Introduction to business continuity & disaster recovery

Standard & Business Critical Feature

- Database and share replication are available to all accounts.
- Replication of other account objects, failover/failback, and Client Redirect require Business Critical (or higher). To inquire about upgrading, please contact Snowflake Support.

This topic describes the main use cases for replication and failover across regions and cloud platforms. The Snowflake replication and failover/failback functionality is composed of the following features:

- Replication and Failover/Failback
- Client Redirect

Collectively, these individual features are designed to support a number of different fundamental business continuity scenarios, including:

- **Planned failovers**: For disaster recovery drills to test preparedness, and measure recovery point and time.
- **Unplanned failovers**: In the case of an outage in a region or a cloud platform, promote secondary account objects and databases in another region or cloud platform to serve as read-write primary objects.
- **Migration**: Move your Snowflake account to a different region or cloud platform without disrupting your business. For example, to maintain business continuity during mergers and acquisitions, or facilitate a change in cloud strategy.
- **Multiple readable secondaries**: Account objects and databases can be replicated to multiple accounts in different regions and cloud platforms, mitigating the risk of multiple region or cloud platform outages.

In addition, Snowflake Secure Data Sharing and Database Replication enable sharing data securely across regions and cloud platforms.

Account replication and failover/failback features

Replication and failover/failback

Replication uses two Snowflake objects, replication group and failover group, to replicate a group of objects with point-in-time consistency from a source account to one or more target accounts. A replication group allows customers to specify what to replicate, where to replicate to, and how often. This means specifying which objects to replicate, to which regions or cloud platforms, at customizable scheduled intervals. A failover group enables the replication ***and*** failover of the objects in the group.

Account objects can include warehouses, users, and roles, along with databases and shares (see Replicated objects for the full list of objects that can be included in a replication or failover group). Account objects can be grouped in one or multiple groups.

In the case of failover, account replication enables the failover of your account to a different region or cloud platform. Each replication and failover group has its own replication schedule, allowing you to set the frequency for replication at different intervals for different groups of objects. In the case of failover groups, it also enables failover of groups individually. You can choose to failover all failover groups, or only select failover groups.

Client Redirect

Client Redirect provides a *connection URL* that can be used by Snowflake clients to connect to Snowflake. The connection URL can redirect Snowflake clients to a different Snowflake account as needed.

Business continuity and disaster recovery

In the event of a massive outage (due to a network issue, software bug, etc.) that disrupts the cloud services in a given region, access to Snowflake will be unavailable until the source of the outage is resolved and services are restored. To ensure continued availability and data durability in such a scenario, replicate your critical account objects to another Snowflake account in your organization in a different region.

With asynchronous replication, secondary replicas typically lag behind the primary objects based on the replication schedule you configure. Secondary replica objects are at most 2x the time interval between scheduled refreshes behind the primary objects. For example, if you choose to replicate a primary replication or failover group every 30 minutes, the secondary objects in the group are at most 60 minutes behind the primary objects during an outage.

Depending on your business needs you could choose to:

- Recover reads first to let client applications read data that is 30 minutes stale.
- Recover writes first to reconcile the last 30 minutes of data on the new primary before opening up reads from client applications.
- Recover both reads and writes simultaneously, that is, open up reads from client applications on data that is 30 minutes stale as you reconcile the last 30 minutes of data on the new primary.

Normal status: Region is operational

Account Object Replication: Replicate the failover group(s) with critical account objects to one or more Snowflake accounts in regions different from that of the account that stores the primary (source) failover group(s). Refresh the failover group(s) frequently.

Region outage

To prioritize reads, writes, or both at the same time, follow the steps in one of the following example scenarios.

Reads before writes

When an outage in a region results in full or partial loss of Snowflake availability, this path allows you to redirect Snowflake clients to read-only replicas of account objects in critical failover group(s) first for minimal downtime. Choosing to operate in read-only mode is often desirable during short-term outages.

A longer-term outage combined with the need for the latest data necessitates read-write mode.

1. **Client Redirect:** Point the connection URL used by clients to a Snowflake account that stores your read-only replica (secondary) failover group(s).
2. **Failover (When Needed):** In the event of a longer-term outage, promote the secondary failover group(s) in the Snowflake account where your connection URL is pointing to serve as read-write primary failover group(s).

Writes before reads

When an outage in a region results in full or partial loss of Snowflake availability, this path allows you to recover failover group(s) with critical account objects and continue to process data first. This option is preferable for account administrators who want to fail over their databases and ETL (Extract, Transform, Load) processes first, and then choose to redirect Snowflake clients only when the data is current.

1. **Failover:** Promote the secondary failover group(s) with critical account objects in a different region to serve as the primary failover group(s), which allows writing to the objects included in each failover group(s). Once the databases in the group(s) are writable, you can use your ETL processes to prioritize writes and reconcile data.
   If you use Snowflake data pipeline objects for ETL processes, you can replicate and fail over those objects. For more information, see Stage, pipe, and load history replication.
   Otherwise, configure separate connection URLs for your data ingestion pipeline and one for your clients (for example, a BI dashboard). After failing over the failover group, fail over the connection URL for data ingestion, and write data to the newly promoted primary objects. After data has been reconciled, fail over the connection URL for your clients to enable reads.

2. **Client Redirect (When Needed):** Point the connection URL used by clients to the Snowflake account that stores the new primary failover group(s).

Prioritize both reads and writes

To prioritize both reads and writes at the same time, fail over both the client connection and secondary failover group(s) without waiting for the secondary objects to be up to date. This enables immediate access for clients to potentially stale data while the newly promoted databases can start reingesting data from data pipelines.

1. **Client Redirect:** Point the connection URL used by clients to a Snowflake account that stores your read-only replica (secondary) failover group(s).
2. **Failover:** Promote the secondary failover group(s) with critical account objects in a different region to serve as the primary failover group(s), which enables writing to the objects included in each failover group(s).

Normal status: Outage is resolved

1. **Replication:** Refresh the failover group(s) in the Snowflake account in the region where the outage occurred.
2. **Failback:** Promote the failover group(s) in the Snowflake account where the outage occurred to again serve as the primary failover group(s).
3. **Client Redirect:** Point the connection URL used by clients to the Snowflake account in the region where the outage occurred.

Account migration

Account migration is the one-time process of migrating (or transferring) the Snowflake objects and your stored data to an account in another region or on a different cloud platform. Typical reasons for migrating your account include a closer proximity to your user base or a preference for a different cloud platform based on your corporate strategy or co-location with other cloud assets (e.g. a data lake).

Account object replication supports the replication of account objects such as warehouses, users, and roles, along with databases and shares. See Replicated objects for the complete list of replicated objects.

Note

Account object replication and failover/failback requires Business Critical (or higher). Snowflake can temporarily waive this requirement for a one-time account migration.

Region support for replication and failover/failback

Customers can replicate across all regions within a Region Group. To replicate between regions in different Region groups (for example, from a Snowflake commercial region to a Snowflake government region), please contact Snowflake Support to enable access.

Transitioning from database replication to group-based replication

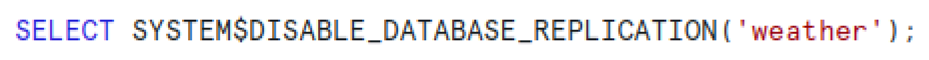
Databases that have been enabled for replication using ALTER DATABASE must have replication disabled ***before*** they can be added to a replication or failover group.

Note
Execute the SQL statements in this section using the ACCOUNTADMIN role.

Step 1. Disable replication for a replication enabled database

Execute the SYSTEM$DISABLE_DATABASE_REPLICATION function to disable replication for a primary database, along with any secondary databases linked to it, in order to add it to a replication or failover group.

Execute the following SQL statement from the source account with the primary database:



```sql
SELECT SYSTEM$DISABLE_DATABASE_REPLICATION('weather');
```

Step 2. Add the database to a primary failover group and create a secondary failover group

Once you have successfully disabled replication for a database, you can add the primary database to a failover group in the source account.

Then create a secondary failover group in the target account. When the secondary failover group is refreshed in the target account, the previously secondary database will automatically be added as a member of the secondary failover group and refreshed with the changes from the primary database.

For more details on creating primary and secondary failover groups, see Workflow.

Note

When you add a previously replicated database to a replication or failover group, Snowflake does ***not*** re-replicate the data that has already been replicated for that database. Only changes since the last refresh are replicated when the group is refreshed.

Workflow

The following SQL statements demonstrate the workflow for enabling account and database object replication and refreshing objects. Each step is discussed in detail below.

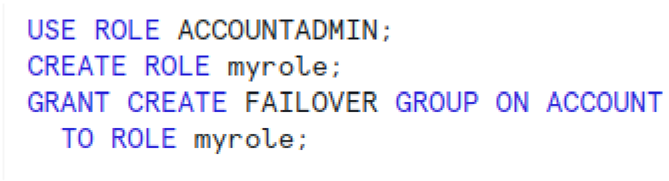
Note
The following examples require replication be enabled for the source and target accounts. For details, see Prerequisite: Enable replication for accounts in the organization.

Examples

Execute the following SQL statements in your preferred Snowflake client to enable account and database object replication and failover, and refresh objects.

Executed on source account

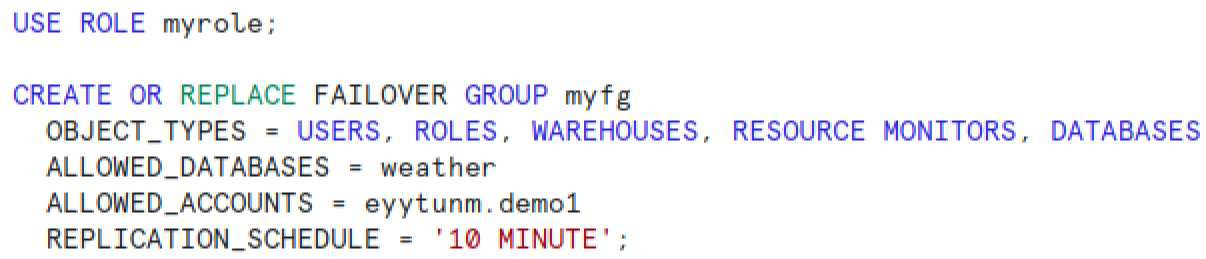
Create a role and grant it the CREATE FAILOVER GROUP privilege. This step is *optional*:



```
USE ROLE ACCOUNTADMIN;
CREATE ROLE myrole;
GRANT CREATE FAILOVER GROUP ON ACCOUNT
  TO ROLE myrole;
```

Create a failover group in the source account and enable replication to specific target accounts.
Note

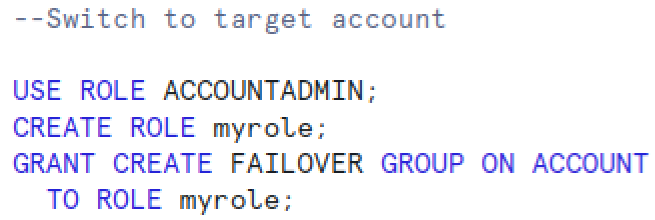
- If you have databases to add to a replication or failover group that have been previously enabled for database replication and failover using ALTER DATABASE, follow the Transitioning from database replication to group-based replication instructions (in this topic) before adding them to a group.
- To add a database to a failover group, the active role must have the MONITOR privilege on the database. For details on database privileges, see Database privileges (in a separate topic).



```
USE ROLE myrole;

CREATE OR REPLACE FAILOVER GROUP myfg
  OBJECT_TYPES = USERS, ROLES, WAREHOUSES, RESOURCE MONITORS, DATABASES
  ALLOWED_DATABASES = weather
  ALLOWED_ACCOUNTS = eyytunm.demo1
  REPLICATION_SCHEDULE = '10 MINUTE';
```

Executed on target account

Create a role in the target account and grant it the CREATE FAILOVER GROUP privilege. This step is *optional*. To do this, you need to switch to the target account.

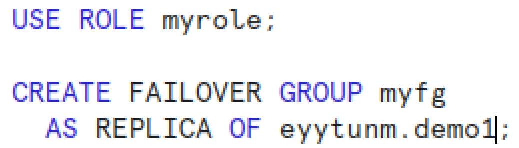


```
--Switch to target account

USE ROLE ACCOUNTADMIN;
CREATE ROLE myrole;
GRANT CREATE FAILOVER GROUP ON ACCOUNT
  TO ROLE myrole;
```

Create a failover group in the target account as a replica of the failover group in the source account.

Note

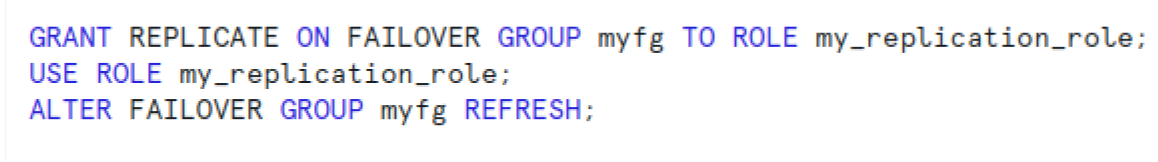
If account objects (e.g. users or roles) exist in the target account that do not exist in the source account, refer to [Initial replication of users and roles](#) before creating a secondary group.



```
USE ROLE myrole;

CREATE FAILOVER GROUP myfg
  AS REPLICA OF eyytunm.demo1;
```

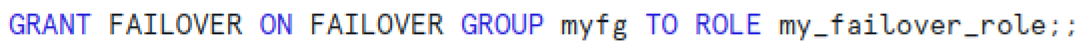
Manually refresh the secondary failover group. This is an *optional* step. If the primary failover group is created with a replication schedule, the initial refresh of the secondary failover group is automatically executed when the secondary failover group is created.

Create a role with the REPLICATE privilege on the failover group. This step is *optional*.
Execute in the target account using a role with the OWNERSHIP privilege on the failover group:
Execute the refresh statement using a role with the REPLICATE privilege:



```
GRANT REPLICATE ON FAILOVER GROUP myfg TO ROLE my_replication_role;
USE ROLE my_replication_role;
ALTER FAILOVER GROUP myfg REFRESH;
```

Create a role with the FAILOVER privilege on the failover group. This step is *optional*.
Execute in the target account using a role with the OWNERSHIP privilege on the failover group:



```
GRANT FAILOVER ON FAILOVER GROUP myfg TO ROLE my_failover_role;;
```

Replicating account objects and databases

The instructions in this section explain how to prepare your accounts for replication, enable the replication of specific objects from the source account to the target account, and synchronize the objects in the target account.

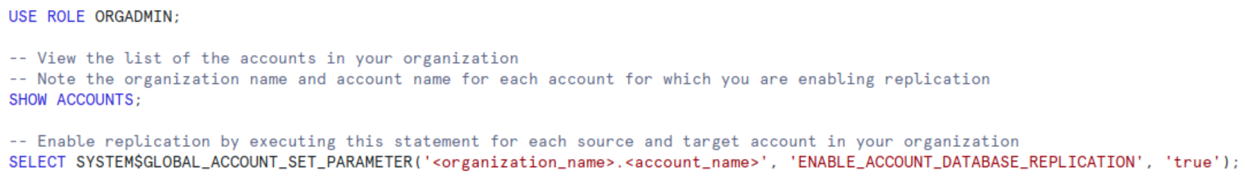
Important
Target accounts do not have Tri-Secret Secure or private connectivity to the Snowflake service, such as AWS PrivateLink, enabled by default. If you require Tri-Secret Secure or private connectivity to the Snowflake service for compliance, security or other purposes, it is your responsibility to configure and enable those features in the target account.

Prerequisite: Enable replication for accounts in the organization

The organization administrator (ORGADMIN role) must enable replication for the source and target accounts.

To enable replication for accounts, a user with the ORGADMIN role uses the SYSTEM$GLOBAL_ACCOUNT_SET_PARAMETER function to set the `ENABLE_ACCOUNT_DATABASE_REPLICATION` parameter to `true`. Note that multiple accounts in an organization can be enabled for replication from the same ORGADMIN account.

Log into an ORGADMIN account to enable replication for each source and target account in your organization.

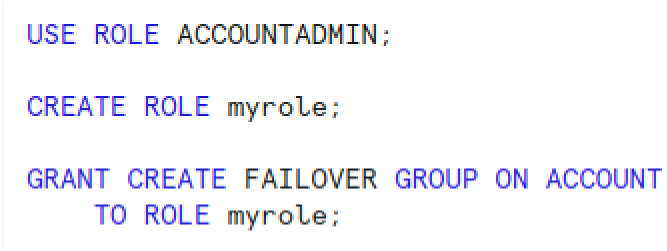


```
USE ROLE ORGADMIN;

-- View the list of the accounts in your organization
-- Note the organization name and account name for each account for which you are enabling replication
SHOW ACCOUNTS;

-- Enable replication by executing this statement for each source and target account in your organization
SELECT SYSTEM$GLOBAL_ACCOUNT_SET_PARAMETER('<organization_name>.<account_name>', 'ENABLE_ACCOUNT_DATABASE_REPLICATION', 'true');
```

Though the SYSTEM$GLOBAL_ACCOUNT_SET_PARAMETER function supports the legacy account locator identifier, it causes unexpected results when an organization has multiple accounts that share the same locator (in different regions).

Step 1: Create a role with the CREATE FAILOVER GROUP privilege in the source account — *Optional*

Create a role and grant it the CREATE FAILOVER GROUP privilege. This step is optional. If you have already created this role, skip to Step 2: Create a primary failover group in a source account.



```
USE ROLE ACCOUNTADMIN;

CREATE ROLE myrole;

GRANT CREATE FAILOVER GROUP ON ACCOUNT
    TO ROLE myrole;
```

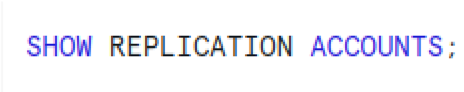
Step 2: Create a primary failover group in a source account

Create a primary failover group and enable the replication and failover of specific objects from the current (source) account to one or more target accounts in the same organization.

View all accounts enabled for replication

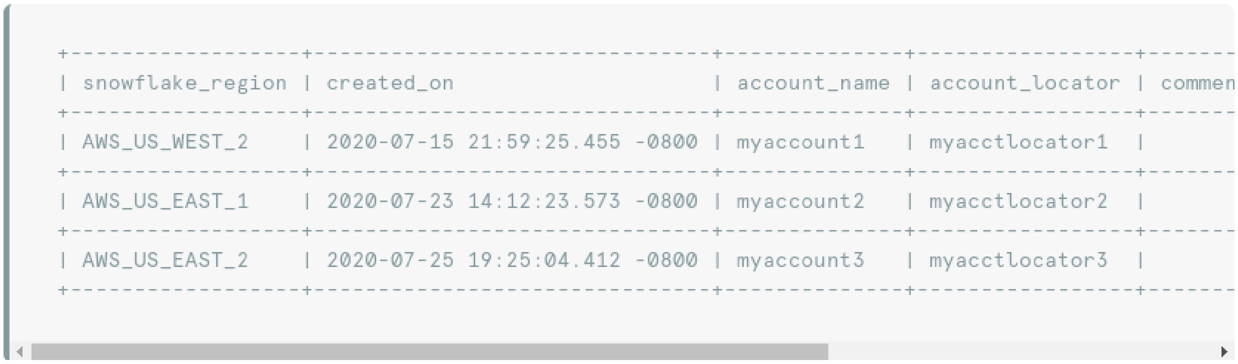
To retrieve the list of accounts in your organization that are enabled for replication, use SHOW REPLICATION ACCOUNTS.

Execute the following SQL statement using the ACCOUNTADMIN role:



```
SHOW REPLICATION ACCOUNTS;
```

Returns:



```
+------------------+-------------------------------+--------------+-----------------+-------
| snowflake_region | created_on                    | account_name | account_locator | commen
+------------------+-------------------------------+--------------+-----------------+-------
| AWS_US_WEST_2    | 2020-07-15 21:59:25.455 -0800 | myaccount1   | myacctlocator1  |
+------------------+-------------------------------+--------------+-----------------+-------
| AWS_US_EAST_1    | 2020-07-23 14:12:23.573 -0800 | myaccount2   | myacctlocator2  |
+------------------+-------------------------------+--------------+-----------------+-------
| AWS_US_EAST_2    | 2020-07-25 19:25:04.412 -0800 | myaccount3   | myacctlocator3  |
+------------------+-------------------------------+--------------+-----------------+-------
```

See the complete list of Region IDs.

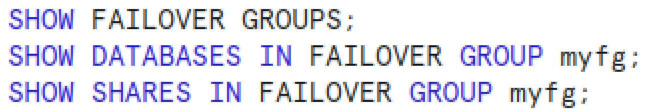
View failover and replication group membership

Account, database, and share objects have constraints on group membership. Before creating new groups or adding objects to existing groups, you can review the list of existing failover groups and the objects in each group.

Note

Only an account administrator (user with the ACCOUNTADMIN role) or the group owner (role with the OWNERSHIP privilege on the group) can execute the SQL statements in this section.

- View all failover groups linked to the current account, and the object types in each group
- View all the databases in failover group `myfg`:
- View all the shares in failover group `myfg`:



```
SHOW FAILOVER GROUPS;
SHOW DATABASES IN FAILOVER GROUP myfg;
SHOW SHARES IN FAILOVER GROUP myfg;
```

Enable replication from a source account to target account

You can create a replication or failover group using Snowsight or SQL.

Note

If you have databases to add to a replication or failover group that have been previously enabled for database replication using ALTER DATABASE, follow the Transitioning from database replication to group-based replication instructions (in this topic) before adding them to a group.

Create a replication or failover group using Snowsight

Preview Feature— Open

Available to accounts in all regions and cloud platforms. Currently this feature is not supported for accounts using private connectivity.

Note

- Only account administrators can create a replication or failover group using Snowsight (refer to Limitations of using Snowsight for replication configuration).
- You must be signed in to the target account as a user with the ACCOUNTADMIN role. If you are not, you will be prompted to sign in.
  Both the source account and the target account must use the same connection type (public internet). Otherwise, signing in to the target account fails.

Complete the following steps to create a new replication or failover group:

1. Sign in to Snowsight and navigate to **Admin** » **Accounts**.
2. Select **Replication**, select **Groups**.
3. Select **+ Add Group**.

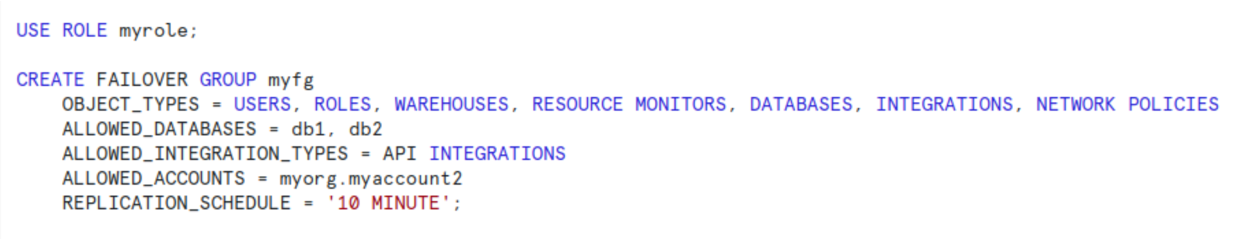
4. Select **Target Account**, then select **Next**.
5. In the **Group Name** box, enter a name for the group that meets the following requirements:
    - Must start with an alphabetic character and cannot contain spaces or special characters unless the identifier string is enclosed in double quotes (e.g. “My object”). Identifiers enclosed in double quotes are also case-sensitive.
      For more information, see Identifier requirements.
    - Must be unique across failover and replication groups in an account.
6. Choose **Select Objects** to add share and account objects to your group.
   Note
   Account objects can only be added to one replication or failover group. If a replication or failover group with any account objects already exists in your account, you cannot select those objects.
7. Choose **Select Databases** to add database objects to your group.
8. Select the **Replication Frequency**.
9. If the account is Business Critical Edition or higher, a failover group is created by default. You can choose to create a replication group instead. To create a replication group, select **Advanced Options**, then unselect **Enable Failover**.
10. Select **Start Replication** to create the replication group.
    If creating the replication group is unsuccessful, refer to Troubleshoot issues with creating and editing replication groups using Snowsight for common errors and how to resolve them.

Create a failover group using SQL

Create a failover group of specified account and database objects in the source account and enable replication and failover to a list of target accounts. See CREATE FAILOVER GROUP for syntax.

For example, enable replication of users, roles, warehouses, resources monitors, and databases `db1` and `db2` from the source account to the `myaccount2` account in the same organization. Set the replication schedule to automatically refresh `myaccount2` every 10 minutes.

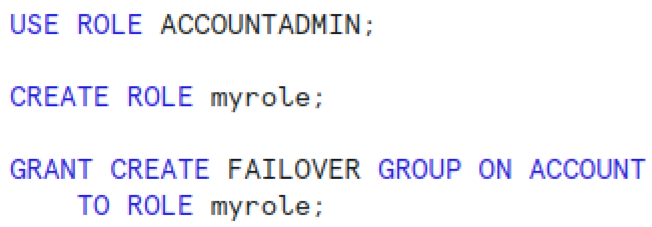
Execute the following statement on the source account:



```
USE ROLE myrole;

CREATE FAILOVER GROUP myfg
    OBJECT_TYPES = USERS, ROLES, WAREHOUSES, RESOURCE MONITORS, DATABASES, INTEGRATIONS, NETWORK POLICIES
    ALLOWED_DATABASES = db1, db2
    ALLOWED_INTEGRATION_TYPES = API INTEGRATIONS
    ALLOWED_ACCOUNTS = myorg.myaccount2
    REPLICATION_SCHEDULE = '10 MINUTE';
```

Executed on target account

Create a role in the target account and grant it the CREATE FAILOVER GROUP privilege. This step is *optional*:

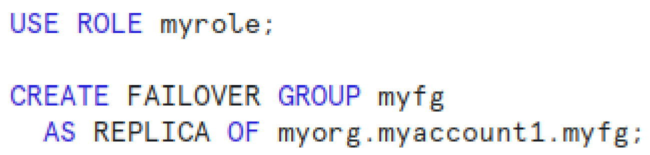


```
USE ROLE ACCOUNTADMIN;

CREATE ROLE myrole;

GRANT CREATE FAILOVER GROUP ON ACCOUNT
    TO ROLE myrole;
```

Create a failover group in the target account as a replica of the failover group in the source account.

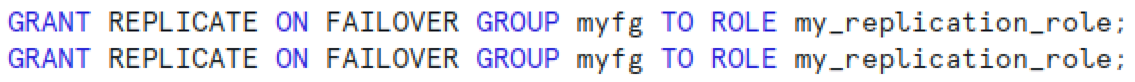
Note

If account objects (e.g. users or roles) exist in the target account that do not exist in the source account, refer to [Initial replication of users and roles]() before creating a secondary group.



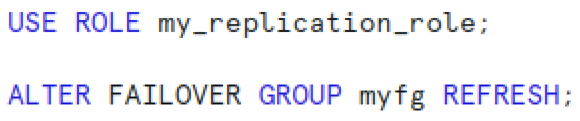
```
USE ROLE myrole;

CREATE FAILOVER GROUP myfg
  AS REPLICA OF myorg.myaccount1.myfg;
```

Manually refresh the secondary failover group. This is an *optional* step. If the primary failover group is created with a replication schedule, the initial refresh of the secondary failover group is automatically executed when the secondary failover group is created.
Create a role with the REPLICATE privilege on the failover group. This step is *optional*.
Execute in the target account using a role with the OWNERSHIP privilege on the failover group:



```
GRANT REPLICATE ON FAILOVER GROUP myfg TO ROLE my_replication_role;
GRANT REPLICATE ON FAILOVER GROUP myfg TO ROLE my_replication_role;
```

Execute the refresh statement using a role with the REPLICATE privilege:



```
USE ROLE my_replication_role;

ALTER FAILOVER GROUP myfg REFRESH;
```

Create a role with the FAILOVER privilege on the failover group. This step is *optional*.
Execute in the target account using a role with the OWNERSHIP privilege on the failover group:

```
GRANT FAILOVER ON FAILOVER GROUP myfg TO ROLE my_failover_role;
```