Micro Partition in snowflake
==========

1. Creating the database and table.
2. Inserting a large volume of data to demonstrate micro-partitioning.
3. Sample queries to show how micro partitions work.
4. Exercise queries

Step 1: Create the Database and Schema

```
CREATE OR REPLACE DATABASE micro_partition_demo;
USE DATABASE micro_partition_demo;

CREATE OR REPLACE SCHEMA demo_schema;
USE SCHEMA demo_schema;
```

Step 2: Create a Large Table

We'll simulate a sales dataset with various fields.

```
CREATE OR REPLACE TABLE sales_data (
    id BIGINT,
    sale_date DATE,
    region STRING,
    product_category STRING,
    sales_amount FLOAT
);
```

Step 3: Insert Large Amount of Data

Use a Snowflake generator function (`SEQUENCE`, `RANDSTR`, `RANDOM`, `UNIFORM`) with `INSERT` + `SELECT` to generate synthetic data.

```
INSERT INTO sales_data
SELECT
    SEQ4() AS id,
    DATEADD(DAY, UNIFORM(0, 3650, RANDOM()), DATE '2015-01-01') AS sale_date,
    CASE UNIFORM(1, 5, RANDOM())
        WHEN 1 THEN 'North'
        WHEN 2 THEN 'South'
        WHEN 3 THEN 'East'
        WHEN 4 THEN 'West'
        ELSE 'Central'
    END AS region,
    CASE UNIFORM(1, 4, RANDOM())
        WHEN 1 THEN 'Electronics'
        WHEN 2 THEN 'Clothing'
        WHEN 3 THEN 'Groceries'
        ELSE 'Home Decor'
    END AS product_category,
    UNIFORM(10, 1000, RANDOM())::FLOAT AS sales_amount
FROM TABLE(GENERATOR(ROWCOUNT => 5000000));
```

This generates 5 million rows with varied `sale_date`, `region`, and `product_category`, which helps in demonstrating micro-partitions and clustering

behavior.

Step 4: Observe Micro Partitions

You can view the partition structure like this:

```
SELECT SYSTEM$CLUSTERING_INFORMATION('sales_data');
```

Sample Queries to Show Micro Partition Behavior

```
-- Query 1: Full table scan
SELECT COUNT(*) FROM sales_data;

-- Query 2: Filter by sale_date (triggers partition pruning if clustered)
SELECT * FROM sales_data
WHERE sale_date BETWEEN '2020-01-01' AND '2020-12-31';

-- Query 3: Filter by region
SELECT region, COUNT(*)
FROM sales_data
WHERE region = 'North'
GROUP BY region;
```

Exercise Queries

1. Partition Pruning Test

```
-- How many records in the year 2019?
SELECT COUNT(*) FROM sales_data
WHERE sale_date BETWEEN '2019-01-01' AND '2019-12-31';
```

2. Clustering Evaluation

```
-- Add a clustering key and test pruning
ALTER TABLE sales_data CLUSTER BY (sale_date);

-- Run the same date query again and compare performance.
```

3. Storage and Partition View

```
-- Explore how many micro-partitions your table has
SELECT SYSTEM$CLUSTERING_INFORMATION('sales_data');
```

Summary:

- Micro Partitions are automatically created in Snowflake (~16MB compressed).
- They store metadata (min/max values, null count) for pruning.
- Partition pruning = faster queries.

- You can improve pruning with clustering keys.
- Use `SYSTEM$CLUSTERING_INFORMATION` to monitor clustering quality.

In Snowflake, if you want to inspect micro-partitions, the correct approach is to use `SYSTEM$CLUSTERING_INFORMATION` or `SYSTEM$CLUSTERING_DEPTH`, and combine it with metadata queries when needed.

Correct Way to Inspect Micro-Partition Info

1. Check Clustering Metadata

```
SELECT SYSTEM$CLUSTERING_INFORMATION('sales_data');
```

> This shows how well clustered your data is, especially if you've defined a clustering key.

2. Clustering Depth (Number of Micro Partitions per Key)

```
SELECT SYSTEM$CLUSTERING_DEPTH('sales_data');
```

3. Storage & Table Size Stats
If you want to view storage-level information (like micro-partition count indirectly):

```
SELECT *
FROM INFORMATION_SCHEMA.TABLE_STORAGE_METRICS
WHERE TABLE_NAME = 'SALES_DATA';
```

Add Clustering Key (To Demonstrate Pruning)

```
ALTER TABLE sales_data CLUSTER BY (sale_date);
```

After that, re-run the `SYSTEM$CLUSTERING_INFORMATION` function to see the pruning potential.