삼성 오픈소스 컨퍼런스

샤딩스피어 어디까지 써봤니?

지극히 개인적인 경험 중심의 데이터 샤딩 이야기

카카오뱅크 | 인프라 | 성동찬 (Chan.c)

2019.10.17

소개

우육빛깔

까칠행원

- https://gywn.net
- https://gywn.blog
- https://www.facebook.com/dongchan.sung

절대 깨지지 않는 견고한 서비스를 지향하는 국내 최초(아마도) 은행 오픈소스 데이터베이스 엔지니어

(KT하이텔 > 티몬 > 카카오 > 한국카카오은행)

목차

1. 샤딩이란 무엇인가!

초간단 훑어보는 샤딩 이야기

1. 샤딩이란 무엇인가!

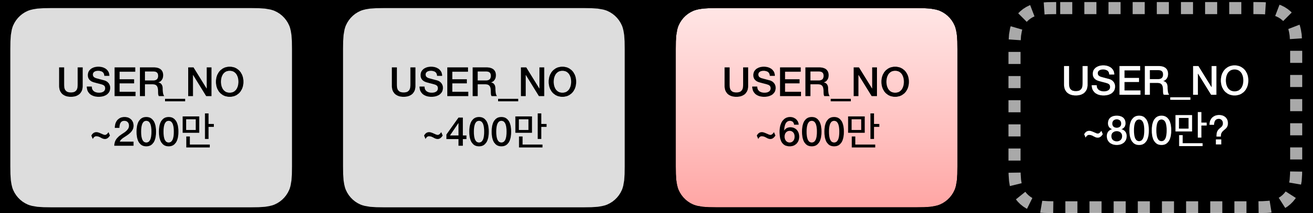
Huge Data

1. 샤딩이란 무엇인가!

1. 샤딩이란 무엇인가!

Range Sharding

- 범위로 데이터를 쪼갬
- 샤드 추가가 상대적으로 쉬움
- 샤드 불균형이 발생할 수 있음



1. 샤딩이란 무엇인가!

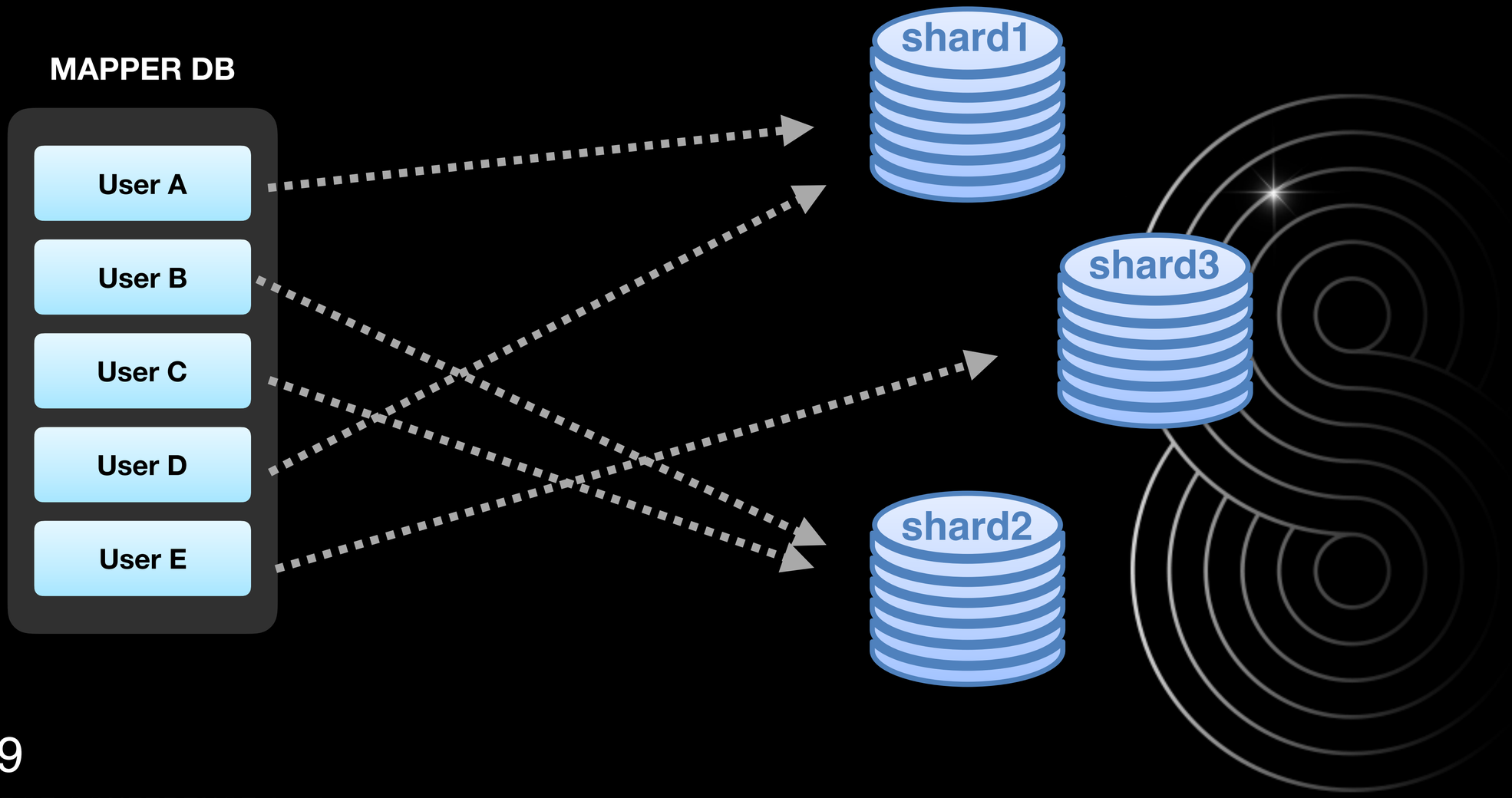
Modulus Sharding

- 나머지 연산으로 데이터를 쪼갬
- 전반적으로 고른 트래픽 분포
- 샤드 확장이 굉장히 어려움

CRC32(USER_ID) mod 4

0	1	2	3

1. 샤딩이란 무엇인가!


MAPPER DB
User A
User B
User C
User D
User E
shard1
shard3
shard2

샤딩이란?

샤딩은 데이터를 쪼개는 방법입니다.

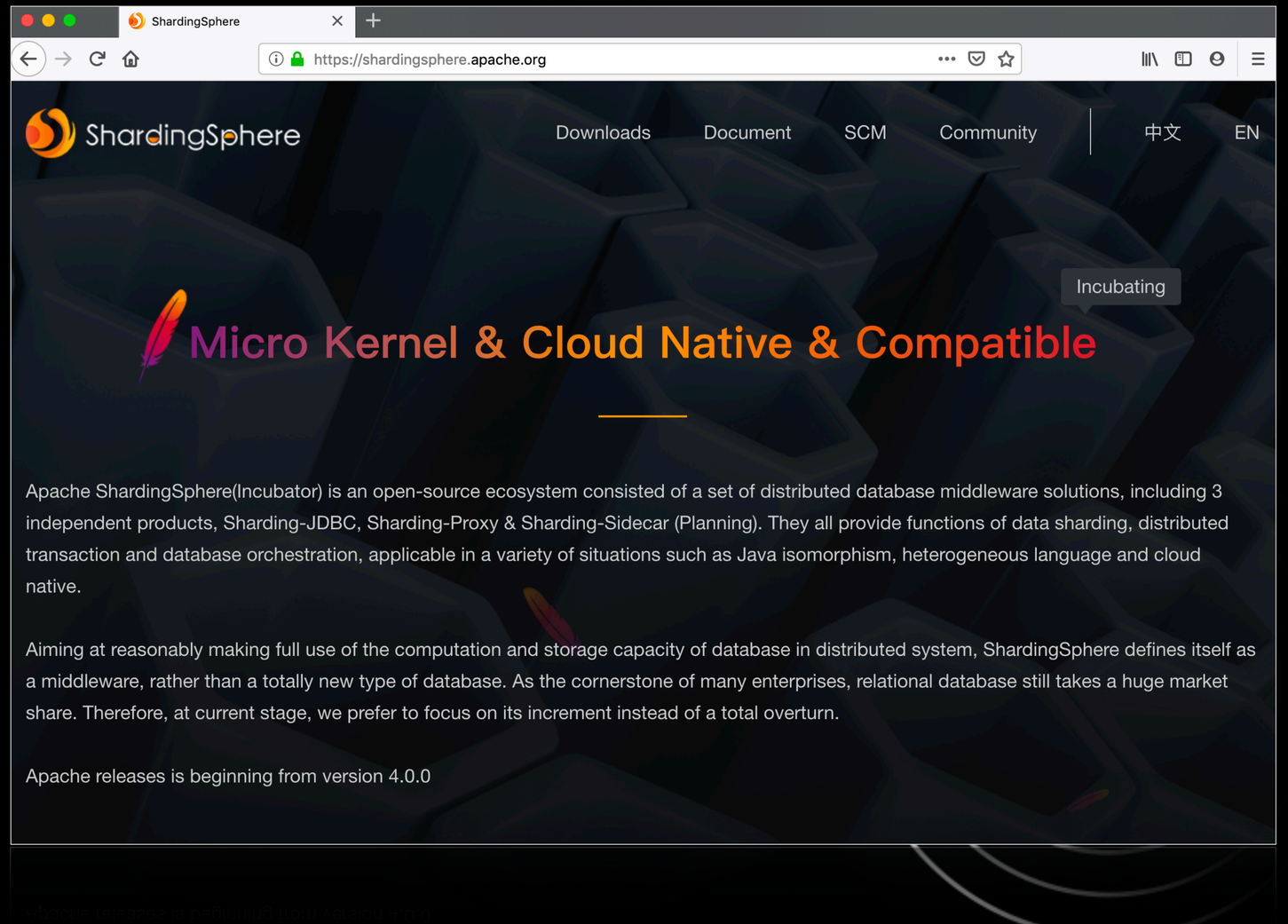
2. 샤딩 스피어가 뭐지?

샤딩스피어도 간단하게 이야기해보아요.

2. 샤딩 스피어가 뭐지?

ShardingSphere

Apache project



2. 샤딩 스피어가 뭐지?

ShardingSphere

made in China

https://cdn.pixabay.com/photo/2017/11/20/12/15/china-2965332_1280.png

2. 샤딩 스피어가 뭐지?

압도적인 트래픽, 모바일 서비스 강국

오히려 배워야할 “IT 선진국”

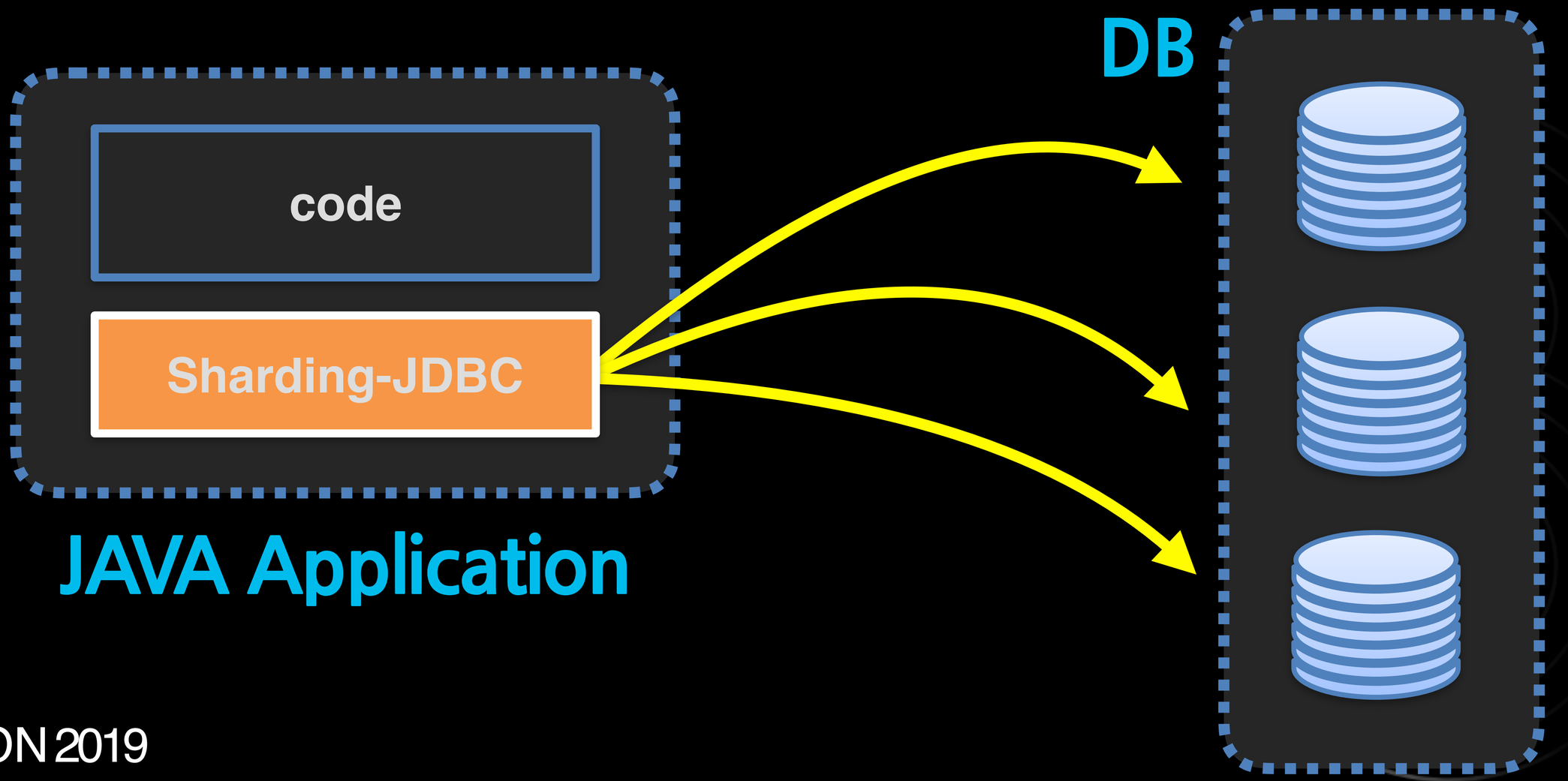
2. 샤딩 스피어가 뭐지?

ShardingSphere

https://shardingsphere.apache.org

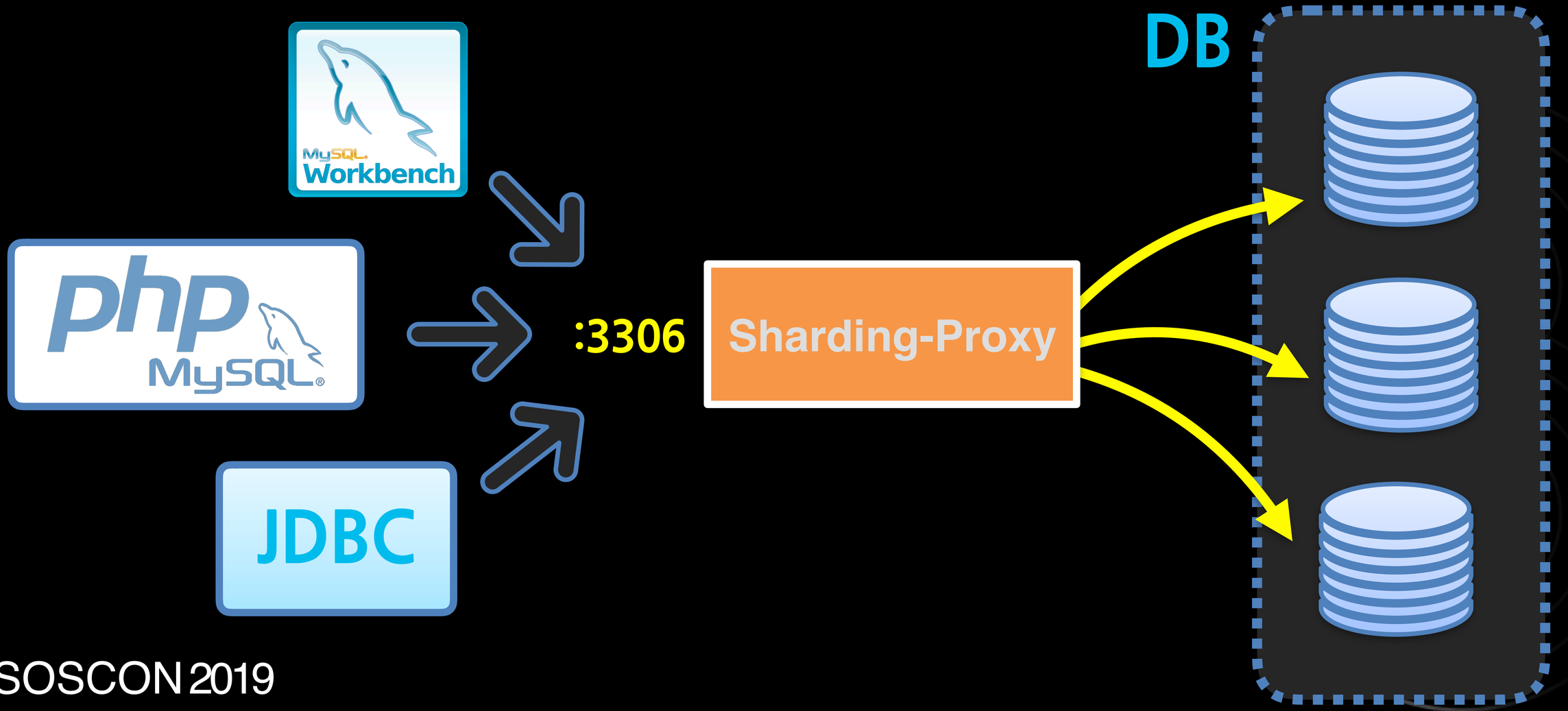
- Sharding-JDBC
- Sharding-Proxy
- Sharding-Sidecar (skip)

2. 샤딩 스피어가 뭐지?

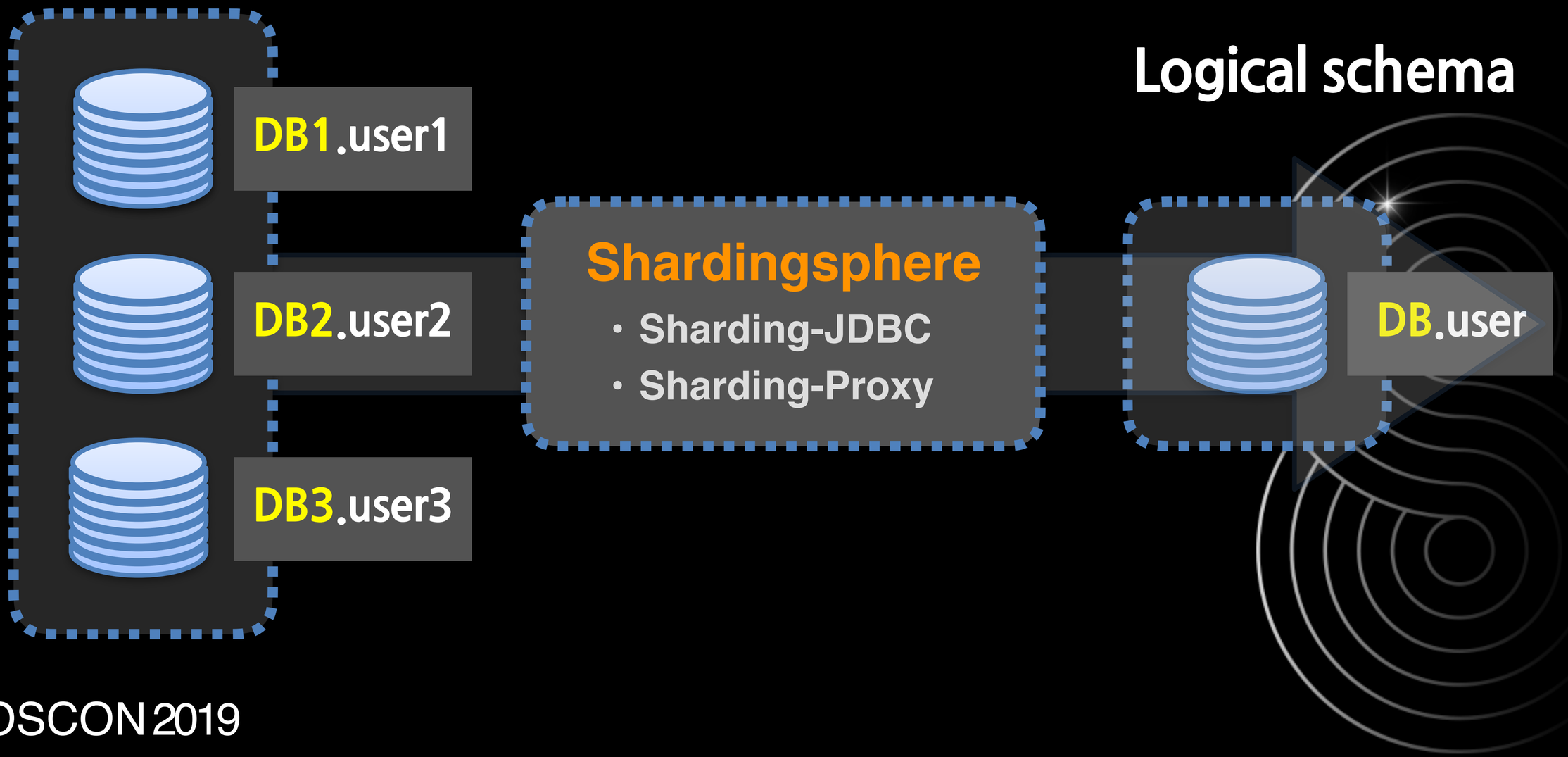


2. 샤딩 스피어가 뭐지?

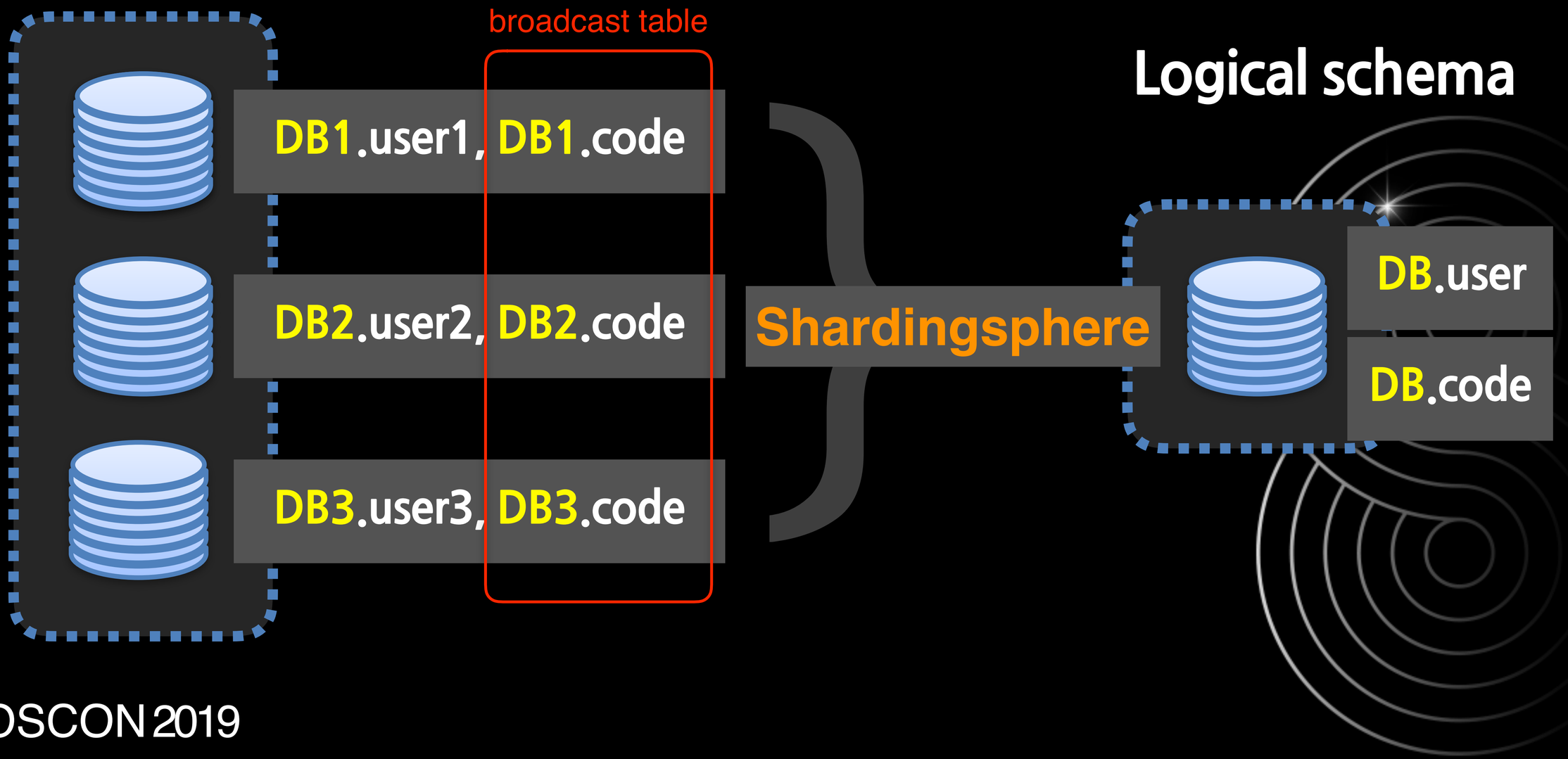
Sharding-Proxy



2. 샤딩 스피어가 뭐지?


Logical schema
DB1.user1
DB2.user2
DB3.user3
Shardingsphere
• Sharding-JDBC
• Sharding-Proxy
Logical schema
DB.user

2. 샤딩 스피어가 뭐지?

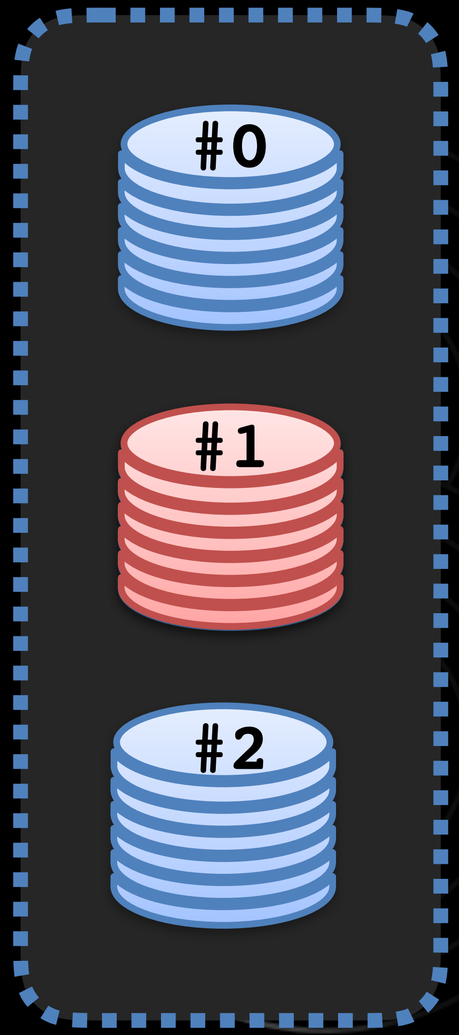

Logical schema
broadcast table
DB1.user1, DB1.code
DB2.user2, DB2.code
DB3.user3, DB3.code
Shardingsphere
Logical schema
DB.user
DB.code

2. 샤딩 스피어가 뭐지?

Logical schema

```
SELECT
   USER.*,
   CODE.TYPE_DETAIL
FROM USER
JOIN CODE on CODE.TYPE = USER.USER_TYPE
WHERE USER.ID = 19810528
```

19810528 % 3 = 1



3. 샤딩스피어 치명적인 매력

샤딩스피어에 대한 지극히 개인적인 의견

3. 샤딩스피어 치명적인 매력

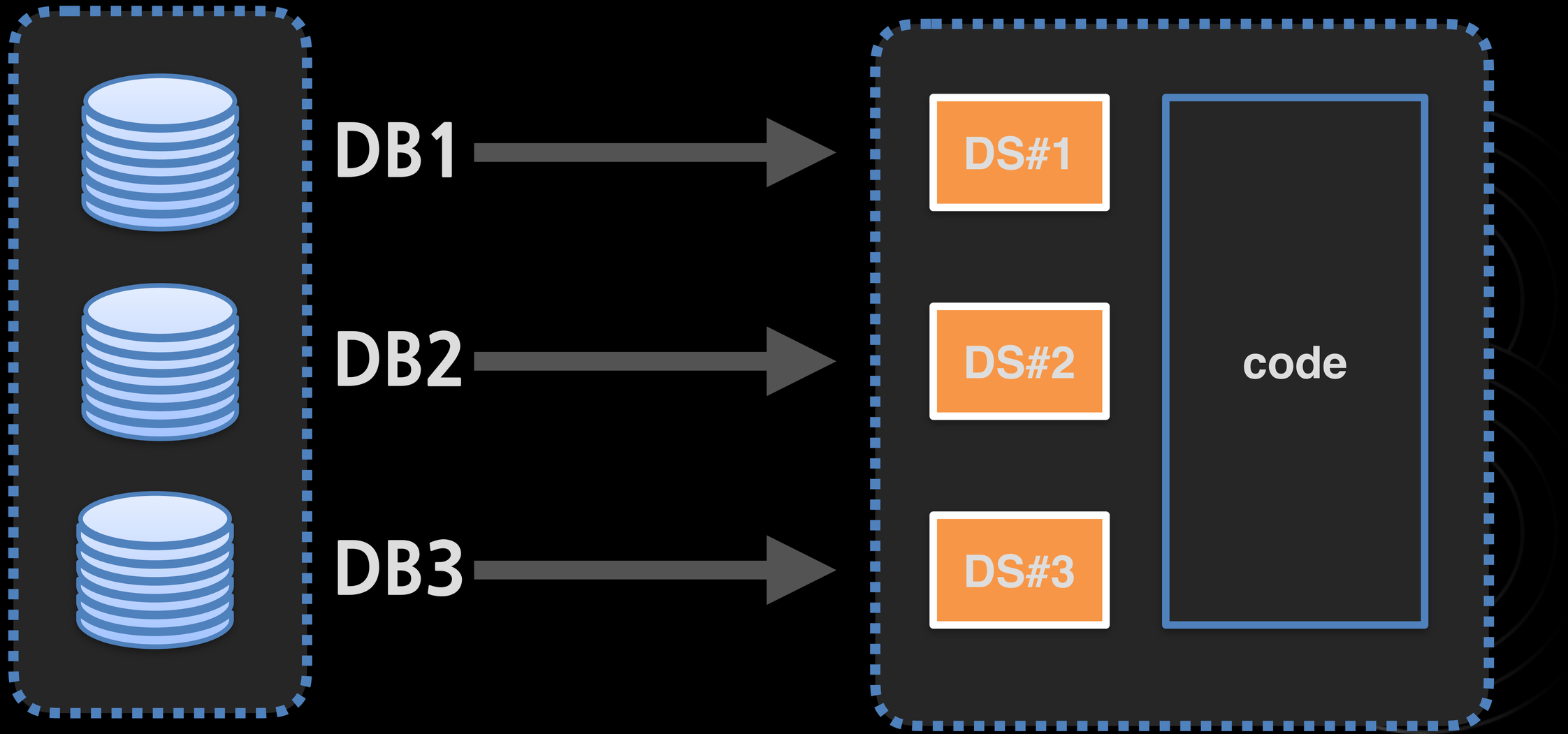
Top Features

- Single Data Source
- SQL Parsing & routing
- Merge ResultSet

https://shardingsphere.apache.org/document/current/en/features/sharding/principle/parse/
https://shardingsphere.apache.org/document/current/en/features/sharding/principle/route/
https://shardingsphere.apache.org/document/current/en/features/sharding/principle/merge/

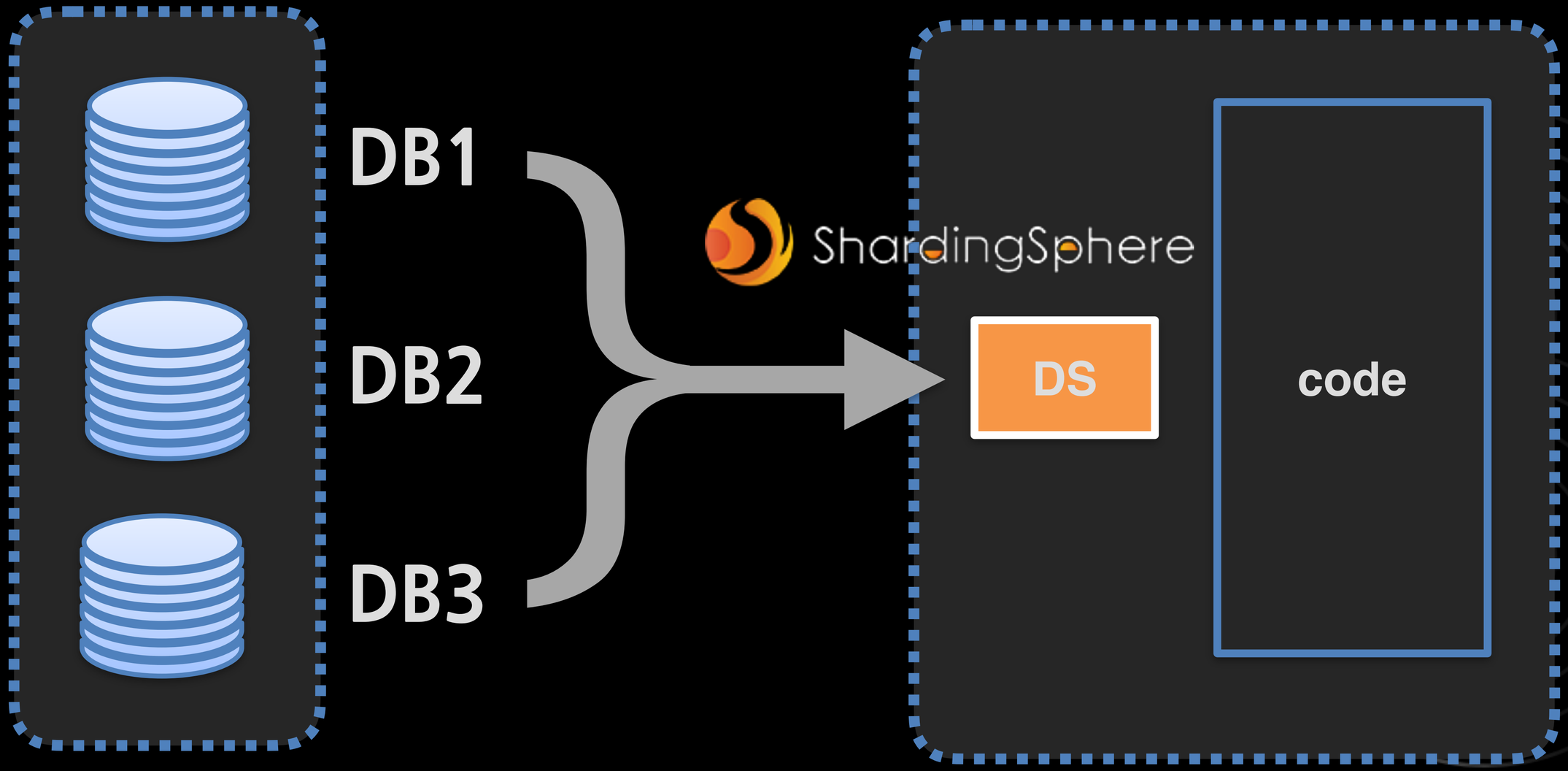
3-1. 샤딩스피어 치명적인 매력

Single Datasource



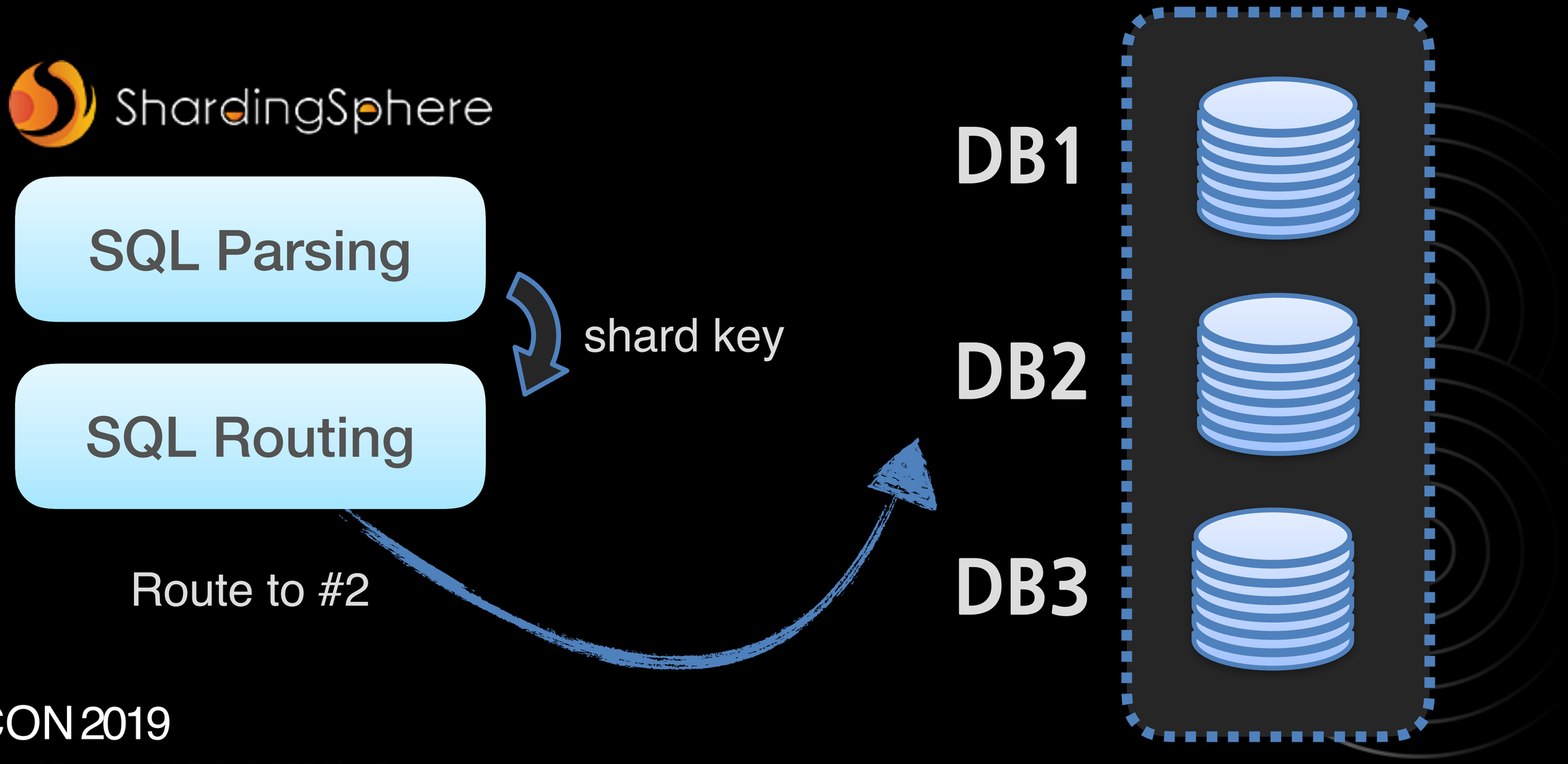
3-1. 샤딩스피어 치명적인 매력

Single Datasource



3-2. 샤딩스피어 치명적인 매력

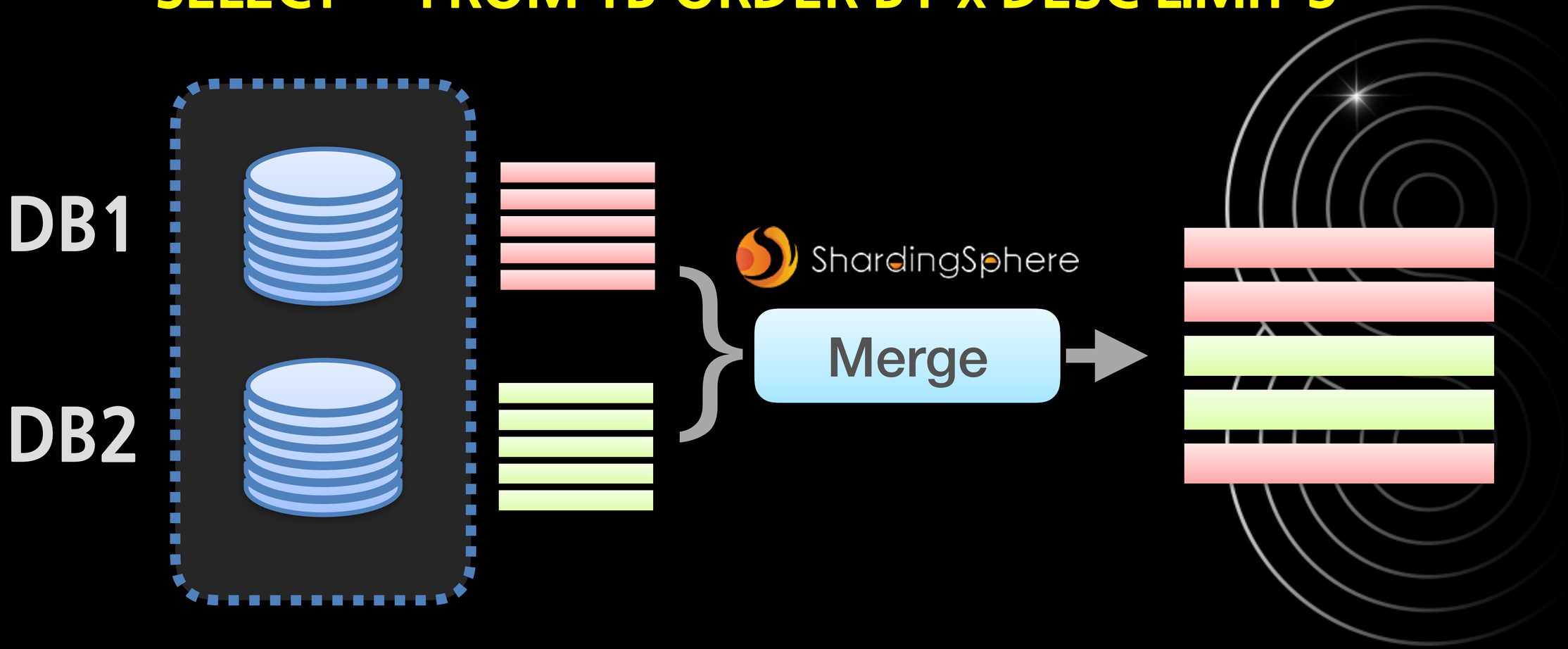
SQL parsing & Routing



3-3. 샤딩스피어 치명적인 매력

Merge ResultSet

SELECT * FROM TB ORDER BY x DESC LIMIT 5



4. 샤딩스피어 사용해보기

sharding-JDBC 시작하기

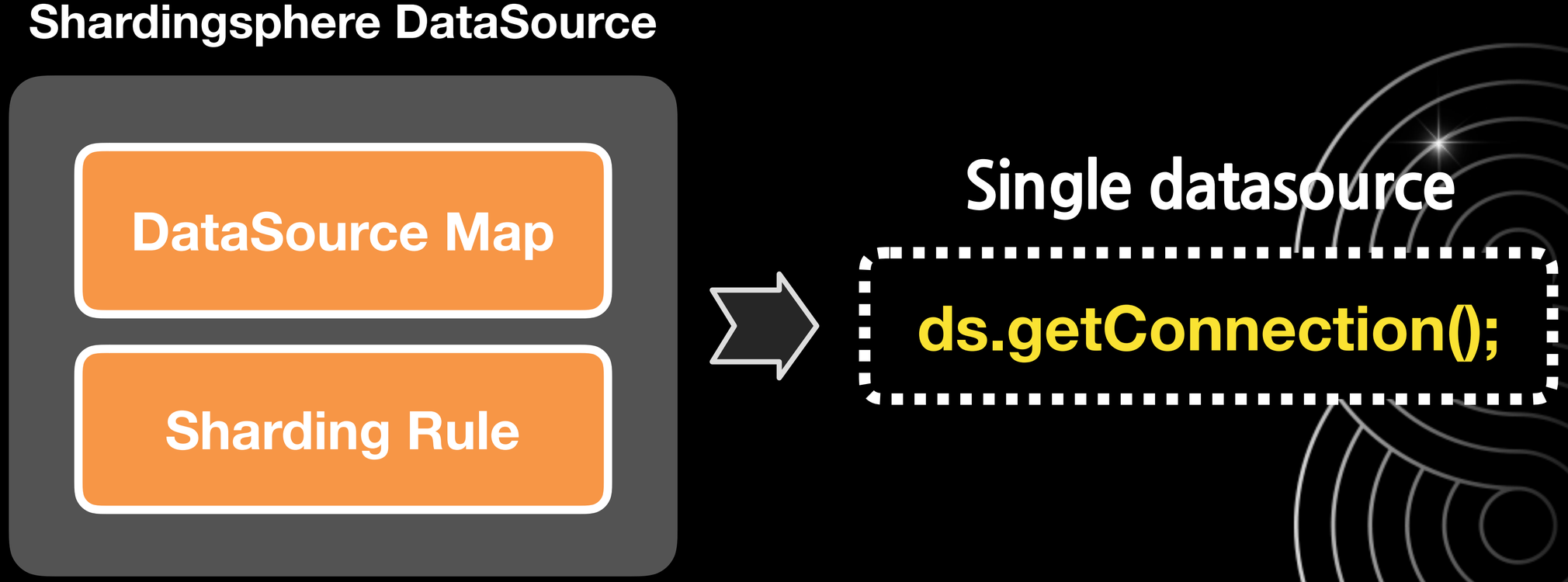
4. 샤딩스피어 사용해보기

maven

```
<dependency>
    <groupId>io.shardingsphere</groupId>
    <artifactId>sharding-jdbc-core</artifactId>
    <version>3.1.0</version>
</dependency>
```

https://shardingsphere.apache.org/document/current/en/quick-start/sharding-jdbc-quick-start/

4. 샤딩스피어 사용해보기



Java configuration

VS

Yaml configuration

4. 샤딩스피어 사용해보기

Java configuration

```
Map<String, DataSource> dataSourceMap = new HashMap<>();
BasicDataSource dataSource1 = new BasicDataSource();
BasicDataSource dataSource1 = new BasicDataSource();
dataSourceMap.put("ds0", dataSource1);
dataSourceMap.put("ds1", dataSource2);

TableRuleConfiguration tableRuleConfig = new TableRuleConfiguration();
tableRuleConfig.setLogicTable("mytable");
tableRuleConfig.setActualDataNodes("ds${0..1}.tb");
tableRuleConfig.setDatabaseShardingStrategyConfig(..);

ShardingRuleConfiguration shardingRuleConfig = new ShardingRuleConfiguration();
shardingRuleConfig.getTableRuleConfigs().add(tableRuleConfig);

DataSource dataSource = ShardingDataSourceFactory.createDataSource(dataSourceMap,
    shardingRuleConfig, new ConcurrentHashMap(), new Properties());
```

4. 샤딩스피어 사용해보기

Yaml configuration

```
String yamFile = "sharding-config.yaml";
DataSource shardingDataSource = YamlShardingDataSourceFactory.createDataSource(new
File(yamFile));
```

4. 샤딩스피어 사용해보기

Yaml configuration

sharding-config.yaml

```
dataSources:
  ds0: !!org.apache.commons.dbcp2.BasicDataSource
    url: jdbc:mysql://127.0.0.1:3306/shard01
    username: shard
    password:
  ds1: !!org.apache.commons.dbcp2.BasicDataSource
    url: jdbc:mysql://127.0.0.1:3306/shard01
    username: shard
    password:

shardingRule:
  tables:
    mytable:
      actualDataNodes: ds${0..1}.tb
      databaseStrategy:
        standard:
          shardingColumn: skey
          preciseAlgorithmClassName: net.gywn.algorithm.PreciseShardingCRC32
  bindingTables:
    - mytable
```

4. 샤딩스피어 사용해보기

PreciseShardingAlgorithm

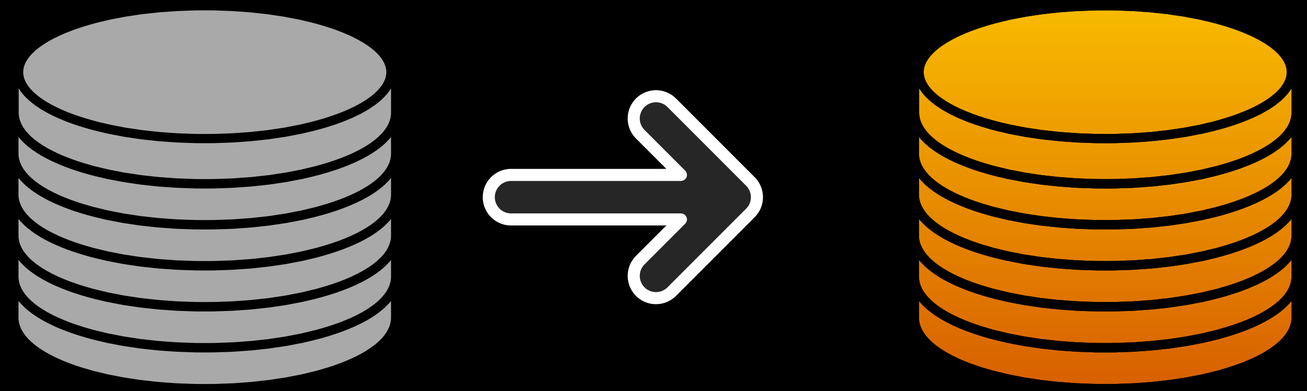
```
public class PreciseShardingCRC32 implements PreciseShardingAlgorithm<Comparable> {

    public String doSharding(Collection<String> availableTargetNames,
            PreciseShardingValue<Comparable> shardingValue) {

        ArrayList<String> list = new ArrayList<String>(availableTargetNames);
        Checksum checksum = new CRC32();
        try {
            byte[] bytes = shardingValue.getValue().toString().getBytes();
            checksum.update(bytes, 0, bytes.length);
            int seq = (int) (checksum.getValue() % list.size());
            return list.get(seq);
        } catch (Exception e) {}
        return list.get(0);

    }
}
```

5. 샤딩 스피어 활용 1탄

샤드로 데이터 찢어서 옮길 때 유용하더군요.

5. 샤딩 스피어 활용 1탄

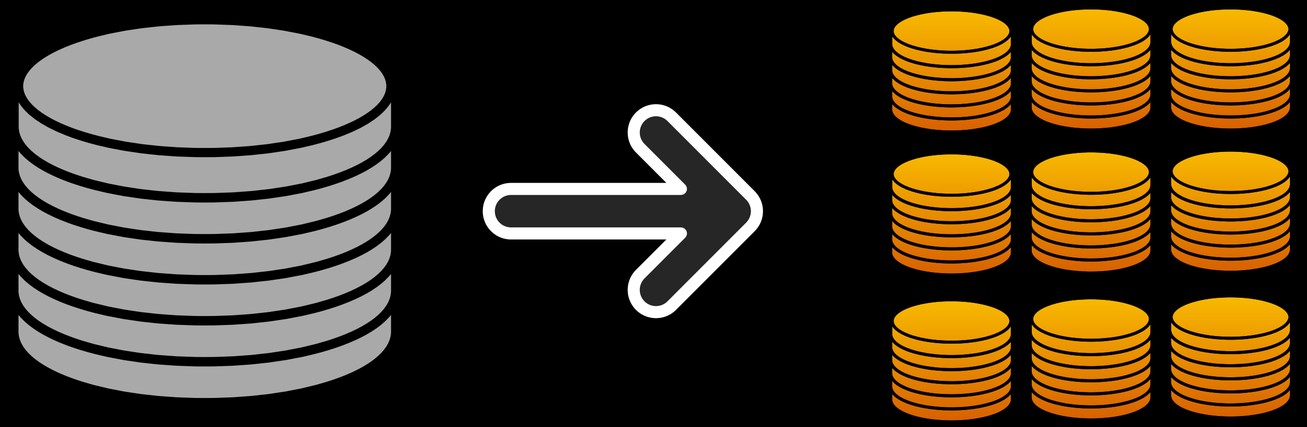
Sharding migration



Export & Import

5. 샤딩 스피어 활용 1탄

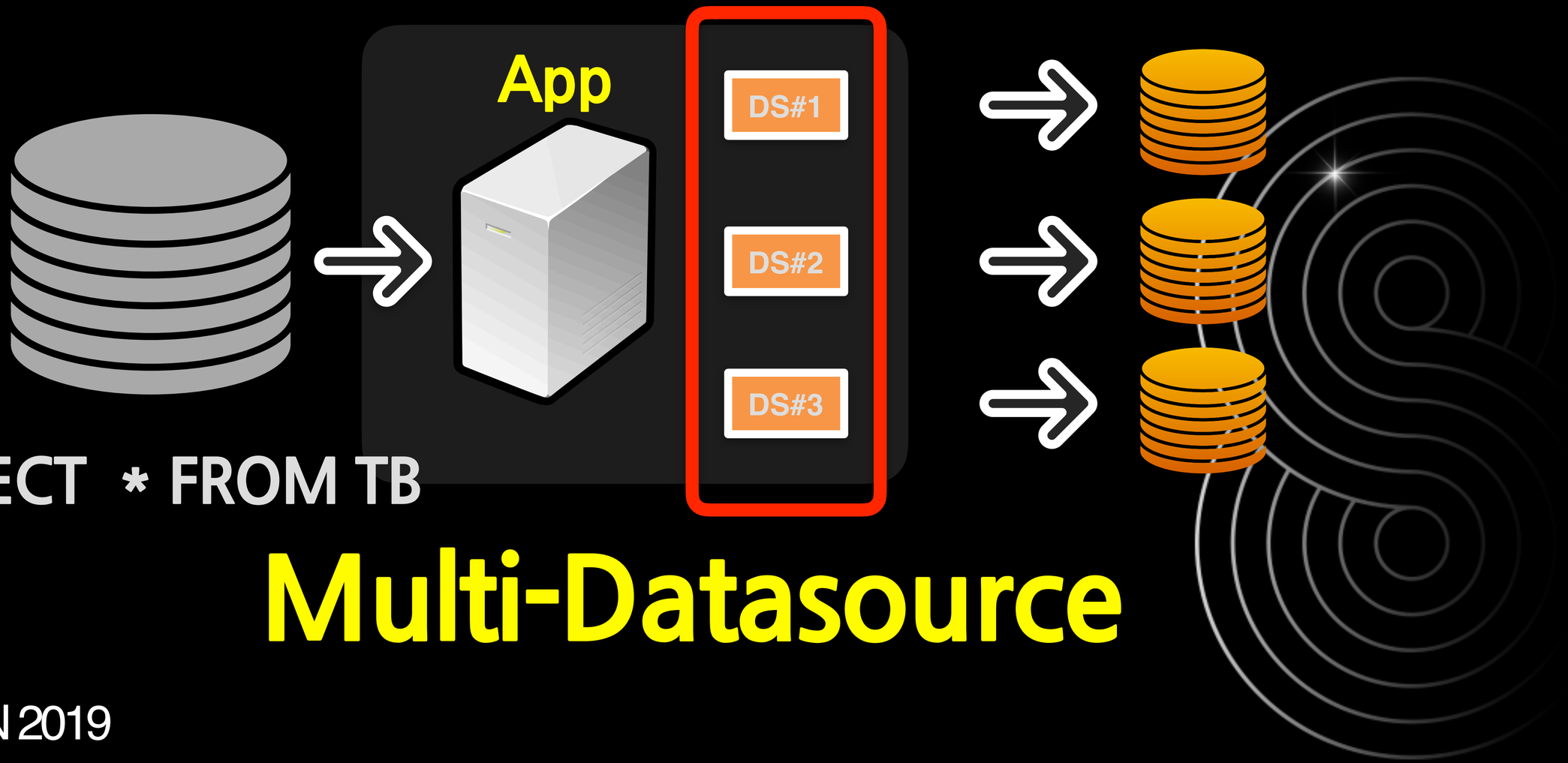
Sharding migration



??????????????

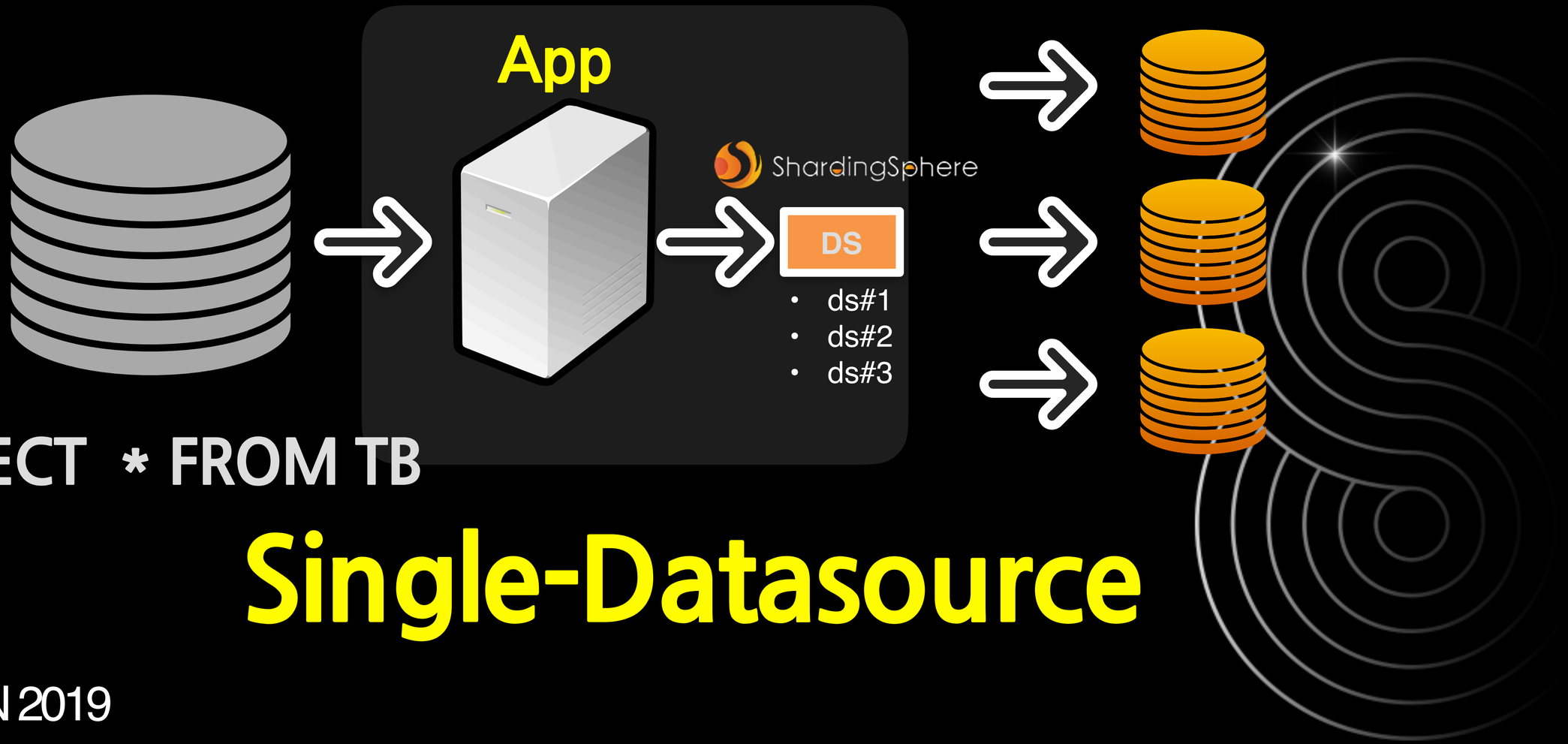
5. 샤딩 스피어 활용 1탄

Sharding migration



5. 샤딩 스피어 활용 1탄

Sharding migration



Single-Datasource

DB 하나에 쿼리 날리듯 작성하면 됨

SQL을 파싱해서 찾아낸 샤딩키 기준으로 데이터를 분산

5. 샤딩 스피어 활용 1탄

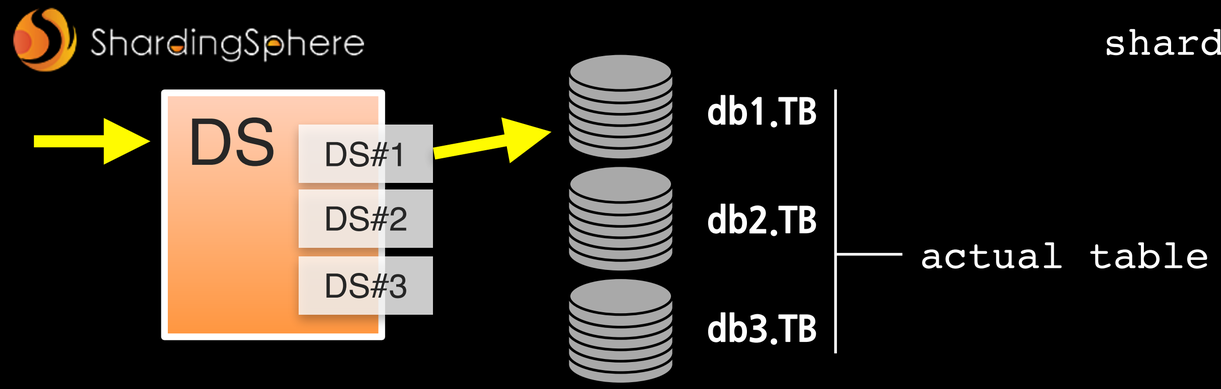
Sharding migration

shardingColumn

logic table

```
INSERT INTO TB (SKEY, INFO) VALUES (1, 'ABCDE');
```

shardingValue



30억 데이터 20시간 이관 성공

500라인 간단한 프로그램으로 가치있는 결과를 얻게 됨

6. 샤딩 스피어 활용 2탄

찢어진 데이터를 한눈에 바라보기

6. 샤딩 스피어 활용 2탄

Log Storage

개인화 서비스

App

ShardingSphere

Shard Rule

- User A
- User B

shard1

shard2

shard3

User History

서비스 관리자

고객 이력 관리

불특정 다수 데이터 접근

다양한 검색 요구 사항

6. 샤딩 스피어 활용 2탄

Log Storage

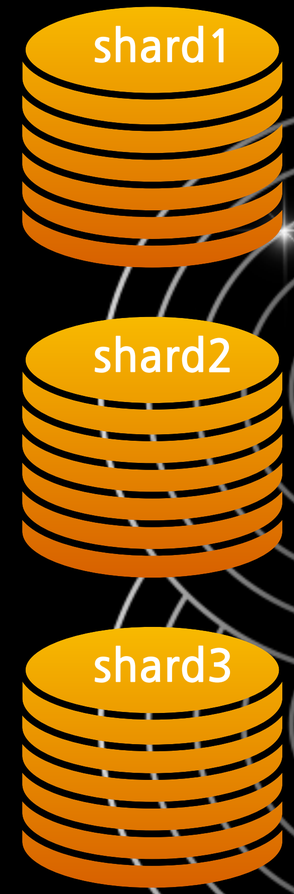
서비스 관리자

App

ShardingSphere

Shard Rule

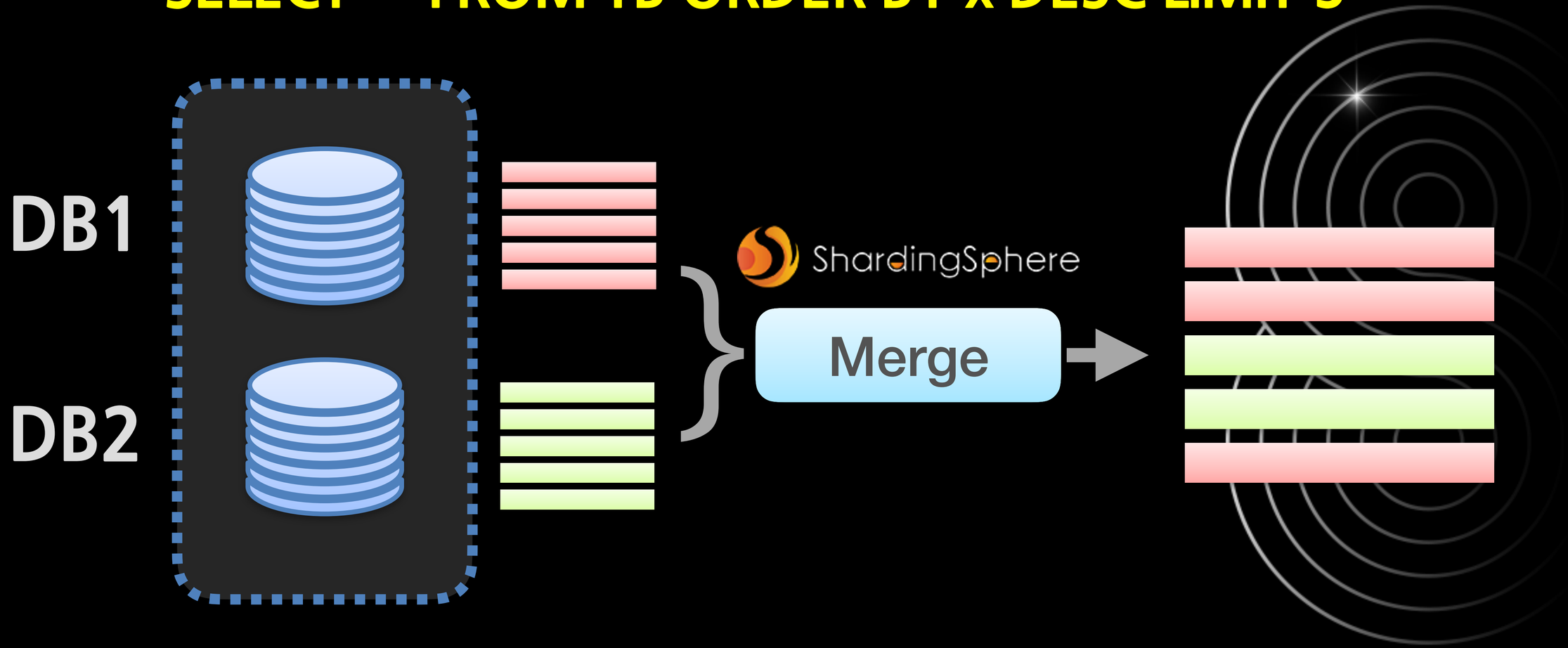
```
where email = 'chan.c@kakaobank.com'
  and visit_date = '2019-05-28'
```



User History

6. 샤딩 스피어 활용 2탄

SELECT * FROM TB ORDER BY x DESC LIMIT 5



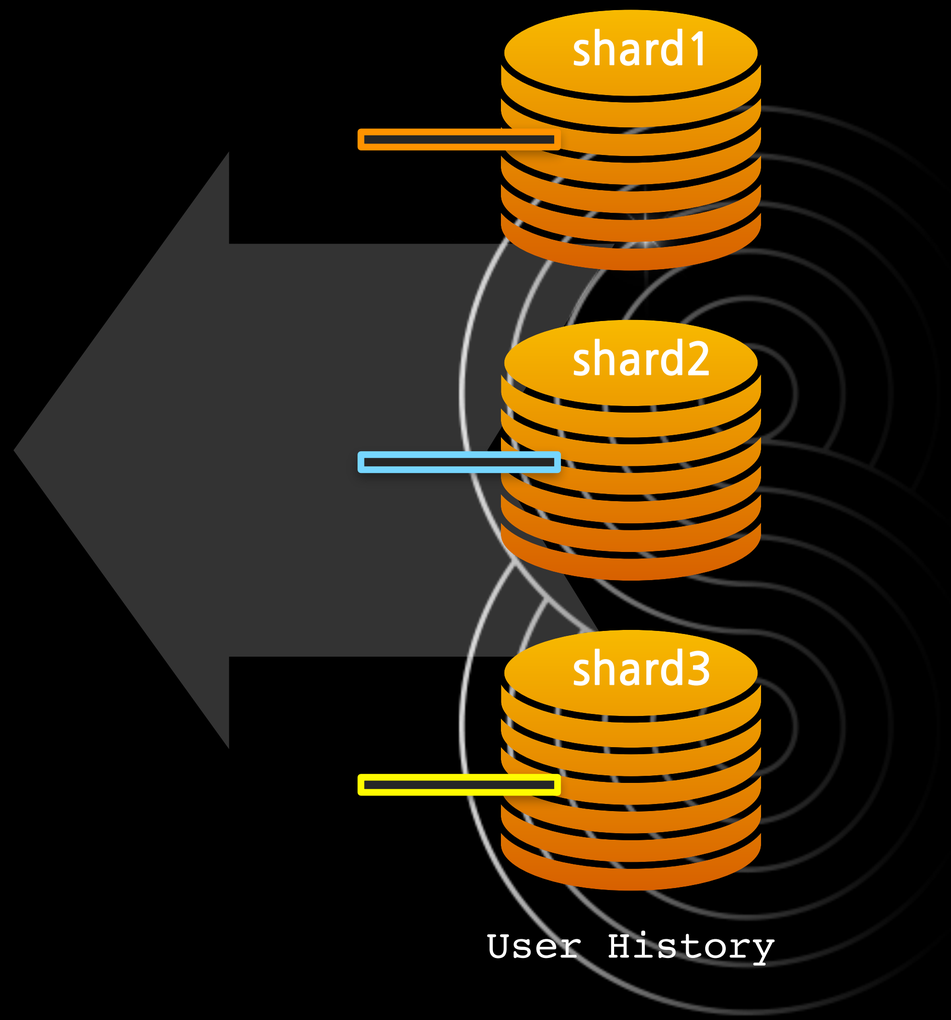
6. 샤딩 스피어 활용 2탄

Log Storage

```
select * from user_history
 where email = 'chan.c@kakaobank.com'
   and visit_date = '2019-05-28'
```

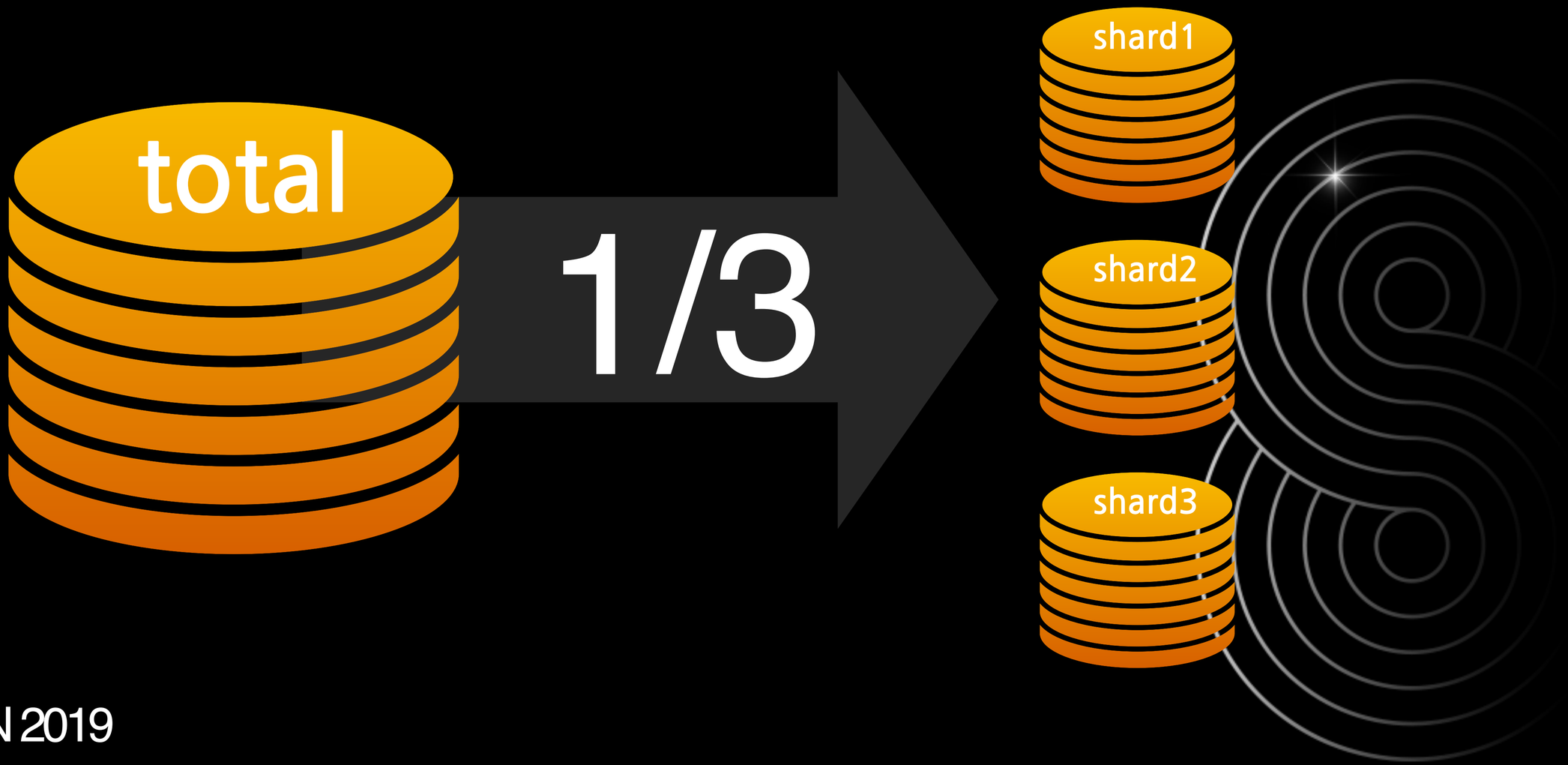
```
select * from user_history
 where nickname = 'chan.c'
   and visit_date = '2019-05-28'
```

```
select count(*) from user_history
 where visit_date = '2019-05-28'
```



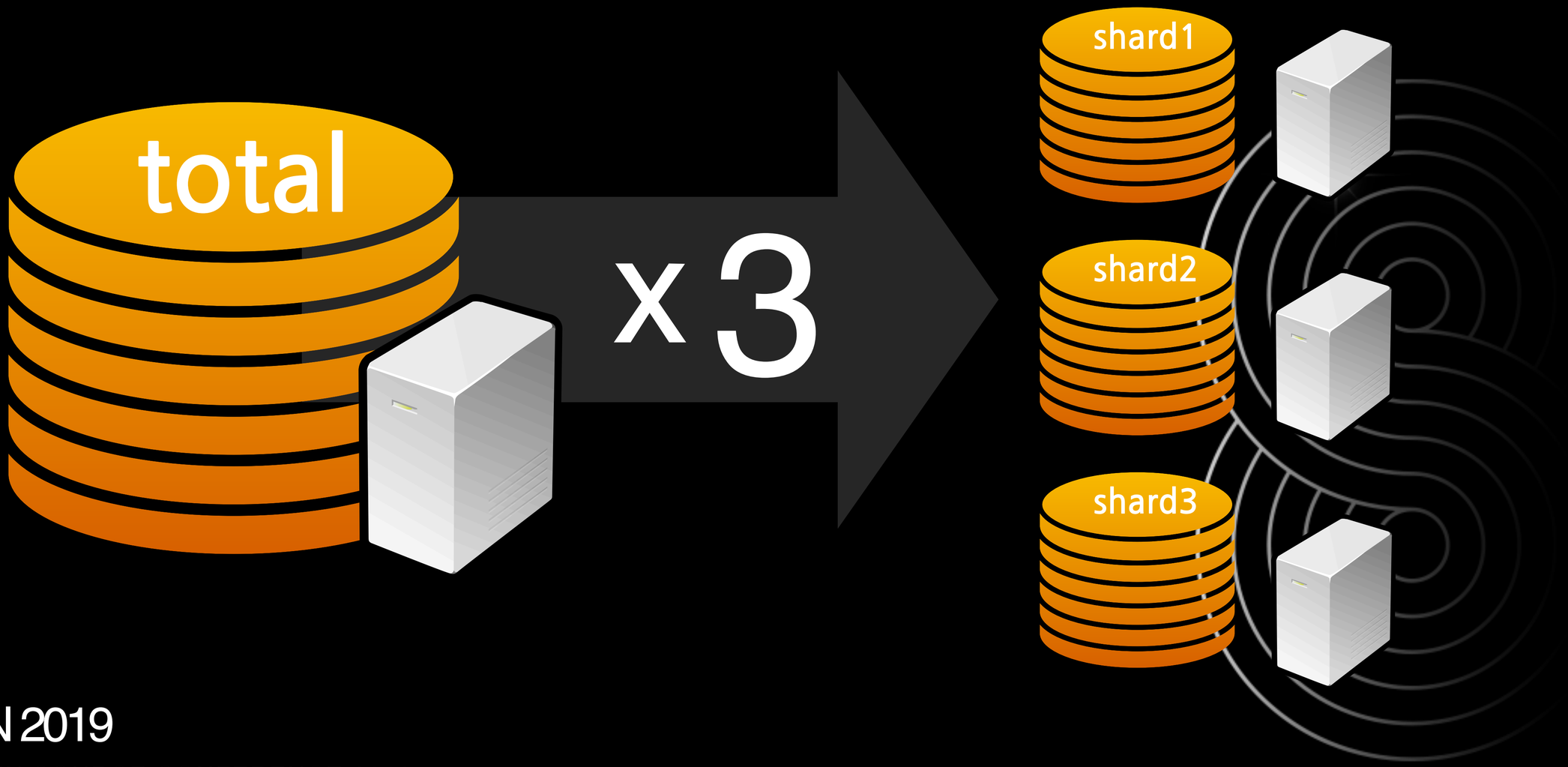
6. 샤딩 스피어 활용 2탄

Log Storage



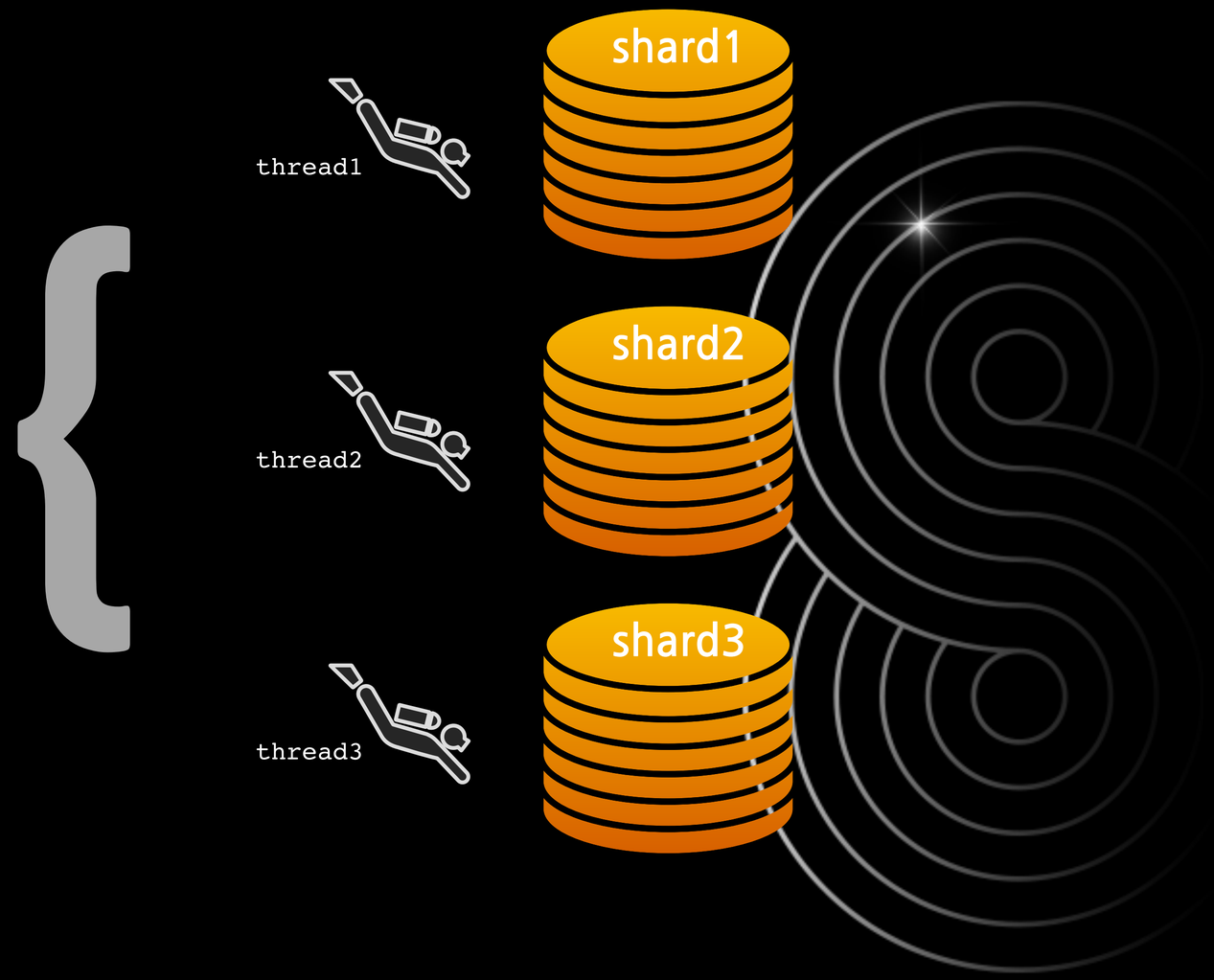
6. 샤딩 스피어 활용 2탄

Log Storage



6. 샤딩 스피어 활용 2탄


```
select count(*)
  from user_history
```



어쩌다 병렬처리

최대 9배 막연한 성능 향상 기대

6. 샤딩 스피어 활용 2탄


```
mysql> select count(*)
    -> from user_history;
+----------+
| count(*) |
+----------+
| 84950612 |
+----------+
1 row in set (2 min 9.23 sec)
```

VS

(sharding-proxy로 쿼리 테스트)

```
mysql> select count(*)
    -> from user_history;
+----------+
| count(*) |
+----------+
| 84950612 |
+----------+
1 row in set (17.13 sec)
```

7. Finish

벌써 정리시간이네요.

7. Finish

샤딩이란 무엇이었지?

데이터를 찢는 기술일 뿐입니다.

- Range Sharding
- Modulus Sharding
- 내맘대로 샤딩(Mapping)

7. Finish

샤딩스피어는 뭐였더라???

“made in China” 샤딩 라이브러리

- Sharding-JDBC, Sharding-Proxy
- 매력포인트
  - Single Data Source
  - SQL Parsing & routing
  - Merge ResultSet

7. Finish

샤딩스피어는 어떻게 써볼까?

서비스 특성에 따라 다양하게 활용해볼 수 있겠음

- 데이터 이관에서, 간단한 코드로 최대의 행복을 얻음
- 대량 데이터 관리 시 **“어쩌다 병렬처리”** 득

7. Finish

개인 소감

세상은 넓고 천재들은 많다.

- 오픈 소스를 활용해보니, 원하는 바를 금방 이루었다.
- 오픈 소스를 써보려고 해보니, 제대로 쓰기 위해서 많은 학습이 필요하다.
- 오픈 소스로 얻은 작은 경험을 공유하는 것이 의미 있겠다.

THANK YOU

SOSCON 2019

SAMSUNG OPEN SOURCE CONFERENCE 2019