Resource Optimization: Billing Metrics

Overview

This resource optimization guide represents one module of the four contained in the series. These guides are meant to help customers better monitor and manage their credit consumption. Helping our customers build confidence that their credits are being used efficiently is key to an ongoing successful partnership. In addition to this set of Snowflake Quickstarts for Resource Optimization, Snowflake also offers community support as well as Training and Professional Services offerings.

Billing Metrics

Billing queries are responsible for identifying total costs associated with the high level functions of the Snowflake Cloud Data Platform, which includes warehouse compute, snowpipe compute, and storage costs. If costs are noticeably higher in one category versus the others, you may want to evaluate what might be causing that.

These metrics also seek to identify those queries that are consuming the most amount of credits. From there, each of these queries can be analyze for their importance (do they need to be run as frequently, if at all) and explore if additional controls need to be in place to prevent excessive consumption (i.e. resource monitors, statement timeouts, etc.).

What You'll Learn

- how to identify and analyze Snowflake consumption across all services
- how to analyze most resource-intensive queries
- how to analyze serverless consumption

What You'll Need

- A Snowflake Account
- Access to view [Account Usage Data Share]

Related Materials

- Resource Optimization: Setup & Configuration
- Resource Optimization: Usage Monitoring
- Resource Optimization: Performance

Query Tiers

Each query within the Resource Optimization Snowflake Quickstarts will have a tier designation just to the right of its name as "(T*)". The following tier descriptions should help to better understand those designations.

Tier 1 Queries

At its core, Tier 1 queries are essential to Resource Optimization at Snowflake and should be used by each customer to help with their consumption monitoring - regardless of size, industry, location, etc.

Tier 2 Queries

Tier 2 queries, while still playing a vital role in the process, offer an extra level of depth around Resource Optimization and while they may not be essential to all customers and their workloads, it can offer further explanation as to any additional areas in which over-consumption may be identified.

Tier 3 Queries

Finally, Tier 3 queries are designed to be used by customers that are looking to leave no stone unturned when it comes to optimizing their consumption of Snowflake. While these queries are still very helpful in this process, they are not as critical as the queries in Tier 1 & 2.

Billing Metrics (T1)

TIER 1

Description:

Identify key metrics as it pertains to total compute costs from warehouses, serverless features, and total storage

How to Interpret Results:

Where are we seeing most of our costs coming from (compute, serverless, storage)? Are seeing excessive costs in any of those categories that are above expectations?

Primary Schema:

Account_Usage

```
/* These queries can be used to measure where costs have been incurred by
  the different cost vectors within a Snowflake account including:
  1) Warehouse Costs
   2) Serverless Costs
  3) Storage Costs
  To accurately report the dollar amounts, make changes to the variables
   defined on lines 17 to 20 to properly reflect your credit price, the initial
  capacity purchased, when your contract started and the term (default 12 months)
   If unsure, ask your Sales Engineer or Account Executive
USE DATABASE SNOWFLAKE;
USE SCHEMA ACCOUNT USAGE;
SET CREDIT PRICE = 4.00; --edit this number to reflect credit price
SET TERM LENGTH = 12; --integer value in months
SET TERM START DATE = '2019-01-01';
SET TERM AMOUNT = 100000.00; --number(10,2) value in dollars
WITH CONTRACT VALUES AS (
      SELECT
              $CREDIT PRICE::decimal(10,2) as CREDIT PRICE
              , $TERM_AMOUNT::decimal(38,0) as TOTAL_CONTRACT_VALUE
              , $TERM START DATE::timestamp as CONTRACT START DATE
              , DATEADD (month, $TERM LENGTH, $TERM START DATE)::timestamp as
```

```
CONTRACT END DATE
),
PROJECTED USAGE AS (
      SELECT
                CREDIT_PRICE
                ,TOTAL_CONTRACT_VALUE
                , CONTRACT START DATE
                , CONTRACT END DATE
                , (TOTAL CONTRACT VALUE)
                    DATEDIFF (day, CONTRACT_START_DATE, CONTRACT_END_DATE) AS
DOLLARS_PER_DAY
                , (TOTAL CONTRACT VALUE/CREDIT PRICE)
                DATEDIFF(day, CONTRACT START DATE, CONTRACT END DATE) AS CREDITS PER DAY
               CONTRACT VALUES
     FROM
--COMPUTE FROM WAREHOUSES
SELECT
        'WH Compute' as WAREHOUSE GROUP NAME
        ,WMH.WAREHOUSE_NAME
        ,NULL AS GROUP CONTACT
        , NULL AS GROUP COST CENTER
       , NULL AS GROUP COMMENT
        ,WMH.START TIME
        ,WMH.END TIME
        ,WMH.CREDITS USED
        ,$CREDIT PRICE
        ,($CREDIT_PRICE*WMH.CREDITS_USED) AS DOLLARS_USED
        , 'ACTUAL COMPUTE' AS MEASURE TYPE
       SNOWFLAKE.ACCOUNT USAGE.WAREHOUSE METERING HISTORY WMH
UNION ALL
--COMPUTE FROM SNOWPIPE
SELECT
        'Snowpipe' AS WAREHOUSE GROUP NAME
        , PUH. PIPE NAME AS WAREHOUSE NAME
        ,NULL AS GROUP CONTACT
        ,NULL AS GROUP_COST_CENTER
        , NULL AS GROUP COMMENT
        , PUH.START TIME
        , PUH.END TIME
        ,PUH.CREDITS USED
        ,$CREDIT PRICE
        ,($CREDIT PRICE*PUH.CREDITS USED) AS DOLLARS USED
        , 'ACTUAL COMPUTE' AS MEASURE TYPE
       SNOWFLAKE.ACCOUNT USAGE.PIPE USAGE HISTORY PUH
from
```

```
UNION ALL
--COMPUTE FROM CLUSTERING
SELECT
        'Auto Clustering' AS WAREHOUSE GROUP NAME
        ,DATABASE_NAME || '.' || SCHEMA_NAME || '.' || TABLE_NAME AS WAREHOUSE_NAME
        ,NULL AS GROUP CONTACT
        , NULL AS GROUP COST CENTER
        , NULL AS GROUP COMMENT
        ,ACH.START TIME
        ,ACH.END TIME
        ,ACH.CREDITS USED
        ,$CREDIT_PRICE
        , ($CREDIT PRICE*ACH.CREDITS USED) AS DOLLARS USED
        , 'ACTUAL COMPUTE' AS MEASURE TYPE
      SNOWFLAKE.ACCOUNT USAGE.AUTOMATIC CLUSTERING HISTORY ACH
from
UNION ALL
--COMPUTE FROM MATERIALIZED VIEWS
SELECT
        'Materialized Views' AS WAREHOUSE GROUP NAME
        ,DATABASE NAME || '.' || SCHEMA NAME || '.' || TABLE NAME AS WAREHOUSE NAME
        ,NULL AS GROUP CONTACT
        ,NULL AS GROUP COST CENTER
        , NULL AS GROUP COMMENT
        , MVH.START TIME
        ,MVH.END TIME
        ,MVH.CREDITS USED
        ,$CREDIT PRICE
        ,($CREDIT_PRICE*MVH.CREDITS_USED) AS DOLLARS_USED
        , 'ACTUAL COMPUTE' AS MEASURE TYPE
      SNOWFLAKE.ACCOUNT USAGE.MATERIALIZED VIEW REFRESH HISTORY MVH
from
UNION ALL
--COMPUTE FROM SEARCH OPTIMIZATION
        'Search Optimization' AS WAREHOUSE GROUP NAME
        ,DATABASE_NAME || '.' || SCHEMA_NAME || '.' || TABLE_NAME AS WAREHOUSE_NAME
        ,NULL AS GROUP CONTACT
        , NULL AS GROUP COST CENTER
        , NULL AS GROUP COMMENT
        ,SOH.START TIME
        ,SOH.END TIME
        ,SOH.CREDITS USED
        ,$CREDIT PRICE
        ,($CREDIT_PRICE*SOH.CREDITS_USED) AS DOLLARS_USED
        , 'ACTUAL COMPUTE' AS MEASURE TYPE
       SNOWFLAKE.ACCOUNT USAGE.SEARCH OPTIMIZATION HISTORY SOH
from
```

```
UNION ALL
--COMPUTE FROM REPLICATION
        'Replication' AS WAREHOUSE GROUP NAME
       , DATABASE NAME AS WAREHOUSE NAME
        ,NULL AS GROUP CONTACT
        , NULL AS GROUP COST CENTER
        , NULL AS GROUP COMMENT
        ,RUH.START TIME
        , RUH.END TIME
        ,RUH.CREDITS USED
        ,$CREDIT PRICE
        ,($CREDIT_PRICE*RUH.CREDITS_USED) AS DOLLARS_USED
        , 'ACTUAL COMPUTE' AS MEASURE TYPE
       SNOWFLAKE.ACCOUNT USAGE.REPLICATION USAGE HISTORY RUH
from
UNION ALL
--STORAGE COSTS
SELECT
        'Storage' AS WAREHOUSE GROUP NAME
        ,'Storage' AS WAREHOUSE_NAME
       ,NULL AS GROUP CONTACT
        , NULL AS GROUP COST CENTER
        , NULL AS GROUP COMMENT
        ,SU.USAGE DATE
       ,SU.USAGE DATE
        ,NULL AS CREDITS USED
        ,$CREDIT PRICE
       , ((STORAGE BYTES + STAGE BYTES +
FAILSAFE BYTES)/(1024*1024*1024*1024)*23)/DA.DAYS_IN_MONTH AS DOLLARS_USED
       , 'ACTUAL COMPUTE' AS MEASURE TYPE
     SNOWFLAKE.ACCOUNT USAGE.STORAGE USAGE SU
JOIN (SELECT COUNT(*) AS DAYS IN MONTH, TO DATE(DATE PART('year', D DATE)||'-
'||DATE_PART('month',D_DATE)||'-01') as DATE_MONTH FROM
SNOWFLAKE SAMPLE DATA.TPCDS SF10TCL.DATE DIM GROUP BY
TO_DATE(DATE_PART('year',D_DATE)||'-'||DATE_PART('month',D_DATE)||'-01')) DA ON
DA.DATE MONTH = TO DATE (DATE PART ('year', USAGE DATE) | | '-
'||DATE PART('month', USAGE DATE)||'-01')
UNION ALL
SELECT
        NULL as WAREHOUSE GROUP NAME
        ,NULL as WAREHOUSE NAME
        ,NULL as GROUP CONTACT
        ,NULL as GROUP COST CENTER
        ,NULL as GROUP COMMENT
        , DA.D DATE::timestamp as START TIME
        , DA.D DATE::timestamp as END TIME
```

```
,PU.CREDITS_PER_DAY AS CREDITS_USED
        , PU.CREDIT PRICE
        , PU. DOLLARS PER DAY AS DOLLARS USED
        , 'PROJECTED COMPUTE' AS MEASURE TYPE
FROM PROJECTED USAGE PU
JOIN SNOWFLAKE SAMPLE DATA.TPCDS SF10TCL.DATE DIM DA ON DA.D DATE BETWEEN
PU.CONTRACT_START_DATE AND PU.CONTRACT_END_DATE
UNION ALL
SELECT
        NULL as WAREHOUSE GROUP NAME
        ,NULL as WAREHOUSE_NAME
        ,NULL as GROUP CONTACT
        ,NULL as GROUP COST CENTER
        , NULL as GROUP COMMENT
        ,NULL as START TIME
        ,NULL as END TIME
        ,NULL AS CREDITS USED
        , PU. CREDIT PRICE
        ,PU.TOTAL CONTRACT VALUE AS DOLLARS USED
        , 'CONTRACT VALUES' AS MEASURE_TYPE
       PROJECTED USAGE PU
FROM
```

Screenshot

| | WAREHOUSE_GROUP_NAME | WAREHOUSE_NAME | GROUP_CONTACT | GROUP_COST_CENTER | GROUP_COMMENT | START_TIME | END_TIME | CREDITS_USED | \$CREDIT_PRICE | DOLLARS_USED | MEASURE_TYPE |
|---|----------------------|----------------|---------------|-------------------|---------------|-------------------------------|-------------------------------|--------------|----------------|----------------|----------------|
| | WH Compute | COMPUTE_WH | | | | 2021-07-09 13:00:00.000 -0700 | 2021-07-09 14:00:00.000 -0700 | 0.399620556 | 4 | 1.598482224 | ACTUAL COMPUTE |
| | WH Compute | RESET_WH | | | | 2021-07-09 12:00:00:000 -0700 | 2021-07-09 13:00:00.000 -0700 | 0.000061389 | 4 | 0.000245556 | ACTUAL COMPUTE |
| | WH Compute | RESET_WH | | | | 2021-07-29 06:00:00.000 -0700 | 2021-07-29 07:00:00.000 -0700 | 2.752222222 | 4 | 11.008888888 | ACTUAL COMPUTE |
| 4 | Storage | Storage | | | | 2021-03-21 00:00:00:00 -0700 | 2021-03-21 00:00:00:00 -0700 | | 4 | 0.625538534603 | ACTUAL COMPUTE |
| 5 | Storage | Storage | | | | 2020-10-10 00:00:00:00 -0700 | 2020-10-10 00:00:00:00 -0700 | | 4 | 0.625538534601 | ACTUAL COMPUTE |
| 6 | Storage | Storage | | | | 2020-08-25 00:00:00:00 -0700 | 2020-08-25 00:00:00:00 -0700 | | 4 | 0.625538534603 | ACTUAL COMPUTE |
| | Storage | Storage | | | | 2020-11-10 00:00:00:00 -0800 | 2020-11-10 00:00:00:00 -0800 | | 4 | 0.646389819087 | ACTUAL COMPUTE |
| 8 | Storage | Storage | | | | 2021-07-04 00:00:00:00 -0700 | 2021-07-04 00:00:00:00 -0700 | | 4 | 0.625538534829 | ACTUAL COMPUTE |
| 9 | Storage | Storage | | | | 2021-05-19 00:00:00:00 -0700 | 2021-05-19 00:00:00:00 -0700 | | 4 | 0.625538534829 | ACTUAL COMPUTE |
| | Storage | Storage | | | | 2020-12-26 00:00:00:000 -0800 | 2020-12-26 00:00:00:00 -0800 | | 4 | 0.6255385346 | ACTUAL COMPUTE |
| | Storage | Storage | | | | 2021-08-06 00:00:00:00 -0700 | 2021-08-06 00:00:00:00 -0700 | | 4 | 0.625539930966 | ACTUAL COMPUTE |
| | Storage | Storage | | | | 2021-02-12 00:00:00.000 -0800 | 2021-02-12 00:00:00.000 -0800 | | 4 | 0.69256052045 | ACTUAL COMPUTE |
| | Storage | Storage | | | | 2021-07-20 00:00:00.000 -0700 | 2021-07-20 00:00:00:00 -0700 | | 4 | 0.625539522069 | ACTUAL COMPUTE |

Most Expensive Queries (T2)

TIER 2

Description:

This query orders the most expensive queries from the last 30 days. It takes into account the warehouse size, assuming that a 1 minute query on larger warehouse is more expensive than a 1 minute query on a smaller warehouse

How to Interpret Results:

This is an opportunity to evaluate expensive queries and take some action. The admin could:

- -look at the query profile
- -contact the user who executed the query
- -take action to optimize these queries

Primary Schema:

Account_Usage

```
WITH WAREHOUSE SIZE AS
     SELECT WAREHOUSE SIZE, NODES
       FROM (
              SELECT 'XSMALL' AS WAREHOUSE SIZE, 1 AS NODES
              UNION ALL
              SELECT 'SMALL' AS WAREHOUSE SIZE, 2 AS NODES
              UNION ALL
              SELECT 'MEDIUM' AS WAREHOUSE SIZE, 4 AS NODES
              SELECT 'LARGE' AS WAREHOUSE SIZE, 8 AS NODES
              UNION ALL
              SELECT 'XLARGE' AS WAREHOUSE SIZE, 16 AS NODES
              SELECT '2XLARGE' AS WAREHOUSE SIZE, 32 AS NODES
              UNION ALL
              SELECT '3XLARGE' AS WAREHOUSE SIZE, 64 AS NODES
              UNION ALL
              SELECT '4XLARGE' AS WAREHOUSE SIZE, 128 AS NODES
),
QUERY_HISTORY AS
     SELECT QH.QUERY ID
          ,QH.QUERY TEXT
           ,QH.USER NAME
           ,QH.ROLE NAME
           ,QH.EXECUTION TIME
           ,QH.WAREHOUSE SIZE
      FROM SNOWFLAKE.ACCOUNT USAGE.QUERY HISTORY QH
     WHERE START TIME > DATEADD(month, -2, CURRENT_TIMESTAMP())
)
SELECT QH.QUERY ID
     ,'https://' || current_account() ||
'.snowflakecomputing.com/console#/monitoring/queries/detail?queryId='||QH.QUERY ID AS
OU
      ,QH.QUERY TEXT
      ,QH.USER NAME
      ,QH.ROLE NAME
      ,QH.EXECUTION_TIME as EXECUTION_TIME_MILLISECONDS
      , (QH.EXECUTION TIME/(1000)) as EXECUTION TIME SECONDS
      , (QH.EXECUTION TIME/(1000^{\star}60)) AS EXECUTION TIME MINUTES
      , (QH.EXECUTION_TIME/(1000 \star 60 \star 60)) AS EXECUTION_TIME_HOURS
      ,WS.WAREHOUSE SIZE
      ,WS.NODES
      , (QH.EXECUTION TIME/(1000\star60\star60)) \starWS.NODES as RELATIVE PERFORMANCE COST
```

```
FROM QUERY_HISTORY QH

JOIN WAREHOUSE_SIZE WS ON WS.WAREHOUSE_SIZE = upper(QH.WAREHOUSE_SIZE)

ORDER BY RELATIVE_PERFORMANCE_COST DESC

LIMIT 200
;
```

Average Cost per Query by Warehouse (T2)

TIER 2

Description:

This summarize the query activity and credit consumption per warehouse over the last month. The query also includes the ratio of queries executed to credits consumed on the warehouse

How to Interpret Results:

Highlights any scenarios where warehouse consumption is significantly out of line with the number of queries executed. Maybe auto-suspend needs to be adjusted or warehouses need to be consolidated.

Primary Schema:

Account_Usage

```
set credit price = 4; --edit this value to reflect your credit price
SELECT
   COALESCE (WC.WAREHOUSE NAME, QC.WAREHOUSE NAME) AS WAREHOUSE NAME
   ,QC.QUERY COUNT LAST MONTH
   , WC. CREDITS USED LAST MONTH
   ,WC.CREDIT_COST_LAST_MONTH
   ,CAST((WC.CREDIT COST LAST MONTH / QC.QUERY COUNT LAST MONTH) AS decimal(10,2))
AS COST PER QUERY
FROM (
   SELECT
      WAREHOUSE NAME
     , COUNT (QUERY ID) as QUERY COUNT LAST MONTH
   FROM SNOWFLAKE.ACCOUNT USAGE.QUERY HISTORY
   WHERE TO DATE(START TIME) >= TO DATE(DATEADD(month, -1, CURRENT TIMESTAMP())))
   GROUP BY WAREHOUSE NAME
     ) QC
JOIN (
   SELECT
       WAREHOUSE NAME
        ,SUM(CREDITS_USED) as CREDITS USED LAST MONTH
        ,SUM(CREDITS USED) * ($CREDIT PRICE) as CREDIT COST LAST MONTH
   FROM SNOWFLAKE.ACCOUNT USAGE.WAREHOUSE METERING HISTORY
   WHERE TO DATE(START TIME) >= TO DATE(DATEADD(month,-1,CURRENT TIMESTAMP())))
   GROUP BY WAREHOUSE NAME
  ) WC
```

```
ON WC.WAREHOUSE_NAME = QC.WAREHOUSE_NAME
ORDER BY COST_PER_QUERY DESC
;
```

Screenshot

| WAREHOUSE_NAME | QUERY_COUNT_LAST_MONTH | CREDITS_USED_LAST_MONTH | CREDIT_COST_LAST_MONTH | COST_PER_QUERY |
|----------------|------------------------|-------------------------|------------------------|----------------|
| DATALAKE_WH | 53 | 8.392294444 | 33.569177776 | 0.63 |
| COMPUTE_WH | 234 | 10.023920556 | 40.095682224 | 0.17 |
| RESET_WH | 109 | 3.836485278 | 15.345941112 | 0.14 |
| ANALYTICS_WH | 189 | 2.526540556 | 10.106162224 | 0.05 |
| LOAD_WH | 219 | 2.752667501 | 11.010670004 | 0.05 |
| TASK_WH | 302 | 1.077893611 | 4.311574444 | 0.01 |
| BI_MEDIUM_WH | 2 | 0.000060000 | 0.000240000 | 0.00 |
| BI_LARGE_WH | 16734 | 0.481809443 | 1.927237772 | 0.00 |

AutoClustering Cost History (by Day by Object) (T3)

TIER 3

Description:

Full list of tables with auto-clustering and the volume of credits consumed via the service over the last 30 days, broken out by day.

How to Interpret Results:

Look for irregularities in the credit consumption or consistently high consumption

Primary Schema:

Account_Usage

SQL

```
SELECT TO_DATE(START_TIME) as DATE
,DATABASE_NAME
,SCHEMA_NAME
,TABLE_NAME
,TABLE_NAME
,SUM(CREDITS_USED) as CREDITS_USED

FROM "SNOWFLAKE"."ACCOUNT_USAGE"."AUTOMATIC_CLUSTERING_HISTORY"

WHERE START_TIME >= dateadd(month,-1,current_timestamp())
GROUP BY 1,2,3,4

ORDER BY 5 DESC
;
```

Screenshot

| CLUSTERING_DATE DATABASE_NAME SCHEMA_NAME TABLE_NAME CREDITS_USED 2020-02-12 ASSISS_LTEST_UBS PUBLIC GENOTYPES_COPY 268.442864197 2020-04-28 PACIONE_TEST PUBLIC COLLAT_AMT_FACT_AC 118.2905364 2020-10-13 CARLINENO TEODS_UNCLUSTERED AUTOCLUSTER_SOLD_DATE_ITEM 78.132447280 2020-10-13 CARLINENG TPCDS_UNCLUSTERED AUTOCLUSTER_TICKET_NUMBER 67.332296503 2020-10-13 CARLINENG TPCDS_UNCLUSTERED AUTOCLUSTER_SOLD_DATE 57.013410088 2020-10-13 CARLINENG TPCDS_UNCLUSTERED AUTOCLUSTER_SOLD_DATE 57.013410088 2020-10-13 CARLINENG TPCDS_UNCLUSTERED AUTOCLUSTER_SOLD_DATE 57.013410088 2020-10-13 CARLINENG TPCDS_UNCLUSTERED AUTOCLUSTER_TSCT_UMBER 67.3322268503 2020-10-13 CARLINENG TPCDS_UNCLUSTERED AUTOCLUSTER_TSCT_UMBER 57.013410088 2020-10-13 CARLINENG TPCDS_UNCLUSTER_TSCT_UMBER AUTOCLUSTER_TSCT_UMBER 67.332248 2019-12-18 DICLAREE_DB PUBLIC | | | | | |
|--|-----------------|------------------|-------------------|---|---------------|
| 2020-04-28 PAGIONE_TEST PUBLIC COLLAT_AMT_FACT_AC 118.20533388 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_SOLD_DATE_ITEM 78.132147280 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_TRUBBER 67.338280803 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_STL, JUMBER 67.338280803 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_STL, JUMBER 57.013410088 2020-01-10 CERCINES_STAGE GENOMEK GENOMEK GENOTYPES 29.837784284 2020-01-12 LICLARKE_DB PUBLIC SNOW12900_CBD 23.40572708 18.709104655 2020-01-18 SKIPSANDBOX PUBLIC SNOW12900_CBTS 18.2169786 18.2169786 2020-01-19 GEOSPATIAL PUBLIC TEST_CLUSTER_IMPACT_ON_CARD_EST 9.950016226 2020-01-19 GEOSPATIAL PUBLIC C_TEST 4.86702140 2020-01-19 GENESANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID-2778292 4.259085144 2020-01-20 SILARKE_DB <td< td=""><td>CLUSTERING_DATE</td><td>DATABASE_NAME</td><td>SCHEMA_NAME</td><td>TABLE_NAME</td><td>CREDITS_USED</td></td<> | CLUSTERING_DATE | DATABASE_NAME | SCHEMA_NAME | TABLE_NAME | CREDITS_USED |
| 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_SOLD_DATE_ITEM 78.19214728 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_TICKET_NUMBER 67.328298503 2020-10-10 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_SOLD_DATE 57013410088 2020-01-10 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_SOLD_DATE 57013410088 2020-01-10 CARLINENG GENOMEIS, STAGE GENOMEIK | 2020-02-19 | ASHISH_TEST_DB | PUBLIC | GENOTYPES_COPY | 266.442864197 |
| 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_TICKET_NUMBER 67.328296802 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_SOLD_DATE 57.013410080 2020-01-08 GENOMES_STAGE GENOMEK GENOTYPES 29.837784249 2019-12-18 LOLARKE_DB PUBLIC SNOW121900_CBD 25.00527086 2020-06-18 SKIPSANDBOX PUBLIC WEB_SALES 18.709104655 2020-06-18 SKIPSANDBOX PUBLIC SNOW121900_CBDS 18.2067026 2020-06-18 SKIPSANDBOX PUBLIC SNOW121900_CBTS 18.2067026 2020-06-18 SKIPSANDBOX PUBLIC TEST_CLUSTER_IMPACT_ON_CARD_EST 9.85601626 2020-07-02 GEOSPATIAL NASANEX NEX_GDOP_ALL 4.867021801 2020-08-29 CLIARKE_DB PUBLIC C_TEST 4.867021801 2020-08-20 CHERNEY_TEST_DB CLUSTER_TEST V_ORDERS_CUSTOMER 3.60734810 2020-08-20 CHERNEY_TEST_DB PUBLIC LINITEM 0.575332281 2020-08-20 CLARKE_DB | 2020-04-28 | PACIONE_TEST | PUBLIC | COLLAT_AMT_FACT_AC | 118.290533383 |
| 2020-10-13 CARLINENG TPCDS_LINCLUSTERED AUTOCLUSTER_SOLD_DATE 57/01341008 2020-01-08 GENOMES_STAGE GENOMEIK GENOTYPES 29.637784249 2019-12-18 CULARKE_DB PUBLIC SNOW121900_CBD 25.405727088 2020-06-18 SKIPSANDBOX PUBLIC WEL_SALES 18.709104655 2019-12-18 SKIPSANDBOX PUBLIC SNOW121900_CBTS 118.21695786 2020-06-18 SKIPSANDBOX PUBLIC TEST_CLUSTER_IMPACT_ON_CARD_EST 9.950910228 2020-07-01 GEOSPATIAL NASANEX NEX_GDDP_ALL 8.324776052 2020-08-25 JCLARKE_DB PUBLIC C_TEST 4.867021801 2020-08-25 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID-2778292 4.259085124 2020-08-26 SKIPSANDBOX PUBLIC LINEITEM 2.0000649 3.167346136 2020-08-26 SKIPSANDBOX PUBLIC LINEITEM 2.00006649 3.167346136 2020-08-22 CALLOS_TEST_DB PUBLIC LINEITEM 2.00006644 2.00006644 < | 2020-10-13 | CARLINENG | TPCDS_UNCLUSTERED | AUTOCLUSTER_SOLD_DATE_ITEM | 78.132147290 |
| 2020-01-08 GENOMES, STAGE GENOMEIK GENOTYPES 29.83778428 2019-12-18 JULARKE, DB PUBLIC SNOW12190Q, GBD 25.405727088 2020-06-18 SKIPSANDBOX PUBLIC WEL, SALES 18.709104655 2019-12-18 JULARKE, DB PUBLIC SNOW12190Q, GBTS 11.821695786 2020-06-18 SKIPSANDBOX PUBLIC TEST, CLUSTERI, MPACT, ON, CARD, EST 9.9806786 2020-07-09 GEOSPATIAL NASANEX NEX, SDOP, JALL 8.324776052 2020-08-12 JULARKE, DB PUBLIC C, TEST 4.86702180 2020-08-25 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE, D: 2778292 4.259085124 2020-08-26 SKIPSANDBOX PUBLIC LINEITEM 2.00066449 2020-08-27 GEOSPATIAL NASANEX V.ORDERS, CUSTOMER 3.367348738 2020-08-28 JULARKE, DB PUBLIC LINEITEM 2.0208578 2020-08-29 JULARKE, DB PUBLIC LINEITEM 2.0208578 2020-08-29 JULARKE, DB PUBLIC | 2020-10-13 | CARLINENG | TPCDS_UNCLUSTERED | AUTOCLUSTER_TICKET_NUMBER | 67.328296503 |
| 2019-12-18 JCLARKE_DB PUBLIC SNOW121800_CBD 25.405727088 2020-06-18 SKIPSANDBOX PUBLIC WEB_SALES 18.709104855 2019-12-18 JCLARKE_DB PUBLIC SNOW121800_CBTS 11.82169788 2020-06-19 KEYSANDBOX PUBLIC TEST_CLUSTER_IMPACT_ON_CARD_EST 9.56016226 2020-07-01 GEOSPATIAL NASANEX NEX_GDOP_ALL 4.86702180 2020-06-25 JCLARKE_DB PUBLIC C_TEST 4.86702180 2020-07-30 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID-2778292 4.25908514 2020-08-28 SKIPSANDBOX PUBLIC LINETEM 2.000566449 2020-08-29 CALLOS_TEST_DB PUBLIC LINETEM 2.000566449 2020-08-20 JCLARKE_DB PUBLIC LINETEM <td< td=""><td>2020-10-13</td><td>CARLINENG</td><td>TPCDS_UNCLUSTERED</td><td>AUTOCLUSTER_SOLD_DATE</td><td>57.013410098</td></td<> | 2020-10-13 | CARLINENG | TPCDS_UNCLUSTERED | AUTOCLUSTER_SOLD_DATE | 57.013410098 |
| 2020-06-18 SKIPSANDBOX PUBLIC WER.SALES 18.70910485 2019-12-18 JCLARKE_DB PUBLIC SNOW121900_CBTS 118.21695786 2020-07-10 EGSRATIAL PUBLIC TEST_CLUSTER_IMPACT_ON_CARD_EST 9.95001228 2020-07-20 GCORATIAL NASANEX NEX_GODP_ALL 4.870721840 2020-06-25 JCLARKE_DB PUBLIC C_TEST 4.870721840 2020-07-20 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID 2778292 4.269085124 2020-04-28 RITERNEY_TEST_DB CLUSTER_TEST V_ORDERS_CUSTOMER 3.167346136 2020-04-28 JCLARKE_DB PUBLIC LINEITEM 2.000566449 2020-04-28 JCLARKE_DB PUBLIC LINEITEM 2.000566449 2020-04-28 JCLARKE_DB PUBLIC C_TEST 1.20280798 2020-04-29 JCLARKE_DB PUBLIC C_TEST 0.000566449 2020-04-29 JCLARKE_DB NASANEX NEX_GODP_ALL 0.00056649 2020-04-29 GCOSSATIAL NASANEX NEX_GODP_ | 2020-01-08 | GENOMES_STAGE | GENOME1K | GENOTYPES | 29.637784249 |
| 2019-12-18 JCLARKE_DB PUBLIC SNOW121800_CBTS 11.821895786 2020-06-18 SKPSANDBOX PUBLIC TEST_CLUSTER_IMPACT_ON_CARD_EST 9.956016226 2020-07-01 GGOSPATIAL NASANEX NEX_GDOP_ALL 8.324776052 2020-06-25 JCLARKE_DB PUBLIC C_TEST 4.867021840 2020-07-30 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID-2778292 4.259085124 2020-04-28 NTERNEY_TEST_DB CLUSTER_TEST V_ORDERS_CUSTOMER 2.0504084 2.060566449 2020-04-28 JCLARKE_DB PUBLIC LINEITEM 2.060566449 1.20288078 2020-04-29 GLARKE_DB PUBLIC C_TEST 1.20288078 1.20288078 2020-04-29 JCLARKE_DB PUBLIC C_TEST 0.575332287 1.20288078 2020-04-29 JCLARKE_DB PUBLIC NEX_GDOP_ALL 0.598955203 0.57533228 2020-04-29 JCLARKE_DB PUBLIC NEX_GDOP_ALL 0.598955203 0.57533228 2020-04-29 NITERNEY_TEST_DB CLUSTER_TEST | 2019-12-18 | JCLARKE_DB | PUBLIC | SNOW121900_CBD | 25.405727086 |
| 2020-06-18 SKIPSANDBOX PUBLIC TEST_CLUSTER_IMPACT_ON_CARD_EST 9.950016226 2020-07-01 GOSBATIAL NASANEX NEX_GDDP_ALL 8.324776052 2020-06-25 JCLARKE_DB PUBLIC C_TEST 4.867021840 2020-07-30 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID:2778292 4.259085124 2020-04-28 TIERREY_TEST_DB CUSTER_TEST V.ORDERS_CUSTOMER 2.0000488 2.000566449 2020-06-29 JCLARKE_DB PUBLIC LINETEM 2.000566449 1.20238078 2020-06-29 JCLARKE_DB PUBLIC C_TEST 1.20238078 1.20238078 2020-06-29 JCLARKE_DB PUBLIC C_TEST 0.57533228 0.57533228 2020-06-29 JCLARKE_DB PUBLIC NEX_GODP_ALL 0.5968450 0.57533228 2020-06-29 MITERNEY_TEST_DB CLUSTER_TEST ORDERS 0.57533228 0.57533228 2020-06-29 CARLOS, TEST_DB PUBLIC LINETEM 0.5000428 0.57533228 | 2020-06-18 | SKIPSANDBOX | PUBLIC | WEB_SALES | 18.709104655 |
| 2020-07-01 GEOSPATIAL NASANEX NEX.GDDP_ALL 8.324776052 2020-08-25 JCLARKE_DB PUBLIC C_TEST 4.867021804 2020-07-30 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_JD:2778292 4.259085124 2020-08-26 VERTENEY_TEST_LDB CLUSTER_TEST V. ORDERS_CUSTOMER 3.167346136 2020-08-22 CALLOS_TEST_LDB PUBLIC LINEITEM 2.050056244 2020-08-23 CALLOS ARE JB PUBLIC C_TEST 1.202380758 2020-08-23 GEOSPATIAL NASANEX NEX_GDDP_ALL 0.698955203 2020-08-28 TRENEY_TEST_LDB CLUSTER_TEST ORDERS 0.57533228 2020-08-29 CARLOS_TEST_LDB PUBLIC LINEITEM 0.36355380 | 2019-12-18 | JCLARKE_DB | PUBLIC | SNOW121900_CBTS | 11.821695786 |
| 2020-06-25 JCLARKE_DB PUBLIC C_TEST 4.867021801 2020-07-30 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID:2778292 4.259085124 2020-04-28 TITERREY_TEST_LDB CLUSTER_TEST V.ORDERS_CUSTOMER 3.16734833 2020-06-29 CALICOS_TEST_LDB PUBLIC LINEITEM 2.000566449 2020-06-29 JCLARKE_DB PUBLIC C_TEST 1.022380798 2020-07-09 GEOSATIAL NASANEX NEX_GDOP_ALL 0.68965203 2020-08-28 TITERREY_TEST_DB CLUSTER_TEST ORDERS 0.57533228 2020-08-29 CARLOS_TEST_LDB PUBLIC LINEITEM 0.36855308 | 2020-06-18 | SKIPSANDBOX | PUBLIC | TEST_CLUSTER_IMPACT_ON_CARD_EST | 9.956016226 |
| 2020-07-30 SKIPSANDBOX PUBLIC SEARCH OPTIMIZATION ON TABLE_ID:2778292 4.250805124 2020-04-28 NTERNEY_TEST_LDB CLUSTER_TEST V_ORDERS_CUSTOMER 3.167348136 2020-06-22 CARLOS_TEST_LDB PUBLIC LINEITEM 2.000506449 2020-06-28 DLARKE_DB PUBLIC C_TEST 1.20280758 2020-07-02 GEOSATIAL NASANEX NEX_SDOP_ALL 0.698955203 2020-04-28 THERNEY_TEST_DB CLUSTER_TEST ORDERS 0.575332286 2020-06-23 CARLOS_TEST_DB PUBLIC LINEITEM 0.36355380 | 2020-07-01 | GEOSPATIAL | NASANEX | NEX_GDDP_ALL | 8.324776052 |
| 2020-04-28 NTERNEY_TEST_DB CLUSTER_TEST V.ORDERS_CUSTOMER 3167348136 2020-06-22 CARLOS_TEST_DB PUBLIC LINEITEM 2,000566449 2020-06-26 DLARKE_DB PUBLIC C_TEST 1,202380788 2020-07-02 GCOSPATIAL NASANEX NEX_GDOP_ALL 0,808955203 2020-04-28 NTERNEY_TEST_DB CLUSTER_TEST ORDERS 0,57533228 2020-06-23 CARLOS_TEST_DB PUBLIC LINEITEM 0,363553380 | 2020-06-25 | JCLARKE_DB | PUBLIC | C_TEST | 4.867021840 |
| 2020-06-22 CARLOS,TEST_DB PUBLIC LINEITEM 2,060586449 2020-06-26 JCLARKE_DB PUBLIC C,TEST 1,202380758 2020-07-02 GCOSPATIAL NASANEX NEX_GDOP_ALL 0,689855203 2020-04-28 TIERNEY_TEST_DB CLUSTER_TEST CROERS 0,57533238 2020-06-23 CARLOS,TEST_DB PUBLIC LINEITEM 0,363553380 | 2020-07-30 | SKIPSANDBOX | PUBLIC | SEARCH OPTIMIZATION ON TABLE_ID:2778292 | 4.259085124 |
| 2020-06-28 LICLARKE_DB PUBLIC C_TEST 1.20280758 2020-07-02 GOSPATIAL NASANEX NEX_GDOP_ALL 0.698955203 2020-04-28 NTERNEY_TEST_DB CLUSTER_TEST ORDERS 0.575332238 2020-08-23 CARLOS_TEST_DB PUBLIC LINEITEM 0.363553980 | 2020-04-28 | NTIERNEY_TEST_DB | CLUSTER_TEST | V_ORDERS_CUSTOMER | 3.167348136 |
| 2020-07-02 GEOSPATIAL NASANEX NEX.GDDP_ALL 0.698955203 2020-04-28 NTERNEY_TEST_DB CLUSTER_TEST ORDERS 0.575332288 2020-06-23 CARLOS_TEST_DB PUBLIC LINEITEM 0.36355380 | 2020-06-22 | CARLOS_TEST_DB | PUBLIC | LINEITEM | 2.060566449 |
| 2020-04-28 NTIERNEY_TEST_DB CLUSTER_TEST ORDERS 0.575332238 2020-06-23 CARLOS_TEST_DB PUBLIC LINEITEM 0.363553380 | 2020-06-26 | JCLARKE_DB | PUBLIC | C_TEST | 1.202380758 |
| 2020-06-23 CARLOS,TEST_DB PUBLIC LINEITEM 0.383553380 | 2020-07-02 | GEOSPATIAL | NASANEX | NEX_GDDP_ALL | 0.698955203 |
| | 2020-04-28 | NTIERNEY_TEST_DB | CLUSTER_TEST | ORDERS | 0.575332238 |
| 2020-09-23 CARLINENO PUBLIC ORDERS 0.212436370 | 2020-06-23 | CARLOS_TEST_DB | PUBLIC | LINEITEM | 0.363553380 |
| | 2020-09-23 | CARLINENG | PUBLIC | ORDERS | 0.212436370 |

Materialized Views Cost History (by Day by Object) (T3)

TIER 3

Description:

Full list of materialized views and the volume of credits consumed via the service over the last 30 days, broken out by day.

How to Interpret Results:

Look for irregularities in the credit consumption or consistently high consumption

Primary Schema:

Account_Usage

SQL

```
SELECT

TO_DATE(START_TIME) as DATE
, DATABASE_NAME
, DATABASE_NAME
, SCHEMA_NAME
, TABLE_NAME
, SUM(CREDITS_USED) as CREDITS_USED

FROM "SNOWFLAKE"."ACCOUNT_USAGE"."MATERIALIZED_VIEW_REFRESH_HISTORY"

WHERE START_TIME >= dateadd(month, -1, current_timestamp())
GROUP BY 1,2,3,4
ORDER BY 5 DESC
;
```

Search Optimization Cost History (by Day by Object) (T3)

TIER 3

Description:

Full list of tables with search optimization and the volume of credits consumed via the service over the last 30 days, broken out by day.

How to Interpret Results:

Look for irregularities in the credit consumption or consistently high consumption

Primary Schema:

Account_Usage

SQL

```
TSELECT

TO_DATE(START_TIME) as DATE

,DATABASE_NAME

,SCHEMA_NAME

,TABLE_NAME

,SUM(CREDITS_USED) as CREDITS_USED

FROM "SNOWFLAKE"."ACCOUNT_USAGE"."SEARCH_OPTIMIZATION_HISTORY"

WHERE START_TIME >= dateadd(month,-1,current_timestamp())

GROUP BY 1,2,3,4

ORDER BY 5 DESC

;
```

Snowpipe Cost History (by Day by Object) (T3)

TIER 3

Description:

Full list of pipes and the volume of credits consumed via the service over the last 30 days, broken out by day.

How to Interpret Results:

Look for irregularities in the credit consumption or consistently high consumption

Primary Schema:

Account_Usage

```
TO_DATE(START_TIME) as DATE

,PIPE_NAME
,SUM(CREDITS_USED) as CREDITS_USED

FROM "SNOWFLAKE"."ACCOUNT_USAGE"."PIPE_USAGE_HISTORY"

WHERE START_TIME >= dateadd(month,-1,current_timestamp())
GROUP BY 1,2
```

```
ORDER BY 3 DESC;
```

Replication Cost History (by Day by Object) (T3)

TIER 3

Description:

Full list of replicated databases and the volume of credits consumed via the replication service over the last 30 days, broken out by day.

How to Interpret Results:

Look for irregularities in the credit consumption or consistently high consumption

Primary Schema:

Account_Usage

```
SELECT
TO_DATE(START_TIME) as DATE
,DATABASE_NAME
,SUM(CREDITS_USED) as CREDITS_USED
FROM "SNOWFLAKE"."ACCOUNT_USAGE"."REPLICATION_USAGE_HISTORY"
WHERE START_TIME >= dateadd(month,-1,current_timestamp())
GROUP BY 1,2
ORDER BY 3 DESC;
```