**STORAGE MONITOR QUERY DEVELOPMENT**

**Note : Flow of execution :**

Step 1 : **Snowflake Prerequisite** Queries.

Step 2 : **Create the Stored Procedures and tasks** Queries.

## **Snowflake Prerequisite Query**

Please refer to the Snowflake **Prerequisite Query** and execute those queries if you have not already executed them before you move on to the next steps.

## **List of Tables Used**

| **Table Name** | **Dashboards** | **Latency** |
| --- | --- | --- |
| Org\_Storage | Storage Usage | Upto 120 minutes |
| Account\_Storage | Storage Usage | Upto 120 minutes |
| DB\_Storage | Storage Usage | Upto 180 minutes |
| Table\_and\_Schema\_Storage | Storage Usage | Upto 90 minutes |

## **Create the Stored Procedure and Tasks**

**Note :** Run the below queries to make sure that you use the appropriate role, warehouse, database and schema before you create the tables and run the queries

**QUERY :**

USE role MONITOR\_ADMIN;

USE warehouse MONITOR\_WH;

USE database MONITOR\_DB;

CREATE schema STORAGE\_MONITOR\_SCHEMA;

USE schema STORAGE\_MONITOR\_SCHEMA;

// CREATE Stored Procedure

CREATE OR REPLACE PROCEDURE Storage\_Monitor\_SP()

RETURNS VARCHAR

LANGUAGE JAVASCRIPT

AS

$$

var orgCommand1="CREATE TRANSIENT TABLE if not exists ORG\_STORAGE as SELECT USAGE\_DATE,SERVICE\_TYPE, AVERAGE\_BYTES/(1024\*1024\*1024\*1024) as AVERAGE\_TB, ACCOUNT\_NAME, REGION FROM SNOWFLAKE.ORGANIZATION\_USAGE.PREVIEW\_STORAGE\_DAILY\_HISTORY;"

var orgCommand2="Truncate table ORG\_STORAGE;"

var orgCommand3="Insert into ORG\_STORAGE SELECT USAGE\_DATE,SERVICE\_TYPE, AVERAGE\_BYTES/(1024\*1024\*1024\*1024) as AVERAGE\_TB,ACCOUNT\_NAME,REGION FROM SNOWFLAKE.ORGANIZATION\_USAGE.PREVIEW\_STORAGE\_DAILY\_HISTORY;"

var accCommand1="CREATE TRANSIENT TABLE if not exists ACCOUNT\_STORAGE as select USAGE\_DATE,STORAGE\_BYTES/(1024\*1024\*1024\*1024) AS STORAGE\_TB ,STAGE\_BYTES/(1024\*1024\*1024\*1024) AS STAGE\_TB,FAILSAFE\_BYTES/(1024\*1024\*1024\*1024) AS FAILSAFE\_TB from SNOWFLAKE.ACCOUNT\_USAGE.STORAGE\_USAGE;"

var accCommand2="Truncate table ACCOUNT\_STORAGE;"

var accCommand3="Insert into ACCOUNT\_STORAGE select USAGE\_DATE,STORAGE\_BYTES/(1024\*1024\*1024\*1024) AS STORAGE\_TB ,STAGE\_BYTES/(1024\*1024\*1024\*1024) AS STAGE\_TB,FAILSAFE\_BYTES/(1024\*1024\*1024\*1024) AS FAILSAFE\_TB from SNOWFLAKE.ACCOUNT\_USAGE.STORAGE\_USAGE;"

var dbCommand1="CREATE TRANSIENT TABLE if not exists DB\_STORAGE as select DATABASE\_NAME,USAGE\_DATE,AVERAGE\_DATABASE\_BYTES/(1024\*1024\*1024\*1024) AS AVERAGE\_DATABASE\_TB ,AVERAGE\_FAILSAFE\_BYTES/(1024\*1024\*1024\*1024) AS AVERAGE\_FAILSAFE\_TB from SNOWFLAKE.ACCOUNT\_USAGE.DATABASE\_STORAGE\_USAGE\_HISTORY;"

var dbCommand2="Truncate table DB\_STORAGE;"

var dbCommand3="Insert into DB\_STORAGE select DATABASE\_NAME,USAGE\_DATE,AVERAGE\_DATABASE\_BYTES/(1024\*1024\*1024\*1024) AS AVERAGE\_DATABASE\_TB ,AVERAGE\_FAILSAFE\_BYTES/(1024\*1024\*1024\*1024) AS AVERAGE\_FAILSAFE\_TB from SNOWFLAKE.ACCOUNT\_USAGE.DATABASE\_STORAGE\_USAGE\_HISTORY;"

var tabCommand1="CREATE TRANSIENT TABLE IF NOT EXISTS TABLE\_AND\_SCHEMA\_STORAGE as select TABLE\_CREATED,TABLE\_NAME, TABLE\_SCHEMA, SCHEMA\_CREATED AS SCHEMA\_CREATION\_DATE, TABLE\_CATALOG AS TABLE\_DATABASE, DELETED AS IS\_TABLE\_DELETED, IS\_TRANSIENT, ACTIVE\_BYTES/(1024\*1024\*1024) AS ACTIVE\_GB,TIME\_TRAVEL\_BYTES/(1024\*1024\*1024) AS TIME\_TRAVEL\_GB,FAILSAFE\_BYTES/(1024\*1024\*1024) AS FAILSAFE\_GB,RETAINED\_FOR\_CLONE\_BYTES/(1024\*1024\*1024) AS RETAINED\_FOR\_CLONE\_GB from SNOWFLAKE.ACCOUNT\_USAGE.TABLE\_STORAGE\_METRICS;"

var tabCommand2="Truncate table TABLE\_AND\_SCHEMA\_STORAGE;"

var tabCommand3="Insert into TABLE\_AND\_SCHEMA\_STORAGE select TABLE\_CREATED,TABLE\_NAME, TABLE\_SCHEMA, SCHEMA\_CREATED AS SCHEMA\_CREATION\_DATE, TABLE\_CATALOG AS TABLE\_DATABASE, DELETED AS IS\_TABLE\_DELETED, IS\_TRANSIENT, ACTIVE\_BYTES/(1024\*1024\*1024) AS ACTIVE\_GB,TIME\_TRAVEL\_BYTES/(1024\*1024\*1024) AS TIME\_TRAVEL\_GB,FAILSAFE\_BYTES/(1024\*1024\*1024) AS FAILSAFE\_GB,RETAINED\_FOR\_CLONE\_BYTES/(1024\*1024\*1024) AS RETAINED\_FOR\_CLONE\_GB from SNOWFLAKE.ACCOUNT\_USAGE.TABLE\_STORAGE\_METRICS;"

var orgCmd\_dict1 = {sqlText: orgCommand1};

var orgCmd\_dict2 = {sqlText: orgCommand2};

var orgCmd\_dict3 = {sqlText: orgCommand3};

var accCmd\_dict1 = {sqlText: accCommand1};

var accCmd\_dict2 = {sqlText: accCommand2};

var accCmd\_dict3 = {sqlText: accCommand3};

var dbCmd\_dict1 = {sqlText: dbCommand1};

var dbCmd\_dict2 = {sqlText: dbCommand2};

var dbCmd\_dict3 = {sqlText: dbCommand3};

var tabCmd\_dict1 = {sqlText: tabCommand1};

var tabCmd\_dict2 = {sqlText: tabCommand2};

var tabCmd\_dict3 = {sqlText: tabCommand3};

var org\_stmt1 = snowflake.createStatement(orgCmd\_dict1);

var org\_rs1 = org\_stmt1.execute();

var org\_stmt2 = snowflake.createStatement(orgCmd\_dict2);

var org\_rs2 = org\_stmt2.execute();

var org\_stmt3 = snowflake.createStatement(orgCmd\_dict3);

var org\_rs3 = org\_stmt3.execute();

var acc\_stmt1 = snowflake.createStatement(accCmd\_dict1);

var acc\_rs1 = acc\_stmt1.execute();

var acc\_stmt2 = snowflake.createStatement(accCmd\_dict2);

var acc\_rs2 = acc\_stmt2.execute();

var acc\_stmt3 = snowflake.createStatement(accCmd\_dict3);

var acc\_rs3 = acc\_stmt3.execute();

var db\_stmt1 = snowflake.createStatement(dbCmd\_dict1);

var db\_rs1 = db\_stmt1.execute();

var db\_stmt2 = snowflake.createStatement(dbCmd\_dict2);

var db\_rs2 = db\_stmt2.execute();

var db\_stmt3 = snowflake.createStatement(dbCmd\_dict3);

var db\_rs3 = db\_stmt3.execute();

var tab\_stmt1 = snowflake.createStatement(tabCmd\_dict1);

var tab\_rs1 = tab\_stmt1.execute();

var tab\_stmt2 = snowflake.createStatement(tabCmd\_dict2);

var tab\_rs2 = tab\_stmt2.execute();

var tab\_stmt3 = snowflake.createStatement(tabCmd\_dict3);

var tab\_rs3 = tab\_stmt3.execute();

//CREATE TASK : CHANGE CRON EXPRESSION TO SCHEDULE SP

snowflake.execute({sqlText:`CREATE OR REPLACE TASK Monitor\_Storage\_Monitor\_Task WAREHOUSE = 'MONITOR\_WH' SCHEDULE = 'USING CRON 0 2 \* \* SUN UTC' AS call Storage\_Monitor\_SP();`});

snowflake.execute({sqlText:`alter task Monitor\_Storage\_Monitor\_Task resume;`});

return 'TRANSIENT TABLES CREATED FOR STORAGE COST MONITORING. TABLES NAME : 1.ORG\_STORAGE\_COST 2.ACCOUNT\_STORAGE\_COST 3.DB\_STORAGE\_COST 4.TABLE\_STORAGE\_COST';

$$;

**Note :** Call the storage monitor once to create all the tables and to create the task that would call the stored procedure Storage\_monitor\_SP() every day.

call Storage\_Monitor\_SP();

**Note :** Alter the task Monitor\_Storage\_Monitor\_Task in order to schedule the calling of the stored procedure Storage\_Monitor\_SP() in order to refresh the table as per the user requirements.

# **Best Practices :-**

* It is mandatory to execute all the insert statements for historical load wherever required before executing SPs for incremental load of data.
* Always run historical load commands only once. If in any case you want to run it again then first truncate the respective custom table.
* You can always schedule the task as per your requirement . Reference for setting up task using cron expression - <https://docs.snowflake.com/en/sql-reference/sql/create-task.html#examples>
* If you want to customize bad query table results then always start by replacing the old table and do historical and incremental load again.