**Snowflake:**

* built from scratch, optimized from cloud, storage and compute is decoupled.
* Software as a service: No software, infrastructure or upgrade to manage.
* Pay only for used compute and storage.
* Scalable: virtual warehouse can scale and storage scale independently.

**History:**

* **Founded in 2002**
* **Came out to stealth mode in 2014**
* **General availability in 2015 for 80 customers.**

**Snowflake Architecture:**

**Snowflake Edition:**

There are 4 edition of snowflake.

1. **Standard Edition:**

Standard Edition is our introductory level offering, providing full, unlimited access to all of Snowflake’s standard features. It provides a strong balance between features, level of support, and cost.

1. **Enterprise Edition:**

Enterprise Edition provides all the features and services of Standard Edition, with additional features designed specifically for the needs of large-scale enterprises and organizations.

1. **Business Critical Edition:**

Business Critical Edition, formerly known as Enterprise for Sensitive Data (ESD), offers even higher levels of data protection to support the needs of organizations with extremely sensitive data, particularly PHI data that must comply with HIPAA and HITRUST CSF regulations

It includes all the features and services of Enterprise Edition, with the addition of enhanced security and data protection. In addition, database failover/failback adds support for business continuity and disaster recovery.

1. **Virtual Private Snowflake (VPS):**

Virtual Private Snowflake offers our highest level of security for organizations that have the strictest requirements, such as financial institutions and any other large enterprises that collect, analyze, and share highly sensitive data.

It includes all the features and services of Business Critical Edition, but in a completely separate Snowflake environment, isolated from all other Snowflake accounts (i.e. VPS accounts do not share any resources with accounts outside the VPS). However, you may choose to enable data sharing with non-VPS customers by contacting Snowflake Support to enable listing auto-fulfillment.

**Feature/Matrx:**

**Release Management:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature/Service | Standard | Enterprise | Business Critical | VPS |
| 24-hour [early access to weekly new releases](https://docs.snowflake.com/en/user-guide/intro-releases.html), which can be used for additional testing/validation before each release is deployed to your production accounts. |  | yes | yes | yes |

**See doc for more detail:**[**https://docs.snowflake.com/en/user-guide/intro-editions#overview-of-editions**](https://docs.snowflake.com/en/user-guide/intro-editions#overview-of-editions)

**Sign up for Free Trial:**

* Duration 30 days
* $400 credits to use
* All functionality

**Snowflake Architecture:**

 It is a hybrid architecture.

Create ware house:

CREATE WAREHOUSE ETL\_WH WITH

WAREHOUSE\_SIZE = 'SMALL'

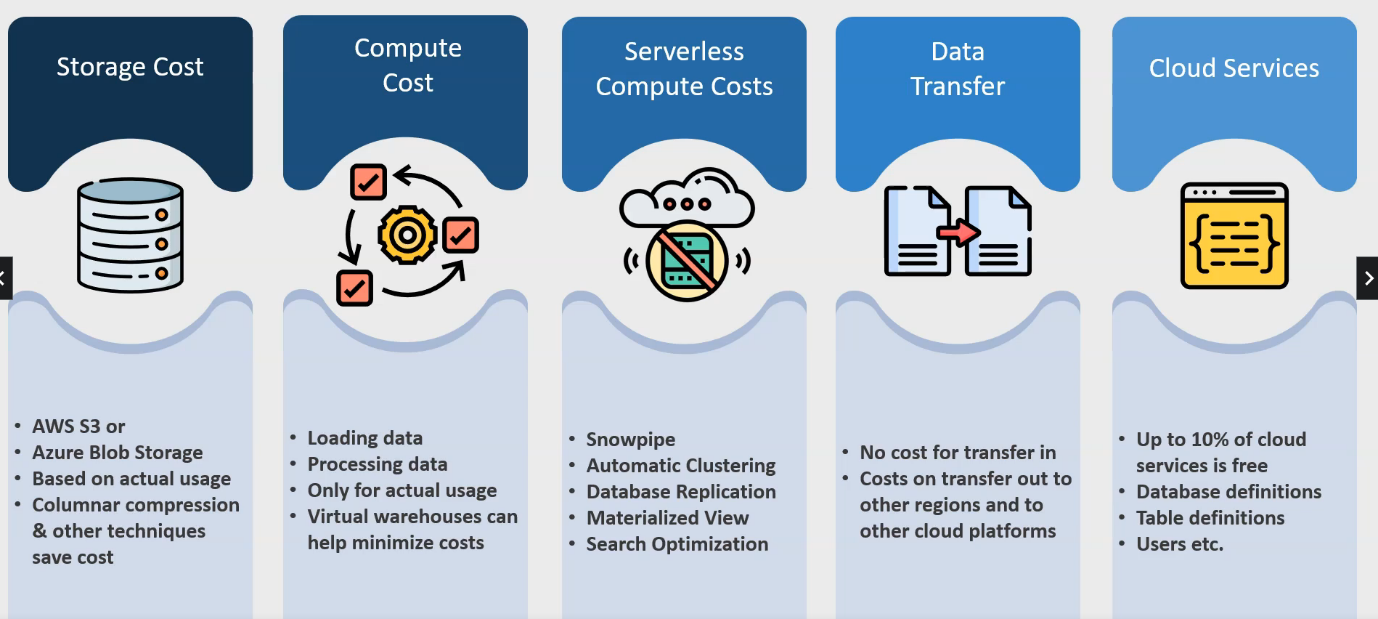
WAREHOUSE\_TYPE = 'STANDARD'

AUTO\_SUSPEND = 300

AUTO\_RESUME = TRUE

**Storage layer: Micro partition:**

* 50-500 MB of uncompressed data per micro-partition.
* Micro partitions are added in the order of data is added to table.
* Micro partition are immutable, This means that new partition will be created for newly inserted data.
* And update to data will result in existing partition being dropped and new partition is created with updated data.
* Snowflake also store metadata of micro partition:
  + Range of values of each column in the micro partition.
  + Count of distinct values for each columns.
  + Additional meta data for optimization
* Within each micro partitions columns are stored in columnar format.
* Columnar values are compressed with-in each partitioned.



**Summary:**

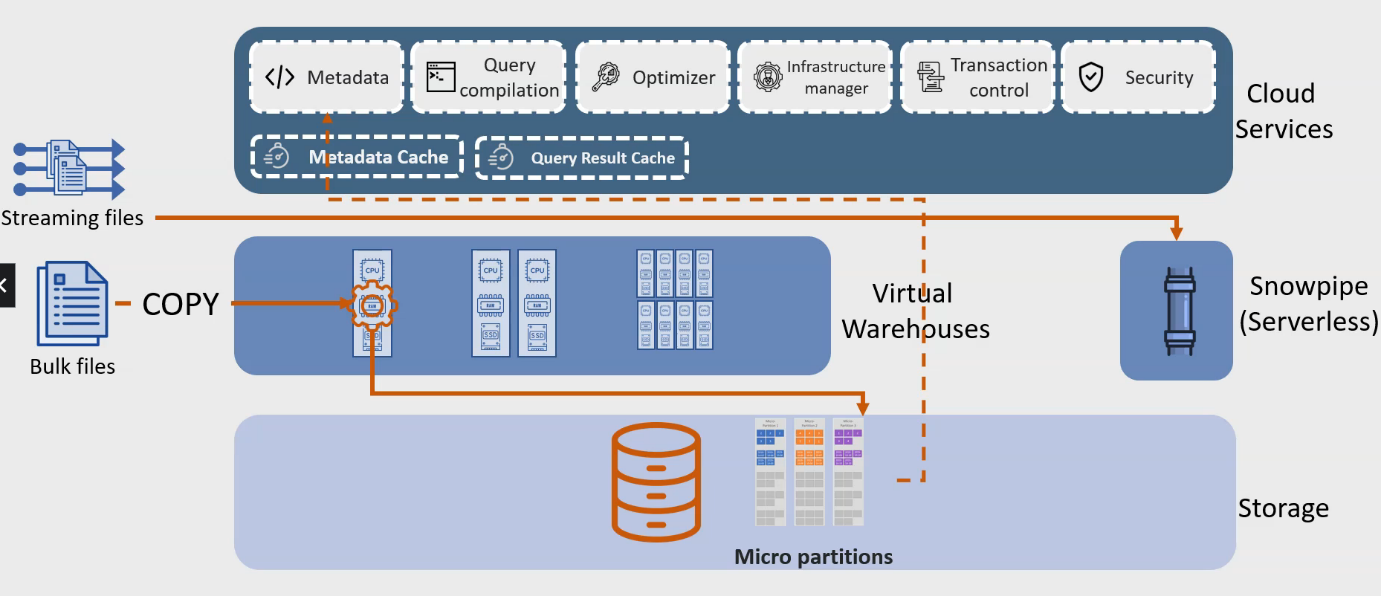
* Snowflake utilizes a hybrid architecture that combines shared nothing and shared-disk architecture.
* Snowflake stores data in a shared place, but processes it by using multiple compute clusters, also known as Virtual Warehouse.
* Snowflake architecture consists of three primary layers
  + The shared storage layer.
  + The compute layer
  + The cloud service layer.
* The cloud service layer is the brain of the whole operation. It is the entry point to a Snowflake system and provides
  + Security
  + Login & user management.
  + Transaction control.
  + Query planning & optimization
  + Metadata management.
* Snowflake uses cloud storage to store data in a proprietary format using micro-partitions:
  + Each micro-partition may hold 50-500MB of uncompressed data.
  + Snowflake chooses the compression methods for each column inside each division.
  + Because micro-partitions are immutable, insert, updates and deletes require the creation of new micro-partitions.
* Virtual Warehouses are on-demand computing clusters for query processing and data loading.
  + Each virtual warehouse access the same shared data.
  + Virtual warehouses size range from a single node X-Small instance to a 512 nodes 6X-large clusters.
  + Virtual warehouses can be suspended until a query load is available.
  + A virtual warehouse’s size can be scaled up or down.
  + Multi-clusters virtual warehouses allow for dynamic workload scalability.

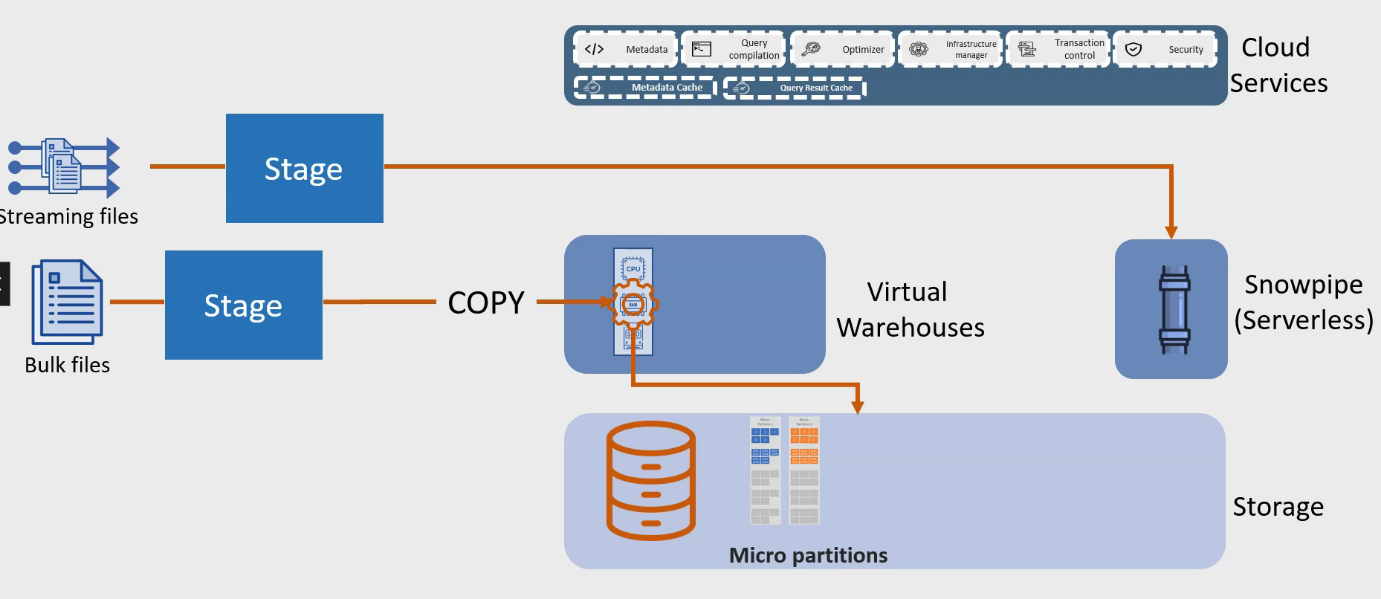
**Snowflake Web UI:**

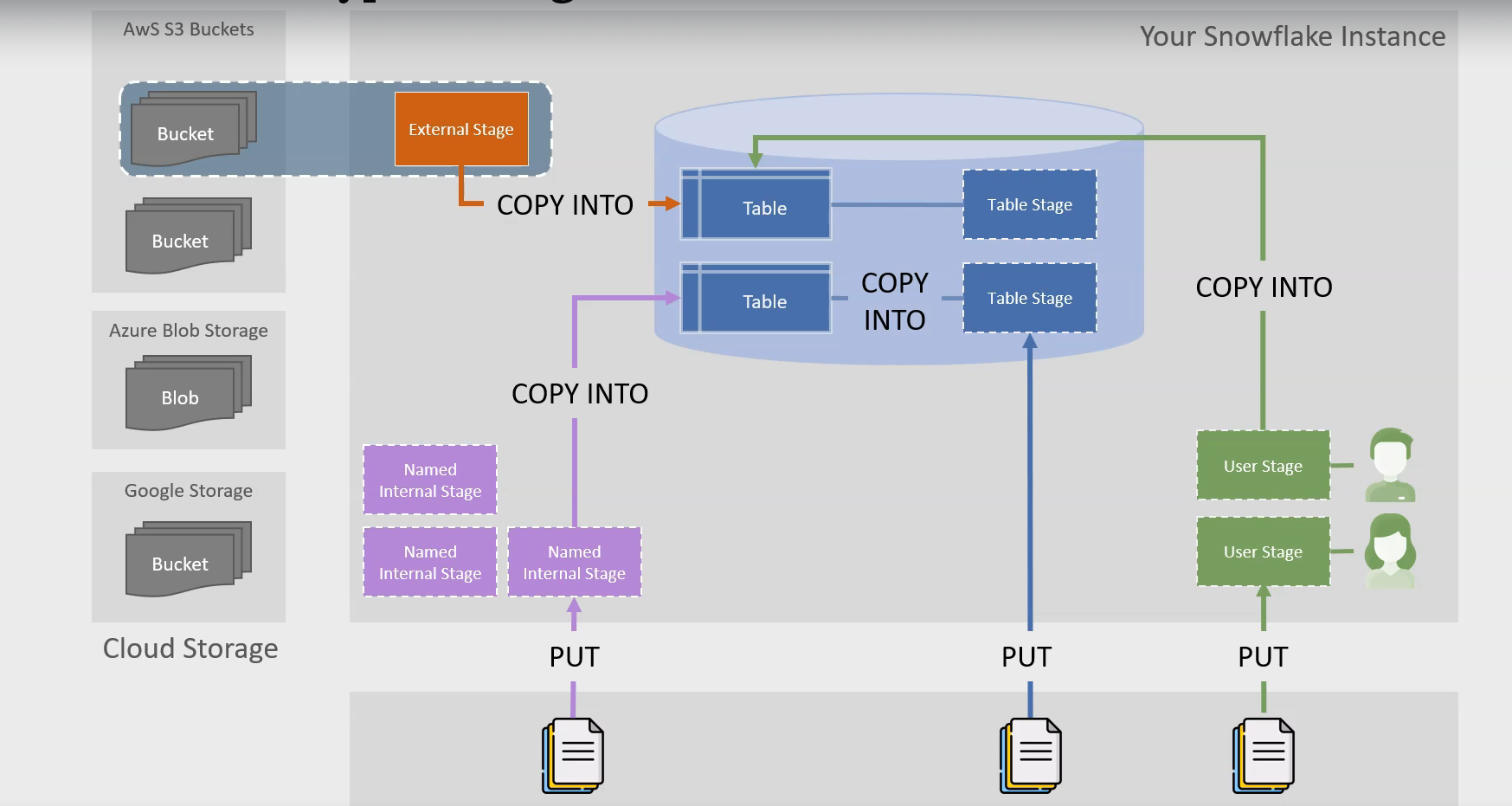
**Data Loading:**

**Data Loading in Snowflake:**

* Over of Data loading in Snowflake.
* Data loading with External stage
* Internal Stage for data loading
* Semi-structure Data support in Snowflake.
* Unloading(exporting) data from Snowflake
* Basic data transformation
* Snowpipe & continues data loading.







External Stage:

* An external stage points to cloud storage location like AWS s3 bucket, Azure Blob, etc.
* External stage definition include
  + Path to cloud storage location
  + Authentication details for connecting to cloud location
  + A key for decrypting data if data is encrypted in cloud storage
  + A file format to describe file structure.

Internal Stage:

These stages are managed by snowflake and reside in snowflake environment.

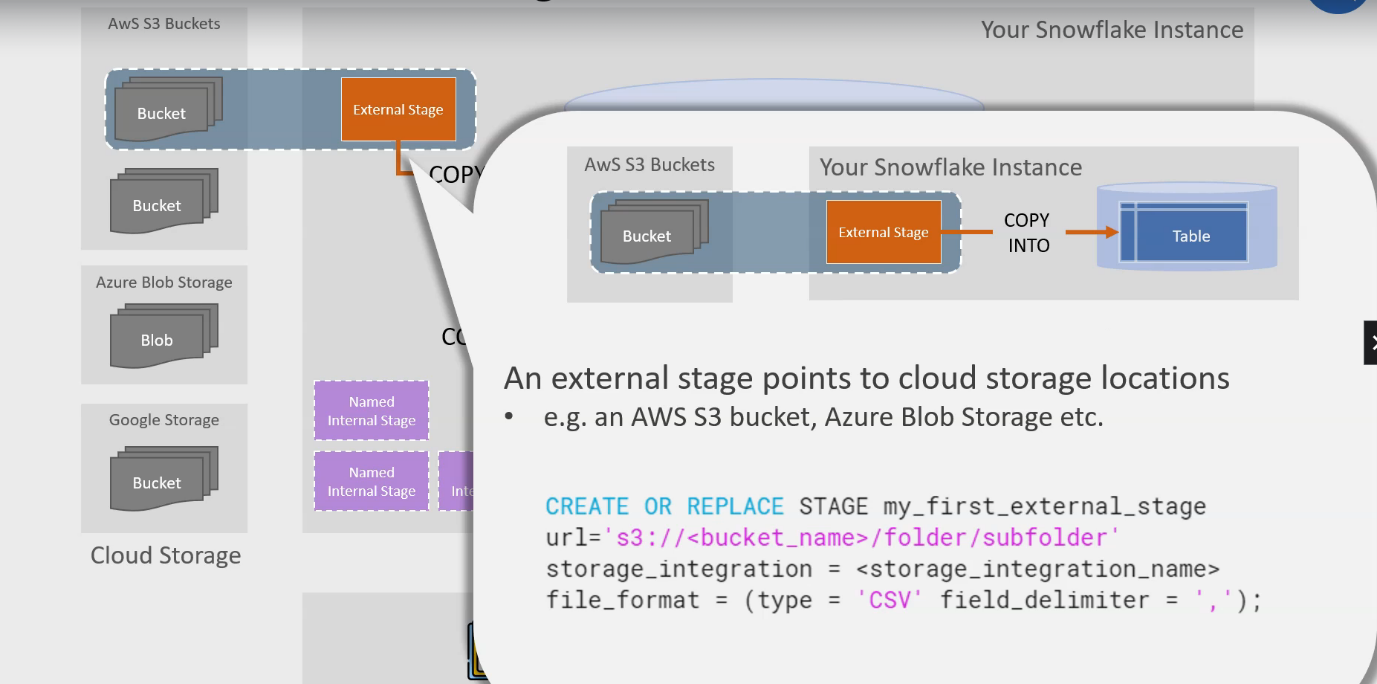
* Table stage:
  + A table stage can not be created and dropped. They are automatically created and available for a tables’ lifetime.
  + They share table’s name and can be referenced using @tablename syntax.
  + Table stage do not support file format options. The copy command must include these arguments.
  + Table stage do not allow data transformation while loading data.
  + They only load into single tables.
* Named stage:
  + A named internal stage is a database objects that can be created, modified and dropped. Compare to table and user stage, they are more flexible. They can be shared and can be used to load multiple files and multiple table.
* User stage.
  + A user stage cannot be created or dropped. They are automatically created and available for user lifetime.
  + They can be referenced using @ syntax.
  + User stage does not support file format options. The copy command must include these arguments.

**External Table:**

* External tablesallow you to create tables on data stored external to snowflake.
  + Definition in Snowflake metadata
  + Data in external location
* External table functionality lets you query an external table like regular table.
  + Join with other tables.
  + Create views on top of external tables.
* External tables are read-only table since they point to external storage.
  + DML operation can not be performed on them.
* External table data does not contribute to Snowflake storage costs.
* External table’s definition contains column information and file format information.
* Materialize vie can be created on top of external table.
  + Materialized view on external tables do not refresh automatically.

**Unloading the data:**

* Unloading data or exporting data is almost the same as loading.
* It uses the same COPY mechanism and the concepts of stages.
* Unload to different file formats.
* Compress & encrypts while unloading.



**Supported File Formats for data unloading:**

* Simple delimited files
* CSV, TSV, any other separator character.
* Parquet
* Json
  + ONLY NDJSON( New Line delimited Json)

**Compression:**

* Automatically compressed to gzip
* Can change compression mechanism
* Can disable compression if require.

**Encryption:**

* Automatically encrypted when exporting to an internal stage.
* Not automatically encrypted when export to external stage.

**Single vs Multiple files:**

* Exported to multiple files by default.
* Can be changed to a single file.

Basic data Transformation during **COPY** command:

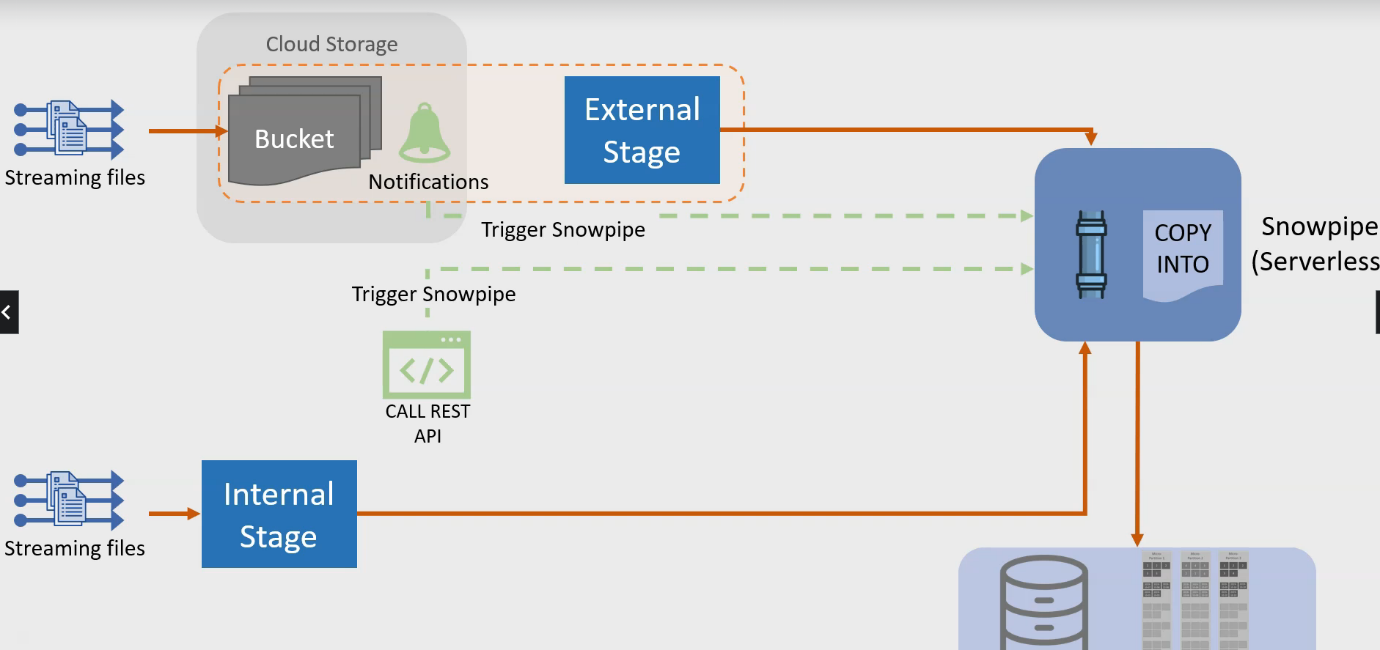
* Omit column or columns.
* Change the order of columns.
* Cast columns into specific data types.
* Truncate

Transformation not supported during **COPY**

* Joins
* Filter i.e WHERE clause
* Aggregation function.
* Flatten function also not supported while using Copy command.

**Snowpipe**:

* Continuous loading of data.
* Data is loaded within minutes of arriving in a stage.
* Serverless
* Managed by Snowflake
* Scale up and scaled-down automatically as needed.
* Does not use user-managed virtual warehouses.
* Billed separately from the virtual warehouse cost



**Data Protection**

Time Travel:

* Travel back to a point in time
* Travel back to a point before a query
* Un-drop objects
* Undo accidental Changes.

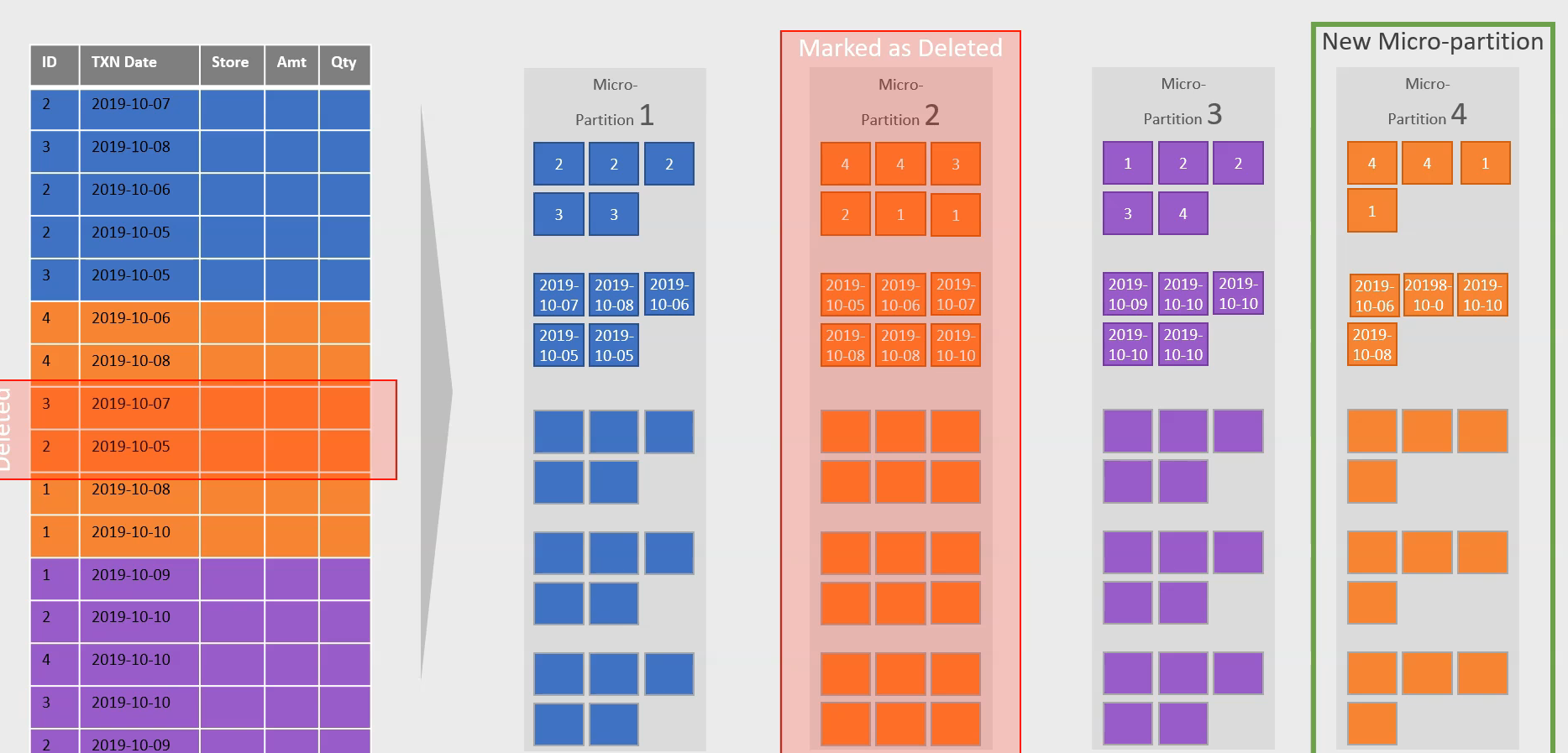
Which Objects:

* Database
* Schema
* Tables

Some use Case:

* Query data in past
* Restore dropped database, tables & Schema
* Combine time travel with cloning.

**How Time Travel work in Snowflake.**



**Time Travel SQL Extension:**

* **AT** and **BEFORE**Clause
* **UNDROP** statement

The AT clause can be used in three different ways.

**SELECT FROM AT ({TIMESTAMP} > =<timestamp> | OFFSET => <time\_difference> | STATEMENT => <ID> } …..**

* Specify a timestamp
* Specify an offset from the current time
* Specify an SQL statement id

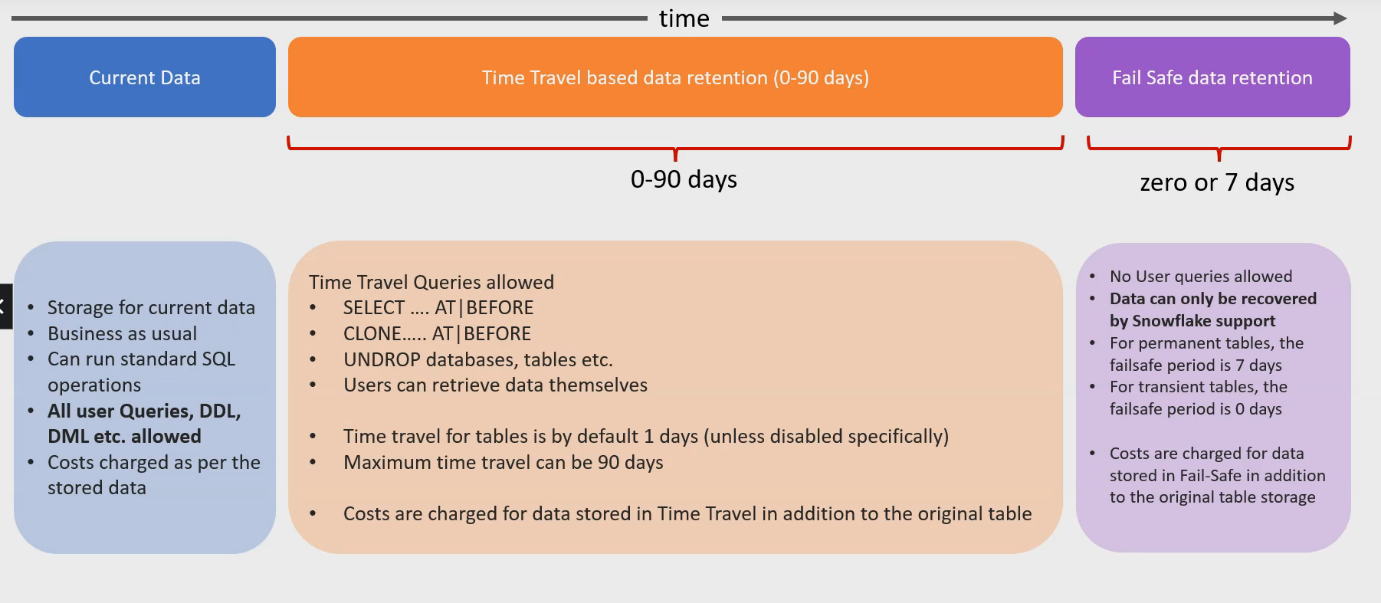
**The BEFORE clause can also be used in three different ways:**

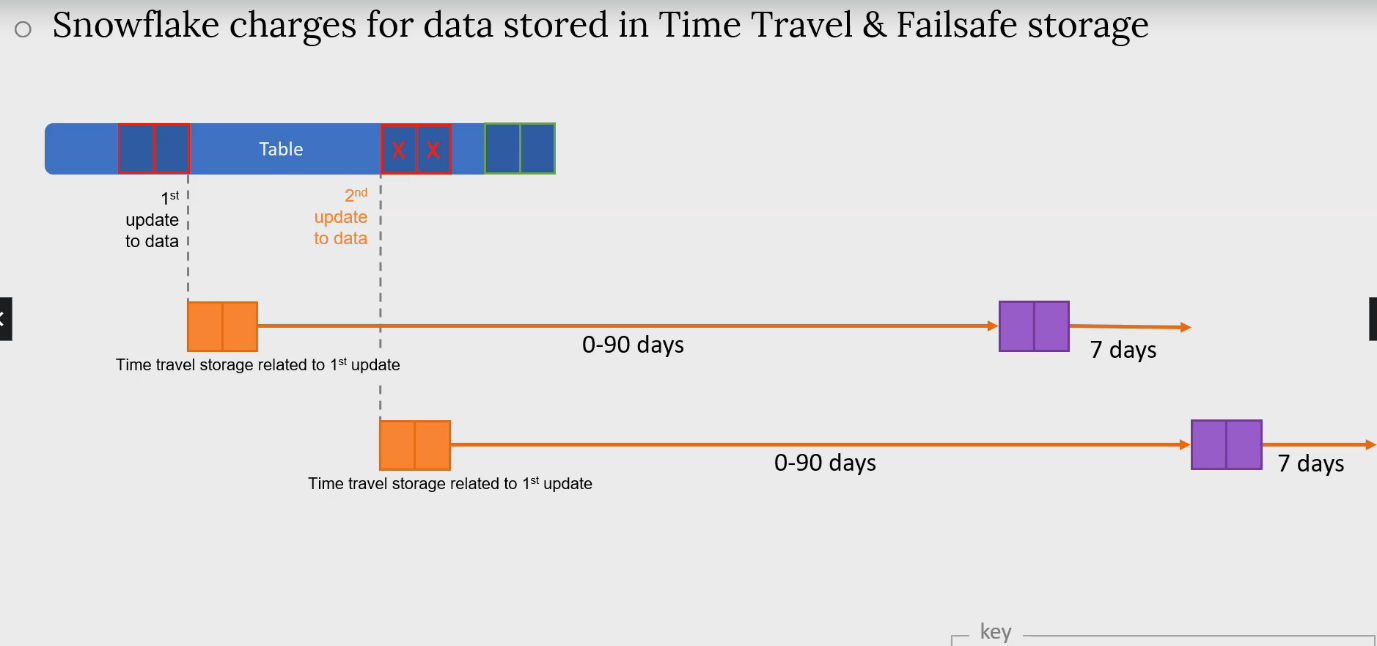
SELECT FROM BEFORE  ({TIMESTAMP} > =<timestamp> | OFFSET => <time\_difference> | STATEMENT => <ID> } …..

* Specify a timestamp
* Specify an offset from the current time
* Specify an SQL statement id

**Failsafe Storage:**

* Failsafe provides a 7-day period during which data is recoverable by snowflake.
* The Failsafe period start immediately after time travel period ends
* The Failsafe data can be recovered only by snowflake support.
* Provide a way to recover lost data in a relatively quick manner.
* Snowflake’s redundant & cheap data storage architecture reduce the need for backup, there still can be scenario for data loss.
* Failsafe storage provides a safety cushion in such scenario and allow you to recover data 7 days after your time travel history period is gone.

****

****

**Types of tables in Snowflake:**

**Temporary**:

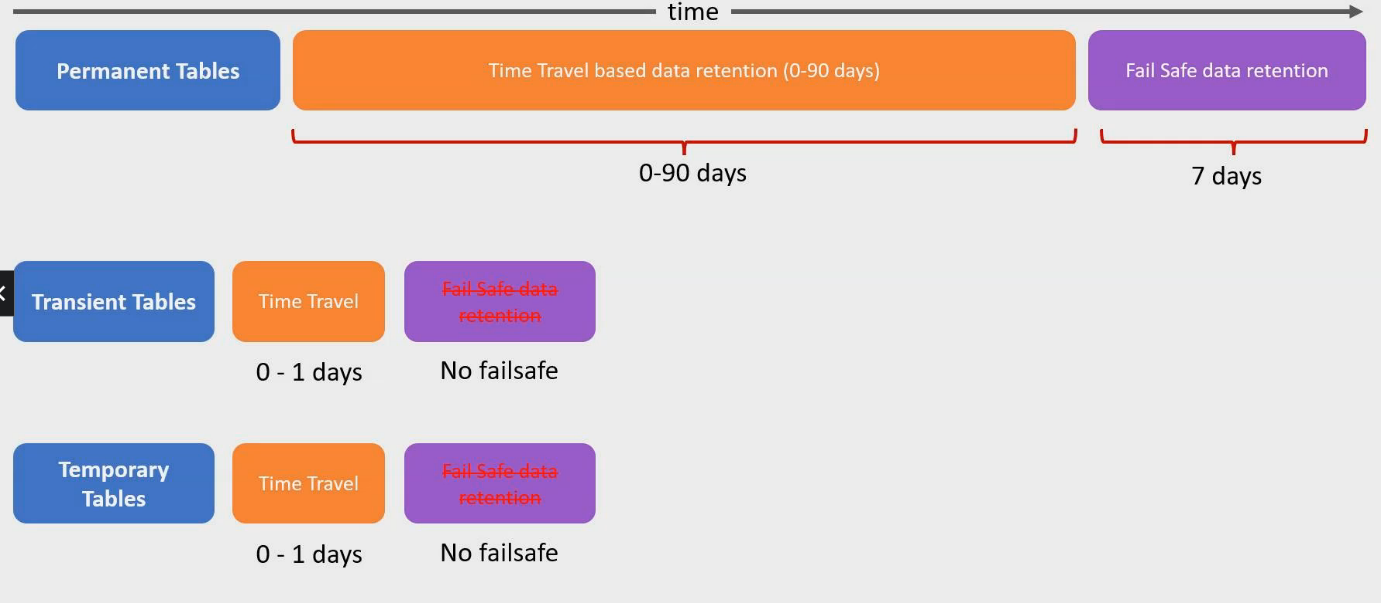
* Exist only for the lifetime of a session
* They are not visible to other session and are removed immediately once the session ends.

**Transient**:

* Similar to temporary tables, but they persist between session. They are designed to hold temporary data that needs to be accessed across session e.g ETL job

Permanent:

* This is the default type of table in snowflake , if you don’t specify the type a table is created permanent.



**Summary:**

* **Time Travel allows to query, retrieve and restore data from tables.**
  + Time travel is available in all snowflake editions.
  + Time Travel duration may be range from 0 to 90 days.
  + The standard edition allow only up to 1 day for Travel.
  + Enterprise edition and above allow up to 90 days of Time travel.
* **Undrop allows users to restore dropped table, schemas and databases.**
  + Tables, Schema and databases are not immediately deleted physically but instead marked as deleted.
  + This allows snowflake to restore the table extremely fast using the UNDROP command.

**Fail-safe:**

* Further seven day of data protection after Time Travel period has finished.
* Only Snowflake support can recover data.
* Can not be turned off or customized.
* Transient and Temporary table do not have fail safe storage.
* Permanent table always have fixed seven days of Fail-safe storage.

**Snowflake Cloning or zero Copy cloning**

**Zero Copy Cloning:**

* Metadata Operation
* Perform by Cloud Service Layer using micro-partition metadata.
* No Physical data movement
* No Extra storage requirement
* Generally a very fast Operation
* Cloning can be used with Time Travel.

**CLONE Command**:

* Supported Objects:
  + Database, Schema & Tables.
  + Stream
  + Stage, File Format, Sequence, Tasks.

**Cloning Caveats:**

* **Cloning is fast:**
  + But make take some time for very large tables.
* Storage:
  + Cloned object does not contribute to storage unless data is changed in source or cloned.
* Clone whole database or schema:
  + All children are cloned
    - (except named internal stages & snowpipe that reference any named internal stage)
* Access Rights
  + Cloned objects do not inherit permission
  + Child objects(in case of database/schema cloned) inherit permission
* Can’t be cloned
  + External tables
  + Named internal stages.

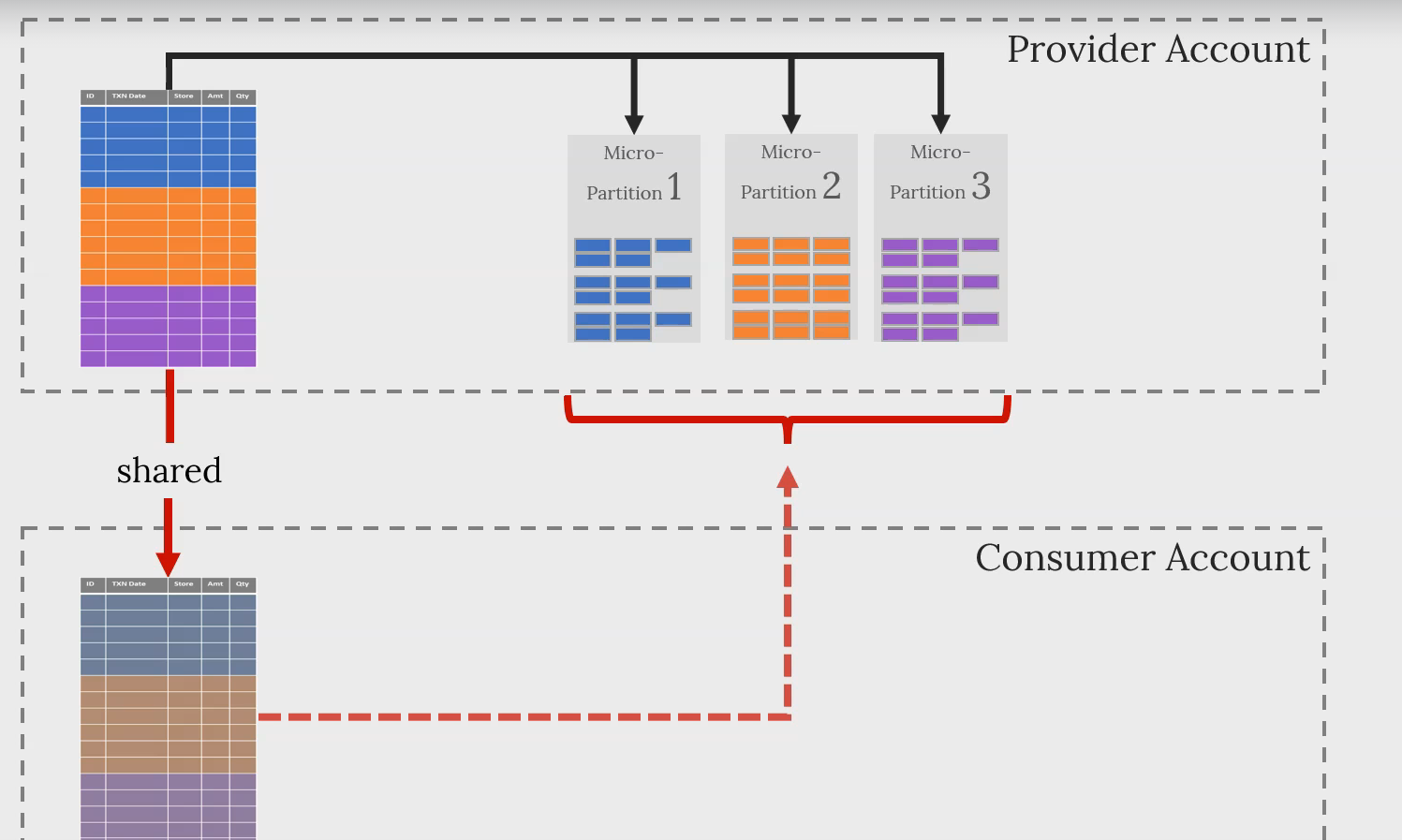
**SECURE DATA**

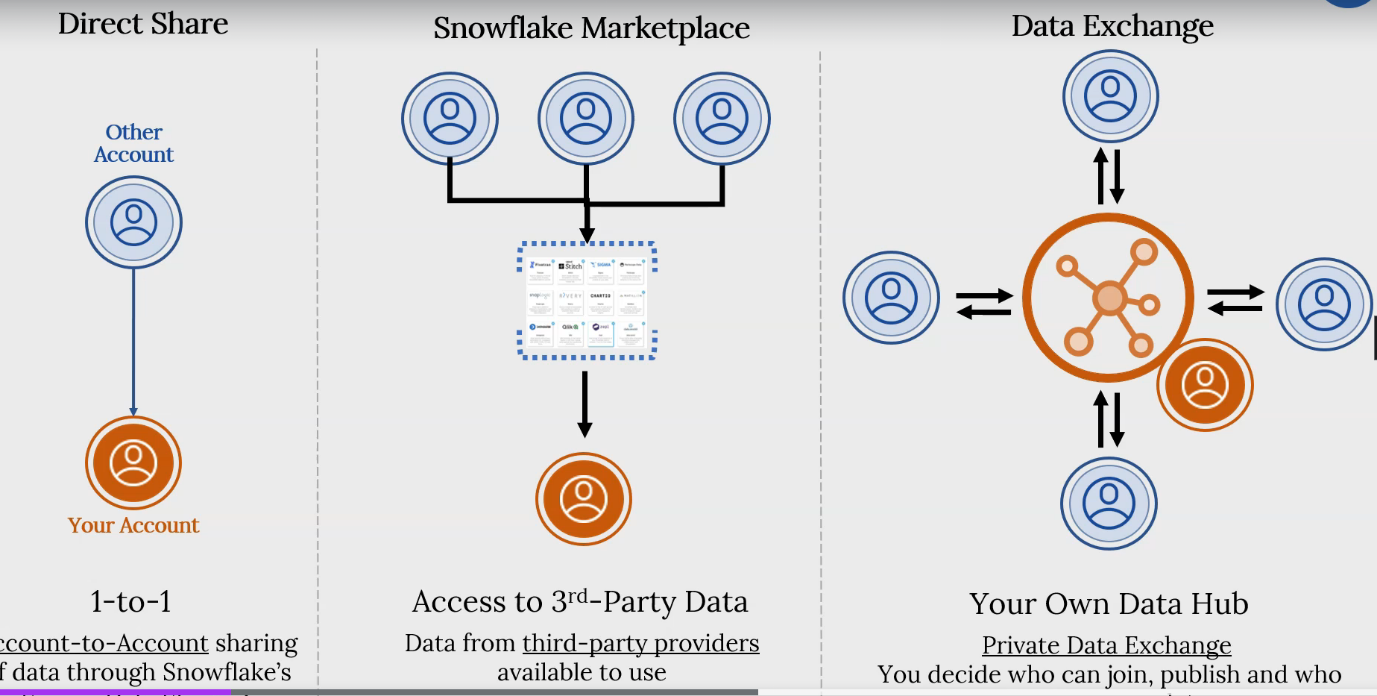
**Secure Data Sharing:**

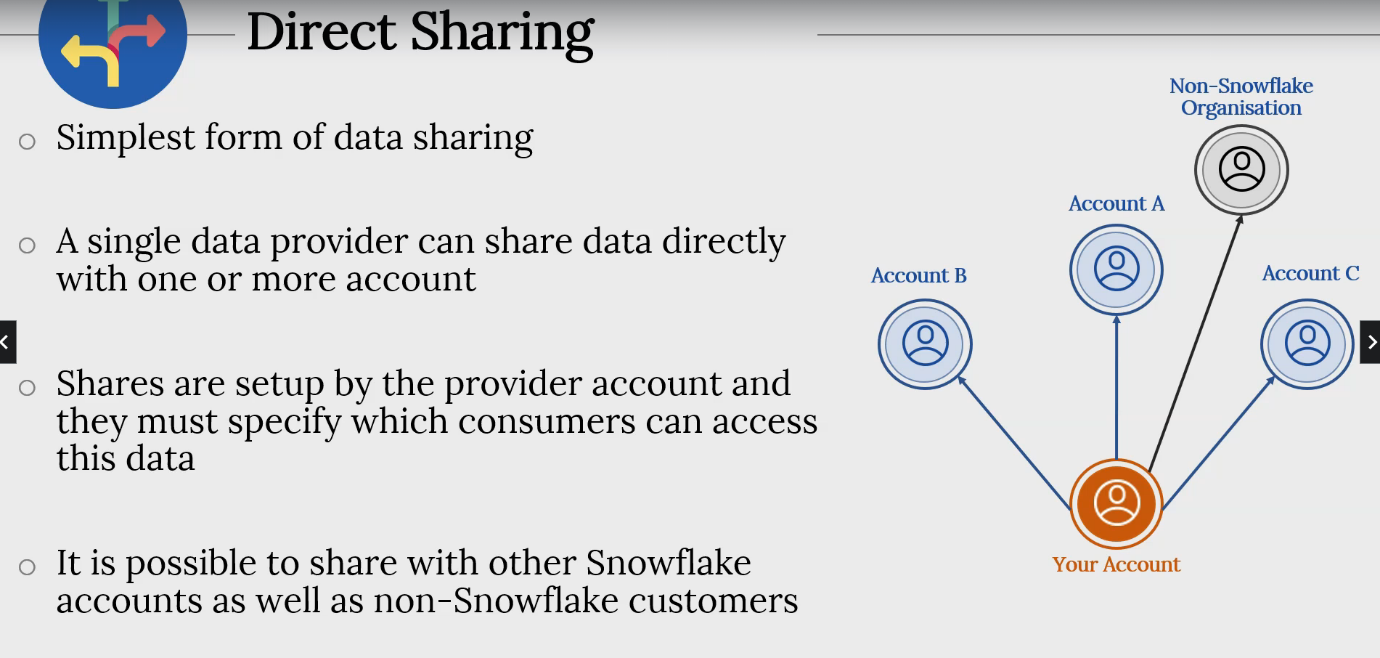
* Meta data Operations
* Performed by Cloud Service layer using micro-partition metadata.
* No Physical data movement.
* Generally, a very fast operation
* Provider can stop sharing anytime.
* No additional storage requirements

Key terms:

* Data Provider
* Share
* Data Consumer





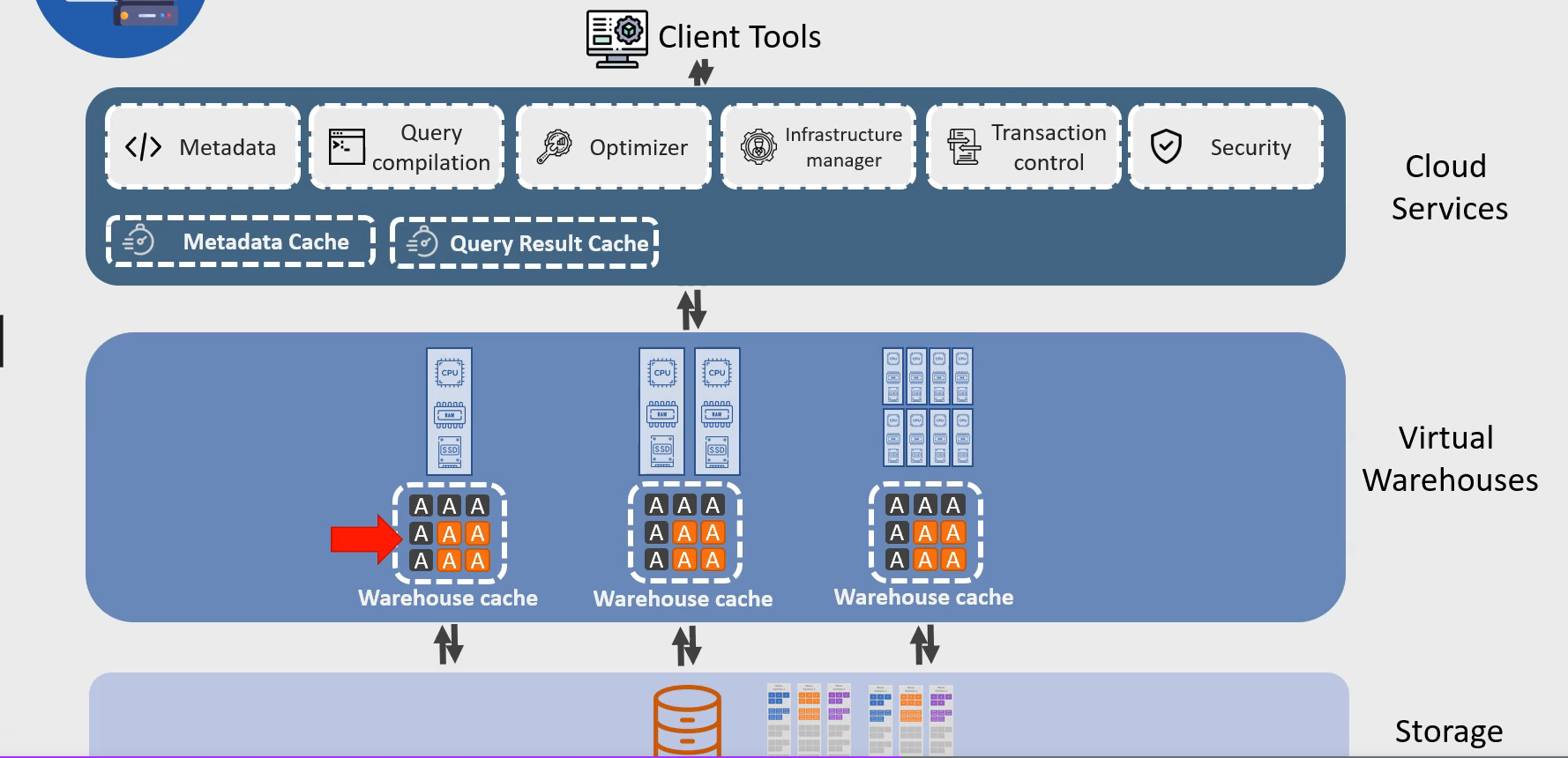


**Direst Sharing Steps:**

* **Data Provider:**
  + Create a Share Object
  + Add tables/objects(schema ,database)  to the share.
  + Add one or more Consumer accounts
* **Data Consumer**
  + **Read only database:** create a database from the share.
  + **Access Control:**Configure security for new database.
  + **Query Shared data:**Using own compute.

**Snowflake Performance Feature and Optimization**

* Caching
  + Metadata cache
  + Query result cache
  + Virtual Warehouse cache
* Scaling up
* Scaling Down
* Scaling Out – multi-cluster virtual warehouse
* Improved Partition pruning via clustering keys
* Materialized Views
* Search Optimization



**Query Profile**:

* Provides a graphical representation of a query’s execution.
* Provides execution detail for a query.
* Statistics and detail for each step
* Statistics and detail for the whole query.

**Query result Cache in Action:**

* Execute a query using sample data to create a query result cache.
* Run the same query again to check if the query result cache is used.
* Set the **USE\_CACHED\_RESULT** parameter to disable the query result cache.

**Metadata Cache:**

* Snowflake stores metadata about new micro-partitions, storing maximum and minimum values for each column for each micro-partition.
* It also stores the count of distinct values for each column in the metadata,
* For queries such a COUNT, MIN or MAX snowflake does not need to read the table but can pull information from the metadata cache.
* Query MAX or MIN for string type data, then it will fetch from table.

Snowflake maintains metadata for each table, which includes statistics for each micro-partition. The statistics include information such as the number of rows for each micro-partition. Thus, Snowflake can answer a simple COUNT query using the metadata by summing up the number of rows for each micro-partition maintained in the statistics.

**Virtual Warehouse cache:**

* Virtual Warehouses maintain a cache of table data that they access while processing queries.
* Warehouse cache may be re-used if subsequent queries can read the required data from cache rather then from the table.
* The Warehouse cache is removed if the virtual warehouse is suspended
* The size of the cache relates to the size of virtual warehouse , bigger the virtual warehouse, larger the cache.

**After a virtual warehouse is suspended and resumed, a new query can not take advantage of the warehouse cache. What is the reason behind this?**

When a virtual warehouse is suspended, all its compute and temporary storage resources related are released or de-provisioned. Thus, the cache stored on the temporary storage gets purged and is no longer available when the virtual warehouse is resumed. After a virtual warehouse is resumed, the cache gets rebuilt over time as queries execute.

**Automatic Clustering:**

* Automatic Clustering
  + Snowflake service responsible for re-clustering tables.
* Automatic Clustering redistribute data according to cluster key
  + Redistribute only if determines that the table will benefits from re-clustering.
* Automatic Clustering is Serverless
  + Does not use a virtual warehouse.
  + Uses Snowflake managed CPU, RAM etc.
  + You are charged for re-clustering costs as it involves compute
* Re-clustering also involves additional storage costs
  + The original micro-partitions are kept for fail-safe and time-travel process.

**Clustering depth:**

**Snowflake stores metadata for clustering**

* Number of micro-partitions in a table
* Number of micro- partitions that overlaps with other micro-partitions(specific column)
* The depth of the overlap

**Clustering Depth:**

* calculate for specific columns
* Starts at 1 and can be a much larger number
* A smaller average depth indicates a well-clustered table.

**Inbuilt Function:**

* SYSTEM$CLUSTERING\_DEPTH
* SYSTEM$CLUSTERING\_INFORMATION
* Monitor clustering health for a table.
* Identify if a table will benefit from an explicit clustering key.
* Clustering depth for an unpopulated table is zero.

**Scaling up and scaling Down:**

* Scale up
  + When query complexity increase
* Scale Down
  + When query complexity descrease
* Multi-clustering virtual warehouses
  + When there is high query concurrency , ie. Large number of simultaneously queries, and
  + The workload fluctuations is un-predictable.

**Scaling Up:**

* A virtual warehouse can be resized as per requirements.
* The resize can be to a larger size in which case new nodes are provisioned
* The charge for the new size takes effect only after all nodes in the new size have been provisioned.
* Currently running queries can not take advantages of increased size.
* Only new queries will be able to take advantage of the increase clusters size.

**Scaling Down:**

* A virtual warehouse can be resized to a smaller size.
* Resizing to a smaller size result in Snowflake decommissioning necessary nodes.
* The resize down can only **complete** once **all active queries running** on that specific virtual warehouse have **completed**.

**Syntax:**

ALTER WAREHOUSE <WAREHOUSE\_NAME>

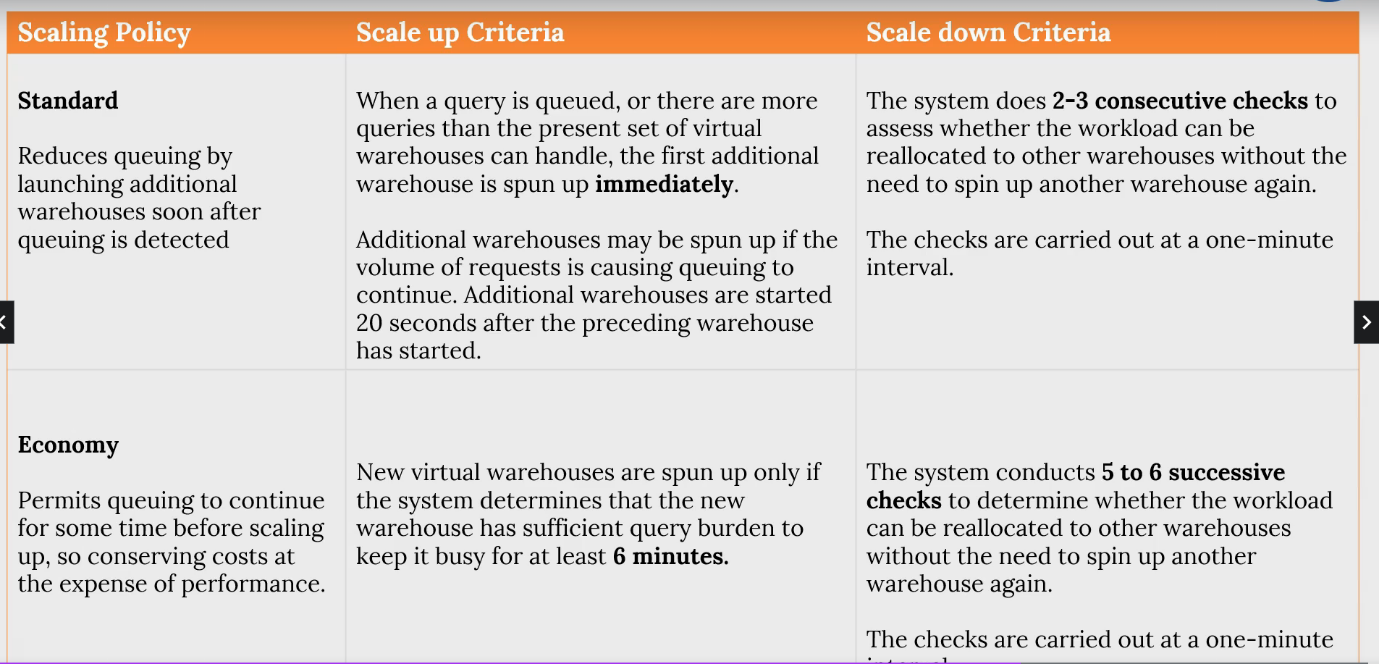
SET WAREHOUSE\_SIZE = XSMALL | SMALL | MEDIUM | LARGE |XLARGE | XXLARGE |XXXLARGE |X4LARGE | X5LARGE | X6LARGE

**Scaling out/ Multi-cluster virtual warehouse:**

* Typing virtual warehouse consists of a single cluster of compute resources.
* Queuing occurs if there are not enough compute resources.

**Multi-cluster virtual warehouse provides a solution:**

* When concurrent queries exceed the capacity of a single virtual warehouse
* Spin up new virtual warehouse automatically.
* Decommission when demand decreases.



**Materialized View:**

* Materialized Views store pre-computed results based on a SELECT query.
* Result are pre-computed and physically stored.
* Data in materialized Views are kept up-to date by a Snowflake service.
* The service ensures that data is always in synch with the base table.
* Snowflake transparently re-routes queries to the base table if the materialized view has not been synched yet.
* Materialized views can be helpful if
  + A query(or its variation) is executed frequently.
  + The query is complex and requires a significant amount of time/resource.
  + The query result is generally consistent and don’t change too often.

**Summary:**

**Metadata Cache:**

* Stored in cloud Service layer.
* Can fulfil COUNT, MIN, MAX etc, queries without requiring a virtual warehouse/ table scan

**Query Result cache:**

* Stores results from a query and re-uses for future matching queries.
* Results stored for 24 hours, unless re-used during that time in which case results are kept for another 24 hours and so on.
* Can be refreshed up-to 31 days after which it is discarded.

**Virtual Warehouse cache:**

* Local cache within a virtual warehouse cluster.
* Caches the columns and micro-partitioned accessd by each query.
* Once the cache is built, requests for columns and micro-partitions can be fulfilled from the virtual warehouse cache.

**Virtual Warehouse Scaling up:**

* Adds additional nodes.
* Handle more complex queries and workloads
* The charging for the new size beings only once all new nodes are provisioned.
* Existing queries are not impacted by the changed size, only new queries are affected.

**Virtual Warehouse Scaling Down:**

* Nodes are removed only when there is no active query.

**Multi-cluster virtual warehouse or scal out:**

* Handles increased user/query concurrency
* Manages query queuing.
* New Clusters are started with increased concurrency.
* Clusters are stopped when concurrency requirements decrease.
* Maximized Mode – all clustered started immediately.
* Auto-Scale Mode – Clustered are started/stopped as per concurrency demands.
  + **Standard:** prefer performance over cost.
  + Economy scaling: Prefer costs over queuing.

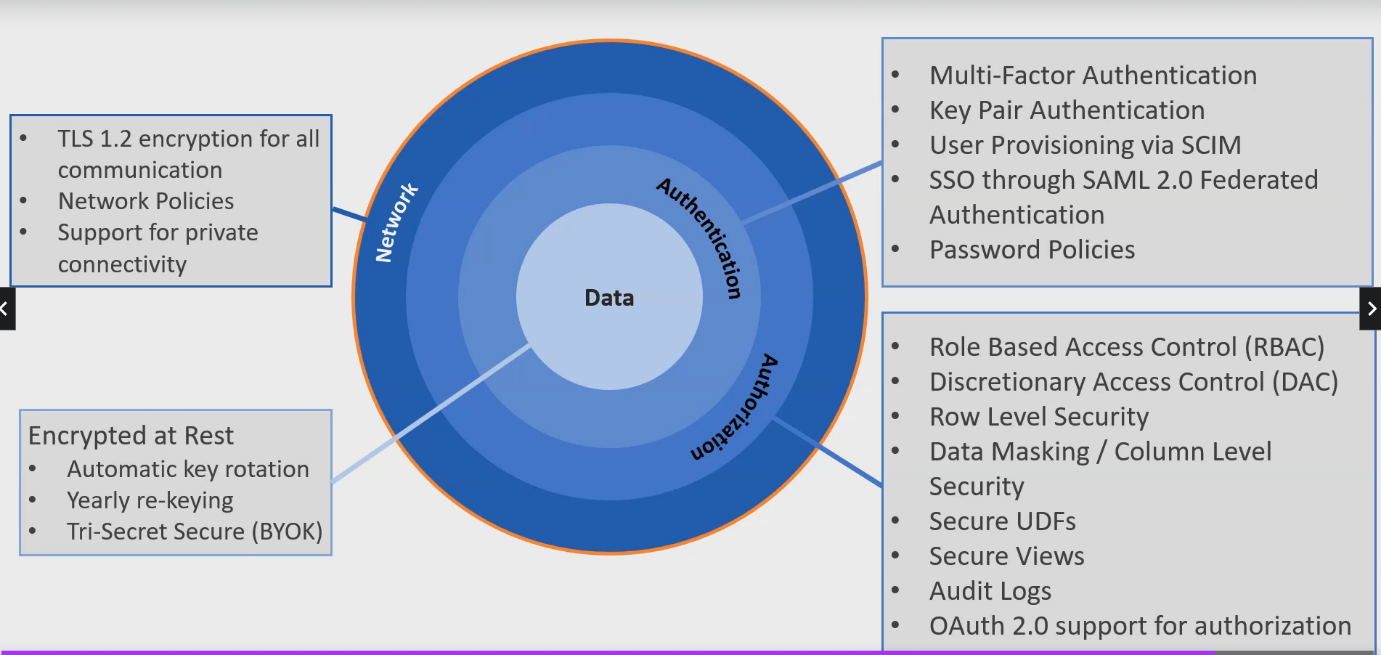
**Multi-cluster:**

* Snowflake table are automatically clustered.
* A custom clustering key can improve performance.
* Data is redistributed according to the clustering key- resulting in better partition pruning.
* Automatic Clustering is a snowflake-managed service that takes care of clustering and redistribution.
* Re-clustering uses Snowflake credits and uses extra storage(time-travel, fail-safe)

**Clustering depth:**

* Calculate for specific columns.
* Starts at 1 and can be a much larger number.
* A smaller average depth indicates a well-clusters table.

**Snowflake Security:**



**Data Encryption a rest:**

AES 256-bit encryption

* Encryption keys are managed by Snowflake
* Keys are rotated every 30 days.
  + Previously active keys retired, used only for decryption from that point.
  + New Keys added- used for encryption and decryption.
* Yearly re-keying:
  + Re-encrypt data protected by a retired encryption key.
  + Requires at least Enterprise Edition
  + Enable by account administrator(not active by default)
* Tri-Secret Secure
  + Additional level of security
  + Composite master key consisting of customer managed key & snowflake managed key
  + If customer managed key is revoked, snowflake can no longer decrypt data.
  + Require minimum of Business Critical edition.

**Multi-factor Authentication:**

* MFA is enabled for all account
* MFA is available in all Snowflake editions
  + So the minimum edition supporting MFA is Basic
* Snowflake recommends that users with ACCOUNTADMIN role should use MFA.
* An administrator can disable MFA for a user.
  + They must re-enrol if they require MFA capability.
  + ALTER USER and set DISABLE\_MFA= TRUE
* AN administrator can also temporarily disable MFA for a user
  + The MFA for that user re-enables after the specified time period has passed.
  + ALTER USER and MINS\_TO\_BYOASS\_MFA = <mins>
* MFA is supported by all Snowflake Client and coonnectorrs.
  + Snowflake Web UI
  + SnowSQL
  + Snowflake ODBC, JDBC drivers
  + Python and other connectors
* **Snowflake uses the push approach by default for clients and connectors.**
* Passcodes can be used on be command line for SnowSQL.
  + **–mfa-passcode <string> or –mfa-passcode-in-passcode**
* Passcodes can be provided in the connection string for drivers
  + Passcode=<passcode\_string> or a variation of passcode\_in\_password=True

**Key Pair Authentication:**

More secure alternative to username/password:

Uses private & public keys

* Private key is assigned to a user in Snowflake.
* Public Key is issued to the user for assisting in authentication.
* A user can have up-to tow public keys/
* Key-pair authentication is supported by all snowflake editions
* Key-pair authentication is supported by SnowSQL & all ther connectors.

**SSO/SAML 2.0 Federated Auth:**

Authenticate through an external identity provider.

* After the IdP has authenticated the user, the user does not require to provide username or password.
* Also known as Single Sing ON(SSO)

Snowflake Support SAML2.0 IdP:

* Native Support for Okta, ADFS.
* Support for Google G suite, Microsoft Azure AD, one login, ping Identity Ping ON

SSO/federated authentication is supported by all Snowflake editions.

**Authorisation:**

* Role Based Access Control(RBAC)
* **Discretionary Access Control (DAC)**
* Row Level Security
* Data Masking/Column Level
* Security
* Secure UDF
* Secure Viws
* Audit Logs
* OAuth2.0 support for authentication

**Role Based Access Control:**

* Snowflake access control is based on RBAC
  + Role Based Access Control
  + Rights given to roles, roles given to users.
* Snowflake also support DAC
  + Discretionary Access Control
  + Owning role can provide access to other roles.

**Column Level Security:**

**Dynamic Data Masking:**

* Dynamic mask data depending on user’s role.

**External Tokenization.**

* Tokenize data prior to loading into Snowflake
* Detokenize during query execution based on user’s role.

**Row level Security:**

* Return specific results based on the role.

**Secure View & Secure UDF:**

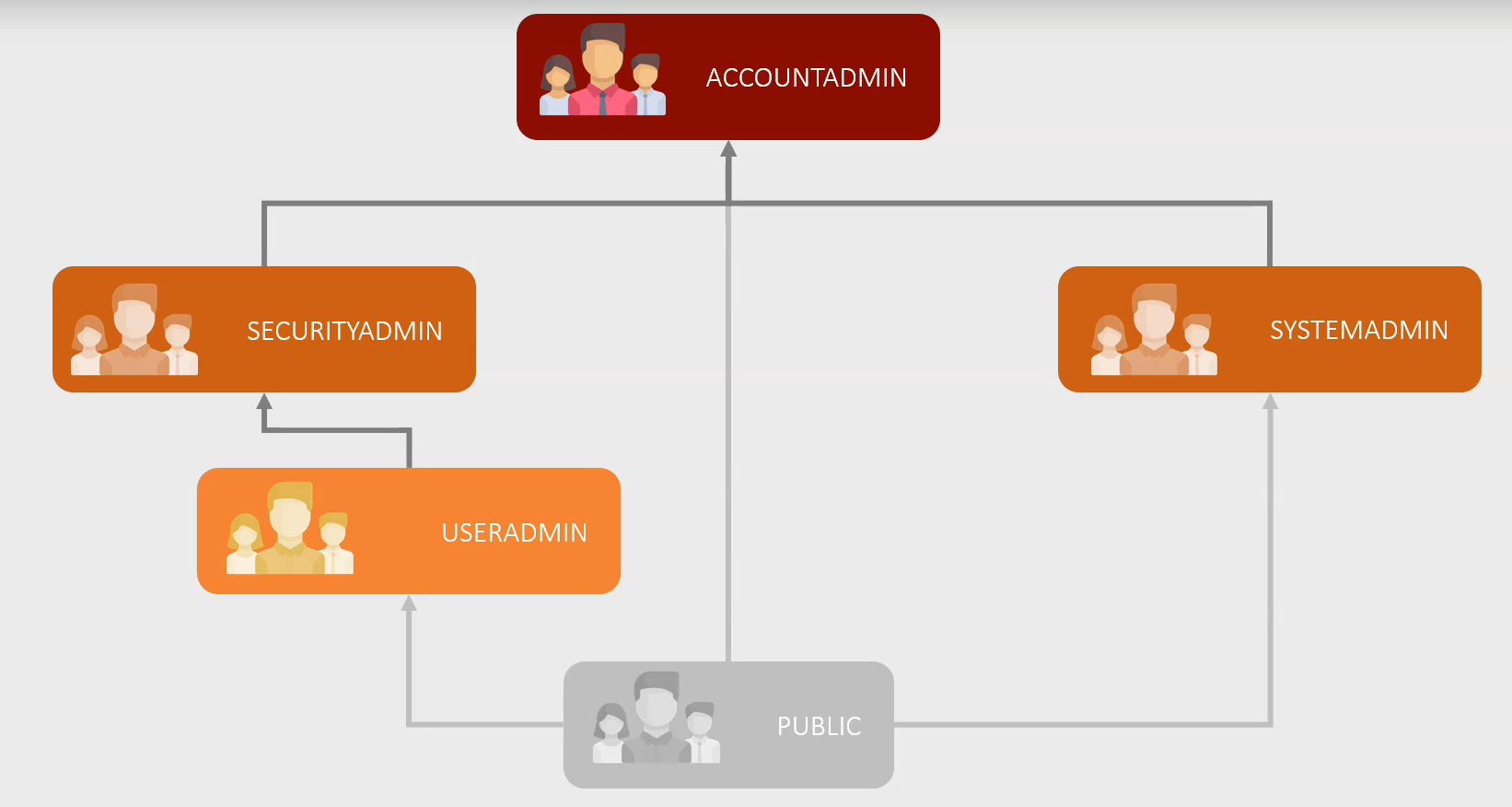
**Secure View:**

* Used for Data Privacy
* Hide the underlying data from the user.
* Use SECURE keyword to create

**Secure UDF:**

* Similar to Secure View
* Hide underlying data from the user.

**Snowflake Built in Role:**

****

**Network Policy:**

Currently support IPV4

**Summary:**

**Data Encryption at Rest:**

* Snowflake stores all customer data using AES-256 bit encryption.
* Snowflake managed keys are rotated every 30 days, and data is re-encrypted with new keys every year(yearly re-keying)
* With **Tri Secret Secure** (require Business Critical edition) you can further protect your data with combination of snowflake managed key and a customer-managed key.

**Authentication:**

* Snowflake supports key pair authentication using private and public keys.
* Snowflake also supports federated authentication (SSO) where users authenticate via an external SAML2.0 identity provider.
* Snowflake support SCIM, an open standard for automatic user and role provisioning based on identity provider data.
* Snowflake also enforce strict password rules.
* All snowflake edition support MFA or multi-factor authentication
* Administrators can disable or temporary disable MFA for user
* Snowflake Web UI, SNowSQL and all other connectors support MFA

**Authorization:**

* Snowflake’s access control system is based on the **RBAC or role-based access control model.**
* Snowflake also offer **DAC** or **discretionary access control Privileges** can be granted on **Securable** objects.
* The GRANT statement grants privileges while the REVOKE statement revokes them
* A **ROLE** is an entity to which privileges on other objects can be granted.
* The roles themselves can then be granted to others users or other roles. A role **hierarchy** is formed when a role is assigned to another role.
* Snowflake includes a small set of system roles.
* **ACCOUNTADMIN** is the full-access account administrator role.
* **USERADMIN** allows you to create USERS and ROLES.
* **SECURITYADMIN** inherits USERADMIN privileges and has the ability to handle object grants globally.
* Most item in snowflake system can be created and managed using **SYSADMIN**.
* **PUBLIC** is the least permissive role and it is assigned to everyone by default.
* When a role is assigned to another role, the higher role obtain all of the privileges of the lower role.
* Snowflake’s roles are arranged in a hierarchical order.

**Authorization:**

* Column-level security feature in Snowflake allows you to mask a column.
* **MASKING policies** are attached to column to enable masking.
* Two methods of masking:
* **1: Dynamic Data masking**
  + Data stored unmasked.
  + Show masked or unmasked data at execution time.
* **2: External tokenization**
  + Data stored in tokenized(encrypted manner)
  + Show masked or unmasked data at execution time
* Snowflake also offer row-level-security:
  + **Determine which** rows to return to users when they execute queries.
  + **Row-level access policies** define which rows should be returned using condition and functions.

**Network:**

* **Snowflake encrypts all data in transit using TLS1.2**
  + Applies to all communication and all methods of connectivity.
  + Web interface, SnowSQL and all the drivers.
* Administrator can define **network policies** to allow or deny **IPv4** addresses
* Network Policies have **allowed IP address & blocked IP address.**
* Snowflake applies the blocked list first if both lists are populated
* Snowflake supports private connectivity
  + Require Business Critical edition and above

**Extending Snowflake Functionality:**

User defined function can be created to create functionality:

Supported Language:

* SQL
* Java
* JavaScript
* Python

External function:

* External function can be written in language that standard UDFs do not support.
* They can also access functions and libraries that typical UDF cannot.
* They can of course make call to external API services

**Secure UDF:**

* Typical UDF uses SQL execution optimization which can enable access to underlying data.
* This may allow access to data to users indirectly.
  + Security concern
* Secure UDF solve this by not using the optimization
  + Thus, ensuring users have no access to underlying data, even indirectly
* Secure UDF solves this issue by hiding the definition
  + Only available to authorize users
* Performance vs Security

**Stored Procedure:**

* Procedural code with if-else logic, looping and other features.
* Construct dynamic SQL statement and execute
* Stored procedures can return a single value or tabulate data
* **UDF** takes an input, performs computation on the input and return a value
* **Store Procedure** execute one or more SQL queries
  + **They are often used for performing administrative tasks.**
* **Stored Procedure supported Language:**
  + SQL
  + JavaScript
  + Snowflake Scripting
  + **& when using Snowpark , we can use Java, Python or Scala**

**Resource Monitor:**

* It is not possible for more than one Resource Monitor to be monitoring the same virtal warehouse.
  + Account Level Monitoring is an exception
* Resource monitors do not control the credits usage by server-less features such as
  + Snowpipe
  + Automatic reclustring
  + Materialized View maintenance
  + Etc.
* Resource monitors can only suspends user-managed virtual warehouse.
* Resource monitors cannot manage the cost of cloud service.

**INFORMATION\_SCHEMA:**

* Data is real-time.
  + There is no data latency
* Data retention is shorter
  + Ranging from **7 days** to a minimum of **6 months(**depending on the view).
* Does NOT show information for deleted objects.

**Snowflake release Process:**

* New release every week.
* Update are **transparent** and cause **no downtime or disruptions**
* The updates may deploy
  + New features
  + Bug fixes
  + Enhancements
* **Two planned release every week**
  + FullRelease- new feature, feature enhancement or bug fixes.
  + Path releases: contains issue fixes only.

****