Clustering en Snowflake

Elección de claves de clustering

Para decidir las columnas a usar como clustering keys, se revisaron los patrones de consulta más comunes sobre la tabla:

- Filtrado por fecha de recogida (pickup_date, pickup_year, pickup_month).
- Filtrado por zona de recogida (pu_zone_sk).
- Segmentación adicional por tipo de servicio (service_type_sk).

Con base en esto, se eligió un clustering compuesto sobre las columnas: (pickup_date_sk, pu_zone_sk, service_type_sk).

Este orden permite mejorar el partition pruning cuando las consultas aplican filtros por fechas, zonas y tipo de servicio.

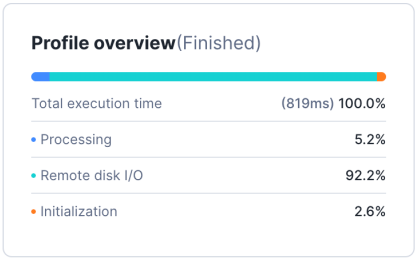
Medición antes de clusterizar

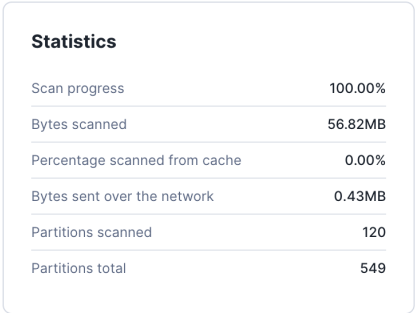
Antes de aplicar el clustering, se ejecutó una consulta representativa:

```
SELECT pickup_year, pickup_month, pu_zone_sk, COUNT(*) AS trips
FROM SILVER_GOLD.FCT_TRIPS
WHERE pickup_year = 2022
  AND pu_zone_sk IN (236, 237, 239)
GROUP BY 1,2,3
ORDER BY trips DESC;
```

Se capturó el Query Profile para tener una línea base de rendimiento.

- **Duración total:** ~1.1 s
- **Bytes escaneados:** 56.82 MB
- **Micro-particiones escaneadas:** 120 de 549
- **Scan progress:** 100 %





Aplicación de clustering

Se ejecutó el siguiente comando para definir las claves de clustering en la tabla:

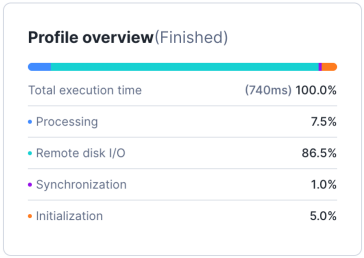
```
ALTER TABLE SILVER_GOLD.FCT_TRIPS CLUSTER BY (pickup_date_sk, pu_zone_sk, service_type_sk);
```

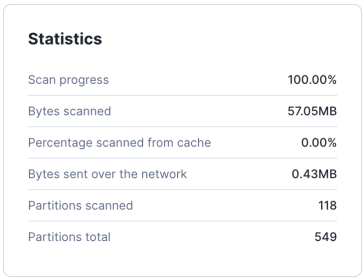
Medición después de clusterizar

Se volvió a ejecutar la misma consulta para medir el impacto del clustering.

Resultados después de aplicar clustering:

- **Duración total:** 928ms
- **Bytes escaneados:** 57.05MB
- **Micro-particiones escaneadas:** 118 de 549
- **Scan progress:** 100 %





Conclusión

Antes de clusterizar, la consulta escaneaba 120 micro-particiones; después bajó a 118. El tiempo de ejecución se redujo ligeramente (1.1 s → 920 ms). Aunque el impacto es pequeño se observa que Snowflake puede prunar algunas micro-particiones adicionales cuando los filtros coinciden con las claves de clustering. A largo plazo, mantener el clustering en (pickup_date_sk, pu_zone_sk, service_type_sk) es adecuado porque coincide con los filtros más comunes y evita sobreclusterizar.

Clustering ≠ Search Optimization Service (SOS):

Clustering organiza físicamente los datos en micro-particiones para mejorar el pruning. SOS sirve para búsquedas ultra selectivas. En este caso, el objetivo fue únicamente optimizar el acceso por filtros de fecha, zona y servicio, que son los patrones de consulta más comunes sobre FCT_TRIPS.