13 Determine Appropriate Warehouse Sizes

13.1 Lab Introduction

13.1.1 Purpose

The purpose of this lab is to familiarize you with how different warehouse sizes impact the performance of a query.

This lab is immediately applicable to many personas. If you're a data analyst, you'll see how different warehouse sizes can save you and your business users valuable time. If you're a data engineer, you'll learn the importance of choosing appropriate warehouse sizes for your work. If you're a database admin, you'll learn how important it is to allocate appropriately-sized warehouses for particular groups of users.

13.1.2 What you'll learn

- How to resize warehouses
- How to use the query profile to monitor query performance
- How to use the INFORMATION_SCHEMA and QUERY_HISTORY to analyze query performance

13.1.3 How to complete this lab

In order to complete this lab, you can type the SQL commands below directly into a worksheet. It is not recommended that you cut and paste from the workbook pdf as that sometimes results in errors.

You can also use the SQL code file for this lab that was provided at the start of the class. You would simply need to open it in TextEdit (Mac) or Notepad (Windows), and then copy and paste the SQL code directly into a worksheet.

13.1.4 Scenario

In this task you will disable the query result cache, and run the same query on different sized warehouses to determine which is the best size for the query. You will suspend your warehouse after each test, to clear the data cache so the performance of the next query is not artificially enhanced.

Remember, scaling warehouse size up (for more complex queries) and down (for less complex queries) is a strategy for either making complex queries more performant (in the case of complex queries) or saving compute (in the case of less complex queries). In this example, we introduce a query that requires a scaling up/down strategy (it has a lot of data, includes numerous joins and has more than one filter in the WHERE clause).

Let's get started!

13.1.5 Navigate to **[Worksheets]**.

13.1.6 Create a new worksheet named *Warehouse Sizing* with the following context:

- ROLE: TRAINING_ROLE
- WAREHOUSE: **LEARNER**_WH
- DATABASE: SNOWFLAKE_SAMPLE_DATA
- SCHEMA: TPCDS_SF10TCL

```
USE ROLE TRAINING_ROLE;
CREATE WAREHOUSE IF NOT EXISTS LEARNER_WH;
USE SNOWFLAKE_SAMPLE_DATA.TPCDS_SF10TCL;
```

13.1.7 Change the size of your warehouse to `Xsmall` :

```
ALTER WAREHOUSE LEARNER_WH
SET WAREHOUSE_SIZE = XSmall;
```

13.1.8 Suspend and resume the warehouse to make sure its Data Cache is empty, disable the query result cache and set a query tag:

```
ALTER WAREHOUSE LEARNER_WH SUSPEND;
ALTER WAREHOUSE LEARNER_WH RESUME;
ALTER SESSION SET USE_CACHED_RESULT = FALSE;
ALTER SESSION SET QUERY_TAG = 'LEARNER_WH_Sizing';
```

13.1.9 Run the following query (your test query - it will be used throughout this task) to list detailed catalog sales data together with a running sum of sales price within the order (it will take several minutes to run):

```
SELECT cs_bill_customer_sk, cs_order_number, i_product_name, cs_sales_price,
   SUM(cs_sales_price)
OVER (PARTITION BY cs_order_number
   ORDER BY i_product_name
   ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) run_sum
FROM catalog_sales, date_dim, item
WHERE cs_sold_date_sk = d_date_sk
AND cs_item_sk = i_item_sk
AND d_year IN (2000) AND d_moy IN (1,2,3,4,5,6)
LIMIT 100;
```


13.1.10 View the query profile, and click on the operator WindowFunction[1].

13.1.11 Take note of the performance metrics for this operator: you should see significant spilling to local storage, and possibly spilling to remote storage.

- The spill to local storage indicates the warehouse did not have enough memory, and so it spilled to local SSD storage.
- If it spills to remote storage, the warehouse did not have enough SSD storage to store the spill from memory on its local SSD drive.

13.1.12 Take note of the performance on the small warehouse.

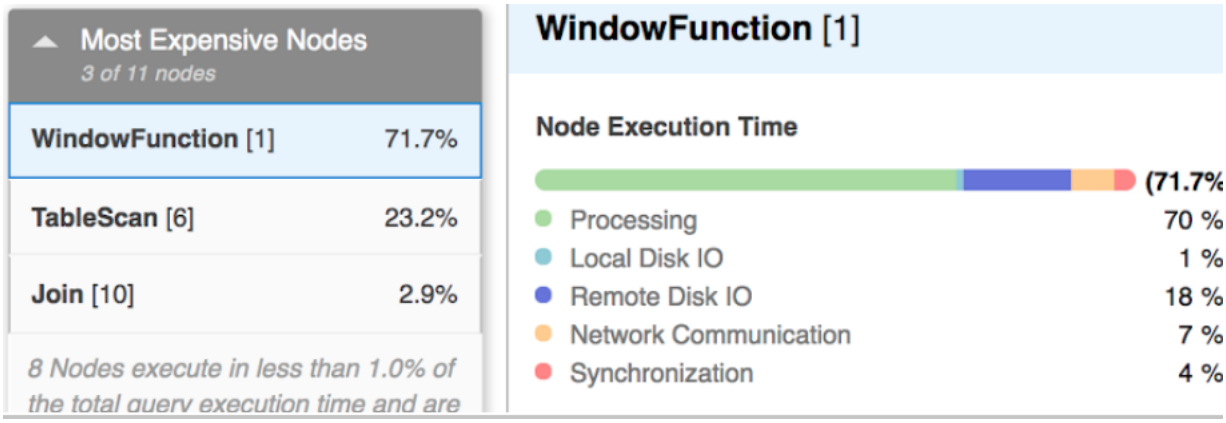

Figure 36: Warehouse Performance

13.2 Run a sample query with a medium warehouse

13.2.1 Suspend your warehouse, change its size to Medium, and resume it:

```
ALTER WAREHOUSE LEARNER_WH SUSPEND;
ALTER WAREHOUSE LEARNER_WH SET WAREHOUSE_SIZE = Medium;
ALTER WAREHOUSE LEARNER_WH RESUME;
```

13.2.2 Re-run the test query:

```
SELECT cs_bill_customer_sk, cs_order_number, i_product_name, cs_sales_price,
   SUM(cs_sales_price)
OVER (PARTITION BY cs_order_number
   ORDER BY i_product_name
   ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) run_sum
FROM catalog_sales, date_dim, item
WHERE cs_sold_date_sk = d_date_sk
AND cs_item_sk = i_item_sk
AND d_year IN (2000) AND d_moy IN (1,2,3,4,5,6)
LIMIT 100;
```


13.2.3 View the query profile, and click the operator WindowFunction[1].

13.2.4 Take note of the performance metrics for this operator. You should see lower amounts spilling to local storage and remote disk, as well as faster execution.

13.3 Run a sample query with a large warehouse

13.3.1 Suspend your warehouse, change its size to Large, and resume it:

```
ALTER WAREHOUSE LEARNER_WH SUSPEND;
ALTER WAREHOUSE LEARNER_WH SET WAREHOUSE_SIZE = Large;
ALTER WAREHOUSE LEARNER_WH RESUME;
```

13.3.2 Re-run the test query:

```
SELECT cs_bill_customer_sk, cs_order_number, i_product_name, cs_sales_price,
    SUM(cs_sales_price)
OVER (PARTITION BY cs_order_number
   ORDER BY i_product_name
   ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) run_sum
FROM catalog_sales, date_dim, item
WHERE cs_sold_date_sk = d_date_sk
AND cs_item_sk = i_item_sk
AND d_year IN (2000) AND d_moy IN (1,2,3,4,5,6)
LIMIT 100;
```

13.3.3 Take note of the performance metrics for this operator. You will see that there is no spilling to local or remote storage.

13.4 Run a sample query with an extra large warehouse

13.4.1 Suspend the warehouse, change it to XLarge in size, and resume it:

```
ALTER WAREHOUSE LEARNER_WH SUSPEND;
ALTER WAREHOUSE LEARNER_WH SET WAREHOUSE_SIZE = XLarge;
ALTER WAREHOUSE LEARNER_WH RESUME;
```

13.4.2 Re-run the test query.

```
SELECT cs_bill_customer_sk, cs_order_number, i_product_name, cs_sales_price,
    SUM(cs_sales_price)
OVER (PARTITION BY cs_order_number
   ORDER BY i_product_name
   ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) run_sum
FROM catalog_sales, date_dim, item
WHERE cs_sold_date_sk = d_date_sk
```



```
AND cs_item_sk = i_item_sk
AND d_year IN (2000) AND d_moy IN (1,2,3,4,5,6)
LIMIT 100;
```

13.4.3 Note the performance.

13.4.4 Suspend your warehouse, and change its size to XSmall:

```
ALTER WAREHOUSE LEARNER_WH SUSPEND;
ALTER WAREHOUSE LEARNER_WH SET WAREHOUSE_SIZE = XSmall;
```

13.5 Run a query showing the query history results for these performance tests using the table function INFORMATION_SCHEMA.QUERY_HISTORY_BY_SESSION()

```
SELECT query_id,query_text, warehouse_size,(execution_time / 1000) Time_in_seconds
    FROM TABLE(information_schema.query_history_by_session())
WHERE  query_tag = 'LEARNER_WH_Sizing'
AND WAREHOUSE_SIZE IS NOT NULL
AND QUERY_TYPE LIKE 'SELECT' ORDER BY start_time DESC;
```

13.6 This query will show more information but the ACCOUNT_USAGE schema has a latency of up to 45 minutes on the QUERY_HISTORY view.

```
SELECT query_id,query_text, warehouse_size,(execution_time / 1000)
   Time_in_seconds, partitions_total, partitions_scanned,
        bytes_spilled_to_local_storage, bytes_spilled_to_remote_storage,
           query_load_percent
FROM "SNOWFLAKE"."ACCOUNT_USAGE"."QUERY_HISTORY"
WHERE  query_tag = 'LEARNER_WH_Sizing'
AND WAREHOUSE_SIZE IS NOT NULL
AND QUERY_TYPE LIKE 'SELECT' ORDER BY start_time DESC;
```

- After running these tests, what size warehouse do you think is best for this query? Why?

13.6.1 Suspend and resize the warehouse

```
ALTER WAREHOUSE LEARNER_WH SET WAREHOUSE_SIZE=XSmall;
ALTER WAREHOUSE LEARNER_WH SUSPEND;
```

13.7 Key Takeaways

- Scaling up can make complex queries more performant; scaling down for less complex queries can save you compute credits.

- You can analyze output from functions provided in the INFORMATION_SCHEMA and the ACCOUNT_USAGE.query_history view to determine query performance.
- Bigger isn't always better when choosing a warehouse size. It's important to choose the right sized warehouse for the performance you need but not needlessly burn compute credits.

