**Time Travel**

Snowflake’s Time Travel lets us access historical data for recovery, auditing, or testing. It supports querying older versions of data, restoring dropped objects, and cloning data as of a specific time(up to 1-90 days depending on edition default is 1 day) or statement. In my projects, I used it mainly to **recover accidentally deleted data** and to **create test environments by cloning historical tables without extra storage cost**.”

**Fail Safe**: After 90 days time travel period also we can get the data back using fail safe needs to contact the snowflake customer support for the continous data protection life cycle. No user operations allowed. Its 7 days on we can get the data back.. It takes several days

show parameters like '%retention%' in account; -- default is 1 and 0, Value can be 1 - 90 days

alter account set DATA\_RETENTION\_TIME\_IN\_DAYS = 1 ;

alter account set MIN\_DATA\_RETENTION\_TIME\_IN\_DAYS = 2 ;

-- effective retention time = max (data\_retention\_time\_in\_days, min\_data\_retention\_time\_in\_days)

alter database PRACTICE\_DB set DATA\_RETENTION\_TIME\_IN\_DAYS = 3 ; --3

show databases;

show tables;

alter database set MIN\_DATA\_RETENTION\_TIME\_IN\_DAYS = 1 ; -- error show parameters like '%retention%' in database; -- only 1

alter schema PRACTICE\_DB.SNOWFLAKE set DATA\_RETENTION\_TIME\_IN\_DAYS = 6 ;

alter schema PRACTICE\_DB.SNOWFLAKE set MIN\_DATA\_RETENTION\_TIME\_IN\_DAYS = 1 ; --error

show parameters like '%retention%' in schema; -- only 1

-- create tables for time travel

create or replace table income\_band as select \* from

SNOWFLAKE\_SAMPLE\_DATA.TPCDS\_SF100TCL.INCOME\_BAND ;

-- Create table client

create or replace table client

( id NUMBER(38,0), first\_name VARCHAR(16), last\_name VARCHAR(50), sex VARCHAR(1), ethinicity VARCHAR(30), ssn VARCHAR(15), street\_address VARCHAR(90),status VARCHAR(10)

);

delete from client ;

insert into client values (111111, 'James', 'Schwartz', 'M', 'American','342-76-9087','5676 Washington Street','ACTIVE') ;

insert into client values (222222, 'Jessica', 'Escobar', 'F', 'Hispanic','456-93-5629','3234 WateringCan

Drive','INACTIVE') ;

insert into client values (333333, 'Ben', 'Hardy', 'M', 'American','876-98-3245','6578 Historic Circle','INACTIVE') ;

insert into client values (444444, 'Anjali', 'Singh', 'F', 'Indian American','435-87-6532','8978 Autumn Day Drive','ACTIVE') ;

insert into client values (555555, 'Dean', 'Tracy', 'M', 'African','767-34-7656','2343 India Street','ACTIVE') ;

-- select data s

select \* from client ; -- 5 rows

-- Time travel selects

show parameters like '%time%' ;

alter session set timezone = 'Asia/Kolkata';

-- Snowflake does not support EST, EDT. It supports ->America/Los\_Angeles, Europe/paris, Asia/Tokyo

select current\_timestamp() ; -- 2025-08-23 10:10:22.950 +0530

-- <>

-- Do some deletes and updates on both the tables save query ID

-- delete

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS WHERE CA\_LOCATION\_TYPE = 'single family';

delete from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS WHERE CA\_LOCATION\_TYPE = 'single family'; -- 01be8dfd-3201-e34d-000f-43660003a106

select count(\*) from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS;

-- Query ID

-- <> -- update

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS WHERE CA\_LOCATION\_TYPE = 'single family' ; -- accidental update, shouyld have been only for 222222 ;

-- Query ID -- <>

-- see the data changes

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS;

update PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS set ca\_country = 'INDIA' where CA\_LOCATION\_TYPE = 'apartment';

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS where ca\_country = 'INDIA';

select current\_timestamp(); -- 2025-08-23 10:21:38.839 +0530 -- 8/23/2025, 10:21:38 AM

-- How can we query using time travel BEFORE | AT

-- using before via timestamp

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS AT (timestamp => '2025-08-23 10:21:19.839'::timestamp\_ltz);

create or replace table customer\_address\_temp as select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS AT (timestamp => '2025-08-23 10:21:19.839'::timestamp\_ltz);

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS\_TEMP;

-- 01be8dfd-3201-e34d-000f-43660003a106

-- shows 5 rows with correct status select \* from income\_band before (timestamp => ''::timestamp\_ltz); -- shows all 20 rows

--using before via query ID

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS AT (statement => '01be8dfd-3201-e34d-000f-43660003a106');

alter table PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS SET DATA\_RETENTION\_TIME\_IN\_DAYS = 90;

show tables;

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS before (statement => '01be8dfd-3201-e34d-000f-43660003a106');

-- get the seconds to be used for offset

select datediff(second, '2025-08-23 10:21:19.839'::timestamp\_ltz, current\_timestamp()) as seconds; -- The offset is in seconds using at -- 566

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS before(offset => -60\*20);

-- get right seconds

select datediff(second, ''::timestamp\_ltz, current\_timestamp()) as seconds;

-- show how the offset can be used

select \* from income\_band at(offset => -);

-- dropped the table

drop table PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS;

select \* from PRACTICE\_DB.SNOWFLAKE.CUSTOMER\_ADDRESS;

-- UNDROP

undrop table CUSTOMER\_ADDRESS;