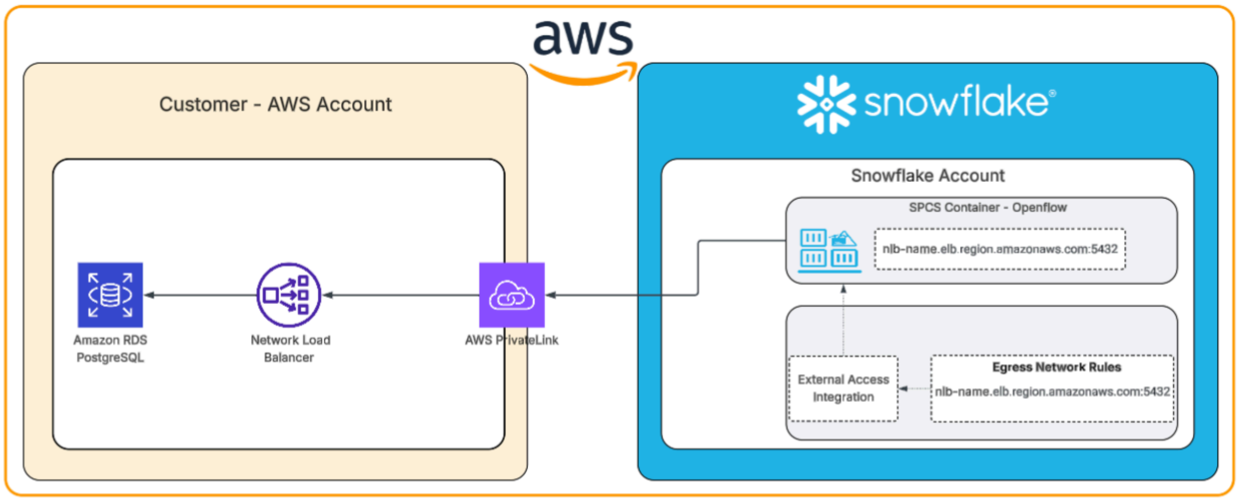
SPCS Private Connectivity to AWS PostgreSQL (RDS)

<u>Note:</u> AWS RDS service is not directly accessible via PrivateLink, private connectivity is only possible through a VPC Endpoint Service and Network Load Balancer in front of your RDS instance.


aws
Customer - AWS Account
Amazon RDS PostgreSQL
Network Load Balancer
AWS PrivateLink
snowflake
Snowflake Account
SPCS Container - Openflow
nlb-name.elb.region.amazonaws.com:5432
External Access Integration
Egress Network Rules
nlb-name.elb.region.amazonaws.com:5432

<u>Requirements:</u>

1. A **VPC Endpoint Service**
2. An **internal Network Load Balancer** attached to the above-mentioned VPCE Service.
3. An RDS PostgreSQL instance

Network Load Balancer configuration:

- Scheme: Internal
- Security: Enforce inbound rules on PrivateLink traffic: **OFF**
  - If you would like to leave this setting ON and apply inbound rules on privatelink traffic, you must allow 10.0.0.0 /8 inbound traffic on TCP port 5432
- Target Group: IP address(es) of the RDS PostgreSQL instance, on TCP port 5432 (or the port the instance accepts connections on)
- Edit load balancer attributes > Availability Zone routing configuration > select **Enable cross-zone load balancing**.
  <u>Note:</u> This has a cost associated with it, but it is mandatory if the NLB and RDS instances are deployed in different and/or multiple Availability Zones within the region.

RDS Postgres Instance configuration:

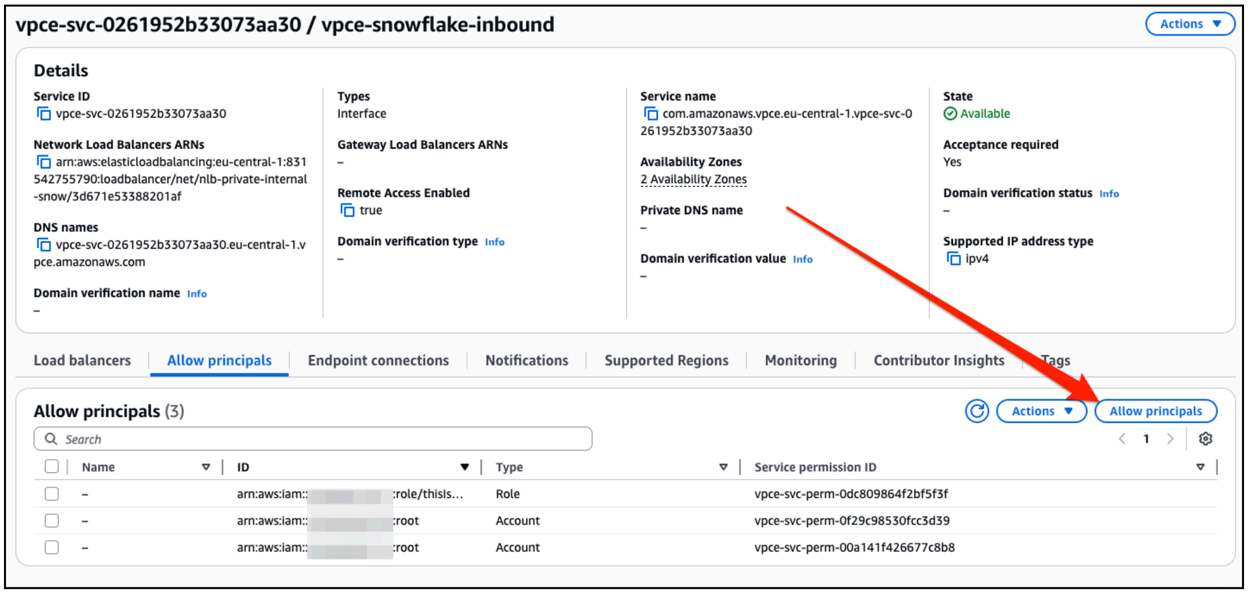
- The Inbound Rule of the security group should include **inbound** traffic from the Network Load Balancer's Security Group on the relevant TCP port.

1. Allow Snowflake to discover your VPCE Service

In your Snowflake account, execute the following statement to retrieve your account principal:

```sql
SQL
SELECT value::string
FROM TABLE(FLATTEN(INPUT => PARSE_JSON(SYSTEM$GET_PRIVATELINK_CONFIG())))
where key = 'privatelink-account-principal';
```

Navigate to your VPC Endpoint Service, under **Allow principals**, add the principal you previously copied in format `arn:aws:iam::000123456789:root`



vpce-svc-0261952b33073aa30 / vpce-snowflake-inbound

Details

Service ID vpce-svc-0261952b33073aa30	**Types** Interface	**Service name** com.amazonaws.vpce.eu-central-1.vpce-svc-0261952b33073aa30	**State** Available
Network Load Balancers ARNs arn:aws:elasticloadbalancing:eu-central-1:831542755790:loadbalancer/net/nlb-private-internal-snow/3d671e53388201af	**Gateway Load Balancers ARNs** –	**Availability Zones** 2 Availability Zones	**Acceptance required** Yes
DNS names vpce-svc-0261952b33073aa30.eu-central-1.vpce.amazonaws.com	**Remote Access Enabled** true	**Private DNS name** –	**Domain verification status** Info –
Domain verification name Info –	**Domain verification type** Info –	**Domain verification value** Info –	**Supported IP address type** ipv4

Allow principals (3)

Name	ID	Type	Service permission ID
–	arn:aws:iam::[redacted]:role/thisIs...	Role	vpce-svc-perm-0dc809864f2bf5f3f
–	arn:aws:iam::[redacted]:root	Account	vpce-svc-perm-0f29c98530fcc3d39
–	arn:aws:iam::[redacted]:root	Account	vpce-svc-perm-00a141f426677c8b8

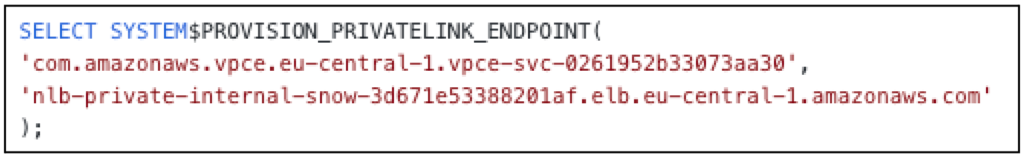
2. Provision a VPC Endpoint from Snowflake to your Service

First, gather:

1. Your **VPC Endpoint Service DNS name**
2. Your **Network Load Balancer DNS name**

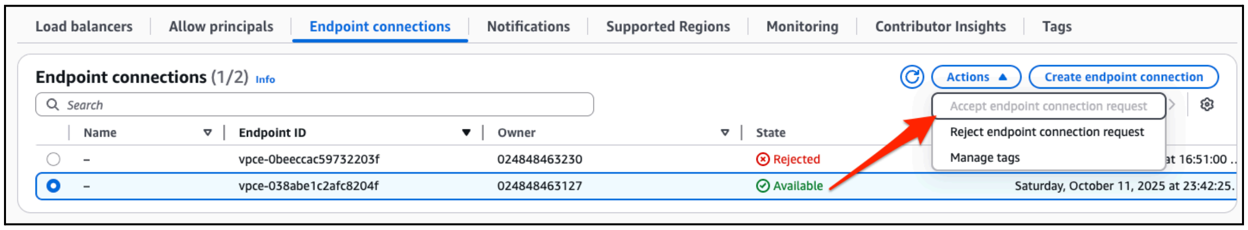
In Snowflake execute the following commands:

```sql
SELECT SYSTEM$PROVISION_PRIVATELINK_ENDPOINT(
'<VPCE-svc-DNS-name>',
'<NLB-DNS-name>'
);
```



```sql
SELECT SYSTEM$PROVISION_PRIVATELINK_ENDPOINT(
'com.amazonaws.vpce.eu-central-1.vpce-svc-0261952b33073aa30',
'nlb-private-internal-snow-3d671e53388201af.elb.eu-central-1.amazonaws.com'
);
```

3. Approve the inbound VPC Endpoint in your VPC Endpoint Service



Endpoint connections (1/2) Info

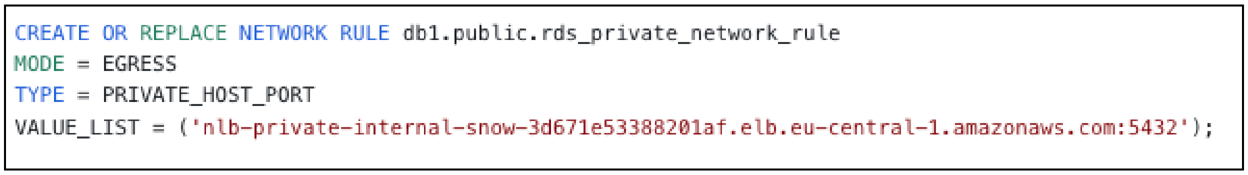
	Name	Endpoint ID	Owner	State	
○	–	vpce-0beeccac59732203f	024848463230	Rejected	at 16:51:00 ..
●	–	vpce-038abe1c2afc8204f	024848463127	Available	Saturday, October 11, 2025 at 23:42:25.

You can verify the status of your VPC Endpoint is **Approved** with the following command:

```sql
SELECT
  parsed_value:provider_resource_id::STRING          AS provider_resource_id,
  parsed_value:snowflake_resource_id::STRING         AS snowflake_resource_id,
  parsed_value:host::STRING                          AS host,
  parsed_value:endpoint_state::STRING                AS endpoint_state,
  parsed_value:subresource::STRING                   AS subresource,
  parsed_value:status::STRING                        AS status
FROM TABLE(
  FLATTEN(
    INPUT => PARSE_JSON(SYSTEM$GET_PRIVATELINK_ENDPOINTS_INFO())
  )
),
LATERAL (
  SELECT PARSE_JSON(value) AS parsed_value
)
WHERE HOST ilike '%elb%amazonaws.com';
```

4. Create an Egress Network Rule

```sql
CREATE OR REPLACE NETWORK RULE rds_private_network_rule
MODE = EGRESS
TYPE = PRIVATE_HOST_PORT
VALUE_LIST = ('<nlb_dns_name>:<database_port>');
```



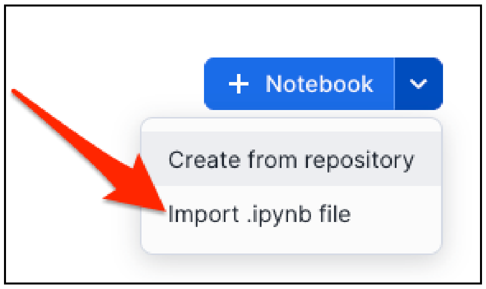
```
CREATE OR REPLACE NETWORK RULE db1.public.rds_private_network_rule
MODE = EGRESS
TYPE = PRIVATE_HOST_PORT
VALUE_LIST = ('nlb-private-internal-snow-3d671e53388201af.elb.eu-central-1.amazonaws.com:5432');
```

5. Create an External Access Integration

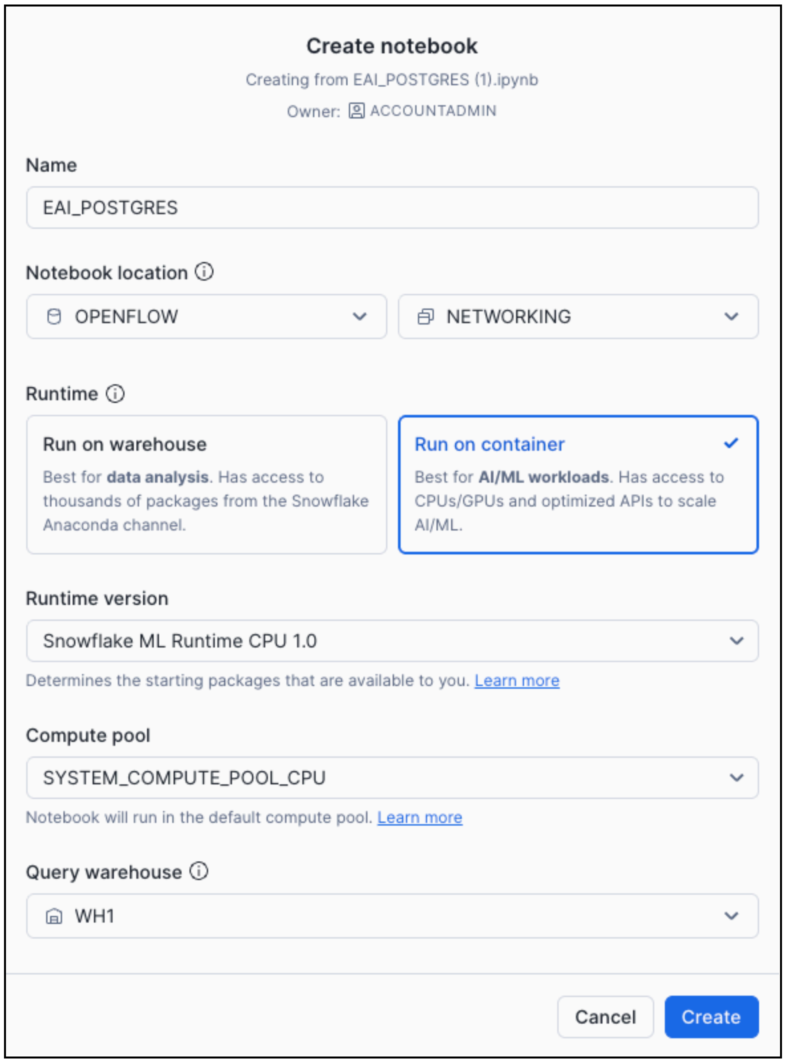
```sql
CREATE OR REPLACE EXTERNAL ACCESS INTEGRATION rds_external_access_integration
ALLOWED_NETWORK_RULES = (rds_private_network_rule)
ENABLED = TRUE;
```

6. Download and install the Notebook to validate the configuration

Download the following Snowflake Notebook file from our Snowflake-Labs GitHub repository: https://github.com/sfc-gh-plewandowski/sfguide-getting-started-with-openflow-spcs/blob/main/notebooks/EAI_POSTGRES/EAI_POSTGRES.ipynb



Import the file as a Notebook in Snowsight > Projects > Notebooks > **Import .ipynb file**



Create notebook

Creating from EAI_POSTGRES (1).ipynb

Owner: ACCOUNTADMIN

Name

EAI_POSTGRES

Notebook location

OPENFLOW

NETWORKING

Runtime

Run on warehouse

Best for **data analysis**. Has access to thousands of packages from the Snowflake Anaconda channel.

Run on container ✓

Best for **AI/ML workloads**. Has access to CPUs/GPUs and optimized APIs to scale AI/ML.

Runtime version

Snowflake ML Runtime CPU 1.0

Determines the starting packages that are available to you. Learn more

Compute pool

SYSTEM_COMPUTE_POOL_CPU

Notebook will run in the default compute pool. Learn more

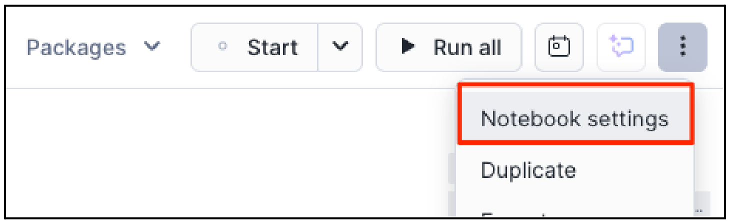
Query warehouse

WH1

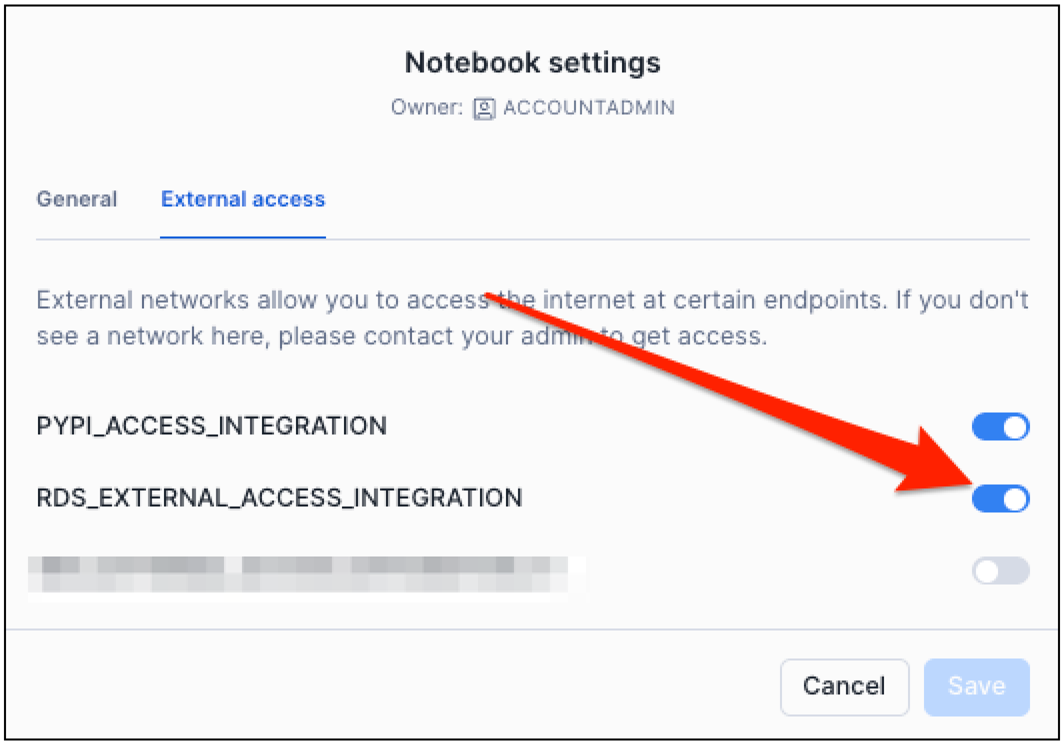
Cancel Create

7. Enable the PostgreSQL EAI in the Notebook

Navigate to the **Notebook settings** at the top right corner of your screen



Under **External access**, enable the PostgreSQL External Access Integration and Save


Notebook settings

Owner: ACCOUNTADMIN

External networks allow you to access the internet at certain endpoints. If you don't see a network here, please contact your admin to get access.

PYPI_ACCESS_INTEGRATION

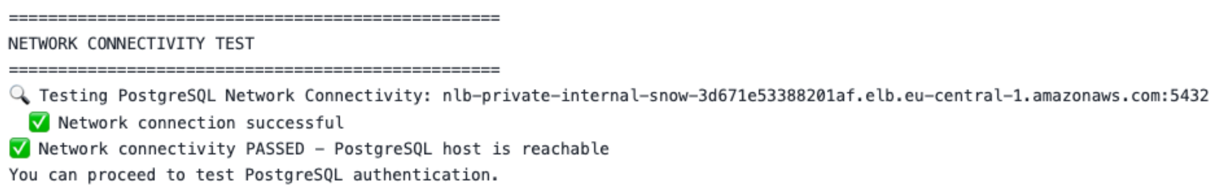
RDS_EXTERNAL_ACCESS_INTEGRATION

[redacted]

8. Configure the variables and execute the Notebook

```SQL
POSTGRES_HOST =
"nlb-private-internal-snow-3d671e53388201af.elb.eu-central-1.amazonaws.com"
POSTGRES_PORT = 5432
POSTGRES_DATABASE = "postgres"
POSTGRES_USER = "sqladminuser"
POSTGRES_PASSWORD = "xxxxxxx"
```

Successful tests will confirm successful private connectivity and authentication to your RDS instance.



```
==================================================
NETWORK CONNECTIVITY TEST
==================================================
🔍 Testing PostgreSQL Network Connectivity: nlb-private-internal-snow-3d671e53388201af.elb.eu-central-1.amazonaws.com:5432
  ✅ Network connection successful
✅ Network connectivity PASSED - PostgreSQL host is reachable
You can proceed to test PostgreSQL authentication.
```

```
==================================================
POSTGRESQL AUTHENTICATION TEST
==================================================
🔍 Testing PostgreSQL Authentication and Basic Query
  📦 Using pg8000 library
  ✅ Authentication successful
  ✅ Database query successful
  📊 PostgreSQL Version: PostgreSQL 17.4 on x86_64-pc-linux-gnu, compiled b...
  📋 Found 1 tables in public schema
     - table1
✅ PostgreSQL authentication PASSED
Your SPCS environment can successfully connect to PostgreSQL!
You can proceed with PostgreSQL integration.
```