ORACLE

MySQL

MySQL Database Architectures: High Availability and Disaster Recovery Solutions

WEBINAR 7 Novembre 2024

Emmanuel COLUSSI

MySQL Solution Architect EMEA

Emmanuel COLUSSI

- MySQL Solution Architect at Oracle
- 25 years of experience in designing and building complex Hight Availibility solutions across industries such as health, telecommunications, insurance, finance, and defense. Specializes in cloud-native architecture, Kubernetes deployment, and enjoys programming in Go
- My workshops at Github: https://github.com/colussim
- Let’s stay in touch:
  - https://www.linkedin.com/in/emmanuel-colussi-b8b29113/

Agenda

- Presentation MySQL Enterprise Edition
- Qu'est-ce que la Haute Disponibilité ?
- Composants Clés pour une Architecture HA avec MySQL
- Démonstration

MySQL Enterprise Edition

Advanced Features

- Scalability
- High Availability
- Authentication
- Audit
- Encryption + TDE
- Firewall
- Masking

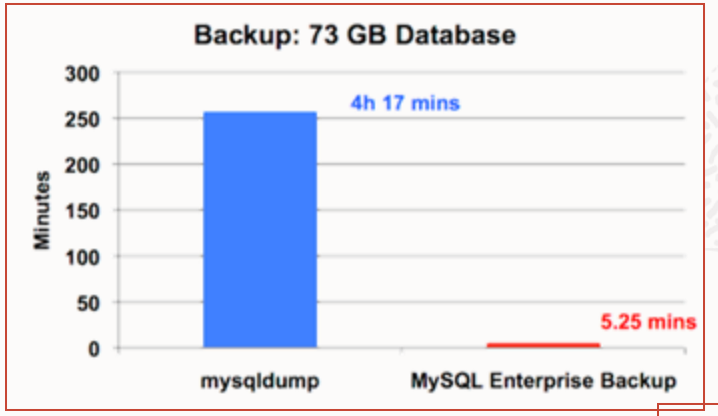
Management Tools

- Monitoring
- Backup
- Development
- Administration
- Migration

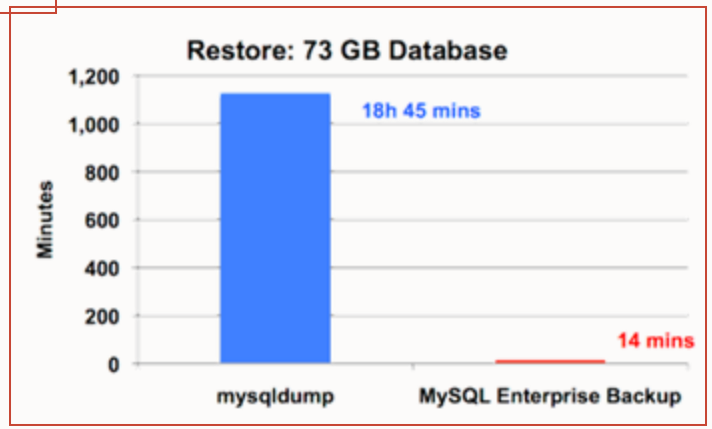
Support

- Technical Support
- Oracle Certifications

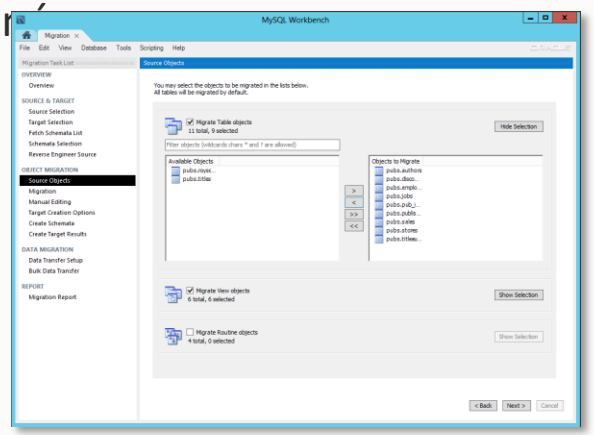
MySQL Commercial tools



- MySQL Enterprise Backup
  - best performances
  - Sauvegarde et récupération en ligne, sans verrouillage
  - Fonctionnalités avancées
    - Sauvegardes incrémentales/différentielles, compression, cryptage …
  - Integration avec MMS



- MySQL Workbench Enterprise
  - Développement, conception et documentation de bases de donn
  - Database migrations
  - Database Administration
    - Performance Tools, EE plugins wizards…



Qu'est-ce que la Haute Disponibilité ?

Réduire ou éliminer le temps d’arrêt

Indicateurs de mesure communs

Concepts – RTO & RPO

✅ Objectif de délai de recuperation (RTO)

- Temps total nécessaire pour terminer une réparation et reconnecter un système.

✅ Objectif de point de recuperation (RPO)

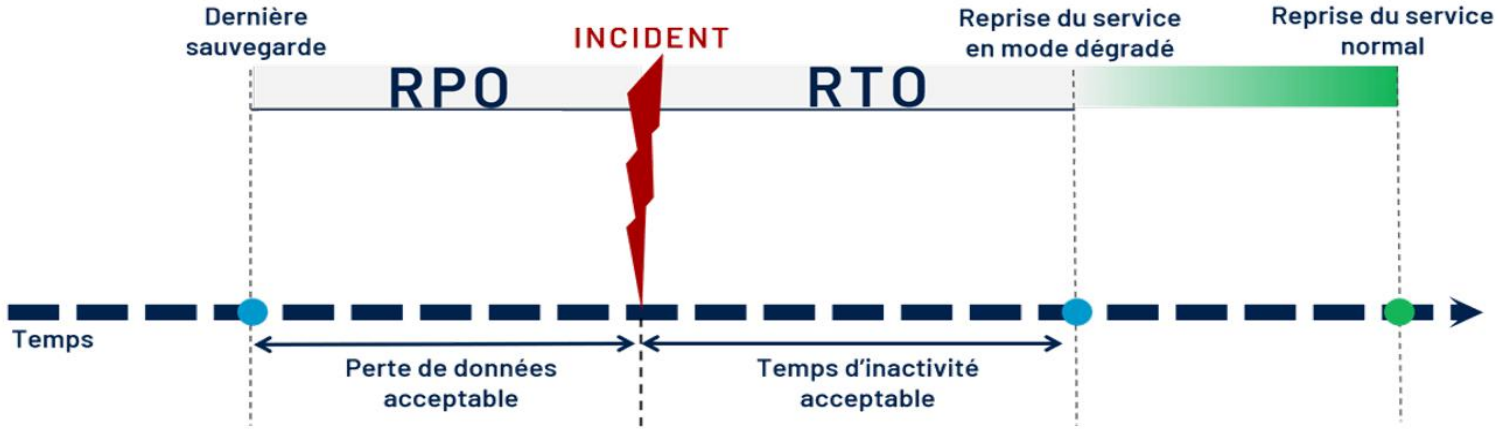
- Moment à partir duquel vous avez besoin de récupérer les données. Il s'agit de la fenêtre des données perdues.

Types d'échec

High Availability: Défaillance d'un seul serveur, partition du réseau

Disaster Recovery: Défaillance totale de la région/du réseau

Human Error: Erreurs DBA, Injection SQL, ...

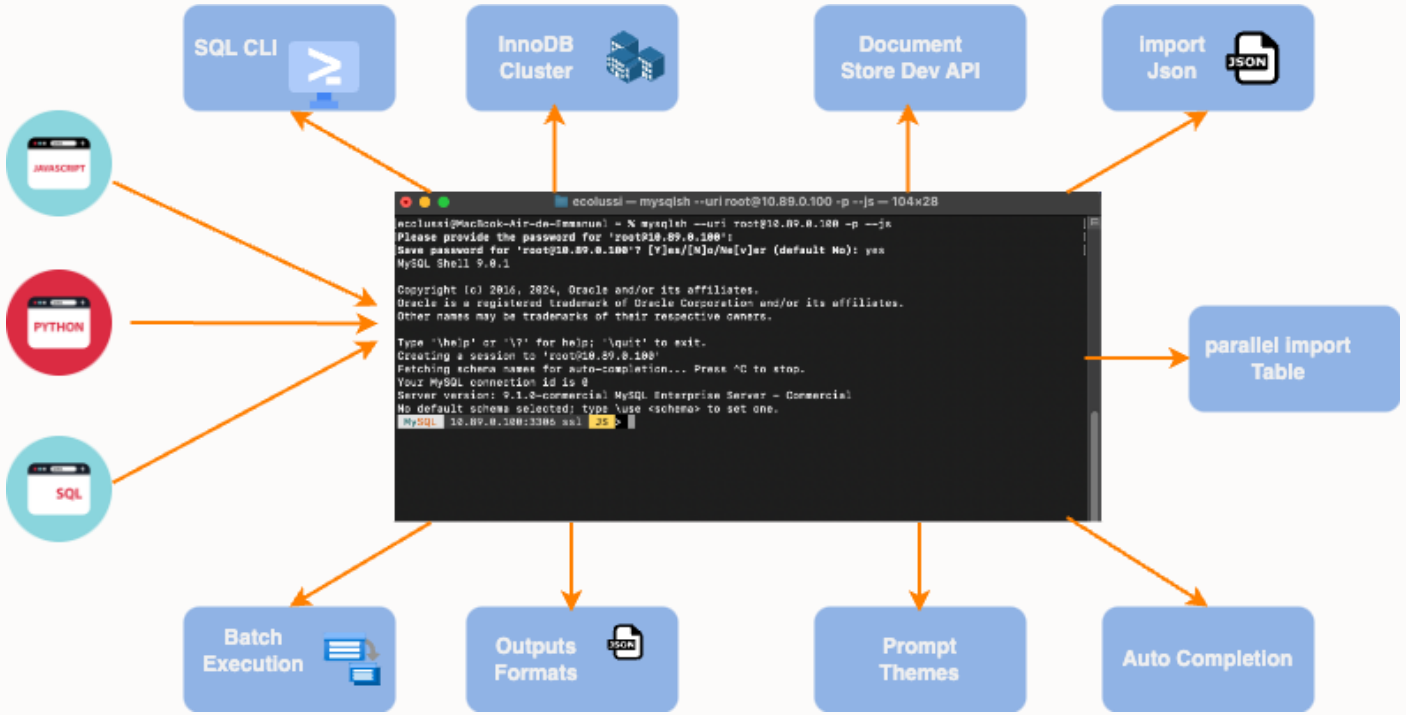


Composants Clés pour une Architecture HA avec MySQL

MySQL Shell

MySQL Shell est un environnement interactif et puissant pour travailler avec MySQL.

Il offre plusieurs modes d'utilisation, notamment le mode SQL, le mode JavaScript et le mode Python, permettant aux développeurs et aux administrateurs de bases de données d'exécuter des scripts et des requêtes de manière efficace.



MySQL Shell

Fonctionnalités :

✅ **Support multi-langs** : Capacité d'exécuter des scripts en SQL, JavaScript ou Python. Cela offre une flexibilité pour les développeurs et les administrateurs selon leurs préférences.

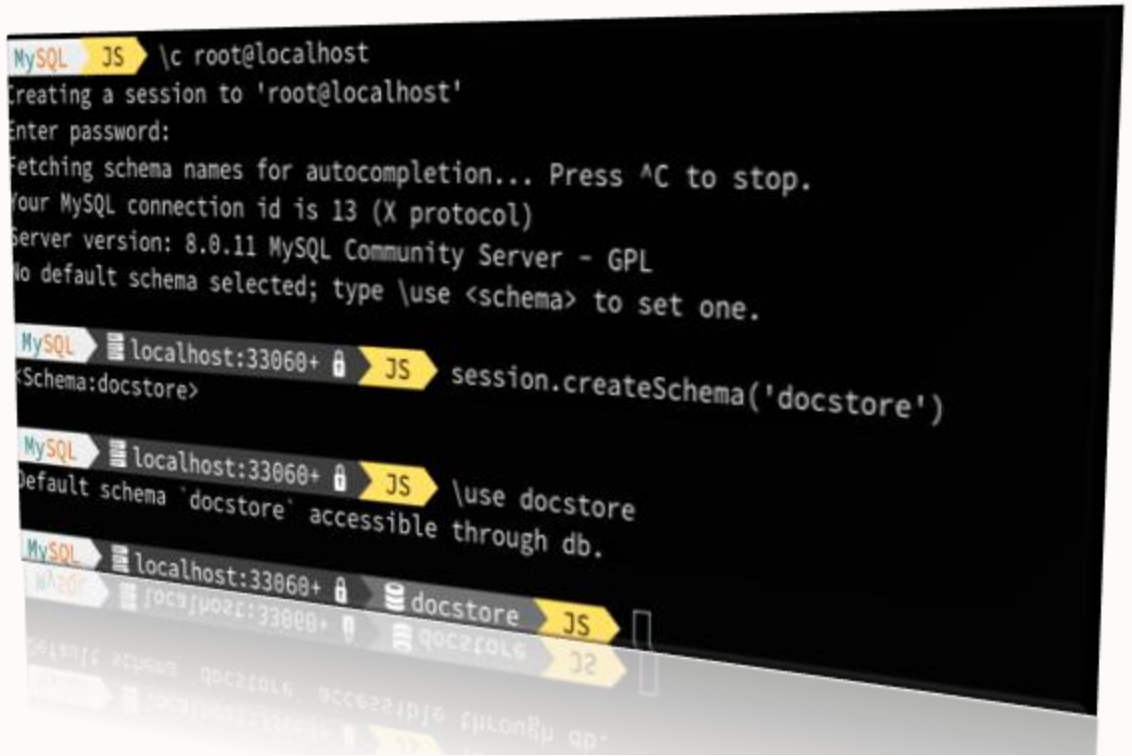
✅ **Automatisation** : Permet l'automatisation des tâches courantes de gestion de bases de données, telles que les sauvegardes, les récupérations, et le déploiement d'InnoDB Cluster.

✅ **Intégration avec MySQL Server** : MySQL Shell se connecte facilement aux instances MySQL, facilitant l'exécution de commandes et la gestion des configurations du serveur.

Avantages :

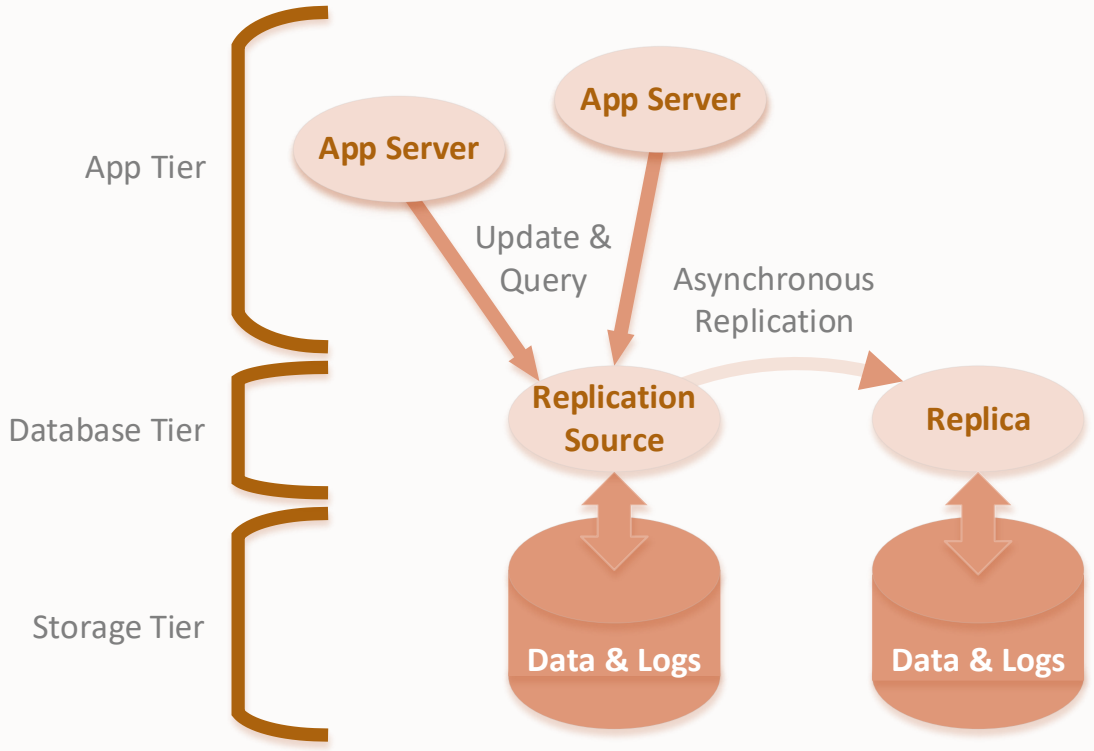
✅ **Productivité accrue** grâce à un environnement interactif avec auto-complétion.

✅ **Facilité de gestion** des clusters MySQL via des commandes simplifiées.



MySQL Replication

La réplication traditionnelle améliore la disponibilité



Génial ! 😀

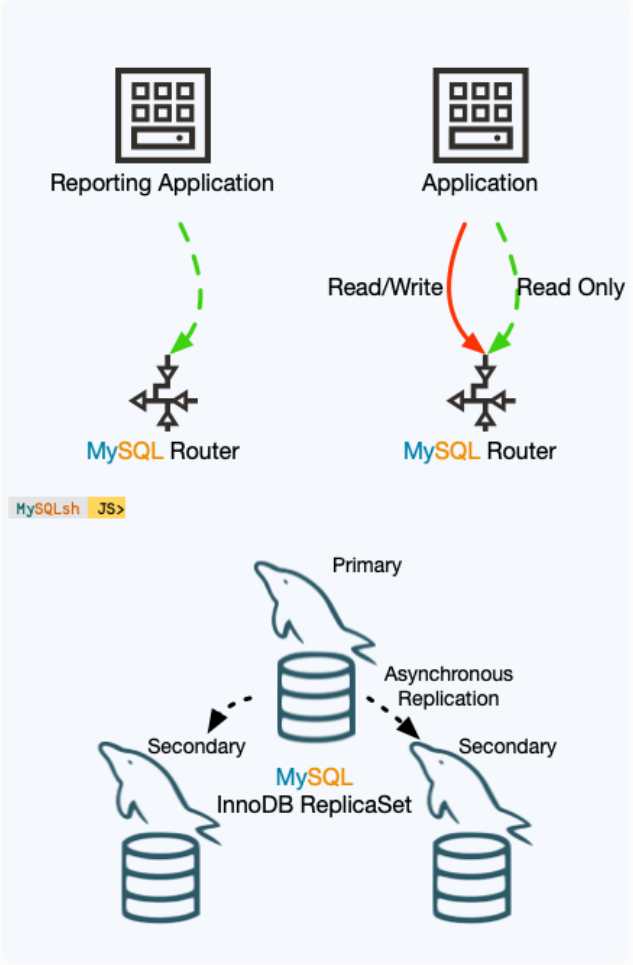
- ✅ Les données sont protégées.
- ✅ La source tombe en panne, mettez la réplique en marche.
- ✅ Vous avez sauvé la situation ! 😀

Mais ce n'est pas fluide. 😥

- ✅ Failover manuel vers la réplique.
- ✅ Reconfigurer manuellement les serveurs d'application vers la réplique.
- ✅ Il peut y avoir une e perte de données avec la réplication asynchrone.

C'est davantage une solution de récupération après sinistre.

MySQL InnoDB ReplicaSet



Vanilla Replication Solution

- Introduit dans MySQL 8.0.20
- Extensibilité des opérations de lecture
- Failover manuel

- MySQL Shell
- MySQL Router
- MySQL Server

```
RPO != 0
RTO = minutes ou plus (failover namuel)
```

Créer un MySQL ReplicaSet

Commandes pour configurer un MySQL ReplicaSet:

```
mysqlsh> dba.configureReplicaSetInstance(”root@localhost:3311")

mysqlsh> \c root@localhost:3311
mysqlsh> rs = dba.createReplicaSet(”MyReplicaSet”)

mysqlsh> rs.addInstance('localhost:3312')
mysqlsh> rs.addInstance('localhost:3313')
mysqlsh> rs.status()

Définissez l'instance MySQL 3312 comme primaire :
mysqlsh> rs.setPrimaryInstance(”localhost:3312")
```

MySQL InnoDB ReplicaSet

Principales fonctionnalités et composants de l'InnoDB ReplicaSet :

1. **Architecture primaire-secondaire** : Un seul serveur est configuré comme primaire (master), recevant toutes les écritures, tandis que les autres sont des réplicas (ou secondaires) qui synchronisent les données depuis le serveur primaire.

2. **Répartition des lectures** : Les lectures peuvent être distribuées entre le serveur primaire et les serveurs secondaires, ce qui permet de mieux répartir la charge et d'améliorer les performances de lecture.

3. **Automatisation de la réplication** : Avec l'outil MySQL Shell, on peut facilement configurer et gérer l'InnoDB ReplicaSet, en automatisant les tâches de configuration et de gestion des membres du ReplicaSet, y compris l'ajout ou le retrait de réplicas.

4. **Gestion des basculements manuels** : En cas de défaillance du serveur primaire, un basculement manuel peut être initié pour promouvoir un serveur secondaire comme nouveau primaire. Contrairement à l'InnoDB Cluster, ce processus n'est pas automatique dans InnoDB ReplicaSet, ce qui le rend mieux adapté aux environnements qui peuvent tolérer un certain temps d'arrêt.

5. **Surveillance** : Le MySQL Shell offre des commandes pour surveiller l'état de la réplication, la latence, et les déconnexions potentielles des réplicas. Ces outils permettent aux administrateurs de suivre l'intégrité et la performance du ReplicaSet.

6. **Configuration simplifiée** : Il ne nécessite pas le consensus ni les mécanismes de verrouillage stricts de Group Replication. Il est donc idéal pour les cas d'utilisation où une réplication simple et une tolérance de panne manuelle suffisent.

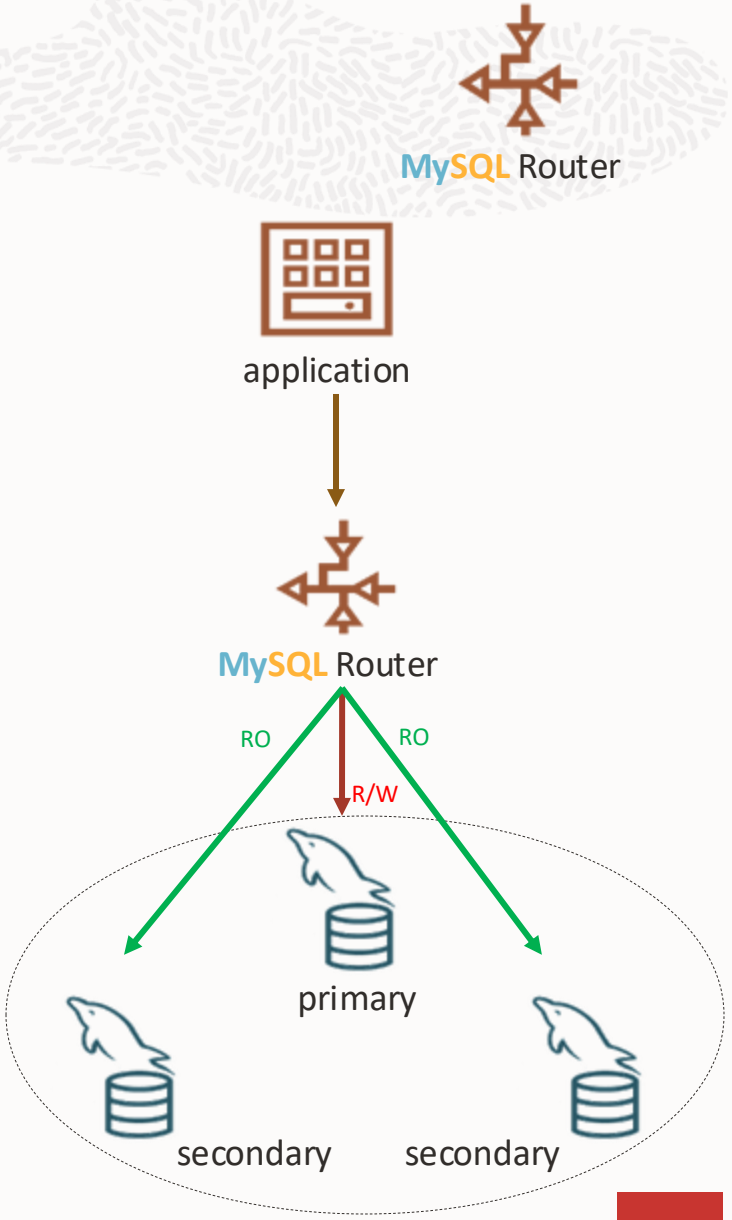
MySQL Router

MySQL Router est un logiciel léger qui sert de point d'entrée pour les applications lorsqu'elles se connectent aux instances

MySQL. Il gère le routage des connexions vers les nœuds appropriés dans un environnement de haute disponibilité.

MySQL Router propose 3 ports

- Connexion au **nœud en lecture/écriture** (6446)
- Connexion au **nœud en lecture seule** (6447)
- **Répartition en Round robin**
- **Répartition transparente des lectures/écritures** (6450)
- Dirige tout le trafic de lecture vers les instances en lecture seule et tout le trafic d'écriture vers les instances en lecture/écriture
- Chaque session client peut communiquer avec une destination en lecture/écriture et une destination en lecture seule



MySQL Router

Fonctionnalités :

✅ **Routage basé sur les rôles** : Dirige intelligemment les requêtes vers le nœud maître pour les écritures et les nœuds esclaves pour les lectures. Cela permet d'optimiser les performances en répartissant la charge.

✅ **Failover automatique** : En cas de défaillance d'un nœud, MySQL Router redirige automatiquement les connexions vers un autre nœud, assurant ainsi une continuité de service.

✅ **Configuration simple** : Facilité de configuration à l'aide de fichiers de configuration simples ou par le biais de MySQL Shell.

Avantages :

✅ **Haute disponibilité** : Garantit que les applications peuvent toujours se connecter à la base de données, même en cas de pannes.

✅ **Optimisation des performances** : Améliore le temps de réponse et réduit la charge sur le serveur maître.

MySQL Router: scénarios de déploiement

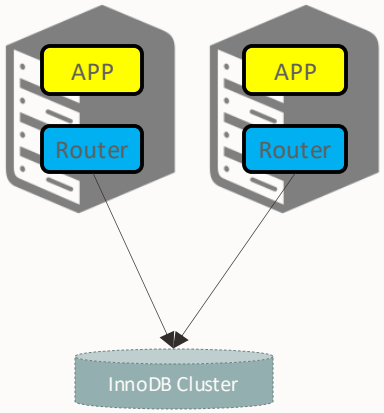
Un par serveur d'application

Avantages :

- Surcharge minimale
- Pas de saut réseau entre l'application et la base de données
- Deploiement facile de nouveau router

Inconvénients:

Le pare-feu entre les couches nécessiste une règle pour chaque serveur d'application



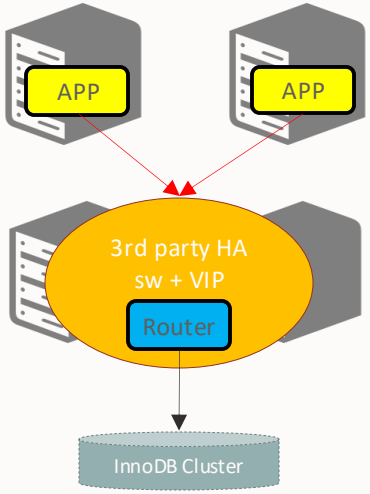
Cluster tiers sur serveurs dédiés

Avantages :

- Pas de point de défaillance unique
- Configuration facile avec les pare-feu

Inconvénients:

- Logiciel tiers requis
- Nécessite un saut réseau supplémentaire



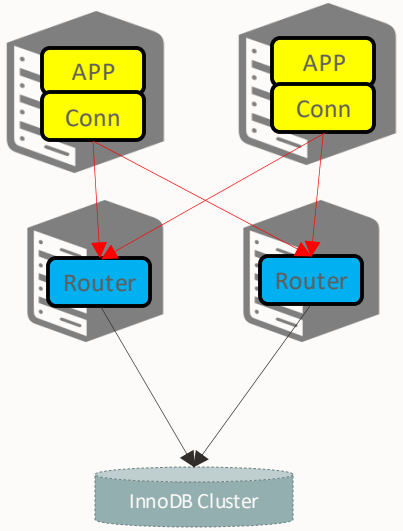
Serveurs dédiés + connecteurs avec haute disponibilité intégrée

Avantages :

- Pas besoin de logiciel tiers ni d'adresse IP virtuelle (VIP)

Inconvénients:

- Tous les connecteurs n'ont pas de capacités de haute disponibilité .
- Nécessite un saut réseau supplémentaire



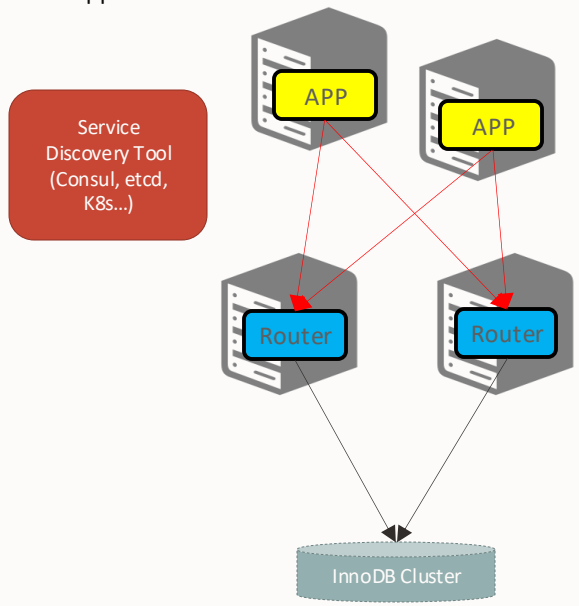
DNS/SRV

Avantages :

- Un seul domaine DNS peut pointer vers plusieurs cibles (serveurs)

Inconvénients:

- Nécessite un logiciel tiers
- Nécessite des connecteurs qui le supportent



MySQL InnoDB Cluster

La solution : Cluster MySQL InnoDB

Objectifs

Un seul produit : MySQL

Facile à utiliser

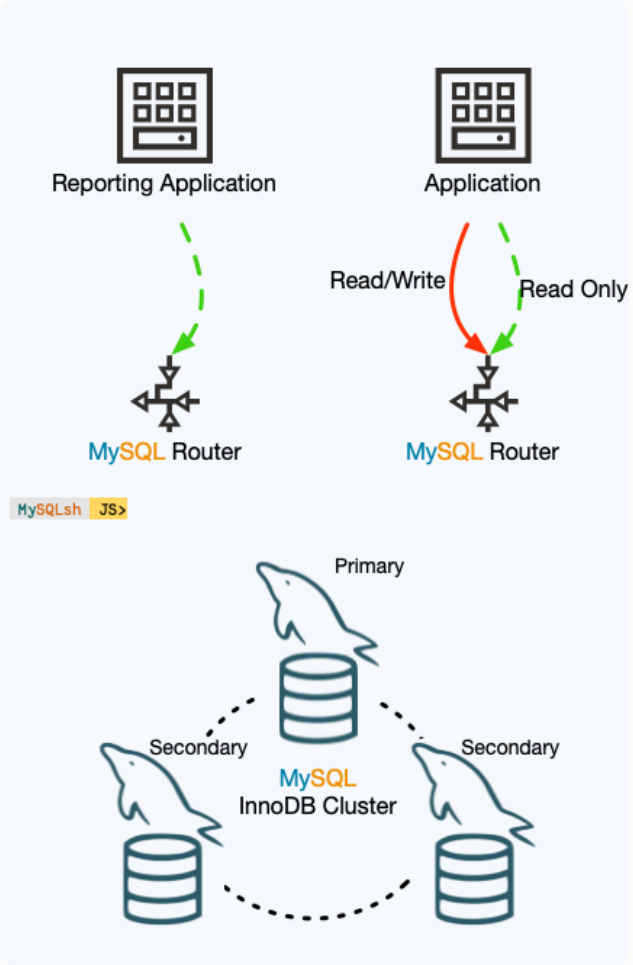
Un client – MySQL Shell:

MySQL Shell

Flexible et moderne

```
{ Job: done }
```

MySQL InnoDB Cluster



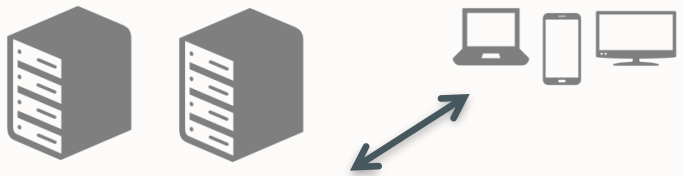
Solution de haute disponibilité basée sur la réplication de groupe, entièrement intégrée

- MySQL Shell
- MySQL Router
- MySQL Server
  - La replication de groupe fournie:
    - Failover automatique / Tolérance de panne
    - Changements automatiques de members
    - Gestion des partitions du réseau
    - Cohérence

RPO = 0
RTO = seconds (failover automatique)

InnoDB Cluster: les composants

- Acheminer la connexion de l'application vers les nœuds disponibled
- Configuration automatique



App Servers with
MySQL Router

MySQL Shell
Setup, Manage, Orchestrate

- Automatiser la création et le fonctionnement du cluster
- Scriptable (Javascript, Python, SQL)

- Fournit une réplication virtuellement synchrone
- Automatisation des opérations
  - Détection et résolution des conflits
  - Détection des pannes, basculement, récupération
  - Gestion et création de groupes

MySQL Server avec Group Replication plugin

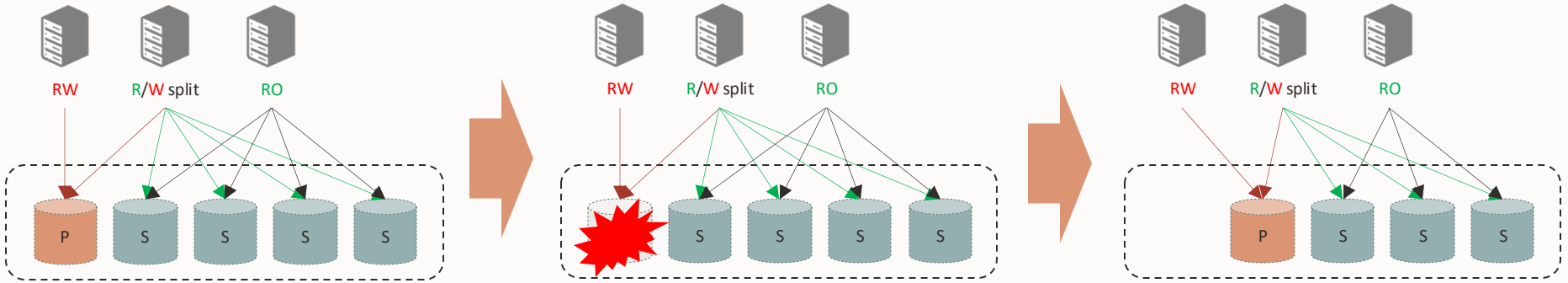
Disponible sur toutes les plateformes prises en charge par MySQL

MySQL Group Replication: Single-Primary Mode

- One Une instance ouverte en lecture/écriture (primaire) ,tous les autres s'ouvrent en lecture seule (secondaires)
- Les ports de lecture/écriture du routeur se connectent toujours à une instance primaire.
  - 6446 est la valeur par défaut pour le mode lecture / ecriture
  - 6448 est la Valeur par default pour les appels d'API en lecture-écriture du client MySQL vers le routeur MySQL

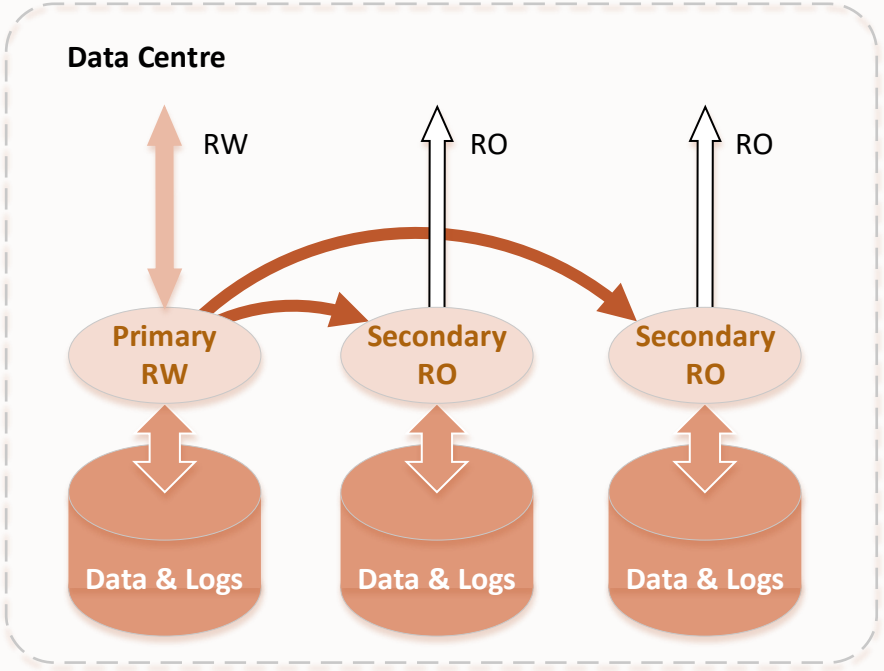
- Les ports en lecture seule du routeur se connectent toujours à une instance secondaire, de manière circulaire.
  - 6447 est la valeur par défaut pour le mode lecture
  - 6449 est la Valeur par default pour les appels en lecture seule du client MySQL vers le routeur MySQL

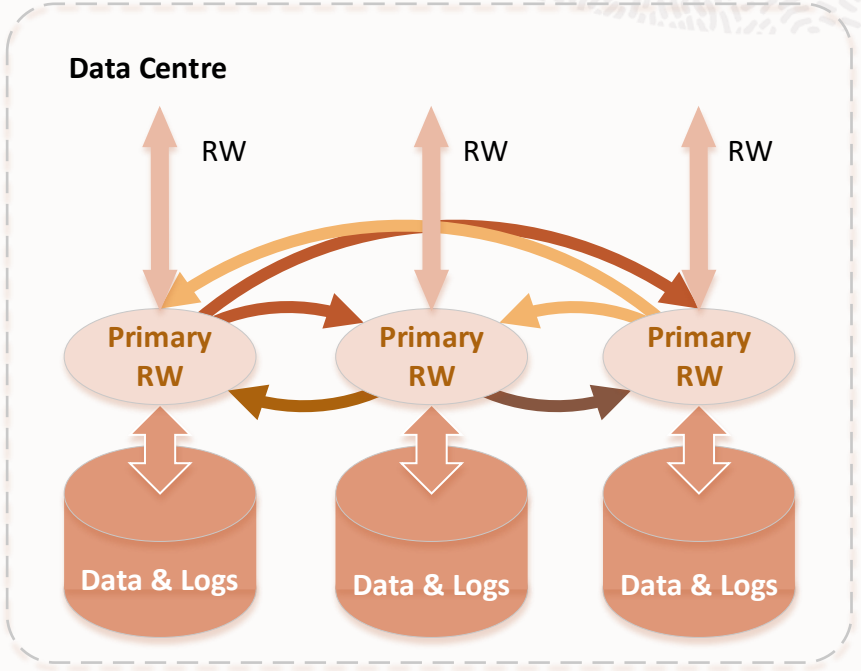
- Router Read/Write split pour diriger tout le trafic de lecture vers les instances en lecture seule et tout le trafic d'écriture vers les instances en lecture-écriture.
  - 6450 est la valeur par défaut pour le mode mode lecture / écriture split



InnoDB Cluster Topologies

Configurations à un et plusieurs nœuds principaux.





Topologie à un seul nœud principal par défaut

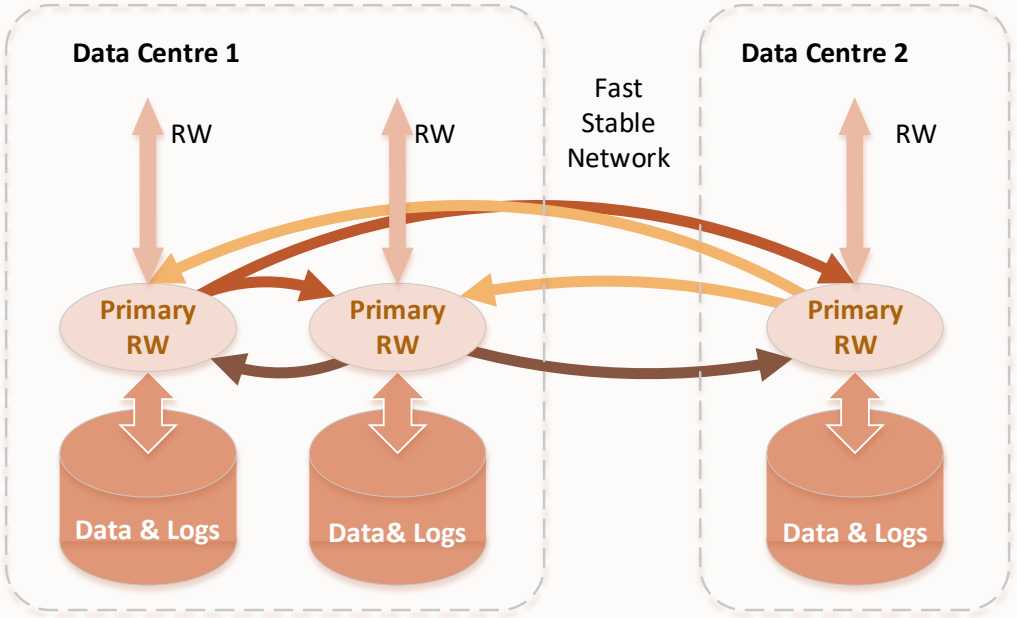
- 3, 5, 7 or 9 noeuds
- Le cluster reste résilient après 1, 2, 3 ou 4 pannes respectivement

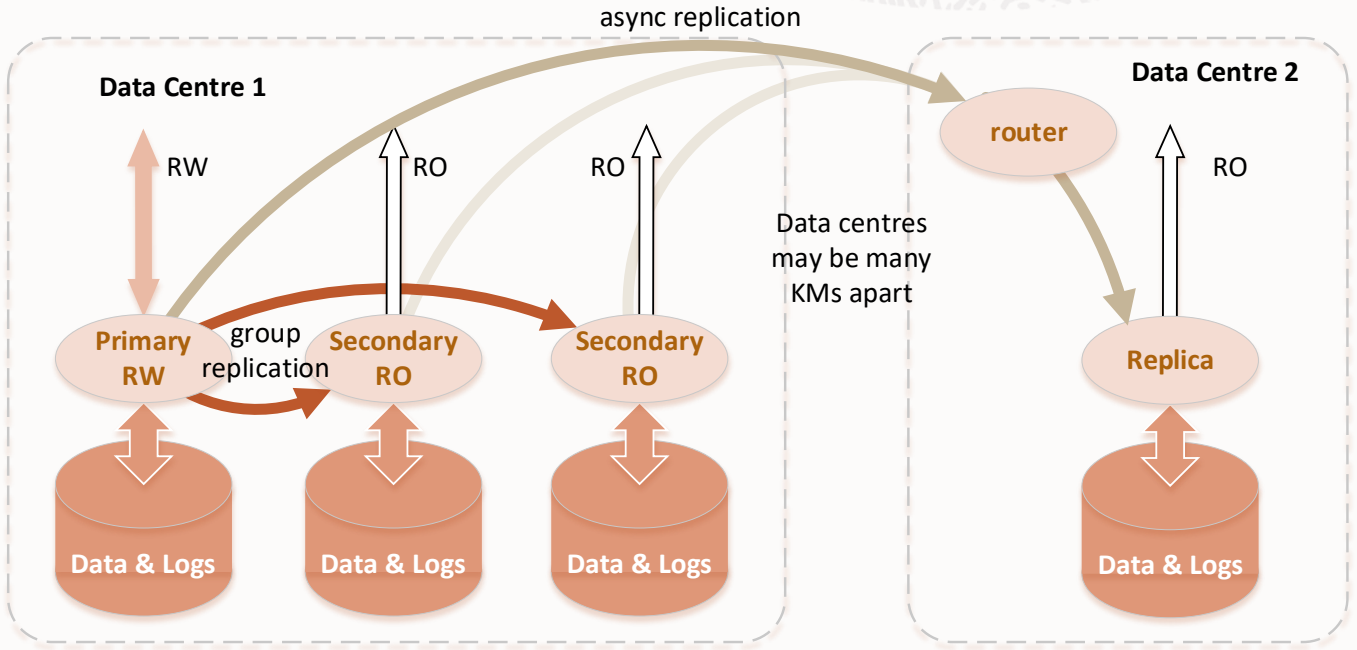
Topologie Multi-Primaire

- Tous les moeuds en lecture / écriture

InnoDB Cluster Topologies

Clusters étendus et géographiques





Cluster étendu – Multi-Primary

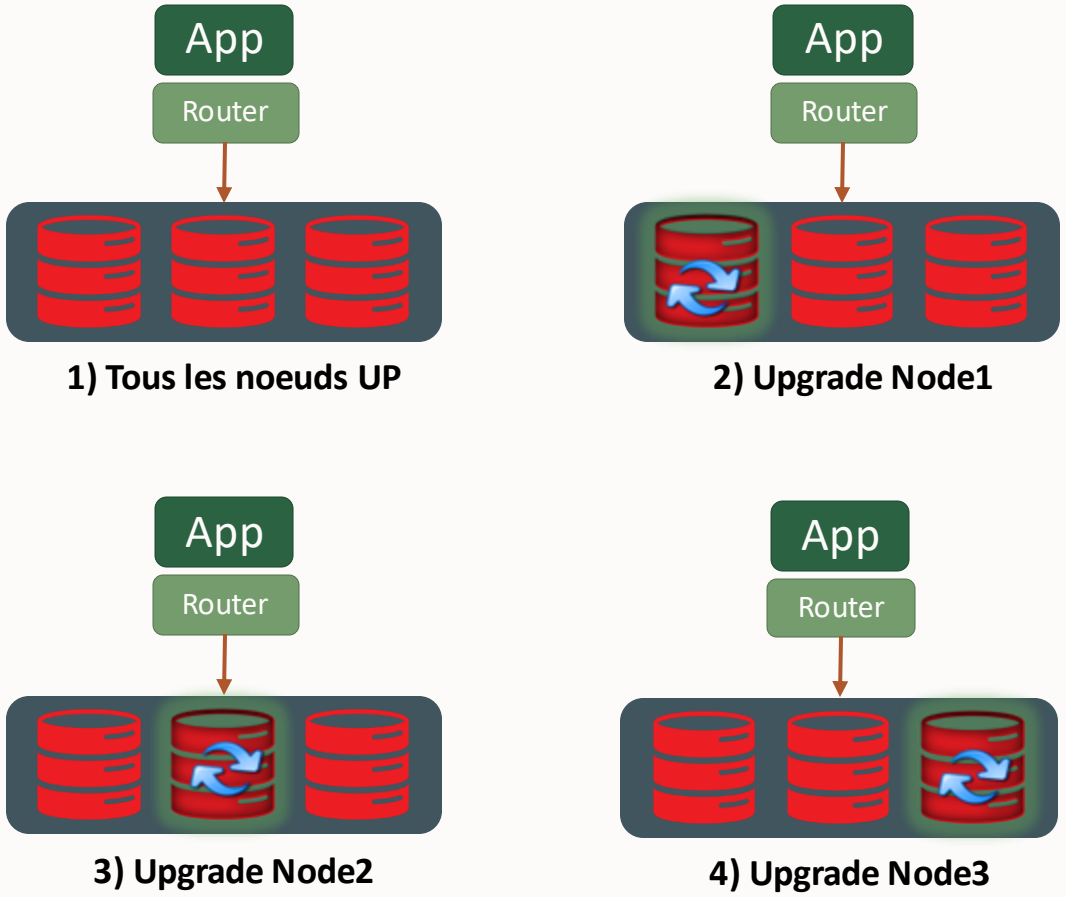
- Un réseau rapide et stable est requis

Geographic DR solution

- Le routeur agit en tant que proxy de réplication pour le cluster.
- La réplication asynchrone est basée sur GTID. Si le primaire échoue, la réplication asynchrone reprendra sur le nouveau primaire là où elle s'était arrêtée.
- Une instance distante (réplique) pourrait être utilisée pour le reporting.

Mise à niveau en ligne d'InnoDB Cluster (OS/MySQL/HW)

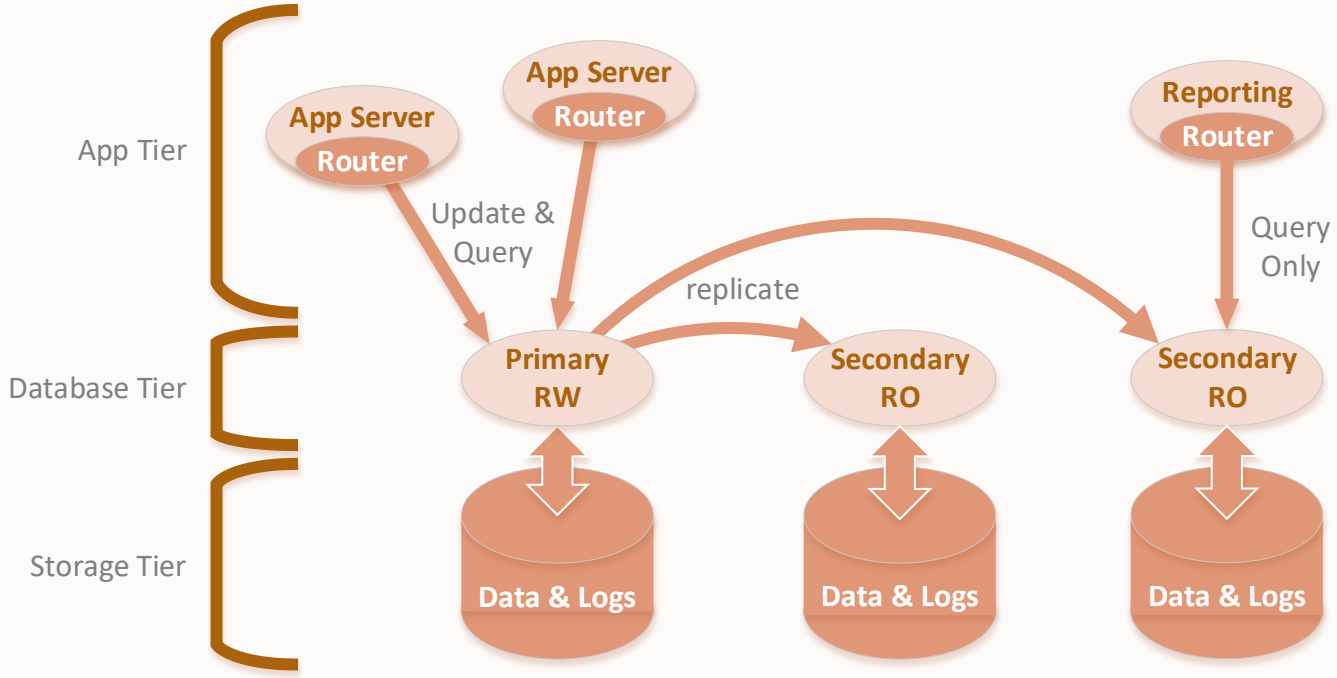
- Mettre à niveau le système d'exploitation, le matériel ou le logiciel sans temps d'arrêt pour l'application.
- **La mise à niveau en ligne** est possible en effectuant une mise à niveau progressive sur chaque serveur, en commençant par les secondaires et en terminant par le primaire :
  - Stop mysql: systemctl stop mysqld
  - Upgrade binaries: yum update mysql
  - Start MySQL: systemctl start mysqld



Failover and Recovery

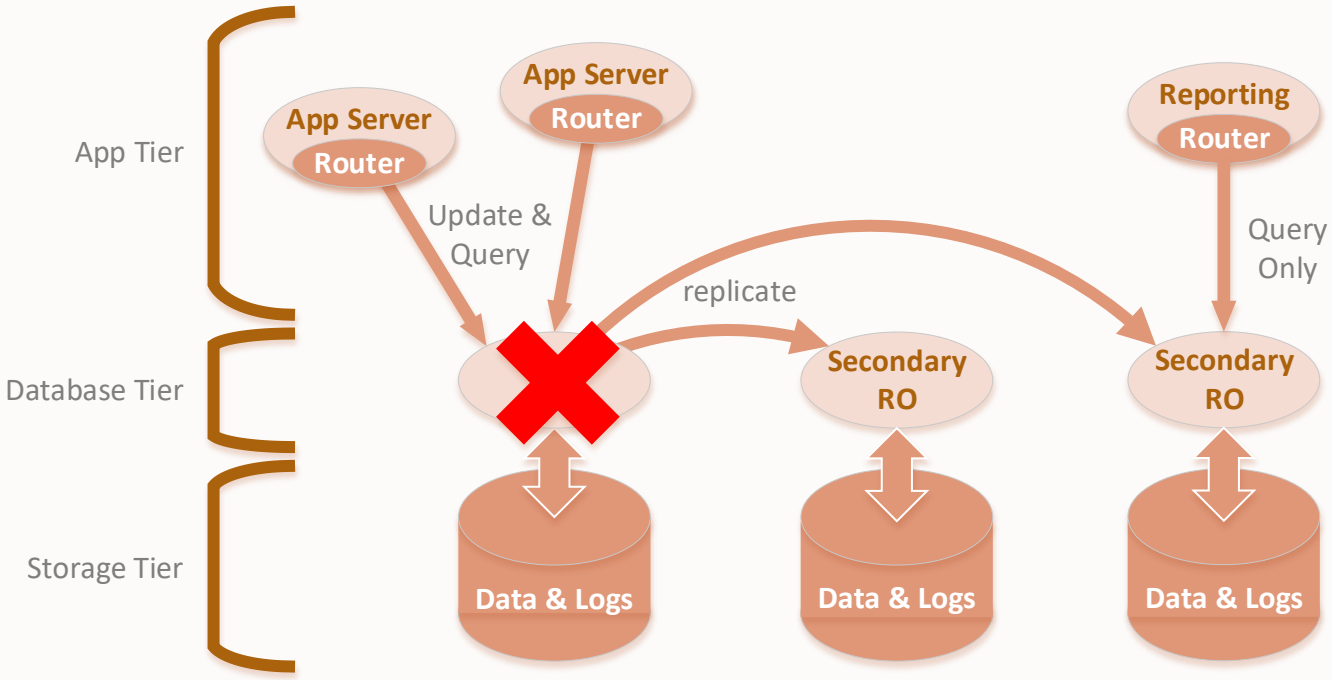
Failover and Recovery

Topologie standard à un seul nœud principal



Failover and Recovery

Topologie standard à un seul nœud principal

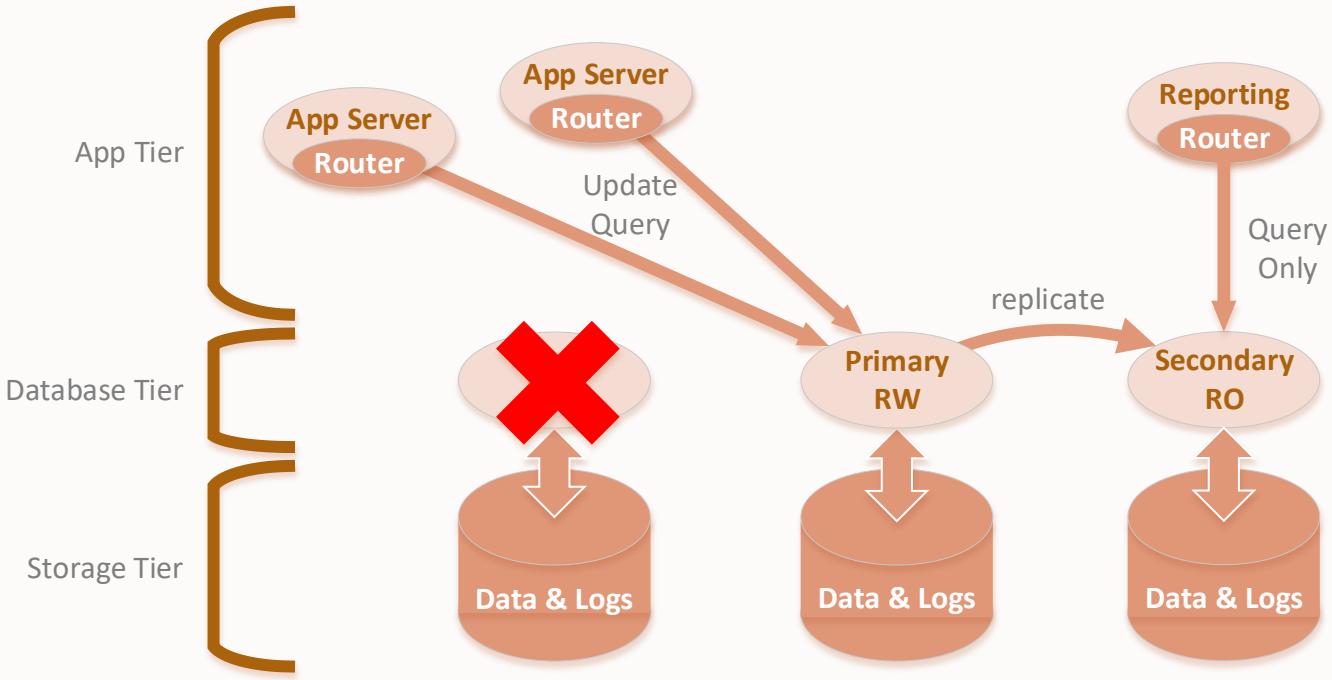


Primary goes down

- The worst case!
- Remaining 2 nodes can form quorum
- Decision made to failover
- Nodes elect a new master
- Decision can be influenced with weighting

Failover and Recovery

Topologie standard à un seul nœud principal

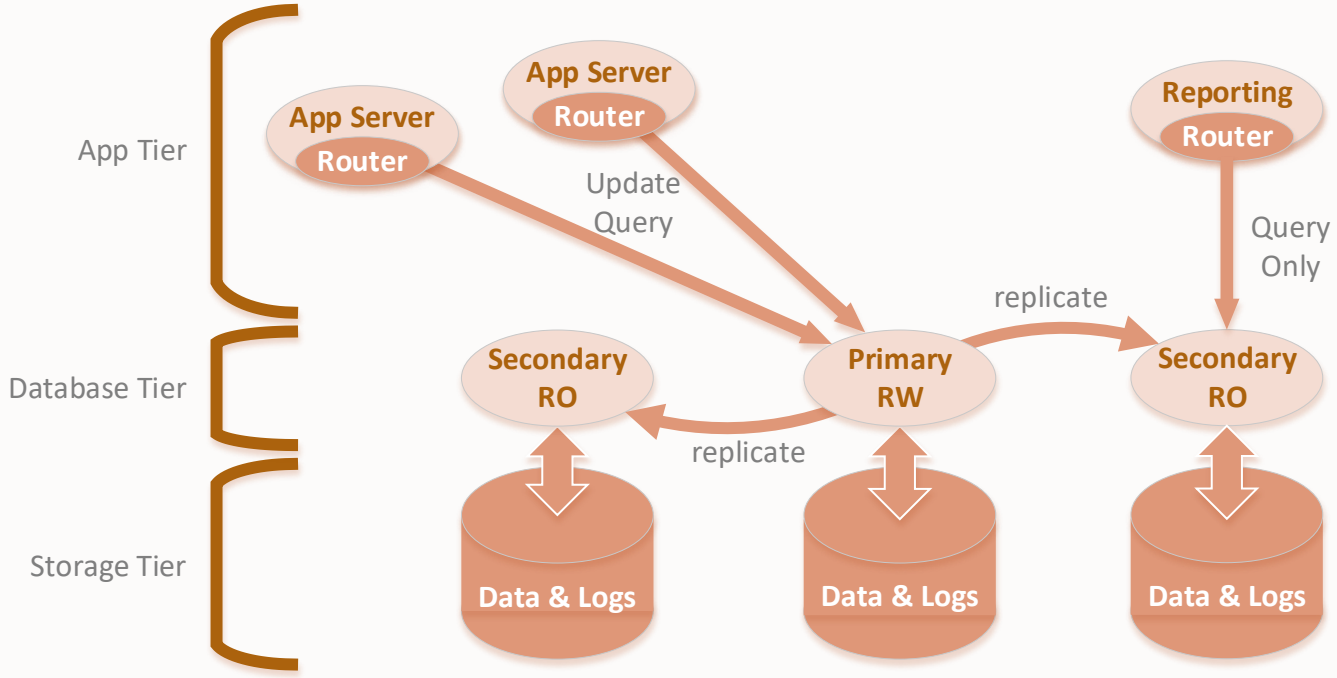


Le basculement se produit

- Les routeurs basculent automatiquement vers le nouveau primaire
- Aucun script n'est nécessaire.

Failover and Recovery

Topologie standard à un seul nœud principal



✅ Réintègre le nœud dans le cluster en tant que secondaire

✅ Applique les données en retard

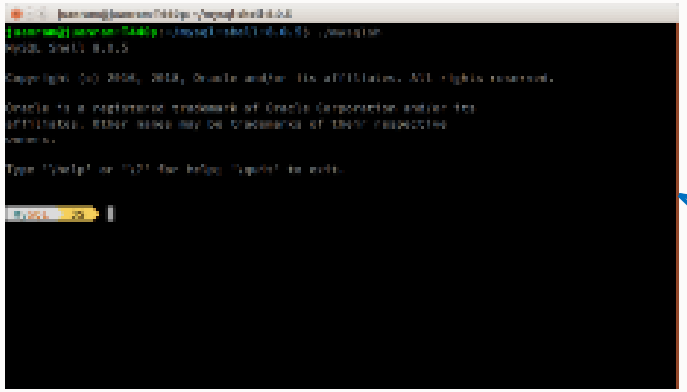
✅ Devient cohérent

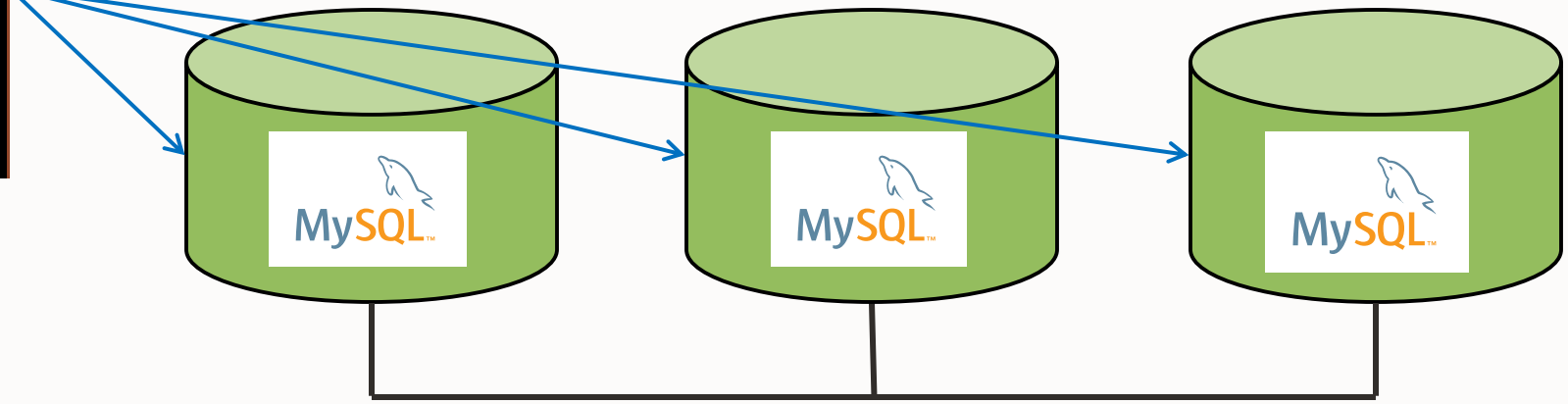
cluster.setPrimaryInstance()

Installation

MySQL Shell: Cluster setup

- Configurer et créer un cluster en utilisant MySQL Shell et 3 instances MySQL
  - Pas de stockage partagé





MySQL InnoDB Cluster: minimal setup

1. Vérifiez la configuration de chaque instance:
   - `MySQL JS > dba.checkInstanceConfiguration('root@host1')`
2. Si ce n'est pas correct, préparez les instances pour la réplication de groupe :
   - `MySQL JS > dba.configureInstance('root@host1')`
3. Connectez-vous à la première instance (le primaire):
   - `MySQL JS > \c root@host1`
4. Créer un cluster:
   - `MySQL host1 … JS > cluster=dba.createCluster('prod_cluster');`

5/6. Ajouter tous les autres nœuds avec la méthode de cluster :

- `MySQL … JS > cluster.addInstance('host2')`
- `MySQL … JS > cluster.addInstance('host3')`

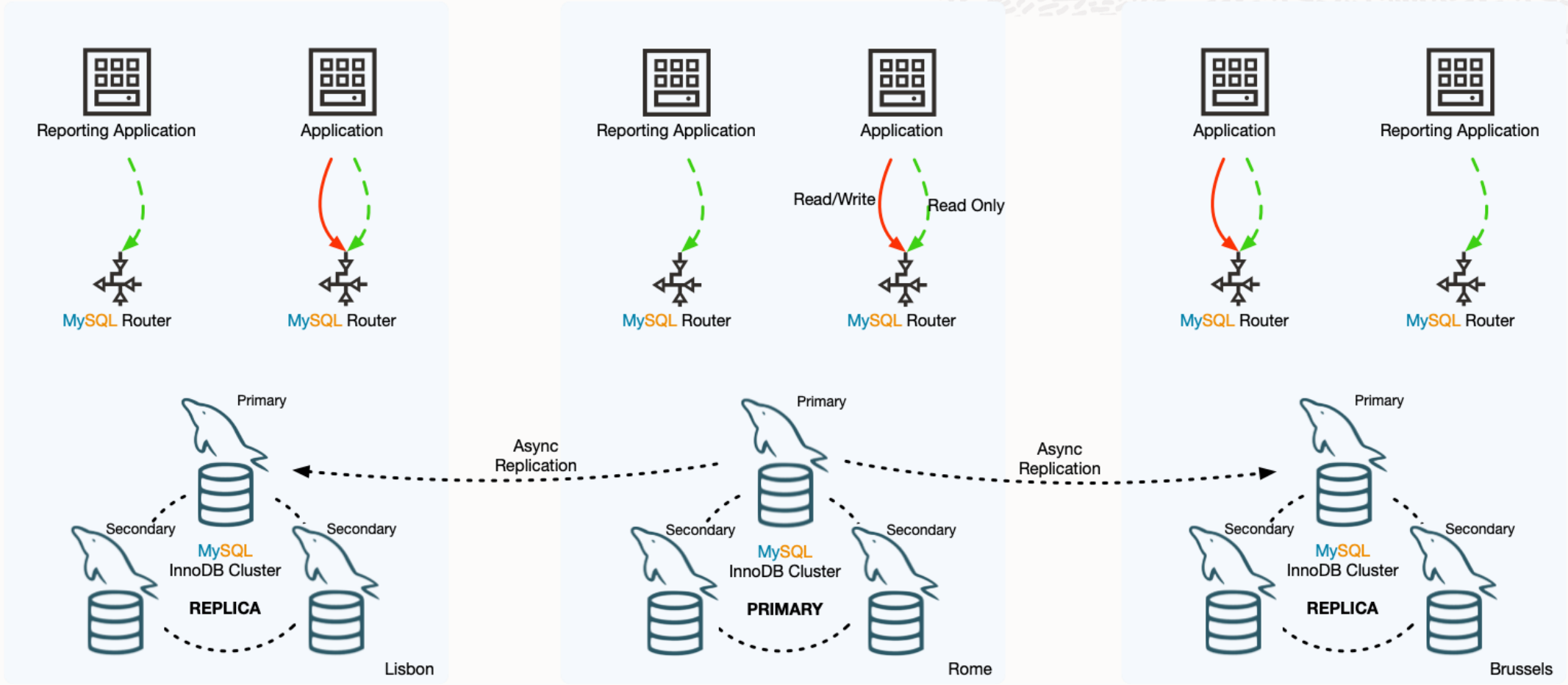
- ***C'est tout !*** 😀

- Maintenant, vous pouvez vérifier l'état du cluster et ajouter les routeurs :

```
MySQL … JS > cluster.status()
$> mysqlrouter --bootstrap root@host1
$> systemctl start mysqlrouter.service
```

InnoDB ClusterSet

MySQL InnoDB ClusterSet – 3 Data Centers



MySQL InnoDB ClusterSet

Principales fonctionnalités :

Gestion Centralisée 🔒 :

✅ Fournit une interface unifiée pour gérer plusieurs clusters MySQL, simplifiant les opérations administratives et la surveillance.

Scalabilité Horizontale 📈 :

✅ Permet d'ajouter facilement de nouveaux clusters pour faire face à la croissance des données et aux exigences de performance.

✅ Idéal pour des applications nécessitant une disponibilité constante et un accès aux données en temps réel.

Répartition des Charges ⚖️ :

✅ Facilite la répartition des charges de travail entre plusieurs clusters, optimisant ainsi les performances et la résilience.

Haute Disponibilité et Récupération 🔄 :

✅ Offre des options avancées pour la tolérance aux pannes, la réplication et la récupération en cas de sinistre.

✅ Intégration des mécanismes de failover pour assurer une continuité de service même lors de défaillances matérielles.

InnoDB ClusterSet: minimal setup

1. Créer les clusters MySQL InnoDB
2. Créer le ClusterSet :
   - `MySQL JS > cs = cluster.createClusterSet('mydrsolution')`
4. Créer les répliques :
   - `MySQL JS > repl = cs.createReplicaCluster('repl_host1', 'repl_cluster')`
- 5. Ajouter les instances :
   - `MySQL JS > repl.addInstance('repl_host2')`
   - `MySQL JS > repl.addInstance('repl_host3')`
     - ***C'est tout !*** 😀
- Vous pouvez maintenant vérifier l'état de votre ClusterSet
   - `MySQL JS > cs.status()`

Synthèse et Perspectives

Synthèse et Perspectives

Flexibilité et Automatisation ⚙️ :

✅ MySQL Shell joue un rôle essentiel dans l'automatisation des tâches et la gestion des configurations, facilitant ainsi les opérations quotidiennes.

Réplication Avancée 💾 :

✅ MySQL Group Replication et MySQL InnoDB Cluster assurent que les données restent disponibles et cohérentes même en cas de défaillance. MySQL InnoDB Cluster facilite le déploiement et la gestion de l'architecture HA dans son ensemble.

Routage Intelligent 🔄 :

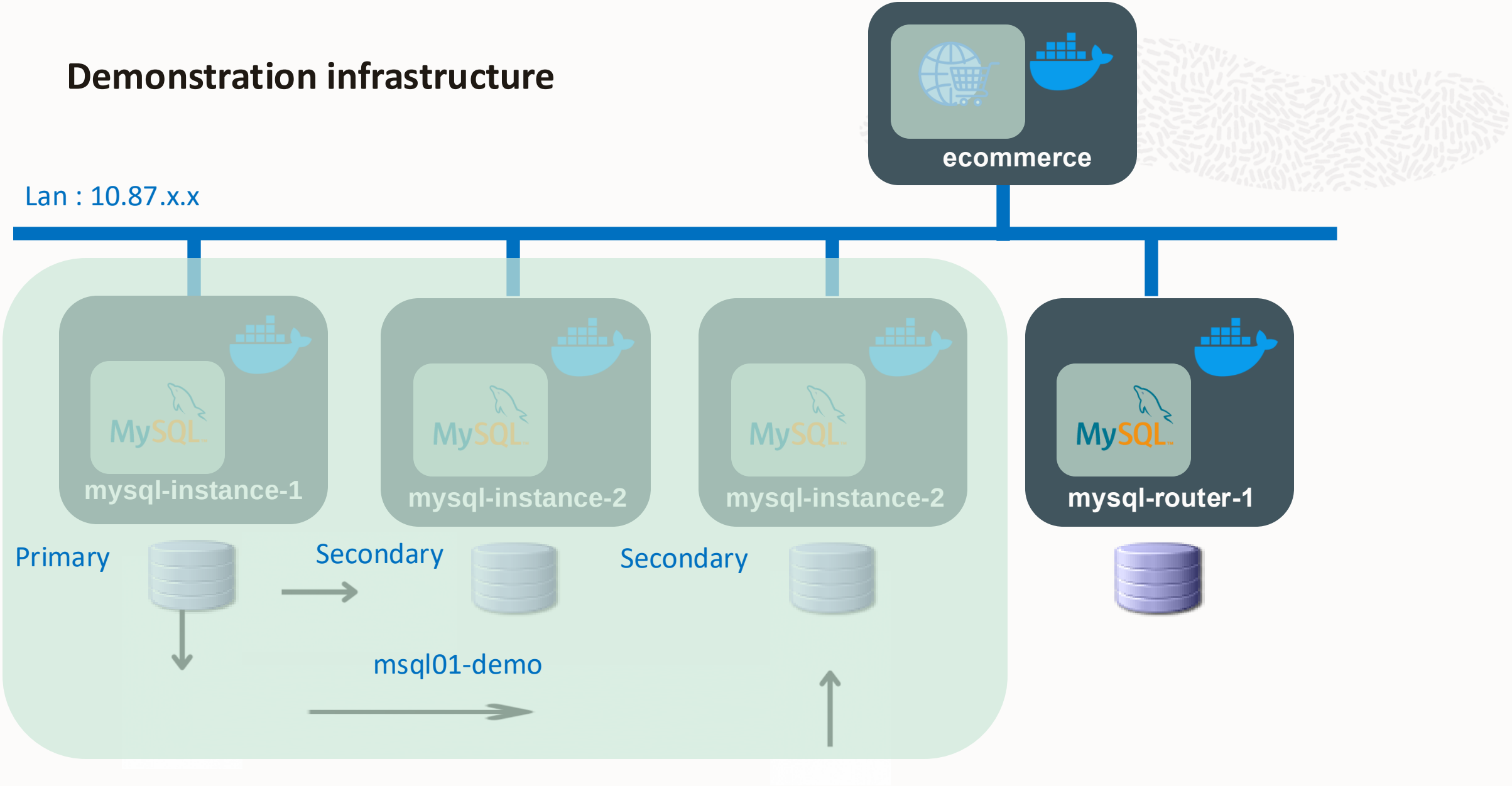
✅ MySQL Router permet un routage optimal des requêtes, améliorant la performance et la disponibilité en séparant les tâches de lecture et d’écriture.

Gestion à Grande Échelle 🌐 :

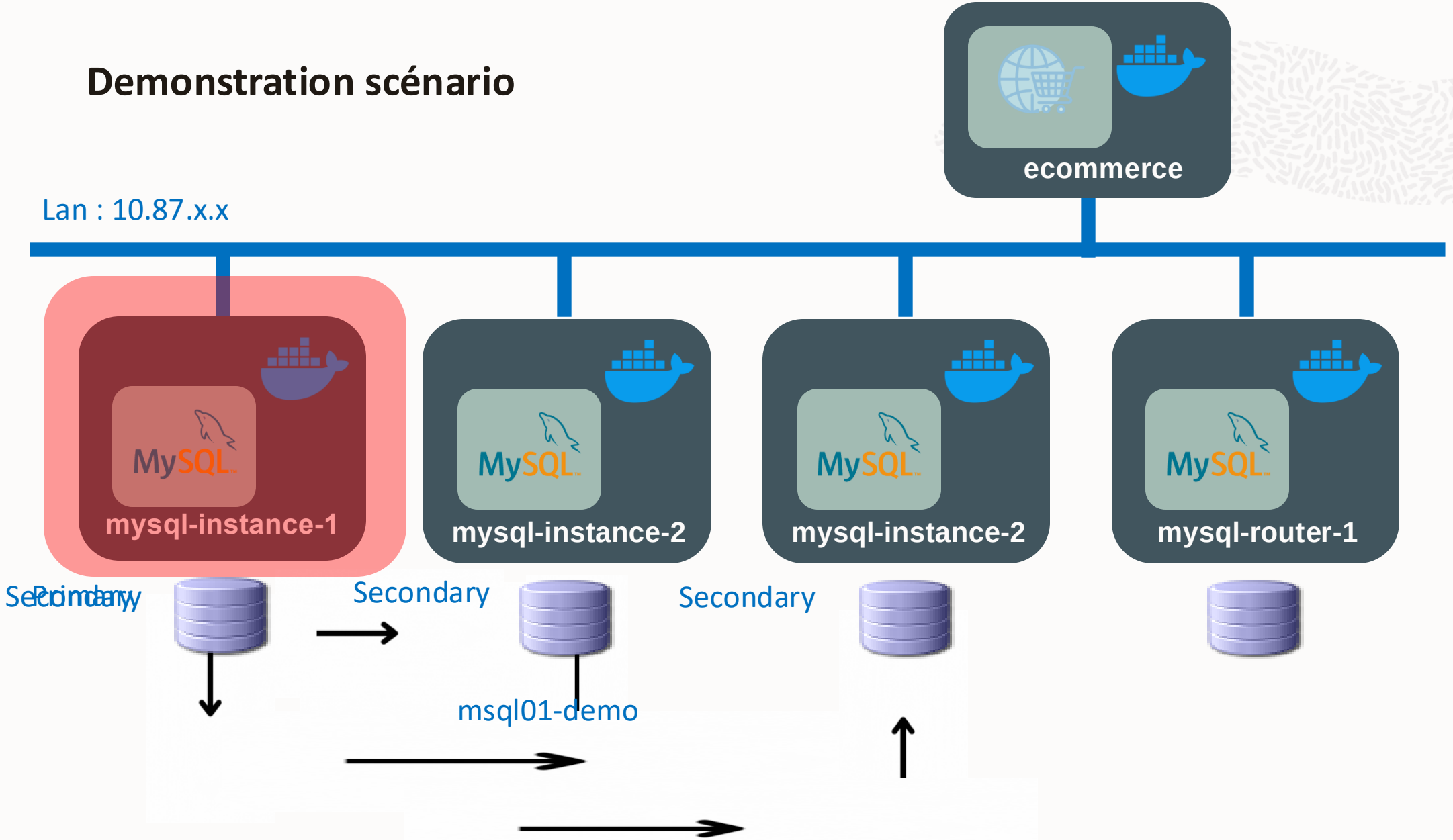
✅ MySQL ClusterSet permet de gérer plusieurs clusters de manière centralisée, offrant à la fois scalabilité et résilience.

Demonstration

Demonstration infrastructure



Demonstration scénario



Questions ?

Resources

InnoDB Cluster

https://dev.mysql.com/doc/mysql-shell/9.1/en/mysql-innodb-cluster.html

Pre-Checking Instance Configuration for InnoDB Cluster Usage

https://dev.mysql.com/doc/mysql-shell/9.1/en/check-instance-configuration.html

Replication Sets

https://dev.mysql.com/doc/mysql-shell/9.1/en/mysql-innodb-replicaset.html

Reference Manual

https://dev.mysql.com/doc/refman/9.1/en/

Editions

https://www.mysql.com/products/

Merci!

Emmanuel COLUSSI

emmanuel.colussi@oracle.com


ORACLE