



Otomatiskan solusi DR Anda untuk database relasional AWS

AWS Panduan Preskriptif



AWS Panduan Preskriptif: Otomatiskan solusi DR Anda untuk database relasional AWS

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

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan hak milik masing-masing pemiliknya, yang mungkin atau mungkin tidak terafiliasi, terkait dengan, atau disponsori oleh Amazon.

Table of Contents

Pengantar	1
Tujuan	1
Gambaran umum	3
Proses failover	4
Arsitektur DR Orchestrator FAILOVER	4
Proses Failback	5
Arsitektur DR Orchestrator FAILBACK	5
DR Orkestrator FAILOVER	7
Rincian parameter	8
DR Orchestrator GAGAL	9
Status Aurora dan mesin status Step Functions	12
Keadaan mantap	12
State kejadian	12
Peralihan	13
Failover manual yang tidak direncanakan	18
Kegagalan	22
Terapkan solusinya	31
Pertanyaan yang Sering Diajukan	33
Apa RPO dan RTO yang dapat saya capai dengan menggunakan pendekatan ini?	33
Apakah wajib menggunakan variabel AWS CloudFormation ekspor?	33
Dapatkah saya menggunakan alur kerja DR Orchestrator FAILOVER untuk gagal lebih dari satu database? AWS	33
Bagaimana saya bisa menghindari InvalidParameterCombination kesalahan saat menjalankan mesin status DR Orchestrator FAILBACK untuk Amazon RDS?	34
Langkah selanjutnya	35
Sumber daya	36
Riwayat dokumen	37
.....	xxxviii

Otomatiskan solusi DR Anda untuk database relasional AWS

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

Mei 2024 ([riwayat dokumen](#))

Saat merencanakan [strategi pemulihan bencana \(DR\)](#) di Amazon Web Services (AWS), Anda dapat menerapkan solusi DR untuk database di organisasi Anda. Anda dapat mengotomatiskan solusi DR dengan mengonfigurasi arsitektur berbasis peristiwa untuk melakukan failover instance database.

Manfaat dari pendekatan ini meliputi:

- Mengurangi kesalahan manusia yang disebabkan oleh koordinasi manual
- Memberikan tujuan waktu pemulihan sesingkat mungkin (RTO) dalam beberapa kasus
- Memfasilitasi pengujian berulang dari solusi DR Anda

Tanpa otomatisasi, pemadaman di primer Anda Wilayah AWS, akan membutuhkan sumber daya di organisasi Anda untuk melakukan langkah-langkah secara manual dalam run-book. Langkah-langkah tersebut mungkin mencakup beberapa proses, seperti mempromosikan replika baca Amazon Relational Database Service (Amazon RDS) di Wilayah AWS sekunder untuk berfungsi sebagai instance database utama yang baru. AWS menyediakan [AWS SDK for Python \(Boto3\)](#) untuk APIs mengotomatiskan tindakan ini, yang dapat membentuk blok bangunan solusi DR untuk kebutuhan organisasi Anda.

Panduan ini membahas orkestrator DR terpusat yang digerakkan oleh manifes yang dapat membantu Anda mencapai solusi DR untuk database di dalam yang sama. AWS Wilayah AWS Akun AWS Keputusan untuk melakukan failover masih membutuhkan intervensi manusia dan berada di luar cakupan otomatisasi.

Target audiens untuk panduan ini adalah arsitek aplikasi, arsitek infrastruktur, arsitek basis data dan konsultan database.

Tujuan

Menerapkan solusi DR Orchestrator Framework dapat membantu Anda mencapai hasil berikut:

- Mengurangi kesalahan manual selama pemadaman — Kinerja manual proses DR dapat rentan terhadap kesalahan. Mengotomatiskan tindakan selama fase aktivitas DR mengurangi kesalahan manual dan membantu memastikan failover atau fallback yang lebih lancar.

- **Kelola DR dalam skala besar** — Jika organisasi Anda memiliki armada database yang besar, otomatisasi bangunan untuk DR membantu Anda meningkatkan skala solusi DR Anda. Anda dapat mengotomatiskan solusi DR AWS database Anda dengan menggunakan DR Orchestrator Framework. DR Orchestrator mengotomatiskan langkah-langkah untuk failover atau failback AWS database dengan menggunakan file parameter dalam format JSON.

Ikhtisar DR Orchestrator Framework

DR Orchestrator Framework menyediakan solusi satu-klik untuk mengatur dan mengotomatiskan DR Lintas wilayah untuk database. AWS Ini menggunakan [AWS Step Functions](#) dan [AWS Lambda](#) untuk melakukan langkah-langkah yang diperlukan selama failover dan failback. Mesin status Step Functions memberikan dasar untuk pengambilan keputusan dalam desain orkestrator. Operasi API untuk melakukan tindakan failover atau failback dikodekan ke dalam fungsi Lambda yang dipanggil dari dalam mesin status. Fungsi Lambda menjalankan [AWS SDK untuk Python \(Boto3\)](#) API untuk berinteraksi dengan AWS database.

DR Orchestrator Framework berisi dua mesin status utama yang sesuai dengan fase failover dan failback.

Untuk Amazon RDS, fase failover mempromosikan replika baca RDS lintas wilayah menjadi instans DB mandiri. Untuk Amazon Aurora, ketika Wilayah utama turun selama pemadaman yang jarang dan tidak terduga, simpul penulisnya tidak tersedia. Replikasi antara node penulis dan cluster sekunder berhenti. Anda harus melepaskan cluster sekunder dari database global dan mempromosikannya sebagai cluster mandiri. Aplikasi dapat menghubungkan dan mengirim lalu lintas tulis ke cluster mandiri. Anda dapat menggunakan proses yang sama ini untuk [beralih ke](#) cluster DB primer dari database global ke Regions sekunder. Gunakan pendekatan ini untuk skenario terkontrol seperti berikut:

- Pemeliharaan operasional
- Prosedur operasional yang direncanakan
- Promosi klaster sekunder Amazon ElastiCache (Redis OSS) sebagai klaster utama baru Anda

Fase failback menetapkan replikasi langsung data antara Wilayah primer hidup dan Wilayah sekunder baru.

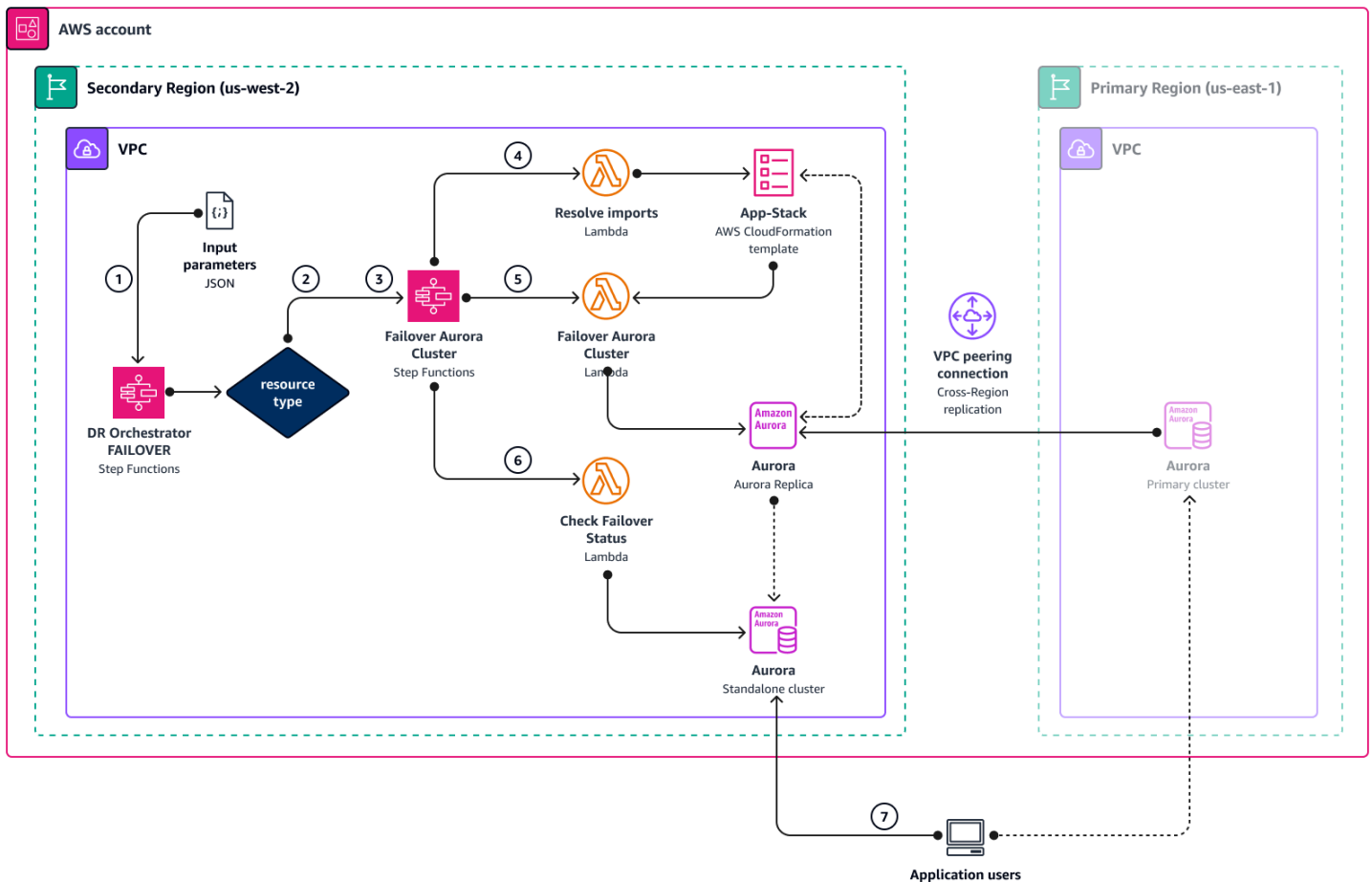
Sangat penting untuk memahami bahwa DR Orchestrator hanya berlaku untuk database. Semua aplikasi yang mereferensikan database ini dan berada di Wilayah yang sama mungkin memerlukan solusi failover tandem yang terpisah. Setelah database gagal ke Wilayah sekunder, aplikasi perlu diperbarui untuk terhubung ke instance database baru, yang akan berfungsi sebagai sumber data.

Proses failover

Untuk melakukan failover, jalankan DR Orchestrator FAILOVER state machine. Pada tahap ini, database sekunder sudah ada di Wilayah sekunder, baik sebagai replika baca (Amazon RDS) atau sebagai cluster sekunder (Amazon Aurora). Ketika Anda menjalankan mesin DR Orchestrator FAILOVER negara, itu mempromosikan database sekunder untuk menjadi yang utama.

Arsitektur DR Orchestrator FAILOVER

Diagram berikut menunjukkan konsep proses failover untuk Amazon Aurora saat menggunakan DR Orchestrator. Amazon Aurora dan Amazon ElastiCache menggunakan alur kerja yang sama tetapi dengan mesin status dan fungsi Lambda yang berbeda.



1. Mesin DR Orchestrator FAILOVER status membaca parameter input JSON.
2. Berdasarkan resourceType parameter, mesin negara memanggil mesin negara lain:Promote RDS Read Replica,Failover Aurora Cluster, atauFailover ElastiCache. Jika lebih dari satu sumber daya dilewatkan dalam input, mesin status ini berjalan secara paralel.

3. Mesin **Failover Aurora Cluster** negara memanggil fungsi Lambda di masing-masing dari tiga langkah berikut.
4. Fungsi **Resolve imports** Lambda menyelesaikan `"! import <export-variable-name>"` dengan nilai aktual dari template. App-Stack AWS CloudFormation
5. Fungsi **Failover Aurora Cluster** Lambda mempromosikan replika baca sebagai instans DB mandiri.
6. Fungsi **Check Failover Status** Lambda memeriksa status instans DB yang dipromosikan. Setelah status **TERSEDIA**, fungsi Lambda mengirimkan token sukses kembali ke mesin status panggilan dan selesai.
7. Anda dapat mengarahkan aplikasi Anda ke database mandiri di DR Region (`us-west-2`), yang sekarang menjadi database utama.

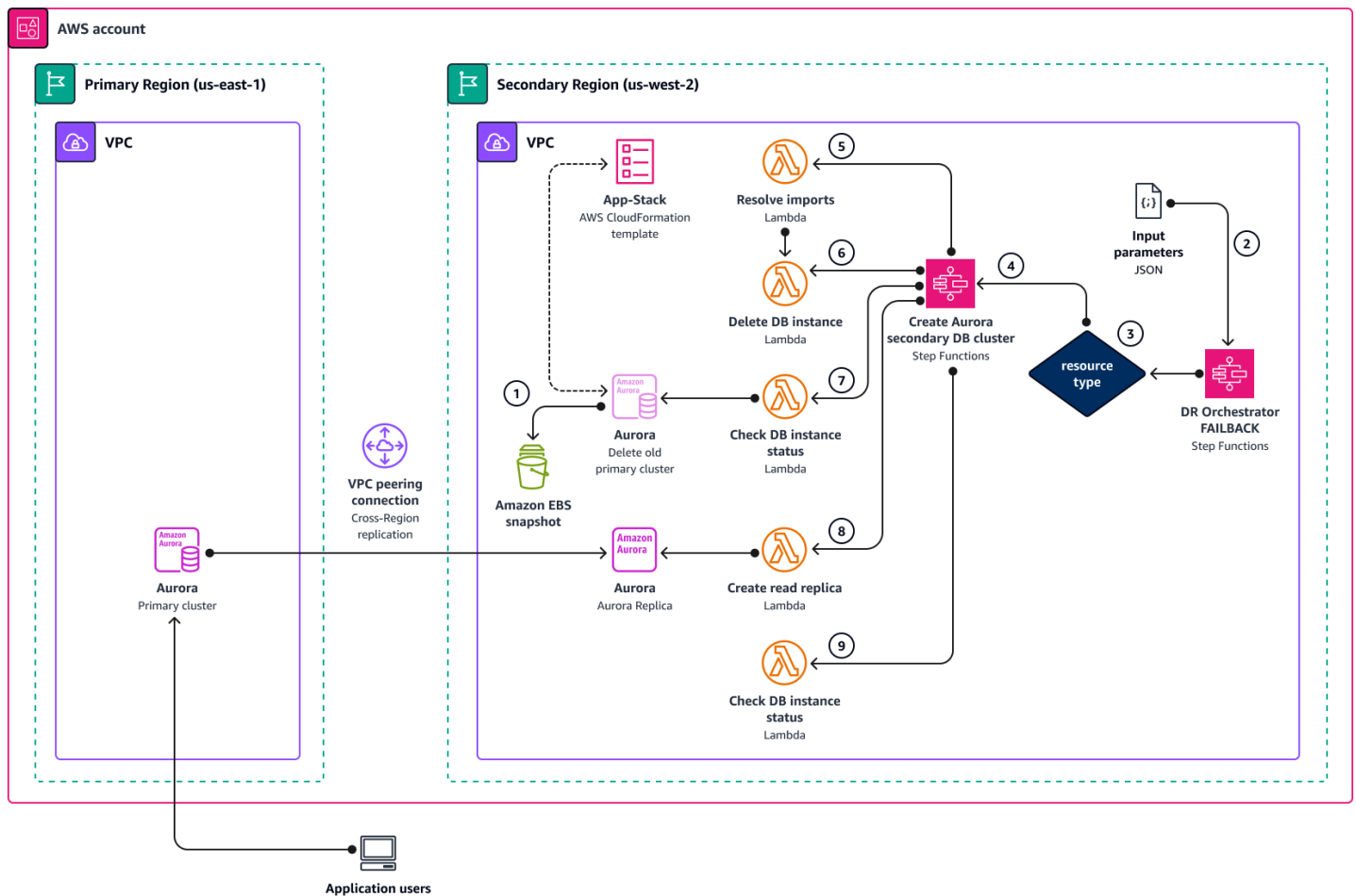
Proses Failback

Setelah Region (`us-east-1`) primer Anda aktif lagi, Anda dapat gagal kembali ke sana, sehingga database `us-east-1` menjadi yang utama lagi. Untuk memulai failback, jalankan mesin DR **Orchestrator FAILBACK** status. Seperti namanya, mesin status ini mulai mereplikasi perubahan di Region (`us-west-2`) primer baru Anda kembali ke Region (`us-east-1`) primer sebelumnya, yang bertindak sebagai sekunder saat ini.

Setelah replikasi dibuat antara dua Wilayah, Anda dapat memulai failback. Untuk gagal kembali dan kembali ke Region (`us-east-1`) primer asli Anda, jalankan mesin DR **Orchestrator FAILOVER** status di Region sekunder saat ini (`us-east-1`) untuk mempromosikannya ke Region primer.

Arsitektur DR Orchestrator FAILBACK

Diagram berikut menunjukkan konsep proses failback untuk Amazon Aurora saat menggunakan DR Orchestrator.



1. Sebelum memulai failback, ambil snapshot DB manual untuk digunakan saat melakukan analisis akar penyebab (RCA).

Selain itu `DeletionProtection`, nonaktifkan cluster Aurora di Region () us-east-1 primer sebelumnya.

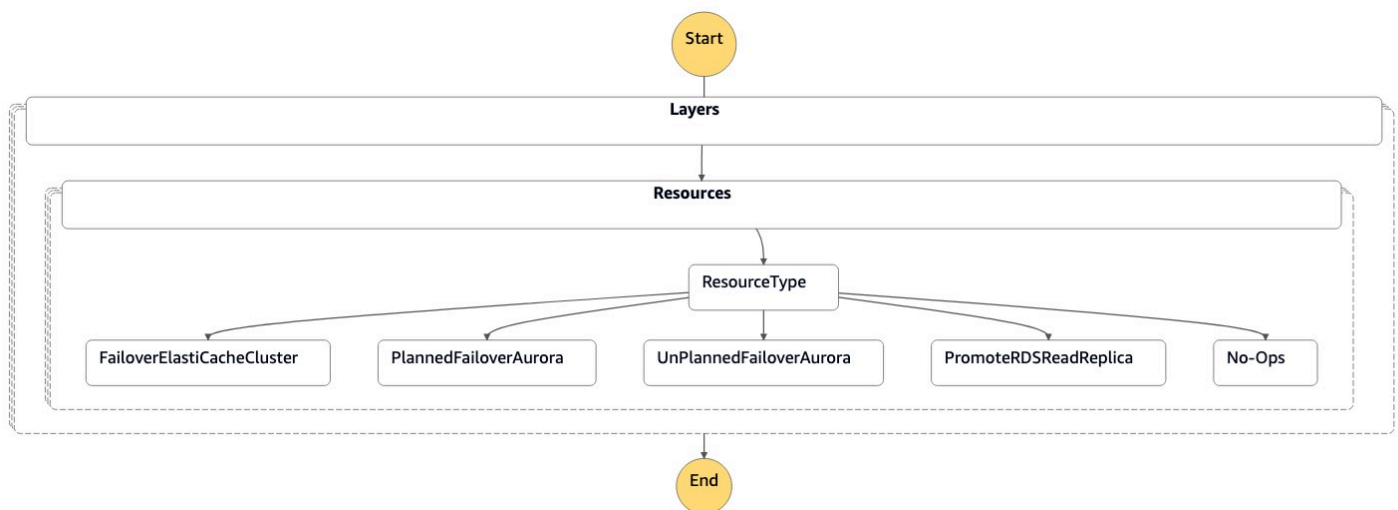
2. Mesin DR Orchestrator FAILBACK status membaca parameter input JSON.
3. Berdasarkan `resourceType`, mesin DR Orchestrator FAILBACK negara memanggil mesin Create Aurora Secondary DB Cluster negara.
4. Mesin Create Aurora Secondary DB Cluster negara memanggil fungsi Lambda di masing-masing dari lima langkah berikut.
5. Fungsi `Resolve import` Lambda menyelesaikan `! import <export-variable-name>` dengan nilai aktual dari template. App-Stack CloudFormation
6. Fungsi `Delete DB Instance` Lambda menghapus contoh primer sebelumnya.

7. Fungsi Check DB instance status Lambda memeriksa Delete DB Instance status sampai DB dihapus.
8. Fungsi Create Read Replica Lambda membuat replika baca di Region sekunder dari instans DB yang ada di Region primer baru.
9. Fungsi Check DB instance status Lambda memeriksa status instans DB replika baca. Ketika status TERSEDIA, fungsi Lambda mengirimkan token sukses kembali ke mesin status panggilan, yang selesai.

DR Orkestrator FAILOVER

Gunakan mesin DR Orchestrator FAILOVER negara dalam acara DR ketika Region utama (us-east-1) sedang down atau selama acara yang direncanakan seperti pemeliharaan operasional.

Fungsi ini dapat dipanggil untuk gagal melalui satu atau beberapa database secara paralel.



Mesin status menerima parameter dalam format JSON seperti yang ditunjukkan pada kode berikut:

```

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

```

```

        "RDSInstanceIdentifier": "!Import rds-mysql-instance-identifier",
        "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"
        }
    }
]
}

```

Rincian parameter

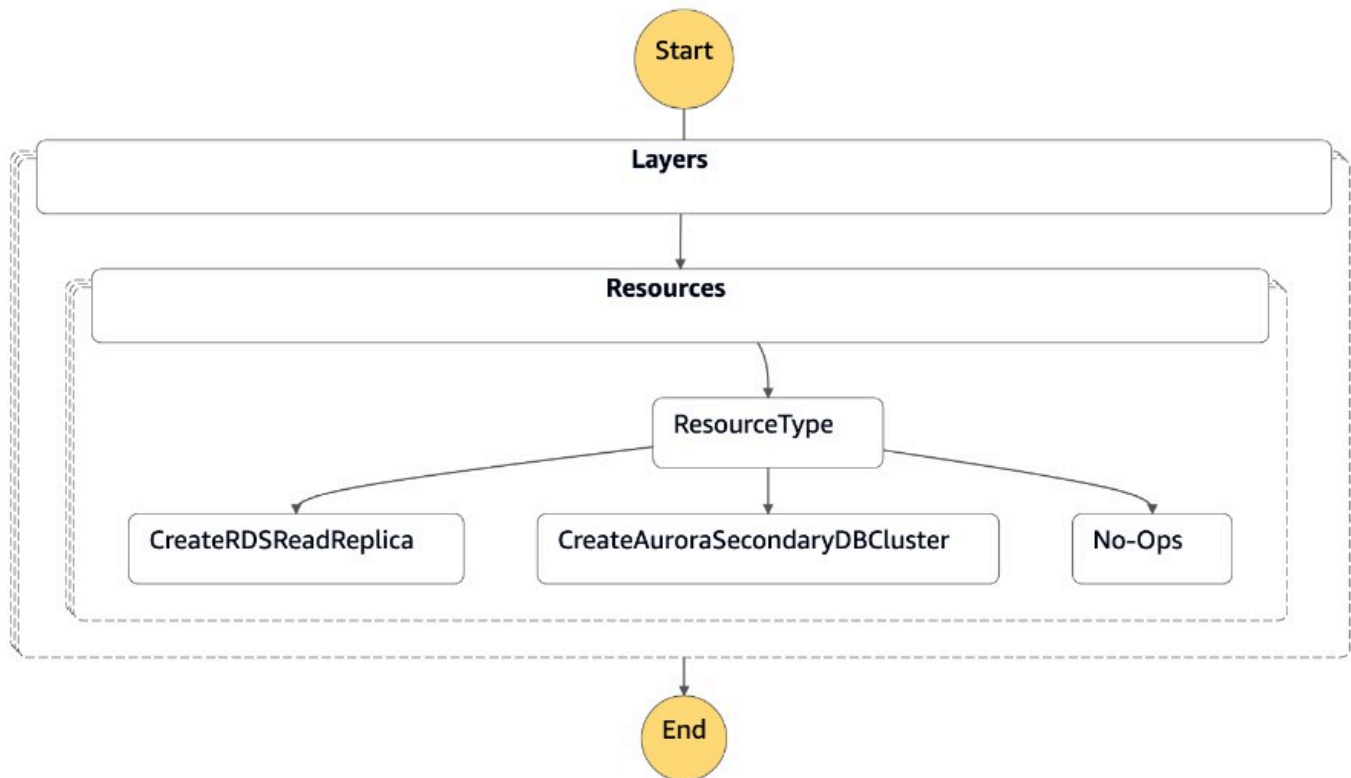
Tabel berikut menunjukkan parameter yang digunakan oleh mesin DR Orchestrator FAILOVER negara.

Nama parameter	Deskripsi	Nilai yang diharapkan
layer(diperlukan: nomor)	Urutan pemrosesan. Semua sumber daya yang didefinisikan dalam lapisan 1 harus dijalankan sebelum sumber daya lapisan 2 dijalankan.	1 atau 2, dan seterusnya
sumber daya (diperlukan: array kamus)	Semua sumber daya dalam satu lapisan berjalan secara paralel.	<pre> { "resourceType": " String", "resourceName": "St ring", "parameters": { "<param1>": "<!Import cft-output-1">, </pre>

Nama parameter	Deskripsi	Nilai yang diharapkan
		<pre>.... }</pre>
resourceType (diperlukan: string)	Jenis sumber daya untuk mengidentifikasi sumber daya	PromoteRDSReadReplica atau FailoverElastiCacheCluster
resourceName (opsional: string)	Untuk mengidentifikasi portofolio aplikasi mana sumber daya ini dimiliki	Promote RDS for MySQL Read Replica
parameter (diperlukan: array kamus)	Daftar parameter yang diperlukan untuk gagal atau gagal kembali AWS database	<pre>{ "<param1>" : "<!Import cft- output-1>", "<param2>" : "<!Import cft- output-2>", }</pre>

DR Orchestrator GAGAL

Gunakan mesin DR Orchestrator FAILBACK status setelah acara DR, ketika Region (us-east-1) primer sebelumnya sudah habis. Anda dapat membuat [replika baca](#) untuk Amazon RDS di Wilayah primer sebelumnya dari Region (us-west-2) primer baru agar sesuai dengan strategi DR Anda. Karena ini adalah acara yang direncanakan, Anda dapat menjadwalkan kegiatan ini selama akhir pekan atau selama jam kerja di luar puncak dengan perkiraan waktu henti.



Mesin status menerima parameter dalam format JSON seperti yang ditunjukkan pada kode berikut:

```

{
  "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"
    }
}
]
```

Status Aurora dan mesin status Step Functions

Bagian ini mencakup proses dan mesin status khusus untuk gagal dan gagal kembali cluster Amazon Aurora. Cluster dikonfigurasi sebagai database global.

Note

Untuk tujuan demonstrasi, contoh ini menggunakan Edisi MySQL-Compatible Aurora. Anda dapat menggunakan langkah serupa untuk Edisi Aurora PostgreSQL-Compatible .

Keadaan mantap

Dalam kondisi mapan, database MySQL-Compatible global Amazon Aurora (`dr-globaldb-cluster-mysql`) telah dibuat dengan dua cluster DB. Cluster DB pertama (`db-cluster-01`) telah dibuat di primary Wilayah AWS (`us-east-1`) untuk melayani read/write beban kerja. Cluster DB kedua (`db-cluster-02`) telah dibuat di Region sekunder (`us-west-2`) untuk server beban kerja hanya-baca.

Selain menyediakan solusi DR, Anda dapat mengurangi beban pada cluster DB primer Anda dengan merutekan kueri baca dari aplikasi Anda ke cluster DB sekunder. Masing-masing cluster ini berisi satu instance database yang disebut `dbcluster-01-use1-instance-1` dan `dbcluster-02-usw2-instance-2`, masing-masing.

State kejadian

Dengan menggunakan database global Amazon Aurora, Anda dapat merencanakan dan memulihkan diri dari bencana dengan cukup cepat. Pemulihan dari bencana biasanya diukur menggunakan nilai untuk tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO). Untuk informasi selengkapnya, lihat [Menggunakan switchover atau failover dalam database global Amazon Aurora](#).

Dengan database global Aurora, ada dua pendekatan berbeda untuk failover:

- Switchover (failover terencana terkelola)
- Failover (failover manual yang tidak direncanakan, atau lepaskan dan promosikan)

Peralihan

Switchover ditujukan untuk lingkungan yang terkendali, seperti pemeliharaan operasional dan prosedur operasional terencana lainnya. Dengan menggunakan failover terencana terkelola, Anda dapat memindahkan cluster DB utama dari database global Aurora Anda ke salah satu Wilayah sekunder. Karena switchover menunggu sampai cluster DB sekunder disinkronkan dengan database utama, RPO adalah 0 (tidak ada kehilangan data). Untuk mempelajari selengkapnya, lihat [Melakukan switchover untuk database global Amazon Aurora](#).

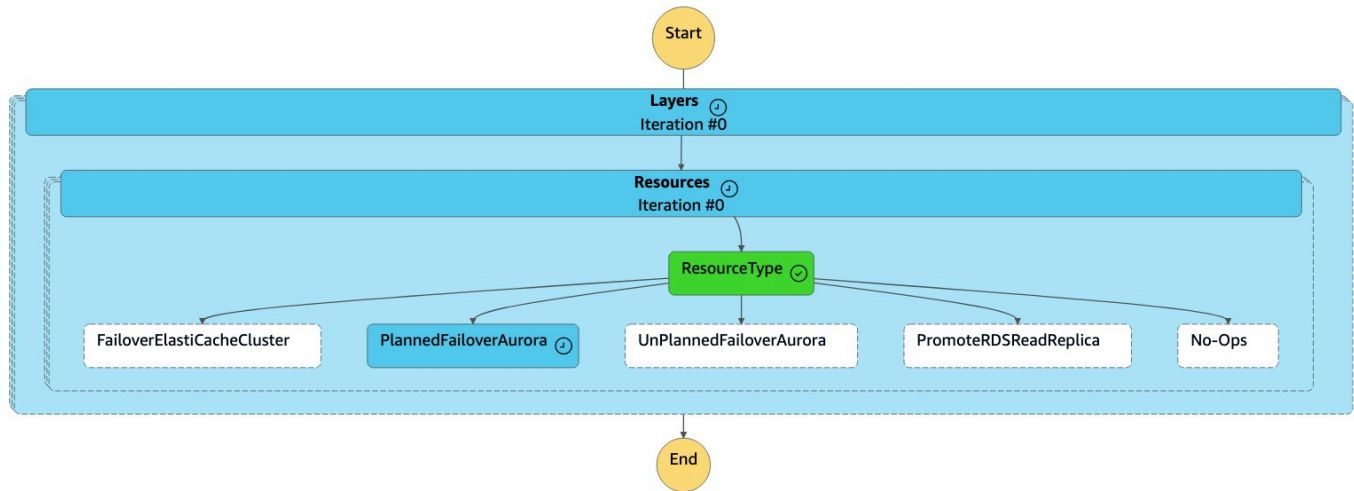
Mesin `dr-orchestrator-stepfunction-FAILOVER` status dipanggil selama status peristiwa untuk mengalihkan cluster utama Anda ke Region sekunder yang Anda pilih (`us-west-2`).

Untuk melakukan peralihan, lakukan hal berikut:

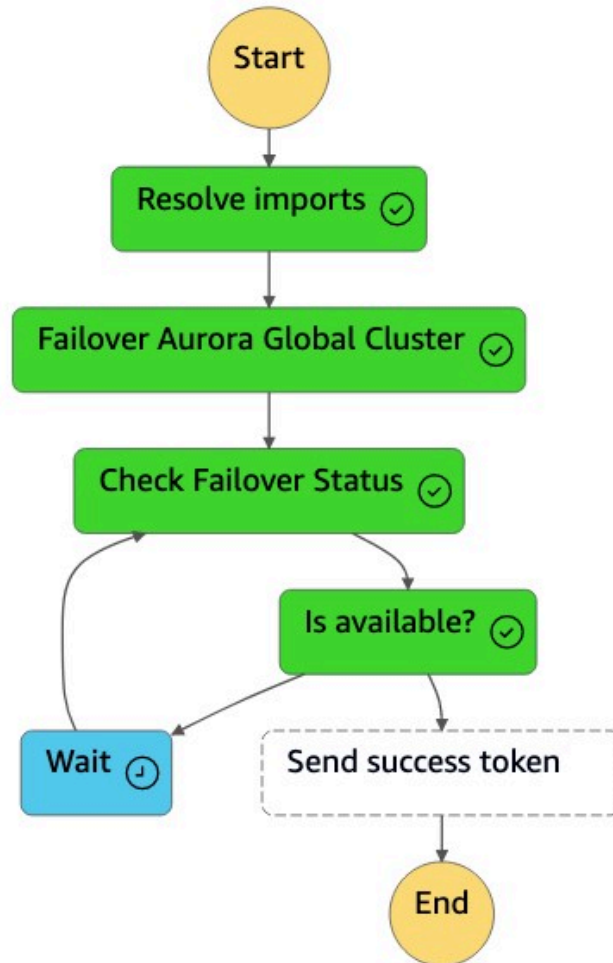
1. Masuk ke Konsol Manajemen AWS.
2. Ubah Wilayah ke Wilayah DR (`us-west-2`).
3. Arahkan ke Layanan, dan pilih Step Functions.
4. Arahkan ke mesin `dr-orchestrator-stepfunction-FAILOVER` negara.
5. Pilih Mulai eksekusi, dan masukkan kode JSON berikut di Input - optional bagian:

```
{
  "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. Mesin `dr-orchestrator-stepfunction-FAILOVER` status membaca tipe sumber daya sebagai `PlannedFailoverAuroraMySQL`, dan memanggil mesin `dr-orchestrator-stepfunction-planned-Aurora-failover` status untuk gagal atas database global Aurora.



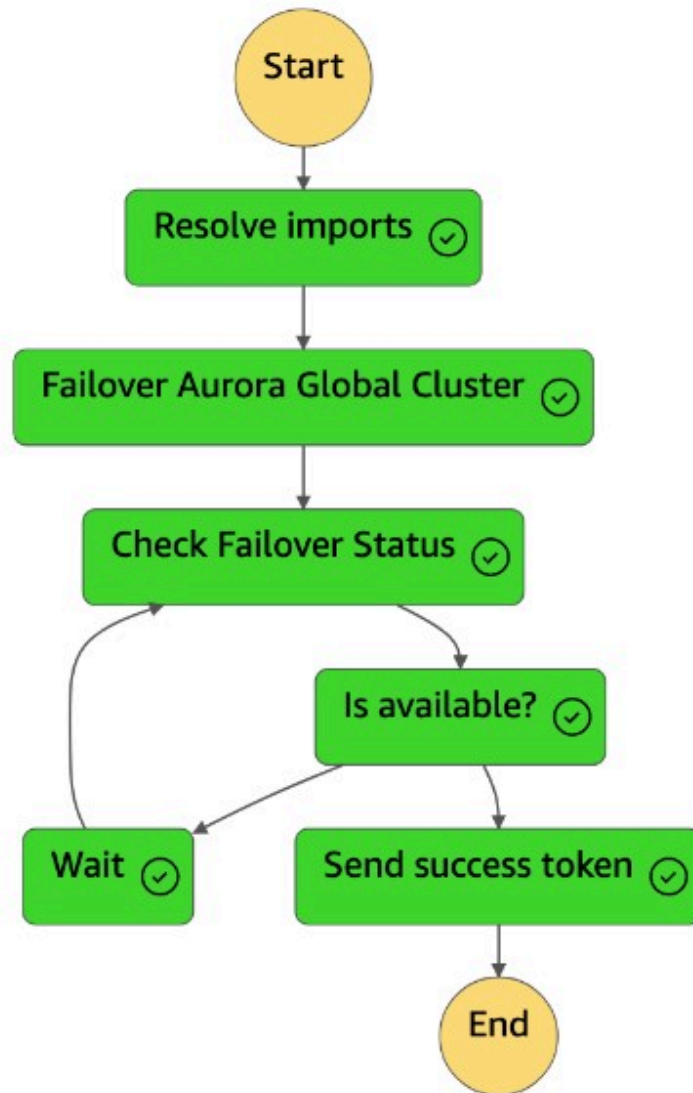
7. Mesin `dr-orchestrator-stepfunction-planned-Aurora-failover` negara melakukan langkah-langkah berikut untuk beralih ke peran database MySQL-Compatible global Aurora.



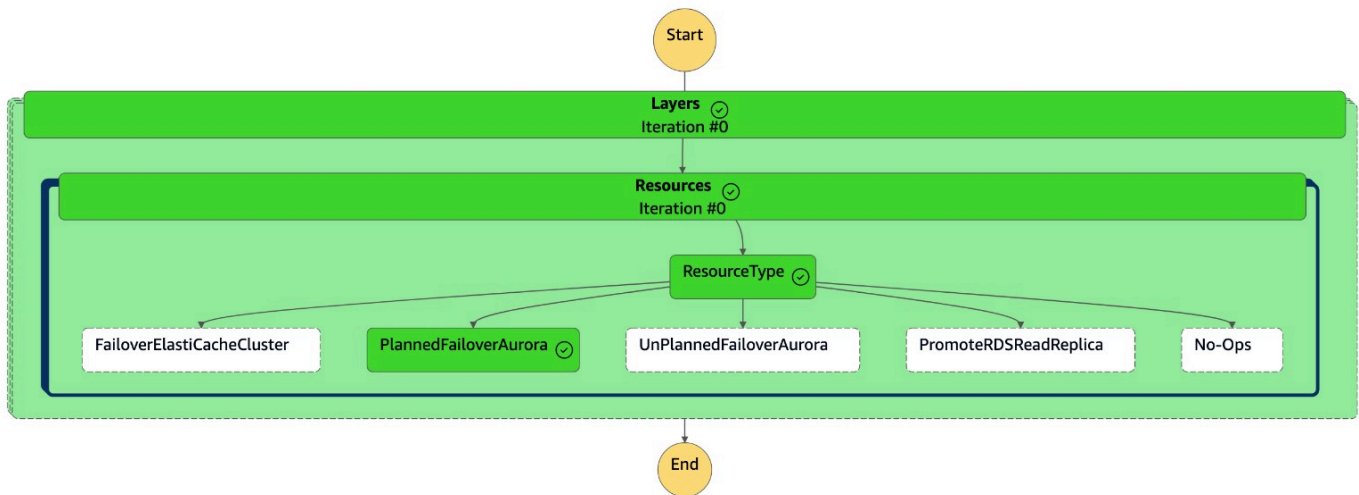
Langkah	Deskripsi	Nilai yang Diharapkan
Selesaikan impor	Fungsi Lambda mengganti kan !Import <variable name> nilai dengan nama sebenarnya.	"!Import dr-global db-cluster-mysql-global-identifier" digantikan oleh "dr-globaldb-cluster-mysql" .

Langkah	Deskripsi	Nilai yang Diharapkan
Gugus Global Failover Aurora	Fungsi Lambda memanggil failover_global_cluster Boto3 API untuk gagal melalui database global Aurora.	<pre>{ 'GlobalCluster': { 'GlobalClusterId entifier': 'dr-globa ldb-cluster-mysql', 'GlobalClusterRe sourceId': 'cluster- cce7f9bec2846db4', 'GlobalClusterArn': 'arn:aws:rds::xxx', 'Status': 'failing- over', } }</pre>
Periksa Status Failover	Fungsi Lambda memanggil describe_db_clusters Boto3 API untuk memeriksa status failover.	memodifikasi, tersedia
Kirim token sukses	Fungsi Lambda memanggil send_task_success Boto3 API dan mengirimkan token sukses kembali ke mesin status. DR Orchestrator Failover	h7XRIcDltd/83p1E0d MccoxlzFhglsdkzpK9 mBVKZsp7d9yrT1W

- Arahkan ke konsol Amazon RDS. Di bawah Status, nilai untuk database global Aurora akan berubah dari Tersedia menjadi Switching over atau Modifying.
- Setelah mesin `dr-orchestrator-stepfunction-planned-Aurora-failover` negara selesai, ia mengirimkan token sukses kembali ke mesin `dr-orchestrator-stepfunction-FAILOVER` negara.



10Mesin dr-orchestrator-stepfunction-FAILOVER negara selesai.



Di konsol, peran cluster Secondary (`dbcluster-02`) sekarang adalah klaster Primer, dan klaster siap melayani read/write beban kerja. Peran cluster primer asli (`dbcluster-01`) sekarang terdaftar sebagai cluster Sekunder.

Failover manual yang tidak direncanakan

Pada kesempatan langka, database global Aurora Anda mungkin mengalami pemadaman tak terduga pada awalnya. Wilayah AWS Jika hal ini terjadi, klaster DB Aurora primer dan simpul penulisnya tidak tersedia, dan replikasi antara klaster primer dan kedua berhenti. Untuk meminimalkan downtime (RTO) dan kehilangan data (RPO), bekerja dengan cepat untuk melakukan failover lintas wilayah dan merekonstruksi basis data global Aurora Anda. Untuk informasi selengkapnya, lihat [Memulihkan database global Amazon Aurora dari pemadaman yang tidak direncanakan](#).

Melakukan failover yang tidak direncanakan mengharuskan Anda melepaskan cluster sekunder Anda dari database global Aurora. Sebelum Anda melakukan failover yang tidak direncanakan, hentikan penulisan aplikasi di cluster Aurora DB utama Anda. Setelah failover selesai dengan sukses, konfigurasi ulang aplikasi untuk menulis ke cluster DB primer yang baru. Pendekatan ini membantu mencegah kehilangan data. Ini juga membantu menghindari inkonsistensi data jika node penulis utama kembali online selama proses failover.

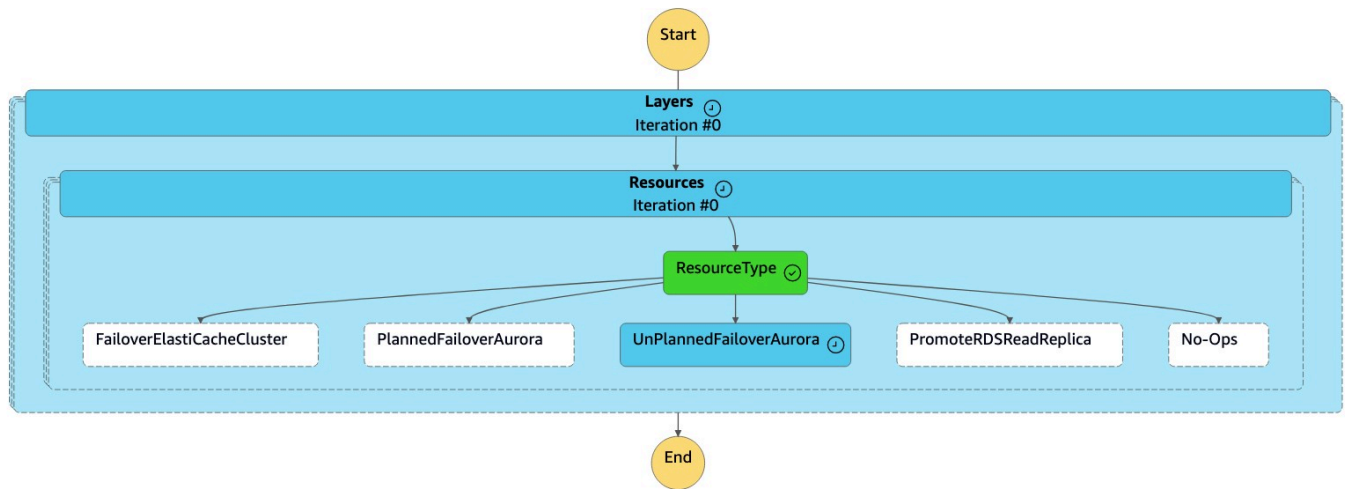
Untuk melakukan failover yang tidak direncanakan, hubungi mesin `dr-orchestrator-stepfunction-FAILOVER` negara. Untuk contoh ini, cluster Sekunder (`db-cluster-02`) berada di DR Region (`us-west-2`) dalam keadaan mapan.

Untuk melakukan failover, lakukan hal berikut:

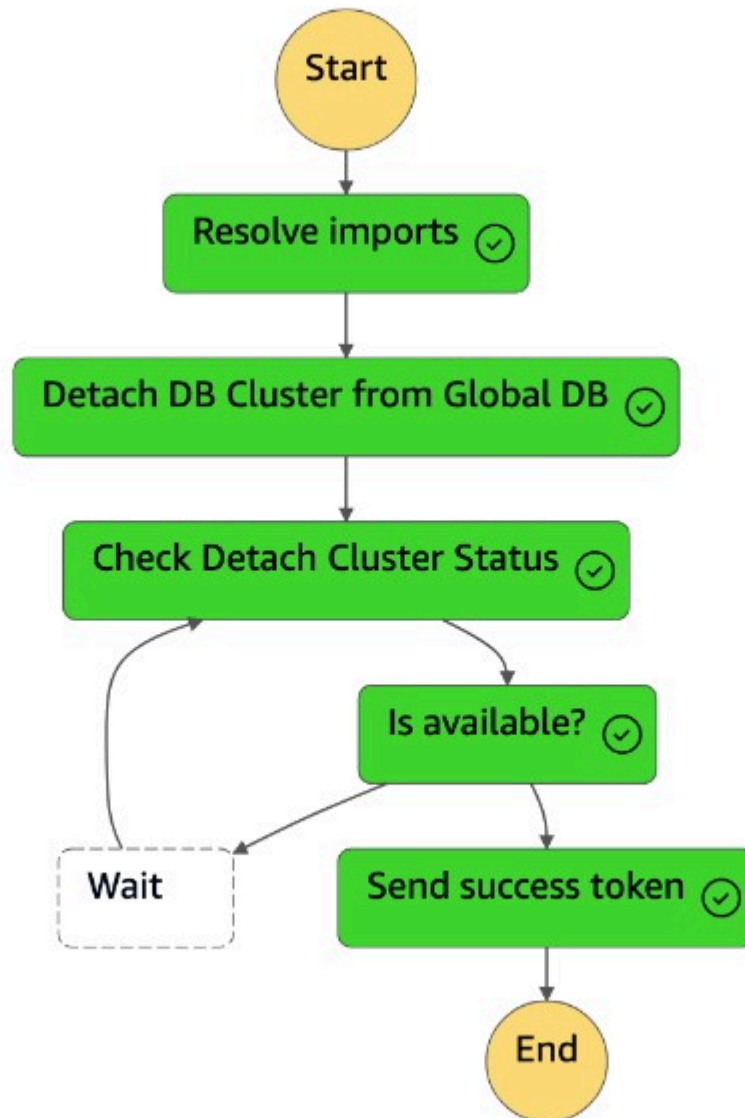
1. Masuk ke konsol tersebut.
2. Ubah Wilayah ke Wilayah DR (us-west-2).
3. Arahkan ke Layanan, dan pilih Step Functions.
4. Arahkan ke mesin `dr-orchestrator-stepfunction-FAILOVER` negara.
5. Pilih Mulai eksekusi, dan masukkan kode JSON berikut di *Input - optional* bagian, gunakan *UnPlannedFailoverAurora* sebagai: *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"
          }
        }
      ]
    }
  ]
}
```

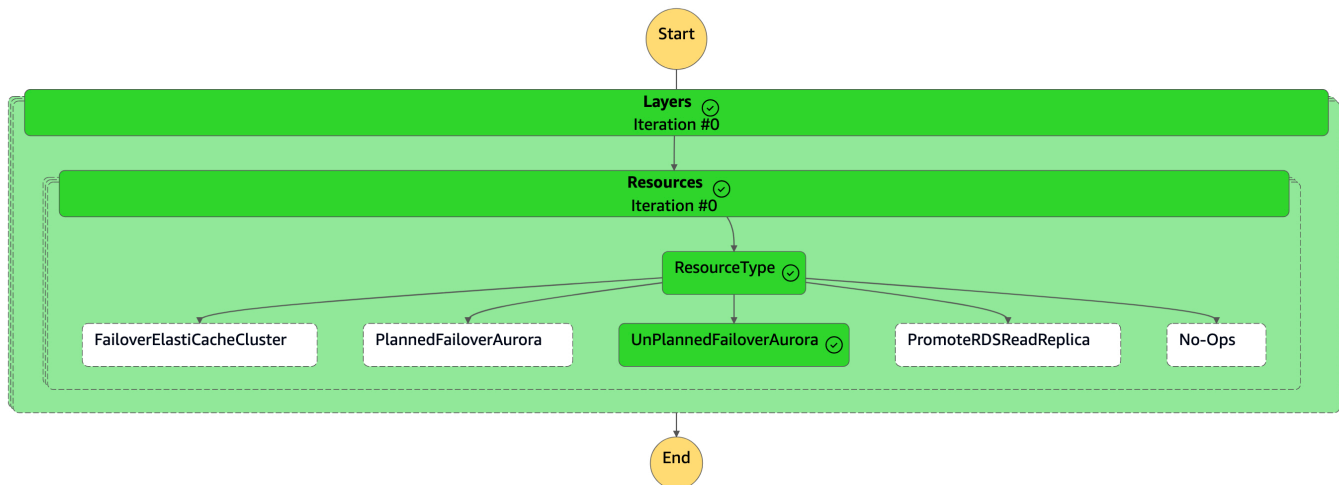
6. Mesin `dr-orchestrator-stepfunction-FAILOVER` status membaca tipe sumber daya sebagai `UnPlannedFailoverAuroraMySQL` dan memanggil tugas `Detach Cluster from Global Database` dari mesin `dr-orchestrator-stepfunction-unplanned-Aurora-failover` status.



7. Detach Cluster from Global Database Tugas melepaskan (menghapus) cluster sekunder dari database global.



8. Cluster sekunder (dbcluster-02) dipromosikan menjadi cluster mandiri, dan dapat melayani beban read/write kerja.
9. Mesin dr-orchestrator-stepfunction-FAILOVER negara selesai.



10.Cluster sekunder (dbcluster-02) terlepas dari database global Aurora, dan menjadi cluster mandiri untuk melayani beban kerja. read/writer

11.Konfigurasi ulang aplikasi Anda untuk mengirim semua operasi penulisan ke cluster Aurora DB mandiri baru ini dengan menggunakan titik akhir cluster barunya.

Kegagalan

Failback mengembalikan database Anda ke lokasi utama asli (atau baru) setelah bencana (atau peristiwa terjadwal) diselesaikan. Ketika pemadaman yang tidak direncanakan telah diselesaikan, Anda mungkin ingin menambahkan Wilayah utama Anda sebelumnya kembali ke database global Aurora. Pertama-tama Anda harus menghapus klaster DB yang ada dari Region primer sebelumnya, membuat klaster DB baru dari Region primer yang baru, dan kemudian menggunakan proses failover terencana terkelola untuk mengalihkan peran klaster baru.

Ini dapat dianggap sebagai kegiatan terencana yang dapat Anda lakukan selama jam-jam sibuk atau pada akhir pekan.

Anda harus secara manual [memodifikasi Amazon Aurora DB Cluster](#) dan menonaktifkan DeletionProtection sebelum Anda menjalankan DR Orchestrator FAILBACK state machine dari Region (us-east-1) primer sebelumnya karena dibuat dengan DeletionProtection

DR Orchestrator Framework menggunakan mesin dr-orchestrator-stepfunction-FAILBACK status untuk mengotomatiskan langkah-langkah untuk menghapus cluster yang ada dan membuat cluster baru di Region primer sebelumnya.

Untuk menonaktifkanDeletionProtection, lakukan hal berikut:

1. Masuk ke konsol tersebut.
2. Ubah Region ke Region primer sebelumnya (us-east-1).
3. Arahkan ke konsol Amazon RDS, pilih nama cluster (dbcluster-01), dan pilih Ubah.
4. Di bawah Perlindungan penghapusan, kosongkan kotak centang Aktifkan perlindungan penghapusan, dan pilih Lanjutkan.
5. Pilih Terapkan segera, lalu pilih Modify cluster.

Mesin DR Orchestrator FAILBACK status dipanggil selama proses failback dari Region () us-east-1 primer sebelumnya.

Untuk melakukan failback, lakukan hal berikut:

1. Masuk ke konsol tersebut.
2. Ubah Region ke Region primer sebelumnya (us-east-1).
3. Arahkan ke Layanan, lalu pilih Step Functions.
4. Arahkan ke mesin DR Orchestrator FAILBACK negara.
5. Pilih Mulai eksekusi, dan masukkan kode JSON berikut di Input - optional bagian:

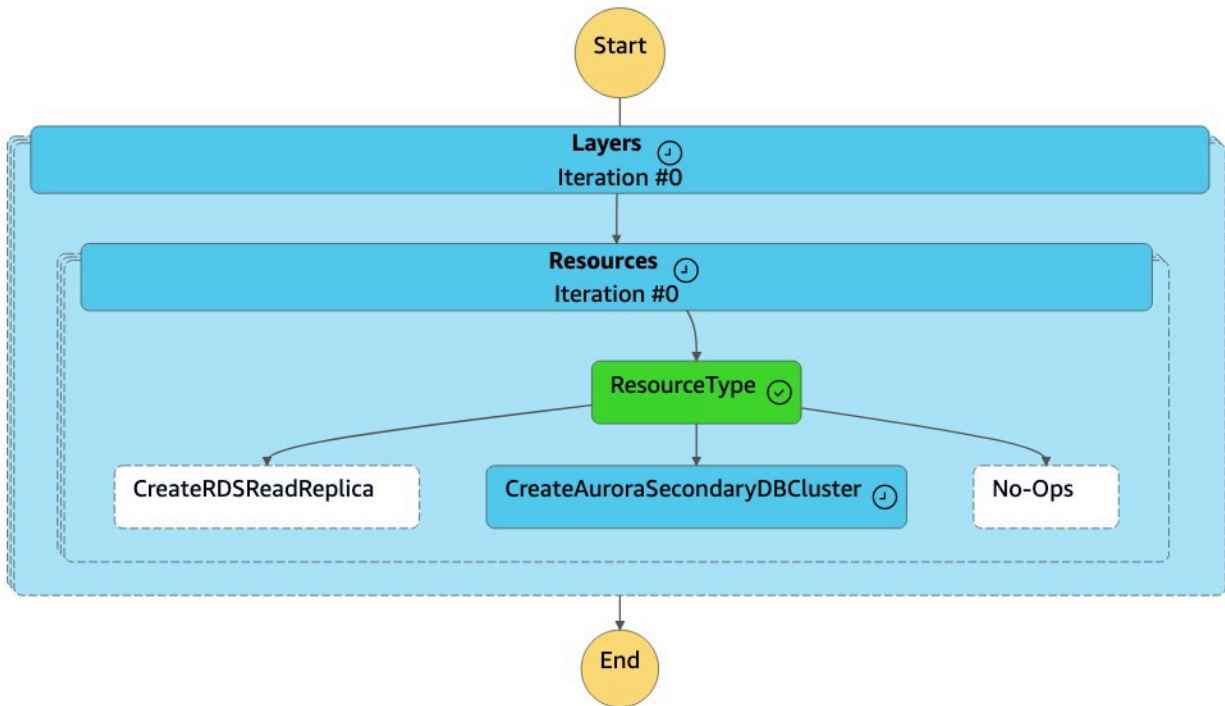
```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "CreateAuroraSecondaryDBCluster",
          "resourceName": "To create secondary Aurora MySQL Global Database Cluster",
          "parameters": {
            "GlobalClusterIdentifier": "!Import 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-identifier",
            "Port": "!Import dr-globaldb-cluster-mysql-port",
            "DBInstanceClass": "!Import dr-globaldb-cluster-mysql-instance-class",
```

```

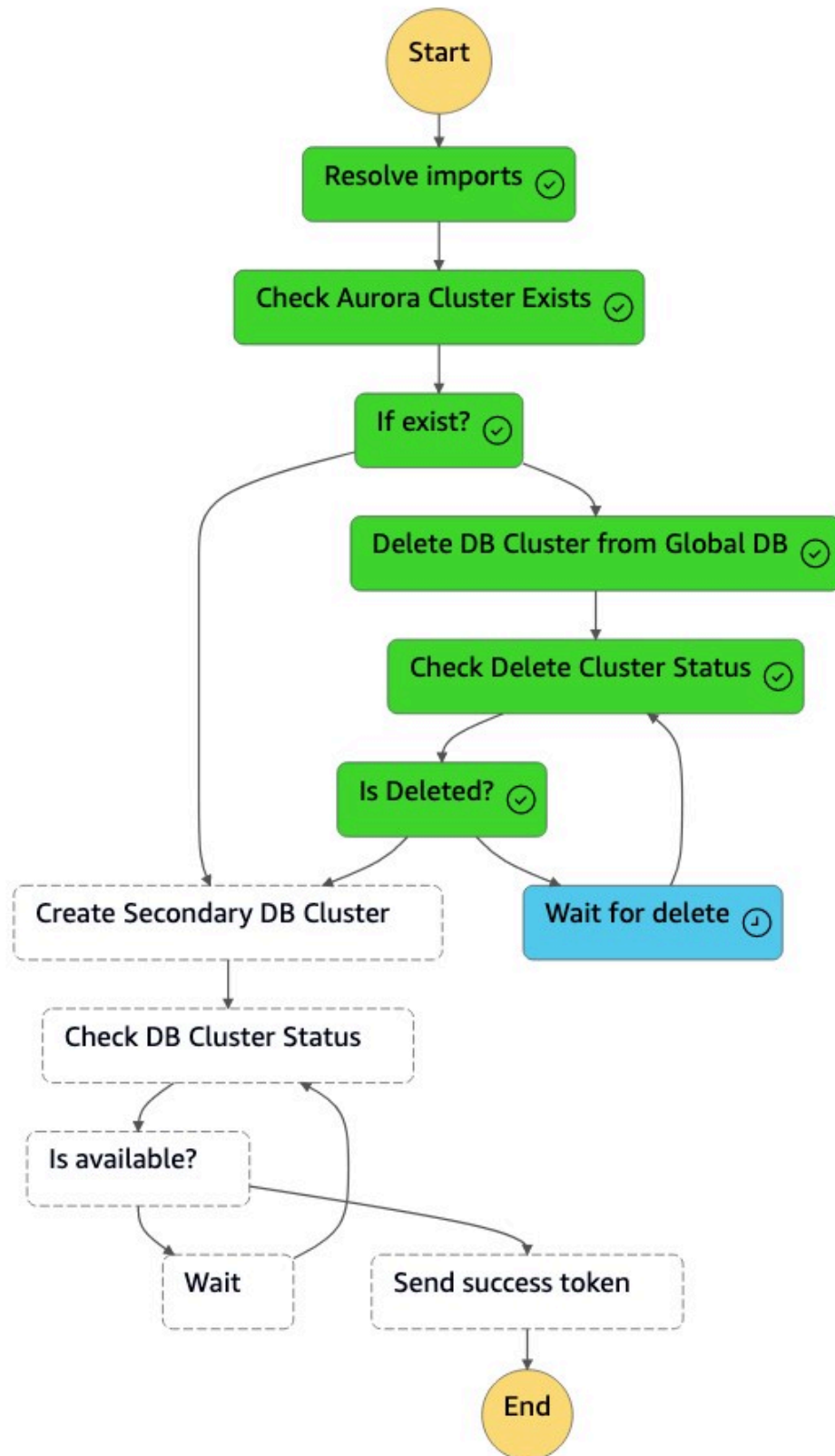
        "DBSubnetGroupName": "!Import dr-globaldb-cluster-mysql-subnet-group-
name",
        "VpcSecurityGroupIds": "!Import dr-globaldb-cluster-mysql-vpc-security-
group-ids",
        "Engine": "!Import dr-globaldb-cluster-mysql-engine",
        "EngineVersion": "!Import dr-globaldb-cluster-mysql-engine-version",
        "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"
    }
}
]
}
]
}

```

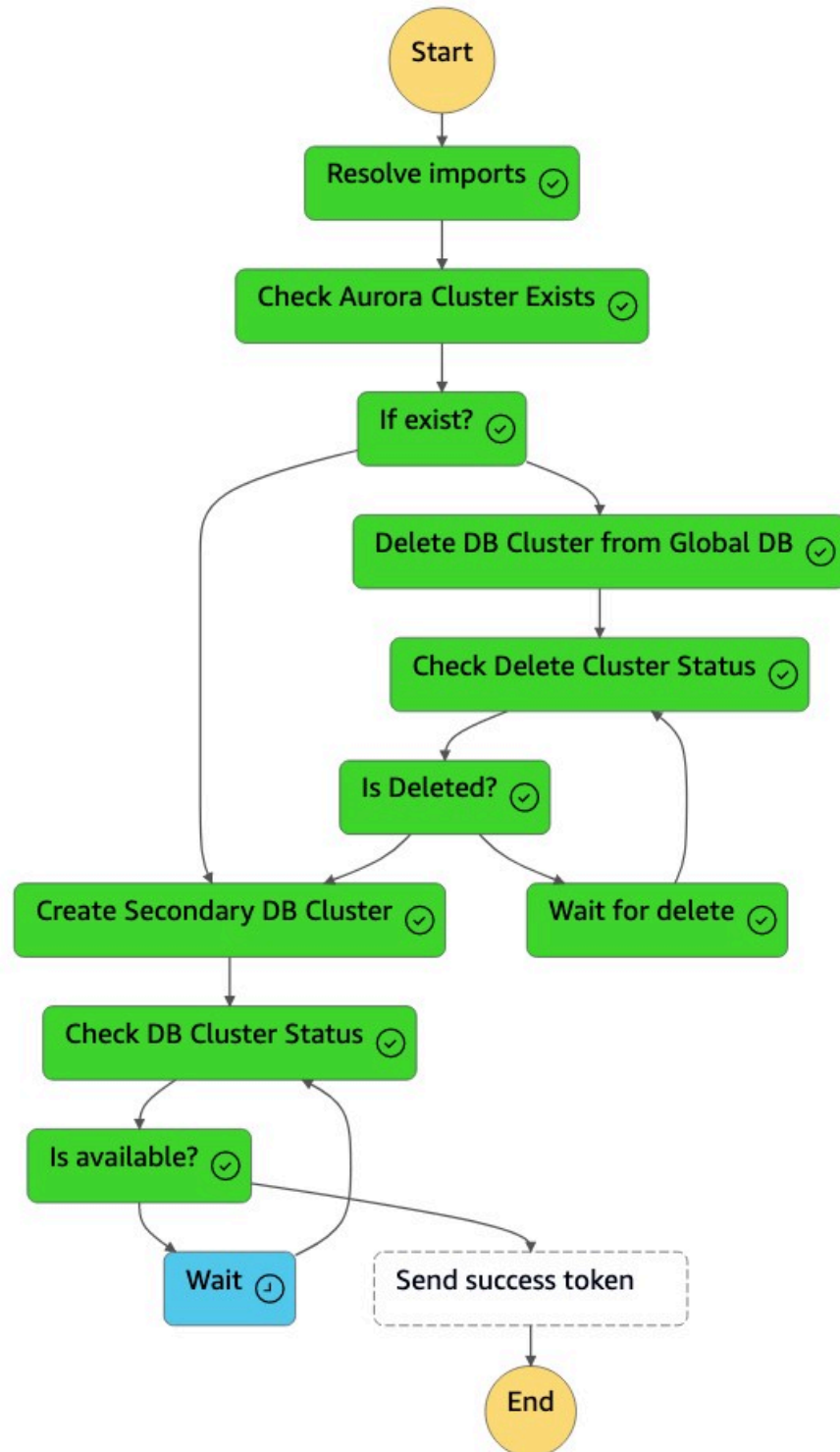
6. Mesin DR Orchestrator FAILBACK status membaca jenis sumber daya sebagai `CreateAuroraSecondaryDBCluster`, dan itu memanggil mesin `dr-orchestrator-stepfunction-create-Aurora-Secondary-cluster` status.



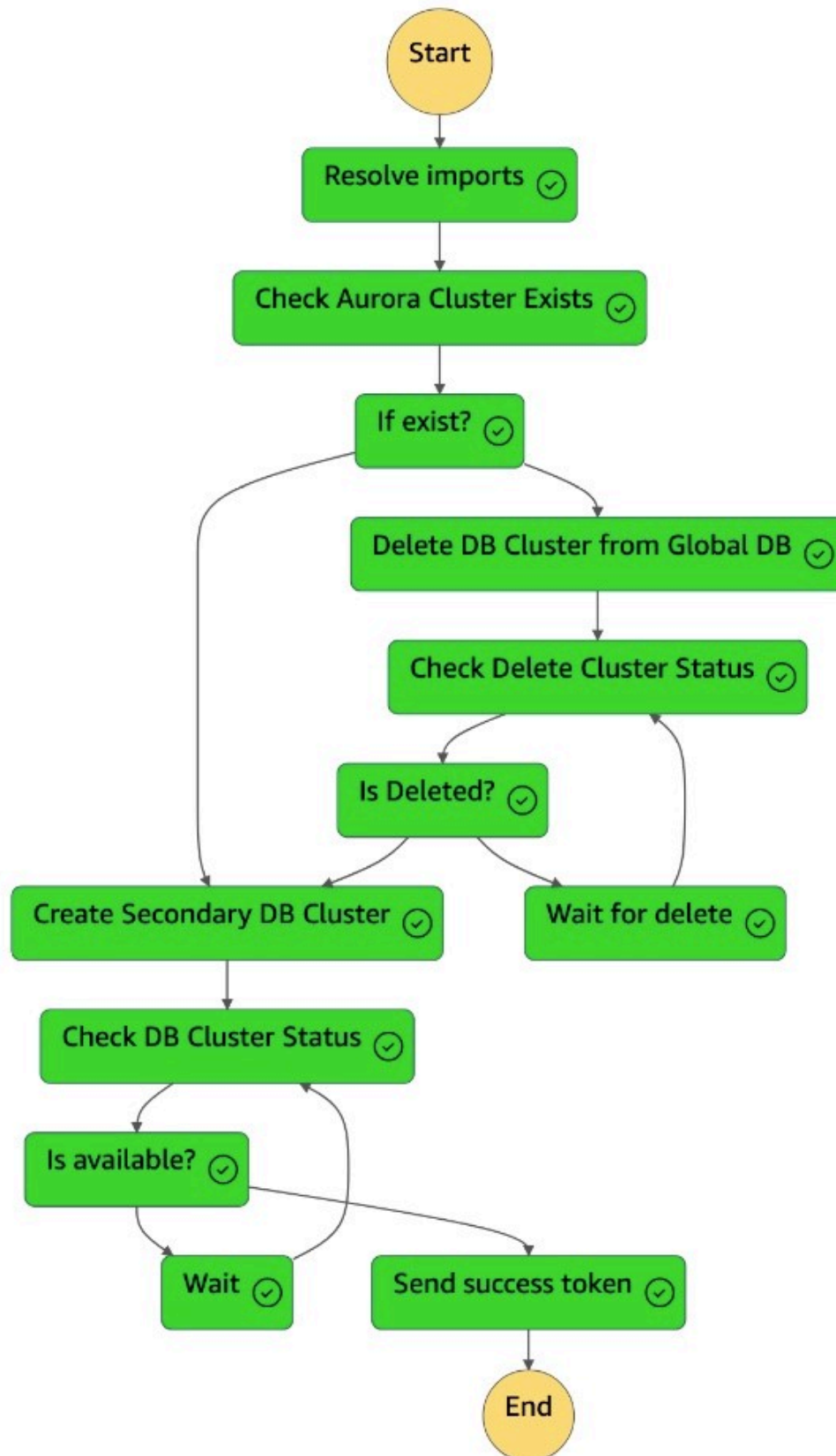
7. Mesin `dr-orchestrator-stepfunction-create-Aurora-Secondary-cluster` state menghapus cluster (`dbcluster-01`) yang ada dari Region (`us-east-1`) primer sebelumnya.



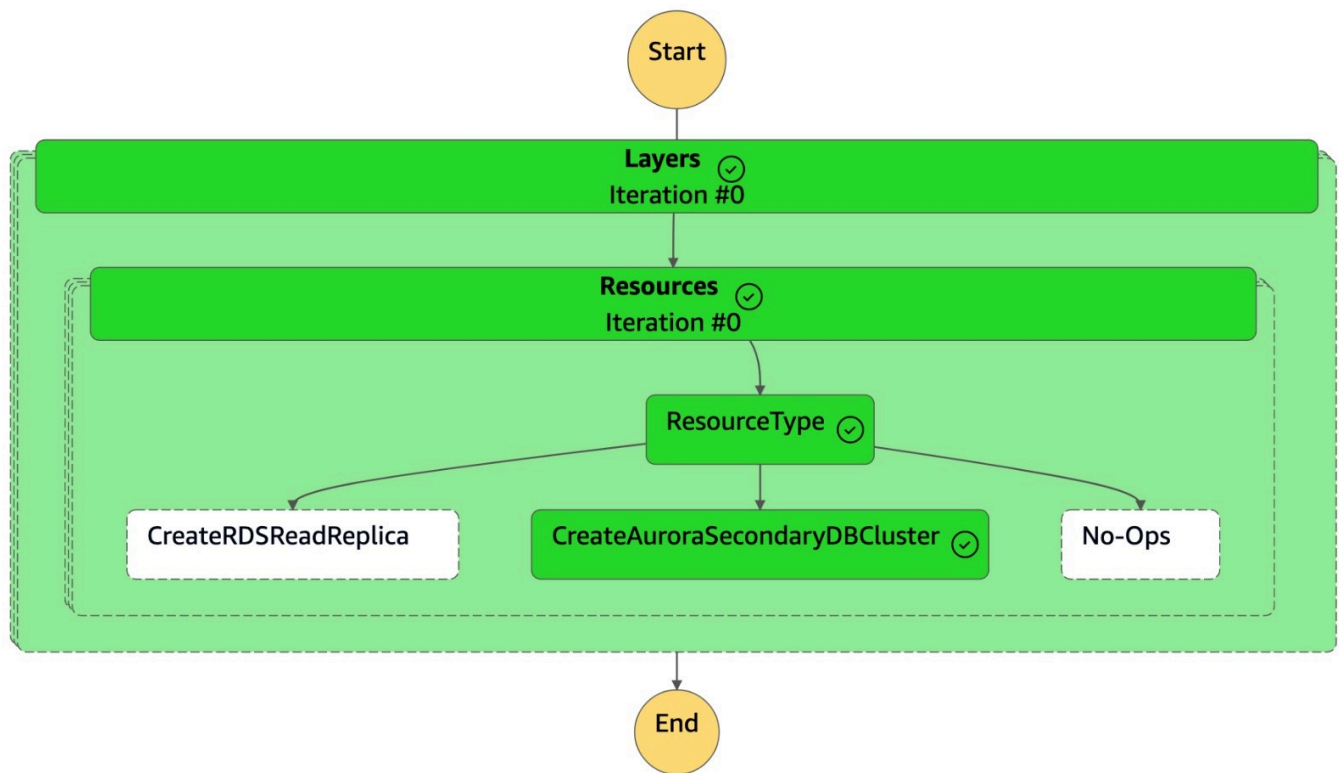
8. Setelah cluster (`dbcluster-01`) dihapus, mesin state membuat cluster (`dbcluster-01`) baru bersama dengan instans DB, dan bergabung dengan database global Aurora sebagai cluster sekunder untuk melayani beban kerja hanya-baca.



9. Setelah cluster sekunder tersedia, mesin `dr-orchestrator-stepfunction-create-Aurora-Secondary-cluster` status selesai, dan mengirimkan token sukses kembali ke mesin `DR Orchestrator Failback` negara.



10 Mesin `dr-orchestrator-stepfunction-FAILBACK` negara selesai.



11 Anda dapat memverifikasi database global Aurora di konsol Amazon RDS.

Jika Anda ingin memindahkan cluster DB utama ke us-east-1 maka Anda dapat mengikuti langkah-langkah yang disebutkan di bagian Switchover.

Terapkan solusinya

Setelah Anda mengatur prasyarat, ikuti instruksi dalam pola [DR](#) Orchestrator Framework. Instruksi disertakan untuk AmazonRDS Aurora, dan Amazon ElastiCache. Untuk menguji solusinya, tinjau prasyarat dan batasan, dan gunakan database contoh. AWS

Prasyarat

- Aktif [Akun AWS](#) untuk menyebarkan solusi ini.
- Izin untuk membuat bucket [Amazon Simple Storage Service \(Amazon S3\)](#) dan sumber daya yang digunakan dalam template. [AWS CloudFormation](#)
- Sebelum Anda menjalankan mesin DR Orchestrator FAILBACK negara:
 - Secara manual [memodifikasi RDS instance](#) atau [memodifikasi Amazon Aurora DB Cluster](#), dan nonaktifkan. DeletionProtection
 - Ambil [snapshot DB](#) sebelum menghapus RDS instans Amazon atau klaster Aurora jika Anda berencana untuk melakukan analisis akar penyebab (). RCA

Batasan

- Ketersediaan dan dukungan fitur bervariasi di seluruh versi tertentu dari setiap mesin database dan di seluruh Wilayah AWS. Untuk informasi selengkapnya tentang fitur dan ketersediaan Wilayah untuk replikasi Lintas wilayah, lihat replika baca [Lintas Wilayah](#).
- Solusi ini telah diuji menggunakan instance database utama dengan satu replika baca. Jika Anda ingin menggunakan lebih dari satu replika baca, uji solusinya secara menyeluruh sebelum menerapkannya di lingkungan produksi.
- Untuk AmazonRDS, solusinya hanya mendukung mesin berikut: [Amazon RDS untuk MySQL](#), [Amazon untuk PostgreSQL](#), dan [Amazon RDS untuk RDS MariaDB](#).

Menyebarkan database contoh pada AWS

Untuk mendemonstrasikan solusi DR Lintas wilayah untuk database AWS, Anda dapat menerapkan [Amazon untuk SQL My, RDS Amazon Aurora My -Compatible Edition, dan ElastiCache Amazon OSS \(Redis\)](#) - Global Datastore. SQL Gunakan instruksi dalam App-Stack README file repositori [aws-cross-region-dr-databases](#) GitHub . Dalam contoh ini, Wilayah primer adalah us-east-1, dan Wilayah sekunder adalah us-west-2.

Tumpukan aplikasi menggunakan bagian AWS CloudFormation [Output](#) untuk mengekspor parameter yang digunakan oleh DR Orchestrator Framework untuk mengotomatiskan failover dan failback database. AWS

Pertanyaan yang Sering Diajukan

Temukan jawaban atas pertanyaan tentang DR Orchestrator Framework untuk failover dan failback.

Apa RPO dan RTO yang dapat saya capai dengan menggunakan pendekatan ini?

Untuk informasi tentang Recovery Time Objective (RTO) dan Recovery Point Objective (RPO), lihat strategi [Disaster recovery strategy untuk database](#). AWS

Apakah wajib menggunakan variabel AWS CloudFormation ekspor?

Tidak, Anda dapat langsung meneruskan nilai database global Amazon Aurora atau instans Amazon RDS DB langsung dalam format JSON (misalnya): - "RDSInstanceIdentifier": "rds-mysql-instance"

Dapatkan saya menggunakan alur kerja DR Orchestrator FAILOVER untuk gagal lebih dari satu database? AWS

Ya, Anda dapat meneruskan lebih dari satu sumber daya dalam file parameter input untuk gagal lebih dari satu AWS database. Contoh kode berikut menunjukkan kegagalan atas replika baca Amazon RDS for MySQL dan datastore global Amazon (Redis OSS) ElastiCache secara paralel:

```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "PromoteRDSReadReplica",
          "resourceName": "Promote RDS MySQL Read Replica",
          "parameters": {
            "RDSInstanceIdentifier": "!Import rds-mysql-instance-identifier",
            "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"
      }
    }
  ]
}
```

Bagaimana saya bisa menghindari InvalidParameterCombination kesalahan saat menjalankan mesin status DR Orchestrator FAILBACK untuk Amazon RDS?

Teks lengkap kesalahan adalah:

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

Untuk menghindari kesalahan, [ubah instance RDS](#) dengan menonaktifkan DeletionProtection sebelum Anda menjalankan mesin status. DR Orchestrator FAILBACK

Langkah selanjutnya

Anda dapat menggunakan DR Orchestrator Framework untuk menambahkan proses persetujuan sebelum memulai aktivitas failover atau failback. Misalnya, saat memulai failover selama acara DR, Anda dapat menambahkan mesin status di atas DR Orchestrator Failover untuk mengirim notifikasi Amazon [Simple Notification Service \(Amazon SNS\)](#) melalui email. Segera setelah persetujuan diberikan, aktivitas failover akan dimulai.

Application failover juga dapat diintegrasikan dengan DR Orchestrator Framework untuk mengalihkan entri DNS dan menunjuk aplikasi ke endpoint baru dari instance database atau cluster.

Anda dapat menggunakan [AWS X-Ray](#) tracing untuk mendapatkan durasi failover untuk menghitung RTO. Anda dapat membangun dasbor pemantauan di atas X-Ray.

Sumber daya

- [Replikasi di seluruh Wilayah AWS menggunakan datastores global](#) (dokumentasi ElastiCache Amazon (Redis OSS))
- [Menggunakan failover dalam database global Amazon Aurora](#) (dokumentasi Aurora)
- [Otomatiskan manajemen endpoint Database Global Amazon Aurora untuk failover yang direncanakan dan tidak direncanakan](#) (posting blog)
- [Pemulihan bencana: 3 skenario failover untuk database global Amazon Aurora Anda dengan Terraform \(Bagian 2\)](#) (posting blog)
- [Strategi pemulihan bencana untuk database pada AWS](#) (Strategi Panduan AWS Preskriptif)
- [Otomatiskan failover dan failback lintas wilayah dengan menggunakan DR Orchestrator Framework](#) (Pola Panduan Preskriptif)AWS

Riwayat dokumen

Tabel berikut menjelaskan perubahan signifikan pada panduan ini. Jika Anda ingin diberi tahu tentang pembaruan masa depan, Anda dapat berlangganan umpan [RSS](#).

Perubahan	Deskripsi	Tanggal
Publikasi awal	—	3 Mei 2024

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.