

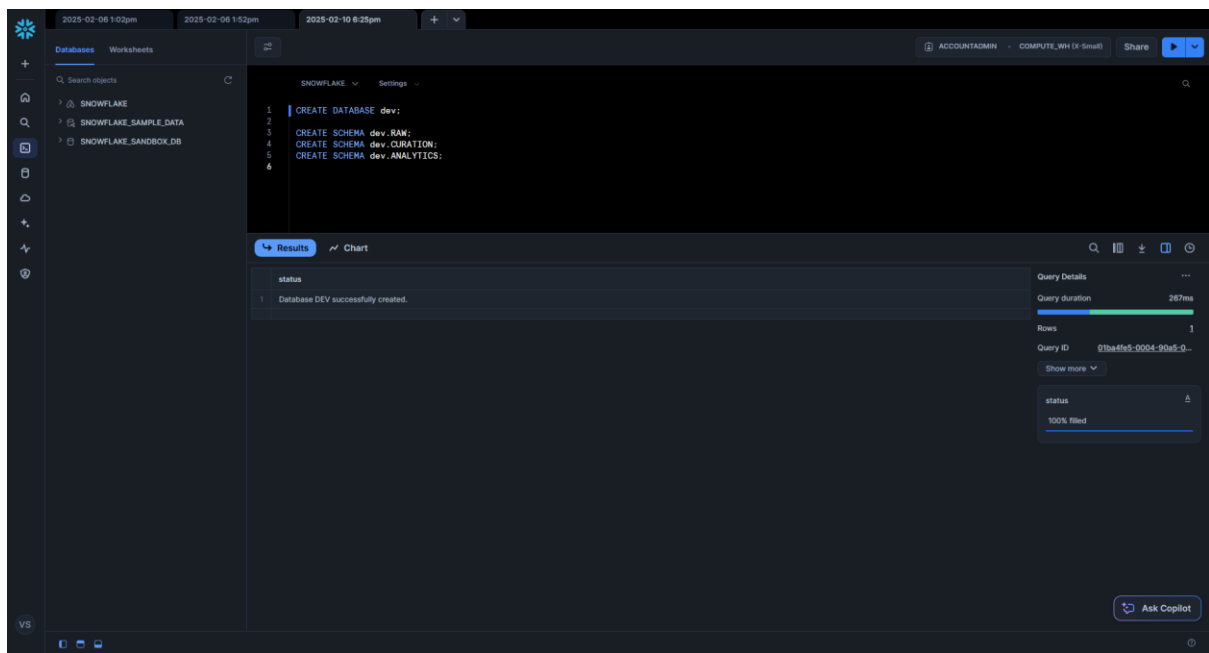
DATA 226- DATAWAREHOUSE

Homework 2

Name : Vimalanandhan Sivanandham

SJSU ID: 017596436

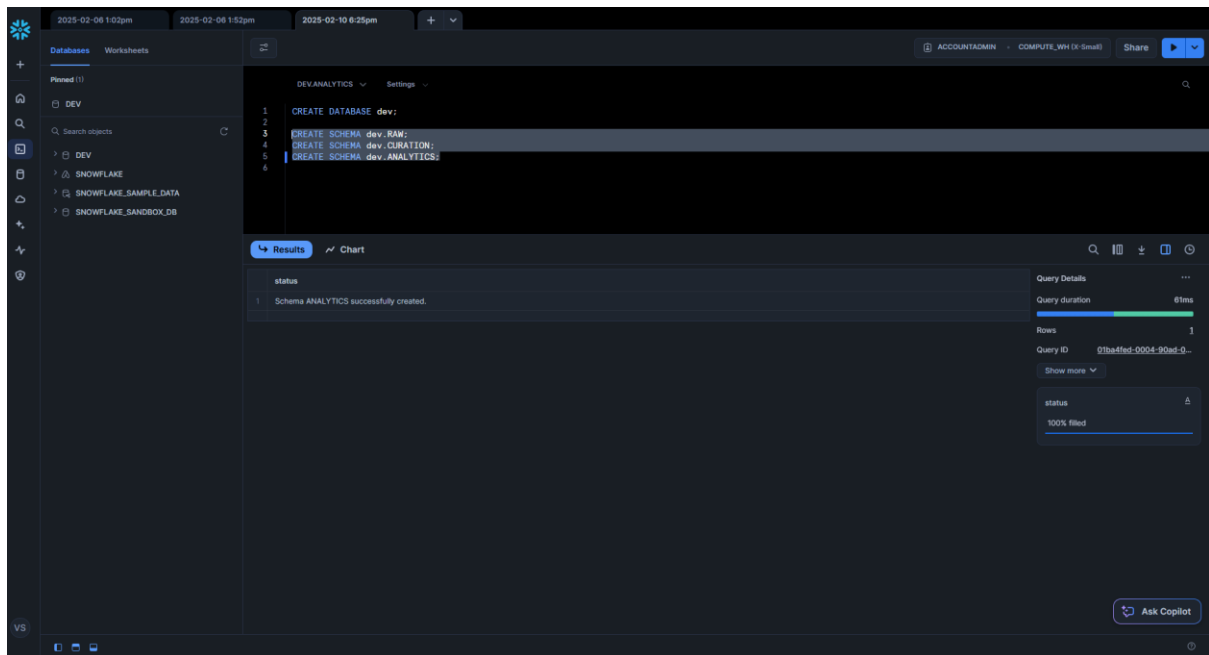
1.(+1) Create database dev and schemas RAW, CURATION and ANALYTICS



CODE:

--- DROP DATABASE DEV

CREATE DATABASE dev;



CODE:

--- CREATE SCHEMA

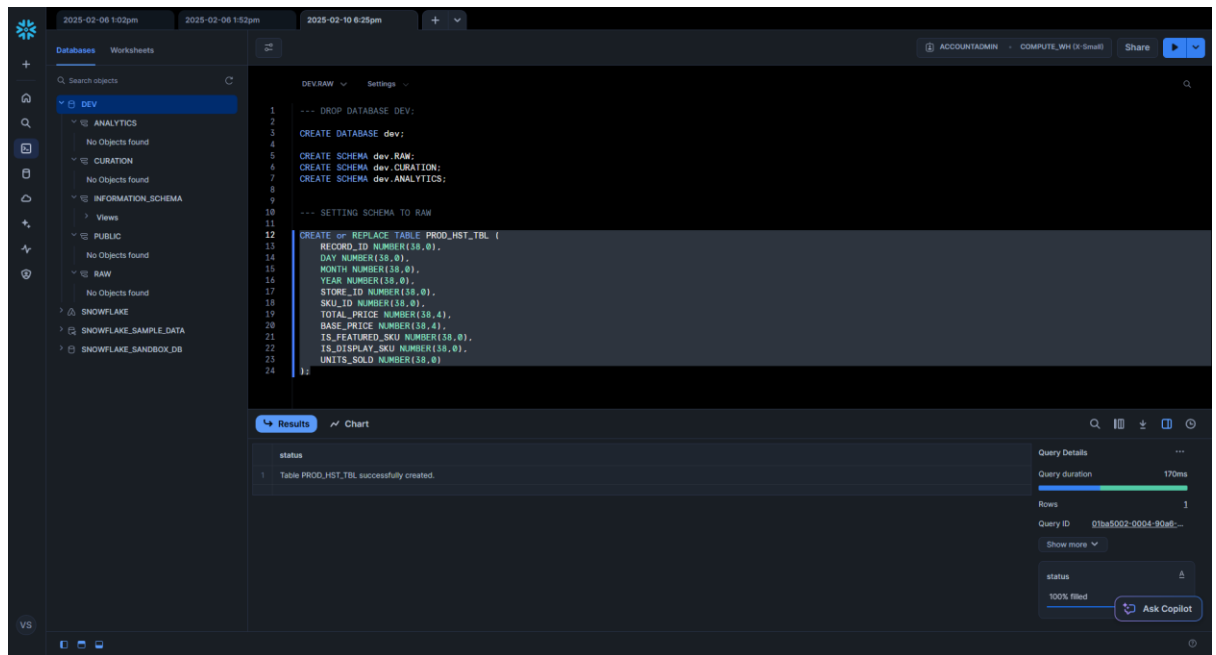
CREATE SCHEMA dev.RAW;

CREATE SCHEMA dev.CURATION;

CREATE SCHEMA dev.ANALYTICS;

2.(+4) In RAW schema, create

- **prod_hst_tbl** - table
- **prod_stg** - stage
- **prod_raw_task** - task (alter the task)
- **prod_stream** – stream



CODE:

--- SETTING SCHEMA TO RAW

CREATE or REPLACE TABLE PROD_HST_TBL (

 RECORD_ID NUMBER(38,0),

 DAY NUMBER(38,0),

 MONTH NUMBER(38,0),

 YEAR NUMBER(38,0),

 STORE_ID NUMBER(38,0),

 SKU_ID NUMBER(38,0),

 TOTAL_PRICE NUMBER(38,4),

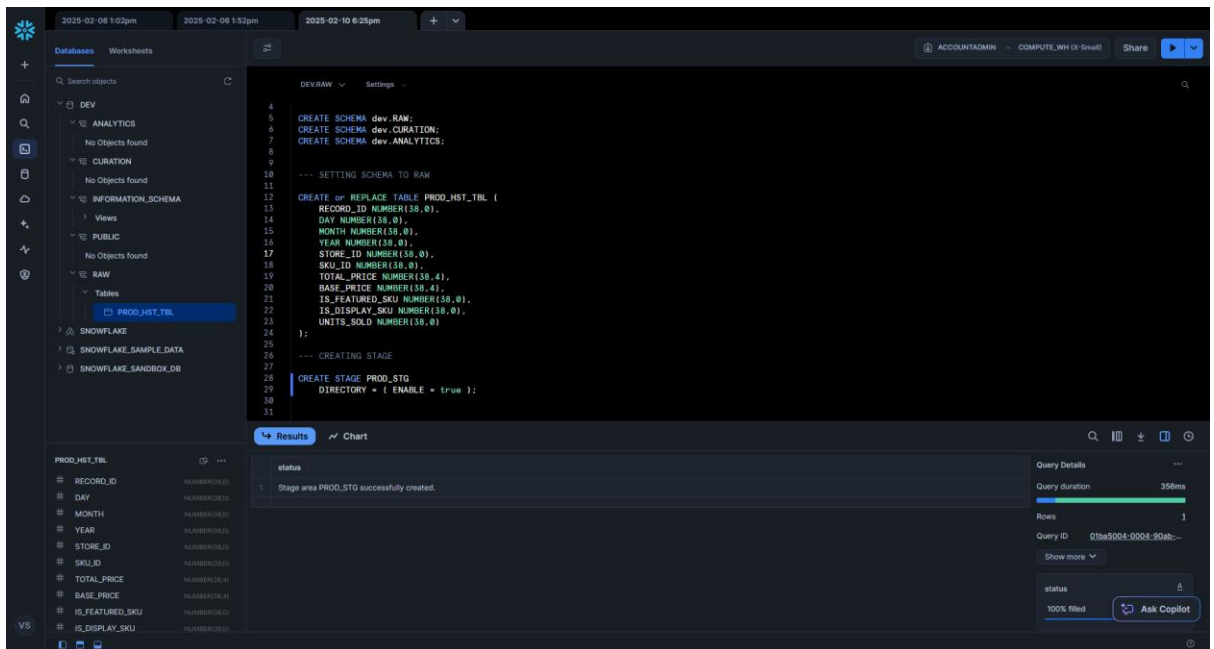
 BASE_PRICE NUMBER(38,4),

 IS_FEATURED_SKU NUMBER(38,0),

 IS_DISPLAY_SKU NUMBER(38,0),

 UNITS_SOLD NUMBER(38,0)

);

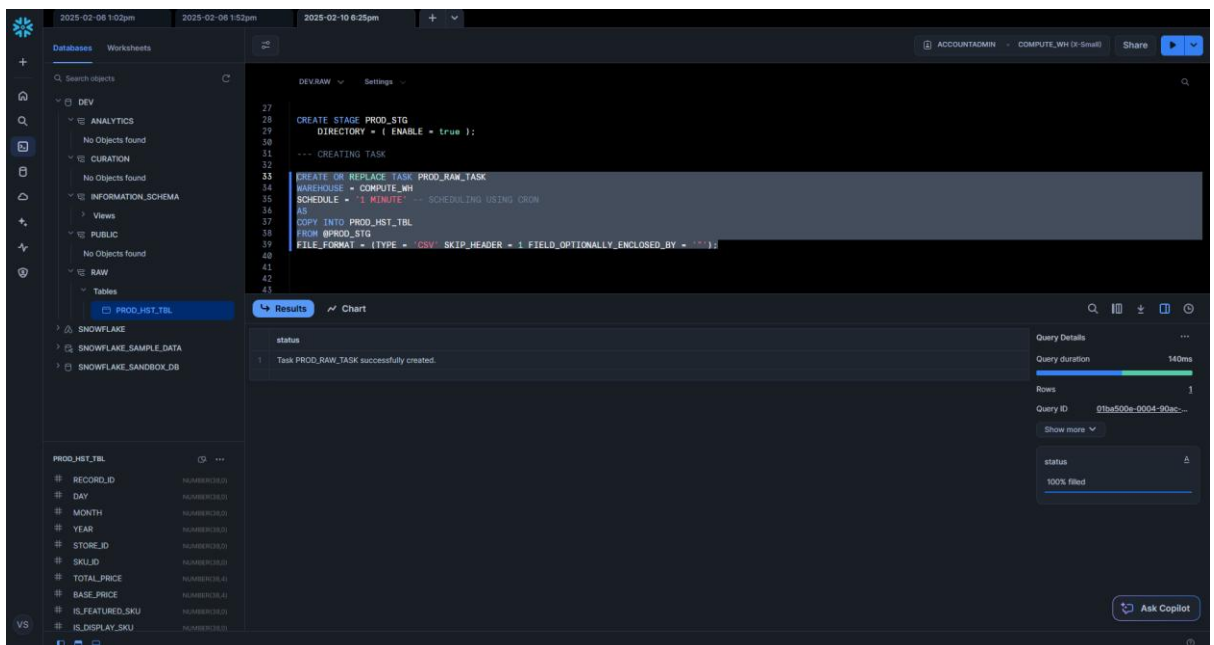


CODE:

--- CREATING STAGE

CREATE STAGE PROD_STG

DIRECTORY = (ENABLE = true);



CODE:

--- CREATING TASK

CREATE OR REPLACE TASK PROD_RAW_TASK

WAREHOUSE = COMPUTE_WH

SCHEDULE = '1 MINUTE' -- SCHEDULING USING CRON

AS

COPY INTO PROD_HST_TBL

FROM @PROD_STG

FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY =
''');

The screenshot displays the Snowflake SQL Editor interface. The top navigation bar shows the current session as 'ACCOUNTADMIN' on 'COMPUTE_WH (X-Small)' warehouse. The left sidebar contains a 'Databases' tab with a search bar and a tree view of the database structure, including 'DEV', 'ANALYTICS', 'CURATION', 'INFORMATION_SCHEMA', 'PUBLIC', 'RAW', and 'SNOWFLAKE'. The 'RAW' database is expanded, showing a table named 'PROD_HST_TBL'. The main editor area shows a SQL query with line numbers 26 through 46. The query includes comments for creating a stage, a task, and a stream. The 'Results' tab is active, showing a single status message: 'Stream PROD_STREAM successfully created.' The right sidebar provides 'Query Details' including a duration of 398ms, 1 row, and a query ID. An 'Ask Copilot' button is visible at the bottom right.

```
26 --- CREATING STAGE
27
28 CREATE STAGE PROD_STG
29   DIRECTORY = ( ENABLE = true );
30
31 --- CREATING TASK
32
33 CREATE OR REPLACE TASK PROD_RAW_TASK
34   WAREHOUSE = COMPUTE_WH
35   SCHEDULE = '1 MINUTE' -- SCHEDULING USING CRON
36   AS
37   COPY INTO PROD_HST_TBL
38   FROM @PROD_STG
39   FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY = '');
40
41 --- CREATING STREAM
42
43 CREATE OR REPLACE STREAM prod_stream
44   ON TABLE PROD_HST_TBL;
```

status
1 Stream PROD_STREAM successfully created.

Query Details

- Query duration: 398ms
- Rows: 1
- Query ID: 91ba500f-0004-929b-9...
- status: 100% filled

Ask Copilot

CODE:

--- CREATING STREAM

CREATE OR REPLACE STREAM prod_stream

ON TABLE PROD_HST_TBL;

SHOW TASKS;

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema, including the 'PROD_HST_TBL' table. The main editor area contains the following SQL code:

```

27
28 CREATE STAGE PROD_STG
29   DIRECTORY = ( ENABLE = true );
30
31 --- CREATING TASK
32
33 CREATE OR REPLACE TASK PROD_RAW_TASK
34   WAREHOUSE = COMPUTE_WH
35   SCHEDULE = '1 MINUTE' --- SCHEDULING USING CRON
36   AS
37   COPY INTO PROD_HST_TBL
38   FROM @PROD_STG
39   FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY = ''');
40
41 --- CREATING STREAM
42
43 CREATE OR REPLACE STREAM prod_stream
44   ON TABLE PROD_HST_TBL;
45
46 SHOW TASKS;
47
48
49
50
51

```

The results pane at the bottom shows the task definition and its status. The task is named 'PROD_RAW_TASK', owned by 'ACCOUNTADMIN', and is currently 'suspended'. The query duration is 57ms.

task_name	owner	comment	warehouse	schedule	predecessors	state	definition	com	Query Details
PROD_RAW_TASK	ACCOUNTADMIN		COMPUTE_WH	1 MINUTE		suspended	COPY INTO PROD_HST_TBL FROM @PROD_STG FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY = ''');		Query duration: 57ms

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema, including the 'PROD_HST_TBL' table. The main editor area contains the following SQL code:

```

27
28 CREATE STAGE PROD_STG
29   DIRECTORY = ( ENABLE = true );
30
31 --- CREATING TASK
32
33 CREATE OR REPLACE TASK PROD_RAW_TASK
34   WAREHOUSE = COMPUTE_WH
35   SCHEDULE = '1 MINUTE' --- SCHEDULING USING CRON
36   AS
37   COPY INTO PROD_HST_TBL
38   FROM @PROD_STG
39   FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY = ''');
40
41 --- CREATING STREAM
42
43 CREATE OR REPLACE STREAM prod_stream
44   ON TABLE PROD_HST_TBL;
45
46 SHOW TASKS;
47
48
49
50
51

```

The results pane at the bottom shows the status of the task. The status is 'Statement executed successfully'.

status	Query Details
Statement executed successfully.	Query duration: 133ms

CODE:

--- ALTER TASK

ALTER TASK PROD_RAW_TASK RESUME;

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema, including the 'PROD_HST_TBL' table. The main editor area contains a SQL script for creating a task named 'PROD_RAM_TASK' that runs every minute, copying data from 'PROD_STG' to 'PROD_HST_TBL'. The 'Results' tab is active, showing a single row of metadata for the task execution.

name	schema_name	owner	comment	warehouse	schedule	predecessors	state	definition
PROD_HST_TBL	RAW	ACCOUNTADMIN		COMPUTE_WH	1 MINUTE		started	COPY INTO PROD_HST_TBL FROM @PROD_STG FILE_FORMAT = (TYPE = 'CSV' SKIP...

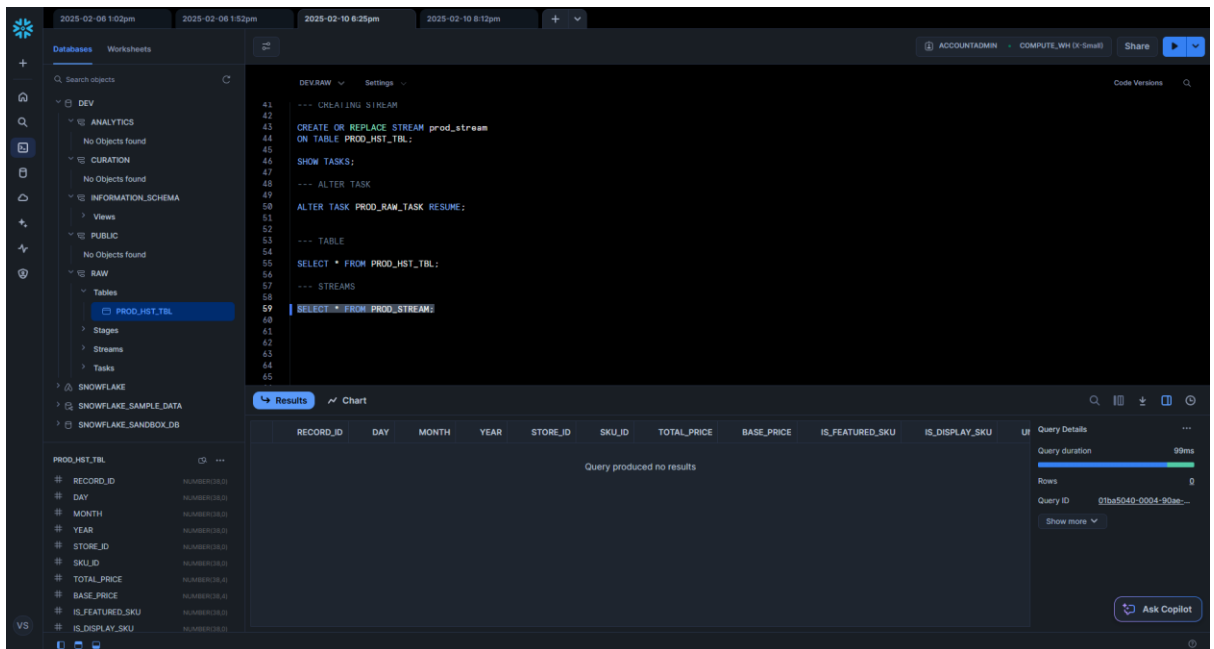
The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema, including the 'PROD_HST_TBL' table. The main editor area contains a SQL script for creating a stream named 'prod_stream' on the 'PROD_HST_TBL' table. The 'Results' tab is active, showing a message indicating that the query produced no results.

RECORD_ID	DAY	MONTH	YEAR	STORE_ID	SKU_ID	TOTAL_PRICE	BASE_PRICE	IS_FEATURED_SKU	IS_DISPLAY_SKU
Query produced no results									

CODE:

--- TABLE

SELECT * FROM PROD_HST_TBL;



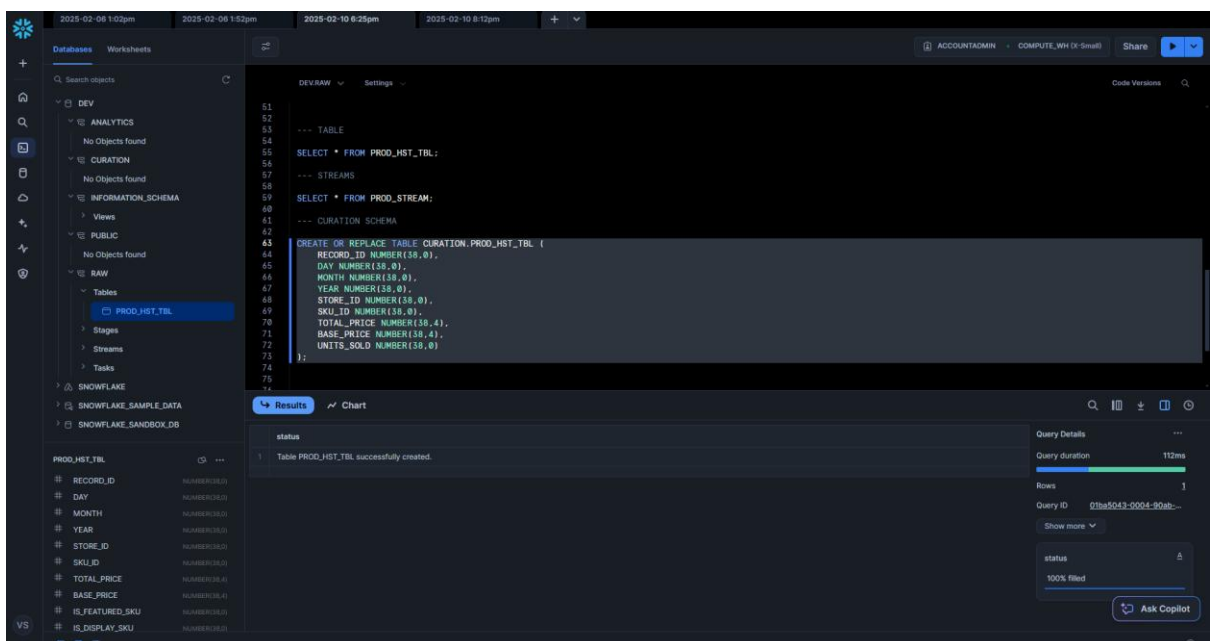
CODE:

--- STREAMS

SELECT * FROM PROD_STREAM;

3.(+3) In CURATION schema, create

- prod_hst_tbl - table
- prod_curation_task using MERGE (alter the task)



CODE:

--- CURATION SCHEMA

CREATE OR REPLACE TABLE CURATION.PROD_HST_TBL (

RECORD_ID NUMBER(38,0),

DAY NUMBER(38,0),

MONTH NUMBER(38,0),

YEAR NUMBER(38,0),

STORE_ID NUMBER(38,0),

SKU_ID NUMBER(38,0),

TOTAL_PRICE NUMBER(38,4),

BASE_PRICE NUMBER(38,4),

UNITS_SOLD NUMBER(38,0)

);

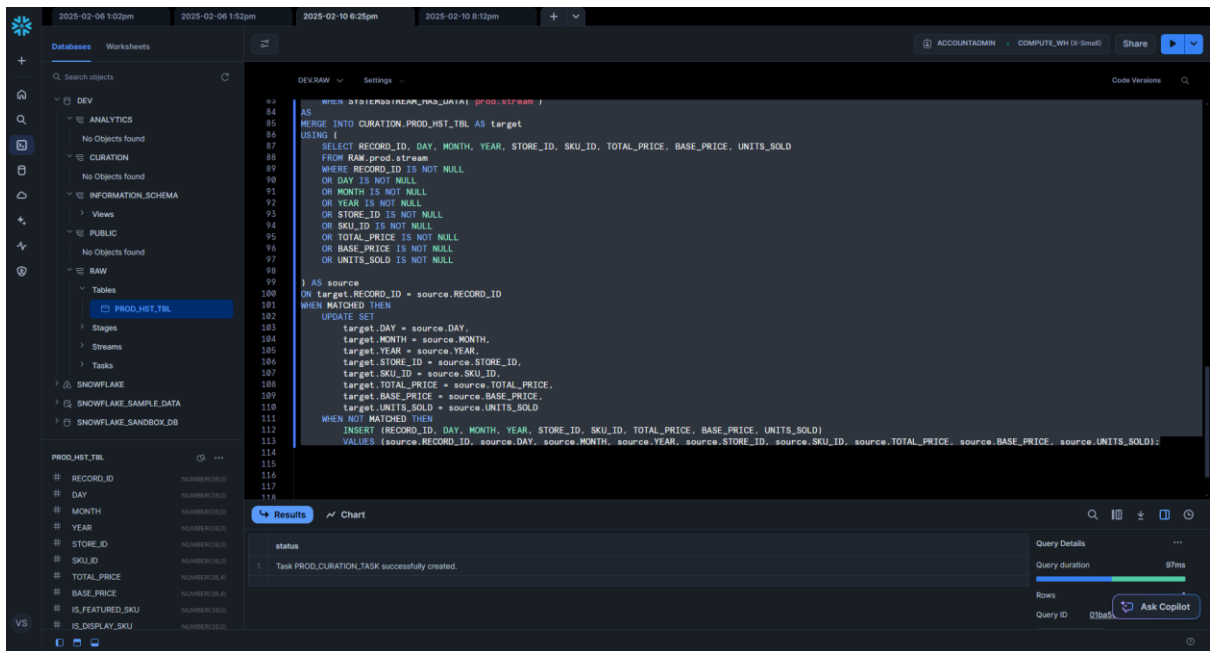
The screenshot displays the Snowflake SQL Editor interface. The left sidebar shows a database schema tree with the following structure:

- DEV
 - ANALYTICS
 - No Objects found
 - CURATION
 - No Objects found
 - INFORMATION_SCHEMA
 - Views
 - PUBLIC
 - No Objects found
 - RAW
 - Tables
 - PROD_HST_TBL
 - Stages
 - Streams
 - Tasks
 - SNOWFLAKE
 - SNOWFLAKE_SAMPLE_DATA
 - SNOWFLAKE_SANDBOX_DB

The main editor area shows the following SQL code:

```
78  
79  
80 --- TASK CREATION  
81  
82 CREATE OR REPLACE TASK RAW_PROD_CURATION_TASK  
83 WAREHOUSE = COMPUTE_WH  
84 WHEN SYSTEMSTREAM_HAS_DATA('prod.stream')  
85 AS  
86 MERGE INTO CURATION.PROD_HST_TBL AS target  
87 USING (  
88 SELECT RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, UNITS_SOLD  
89 FROM RAW.prod-stream  
90 WHERE RECORD_ID IS NOT NULL  
91 OR DAY IS NOT NULL  
92 OR MONTH IS NOT NULL  
93 OR YEAR IS NOT NULL  
94 OR STORE_ID IS NOT NULL  
95 OR SKU_ID IS NOT NULL  
96 OR TOTAL_PRICE IS NOT NULL  
97 OR BASE_PRICE IS NOT NULL  
98 OR UNITS_SOLD IS NOT NULL  
99 ) AS source  
100 ON target.RECORD_ID = source.RECORD_ID  
101 WHEN MATCHED THEN  
102 UPDATE SET  
103 target.DAY = source.DAY,  
104 target.MONTH = source.MONTH,  
105 target.YEAR = source.YEAR,  
106 target.STORE_ID = source.STORE_ID,  
107 target.SKU_ID = source.SKU_ID,  
108 target.TOTAL_PRICE = source.TOTAL_PRICE,  
109 target.BASE_PRICE = source.BASE_PRICE,  
110 target.UNITS_SOLD = source.UNITS_SOLD  
111 WHEN NOT MATCHED THEN  
112 INSERT (RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, UNITS_SOLD)
```

The bottom of the editor shows the 'Results' tab with a status message: 'Task PROD_CURATION_TASK successfully created.' The 'Query Details' panel on the right indicates a query duration of 57ms and 0 rows returned.



CODE:

--- TASK CREATION

CREATE OR REPLACE TASK RAW.PROD_CURATOR_TASK

WAREHOUSE = COMPUTE_WH

WHEN SYSTEM\$STREAM_HAS_DATA('prod.stream')

AS

MERGE INTO CURATION.PROD_HST_TBL AS target

USING (

SELECT RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE,
BASE_PRICE, UNITS_SOLD

FROM RAW.prod.stream

WHERE RECORD_ID IS NOT NULL

OR DAY IS NOT NULL

OR MONTH IS NOT NULL

OR YEAR IS NOT NULL

OR STORE_ID IS NOT NULL

OR SKU_ID IS NOT NULL

OR TOTAL_PRICE IS NOT NULL

OR BASE_PRICE IS NOT NULL

OR UNITS_SOLD IS NOT NULL

) AS source

ON target.RECORD_ID = source.RECORD_ID

WHEN MATCHED THEN

UPDATE SET

target.DAY = source.DAY,

target.MONTH = source.MONTH,

target.YEAR = source.YEAR,

target.STORE_ID = source.STORE_ID,

target.SKU_ID = source.SKU_ID,

target.TOTAL_PRICE = source.TOTAL_PRICE,

target.BASE_PRICE = source.BASE_PRICE,

target.UNITS_SOLD = source.UNITS_SOLD

WHEN NOT MATCHED THEN

INSERT (RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE,
BASE_PRICE, UNITS_SOLD)

VALUES (source.RECORD_ID, source.DAY, source.MONTH, source.YEAR,
source.STORE_ID, source.SKU_ID, source.TOTAL_PRICE, source.BASE_PRICE,
source.UNITS_SOLD);

The screenshot displays the Snowflake SQL IDE interface. The left sidebar shows a database schema with tables like RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, IS_FEATURED_SKU, and IS_DISPLAY_SKU. The main editor shows a SQL query for a MERGE operation. The query starts with a SELECT from RAW_PROD_STREAM, followed by a MERGE INTO CURATION.PROD_HST_TBL AS target. The query includes a WHEN MATCHED THEN UPDATE SET clause and a WHEN NOT MATCHED THEN INSERT clause. The task definition at the bottom shows a task named COPY INTO PROD_HST_TBL FROM @PROD_STG_FILE_FORMAT, which is scheduled to run every 1 minute.

```
88 FROM RAW_PROD_STREAM
89 WHERE RECORD_ID IS NOT NULL
90 OR DAY IS NOT NULL
91 OR MONTH IS NOT NULL
92 OR YEAR IS NOT NULL
93 OR STORE_ID IS NOT NULL
94 OR SKU_ID IS NOT NULL
95 OR TOTAL_PRICE IS NOT NULL
96 OR BASE_PRICE IS NOT NULL
97 OR UNITS_SOLD IS NOT NULL
98
99 ) AS source
100 ON target.RECORD_ID = source.RECORD_ID
101 WHEN MATCHED THEN
102 UPDATE SET
103   target.DAY = source.DAY,
104   target.MONTH = source.MONTH,
105   target.YEAR = source.YEAR,
106   target.STORE_ID = source.STORE_ID,
107   target.SKU_ID = source.SKU_ID,
108   target.TOTAL_PRICE = source.TOTAL_PRICE,
109   target.BASE_PRICE = source.BASE_PRICE,
110   target.UNITS_SOLD = source.UNITS_SOLD
111 WHEN NOT MATCHED THEN
112 INSERT (RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, UNITS_SOLD)
113 VALUES (source.RECORD_ID, source.DAY, source.MONTH, source.YEAR, source.STORE_ID, source.SKU_ID, source.TOTAL_PRICE, source.BASE_PRICE, source.UNITS_SOLD);
114
115
116
117
118
119
120
121
122
```

Task Definition:

comment	warehouse	schedule	predecessors	state	definition	condition	Query Details
	JNTADMIN	COMPUTE_WH		suspended	MERGE INTO CURATION.PROD_HST_TBL AS target USING (SELECT RECORD_ID, DAY, M	SYSTEMSTREAM_HAS	Query duration 84ms
	JNTADMIN	COMPUTE_WH	1 MINUTE	started	COPY INTO PROD_HST_TBL FROM @PROD_STG_FILE_FORMAT = (TYPE = 'CSV' SKIP HEAD		Rows

CODE:

SHOW TASKS;

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema with the following structure:

- DEV
 - ANALYTICS (No Objects found)
 - CURATION (No Objects found)
 - INFORMATION_SCHEMA
 - PUBLIC (No Objects found)
 - RAW
 - PROD_HST_TBL (Selected)
 - Stages
 - Streams
 - Tasks
- SNOWFLAKE
- SNOWFLAKE_SAMPLE_DATA
- SNOWFLAKE_SANDBOX_DB

The main editor displays the following SQL code:

```
91 OR MONTH IS NOT NULL
92 OR YEAR IS NOT NULL
93 OR STORE_ID IS NOT NULL
94 OR SKU_ID IS NOT NULL
95 OR TOTAL_PRICE IS NOT NULL
96 OR BASE_PRICE IS NOT NULL
97 OR UNITS_SOLD IS NOT NULL
98
99 ) AS source
100 ON target.RECORD_ID = source.RECORD_ID
101 WHEN MATCHED THEN
102   UPDATE SET
103     target.DAY = source.DAY,
104     target.MONTH = source.MONTH,
105     target.YEAR = source.YEAR,
106     target.STORE_ID = source.STORE_ID,
107     target.SKU_ID = source.SKU_ID,
108     target.TOTAL_PRICE = source.TOTAL_PRICE,
109     target.BASE_PRICE = source.BASE_PRICE,
110     target.UNITS_SOLD = source.UNITS_SOLD
111 WHEN NOT MATCHED THEN
112   INSERT (RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, UNITS_SOLD)
113     VALUES (source.RECORD_ID, source.DAY, source.MONTH, source.YEAR, source.STORE_ID, source.SKU_ID, source.TOTAL_PRICE, source.BASE_PRICE, source.UNITS_SOLD);
114
115 SHOW TASKS;
116
117 SELECT * FROM CURATION.PROD_HST_TBL;
```

The bottom panel shows the 'Results' tab with a table header:

RECORD_ID	DAY	MONTH	YEAR	STORE_ID	SKU_ID	TOTAL_PRICE	BASE_PRICE	UNITS_SOLD
-----------	-----	-------	------	----------	--------	-------------	------------	------------

Below the header, it states: "Query produced no results". The right sidebar shows query details: "Query duration: 46ms", "Rows: 0", and "Query ID: 01ba509e-0004-80a7-...".

CODE:

SELECT * FROM CURATION.PROD_HST_TBL;

4.(+2) In ANALYTICS schema, create

- book_dy_tbl – Dynamic Table

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema with the following structure:

- DEV
 - ANALYTICS (No Objects found)
 - CURATION (No Objects found)
 - INFORMATION_SCHEMA
 - PUBLIC (No Objects found)
 - RAW
 - PROD_HST_TBL (Selected)
 - Stages
 - Streams
 - Tasks
- SNOWFLAKE
- SNOWFLAKE_SAMPLE_DATA
- SNOWFLAKE_SANDBOX_DB

The main editor displays the following SQL code:

```
100 ON target.RECORD_ID = source.RECORD_ID
101 WHEN MATCHED THEN
102   UPDATE SET
103     target.DAY = source.DAY,
104     target.MONTH = source.MONTH,
105     target.YEAR = source.YEAR,
106     target.STORE_ID = source.STORE_ID,
107     target.SKU_ID = source.SKU_ID,
108     target.TOTAL_PRICE = source.TOTAL_PRICE,
109     target.BASE_PRICE = source.BASE_PRICE,
110     target.UNITS_SOLD = source.UNITS_SOLD
111 WHEN NOT MATCHED THEN
112   INSERT (RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, UNITS_SOLD)
113     VALUES (source.RECORD_ID, source.DAY, source.MONTH, source.YEAR, source.STORE_ID, source.SKU_ID, source.TOTAL_PRICE, source.BASE_PRICE, source.UNITS_SOLD);
114
115 SHOW TASKS;
116
117 SELECT * FROM CURATION.PROD_HST_TBL;
```

Below the code, a comment reads: "---- CREATION OF DYNAMIC TABLE IN ANALYTICS SCHEMA". The bottom panel shows the 'Results' tab with a table header:

status
Dynamic table BOOK_DY_TBL successfully created.

The right sidebar shows query details: "Query duration: 1.7s", "Rows: 1", and "Query ID: 01ba509e-0004-80a7-...".

CODE:

--- CREATION OF DYNAMIC TABLE IN ANALYTICS SCHEMA

CREATE OR REPLACE DYNAMIC TABLE ANALYTICS.BOOK_DY_TBL

WAREHOUSE = 'COMPUTE_WH'

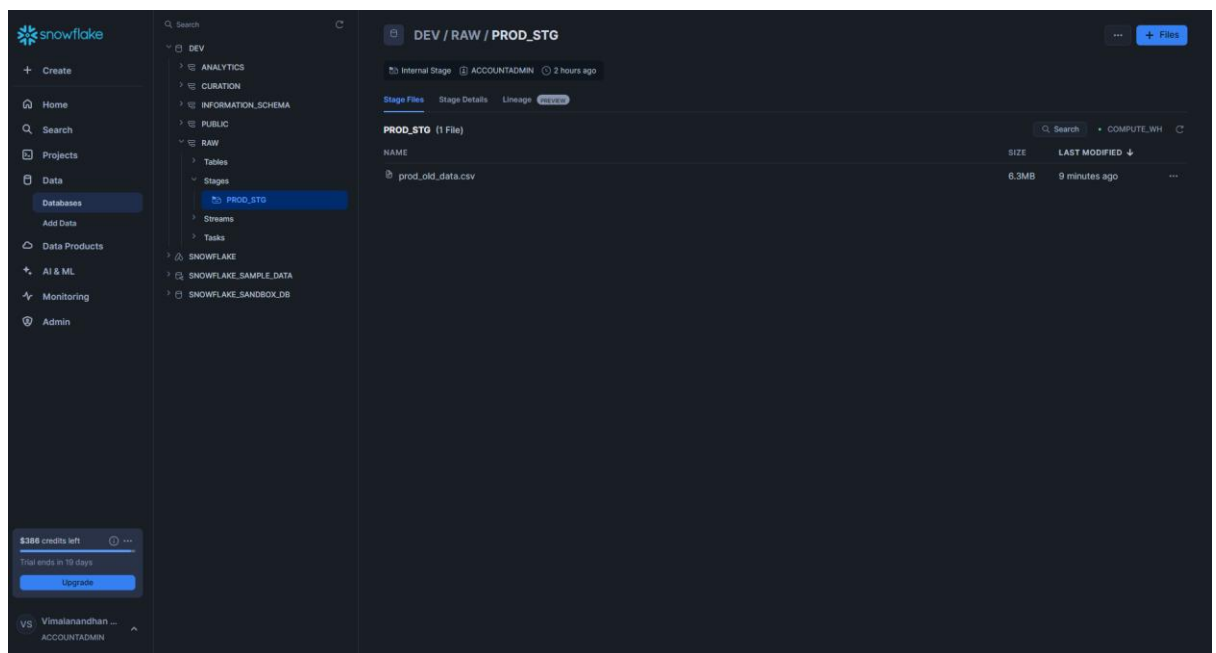
LAG = '1 MINUTE'

AS

SELECT DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE,
UNITS_SOLD FROM CURATION.PROD_HST_TBL

WHERE SKU_ID = '216425';

Uploading the old_csv_file:



The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema with the following structure:

- DEV
 - ANALYTICS
 - Dynamic Tables
 - CURATION
 - Tables
 - INFORMATION_SCHEMA
 - Views
 - PUBLIC
 - No Objects found
 - RAW
 - Tables
 - PROD_HST_TBL
 - Stages
 - Streams
 - Tasks
- SNOWFLAKE
 - SNOWFLAKE_SAMPLE_DATA
 - SNOWFLAKE_SANDBOX_DB

The main editor shows the following SQL script:

```

34 WAREHOUSE = COMPUTE_WH
35 SCHEDULE = '1 MINUTE' -- SCHEDULING USING CRON
36 AS
37 COPY INTO PROD_HST_TBL
38 FROM @PROD_STG
39 FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY = '');
40
41 --- CREATING STREAM
42
43 CREATE OR REPLACE STREAM prod_stream
44 ON TABLE PROD_HST_TBL;
45
46 SHOW TASKS;
47
48 --- ALTER TASK
49
50 ALTER TASK PROD_RAW_TASK RESUME;
51
52
53 --- TABLE
54
55 SELECT * FROM PROD_HST_TBL;
56
57 --- STREAMS
58
59 SELECT * FROM PROD_STREAM;
60
61 --- CURATION SCHEMA
  
```

The results pane shows the output of the query `SELECT * FROM PROD_HST_TBL`. The table has 129,446 rows. The columns are: RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, IS_FEATURED_SKU, IS_DISPLAY_SKU, and UNITS_SOLD. The first 8 rows are displayed:

RECORD_ID	DAY	MONTH	YEAR	STORE_ID	SKU_ID	TOTAL_PRICE	BASE_PRICE	IS_FEATURED_SKU	IS_DISPLAY_SKU	UNITS_SOLD
1	68037	2023	10	31	9672	216425	134.6625	0	0	36
2	68051	2023	10	31	9672	245338	498.0375	498.0375	0	1
3	68048	2023	10	31	9672	223153	178.1250	228.0000	0	0
4	68047	2023	10	31	9672	223245	213.7500	213.7500	0	0
5	68046	2023	10	31	9672	222785	234.4125	234.4125	0	0
6	68045	2023	10	31	9672	222087	226.5750	226.5750	0	0
7	68044	2023	10	31	9672	219029	327.0375	327.0375	0	0
8	68043	2023	10	31	9672	219009	197.3625	228.7125	0	0

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema with the following structure:

- DEV
 - ANALYTICS
 - Dynamic Tables
 - CURATION
 - Tables
 - INFORMATION_SCHEMA
 - Views
 - PUBLIC
 - No Objects found
 - RAW
 - Tables
 - PROD_HST_TBL
 - Stages
 - Streams
 - Tasks
- SNOWFLAKE
 - SNOWFLAKE_SAMPLE_DATA
 - SNOWFLAKE_SANDBOX_DB

The main editor shows the following SQL script:

```

34 WAREHOUSE = COMPUTE_WH
35 SCHEDULE = '1 MINUTE' -- SCHEDULING USING CRON
36 AS
37 COPY INTO PROD_HST_TBL
38 FROM @PROD_STG
39 FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY = '');
40
41 --- CREATING STREAM
42
43 CREATE OR REPLACE STREAM prod_stream
44 ON TABLE PROD_HST_TBL;
45
46 SHOW TASKS;
47
48 --- ALTER TASK
49
50 ALTER TASK PROD_RAW_TASK RESUME;
51
52
53 --- TABLE
54
55 SELECT * FROM PROD_HST_TBL;
56
57 --- STREAMS
58
59 SELECT * FROM PROD_STREAM;
60
61 --- CURATION SCHEMA
  
```

The results pane shows the output of the query `SELECT * FROM PROD_HST_TBL`. The table has 124,746 rows. The columns are: RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE, BASE_PRICE, IS_FEATURED_SKU, IS_DISPLAY_SKU, and UNITS_SOLD. The first 11 rows are displayed:

RECORD_ID	DAY	MONTH	YEAR	STORE_ID	SKU_ID	TOTAL_PRICE	BASE_PRICE	IS_FEATURED_SKU	IS_DISPLAY_SKU	UNITS_SOLD
1	68037	2023	10	31	9672	216425	134.6625	0	0	36
2	68051	2023	10	31	9672	245338	498.0375	498.0375	0	1
3	68048	2023	10	31	9672	223153	178.1250	228.0000	0	0
4	68047	2023	10	31	9672	223245	213.7500	213.7500	0	0
5	68046	2023	10	31	9672	222785	234.4125	234.4125	0	0
6	68045	2023	10	31	9672	222087	226.5750	226.5750	0	0
7	68044	2023	10	31	9672	219029	327.0375	327.0375	0	0
8	68043	2023	10	31	9672	219009	197.3625	228.7125	0	0
9	68039	2023	10	31	9672	217390	169.5750	169.5750	0	0
10	68038	2023	10	31	9672	216233	134.6625	134.6625	0	0
11	68053	2023	10	31	9611	216418	104.7375	104.7375	0	0

--- TASK CREATION

CREATE OR REPLACE TASK RAW.PROD_CURATION_TASK

WAREHOUSE = COMPUTE_WH

WHEN SYSTEM\$STREAM_HAS_DATA('RAW.PROD_STREAM')

AS

MERGE INTO CURATION.PROD_HST_TBL AS target

USING (

```
SELECT RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE,  
BASE_PRICE, UNITS_SOLD
```

```
FROM RAW.prod.stream
```

```
WHERE RECORD_ID IS NOT NULL
```

```
OR DAY IS NOT NULL
```

```
OR MONTH IS NOT NULL
```

```
OR YEAR IS NOT NULL
```

```
OR STORE_ID IS NOT NULL
```

```
OR SKU_ID IS NOT NULL
```

```
OR TOTAL_PRICE IS NOT NULL
```

```
OR BASE_PRICE IS NOT NULL
```

```
OR UNITS_SOLD IS NOT NULL
```

```
) AS source
```

```
ON target.RECORD_ID = source.RECORD_ID
```

```
WHEN MATCHED THEN
```

```
UPDATE SET
```

```
target.DAY = source.DAY,
```

```
target.MONTH = source.MONTH,
```

```
target.YEAR = source.YEAR,
```

```
target.STORE_ID = source.STORE_ID,
```

```
target.SKU_ID = source.SKU_ID,
```

```
target.TOTAL_PRICE = source.TOTAL_PRICE,
```

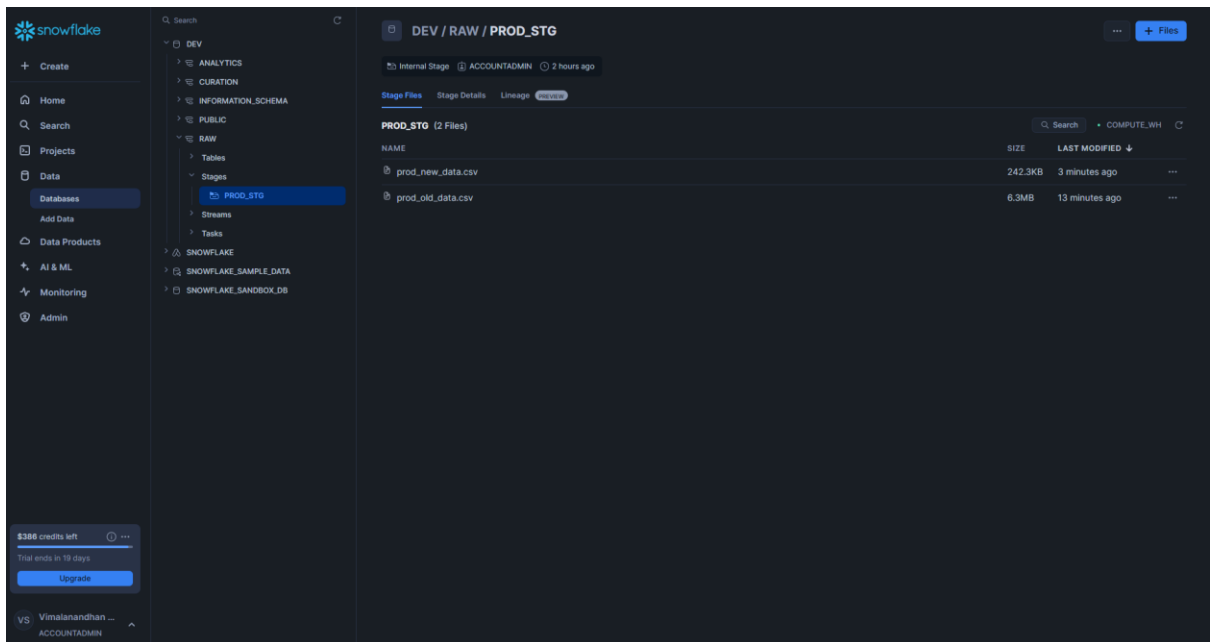
```
target.BASE_PRICE = source.BASE_PRICE,
```

```
target.UNITS_SOLD = source.UNITS_SOLD
```

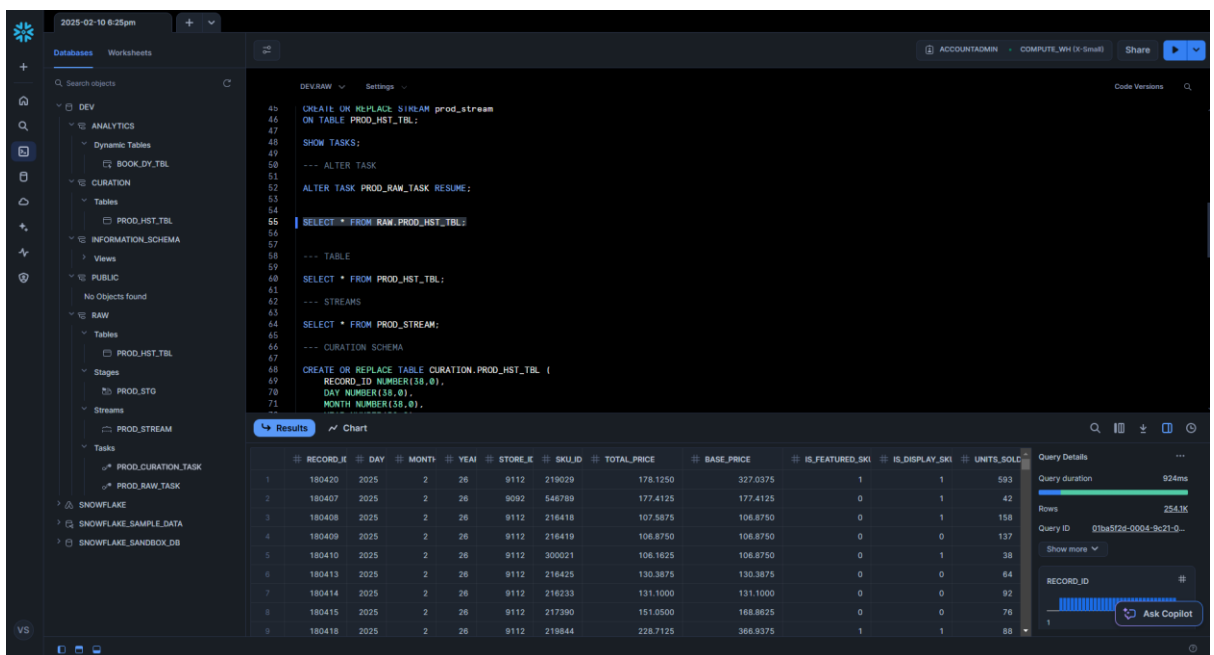
```
WHEN NOT MATCHED THEN
```

```
INSERT (RECORD_ID, DAY, MONTH, YEAR, STORE_ID, SKU_ID, TOTAL_PRICE,  
BASE_PRICE, UNITS_SOLD)
```

```
VALUES (source.RECORD_ID, source.DAY, source.MONTH, source.YEAR,  
source.STORE_ID, source.SKU_ID, source.TOTAL_PRICE, source.BASE_PRICE,  
source.UNITS_SOLD);
```



Uploaded the prod_new_data.csv



5.(+4) Explain end-to-end process based on your understanding.

In order for raw data to be converted into analytics-ready format, Snowflake data pipeline has a documented three-step process: RAW → CURATION → ANALYTICS. Data is first imported from Stages like internal and external Amazon S3 (External Stage) in RAW schema and staged in prod_stg. Data in the stage is loaded in RAW automatically via a Task (prod_raw_task). A stream (prod_stream) on the other hand streams newly added and updated records. All changes to the raw data are efficiently logged and available for further processing due to this setup. New and updated records are added to CURATION by utilizing the prod_stream in the CURATION schema. PROD_HST_TBL ensures data consistency and integrity using a MERGE process. Last but not least, the data that has been curated is updated periodically by a Dynamic Table (book_dy_tbl) in the ANALYTICS schema, resulting in a formatted, real-time data set that can be used for reporting and analysis.