**Secure data SHARE in Snowflake**

A complete, best-practice-based hands-on lab on how to securely share data with a third-party vendor using Snowflake’s secure data sharing capabilities. This will include:

* Creating a sample database, schema, and table
* Implementing **masking policies** and **row-level security**
* Creating a **secure view** for sharing
* Creating a **share** and adding objects to it
* Easy **revocation** mechanism
* No data replication or export – just **read-only access**

**Step 1: Create Database, Schema, and Dummy Table**

-- Create a database and schema for demonstration

CREATE OR REPLACE DATABASE VENDOR\_SHARE\_DB;

CREATE OR REPLACE SCHEMA VENDOR\_SHARE\_DB.PUBLIC;

-- Create dummy customer orders table

CREATE OR REPLACE TABLE VENDOR\_SHARE\_DB.PUBLIC.CUSTOMER\_ORDERS (

ORDER\_ID INT,

CUSTOMER\_NAME STRING,

CUSTOMER\_EMAIL STRING,

CUSTOMER\_REGION STRING,

PRODUCT STRING,

AMOUNT NUMBER(10,2),

ORDER\_DATE DATE,

IS\_INTERNAL BOOLEAN

);

-- Insert demo records

INSERT INTO VENDOR\_SHARE\_DB.PUBLIC.CUSTOMER\_ORDERS VALUES

(1, 'Alice', 'alice@example.com', 'US', 'Laptop', 1200.00, '2024-12-01', FALSE),

(2, 'Bob', 'bob@example.com', 'EU', 'Tablet', 450.00, '2024-12-02', FALSE),

(3, 'Charlie', 'charlie@example.com', 'US', 'Phone', 700.00, '2024-12-03', TRUE),

(4, 'David', 'david@example.com', 'ASIA', 'Camera', 300.00, '2024-12-04', FALSE),

(5, 'Eve', 'eve@example.com', 'US', 'Monitor', 250.00, '2024-12-05', FALSE),

(6, 'Frank', 'frank@example.com', 'EU', 'Keyboard', 120.00, '2024-12-06', FALSE),

(7, 'Grace', 'grace@example.com', 'US', 'Mouse', 80.00, '2024-12-07', TRUE),

(8, 'Heidi', 'heidi@example.com', 'ASIA', 'Speaker', 150.00, '2024-12-08', FALSE),

(9, 'Ivan', 'ivan@example.com', 'US', 'Tablet', 420.00, '2024-12-09', FALSE),

(10, 'Judy', 'judy@example.com', 'EU', 'Laptop', 1100.00, '2024-12-10', FALSE),

(11, 'Karl', 'karl@example.com', 'US', 'Phone', 800.00, '2024-12-11', FALSE),

(12, 'Laura', 'laura@example.com', 'EU', 'Monitor', 240.00, '2024-12-12', FALSE),

(13, 'Mallory', 'mallory@example.com', 'ASIA', 'Camera', 290.00, '2024-12-13', TRUE),

(14, 'Niaj', 'niaj@example.com', 'US', 'Laptop', 1250.00, '2024-12-14', FALSE),

(15, 'Olivia', 'olivia@example.com', 'EU', 'Phone', 750.00, '2024-12-15', FALSE),

(16, 'Peggy', 'peggy@example.com', 'ASIA', 'Tablet', 400.00, '2024-12-16', FALSE),

(17, 'Quentin', 'quentin@example.com', 'US', 'Keyboard', 130.00, '2024-12-17', FALSE),

(18, 'Rupert', 'rupert@example.com', 'EU', 'Mouse', 70.00, '2024-12-18', FALSE),

(19, 'Sybil', 'sybil@example.com', 'US', 'Speaker', 160.00, '2024-12-19', TRUE),

(20, 'Trent', 'trent@example.com', 'ASIA', 'Monitor', 255.00, '2024-12-20', FALSE);

**Step 2: Create a Masking Policy (for sensitive columns)**

-- Create a masking policy for email

CREATE OR REPLACE MASKING POLICY MASK\_EMAIL\_POLICY AS (val STRING)

RETURNS STRING ->

CASE

WHEN CURRENT\_ROLE() IN ('VENDOR\_READER') THEN '\*\*\*@masked.com'

ELSE val

END;

-- Apply the masking policy to CUSTOMER\_EMAIL

ALTER TABLE VENDOR\_SHARE\_DB.PUBLIC.CUSTOMER\_ORDERS

MODIFY COLUMN CUSTOMER\_EMAIL

SET MASKING POLICY MASK\_EMAIL\_POLICY;

**Step 3: Create a Row Access Policy**

-- Row-level policy to restrict internal records

CREATE OR REPLACE ROW ACCESS POLICY EXCLUDE\_INTERNAL\_ORDERS

AS (IS\_INTERNAL BOOLEAN)

RETURNS BOOLEAN ->

CASE

WHEN CURRENT\_ROLE() = 'VENDOR\_READER' THEN IS\_INTERNAL = FALSE

ELSE TRUE

END;

-- Apply it to the table

ALTER TABLE VENDOR\_SHARE\_DB.PUBLIC.CUSTOMER\_ORDERS

ADD ROW ACCESS POLICY EXCLUDE\_INTERNAL\_ORDERS ON (IS\_INTERNAL);

**Step 4: Create a Secure View (best practice for sharing)**

-- Create a secure view to control exposure

CREATE OR REPLACE SECURE VIEW VENDOR\_SHARE\_DB.PUBLIC.SAFE\_VENDOR\_VIEW AS

SELECT

ORDER\_ID,

CUSTOMER\_NAME,

CUSTOMER\_EMAIL,

CUSTOMER\_REGION,

PRODUCT,

AMOUNT,

ORDER\_DATE

FROM VENDOR\_SHARE\_DB.PUBLIC.CUSTOMER\_ORDERS;

**Step 5: Create Role & Grant for the Vendor (Optional in Real Share)**

-- Simulate a role that the third-party will use

CREATE OR REPLACE ROLE VENDOR\_READER;

GRANT USAGE ON DATABASE VENDOR\_SHARE\_DB TO ROLE VENDOR\_READER;

GRANT USAGE ON SCHEMA VENDOR\_SHARE\_DB.PUBLIC TO ROLE VENDOR\_READER;

GRANT SELECT ON VIEW VENDOR\_SHARE\_DB.PUBLIC.SAFE\_VENDOR\_VIEW TO ROLE VENDOR\_READER;

**Step 6: Create a Share & Add the View**

-- Create a secure share

CREATE OR REPLACE SHARE VENDOR\_SECURE\_SHARE;

-- Add the view to the share

ALTER SHARE VENDOR\_SECURE\_SHARE ADD

VIEW VENDOR\_SHARE\_DB.PUBLIC.SAFE\_VENDOR\_VIEW;

-- Grant usage on the database and schema in the share

GRANT USAGE ON DATABASE VENDOR\_SHARE\_DB TO SHARE VENDOR\_SECURE\_SHARE;

GRANT USAGE ON SCHEMA VENDOR\_SHARE\_DB.PUBLIC TO SHARE VENDOR\_SECURE\_SHARE;

## **Step 7: Revocation is Easy**

You can **remove the view** or **revoke objects**:

-- Remove the view from the share (immediate revocation)

ALTER SHARE VENDOR\_SECURE\_SHARE REMOVE VIEW VENDOR\_SHARE\_DB.PUBLIC.SAFE\_VENDOR\_VIEW;

You can also **drop the share entirely**:

DROP SHARE VENDOR\_SECURE\_SHARE;

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Covered?** | **Notes** |
| **Secure sharing** | ✅ | Via secure view and Snowflake's native share |
| **Governance** | ✅ | Masking policy and row-level policy |
| **Revocation** | ✅ | Remove view or drop share |
| **No replication/export** | ✅ | Only read-only secure view exposed |
| **Naive-user friendly demo** | ✅ | Simple data + direct steps |

Happy learning

Regards

Saransh Jain