**Lab Exercise 12– Incremental Materialization in dbt Cloud for Snowflake**

This lab exercise will walk you through how to implement **incremental materialization** in dbt for Snowflake using dbt Cloud. Incremental models in dbt are used when you want to update only the new or modified records in a table rather than reloading the entire table. This is particularly useful for large datasets where you want to minimize the processing time.

**Objective**

* Learn how to set up an incremental model in dbt for Snowflake.
* Use dbt Cloud to implement incremental materialization.
* Efficiently update the target table with new or changed data without full reprocessing.

**Prerequisites:**

1. You have access to dbt Cloud.
2. You have a Snowflake account with an active connection to dbt Cloud.
3. You have a dbt project set up in dbt Cloud.

**Steps to Implement Incremental Materialization in dbt Cloud**

**Step 1: Create or Set Up a Snowflake Table**

For this exercise, we will assume that you have an existing **orders** table in Snowflake that you want to update incrementally. If you don't have this table, you can create one using the following SQL:

CREATE OR REPLACE TABLE orders (

order\_id INT,

order\_date DATE,

customer\_id INT,

total\_amount DECIMAL(10, 2),

updated\_at TIMESTAMP

);

Now, populate the orders table with some sample data (you can modify this as needed):

INSERT INTO raw\_db.raw\_data.orders (order\_id, order\_date, customer\_id, total\_amount,updated\_at)

VALUES

(1, '2024-01-01', 101, 100.50,current\_timestamp()),

(2, '2024-01-02', 102, 150.75,current\_timestamp()),

(3, '2024-01-03', 103, 200.25,current\_timestamp());

**Step 2: Configure dbt Project for Incremental Model**

1. **Set up your dbt project** and configure it to use Snowflake as your data warehouse. If you haven't already set up a dbt project, you can follow the setup guide on dbt Cloud.
2. In your **dbt project**, go to the **models** directory, and create a new SQL file named incremental\_orders.sql:

-- models/incremental\_orders.sql

{{ config(

materialized='incremental',

unique\_key='order\_id'

) }}

WITH new\_orders AS (

SELECT

order\_id,

order\_date,

customer\_id,

total\_amount,

updated\_at AS updated\_time, -- Adding updated\_at field from the source table as updated\_time

current\_timestamp() AS dbt\_updated\_time -- Adding dbt\_updated\_time as the current timestamp

FROM {{ source('raw', 'orders') }} -- Referencing the source table

{% if is\_incremental() %}

-- Only fetch records that are new or changed (i.e., orders after the last update)

WHERE order\_date > (SELECT MAX(order\_date) FROM {{ this }})

{% endif %}

)

SELECT \*

FROM new\_orders;

**Explanation of the SQL Code:**

* **materialized='incremental'**: This specifies that the model should be materialized as an incremental model.
* **unique\_key='order\_id'**: This is the unique key that dbt will use to identify new or updated records. It's important that this key uniquely identifies a record in the table.
* **is\_incremental()**: This function checks if the model is being run incrementally or as a full refresh. During an incremental run, only new or updated records are processed.
* **WHERE order\_date > (SELECT MAX(order\_date) FROM {{ this }})**: This part ensures that only records with an order\_date greater than the maximum order\_date already in the target table are selected.

**Step 3: Add a Source in dbt**

**Step 4: Run the dbt Model**

Once you have the incremental\_orders.sql model and sources.yml defined, you're ready to run your model.

1. **Test your dbt project**:

In the dbt Cloud interface, run the following:

dbt run --models incremental\_orders

1. On the first run, dbt will create the target table (incremental\_orders) and load all records from the source orders table.

**Step 5: Update the Model with New Records (Incremental Run)**

Once the model has been run, dbt will track the state of the target table and only insert new records during subsequent runs.

To add new records, insert new data into the orders table in Snowflake:

INSERT INTO raw\_db.raw\_data.orders (order\_id, order\_date, customer\_id, total\_amount,updated\_at)

VALUES

(4, '2024-01-05', 101, 100.50,current\_timestamp()),

(5, '2024-01-06', 102, 150.75,current\_timestamp()),

(6, '2024-01-07', 103, 200.25,current\_timestamp());

Now, run the incremental model again using dbt:

dbt run --models incremental\_orders

Only the new records with order\_id 4 and 5 will be inserted into the incremental\_orders model.

**Step 6: Check the Results**

To confirm the success of the incremental load, query the incremental\_orders table in Snowflake:

SELECT \* FROM your\_database.your\_schema.incremental\_orders;

You should see the new records (order\_id 4 and 5) added to the table.

To load full data run following command

dbt run --models incremental\_orders --full-refresh

SELECT \* FROM your\_database.your\_schema.incremental\_orders;

**Summary**

* **Incremental Materialization** allows you to update only new or modified records, improving performance for large datasets.
* **is\_incremental()** function is key to controlling what data is loaded.
* Use **unique\_key** to specify the field(s) that identify unique records in the model.
* When running dbt, only new records that match the criteria will be inserted or updated in the target table.

This lab exercise helps you set up an incremental model in dbt Cloud for Snowflake, enabling efficient data processing and keeping your models up to date with minimal resource consumption.