

自由构建 探索无限

亚马逊科技 DEV DAY

Apache ShardingSphere 分布式数据库生态的云原生实践

端正强

SphereEx 高级中间件工程师

Apache ShardingSphere PMC

CONTENTS

- 1. ShardingSphere 简介
- 2. 数据库行业发展趋势
- 3. ShardingSphere 核心功能与架构
- 4. ShardingSphere 云原生实践

ShardingSphere 简介

ShardingSphere 简介

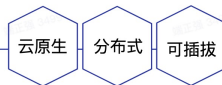
Apache ShardingSphere 是一款**分布式数据库增强计算平台**，可以将任意数据库转换为分布式数据库系统，并提供分布式数据库、弹性扩缩容、数据安全脱敏等解决方案。

ShardingSphere 遵循 Database Plus 理念，旨在构建异构数据库上层的服务标准和生态。



ShardingSphere

前沿多领域技术融合



强大的产品功能及特性

| | | |
|-------|--------|----------|
| 联合查询 | 分布式事务 | 数据分片 |
| 读写分离 | 分布式治理 | 弹性伸缩 |
| 数据加密 | 影子库压测 | Dist SQL |
| 可插拔架构 | | |

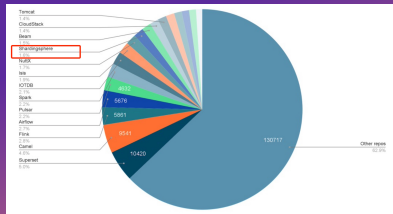
ShardingSphere 社区



全球最顶级的开源软件基金会
管理超过两亿行代码
成功孵化 300+ 顶级开源项目



Projects by Number
of Commits, 2021



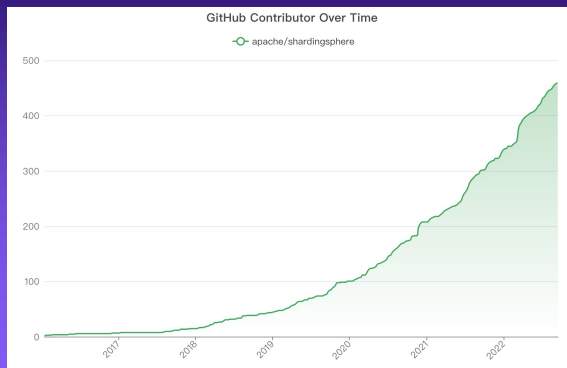
- 17000+ Stars
- 5000+ Forks
- 400+ Contributors
- 10000+ Pull Requests

基金会认可 Apache 软件基金会顶级项目

社区活跃度 2021 年度 Apache 基金会年度报告代码提交量位列前十

学术界认可 数据库顶会 ICDE 2022 发表论文《A Holistic and Pluggable Platform for Data Sharding》

ShardingSphere 社区



- 社区优于代码
- 尊重和信任合作伙伴
- 保持友好和开放

August 15, 2022 – September 15, 2022

Period: 1 month

Overview

562 Active pull requests

269 Active issues

548

Merged pull requests

14

Open pull requests

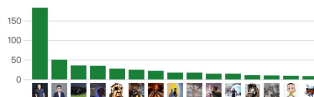
159

Closed issues

110

New Issues

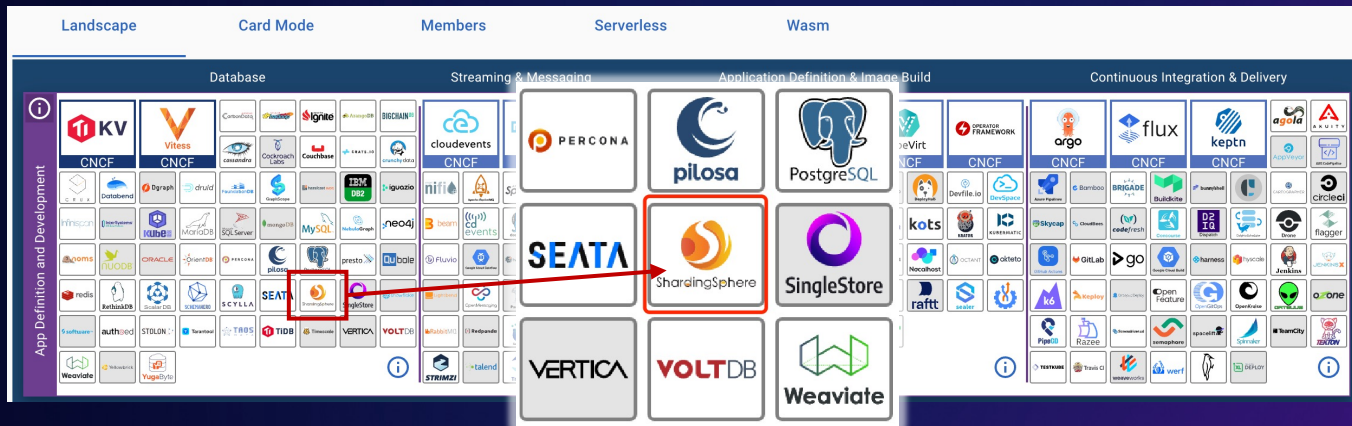
Excluding merges, **49 authors** have pushed **543 commits** to master and **557 commits** to all branches. On master, **2,963 files** have changed and there have been **51,838 additions** and **26,318 deletions**.



1 Release published by 1 person

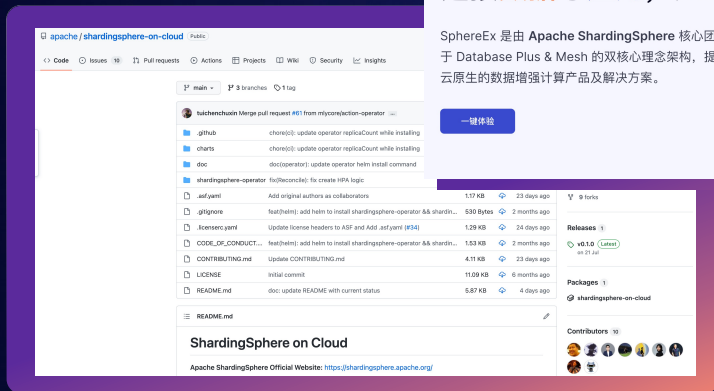
- 测试自动化
- 创建自动渠道
- 公开和远程的工作方式

CNCF：云原生计算基金会



CNCF：Linux 基金会旗下基金会。旨在为云原生应用者提供一个全景图，帮助企业和开发人员快速了解云原生体系的全貌。
ShardingSphere 2019 进入 CNCF Landscape，未来会在开源领域及云原生领域会持续拓展，不断精进，构建有机开源生态圈。

SphereEx 开源商业化公司



SphereEx 产品 文档 社区 关于我们 中文

连接数据与应用, 如此简单

SphereEx 是由 Apache ShardingSphere 核心团队创立, 基于 Database Plus & Mesh 的双核心理念架构, 提供企业级、云原生的数据增强计算产品及解决方案。

一键体验

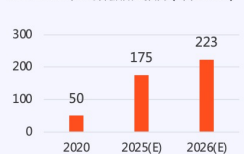
- SphereEx——专注于分布式数据库解决方案的开源商业化公司
- 持续引导 ShardingSphere 及 ShardingSphere on cloud 云原生项目发展

数据库行业发展趋势

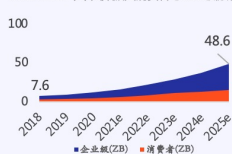
数据库碎片化

企业级数据成为数据增量核心引擎¹

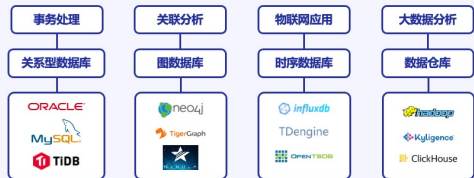
2018-2026年全球数据量预测 (单位: ZB)¹



2018-2025年中国数据圈消费者与企业级份额¹



场景分化&数据量提升&技术革新带来底层数据进一步多元化演进



数据来源: IDC, 艾瑞研究院

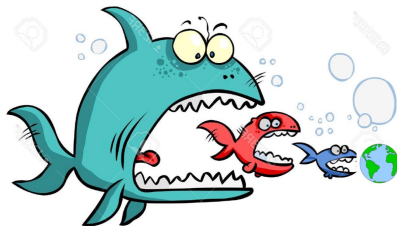
底层数据技术栈及 DBA 逐步形成多样化 & 复杂化生态

典型大中型企业 (2010~202X)



数据库上云

- 软件吞噬世界
- 开源吞噬软件
- 云吞噬开源



Apache Flink



HIVE



PostgreSQL



MySQL™



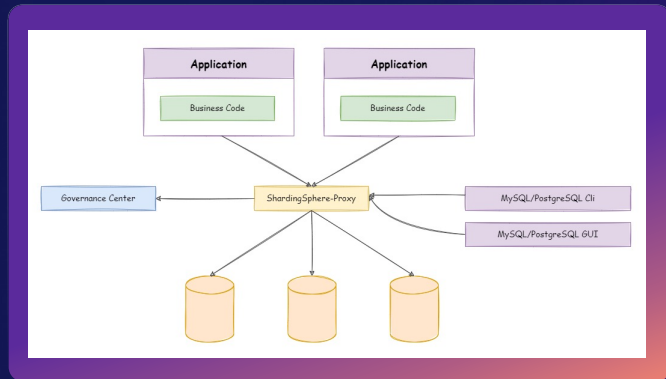
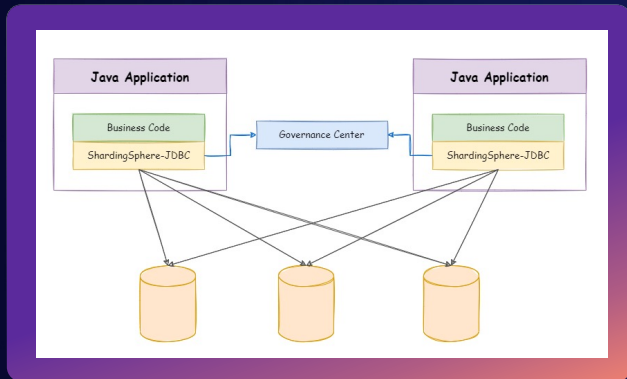
COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"

THE WORLD'S LARGEST OPEN SOURCE FOUNDATION

- All volunteer community
- 271M+ lines of code in stewardship
- 4.8B+ lines of code changed
- 4.6M+ code commits
- 850+ individual ASF Members
- 8,200+ Apache Committers
- 49,000+ code contributors
- 640,000+ people involved in our communities
- 350+ Projects and Initiatives
- 300+ Top-Level Projects
- 37 podlings in the Apache Incubator
- ~2 Petabytes source code/downloads from Apache mirrors
- 28M+ emails across 1,400+ mailing lists
- Web requests received from every internet-connected country on the planet
- 286M+ weekly page views across apache.org

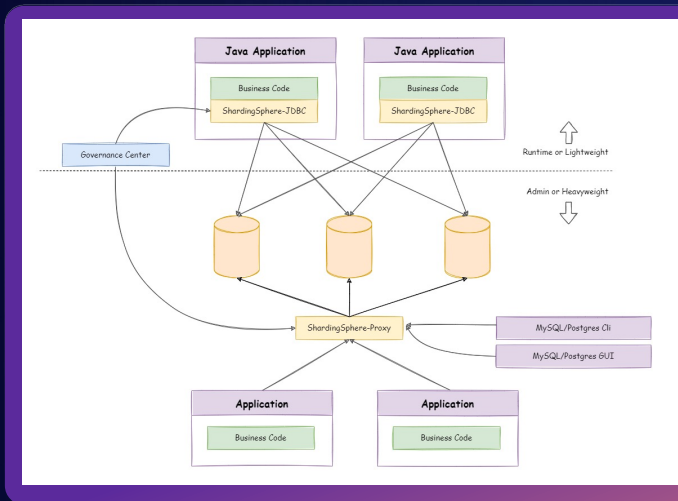
ShardingSphere 核心功能与架构

接入端



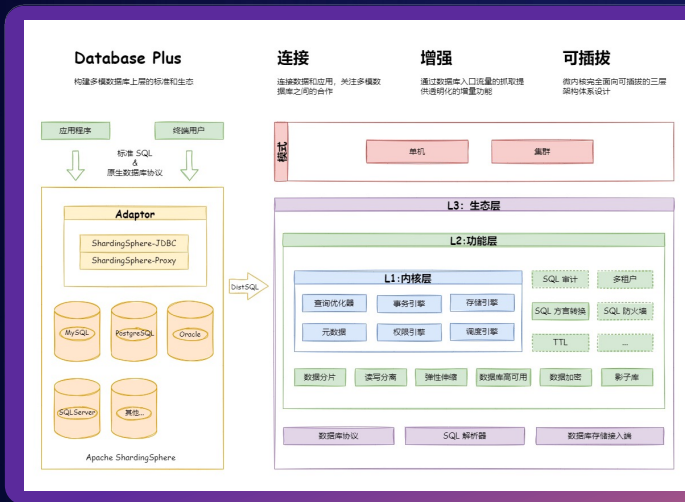
- ShardingSphere 由 JDBC、Proxy 和 Sidecar（规划中）3 个接入端组成，既支持独立部署，又可以混合部署；
- JDBC、Proxy 和 Sidecar 3 个接入端均提供标准化的增量功能，可适用于 Java 同构、异构语言以及云原生等各种应用场景；

接入端混合部署



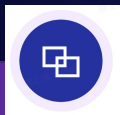
- ShardingSphere-JDBC 采用无中心化架构，与应用程序共享资源，适用于 Java 开发的高性能的轻量级 OLTP 应用；
- ShardingSphere-Proxy 提供静态入口以及异构语言的支持，独立于应用程序部署，适用于 OLAP 应用以及对分片数据库进行管理和运维的场景。

核心功能



- L1 内核层：**面向数据库内核**，包括数据库事务引擎，查询优化器等；
- L2 功能层：**ShardingSphere 最核心所在**，**可定制化开发平台**。具有高定制化、高度内聚、灵活扩展等特点；
- L3 生态层：通过三个接口分别实现数据库协议、SQL 方言和数据库存储对接，用于**打造异构数据网关**；

核心功能



连接

连接是 ShardingSphere 的基础能力，可以有效**简化数据和应用之间的连接**。连接的设计要点在于**强大的数据库的兼容性**，在应用和数据之间搭建了一层与具体数据库实现无关的桥梁，为增量能力提供了基础。



增强

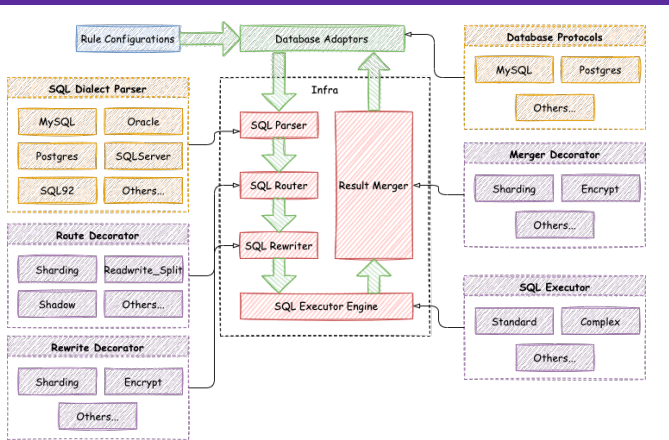
增强是 ShardingSphere 的主要能力，在拦截访问数据库流量的前提下，透明化的提供增量功能。增强包含了**流量的重定向**（数据分片、读写分离、影子库）、**流量变形**（数据加密）、**流量鉴权**（SQL 审计、权限）、**流量治理**（熔断、限流）以及**流量分析**（可观察性、服务质量分析）等。



可插拔

可插拔是 ShardingSphere 的设计理念，架构内核是**完全面向顶层接口设计的**，**内核模块完全不感知具体功能的存在**。它为分库分表、读写分离等每一个功能插件赋予单独部署和协同配合的能力。

可插拔架构



- ShardingSphere 可插拔架构提供了数十个基于 SPI 的扩展点，开发者可以十分方便的对功能进行定制化扩展；
- 按照扩展点是基于技术还是基于功能实现，可以将扩展点划分为功能扩展点和技术扩展点。
- 基于扩展点，ShardingSphere 默认实现了数据分片、读写分离、数据加密、影子库压测、高可用等功能；

ShardingSphere 云原生实践

ShardingSphere 虚拟机部署

《Creating a Secure Distributed Database Cluster Leveraging Your Existing Database Management System》文章中介绍了如何在 AWS 云上，使用 ShardingSphere-Proxy 和 Aurora PostgreSQL-Compatible Edition 联合打造分布式数据库，主要步骤如下：

1. 为 ShardingSphere-Proxy 创建 EC2

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Red Hat Enterprise Linux 8.3 (HVM), SSD Volume Type - ami-0c04ac6974e10f832 (64-bit x86) / ami-06533535d7ffb2572 (64-bit Arm)

Red Hat Enterprise Linux version 8.3 (HVM), EBS General Purpose (SSD) Volume Type

Red Hat

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select

64-bit (x86)
 64-bit (Arm)

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance families Current generation Show/Hide Columns

Currently selected: t2.medium (- ECUs, 2 vCPUs, 2.3 GHz, -, 4 GiB memory, EBS only)

| | Family | Type | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance | IPv6 Support |
|-------------------------------------|--------|---------------------------|-------|--------------|-----------------------|-------------------------|---------------------|--------------|
| <input type="checkbox"/> | t2 | t2.micro | 1 | 1 | EBS only | - | Low to Moderate | Yes |
| <input type="checkbox"/> | t2 | t2.small | 1 | 2 | EBS only | - | Low to Moderate | Yes |
| <input checked="" type="checkbox"/> | t2 | t2.medium | 2 | 4 | EBS only | - | Low to Moderate | Yes |

ShardingSphere 虚拟机部署

2. 创建 Aurora 数据库

RDS > Create database

Create database

Engine options

Engine type [Info](#)

Amazon Aurora



MySQL



PostgreSQL



Oracle

ORACLE



Edition

Amazon Aurora MySQL-Compatible Edition

Amazon Aurora PostgreSQL-Compatible Edition

Capacity type [Info](#)

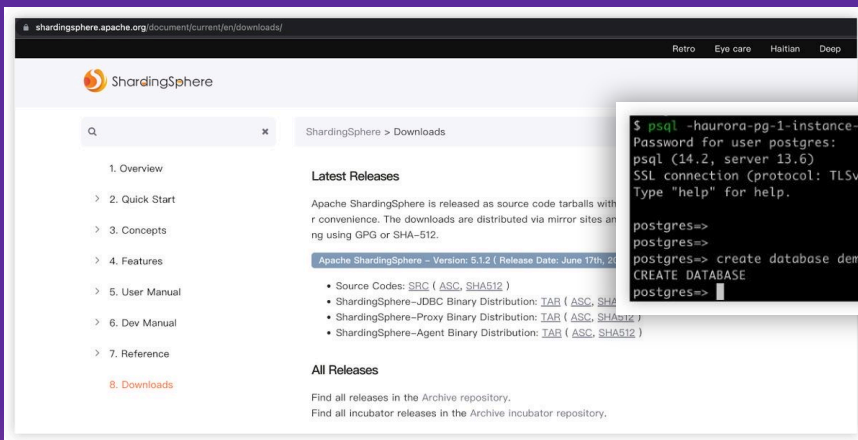
Provisioned

You provision and manage the server instance sizes.

| DB identifier | Role | Engine | Region & AZ | Size |
|--|------------------|-------------------|-------------|--------------|
| <input type="radio"/> aurora-pg-1 | Regional cluster | Aurora PostgreSQL | cn-north-1 | 1 instance |
| <input type="radio"/> aurora-pg-1-instance-1 | Writer instance | Aurora PostgreSQL | cn-north-1d | db.t3.medium |
| <input type="radio"/> aurora-pg-2 | Regional cluster | Aurora PostgreSQL | cn-north-1 | 1 instance |
| <input type="radio"/> aurora-pg-2-instance-1 | Writer instance | Aurora PostgreSQL | cn-north-1d | db.t3.medium |

ShardingSphere 虚拟机部署

3. 部署 ShardingSphere-Proxy & 执行数据库操作

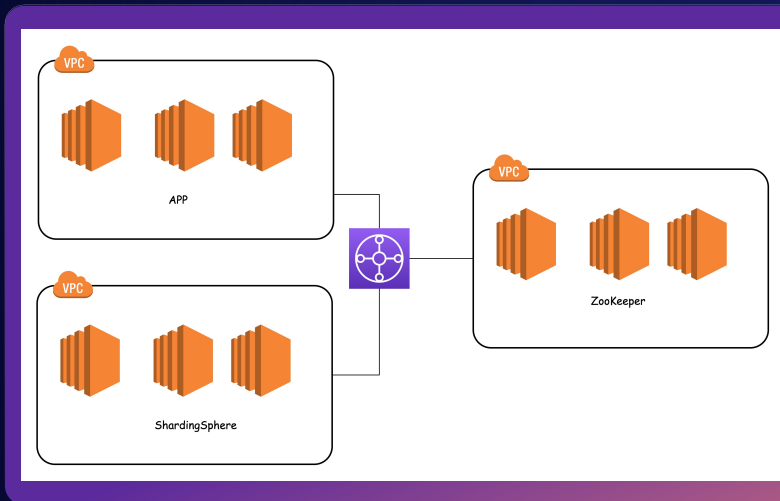


The screenshot shows the Apache ShardingSphere website's Downloads page. The page title is "ShardingSphere > Downloads". On the left, there is a navigation menu with items: 1. Overview, 2. Quick Start, 3. Concepts, 4. Features, 5. User Manual, 6. Dev Manual, 7. Reference, and 8. Downloads (highlighted). The main content area is titled "Latest Releases" and contains the following text: "Apache ShardingSphere is released as source code tarballs with r convenience. The downloads are distributed via mirror sites an ng using GPG or SHA-512." Below this, there is a blue bar indicating the current version: "Apache ShardingSphere - Version: 5.1.2 (Release Date: June 17th, 20...". Underneath, there is a list of download links: "Source Codes: [SRC](#) ([ASC](#), [SHA512](#))", "ShardingSphere-JDBC Binary Distribution: [TAR](#) ([ASC](#), [SHA512](#))", "ShardingSphere-Proxy Binary Distribution: [TAR](#) ([ASC](#), [SHA512](#))", and "ShardingSphere-Agent Binary Distribution: [TAR](#) ([ASC](#), [SHA512](#))". At the bottom, there is a section titled "All Releases" with instructions to find releases in the Archive repository and the Archive incubator repository.

```
$ psql -haurora-pg-1-instance-1.cm1ktganzxak.rds.cn-north-1.amazonaws.com.cn -Upostgres -dpostgres
Password for user postgres:
psql (14.2, server 13.6)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

postgres=>
postgres=>
postgres=> create database demo_ds_0;
CREATE DATABASE
postgres=>
```

ShardingSphere 虚拟机部署



在传统的部署模式下，治理节点、计算节点和应用单独部署在各自的 EC2 上。

对于 Apache ShardingSphere 来说，需要一些 EC2 机器安装部署所有组件，并且运维的同事需要管理和维护每一个安装组件的机器及其上运行的应用。

ShardingSphere 虚拟机部署

传统虚拟机部署的问题

- 水平扩展能力不足
- 宿主机资源利用率不高
- 服务状态管理复杂



云上部署的优势

- 优秀的弹性伸缩能力
- 提高服务器资源利用率
- 自动化服务状态管理

ShardingSphere 云原生解决方案

Apache ShardingSphere 5.1.2 发布 | 全新驱动 API + ...

MONDAY, JUNE 20, 2022

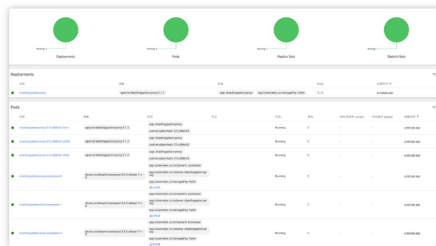


此次更新带来 ShardingSphere 原生驱动、Helm Chart、SQL 翻译等全新功能，打造极具生产力的数据网关

- 使用 Helm 部署 ShardingSphere-Proxy

ShardingSphere-Proxy 提供了 Docker 镜像以便于用户容器化部署。不过，对于需要在 Kubernetes 部署 ShardingSphere-Proxy 的用户，还需要自行处理数据库驱动挂载、配置挂载、自定义算法挂载等必要步骤，部署过程相对繁琐，运维成本相对较高。

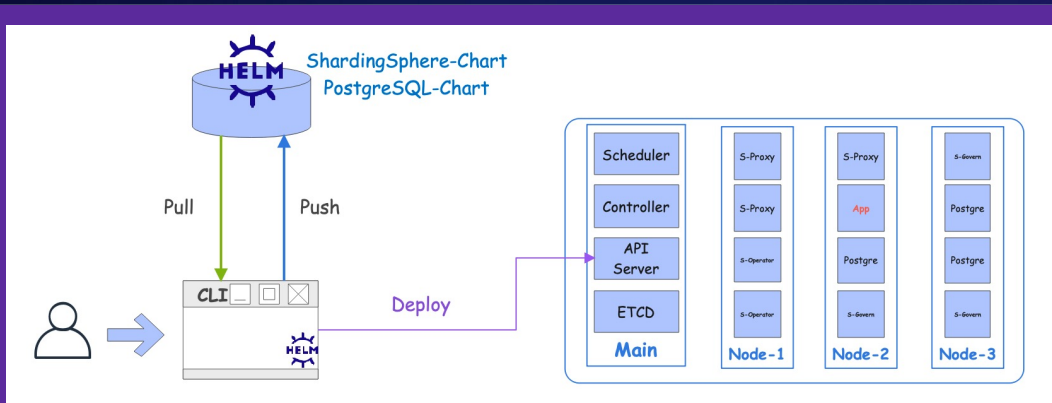
ShardingSphere 本次更新带来了全新的 ShardingSphere-Proxy Helm Chart。这项新功能由企业级、云原生数据增强计算产品及解决方案提供商 **SphereEx** 向 **Apache ShardingSphere** 社区捐赠，推动 Apache ShardingSphere 在云原生方向前进。



ShardingSphere 在集群模式下依赖注册中心存储元数据，ShardingSphere-Proxy 的 Helm Chart 能够自动部署 ZooKeeper 集群，帮助用户快速搭建 ShardingSphere-Proxy 集群。

受限于开源协议，ShardingSphere-Proxy 的二进制发布包、Docker 镜像受限于开源协议，无法打包 MySQL JDBC 驱动，用户需要手动添加 MySQL JDBC 驱动到 classpath 才能使用 MySQL 作为 ShardingSphere 的存储节点。对于这类情况，ShardingSphere-Proxy Helm Chart 能够在 Pod 的 Init 容器自动获取 MySQL JDBC 驱动，降低了用户的部署操作成本。

ShardingSphere Helm 云上部署

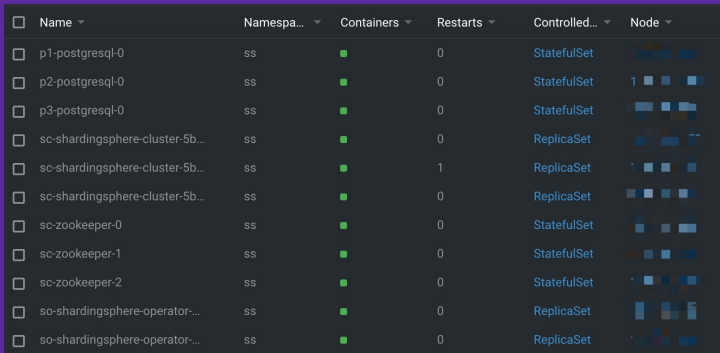


云上部署模式下，治理节点、计算节点和应用共享一个 Kubernetes 资源池。

依托于 Kubernetes 自身特性，能够做到对基础设施层面的资源进行池化，使集群中的应用共享所有的机器资源，提高每一台机器的利用率。

在 Kubernetes 中，应用可以通过一定配置，可以在 Kubernetes 的帮助下实现自愈和自运维，从而减少了运维同事的工作压力。

ShardingSphere Helm 云上部署



| Name | Namespa... | Containers | Restarts | Controlled... | Node |
|--|------------|------------|----------|---------------|------|
| <input type="checkbox"/> p1-postgresql-0 | ss | ● | 0 | StatefulSet | 1/1 |
| <input type="checkbox"/> p2-postgresql-0 | ss | ● | 0 | StatefulSet | 1/1 |
| <input type="checkbox"/> p3-postgresql-0 | ss | ● | 0 | StatefulSet | 1/1 |
| <input type="checkbox"/> sc-shardingsphere-cluster-5b... | ss | ● | 0 | ReplicaSet | 3/3 |
| <input type="checkbox"/> sc-shardingsphere-cluster-5b... | ss | ● | 1 | ReplicaSet | 3/3 |
| <input type="checkbox"/> sc-shardingsphere-cluster-5b... | ss | ● | 0 | ReplicaSet | 3/3 |
| <input type="checkbox"/> sc-zookeeper-0 | ss | ● | 0 | StatefulSet | 3/3 |
| <input type="checkbox"/> sc-zookeeper-1 | ss | ● | 0 | StatefulSet | 3/3 |
| <input type="checkbox"/> sc-zookeeper-2 | ss | ● | 0 | StatefulSet | 3/3 |
| <input type="checkbox"/> so-shardingsphere-operator... | ss | ● | 0 | ReplicaSet | 3/3 |
| <input type="checkbox"/> so-shardingsphere-operator... | ss | ● | 0 | ReplicaSet | 3/3 |

Helm 简化了对 Kubernetes 中使用的资源对象的配置编写过程。使得 Apache ShardingSphere 能够在不同环境中快速部署、快速复制。

```
governance:
  ...
  zookeeper:
    replicaCount: 3
  ...
  compute:
  ...
  serverConfig:
    authority:
      privilege:
        type: ALL_PRIVILEGES_PERMITTED
      users:
        - password: root
          user: root@%
    mode:
      overwrite: true
    repository:
      props:
        maxRetries: 3
        namespace: governance_ds
        operationTimeoutMilliseconds: 5000
        retryIntervalMilliseconds: 500
        server-lists: "[[ printf \"%s-zookeeper.%s:2181\" .Release.Name .Release.Namespace ]]
        timeToLiveSeconds: 600
      type: ZooKeeper
    type: Cluster
```

ShardingSphere Operator 自动化运维

Operator 是 Kubernetes API 的客户端，充当自定义资源（ CustomResourceDefinition ）的控制器。

Operator 允许你在不修改 Kubernetes 自身代码的情况下，通过为一个或多个自定义资源关联控制器来实现对集群的扩展能力。

ShardingSphere Operator，可以应对复杂部署形态的 Kubernetes，实现自动化运维，完成 ShardingSphere 在 K8S 中的生命周期管理。



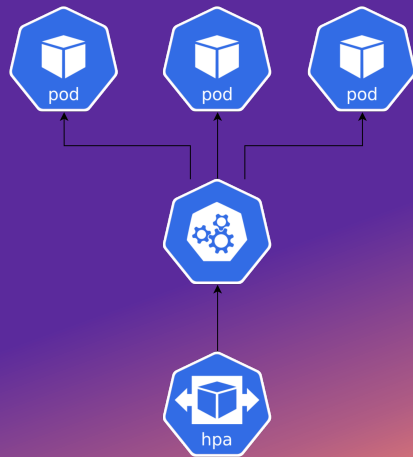
ShardingSphere Operator 自动化运维

水平自动缩放器 (HPA)

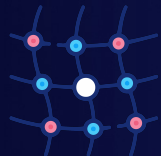
在负载高峰到达的时候，可以实现 ShardingSphere 集群的自动扩容。当高峰过去之后，自动进行缩容，在保证 ShardingSphere 高可用的同时，也节省了成本。

第三方指标

Apache ShardingSphere 支持专有的 Agent 采集运行状态，在 Operator 中可以实现基于 ShardingSphere 指标的 Adapter，便于对接各种指标消费组件。



ShardingSphere 云原生规划



混沌测试可以做到

高可用测试 (Pod 故障注入)

延迟测试

...

混沌测试优势

无侵入

云原生

...

ShardingSphere 云原生规划

- Apache ShardingSphere 云上更多场景的解决方案
(例如: Amazon CloudFormation , ...)
- 治理中心云上方案优化 (Kafka 3.0 去 Zookeeper 版本)

社区交流

项目地址：<https://shardingsphere.apache.org>

GitHub 地址：<https://github.com/apache/shardingsphere>

开发者邮件列表：dev-subscribe@shardingsphere.apache.org

中文社区：<https://community.sphere-ex.com>



技术干货



技术交流

Thank you!

The background features a smooth gradient from dark blue on the left to a vibrant purple and pink on the right. On the right side, there is a complex pattern of thin, white, perspective-distorted lines that create a sense of depth and movement, resembling a grid or architectural structure.