**Dynamic Data Masking**

Dynamic Data Masking in Snowflake allows us to secure sensitive columns by applying masking policies. Depending on the user’s role, Snowflake either shows the real data or a masked value.for example In the Hospitality Management System (HMS) project in HCL, we were dealing with customer data from hotels, including **credit card numbers, phone numbers, and personal details**. To comply with **PCI-DSS** and **GDPR** regulations, we implemented **Dynamic Data Masking** in Snowflake.

For example, credit card numbers were masked such that **only authorized finance or audit users** could see the full number, while other roles like support or reporting teams saw only a **masked version (XXXX-XXXX-XXXX-1234)**.

This ensured **data privacy and security**, while still allowing teams to run analytics without exposing sensitive information.”

create role dev\_team;

create role QA\_TEAM;

grant role QA\_TEAM to user NAGENDRA;

grant usage on database practice\_db to role DEV\_TEAM;

grant USAGE on SCHEMA practice\_SCHEMA to role DEV\_TEAM;

grant USAGE on warehouse compute\_wh\_m to role DEV\_TEAM;

grant select on table practice\_db.practice\_SCHEMA.customer\_dm to role DEV\_TEAM;

grant usage on database practice\_db to role qa\_TEAM;

grant USAGE on SCHEMA practice\_SCHEMA to role qa\_TEAM;

grant USAGE on warehouse compute\_wh\_m to role qa\_TEAM;

grant select on table practice\_db.practice\_SCHEMA.customer\_dm to role qa\_TEAM;

show users;

use role dev\_team;

create table customer\_dm as

select \* from SNOWFLAKE\_SAMPLE\_DATA.TPCH\_SF1.CUSTOMER;

select \* from snowflake.customer\_dm;

create or replace schema my\_policies;

create or replace masking policy my\_policies.customer\_dm\_phone

as (val string) returns string->

case

when current\_role() = 'DEV\_TEAM' then val

else '##-###-###-'||substr(val,12,4)

end;

create or replace masking policy my\_policies.customer\_dm\_bal

as (val number) returns number->

case

when current\_role() in ('ACCOUNTADMIN','DEV\_TEAM') then val

else 0

end;

alter table practice\_db.snowflake.customer\_dm

modify column c\_acctbal set masking policy practice\_db.my\_policies.customer\_dm\_bal;

alter table practice\_db.snowflake.customer\_dm

modify column c\_phone set masking policy practice\_db.my\_policies.customer\_dm\_phone;

use role dev\_team;

select \* from practice\_db.snowflake.customer\_dm;

use role qa\_team;

show masking policies in schema my\_policies;

desc masking policy practice\_db.my\_policies.customer\_dm\_phone;

// To see wherever you applied the policy

SELECT \* FROM table(information\_schema.policy\_references(policy\_name=>'practice\_db.my\_policies.customer\_dm\_phone'));

drop masking policy practice\_db.snowflake.customer\_dm\_phone;

alter masking policy practice\_db.my\_policies.customer\_dm\_phone set body->

case when current\_role() = 'ACCOUNTADMIN' then val

else '#########'

end;

select \* from snowflake.customer\_dm;

alter table snowflake.customer\_dm modify column c\_phone

unset masking policy;

alter table snowflake.customer\_dm modify column C\_ACCTBAL

unset masking policy;