MySQL Orchestrator 기반의 고가용성 DB 클러스터 관리 시스템

목차

Part 1

팀원 소개

MySQL 담당

김혜수

Orchestrator 담당

신영민

Proxy 담당

신지혜

프로젝트 개요

프로젝트 목적	Docker 기반 MySQL 클러스터를 구축하여 Orchestrator로 자동 Failover를 구현하고, ProxySQL을 통해 읽기/쓰기 분산과 고가용성 확보
대상 애플리케이션	MySQL을 사용하는 Java 기반의 웹 애플리케이션
주요 기술 스택	Ubuntu 20.04, Docker, MySQL, Orchestrator, ProxySQL

시스템 구성 시나리오

Docker

컨테이너 기반 실행 환경 준비 및 서버 간 통신 가능하도록 설정

MySQL Master/Slave

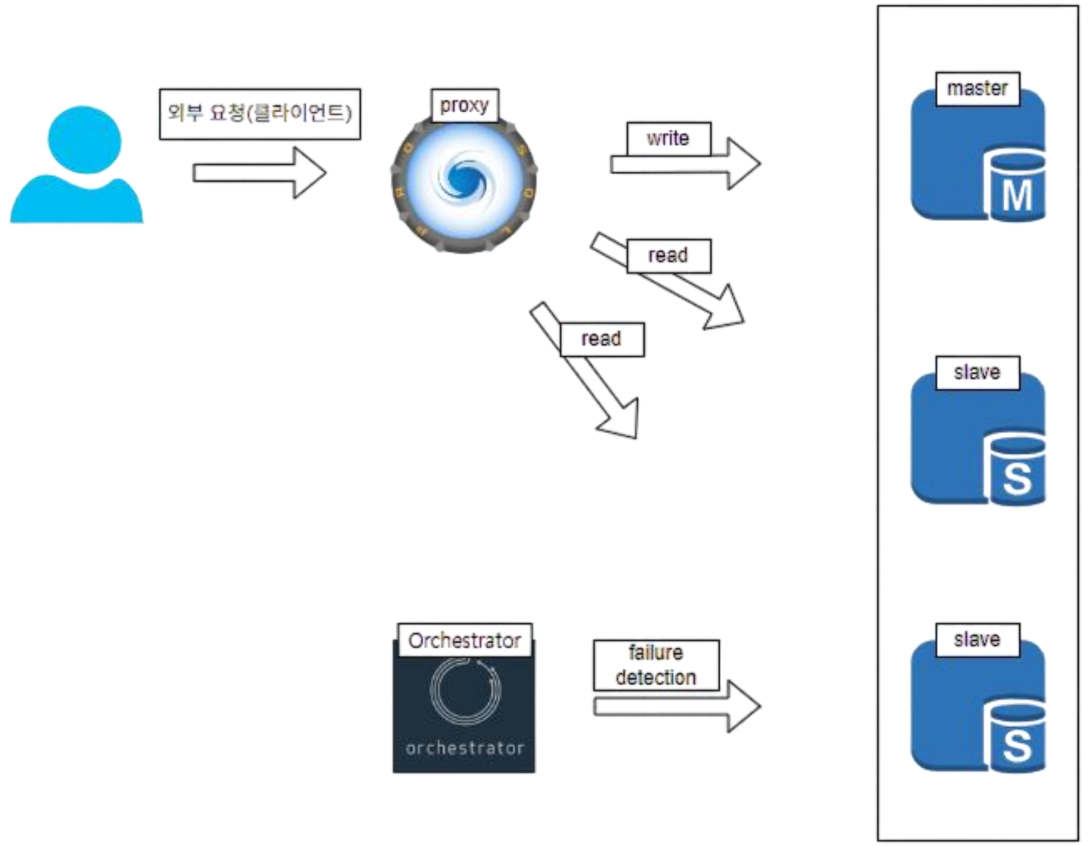
데이터 쓰기 중심 노드 구성, GTID 및 binlog 설정,
Master 복제 설정 (CHANGE MASTER TO), 읽기 부하 분산

Orchestrator

Master 장애 감지 및 자동 Failover 수행, 상태 시각화 가능

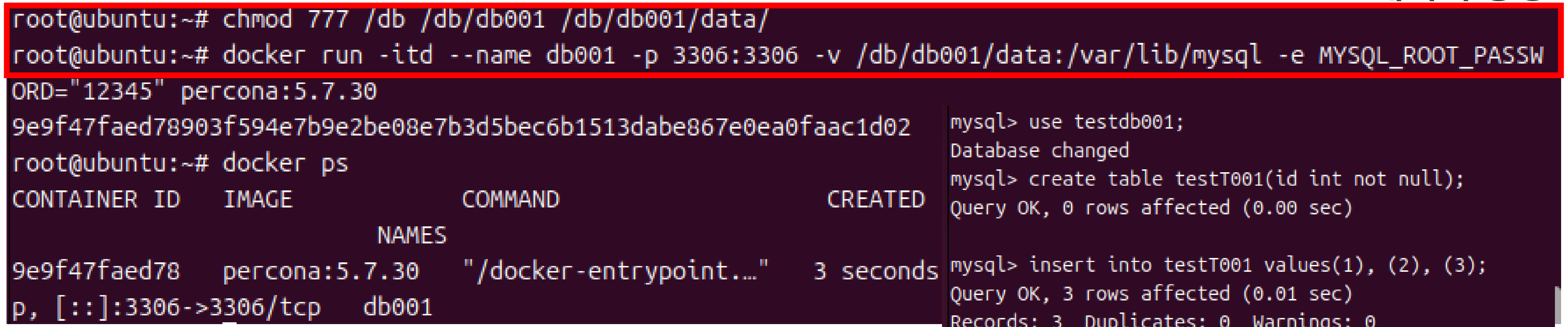
ProxySQL

클라이언트 요청을 ProxySQL에서 라우팅, 애플리케이션은 단일 접점 사용



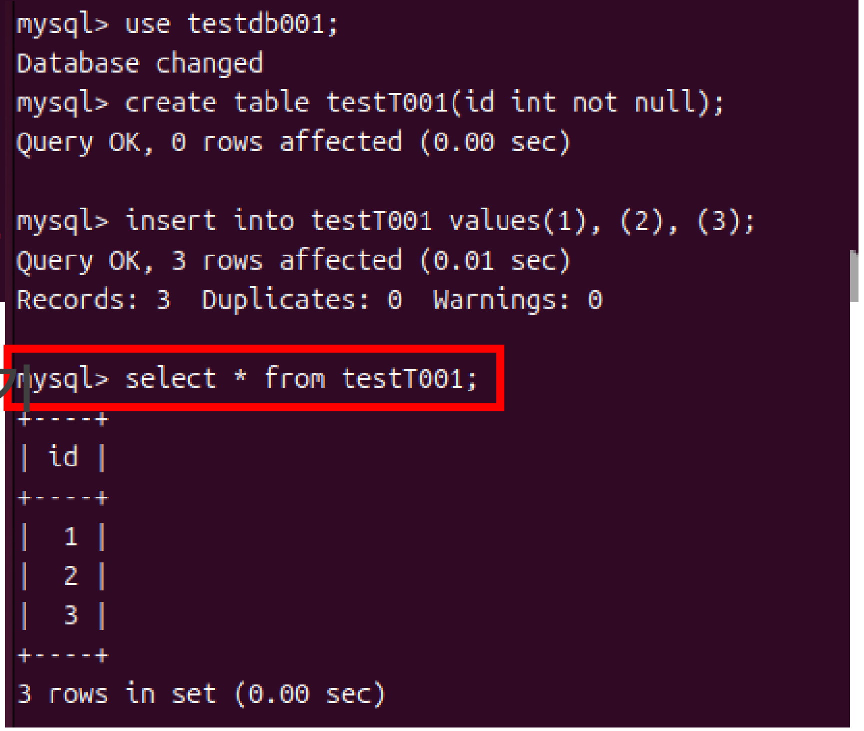
컨테이너 만들기

디렉터리 생성 및 권한 부여
컨테이너 생성



```
root@ubuntu:~# chmod 777 /db /db/db001 /db/db001/data/
root@ubuntu:~# docker run -itd --name db001 -p 3306:3306 -v /db/db001/data:/var/lib/mysql -e MYSQL_ROOT_PASSW
ORD="12345" percona:5.7.30
9e9f47faed78903f594e7b9e2be08e7b3d5bec6b1513dabe867e0ea0faac1d02
root@ubuntu:~# docker ps
CONTAINER ID   IMAGE            COMMAND                    CREATED
                         NAMES
9e9f47faed78   percona:5.7.30   "/docker-entrypoint.…"   3 seconds
p, [::]:3306->3306/tcp   db001
```

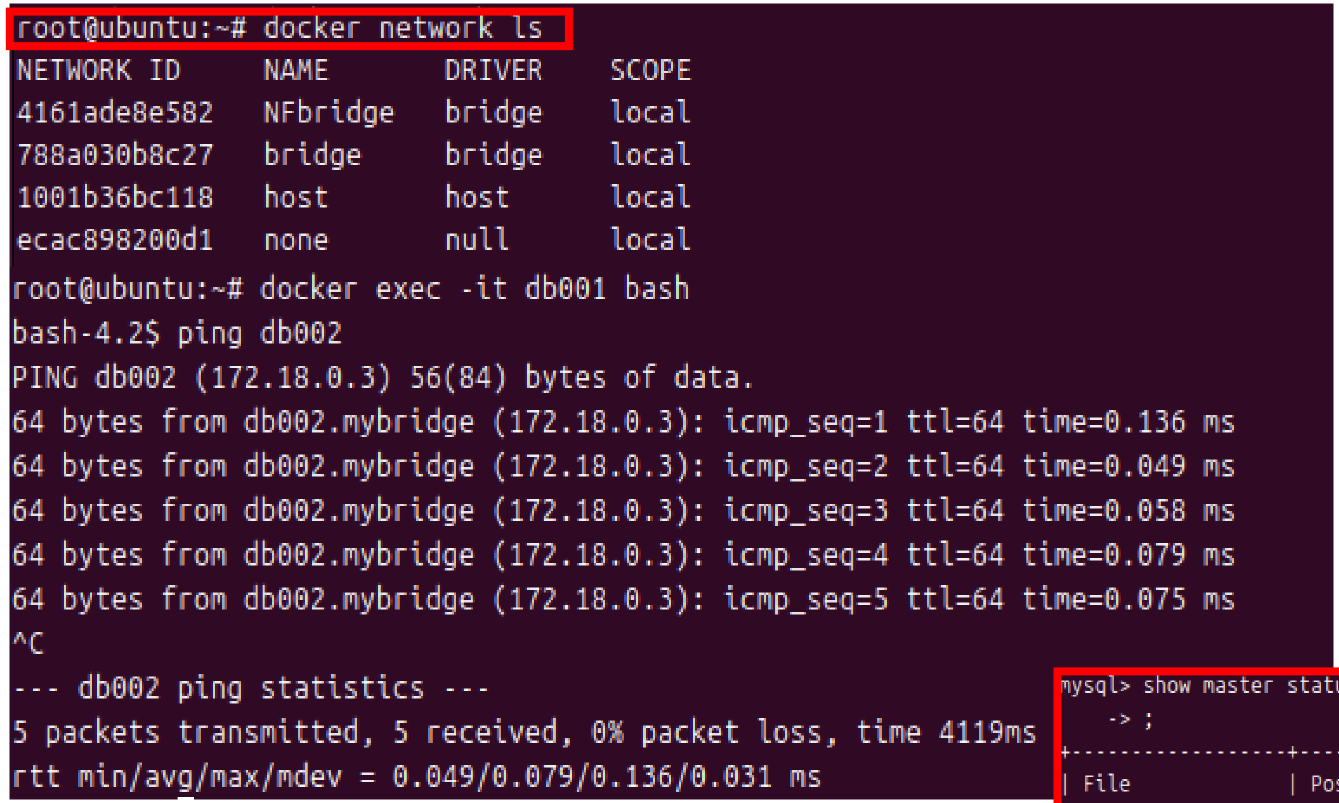
컨테이너 접속 후 MySQL Test 테이블 만들기



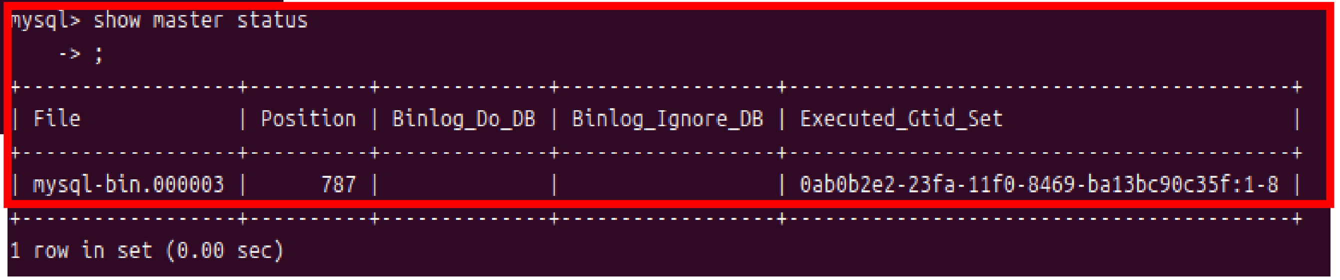
```
mysql> use testdb001;
Database changed
mysql> create table testT001(id int not null);
Query OK, 0 rows affected (0.00 sec)

mysql> insert into testT001 values(1), (2), (3);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from testT001;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
+----+
3 rows in set (0.00 sec)
```

Docker Network 설정 후 컨테이너 연결 및 접속

Docker network 설정 (Nfbridge)



```
root@ubuntu:~# docker network ls
NETWORK ID     NAME       DRIVER    SCOPE
4161ade8e582   NFbridge   bridge    local
788a030b8c27   bridge     bridge    local
1001b36bc118   host       host      local
ecac898200d1   none       null      local
root@ubuntu:~# docker exec -it db001 bash
bash-4.2$ ping db002
PING db002 (172.18.0.3) 56(84) bytes of data.
64 bytes from db002.mybridge (172.18.0.3): icmp_seq=1 ttl=64 time=0.136 ms
64 bytes from db002.mybridge (172.18.0.3): icmp_seq=2 ttl=64 time=0.049 ms
64 bytes from db002.mybridge (172.18.0.3): icmp_seq=3 ttl=64 time=0.058 ms
64 bytes from db002.mybridge (172.18.0.3): icmp_seq=4 ttl=64 time=0.079 ms
64 bytes from db002.mybridge (172.18.0.3): icmp_seq=5 ttl=64 time=0.075 ms
^C
--- db002 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4119ms
rtt min/avg/max/mdev = 0.049/0.079/0.136/0.031 ms
```

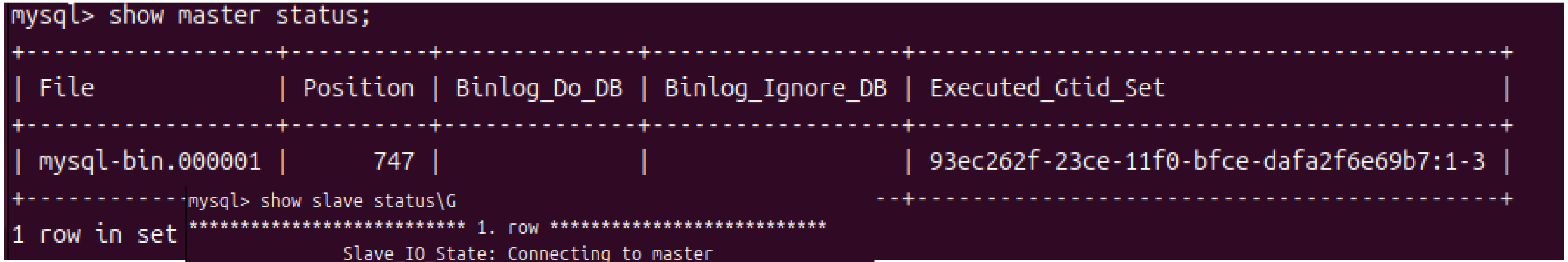


```
mysql> show master status
    -> ;
+------------------+----------+--------------+------------------+------------------------------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set                        |
+------------------+----------+--------------+------------------+------------------------------------------+
| mysql-bin.000003 |      787 |              |                  | 0ab0b2e2-23fa-11f0-8469-ba13bc90c35f:1-8 |
+------------------+----------+--------------+------------------+------------------------------------------+
1 row in set (0.00 sec)
```

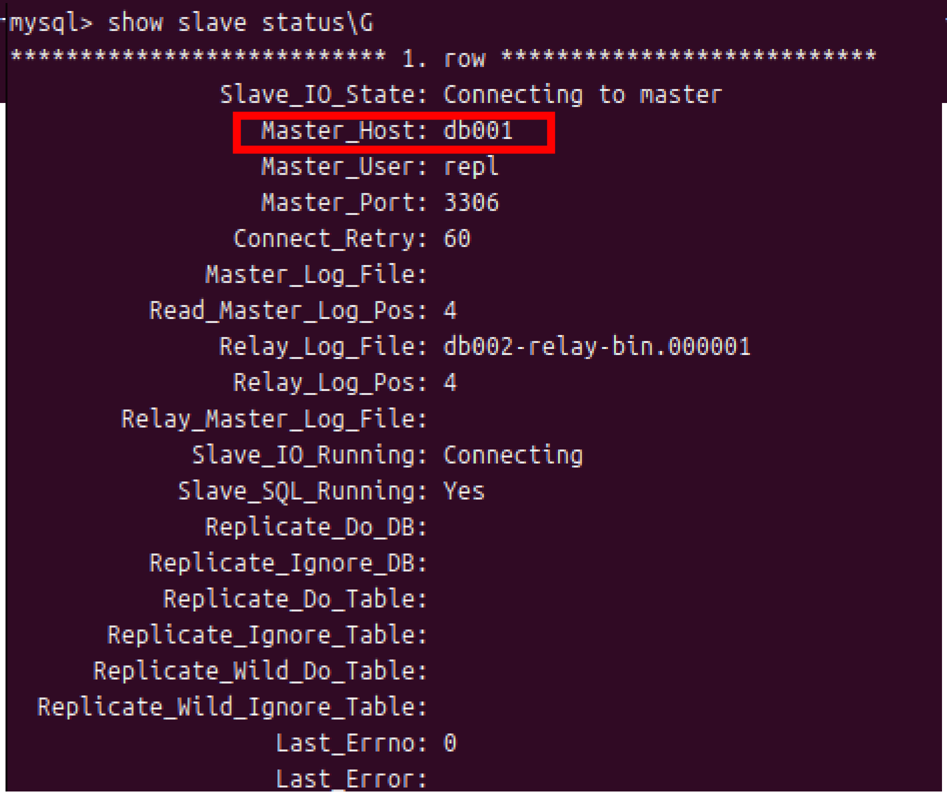
db001 컨테이너 접속 후 master 설정

Master 설정

db001을 Master로 설정



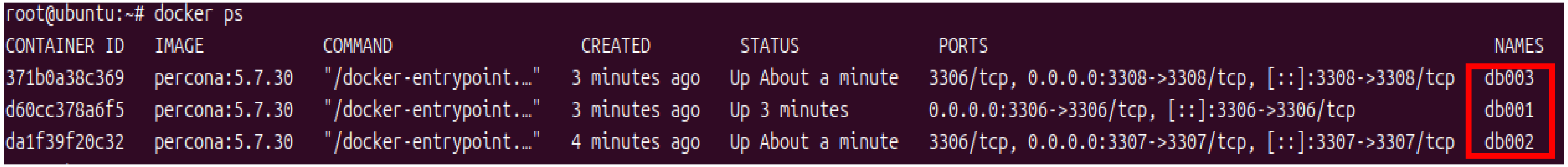
```
mysql> show master status;
+------------------+----------+--------------+------------------+------------------------------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set                        |
+------------------+----------+--------------+------------------+------------------------------------------+
| mysql-bin.000001 |      747 |              |                  | 93ec262f-23ce-11f0-bfce-dafa2f6e69b7:1-3 |
+-----------                                                   --+------------------------------------------+
1 row in set
```



```
mysql> show slave status\G
*************************** 1. row ***************************
               Slave_IO_State: Connecting to master
                  Master_Host: db001
                  Master_User: repl
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File:
          Read_Master_Log_Pos: 4
               Relay_Log_File: db002-relay-bin.000001
                Relay_Log_Pos: 4
        Relay_Master_Log_File:
             Slave_IO_Running: Connecting
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
```

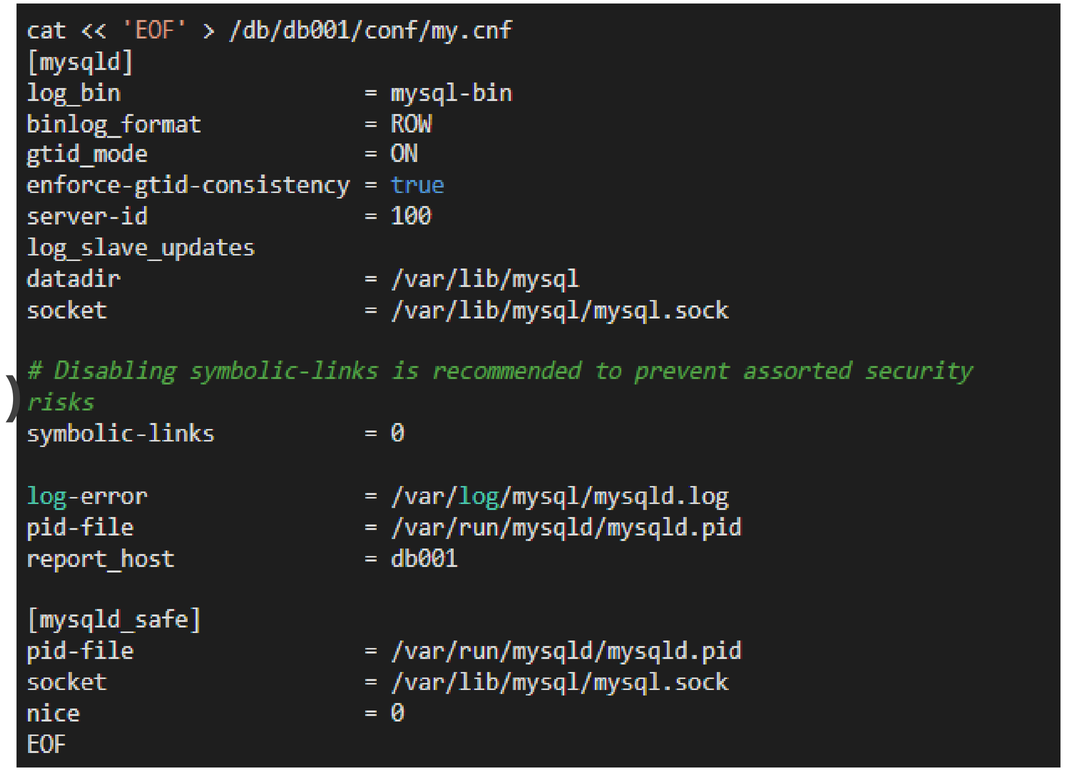
Slave(db002/db003) 에서 확인
Master_Host : db001

권한 설정



```
root@ubuntu:~# docker ps
CONTAINER ID   IMAGE           COMMAND                  CREATED         STATUS              PORTS                                               NAMES
371b0a38c369   percona:5.7.30  "/docker-entrypoint.…"   3 minutes ago   Up About a minute   3306/tcp, 0.0.0.0:3308->3308/tcp, [::]:3308->3308/tcp   db003
d60cc378a6f5   percona:5.7.30  "/docker-entrypoint.…"   3 minutes ago   Up 3 minutes        0.0.0.0:3306->3306/tcp, [::]:3306->3306/tcp             db001
da1f39f20c32   percona:5.7.30  "/docker-entrypoint.…"   4 minutes ago   Up About a minute   3306/tcp, 0.0.0.0:3307->3307/tcp, [::]:3307->3307/tcp   db002
```

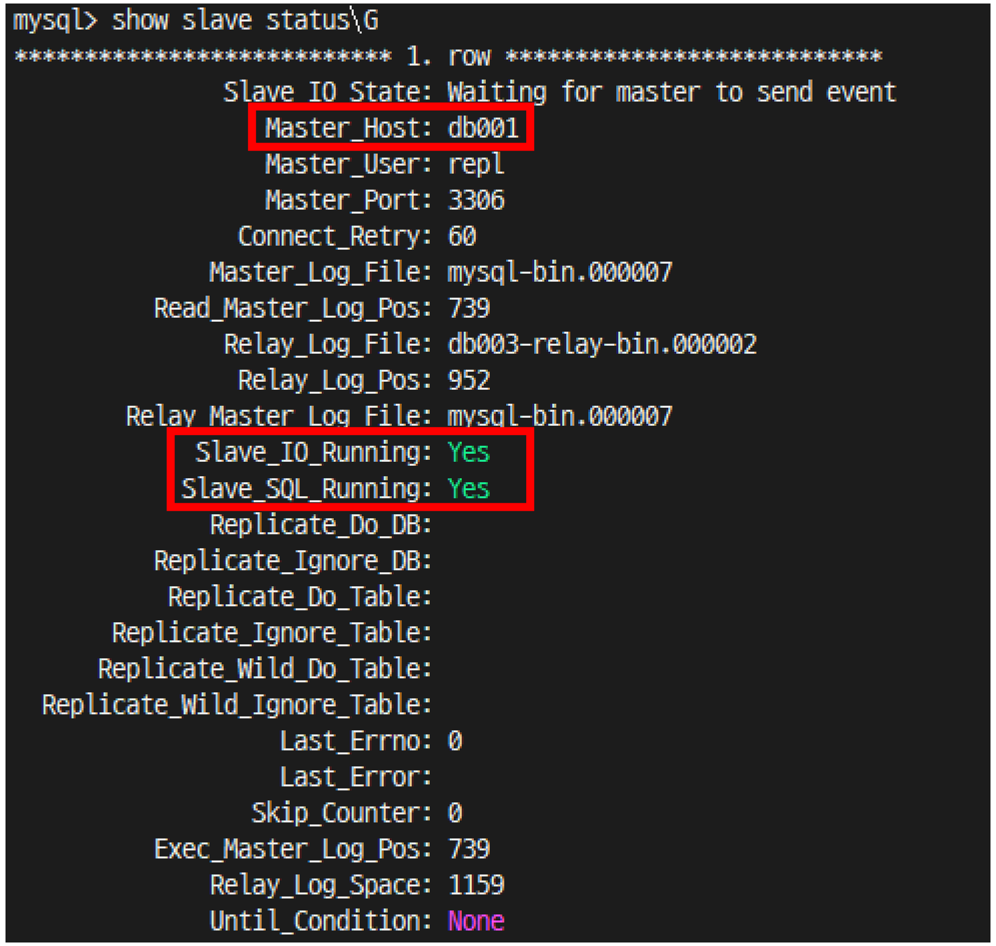
db001 / db002 / db003 컨테이너 생성 확인
-Hostname: db001 / db002 / db003
-Container name: db001 / db002 / db003
-Port : 3306 / 3307 / 3308
-MySQL 데이터 파일 저장 위치를 호스트 /db/db001-003/data로 연결
-Docker image: (percona MySQL 5.7.30 버전)



```
cat << 'EOF' > /db/db001/conf/my.cnf
[mysqld]
log_bin                    = mysql-bin
binlog_format              = ROW
gtid_mode                  = ON
enforce-gtid-consistency   = true
server-id                  = 100
log_slave_updates
datadir                    = /var/lib/mysql
socket                     = /var/lib/mysql/mysql.sock

# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links             = 0

log-error                  = /var/log/mysql/mysqld.log
pid-file                   = /var/run/mysqld/mysqld.pid
report_host                = db001

[mysqld_safe]
pid-file                   = /var/run/mysqld/mysqld.pid
socket                     = /var/lib/mysql/mysql.sock
nice                       = 0
EOF
```

설정파일 만들기> 권한부여

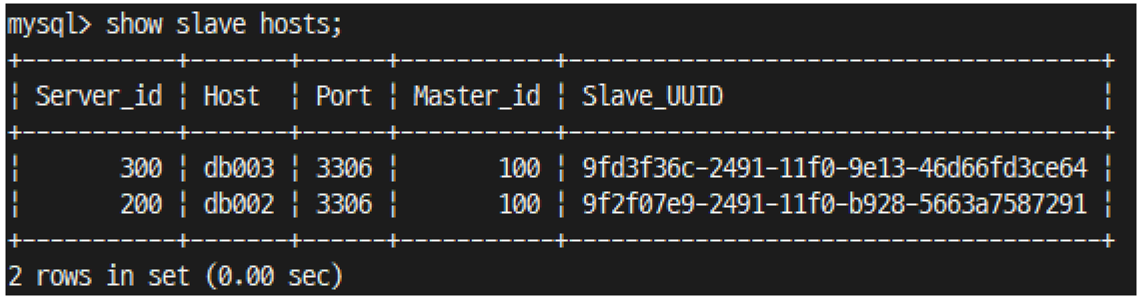
Slave 설정 및 연결 확인



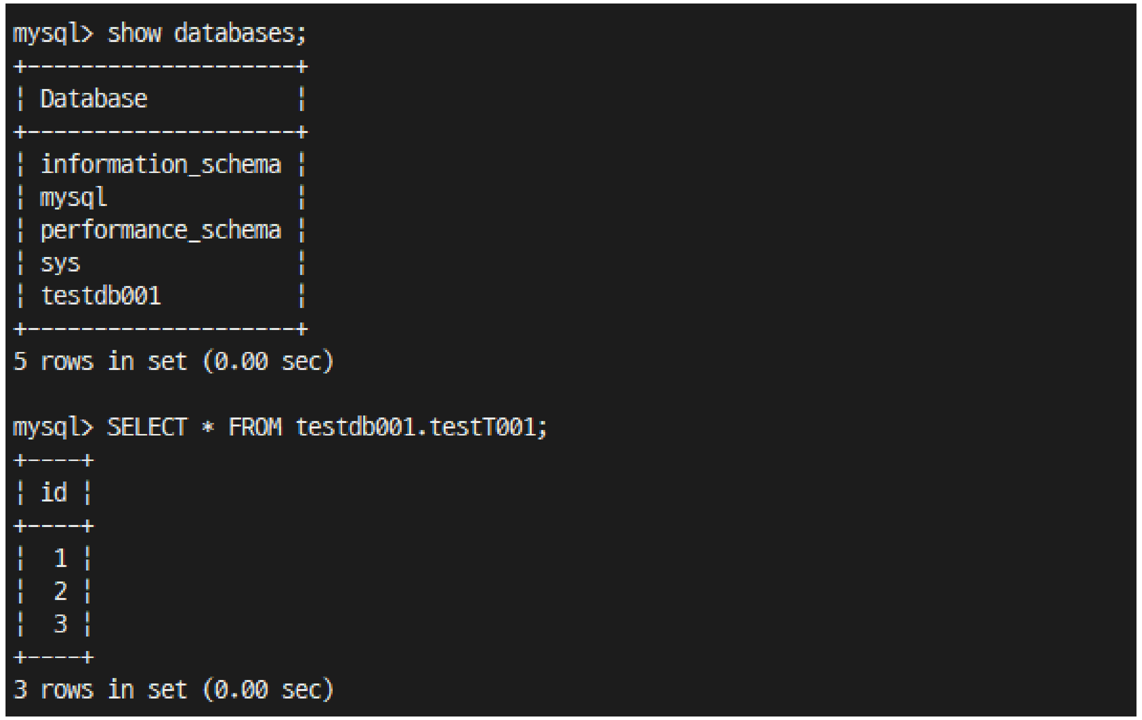
```
mysql> show slave status\G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: db001
                  Master_User: repl
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000007
          Read_Master_Log_Pos: 739
               Relay_Log_File: db003-relay-bin.000002
                Relay_Log_Pos: 952
        Relay_Master_Log_File: mysql-bin.000007
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 739
              Relay_Log_Space: 1159
              Until_Condition: None
```

db002 / db003 –> Master 확인

Master(db001) –> slave 연결 확인



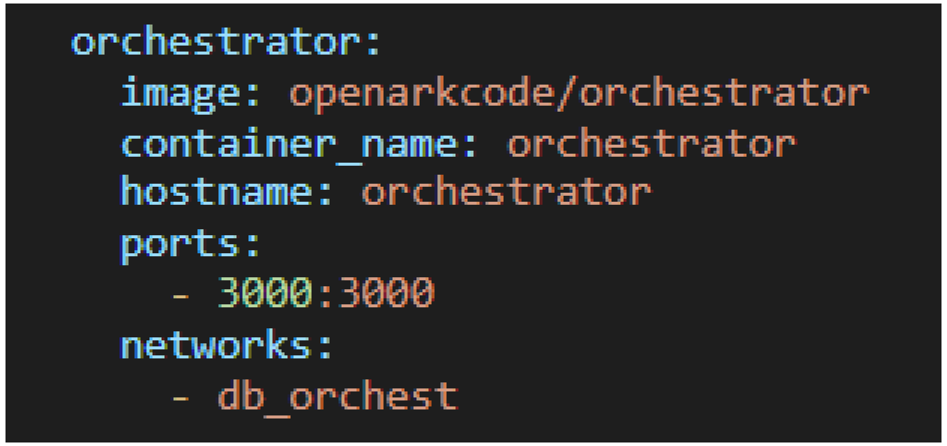
```
mysql> show slave hosts;
+-----------+-------+------+-----------+--------------------------------------+
| Server_id | Host  | Port | Master_id | Slave_UUID                           |
+-----------+-------+------+-----------+--------------------------------------+
|       300 | db003 | 3306 |       100 | 9fd3f36c-2491-11f0-9e13-46d66fd3ce64 |
|       200 | db002 | 3306 |       100 | 9f2f07e9-2491-11f0-b928-5663a7587291 |
+-----------+-------+------+-----------+--------------------------------------+
2 rows in set (0.00 sec)
```



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| testdb001          |
+--------------------+
5 rows in set (0.00 sec)

mysql> SELECT * FROM testdb001.testT001;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
+----+
3 rows in set (0.00 sec)
```

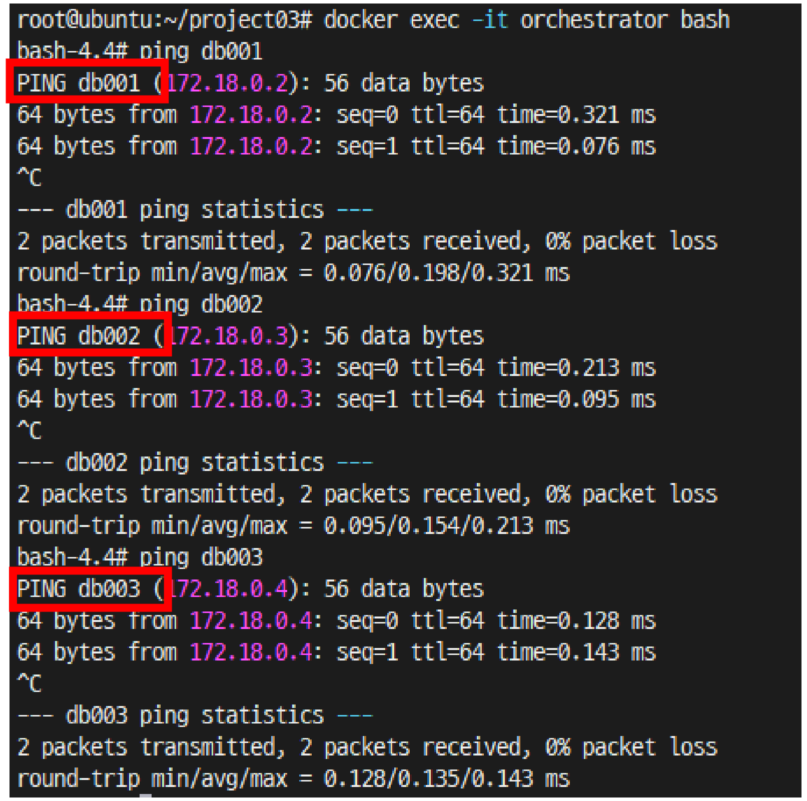
Slave 에서 동기화 된 DB 및 테이블 확인

Orchestrator – DB 컨테이너

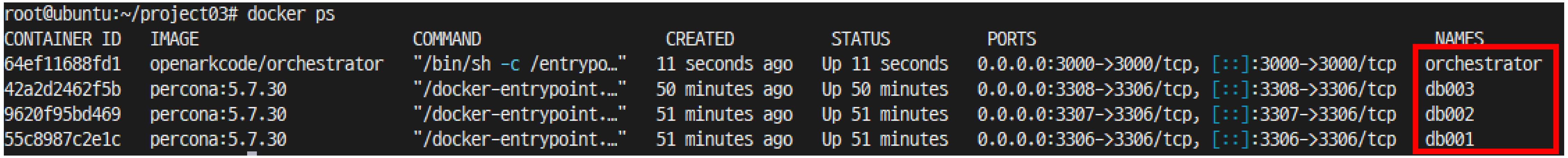


```
orchestrator:
  image: openarkcode/orchestrator
  container_name: orchestrator
  hostname: orchestrator
  ports:
    - 3000:3000
  networks:
    - db_orchest
```

docker-compose 파일에 추가



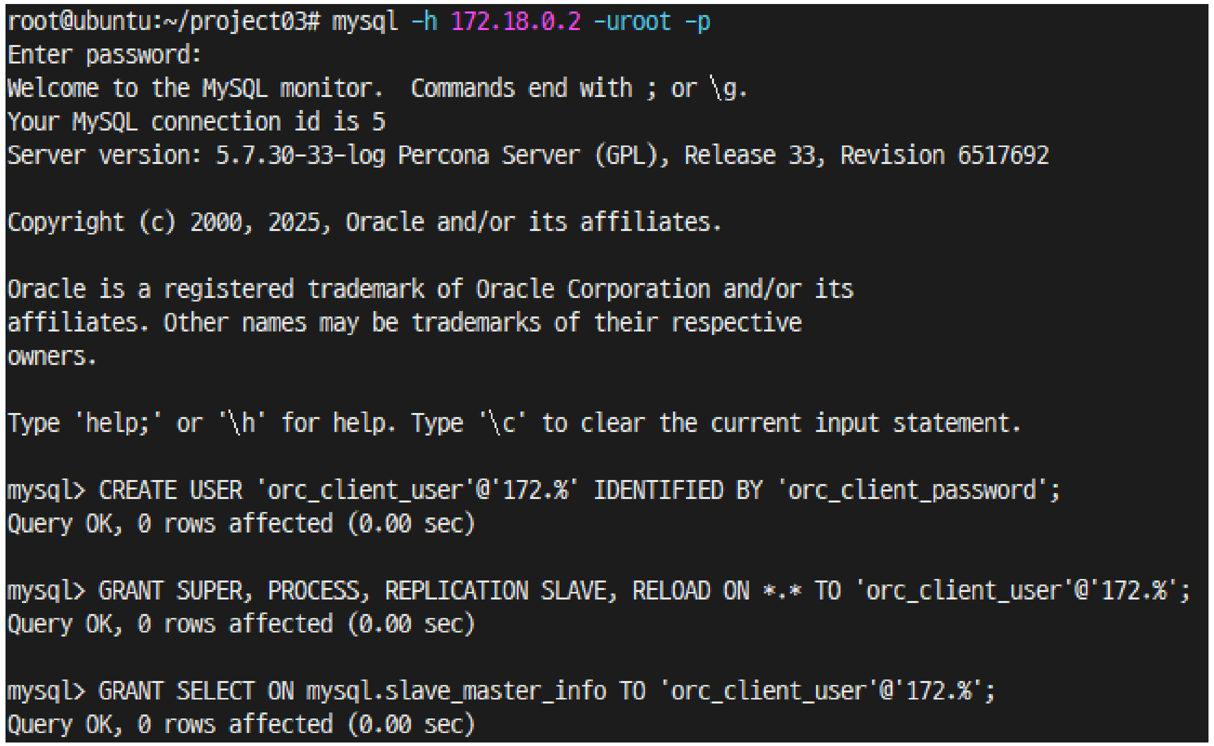
```
root@ubuntu:~/project03# docker exec -it orchestrator bash
bash-4.4# ping db001
PING db001 (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: seq=0 ttl=64 time=0.321 ms
64 bytes from 172.18.0.2: seq=1 ttl=64 time=0.076 ms
^C
--- db001 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.076/0.198/0.321 ms
bash-4.4# ping db002
PING db002 (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: seq=0 ttl=64 time=0.213 ms
64 bytes from 172.18.0.3: seq=1 ttl=64 time=0.095 ms
^C
--- db002 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.095/0.154/0.213 ms
bash-4.4# ping db003
PING db003 (172.18.0.4): 56 data bytes
64 bytes from 172.18.0.4: seq=0 ttl=64 time=0.128 ms
64 bytes from 172.18.0.4: seq=1 ttl=64 time=0.143 ms
^C
--- db003 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.128/0.135/0.143 ms
```



```
root@ubuntu:~/project03# docker ps
CONTAINER ID   IMAGE                      COMMAND                  CREATED          STATUS          PORTS                                          NAMES
64ef11688fd1   openarkcode/orchestrator   "/bin/sh -c /entrypo…"   11 seconds ago   Up 11 seconds   0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp   orchestrator
42a2d2462f5b   percona:5.7.30             "/docker-entrypoint.…"   50 minutes ago   Up 50 minutes   0.0.0.0:3308->3306/tcp, [::]:3308->3306/tcp   db003
9620f95bd469   percona:5.7.30             "/docker-entrypoint.…"   51 minutes ago   Up 51 minutes   0.0.0.0:3307->3306/tcp, [::]:3307->3306/tcp   db002
55c8987c2e1c   percona:5.7.30             "/docker-entrypoint.…"   51 minutes ago   Up 51 minutes   0.0.0.0:3306->3306/tcp, [::]:3306->3306/tcp   db001
```

Orchestrator 컨테이너와
DB 컨테이너의 통신 확인

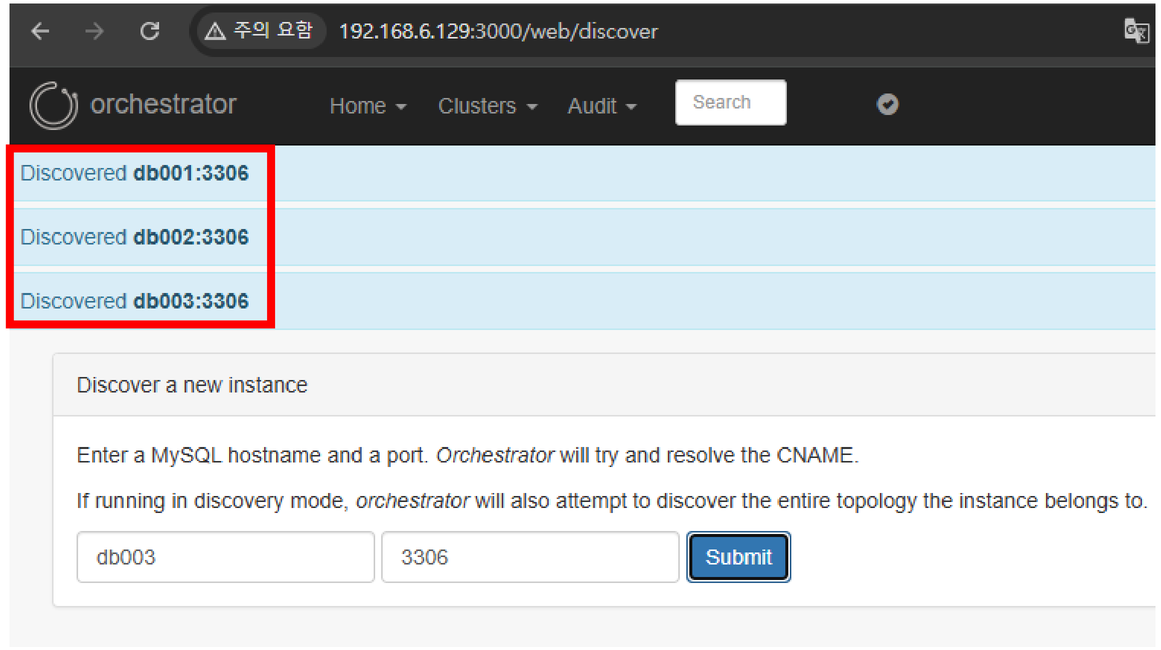
Orchestrator 설정



```
root@ubuntu:~/project03# mysql -h 172.18.0.2 -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.30-33-log Percona Server (GPL), Release 33, Revision 6517692

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE USER 'orc_client_user'@'172.%' IDENTIFIED BY 'orc_client_password';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT SUPER, PROCESS, REPLICATION SLAVE, RELOAD ON *.* TO 'orc_client_user'@'172.%';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT SELECT ON mysql.slave_master_info TO 'orc_client_user'@'172.%';
Query OK, 0 rows affected (0.00 sec)
```

Orchestrator가 사용할 MySQL user 생성 및 권한 추가
Master 에서 실행

Orchestrator 설정



Discovered **db001:3306**

Discovered **db002:3306**

Discovered **db003:3306**

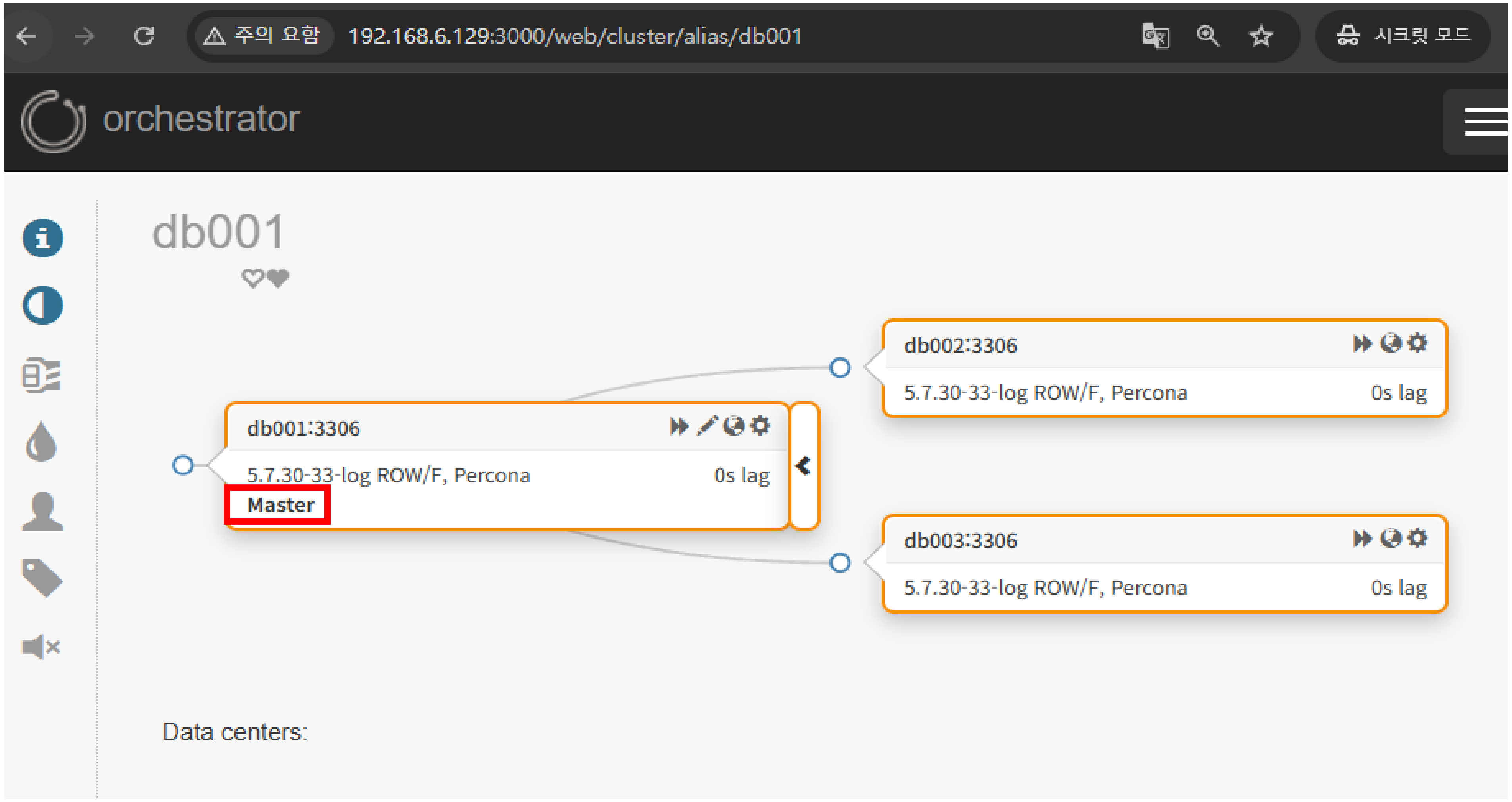
Discover a new instance

Enter a MySQL hostname and a port. *Orchestrator* will try and resolve the CNAME.

If running in discovery mode, *orchestrator* will also attempt to discover the entire topology the instance belongs to.

db003	3306	Submit

DB 연결 확인



db001

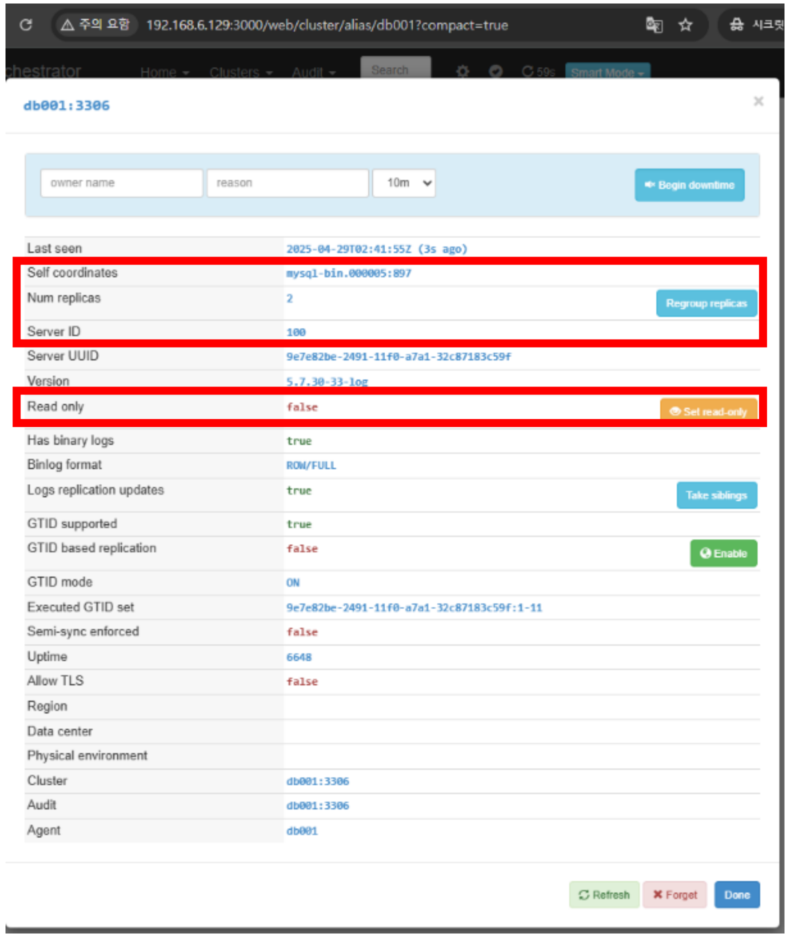
db002:3306
5.7.30-33-log ROW/F, Percona 0s lag

db001:3306
5.7.30-33-log ROW/F, Percona 0s lag
Master

db003:3306
5.7.30-33-log ROW/F, Percona 0s lag

Data centers:

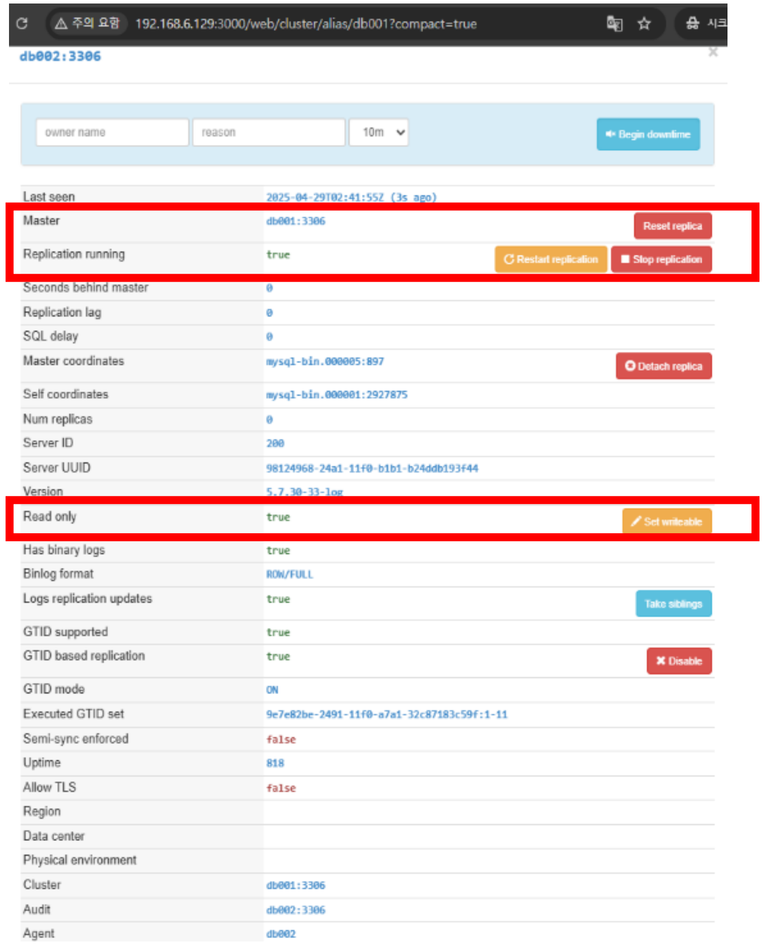
DB 상세 정보



db001:3306

Last seen	2025-04-29T02:41:55Z (3s ago)
Self coordinates	mysql-bin.000005:897
Num replicas	2
Server ID	100
Server UUID	9e7e82be-2491-11f0-a7a1-32c87183c59f
Version	5.7.30-33-log
Read only	false
Has binary logs	true
Binlog format	ROW/FULL
Logs replication updates	true
GTID supported	true
GTID based replication	false
GTID mode	ON
Executed GTID set	9e7e82be-2491-11f0-a7a1-32c87183c59f:1-11
Semi-sync enforced	false
Uptime	6648
Allow TLS	false
Region	
Data center	
Physical environment	
Cluster	db001:3306
Audit	db001:3306
Agent	db001

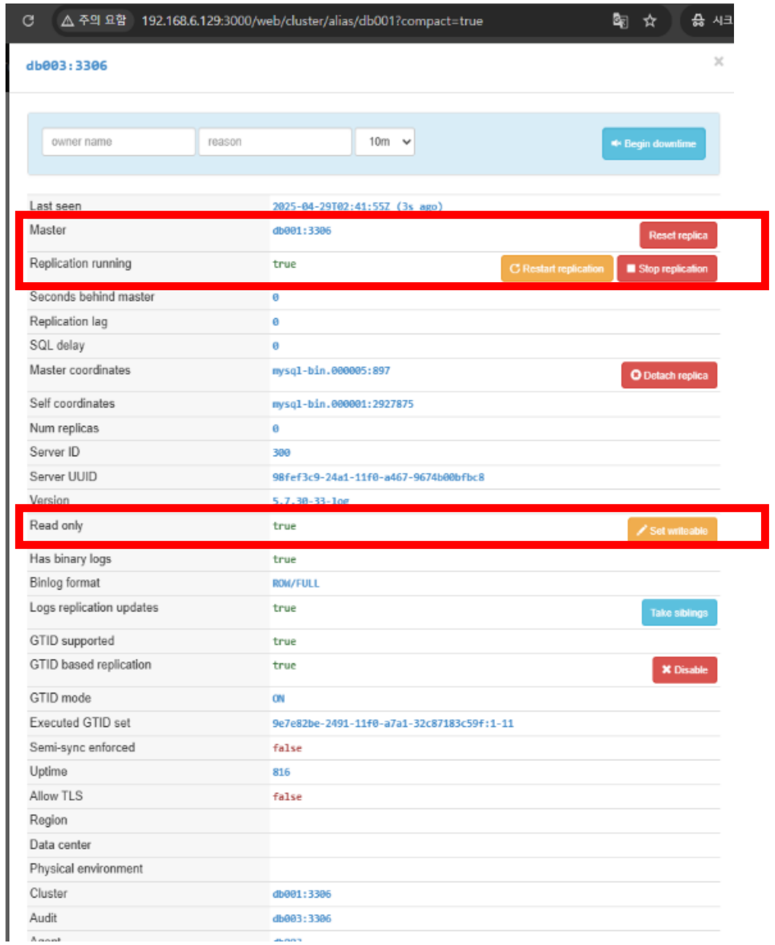
DB 001



db002:3306

Last seen	2025-04-29T02:41:55Z (3s ago)
Master	db001:3306
Replication running	true
Seconds behind master	0
Replication lag	0
SQL delay	0
Master coordinates	mysql-bin.000005:897
Self coordinates	mysql-bin.000001:2927875
Num replicas	0
Server ID	200
Server UUID	98124968-24a1-11f0-b1b1-b24ddb193f44
Version	5.7.30-33-log
Read only	true
Has binary logs	true
Binlog format	ROW/FULL
Logs replication updates	true
GTID supported	true
GTID based replication	true
GTID mode	ON
Executed GTID set	9e7e82be-2491-11f0-a7a1-32c87183c59f:1-11
Semi-sync enforced	false
Uptime	818
Allow TLS	false
Region	
Data center	
Physical environment	
Cluster	db001:3306
Audit	db002:3306
Agent	db002

DB
002



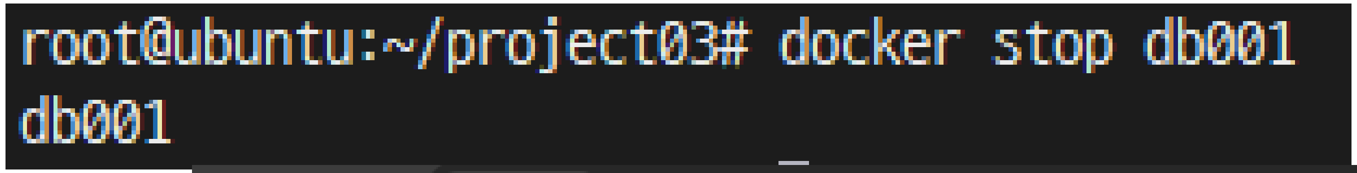
db003:3306

Last seen	2025-04-29T02:41:55Z (3s ago)
Master	db001:3306
Replication running	true
Seconds behind master	0
Replication lag	0
SQL delay	0
Master coordinates	mysql-bin.000005:897
Self coordinates	mysql-bin.000001:2927875
Num replicas	0
Server ID	300
Server UUID	98fef3c9-24a1-11f0-a467-9674b00bfbc8
Version	5.7.30-33-log
Read only	true
Has binary logs	true
Binlog format	ROW/FULL
Logs replication updates	true
GTID supported	true
GTID based replication	true
GTID mode	ON
Executed GTID set	9e7e82be-2491-11f0-a7a1-32c87183c59f:1-11
Semi-sync enforced	false
Uptime	816
Allow TLS	false
Region	
Data center	
Physical environment	
Cluster	db001:3306
Audit	db003:3306

DB 003

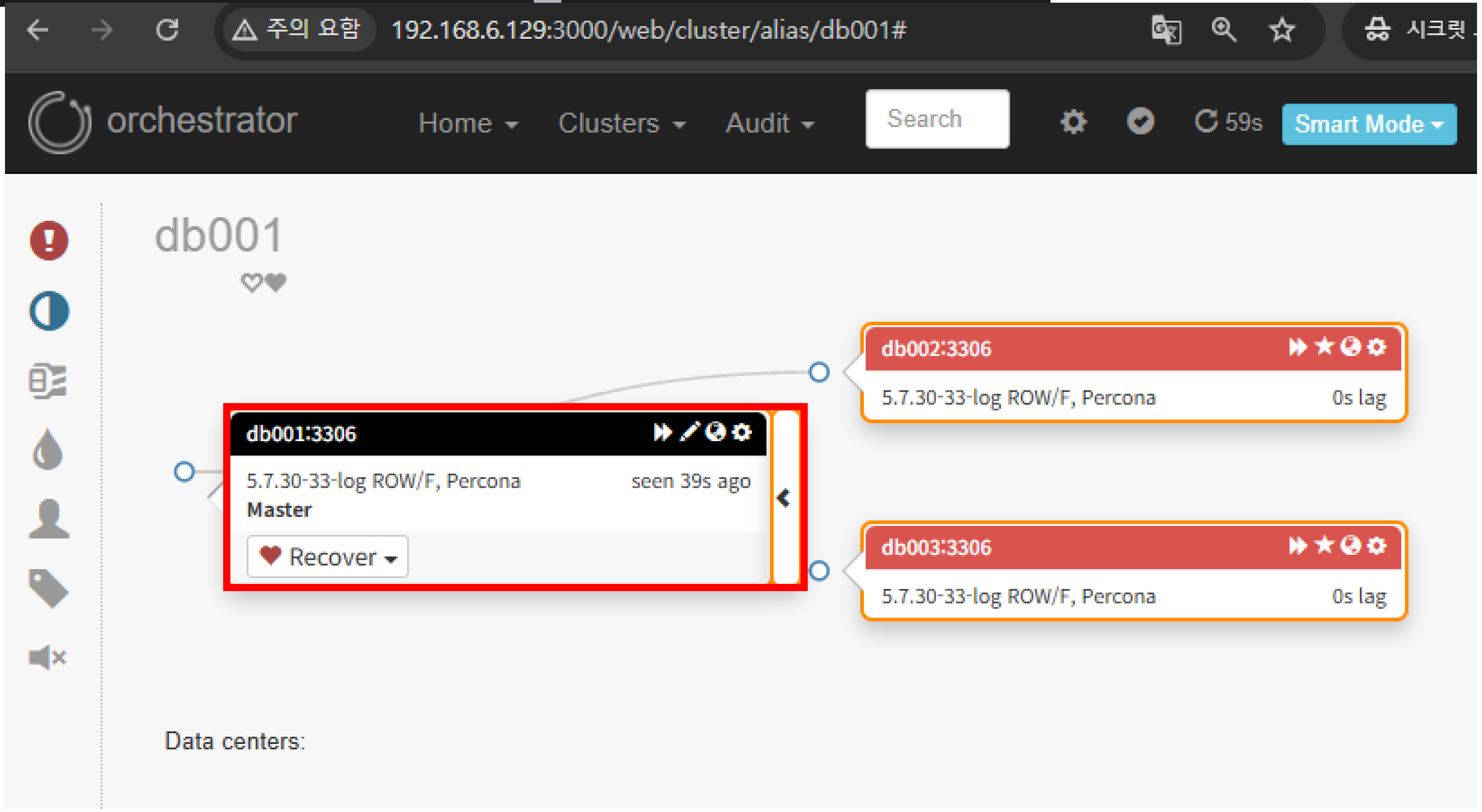
Master 장애 발생

db001 STOP



```
root@ubuntu:~/project03# docker stop db001
db001
```

장애 발생

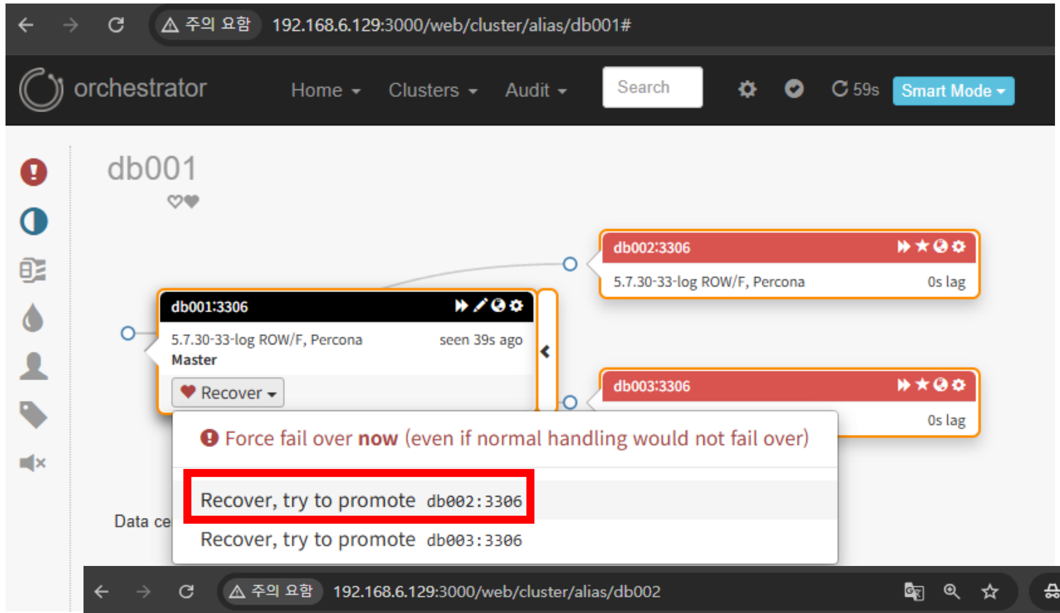


db001

```
db001:3306
5.7.30-33-log ROW/F, Percona          seen 39s ago
Master
Recover

db002:3306
5.7.30-33-log ROW/F, Percona          0s lag

db003:3306
5.7.30-33-log ROW/F, Percona          0s lag
```

Data centers:

db002 : Master 수동으로 db002를 Master로 승격



192.168.6.129:3000/web/cluster/alias/db001#

db001

db001:3306 — 5.7.30-33-log ROW/F, Percona — seen 39s ago — Master

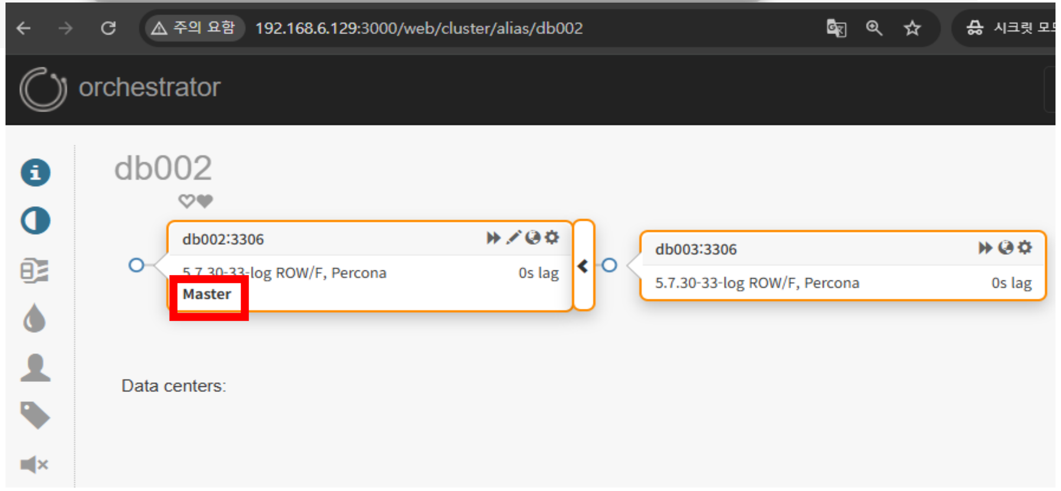
db002:3306 — 5.7.30-33-log ROW/F, Percona — 0s lag

db003:3306 — 0s lag

Force fail over **now** (even if normal handling would not fail over)

Recover, try to promote `db002:3306`

Recover, try to promote `db003:3306`



192.168.6.129:3000/web/cluster/alias/db002

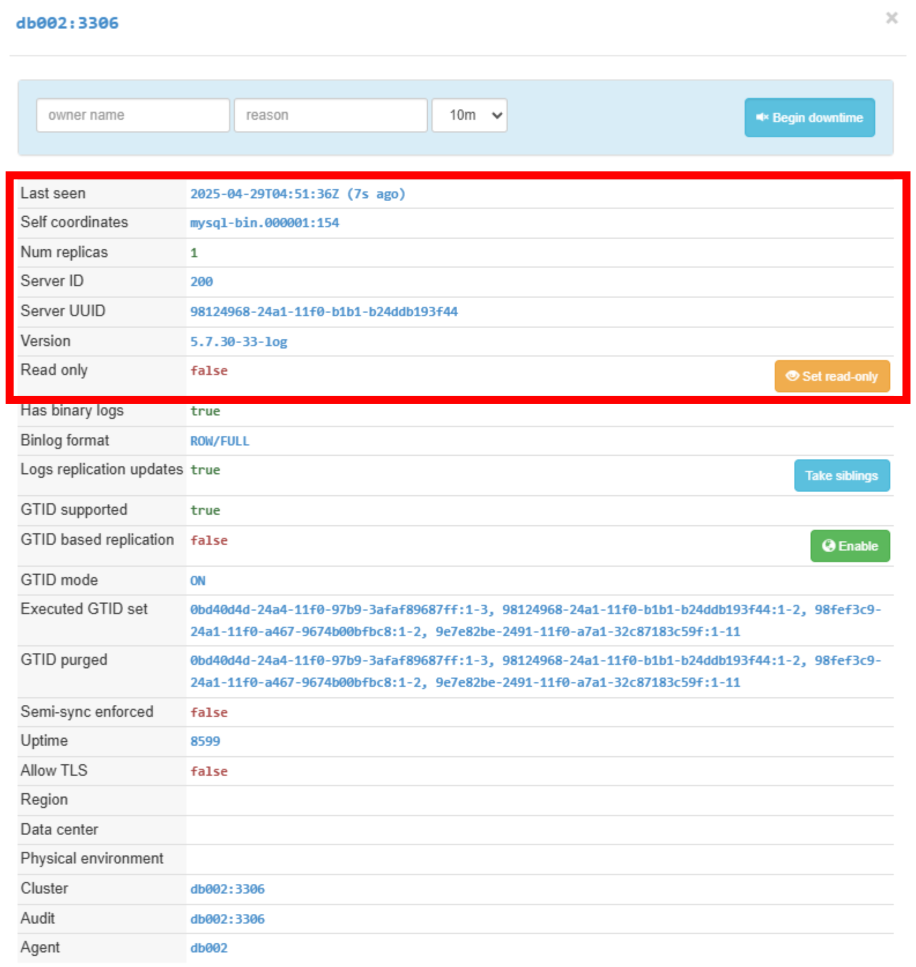
db002

db002:3306 — 5.7.30-33-log ROW/F, Percona — 0s lag — Master

db003:3306 — 5.7.30-33-log ROW/F, Percona — 0s lag

Data centers:

db003은 자동으로 db002를 MASTER로 인식

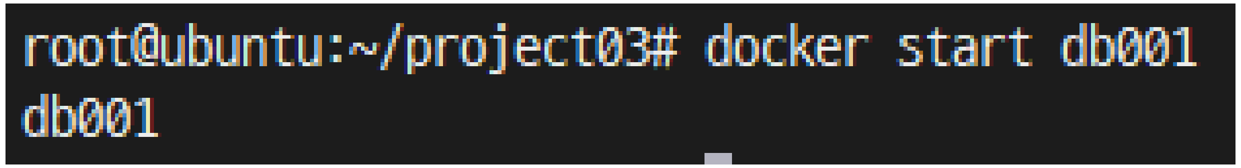


db002:3306

Last seen	2025-04-29T04:51:36Z (7s ago)
Self coordinates	mysql-bin.000001:154
Num replicas	1
Server ID	200
Server UUID	98124968-24a1-11f0-b1b1-b24ddb193f44
Version	5.7.30-33-log
Read only	false
Has binary logs	true
Binlog format	ROW/FULL
Logs replication updates	true
GTID supported	true
GTID based replication	false
GTID mode	ON
Executed GTID set	0bd40d4d-24a4-11f0-97b9-3afaf89687ff:1-3, 98124968-24a1-11f0-b1b1-b24ddb193f44:1-2, 98fef3c9-24a1-11f0-a467-9674b00bfbc8:1-2, 9e7e82be-2491-11f0-a7a1-32c87183c59f:1-11
GTID purged	0bd40d4d-24a4-11f0-97b9-3afaf89687ff:1-3, 98124968-24a1-11f0-b1b1-b24ddb193f44:1-2, 98fef3c9-24a1-11f0-a467-9674b00bfbc8:1-2, 9e7e82be-2491-11f0-a7a1-32c87183c59f:1-11
Semi-sync enforced	false
Uptime	8599
Allow TLS	false
Region	
Data center	
Physical environment	
Cluster	db002:3306
Audit	db002:3306
Agent	db002

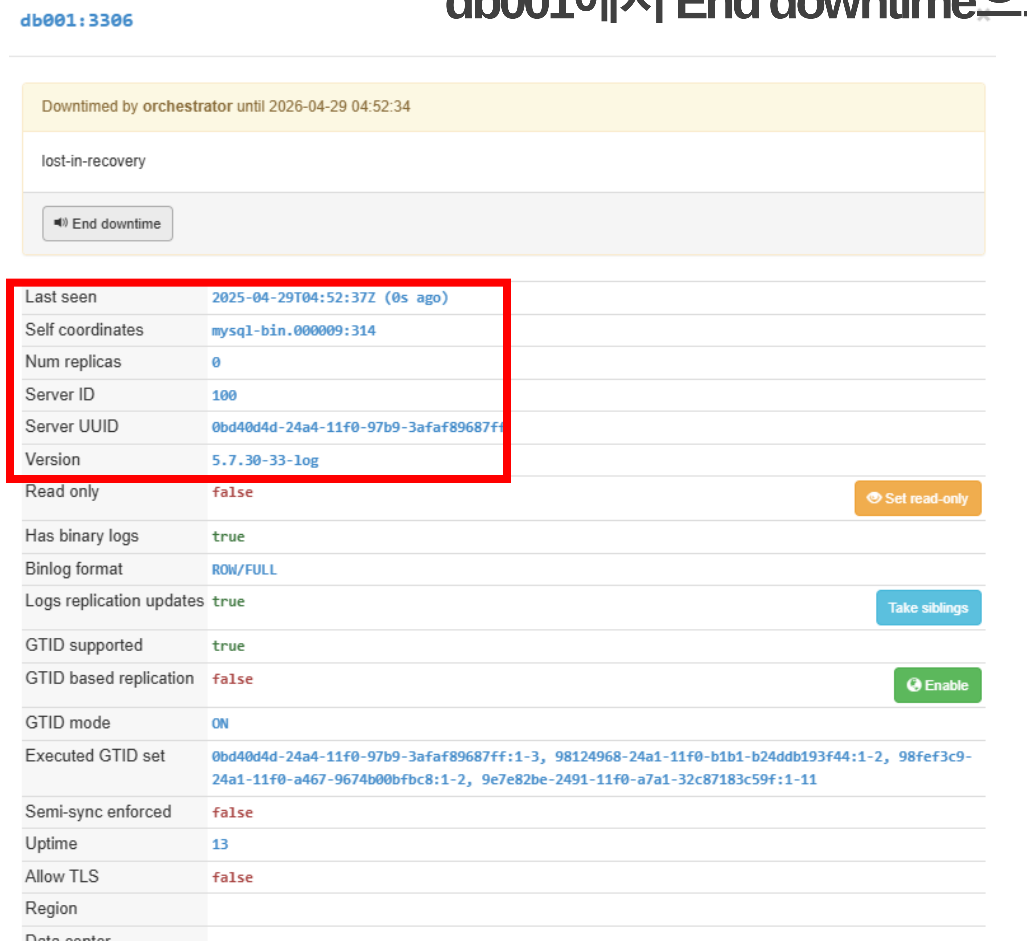
Master 로 승격된 db002 상세정보

db001 장애 복구 (수동)



```
root@ubuntu:~/project03# docker start db001
db001
```

db001에서 End downtime으로 복구 설정

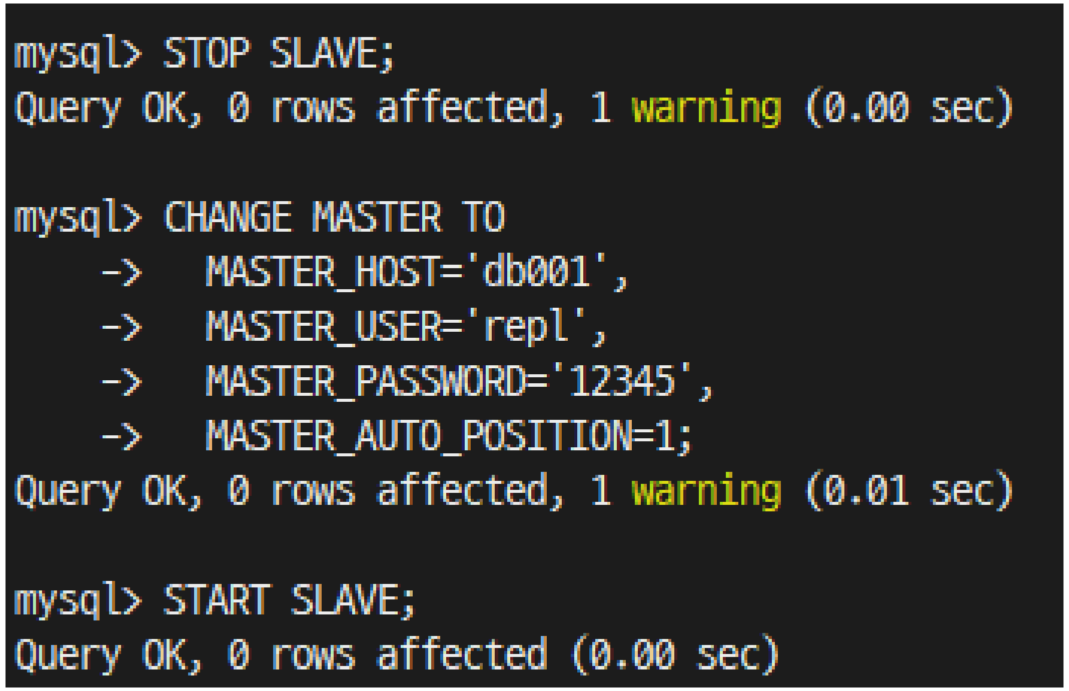


db001:3306

Downtimed by **orchestrator** until 2026-04-29 04:52:34

lost-in-recovery

Last seen	2025-04-29T04:52:37Z (0s ago)
Self coordinates	mysql-bin.000009:314
Num replicas	0
Server ID	100
Server UUID	0bd40d4d-24a4-11f0-97b9-3afaf89687ff
Version	5.7.30-33-log
Read only	false
Has binary logs	true
Binlog format	ROW/FULL
Logs replication updates	true
GTID supported	true
GTID based replication	false
GTID mode	ON
Executed GTID set	0bd40d4d-24a4-11f0-97b9-3afaf89687ff:1-3, 98124968-24a1-11f0-b1b1-b24ddb193f44:1-2, 98fef3c9-24a1-11f0-a467-9674b00bfbc8:1-2, 9e7e82be-2491-11f0-a7a1-32c87183c59f:1-11
Semi-sync enforced	false
Uptime	13
Allow TLS	false
Region	

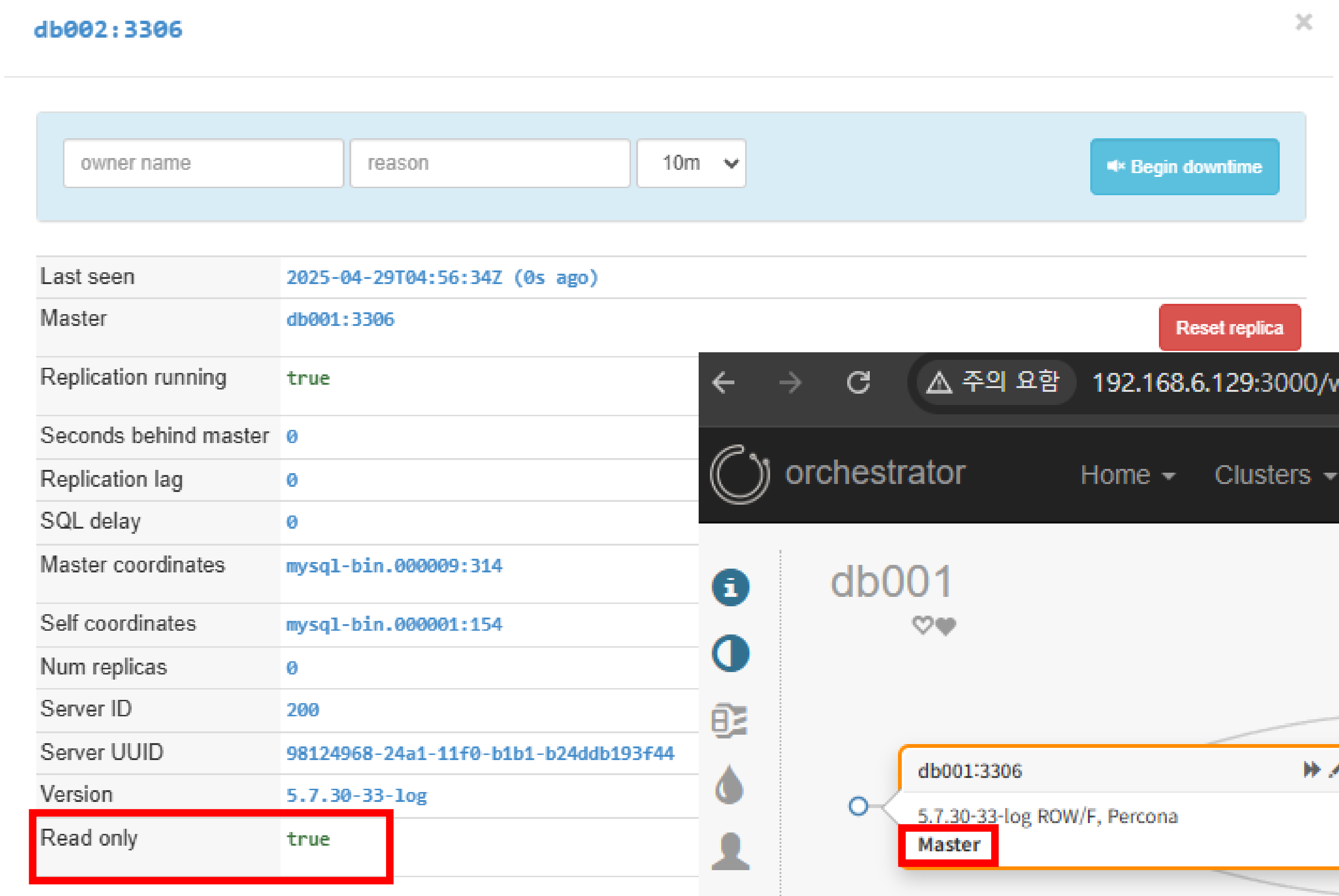


```
mysql> STOP SLAVE;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> CHANGE MASTER TO
    ->     MASTER_HOST='db001',
    ->     MASTER_USER='repl',
    ->     MASTER_PASSWORD='12345',
    ->     MASTER_AUTO_POSITION=1;
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> START SLAVE;
Query OK, 0 rows affected (0.00 sec)
```

db002와 db003에서 다시 db001을 마스터로 변경

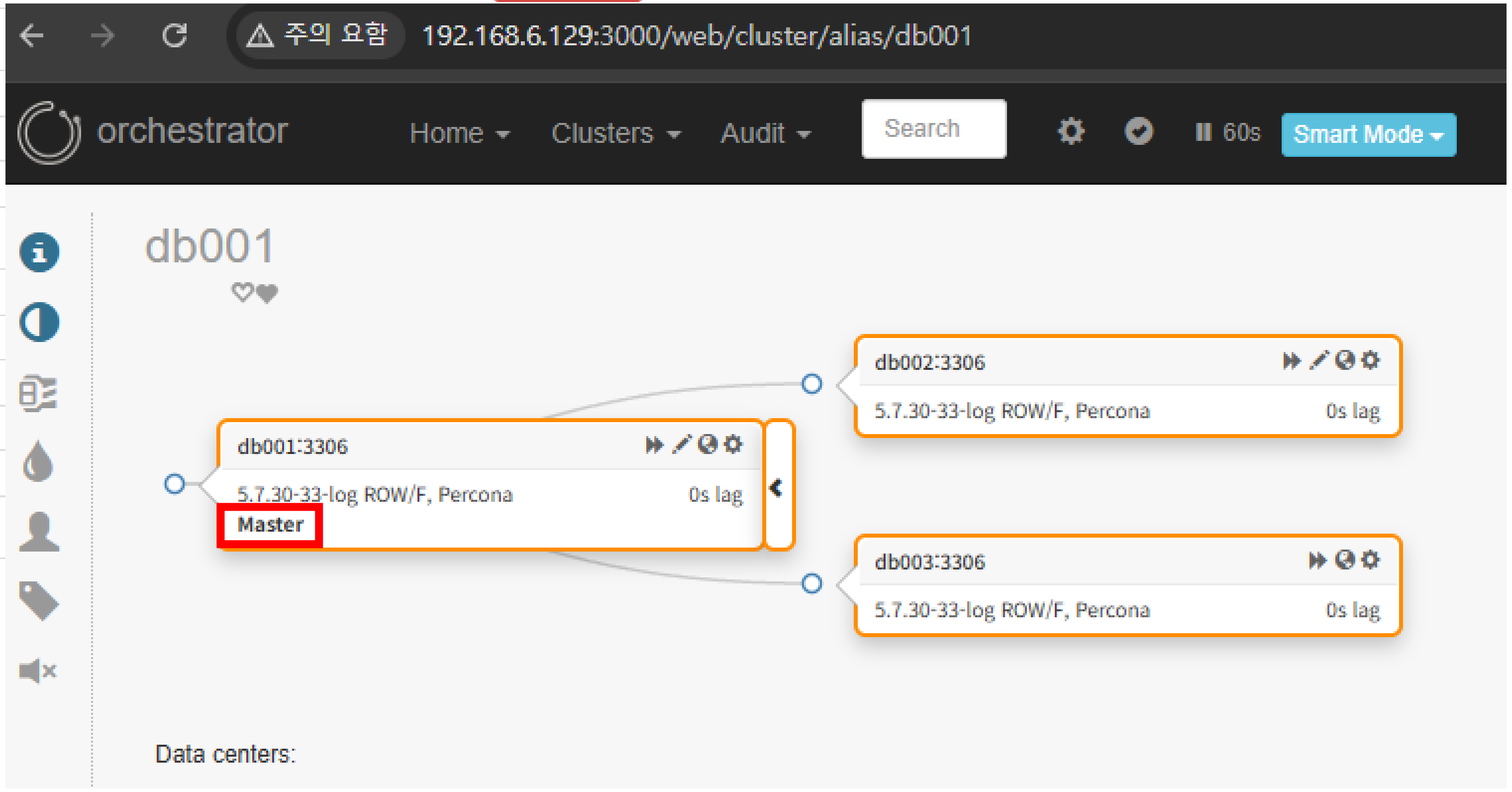
db002 설정

Slave 로 강등된 db002를 다시 read-only로 설정



db002:3306

Last seen	2025-04-29T04:56:34Z (0s ago)
Master	db001:3306
Replication running	true
Seconds behind master	0
Replication lag	0
SQL delay	0
Master coordinates	mysql-bin.000009:314
Self coordinates	mysql-bin.000001:154
Num replicas	0
Server ID	200
Server UUID	98124968-24a1-11f0-b1b1-b24ddb193f44
Version	5.7.30-33-log
Read only	true

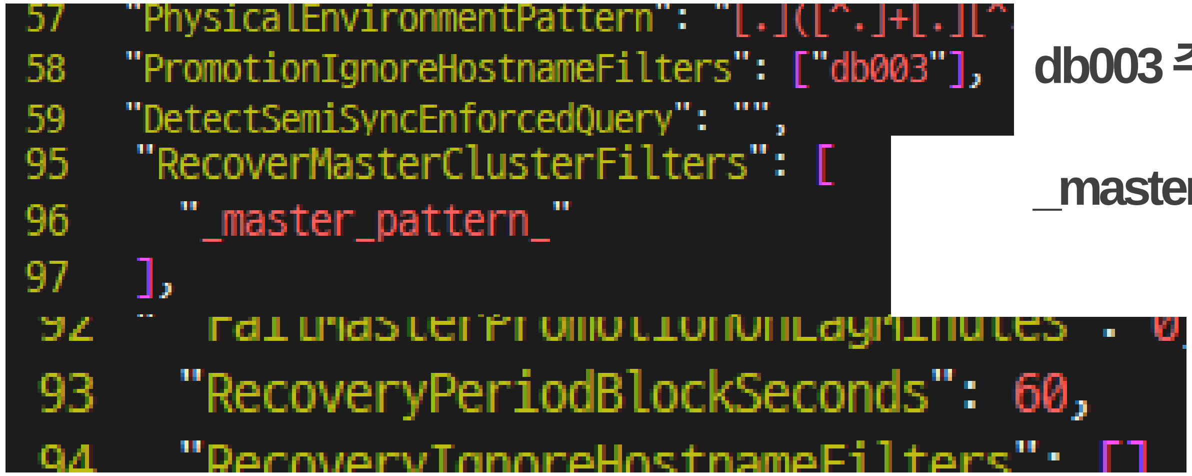


db002:3306
5.7.30-33-log ROW/F, Percona 0s lag

db001:3306
5.7.30-33-log ROW/F, Percona 0s lag
Master

db003:3306
5.7.30-33-log ROW/F, Percona 0s lag

Orchestrator 장애 복구 (자동)

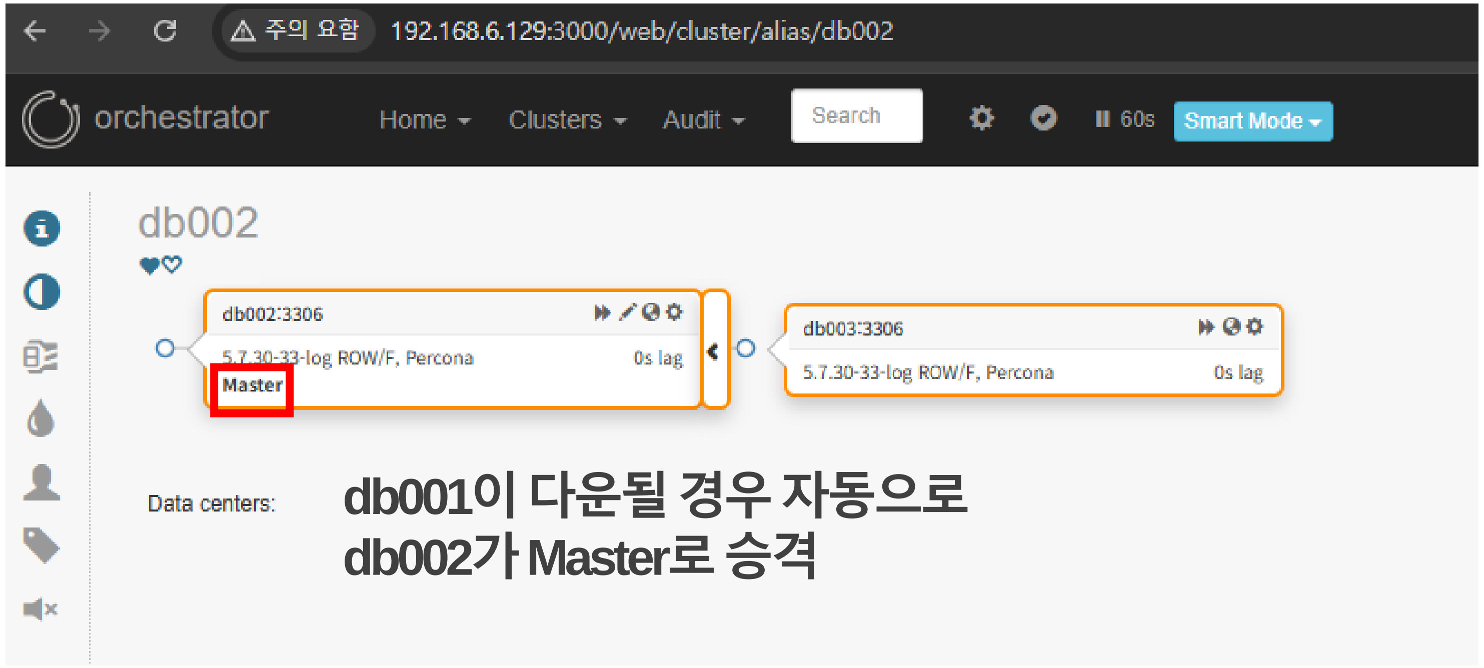


```
"PhysicalEnvironmentPattern": "[.]([^.]+[.][^
"PromotionIgnoreHostnameFilters": ["db003"],
"DetectSemiSyncEnforcedQuery": "",
 "RecoverMasterClusterFilters": [
   "_master_pattern_"
 ],
  "FailMasterPromotionOnLagMinutes": 0,
  "RecoveryPeriodBlockSeconds": 60,
  "RecoveryIgnoreHostnameFilters": []
```

db003 추가(db003은 master로 설정하지 않음)

_master_pattern_ => *

임시로 60초로 설정
- 해당 옵션은 해당 시간동안 같은 클러스터에 대해 추가적인 장애 복구를 막는 효과

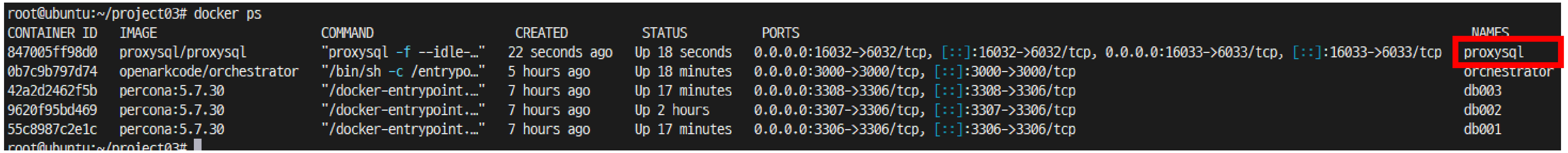


```
db002:3306
5.7.30-33-log ROW/F, Percona    0s lag
Master

db003:3306
5.7.30-33-log ROW/F, Percona    0s lag
```

db001이 다운될 경우 자동으로
db002가 Master로 승격

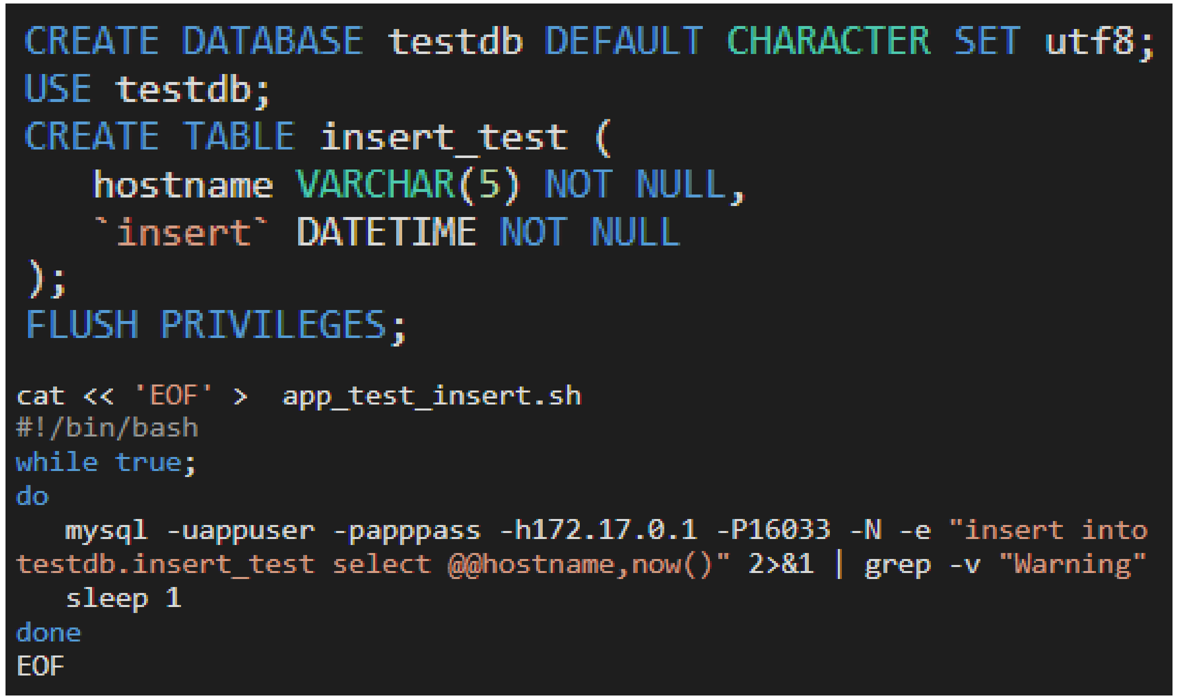
Proxy

Proxy 컨테이너 상태 확인



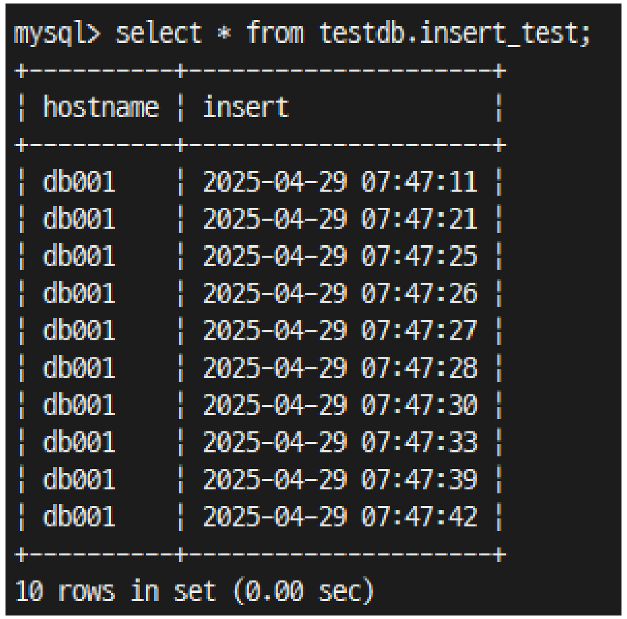
```
root@ubuntu:~/project03# docker ps
CONTAINER ID   IMAGE                         COMMAND                  CREATED          STATUS          PORTS                                                                                          NAMES
847005ff98d0   proxysql/proxysql             "proxysql -f --idle-…"   22 seconds ago   Up 18 seconds   0.0.0.0:16032->6032/tcp, [::]:16032->6032/tcp, 0.0.0.0:16033->6033/tcp, [::]:16033->6033/tcp   proxysql
0b7c9b797d74   openarkcode/orchestrator      "/bin/sh -c /entrypo…"   5 hours ago      Up 18 minutes   0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp                                                    orchestrator
42a2d2462f5b   percona:5.7.30                "/docker-entrypoint.…"   7 hours ago      Up 17 minutes   0.0.0.0:3308->3306/tcp, [::]:3308->3306/tcp                                                    db003
9620f95bd469   percona:5.7.30                "/docker-entrypoint.…"   7 hours ago      Up 2 hours      0.0.0.0:3307->3306/tcp, [::]:3307->3306/tcp                                                    db002
55c8987c2e1c   percona:5.7.30                "/docker-entrypoint.…"   7 hours ago      Up 17 minutes   0.0.0.0:3306->3306/tcp, [::]:3306->3306/tcp                                                    db001
root@ubuntu:~/project03#
```

INSERT 테스트용 데이터베이스 및 테이블 생성
INSERT 테스트용 파일 생성



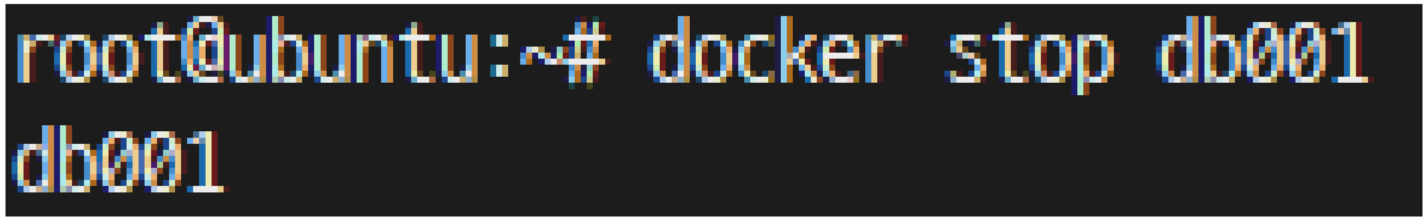
```
CREATE DATABASE testdb DEFAULT CHARACTER SET utf8;
USE testdb;
CREATE TABLE insert_test (
    hostname VARCHAR(5) NOT NULL,
    `insert` DATETIME NOT NULL
);
FLUSH PRIVILEGES;

cat << 'EOF' >  app_test_insert.sh
#!/bin/bash
while true;
do
   mysql -uappuser -papppass -h172.17.0.1 -P16033 -N -e "insert into
testdb.insert_test select @@hostname,now()" 2>&1 | grep -v "Warning"
   sleep 1
done
EOF
```

INSERT 테스트 결과
- INSERT 명령은 db001로만 연결



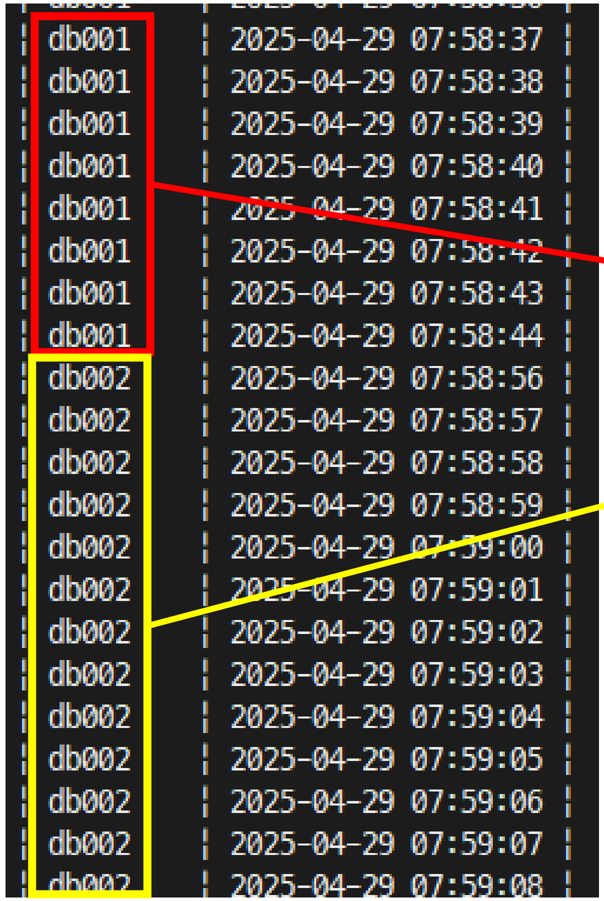
```
mysql> select * from testdb.insert_test;
+----------+---------------------+
| hostname | insert              |
+----------+---------------------+
| db001    | 2025-04-29 07:47:11 |
| db001    | 2025-04-29 07:47:21 |
| db001    | 2025-04-29 07:47:25 |
| db001    | 2025-04-29 07:47:26 |
| db001    | 2025-04-29 07:47:27 |
| db001    | 2025-04-29 07:47:28 |
| db001    | 2025-04-29 07:47:30 |
| db001    | 2025-04-29 07:47:33 |
| db001    | 2025-04-29 07:47:39 |
| db001    | 2025-04-29 07:47:42 |
+----------+---------------------+
10 rows in set (0.00 sec)
```

장애 발생 및 복구(자동)



```
root@ubuntu:~# docker stop db001
db001
```

db001에 장애 발생을 가정



```
| db001    | 2025-04-29 07:58:37 |
| db001    | 2025-04-29 07:58:38 |
| db001    | 2025-04-29 07:58:39 |
| db001    | 2025-04-29 07:58:40 |
| db001    | 2025-04-29 07:58:41 |
| db001    | 2025-04-29 07:58:42 |
| db001    | 2025-04-29 07:58:43 |
| db001    | 2025-04-29 07:58:44 |
| db002    | 2025-04-29 07:58:56 |
| db002    | 2025-04-29 07:58:57 |
| db002    | 2025-04-29 07:58:58 |
| db002    | 2025-04-29 07:58:59 |
| db002    | 2025-04-29 07:59:00 |
| db002    | 2025-04-29 07:59:01 |
| db002    | 2025-04-29 07:59:02 |
| db002    | 2025-04-29 07:59:03 |
| db002    | 2025-04-29 07:59:04 |
| db002    | 2025-04-29 07:59:05 |
| db002    | 2025-04-29 07:59:06 |
| db002    | 2025-04-29 07:59:07 |
| db002    | 2025-04-29 07:59:08 |
```

ProxySQL에서 자동으로 db001에서 db002로 전환 완료

QUESTION

THANK YOU