



Automatisez votre solution de reprise après sinistre pour les bases de données relationnelles sur AWS

AWS Conseils prescriptifs



AWS Conseils prescriptifs: Automatisez votre solution de reprise après sinistre pour les bases de données relationnelles sur AWS

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques et la présentation commerciale d'Amazon ne peuvent être utilisées en relation avec un produit ou un service qui n'est pas d'Amazon, d'une manière susceptible de créer une confusion parmi les clients, ou d'une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d'Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Introduction	1
Objectifs	2
Présentation de	3
Le processus de basculement	4
Architecture FAILOVER de DR Orchestrator	4
Le processus de failback	5
Architecture FAILBACK de DR Orchestrator	5
DR Orchestrator FAILOVER	7
Détails des paramètres	8
DR Orchestrator FAILBACK	9
Machines d'état Aurora et Step Functions	12
État stable	12
État de l'événement	12
Basculement	13
Basculement manuel non planifié	18
Failback	22
Déployez la solution	31
FAQ	33
Quels sont les RPO et RTO que je peux atteindre en utilisant cette approche ?	33
Est-il obligatoire d'utiliser des variables AWS CloudFormation d'exportation ?	33
Puis-je utiliser le flux de travail DR Orchestrator FAILOVER pour basculer sur plusieurs bases de données ? AWS	33
Comment puis-je éviter l' InvalidParameterCombination erreur lorsque j'exécute la machine d'état DR Orchestrator FAILBACK pour Amazon RDS ?	34
Étapes suivantes	35
Ressources	36
Historique de la documentation	37
.....	xxxviii

Automatisez votre solution de reprise après sinistre pour les bases de données relationnelles sur AWS

Jitendra Kumar, Oliver Francis et Pavithra Balasubramanian, Amazon Web Services (AWS)

Mai 2024 ([historique du document](#))

Lorsque vous planifiez une [stratégie de reprise après sinistre \(DR\)](#) sur Amazon Web Services (AWS), vous pouvez implémenter une solution de reprise après sinistre pour les bases de données de votre organisation. Vous pouvez automatiser la solution de reprise après sinistre en configurant une architecture axée sur les événements pour effectuer le basculement des instances de base de données. Les avantages de cette approche sont notamment les suivants :

- Réduire les erreurs humaines causées par la coordination manuelle
- Fournir un objectif de temps de restauration (RTO) le plus court possible dans certains cas
- Faciliter les tests reproductibles de votre solution de reprise après sinistre

Sans automatisation, une panne de votre système principal Région AWS nécessiterait des ressources au sein de votre organisation pour effectuer manuellement les étapes d'un manuel d'exécution. Ces étapes peuvent inclure plusieurs processus, tels que la promotion des répliques de lecture Amazon Relational Database Service (Amazon RDS) dans la Région AWS secondaire pour qu'elles servent de nouvelle instance de base de données principale. AWS fournit un [AWS SDK pour Python \(Boto3\) permettant](#) APIs d'automatiser ces actions, qui peuvent constituer les éléments de base d'une solution de reprise après sinistre adaptée aux besoins de votre entreprise.

Ce guide décrit un orchestrateur de reprise après sinistre centralisé piloté par un manifeste qui peut vous aider à obtenir une solution de reprise après sinistre pour les AWS bases de données Régions AWS au sein d'une même instance. Compte AWS La décision d'effectuer un basculement nécessiterait tout de même une intervention humaine et est hors de portée de l'automatisation.

Les publics cibles de ce guide sont les architectes d'applications, les architectes d'infrastructure, les architectes de bases de données et les consultants en bases de données.

Objectifs

La mise en œuvre de la solution DR Orchestrator Framework peut vous aider à atteindre les résultats suivants :

- Réduisez les erreurs manuelles lors d'une panne — Les performances manuelles des processus de reprise après sinistre peuvent être sujettes à des erreurs. L'automatisation des actions pendant les phases d'une activité de reprise après sinistre réduit les erreurs manuelles et permet de garantir un basculement ou un retour en arrière plus fluide.
- Gérez la reprise après sinistre à grande échelle : si votre entreprise dispose d'un important parc de bases de données, l'automatisation des bâtiments pour la reprise après sinistre vous permet de faire évoluer votre solution de reprise après sinistre. Vous pouvez automatiser votre solution de reprise après AWS sinistre de base de données à l'aide de DR Orchestrator Framework. DR Orchestrator automatise les étapes de basculement ou de restauration des AWS bases de données à l'aide d'un fichier de paramètres au format JSON.

Présentation de DR Orchestrator Framework

DR Orchestrator Framework fournit une solution en un clic pour orchestrer et automatiser la reprise après sinistre entre régions pour les bases de données. AWS Il utilise [AWS Step Functions](#) et exécute [AWS Lambda](#) les étapes requises lors du basculement et du retour en arrière. Les machines d'état Step Functions constituent la base de la prise de décision dans le cadre de la conception de l'orchestrateur. Les opérations d'API permettant d'effectuer des actions de basculement ou de retour sont codées dans des fonctions Lambda appelées depuis la machine à états. Les fonctions Lambda exécutent des [AWS SDK pour Python \(Boto3\)](#) API pour interagir avec AWS les bases de données.

DR Orchestrator Framework contient deux machines d'état principales qui correspondent aux phases de basculement et de retour en arrière.

Pour Amazon RDS, la phase de basculement favorise la création d'une réplique de lecture RDS interrégionale dans une instance de base de données autonome. Pour Amazon Aurora, lorsque la région principale est en panne lors d'une panne rare et inattendue, son nœud d'écriture n'est pas disponible. La réplication entre le nœud d'écriture et les clusters secondaires s'arrête. Vous devez détacher le cluster secondaire de la base de données globale et le promouvoir en tant que cluster autonome. Les applications peuvent se connecter et envoyer du trafic d'écriture au cluster autonome. Vous pouvez utiliser ce même processus pour transférer [le cluster de](#) base de données principal de la base de données globale vers les régions secondaires. Utilisez cette approche pour des scénarios contrôlés tels que les suivants :

- Maintenance opérationnelle
- Procédures opérationnelles planifiées
- Promotion d'un cluster secondaire Amazon ElastiCache (Redis OSS) en tant que nouveau cluster principal

La phase de repli établit la réplication en direct des données entre une région principale active et une nouvelle région secondaire.

Il est essentiel de comprendre que DR Orchestrator s'applique uniquement aux bases de données. Toutes les applications qui font référence à ces bases de données et qui se trouvent dans la même région peuvent avoir besoin d'une solution de basculement en tandem distincte. Après le basculement des bases de données vers la région secondaire, les applications doivent être mises

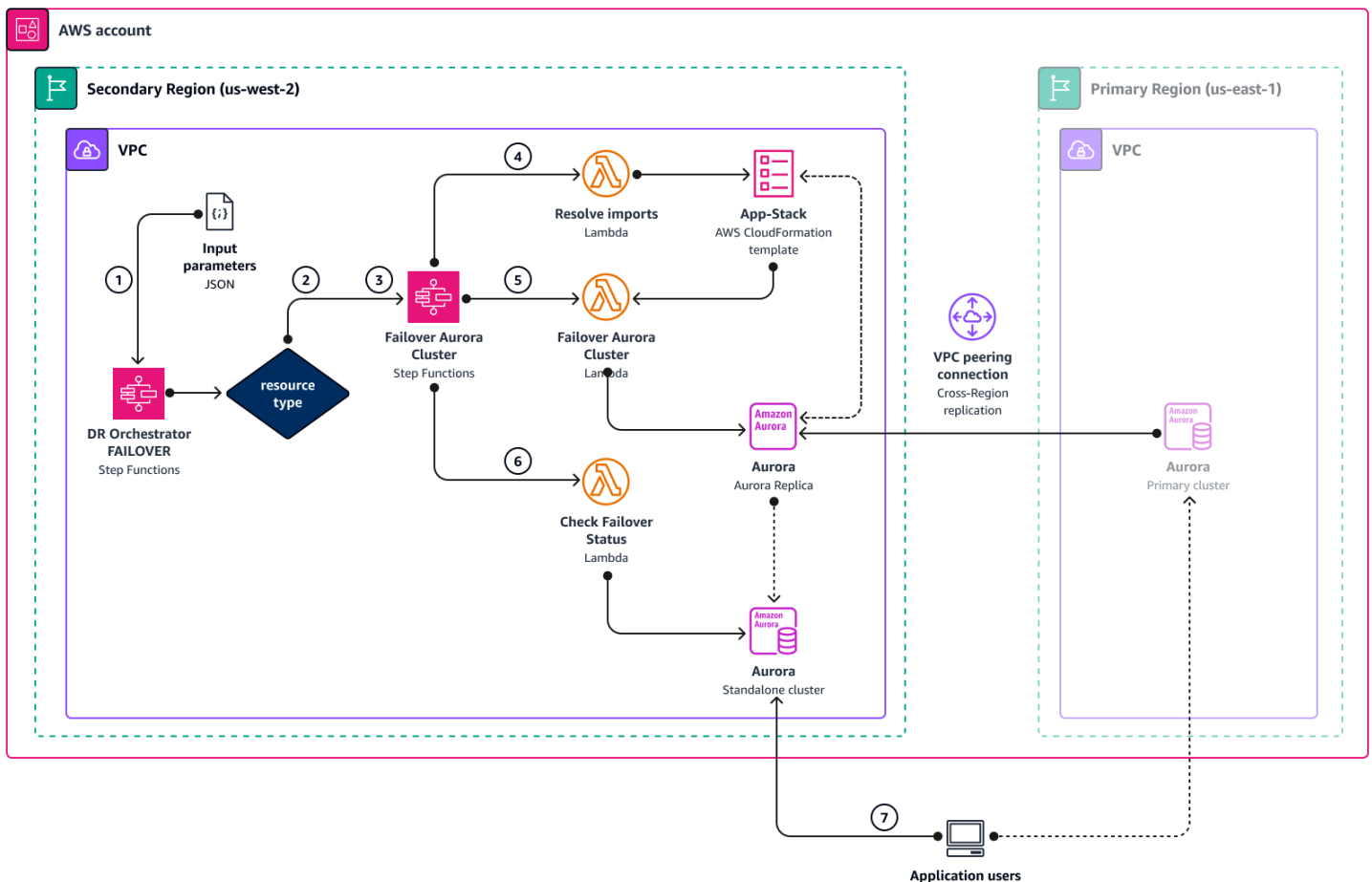
à jour pour se connecter aux nouvelles instances de base de données, qui serviront de source de données.

Le processus de basculement

Pour effectuer un basculement, exécutez la machine DR Orchestrator FAILOVER d'état. À ce stade, une base de données secondaire est déjà présente dans la région secondaire, soit en tant que réplique en lecture (Amazon RDS), soit en tant que cluster secondaire (Amazon Aurora). Lorsque vous exécutez la machine DR Orchestrator FAILOVER d'état, la base de données secondaire devient la base de données principale.

Architecture FAILOVER de DR Orchestrator

Le schéma suivant montre les concepts du processus de basculement pour Amazon Aurora lors de l'utilisation de DR Orchestrator. Amazon Aurora et Amazon ElastiCache utilisent le même flux de travail, mais avec des machines à états et des fonctions Lambda différentes.



1. La machine `DR Orchestrator FAILOVER` d'état lit les paramètres JSON d'entrée.
2. Sur la base du `resourceType` paramètre, la machine à états appelle les autres machines à états : `Promote RDS Read Replica`, `Failover Aurora Cluster`, ou `Failover ElastiCache`. Si plusieurs ressources sont transmises en entrée, ces machines d'état s'exécutent en parallèle.
3. La machine à `Failover Aurora Cluster` états appelle les fonctions Lambda dans chacune des trois étapes suivantes.
4. La fonction `Resolve imports Lambda` est résolue "`! import <export-variable-name>`" avec les valeurs réelles du `App-Stack AWS CloudFormation` modèle.
5. La fonction **`Failover Aurora Cluster`** Lambda promeut la réplique en lecture en tant qu'instance de base de données autonome.
6. La fonction **`Check Failover Status`** Lambda vérifie l'état de l'instance de base de données promue. Une fois que le statut est `DISPONIBLE`, la fonction Lambda renvoie un jeton de réussite à la machine d'état appelante et termine.
7. Vous pouvez rediriger vos applications vers la base de données autonome de la région DR (`us-west-2`), qui est désormais la base de données principale.

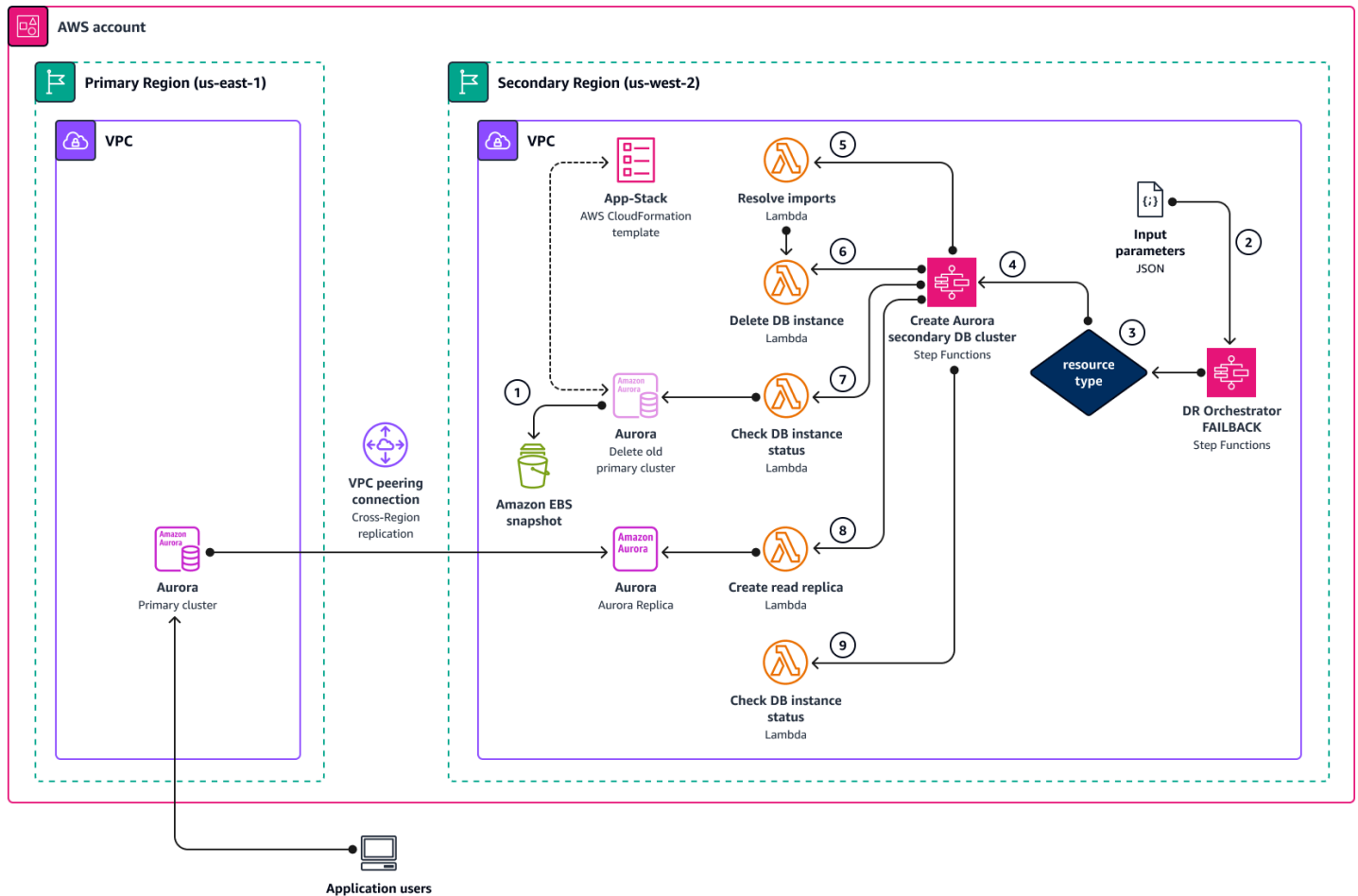
Le processus de failback

Une fois que votre ancienne région principale (`us-east-1`) sera de nouveau active, vous pouvez y revenir, de sorte que la base de données `us-east-1` redevienne la région principale. Pour démarrer le failback, exécutez la machine `DR Orchestrator FAILBACK` d'état. Comme son nom l'indique, cette machine à états commence à répliquer les modifications de votre nouvelle région principale (`us-west-2`) vers l'ancienne région principale (`us-east-1`), qui agit en tant que région secondaire actuelle.

Une fois la réplication établie entre les deux régions, vous pouvez lancer le retour en arrière. Pour revenir en arrière et revenir à votre région principale d'origine (`us-east-1`), exécutez la machine d'`DR Orchestrator FAILOVER` d'état de la région secondaire actuelle (`us-east-1`) pour la promouvoir en région principale.

Architecture **FAILBACK** de **DR Orchestrator**

Le schéma suivant montre les concepts du processus de repli pour Amazon Aurora lors de l'utilisation de `DR Orchestrator`.



1. Avant de commencer le retour en arrière, prenez un instantané manuel de base de données à utiliser lors de l'analyse des causes premières (RCA).

Désactivez également le DeletionProtection pour le cluster Aurora dans la région principale précédente (us-east-1).

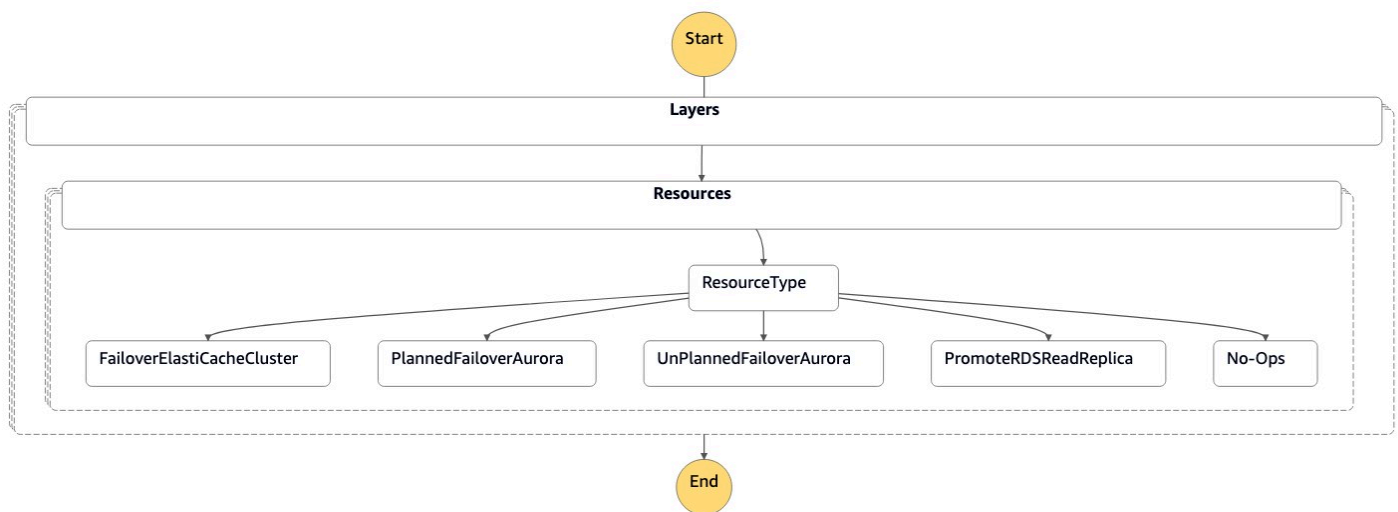
2. La machine DR Orchestrator FAILBACK d'état lit les paramètres JSON d'entrée.
3. Sur la base de resourceType, la machine DR Orchestrator FAILBACK d'état appelle la machine Create Aurora Secondary DB Cluster d'état.
4. La machine à Create Aurora Secondary DB Cluster états appelle les fonctions Lambda dans chacune des cinq étapes suivantes.
5. La fonction Resolve import Lambda est résolue `! import <export-variable-name>` avec les valeurs réelles du App-Stack CloudFormation modèle.
6. La fonction Delete DB Instance Lambda supprime l'ancienne instance principale.

7. La fonction `Check DB instance status` Lambda vérifie `Delete DB Instance status` jusqu'à ce que la base de données soit supprimée.
8. La fonction `Create Read Replica` Lambda crée une réplique de lecture dans la région secondaire à partir de l'instance de base de données située dans la nouvelle région principale.
9. La fonction `Check DB instance status` Lambda vérifie l'état de l'instance de base de données de réplique lue. Lorsque le statut est `DISPONIBLE`, la fonction Lambda renvoie un jeton de réussite à la machine d'état appelante, ce qui est terminé.

DR Orchestrator FAILOVER

Utilisez la machine `DR Orchestrator FAILOVER` d'état en cas de reprise après sinistre lorsque la région principale (`us-east-1`) est en panne ou lors d'événements planifiés tels que la maintenance opérationnelle.

La fonction peut être appelée pour basculer sur une ou plusieurs bases de données en parallèle.



La machine d'état accepte les paramètres au format JSON, comme indiqué dans le code suivant :

```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "PromoteRDSReadReplica",
          "resourceName": "Promote RDS MySQL Read Replica",

```

```
    "parameters": {
      "RDSInstanceIdentifier": "!Import rds-mysql-instance-identifiant",
      "TargetClusterIdentifier": "!Import rds-mysql-instance-global-arn"
    },
    {
      "resourceType": "FailoverElastiCacheCluster",
      "resourceName": "Failover ElastiCache Cluster",
      "parameters": {
        "GlobalReplicationGroupId": "!Import demo-redis-cluster-global-replication-group-id",
        "TargetRegion": "!Import demo-redis-cluster-target-region",
        "TargetReplicationGroupId": "!Import demo-redis-cluster-target-replication-group-id"
      }
    }
  ]
}
```

Détails des paramètres

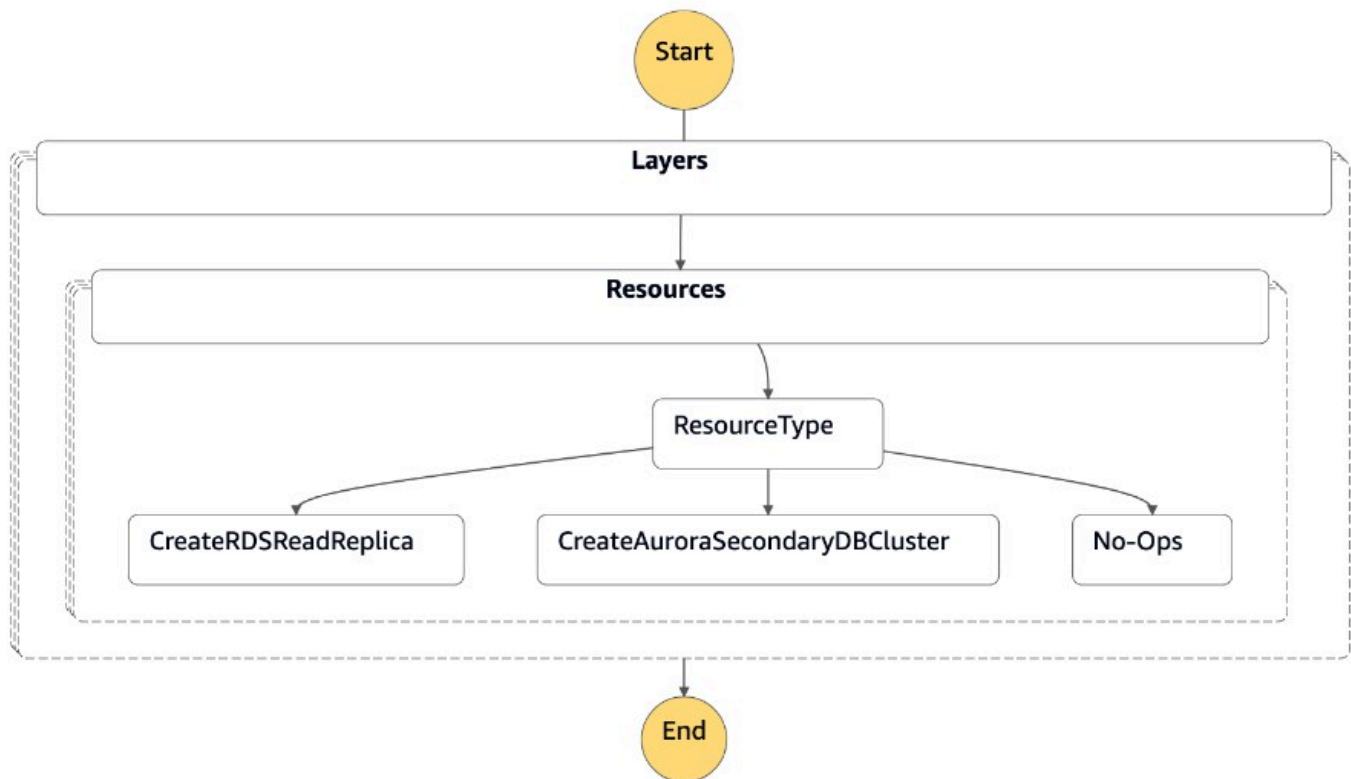
Le tableau suivant indique les paramètres utilisés par la machine DR Orchestrator FAILOVER d'état.

Nom du paramètre	Description	Valeurs attendues
layer(obligatoire : numéro)	La séquence de traitement. Toutes les ressources définies dans la couche 1 doivent être exécutées avant que les ressources de couche 2 ne soient exécutées.	1 ou 2, et ainsi de suite
ressources (obligatoire : tableau de dictionnaires)	Toutes les ressources d'une même couche s'exécutent en parallèle.	<pre>{ "resourceType": "String", "resourceName": "String", </pre>

Nom du paramètre	Description	Valeurs attendues
		<pre>"parameters":{ "<param1>":"<!Import cft-output-1">, }</pre>
resourceType (obligatoire : chaîne)	Type de ressource pour identifier la ressource	PromoteRDSReadReplica ou FailoverElastiCacheCluster
resourceName (facultatif : chaîne)	Pour identifier à quel portefeuille d'applications appartiennent ces ressources	Promote RDS for MySQL Read Replica
paramètres (obligatoire : tableau de dictionnaire)	Liste des paramètres requis pour basculer ou revenir en arrière dans la AWS base de données	<pre>{ "<param1> ": "<!Import cft- output-1">", "<param2>":"<!Import cft- output-2">", }</pre>

DR Orchestrator FAILBACK

Utilisez la machine à DR Orchestrator FAILBACK états après l'événement DR, lorsque l'ancienne région principale (us-east-1) est active. Vous pouvez créer la [réplique en lecture](#) pour Amazon RDS dans l'ancienne région principale à partir de la nouvelle région principale (us-west-2) afin de vous conformer à votre stratégie de reprise après sinistre. Comme il s'agit d'un événement planifié, vous pouvez planifier cette activité pendant le week-end ou en dehors des heures de bureau avec une estimation des temps d'arrêt.



La machine d'état accepte les paramètres au format JSON, comme indiqué dans le code suivant :

```

{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "CreateRDSReadReplica",
          "resourceName": "Create RDS for MySQL Read Replica",
          "parameters": {
            "RDSInstanceIdentifier": "!Import rds-mysql-instance-identifier",
            "TargetClusterIdentifier": "!Import rds-mysql-instance-global-arn",
            "SourceRDSInstanceIdentifier": "!Import rds-mysql-instance-source-identifier",
            "SourceRegion": "!Import rds-mysql-instance-SourceRegion",
            "MultiAZ": "!Import rds-mysql-instance-MultiAZ",
            "DBInstanceClass": "!Import rds-mysql-instance-DBInstanceClass",
            "DBSubnetGroup": "!Import rds-mysql-instance-DBSubnetGroup",
            "DBSecurityGroup": "!Import rds-mysql-instance-DBSecurityGroup",
            "KmsKeyId": "!Import rds-mysql-instance-KmsKeyId",
            "BackupRetentionPeriod": "7",

```

```
        "MonitoringInterval": "60",
        "StorageEncrypted": "True",
        "EnableIAMDatabaseAuthentication": "True",
        "DeletionProtection": "True",
        "CopyTagsToSnapshot": "True",
        "AutoMinorVersionUpgrade": "True",
        "Port": "!Import rds-mysql-instance-DBPortNumber",
        "MonitoringRoleArn": "!Import rds-mysql-instance-RDSMonitoringRole"
    }
}
]
```

Machines d'état Aurora et Step Functions

Cette section couvre les processus et les machines d'état spécifiques au basculement et au retour en panne des clusters Amazon Aurora. Les clusters sont configurés en tant que base de données globale.

Note

À des fins de démonstration, cet exemple utilise l' MySQL-Compatible édition Aurora. Vous pouvez suivre des étapes similaires pour Aurora PostgreSQL-Compatible Edition.

État stable

À l'état stable, une base de données MySQL-Compatible globale Amazon Aurora (`dr-globaldb-cluster-mysql`) a été créée avec deux clusters de bases de données. Le premier cluster de base de données (`db-cluster-01`) a été créé dans le cluster principal Région AWS (`us-east-1`) pour servir la read/write charge de travail. Le deuxième cluster de base de données (`db-cluster-02`) a été créé dans la région secondaire (`us-west-2`) pour servir la charge de travail en lecture seule.

En plus de fournir la solution DR, vous pouvez réduire la charge sur votre cluster de base de données principal en acheminant les requêtes de lecture de vos applications vers le cluster de base de données secondaire. Chacun de ces clusters contient une instance de base de données appelée `dbcluster-01-use1-instance-1` et `dbcluster-02-usw2-instance-2`, respectivement.

État de l'événement

À l'aide d'une base de données mondiale Amazon Aurora, vous pouvez planifier et reprendre vos activités en cas de sinistre assez rapidement. La reprise après sinistre est généralement mesurée à l'aide des valeurs de l'objectif de temps de restauration (RTO) et de l'objectif du point de reprise (RPO). Pour plus d'informations, consultez [Utilisation du basculement ou du basculement dans une base de données globale Amazon Aurora](#).

Avec une base de données globale Aurora, il existe deux approches différentes pour le basculement :

- Basculement (basculement planifié géré)
- Basculement (basculement manuel non planifié, ou détachement et promotion)

Basculement

Le passage au numérique est destiné aux environnements contrôlés, tels que la maintenance opérationnelle et les autres procédures opérationnelles planifiées. En utilisant un basculement planifié géré, vous pouvez déplacer le cluster de base de données principal de votre base de données globale Aurora vers l'une des régions secondaires. Comme le basculement attend que les clusters de base de données secondaires soient synchronisés avec la base de données principale, le RPO est égal à 0 (aucune perte de données). Pour en savoir plus, consultez la section [Effectuer des commutations pour les bases de données mondiales Amazon Aurora](#).

La machine `dr-orchestrator-stepfunction-FAILOVER` d'état est invoquée pendant l'état de l'événement pour faire basculer votre cluster principal vers la région secondaire que vous avez choisie (`us-west-2`).

Pour effectuer le basculement, procédez comme suit :

1. Connectez-vous au Console de gestion AWS.
2. Remplacez la région par la région DR (`us-west-2`).
3. Accédez à Services, puis sélectionnez Step Functions.
4. Accédez à la machine à `dr-orchestrator-stepfunction-FAILOVER` états.
5. Choisissez Démarrer l'exécution, puis entrez le code JSON suivant dans la Input - optional section :

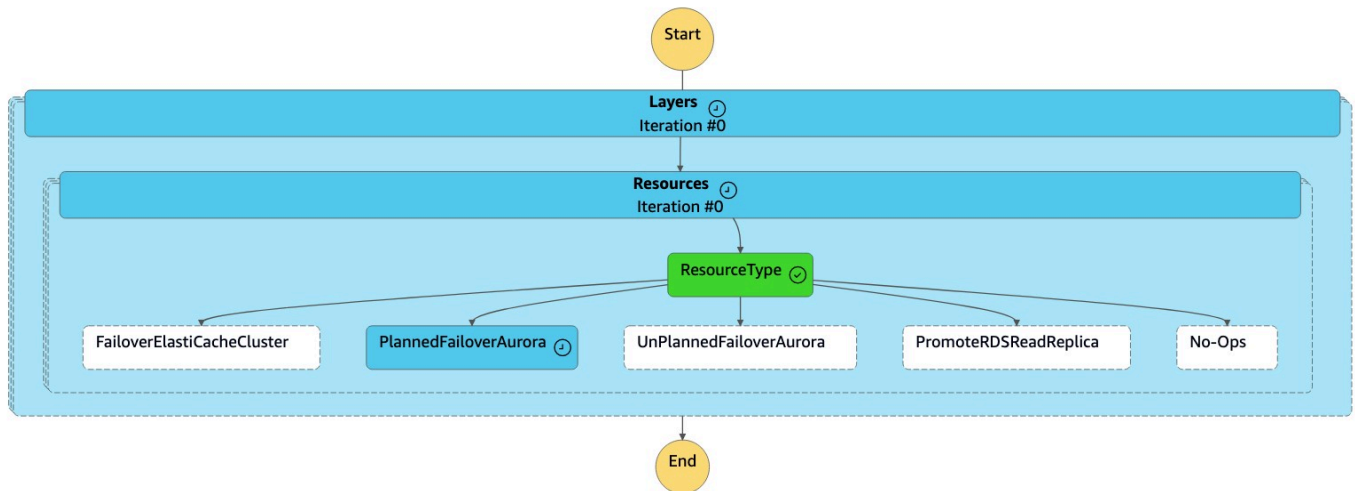
```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "PlannedFailoverAurora",
          "resourceName": "Switchover (planned failover) of Amazon Aurora global
databases (MySQL)",
          "parameters": {
            "GlobalClusterIdentifier": "!Import dr-globaldb-cluster-mysql-global-
identifier",
            "DBClusterIdentifier": "!Import dr-globaldb-cluster-mysql-cluster-
identifier"
          }
        }
      ]
    }
  ]
}
```

```

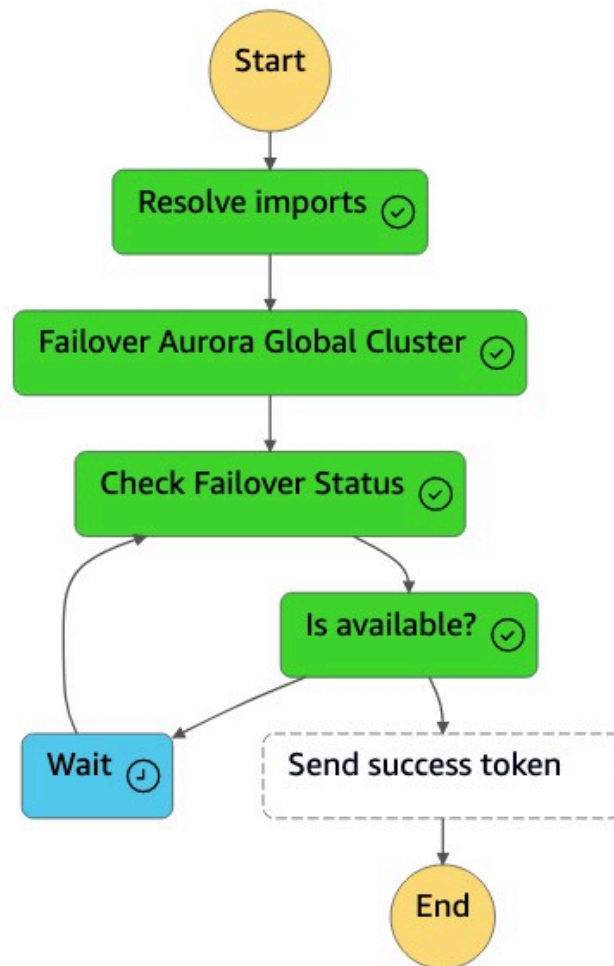
    ]
  }
]
}

```

6. La machine `dr-orchestrator-stepfunction-FAILOVER` d'état lit le type de ressource `PlannedFailoverAuroraMySQL` et appelle la machine `dr-orchestrator-stepfunction-planned-Aurora-failover` d'état pour qu'elle bascule sur la base de données globale Aurora.



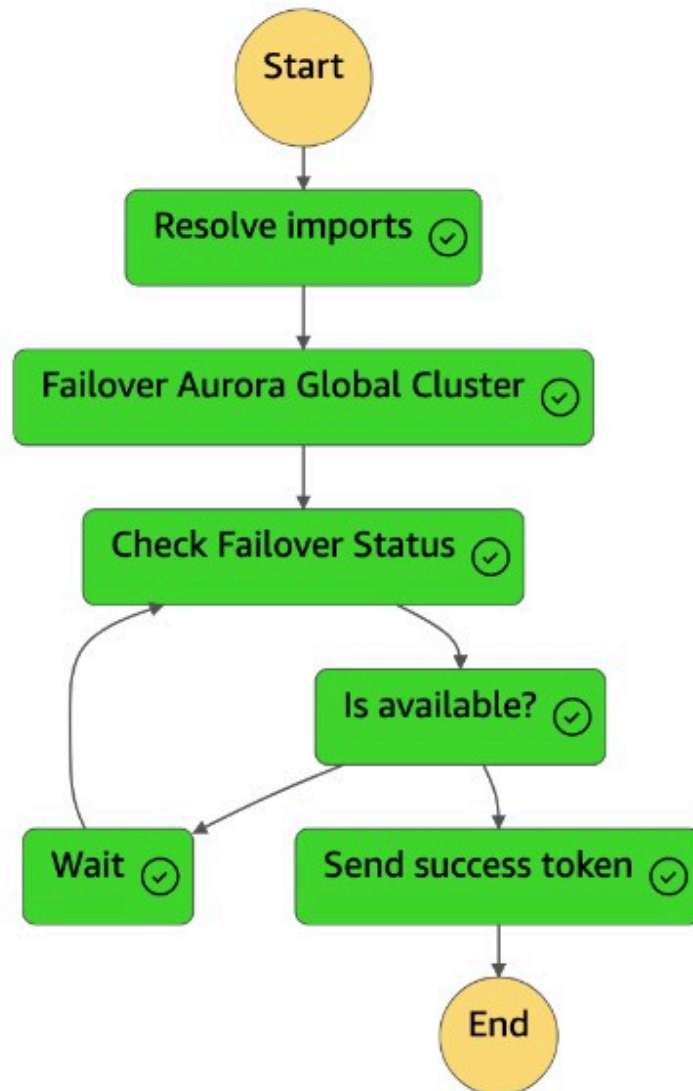
7. La machine `dr-orchestrator-stepfunction-planned-Aurora-failover` d'état exécute les étapes suivantes pour changer le rôle de base de données MySQL-Compatible globale Aurora.



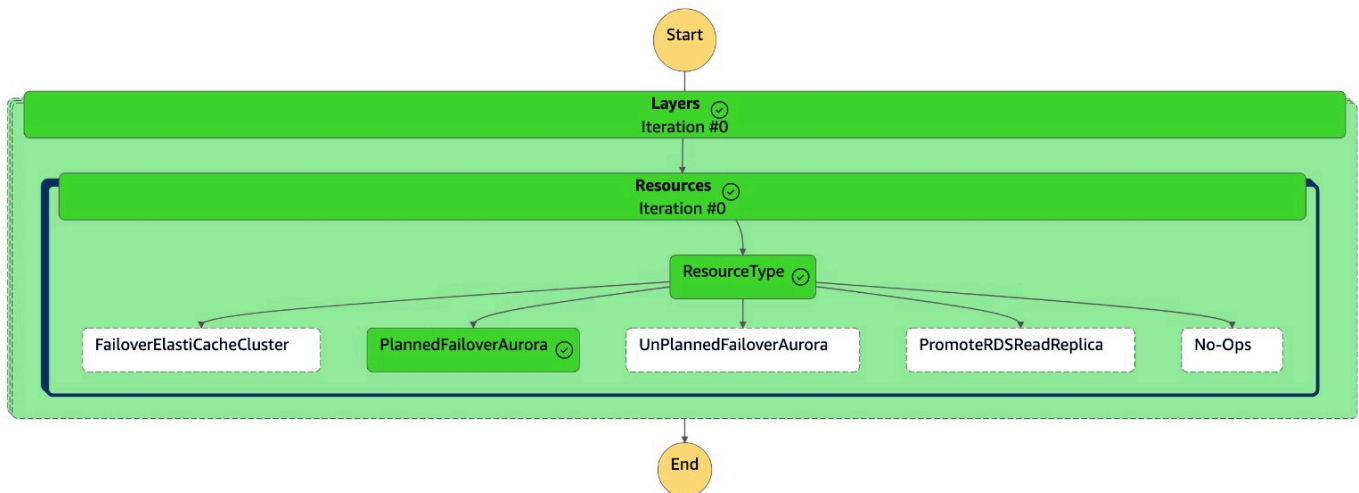
Step (Étape)	Description	Valeurs attendues
Résoudre les importations	Une fonction Lambda remplace !Import <variable name> les valeurs par le nom réel.	"!Import dr-global db-cluster-mysql-global-identifiant" est remplacé par "dr-global db-cluster-mysql" .

Step (Étape)	Description	Valeurs attendues
Cluster mondial Failover Aurora	Une fonction Lambda appelle les API Boto3 failover_global_cluster pour qu'elles survolent la base de données globale Aurora.	<pre>{ 'GlobalCluster': { 'GlobalClusterId entifieur': 'dr-globa ldb-cluster-mysql', 'GlobalClusterRe sourceId': 'cluster- cce7f9bec2846db4', 'GlobalClusterArn': 'arn:aws:rds::xxx', 'Status': 'failing- over', } }</pre>
Vérifier l'état du basculement	Une fonction Lambda appelle les API Boto3 describe_db_clusters pour vérifier l'état du basculement.	modificatif, disponible
Envoyer des jetons de réussite	Une fonction Lambda appelle les API Boto3 send_task_success et renvoie un jeton de réussite à la machine d'état. DR Orchestrator Failover	h7XRiCdLtd/83p1E0dMccoxlzFhglsdkzpK9mBVKZsp7d9yrT1W

- Accédez à la console Amazon RDS. Sous Status, les valeurs de la base de données globale Aurora passeront de Disponible à Switching over ou Modifying.
- Une fois que la machine `dr-orchestrator-stepfunction-planned-Aurora-failover` d'état est terminée, elle renvoie un jeton de réussite à la machine `dr-orchestrator-stepfunction-FAILOVER` d'état.



10 La machine à dr-orchestrator-stepfunction-FAILOVER états est terminée.



Sur la console, le rôle du cluster secondaire (`dbcluster-02`) est désormais le cluster principal, et le cluster est prêt à traiter les read/write charges de travail. Le rôle du cluster principal d'origine (`dbcluster-01`) est désormais répertorié comme cluster secondaire.

Basculement manuel non planifié

Dans de rares cas, votre base de données globale Aurora peut rencontrer une panne inattendue dans sa base de données principale Région AWS. Si cela se produit, votre cluster de base de données Aurora principal et son nœud de scripteur ne sont pas disponibles, et la réplication entre le cluster principal et les clusters secondaires s'interrompt. Pour minimiser à la fois les temps d'arrêt (RTO) et les pertes de données (RPO), effectuez rapidement un basculement entre régions et reconstruisez votre base de données globale Aurora. Pour plus d'informations, consultez [Restaurer une base de données globale Amazon Aurora suite à une panne imprévue](#).

Pour effectuer un basculement imprévu, vous devez détacher votre cluster secondaire de la base de données globale Aurora. Avant d'effectuer le basculement imprévu, arrêtez les écritures d'applications sur votre cluster de base de données Aurora principal. Une fois le basculement effectué avec succès, reconfigurez l'application pour écrire sur le nouveau cluster de base de données principal. Cette approche permet d'éviter les pertes de données. Cela permet également d'éviter les incohérences dans les données si le nœud d'écriture principal revient en ligne pendant le processus de basculement.

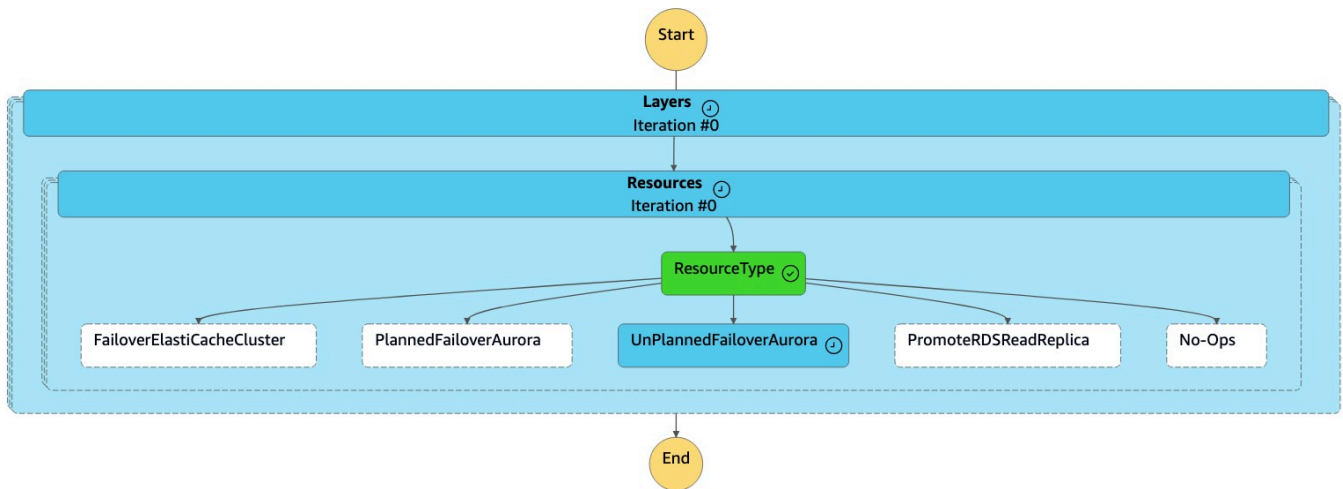
Pour effectuer le basculement imprévu, appelez la machine d'`dr-orchestrator-stepfunction-FAILOVER` état. Pour cet exemple, le cluster secondaire (`db-cluster-02`) se trouve dans la région DR (`us-west-2`) à l'état stable.

Pour effectuer le basculement, procédez comme suit :

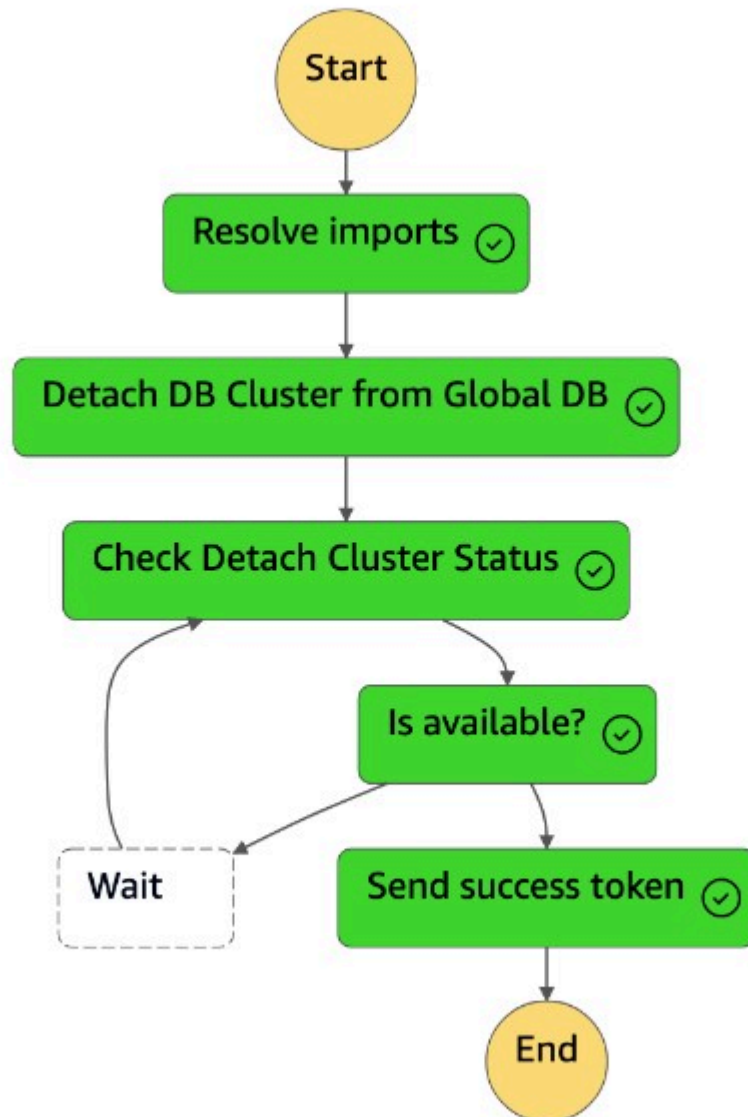
1. Connectez-vous à la console.
2. Remplacez la région par la région DR (us-west-2).
3. Accédez à Services, puis sélectionnez Step Functions.
4. Accédez à la machine à `dr-orchestrator-stepfunction-FAILOVER` états.
5. Choisissez Démarrer l'exécution, puis entrez le code JSON suivant dans la Input - optional section, en utilisant `UnPlannedFailoverAurora` comme suit `resourceType` :

```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "UnPlannedFailoverAurora",
          "resourceName": "Performing unplanned failover for Amazon Aurora global
databases (MySQL)",
          "parameters": {
            "GlobalClusterIdentifier": "!Import dr-globaldb-cluster-mysql-global-
identifier",
            "DBClusterIdentifier": "!Import dr-globaldb-cluster-mysql-cluster-
identifier",
            "ClusterRegion": "!Import dr-globaldb-cluster-mysql-cluster-region"
          }
        }
      ]
    }
  ]
}
```

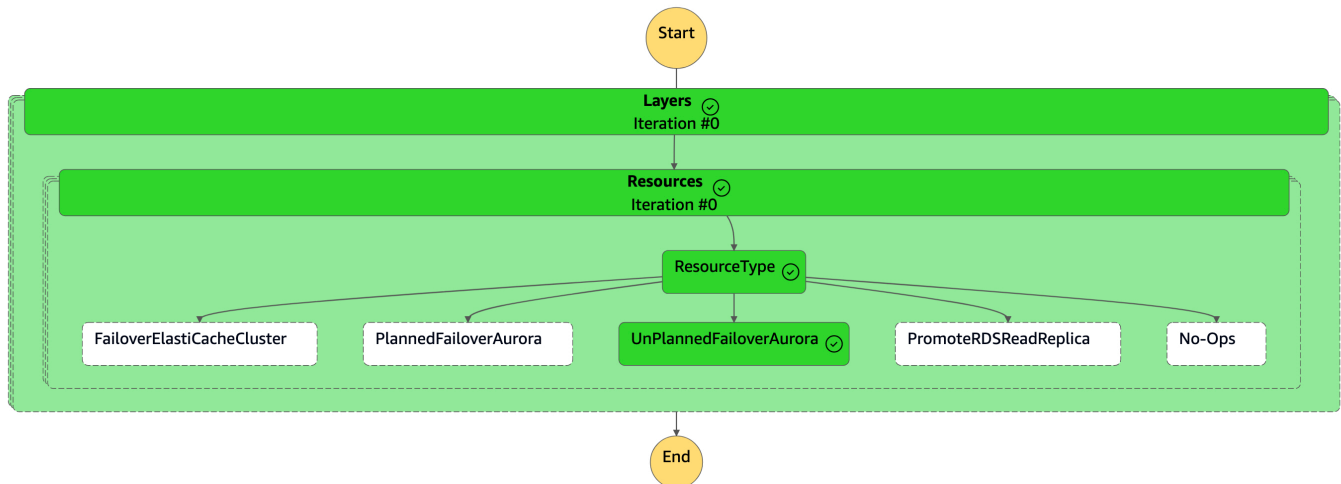
6. La machine `dr-orchestrator-stepfunction-FAILOVER` d'état lit le type de ressource `UnPlannedFailoverAuroraMySQL` et appelle la tâche `Detach Cluster from Global Database` depuis la machine `dr-orchestrator-stepfunction-unplanned-Aurora-failover` d'état.



7. La `Detach Cluster from Global Database` tâche détache (supprime) le cluster secondaire de la base de données globale.



8. Le cluster secondaire (dbcluster-02) est promu au rang de cluster autonome, et il peut prendre en charge des charges de read/write travail.
9. La machine à dr-orchestrator-stepfunction-FAILOVER états est terminée.



- 10 Le cluster secondaire (dbcluster-02) est détaché de la base de données globale Aurora et devient un cluster autonome pour répondre à la read/writer charge de travail.
- 11 Reconfigurez votre application pour envoyer toutes les opérations d'écriture à ce nouveau cluster de base de données Aurora autonome en utilisant son nouveau point de terminaison de cluster.

Failback

Un retour de secours ramène votre base de données à son emplacement principal d'origine (ou à son nouvel emplacement) après la résolution d'un sinistre (ou d'un événement planifié). Lorsque la panne imprévue aura été résolue, vous souhaitez peut-être ajouter à nouveau votre ancienne région principale à la base de données mondiale Aurora. Vous devez d'abord supprimer le cluster de base de données existant de l'ancienne région principale, créer un nouveau cluster de base de données à partir de la nouvelle région principale, puis utiliser le processus de basculement planifié géré pour changer le rôle du nouveau cluster.

Cela peut être considéré comme une activité planifiée que vous pouvez effectuer en dehors des heures de pointe ou le week-end.

Vous devez [modifier manuellement le cluster de base de données Amazon Aurora](#) et le désactiver `DeletionProtection` avant d'exécuter la machine d'OR Orchestrateur `FAILBACK` état depuis l'ancienne région principale (us-east-1), car elle a été créée avec `DeletionProtection`.

DR Orchestrator Framework utilise la machine à `dr-orchestrator-stepfunction-FAILBACK` états pour automatiser les étapes de suppression du cluster existant et de création d'un nouveau cluster dans l'ancienne région principale.

Pour le désactiver `DeletionProtection`, procédez comme suit :

1. Connectez-vous à la console.
2. Remplacez la région par l'ancienne région principale (`us-east-1`).
3. Accédez à la console Amazon RDS, sélectionnez le nom du cluster (`dbcluster-01`), puis choisissez Modifier.
4. Sous Protection contre la suppression, décochez la case Activer la protection contre la suppression, puis choisissez Continuer.
5. Choisissez Appliquer immédiatement, puis Modifier le cluster.

La machine DR `Orchestrator FAILBACK` d'état est invoquée pendant le processus de repli depuis l'ancienne région principale (`us-east-1`).

Pour effectuer le retour en arrière, procédez comme suit :

1. Connectez-vous à la console.
2. Remplacez la région par l'ancienne région principale (`us-east-1`).
3. Accédez à Services, puis sélectionnez Step Functions.
4. Accédez à la machine à DR `Orchestrator FAILBACK` états.
5. Choisissez Démarrer l'exécution, puis entrez le code JSON suivant dans la Input - optional section :

```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "CreateAuroraSecondaryDBCluster",
          "resourceName": "To create secondary Aurora MySQL Global Database Cluster",
          "parameters": {
            "GlobalClusterIdentifier": "!Import dr-globaldb-cluster-mysql-global-identifier",
            "DBClusterIdentifier": "!Import dr-globaldb-cluster-mysql-cluster-identifier",
            "DBClusterName": "!Import dr-globaldb-cluster-mysql-cluster-name",
            "SourceDBClusterIdentifier": "!Import dr-globaldb-cluster-mysql-source-cluster-identifier",

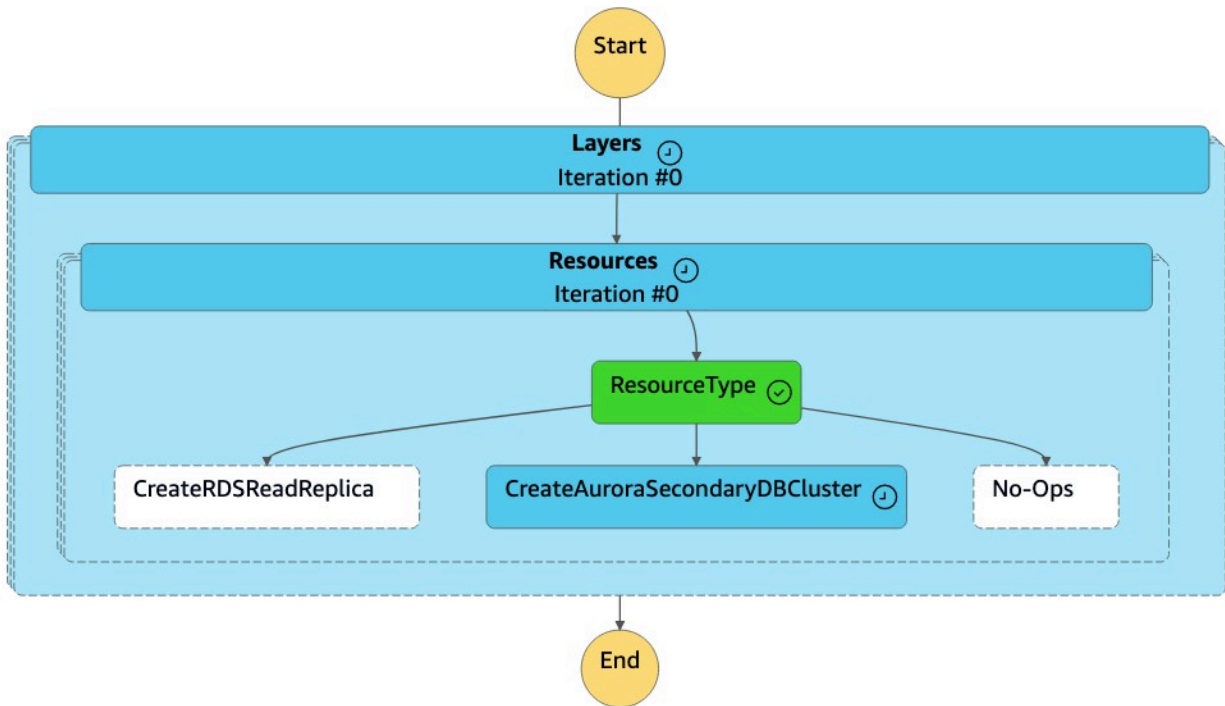
```

```

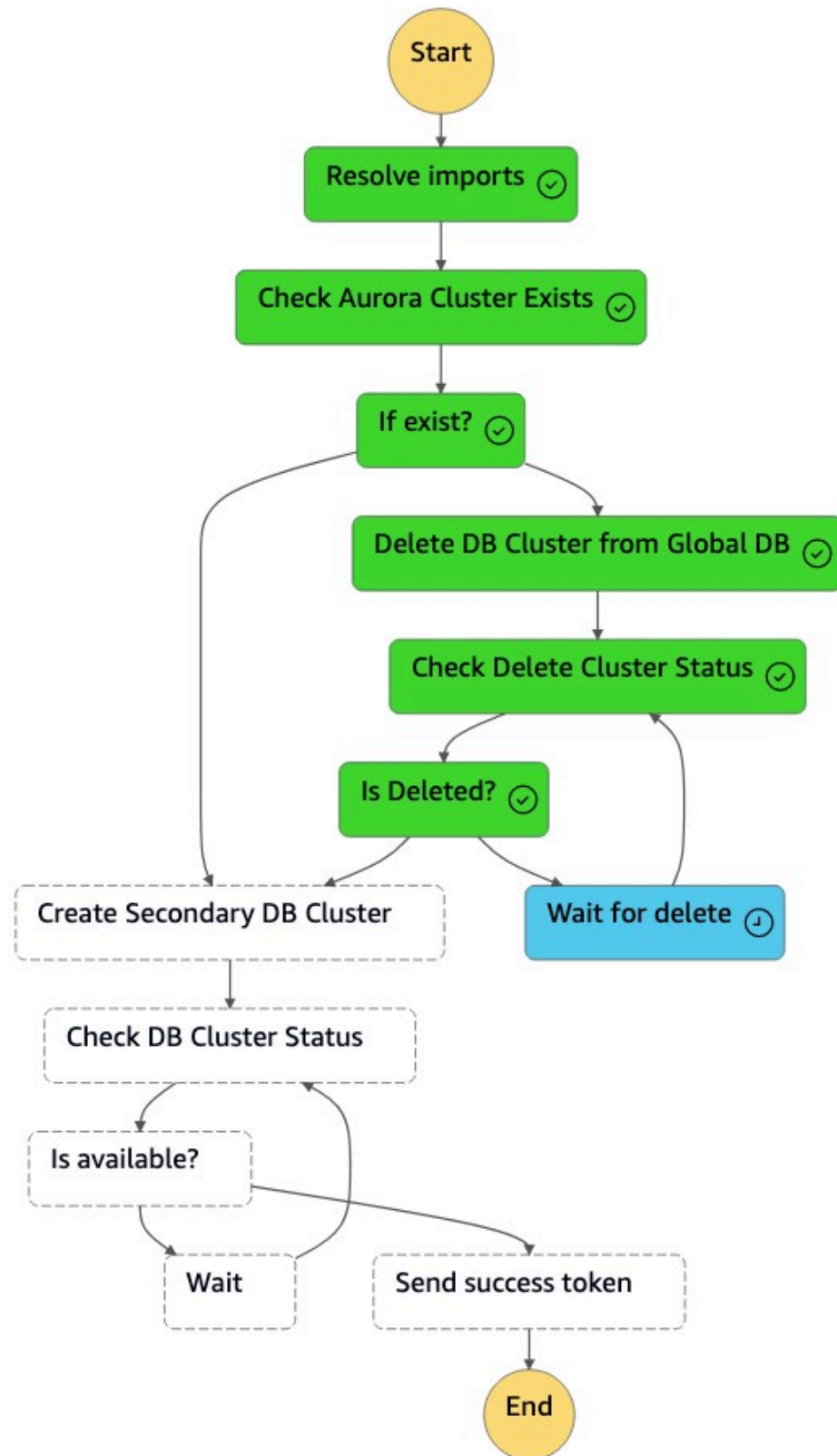
        "DBInstanceIdentifier": "!Import dr-globaldb-cluster-mysql-instance-
identifiant",
        "Port": "!Import dr-globaldb-cluster-mysql-port",
        "DBInstanceClass": "!Import dr-globaldb-cluster-mysql-instance-class",
        "DBSubnetGroupName": "!Import dr-globaldb-cluster-mysql-subnet-group-
name",
        "VpcSecurityGroupIds": "!Import dr-globaldb-cluster-mysql-vpc-security-
group-ids",
        "Engine": "!Import dr-globaldb-cluster-mysql-engine",
        "EngineVersion": "!Import dr-globaldb-cluster-mysql-engine-version",
        "KmsKeyId": "!Import dr-globaldb-cluster-mysql-KmsKeyId",
        "SourceRegion": "!Import dr-globaldb-cluster-mysql-source-region",
        "ClusterRegion": "!Import dr-globaldb-cluster-mysql-cluster-region",
        "BackupRetentionPeriod": "7",
        "MonitoringInterval": "60",
        "StorageEncrypted": "True",
        "EnableIAMDatabaseAuthentication": "True",
        "DeletionProtection": "True",
        "CopyTagsToSnapshot": "True",
        "AutoMinorVersionUpgrade": "True",
        "MonitoringRoleArn": "!Import rds-mysql-instance-RDSMonitoringRole"
    }
}
]
}
]
}

```

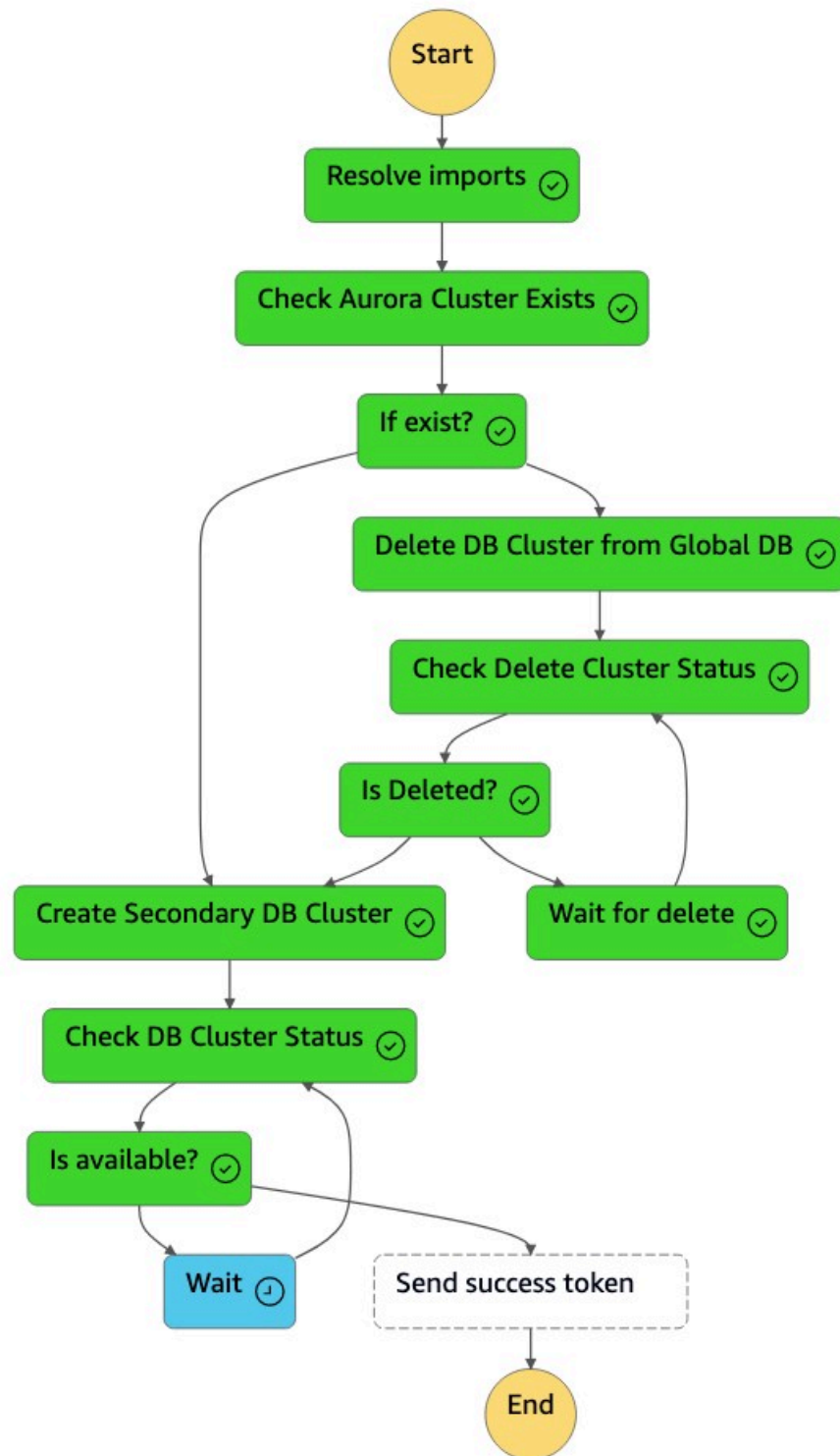
6. La machine DR Orchestrator FAILBACK d'état lit le type de ressource comme `CreateAuroraSecondaryDBCluster`, et elle appelle la machine `dr-orchestrator-stepfunction-create-Aurora-Secondary-cluster` d'état.



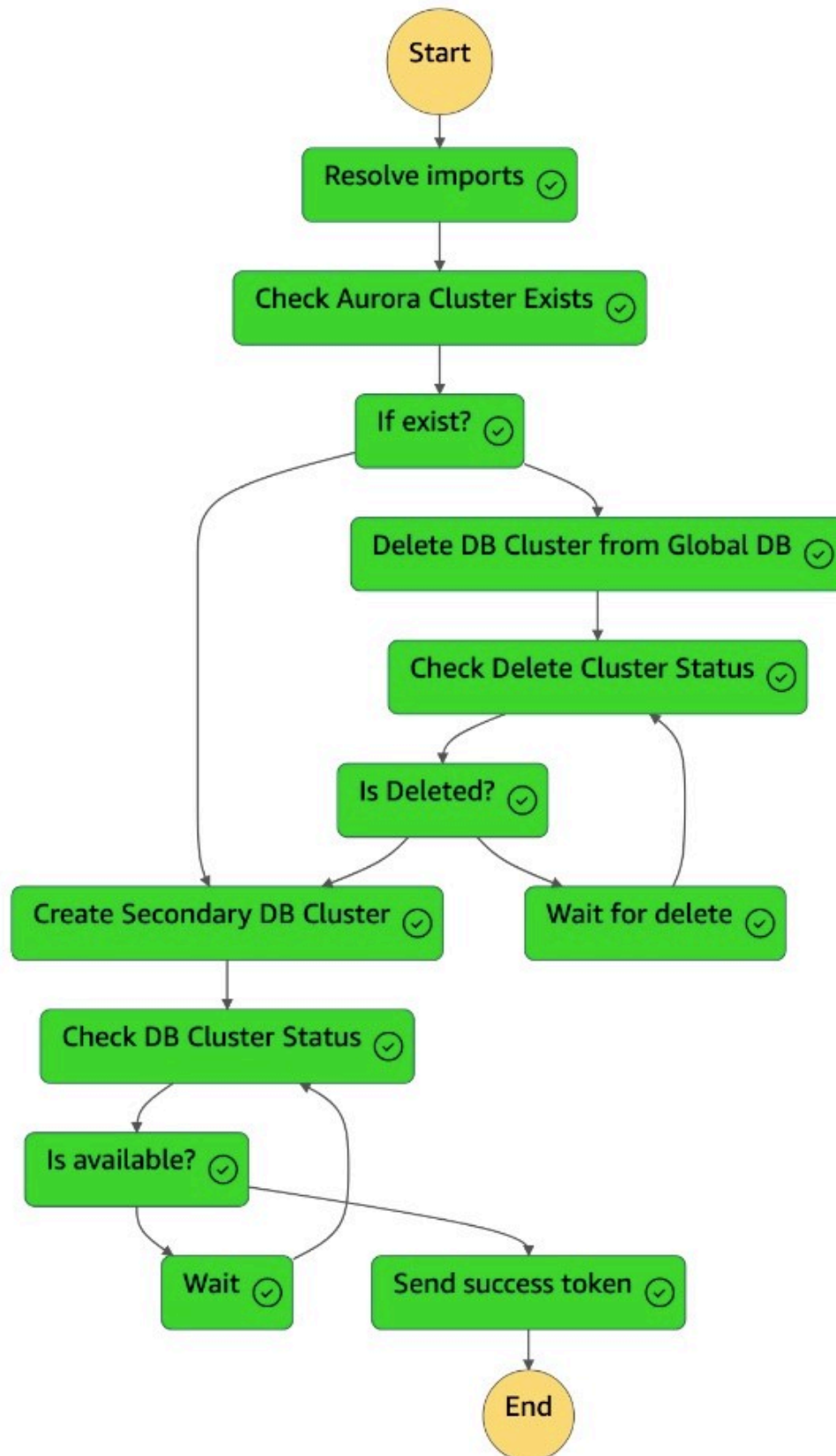
7. La machine `dr-orchestrator-stepfunction-create-Aurora-Secondary-cluster` d'état supprime le cluster existant (`dbcluster-01`) de l'ancienne région principale (`us-east-1`).



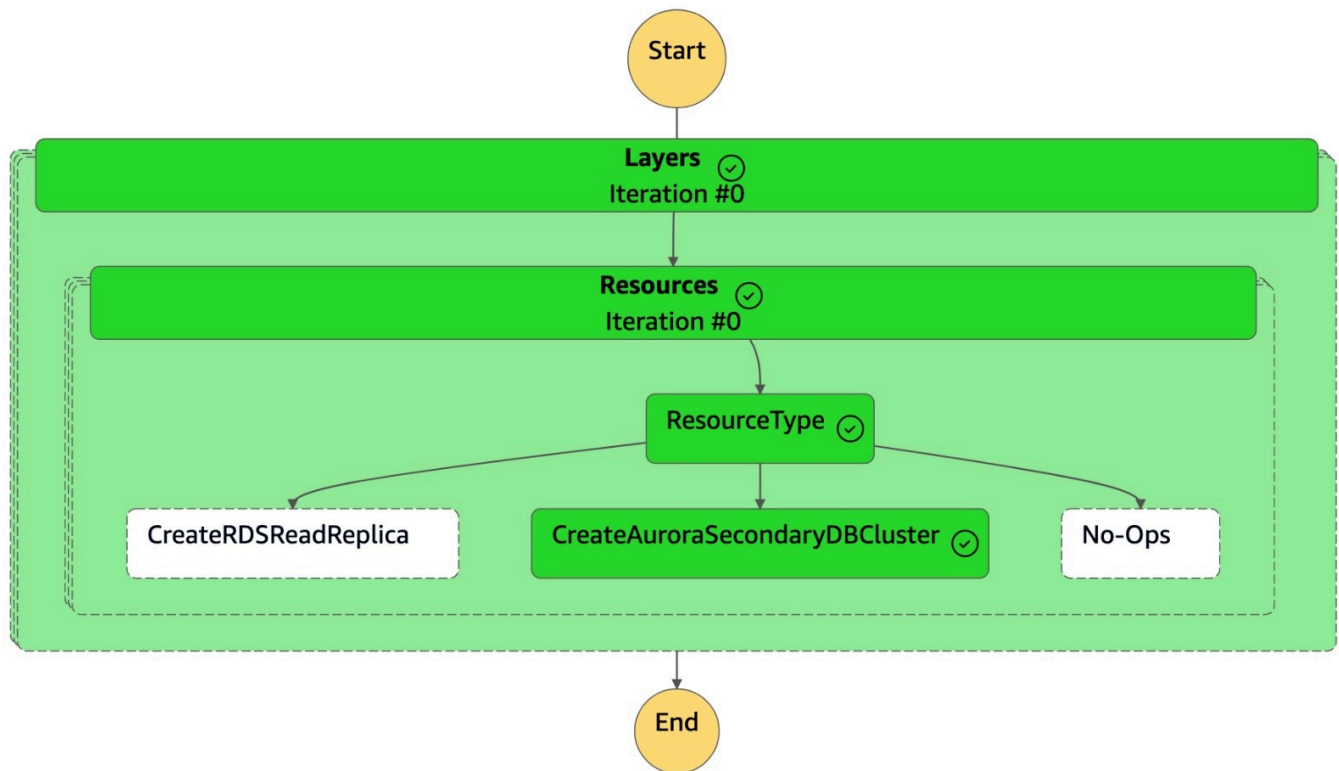
8. Une fois le cluster (dbcluster-01) supprimé, la machine d'état crée un nouveau cluster (dbcluster-01) avec l'instance de base de données, et il rejoint la base de données globale Aurora en tant que cluster secondaire pour traiter les charges de travail en lecture seule.



9. Une fois que le cluster secondaire est disponible, la machine d'`dr-orchestrator-stepfunction-create-Aurora-Secondary-cluster` état est terminée et renvoie un jeton de réussite à la machine DR Orchestrator Failback d'état.



10 La machine à dr-orchestrator-stepfunction-FAILBACK états est terminée.



11. Vous pouvez vérifier la base de données globale Aurora sur la console Amazon RDS.

[Si vous souhaitez déplacer le cluster de base de données principal vers us-east-1, vous pouvez suivre les étapes mentionnées dans la section Switchover.](#)

Déployez la solution

Après avoir configuré les prérequis, suivez les instructions du modèle [DR Orchestrator Framework](#). Des instructions sont incluses pour Amazon RDS Aurora et Amazon ElastiCache. Pour tester la solution, passez en revue les conditions préalables et les limites, puis déployez des exemples de bases de données sur AWS.

Prérequis

- Un actif [Compte AWS](#) pour déployer cette solution.
- Autorisations pour créer des compartiments [Amazon Simple Storage Service \(Amazon S3\)](#) et les ressources utilisées dans les modèles. [AWS CloudFormation](#)
- Avant d'exécuter la machine à DR Orchestrator FAILBACK états :
 - [Modifiez manuellement l'RDSinstance](#) ou [modifiez le cluster de base de données Amazon Aurora](#), puis désactivez-le `DeletionProtection`.
 - Prenez un [instantané](#) de base de données avant de supprimer l'RDSinstance Amazon ou le cluster Aurora si vous prévoyez d'effectuer une analyse des causes premières (RCA).

Limites

- La disponibilité et le support des fonctionnalités varient selon les versions spécifiques de chaque moteur de base de données et selon les versions Régions AWS. Pour plus d'informations sur la disponibilité des fonctionnalités et des régions pour la réplication entre régions, voir [Répliques de lecture entre régions](#).
- Cette solution a été testée à l'aide d'une instance de base de données principale avec une réplique en lecture. Si vous souhaitez utiliser plusieurs répliques de lecture, testez soigneusement la solution avant de l'implémenter dans un environnement de production.
- Pour AmazonRDS, la solution ne prend en charge que les moteurs suivants : [Amazon RDS for My SQL](#), [Amazon RDS pour Postgre SQL](#) et [Amazon RDS pour MariaDB](#).

Déployez des exemples de bases de données AWS

Pour démontrer la solution de reprise après sinistre interrégionale pour les bases de données sur AWS, vous pouvez déployer [Amazon RDS for My SQL](#), [Amazon Aurora My SQL -Compatible Edition](#) et [Amazon ElastiCache \(RedisOSS\)](#) - Global Datastore. Suivez les instructions contenues dans le

App-Stack README fichier du GitHub référentiel [aws-cross-region-dr-databases](#). Dans cet exemple, la région principale est us-east-1 et la région secondaire est us-west-2.

La pile d'applications utilise la section AWS CloudFormation [Outputs](#) pour exporter les paramètres utilisés par DR Orchestrator Framework pour automatiser le basculement et le retour en arrière des bases de données. AWS

FAQ

Trouvez les réponses aux questions concernant DR Orchestrator Framework pour le basculement et le retour sur incident.

Quels sont les RPO et RTO que je peux atteindre en utilisant cette approche ?

Pour plus d'informations sur l'objectif de temps de restauration (RTO) et l'objectif de point de restauration (RPO), consultez la stratégie [Stratégie de reprise après sinistre pour les bases de données sur AWS](#).

Est-il obligatoire d'utiliser des variables AWS CloudFormation d'exportation ?

Non, vous pouvez directement transmettre la valeur de la base de données globale Amazon Aurora ou de l'instance de base de données Amazon RDS directement au format JSON (par exemple :- "RDSInstanceIdentifier": "rds-mysql-instance").

Puis-je utiliser le flux de travail DR Orchestrator FAILOVER pour basculer sur plusieurs bases de données ? AWS

Oui, vous pouvez transmettre plusieurs ressources dans le fichier de paramètres d'entrée pour basculer sur plusieurs AWS bases de données. L'exemple de code suivant montre l'échec d'une réplique de lecture Amazon RDS for MySQL et d'une banque de données globale ElastiCache Amazon (Redis OSS) en parallèle :

```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "PromoteRDSReadReplica",
          "resourceName": "Promote RDS MySQL Read Replica",
```

```
    "parameters": {
      "RDSInstanceIdentifier": "!Import rds-mysql-instance-identifiant",
      "TargetClusterIdentifier": "!Import rds-mysql-instance-global-arn"
    },
    {
      "resourceType": "FailoverElastiCacheCluster",
      "resourceName": "Failover ElastiCache Cluster",
      "parameters": {
        "GlobalReplicationGroupId": "!Import demo-redis-cluster-global-replication-
group-id",
        "TargetRegion": "!Import demo-redis-cluster-target-region",
        "TargetReplicationGroupId": "!Import demo-redis-cluster-target-replication-
group-id"
      }
    }
  ]
}
```

Comment puis-je éviter l' InvalidParameterCombination erreur lorsque j'exécute la machine d'état DR Orchestrator FAILBACK pour Amazon RDS ?

Le texte complet de l'erreur est le suivant :

```
"errorMessage": "An error occurred (InvalidParameterCombination) when calling the DeleteDBInstance operation: Cannot delete protected DB Instance, please disable deletion protection and try again."
```

Pour éviter cette erreur, [modifiez l'instance RDS](#) en la désactivant DeletionProtection avant d'exécuter la machine DR Orchestrator FAILBACK d'état.

Étapes suivantes

Vous pouvez utiliser DR Orchestrator Framework pour ajouter un processus d'approbation avant de lancer les activités de basculement ou de retour en arrière. Par exemple, lorsque vous initiez le basculement lors d'un événement de reprise après sinistre, vous pouvez ajouter une machine à états par-dessus DR Orchestrator Failover pour envoyer une [notification Amazon Simple Notification Service \(Amazon SNS\)](#) par e-mail. Dès que l'approbation est accordée, l'activité de basculement commence.

Le basculement des applications peut également être intégré à DR Orchestrator Framework pour faire passer les applications d'entrée et de point DNS vers le nouveau point de terminaison de l'instance ou du cluster de base de données.

Vous pouvez utiliser le [AWS X-Ray](#) suivi pour obtenir la durée du basculement afin de calculer le RTO. Vous pouvez créer un tableau de bord de surveillance au-dessus de X-Ray.

Ressources

- [Réplication à Régions AWS l'aide de banques de données globales](#) (documentation Amazon ElastiCache (Redis OSS))
- [Using failover in an Amazon Aurora global database](#) (documentation Aurora)
- [Automatisez la gestion des points de terminaison de la base de données globale Amazon Aurora pour les basculements planifiés et imprévus](#) (article de blog)
- [Reprise après sinistre : 3 scénarios de basculement pour votre base de données globale Amazon Aurora avec Terraform \(partie 2\)](#) (article de blog)
- [Stratégie de reprise après sinistre pour les bases de données sur AWS](#) (stratégie d'orientation AWS prescriptive)
- [Automatisez le basculement et le retour en arrière entre régions à l'aide de DR Orchestrator Framework](#) (AWS modèle de guidage prescriptif)

Historique du document

Le tableau suivant décrit les modifications importantes apportées à ce guide. Pour être averti des mises à jour à venir, abonnez-vous à un [fil RSS](#).

Modification	Description	Date
Publication initiale	—	3 mai 2024

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.