**Snowflake SP - 18 - Delta/Incremental Load Automation**

Here is the requirement for the Delta/Incremental load process, written in the style of the second screenshot.

Delta/Incremental load process:

Step 1: Check if the target table has a Primary Key defined on the table, if not, raise one exception.

Step 2: Prepare the Merge query required

Get the list of columns of the Primary Key

Get the list of columns required to form the update query

Get the list of columns and values required to form the insert query

Combine all these and prepare the Merge Query

Step 3: Run the Merge query to load the data

CREATE OR REPLACE PROCEDURE EMP.PROCS.SP\_SRC\_TO\_TGT\_DELTA\_LOAD("SOURCE\_TABLE" VARCHAR, "TARGET\_TABLE" VARCHAR)

RETURNS VARCHAR

LANGUAGE SQL

EXECUTE AS CALLER

AS

$$

DECLARE

pk\_count INTEGER;

tgt\_db\_name VARCHAR;

tgt\_schema\_name VARCHAR;

tgt\_table\_name VARCHAR;

join\_keys VARCHAR;

update\_columns VARCHAR;

insert\_columns VARCHAR;

insert\_values VARCHAR;

merge\_statement VARCHAR;

no\_keys\_found EXCEPTION (-20100, 'As there is no Primary Key present in Target table, Delta Load can''t be done.');

BEGIN

-- Step 1: Check for Primary Key existence on the target table

SELECT

COUNT(\*) INTO pk\_count

FROM

INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS

WHERE

CONSTRAINT\_TYPE = 'PRIMARY KEY'

AND TABLE\_NAME = :TARGET\_TABLE;

IF (pk\_count = 0) THEN

RAISE no\_keys\_found;

END IF;

-- Extract database and schema names from the target table string

SELECT

SPLIT\_PART(:TARGET\_TABLE, '.', 1),

SPLIT\_PART(:TARGET\_TABLE, '.', 2),

SPLIT\_PART(:TARGET\_TABLE, '.', 3)

INTO

tgt\_db\_name,

tgt\_schema\_name,

tgt\_table\_name;

-- Step 2: Prepare the dynamic MERGE query

-- Get primary key columns for the JOIN clause

SELECT

LISTAGG(COLUMN\_NAME, ' AND ') INTO join\_keys

FROM

INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS T

JOIN INFORMATION\_SCHEMA.KEY\_COLUMN\_USAGE K

ON T.CONSTRAINT\_NAME = K.CONSTRAINT\_NAME

WHERE

T.CONSTRAINT\_TYPE = 'PRIMARY KEY'

AND T.TABLE\_NAME = :TARGET\_TABLE;

-- Get all columns to form the UPDATE and INSERT clauses

SELECT

LISTAGG(COLUMN\_NAME || ' = SOURCE.' || COLUMN\_NAME, ', ') WITHIN GROUP (ORDER BY ORDINAL\_POSITION) INTO update\_columns,

LISTAGG(COLUMN\_NAME, ', ') WITHIN GROUP (ORDER BY ORDINAL\_POSITION) INTO insert\_columns,

LISTAGG('SOURCE.' || COLUMN\_NAME, ', ') WITHIN GROUP (ORDER BY ORDINAL\_POSITION) INTO insert\_values

FROM

INFORMATION\_SCHEMA.COLUMNS

WHERE

TABLE\_NAME = :TARGET\_TABLE

AND COLUMN\_NAME NOT IN (

SELECT COLUMN\_NAME

FROM INFORMATION\_SCHEMA.KEY\_COLUMN\_USAGE

WHERE TABLE\_NAME = :TARGET\_TABLE AND CONSTRAINT\_NAME IN (

SELECT CONSTRAINT\_NAME

FROM INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS

WHERE TABLE\_NAME = :TARGET\_TABLE AND CONSTRAINT\_TYPE = 'PRIMARY KEY'

)

);

-- Combine everything to form the final MERGE statement

merge\_statement := 'MERGE INTO ' || :TARGET\_TABLE || ' AS TARGET USING ' || :SOURCE\_TABLE || ' AS SOURCE ON TARGET.' || join\_keys || ' WHEN MATCHED THEN UPDATE SET ' || update\_columns || ' WHEN NOT MATCHED THEN INSERT (' || insert\_columns || ') VALUES (' || insert\_values || ');';

-- Step 3: Run the dynamic MERGE query

EXECUTE IMMEDIATE :merge\_statement;

RETURN 'Delta Load completed successfully.';

EXCEPTION

WHEN no\_keys\_found THEN

RETURN 'Error: Delta Load failed. ' || SQLERRM;

WHEN OTHER THEN

RETURN 'An unexpected error occurred: ' || SQLERRM;

END;

$$;