MySQL High Availability and Disaster Recovery
14+ Years of Experience
250+ Global Customers
15+ DC Exit to Cloud
8+ Countries

800+ Cloud Certifications
500+ Global Talent Pool
250+ Cloud/DB Engineers
150+ Application Consultants
2 Oracle Certified Masters
25 Specializations
INFOLOB is the "ONE and ONLY" first Oracle partner to complete leading Oracle MySQL certifications including MySQL Heatwave.

- MySQL Heatwave Implementation Certified Associate
- Oracle Certified Professional, MySQL 8.0 Database Administrator
- MySQL 2021 Certified Implementation Specialist
- Solution Engineering Specialist - Oracle MySQL
- Solution Engineering Specialist - Oracle MySQL Heatwave
Introducing our book

This book offers step-by-step guidance on installing, upgrading, and establishing robust high availability and disaster recovery capabilities for MySQL databases. It also covers high availability with InnoDB and NDB clusters, MySQL routers and enterprise MySQL tools, and robust security design and performance techniques.

- Chapter 1: MySQL Installation and Upgrade
- Chapter 2: MySQL Utilities
- Chapter 3: MySQL Server Administration
- Chapter 4: MySQL Tablespace Management and Partitioning
- Chapter 5: MySQL High Availability, Replication, and Scalability
- Chapter 6: MySQL InnoDB Cluster and Cluster Set
- Chapter 7: MySQL NDB Cluster
- Chapter 8: MySQL Logical Backup
- Chapter 9: MySQL Enterprise Backup and Recovery
- Chapter 10: MySQL Security
- Chapter 11: MySQL Performance Tuning
- Chapter 12: MySQL Enterprise Monitor
- Chapter 13: Monitoring MySQL Using Oracle Enterprise Manager Cloud Control 13c
- Chapter 14: MySQL Troubleshooting
Y V Ravi Kumar

• 26+ years of industry experience
• EB1-A "Einstein Green Card" Recipient from United States
• Oracle Certified Master (OCM)
• Published x(11) books on Oracle Technology
  • Co-author of (x6) books
  • Technical Reviewer of (x5) books
• Published 100+ Oracle Technology Network (OTN) - English, Portuguese & Spanish
• Speaker 4x @Oracle Open/Cloud World, US
• Multi-Cloud Certified Architect
Agenda

- Introduction
- Architecture of MySQL InnoDB Cluster
- MySQL Group Replication Plug-In
- How MySQL Group Replication will help in the InnoDB Cluster
- Role of MySQL Shell in InnoDB Cluster
- MySQL Router and its usage
- InnoDB Clusterset Architecture and configuration
- InnoDB Cluster and Clusterset scenarios
Common causes of IT disasters and system-wide outages

- Power outages
- Hardware failure
- Software issues
- Human error
- Network failures
- Natural disaster
MySQL Replication primer

- Replica connects to Source
- I/O thread gets data
- Binlog dump sends data to the I/O thread
- SQL thread applies data
- Statement based vs. Row based vs. Mixed
- Asynchronous vs Semi Synchronous

Source

Database

Database

Replica

Binary log

Relay log

Applied to Database

I/O

DML

Binlog dump

SQL
MySQL Replication topologies

- Source - Replica
- Source - Multiple Replicas
- Multiple Source – Replica
- Source – Replica – Multiple Replicas
- Multi-Source
MySQL Group Replication

- An elastic, highly-available, fault-tolerant replication topology
- Offered as a plugin to MySQL server
- Operates in a single primary mode with automatic primary election
- Built-in group membership service to guarantee database service availability
- Client connections need to be redirected or failed over if the primary becomes unavailable

P = Primary; S = Secondary
Primary Cluster with DR Cluster

P = Primary; S = Secondary
Arun Samayam

• 15+ years of experience with different database platforms as a DBA and Architect

• Multi-cloud certified professional

• Co-author: Mastering MySQL Administrator

• Technical reviewer: Oracle Cloud Infrastructure (OCI) Goldengate

• Speaker(x2): Oracle OpenWorld
MySQL InnoDB Cluster

- Provides a highly available and scalable solution
- Key components
  - MySQL Server
  - MySQL Group Replication
  - MySQL Shell
  - MySQL Router
- MySQL Group Replication: replicate data between all the servers in the cluster with automatic failover management
- MySQL Shell: Advanced client and code editor provides scripting capabilities in Python and JavaScript
- MySQL Router: Transparent client connection routing between the application and the cluster

P = Primary; S = Secondary
MySQL InnoDB Cluster Architecture

- Relies on MySQL Group replication which is installed on each server instance
- Group replication enables you to create elastic replication topologies
- Has the ability to reconfigure itself automatically if a server within the cluster goes offline
- Atleast 3 servers to form a group
- Operates in a single-primary mode or multi-primary mode
- Group replication plugin was introduced in 5.7 but it is tricky to work with directly
- MySQL InnoDB cluster introduced new components that are integrated tightly which made it easier to set up and administer
MySQL InnoDB Cluster creation

- Pre-check instance configuration for InnoDB Cluster usage
  
  MySQL localhost:3306+ ssl JS > `dba.checkInstanceConfiguration('mysqlclusteradmin@localhost:3306')`  

- Create a cluster named
  
  MySQL localhost:3306+ ssl JS > `dba.createCluster("myPrimaryCluster")`  

- Create a variable named “cluster”
  
  MySQL localhost:3306+ ssl JS > `var cluster=dba.getCluster()`  

- Add an instance to the cluster
  
  MySQL localhost:3306+ ssl JS > `cluster.addInstance('mysqlclusteradmin@mysql-b:3306',{recoveryMethod:'clone'})`  

- Check cluster status
  
  MySQL localhost:3306+ ssl JS > `cluster.status()`
MySQL localhost:3306+ ssl > cluster.status()
{
    "clusterName": "myPrimaryCluster",
    "defaultReplicaSet": {
        "name": "default",
        "primary": "mysql-a:3306",
        "ssl": "REQUIRED",
        "status": "OK",
        "statusText": "Cluster is ONLINE and can tolerate up to ONE failure.",
        "topology": {
            "mysql-a:3306": {
                "address": "mysql-a:3306",
                "memberRole": "PRIMARY",
                "mode": "R/W",
                "readReplicas": {},
                "replicationLag": "applier_queue_applied",
                "role": "HA",
                "status": "ONLINE",
                "version": "8.0.34"
            },
            "mysql-b:3306": {
                "address": "mysql-b:3306",
                "memberRole": "SECONDARY",
                "mode": "R/O",
                "readReplicas": {},
                "replicationLag": "applier_queue_applied",
                "role": "HA",
                "status": "ONLINE",
                "version": "8.0.34"
            },
            "mysql-c:3306": {
                "address": "mysql-c:3306",
                "memberRole": "SECONDARY",
                "mode": "R/O",
                "readReplicas": {},
                "replicationLag": "applier_queue_applied",
                "role": "HA",
                "status": "ONLINE",
                "version": "8.0.34"
            }
        },
        "topologyMode": "Single-Primary"
    },
    "groupInformationSourceMember": "mysql-a:3306"
}
MySQL InnoDB Clusterset

MySQL Connector

MySQL Shell

MySQL Admin API

Group Replication

Primary Cluster

P = Primary; S = Secondary

Asynchronous Replication

HA & DR

P = Primary; S = Secondary

Group Replication

MySQL Connector

MySQL Shell

MySQL Admin API

Group Replication

DR Cluster
• Pre-check instance configuration for InnoDB Cluster usage

  MySQL localhost:3306+ ssl JS > **dba.checkInstanceConfiguration('mysqlclusteradmin@localhost:3306')**

• Create a primary cluster named

  MySQL localhost:3306+ ssl JS > **dba.createCluster("myPrimaryCluster")**

• Create a variable named “cluster”

  MySQL localhost:3306+ ssl JS > **var cluster=dba.getCluster()**

• Add an instance to the cluster

  MySQL localhost:3306+ ssl JS > **cluster.addInstance('mysqlclusteradmin@mysql-b:3306',{recoveryMethod:'clone'})**

• Check cluster status

  MySQL localhost:3306+ ssl JS > **cluster.status()**

• Create a clusterset named “myClusterset”

  MySQL localhost:3306+ ssl JS > **clusterset=cluster.createClusterSet("myclusterset")**

• Create a replica cluster named “mydrcluster”

  MySQL localhost:3306+ ssl JS > **mydrcluster=myclusterset.createReplicaCluster("mysqlclusteradmin@mysql-d:3306", "mydrcluster", {recoveryProgress:1, timeout:10})**
• Check ClusterSet status

MySQL localhost:3306+ ssl JS > myclusterset.status()
{
    "clusters": {
        "myPrimaryCluster": {
            "clusterRole": "PRIMARY",
            "globalStatus": "OK",
            "primary": "mysql-a:3306"
        },
        "mydrcluster": {
            "clusterRole": "REPLICA",
            "clusterSetReplicationStatus": "OK",
            "globalStatus": "OK"
        }
    },
    "domainName": "myclusterset",
    "globalPrimaryInstance": "mysql-a:3306",
    "primaryCluster": "myPrimaryCluster",
    "status": "HEALTHY",
    "statusText": "All Clusters available."
}

MySQL localhost:3306+ ssl JS > myclusterset.status({extended:1})
Setup MySQL Router

MySQL localhost:3306+ ssl JS > `cluster.setupRouterAccount("routeradmin")`

Create an OS user to run MySQL router and bootstrap MySQL Router

```bash
# useradd routeruser

# mysqlrouter --bootstrap root@localhost:3306 --directory /home/routeruser --conf-use-sockets --account routeradmin --user=routeruser --force
```

Router configuration file default location: `<router_home_directory>/mysqlrouter.conf`

Start router

```bash
# /home/routeruser/start.sh
```

Check router status

MySQL localhost:3306+ ssl JS > `cluster.listRouters()`
Demo
Scenario 1: Switchover

Group Replication

MySQL Connector
MySQL Shell
MySQL Admin API

Primary Cluster

mysql-a
mysql-b
mysql-c

Group Replication

P = Primary; S = Secondary

Asynchronous Replication

DR Cluster

mysql-d
mysql-e
mysql-f

MySQL Connector
MySQL Shell
MySQL Admin API
Scenario 1: After Switchover

Group Replication

MySQL Connector

MySQL Shell

MySQL Admin API

Asynchronous Replication

P = Primary; S = Secondary
Scenario 2: Primary MySQL server in Primary Cluster fails

MySQL Connector
MySQL Shell
MySQL Admin API

Primary Cluster

Asynchronous Replication

DR Cluster

P = Primary; S = Secondary
Scenario 2: Primary MySQL server in Primary Cluster fails

MySQL Connector

MySQL Shell

MySQL Admin API

mysql-b

mysql-c

New Primary

Group Replication

P = Primary; S = Secondary

Asynchronous Replication

DR Cluster

mysql-d

mysql-e

mysql-f

Group Replication

P

S

S

S