Login as SECURITYADMIN (or ACCOUNTADMIN) and create new roles to demonstrate masking policy:

**======================== Creates and Grants ======================**

use role accountadmin;

create role ecommerce\_admin;

create role ecommerce\_users;

grant usage on warehouse ecommerce\_wh to role ecommerce\_admin;

grant usage on warehouse ecommerce\_wh to role ecommerce\_users;

grant ownership on database ecommerce\_database to role ecommerce\_admin;

grant ownership on schema ecommerce\_database.public to role ecommerce\_admin;

grant usage on database ecommerce\_database to role ecommerce\_users;

grant usage on schema ecommerce\_database.public to role ecommerce\_users;

grant select on table ecommerce\_database.public.orders to role ecommerce\_admin;

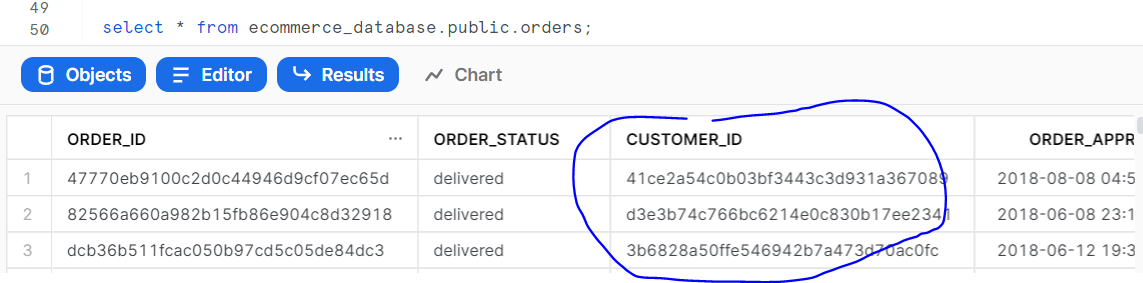
grant select on table ecommerce\_database.public.orders to role ecommerce\_users;

**============= Checking data before the creation / application of masking policy ===========**

use role ecommerce\_admin;

select \* from ecommerce\_database.public.orders;

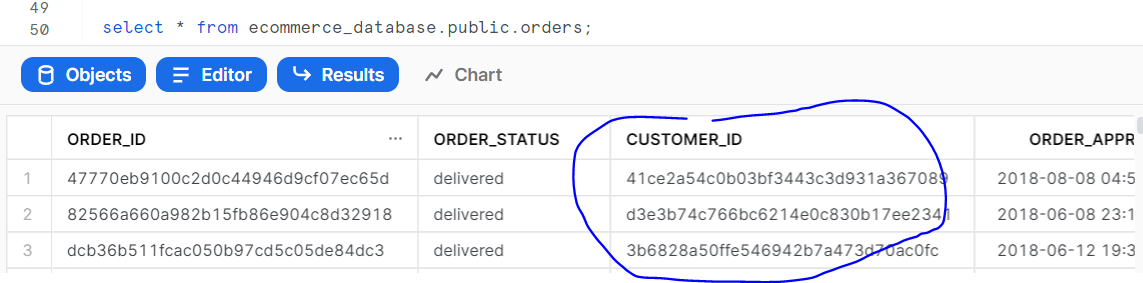
You will be able to see the customer\_id.



use role ecommerce\_users;

select \* from ecommerce\_database.public.orders;

You will be able to see the customer\_id.



**============= Create masking policy and apply on table / column ===============**

create or replace masking policy customer\_id\_policy

as (val varchar) returns varchar ->

case

when current\_role() in ('ECOMMERCE\_ADMIN', 'ACCOUNTADMIN') then val

else '#######'

end;

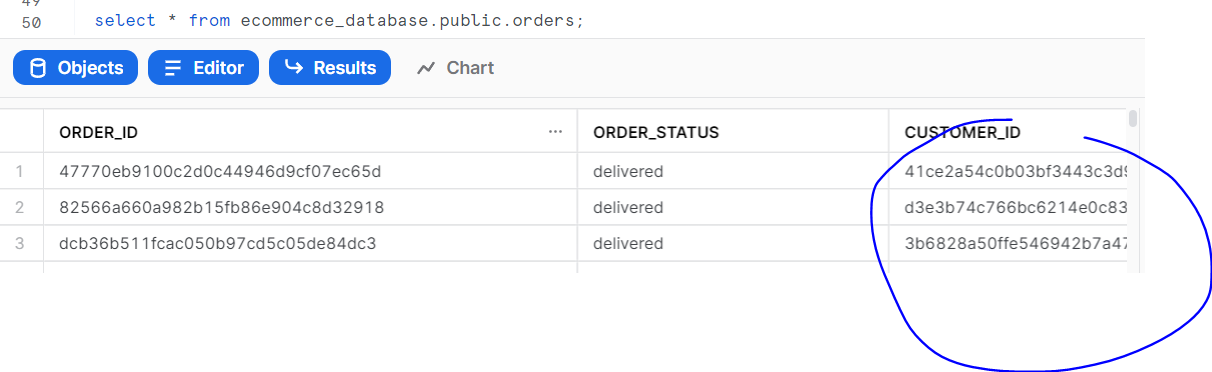
ALTER TABLE IF EXISTS ecommerce\_database.public.orders MODIFY COLUMN customer\_id

SET MASKING POLICY customer\_id\_policy;

**============= Checking data after the creation / application of masking policy ===========**

use role ecommerce\_admin;

select \* from ecommerce\_database.public.orders;

use role ecommerce\_users;

select \* from ecommerce\_database.public.orders;

