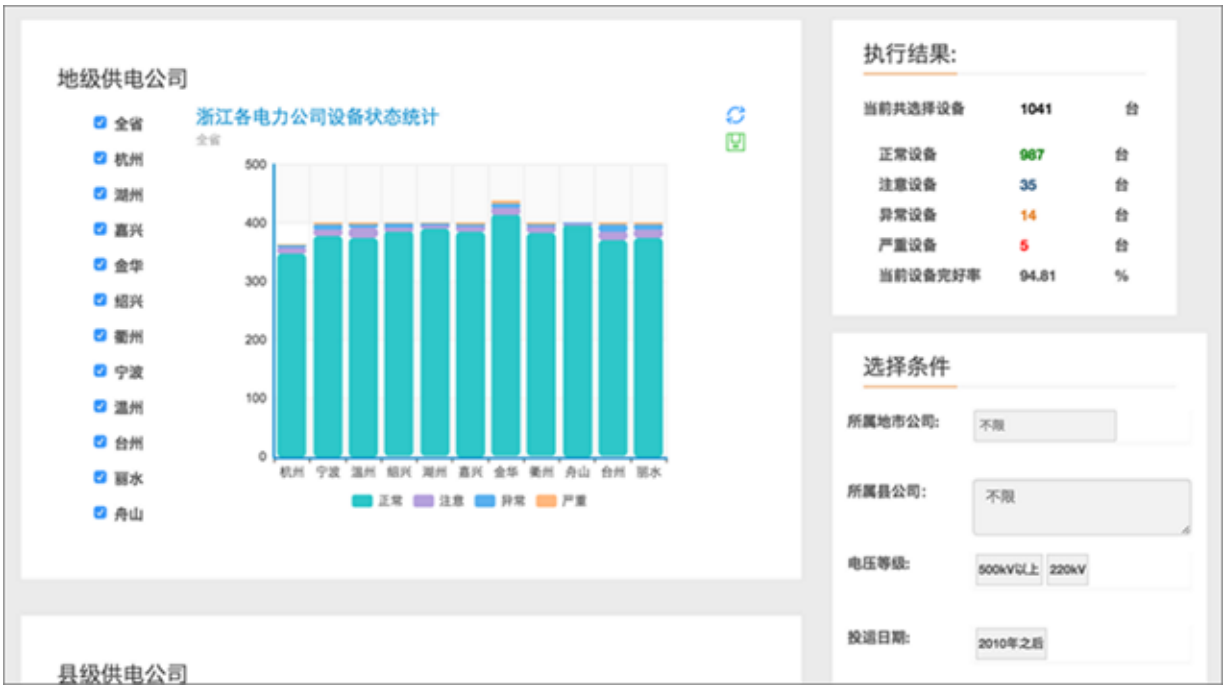
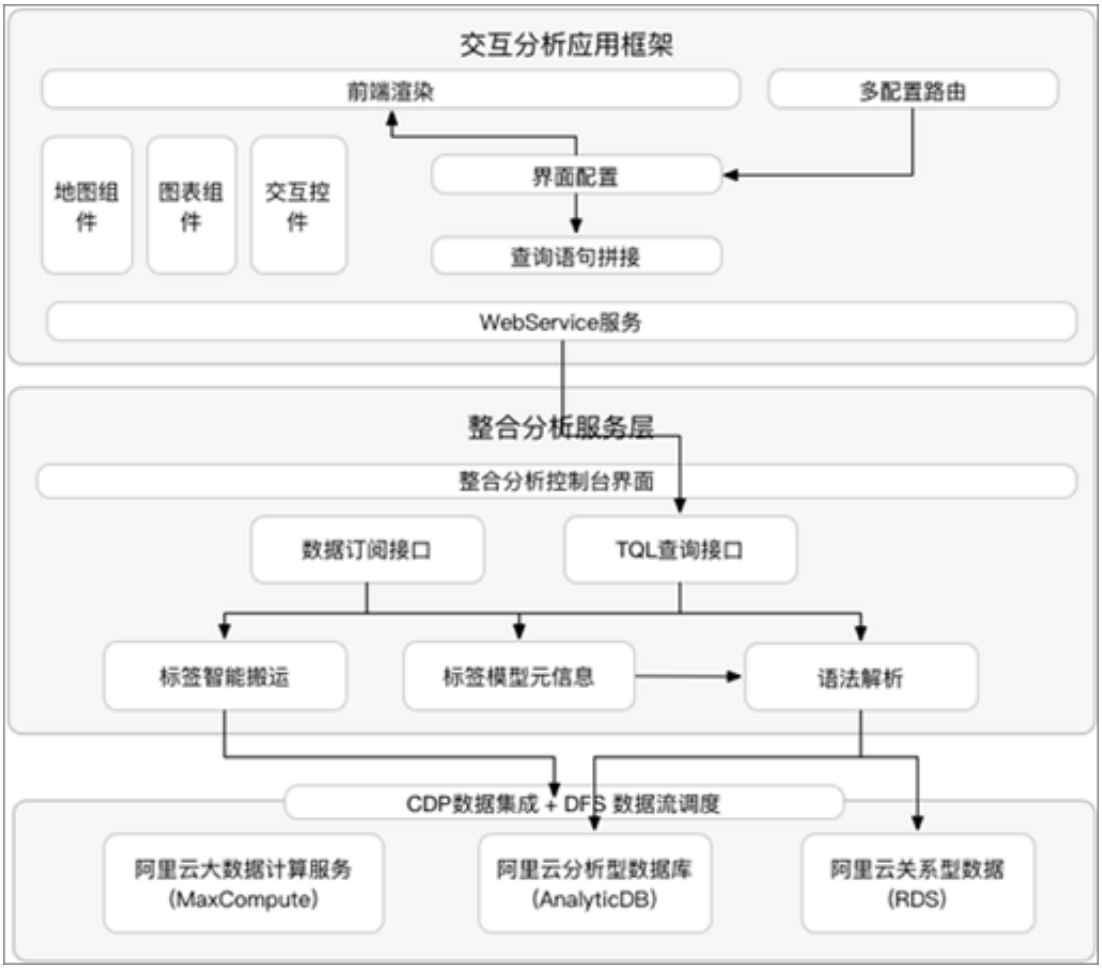


图 36-25: 设备明细

36.4.2.4 技术架构

画像分析的技术架构如图 *36-26:* 技术架构图所示。

图 36-26: 技术架构图



从技术架构来讲分为两个部分，在服务层部分通过TQL（Tag Query Laugage） 查询接口接收您输入的参数，结合标签模型中的元信息会进行真实SQL的语法解析，然后把相应的SQL传送给相应的计算资源进行计算，通过接口获得返回的计算结果。使用Debug模式同时返回所解析的真实SQL以及每一步计算所耗费的时长，可以用于优化相应的查询语句。此处的语法解析主要是把您在扁平化的标签模型视图上的查询逻辑，翻译为真实的多个表之间关联JOIN的查询。

数据订阅部分可以通过界面实现数据的一键搬迁，控制台界面会调用数据订阅的接口获取相应数据的元信息，调用标签中心底层智能搬运的接口，在相应的计算存储资源会自动建表建立索引后，触发调度任务进行数据同步。

在交互分析的应用框架层，会提供应用开发配置框架的源代码。其中包括相应的前端组件、WebService后端服务、界面配置文件以及根据界面配置文件配置的查询接口。同时配置文件还能够进行相应的路由配置，让不同的页面URL可以路由到不同的配置，让不同的人看到不同的界面。

36.4.2.5 产品特性

DTBoost的特性如下所示。

- **一键数据整合**

  针对不同的分析主体，您可将多个存储资源当中的多个标签，通过一键数据整合功能同步至在线查询数据库中，并可进行索引操作，像管理一张表一样管理不同的数据源。兼容的在线分析数据库既可以是阿里云分析型数据库（Analytics DataBase），也可以是阿里云RDS关系型数据库。

- **Web开发友好**

  Web应用开发者直接通过与画像分析查询和标签元信息API接口的交互，结合阿里云DataV或是其他图表组件，即可快速搭建自己的分析型数据产品。

- **查询表达简单**

  在扁平化的标签体系上，一定程度简化了表关联和子查询的表达，让Web应用开发人员更加关注在应用逻辑而非数据表的组织逻辑。查询参数提供JSON对象模式，也提供与SQL相似的TQL模式。

- **与DTBoost其他模块无缝结合**

  由于标签体系下，多个模块之间共享同一个标签视图，以及同一个标签在不同的存储计算资源能够自动搬迁，使得画像分析能够与DTBoost的算法模块、特征工程模块、实时预警模块产出的数据有一致性的表达，相互打通无缝结合。

- **交互式分析应用框架**

以SDK代码的方式提供分析界面配置工具，即刻生成交互式的分析应用。相比传统的BI工具，配置出来的分析界面像一个独立的交互分析产品，可以整合到您整体的分析系统当中，更容易让您对实体的属性、行为、地理出行等进行准确的分析。

36.4.3 标签工厂

36.4.3.1 概念说明

标签工厂设计思路是进行数据的再次开发，开发出来的数据落地后可以继续二次开发，不断满足不同的业务场景需求。设计中根据原有OLT标签体系中的标签数据进行再加工生成新标签数据并落地到OLT标签体系。

36.4.3.2 功能简介

计划

计划（scheduler）是用户创建的每个生产标签的任务，是指根据原有OLT标签系统数据通过SQL、TQL或算法生成新的标签数据并关联到对应的OLT模型的一条记录内容。它包含有原有的OLT标签、SQL或TQL或算法部分、以及新产生标签数据并关联到对应的OLT模型的关系映射。

目前支持图形界面配置计划任务、文本直接写TQL配置计划任务、算法配置计划任务三种形式。

计划通过提交、执行就可以将产生的新的标签数据落地到OLT标签系统。

任务

计划执行之后，就会在任务列表里面有对应的记录，任务就是计划的执行记录。

标签工厂任务列表展示了计划的执行人、提交时间、开始执行时间、执行结束时间、执行状态，同时支持执行参数和日志的查看、执行任务根据执行日期和计划名称（ID）进行全局搜索等功能。

36.4.4 规则引擎

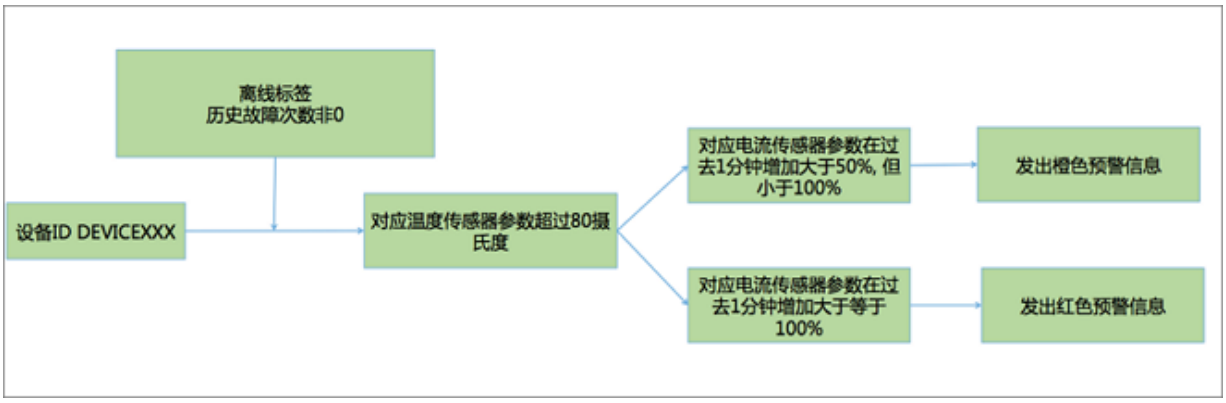
36.4.4.1 概念说明

基于流计算的规则引擎适用于业务决策的实时评估，包括风险拦截和预警、资源分配、流程改进等，特别适用于在状态或行为数据频繁变化的场景下，针对特定的业务目标进行实时决策。

规则引擎实时处理数据，当自定义规则命中时、或当算法模型捕捉到业务数据中的特定模式时，实时返回结果。

规则引擎核心的概念即规则。一个可能的规则结构如图 *36-27:* 规则结构展示所示。

图 36-27: 规则结构展示



规则通常来说是一个if...then...结构，即条件与行为构成了一个完整的规则。在上述示例中，包括如下几个条件。

1. 该设备ID为DEVICEXXXX
2. 该设备历史故障次数非0
3. 该设备对应温度传感器参数超过80
4. 对应电流传感器参数在过去1分钟增加大于50%小于100%
5. 对应电流传感器参数在过去1分钟增加大于等于100%

而规则的行为如下所示。

- 发出橙色预警信息
- 发出红色预警信息

当满足条件1、2、3、4时，发出橙色预警信息。当满足条件1、2、3、5时，发出红色预警信息。同时，规则条件的组成部分包括数据与对应计算逻辑。

在规则引擎中，数据在物理上就是一个个的表，被抽象为DTBoost中的标签。计算逻辑是基于数据需要执行的计算流程，在规则引擎中被抽象为Function，比如最简单的大于小于等于，复杂一点的可以是跳变，多次跳变等。

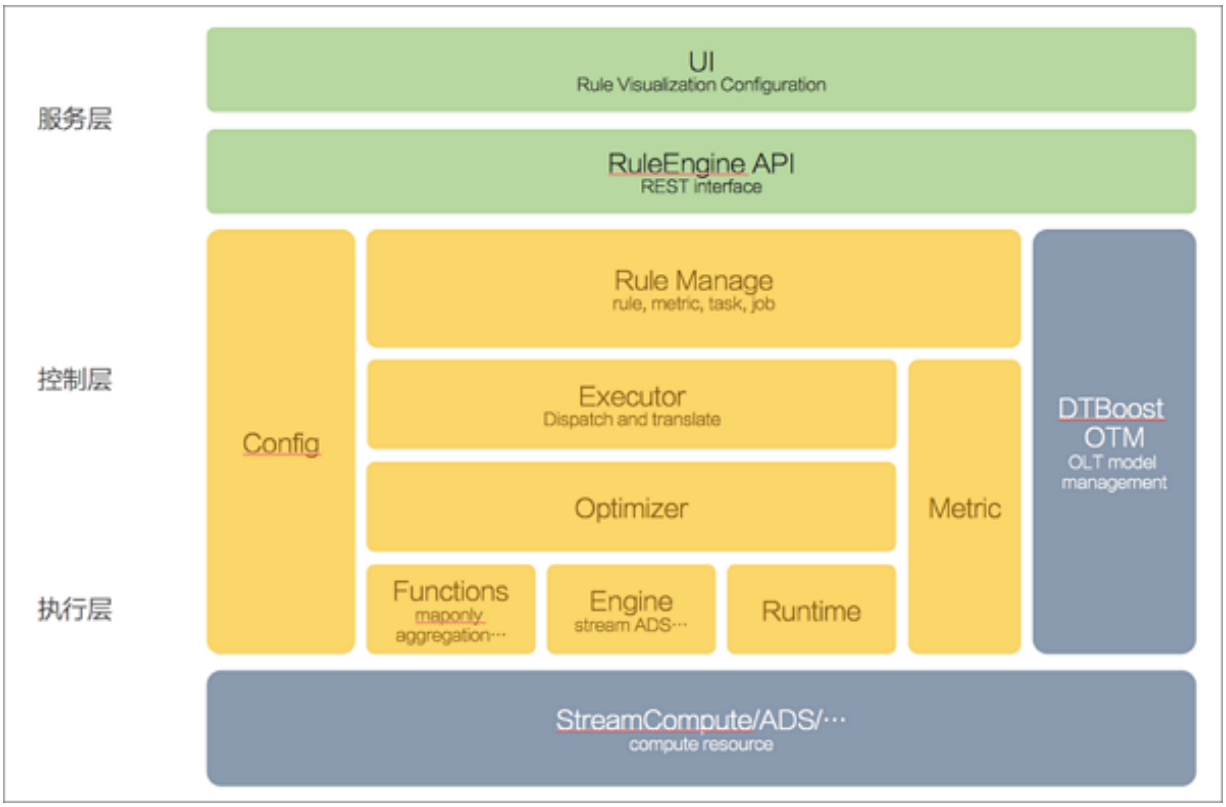
实际规则的执行，是使用标签所描述的数据，去放到规则配置的Function 中进行执行，对得到的结果进行判断。

36.4.4.2 技术架构

36.4.4.2.1 功能模块

规则引擎的功能模块拆解，如图 *36-28:* 规则引擎功能模块所示。

图 36-28: 规则引擎功能模块



规则引擎整体分为三层。

- **服务层**

  服务层包含UI，提供了基本的规则管理功能。UI模块下的API模块提供了一组RESTful风格的API。规则引擎同时可以通过UI和API向外提供服务。您可通过UI访问规则引擎，也可通过API完成系统间的调用。同时，如果您有深度定制的需求，规则引擎的API模块提供了完备的功能供您基于不同行业需求进行二次开发。

- **控制层**

  控制层包括对规则管理，执行，优化等功能，并且提供Metric采集功能，能够查看所有规则运行相关Metric。

  说明：
  Metric是一种运维统计指标，可统计每个规则的输入数据量、产出预警量、延迟情况、处理速度等信息。

- **执行层**

执行层包括了规则实际提交到计算引擎，以及运行态的计算逻辑。此处提供了Function 扩展模块，可以通过对Function进行扩展，以支持新的判断条件。

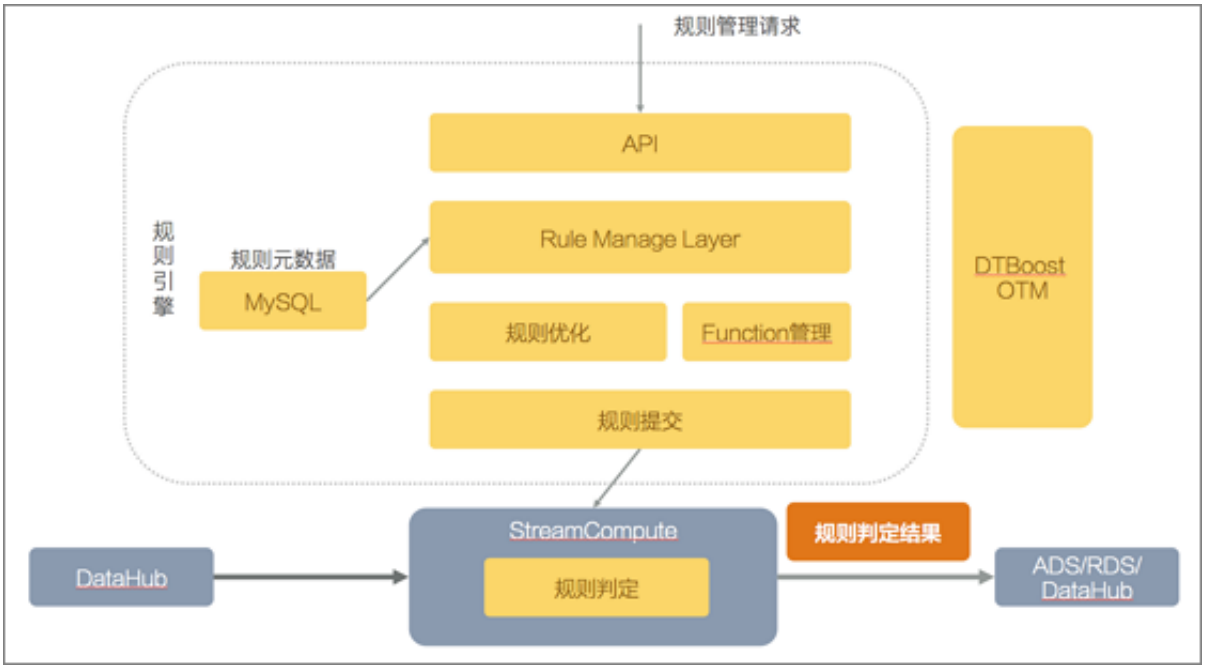
最底层的灰色模块为外部的计算引擎。

侧面灰色的OTM（Open Tag Management）模块是DTBoost所提供的标签管理模块，用于对云计算资源、实体关系以及标签进行管理。

36.4.4.2.2 规则提交流程

规则引擎中规则的提交执行流程，如图 *36-29:* 规则提交流程所示。

图 36-29: 规则提交流程



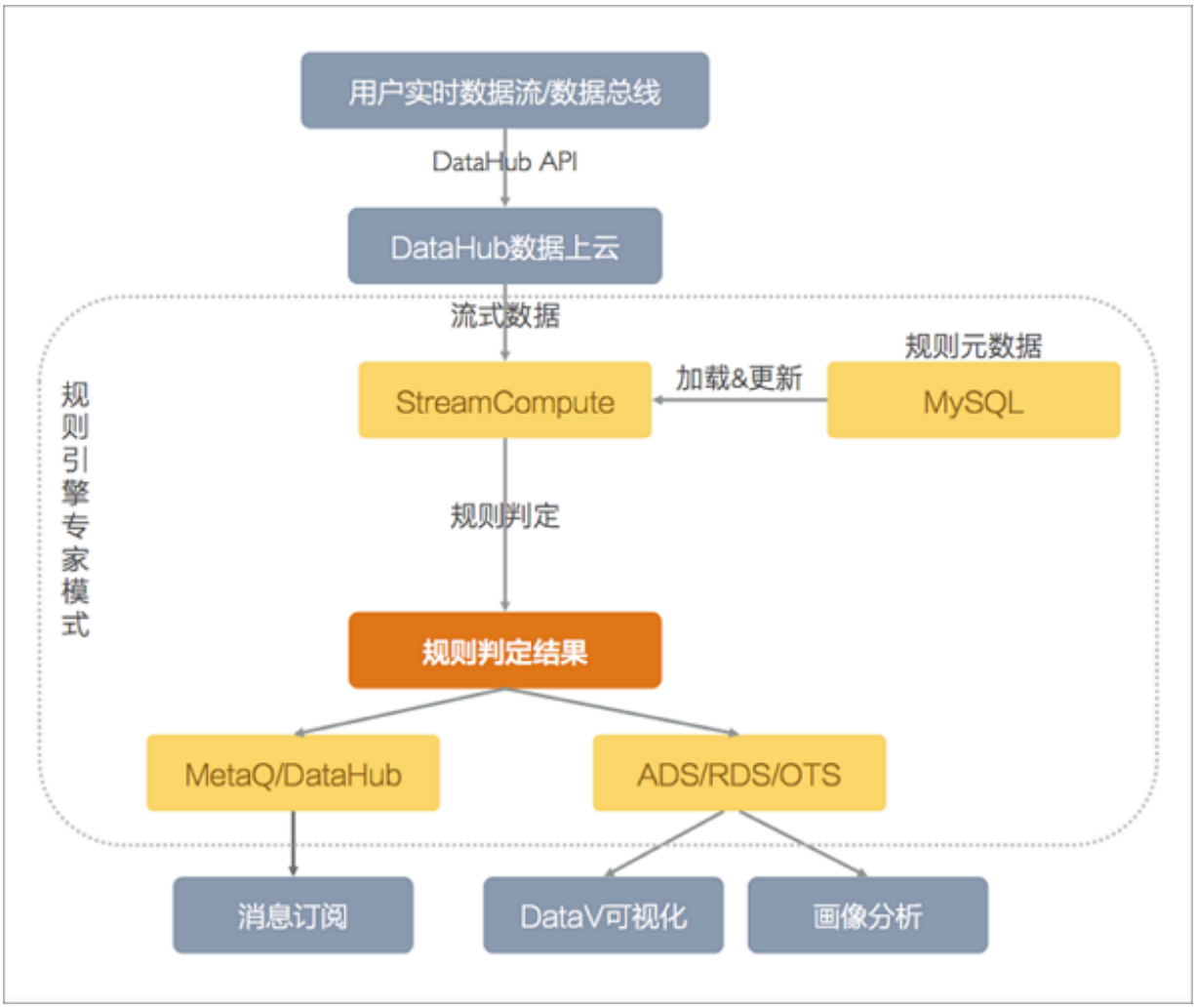
规则通过管理模块进行元数据注册，然后进行翻译，优化，并转化为对应function逻辑。同时，根据规则配置的计算引擎类型，提交到对应的计算引擎，作为计算引擎中的作业进行执行。

在规则执行过程中，不断消费上游发来的实时数据，进行规则function执行，并输出规则判定结果到对应下游目标存储中。

36.4.4.2.3 规则运行流程

规则运行流程，如图 *36-30:* 规则数据流所示。

图 36-30: 规则数据流



36.4.4.3 产品特性

- **拥有海量实时数据流处理能力**

  规则引擎使用了高效的流计算引擎，规则所对应的数据量每秒可达到上万条，规则所对应的数据量每秒可达到上万条。例如在工业场景中，如果有上万台设备，每个设备上有数十个传感器，若在规则引擎中进行规则配置，基于传感器数值对设备进行监控，可以支持每个传感器每隔1秒上传状态数据。

- **支持十万级别规则并发**

  规则引擎支持十万级别规则的并发执行。例如在电商行业中，若用户出现违规行为，需要实时告警。告警出现延时则可能让欺诈得逞，带来钱款的损失。对于这个场景，需要由数十名小二制定数万条规则。由这些规则同时对海量电商用户进行监控，发现其中的违规欺诈行为，每个用户每时每刻的行为都将接受数万规则的检测。

- **支持时间划窗、时空轨迹等流式计算复杂规则**

规则引擎支持基于时间窗口的规则计算，可以在一段时间窗口内进行统计或计算。例如，判断设备温度在1小时内升高是否超过100%，或判断潜在用户在过去3天购买兴趣是否包含3C数码。

同时，规则引擎还支持判断预设的多个条件是否按照预设顺序被触发。例如风险卖家先提取了全部现金，再拒绝了全部退款申请。

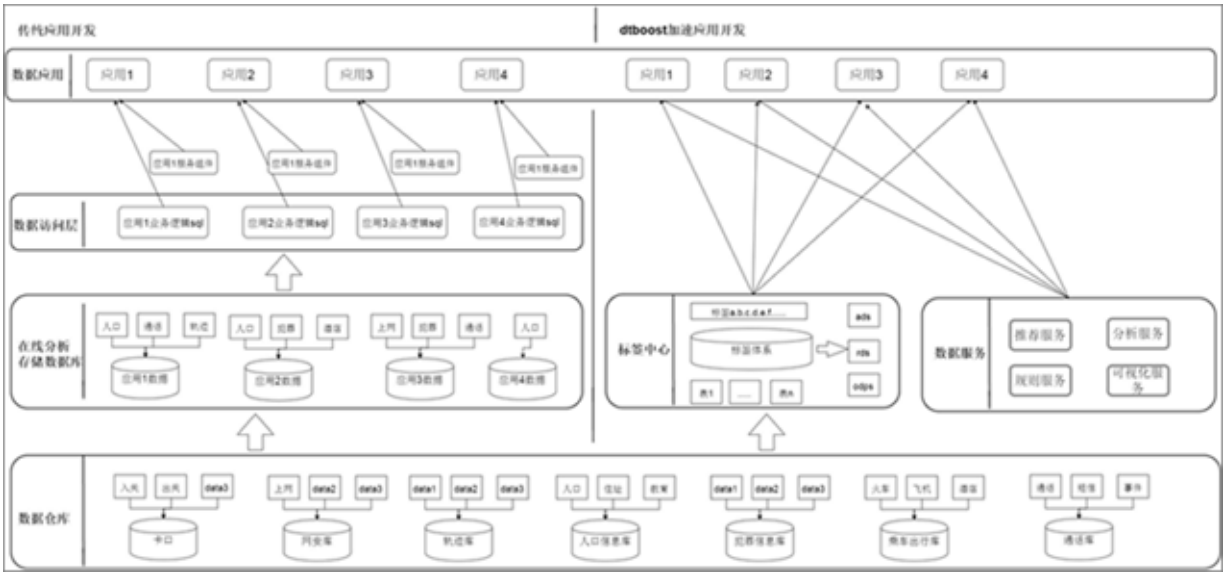
- **支持规则热升级，升级前后状态数据不丢失**

规则引擎中的规则通常按照专家的经验进行配置，然后根据实际执行的结果或实际的数据进行调整。规则引擎提供了规则配置热升级的功能，可以确保在规则参数修改后，规则执行的中间状态不会丢失，对于包含时间窗口的规则尤为有用，同时也可有效避免规则上下线过程中的漏报。

36.5 应用场景

您除了通过控制台对各个模块进行配置操作外，各个模块从数据元信息到数据服务的操作处理都可以透过开放API整合入自己的应用系统当中。这种服务化的方式一方面提高了系统整合的便利性，另一方面也对企业数据应用管理上提供了便利。

图 36-31: 加速开发流程



如图 *36-31:* 加速开发流程所示，从企业IT架构上来看，IT或者数据部门可以通过DTBoost以数据服务化的方式把计算资源、数据资源、数据计算方法打包在一起，提供给业务部门开发、外部合作伙伴。一方面对应用开发者来说即开通即使用，方便快捷。另一方面从IT部门来说，对于平台的资源管控更加有效，一定程度上降低了数据的冗余存储与加工，特别针对于业务算法、消费者画像这些需要使用到明细数据计算的场景，既能够使用到明细数据，又不会影响到原始数据的生产，不造成大数据量的冗余拷贝，还能够降低数据使用的门槛，提供了有力的支撑。

37 大数据管家BCC

37.1 什么是大数据管家

大数据管家Big Data Manager（原名BCC）是为大数据产品量身定做的运维管理平台，当前运维的产品包括MaxCompute（原名ODPS）、DataWorks（原名大数据开发套件）、AnalyticDB（原名ADS）、StreamCompute、BigGraph（在线图计算服务）、Quick BI、IPlus（关系网络分析）、Apsara（飞天操作系统）、OSS（对象存储）、ES（ElasticSearch on MaxCompute）、ElasticSearch、DataPhin（智能数据引擎）、DataHub（大数据总线）、PAI（机器学习）和AIMaster（算法服务敏捷平台）。

大数据管家支持对所有产品进行健康监控和资源监控，支持查看所有产品和产品子服务的指标信息、Metrics信息、服务器信息、告警和进程错误信息等，方便用户快速掌握产品运行状态。同时，大数据管家还支持对各产品进行软件升级，以及对自身的监控项进行热升级。

除以上公共功能外，对于部分产品和 产品子服务，大数据管家还提供以下功能：

产品名称	支持功能
AnalyticDB	• 支持查看资源概览、数据库信息概览、空闲机器，以及查询meta/sysdb。 • 支持查看slave详情。 • 支持日志搜索。 • 支持查看表详情、表使用情况概览、查询分析、节点信息、导入任务列表、资源变更历史、版本变更历史。 • 支持查看节点信息。 • 支持创建业务运维任务。
Apsara	• 支持查看master详情、slave详情。 • 支持日志搜索。
BCC	• 支持查看BCC拓扑结构图、配置。 • 支持日志搜索。 • 支持查看各产品的配置。
Dataworks	• 支持应用参数调整。 • 支持日志搜索。 • 支持BaseBizAlisa资源管理和Gateway管理。
MaxCompute	• 支持查看MaxCompute的剩余存储空间、可服务天数。

产品名称	支持功能
	• 支持参数调整。 • 支持项目空间管理。 • 支持查看实例信息、quota信息。 • 支持创建业务运维任务。
StreamCompute	支持创建业务运维任务。

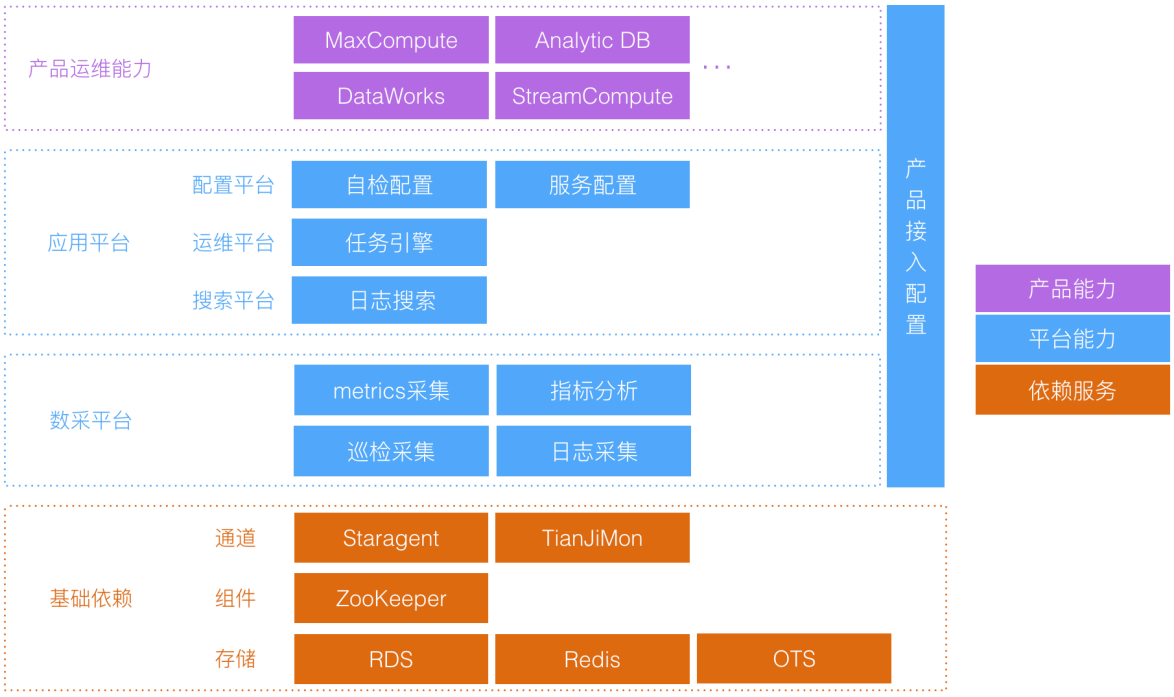
通过大数据管家的赋能，专有云驻场人员可以轻松地管理大数据产品，比如查看大数据产品的运行指标、修改大数据产品运行配置、对大数据产品进行自检、搜索日志等功能。

37.2 产品架构

大数据管家采用微服务设计理念，在此之上提供数据整合、接口整合以及功能整合的统一整合平台，暴露统一标准的服务接口。基于这样的设计理念，在大数据管家上的不同产品的页面操作和运维体验完全一致，减轻现场运维学习成本，降低运维风险。

整个系统主要由基础依赖、数采平台、应用平台、以及产品运维能力组成，如图 *37-1: 产品架构*所示。

图 37-1: 产品架构



37.2.1 基础依赖

大数据管家采用了集团统一的通道服务staragent和TianJiMon来完成远程命令和远程数据采集指令，通过zookeeper来协调主从服务，保证服务可用性。在存储方面，大数据管家主要使用RDS来存储元数据，使用redis作为缓存服务，使用OTS来存储大量的自检采集信息，提升吞吐量。

37.2.2 数采平台

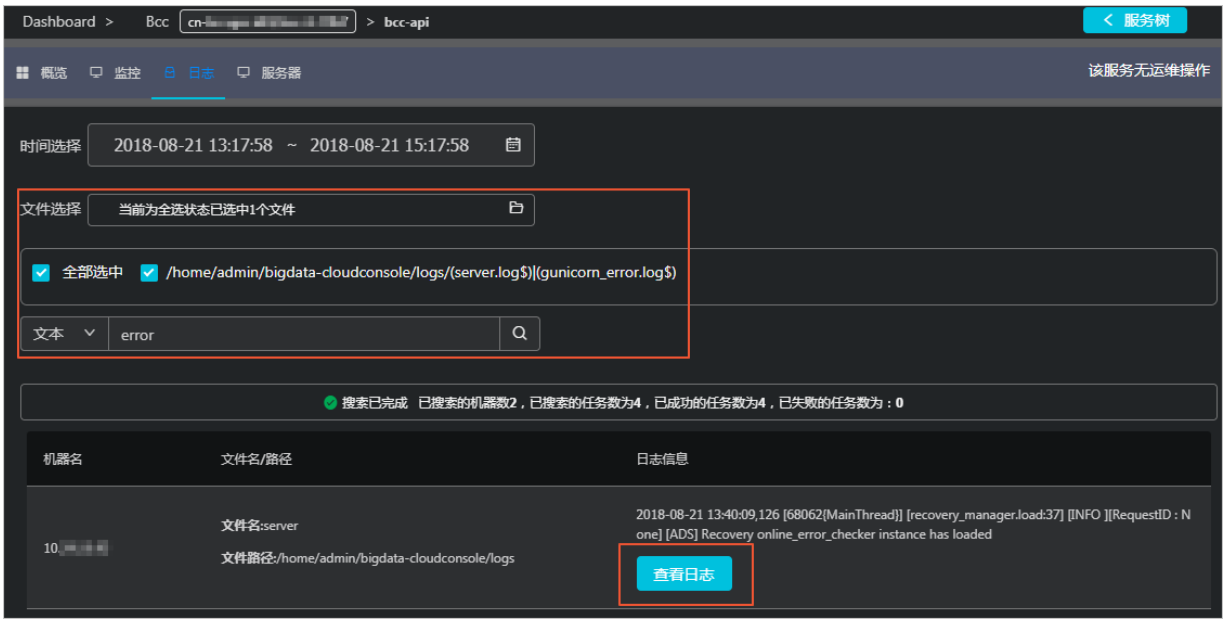
依赖TianJiMon，大数据管家开发了针对实时运维数据采集的采集脚本和针对服务状态采集的巡检脚本，通过回收这些脚本采集的数据，可以快速了解服务整体运行数据和状态，通过一定规则的聚合、筛选和甄别发现有价值信息。日志采集则是依赖staragent完成实时采集任务，让您可以实时分布式抓取日志信息。

37.2.3 应用平台

应用平台根据底层提供的能力以及数据采集的数据信息，构筑出多个提供上层服务能力的平台，这里主要包含配置平台、运维平台以及搜索平台。

- 配置平台可以方便的让您了解指定服务的所有配置信息，主要包含自检的配置和服务本身的配置，并且这些配置可以在这里修改提交。
- 运维平台主要提供运维的能力，并且可以智能识别您当前所在的服务而动态填写参数。
- 日志搜索则提供了针对指定服务的指定日志路径的实时日志搜索能力，如图 37-2: 日志搜索所示。

图 37-2: 日志搜索



37.2.4 产品运维能力

配置平台、运维平台以及搜索平台并不意味着就是所有大数据管家上提供的产品运维能力，因为部分跟产品有关的定制化运维功能则依托在每一个产品运维能力本身上。他们都会深度的与具体需要运维的产品融合和定制，达到一致性和特殊性的统一。例如AnalyticDB 会在日志平台的能力上，再提供更高层次封装的针对其服务的查询分析，并结构化展示分析数据，让您可以快速了解日志信息。这些定制都是透明的，您在具体运维场景上会顺利地了解这些定制内容。

37.3 功能特性

37.3.1 层次化服务管控

大数据管家主要使用层次化服务树来分解应用，也同时引导您了解被运维产品的特性，如图 *37-3:* 服务管控所示。

图 37-3: 服务管控

层次化服务树主要是对被运维应用的一种树形产品模块梳理，这种梳理可能面向服务组件梳理，也可能面向业务场景梳理，梳理出来后，每一个层次上的节点我们都称之为服务。

在这个树里您可以轻松的发现这个服务的父服务，以及它所依赖的子服务。对于服务本身的所有操作，都可以在这个服务的页面上找到。

服务树主要有以下几个特性。

- 产品树里面的服务可配置并实时动态更新。
- 支持跨产品依赖。
- 飞天、机器信息统一配置并快速接入。
- 按照产品树做信息展示和信息汇聚。

37.3.2 自我管控

大数据管家有自我管控能力，可以在应用平台层或底层依赖服务出现问题时显示问题定位和原因，方便您快速恢复大数据管家。

37.3.3 管控服务列表

大数据管家下管理的产品及它们的原名如下表所示。

产品名称	原名
MaxCompute	ODPS
AnalyticDB	ADS
DataWorks	Base
DTBoost	-
IPlus	I+
StreamCompute	-
PAI	-
Quick BI	-
Apsara	-
Big Data Manager	BCC
OSS	-
ES	-
DataHub	-
DataPhin	-
AIMaster	-

37.4 故障处理

常见故障处理

- **登录故障**

如出现无法登录的情况，请先清理浏览器的缓存、cookie，重新尝试登录。

如果登录框提示登录异常，请根据提示异常检查以下内容。

是否密码不对

是否账号锁定

是否账号被关停

- **其他异常**

  请寻找技术支持。

38 E-MapReduce

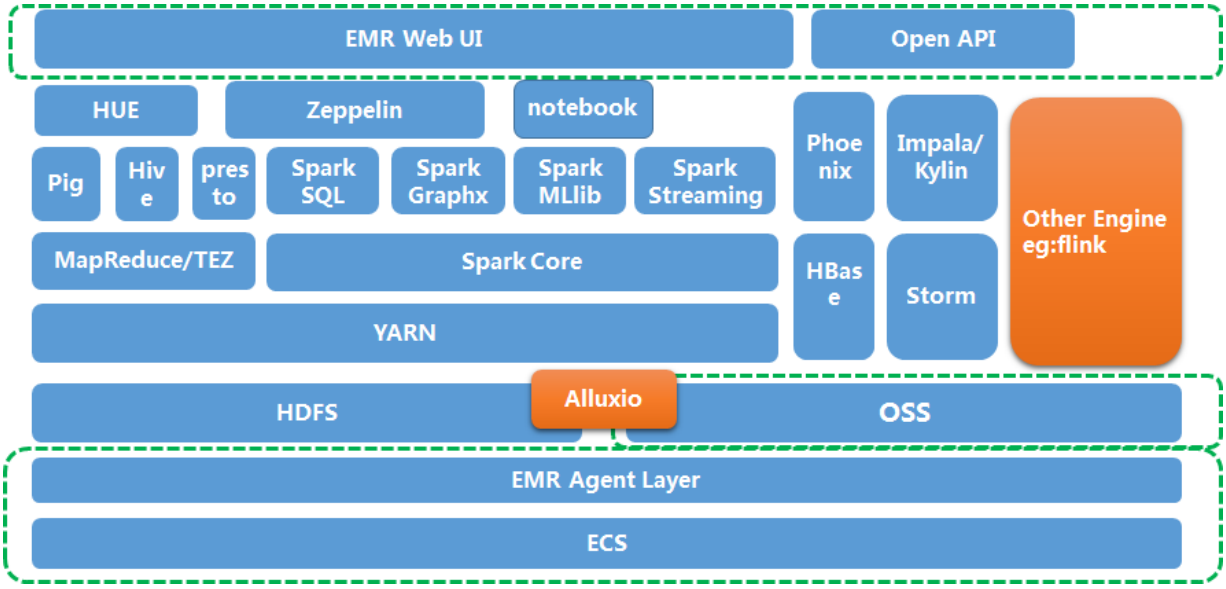
38.1 产品概述

Elastic MapReduce（以下简称E-MapReduce ）是一套利用开源大数据生态系统，包括Hadoop、Spark、Kafka、Storm，为用户提供集群、作业、数据等管理的一站式大数据处理分析服务。

38.2 产品架构

E-MapReduce的产品架构如图 *38-1: E-MapReduce*的产品架构所示。

图 38-1: E-MapReduce的产品架构



E-MapReduce集群基于Hadoop的生态环境来搭建，构建于阿里云ECS之上，同时可以跟阿里云的对象存储服务（OSS）、云数据库（RDS）等云服务进行无缝数据交换，方便您将数据在多个系统之间进行共享和传输，以满足不同业务类型的访问需要。E-MapReduce开放了一系列OpenAPI，方便您对集群、作业和执行计划进行调用操作。

关于Hadoop生态中各个组件的介绍，请参见基本概念。

38.3 功能模块

38.3.1 集群

一个E-MapReduce集群是由一个或多个阿里云ECS instance组成的Hadoop和Spark集群。

E-MapReduce提供了集群管理工具的集成解决方案，如主机选型、环境部署、集群搭建、集群配置、集群运行、作业配置、作业运行、集群管理、性能监控等，用户可以从集群构建各种繁琐的采购、准备、运维等工作中解放出来，只关心自己应用程序的处理逻辑即可。

E-MapReduce还给用户提供了灵活的搭配组合方式，用户可以根据自己的业务特点选择不同的集群服务。例如，如果用户的需求是对数据进行日常统计和简单的批量运算，则可以只选择在E-MapReduce中运行Hadoop服务；而如果用户还需要流式计算和实时计算的需求，则可以在Hadoop服务基础上再加入Spark服务。

38.3.2 作业

用户要运行一个计算任务，首先需要定义一个作业。

E-MapReduce支持Spark、Hadoop、Hive、Pig、Sqoop、Spark SQL和Shell等多种作业类型，用户可根据实际情况选择作业类型后，定义要执行的命令以及执行失败后的策略。

38.3.3 执行计划

执行计划是一组作业的集合，支持在一个现有的E-MapReduce集群上运行，也支持动态的按需创建出一个临时集群来运行作业。通过配置调度策略，可以被一次性或者周期性的执行。执行计划最大的优势就是执行多少作业就用多少资源，最大化的节省资源。

E-MapReduce支持如下执行计划调度策略：

- 周期调度：定义周期调度的频率与启动调度的时间后，执行计划会按设置的时间周期进行调度。
- 手动执行：执行计划只有在用户手动单击的情况下才会进行调度。

38.3.4 报警管理

E-MapReduce支持报警管理功能，支持将执行计划和报警接收组进行关联。在执行计划管理页面打开“报警通知”后，当执行计划执行完成时，关联的报警接收组中的联系人，都将会接收到短信通知。短信内容包含执行计划名、作业的执行情况（成功多少、失败多少）、对应的执行集群名以及具体的执行时长信息。

38.4 产品优势

与自建集群相比，E-MapReduce能给您提供相对方便可控的手段，从各方面管理自己的集群。此外，它还具有以下优势：

- 深度整合

  与阿里云其它产品如OSS、MNS、RDS、MaxCompute等深度整合，使其可作为E-MapReduce产品中Hadoop/Spark计算引擎的输入源或者输出目的地。

- 安全

  E-MapReduce整合了阿里云RAM资源权限管理系统，通过主子账号对服务权限进行隔离。

39 Quick BI

39.1 什么是Quick BI

Quick BI是一个基于云计算的灵活的轻量级的自助BI工具服务平台。

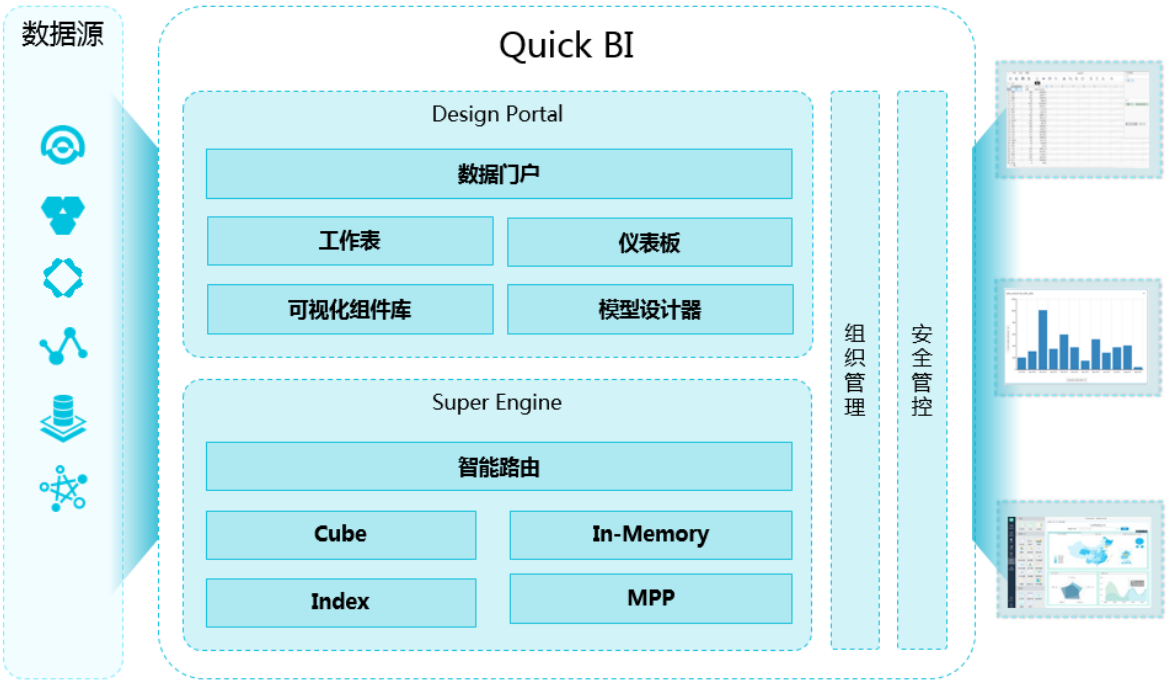
Quick BI支持众多种类的数据源，既可以连接MaxCompute（ODPS）、HybirdDB for MySQL、AnalyticDB、HybridDB（Greenplum）等云数据源，也支持连接ECS上您自有的MySQL数据库，同时，还支持连接来自VPC的数据源。Quick BI可以为您提供海量数据实时在线分析服务，通过提供智能化的数据建模工具，极大降低了数据的获取成本和使用门槛，通过拖拽式的操作和丰富的可视化图表控件，帮助您轻松自如地完成数据透视分析、自助取数、业务数据探查、报表制作和搭建数据门户等工作。

Quick BI不止是业务人员看数据的工具，更能让每个人都成为数据分析师，帮助企业实现数据化运营。

39.2 产品架构

Quick BI产品架构如下图所示。

图 39-1: Quick BI架构



Quick BI的主要模块和相关功能。

- **数据连接模块**

  负责适配各种云数据源，包括MaxCompute、HybirdDB for MySQL（MySQL、PostgreSQL、SQL Server）、AnalyticDB、HybridDB（MySQL、PostgreSQL）等，封装数据源的元数据或者数据的标准查询接口。

- **数据预处理模块**

  负责针对数据源的轻量级 ETL 处理。目前主要是支持MaxCompute的自定义SQL功能，未来会扩展到其他数据源。

- **数据建模**

  负责数据源的OLAP建模过程，将数据源转化为多维分析模型，支持维度（包括日期型维度、地理位置型维度）、度量、星型拓扑模型等标准语义，并支持计算字段功能，允许您使用当前数据源的SQL语法对维度和度量进行二次加工。

- **工作表/电子表格**

  负责在线电子表格（Workbook）的相关操作功能，涵盖行列筛选、普通或高级过滤、分类汇总、自动求和、条件格式等数据分析功能，并支持数据导出，以及文本处理、表格处理等功能。

- **仪表板**

  负责将可视化图表控件组装为仪表板。支持线图、饼图、柱状图、漏斗图、树图、气泡地图、色彩地图、指标看板等17种图表，支持查询条件、TAB、IFRAME、PIC和文本框5种基本控件，支持图表间数据联动效果。

- **数据门户**

  负责将仪表板组装为数据门户，支持内嵌链接（仪表板）和外嵌链接（第三方URL），支持模板和菜单栏的基本设置。

- **QUERY引擎**

  负责针对数据源的查询过程。

- **组织权限管理**

  负责组织管理和工作空间管理的两级权限架构体系管控，以及工作空间下的用户角色体系管控，实现基本的权限管理，实现同一份报表，不同的人可以看不同的内容。

- **行级权限管理**

  负责数据的行级粒度权限管控，实现同一张报表，不同的人可以看不同的数据。

- **分享/公开**

支持将工作表/电子表格、仪表板、数据门户分享给其他的用户，支持将仪表板公开到互联网供非登录用户查看。

39.2.1 部署方案

通过天基进行Quick BI自动化部署。

39.2.2 组件及作用

Quick BI 分为如下几类服务角色。

- base-biz-yunbi-dbinit：进行元数据初始化
- quickbi-redis-slave： redis 缓存 slave
- quickbi-redis-master：redis 缓存 master
- base-biz-yunbi-executor： quickbi agent 服务
- base-biz-yunbi： quickbi 前台页面 web 服务
- ServiceTest： 自动化测试服务

39.3 功能特性

Quick BI提供以下功能：

无缝集成云上数据库

支持阿里云多种数据源，包括MaxCompute、HybirdDB for MySQL（MySQL、PostgreSQL、SQL Server）、AnalyticDB、HybridDB（MySQL、PostgreSQL）等。

图表

丰富的数据可视化效果。系统内置柱状图、线图、饼图、雷达图、散点图等多种可视化图表，满足不同场景的数据展现需求，同时自动识别数据特征，智能推荐合适的可视化方案。

分析

多维数据分析。基于Web页面的工作环境，通过拖拽式的操作和类似于Excel的页面展示，实现数据的一键导入和实时分析，并可以灵活地切换数据分析的视角，无需您重新建模。

快速搭建数据门户

通过拖拽式操作、强大的数据建模和丰富的可视化图表，帮助您快速搭建数据门户。

实时

支持海量数据的在线分析。您无需提前进行大量的数据预处理，大大提高数据的分析效率。

数据权限

内置组织成员管理功能，支持行级数据权限，满足不同人看不同的报表，以及同一份报表，不同的人查看不同的内容。

39.4 产品优势

Quick BI的总体优势可以总结为多、快、强大和易用。

多

支持HybirdDB for MySQL、MaxCompute、AnalyticDB等多种云数据源。

快

亿级数据秒级响应。

强大

内置完整的电子表格工具，可以让您轻松制作复杂的中国式报表。

易用

丰富的数据可视化功能，自动识别数据特征，自动为您推荐合适的图表。

40 关系网络分析I+

40.1 什么是关系网络分析

阿里云关系网络分析（简称I+）是阿里云推出的一款基于关系网络的海量数据智能可视化研判平台。

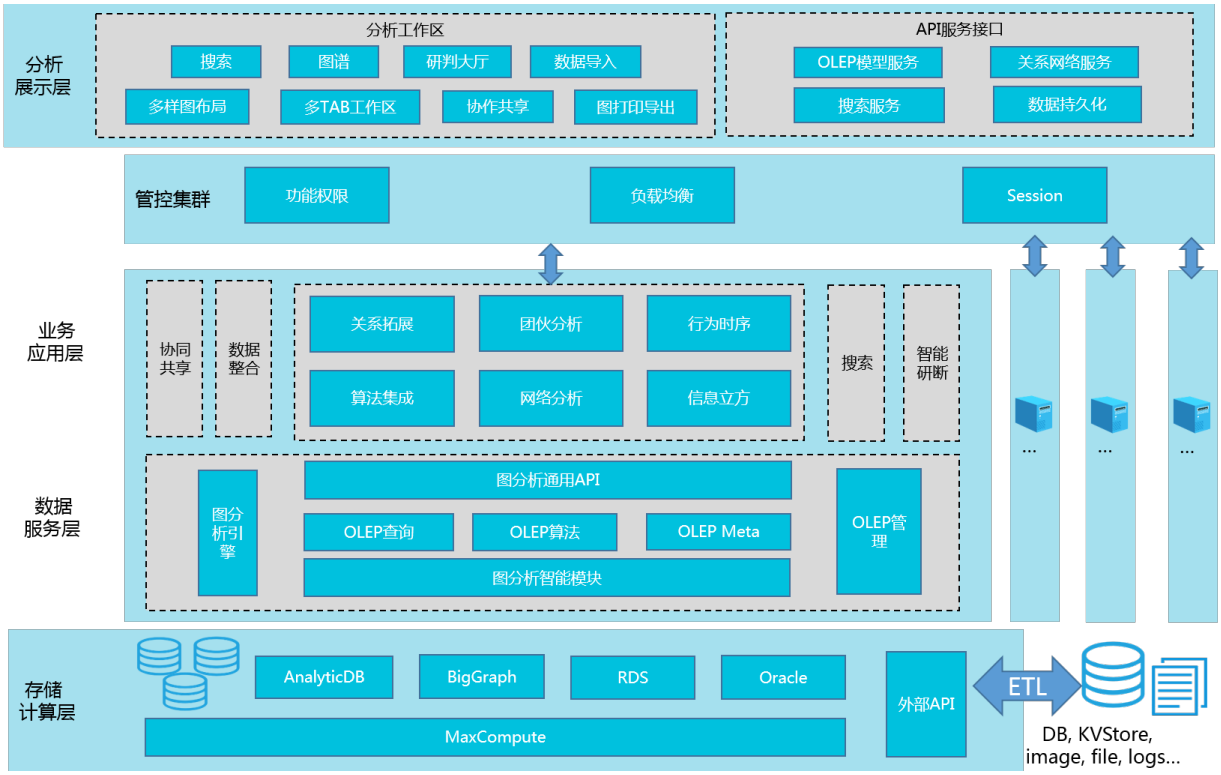
产品面向公安、工商、税务、海关、银行、保险、互联网金融等领域的大数据情报分析，为案件分析研判、反洗钱、反欺诈、反腐反贪、关联交易等调查分析提供强力支撑，帮助分析人员洞察数据中的关键信息，快速、智能的寻找破案线索及有价值的情报。

40.2 产品架构

I+采用组件化、服务化设计理念，多层次体系架构，产品架构如图 *40-1: 产品架构图*所示。。

整个系统分为存储计算层、数据服务层、业务应用层、分析展示层。

图 40-1: 产品架构图



- **存储计算层**：基于阿里云自主研发的大数据基础服务平台上，支持PB/EB 级别数据的存储和计算，支持多种开放数据源，具有强大的数据整合、处理、分析、计算能力。

  计算平台分为离线和在线：

离线计算为 MaxCompute，实现数据的整合、处理。

在线计算实现数据的实时计算，包括分析型数据库 AnalyticDB、在线图计算服务（BigGraph）。

- **数据服务层**：按照关系域、关系类型、关系事项抽象出的**实体—属性—关系**模型，提取自然对象关系、社会对象关系，进行业务关系逻辑建模，通过逻辑业务定义整合异域多源的数据，支持逻辑模型的灵活管理和维护。数据服务引擎为业务应用层提供统一的业务逻辑查询语言，执行各种复杂的关系网络查询、算法分析。
- **业务应用层**：将关联网络、搜索网络、信息立方、智能研判、协作共享、动态建模等多业务业务应用封装成API接口，提供给分析层调用。
- **分析展示层**：提供多元智能可视化交互分析界面，支持多种终端。提供外部 API 接口及可视化组件服务，支持第三方系统的接入。

40.3 功能特性

40.3.1 关系网络

关系网络是I＋研判分析平台的核心模块，所有实体之间的关系拓扑、业务计算、可视化布局均在该模块中进行。同时两大辅助分析组件用户空间和信息立方会对关系网络研判进行补充，使之可以涵盖大部分研判业务场景。

40.3.2 搜索网络

搜索网络提供检索对象信息的功能，用户在分析过程中逐步递进拓展信息，从线索关键词开始细化分析，如**电话号码：138*****001**，**姓名：张三**，**住址：杭州市西湖区文三路**号**等。搜索功能提供的检索工具可以帮助用户快速定位对象信息，同时作为关系网络的入口，可以将检索到的对象信息引入到**关系网络**中继续进行拓展分析。

40.3.3 信息立方

I+提供图形、表格等多视角信息呈现方式，包括行为分析、时序分析、行为明细、网络统计、属性分布统计等，帮助用户进行多维信息洞察。

40.3.4 智能研判

智能研判是I＋平台上旨在研判过程中为业务提供智能化线索的算法应用。这些算法是I＋算法同学深入研判领域如公安、反恐以及税务中沉淀出的业务智能。目前I＋已经沉淀的伴随分析、涉恐指数、吸毒指数等应用，在多个项目中取得重大成果。

40.3.5 动态建模

复杂时空大数据往往以复杂关联网络形式存在，基于本体论和语义网技术，产品中设计实现大数据背景下通用领域抽象数据模型。用对象、关系的链接抽象的方式来组织刻画数据，将数据析解成对象（Object）、属性（Property）和事件（Event）以及对象间的关联关系（Link），形成OLEP数据模型。OLEP数据模型支持用户快速组织整合数据，同时支持业务模型灵活扩展。

40.4 产品优势

40.4.1 超大规模计算及存储

I+的数据存储计算平台建立在阿里云自主研发的大数据基础服务平台（数加平台）上，支持PB / EB级别数据规模的处理。

I+的计算存储平台包括大数据计算服务（MaxCompute）、分析型数据库（AnalyticDB）、在线图计算服务（BigGraph）等。

大数据计算服务（MaxCompute）

阿里云大数据服务 MaxCompute，单个集群的规模可达5000台，并且具备跨机房的线性扩展能力，轻松处理海量数据。离线调度支持百万级任务量，实时监控告警。

核心指标：

- 支持万亿级数据的处理，百万级并发作业，以及每天PB级别的作业吞吐。
- 具备跨集群（机房）数据共享能力，支持万级别的集群数，扩容不受限制。
- 提供功能强大易用的SQL、MR引擎，兼容大部分标准SQL语法。
- MaxCompute（原ODPS）采用三重备份、读写请求鉴权、应用沙箱、系统沙箱等。多层次数据存储和访问安全机制保护用户的数据不丢失、不泄露、不被窃取。

分析型数据库（AnalyticDB）

阿里云分析型数据库（Analytic DB）是基于MPP架构并融合了分布式检索技术的分布式实时计算系统。

核心指标：

- 拥有快速处理千亿级别大数据的能力，使得数据分析中使用的数据可以不再是抽样的，而是业务系统中产生的全量数据，使得数据分析的结果具有最大的代表性。
- 采用分布式计算技术，拥有强大的实时计算能力，通常可以在数百毫秒内完成百亿级的数据计算，使得使用者可以根据自己的想法在海量数据中自由的进行 探索。
- 支撑较高并发查询量，并且通过动态的多副本数据存储计算技术来保证较高的系统可用性。

在线图计算服务（BigGraph）

在线图计算服务（BigGraph）是一款低延时高可用的分布式图计算产品，适用于大数据上的交互式图分析场景。

核心指标：

- 分布式存储
- 分布式计算
- 兼容Gremlin图遍历语言
- 基于多备份的高可用机制

40.4.2 跨数据源建模，多源整合计算，灵活高效部署

与耗时耗力的传统数据仓库物理模型不同，I+为分布在多个计算存储资源上的明细数据建立统一的OLEP逻辑模型。用户在对业务数据进行理解和梳理后，可以通过I+管理后台进行业务分析功能的建模和定义，即自定义业务逻辑模型、配置逻辑模型和实际物理数据存储的映射关系、配置应用场景的参数等。当数据源或业务有变动时，逻辑模型可以进行灵活修改并能够实时部署生效。

以目前I+实际落地的项目来看，在充分理解业务数据的基础上，通常在1-2天就能完成系统的部署和应用的上线使用。

40.4.3 智能算法组件集成，挖掘数据价值

I＋平台的关系网络引擎为关系型数据的挖掘提供了多种关系网络模型和智能算法，融入机器学习、业务算法，实现智能分析，如犯罪系数、骨干分析、路径分析等，帮助用户快速实现对关系网络数据的复杂挖掘。

40.4.4 智能可视化交互，提升用户体验

I＋提供可视化的交互界面，支持将海量数据以关系网络、信息立方的方式呈现给用户，方便用户从网络、时间两个维度视角来探查分析。同时提供各种常用应用工具的操作，支持用户常用操作习惯，提升用户体验。

40.4.5 高度参数配置化，实现灵活的项目定制

I+平台支持对数据源、业务模型、样式图标、用户角色权限、技术参数和系统参数的配置管理。用户可以根据项目的实际情况在管理控制台灵活自定义配置，满足具体项目的实际业务需求。

40.5 产品价值

目前I+已经作为亮点应用参与了多个国家级重点项目，涉及包括公安、金融、国税、保险、互联网金融等多个行业多个领域，尤其在安保、反恐、反洗钱、税务欺诈等细分领域的应用让客户耳目一新，业务价值得到深度考验和认可。

下面列举一些典型行业的典型应用案例，说明I+在真实场景中如何通过交互式的所见及所得的方式，让用户在复杂数据环境中理清数据之间的关联和逻辑关系，并通过直观和友好的用户界面展现给用户。

说明：
案例截图来源于真实应用系统，相关的案例敏感数据已做脱敏处理。

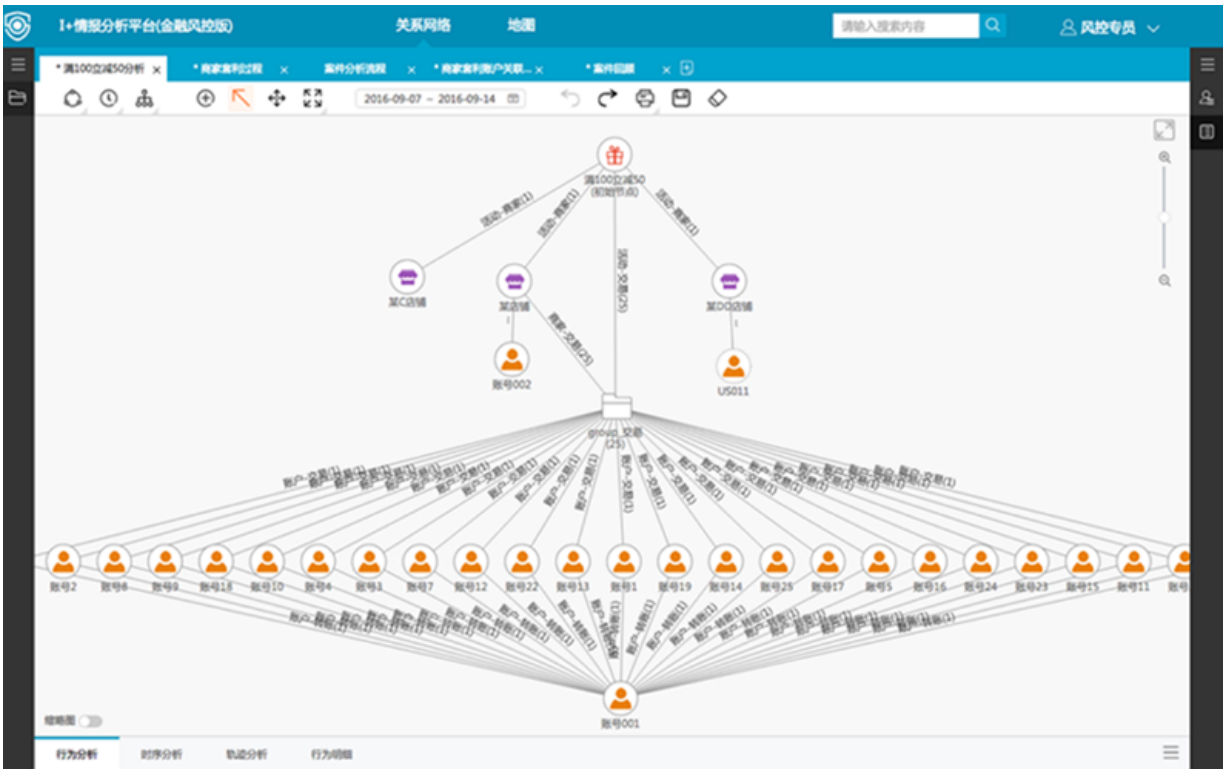
40.5.1 金融行业应用

此案例描述的是互联网金融风控专员如何通过I＋进行**营销反作弊**的分析。

某互联网企业投放大额资金在线下店铺开展买100立减50的纳新营销活动，部分店铺企图套取营销资源，通过事先批量注册小号并在活动前进行转账激活，活动当天在店铺购买物品产生虚假交易，骗取营销资源并获利。

该互联网企业综合利用转账、位置、设备、环境等信息，构建可疑关系网络，通过I+情报分析平台进行快速定位分析，如图 *40-2:* 营销反作弊分析所示，最终追回被套利资源，避免了营销资源的浪费，同时对相应的环境及账号进行布控，对该店铺进行相应的处罚。

图 40-2: 营销反作弊分析



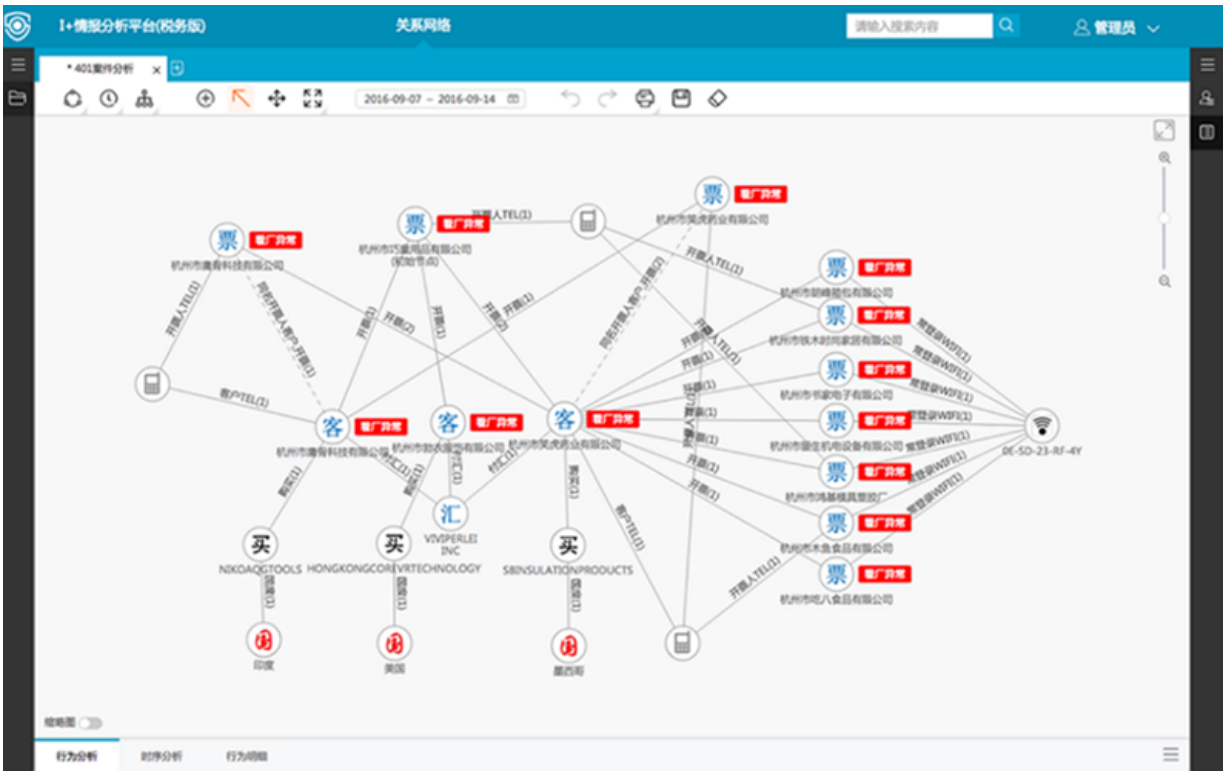
40.5.2 税务行业应用

此案例描述的是税务分析人员如何通过I+进行**出口骗税**的分析。

某外贸综合服务平台为中小企业提供专业、低成本的通关、外汇、退税以及配套物流和金融服务，但是部分企业存在投机取巧、造假骗税的情况，骗税金额不断攀升，对平台的外贸资质带来重大负面影响。

分析人员利用I+情报分析平台的关联反查、群体分析、骨干分析、共同邻居、信息立方等功能，如图 *40-3:* 出口骗税分析所示，从被举报的企业入手，查获了一个涉及13家企业的大规模骗税团伙，该团伙在平台上注册不同角色，内部间虚开发票、作假交易，向贸易平台申报退税金额达到上亿人民币，实地看厂调查后，对该团伙实施了相应的处罚，并追回损失。

图 40-3: 出口骗税分析



41 采云间DPC

41.1 数据集成平台

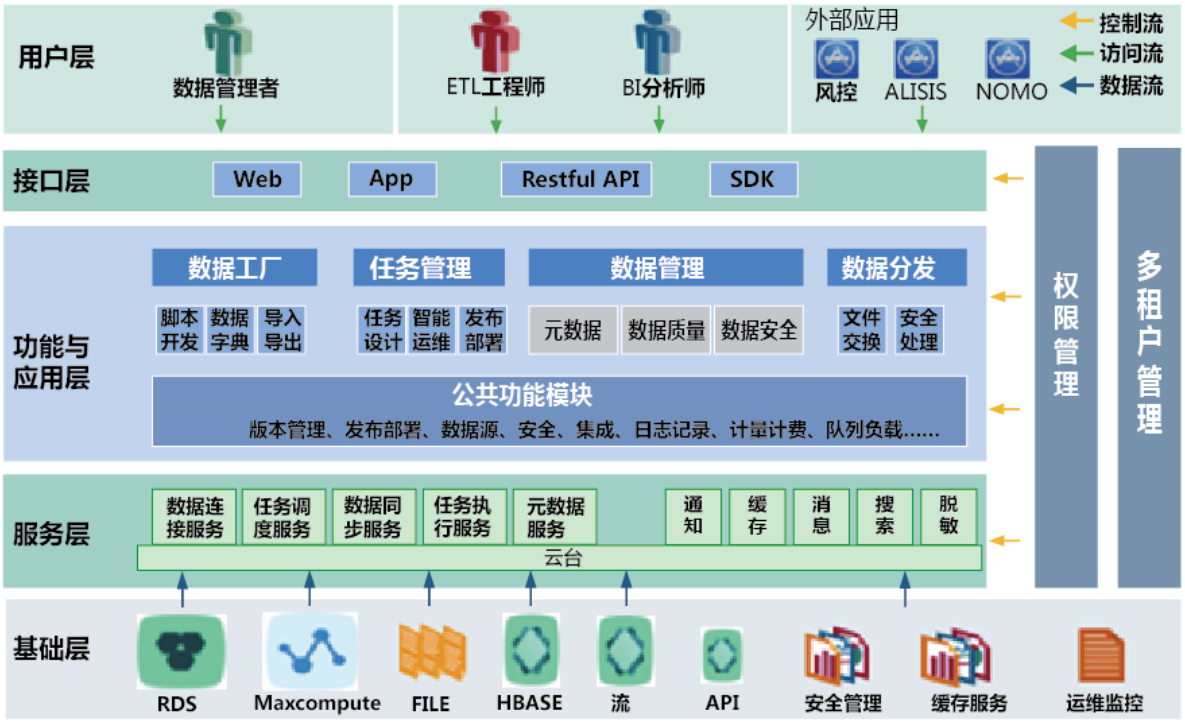
41.1.1 什么是数据集成平台

阿里云专有云数据集成平台（Data Integration Platform，简称DIP）是阿里云专有云大数据平台解决方案的一个分支产品，DIP基于MaxCompute提供强大的产品功能，支持复杂的企业级数据集成，涵盖数据的集成、ETL开发、任务调度、任务发布和运维等功能，完成数据的价值重构和可信赖交付，帮助企业从大数据中获得更多价值。

41.1.2 产品架构

阿里云专有云数据集成平台系统架构如图 *41-1:* 阿里云数据集成平台系统架构所示。

图 41-1: 阿里云数据集成平台系统架构



阿里云专有云数据集成平台的基础架构分为五层：

- 用户层：系统的用户包括实际的操作用户和应用用户，支持不同角色的用户来访问。系统对不同用户，开放不同的权限。
- 接口层：系统提供PC Web、API和SDK等方式，供外部客户使用。

- 功能与应用层：系统提供数据工厂、任务管理、数据管理和数据分发等功能，涵盖了从数据采集、数据加工到数据发布的全流程。
- 服务层：系统通过云台屏蔽了不同PAAS平台的差异，提供统一的基础数据服务。
- 基础层：一般是IAAS层和PAAS层提供的基础能力。

41.1.3 功能特性

41.1.3.1 数据工厂

数据工厂提供基于大数据平台的ETL开发的编程环境。

ETL设计与开发是大数据的DW和BI建设的关键环节之一，负责系统数据的生成，将数据在EDW数据架构的层次之间进行加工传输。

数据工厂主要包含的功能模块如下：

- 脚本开发
- 数据字典
- 数据管道

41.1.3.1.1 脚本开发

脚本开发模块提供脚本编辑和调试的开发环境，可以进行数据清洗和加工的脚本开发，也可以作为灵活查询的工具，执行查询命令。

脚本开发具备如下功能特性：

- 提供个人和团队进行ETL脚本协同开发。
- 便捷的SQL执行和调试功能，可以对执行结果集进行各类操作，满足灵活查询的需要。
- 以文件树的方式记录、编辑和管理代码脚本，支持文件加锁和版本管理。
- 支持UDF函数上传，支持数据分析师上传自己的数据进行关联查询。

41.1.3.1.2 数据字典

通过数据字典功能，您可以查看表清单和字段信息，也可以收藏个人关心的表。

数据字典具备如下功能特性：

- 支持查询指定数据源的所有表和我创建的表。
- 支持查看表详情，包括创建人、存储和字段清单。
- 支持分组收藏关心的表，方便使用。
- 支持自动生成常见SQL。

- 支持把表授权给他人，并查看授权记录。

41.1.3.1.3 数据管道

数据管道支持将数据导入到MaxCompute（原ODPS），也支持将数据从MaxCompute（原ODPS）中导出。

数据管道具备如下功能特性：

- 支持最大500M文件的上传，支持*.csv*和*.txt*格式。
- 支持导出查询结果。

> **说明**：
> 默认该功能不可用，需要设置。

41.1.3.2 任务管理

任务管理提供数据任务的全生命周期管理功能，可以周期性的运行数据采集、加工等任务，保证数据的日常开发。

任务管理主要包含的功能模块如下：

- 任务管理
- 任务监控
- 报警配置
- 发布部署
- 智能运维

41.1.3.2.1 任务管理

任务管理支持任务的创建、编辑、上线和删除等操作，可以灵活设置任务依赖。

任务管理具备如下功能特性：

- 云端数据同步

  通过简单易用的配置，实现RDS、Table Store、MaxCompute（原ODPS）等异构数据库之间的双向同步。
- 图形化任务调度的设置

  支持复杂的任务调度规则，包括任务定义、任务上下线、任务挂起、补数据、任务监控和异常报警等。

- 支持自动解析依赖，减少手动配置工作。

41.1.3.2.2 任务监控

任务监控提供任务运行的实时监控大图，方便查看任务执行进度，也可快速发现和解决出错任务。

任务监控具备如下功能特性：

- 实时数据刷新，第一时间发现问题。
- 可视化的任务流显示，方便查找异常任务。
- 支持任务的挂起、跳过和重试等功能。

41.1.3.2.3 报警配置

报警配置功能支持设置报警规则，以便在任务执行出现异常时第一时间发出报警。

报警配置具备如下功能特性：

- 支持通过多种渠道提醒。
- 支持配置多种预警规则类型：成功、失败、超时未开始、超时未结束和运行时长超时。
- 支持任务的挂起、跳过和重试等功能。

41.1.3.2.4 发布部署

发布部署支持把已经开发好的任务和脚本，发布到其他环境。

发布部署支持同集群发布和跨集群发布。

在发布部署模块，您可以：

- 创建发布包

  选择待发布的任务，设置发布目标环境。

- 发布监测

  系统会对发布的任务自动检测，确认在目标环境中的依赖是否完整，数据源是否存在。

- 发布审核

  发布管理员可以审阅待发布的发布包，以决定该发布包是否需要发布。

41.1.3.2.5 智能运维

智能运维提供运维监控大屏，可以实时显示当前任务执行的情况。

运维监控大屏主要显示以下监控项：

- 异常任务：包括任务运行超时和启动延迟。
- 任务实况：显示任务的实时运行情况。
- 刷新任务运行时长Top10：显示运行时长Top10的刷新任务。
- 同步任务运行时长Top10：显示运行时长Top10的同步任务。
- 任务报警次数Top10：显示报警次数Top10的任务。

41.2 机器学习平台

41.2.1 什么是机器学习平台

阿里云采云间机器学习平台是一套基于MaxCompute（原ODPS）的数据挖掘、建模、预测的工具。提供算法开发、分享、模型训练、部署、监控等一站式算法服务。

用户可以通过可视化的操作界面来操作整个实验流程，同时也支持PAI命令，让用户通过命令行来操作实验。阿里云机器学习平台沉淀了阿里巴巴的机器学习算法体系和经验，从数据的预处理、到机器学习算法、模型的评估和预测动能。

在专有云方面，阿里云机器学习平台的运行需要依赖于MaxCompute，通过将算法包部署到MaxCompute集群中，用户通过阿里云机器学习平台调用算法，实现算法的应用和计算引擎的解耦。

阿里云机器学习平台丰富的算法和技术保障支持也给用户解决自身业务场景带来了更多的可能性和想象空间。在DT时代，通过使用阿里云机器学习平台可以真正实现数据驱动业务的目的。

41.2.2 产品架构

如图 *41-2:* 阿里云机器学习平台系统架构所示，阿里云机器学习平台的基础架构分为四层，用户通过阿里云机器学习平台调用下一层的模型和算法，然后系统将相应的算法转换成对应的计算类型，比如说对于两个表的join操作，会自动生成SQL的workflow传到底层MaxCompute（原ODPS）进行运算和操作。所有算法都是以plugin的形式存放在底层的计算引擎当中，用户在使用算法的时候只关心算法的调用即可，真正实现了算法和计算引擎的解耦。

图 41-2: 阿里云机器学习平台系统架构

41.2.3 功能特性

41.2.3.1 完善的数据挖掘组件

机器学习平台提供整套的数据挖掘组件，包括数据的预处理、特征抽取、模型训练、预测和评估。

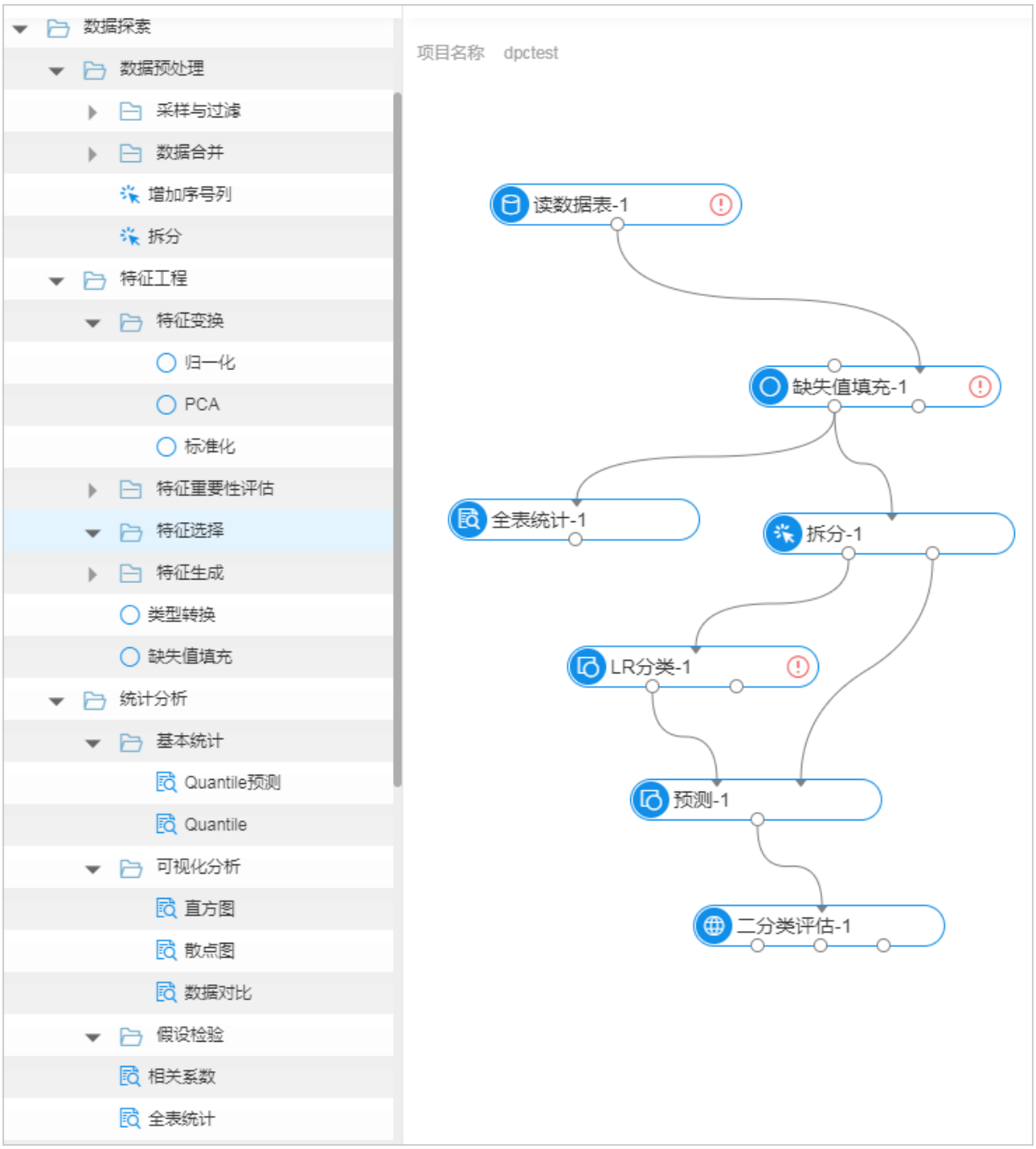
用户将数据导入平台，可以借助实验组件灵活的拼装自己的试验流程来解决自身的业务场景。

41.2.3.2 可视化建模

如图 *41-3:* 可视化建模所示，在操作界面，通过拖拉拼接实验。

从左边的组件框中拖拉组件到右边的实验区进行实验的搭建。

图 41-3: 可视化建模

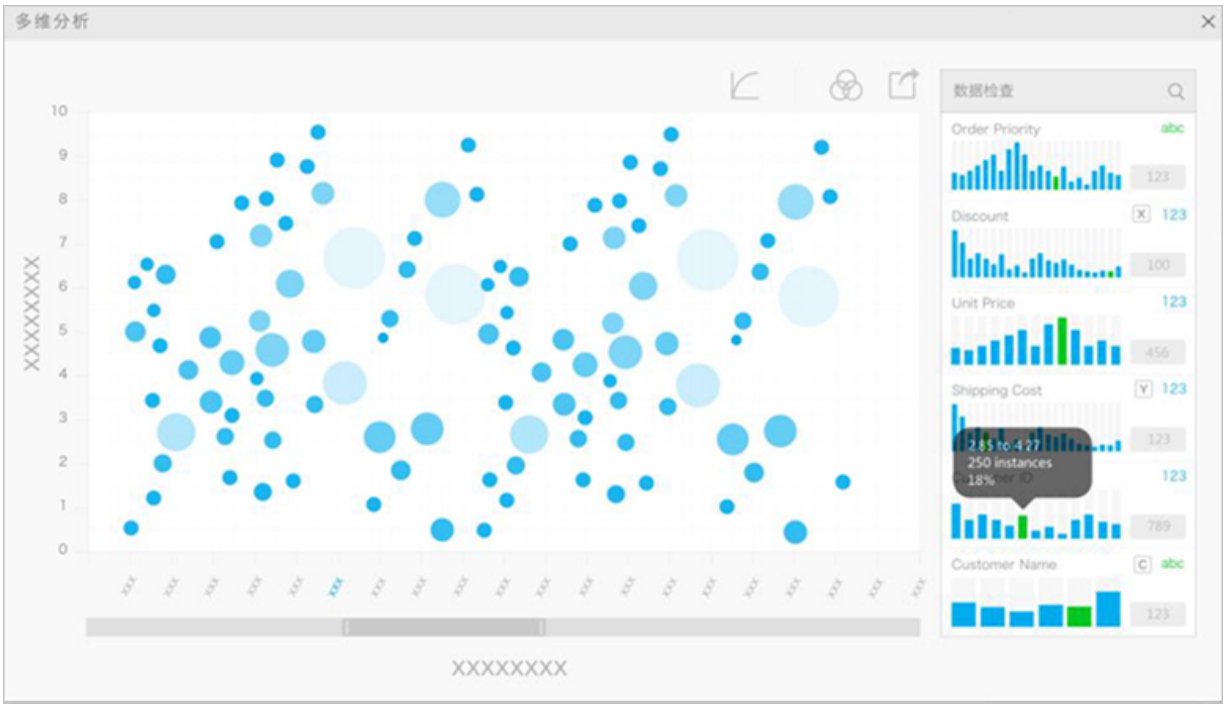


41.2.3.3 数据可视化

对于每个输出型组件，都可以通过右键组件来查看可视化输出模型，如图 *41-4:* 数据可视化所示。

可视化输出有多种表示方法，包括折线图、点图和柱形图。

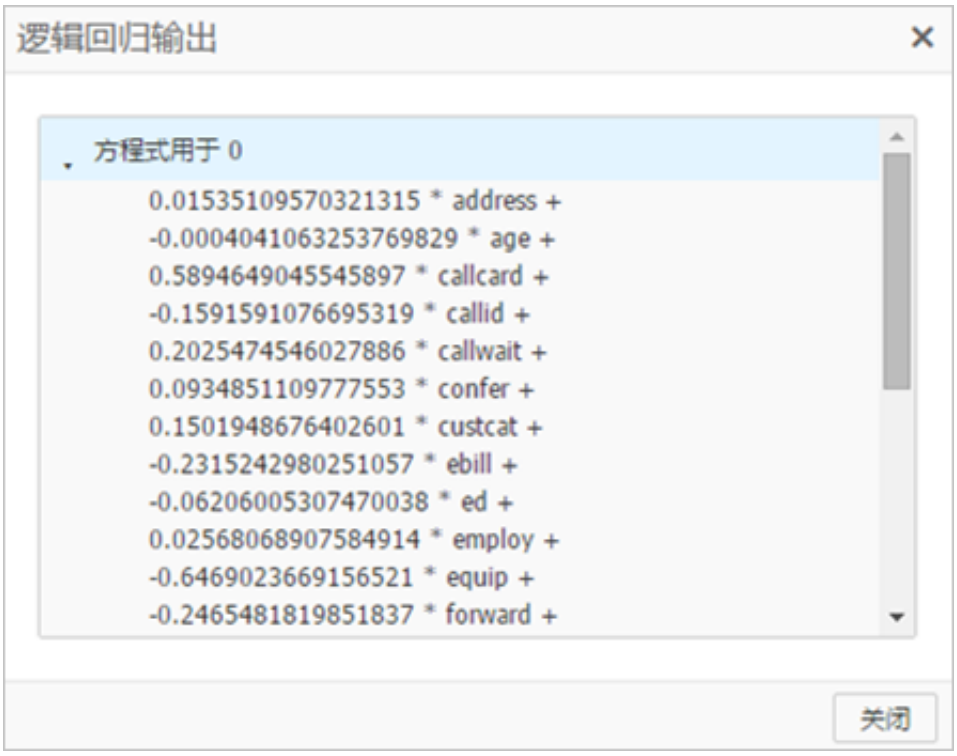
图 41-4: 数据可视化



41.2.3.4 模型可视化

在最左侧的菜单页签中选择模型页签，有查看模型选项，在相应的机器学习算法文件夹下就可以查看相应的模型结果，如图 *41-5:* 模型可视化所示。

图 41-5: 模型可视化


逻辑回归输出
方程式用于 0
0.01535109570321315 * address +
-0.0004041063253769829 * age +
0.5894649045545897 * callcard +
-0.1591591076695319 * callid +
0.2025474546027886 * callwait +
0.0934851109777553 * confer +
0.1501948676402601 * custcat +
-0.2315242980251057 * ebill +
-0.06206005307470038 * ed +
0.02568068907584914 * employ +
-0.6469023669156521 * equip +
-0.2465481819851837 * forward +
关闭

42 机器学习

42.1 什么是机器学习

机器学习指通过统计学算法，对大量的历史数据进行学习从而生成经验模型，利用经验模型指导业务。机器学习主要在以下几方面发挥作用：

- 营销类场景：商品推荐、用户群体画像、广告精准投放。
- 金融类场景：贷款发放预测、金融风险控制、股票走势预测、黄金价格预测。
- SNS 关系挖掘：微博粉丝领袖分析、社交关系链分析。
- 文本类场景：新闻分类、关键词提取、文章摘要、文本内容分析。
- 非结构化数据处理场景：图片分类、图片文本内容提取OCR。
- 其它各类预测场景：降雨预测、足球比赛结果预测。

阿里云机器学习是一套基于MaxCompute（原ODPS）的数据挖掘、建模、预测的工具。通过阿里云机器学习，您可以：

- 获得算法开发、分享、模型训练、部署、监控等一站式算法服务。
- 您可以通过可视化的操作界面来操作整个实验流程，同时也支持通过PAI命令来操作实验。主要面向数据挖掘人员、分析师、算法开发者、数据探索者。
- 在专有云方面，阿里云机器学习平台的运行需要依赖于MaxCompute，将算法包部署到MaxCompute集群后，您可以通过阿里云机器学习平台调用算法，实现算法的应用和计算引擎的解耦。
- 阿里云机器学习平台丰富的算法和技术保障支持，为您解决自身业务场景带来了更多的可能性和想象空间。在DT时代，通过使用阿里云机器学习平台可以真正的实现数据驱动业务的目的。

42.2 产品架构

42.2.1 基础架构

阿里云机器学习平台的基础架构分为四层，用户通过阿里云机器学习平台调用下一层的模型和算法，然后系统将相应的算法转换成对应的计算类型。例如当两个表的进行联合操作时，机器学习平台会自动生成 SQL 工作流，传输到底层 MaxCompute进行运算和操作。所有算法都是以插件的形式存放在底层的计算引擎当中，用户在使用算法的时候只关心算法的调用即可，真正实现了算法和计算引擎的解耦。

42.2.2 系统框架

阿里云机器学习由多个组件系统架构而成，系统框架如图 *42-1:* 阿里云机器学习框架图所示。

图 42-1: 阿里云机器学习框架图

说明如下：

- 基础设施层，是CPU计算集群。
- 计算框架层，包括MapReduce/SQL/MPI等计算方式，分布式计算架构主要解决计算任务的并行化计算分发作用。
- 模型算法层，包含数据预处理、特征工程、机器学习算法等基本组件，所有算法组件全部脱胎于阿里巴巴集团内部成熟的算法体系，经受过PB级别业务数据的锤炼。
- 业务应用层，阿里巴巴内部的搜索、推荐、蚂蚁金服等项目在进行数据挖掘工作时，都是依赖PAI平台产品。可以基于PAI平台搭建各种业务场景，覆盖金融、医疗、教育、交通、安全等领域。

42.2.3 组件及作用

文档中描述了机器学习总共由三大组件（Cap、Dmscloud、Jcs）以及三个基础中间件服务（RDS、OCS、Zookeeper）组成。

RDS

RDS 为机器学习的数据库层，记录了用户相关数据（包括用户、租户、项目）、环境数据、实验数据，所有持久化的数据都在RDS里。

OCS

OCS 为机器学习的缓存层，用以缓存KV数据以及用户Session数据。

Zookeeper

Zookeeper 负责Jcs服务的巡检，以及任务状态的控制与调度。Dmscloud 把任务给提交给Jcs后，Zookeeper 记录任务的初始状态，根据当前任务队列的情况，将任务提交给空闲的 Jcs。

Cap

Cap 是机器学习的登录控制模块，定期同步Base的用户、租户数据，用户登录机器学习时会先跳转到Cap做登录验证，验证成功后根据同步过来的租户数据选择对应的租户，工作空间和项目。

Dmscloud

Dmscloud 是机器学习的核心服务，作为机器学习平台的前端，展示机器学习的各个组件。用户可以在 Dmscloud上拖拽不同的组件拼接成实验，进行机器学习，数据分析等操作。

Jcs

Jcs 是机器学习的任务控制模块，处理用户在 Dmscloud 提交的实验任务，进行任务的调度和执行。

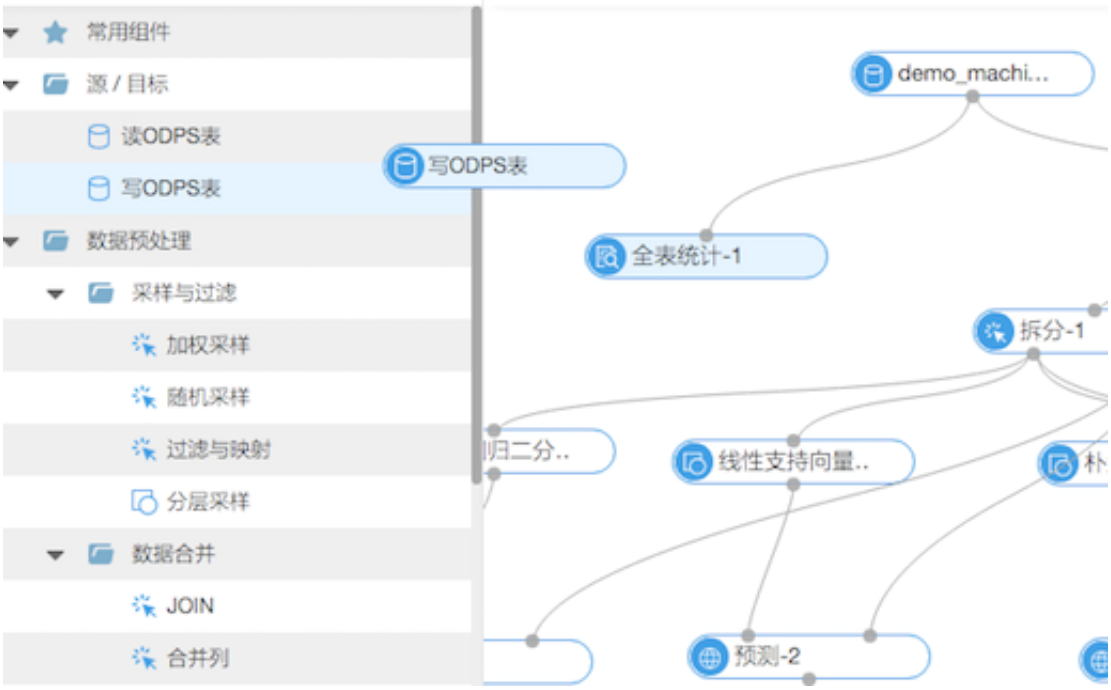
42.3 产品特性和核心优势

阿里云机器学习平台的产品主要优势可以概括为以下几方面：

良好的交互设计

通过拖拽的方式搭配实验，并且提供了数据模型的可视化功能。缩短了用户与数据的距离，真正实现了数据的触手可及。同时也提供了命令行工具，方便用户将算法嵌入到自身的工程中。

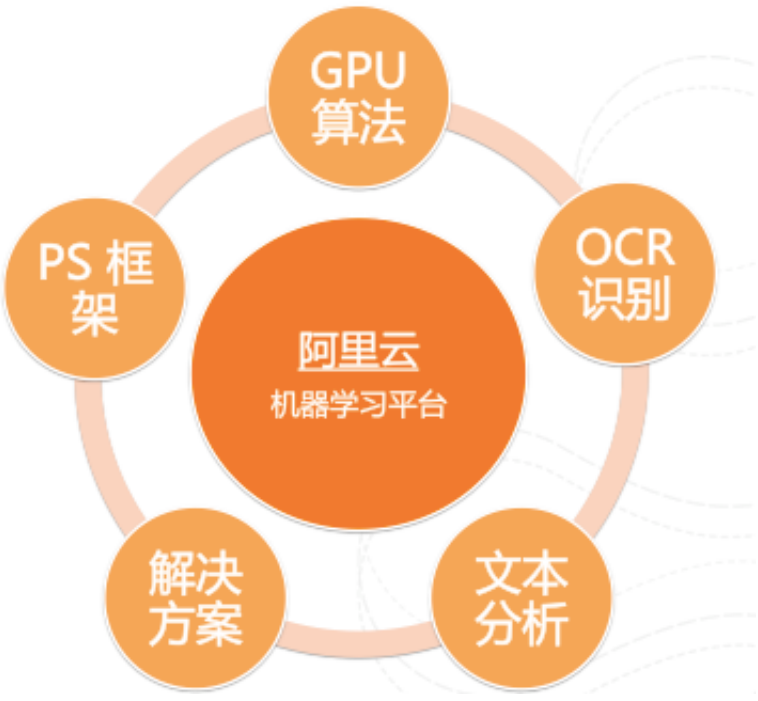
图 42-2: 操作界面



优质、丰富的机器学习算法

平台上边的机器学习算法都是经过阿里大规模业务锤炼的。从算法的丰富性角度来看，阿里云机器学习平台不仅提供了基础的聚类、回归等机器学习算法，也提供了文本分析、特征处理的算法。

图 42-3: 算法框架



与阿里系的融合

使用阿里云机器学习平台计算的模型直接存储在 MaxCompute上。

可以配合其它阿里云的产品组件加以利用。

图 42-4: 阿里云数加产品图

优质的技术保障

阿里云机器学习算法平台的背后是阿里巴巴 IDST 的算法科学家和阿里云的技术保障团队，在使用过程中遇到任何问题都可以到工单系统提交工单或者直接与相关接口人联系。

图 42-5: 工单系统

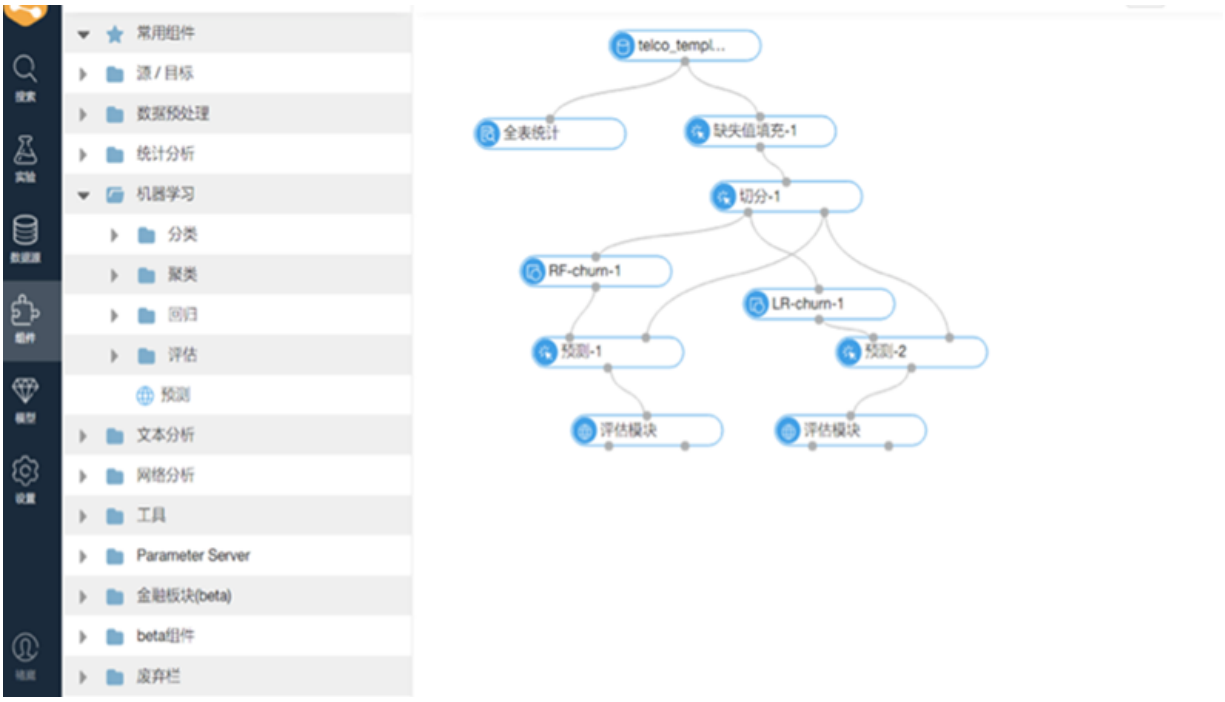
42.4 功能描述

完善的数据挖掘组件

提供整套的数据挖掘组件，包括数据的预处理、特征抽取、模型训练、预测和评估。用户将数据导入平台，可以借助实验组件灵活的拼装自己的试验流程来解决自身的业务场景。

可视化建模

操作界面：通过拖拉拼接实验。从左边的组件框中拖拉组件到右边的实验区进行实验的搭建。

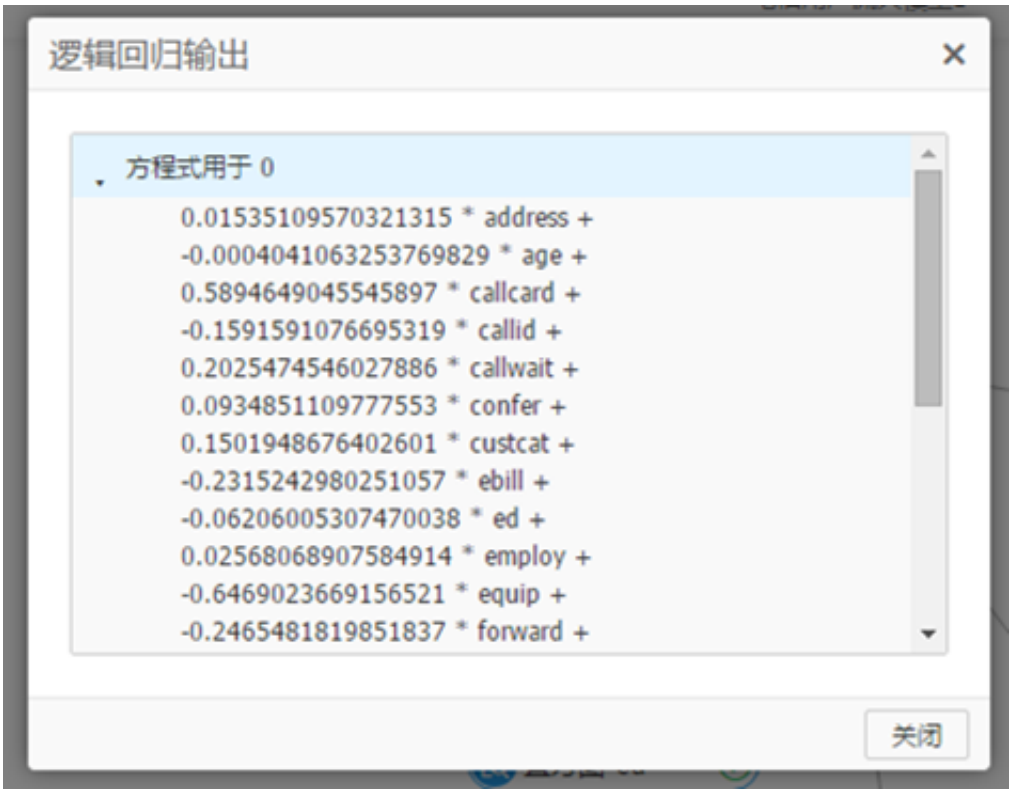


数据可视化

数据的可视化：对于每个输出型组件，都可以通过右键组件来查看可视化输出模型。可视化输出有多种表示方法，有折线图、点图和柱形图。

模型可视化

模型的可视化管理：在最左侧的菜单页签中选择模型页签，有查看模型选项，在相应的机器学习算法文件夹下就可以查看相应的模型结果。



算法组件

机器学习产品部署完成之后， 在可视化界面上可包括如下算法组件：

表 42-1: 算法组件

一级目录	二级目录	算法组件	说明
源/目标			机器学习 PAI 输入数据源。
		读 MaxCompute 表	读取数据源。
		写 MaxCompute 表	写数据源。
数据预处理	采样与过滤	加权采样	按照一定比例采样。
		随机采样	按照随机采样。
		过滤与映射	利用 sql where 过滤。
		分层采样	按照等级抽样。
	数据合并	拆分	按照比例拆分数据。
		合并列	合并两个表的两列。
		UNION	SQL UNION。

一级目录	二级目录	算法组件	说明
	其它	标准化	对表的某一列进行标准化处理。
		归一化	归一化是一种简化计算的方式，即将有量纲的表达式，经过变换，化为无量纲的表达式，成为标量。
		增加序列号	在数据表中增加自增 id 列。
特征工程	特征变换	主成分分析 PCA	降维算法。
	特征重要性评估	线性特征重要性	对应算法的特征评估。
		随机森林特征重要性	
统计分析		百分位	计算某列的百分位。
		全表统计	计算全表的每个字段的统计信息，包括缺省值、最大值、最小值、方差、偏值等。
		皮尔森系数	计算两字段（数值型）的皮尔森相关系数。
		直方图	多个字段查看直方图。
		散点图	数据点在直角坐标系平面上的分布图。
		相关系数矩阵	计算多个字段的相关系数矩阵。
机器学习	二分类	线性支持向量机	支持向量机 SVM 是一个有监督的学习模型，通常用来进行模式识别、分类、以及回归分析。
		逻辑回归二分类	通过逻辑回归算法对数据进行训练生成二分类

一级目录	二级目录	算法组件	说明
			模型，属于有监督的机器学习。
		GBDT二分类	GBDT 是一种迭代的决策树算法，该算法由多颗决策树组成，所有树的结论累积起来做最终答案。
	多分类	逻辑回归多分类	线性回归的多分类。
		随机森林	随机森林指的是利用多棵树对样本进行训练并且预测的一种分类器。
		朴素贝叶斯	朴素贝叶斯法是基于贝叶斯定理与特征条件独立假设的分类方法。
	回归	GBDT 回归	利用 GBDT 树状结构最回归。
	聚类	K均值聚类	聚类相似度是利用各聚类中对象的均值所获得的一个中心对象来进行计算的。
	评估	二分类评估	对于上述算法的评估和预测。
		多分类评估	
		回归模型评估	
		聚类模型评估	
		混淆矩阵	
	预测	预测	
	关联推荐	协同过滤 etrec	etrec 是一个 item base 的协同过滤算法，输入为两列，输出为 item 之间相似度 topK。
文本分析		分词	对指定的文本内容列进行分词，目前仅支持中

一级目录	二级目录	算法组件	说明
			文淘宝分词和互联网分词。
		词频统计	在分词基础上，按行保序输出对应文章 ID 对应的词，统计指定文章 ID 列对应内容的词频。
		TF-IDF	TF-IDF 是一种统计方法，用以评估一个词对于文件集或一个语料库中的其中一份文件的重要程度。
		PLDA	给出每篇文档中主题的概率真分布。
		Word2Vec	将词表转为向量。
		三元组转 KV	将给定的三元组（row, col, value) 转成 kv 格式（row,[col_id,value]）
		文本摘要	使用算法在文章中提取文摘。
		关键词提取	从指定的文本中提取和这篇文章意义最相关的词组。
		句子拆分	按照句子标点符号进行拆分。
工具		MaxCompute SQL	执行 MaxCompute SQL。
网络分析		K-Core	一个图的 Kcore 是指反复去除度小于或等于 K 的节点后，所剩余的子图。
		单源最短路径（SSSP）	计算最短路径。

一级目录	二级目录	算法组件	说明
		PageRank	计算网页 Rank 值。
		标签传播聚类（LabelPropagationClustering）	基于图的半监督学习方法，依赖其邻居节点的标签信息，影响程度由节点相似度决定，并通过传播迭代更新达到稳定。
		标签传播分类（LabelPropagationClassification）	该算法为半监督分类算法，原理为用已标记节点的标签信息去预测未标记节点的标签信息。
		Modularity	Modularity 是一种评估社区网络结构的指标，来评估网络结构中划分出来的社区的紧密程度，往往0.3以上是比较明显的社区结构。
		最大联通子图（maximalConnectedComponent）	在无向图 G 中，从连通的顶点寻找可以顶点相通的子图，寻找到的子图数据量称为最大联通子图。
		点聚类系数（nodeDensity）	在无向图 G 中，计算每一个节点周围的稠密度，星状网络稠密度为0，全联通网络稠密度为1。
		边聚类系数（edgeDensity）	在无向图 G 中，计算每一条边周围的稠密度。
		计数三角形（triangleCount）	在无向图 G 中，输出所有三角形。
		树深度（treeDepth）	对于众多树组成的网络，输出每个节点所处深度和树id。

42.5 应用场景

泰坦尼克号沉船事件：通过分析泰坦尼克号沉船事件幸存者和丧生者的数据，判断拥有什么样的属性的人有更大的概率获救。

数据准备

图 42-6: 数据准备

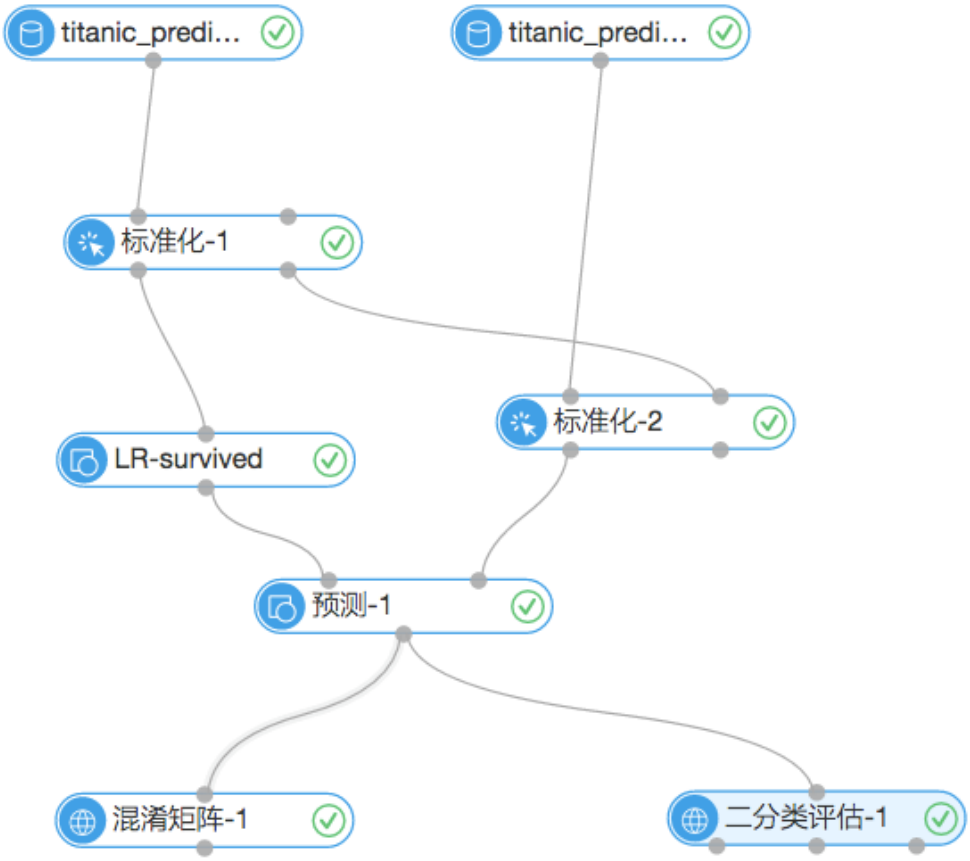
passengerid	survived	pclass	sex	age	sibsp	parch	fare	cabin	embarked
1.0	0.0	3.0	1.0	22.0	1.0	0.0	7.25	0.0	1.0
2.0	1.0	1.0	0.0	38.0	1.0	0.0	71.2833	1.0	2.0
4.0	1.0	1.0	0.0	35.0	1.0	0.0	53.1	1.0	1.0
5.0	0.0	3.0	1.0	35.0	0.0	0.0	8.05	0.0	1.0
7.0	0.0	1.0	1.0	54.0	0.0	0.0	51.8625	1.0	1.0
8.0	0.0	3.0	1.0	2.0	3.0	1.0	21.075	0.0	1.0
9.0	1.0	3.0	0.0	27.0	0.0	2.0	11.1333	0.0	1.0
10.0	1.0	2.0	0.0	14.0	1.0	0.0	30.0708	0.0	2.0

将数据导入MaxCompute：

- passengerId：用户的ID号
- survived： 乘客是否获救，1表示获救，0表示没有获救。目标队列（target）
- pclass：乘客的社会阶层，1表示Upper，2表示Middle，3表示Lower
- sex：乘客的性别，1表示男，0表示女
- age：乘客的年龄
- sibsp：乘客在船上的配偶数量或兄弟姐妹数量
- parch：乘客在船上的父母或子女数量
- fare：乘客的船费
- cabin：是否住在独立的房间，1表示是，0为否
- embarked：表示乘客上船的码头距离泰坦尼克出发码头的距离，数值越大表示距离越远

搭建实验流程

图 42-7: 搭建实验流程



1. 首先将数据集按照7：3进行拆分，一部分作为 Titanic 训练集，一部分作为预测集。
2. 将数据进行标准化处理，去除量纲对于数据造成的干扰。
3. 训练数据通过逻辑回归 LR 算法生成模型。
4. 对预测集进行预测。
5. 通过 ROC 曲线和混淆矩阵来对结果进行评估。

评估结果

- 混淆矩阵结果

图 42-8: 混淆矩阵结果

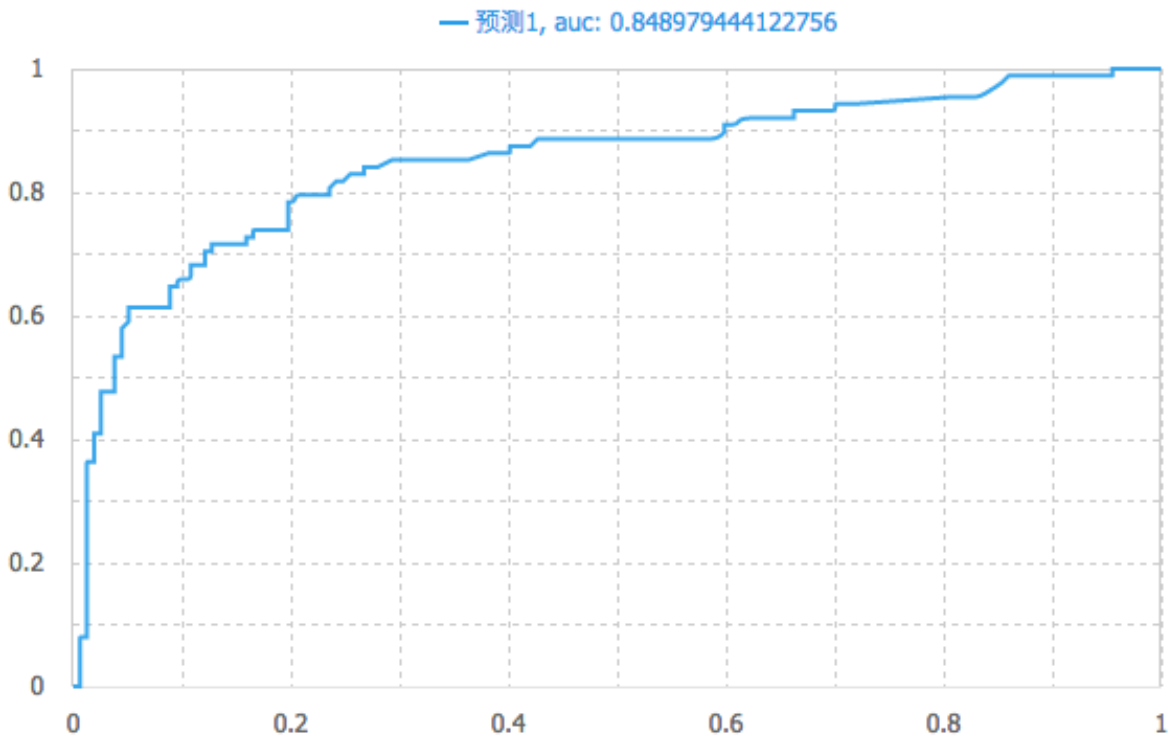
预测1

模型	正确个数	错误个数	总计	正确率	召回率	F1指标
1	65	27	92	70.652%	73.864%	72.222%
0	130	23	153	84.967%	82.803%	83.871%

1. 正确率：预测正确的正例个数占预测为正例的比例，即 P=TP/(TP+FP)。
2. 召回率： 预测正确的正例个数占实际正例的比例，即 R=TP/(TP+FN)。
3. F1 指标：P 和 R 的中权调和平均，即 F1=2PR/(P+R)。当 F1 较高时说明实验方法比较理想。

- ROC 曲线评估结果

图 42-9: ROC曲线评估结果



1. TPR=TP/(TP+FN)
2. FPR=FP/(FP+TN)
3. ROC 曲线

a. Y轴：TPR; X轴：FPR

b. (0,1)：FRP=0,TPR=1。FN=0,FP=0，将所有样本都正确分类

c. (1,0)：即，FPR=1,TPR=0，预测结果相反，正变负，负变正

d. (0,0)：FRP=TPR=0，将所有样本划分为负样本

e. (1,1)：所有样本划分为正样本

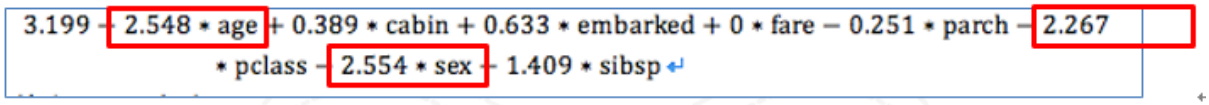
f. ROC 曲线越接近左上角，该分类器越好

4. y=x，随机猜测，一半正样本，一半负样本
5. AUC：ROC 曲线下面积，应该介于0.5~1之间。AUC 越大，分类器越好

模型分析

逻辑回归生成模型：

图 42-10: 模型分析



根据 Logical regression 的特性，model 输出的是每个特征的线性组合。3.199为常数项，不予考虑。其它系数绝对值越大说明对结果影响越大。通过这一结论得出 age、sex 和 pclass 对于结果影响最大。

根据 sigmoid 函数得出负号系数的绝对值越大其结果的正例可能性越大。所以我们可以得出结论，age、pclass 和 sex 的值越小，目标值越大。

也就是有钱人家的女人和小孩有更大的获救概率。

通过真实数据比对也印证了我们的分析结果：

图 42-11: 分析结果

age	cabin	embarked	fare	parch	pclass	sex	sibsp	survived	prediction_result
26.0	0.0	1.0	7.925	0.0	3.0	0.0	0.0	1.0	1
30.0	0.0	3.0	8.4583	0.0	3.0	1.0	0.0	0.0	0
58.0	1.0	1.0	26.55	0.0	1.0	0.0	0.0	1.0	1
14.0	0.0	1.0	7.8542	0.0	3.0	0.0	0.0	0.0	1
35.0	0.0	1.0	26.0	0.0	2.0	1.0	0.0	0.0	0
15.0	0.0	3.0	8.0292	0.0	3.0	0.0	0.0	1.0	1
19.0	1.0	1.0	263.0	2.0	1.0	1.0	3.0	0.0	0
30.0	0.0	3.0	7.8792	0.0	3.0	0.0	0.0	1.0	1
30.0	0.0	3.0	7.75	0.0	3.0	0.0	0.0	1.0	1
30.0	0.0	2.0	7.2292	0.0	3.0	1.0	0.0	1.0	0
21.0	0.0	1.0	8.05	0.0	3.0	1.0	0.0	0.0	0
3.0	0.0	2.0	41.5792	2.0	2.0	0.0	1.0	1.0	1
30.0	0.0	3.0	15.5	0.0	3.0	1.0	1.0	0.0	0
30.0	0.0	3.0	7.75	0.0	3.0	0.0	0.0	1.0	1
30.0	0.0	2.0	21.6792	0.0	3.0	1.0	2.0	0.0	0

43 智能数据引擎Dataphin

43.1 什么是智能数据构建与管理Dataphin

Dataphin（智能数据构建与管理）是智能大数据平台建设引擎，旨在面向各行各业大数据建设、管理及应用诉求，通过输出阿里巴巴集团数据中台长达十年实战沉淀的大数据建设体系——OneData+OneID+OneService（产品+技术+方法论），一站式提供集数据引入、规范定义、数据建模、数据研发、数据萃取、数据资产管理和数据服务的全链路智能数据构建及管理服务，助力政府机构和企业打造属于自己的标准统一、融会贯通、资产化、服务化和闭环自优化的智能数据体系以驱动创新。

我们致力于屏蔽不同计算与存储环境差异，帮助用户快速引入数据、标准规范化构建数据、建模化方式自动开发数据、萃取以实体对象为中心的标签数据体系、沉淀业务数据知识与数据资产、治理数据问题，同时支持数据表查询、智能语音查询等多种类型数据服务。

图 43-1: 研发工作台

43.2 产品优势

Dataphin 包含以下优势。

- 数据规范统一：采用维度事实建模理论，对维度、维度属性、业务过程、指标字段等进行严格的标准化规范化定义，保障数据质量，避免数据指标定义的二义性。
- 高效且自动化的编码：基于函数化理念，对通用数据计算逻辑组件化定义并可自由组建统计指标，从而自助地实现建模研发、系统自动生成代码执行生产数据。
- 智能计算优化：支持以业务视角进行逻辑建模，逻辑模型发布后，系统自动化进行物理建模、编码，降低对专业大数据开发人员的技术能力依赖。
- 一站式研发体验：数据引入、建模、研发、运维、数据查找及探查等过程一气呵成，研发链路统一而高效。
- 系统化构建数据目录：基于规范化建模、高效自动化的元数据抽取，以标准的技术框架系统地构建规范可读的业务化数据目录，形成数据资产地图，方便业务查找及应用。
- 语义化智能数据检索：基于元数据及数据构建数据图谱，实现数据表及数据简单且快速的智能检索。
- 可视化数据资产：系统化构建业务数据资产大图，数据视角还原业务系统、提取业务数据知识，并可快速提炼业务关键环节及数据。
- 数据使用简单可依赖：定义即服务，研发构建的业务主题式数据逻辑表可被直接、快速地查询和访问，可简化约80%的查询代码。
- 提升效率：提供全链路、一站式、智能化的数据构建与管理工具，降低数据建设门槛，各背景的开发同学可自助ETL、快速完成数据需求，而其中贯穿的OneData、OneID、OneService思想与方法论（已申请专利）可完成模型&指标抽象与自助定义、代码自动化生产、主题数据自动聚合并输出服务。
- 降低成本：以元数据为基础、算法智能为驱动，实现物理+逻辑分层的智能自动化生产、数据资产全链路分析追踪与优化，保证最优计算及存储资源分配、最大降低数据使用成本。

43.3 产品特点

- 兼容计算引擎：支持MaxCompute、Hadoop等多计算引擎
- 数据引入：支持本地及阿里云数据库、非结构化存储、大数据存储等多种数据源，进行数据引入及结构化
- 全局规划：支持数据中心就业务板块、数据主题域、运行资源空间等进行全局规划设计
- 规范定义：工具化方式对数据进行规范定义，可快速批量结构化定义统计指标并自动加工输出
- 数据建模、研发：可视化方式支持数据逻辑模型构建，独有智能黑盒技术支持自动创建物理模型及相关代码的同时，兼容自定义编码方式进行数据开发
- 调度运维：可视化方式进行任务调度管理、运行维护

- 元数据管理：系统化、自动化抽取元数据，处理形成统一的元数据中心
- 资产分析：可视化业务数据资产大图和数据资产地图，帮助全局了解数据及详细的数据查找与分析
- 数据安全：支持项目、表、字段多层次权限控制
- 数据服务：支持基于业务主题式的逻辑表查询访问服务，也支持物理表的查询访问服务

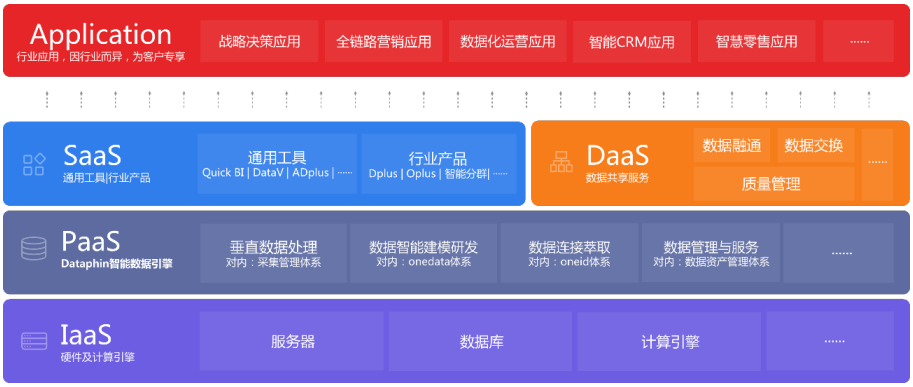
43.4 产品架构

Dataphin在业务系统中的位置如图 *43-2: 系统架构*所示。

图 43-2: 系统架构

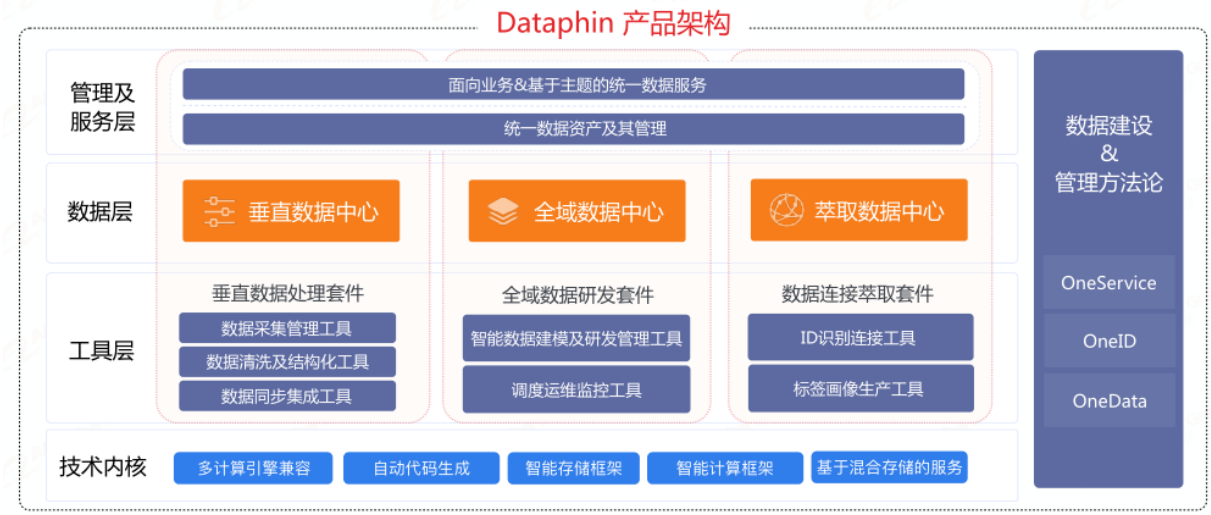
对于业务系统而言，配套的业务平台成为数据平台的PaaS基础层，可以快速地输出易用的数据服务来支撑上层多样化的数据产品，从而让业务数据化运营起来。作为数据平台的基础层，向下可兼容不同的硬件设施，向上可对接各类应用产品，从而构建出从IaaS到SaaS的数据通路，输出面向业务的规范标准、连接融合和可管理查询的数据，如图 *43-3: 业务系统*所示。

图 43-3: 业务系统



Dataphin产品及架构，以及架构间的关系如图 *43-4: 产品架构*所示。

图 43-4: 产品架构



基于OneData、OneID、OneService的数据建设与管理方法论，Dataphin系统由四部分组成，它们分别是**技术内核、工具层、数据层和管理服务层**。通过这四部分的协同合作，最终形成如下可控的数据流向，如图 *43-5: 数据流*所示。

图 43-5: 数据流

43.4.1 技术内核

一套屏蔽底层计算、存储、软件系统差异的技术框架，保证数据研发可兼容多计算引擎与计算时效，代码自动化生成并实现智能存储与计算，数据服务支持混合存储等。

43.4.2 工具层

面向开发者的数据构建与管理的工具，包括基础数据的标准规范及集成引入，公共数据的标准规范定义、智能建模研发、调度运维及机器学习，萃取数据的ID识别连接及标签生产。

43.4.3 数据层

在技术内核基础上，通过工具加工生产，输出三种层次的结构化数据，构建出高保真、面向各业务的基础数据中心，模型化、面向主题的公共数据中心，深度加工、以实体为中心的萃取数据中心。

43.4.4 管理及服务层

将数据及数据服务以资产化视角进行管理，以支持数据研发人员及业务人员都可以获取高质量且统一的数据资产；从业务视角将已有数据包装加工为主题式的数据服务，以保障业务可以统一地查询与调用数据。

43.5 功能特性

Dataphin包含以下几个功能模块：

平台功能

支持对整个产品系统地了解熟悉、全局化功能设置，帮助学习使用产品功能、快速开始工作，以及进行必要的系统管理与控制、保障各模块正常运转。

全局设计

支持从业务全局出发，拆解并设计对应的业务数据总线，从顶层进行数据中心的命名空间划分、主题域及相关名词定义、管理单元（即项目）划分、数据源定义。

数据引入

基于全局设计定义的项目空间与物理数据源，支持将各业务系统、各类型的数据抽取加载至数据库，完成数据的同步与集成，并通过各种清洗策略等，完成基础数据中心的建设。

规范定义

基于全局设计定义的业务总线、数据引入构建的基础数据中心，支持根据业务数据需求，结构化地定义、组件化地构建标签与统计指标等，以保证业务数据无二义性地标准化、规范化生产。

建模研发

基于规范定义的数据元素，支持可视化地设计与构建数据模型，提交发布后由系统智能自动化地生成代码与调度任务，完成公共数据中心的全托管生产。

编码研发

基于通用的代码编辑界面，支持自由灵活地进行个性化的数据编码研发，并完成对应任务发布。

资源及函数管理

支持各种资源包（如JAR、文档文件）管理以满足部分数据处理需求，支持原生的系统函数查找与使用，支持自定义函数以满足数据研发特殊的函数加工需求。

数据萃取

在基础数据中心及公共数据中心基础上，支持以“目标对象”为中心，用参数选配的方式可视化地识别与连接业务中对象ID、提取对象行为与标签，智能地完成数据打通与深度挖掘并生成代码与调度任务，完成萃取数据中心的全托管生产。

调度运维

支持基于策略对建模研发、编码研发、数据萃取生成的代码任务进行调度与运维管控，包括数据生产任务部署、任务运行及依赖情况查看并管理维护，以确保所有任务都有按照时间正确有序地生产数据。

元数据中心

支持采集、解析、管理基础数据中心、公共数据中心、萃取数据中心的元数据。

资产分析

在元数据中心基础上，支持元数据深度分析并实现资产化管理数据，以可视化地呈现资产分布、元数据详情等，以便捷查找及深度了解数据资产。

安全管理

支持从质量管理、安全管理等角度进行标准定义、分析呈现、流程管理、监控报警、从数据源到数据应用端的全链路追踪等，发现资产优化空间并提供治理建议。

即席查询

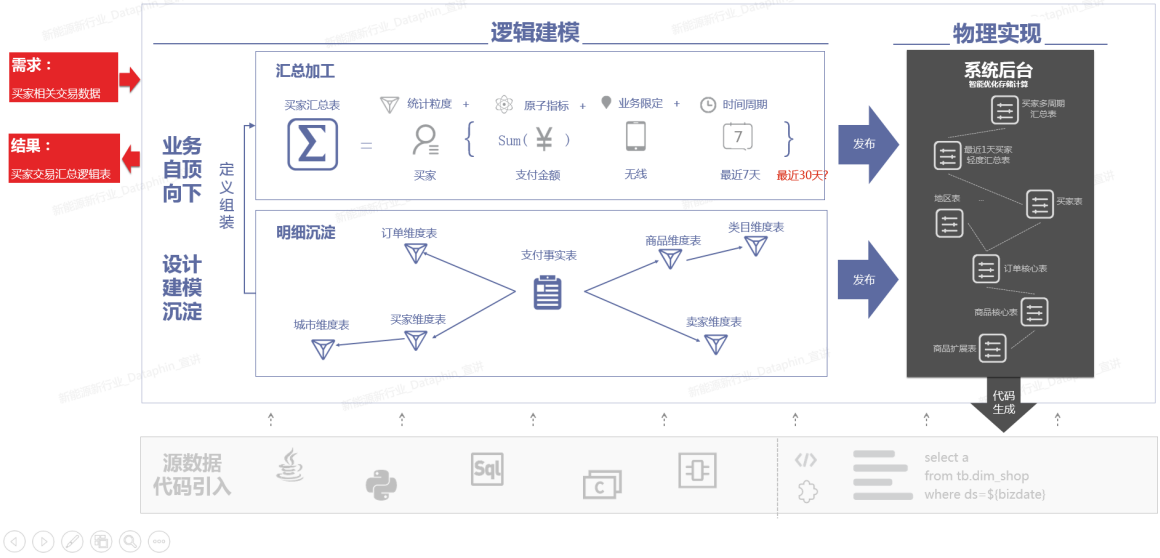
支持自定义SQL等方式查询数据资产中的数据，并通过查询分析引擎快速实现物理表及面向主题的逻辑表（也即数据模型，或逻辑模型）数据查询及结果获取。

43.5.1 能够解决

通过Dataphin，您可以解决以下问题：

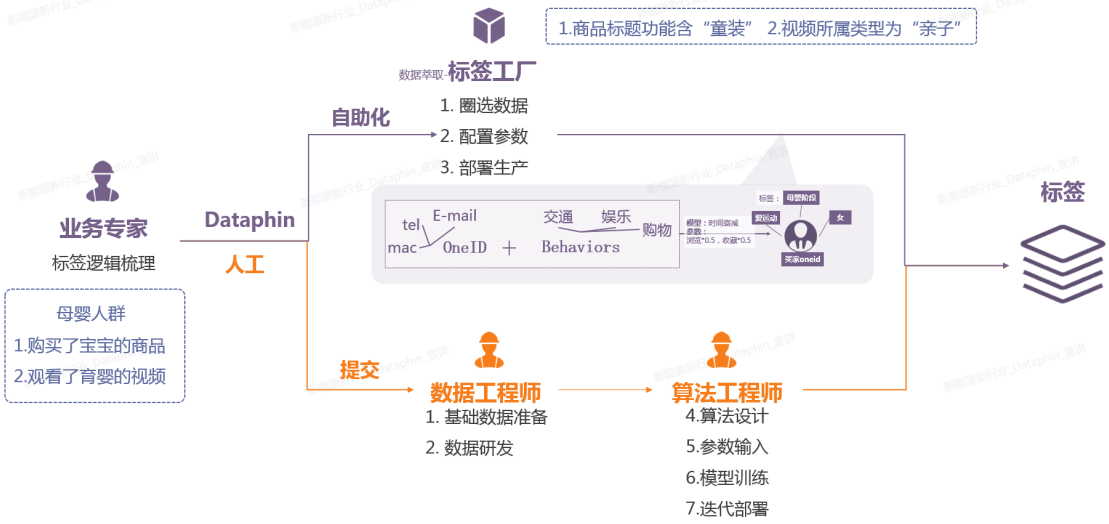
- 建模研发支持通过可视化定义SQL表达式的方式完成模型设计，而系统自动发布生成任务与生产数据，且所有数据指标标准规范无二义性。

Dataphin 亮点：建模研发自动化



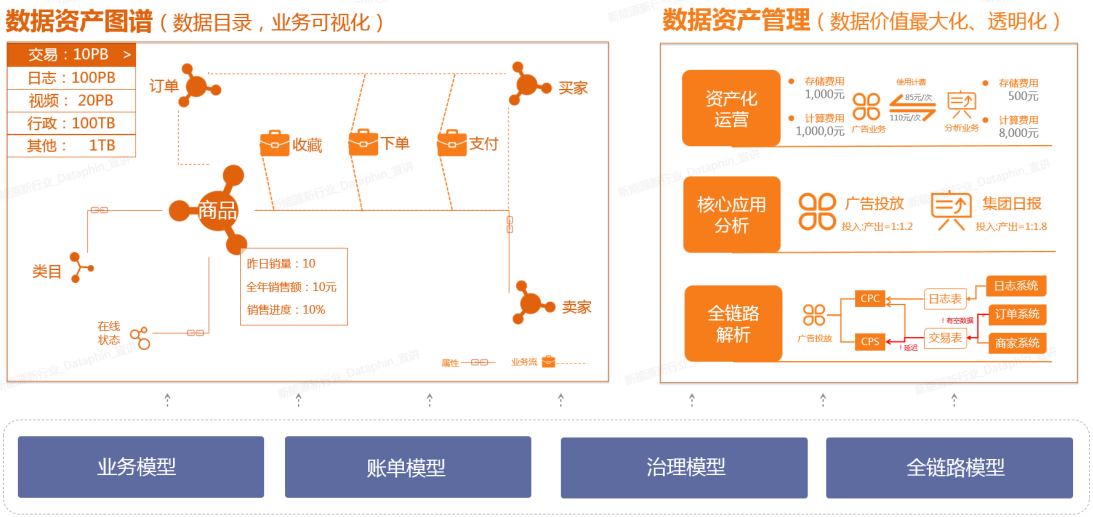
- 数据萃取（敬请期待）支持通过自定义配置参数的方式三步完成以实体对象为中心的业务主数据提炼、DMP构建，实现ID识别连接、标签标准规范地自动化生产，消除数据孤岛。

Dataphin 亮点 ：萃取连接自动化



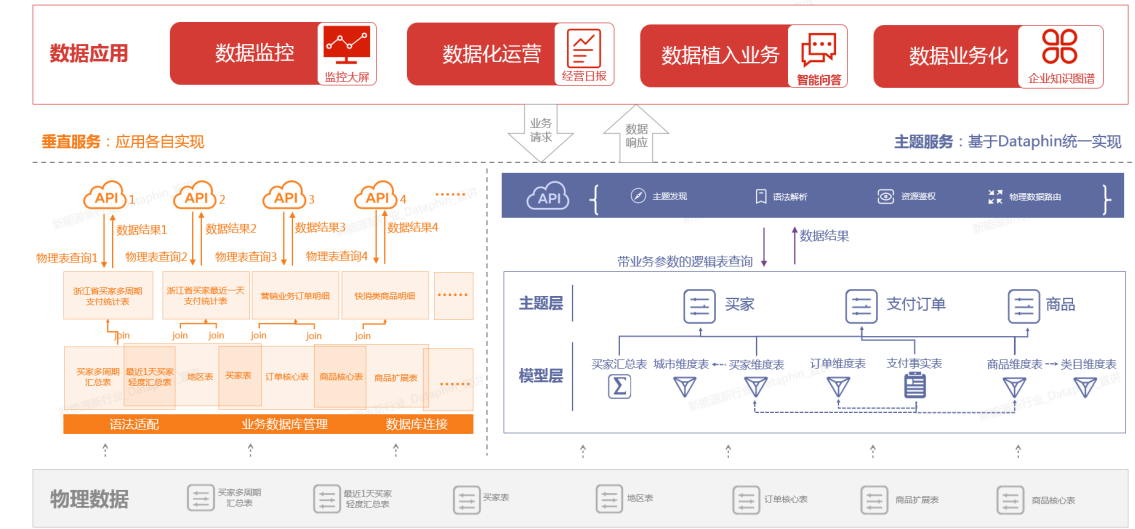
- 资产分析及治理支持资产化视角构建及管理数据体系并一目了然数据价值。

Dataphin 亮点：资产分析与治理清晰化



- 即席查询支持面向主题的逻辑表查询，保证数据被快捷方便地查找定位、数据查询SQL得到最大的简化，提高效率的同时，保证规范、标准、无二义的数据输出至业务应用。

Dataphin 亮点 ：查询应用一体化、智能化



43.5.2 平台管理

产品基础功能，以保证Dataphin系统中所有用户可控、有序、顺利地进行数据研发。本模块主要包括必要的全局功能设置—账号管理、计算管理，快速入门引导、首页及相关资料获取，以保证超级管理员角色的用户能把握平台全局，其他角色的用户能快速进入目标模块。

账号管理

从系统功能使用安全的角度进行用户账号控制，根据企业已有的账号系统识别、配置产品的用户范围，由最高权限的用户管理其他用户账号及权限。

计算管理

作为PaaS层的平台产品，从系统计算稳定、统一的角度进行计算类型、底座硬件的设置与管理，以兼容IaaS层计算引擎，实现不同环境下数据建设工作。支持Maxcompute、Hadoop生态两类主流计算引，支持对两类引擎元数据的自动采集与解析，元数据采集部署初始化相关内容参见元数据中心。

首页引导

数据构建与管理的入口，统一的产品指南、工作空间集大成的门户，展示数据生产、管理与服务的全流程，便于正式开始工作前系统地了解学习产品功能内容，并根据需求快速进入相应的功能模块。

国际化-语言支持

Dataphin产品根据系统的语言版本识别并默认选择语言，包括中文、English，方便不同国家地区用户使用产品，默认为中文。

43.5.3 全局设计

从业务全局出发进行数据架构的顶层设计，是数据建设工作中奠基的一步，保证数据的管理可控，数据研发、萃取、管理时定义及设计的数据体系满足中长期的业务需求，业务获取的数据与服务统一、面向主题、易用。

本模块包括基于业务特征划分业务总线——业务板块及数据域定义维护及权限控制、公共定义的全局性统计周期设置与管理，基于数据独立管理及开发协作需求划分项目空间——项目基本信息及计算资源配置管理、成员管理，基于项目计算资源及业务数据需求定义数据来源——数据源的配置管理。

业务总线

业务总线是基于业务特征以划分定义逻辑意义上的命名空间、主题分类、名词术语，以实现从管理顶层设计、控制建设过程中的数据定义标准化。

项目空间

项目空间是基于数据研发与管理团队对数据研发项目独立管理、对数据资源质量等高效管理的需求，为实现资源隔离、用户成员分组、数据建设约束条件配置等，而定义的物理命名空间。

物理数据源

支持数据源创建、修改等操作以注册及注销所需数据库，支持数据源类型包括MaxCompute、Mysql、SQL Server、Postgre SQL等，数据源一方面作为数据同步传输的来源或者目标，另一方面一些特殊数据源类型（如MaxCompute）可作为对应计算引擎类型设置后项目计算存储基座。

43.5.4 数据引入

数据引入是基于企业数据全局架构中基础数据层设计，选定所需的业务数据存储，并根据存储及数据时效、数据质量等需求，制定的数据同步、清洗、结构化策略。

作为数据中台建设的初始化环节，数据同步套件基于阿里内部在业务数据、日志数据等多种类型数据交换同步方面多年实践的沉淀。可实现业务原始数据高效地录入，并通过管道对传输元数据的采集与统计能力，对数据传输量和数据内容方面，可支持简单规则校验及数据容错自定义机制等灵活管控，实现数据高质的同步。

数据源配置

数据源配置管理模块，支持多数据源的接入与管理，清单列表的展示方式可一目了然管控已接入的数据源，同时支持多种类型的数据源新增。数据同步中心目前支持数据源包括MaxCompute、MySQL、SQLServer、PostgreSQL、Hive等。

数据同步

数据同步模块，支持来源数据与目标数据的选择，配置相关参数实现增量或全量同步，确定源数据与目标数据字段的映射关系，同时支持传输流量、并发数等的配置，并生成相关任务节点发布后进行调度。

43.5.5 规范定义

在传统的数据研发过程中，数据建模以及指标定义等具体而重要数据建设与研发工作，很多情况下依赖于研发人员的专业能力，命名方式没有统一的标准，仅基于个性且变动的文档进行研发工作规范及设计的传递说明，非常容易造成不同指标命名冲突或者重复计算等一系列的问题。

Dataphin基于OneData方法论，标准规范化维度、业务过程、指标定义等重要数据元素的定义，保证口径、算法、命名等的唯一性，从数据设计的顶层杜绝指标二义性的产生。此外，Dataphin也可以帮助用户基于表单式操作，便捷、批量地创建指标，降低数据研发门槛，可快速赋能有基本数据分析能力的业务人员，也大大提升了研发效率。

规范定义主要包括维度、业务过程、原子指标、业务限定、派生指标共5个模块，基于数据域实现进一步数据建设顶层总线设计、公用定义复用、标准数据元素沉淀——数据仓库主题（包含对象及关系、事务）、指标构建元素（最小计算逻辑单元、修饰）。

43.5.5.1 维度

- 业务板块内唯一，并唯一归属一个数据域，实现命名与主题分类上归一与规范化。
- Dataphin支持主子维度关系定义，以统一维度对象、归一维度特征。
- Dataphin支持不同类型的维度定义，包括枚举、虚拟、普通（层级）、普通维度。
- Dataphin支持对业务板块内、项目内两个范围的已有维度的清单查看和管理，也支持对单个维度的快速查看了解与编辑。

维度清单查看与管理

支持对已选择的项目下的维度清单进行查看，包括维度的名称、创建人、发布状态。支持清单列表中查找或搜索查找以定位具体的维度，并对该维度进行编辑、下线、删除等操作。

维度查看与管理

支持表单方式查看维度及标准化新建及编辑维度，维度作为业务关键概念。需要如下两类信息：

- 基本信息：维度所属数据域，维度中文名、英文名以及描述，其中英文名默认带dim_的前缀，以保证命名唯一。
- 逻辑信息：用确定性方式描述圈定维度对象范围，以逻辑化固定对象的特征，保证维度定义及后续细化时该维度真实且唯一性存在。针对Dataphin每种维度类型，所需填写内容不同，以满足不同维度对象建设需求。
- 快捷查看维度：用户可快速轻量查看维度基本信息，在不影响已有操作情况下了解所需维度的基本信息，并快速触达相关操作。

43.5.5.2 业务过程

业务过程指的在业务中发生的最小单元的行为或事务，比如创建订单，浏览网页等等。业务过程产生的行为明细，比如支付了一笔订单，浏览了某个网页，最终都会汇集到事实表中，而大部分情况下，事实表都会聚焦于某个特定的业务过程。

与维度的意义一样，业务过程是OneData方法论的顶层设计概念之一，与维度共同明确数据建设的架构，Dataphin支持业务过程的规范定义，不仅可以看到组织的业务全貌，还可以通过业务过程，方便地对事实表进行分类管理。

为保障后续事实模型统一、标准、规范地构建，业务过程在业务板块内唯一，并唯一归属一个数据域，实现命名与主题分类上归一与规范化。

Dataphin支持对业务板块内、项目内两个范围的已有业务过程的清单查看和管理，也支持对单个业务过程的快速查看了解与编辑。

业务过程清单查看与管理

支持对已选择的项目下的业务过程清单进行查看，包括业务过程的名称、创建人、发布状态。支持清单列表中查找或搜索查找以定位具体的业务过程，并对该业务过程进行编辑、删除等操作。

业务过程查看与管理

支持表单方式查看业务过程及标准化新建及编辑业务过程，业务过程作为业务关键概念，需要如下信息：所属数据域，以及中文名、英文名及描述等信息。

43.5.5.3 原子指标

原子指标是对指标统计口径、具体算法的一个抽象。为了从根源上解决定义、研发不一致的问题，Dataphin创新性地提出了“设计即开发”的理念，指标定义同时即明确设计统计口径（即计算逻辑），不需要ETL二次或者重复研发，从而提升了研发效率，也保证了统计结果的一致性。根据计算逻辑复杂性，Dataphin将原子指标分为两种：原生的原子指标，如支付金额，衍生原子指标-基于原子指标组合构建，如客单价为支付金额除以买家数。

为保障所有统计指标统一、标准、规范地构建，原子指标在业务板块内唯一，并唯一归属一个来源逻辑表，计算逻辑也以该来源逻辑表模型的字段为准，进行确定性定义，每个原子指标取来源逻辑表模型相关的所有逻辑表追溯所属分类，故而可能归属多个数据域，从而实现命名、逻辑归一，主题分类规范化、标准化、系统化。

原子指标清单查看与管理

支持对已选择的项目下的原子指标清单进行查看，包括原子指标的名称、创建人、发布状态。支持清单列表中查找或搜索查找以定位具体的原子指标，并对该原子指标进行编辑、下线、删除等操作。

原子指标查看与管理

- 原子指标

  为保证原子指标生产规范化，仅支持从逻辑表及其对应模型上定义原子指标，用户可以选择来源表，在对应雪花或星型模型上选择字段，定义基于该字段的计算逻辑为原子指标。

- 衍生原子指标

  衍生原子指标即通过其他的原子指标计算得来的。比如下单支付转化率这个原子指标，可以事先定义好支付买家数和下单买家数两个原子指标，通过支付买家数/下单买家数的计算逻辑定义得到。

43.5.5.4 业务限定

原子指标是计算逻辑的标准化定义，业务限定则是条件限制的标准化定义，同原子指标，为保障所有统计指标统一、标准、规范地构建，业务限定在业务板块内唯一，并唯一归属一个来源逻辑表，计算逻辑也以该来源逻辑表模型的字段为准，进行确定性定义，每个业务限定取来源逻辑表模型相关的所有逻辑表追溯所属分类，故而可能归属多个数据域，从而实现命名、逻辑归一，主题分类规范化、标准化、系统化。

业务限定清单查看与管理

支持对已选择的项目下的业务限定清单进行查看，包括业务限定的名称、创建人、发布状态。支持清单列表中查找或搜索查找以定位具体的业务限定，并对该业务限定进行编辑、下线、删除等操作。

业务限定查看与管理

为保证业务限定生产规范化，仅支持从逻辑表及其对应模型上定义业务限定，用户可以选择来源表，在对应雪花或星型模型上选择字段，定义基于该字段的计算逻辑为业务限定。

43.5.5.5 派生指标

派生指标即常见的统计指标，为保证统计指标标准、规范、无二义性地生成，OneData方法论抽象为如下四部分：

- 原子指标：明确统计口径，即计算逻辑
- 业务限定：统计的业务范围，筛选出符合业务规则的记录
- 统计周期：统计的时间范围，比如最近一天，最近30天等
- 统计粒度：统计分析的对象或视角，定义数据需要汇总的程度，可理解为聚合运算时的分组条件（即SQL中的group by）。粒度是维度的一个组合，比如某个指标是某个卖家在某个省份的成交额，那么粒度就是卖家、地区这两个维度的组合。

基于上述定义组合机制，即可完成派生指标批量、无重复地快速创建，且保证概念定义、计算逻辑明确而不会重复，业务人员也可自主定义生产。派生指标作为与字段同级的概念，在同一个统计粒度内唯一，保证对象组合的统计数据定义的唯一与确定。

派生指标清单查看与管理

支持对已选择的项目下的派生指标清单进行查看，包括派生指标的名称、创建人、发布状态。支持清单列表中查找或搜索查找以定位具体的派生指标，并对该派生指标进行编辑、下线、删除等操作。

派生指标查看与管理

为保证标准化生产派生指标，指标计算统计的范围与对象都依赖与统计计算逻辑而定，故而需要选定一个原子指标为基础，进一步选择与该原子指标有关的统计粒度、统计周期、业务限定进行组合，一键批量、规范、标准化生成新的派生指标。

- 确定统计粒度

  选择统计粒度即选择此粒度对应的维度组合，选择框中的维度为原子指标所属逻辑表模型的所有，有关联关系的维度，从而保证该统计粒度的统计计算有意义且可实现。

- 确定统计周期

  Dataphin系统提供常用统计周期供使用，也支持用户在全局设计-业务总线自定义增加更多的统计周期，满足不同统计时间范围计算需求。

- 确定业务限定

  业务限定即在逻辑表上定义生成的限制或过滤条件。业务数据需求会存在一组或一类的情况，如需要最近1天、最近7天、最近30天等不同统计周期、相同统计范围与计算逻辑的指标，故而Dataphin支持统计粒度、统计周期、业务限定定义多个，以自由组合批量生成指标，规范标准的同时提高开发效率。

43.5.6 建模研发

Dataphin提供体系化、系统化建模及研发的功能，将数据仓库理论以工具化半自动化实现：自顶向下快速构建业务维度、业务过程，并进一步细化开发维度表、事实表、汇总表、应用层，沉淀标准统一的数据资产，便于业务分层数据应用的同时优化计算存储。

43.5.6.1 维度逻辑表

维度逻辑表是对维度逻辑模型的细化，包含维度的详细信息。支持对已有维度逻辑表的清单查看和管理，对单张维度逻辑表的可视化查看及编辑。

维度逻辑表清单查看与管理

支持对已选择的项目下的维度逻辑表清单进行查看，包括维度逻辑表的名称、描述、创建人、创建时间、发布状态。支持清单列表中查找或搜索查找以定位具体的维度逻辑表，并对该逻辑表进行编辑、删除等操作。

支持可视化的方式进行单张维度逻辑表模型的详情查看，包括：主表涉及的已定义的主键、关联维度、属性等；关联维度表在内的星型模型和雪花模型概况；父子维度表的模型信息（如果已定义继承关系）。支持解锁编辑后发布；支持对画布缩放查看；支持跳转查看线上版本。支持可视化的方式对单张维度逻辑表模型进行编辑，包括：定义维度属性、关联维度、子维度；配置逻辑表物理化参数。

43.5.6.2 事实逻辑表

Dataphin支持用事实逻辑表来描述特定过程（即业务过程，如下单、支付等）或者状态度量（如账户余额、库存量等）属性的数据仓库模型。事实逻辑表的构建是基于优化的类雪花模型完成的，即：允许将除度量和关联维度外的属性信息退化至事实表，作为事实属性，并可以进行归类，降低了模型设计复杂度和用户使用友好度。

事实逻辑表清单查看与管理

支持对已选择的项目下的事实逻辑表清单进行查看，包括事实逻辑表的名称、创建人、发布状态。支持清单列表中查找或搜索查找以定位具体的事实逻辑表，并对该逻辑表进行编辑、下线、删除等操作。

单事实逻辑表查看与编辑

支持表单及可视化的方式进行单张事实逻辑表模型的详情查看，包括：主表涉及的关联维度、度量、事实属性等；主表关联的维度逻辑表。支持解锁编辑后发布；支持对画布缩放查看；支持跳转

查看线上版本。支持可视化的方式对单张事实逻辑表模型进行编辑，包括：基本信息定义、主键定义、字段定义；配置逻辑表物理化参数。

43.5.6.3 汇总逻辑表

汇总逻辑表是重要的数据仓库模型之一，模型内涉及两类元素：一类是描述某种统计粒度（由N个维度组成，N>=0，如：省份+产品线）的各种统计值（即派生指标，如最近7天销售额），另一类是组成统计粒度的各个维度（如：省份、产品线）的属性信息（如：省份名称、产品线名称、产品线等级等）。

汇总逻辑表清单查看与管理

支持对已选择的项目下的汇总逻辑表清单进行查看，包括汇总逻辑表的名称、创建时间。支持清单列表中查找或搜索查找以定位具体的汇总逻辑表，并对该逻辑表进行编辑、下线、删除等操作。

单汇总逻辑表查看与编辑

汇总逻辑表构建支持两种情况：自动化汇聚规范定义的派生指标，兼容编码研发的物理表字段上挂。

43.5.6.4 代码自动化

维度逻辑表、事实逻辑表、汇总逻辑表提交发布后将自动完成物理模型设计、自动完成相关代码、自动生成对应调度任务（一个逻辑表一般会涉及多个任务）以生产所需数据，可在调度运维中查看了解其执行逻辑。

43.5.7 编码研发

编码研发是Dataphin数据处理研发过程与建模研发并行的重要数据研发方式之一，提供用户基于计算引擎的代码编写方式编辑脚本文件，并可提交至调度系统进行任务生产，同时支持追溯查看节点版本，从而完成通用数据开发。脚本文件会有多种类型（包括SQL、Shell、MapReduce等），对应不同的编写配置要求（如代码的语法特征、调度配置要求），提交发布成功会生成对应任务进行运行、生产数据，在调度运维的DAG图中又被称为节点。编码研发的核心功能点包括：代码文件管理（增删改查）、代码编辑、任务调度配置及发布、节点版本管理。其中数据同步任务DataX见数据引入。

43.5.7.1 代码编辑器

代码编辑器提供在线代码编辑界面以完成数据开发任务。支持SQL编程、MR编程、Spark编程、Shell编程。

43.5.7.2 任务调度配置及发布

调度配置

支持手动和周期性任务的调度配置， 可对已经完成配置的任务进行发布，系统支持自动检测任务调度配置的完整性，对配置完整的方会放行进行发布，所有发布的任务会生成周期性任务在调度运维-周期性任务列表中展示。

提交发布

有权限控制地允许项目成员操作，仅参数配置全面、依赖关系内容有效、非循环依赖的调度可以完成提交发布，生成调度任务，保证数据可以如期、稳定、有序生产。

43.5.7.3 代码管理

支持代码文件的新增、删除、更新、重命名、文件夹分类管理、内容编辑与查看等操作，方便代码文件整理、使用。

文件管理

支持单文件编辑、删除、下线、重命名操作管理之余，可查看单代码文件的发布状态、创建人和创建时间。这样，可实现整个大数据编码研发的文件的便捷创建、清晰查看与系统化管理。

文件夹管理

代码文件过多时需要进行分类管理，保证文件有序分布与展示，支持新建、重命名、删除文件夹，历史文件和新建的代码文件可被操作移动至对应文件夹目录内进行管理，支持多级文件夹管理。

43.5.7.4 团队协作编程

节点版本管理

支持对任务节点版本的历史追溯，可查看版本号、提交人、提交时间、提交备注，并可以查看各版本的代码详情，以比较差异性。目前面向Maxcompute_SQL、ODPS MR、Shell等节点类型。

协作开发

为了支持多人同时编辑协作，提高开发效率，提供脚本文件的锁机制，以有效保障协同编辑与冲突解决，即保证同一个代码在同一时间内只能被一个用户所编辑，用户通过获取锁的方式来得到编辑权限，被偷锁者也可获取偷锁进行，进行相应处理操作。

43.5.8 资源及函数管理

资源及函数管理是编码研发中重要的辅助功能，数据开发者可以上传本地的资源，在节点运行时调用以实现特殊的加工诉求，或者直接使用计算引擎支持的脚本语言所对应系统自带函数，实现常见的数据加工处理。特别地，对于较高频率处理某类数据数据逻辑且无法用系统内建函数解决（如：按照某种业务逻辑进行数据转换）时，可基于上传的资源注册自定义函数实现。

43.5.8.1 资源管理

支持当前项目内数据开发者进行资源的新增、修改等操作。通过新建、编辑，可完成资源文件的命名、上传，然后复制引用至代码中，也可以手工删除不需要的资源文件。

资源新建与上传

支持上传的本地资源文件类型包括如下，新增文件类型在标准接口下支持3日快速扩展，资源名称在项目空间中唯一、提交成功后文件名及上传的资源包不可改、上传仅支持单个文件并提示文件与所选类型保持一致。

资源引用

通过copy引用即可复制并粘贴本资源至所需应用的代码编辑框的对应代码位置，编写相关执行语句调用该资源。

资源更新

支持对已管理的资源进行描述信息的编辑更新，也支持删除已有资源，达到空间合理利用的目的。

43.5.8.2 函数管理

函数管理可以满足函数的查找、使用、管理。函数分为两类：一类是系统默认的内建函数，另一类是基于已上传的jar或python等资源的自定义函数。自定义函数支持在引用标准化的函数基础上进行扩展。

新建自定义函数

自定义函数要求其名称在项目空间中唯一，且注册后不可修改名称。

函数引用

系统内建函数或自定义函数，支持copy引用即可复制并粘贴本函数名至所需应用的代码编辑框的对应代码位置，以示例的命令格式编写相关执行语句进行加工处理。

函数更新

支持自定义函数的更新，即：可以编辑变更函数的相关信息（名称除外），也可以删除注销不再需要的自定义函数。

43.5.9 调度运维

调度运维子产品作为数据研发工作后期环节的日常化维护控制部分，提供所有数据处理任务（周期性任务及手动任务）的清单、任务依赖关系DAG图、实际任务（周期性任务、手动任务、补数据任务）运行的实例清单、运行实例的依赖关系及状态DAG图，可实现任务执行时序安排、执行进程拆分、执行机器资源最佳分配、异常任务发现，保证所有任务能如期、稳定、可靠地执行并生产出数据，如有运行问题可及时管控。目前调度运维的功能模块包括任务列表和任务运维2大部分。

任务列表

任务列表提供不同项目下的周期性任务及手动任务的任务清单列表及任务依赖关系DAG图。

周期性任务

周期性任务，支持任务清单、任务查找、单任务的依赖关系查看。支持切换项目查看和查找任务，支持以任务节点名称和节点ID的模糊匹配搜索查找任务，支持对指定人员的任务节点和今天发布的任务节点的二次筛选，可对任务进行缩小范围或精确定位进行运维。

手动任务

支持任务清单、任务查找、单任务的详细情况查看。支持切换项目查看和查找任务，支持以任务节点名称和节点ID的模糊匹配搜索查找任务，支持对指定人员的任务节点和今天发布的任务节点的二次筛选，可对任务进行缩小范围或精确定位进行运维。

实例运维

提供不同项目下的周期性任务实例、手动任务实例、补数据实例的清单列表及任务实例运行详情。

周期性实例

支持实例清单查看、实例查找、单实例的详情查看。可查看所有常规实例的运行状态、对应任务的唯一性编号NodeID、节点名称、任务的负责人、任务运行时间开始和结束时间及时长等信息。支持切换项目查看和查找任务实例，支持以任务节点名称和节点ID的模糊匹配搜索查找任务，支持对我

的实例、出错实例、未完成节点的二次筛选，同时也支持对运行日期进行二次筛选，可实现对实例缩小范围或精确定位进行运维。

手动实例

支持实例清单查看、实例查找、单实例的详情查看。可查看所有手动实例的运行状态、对应任务的唯一性编号NodeID、节点名称、任务的负责人、任务运行时间开始和结束时间及时长等信息。支持切换项目查看和查找任务实例，支持以任务节点名称和节点ID的模糊匹配搜索查找任务，支持对我的实例、今天发布实例的二次筛选，可实现对实例缩小范围或精确定位进行运维。

补数据实例

提供补数据实例的清单，可查看补数据的节点名称，补数据的时间分区范围信息及状态，被补数据的任务节点ID、名称、负责人，补数据的运行时长信息。也支持对补数据实例的搜索、筛选等查找，便捷定位所需查看的具体实例。

43.5.10 元数据中心

Dataphin包含强大的元数据管理能力，支持MaxCompute、Hadoop、Hive、MySQL、PostgreSQL、Oracle等元数据的采集与抽取，支持对于上述计算/存储引擎中元数据的实时追踪，并通过对不同类型存储元数据的抽象，构建统一元数据模型，支持多类型元数据快速扩展，通过元数据中心建设，实现元数据丰富多样、标准统一、保障稳定，为数据地图、数据治理提供强大的元数据保障。

元数据中心是数据资产管理的核心基础，元数据中心建设内容包括：

- 元数据采集规范：维护统一标准的数据建设规范，确保模型建设、数据表创建、血缘依赖关系记录都能够一致，提高元数据在检索及服务中的可用性。
- 元数据时效与质量：保障元数据产出时间、数据质量，提高资产管理应用数据时效性及研发用户的精准检索。
- 元数据模型体系：通过构建统一的元数据公共模型，兼容多种数据类型，保障数据地图的一站式服务能力。

43.5.11 资产分析

作为数据工作后期环节的发现获取部分，基于OneData和数据资产的方法论设计应用原则以及元数据全面采集提取、解析管理及加工的技术内核，实现所有数据如资产般分类整理、异常发现优化，保证所有数据的成本最小化消耗、价值最大化呈现并用于业务。

驱动数据资产管理功能的是一系列技术内核，实时事件/订阅服务支撑表、任务等元数据的实时更新，规则引擎支撑治理项规则的高效准确判定与健康分模型的构建，日志动态解析支撑每日海量生

产任务执行与机器运维日志的分析，图计算则支撑起数据血缘关系的分析与构建，Onelog全链路数据追踪技术则支撑起数据生产、服务与消费过程中端端元数据的互通，插件式的元数据接入与加工架构，则支撑了对多计算/存储引擎的兼容数据资产管理是阿里巴巴基于多年海量数据管理经验沉淀出的一整套集分析、治理、应用、运营为一体的方法论与产品，覆盖数据的建、管、用、销全生命周期。

资产分析的主题包含两个关键词：“全域”和“融通”。“全域”是对全域数据的盘点，是通过OneData体系中维度、业务过程、关联关系等构建起数据资产大图，即以模型的语言来盘点描述数据资产； “融通”则是对数据资产在生产过程中的成本与价值进行分析，通过连接度、贡献度两大模型，描述出数据资产中不同数据集在资产大版图中发挥的不同作用。

Dataphin数据地图模块，以对企业数据资产分析构建起的数据资产目录为基础，综合用户使用行为，通过元数据画像+搜索引擎，实现对企业数据资产的高效检索。

资产全景

基于OneData构建的企业数据资产，可以结构化展示，以不同形状的物料组件表示业务实体，以不同样式的连线表示实体间不同的业务过程关系， 如此将同一业务板块下数据全景清晰描绘。

资产地图

通过业务板块-数据域-维度&业务过程的关系汇总展示了企业数据构成，并与企业资产全景相对应；同时，结合用户搜索、访问及收藏等主动行为，为用户提供高效、快捷、准确的数据查找与探索的入口。

43.5.12 安全管理

随着对大数据技术的应用深化，数据安全也成为重要课题，国内 《中华人民共和国网络完全法》已于2017年6月1日开始实施，鼓励开发网络数据安全保护和利用技术，国际上《欧盟数据保护条例》(General Data Protection Regulation) 将于2018年5月25日生效，旨在加强个人信息等数据保护。Dataphin聚焦智能数据的构建与管理，对于数据安全管理非常重视，面向数据从产生到销毁全生命周期，提供数据访问控制与隔离、数据安全分类分级、个人信息合规管理、数据脱敏、数据使用安全审计等能力，全面保障数据安全。

数据安全管理最首要的工作是数据访问控制与隔离，Dataphin提供了完善的数据使用权限申请、审批及其生命周期管理，支持多租户访问隔离措施，支持字段粒度的权限管控，并提供基于ACL的数据访问授权模型。

Dataphin基于数据生命周期构建全面的数据安全保障体系，从数据行为、数据内容。数据环境等角度提供技术和管理措施，在大数据构建与管理过程中，Dataphin结合阿里云数据安全管控体系，提供“可用不可见”的大数据交换共享安全环境、字段粒度的权限访问管控、严格的权限申请审批流程管控、健全的数据使用的行为追踪及审计能力等，保障大数据在“存储、流通、使用”全过程中的安全管控。

目前Dataphin提供权限分层分级及权限的申请、审批、赋权、交还及鉴权的全链路管理流程。

43.5.12.1 权限类型

为了保证用户能安全、有控制地使用产品及访问数据，Dataphin提供角色和资源两个类型的权限机制。

角色权限

为了统一管理用户在平台上的操作，Dataphin除了提供账号管理机制获取超级管理员及系统成员，从平台层面控制用户准入方式，也提供基于项目管理的组织粒度的数据资源获取与操作的权限管理，称之为角色管理，以批量、可管控地赋予系统内用户一组数据资源及操作权限。

资源权限

为了统一管理用户在对项目中数据资源操作，Dataphin提供数据访问控制机制，在项目空间独立管理、成员与资源逻辑隔离的情况下，控制跨项目的存储资源的访问，实现数据在不做迁移时也可在不同项目空间内使用，达到数据共享的目的。

43.5.12.2 权限管理

权限申请

数据研发者在使用“数据地图”找到所需数据表并查看该表详细元数据信息后，使用该表需要预先申请权限。

在具体的权限申请过程中，Dataphin支持根据来源数据表信息，自动显示该表类型和业务板块归属等信息，同支持显示该表字段元数据。权限申请流程支持对遵从最小化按需申请原则的响应，即：支持字段级别的权限申请，支持多种权限时效的选择（自定义选择日期区间或者快速选择30天、90天、180天及1年），支持用户输入权限申请使用场景及目的说明，以便审批人可以据此判断是否可授权从而实现根据场景按需申请。

申请记录管理

支持通过权限列表查看申请记录及当前审批状态，并可通过点击“详情”可查看申请信息，点击“撤回”申请工单撤销。对于审批通过的权限，可以查看相关列表清单及具体的字段级信息。

权限审批

权限提交后，系统支持将权限审批的工单随机分发给数据表所在项目的管理员进行审批，在审批人视角，可以通过“我的审批”查看申请人提交的申请信息，并进行授权或驳回。

权限交还

当用户的角色有转岗或者离职变动时候，需要提前交还权限，以防止数据及对应生产任务无人管理的情况发生。Dataphin的安全管理支持在“我的权限”页面，点击“交还”按钮，将权限转交给项目管理员，实现权限的有效回收。

43.5.13 即席查询

即席查询，是基于Dataphin强大的OneService引擎，提供高性能的数据临时查询与探索需求，既支持传统的简单查询方式，也支持主题式查询，在代码简洁度、查询执行速度等方面表现优异。

语法特点

- 支持基于所有建模的逻辑表的离线查询，智能查询引擎会基于产出时间、查询性能等因素选择最优的物理表。
- 支持基于雪花模型的关联查询, 使SQL更加简洁和智能。
- 支持物理表、逻辑表以及物理表和逻辑表的混合查询。
- 支持 MaxCompute SQL、Hive SQL等多计算引擎的语法。
- 支持SQL的智能提示、预编译、 格式化等功能。
- 支持逻辑表与物理表字段级别的权限管理与鉴权。

查询执行

在查询脚本文件中，可以根据需要自由输入查询语句，脚本编辑器会根据输入智能给出提示，快速定位所需要的数据表或者字段，同时也会帮助用户校验脚本语法的有效性。

44 实时数据分发平台DataHub

44.1 什么是DataHub

阿里云实时数据分发平台（DataHub）是流式数据（Streaming Data）的处理平台，提供对流式数据的发布（Publish）、订阅（Subscribe）及分发功能，让用户可以轻松构建基于流式数据的分析和应用。

DataHub服务可以对各种移动设备、应用软件、网站服务、传感器等产生的大量流式数据进行持续不断的采集、存储和处理。用户可以编写应用程序或者使用流计算引擎来处理写入到DataHub的流式数据（例如：实时web访问日志、应用日志、各种事件等），并且能够产出各种实时的数据处理结果（例如：实时图表、报警信息、实时统计等）。

DataHub服务基于阿里云自研的飞天操作平台，具有高可用、低延迟、高可扩展、高吞吐的特点。DataHub与阿里云流计算引擎StreamCompute无缝连接，用户可以轻松使用SQL进行流数据分析。

DataHub服务也提供分发流式数据到各种云产品的功能，目前支持分发到MaxCompute、OSS等。

44.2 产品特性和核心优势

高吞吐

单**主题**（`Topic`）最高支持每日TB级别的数据量写入；每个**分片**（`Shard`）最高支持每日8000万Record级别的数据量写入。

实时性

通过DataHub，用户可以实时的收集通过各种方式生成的数据并进行实时的处理，对用户的业务产生快速的响应。

易用性

- DataHub提供丰富的SDK包，包括C++、JAVA、Pyhon、Ruby、Go等语言。
- DataHub服务也提供Restful API规范，用户可以使用自己的方式实现接口访问。
- 除了SDK以外，DataHub还提供一些常用的客户端插件，包括：Fluentd、LogStash、Flume等，用户可以使用这些客户端工具向DataHub中写入流式数据。
- DataHub同时支持强Schema的结构化数据和无类型的非结构化数据，用户可以自由选择。

高可用

- 规模自动扩展，不影响对外服务。

- 数据自动多重冗余备份。

动态伸缩

每个**主题**（**Topic**）的数据流吞吐能力可以动态扩展和减少，最高可达到每主题256000 Records/s的吞吐量。

高安全性

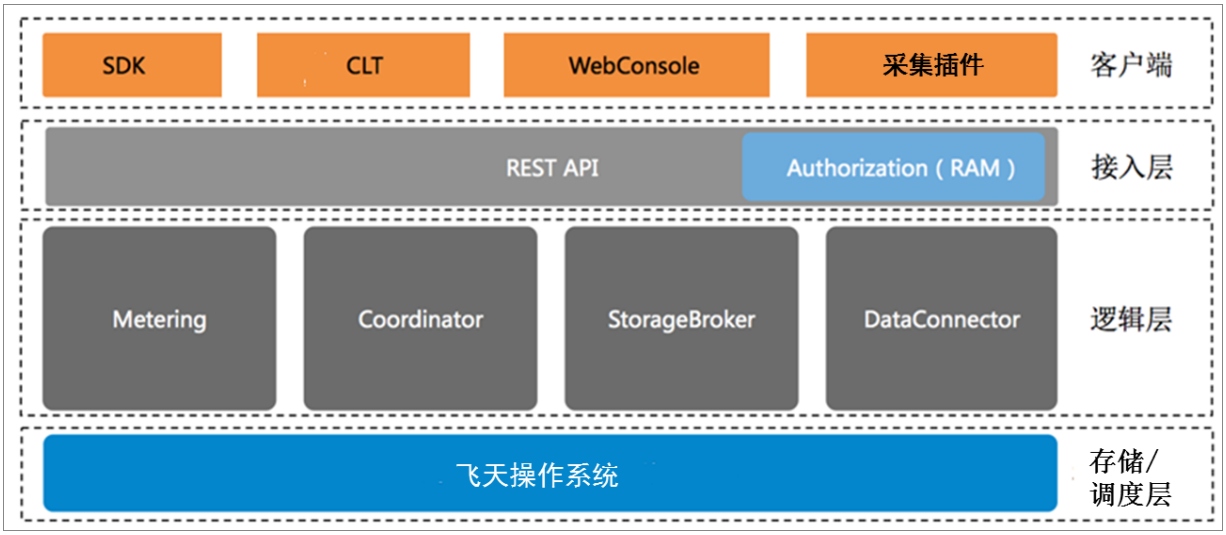
- 提供企业级多层次安全防护，多用户资源隔离机制。
- 提供多种鉴权和授权机制及白名单、主子账号功能。

44.3 产品架构

44.3.1 功能架构

DataHub的产品功能架构如图 *44-1: DataHub产品功能架构图*所示：

图 44-1: DataHub产品功能架构图



DataHub从功能上可分为4层，分别为**客户端**、**接入层**、**逻辑层**及**存储/调度层**。

客户端

DataHub的客户端有以下几种形式：

- SDK：提供C++、JAVA、Pyhon、Ruby、Go等语言的SDK。
- CLT（Command Line Tool）：支持在Windows/Linux/Mac下运行的命令行工具，可以通过命令进行project/topic的管理。

- Web Console：可视化页面，可以在页面上进行project/topic管理、创建订阅、查看shard状态、topic性能监控、管理Connector等操作。
- 采集插件：包括Logstash、Fluntd、Flume以及Oracle GoldenGate。

接入层

DataHub在接入层提供Http（Https）服务和RAM鉴权，支持水平扩展。

逻辑层

DataHub逻辑层处理整个服务的核心逻辑，提供Project/Topic管理、数据读/写、点位存储、流量统计、数据对接等核心功能，根据基本功能可以分为以下几个模块：StorageBroker、Metering、Coordinator、DataConnector。

- StorageBroker：提供最核心的数据读写功能，使用飞天分布式文件系统的LogFile为存储模型，比传输的WAL降低一半集群磁盘读写；一份数据三份存储，单机异常不丢数据；支持跨机房容灾；实时数据写入缓存，保证实时消费场景；历史数据有独立的读缓存，保证读历史数据多份订阅的性能。
- Metering：支持shard维度使用时长计费。
- Coordinator：提供点位存储功能，单机QPS高达150000，支持水平扩展。
- DataConnector：提供数据同步功能，支持将DataHubk的数据自动同步到阿里云产品线的其它服务，目前支持以下服务：MaxCompute、OSS、AnalyticDB、RDS、TableStore、Elasticsearch。

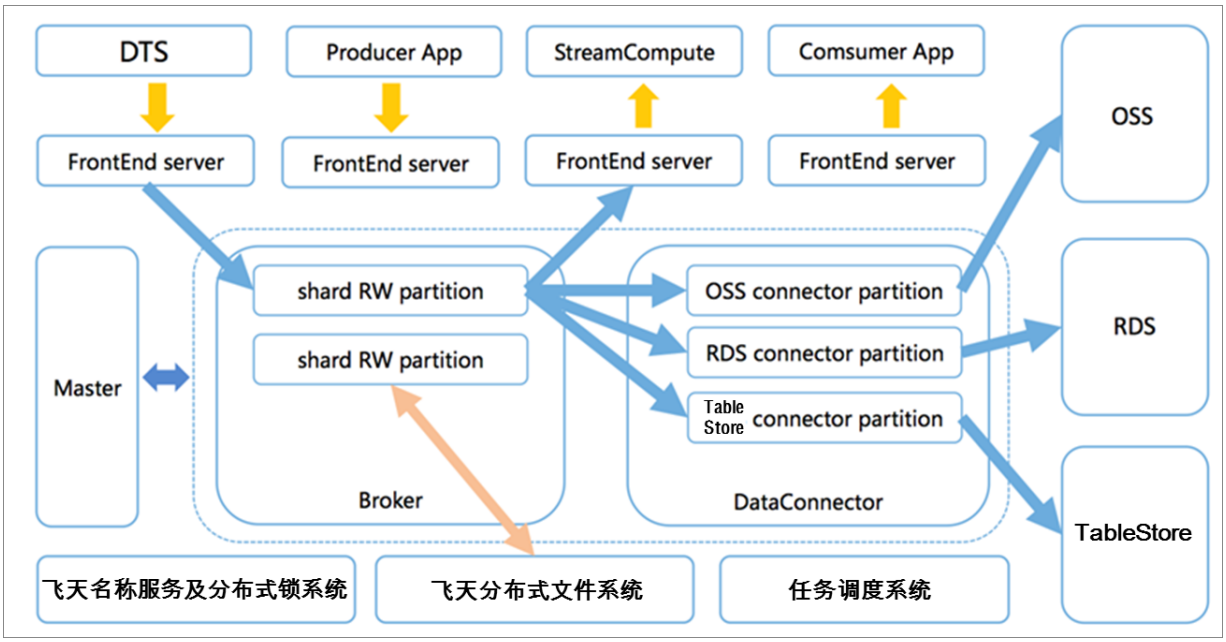
存储/调度层

- 存储：存储层基于飞天分布式文件系统的LogFile做为存储模型，Append模式写入数据，支持SSD混合存储，支持timestamp做为数据索引，单shard写单个飞天分布式文件系统文件，基于时间轴切分存储数据。
- 调度：基于任务调度系统的调度模块，结合partition模式和machine模式实现业务维度的调度算法，即按流量配置每台机器上的负责的shard；不占用任务调度系统的CPU/Memory单元，单机partition数据无上限；能够实现毫秒级Failover，支持热升级。

44.3.2 技术架构

DataHub的技术架构如图 *44-2: DataHub*技术架构图所示：

图 44-2: DataHub技术架构图



上图为数据从产生到消费一个基本流程。

1. DataHub中Shard做为数据管理的最小单位，逻辑上可以看做一个先进先出队列。
2. 物理上每个Shard对应飞天分布式文件系统上一组Logfile。
3. Master是以Shard为单位进行调度，每个shard会被调度到一台Broker机器上，每台Broker负责多个Shard读写。
4. Frontend server会根据请求中的project/topic/shard组合定位到一台broker并将数据请求转发到Broker上。
5. DataHub Connector做为服务的内部模块可以直接从Broker上读取数据，并将数据转发给其它服务。

数据采集

DataHub支持多种数据写入方式：使用SDK开发的数据应用、DataHub的数据采集插件（LogStash/Fluntd/Flume）、数据采集服务（DTS）、流利算（StreamCompute）产生的数据。

Frontend Server

提供Rest API接口和鉴权服务，做为DataHub服务的统一入口层，支持水平扩展。

Master

提供Meta管理服务和Shard调用服务，支持project和topic基本的增删改查、shard分裂/合并以及shard调度。

Broker

负责每个Shard中数据的读/写功能，包括数据的索引、缓存、数据文件的组织和管理。

Connector

Connector的功能是从DataHub中将数据转发到阿里云的其它服务中。针对不同的服务Connector提供不同的功能点，例如MaxCompute的自动创建分区、OSS的自动切分文件等功能。

44.4 产品价值

DataHub具有如下的价值点：

表 44-1: 价值点

价值点	DataHub
安全性	高（基于阿里云RAM权限体系）
运维难度	容易（自动化的上下线）
资源隔离	租户隔离
生态整合	丰富
规模扩展性	强（经历过双十一的考验）
读写功能	索引机制导致实时入库数据难以支持分析型应用，海量数据碰撞比对、分析预测计算时间可能超过24小时，性能无法满足需求。
高可用	丰富
与阿里云上下游产品整合	无缝

44.5 功能描述

44.5.1 数据队列

DataHub的基本功能，单shard内数据保序；DataHub对Shard的读/写性能提供SLA的保障；单topic的性能以shard数为单位水平扩展。

44.5.2 点位存储

支持消费应用将消费点位保存到DataHub服务，保证消费应用在Failover后可以从保存的点位进行消费。

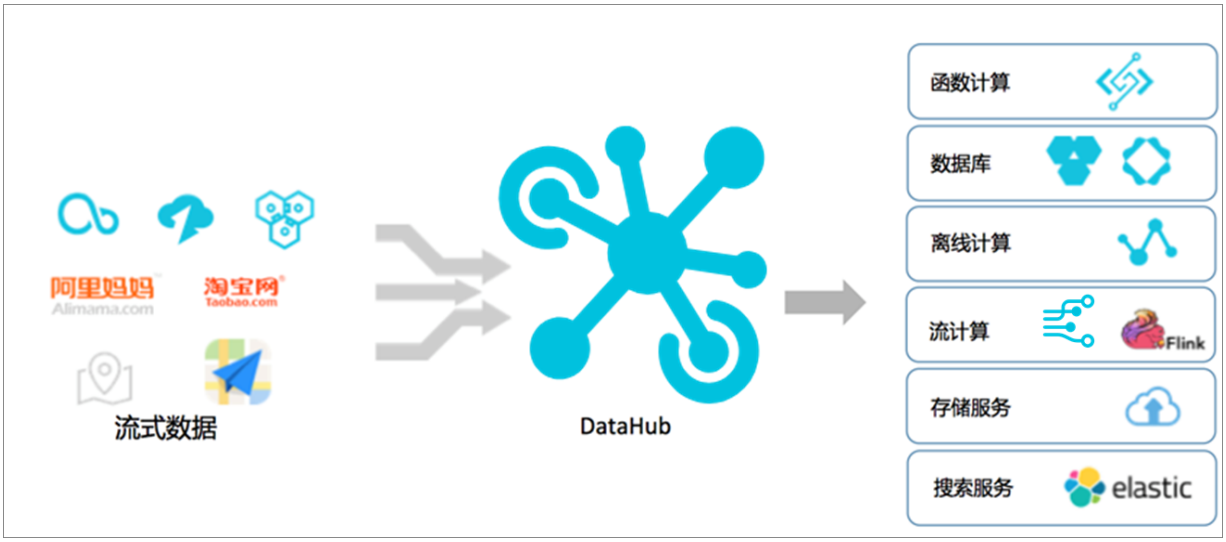
44.5.3 数据同步

DataHub中的数据自动同步到阿里云其它服务，目前支持的服务包括：MaxCompute、OSS、AnalyticDB、RDS、TableStore、Elasticsearch。支持标done功能，确认某一个时间点之前的数据已经全部同步完成。

44.6 应用场景

44.6.1 数据上云入口

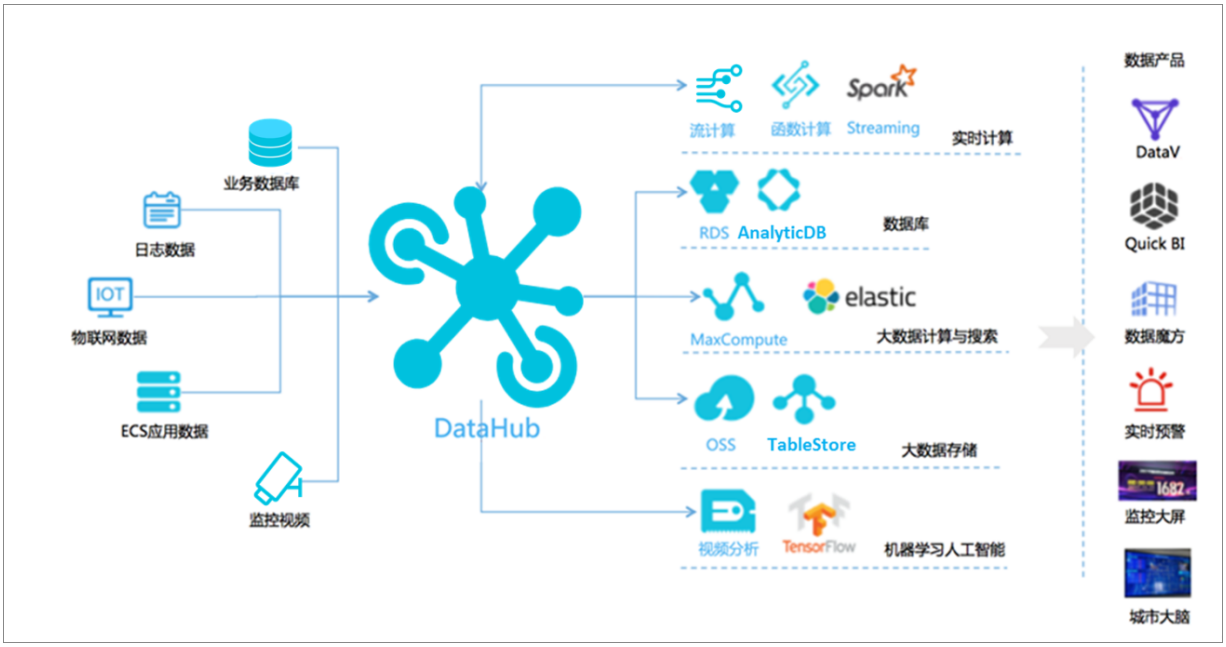
图 44-3: 数据上云入口



DataHub做为数据上云的入口，可以简化用户将数据上传到阿里云的接口，让用户做到一次写入多处使用，无需对接多种云服务的接口。

44.6.2 数据采集通道

图 44-4: 数据采集通道



DataHub提供丰富的采集端插件，支持多种业务数据的采集，让用户可以方便的将现有的业务数据采集到DataHub，在云端进行更多的处理和使用。目前DataHub支持的采集端包括日志采集（Logstash/Fluntd/Flume）、数据库binlog采集（dts/oracle goldengate）、视频采集（GB28181协议的监控/安防视频）。

45 在线图计算服务BigGraph

45.1 什么是BigGraph

在线图计算服务（BigGraph）是阿里巴巴内部研发的一款低延时高可用的分布式图计算产品，目前在阿里巴巴集团内部用于账户安全风控、安全流量分析、知识图谱等场景。相对于传统图数据库，BigGraph不仅具备低延时的交互式查询能力，还具备图分析/计算能力，可通过接口实现用户算法，满足图分析需求。

BigGraph产品围绕整个图实例元数据创建、数据导入、数据查询及展现等几个方面，其提供的接口和API，全面支撑图查询应用的业务构建。当前构建在BigGraph上的应用有阿里云关系网络分析平台、阿里集团搜索商品风控平台等。

依托BigGraph可以一站式的搭建图查询应用，可根据具体场景需要实现不同的图应用产品，例如专注场景查询的可以对接**查询研发**块，想致力于研发满足通用业务场景解决方案的可以对接所有图管理、研发相关的API，产品层面进行统一的功能封装。

45.2 产品优势

海量图数据查询

BigGraph能够在百亿节点、千亿边、TB级别数据量级中，对用户数据进行实时查询和响应。

丰富的查询接口

BigGraph不仅支持Apache TinkerPop™ Gremlin图查询语言，也支持类GAS算法查询接口，产品兼备Gremlin语言的便捷性和GAS接口的高效性、灵活性。

高效的执行引擎

BigGraph重新实现Apache官方的Gremlin执行引擎，充分利用分布式的计算能力，以key查询可达毫秒级。GAS接口搭配分布式执行引擎，兼顾效率和灵活性，在秒级别完成分析型任务的执行。

多租户支持

BigGraph支持图实例资源、数据、权限完全隔离，可管理和使用不同的图实例满足不同的业务需求。

图实例的高可靠和高可用

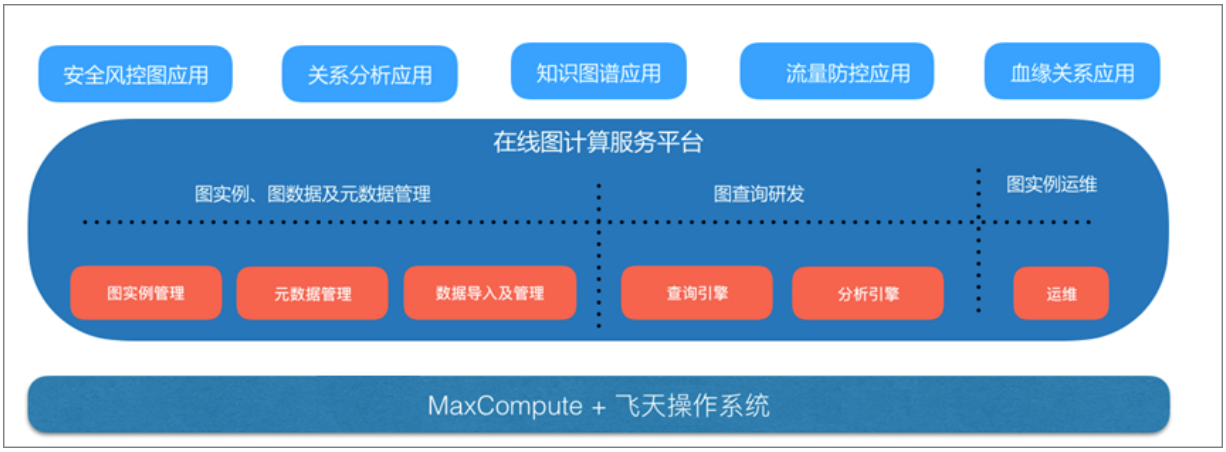
- 依靠图多副本策略，可以实现图实例的高可靠。
- 依靠飞天操作系统提供的调度能力，可以实现图实例的高可用。

45.3 产品架构

45.3.1 功能架构

BigGraph的架构如下图所示：

图 45-1: BigGraph架构图

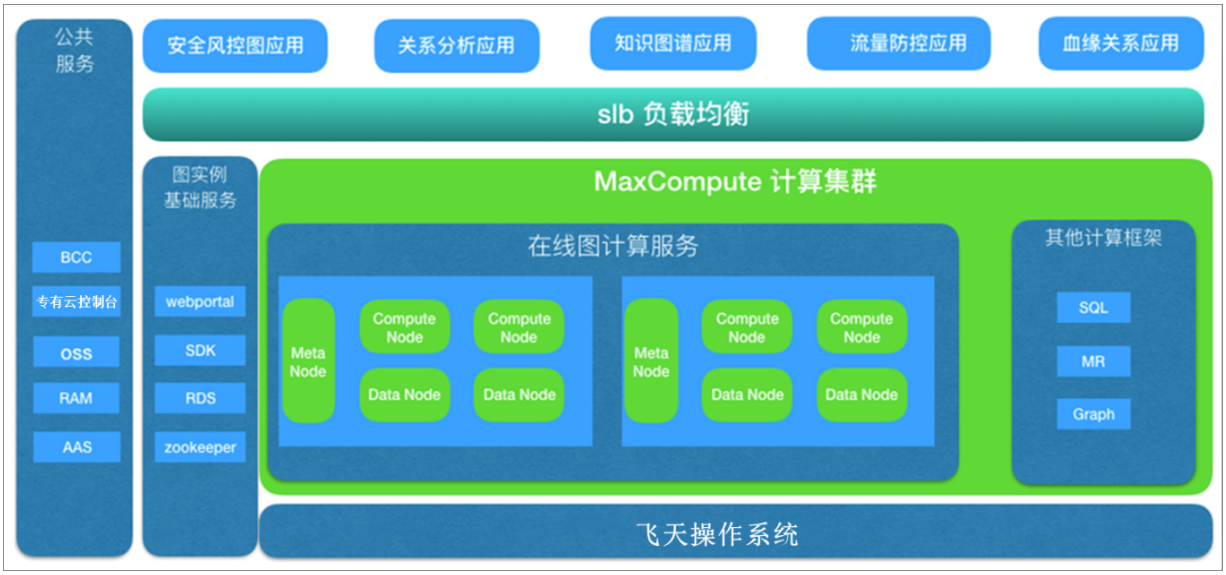


在线图计算平台提供全面的图应用开发能力，包括如下模块：

- 图实例管理：提供图实例初始化、图实例权限管理等。
- 元数据管理：管理图中点、边的属性、主键及索引等。
- 数据导入及管理：提供一站式MaxCompute数据导入至图服务，并能够管理数据导入任务和已导入的数据。
- 查询引擎：又称交互式查询引擎，支持Apache TinkerPop™的Gremlin查询语言，并优化原有单机引擎为分布式引擎。
- 分析引擎：针对分析型作业（子图查询，秒级到分钟级），提供分布式计算引擎能力。
- 运维管理：提供图实例集群管理，图实例热升级、扩缩容、启动、停止等可视化运维操作。

45.3.2 系统架构

图 45-2: BigGraph系统架构



在线图计算服务平台（BigGraph）基于阿里云飞天操作系统之上的MaxCompute平台，提供了完整的资源分配、资源调度、图实例高可用的解决方案。不仅能够按需分配资源，而且可以在线对资源进行弹性伸缩。通过飞天资源调度系统能自动对宕机节点进行容错，确保系统遇环境故障能够自动恢复。

单个图实例分为计算节点、存储节点和元数据节点。系统通过双副本策略，不同角色的节点均有备份节点，能够做到容忍单点故障。

45.3.3 安全架构

在线图计算服务平台的安全架构，是由平台自身的安全实现层、平台内置的安全服务层构成的：

- 平台自身的安全实现层：保障平台在代码实现和部署配置时的产品自身安全性。
- 平台内置的安全服务层：为租户和其用户提供平台基础性的安全服务能力，例如：租户资源隔离。

45.3.4 多租户模型

在线图计算服务平台拥有自己的多租户权限模型：

- 弹性的存储和计算资源，租户可按需申请资源配额，独立管理自己的资源。
- 租户独立管理自有的数据、权限，彼此隔离，以确保数据安全。

45.4 功能描述

45.4.1 图实例管理

图实例管理可以分为图实例管理员和平台运维管理员两种角色进行管理，两种角色管理的功能各有不同。其中平台运维管理员可进行图实例的创建、修改、删除、热升级、停止等操作，主要侧重图实例服务的管理；而图实例管理员则可以进行图实例的授权和全局配置的初始化。

45.4.1.1 图实例运维侧管理

运维侧管理功能主要针对系统级的运维，侧重实例部署、升级、资源的申请、状态查看等内容。目前已实现的运维侧管理功能具体包括如下内容：

- 管理多个图实例集群，可以针对不同的计算集群创建多个图实例集群。
- 管理图实例。
  - # 按内存、CPU核数等资源配置，创建图实例。
  - # 在线热升级图实例（必须事先创建好对应版本）。
  - # 按需在线扩缩容图实例。
  - # 删除、停止图实例。
  - # 查看图实例状态、分配的机器等。

其中，扩缩容和热升级均支持在线操作。在线热升级的策略是**`rolling update`**（轮转升级），这就保证了一定范围内不影响用户服务，在热升级过程中可服务的节点/副本数会降低，所以在进行在线热升级的过程中，需要对业务水位进行控制。在线扩缩容则是指在正常服务过程中弹性伸缩可服务节点，避免停机重启、重新部署操作。

45.4.1.2 图实例用户侧管理

用户侧管理功能主要针对业务意义上的图访问权限、图API调用权限的管理以及对图的初始化，具体包括如下功能：

- API授权码查看及重置。
- 图实例管理员、普通用户权限的授予和删除。
- 图实例全局参数初始化，一般初始化分区数、副本数、数据导入方式等等。

图实例权限管理

图实例具备一定的访问控制能力，默认没有权限的用户不能访问对应的图实例。分为图实例管理员权限和图实例普通用户权限，不同权限间区别如下表所示。

说明：

在专有云体系内，用户一般指的是云账号，因此用户权限即云账号权限。

表 45-1: 不同权限间区别

角色	可访问图实例	可操作图实例	可授权
管理员权限	√	√	√
用户权限	√	√	×
未授权用户	×	×	×

45.4.2 元数据管理

元数据管理模块主要负责管理整个图实例的点、边类型的定义（或者叫元数据、Schema）。在线图计算服务平台中的图实例的元数据支持的是属性图模型，所以点、边类型上可以定义属性。具体支持如下功能：

- 点属性增删改。
- 点属性主键、分区键、索引增删改。
- 边属性增删改。
- 边属性索引增删改。

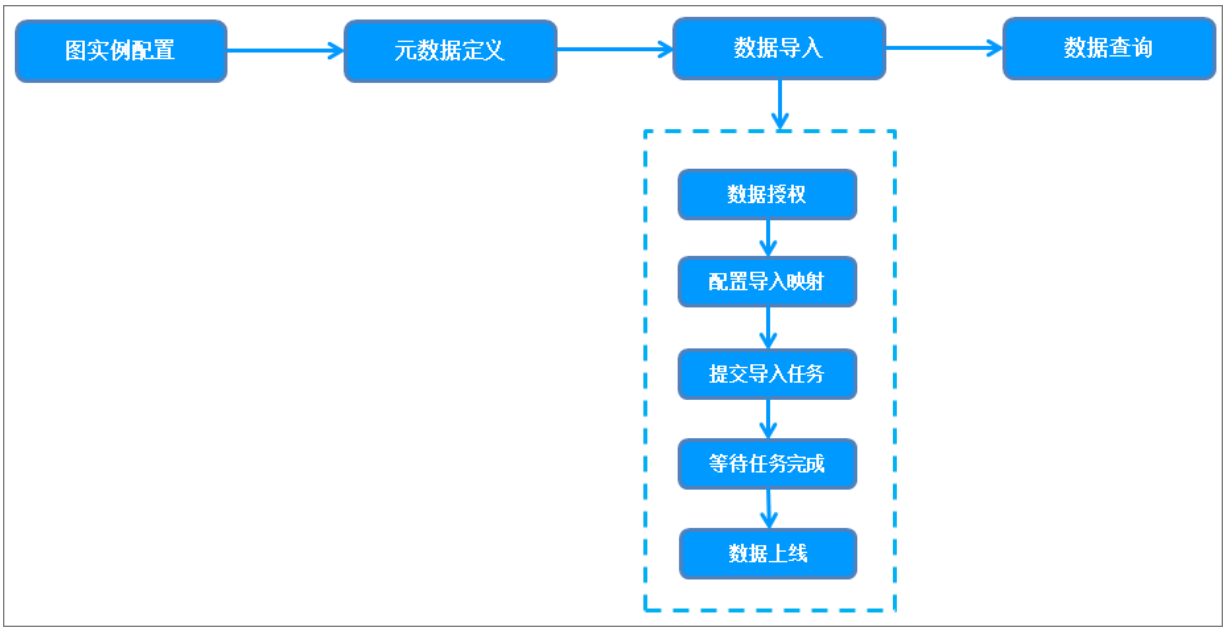
需要注意，图实例中的点类型必须要定义主键和分区键，分区键为空时默认使用主键。 定义主键是为了导入点、边数据时生成唯一的内置id，以便点、边关系能够被正确的映射。可以选择对属性定义索引，一般来说索引能够通过减少查询IO量，加快数据的查询效率；副作用则是会增加存储空间，即所谓用空间换时间。业务应用可以按照业务的查询、过滤条件选择要创建的索引。

元数据新增、修改或是删除操作完成后，必须等下一次导入才能生效。所以修改某类型元数据后在该类型数据导入之前，修改后的元数据不会体现在数据中。

45.4.3 数据导入

在图实例的元数据定义完成之后，可以开始进行数据导入，在线图计算平台提供一站式的从MaxCompute映射图数据并一键导入数据的功能。 一个完整的数据导入的Step by Step过程如下所示：

图 45-3: 数据导入流程



从上图可以看出，需要经过图实例配置、元数据定义之后，才能进行数据导入。数据导入本身也分为**数据授权**、**配置导入映射**、**提交导入任务**、**等待任务完成**以及**数据上线**等几个小步骤。

45.4.3.1 数据导入映射

用户需给要导入的表授权，授权之后才能配置数据导入任务，提交任务之后，就可以等待任务完成，任务完成时数据会自动上线。即任务完成之后，可以在可视化查询页面使用Gremlin语言进行数据查询。

对于给要导入的表授予权限，默认需要授予**test1000000001@aliyun.com**（默认内部账号）读数据的权限，具体授予如下的policy权限：

```
{
            "Action": ["odps:Select","odps:Describe"
            ],
            "Condition": {
                "StringEquals": {
                    "odps:TaskType": ["MapReduce","MR", "LOT"]
                }
            },
            "Effect": "Allow",
            "Principal": ["ALIYUN$test1000000001@aliyun.com"],
            "Resource": ["acs:odps:*:projects/${user_project}/tables/
*"]
```

```
        }
```

45.4.3.2 数据导入模式

数据导入的配置分为两种模式：横表导入模式和竖表导入模式。横表、竖表两种模式的导入支持提炼于阿里巴巴集团内真实场景，较为通用。具体导入模式描述如下。

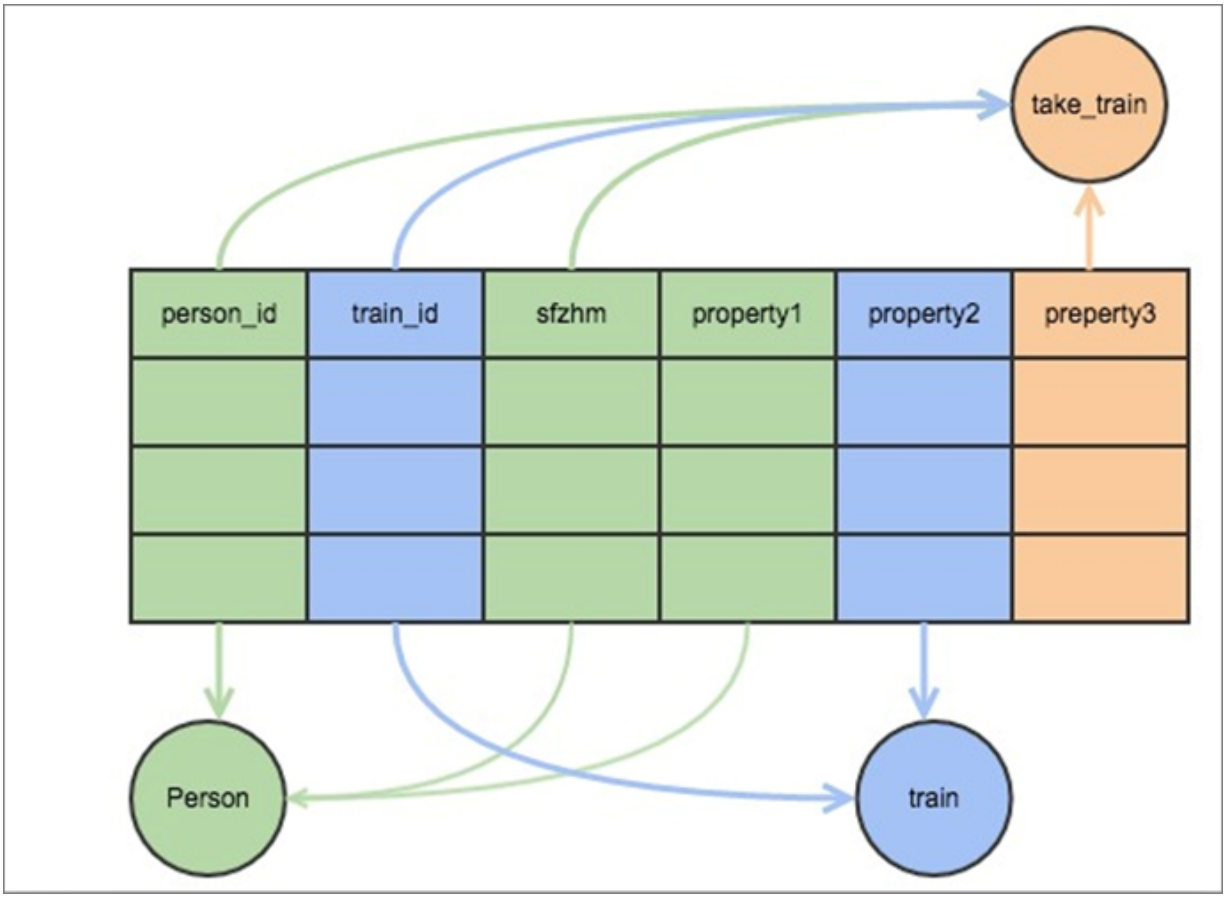
横表导入模式说明

假设源数据存在关系表中，例如MaxCompute。数据按行存储，有固定的列。要将关系数据构建为图数据，需要一个描述文件告知如何做这样的转换。BigGraph ETL目前支持以下几种模式的转换映射：

- 将一张关系表里的每一条记录构建为N个类型的点或边。
- 对表定义`Filter`，将表中满足过滤条件的记录构建为N个类型的点或边。

> **说明：**
> 暂时只支持等值Filter。

图 45-4: 转换映射



如上图所示，每一条记录被构建为一个Person类型点，一个train类型点，和一条take_train类型边。

多个不同的表可以构建相同的点或边类型。例如上图表中构建的Person类型的点，其他表同样也可以构建Person类型的点，它们一起构成了图中Person点数据。

关系表中的一行记录可以构建为多个不同类型的点或边，即1->n映射，但是不支持多行记录构建同一个点或边，即不支持n->1的映射。

竖表导入模式说明

竖表是指表中的多行记录映射成一个点或一条边的数据组织方式。BigGraph支持从竖表导入数据，但要求数据源必须满足一定的格式。

- 竖表之实体表（点表）格式。

  # 有一个列可以作为主键，唯一标识一个点。
  # 每行数据定义一个属性，用一列存储属性名，一列存储属性值。
  # 用一列存储实体类型。

  表格式的示例如下所示。

  表 45-2: 实体表格式示例

ID	PropertyName	PropertyValue	Type
123	Xm	张三	Person
123	Age	14	Person
125	Number	C77	Card

  从上述表格中可以看出，竖表之实体表（点表）中某一类型的点数据是由多行组成的。

- 竖表之关系表（边表）格式。

  关系表用于构建BigGraph中的边类型，表中数据每行定义某种关系的一个属性，多行属性共同构成一个完整的关系。关系表的格式需要满足以下条件：

  # 用一列定义起点主键，这些列的值必须能够唯一标识一个起点实体。BigGraph数据导入时，会通过这些主键的值查询到起点实体，并为其添加关系。
  # 用一列说明起点的类型。
  # 用一列定义终点主键，这些列的值必须能够唯一标识一个终点实体。BigGraph数据导入时，会通过这些主键的值查询到终点实体，并为其添加关系。
  # 用一列说明终点的类型。

用一列说明每行属性的属性名，例如：**“通话时长”**、**“乘坐车厢”**。
用一列说明每行属性的属性值。
用一列说明关系的类型。
如果两点之间相同类型的关系有多个，需要用一个特殊列和起点终点主键、起点终点类型、边类型一起构成一个六元组，唯一标识一条关系/边。

表格式的示例如下所示。

表 45-3: 关系表格式示例

SrcID	SrcType	DstID	DstType	PropertyName	PropertyValue	EdgeType	ShuffleKey
123	Person	125	Train	乘坐车厢	3车	TakeTrain	Null
123	Person	130	Person	通话时长	100	Call	Null
123	Person	130	Person	呼出地点	杭州	Call	Null

在上述表格中，描述了两个关系，可以构建BigGraph的两条边，分别是：

1. 主键为**“123”**的人乘坐了主键为**“125”**的火车，包含一个属性：**“乘坐车厢”**。
2. 主键为**“123”**的人给主键为**“130”**的人打了电话，包含两个属性：**“通话时长”**和**“呼出地点”**。

45.4.3.3 数据导入任务

配置完数据导入后，可通过单击页面上的**执行**按钮，提交数据导入任务，系统能够根据已提交的任务，自动维护任务从提交到上线状态的完整作业生命周期。可以通过观察作业列表显示的作业状态来确定数据有没有成功构建，如果数据已经被成功构建则需要关注上线状态，如果均为**SUCCEEDED**，则表示已被成功上线。上线之后可以使用Gremlin语句进行查询。

需要特别注意的是，作业可能会由于各种原因失败，包括数据本身可能会出现重复的主键、构建机器硬件故障或者网络等问题。如果作业失败，首先可以通过作业链接查看具体日志。

45.4.3.4 图实例数据管理

在线图计算服务平台能够方便查看并管理实例内部的数据，即可以按版本下线数据（如果图实例内部仅存在一个版本，下线后图实例将无数据可用）；如果图实例选择的是增量导入模式，还可以按类型下线数据。

45.4.4 图实例在线查询

提供在线可视化图查询能力，支持在线执行Gremlin查询语言，并将图结果（非点、边图结果暂时无法展现）转换成图，同时能够在图上直观的查看点、边属性。另外，可以锁定图上某点并从该点往外扩展探索，非常有助于帮助业务探索数据。

45.4.5 图计算引擎能力

- 提供交互式查询引擎，分布式执行Gremlin算法。
- 提供分析引擎，使用GAS（类似于Power Graph）接口，实现分析型图算法。适用于算法类任务，例如：最短路径、PageRank等算法。

45.4.6 图实例可视化运维

提供图实例集群管理、图实例热升级、扩缩容、启动、停止等可视化运维操作。

46 算法服务敏捷平台AIMaster

46.1 什么是算法服务敏捷平台

算法服务敏捷平台（以下简称AIMaster）为基于大数据开发的智能应用或产品提供一套一站式的算法/数据研发管理。它是基于业务场景的流程配置，跨多种异构存储/计算平台快速部署和复制的能力的基础平台工具。

算法在智能系统中的使用

算法在具备智能的项目或产品（智能系统）中会大量被使用，处于比较关键的位置，但算法只是智能系统的一部分。算法计算所需要的数据其来源多种多样，算法的计算结果还需要进一步的加工和处理，这些功能包含在智能系统中，但和算法开发的关系不大。

在交通应用中，算法使用的数据来自方方面面：GPS数据、视频数据、微波数据等。算法利用这些数据计算出来的信号灯控制策略需要通过专门的控制通道落实到信号灯上。在搜索引擎中，算法使用的数据主要来自日志系统，算法提供的搜索结果则需要通过前端渲染才能最终展示在您面前。

算法在智能系统中存在的问题

算法是智能系统中不稳定的部分，尤其是机器学习算法，需要不断的优化。一方面算法优化可以不涉及代码的变更，另一方面算法优化则需要更新代码，甚至重新开发部署。算法也会受到数据的影响，当有新的数据可用时，算法通常也需要重新设计。

设计良好的智能系统会把算法集中起来，构成算法模块，和系统其他部分通过接口进行交互，尽量减少算法变更对系统的影响；缺乏设计的系统则有可能到处遍布算法的身影，这使得算法优化非常困难。AIMaster的一个主要目的就是通过工具来强制规范项目或产品中使用算法的方式，不但把系统中使用的算法集中起来构成算法模块，还会规范算法模块的使用方式。

AIMaster的作用

AIMaster除了在系统层面上集中管理算法，降低算法模块和系统其他组件的耦合之外，也对算法模块内部的实现提供了一组规范。

实际的业务系统中由于数据来源多种多样，对数据处理要求的时延和数据量的不同，需要综合多种不同的计算平台，包括离线、在线、专有计算环境等，上一个平台计算完毕后把数据传递给下一个平台进行计算，互相配合完成任务。AIMaster定义并实现了各平台间数据流转的规范，以及如何定义各平台算法执行顺序的方法，从而让算法开发人员能够更加专注于算法开发，不必关注底层的技术细节。

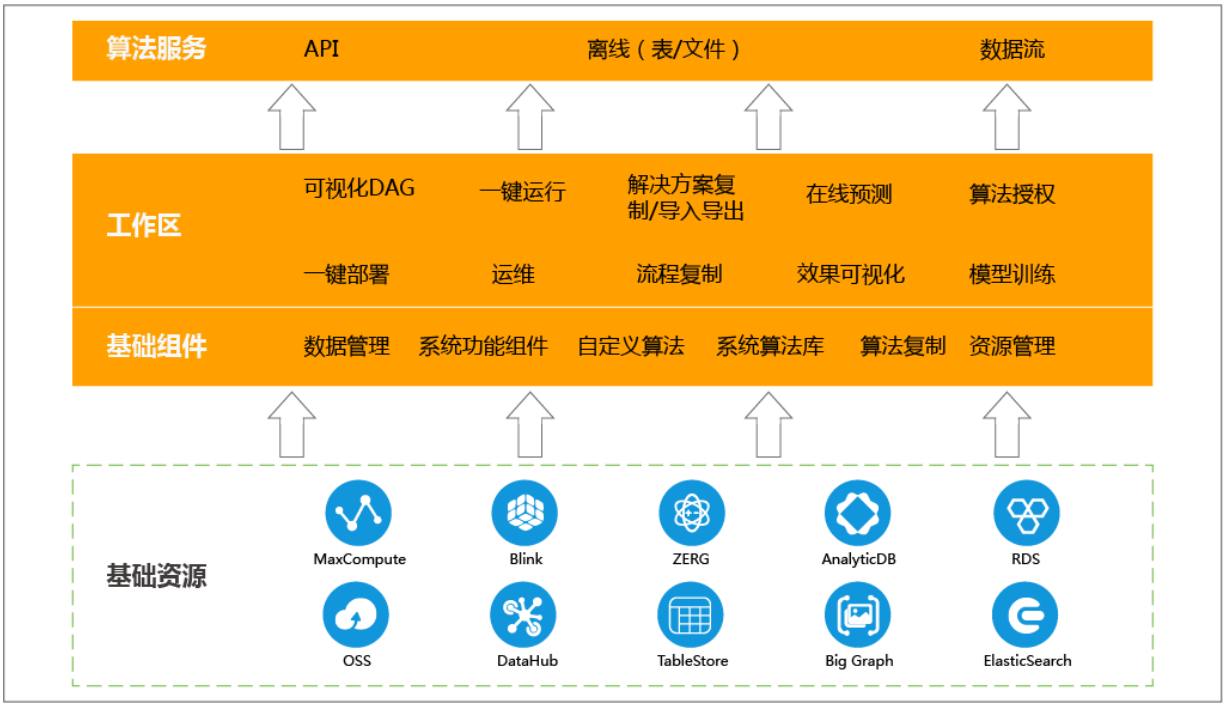
由于AIMaster规范了各个算法接口，也在一定程度上提高了算法复用的能力，降低了多人协作算法开发的实践难度。

AIMaster除了提供算法的规范化和集中托管能力之外， 还提供了基于业务视角，跨多种存储和计算平台的算法流程配置和快速部署上线的能力。 使得算法工程师或者数据开发工程师在构建数据应用的时候从解决实际某一业务场景出发，组合配置多种类型的算法流程， 从而快速实现和沉淀解决该业务问题的工作流。 依托AIMaster的跨平台多异构任务的一键部署和上线能力，极大提高工程效率。

AIMaster是一个主要面向算法开发人员的工具，其目标是优化智能系统的架构，提高系统的工程化水平，让算法开发人员能够集中精力处理算法问题，提高算法开发效率。

46.2 产品架构

46.2.1 功能架构



AIMaster平台提供了以下几大类的模块能力。

- 数据格式（SDF）：提供标准的算法输入输出格式定义，支持MaxCompute、MySQL、DataHub、AnalyticDB等，支持SQL或者JSON。
- 算法管理：提供算法的托管，算法类型支持数据同步、MaxCompute（SQL/MR/SHELL）、Blink SQL、PAI、离线自定义算法（Zerg-Standalone）、在线服务算法（Zerg-Service）。

- 场景开发：跨多种计算平台，组合多种算法的面向业务场景的算法或者数据流开发。
- 数据/算法服务：提供API或者SDK，实现数据或者算法服务的实时调用。
- 跨平台快速部署：支持基于场景的发布，实现多种计算平台的快速上线。
- 解决方案：提供多场景组合的解决方案创建，导入导出场景，同时支持基于解决方案快速生成新场景的快速复制能力。
- 监控运维：提供基于场景的监控运维能力， 并无缝衔接各种目标平台的运维系统。

46.2.2 系统架构

AIMaster整个系统由以下四大模块组成。

- **AIMaster-Web UI**：AIMaster的前端用户交互Web界面，基于React组件化技术开发，提供基于B/S架构的图形界面操作。
- **AIMaster-Biz**：负责与AIMaster-Web UI交互，提供前端页面功能的API，根据产品功能，内部涵盖了以下的6大块内容：

  - 用户管理：负责用户的添加和角色的配置。
  - 工作空间管理：完成工作空间的创建，采用工作空间隔离云计算和场景等资源。
  - 云计算资源管理：完成各种阿里云平台计算和存储资源的注册和托管，比如MaxCompute、Blink和TableStore等。
  - SDF管理：负责算法标准输入输出的创建等。
  - 算法管理：负责各类算法的注册和发布，AIMaster支持丰富的算法类型，比如MaxCompute SQL/MR、Blink SQL和自定义算法，在线服务算法等。
  - 场景管理：负责场景的创建，场景内算法Flow DAG的编辑与配置，以及最终的发布、测试运行和上线。
- **AZK**：AIMaster核心模块之一，主要服务于场景中算法Flow DAG的发布、执行等核心逻辑。主要包括以下四大模块：
  - 权限管控：内置了多租户的隔离功能。在租户内基于RBAC实现了您按照不同角色拥有不同的权限点功能，执行不同登录账号的安全管控。
  - 算法管控：实现算法的参数、输入输出和代码的集中管理。
  - 算法流程管控：完成算法Flow DAG的定义保存及部署。
  - 流程执行管控：完成算法Flow DAG在各种异构平台的依赖执行，有序保障结果产出的正确性。
- **DTB-Zerg**：AIMaster核心模块之一，主要实现的算法的服务化和微服务能力，提供面向不同需求的Zerg-StandAlone和Zerg-Service两大类服务。主要包括以下六大模块：
  - 集群资源管控：提供不同用户、不同服务根据自身特点，配置管理不同的服务器集群，满足服务隔离和资源的需求。
  - Zerg-Standalone：提供自定义、单机、异步、不定时的算法执行场景。
  - Zerg-Service：提供在线算法的服务化输出，将普通的通过Java编写的程序实现服务化的部署，并提供SDK在线调用。
  - 服务路由：Zerg-Serivce发布后，根据服务的部署拓扑结构和负载情况，动态路由服务调用到不同机器，实现负载均衡。
  - 服务治理：支持服务的扩容和缩容，并且对终端用户零感知，实现透明化的服务治理能力。
  - 服务的监控：提供集群机器及其之上的服务监控能力，包括服务的资源使用情况、服务的调用量和响应时间等。

46.2.3 安全架构

AIMaster的安全架构，是由平台自身的安全实现层、平台内置的安全服务层构成。

- 平台自身的安全实现层：保障平台在代码实现和部署配置时的产品自身安全性。
- 平台内置的安全服务层：为租户和其用户提供平台基础性的安全服务能力，如租户资源隔离、身份认证、权限鉴别和日志合规审计等。

46.2.4 多租户模型

数据资源平台拥有自己的多租户权限模型。

- 弹性的存储和计算资源：租户可按需配置自己的工作区、场景、云计算资源。
- 租户独立管理自有的数据、权限、用户、角色，彼此隔离，以确保数据安全。

46.3 工作原理

46.3.1 多种大数据存储和计算平台支持

AIMaster作为数据、算法和服务支撑平台，为了支撑不同的业务需求、功能实现，实现了与阿里云主流的大数据存储及计算平台的对接。

- **MaxCompute**

  作为阿里云大数据离线处理的核心能力，大量的ETL和算法服务依托于MaxCompute运行。AIMaster支持您创建运行在MaxCompute上的SQL及MR任务算法，同时通过可视化的流程配置，将MaxCompute任务串联，作为工作流发布到DataWorks上运行。

- **Blink**

  作为阿里云实时流式数据处理的核心能力，大量的实时ETL和算法依托于Blink运行。 AIMaster支持您创建运行在Blink上的Flink SQL任务算法，同时通过可视化的流程配置，将实时流式任务发布到Blink平台。

- **RDS/AnalyticDB**

  这两个关系型数据库，面向您的数据存储、数据分析场景。RDS面向千万级别以下数据量，AnalyticDB面向千万级别以上数据量。 AIMaster支持您创建面向两者的SQL算法节点。

- **Zerg-Standalone**

  很多算法或者个性化服务，依托于阿里云大数据产品之外的第三方开源平台运行， 比如TensorFlow、自研算法。AIMaster支持您创建Zerg-Standalone类算法，将自研或者第三方算法

包（jar包或者python脚本）托管到AIMaster平台，通过动态生成API的方式，提供用户方便的托管和运行算法的能力。

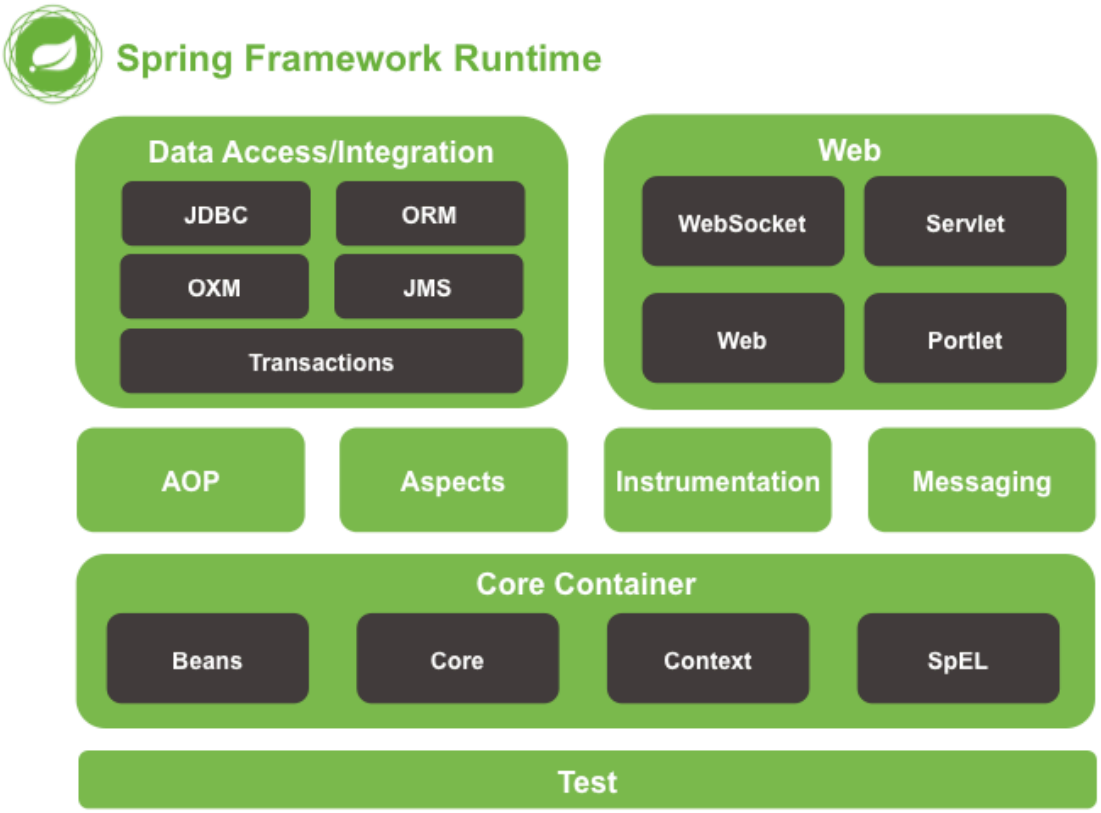
- **Zerg-Service**

  在很多场景中有在线服务类算法的需求，比如在线预测。通过AIMaster，算法工程师能够把普通jar包托管到AIMaster， AIMaster会根据服务的能力要求，动态将其发布为在线服务，提供动态扩缩容、高可用、负载均衡能力，并快速将算法服务化。

46.3.2 Spring Boot架构

Spring作为J2EE技术的事实标准，被大量应用于企业级应用开发场景。特别是针对如企业ERP、电子商务、社交网络等大型高并发高可用场景，它提供了开发的标准实现。使得开发和部署N层架构、面向Web和服务的企业级应用具有高可扩展性、高可靠性和高灵活性。

Spring Framework的架构设计如下图所示。



而作为最新的基于Spring框架的Spring Boot项目，简化了基于Spring的应用开发，只需要run就能创建一个独立的、产品级别的Spring应用，它为Spring平台及第三方库提供开箱即用的设置。使用

Spring Boot创建Java应用，并使用java -jar启动它或采用传统的war部署方式。并且提供一系列大型项目常用的非功能性特征，比如：内嵌服务器，安全，指标，健康检测，外部化配置。

46.3.3 微服务架构托管平台

微服务是一种架构风格，一个大型复杂软件应用由一个或多个微服务组成。系统中的各个微服务可被独立部署，各个微服务之间是松耦合的。每个微服务仅关注于完成一件任务并很好地完成该任务。在所有情况下，每个任务代表着一个小的业务能力。

AIMaster的微服务能力表现在以下四个方面。

- 将用户普通jar文件快速发布成微服务，使得普通用户在不懂微服务架构的情况下也能得到微服务架构带来的红利。
- 算法微服务化后动态的扩缩容和负载均衡能力。
- 客户端调度，服务的动态发现。
- 服务调用的性能监控和容错。

同时在微服务的基础之上， AIMaster提供了服务之间基于OSGi技术的服务和状态隔离能力，解决了服务间的jar包冲突、服务间错误影响等多服务交叉部署时候遇到的问题。

AIMaster作为数据、算法和服务支撑平台，提供了微服务托管能力。针对不同场景对微服务对外服务形式的要求，AIMaster提供多协议服务化能力。

- REST：作为目前基于HTTP协议之上的面向资源的软件架构风格，AIMaster支持将您的算法发布为RESTful服务。您只需按照JAX-RS规范，在算法开始的时候，标记JAX-RS的Annotation，AIMaster及可以将其算法包发布为RESTful服务的算法供外部调用。
- Spring Remoting：最为高效的基于HTTP、Hessian和Spring的RPC通信协议，AIMaster支持将您的算法发布为Spring Remoting服务，并且提供给您SDK方便基于Java Interface来调用服务，更符合服务嵌入您应用的场景。
- HSF：作为阿里巴巴分布式服务治理和服务框架，HSF正如其名具备高性能的特点，并且提供服务发现和动态路由等微服务基础能力。在面向高可用、高并发和高性能的场景下，AIMaster能将您普通的算法jar包发布为HSF服务，同时提供SDK方便地基于Java Interface来调用服务，您完全不感知HSF，减少了学习和上线成本。

46.4 功能特性

46.4.1 数据格式（SDF）

标准数据格式（SDF）是一种json形式的数据格式抽象，它约定了算法的输入和输出贵方。

标准数据格式（SDF） 包含两部分：基于各种存储资源的建表语句、每个SDF包含的字段信息和字段约束。目前支持的存储资源类型有五种，包括Analytic DB、MaxCompute、ApsaraDB for RDS、Table Store、DataHub。每一种SDF均可分别定义在五种存储资源类型下的建表语句。

标准数据格式（SDF）的可见性属于租户下的定义，不同租户进行隔离， 算法的输入输出通过SDF来约束。在场景配置算法流程时，上下游节点是否能够建立关系是通过判断对应的SDF是否匹配来决定。

46.4.2 算法管理

算法包含了基本信息、算法资源、输入、输出、参数五个要素，在新增算法时进行定义。

- **基本信息**：包含算法code、算法名称、算法描述、算法分类。
- **算法资源**：包含服务类型、计算资源类型、启动脚本、算法文件等。

  # 服务类型包括非在线服务、在线服务、流计算。

  # 计算资源类型包括MaxCompute、Zerg-Service、Zerg-Standalone、Blink。

  # 算法文件包含算法逻辑文件以及算法依赖文件。
- **算法输入**：包含输入code、数据格式、数据源类型、是否必选、支持多个等。

  # 数据格式在数据格式中定义，此处可选择已有的数据格式，或者新增。

  # 数据源类型受所选择的服务类型、计算资源类型约束。

    如用户在算法资源部分选择了服务类型为非在线服务、计算资源类型为MaxCompute,则数据源类型只能选择MaxCompute。（此处非平台功能限制，是数据源类型与计算资源类型存在匹配关系。）

  # 是否必选是指该输入是否是算法的必选输入，默认为必选。

  # 支持多个是指算法输入是否支持多个数据源文件，由算法实现拼接。
- **算法输出**：包含输入code、数据格式、数据源类型。

  数据格式和数据源类型，与算法输入一样，受限于数据源类型受所选择的服务类型、计算资源类型约束。数据源类型同时受限于数据格式中的DDL信息。

算法定义保存后，首先是保存在测试环境，用户在测试环境对算法进行修改和调优。调优完成后，点击发布，指定发布版本号，比如V1.0.0，填写版本备注，将算法发布到线上环境。发布环境

的算法版本，能在流程中被引用。版本可以被废弃，废弃后的版本只有算法创建者可见，非创建者不可见。算法版本被创建后，不能在新增的流程中被引用，但已经引用过的流程中依然可用。

46.4.3 场景开发

场景定义数据输出的方式，将一个应用的算法解决方案定义为一个场景。例如针对特殊车辆交通灯控制的应用，开发一套数据、算法为应用服务，即可定义为一个场景。场景中包含了场景的输入、输出（包含输入输出的存储资源、数据格式）、算法名称、算法版本、计算资源、算法之间依赖关系、调度周期等等。

每个场景可以编辑场景的流程，场景与流程一一对应，每个流程是一个工作画布。画布的左侧是已经定义好的算法和系统节点，您通过拖、拉的方式自定义算法流程、输入输出。对于依赖的计算资源相同的算法节点，可以定义算法节点组进行组合，被组合的算法在部署、运行时被打包在一起进行调度。

支持对每个算法节点单独设置基础信息、资源、调度周期信息等。

场景是定义在工作区下，同一个场景区分测试和线上环境，测试线上环境分别对应一个流程，为了支持回滚的功能，流程需要区分不同的版本。

测试环境定义好的算法流程后，您进入部署环节。单击DAG图右上方的**部署**，此时系统会检查每个算法节点的定义的计算资源、输入输出数据实例是否完备，输入、输出节点的数据实例定义是否完备。在部署的过程中，提供部署日志查看功能。支持对流程同步编辑、保存。对部署失败的流程，提供具体的日志页面看到失败节点的部署日志详情。

部署成功后，进入到运行环节。单击DAG图右上方的**运行**，进入运行状态。运行阶段提供运行日志，每一次运行均有单独的运行日志，对运行结果进行查看和确认。测试环境中支持对场景流程进行调整、验证。

测试场景上线为线上场景，并完成线上环境的部署。场景被发布到线上环境后，在线上环境中能查看到场景信息。查看到场景流程DAG图、每个节点的算法信息、计算资源、存储资源、调度周期、输入输出等的配置。上线成功后，进入到运行环节。单击DAG图右上方的**运行**，进入运行阶段。运行阶段提供运行日志，每一次运行均有单独的运行日志，对运行结果进行查看和确认。

46.4.4 数据/算法服务

场景中的数据或者算法流程上线发布之后，AIMaster会独立生成可调用RESTful API。您可以按照自己的调度周期调用API完成场景工作流的执行，也可以将API提供给应用开发，完成与业务系统的快速对接，从而实现数据或者算法的服务化。

46.4.5 跨平台快速部署

在AIMaster中创建的一个场景，可以包含来自多种计算平台的算法节点，目前主要支持以下几大类：

- 离线：MaxCompoute SQL/MR/SHELL +自定义算法
- PAI
- 在线服务化算法：支持Java或者Python编写
- Blink SQL

在不使用AIMaster的情况下， 您要搭建一个完整跨多种计算平台的工作流，需要对接不同的目标平台，这会导致流程不能统一管控，操作繁琐，同时没有一个集中的地方展示整个数据或者算法处理流程，无法直观展示自己的方案。

但是依托AIMaster，可以在场景中拖拉来自不同平台的算法构成完成的流程链路。不仅如此，在您编辑和配置完整个流程后，AIMaster提供一键部署工作流到异构计算平台的能力，非常便捷。

46.4.6 解决方案

解决方案的核心思想是帮助沉淀成熟的场景案例，同时在多地复制输出的时候提供快速的交付能力和完全相同的交付质量。

其使用方式是将在某地成功实施的一到多个场景（算法流程）保存为解决方案，之后可将解决方案导出并在新环境重新导入，从而实现解决方案的在新环境（本环境不同工作区或者全新的物理环境）的迁移。在新环境，可以将解决方案中的场景重新实例化并重新部署上线。通过这种方式，类似比如城市大脑、工业大脑等行业解决方案可以快速在新的城市或者行业落地。交付能力极大提高的同时，交付质量同样能够得到保障。

46.4.7 平台管理

平台管理主要从系统层面，为管理者对参与AIMaster平台使用的用户进行对应管控，项目空间作为代码管理、成员管理、角色和权限分配的基本单元，每个团队都可具有独立的项目空间。用户加入项目空间并被分配相关权限之后，才能够查看或编辑代码。一个用户可以同时加入多个项目空间，在不同的项目空间中被授予不同的角色。

46.4.7.1 组织管理

显示组织详情信息以及组织Owner账号、AK信息，并可对组织对应人员进行成员管控。

46.4.7.2 工作空间管理

AIMaster租户管理员可对您的工作空间进行列表展现，并提供创建、配置、激活项目空间的对应管理功能，方便AIMaster租户管理员进行工作空间整体管控。

46.4.7.3 工作空间成员管理

项目成员列表，以列表的形式显示本工作空间的成员名称、登录名称、成员角色等信息。支持模糊搜索工作区空间成员并可将用户移出本工作区空间的操作。您加入工作空间，只针对项目管理员才可新增工作区成员，支持模糊匹配查找的方式用户添加至本项目空间。在添加用户到项目空间时必须指定至少一种角色。您退出工作区空间，支持项目管理员主动清退用户。用户移出本工作区后失去之前在本工作区分配的所有权限。

46.4.7.4 权限管理

权限管理主要完成平台用户、角色、权限等管理由统一的管理提供，对应权限特征如下表所示。

数据资源平台角色	平台权限特征
管理员	管理员除了能够创建管理SDF、云计算资源、算法、场景工作流、解决方案和场景部署上线之外，还具有管理工作区成员和成员权限功能。
开发人员	开发人员能够创建管理SDF、云计算资源、算法、场景工作流、解决方案和场景部署上线。
普通用户	运维角色的用户由项目管理员分配运维权限；拥有发布及线上运维的操作权限，没有数据开发的操作权限。

46.5 产品价值

46.5.1 B/S架构Web化的软件服务

可在互联网/内部网络环境下直接使用，无需安装部署，拎包入驻，开箱即用。

46.5.2 丰富的算法类型支持

AIMaster的核心功能之一是帮忙算法的标准化和沉淀，因此支持非常丰富的算法类型。

- 离线算法

  # Maxcompute：SQL/MR/SHELL

  # PAI

自定义离线算法：Zerg-Standalone

- 在线算法

  # Zerg-Service-Java：自定义Java编写的服务化算法

  # Zerg-Service-Python：自定义Python编写的服务化算法

- 流式算法

  # Blink SQL

46.5.3 跨多种异构计算平台的算法流配置

- 支持按照业务视角来配置整个数据或者算法流程。
- 流程中涉及的算法可以跨多种计算平台，比如可以有MaxCompute的算法节点，也可以有Blink SQL节点，同时也可以有自定义算法。
- 多异构平台算法流的一键部署，减轻多平台对接的繁琐操作。

46.5.4 可控的在线服务算法的扩展能力

针对在线算法，可根据算法对外提供的负载能力要求，支持自由的扩缩容能力。同时可以按照服务隔离要求，将服务部署到不同的目标集群，服务不同的客户群体。

46.5.5 快速的场景复制满足多地复用的需求

依托AIMaster中的**解决方案**模块， 可以将一到多个场景打包为一个解决方案。解决方案可以被整体打包输出。在新的环境重新导入，进而将解决方案中包含的场景在新环境快速完成克隆，克隆的内容包块算法及场景工作流，再依托场景一键部署快速上线的能力，实现快速的场景复制，满足多地复用的需求，极大提高交付能力。

46.5.6 多租户权限模型

多租户模型确保用户数据被安全隔离，以租户为单位进行统一的权限管控、数据管理、调度资源管理和成员管理工作。

46.5.7 开放的平台

所有模块已实现组件化、服务化，您可基于AIMaster的Open API来定制开发扩展功能。

46.6 性能指标

算法敏捷服务平台AIMaster的主要性能指标依赖于算法运行的目标云计算资源，因为目标云计算资源是由用户提供，因此计算能力可直接由用户决定。

- 离线算法：主要依赖用户的MaxCompute Project的Quota计算能力。
- 在线服务算法：依赖用户提供的Agent机器性能指标。
- 流式算法：主要依赖用户的Blink Project的Quota计算能力。