Insight Into Apache ShardingSphere

SphereEx@Zhengqiang Duan

About me

Zhengqiang Duan 端正强

- SphereEx Java Senior Software Engineer
- Committer of Apache ShardingSphere
- Love open source, currently focusing on the development of Apache ShardingSphere database middleware.

SphereEx

ShardingSphere

CONTENTS

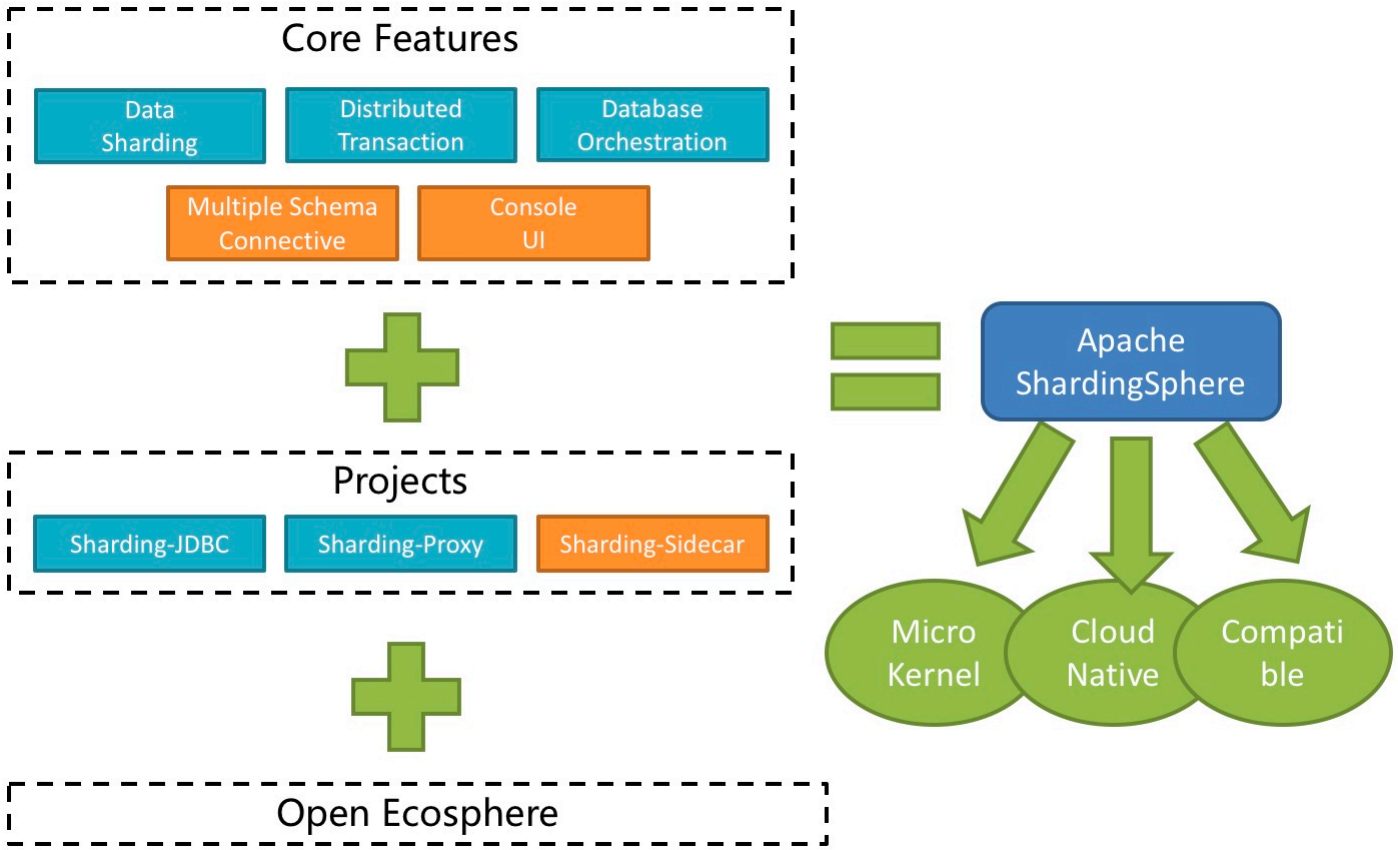
目录

01

Introduction

简介

Introduction



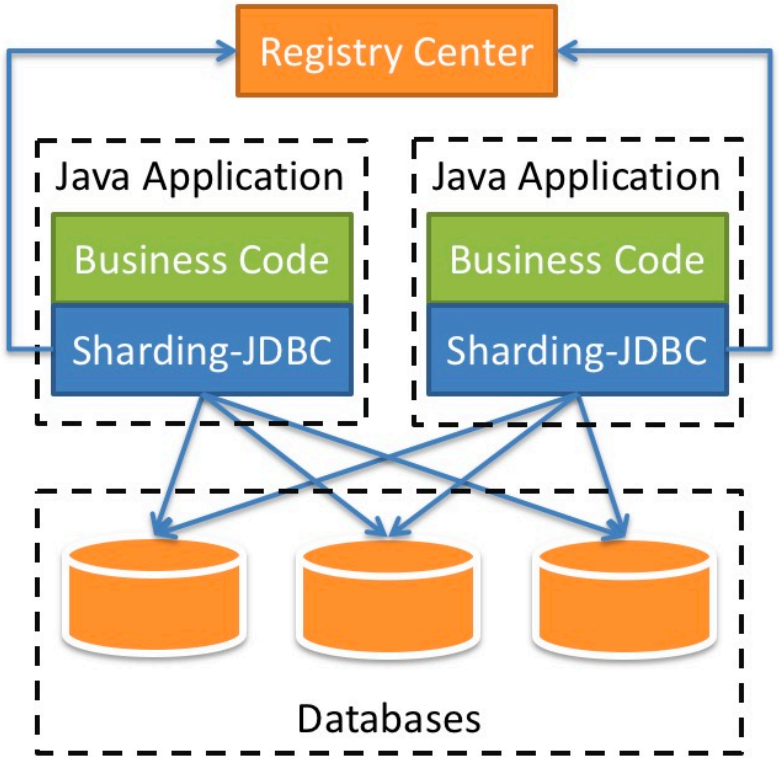
- Apache Top Level Project
- Open-source distributed database ecosystem
- Provide data sharding, distributed transaction and distributed governance functions
- Support MySQL, Oracle, SQLServer, PostgreSQL and any SQL92 followed databases

02

Architecture

架构

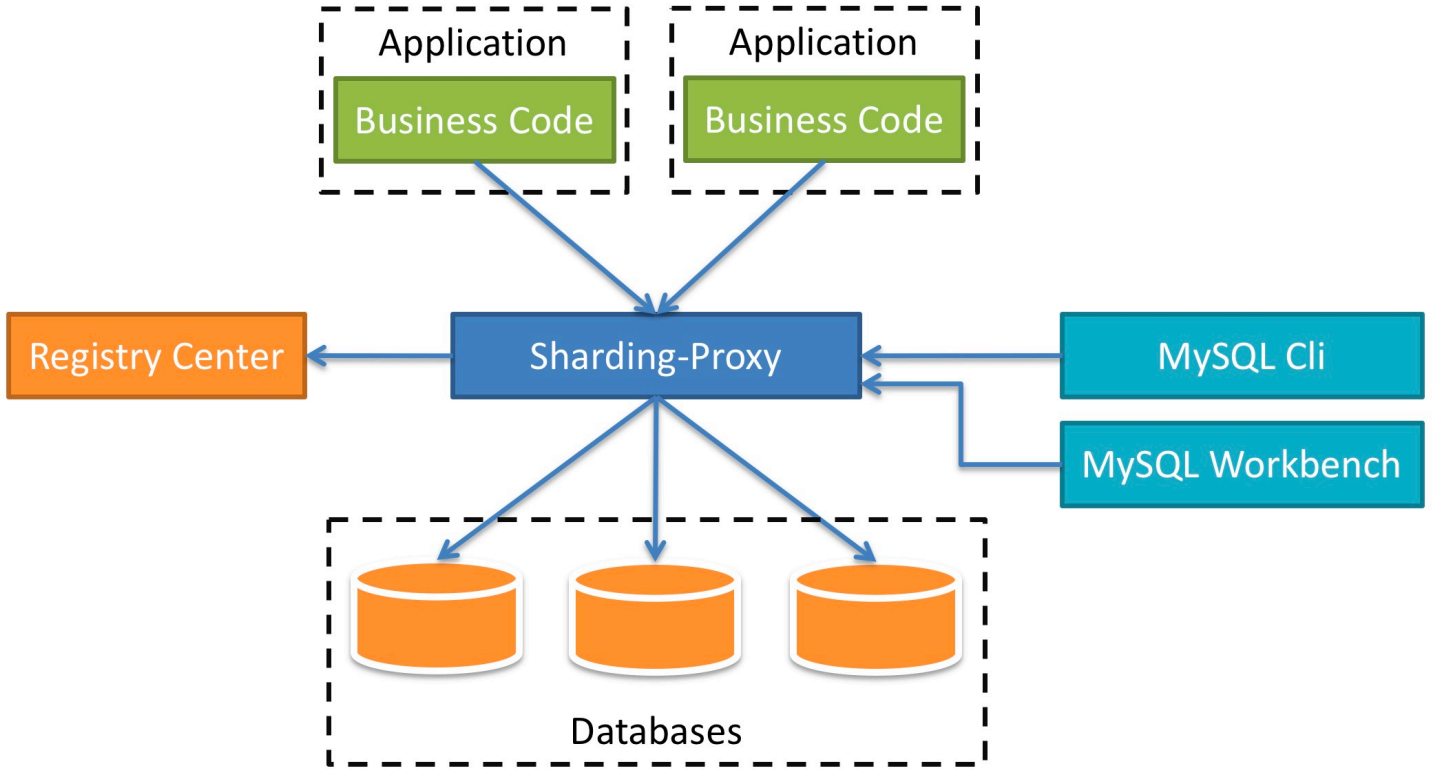
Architecture-Sharding-JBDC



Lightweight Java framework that provides extra service at Java JDBC layer.

- Applicable in any ORM framework based on JDBC
- Support any third-party database connection pool
- Support any kind of JDBC standard database——MySQL, Oracle, SQLServer, PostgreSQL and any SQL92 followed databases

Architecture-Sharding-Proxy

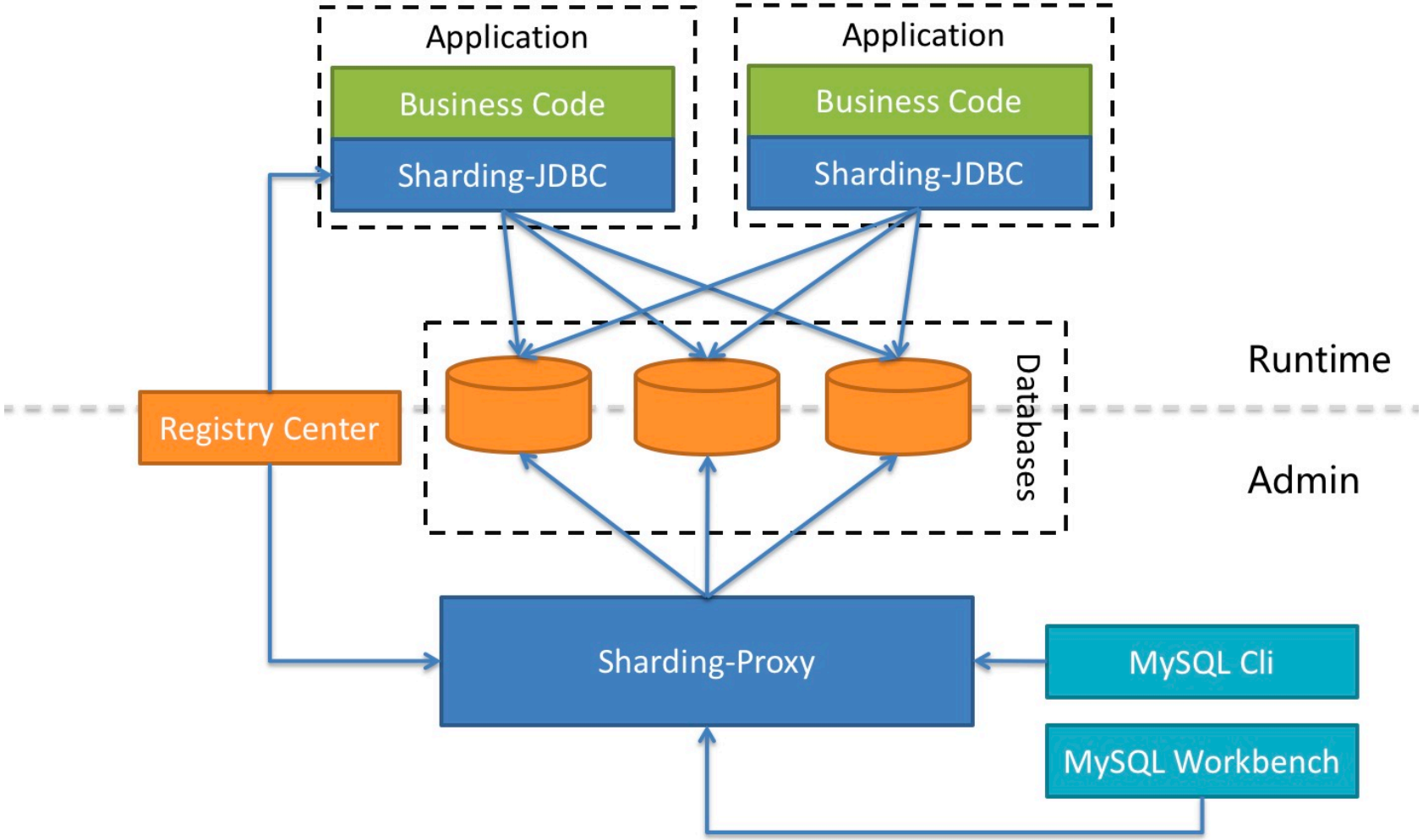


- Transparent to applications, it can be used directly as MySQL/PostgreSQL
- Applicable to any kind of terminal that is compatible with MySQL and PostgreSQL protocol

Architecture-Compare

	Sharding-JDBC	Sharding-Proxy
Database	Any	MySQL/PostgreSQL
Connections Count Cost	High	Low
Supported Languages	Java Only	Any
Performance	Low loss	Relatively High loss
Decentralization	Yes	No
Static Entry	No	Yes

Architecture-Hybrid



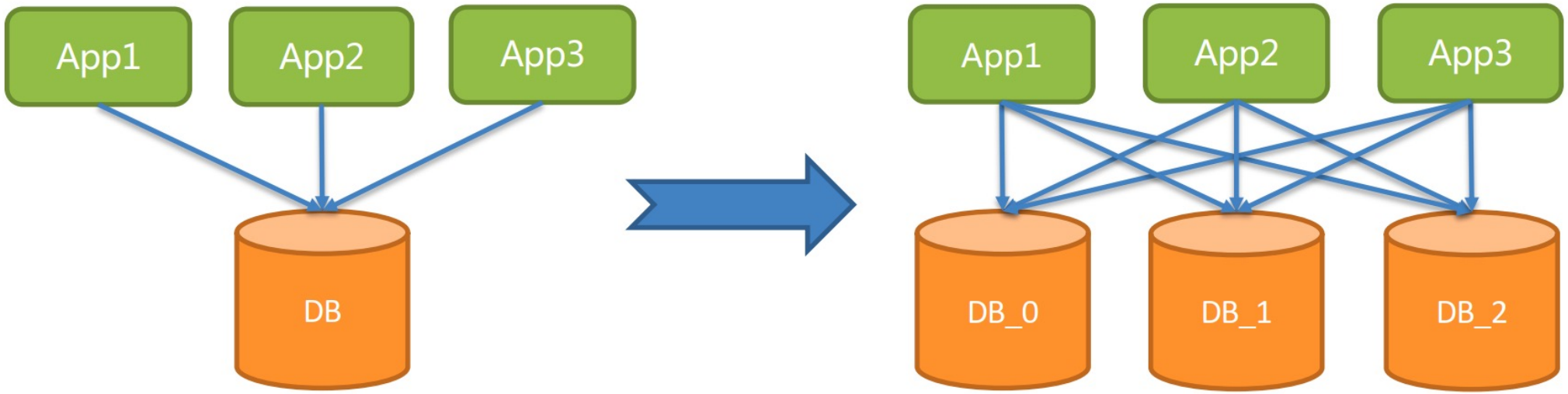
03

Features

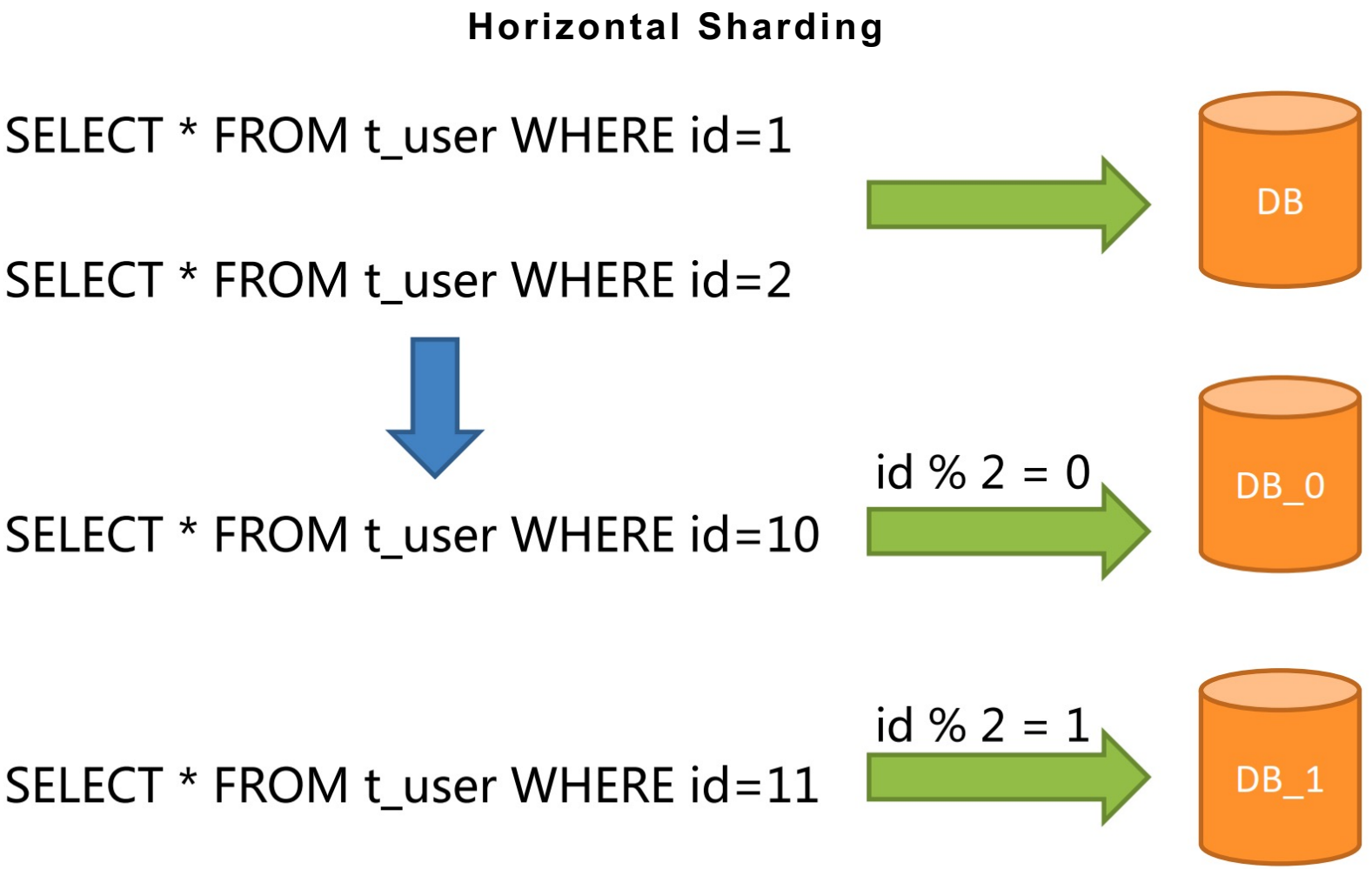
特 性

Features - Data sharding

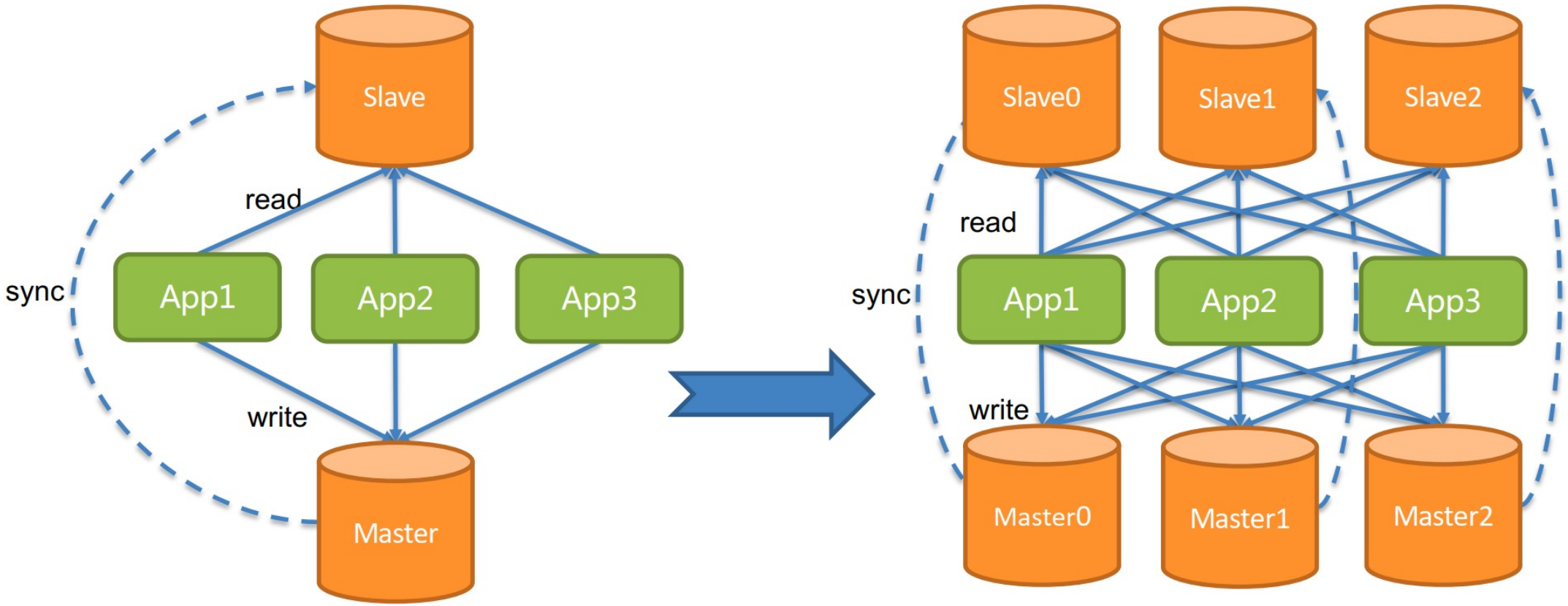
Vertical Sharding



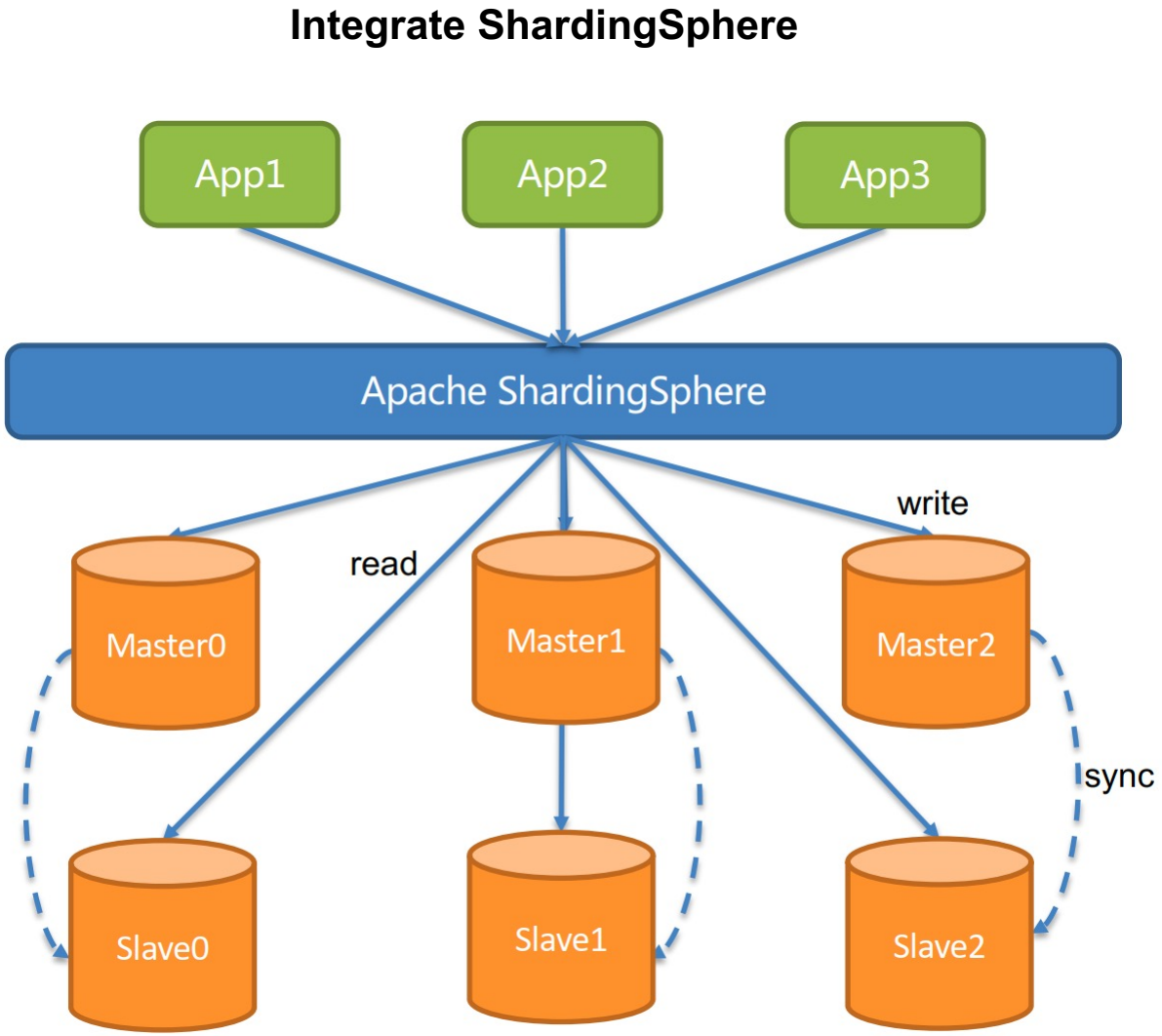
Features - Data sharding



Features - Data sharding



Features - Data sharding



Features - Data sharding

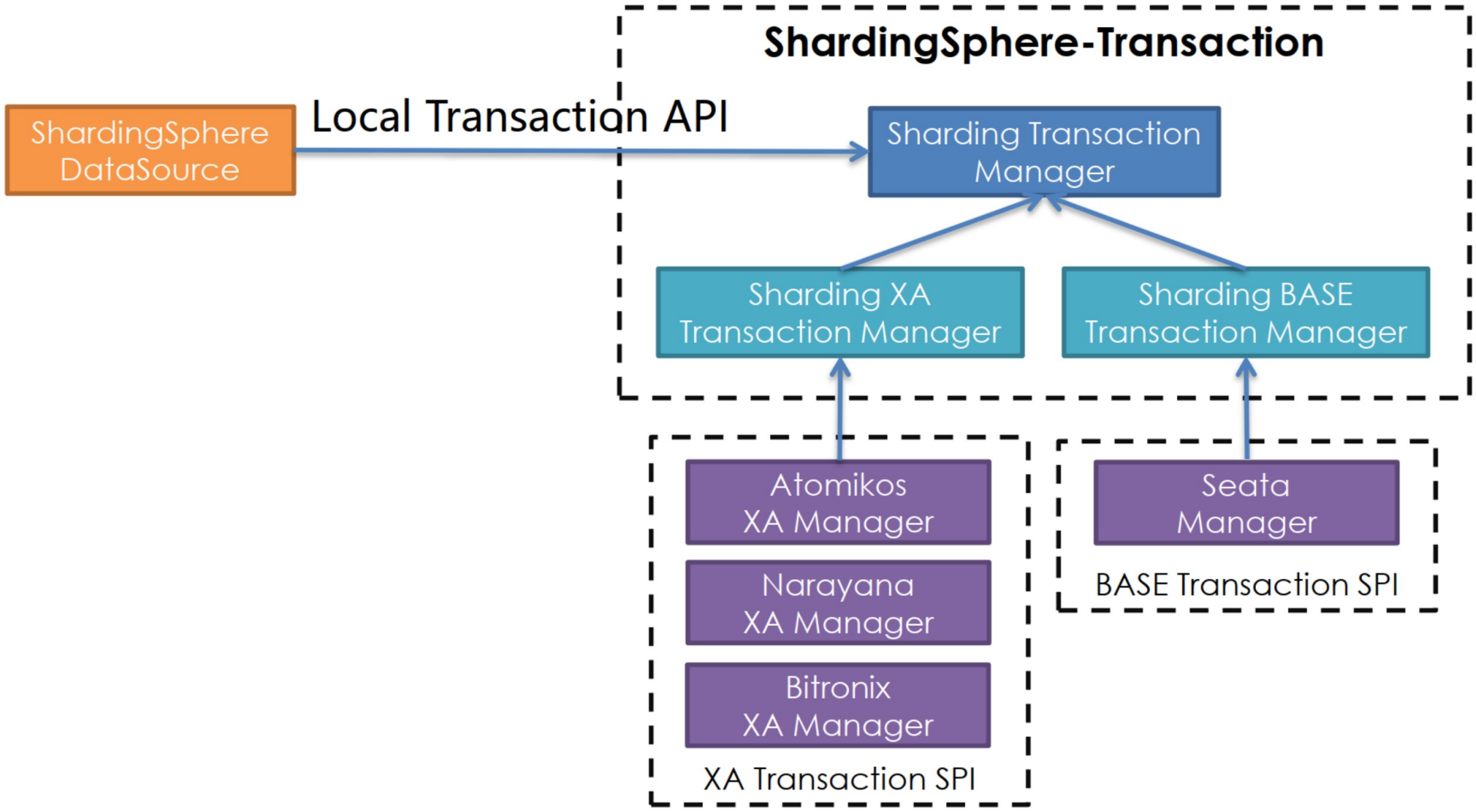
- Execute SQL transparently: oriented to logical SQL operations

- Customized Sharding Algorithm
  - Java Class: xxx.xxx. XXXShardingAlgorithm
  - Inline Expression: t_order_$->{order_id % 2}
  - Built-in algorithm: hash, range, time

- Built-in distributed ID generator
  - UUID
  - SNOWFLAKE

```
db_0
    ├── t_order_0
    └── t_order_1
db_1
    ├── t_order_0
    └── t_order_1

logic sql: select * from t_order where order_id = ?;
actual sql: select * from db_0.t_order_0 where order_id = ?;
```

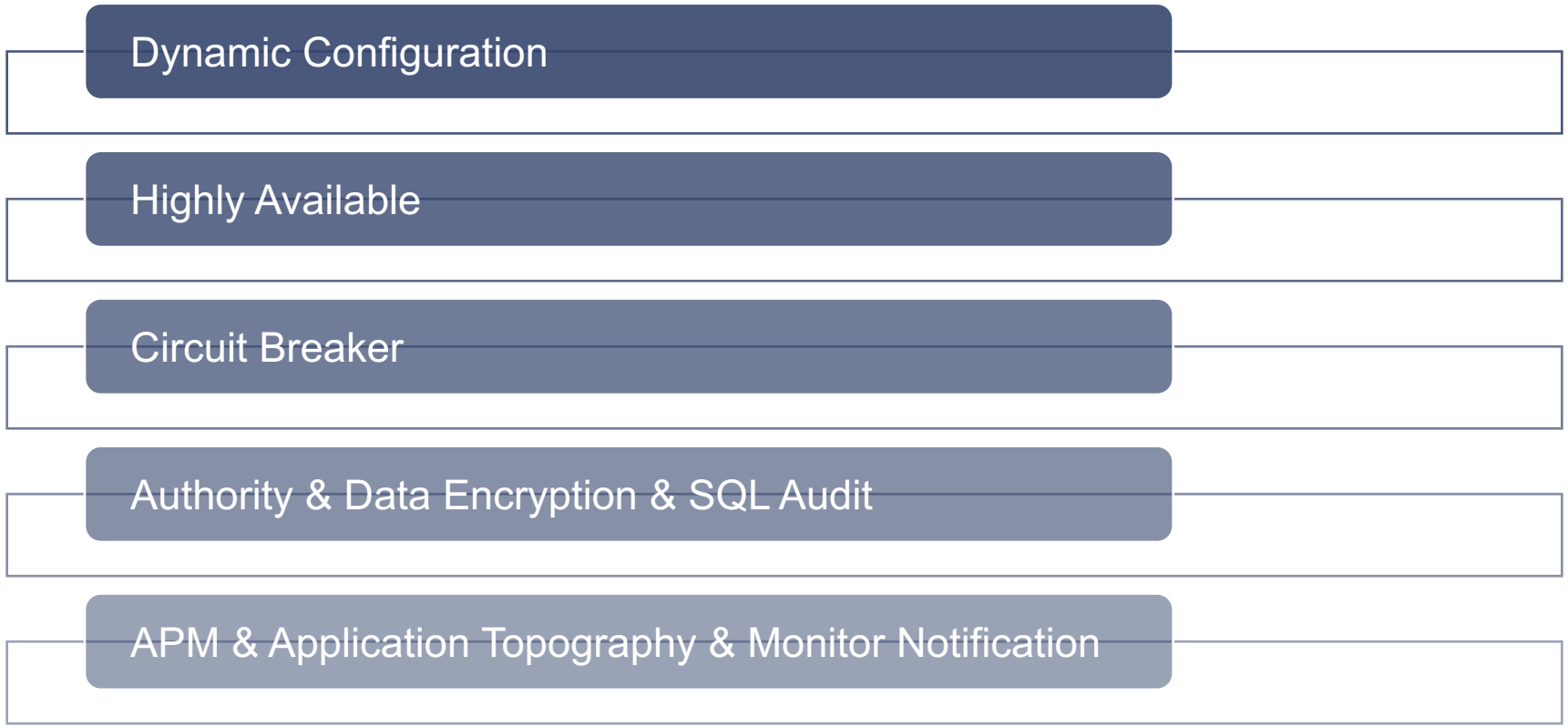
Features - Distributed transaction



Features - Distributed transaction

	Local transaction	2PC (3PC) transaction	BASE transaction
Business transformation	None	None	Relevant interface
Consistency	Not support	Support	Eventual consistency
Isolation	Not support	Support	Business-side guarantee
Concurrency performance	No influence	Serious recession	Minor recession
Situation	Inconsistent operation at business side	Short transaction & low concurrency	Long transaction & high concurrency

Features - Database Governance



Features - New Features

- Dist SQL (Distributed SQL): design to break the boundary between middleware and database
  - RDL (Resource & Rule Definition Language)
  - RQL (Resource & Rule Query Language)
  - SCTL (ShardingSphere Control Language)

- Data Scaling (Dev): common solution for migrating or scaling data

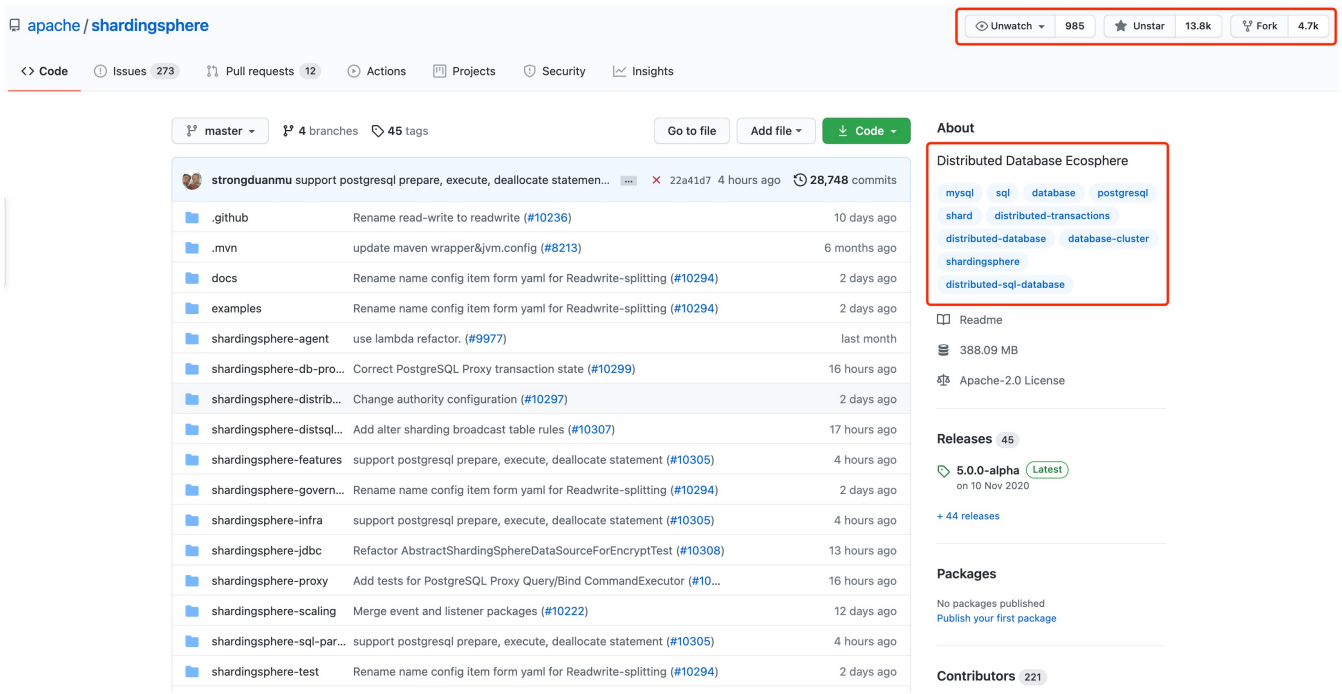
```
CREATE datasources (
ds0=127.0.0.1:3306:demo_ds_0:root:pwd,
ds1=127.0.0.1:3306:demo_ds_1:root:pwd)

CREATE SHARDING RULE (
t_order=hash_mod(order_id, 4),
t_item=mod(item_id, 2)
)
```

04

Community

社区

Community



Website: https://shardingsphere.apache.org
GitHub: https://github.com/apache/shardingsphere

Community

https://shardingsphere.apache.org/community/en/powered-by/

Welcome on board

https://shardingsphere.apache.org/

https://github.com/apache/shardingsphere

mailto:dev-subscribe@shardingsphere.apache.org

wechat group

THANKS