SNOWFLAKE - Mini Project 5

1. Create a Database & Schema
    1. Database : PRODUCT_DB
        1. Schema : PRODUCT_DATA

---

2. Create PRODUCT_SRC Table in the PRODUCT_DATA schema
    1. Table:
        1. Database: PRODUCT_DB
        2. Schema: PRODUCT_DATA
        3. Name: PRODUCT_SRC
    2. Table structure

        PRODUCTID STRING
        PRODUCTNAME STRING
        CATEGORY STRING
        PRICE FLOAT
        STOCKQUANTITY FLOAT
        SUPPLIER STRING
        RATING FLOAT

---

3. Create PRODUCT_TGT Table in the PRODUCT_DATA schema
    3. Table:
        1. Database: PRODUCT_DB
        2. Schema: PRODUCT_DATA
        3. Name: PRODUCT_TGT
    4. Table structure

        PRODUCTID STRING
        PRODUCTNAME STRING
        CATEGORY STRING
        PRICE FLOAT
        STOCKQUANTITY FLOAT
        SUPPLIER STRING
        RATING FLOAT

---

4. Create a stream named PRODUCT_STREAM on the PRODUCT_SRC table to track incremental changes in the data
    1. Stream:
        1. Database: PRODUCT_DB
        2. Schema: PRODUCT_DATA
        3. Name: PRODUCT_STREAM

---

5. Create a task named PRODUCT_TASK that runs a MERGE statement every 1 minute to implement Slowly Changing Dimension Type 1 (SCD1) logic.
    1. Task:
        1. Database: PRODUCT_DB
        2. Schema: PRODUCT_DATA
        3. Name: PRODUCT_TASK
    2. Merge Logic:
        1. Primary Key: PRODUCTID
        2. Source Table: PRODUCT_DB.PRODUCT_DATA.PRODUCT_STREAM
        3. Target Table: PRODUCT_DB.PRODUCT_DATA.PRODUCT_TGT

6. Create an Internal Stage
    1. Create an Internal Stage in the PRODUCT_DATA schema
        1. Database: PRODUCT_DB
        2. Schema: PRODUCT_DATA
        3. Name: PRODUCT_STAGE

---

7. Upload the product_fulldata.csv file to the Internal Stage
    1. Upload the product_fulldata.csv data file in the internal stage PRODUCT_STAGE
    2. List the file in the internal stage

---

8. Load product_fulldata.csv Data from the Stage into the PRODUCT_SRC Table
    1. To load the product_fulldata.csv into the PRODUCT_SRC table, use the COPY INTO command

---

9. Verify that the product_fulldata.csv file is automatically loaded into the PRODUCT_TGT table using the configured stream and task after 1 min

---

10. Upload the product_changedata.csv file to the Internal Stage
    1. Upload the product_changedata.csv data file in the internal stage PRODUCT_STAGE
    2. List the file in the internal stage

---

11. Load product_changedata.csv Data from the Stage into the PRODUCT_SRC Table
    1. To load the product_changedata.csv into the PRODUCT_SRC table, use the COPY INTO command

---

12. Confirm that the product_changedata.csv file is automatically loaded into the PRODUCT_TGT table using the configured stream and task, and verify the implementation of SCD1 logic after 1 min

Solution:

```
-- ##########################################################
-- ###### (1) Create a Database & Schema
-- ##########################################################

-- SET ROLE AND WAREHOUSE:
USE ROLE ACCOUNTADMIN;
USE WAREHOUSE COMPUTE_WH;

-- CREATE DATABASE AND SCHEMA:
CREATE DATABASE PRODUCT_DB;
CREATE SCHEMA PRODUCT_DATA;
```

```
-- ###################################################################
-- ###### (2) Create PRODUCT_SRC Table in the PRODUCT_DATA schema
-- ###################################################################

USE SCHEMA PRODUCT_DB.PRODUCT_DATA;

-- CREATE TABLE:
CREATE OR REPLACE TABLE PRODUCT_DB.PRODUCT_DATA.PRODUCT_SRC
    (PRODUCTID STRING,
    PRODUCTNAME STRING,
    CATEGORY STRING,
    PRICE FLOAT,
    STOCKQUANTITY FLOAT,
    SUPPLIER STRING,
    RATING FLOAT);

SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_SRC;
```

```
-- ###################################################################
-- ###### (3) Create PRODUCT_TGT Table in the PRODUCT_DATA schema
-- ###################################################################

-- CREATE TABLE:
CREATE OR REPLACE TABLE PRODUCT_DB.PRODUCT_DATA.PRODUCT_TGT
    (PRODUCTID STRING,
    PRODUCTNAME STRING,
    CATEGORY STRING,
    PRICE FLOAT,
    STOCKQUANTITY FLOAT,
    SUPPLIER STRING,
    RATING FLOAT);

SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_TGT;
```

```
-- ##########################################################################
-- ###### (4) Create a stream named PRODUCT_STREAM on the PRODUCT_SRC table
-- ######     to track incremental changes in the data
-- ##########################################################################

-- CREATE A STANDARD STREAM ON TABLE:
CREATE OR REPLACE STREAM PRODUCT_DB.PRODUCT_DATA.PRODUCT_STREAM ON TABLE
PRODUCT_DB.PRODUCT_DATA.PRODUCT_SRC;

SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_STREAM;
```

```
-- ################################################################################
-- ###### (5) Create a task named PRODUCT_TASK that runs a MERGE statement every
-- ######     1 minute to implement Slowly Changing dimension Type 1 (SCD1) logic.
-- ################################################################################

-- CREATE A SCD1 TASK USING MERGING LOGIC:
CREATE OR REPLACE TASK PRODUCT_DB.PRODUCT_DATA.PRODUCT_TASK
WAREHOUSE = COMPUTE_WH
SCHEDULE = '1 MINUTE'
AS

    MERGE INTO PRODUCT_DB.PRODUCT_DATA.PRODUCT_TGT AS T
    USING PRODUCT_DB.PRODUCT_DATA.PRODUCT_STREAM AS S
    ON T.PRODUCTID = S.PRODUCTID

    -- STEP 1: UPDATING EXISTING ROWS
    WHEN MATCHED AND (
            T.PRODUCTNAME <> S.PRODUCTNAME OR
            T.CATEGORY <> S.CATEGORY OR
            T.PRICE <> S.PRICE OR
            T.STOCKQUANTITY <> S.STOCKQUANTITY OR
            T.SUPPLIER <> S.SUPPLIER OR
            T.RATING <> S.RATING)
    THEN UPDATE SET
        PRODUCTNAME = S.PRODUCTNAME,
        CATEGORY = S.CATEGORY,
        PRICE = S.PRICE,
        STOCKQUANTITY = S.STOCKQUANTITY,
        SUPPLIER = S.SUPPLIER,
        RATING = S.RATING


    -- STEP 2: INSERT NEW ROWS
    WHEN NOT MATCHED THEN
        INSERT (PRODUCTID, PRODUCTNAME, CATEGORY, PRICE, STOCKQUANTITY, SUPPLIER,
                 RATING)
        VALUES (S.PRODUCTID, S.PRODUCTNAME, S.CATEGORY, S.PRICE, S.STOCKQUANTITY,
                 S.SUPPLIER, S.RATING);

SHOW TASKS;
```

```sql
-- ###################################################################
-- ###### (6) Create an Internal Stage
-- ###################################################################

-- CREATE AN INTERNAL STAGE:
CREATE OR REPLACE STAGE PRODUCT_DB.PRODUCT_DATA.PRODUCT_STAGE;
```

```sql
-- #######################################################################
-- ###### (7) Upload the product_fulldata.csv file to the Internal Stage
-- #######################################################################

-- List the file in the internal stage before uploading the file
LIST @PRODUCT_DB.PRODUCT_DATA.PRODUCT_STAGE;

------------------------------------------------------------------------------------
--- UPLOAD MANUALLY THE product_fulldata.csv FILE INTO THE INTERNAL PRODUCT_STAGE --
------------------------------------------------------------------------------------

-- List the file in the internal stage after uploading the file
LIST @PRODUCT_DB.PRODUCT_DATA.PRODUCT_STAGE;

-- #################################################################################
-- #### (8) Load product_fulldata.csv Data from the Stage into the PRODUCT_SRC Table
-- #################################################################################

-- CREATE A CSV FILE FORMAT:
CREATE OR REPLACE FILE FORMAT CSV_FORMAT
TYPE = 'CSV'
FIELD_DELIMITER = ','
RECORD_DELIMITER = '\n'
SKIP_HEADER = 1;

COPY INTO PRODUCT_DB.PRODUCT_DATA.PRODUCT_SRC
FROM @PRODUCT_DB.PRODUCT_DATA.PRODUCT_STAGE/product_fulldata.csv
FILE_FORMAT = (FORMAT_NAME = PRODUCT_DB.PRODUCT_DATA.CSV_FORMAT);

SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_SRC;

-- #################################################################################
-- ##### (9) Verify that the product_fulldata.csv file is automatically loaded into
-- #####     the PRODUCT_TGT table using the configured stream and task after 1 min.
-- #################################################################################

-- CHECK THE MERGING OF TWO TABLES:
SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_TGT;

-- ALTER STATUS OF THE TASK TO RESUME:
ALTER TASK PRODUCT_DB.PRODUCT_DATA.PRODUCT_TASK RESUME;
SHOW TASKS;

-- CHECK THE MERGING OF TWO TABLES:
SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_TGT;
```

```sql
-- ################################################################################
-- ##### (10) Upload the product_changedata.csv file to the Internal Stage.
-- ################################################################################

-- List the file in the internal stage before uploading the file
LIST @PRODUCT_DB.PRODUCT_DATA.PRODUCT_STAGE;

-------------------------------------------------------------------------------------
-- UPLOAD MANUALLY THE product_changedata.csv FILE INTO THE INTERNAL PRODUCT_STAGE -
-------------------------------------------------------------------------------------

-- List the file in the internal stage after uploading the file
LIST @PRODUCT_DB.PRODUCT_DATA.PRODUCT_STAGE;


-- ##################################################################################
-- # (11) Load product_changedata.csv Data from the Stage into the PRODUCT_SRC Table
-- ##################################################################################


COPY INTO PRODUCT_DB.PRODUCT_DATA.PRODUCT_SRC
FROM @PRODUCT_DB.PRODUCT_DATA.PRODUCT_STAGE/product_changedata.csv
FILE_FORMAT = (FORMAT_NAME = PRODUCT_DB.PRODUCT_DATA.CSV_FORMAT);

SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_SRC;


-- ##################################################################################
-- ##### (12) Confirm that the product_changedata.csv file is automatically loaded
-- #####      into the PRODUCT_TGT table using the configured stream and task, and
-- #####      verify the implementation of SCD1 logic after 1 min.
-- ##################################################################################


-- CHECK THE MERGING OF TWO TABLES:
SELECT * FROM PRODUCT_DB.PRODUCT_DATA.PRODUCT_TGT;










-- ################################################################################
-- ALTER STATUS OF THE TASK TO SUSPEND:
ALTER TASK PRODUCT_DB.PRODUCT_DATA.PRODUCT_TASK SUSPEND;
SHOW TASKS;
```