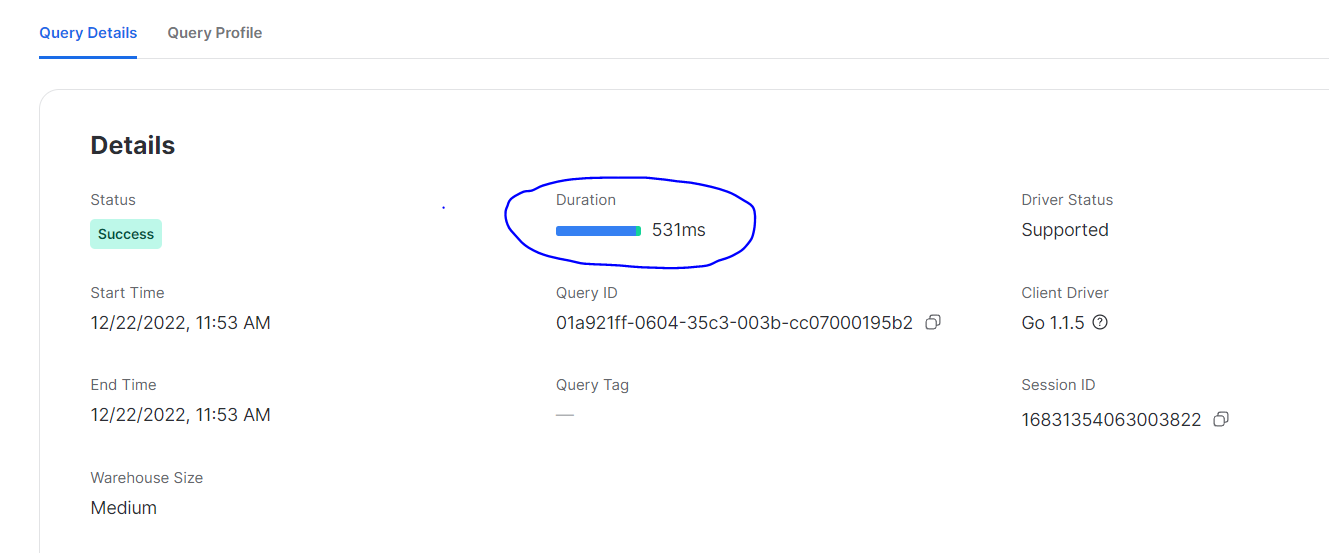
To help improve performance, you can store certain aggregated results in materialized views (MV).

**Before creation of MV:**

select customer\_city as city, count(\*) as number\_of\_orders

from ecommerce\_database.public.orders

group by customer\_city;



**Creation of MV:**

CREATE OR REPLACE MATERIALIZED VIEW ORDERS\_BY\_CITY

AS

select customer\_city as city, count(\*) as number\_of\_orders

from ecommerce\_database.public.orders

group by customer\_city;

-- disable global caching

-- Snowflake automatically caches results. To be able to see performance improvement based on MV -- alone instead of caching, we are disabling caching.

ALTER SESSION SET USE\_CACHED\_RESULT=FALSE;

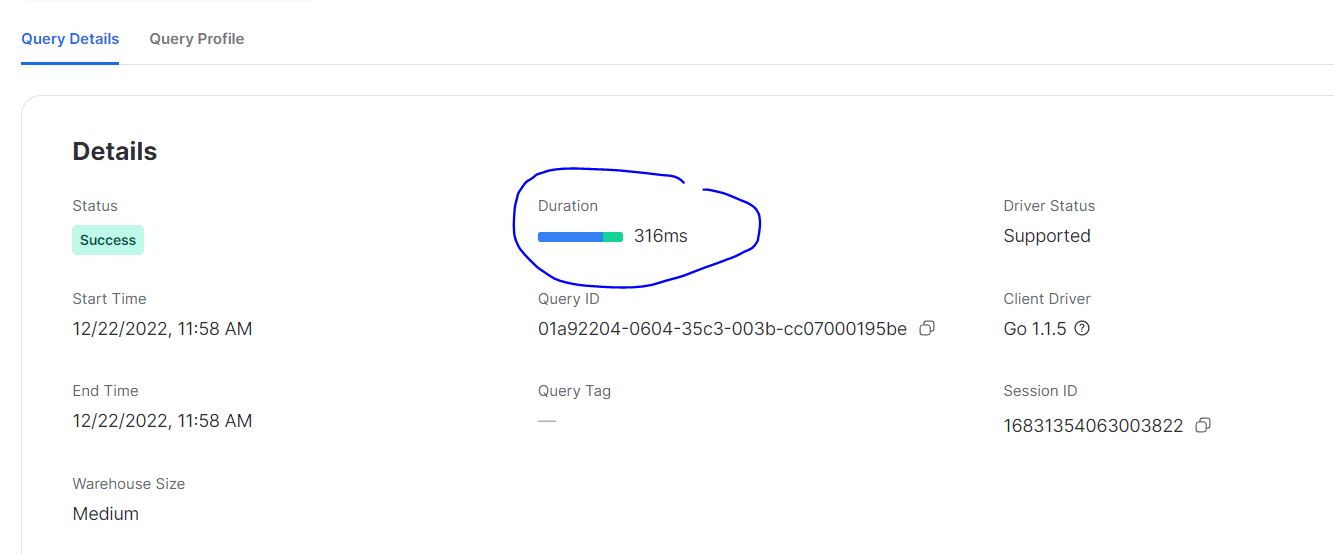
**After creation of MV:**

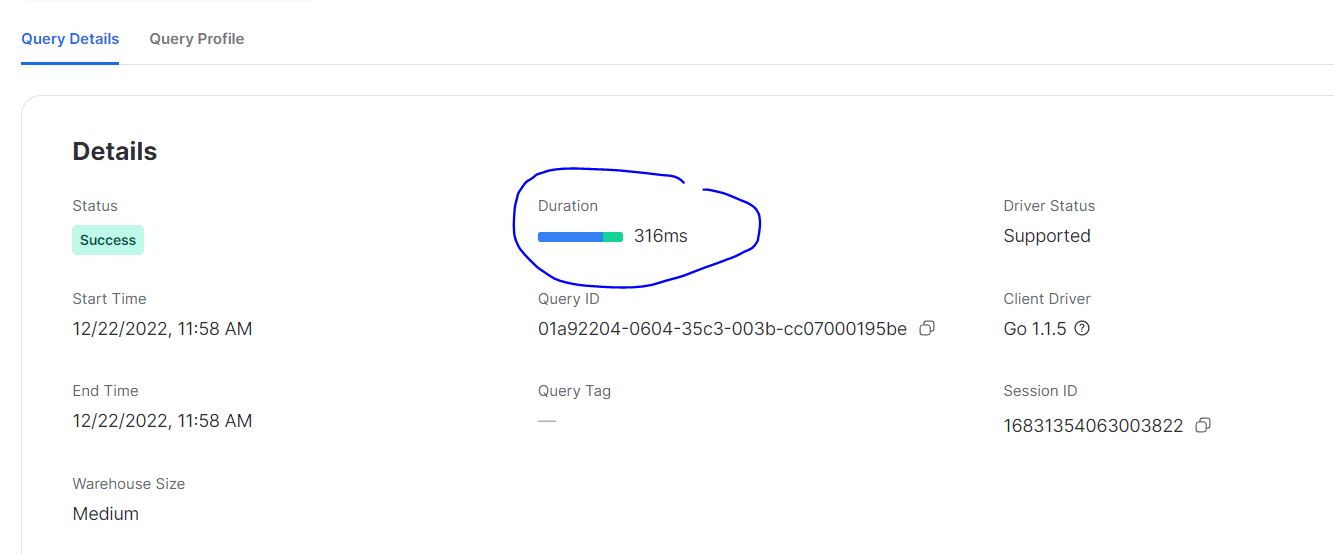
// You can see that the performance has improved.

// This is a small use-case (small table / simple query).

// The improvement will be much more visible for huge tables / complex queries.

select \* from ORDERS\_BY\_CITY;





**Use Cases for MV:**

The maintenance of MVs is taken care of by Snowflake itself but you pay for it.

Maintenance is making sure when the underlying data is changed, the MV reflects that change.

Keeping the cost in mind, the ideal use case for MVs is a table/view/dataset that changes rarely but queried often.

Otherwise, using Tasks and Streams might be a better way to handle the situation.

You have atleast better control over when the changes must be captured (all changes might not have to be reflected immediately).

Also, there are some important limitations for MVs.

MVs do not support joins, including self-joins.

Not all aggregation functions are supported.

Also, MVs do not support UDFs, having by clause, order by clause, limit clause.