

# DYNAMIC TABLES IN SNOWFLAKE

Data Sheet | LumenData

Snowflake provides a cost-effective, reliable, and automated way to transform our data. Instead of creating a sequential object and monitoring it to transform our data, we can simply define the end objective of the transformation in dynamic tables, then the pipeline management will be managed by Snowflake.

To demonstrate this, let's create two tables and monitor how Change Data Capture is achieved. This can be done in two ways.

- Creating streams and Tasks for Automating Change Data Capture
- Creating Dynamic Tables for Automating Change Data Capture

## Creating streams and Tasks for Automating Change Data Capture

1) Creating a Table named "EMPLOYEE\_DETAILS\_RAW".

```
CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (  
  EMPLOYEE_ID NUMBER(5,0),  
  EMPLOYEE_FIRST_NAME VARCHAR(50),  
  EMPLOYEE_LAST_NAME VARCHAR(50),  
  EMPLOYEE_ADDRESS VARCHAR(50)  
);
```

LD.PUBLIC \* Settings \*

```
1 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (  
2   EMPLOYEE_ID NUMBER(5,0),  
3   EMPLOYEE_FIRST_NAME VARCHAR(50),  
4   EMPLOYEE_LAST_NAME VARCHAR(50),  
5   EMPLOYEE_ADDRESS VARCHAR(50)  
6 );
```

Results Chart

status	
1	Table EMPLOYEE_DETAILS_RAW successfully created.

2) Creating a table named "EMPLOYEE\_DETAILS\_CONFIRM".

```
CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_CONFIRM (  
  EMPLOYEE_ID NUMBER(5,0),  
  EMPLOYEE_FIRST_NAME VARCHAR(50),  
  EMPLOYEE_LAST_NAME VARCHAR(50),  
  EMPLOYEE_ADDRESS VARCHAR(50)  
);
```

LD.PUBLIC \* Settings \*

```
1 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (  
2   EMPLOYEE_ID NUMBER(5,0),  
3   EMPLOYEE_FIRST_NAME VARCHAR(50),  
4   EMPLOYEE_LAST_NAME VARCHAR(50),  
5   EMPLOYEE_ADDRESS VARCHAR(50)  
6 );  
7  
8 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_CONFIRM (  
9   EMPLOYEE_ID NUMBER(5,0),  
10  EMPLOYEE_FIRST_NAME VARCHAR(50),  
11  EMPLOYEE_LAST_NAME VARCHAR(50),  
12  EMPLOYEE_ADDRESS VARCHAR(50)  
13 );
```

Results Chart

status	
1	Table EMPLOYEE_DETAILS_CONFIRM successfully created.

3) Now we try to achieve the Change Data Capture that happened in "EMPLOYEE\_DETAILS\_RAW" using streams and update the "EMPLOYEE\_DETAILS\_CONFIRM".

4) Creating Stream named "EMPLOYEE\_STREAM" on "EMPLOYEE\_DETAILS\_RAW".

```
CREATE OR REPLACE STREAM EMPLOYEE_STREAM ON TABLE  
EMPLOYEE_DETAILS_RAW;
```



```
LD/PUBLIC Settings  
1 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (  
2   EMPLOYEE_ID NUMBER(5,0),  
3   EMPLOYEE_FIRST_NAME VARCHAR(50),  
4   EMPLOYEE_LAST_NAME VARCHAR(50),  
5   EMPLOYEE_ADDRESS VARCHAR(50)  
6 );  
7  
8 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_CONFIRM (  
9   EMPLOYEE_ID NUMBER(5,0),  
10  EMPLOYEE_FIRST_NAME VARCHAR(50),  
11  EMPLOYEE_LAST_NAME VARCHAR(50),  
12  EMPLOYEE_ADDRESS VARCHAR(50)  
13 );  
14  
15 --CREATING STREAM ON EMPLOYEE_DETAILS_RAW  
16 CREATE OR REPLACE STREAM EMPLOYEE_STREAM ON TABLE EMPLOYEE_DETAILS_RAW;  
17  
18
```

Results Chart

status
1 Stream EMPLOYEE_STREAM successfully created.

5) Inserting the records into "EMPLOYEE\_DETAILS\_RAW" and verifying the stream.

```
INSERT INTO EMPLOYEE_DETAILS_RAW(EMPLOYEE_ID,  
EMPLOYEE_FIRST_NAME, EMPLOYEE_LAST_NAME,  
EMPLOYEE_ADDRESS)  
VALUES  
(1,'PRADEEP','KOLLI','USA'),  
(2,'ANKIT','KUMAR','DELHI'),  
(3,'SAI','BHARADWAJA','BANGALORE');  
  
SELECT * FROM EMPLOYEE_STREAM;
```

6) From the below figure, we can see that the Change Data Capture has been captured by Stream "EMPLOYEE\_STREAM".

```
LDPUBLIC * Settings *
4      EMPLOYEE_LAST_NAME VARCHAR(50),
5      EMPLOYEE_ADDRESS VARCHAR(50)
6  };
7
8  CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_CONFIRM (
9      EMPLOYEE_ID NUMBER(5,0),
10     EMPLOYEE_FIRST_NAME VARCHAR(50),
11     EMPLOYEE_LAST_NAME VARCHAR(50),
12     EMPLOYEE_ADDRESS VARCHAR(50)
13 );
14
15 --CREATING STREAM ON EMPLOYEE_DETAILS_RAW
16 CREATE OR REPLACE STREAM EMPLOYEE_STREAM ON TABLE EMPLOYEE_DETAILS_RAW;
17
18 --INSERTING DATA INTO EMPLOYEE_DETAILS_RAW TABLE AND VERIFYING THE IT IN THE STREAM
19
20 INSERT INTO EMPLOYEE_DETAILS_RAW(EMPLOYEE_ID, EMPLOYEE_FIRST_NAME, EMPLOYEE_LAST_NAME, EMPLOYEE_ADDRESS)
21 VALUES
22 ('1','PRADEEP','KOLLI','USA'),
23 ('2','ANKIT','KUMAR','DELHI'),
24 ('3','SAI','BHARADWAJA','BANGALORE');
25
26 | SELECT * FROM EMPLOYEE_STREAM;
27
28
```

Results Chart

	EMPLOYEE_ID	EMPLOYEE_FIRST_NAME	EMPLOYEE_LAST_NAME	EMPLOYEE_ADDRESS	METADATA\$ACTION	METADATA\$ISUPDATE	METADATA\$ROW_ID
1	1	PRADEEP	KOLLI	USA	INSERT	FALSE	8ac750ee48fc3b9570df9daec9e3ac381409adee
2	2	ANKIT	KUMAR	DELHI	INSERT	FALSE	89dc56da4ad799bccd63247250c1c8c4607f78a2
3	3	SAI	BHARADWAJA	BANGALORE	INSERT	FALSE	ee0e4c04d61b940838b733b8103b4f1bbff8acd7

7) Now we create a Task named "EMPLOYEE\_TASK" to automate the Change Data Capture (CDC) captured by the stream and update the "EMPLOYEE\_DETAILS\_CONFIRM" on a scheduled basis.

```
CREATE OR REPLACE TASK EMPLOYEE_TASK
```

```
WAREHOUSE = COMPUTE_WH
```

```
SCHEDULE = '1 minute'
```

```
WHEN
```

```
SYSTEM$STREAM_HAS_DATA('EMPLOYEE_STREAM')
```

```
AS
```

```
MERGE INTO EMPLOYEE_DETAILS_CONFIRM a
```

```
USING (
```

```
SELECT * FROM EMPLOYEE_STREAM
```

```
) b ON a.EMPLOYEE_ID = b.EMPLOYEE_ID
```

```
WHEN MATCHED AND b.METADATA$ACTION = 'DELETE' AND  
b.METADATA$ISUPDATE = 'FALSE' THEN DELETE
```

```
WHEN MATCHED AND b.METADATA$ACTION = 'INSERT' AND  
b.METADATA$ISUPDATE = 'TRUE' THEN
```

```
UPDATE SET a.EMPLOYEE_FIRST_NAME =  
b.EMPLOYEE_FIRST_NAME, a.EMPLOYEE_LAST_NAME =  
b.EMPLOYEE_LAST_NAME, a.EMPLOYEE_ADDRESS =  
b.EMPLOYEE_ADDRESS
```

```
WHEN NOT MATCHED AND b.METADATA$ACTION = 'INSERT' THEN  
INSERT (EMPLOYEE_ID, EMPLOYEE_FIRST_NAME,  
EMPLOYEE_LAST_NAME, EMPLOYEE_ADDRESS)  
VALUES (b.EMPLOYEE_ID, b.EMPLOYEE_FIRST_NAME,  
b.EMPLOYEE_LAST_NAME, b.EMPLOYEE_ADDRESS);
```

```
ALTER TASK EMPLOYEE_TASK RESUME;
```



The screenshot shows a SQL script in a text editor with line numbers 26 to 49. The script includes a SELECT statement, a comment, and a MERGE INTO statement with various WHEN clauses for DELETE, UPDATE, and INSERT operations. Below the script, the 'Results' tab is active, showing a single row with the status 'Task EMPLOYEE\_TASK successfully created.'

```
LD.PUBLIC - Settings -  
26 SELECT * FROM EMPLOYEE_STREAM;  
27  
28  
29 --Creating the task to merge data from raw to confirm table based on stream  
30  
31 CREATE OR REPLACE TASK EMPLOYEE_TASK  
32 WAREHOUSE = COMPUTE_WH  
33 SCHEDULE = '1 minute'  
34 WHEN  
35 SYSTEMSSTREAM_HAS_DATA('EMPLOYEE_STREAM')  
36 AS  
37 MERGE INTO EMPLOYEE_DETAILS_CONFIRM a  
38 USING (  
39 SELECT * FROM EMPLOYEE_STREAM  
40 ) b ON a.EMPLOYEE_ID = b.EMPLOYEE_ID  
41 WHEN MATCHED AND b.METADATA$ACTION = 'DELETE' AND b.METADATA$ISUPDATE = 'FALSE' THEN DELETE  
42 WHEN MATCHED AND b.METADATA$ACTION = 'INSERT' AND b.METADATA$ISUPDATE = 'TRUE' THEN  
43 UPDATE SET a.EMPLOYEE_FIRST_NAME = b.EMPLOYEE_FIRST_NAME, a.EMPLOYEE_LAST_NAME = b.EMPLOYEE_LAST_NAME, a.EMPLOYEE_ADDRESS = b.EMPLOYEE_ADDRESS  
44 WHEN NOT MATCHED AND b.METADATA$ACTION = 'INSERT' THEN  
45 INSERT (EMPLOYEE_ID, EMPLOYEE_FIRST_NAME, EMPLOYEE_LAST_NAME, EMPLOYEE_ADDRESS)  
46 VALUES (b.EMPLOYEE_ID, b.EMPLOYEE_FIRST_NAME, b.EMPLOYEE_LAST_NAME, b.EMPLOYEE_ADDRESS);  
47  
48 ALTER TASK EMPLOYEE_TASK RESUME;  
49
```

Results | Chart

status
1 Task EMPLOYEE_TASK successfully created.

8) Once the task was completed, we can see the "EMPLOYEE\_DETAILS\_CONFIRM" has the data init.

```

LD.PUBLIC * Settings *
30
31 CREATE OR REPLACE TASK EMPLOYEE_TASK
32 WAREHOUSE = COMPUTE_WH
33 SCHEDULE = '1 minute'
34 WHEN
35 SYSTEMSTREAM_HAS_DATA('EMPLOYEE_STREAM')
36 AS
37 MERGE INTO EMPLOYEE_DETAILS_CONFIRM a
38 USING (
39     SELECT * FROM EMPLOYEE_STREAM
40     ) b ON a.EMPLOYEE_ID = b.EMPLOYEE_ID
41 WHEN MATCHED AND b.METADATASACTION = 'DELETE' AND b.METADATASISUPDATE = 'FALSE' THEN DELETE
42 WHEN MATCHED AND b.METADATASACTION = 'INSERT' AND b.METADATASISUPDATE = 'TRUE' THEN
43     UPDATE SET a.EMPLOYEE_FIRST_NAME = b.EMPLOYEE_FIRST_NAME, a.EMPLOYEE_LAST_NAME = b.EMPLOYEE_LAST_NAME, a.EMPLOYEE_ADDRESS = b.EMPLOYEE_ADDRESS
44 WHEN NOT MATCHED AND b.METADATASACTION = 'INSERT' THEN
45     INSERT (EMPLOYEE_ID, EMPLOYEE_FIRST_NAME, EMPLOYEE_LAST_NAME, EMPLOYEE_ADDRESS)
46     VALUES (b.EMPLOYEE_ID, b.EMPLOYEE_FIRST_NAME, b.EMPLOYEE_LAST_NAME, b.EMPLOYEE_ADDRESS);
47
48 ALTER TASK EMPLOYEE_TASK RESUME;
49
50
51
52
53 | SELECT * FROM EMPLOYEE_DETAILS_CONFIRM;

```

Results

Chart

	EMPLOYEE_ID	EMPLOYEE_FIRST_NAME	EMPLOYEE_LAST_NAME	EMPLOYEE_ADDRESS
1	1	PRADEEP	KOLLI	USA
2	2	ANKIT	KUMAR	DELHI
3	3	SAI	BHARADWAJA	BANGALORE

9) Now let's perform some delete operation on "EMPLOYEE\_DETAILS\_RAW" and verify the "EMPLOYEE\_DETAILS\_CONFIRM"

delete from employee\_details\_raw where employee\_id ='2';  
select \* from employee\_stream;  
select \* from employee\_details\_confirm;

```

LD.PUBLIC * Settings *
36 AS
37 MERGE INTO EMPLOYEE_DETAILS_CONFIRM a
38 USING (
39     SELECT * FROM EMPLOYEE_STREAM
40     ) b ON a.EMPLOYEE_ID = b.EMPLOYEE_ID
41 WHEN MATCHED AND b.METADATASACTION = 'DELETE' AND b.METADATASISUPDATE = 'FALSE' THEN DELETE
42 WHEN MATCHED AND b.METADATASACTION = 'INSERT' AND b.METADATASISUPDATE = 'TRUE' THEN
43     UPDATE SET a.EMPLOYEE_FIRST_NAME = b.EMPLOYEE_FIRST_NAME, a.EMPLOYEE_LAST_NAME = b.EMPLOYEE_LAST_NAME, a.EMPLOYEE_ADDRESS = b.EMPLOYEE_ADDRESS
44 WHEN NOT MATCHED AND b.METADATASACTION = 'INSERT' THEN
45     INSERT (EMPLOYEE_ID, EMPLOYEE_FIRST_NAME, EMPLOYEE_LAST_NAME, EMPLOYEE_ADDRESS)
46     VALUES (b.EMPLOYEE_ID, b.EMPLOYEE_FIRST_NAME, b.EMPLOYEE_LAST_NAME, b.EMPLOYEE_ADDRESS);
47
48 ALTER TASK EMPLOYEE_TASK RESUME;
49
50
51
52
53 | SELECT * FROM EMPLOYEE_DETAILS_CONFIRM;
54
55 --deleting the records from table
56
57 delete from employee_details_raw where employee_id ='2';
58
59 select * from employee_stream;

```

Results

Chart

	EMPLOYEE_ID	EMPLOYEE_FIRST_NAME	EMPLOYEE_LAST_NAME	EMPLOYEE_ADDRESS
1	1	PRADEEP	KOLLI	USA
2	3	SAI	BHARADWAJA	BANGALORE

# Creating Dynamic Tables for Automating Change Data Capture

10) Creating a Table named "EMPLOYEE\_DETAILS\_RAW" and view the data in the table.

```
CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (  
  EMPLOYEE_ID NUMBER(5,0),  
  EMPLOYEE_FIRST_NAME VARCHAR(50),  
  EMPLOYEE_LAST_NAME VARCHAR(50),  
  EMPLOYEE_ADDRESS VARCHAR(50)  
);  
SELECT * FROM EMPLOYEE_DETAILS_RAW;
```

The screenshot shows a database query editor with the following SQL code:

```
LD.PUBLIC * Settings *  
47  
48  
49 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (  
50   EMPLOYEE_ID NUMBER(5,0),  
51   EMPLOYEE_FIRST_NAME VARCHAR(50),  
52   EMPLOYEE_LAST_NAME VARCHAR(50),  
53   EMPLOYEE_ADDRESS VARCHAR(50)  
54 );  
55  
56  
57 SELECT * FROM EMPLOYEE_DETAILS_RAW;  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83
```

Below the editor, there are two tabs: "Results" and "Chart". The "Results" tab is active and shows a single message: "Task EMPLOYEE\_TASK successfully created." The "Chart" tab is inactive.

The screenshot shows the same database query editor as above, but with the "Results" tab active. The SQL code is identical. The results pane shows a table with the following columns: EMPLOYEE\_ID, EMPLOYEE\_FIRST\_NAME, EMPLOYEE\_LAST\_NAME, and EMPLOYEE\_ADDRESS. The table is empty, and the message "Query produced no results" is displayed below the table.

EMPLOYEE_ID	EMPLOYEE_FIRST_NAME	EMPLOYEE_LAST_NAME	EMPLOYEE_ADDRESS
-------------	---------------------	--------------------	------------------

Query produced no results

11) Creating a Dynamic table on a schedule basis to capture the Change Data Capture and update it.

```
CREATE OR REPLACE DYNAMIC TABLE  
EMPLOYEE_DETAILS_CONFIRM  
TARGET_LAG = '1 minute'  
WAREHOUSE = COMPUTE_WH  
AS  
SELECT * FROM EMPLOYEE_DETAILS_RAW;
```



```
LD.PUBLIC * Settings *  
59  
60 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (  
61     EMPLOYEE_ID NUMBER(5,0),  
62     EMPLOYEE_FIRST_NAME VARCHAR(50),  
63     EMPLOYEE_LAST_NAME VARCHAR(50),  
64     EMPLOYEE_ADDRESS VARCHAR(50)  
65 );  
66  
67 SELECT * FROM EMPLOYEE_DETAILS_RAW;  
68  
69  
70 CREATE OR REPLACE DYNAMIC TABLE EMPLOYEE_DETAILS_CONFIRM  
71 TARGET_LAG = '1 minute'  
72 WAREHOUSE = COMPUTE_WH  
73 AS  
74 SELECT * FROM EMPLOYEE_DETAILS_RAW ;  
75  
76  
77  
78  
79  
80  
81  
82  
83
```

Results Chart

status
1 Dynamic table EMPLOYEE_DETAILS_CONFIRM successfully created.

12) Now let's insert the records in "EMPLOYEE\_DETAILS\_RAW" and verify the Dynamic table "EMPLOYEE\_DETAILS\_CONFIRM" Table.

```
INSERT INTO EMPLOYEE_DETAILS_RAW(EMPLOYEE_ID,  
EMPLOYEE_FIRST_NAME, EMPLOYEE_LAST_NAME,  
EMPLOYEE_ADDRESS)  
VALUES  
(1,'PRADEEP','KOLLI','USA'),  
(2,'ANKIT','KUMAR','DELHI'),  
(3,'SAI','BHARADWAJA','BANGALORE');  
  
SELECT * FROM EMPLOYEE_DETAILS_CONFIRM;
```



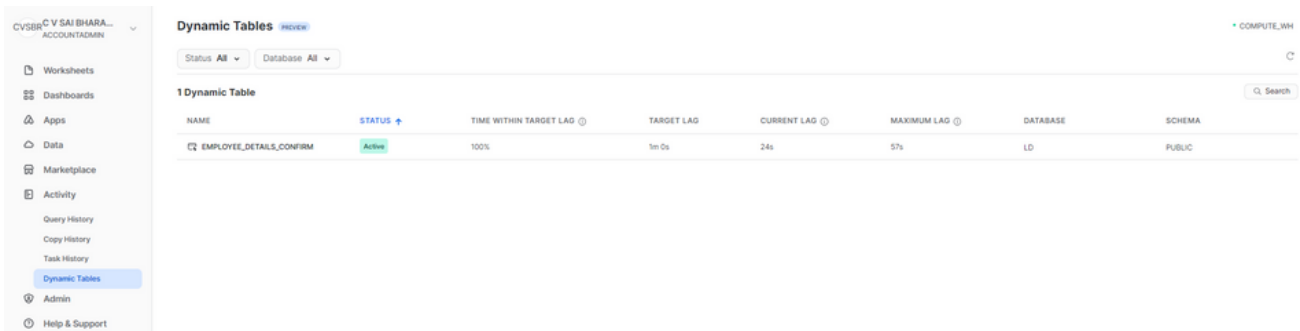
```

LD.PUBLIC * Settings *
59
60 CREATE OR REPLACE TABLE EMPLOYEE_DETAILS_RAW (
61     EMPLOYEE_ID NUMBER(5,0),
62     EMPLOYEE_FIRST_NAME VARCHAR(50),
63     EMPLOYEE_LAST_NAME VARCHAR(50),
64     EMPLOYEE_ADDRESS VARCHAR(50)
65 );
66
67 SELECT * FROM EMPLOYEE_DETAILS_RAW;
68
69 CREATE OR REPLACE DYNAMIC TABLE EMPLOYEE_DETAILS_CONFIRM
70     TARGET_LAG = '1 minute'
71     WAREHOUSE = COMPUTE_WH
72 AS
73     SELECT * FROM EMPLOYEE_DETAILS_RAW ;
74
75
76 INSERT INTO EMPLOYEE_DETAILS_RAW(EMPLOYEE_ID, EMPLOYEE_FIRST_NAME, EMPLOYEE_LAST_NAME, EMPLOYEE_ADDRESS)
77 VALUES
78     ('1', 'PRADEEP', 'KOLLI', 'USA'),
79     ('2', 'ANKIT', 'KUMAR', 'DELHI'),
80     ('3', 'SAI', 'BHARADWAJA', 'BANGALORE');
81
82 | SELECT * FROM EMPLOYEE_DETAILS_CONFIRM;
83

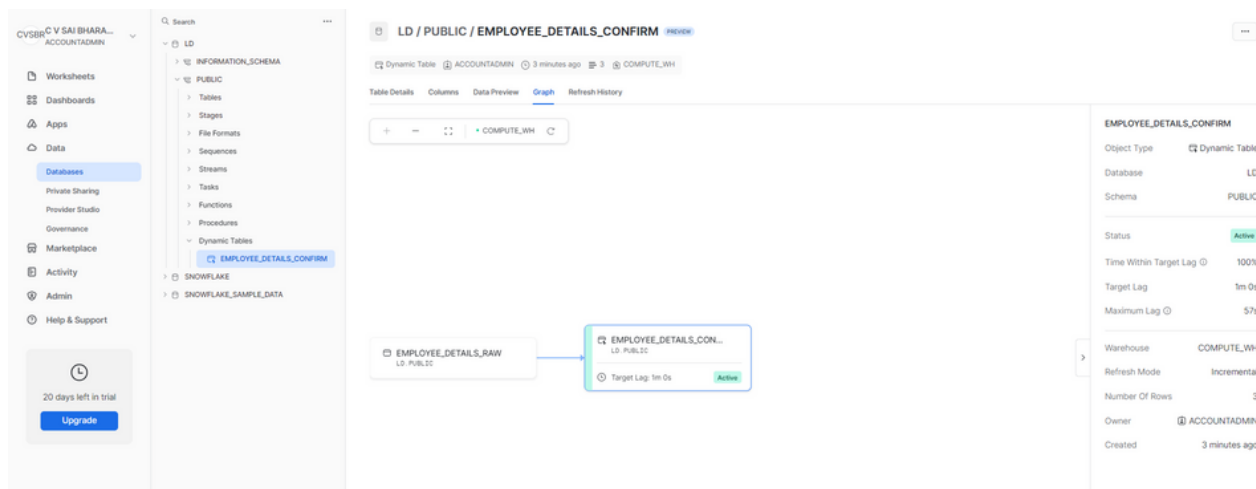
```

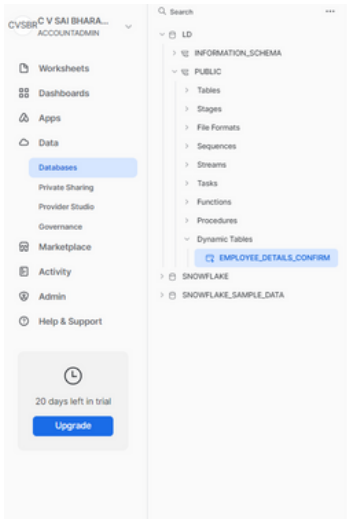
	EMPLOYEE_ID	EMPLOYEE_FIRST_NAME	EMPLOYEE_LAST_NAME	EMPLOYEE_ADDRESS
1	1	PRADEEP	KOLLI	USA
2	2	ANKIT	KUMAR	DELHI
3	3	SAI	BHARADWAJA	BANGALORE

13) We can view the 'Dynamic Table History' under 'Activity Tab' as shown below from Snowflake UI.



14) When we click the Dynamic Table, we see the Graph of it and the refresh history as shown in below figures.





LD / PUBLIC / EMPLOYEE\_DETAILS\_CONFIRM [Refresh](#)

Dynamic Table ACCOUNTADMIN 3 minutes ago 3 COMPUTE\_WH

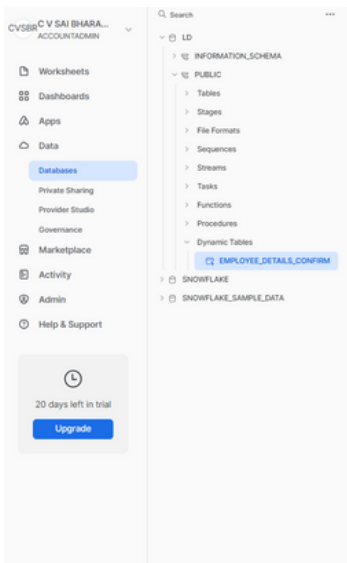
Table Details Columns Data Preview Graph [Refresh History](#)

100% Time Within Target Lag 1m 0s Target Lag 57s Current Lag 57s Maximum Lag

5 Refreshes (Aug 30, 2023, 10 AM - Aug 31, 2023, 10 AM) COMPUTE\_WH

SOURCE DATA TIMESTAMP	STATUS	REFRESH DURATION	REFRESH LAG	ROWS CHANGED	QUERY PROFILE
Aug 31, 2023, 9:18:00 AM	Scheduled	---	---	---	<a href="#">🔗</a>
Aug 31, 2023, 9:17:12 AM	Succeeded	290ms	53s	+0 -0	<a href="#">🔗</a>
Aug 31, 2023, 9:16:24 AM	Succeeded	169ms	58s	+0 -0	<a href="#">🔗</a>
Aug 31, 2023, 9:15:36 AM	Succeeded	584ms	56s	+3 -0	<a href="#">🔗</a>
Aug 31, 2023, 9:14:48 AM	Succeeded	453ms	---	+0 -0	<a href="#">🔗</a>

15) We can see the number of records that got changed or deleted from the refresh history based on the job scheduled as shown in the below figure.



LD / PUBLIC / EMPLOYEE\_DETAILS\_CONFIRM [Refresh](#)

Dynamic Table ACCOUNTADMIN 3 minutes ago 3 COMPUTE\_WH

Table Details Columns Data Preview Graph [Refresh History](#)

100% Time Within Target Lag 1m 0s Target Lag 57s Current Lag 57s Maximum Lag

5 Refreshes (Aug 30, 2023, 10 AM - Aug 31, 2023, 10 AM) COMPUTE\_WH

SOURCE DATA TIMESTAMP	STATUS	REFRESH DURATION	REFRESH LAG	ROWS CHANGED	QUERY PROFILE
Aug 31, 2023, 9:18:00 AM	Scheduled	---	---	---	<a href="#">🔗</a>
Aug 31, 2023, 9:17:12 AM	Succeeded	290ms	53s	+0 -0	<a href="#">🔗</a>
Aug 31, 2023, 9:16:24 AM	Succeeded	169ms	58s	+0 -0	<a href="#">🔗</a>
Aug 31, 2023, 9:15:36 AM	Succeeded	584ms	56s	+3 -0	<a href="#">🔗</a>
Aug 31, 2023, 9:14:48 AM	Succeeded	453ms	---	+0 -0	<a href="#">🔗</a>

We can conclude that dynamic tables update the results of the query without creating separate objects, writing the code, scheduling tasks, or monitoring them.

## ABOUT LUMENDATA:

LumenData is a leading provider of Enterprise Data Management, Cloud & Analytics solutions. We help businesses navigate their data visualization and analytics anxieties and enable them to accelerate their innovation journeys. Founded in 2008, with locations in multiple countries, LumenData is privileged to serve over 100 leading companies, including KwikTrip, Versant Health, US Food & Drug Administration, US Department of Labor, Cummins Engine, BCG, and others. LumenData is SOC2 certified and has instituted extensive controls to protect client data, including adherence to GDPR and CCPA regulations.

Get in touch to discuss how we can facilitate data-driven transformation for your organization.

---

## MEET OUR AUTHORS

---



**Sai Bharadwaja Reddy**  
Consultant



**Ankit Kumar**  
Technical Lead