MySQL Enterprise Edition on Docker

John Kehoe, OCP, OCP MySQL NDB Cluster, OCI Associate Architect
Principal Solution Architect

John.Kehoe@oracle.com
+1.312.651.8756

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

MySQL on Docker

- About MySQL
- Why
- How
- Security Concerns
- Securing MySQL on Docker

About MySQL

What is the Draw to MySQL

- Most widely used Open Source RDMS
- Transactional, RDBMS, Document Store and NoSQL in one
- Open Source
- Low barrier to entry
- Replication without shared instances
- Over 20 years of corporate stewardship and backing

Why Docker

- Docker containers can provision entire application stacks quickly
- Easy returns to an initial state
- Ephemeral design makes for fast scaling

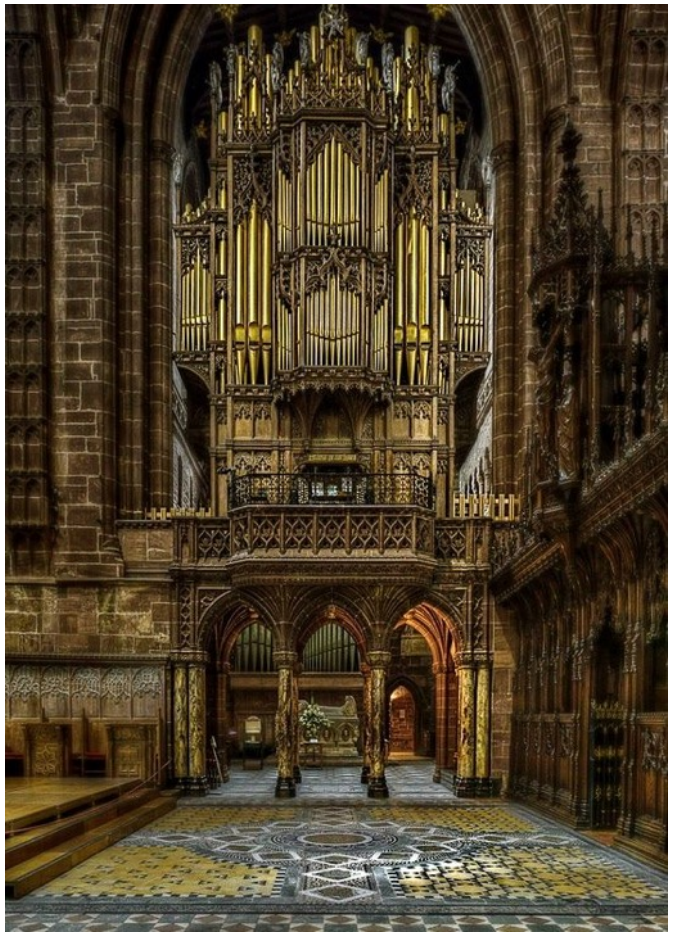
But Why a Stateful/Persistent Database on Docker

- Same reason you'd use Docker for Stateless Containers
- Fast to provision
- Flexibility in configuration

However there are things we need to seriously reflect upon when running databases on a stateless infrastructure

Hand Crafted v Mass Production –Both have a place

But then Sometimes…



Databases on Docker Considerations

- Containers are Ephemeral
- Shared Kernel, limited isolation, no username space
- Noisy neighbors, shared resources
- Persisting volumes stored outside of container
- Hardening and Securing data

”Who owns your data?”, can you affirm you own it when the auditors come to visit (GDPR and CCPA)

Operational Problems to Solve

- Orchestration of MySQL (InnoDB Cluster)
- Securing data files
- Limiting impact of noisy neighbors
- Auditing
- Monitoring

Get InnoDB Cluster Running on Docker

MySQL Enterprise Edition that is

Many Sources of MySQL Docker Containers

- Docker Hub
- Oracle (Enterprise Edition)
- Roll your own

The Basics -InnoDB Cluster and Docker

Instructions from Robert Neumayer from the MySQL Release team https://mysqlrelease.com/2018/03/docker-compose-setup-for-innodb-cluster/

https://github.com/neumayer/mysql-docker-compose-examples

This will provide images and instructions for a non-persisted, Community Edition, 3 node InnoDB Cluster and MySQL Router container

Servers and Connections

Network Created for InnoDB Cluster: 172.x:0.0/16

We'll access via 127.0.0.1 and the Exposed Port

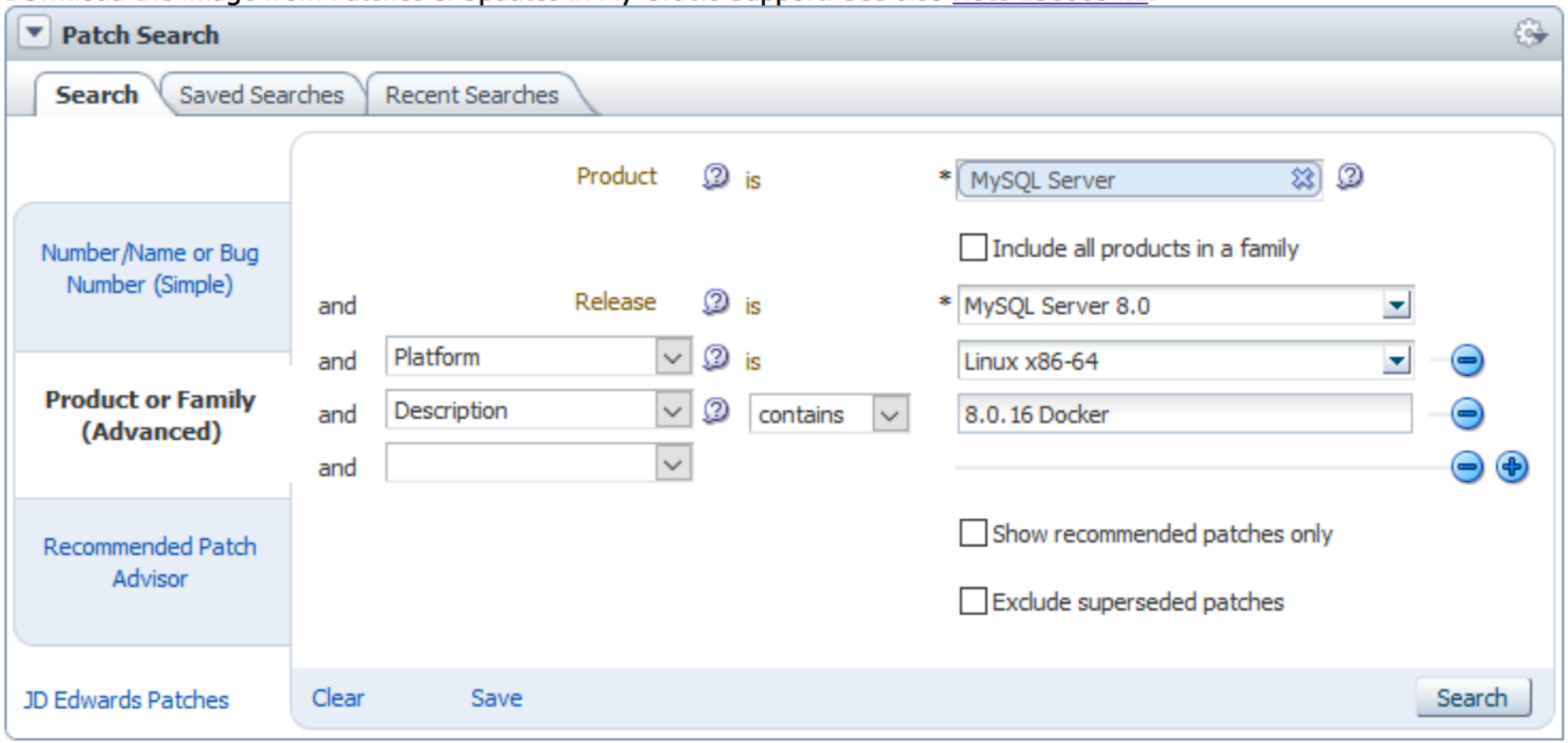
Container	Exposed Port : Internal Port
innodb-cluster_mysql-router_1	6446:6446
innodb-cluster_mysql-server-3_1	3303:3306
innodb-cluster_mysql-server-2_1	3302:3306
innodb-cluster_mysql-server-3_1	3301:3306

Advanced InnoDB Cluster and Docker

- To use Security Enhancements, we need the Enterprise Edition of MySQL, available from MOS
- Additionally, we need MySQL Enterprise Monitor to measure replication performance. That is is covered in another session.

Get Enterprise Edition Docker Image from Support (1)

1. Download the image from Patches & Updates in My Oracle Support. See also Note 1300654.1.



Get Enterprise Edition Docker Image from Support (2)

2. Extract the image from the downloaded ZIP file, for example:

```
shell$ unzip p29703511_580_Linux-x86-64.zip
```

3. To load the image, use the `docker load` command with the extracted TAR file, for example:

```
shell$ docker load -i mysql-enterprise-server-8.0.16.tar
```

4. Verify the image was loaded:

```
shell$ docker images
```

The image should be available as *mysql/enterprise-server*.

Docker-compose.yml Edits

```
image: mysql/mysql-server:8.0.16
ports:
      - "3301:3306"
command: ["mysqld","--server_id=1","--binlog_checksum=NONE","--gtid_mode=ON","--
enforce_gtid_consistency=ON","--log_bin","--log_slave_updates=ON","--
master_info_repository=TABLE","--relay_log_info_repository=TABLE","--
transaction_write_set_extraction=XXHASH64","--user=mysql","--skip-host-cache","--
skip-name-resolve", "--default_authentication_plugin=mysql_native_password",
"--performance_schema_consumer_events_statements_history_long=ON",
"--performance_schema_events_statements_history_long_size=16000"]
```

Changed to use the EE docker image from Oracle Support

Turn on statement collection and increase SQL length captured for performance monitoring

Taking Docker Design Considerations into account

- Persist the database
- Secure files from prying eye
- Limit impact of noisy neighbors
- Central Authentication

Create Persisted Volumes

/var/lib/mysql is the data directory in the base image, it needs to be persisted outside of the container

This is done via

volumes:

-[host path]:[container path substituted]

We create a directory per MySQL instance off of **~/Docker/PersistedMounts**

Additional Mounts

Two additional volumes are made available:

A shared alias to for Enterprise RPM's and scripts

~/Docker/SharedSource:/usr1/shared

A common directory for Enterprise Backup

~/Docker/SharedBackup:/usr1/backup

Docker-compose.yml Edits –Persisting Data

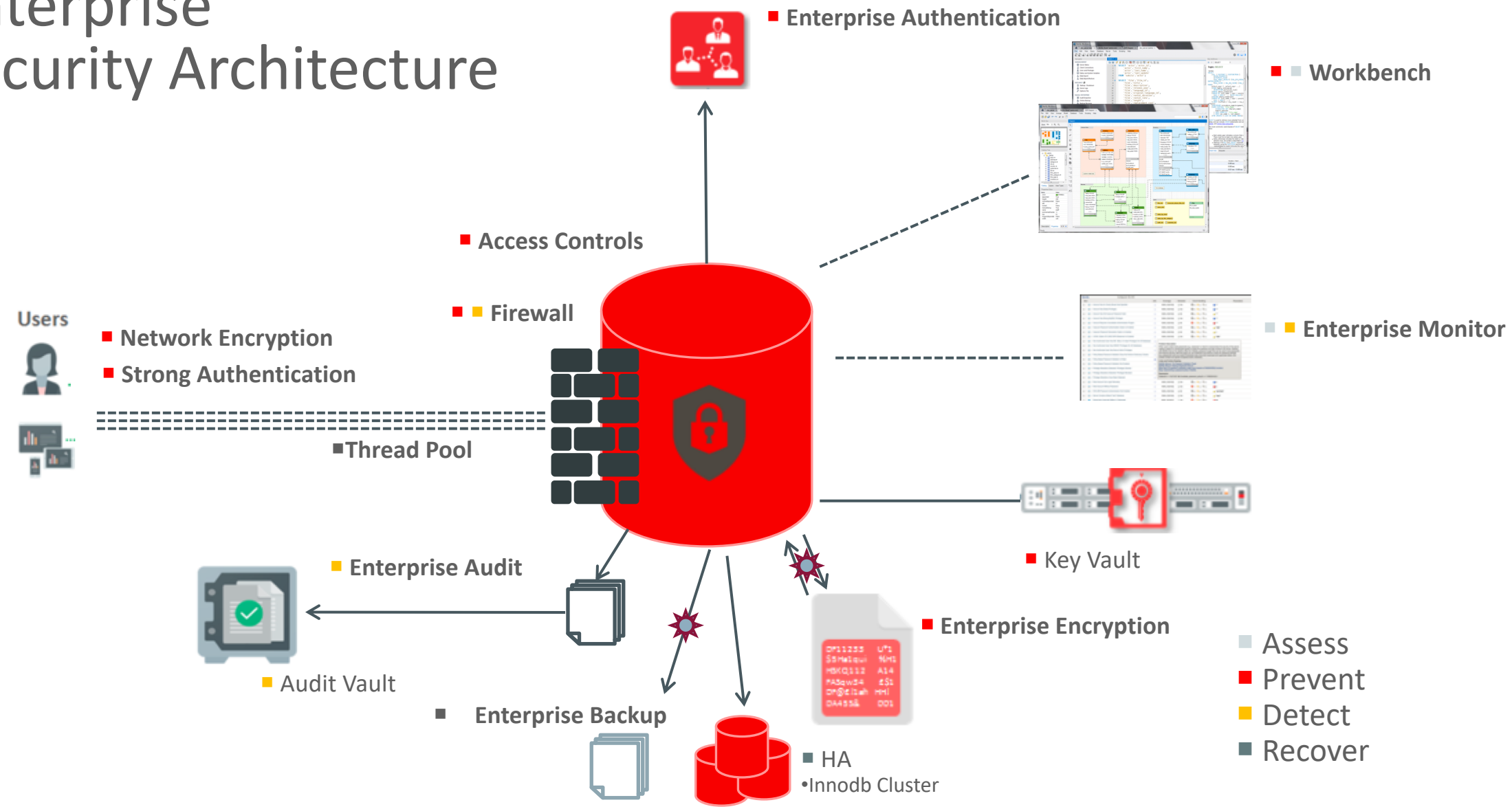
```
image: mysql/mysql-server:8.0.16
ports:
      - "3301:3306"
command: ["mysqld","--server_id=1","--binlog_checksum=NONE","--gtid_mode=ON","--
enforce_gtid_consistency=ON","--log_bin","--log_slave_updates=ON","--
master_info_repository=TABLE","--relay_log_info_repository=TABLE","--
transaction_write_set_extraction=XXHASH64","--user=mysql","--skip-host-cache","--
skip-name-resolve", "--default_authentication_plugin=mysql_native_password",
"--performance_schema_consumer_events_statements_history_long=ON",
"--performance_schema_events_statements_history_long_size=16000"]
volumes:
- /opt/docker/mysql-server-1:/var/lib/mysql
```

We've added a Docker Volumes identifier mapping the container /var/lib/mysql/ path to a mountpoint referring to /opt/docker[server-name] on the container's host

Securing MySQL in a Docker Container

Data is the Crown Jewel –Defend It

Enterprise Security Architecture



Back to Database Security –We can't just rely on others

- How do we survive Denial of Service attack
- How do we secure database files that are persisted elsewhere
- How do we harden our SQL
- How do we prove our databases are compliant with regulations

Security Requirement to Resolution Mapping

Requirement	Resolution
DoS Survival	Thread Pool
Secure Persisted Data	Transparent Data Encryption
Login Management	Authentication and LDAP
Answering Auditors	Audit Log

This in addition to online backups and monitoring

MySQL Enterprise Edition

The Roman Legion, All from the Fountainhead, Designed to Work Together!

Advanced Features

- Scalability
- Authentication
- Audit
- Encryption + TDE
- Firewall
- De-identification
- Data Masking

Management Tools

- Monitoring
- Backup
- Development
- Administration
- Migration

Support

- Technical Support
- Consultative Support
- Oracle Certifications

Hardening MySQL on Docker

- Modify docker-compose.yml to load required plugins
- Execute any installation SQL installation scripts
- Create the InnoDB Cluster

Docker Compose Changes –Adding Plugins

```
...
command: ["mysqld","--server_id=1","--binlog_checksum=NONE","--gtid_mode=ON","--enforce_gtid_consistency=ON","--log_bin","--log_slave_updates=ON","--master_info_repository=TABLE","--relay_log_info_repository=TABLE","--transaction_write_set_extraction=XXHASH64","--user=mysql","--skip-host-cache","--skip-name-resolve", "--default_authentication_plugin=mysql_native_password",
"--performance_schema_consumer_events_statements_history_long=ON",
"--performance_schema_events_statements_history_long_size=16000”, ”--early-plugin-load=keyring_file.so", "--keyring_file_data=/var/lib/mysql-keyring/keyring", "--plugin-load-add=thread_pool.so;audit_log.so"
]

...
```

We’re loading TDE (**keyring.so)** early, and setting the **keyring_file_data** directory

Lastly, we add Thread Pool (**thread_pool.so**) and Audit Logging (**audit_log.so**

Docker Compose Changes -Plugin Installation SQL

Adjust the SQL script called from **docker-compose.yml**

The script is **~/Docker/docker-idbc-master/innodb-cluster/scripts/db.sql**

Append for additional installation scripts

```
source /usr/share/mysql-8.0/audit_log_filter_linux_install.sql;
```

Create and start the stack

From **~/Docker/docker-idbc-master/innodb-cluster/**

docker-compose up

In about 90 second, you'll see the process complete and use your tool of choice to connect to the Cluster

Monitoring Docker Deployments

Quality and Performance at Industrial Scale

Monitoring

- Docker commandline
- Docker GUI's (I use Portainer, ctop and SimpleDockerUI for my examples)
- MySQL Shell
- MySQL Enterprise Monitor (MEM)

Configuring MEM

- Recall our Network
  - Servers are mapped to localhost
  - Instances are Mapped to ports on the localhost

- To add our docker instances to MEM, we need a string to bulk load them into MEM

  127.0.0.1:3301, 127.0.0.1:3302,127.0.0.1:3303

- We'll use that to bulk load instances into MEM

MEM Demo

Wrapping up

- MySQL is up to the task of running InnoDB Cluster on Docker containers
- Docker introduces design constraints
- MySQL needs to be secured mitigate Docker risk profile
- EE has the tools to secure MySQL/Docker
- Monitoring is a critical aspect of MySQL/Docker deployments that MySQL Enterprise Edition solves

Questions

MySQL Enterprise Edition On Docker

John Kehoe, OCP, OCP MySQL NDB Cluster, OCI Associate Architect
Principal Solution Architect

John.Kehoe@oracle.com
+1.312.651.8756


ORACLE®