

Apache ShardingSphere 源码解读

张亮
2022.04.23

分享大纲

什么是 Apache ShardingSphere

代码研读建议

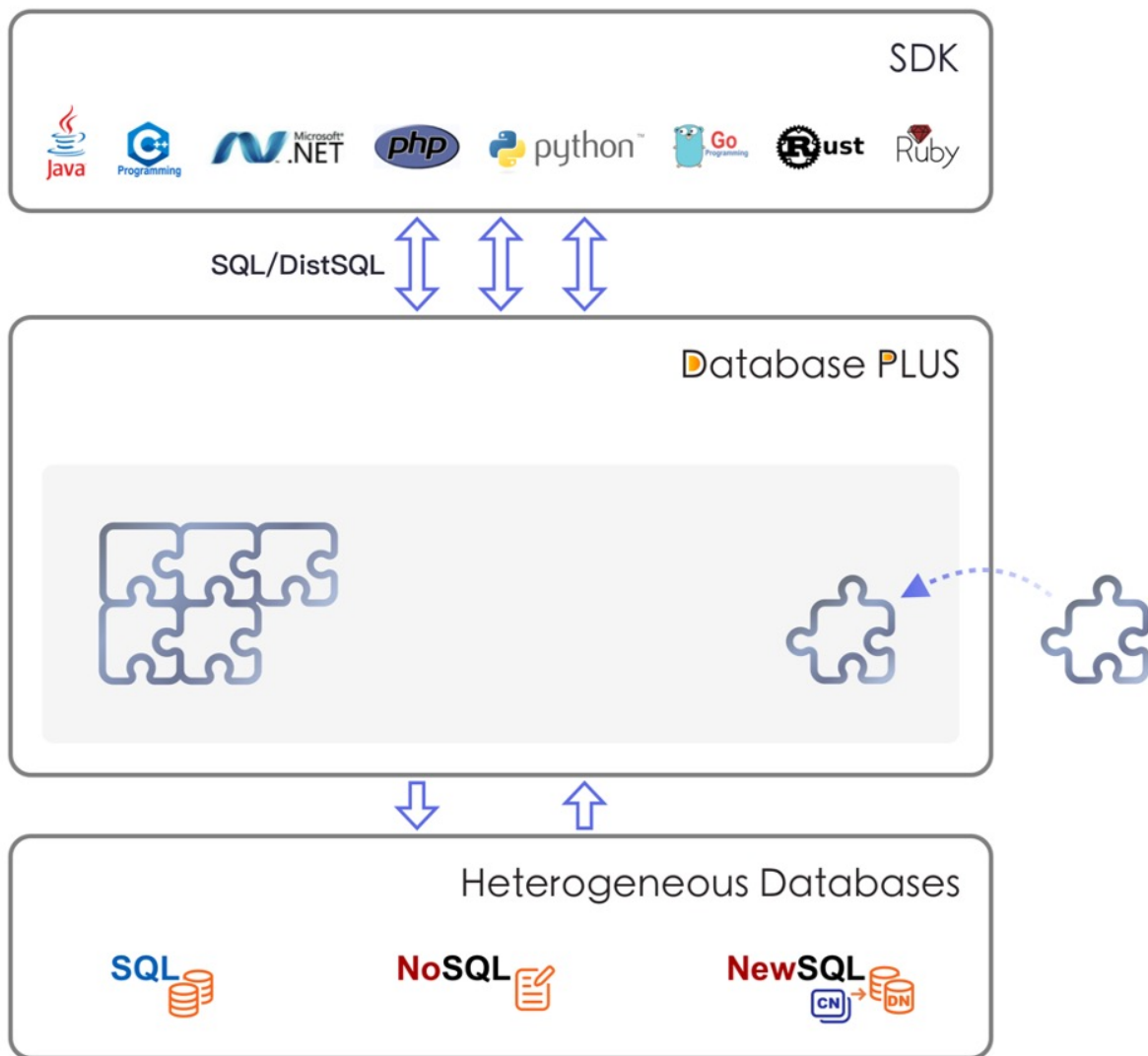
主要结构和知识点梳理

可插拔设计模式以及 SPI 内核实现

Apache ShardingSphere 代码规范



设计哲学：Database Plus



Database Plus：一种分布式数据库系统的设计理念。

旨在碎片化的异构数据库上层构建生态，在最大限度的复用数据库原生存算能力的前提下，进一步提供面向全局的扩展和叠加计算能力（例如：数据分片、数据加密等）。使应用和数据库间的交互面向 Database Plus 构建的标准，从而屏蔽数据库碎片化对上层业务带来的差异化影响。



连接

打造数据库上层标准



增强

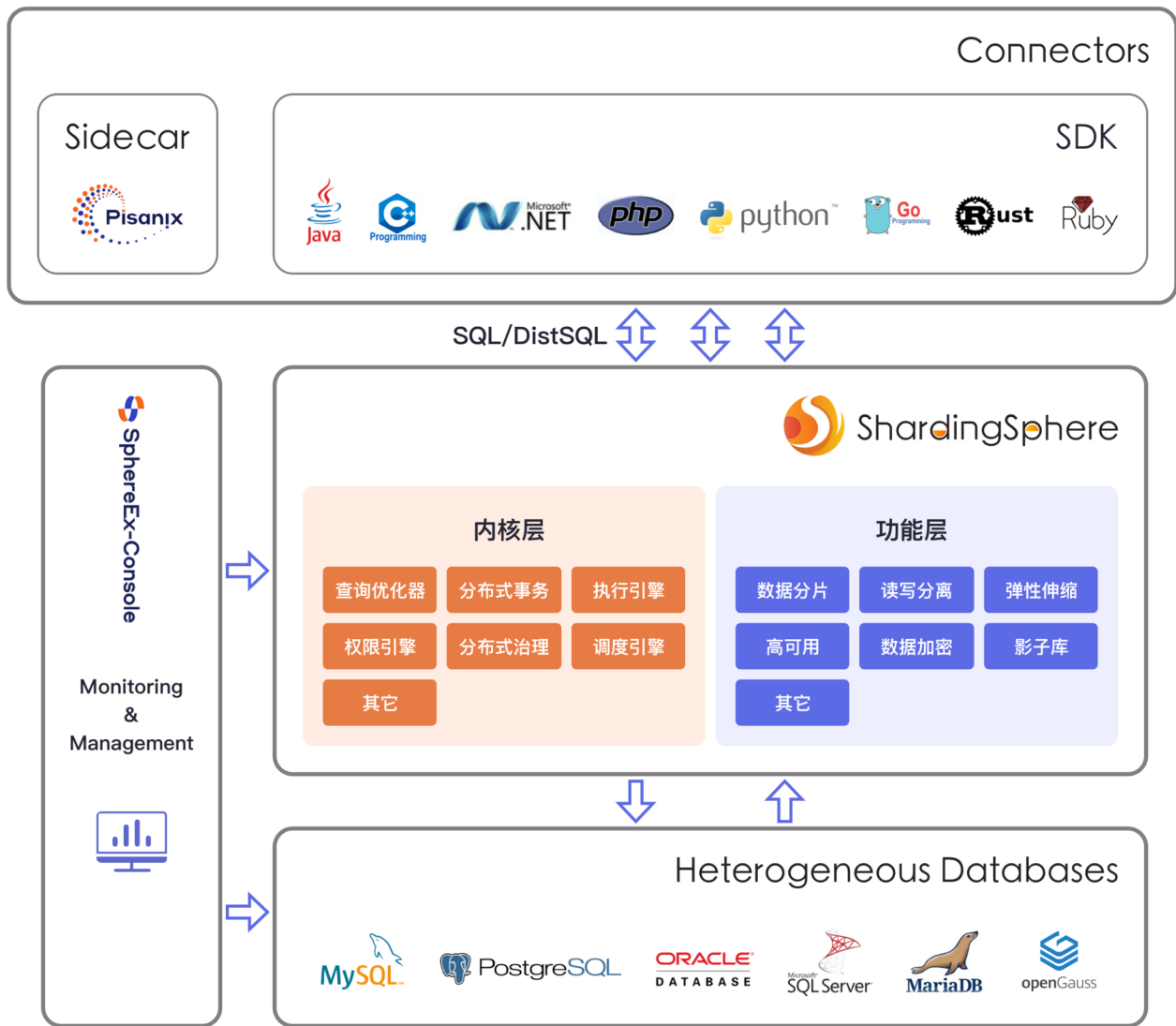
数据库计算增强引擎



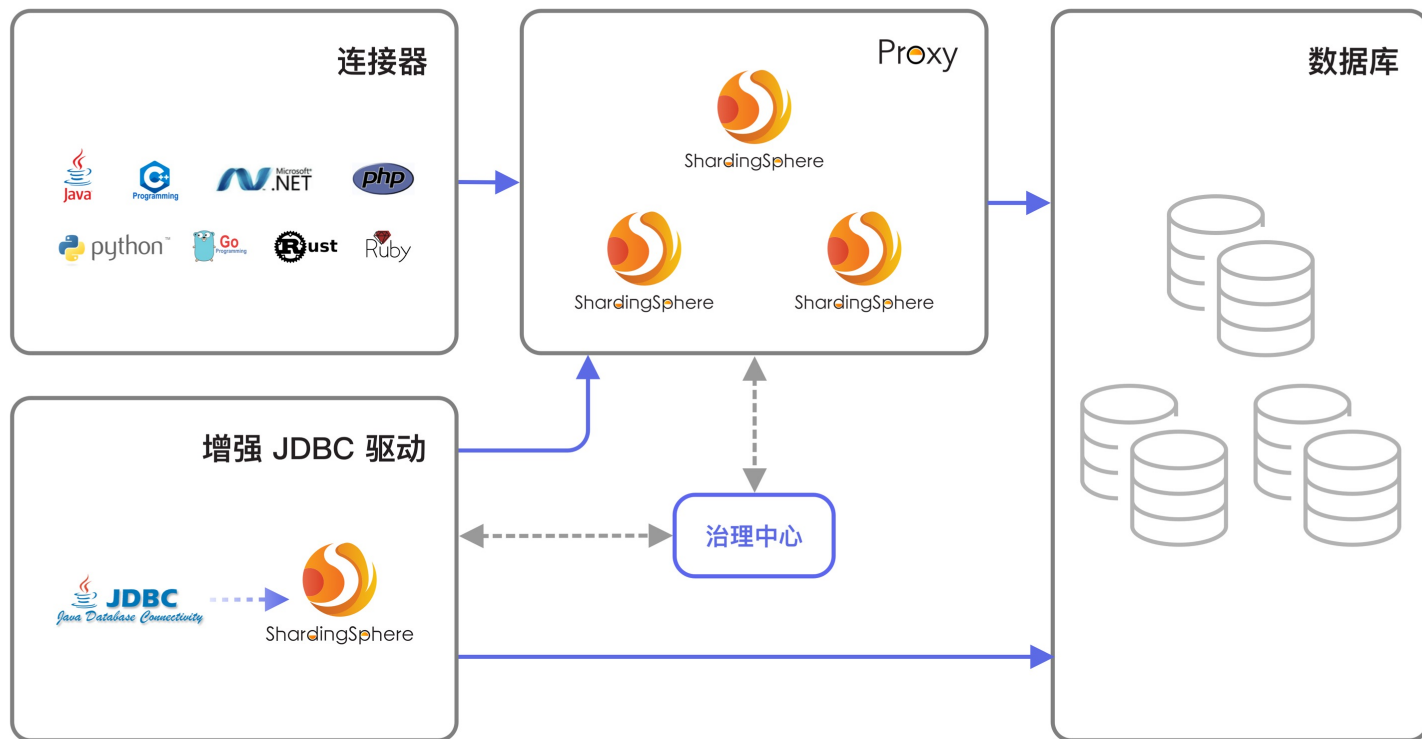
可插拔

构建数据库功能生态

理论实践：Apache ShardingSphere



混合部署的架构拓扑



零侵入 & 低耦合

使用方式与数据库和原生 SDK 一致，零学习成本，对遗留系统无侵入；增强功能与业务应用和数据库完全解耦，防止被数据库和云供应商绑定，对异构数据库、多云和混合云天然友好。

多端访问 & 易运维

多类型接入端可同时在线并组成一体化集群，方便监控和运维。接入端之间可以根据 SQL 规则横向路由，有效梳理集群流量。

高性能 & 单元化

Java SDK 具备极致高并发能力，与业务应用绑定部署，更加适合微服务后端的数据单元化架构。

代码研读建议

代码规模

1. 模块数量：211
2. 代码行数：704,348

代码特征

1. 模块众多
2. 层次结构分明

研读建议

1. 掌握整体脉络
2. 根据兴趣点深入探索

```

> shardingsphere-agent
> shardingsphere-charts
> shardingsphere-db-protocol
> shardingsphere-distribution
> shardingsphere-distsql
> shardingsphere-features
  > shardingsphere-db-discovery
  > shardingsphere-encrypt
  > shardingsphere-readwrite-splitting
  > shardingsphere-shadow
  > shardingsphere-sharding
    > shardingsphere-sharding-api
    > shardingsphere-sharding-core
    > shardingsphere-sharding-distsql
      > shardingsphere-sharding-distsql-handler
      > shardingsphere-sharding-distsql-parser
      > shardingsphere-sharding-distsql-statement
    > target
    m pom.xml
    shardingsphere-sharding-distsql.iml
  > shardingsphere-sharding-spring
  > target
  m pom.xml
  shardingsphere-sharding.iml
> target
m pom.xml
shardingsphere-features.iml

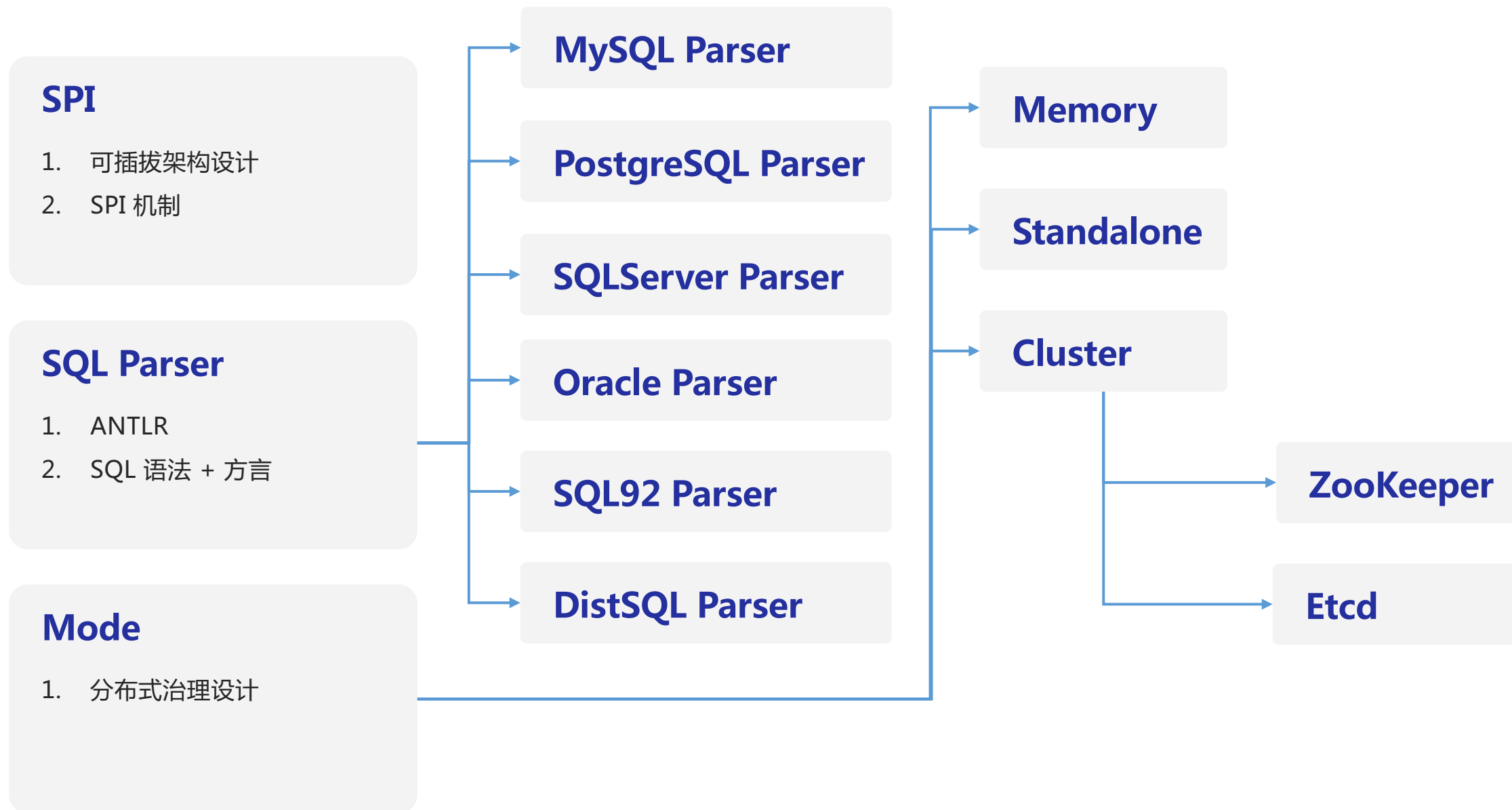
```

```

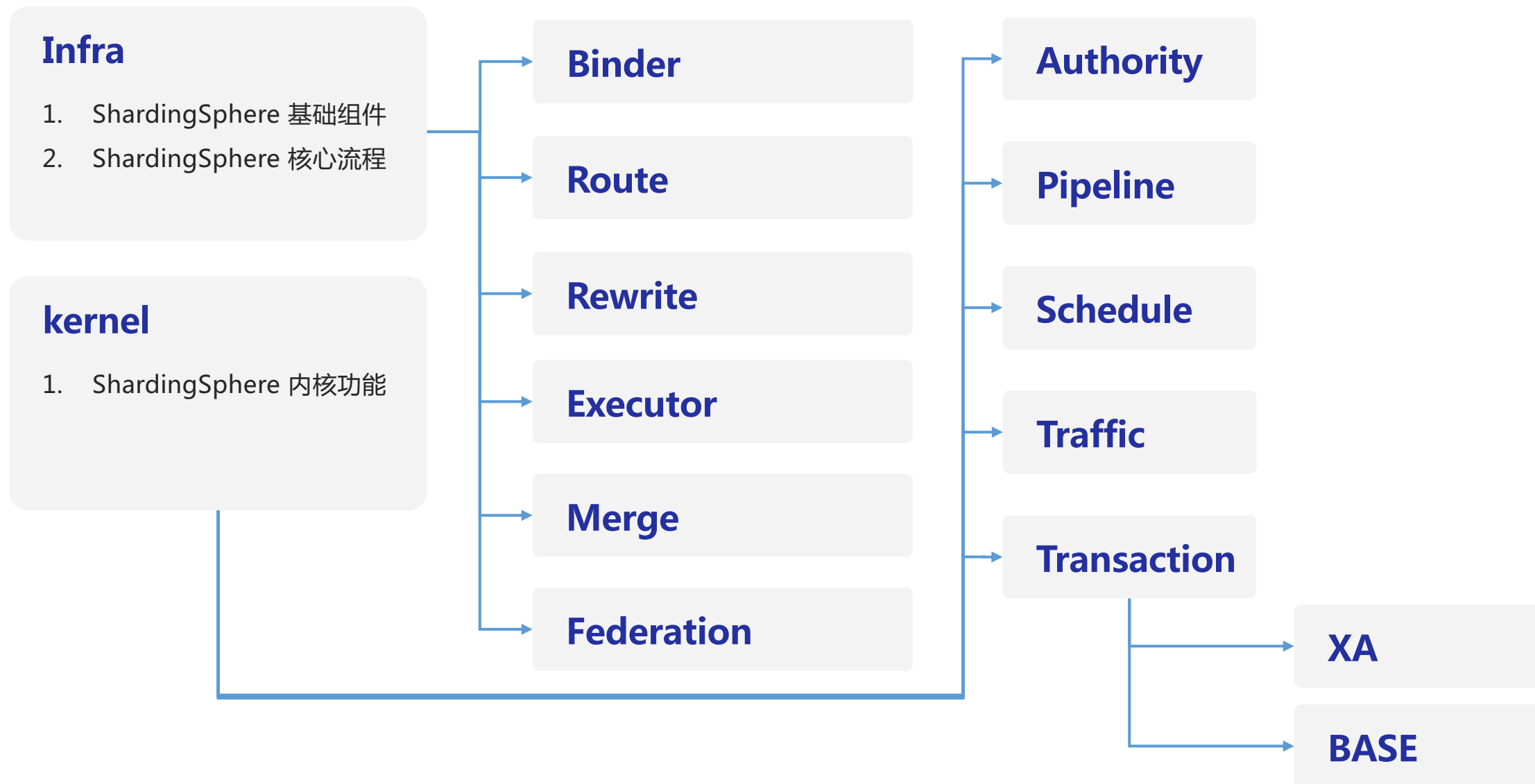
> shardingsphere-infra
> shardingsphere-jdbc
> shardingsphere-kernel
> shardingsphere-mode
> shardingsphere-proxy
> shardingsphere-spi
> shardingsphere-sql-parser
  > shardingsphere-sql-parser-dialect
    > shardingsphere-sql-parser-mysql
    > shardingsphere-sql-parser-opengauss
    > shardingsphere-sql-parser-oracle
    > shardingsphere-sql-parser-postgresql
    > shardingsphere-sql-parser-sql92
    > shardingsphere-sql-parser-sqlserver
  > target
  m pom.xml
  shardingsphere-sql-parser-dialect.iml
  > shardingsphere-sql-parser-engine
  > shardingsphere-sql-parser-spi
  > shardingsphere-sql-parser-statement
  > target
  m pom.xml
  shardingsphere-sql-parser.iml
> shardingsphere-test

```

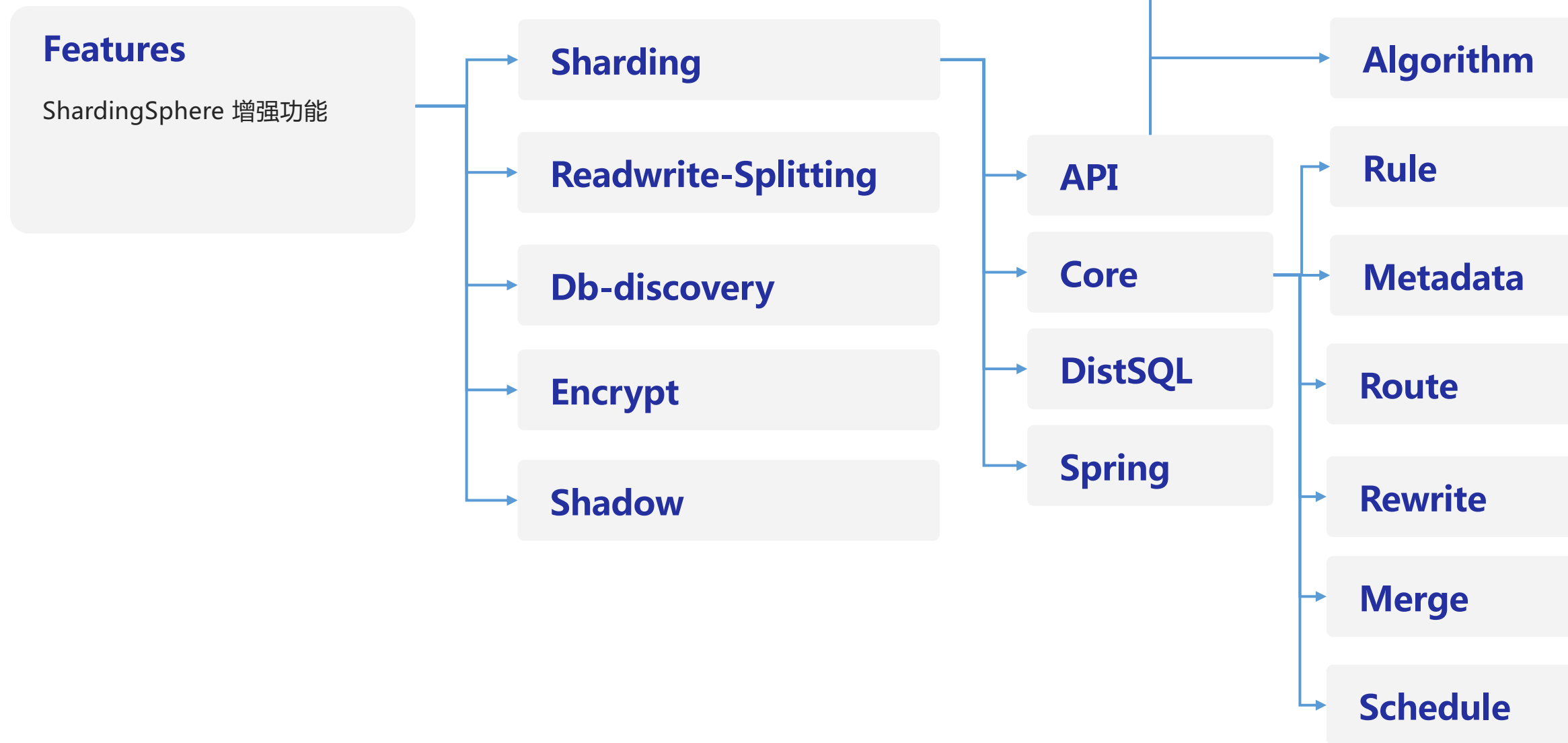
代码结构和主要知识点梳理：基础模块



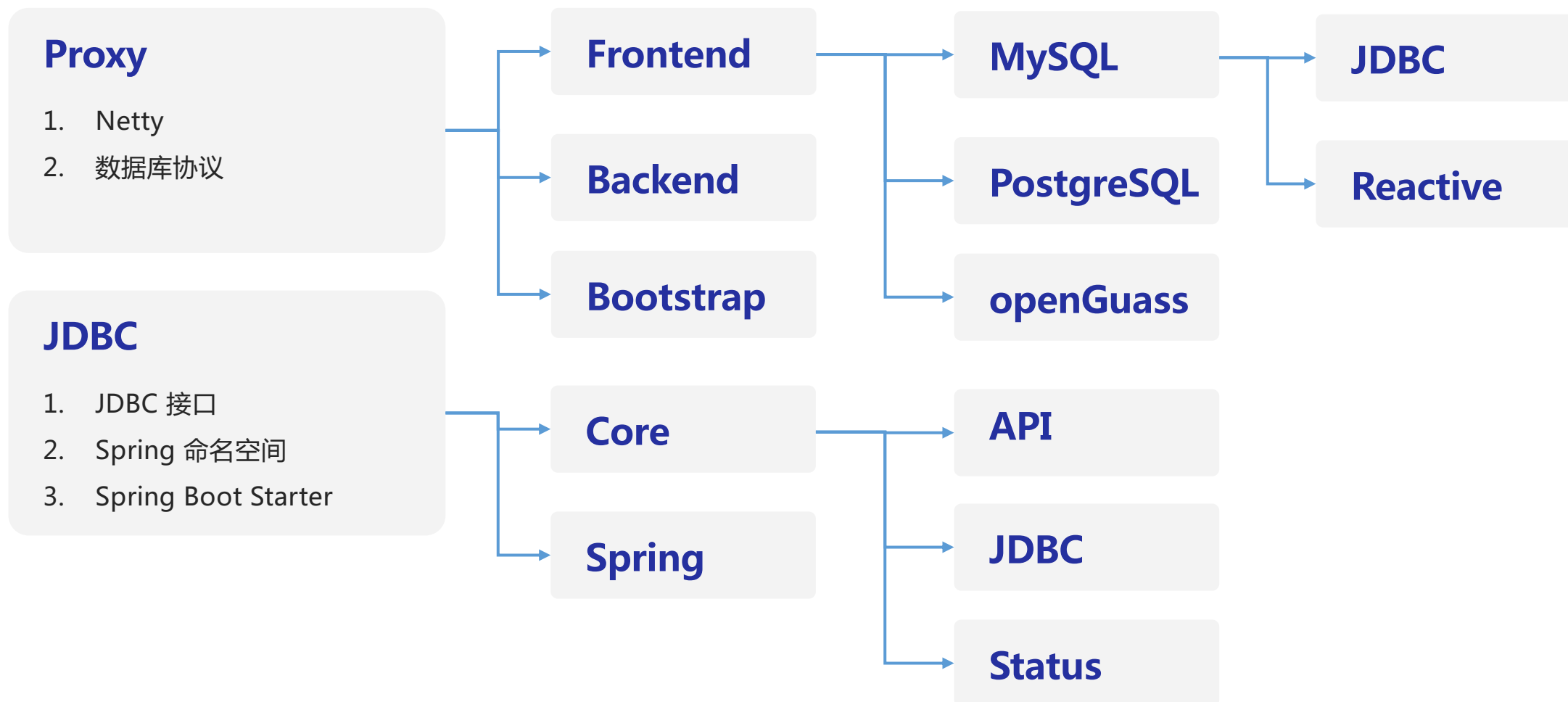
代码结构和主要知识点梳理：核心模块



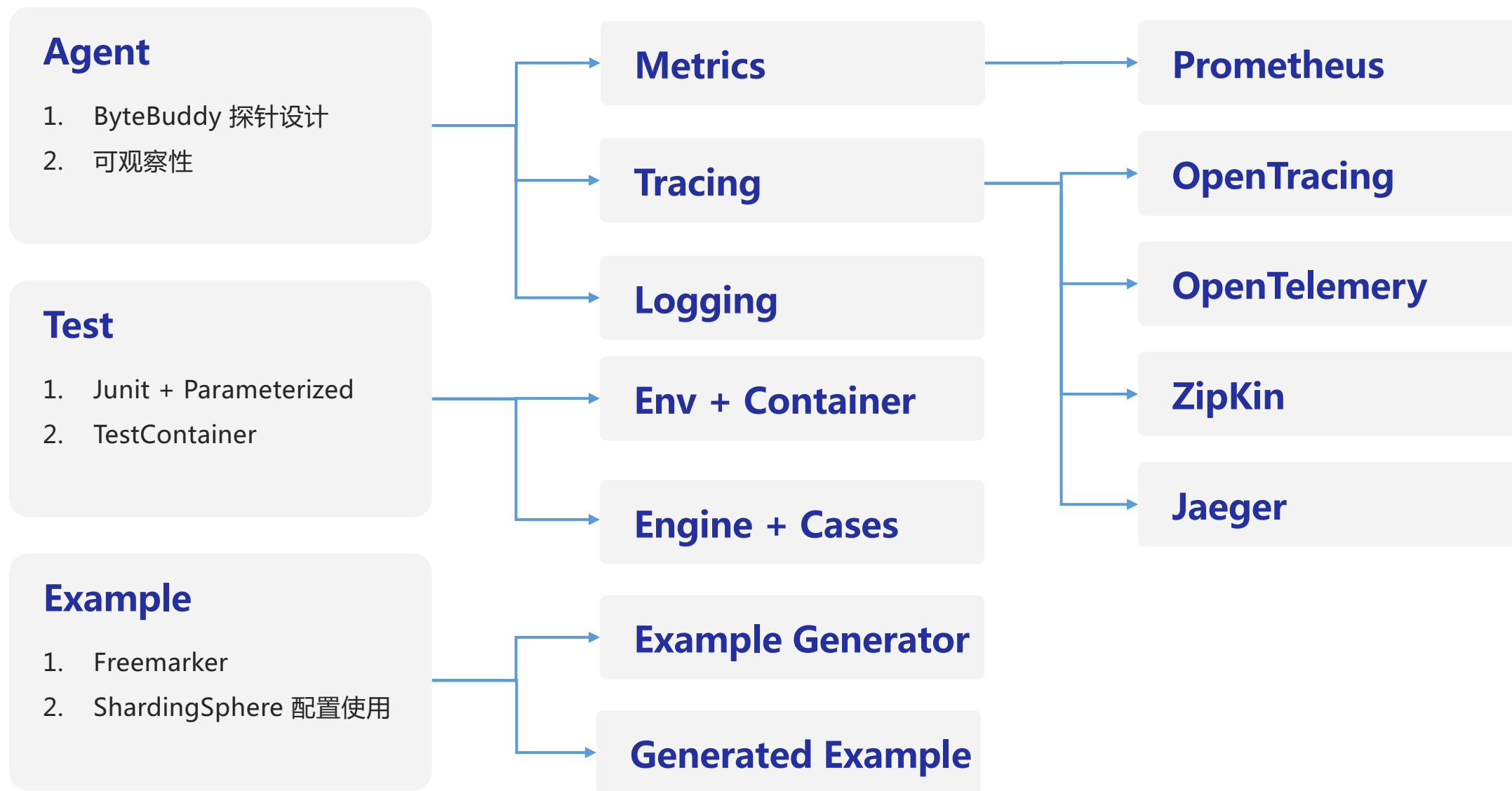
代码结构和主要知识点梳理：功能模块



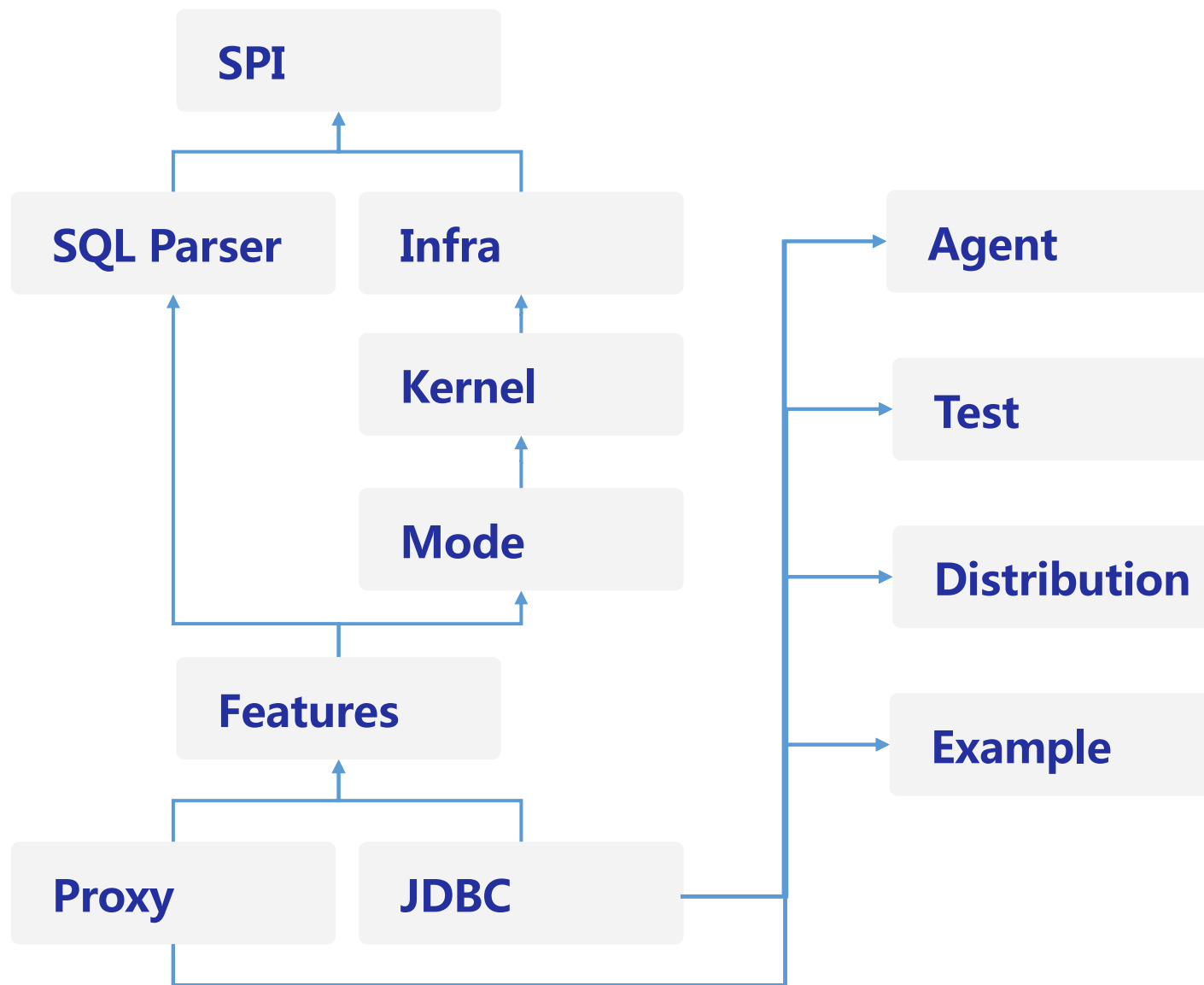
代码结构和主要知识点梳理：接入端



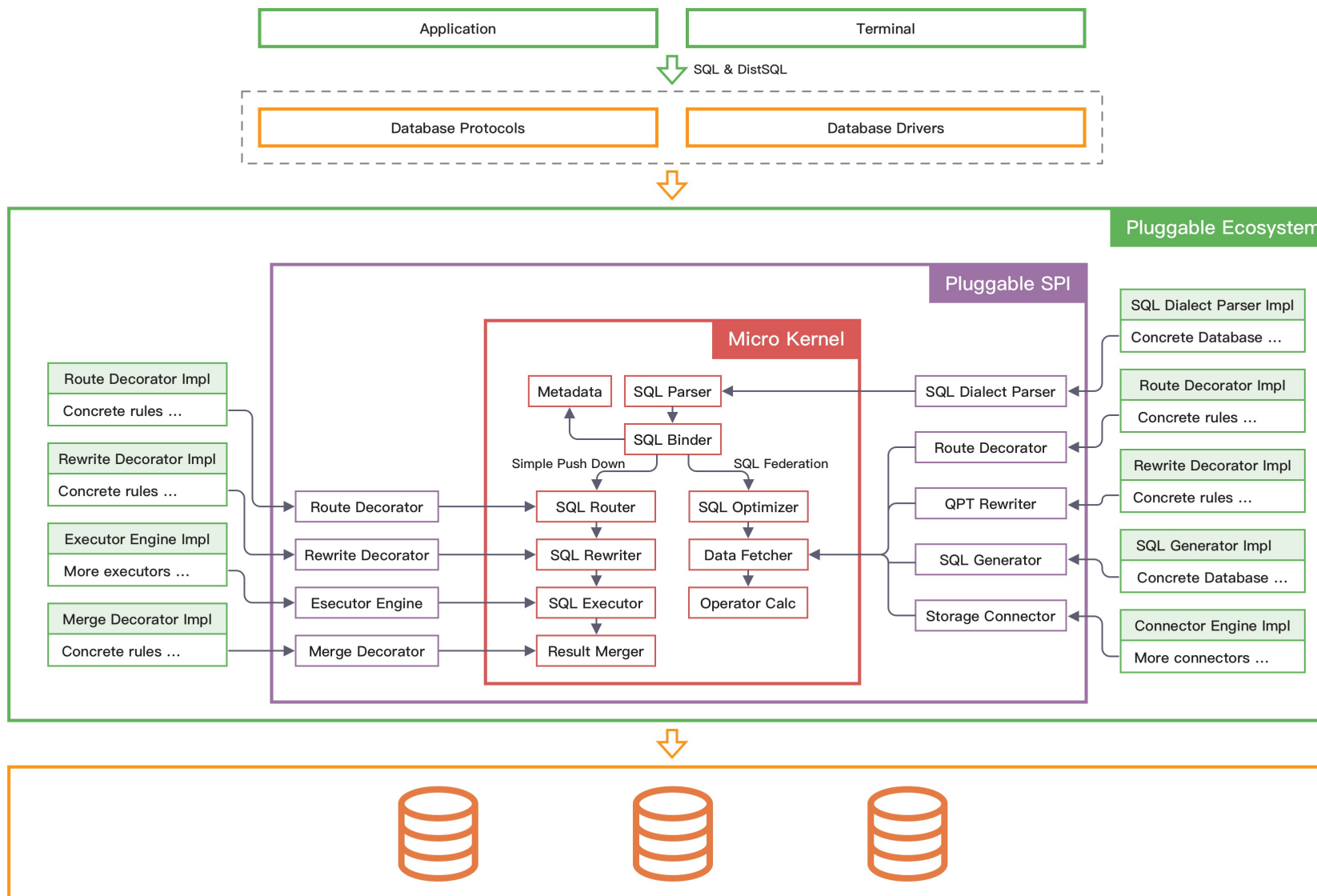
代码结构和主要知识点梳理：监控 & 测试 & 示例



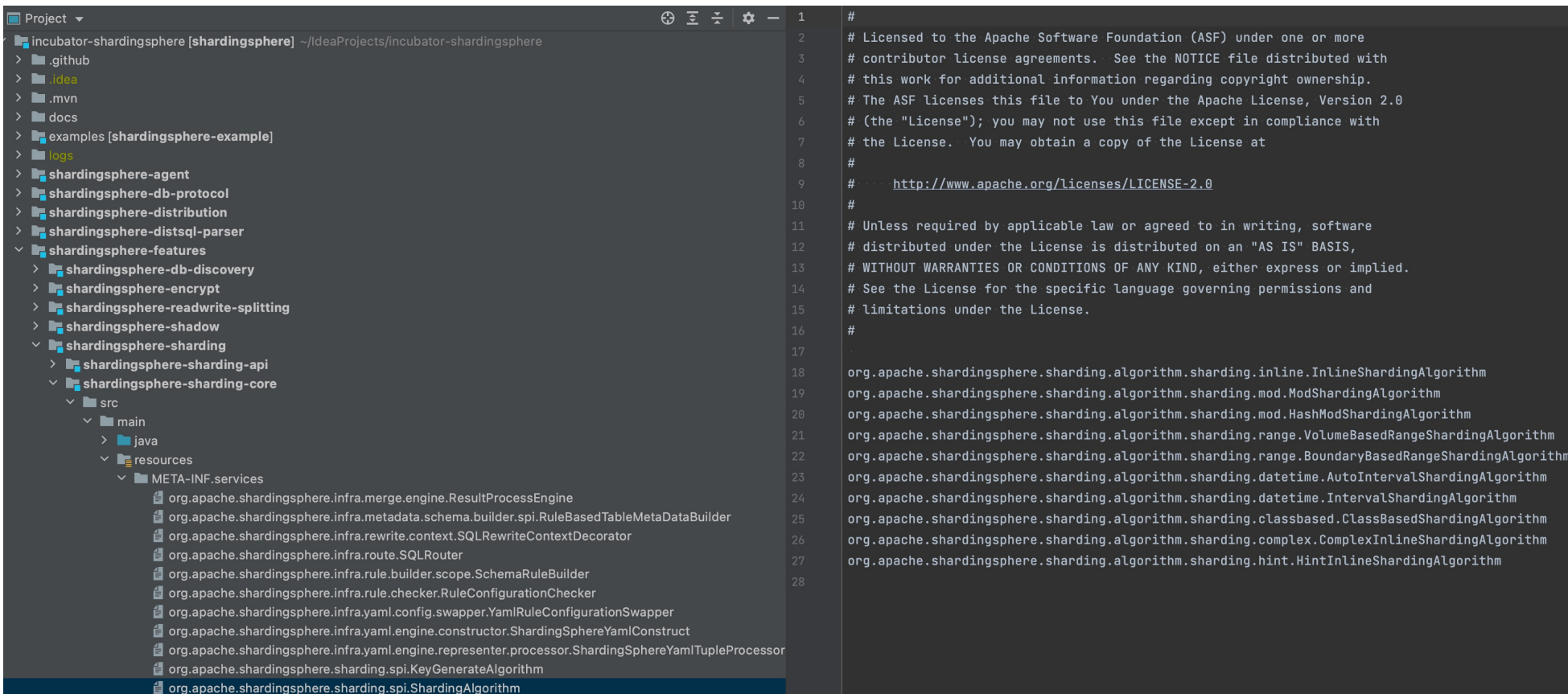
模块依赖关系



可插拔内核架构



ShardingSphere SPI



The screenshot shows an IDE window with a project structure on the left and a license notice on the right.

Project Structure (Left):

- incubator-shardingsphere [shardingsphere]
 - .github
 - .idea
 - .mvn
 - docs
 - examples [shardingsphere-example]
 - logs
 - shardingsphere-agent
 - shardingsphere-db-protocol
 - shardingsphere-distribution
 - shardingsphere-distsql-parser
 - shardingsphere-features
 - shardingsphere-db-discovery
 - shardingsphere-encrypt
 - shardingsphere-readwrite-splitting
 - shardingsphere-shadow
 - shardingsphere-sharding
 - shardingsphere-sharding-api
 - shardingsphere-sharding-core
 - src
 - main
 - java
 - resources
 - META-INF.services
 - org.apache.shardingsphere.infra.merge.engine.ResultProcessEngine
 - org.apache.shardingsphere.infra.metadata.schema.builder.spi.RuleBasedTableMetaDataBuilder
 - org.apache.shardingsphere.infra.rewrite.context.SQLRewriteContextDecorator
 - org.apache.shardingsphere.infra.route.SQLRouter
 - org.apache.shardingsphere.infra.rule.builder.scope.SchemaRuleBuilder
 - org.apache.shardingsphere.infra.rule.checker.RuleConfigurationChecker
 - org.apache.shardingsphere.infra.yaml.config.swapper.YamlRuleConfigurationSwapper
 - org.apache.shardingsphere.infra.yaml.engine.constructor.ShardingSphereYamlConstruct
 - org.apache.shardingsphere.infra.yaml.engine.representer.processor.ShardingSphereYamlTupleProcessor
 - org.apache.shardingsphere.sharding.spi.KeyGenerateAlgorithm
 - org.apache.shardingsphere.sharding.spi.ShardingAlgorithm

License Notice (Right):

```

1 #
2 # Licensed to the Apache Software Foundation (ASF) under one or more
3 # contributor license agreements. See the NOTICE file distributed with
4 # this work for additional information regarding copyright ownership.
5 # The ASF licenses this file to You under the Apache License, Version 2.0
6 # (the "License"); you may not use this file except in compliance with
7 # the License. You may obtain a copy of the License at
8 #
9 #   http://www.apache.org/licenses/LICENSE-2.0
10 #
11 # Unless required by applicable law or agreed to in writing, software
12 # distributed under the License is distributed on an "AS IS" BASIS,
13 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
14 # See the license for the specific language governing permissions and
15 # limitations under the License.
16 #
17
18 org.apache.shardingsphere.sharding.algorithm.sharding.inline.InlineShardingAlgorithm
19 org.apache.shardingsphere.sharding.algorithm.sharding.mod.ModShardingAlgorithm
20 org.apache.shardingsphere.sharding.algorithm.sharding.mod.HashModShardingAlgorithm
21 org.apache.shardingsphere.sharding.algorithm.sharding.range.VolumeBasedRangeShardingAlgorithm
22 org.apache.shardingsphere.sharding.algorithm.sharding.range.BoundaryBasedRangeShardingAlgorithm
23 org.apache.shardingsphere.sharding.algorithm.sharding.datetime.AutoIntervalShardingAlgorithm
24 org.apache.shardingsphere.sharding.algorithm.sharding.datetime.IntervalShardingAlgorithm
25 org.apache.shardingsphere.sharding.algorithm.sharding.classbased.ClassBasedShardingAlgorithm
26 org.apache.shardingsphere.sharding.algorithm.sharding.complex.ComplexInlineShardingAlgorithm
27 org.apache.shardingsphere.sharding.algorithm.sharding.hint.HintInlineShardingAlgorithm
28

```

SPI 类型

1. TypedSPI, 如: 算法
2. OrderedSPI, 如: 规则
3. RequiredSPI, 如: 内核
4. OptionalSPI, 如: 附加

SPI 创建

1. Singleton
2. PropertiesRequired
3. PostProcessor

ShardingSphere 代码规范



用心

保持责任心和敬畏心，以工匠精神持续雕琢。



可读

代码无歧义，通过阅读而非调试手段浮现代码意图。



整洁

认同《重构》和《代码整洁之道》的理念，追求整洁优雅代码。



一致

代码风格、命名以及使用方式保持完全一致。



精简

极简代码，以最少的代码表达最正确的意思。高度复用，无重复代码和配置。及时删除无用代码。



抽象

层次划分清晰，概念提炼合理。保持方法、类、包以及模块处于同一抽象层级。



极致

拒绝随意，保证任何一行代码、任何一个字母、任何一个空格都有其存在价值。

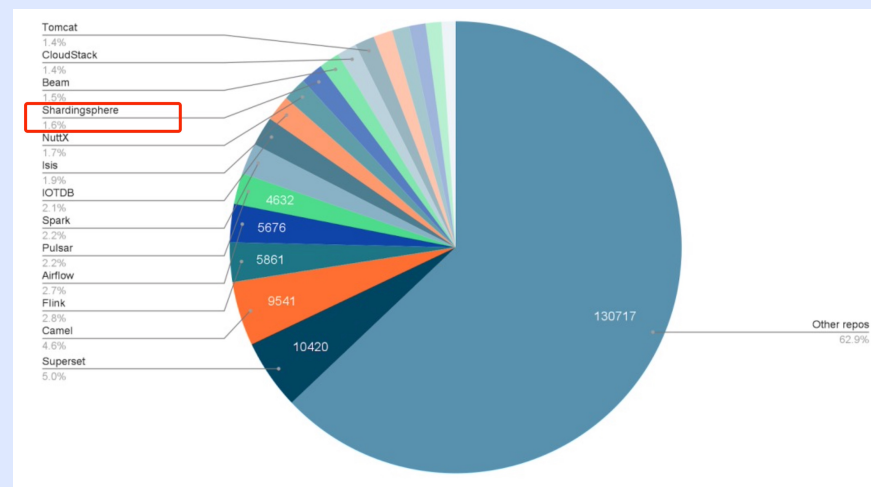
ShardingSphere 等你来贡献!



- 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



- **15,000+** Stars
- **300+** Contributors
- **5,000+** Forks
- **8,000+** 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 年度报告代码提交数量位列前十
- ◆ **学术界认可**：数据库顶会 ICDE 2022 发表论文《A Holistic and Pluggable Platform for Data Sharding》

欢迎关注我们！



技术干货



加入交流群

SphereEx 官网 : <https://sphere-ex.com>

Apache ShardingSphere Website : <https://shardingsphere.apache.org>

Apache ShardingSphere GitHub : <https://github.com/apache/shardingsphere>

Apache ShardingSphere Slack Channel : <https://apacheshardingsphere.slack.com>

谢谢观看