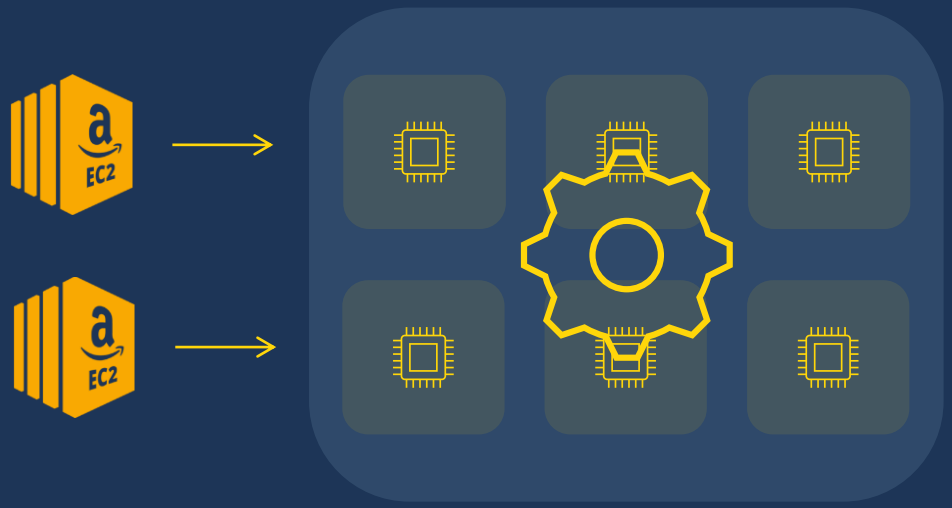
What is a Virtual Warehouse?

Virtual Warehouse Overview



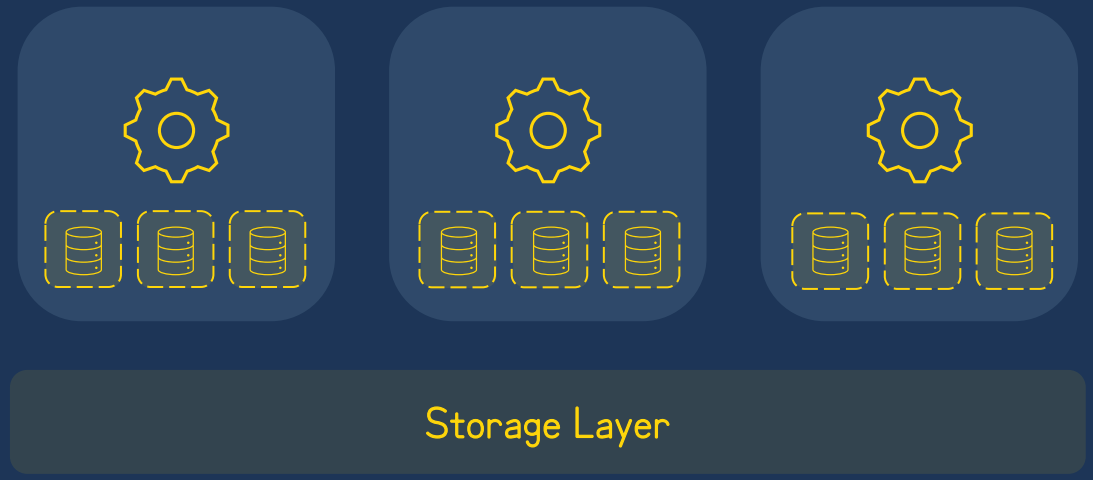
A Virtual Warehouse is a named abstraction for a Massively Parallel Processing (MPP) compute cluster.

Virtual Warehouses execute:

- DQL operations (SELECT)
- DML operations (UPDATE)
- Data loading operations (COPY INTO)

As a user you only interact with the named warehouse object not the underlying compute resources.

Virtual Warehouse Overview



Spin up and shut-down a virtually unlimited number of warehouses without resource contention.

Virtual Warehouse configuration can be changed on-the-fly.

Virtual Warehouses contain local SSD storage used to store raw data retrieved from the storage layer.

Virtual Warehouses are created via the Snowflake UI or through SQL commands.

Virtual Warehouse Overview

```
DROP WAREHOUSE MY_WAREHOUSE;
```

```
CREATE WAREHOUSE MY_MED_WH
WAREHOUSE_SIZE='MEDIUM';
```

```
ALTER WAREHOUSE MY_WH SUSPEND;
```

```
ALTER WAREHOUSE MY_WH_2 SET
WAREHOUSE_SIZE=MEDIUM;
```

```
CREATE WAREHOUSE MY_WH_3
MIN_CLUSTER_COUNT=1
MAX_CLUSTER_COUNT=3
SCALING_POLICY=STANDARD;
```

Sizing and Billing

Virtual Warehouse Sizes

Virtual Warehouses can be created in 10 t-shirt sizes.

Underlying compute power approximately doubles with each size.

In general the larger the Virtual Warehouse the better the query performance.

Choosing a size is typically done by experimenting with a representative query of a workload.

Data loading does not typically require large Virtual Warehouses and sizing up does not guarantee increased data loading performance.

Virtual Warehouse Billing

Virtual Warehouse Size	X-Small	Small	Medium	Large	X-Large	2X-Large	3X-Large	4X-Large	5X-Large	6X-Large
Credits / Hour	1	2	4	8	16	32	64	128	256	512
Credits / Second	0.0003	0.0006	0.0011	0.0022	0.0044	0.0089	0.0178	0.0356	0.0711	0.1422

Credit Calculation

 Running Time	 Credits
0-60 Seconds	0.017
61 Seconds	0.017
2 Minutes	0.033
10 Minutes	0.167
1 Hour	1.000

The first 60 seconds after a virtual warehouse is provisioned and running are always charged.

Credit Pricing

Table 2: On Demand Credit Pricing					
Cloud Provider	Region	Snowflake Service Edition			
		Standard	Enterprise	Business Critical	VPS
AWS	Europe (London)	$2.70	$4.00	$5.40	$8.10
AWS	AP Northeast 1 (Tokyo)	$2.85	$4.30	$5.70	$8.55
Azure	North Europe (Ireland)	$2.60	$3.90	$5.20	$7.80
GCP	Europe West 2 (London)	$2.70	$4.00	$5.40	$8.10

ⓘ Credit price is determined by **region** & **Snowflake edition.**

Credit Pricing

Table 2: On Demand Credit Pricing					
Cloud Provider	Region	Snowflake Service Edition			
		Standard	Enterprise	Business Critical	VPS
AWS	Europe (London)	$2.70	$4.00	$5.40	$8.10
AWS	AP Northeast 1 (Tokyo)	$2.85	$4.30	$5.70	$8.55
Azure	North Europe (Ireland)	$2.60	$3.90	$5.20	$7.80
GCP	Europe West 2 (London)	$2.70	$4.00	$5.40	$8.10

Virtual Warehouse

If a XS Virtual Warehouse is active
for 1 hour
on the Standard Edition of Snowflake
deployed in AWS Europe (London) Region
it will consume 1 Snowflake Credit
costing $2.70 (Nov 2021)

If a L Virtual Warehouse is active
for 3 hours
on the Enterprise Edition of Snowflake
deployed in AWS AP Northeast 1 (Tokyo) Region
it will consume 24 Snowflake Credit
costing $72.00 (Nov 2021)

Virtual Warehouse State and Properties

Virtual Warehouse State

STARTED

SUSPENDED

RESIZING

Virtual Warehouse State

```
CREATE WAREHOUSE MY_MED_WH WITH
WAREHOUSE_SIZE='MEDIUM';
```

By default when a Virtual Warehouse is created it is in the STARTED state.

```
ALTER WAREHOUSE MY_WH SUSPEND;
```

Suspending a Virtual Warehouse puts it in the SUSPENDED state, removing the compute nodes from a warehouse.

```
ALTER WAREHOUSE MY_WH RESUME;
```

Resuming a Virtual Warehouse puts in back into the STARTED state and can execute queries.

Virtual Warehouse State Properties

AUTO SUSPEND

```
CREATE WAREHOUSE MY_MED_WH
AUTO_SUSPEND=300;
```

Specifies the number of seconds of inactivity after which a warehouse is automatically suspended.

AUTO RESUME

```
CREATE WAREHOUSE MY_MED_WH
AUTO_RESUME=TRUE;
```

Specifies whether to automatically resume a warehouse when a SQL statement is submitted to it.

INITIALLY SUSPENDED

```
CREATE WAREHOUSE MY_MED_WH
INITIALLY_SUSPENDED=TRUE;
```

Specifies whether the warehouse is created initially in the 'Suspended' state.

Resource Monitors

Resource Monitors

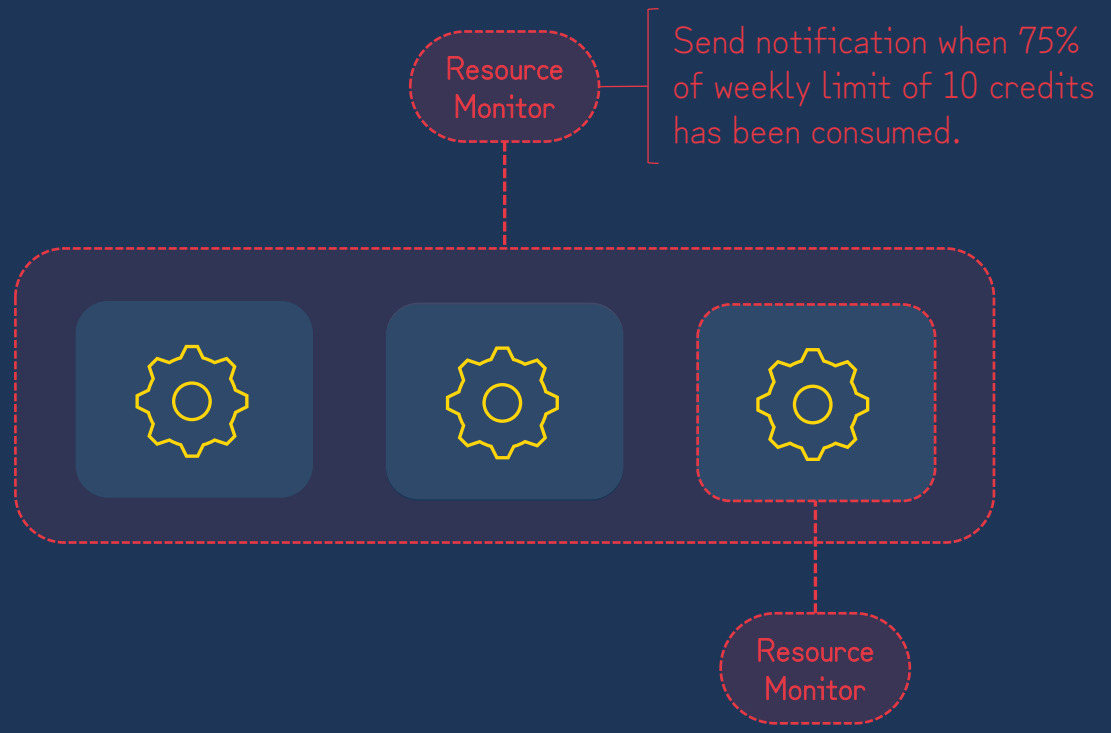
Resource Monitors are objects allowing users to set credit limits on user managed warehouses.

Resource Monitors can be set on either the account or individual warehouse level.

Limits can be set for a specified interval or data range.

When limits are reached an action can be triggered, such as notify user or suspend warehouse.

Resource Monitors can only be created by account administrators.



Resource Monitors

```
CREATE RESOURCE MONITOR ANALYSIS_RM
1   WITH CREDIT_QUOTA=100
2   FREQUENCY=MONTHLY
3   START_TIMESTAMP='2023-01-04 00:00 GMT'
4   TRIGGERS ON 50 PERCENT DO NOTIFY
             ON 75 PERCENT DO NOTIFY
             ON 95 PERCENT DO SUSPEND
             ON 100 PERCENT DO SUSPEND_IMMEDIATE;
```

1

Number of credits allocated to the resource monitor per frequency interval.

2

DAILY, WEEKLY, MONTHLY, YEARLY or NEVER.

3

Start timestamp determines when a resource monitor will start once applied to a warehouse or account. The frequency is relative to the start timestamp.

4

Triggers determine the condition for a certain action to take place.

```
ALTER ACCOUNT SET RESOURCE_MONITOR = ANALYSIS_RM;
```

Virtual Warehouse Concurrency & Query Complexity

Scaling Up: Resizing Virtual Warehouses

X-Small

Large

```
ALTER WAREHOUSE MY_WH
SET WAREHOUSE_SIZE=LARGE;
```

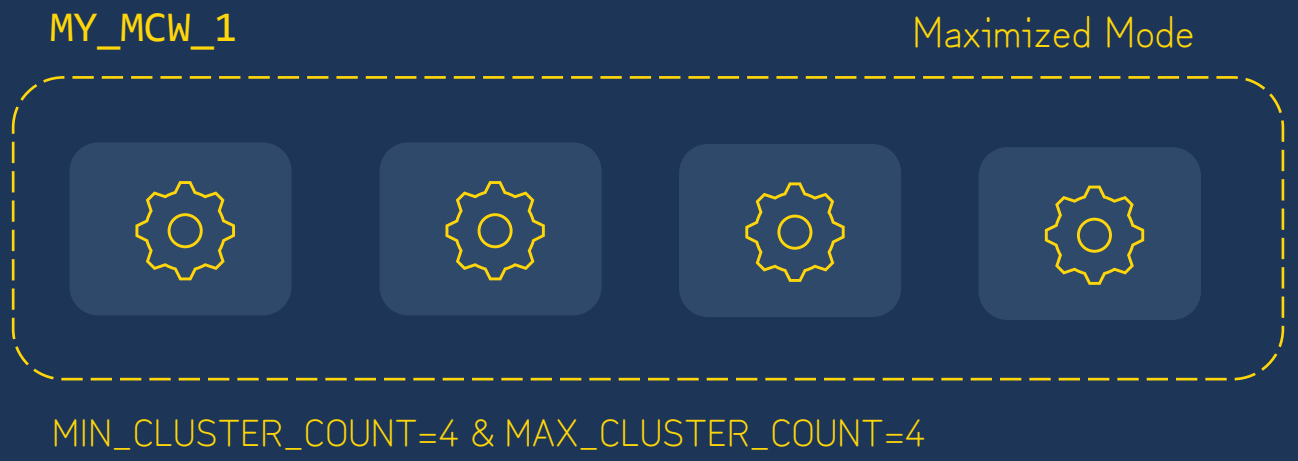
Scaling up a Virtual Warehouse is intended to improve query performance.

Virtual Warehouses can be manually resized via the Snowflake UI or SQL commands.

Resizing a running warehouse does not impact running queries. The additional compute resources are used for queued and new queries.

Decreasing size of running warehouse removes compute resources from the warehouse and clears the warehouse cache.

Scaling out: Multi-cluster Warehouses



A multi-cluster warehouse is a named group of virtual warehouses which can automatically scale in and out based on the number of concurrent users/queries.

MIN_CLUSTER_COUNT specifies the minimum number of warehouses for a multi-cluster warehouse.

MAX_CLUSTER_COUNT specifies the maximum number of warehouses for a multi-cluster warehouse.

Setting these two values the same will put the multi-cluster warehouse in MAXIMIZED mode.

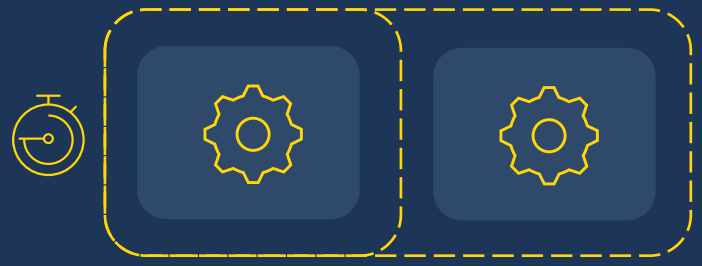
Setting these two values differently will put the multi-cluster warehouse in AUTO-SCALE mode.

Standard Scaling Policy

Standard Scaling Policy

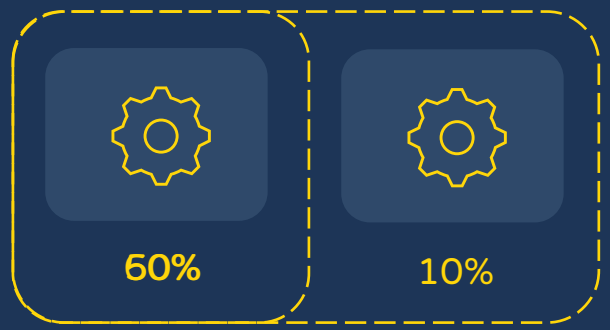
```
CREATE WAREHOUSE MY_MCW_1
MIN_CLUSTER_COUNT=1
MAX_CLUSTER_COUNT=4
SCALING_POLICY=STANDARD;
```

Scaling Out



When a query is queued a new warehouse will be added to the group immediately.

Scaling In



Every minute a background process will check if the load on the least busy warehouse can be redistributed to another warehouse.

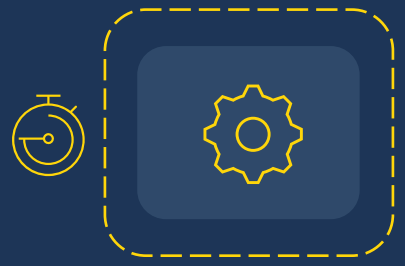
If this condition is met after 2 consecutive minutes a warehouse will be marked for shutdown.

Economy Scaling policy

Economy Scaling Policy

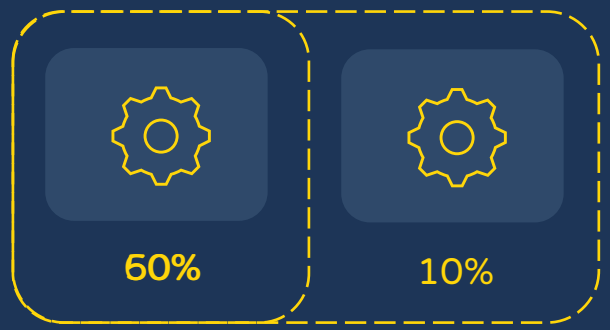
```
CREATE WAREHOUSE MY_MCW_2
MIN_CLUSTER_COUNT=1
MAX_CLUSTER_COUNT=4
SCALING_POLICY=ECONOMY;
```

Scaling Out



When a query is queued the system will estimate if there's enough query load to keep a new warehouse busy for 6 minutes.

Scaling In



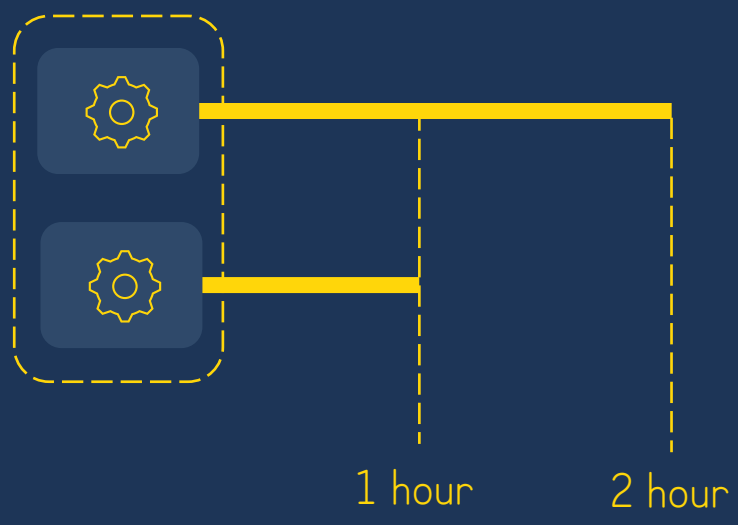
Every minute a background process will check if the load on the least busy warehouse can be redistributed to another warehouse.

If this condition is met after 6 consecutive minutes a warehouse will be marked for shutdown.

Multi-cluster Warehousing Billing

```
CREATE WAREHOUSE MY_MCW
MIN_CLUSTER_COUNT=1
MAX_CLUSTER_COUNT=3
WAREHOUSE_SIZE='MEDIUM';
```

3 x 4 Credits = **12 Credits** / Hour



The total credit cost of a multi-cluster warehouse is the sum of all the individual running warehouses that make up that cluster.

The maximum number of credits a multi-cluster can consume is the number of warehouses multiplied by the hourly credit rate of the size of the warehouses.

Because multi-cluster warehouses scale in and out based on demand it's typical to get some fraction of the maximum credit consumption.

Concurrency Behaviour Properties

MAX CONCURRENCY LEVEL

```
CREATE WAREHOUSE MY_MED_WH
MAX_CONCURRENCY_LEVEL=6;
```

Specifies the number of concurrent SQL statements that can be executed against a warehouse before either it is queued or additional compute power is provided.

STATEMENT QUEUED TIMEOUT IN SECONDS

```
CREATE WAREHOUSE MY_MED_WH
STATEMENT_QUEUED_TIMEOUT_IN_SECONDS=60;
```

Specifies the time, in seconds, a SQL statement can be queued on a warehouse before it is aborted.

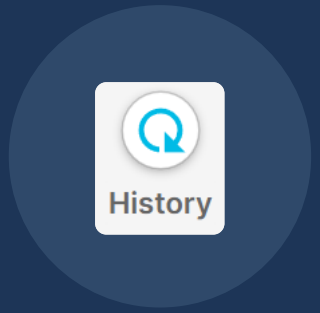
STATEMENT TIMEOUT IN SECONDS

```
CREATE WAREHOUSE MY_MED_WH
STATEMENT_TIMEOUT_IN_SECONDS=600;
```

It specifies the time, in seconds, after which any running SQL statement on a warehouse is aborted.

Performance and Tuning Overview

Query Performance Analysis Tools



History Tab

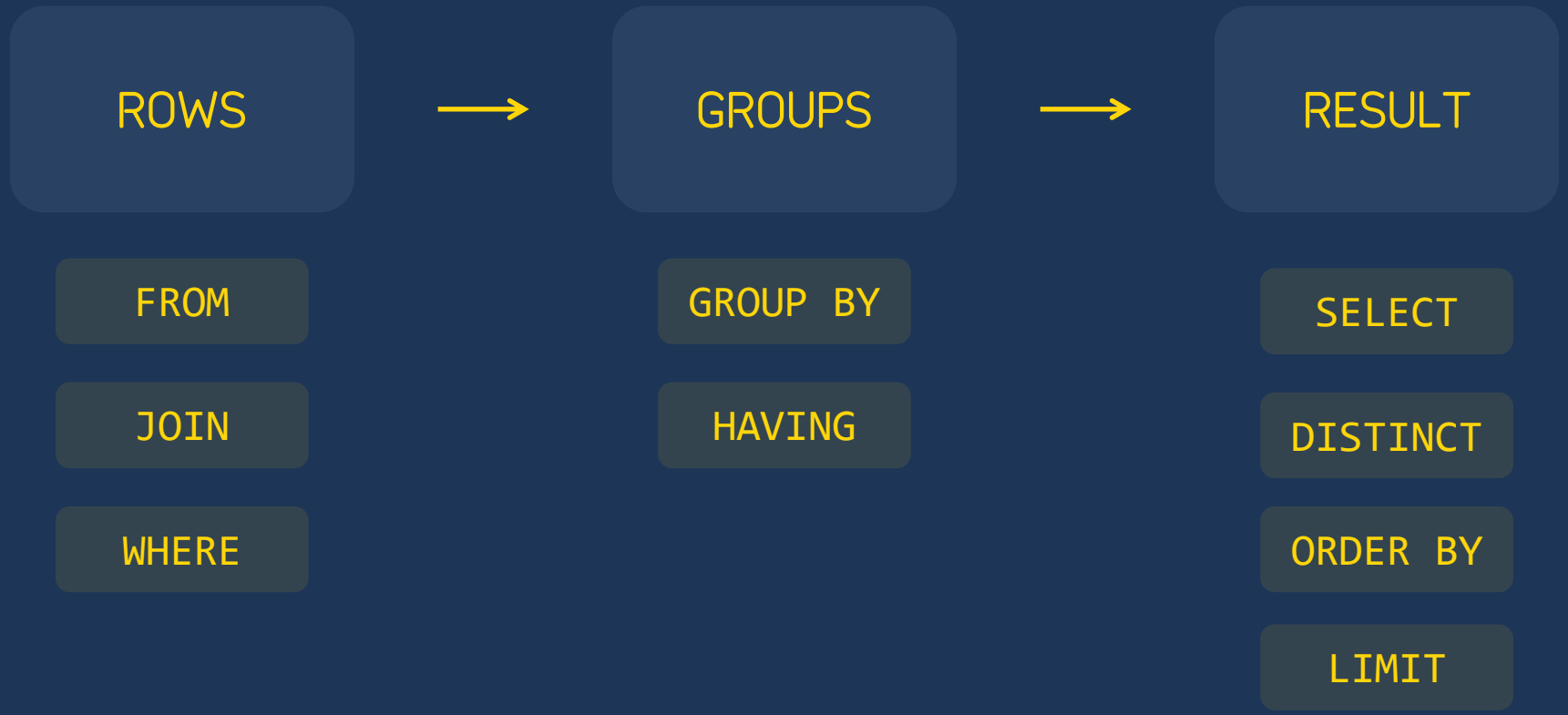
Query History

Query Profile

1. History Tab displays query history for the last 14 days.
2. Users can view other users queries but cannot view their query results.

SQL Tuning

Database Order Of Execution



Join Explosion

Orders

ORDER_DATE	PRODUCT_NAME	CUSTOMER_NAME	ORDER_AMOUNT
01/12/2022	Apple MacBook Air	Arun	1
13/12/2021	Sony Playstation 5	Ana	1
01/01/2022	LG Blu-ray Player	Pawel	2
21/02/2020	Sony Playstation 5	Susan	1

Products

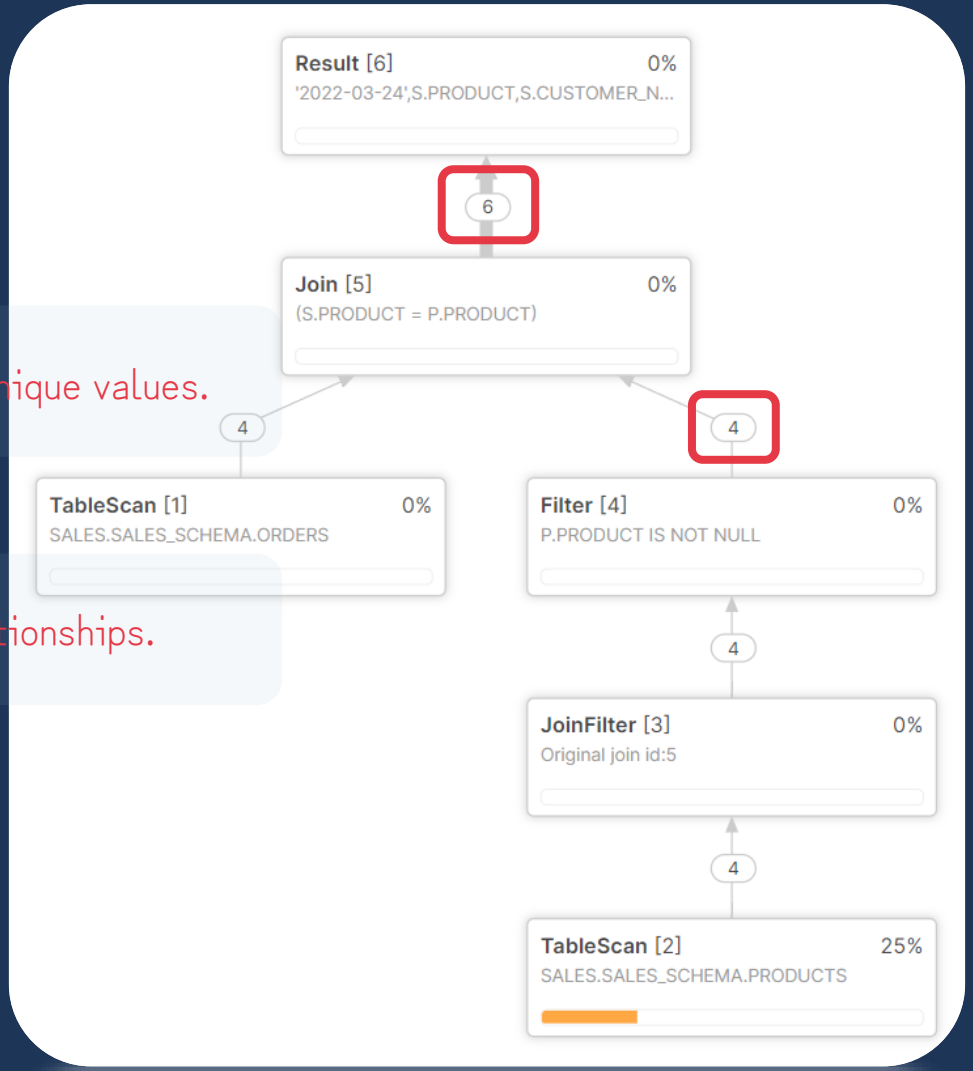
PRODUCT_NAME	PRODUCT_PRICE	ORDER_DATE
Apple MacBook Air	899.99	01/12/2022
LG Blu-ray Player	110.00	01/01/2022
Sony Playstation 5	449.00	12/11/2020
Sony Playstation 5	429.00	10/06/2021

```
SELECT *, (O.ORDER_AMOUNT * P.PRODUCT_PRICE)
FROM ORDERS O
LEFT JOIN PRODUCTS P ON O.PRODUCT = P.PRODUCT;
```

ORDER_DATE	PRODUCT_NAME	CUSTOMER_NAME	ORDER_AMOUNT	ORDER_TOTAL
01/12/2022	Apple MacBook Air	Arun	1	899.99
13/12/2021	Sony Playstation 5	Ana	1	449.00
01/01/2022	LG Blu-ray Player	Pawel	2	220.00
21/02/2020	Sony Playstation 5	Susan	1	449.00
21/02/2020	Sony Playstation 5	Susan	1	429.00
13/12/2021	Sony Playstation 5	Ana	1	429.00

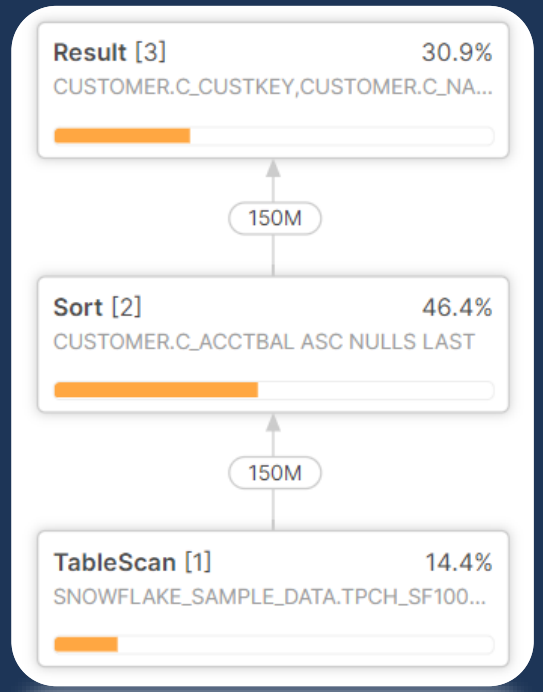
Join Explosion

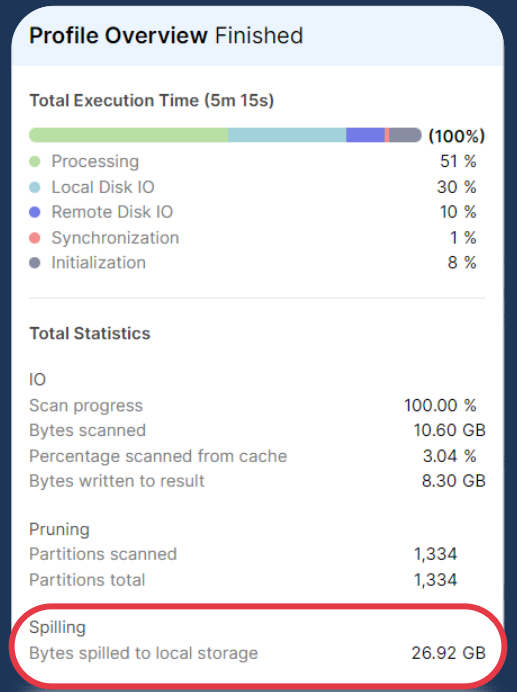
1. Join on columns with unique values.
2. Understand table relationships.



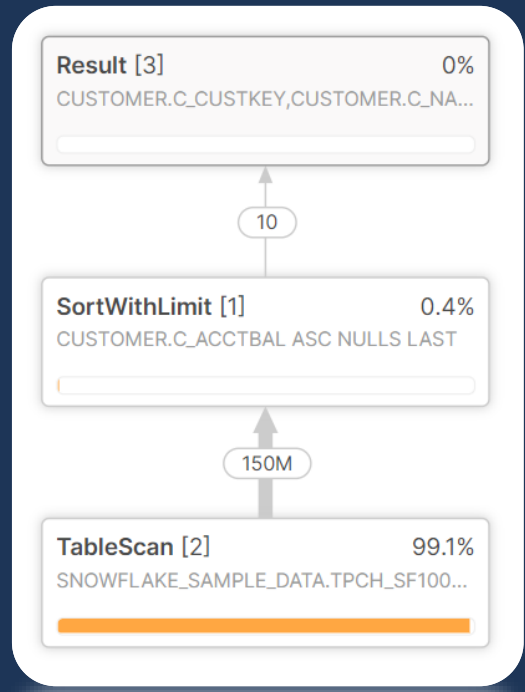
Limit & Order By

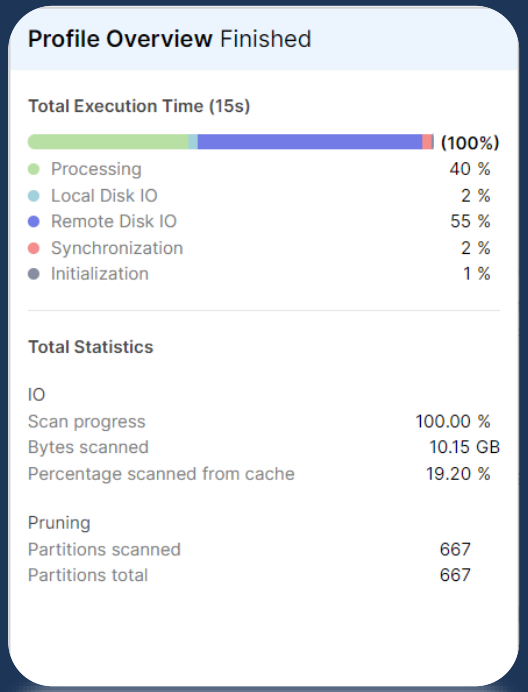
```
SELECT * FROM CUSTOMER
ORDER BY C_ACCTBAL;
```



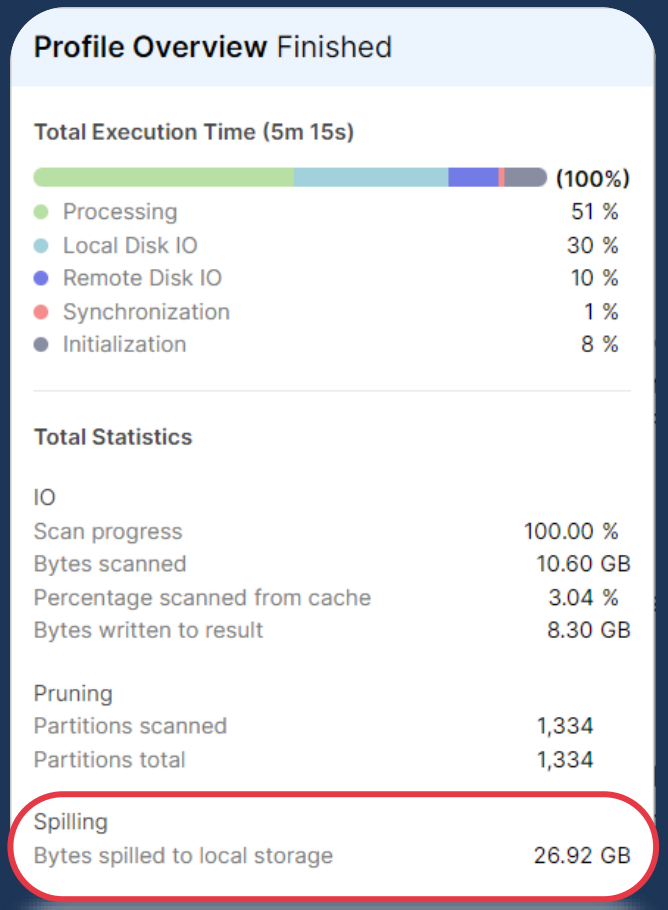



```
SELECT * FROM CUSTOMER
ORDER BY C_ACCTBAL
LIMIT 10;
```





Spilling to Disk



"Bytes spilled to local storage"

Volume of data spilled to virtual warehouse local disk.

"Bytes spilled to remote storage"

Volume of data spilled to remote disk.

1. Process less data.
2. Virtual Warehouse size.

Order By Position

```
SELECT C_NATIONKEY, R.R_NAME, TOTAL_BAL FROM
(
  SELECT
       C_NATIONKEY,
       COUNT(C_ACCTBAL) AS TOTAL_BAL
  FROM CUSTOMER
      GROUP BY C_NATIONKEY
      ORDER BY C_NATIONKEY
) C JOIN REGION R ON (C.C_NATIONKEY = R.R_REGIONKEY)
 ORDER BY TOTAL_BAL;
```

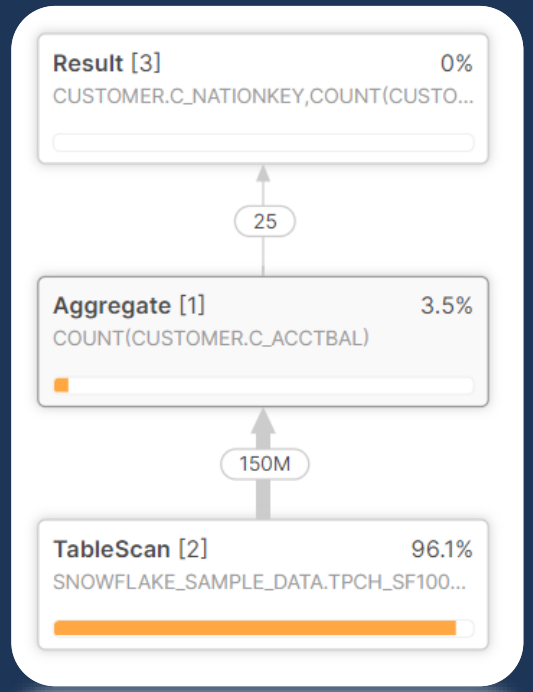
Redundant

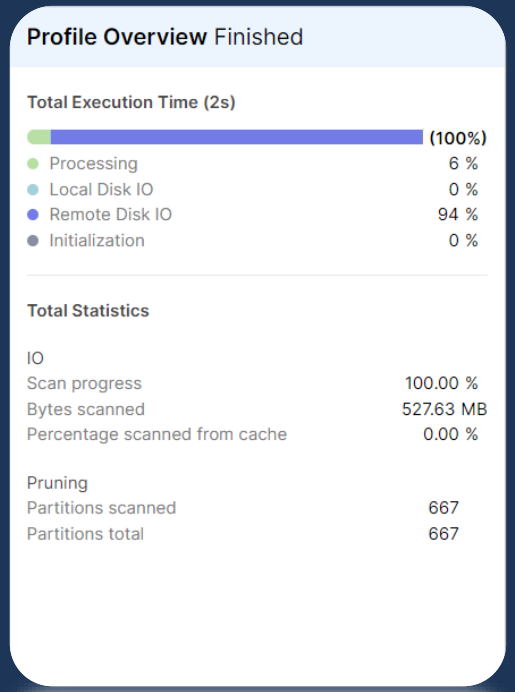
Top-level

1. ORDER BY in top-level select only.

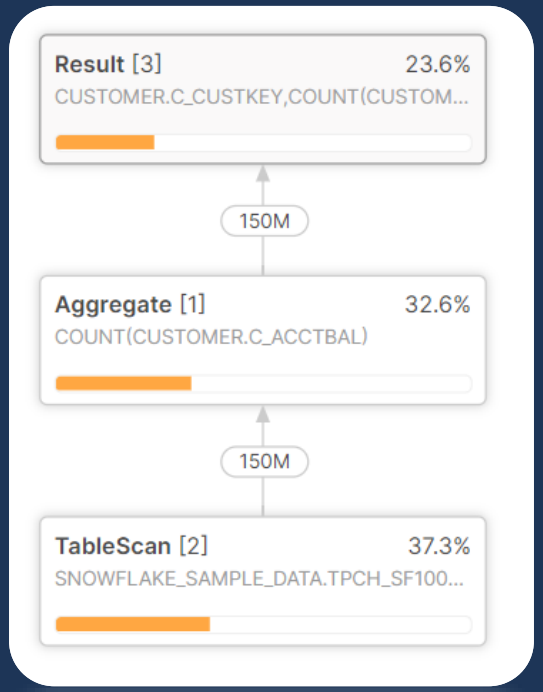
Group By

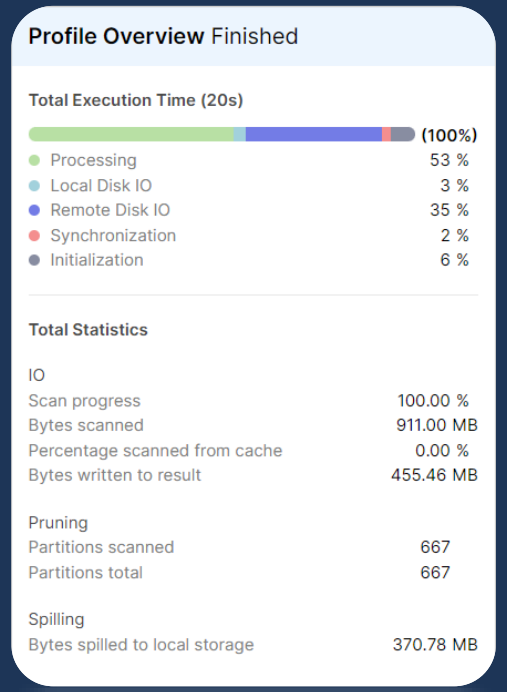
```
SELECT C_NATIONKEY, COUNT(C_ACCTBAL)
FROM CUSTOMER
GROUP BY C_NATIONKEY; -- Low Cardinality
```





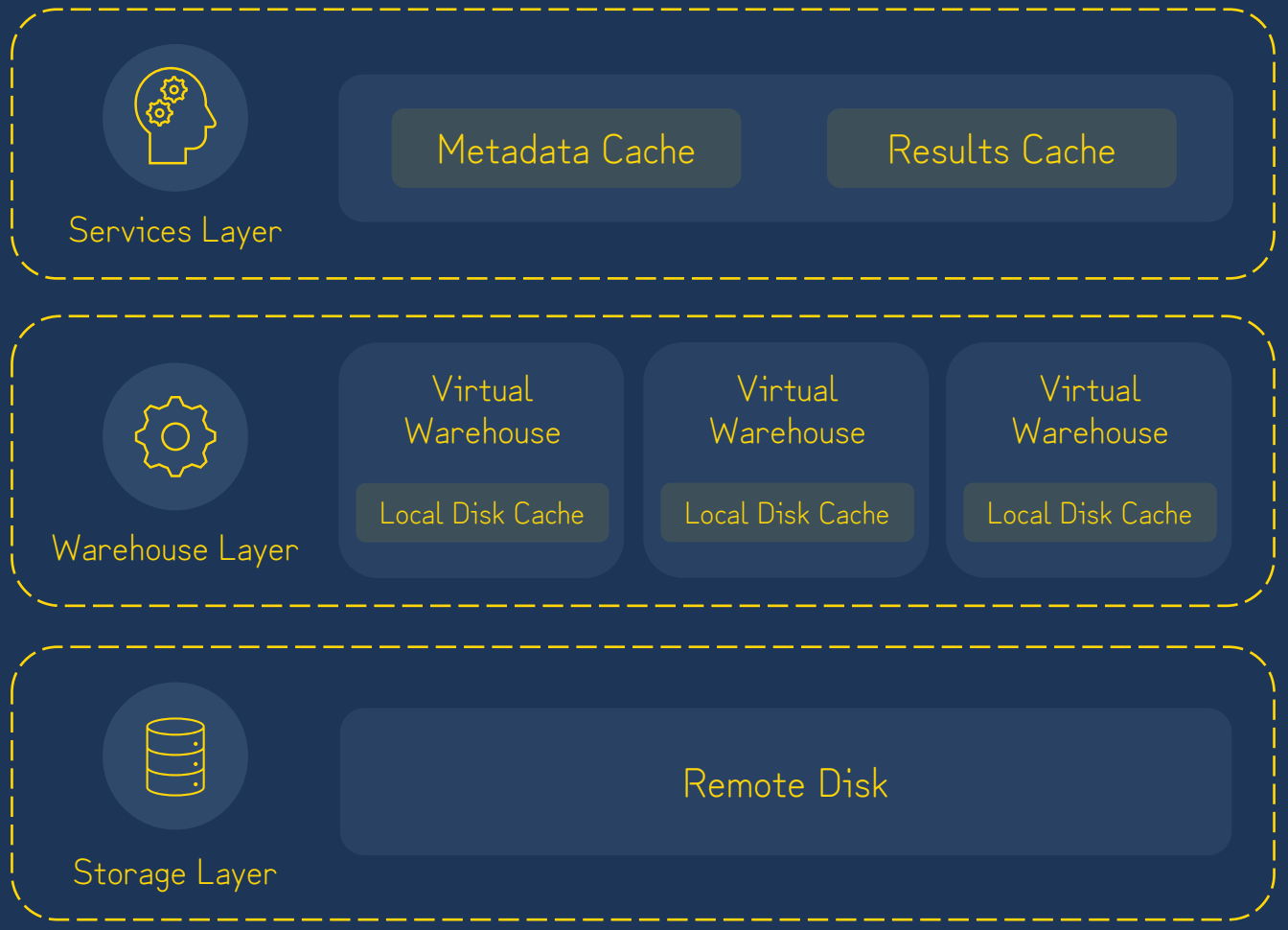

```
SELECT C_CUSTKEY, COUNT(C_ACCTBAL)
FROM CUSTOMER
GROUP BY C_CUSTKEY; -- High Cardinality
```



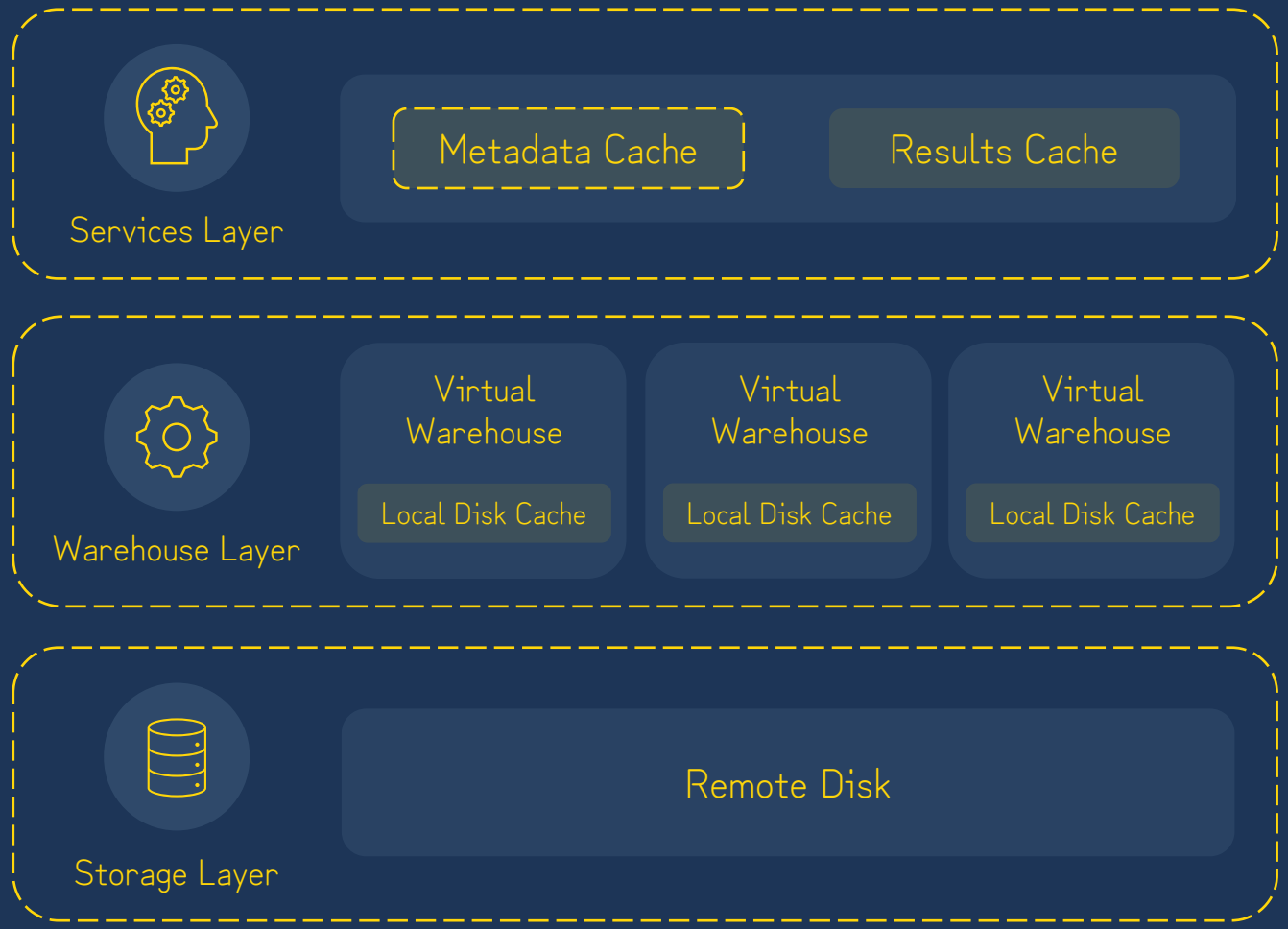


Caching

Caching


Services Layer
Metadata Cache
Results Cache
Warehouse Layer
Virtual Warehouse
Local Disk Cache
Virtual Warehouse
Local Disk Cache
Virtual Warehouse
Local Disk Cache
Storage Layer
Remote Disk

Caching



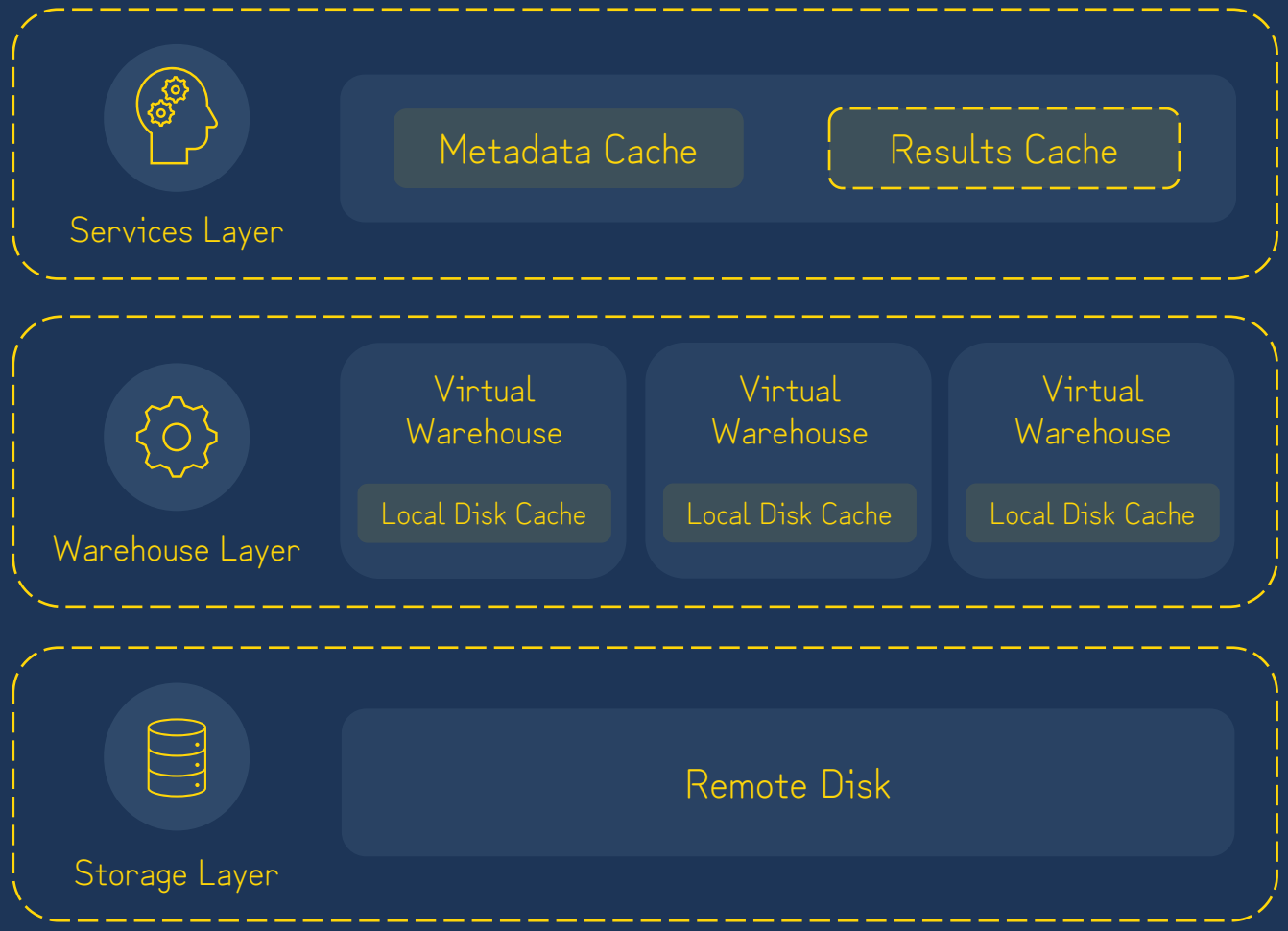
Metadata Cache

Snowflake has a high availability metadata store which maintains metadata object information and statistics.

Some queries can be completed purely using this metadata, not requiring a running virtual warehouse.

```
SELECT COUNT(*) FROM MY_TABLE;

SELECT SYSTEM$WHITELIST();

SELECT CURRENT_DATABASE();

DESCRIBE TABLE MY_TABLE;

SHOW TABLES;
```

Caching



Result Cache

24hr

31 Days

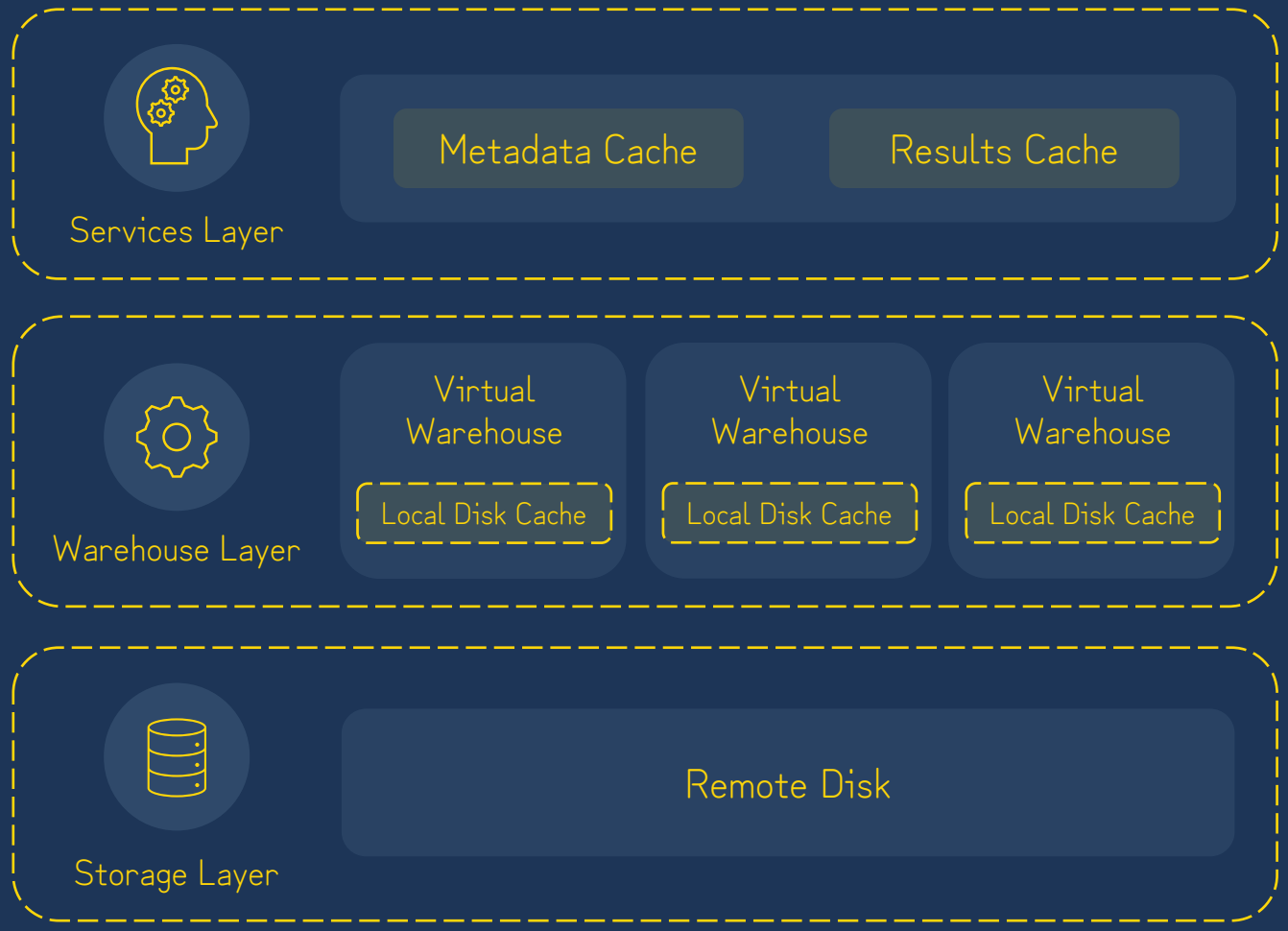
To reuse a result:

- New query exactly matches previous query.
- The underlying table data has not changed.
- The same role is used as the previous query.

If time context functions are used, such as `CURRENT_TIME()`, the result cache will not be used.

Result reuse can be disabled using the session parameter `USE_CACHED_RESULT`.

Caching



Warehouse Cache

Virtual Warehouses have local SSD storage which maintains raw table data used for processing a query.

The larger the virtual warehouse the greater the local cache.

It is purged when the virtual warehouse is resized, suspended or dropped.

Can be used partially, retrieving the rest of the data required for a query from remote storage.

Materialized Views

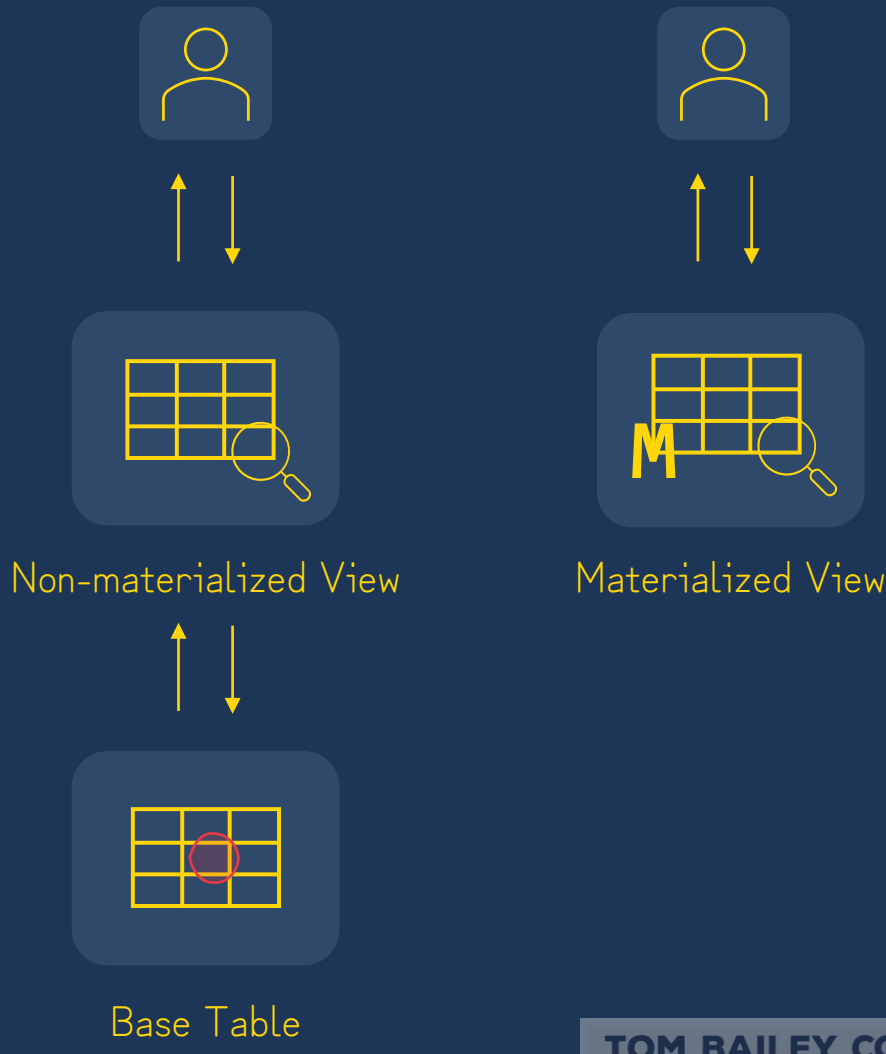
Materialized Views

"A Materialized View is a pre-computed & persisted data set derived from a SELECT query."

MVs are updated via a background process ensuring data is current and consistent with the base table.

MVs improve query performance by making complex queries that are commonly executed readily available.

MVs are an enterprise edition and above serverless feature.



Materialized Views

MVs use compute resources to perform automatic background maintenance.

MVs use storage to store query results, adding to the monthly storage usage for an account.

MATERIALIZED_VIEW_REFRESH_HISTORY

MVs can be created on top of External Tables to improve their query performance.

MVs are limited in the following ways:

- Single Table
- JOIN
- UDF, HAVING, ORDER BY, LIMIT, WINDOW FUNCTIONS

```
CREATE OR REPLACE MATERIALIZED VIEW MV1 AS
SELECT COL1, COL2 FROM T1;
```

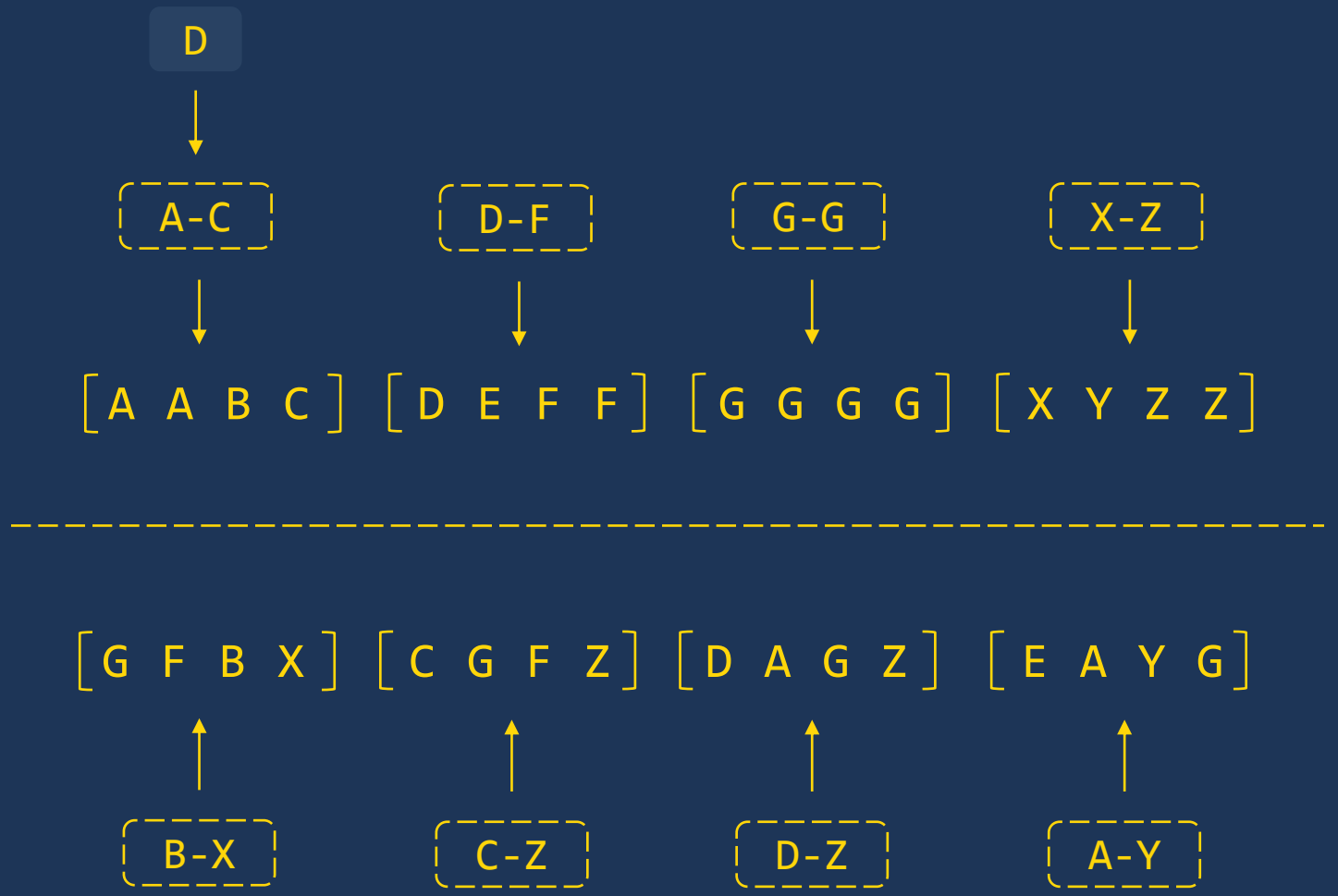
```
ALTER MATERIALIZED VIEW MV1 SUSPEND;
```

```
ALTER MATERIALIZED VIEW MV1 RESUME;
```

```
SHOW MATERIALIZED VIEWS LIKE 'MV1%';
```

Clustering

Natural Clustering



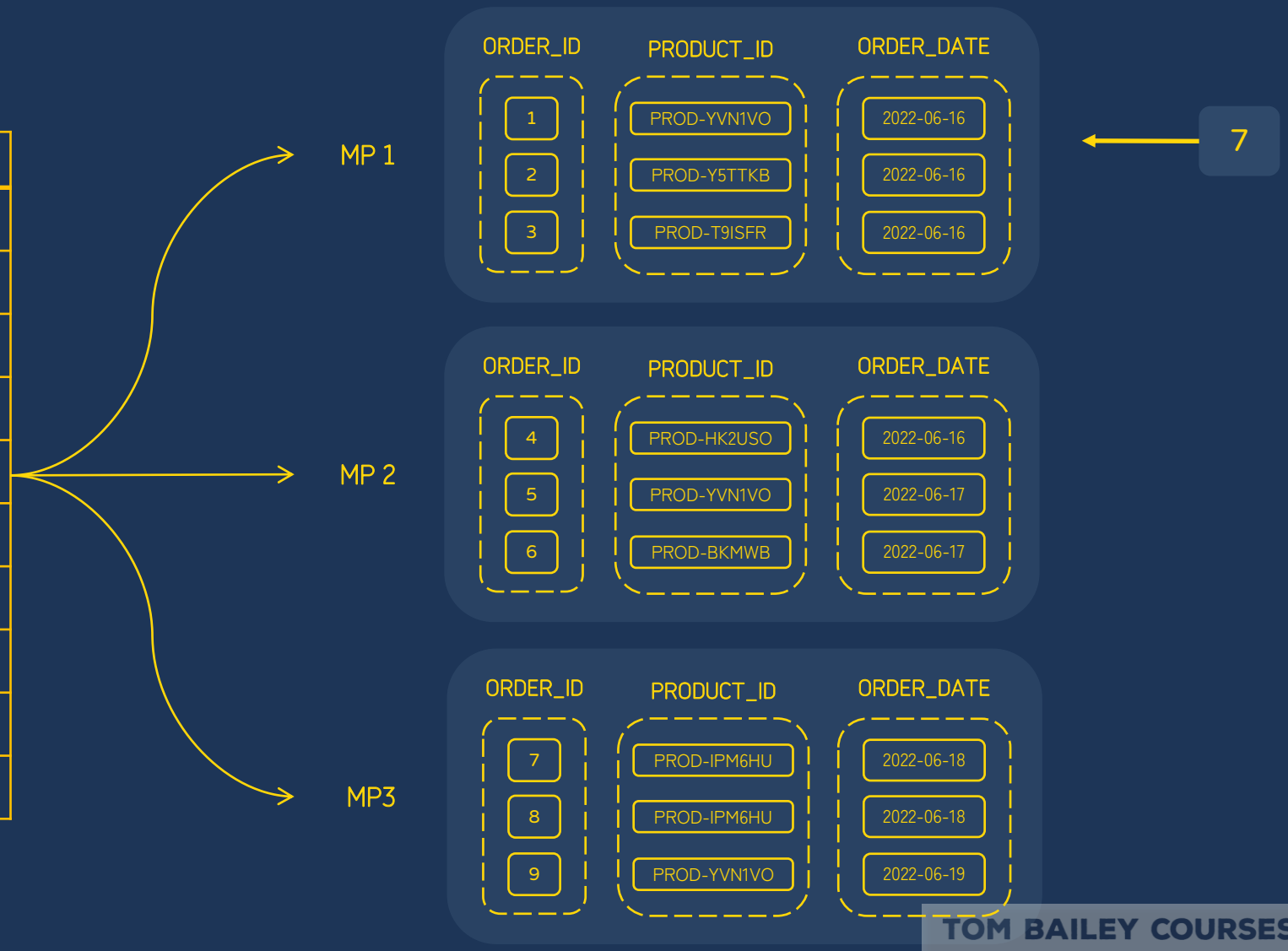
Clustering

`Products.csv`

ORDER_ID	PRODUCT_ID	ORDER_DATE
1	PROD-YVN1VO	2022-06-16
2	PROD-Y5TTKB	2022-06-16
3	PROD-T9ISFR	2022-06-16
4	PROD-HK2USO	2022-06-16
5	PROD-YVN1VO	2022-06-17
6	PROD-BKMWB	2022-06-17
7	PROD-IPM6HU	2022-06-18
8	PROD-IPM6HU	2022-06-18
9	PROD-YVN1VO	2022-06-19
...	...	...

Clustering

ORDER_ID	PRODUCT_ID	ORDER_DATE
1	PROD-YVN1VO	2022-06-16
2	PROD-Y5TTKB	2022-06-16
3	PROD-T9ISFR	2022-06-16
4	PROD-HK2USO	2022-06-16
5	PROD-YVN1VO	2022-06-17
6	PROD-BKMWB	2022-06-17
7	PROD-IPM6HU	2022-06-18
8	PROD-IPM6HU	2022-06-18
9	PROD-YVN1VO	2022-06-19
...	...	...



Clustering Metadata

Snowflake maintains the following clustering metadata for micro-partitions in a table:

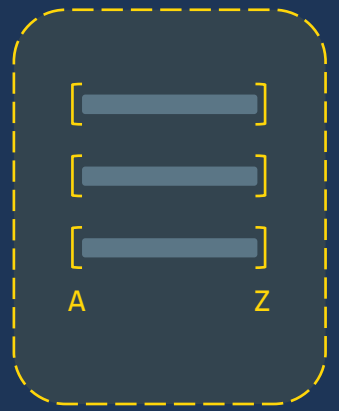
1. Total Number of Micro-partitions
2. Number of Overlapping Micro-partitions
3. Depth of Overlapping Micro-partitions

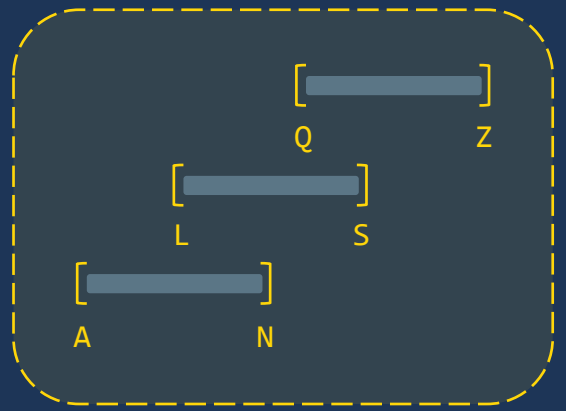
SYSTEM$CLUSTERING_INFORMATION

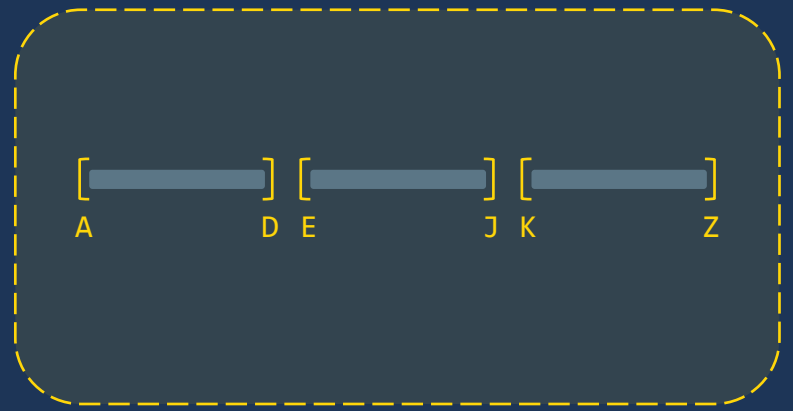
```
SELECT system$clustering_information('table','(col1,col3)');
```


```
"cluster_by_keys" : "(COL1, COL3)",
"total_partition_count" : 1156,
"total_constant_partition_count" : 0,
"average_overlaps" : 117.5484,
"average_depth" : 64.0701,
"partition_depth_histogram" : {
  "00000" : 0,
  "00001" : 0,
  "00002" : 3,
  "00003" : 3,
  "00004" : 4,
  "00005" : 6,
  "00006" : 3,
  "00007" : 5,
  "00008" : 10,
  "00009" : 5,
  "00010" : 7,
  "00011" : 6,
  "00012" : 8,
  "00013" : 8,
  "00014" : 9,
  "00015" : 8,
  "00016" : 6,
  "00032" : 98,
  "00064" : 269,
  "00128" : 698
}
```

Clustering Depth







Overlapping Micro-partitions	3	3	0
Overlap Depth	3	2	1

Automatic Clustering

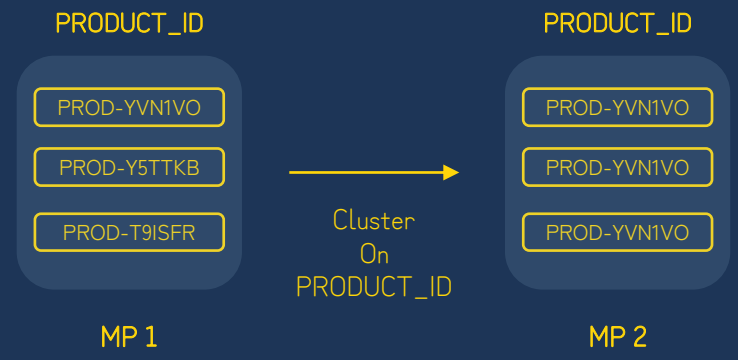
Automatic Clustering

Snowflake supports specifying one or more table columns/expressions as a clustering key for a table.

Clustering aims to co-locate data of the clustering key in the same micro-partitions.

Clustering improves performance of queries that frequently filter or sort on the clustered keys.

Clustering should be reserved for large tables in the multi-terabyte range.



`WHERE` `JOIN` `ORDER BY` `GROUP BY`

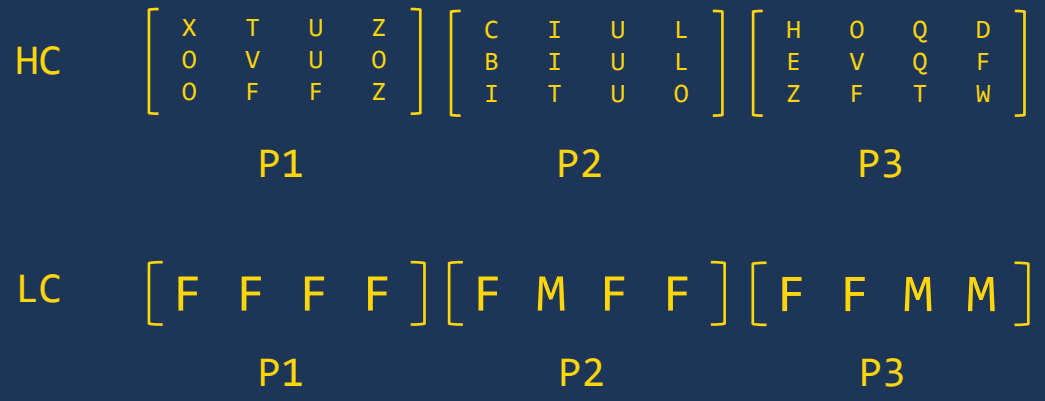
`>1TB`

Choosing a Clustering Key

Snowflake recommended a maximum of 3 or 4 columns (or expressions) per key.

Columns used in common queries which perform filtering and sorting operations.

Consider the cardinality of the clustering key:




```
CREATE TABLE T1 (C1 date, c2 string, c3 number)
CLUSTER BY (C1, C2);
```

```
CREATE TABLE T1 (C1 date, c2 string, c3 number)
CLUSTER BY (MONTH(C1), SUBSTRING(C2,0,10));
```

```
ALTER TABLE T1 CLUSTER BY (C1, C3);
```

```
ALTER TABLE T2 CLUSTER BY (SUBSTRING(C2,5,10),
MONTH(C1));
```

Reclustering and Clustering Cost

As DML operations are performed on a clustered table, the data in the table might become less clustered.

Reclustering is a background process which transparently reorganizes data in the micro-partitions by the clustering key.

Initial clustering and subsequent reclustering operations consume compute & storage credits.

COMPUTE STORAGE

Clustering is recommended for large tables which do not frequently change and are frequently queried.

Search Optimization

Search Optimization Service

Search optimization service is a table level property aimed at improving the performance of selective point lookup queries.

1

```
SELECT NAME, ADDRESS FROM USERS
WHERE USER_EMAIL = 'semper.google.edu';
```

2

```
SELECT NAME, ADDRESS FROM USERS
WHERE USER_ID IN (4,5);
```

USER_ID	USER_NAME	USER_ADDRESS	USER_EMAIL
1	Duff Joisce	81 Mandrake Center	djoisce0@nasa.gov
2	Ira Downing	33214 Barnett Junction	idowning1@trellian.com
3	Alis Litel	9259 Russell Point	semper.google.edu
4	Cory Calderon	9266 New Castle Hill	ccalderon3@nydailynews.com
5	Pearl Denyuk	499 Thierer Hill	pdenyuk4@si.edu

The search optimization service is an enterprise edition feature.

Search Optimization Service

A background process creates and maintains a search access path to enable search optimization.

```
ALTER TABLE MY_TABLE ADD SEARCH OPTIMIZATION;
```

```
ALTER TABLE MY_TABLE DROP SEARCH OPTIMIZATION;
```

```
SHOW TABLES LIKE '%MY_TABLE%';
```

The access path data structure requires space for each table on which search optimization is enabled. The larger the table, the larger the access path storage costs.

10 Snowflake credits per Snowflake-managed compute hour

5 Snowflake credits per Cloud Services compute hour