Snowflake

Introduction

* Snowflake is a Cloud based data warehousing system which is built on top of any Cloud Providers such as AWS, Azure, GCP. etc.
* It is a single Platform where you can build Data warehousing, data lakes, data engineering and data science workloads.

Snowflake Architecture

* Snowflake is having a hybrid model of traditional shared-disk and shared-nothing database architectures. Like shared-disk architectures, Snowflake uses a central data repository for storing persisted data that is accessible from all compute nodes in the cluster. like shared-nothing architectures, Snowflake processes queries using MPP (massively parallel processing) compute clusters where each node in the cluster stores a portion of the entire data set locally.

Three layers of Snowflake

Database Storage layer

* When data is loaded into Snowflake, Snowflake reorganizes that data into its internal optimized, compressed, columnar format. Snowflake stores this optimized data in cloud storage.
* Snowflake is managing data storage, compression, file size, Schema structure, Metadata and everything. The data objects stored by Snowflake is not visible or accessible by customers. This is accessible only by Snowflake SQL query.

Query Processing Layer

* Query execution is performed in Processing Layer. Snowflake process the query using Virtual Warehouses. Each Virtual warehouse is a MPP cluster composed by multiple compute nodes allocated by Snowflake from Cloud provider.
* Each virtual warehouses are independent; it does not share compute resources with other Virtual warehouses. So that, each virtual warehouse has no impact on performance of other Virtual warehouses.

Cloud Services layer

* It is a collection of services that coordinate activities across Snowflake. These services tied up together with all other components to process query execution from Login to query dispatch. These cloud services are also running on the compute instance allocated by Snowflake from Cloud provider.
* Services managed by Cloud Services
  + Authentication
  + Infrastructure Management
  + Metadata management
  + Query optimizer
  + Security

A black background with text and symbols

Description automatically generated

Virtual Warehouses Sizes

- XS - 1

- S - 2

- M - 4

- L - 8

- XL - 16

- 2XL -

- 4XL - 128

Types of Virtual warehouse cluster

- Single cluster

- Multi Cluster

Multi Clustering

* Multi cluster warehouses helps to automatically scale out, spin up more warehouses based on the User concurrency needs.

Two types of Scaling modes to set up a multi-cluster warehouse (Scaling Modes)

* Maximized mode - The minimum and maximum number of clusters are the same, so the multi-cluster virtual warehouse always has the maximum compute resources available.
* Auto-scalability - It will dynamically manages the clusters between the specified minimum and maximum according to the demand on the data warehouse.

Scaling Policy

* The scaling policy for a multi-cluster warehouse only applies if it is running in Auto-scale mode.

Two types of Scaling Policy

* Standard 🡪 Default 🡪 This policy is more expensive than ‘Economy’. However, it provides better performance.
  + The objective of this policy is to prevent or minimize queuing of queries.
* Economy
  + It gives low performance than Standard. It does Spin-up new cluster when at least existing cluster should busy at least for 6 mints.
  + Until than Query will be queued.

A screenshot of a computer

Description automatically generated

Snowflake supports two ways to scale warehouses: (Supports Both Horizontal Scaling and vertical Scaling)

Vertical scaling

* Scale up by resizing a warehouse. (Vertical scaling - Adding more power (e.g. CPU, RAM) to an existing machine)

Horizontal scaling

* Scale out by adding clusters to a multi-cluster warehouse (requires Snowflake Enterprise Edition or higher). - (Horizontal scaling - adding more machines to your pool of resources)

Snowflake Editions: -

* Standard
* Enterprise
* Business Critical
* Virtual Private

A screenshot of a black and white screen

Description automatically generated

Snowflake Roles

* Account Admin
  + Security Admin
  + User Admin
  + Public
* SysAdmin
  + Custom Role1
  + Custom Role2
  + Custom Role3

A diagram of a company

Description automatically generated

File formats supported by Snowflake

- CSV

- JSON

- Parquet

Connecting to Snowflake

* Web Interface
* CLI (SnowSQL)
* ODBC and JDBC
* Native Connectors (Python and Spark)
* Third party connectors which helps to connect Other ETL tools like Informatica and BI tools to Snowflake.

Data Loading

* Snowflake supports loading data from files staged in any of the following locations,
  + Internal (i.e. Snowflake) stages
  + Amazon S3
  + Google Cloud Storage
  + Microsoft Azure blob storage

Loading Data into Snowflake

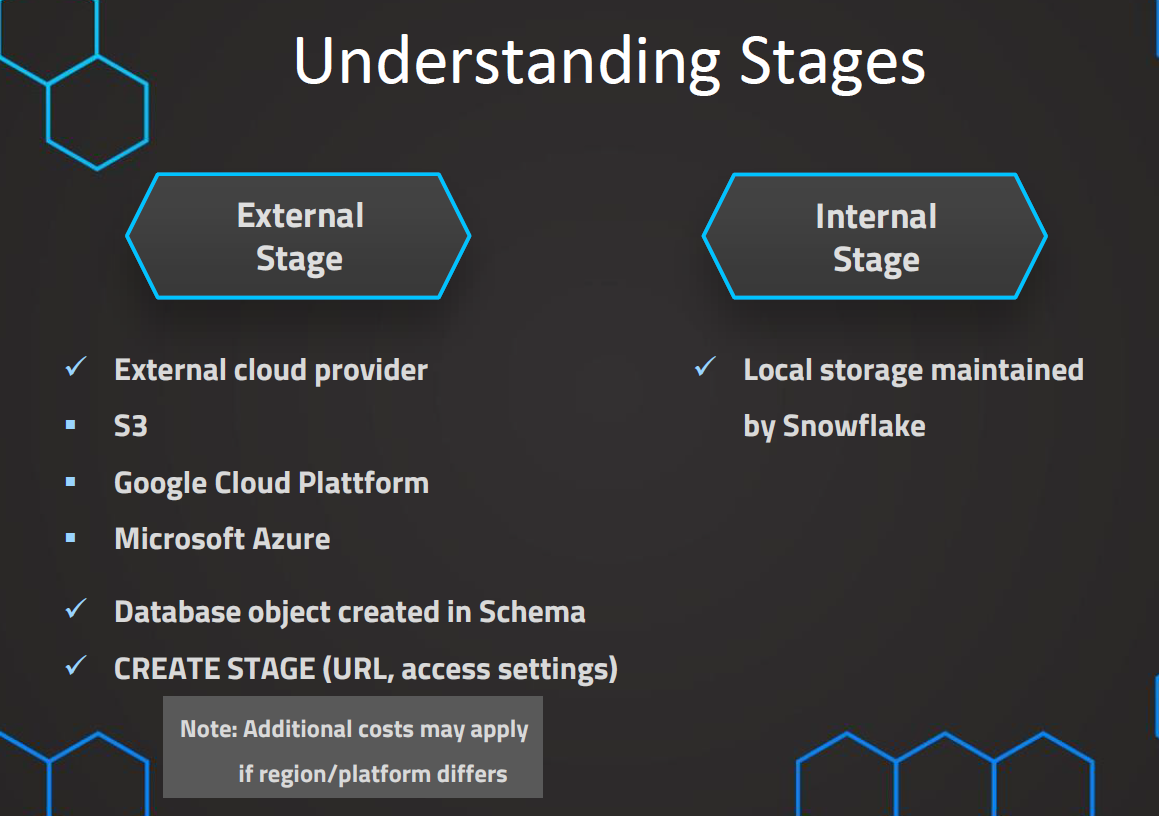
* Supported File Locations
  + External Stages
  + Internal Stages
* Bulk vs Continuous Loading
  + Bulk Loading Using the COPY Command
  + Continuous Loading Using Snowpipe
* Loading Data from Apache Kafka Topics
* Detection of Column Definitions in Staged Semi-structured Data Files
* Alternatives to Loading Data
  + External Tables (Data Lake)

Understanding of Stages

* Stages are nothing but locations where data files are placed and imported to Snowflake.

Types of Stages

* Internal Stage
  + If we don't have access to External Cloud providers, Snowflake provides Internal Stages. (Local Storage maintained by Snowflake)
* External Stage
  + External Stages are External locations provided by cloud providers such as S3, Blop storage, GCS.
  + Stages are database objects which will be created by using create Stage command.
  + Additional cost will be applied if Region/Platform differs.
  + Which is used on common use cases.
  + External cloud provider location such as S3, Blop storage, GCS.
  + Stages are database objects which are created in Schema.
  + It will be created by using create Stage command (CREATE STAGE(URL, access settings - Connection to that location)
  + Note: Additional cost may apply if Region/Platform differs.



How to Load Data into Snowflake.?

* Loading data into Tables and Databases.

Two main methods of Loading data

* Bulk Loading (Batch Loading)
  + It is most frequent option to load large volume of data from Stages using Copy Command. It uses Virtual warehouse to loading data. Transformation is possible.
  + Designed for Large Volume of data
  + Most Frequent method
  + Uses Virtual warehouses to Compute
  + Loading from Stages
  + COPY Command
  + Transformation Possible
* Continuous Loading
  + When we want to load the data immediately to snowflake once added to Stage than Snowpipe would be best choice.
  + Designed for Small Volumes of data and data need to upload immediately.
  + Automatically once they are added to stages
  + Latest results for Analysis
  + It is not using Virtual warehouses for processing. it uses Snowpipe (Serverless Feature)

A screenshot of a black screen

Description automatically generated

Creating Stage

A screenshot of a computer program

Description automatically generated

Copy Command

A screenshot of a computer program

Description automatically generated

Copy Options

* ON\_ERROR - What can be done if any errors come during copy process.
  + CONTINUE - It will Skip the Error files and load Successful data
  + ABORT\_STATEMENT (default)
  + SKIP\_FILE
    - SKIP\_FILE\_<number>
    - SKIP\_FILE\_<percentage\_number>
* VALIDATION\_MODE
  + It will validate the data instead of loading data into target
  + Two modes
    - RETURN\_n\_ROWS
      * It will validate and return n number of Rows if no error in COPY command.
      * Error will be returned if any error in Copy command.
* RETURN\_ERRORS
  + It will validate the COPY command and return list of Errors if any error in the command
  + IF no Errors found in Copy command, it won't return any Rows.

A screenshot of a computer program

Description automatically generated

* Size\_limit
  + Specify Max. Size (in Bytes) of data loaded in that Copy command (at least one file)
  + When the threshold reaches, Copy operation stops loading.
* RETURN\_FAILED\_ONLY - Specifies whether to return only files that have failed to load data.
  + TRUE
  + FALSE (Default)
* TRUNCATECOLUMNS - Specifies whether to truncate text string that exceeds the Target column length.
  + TRUE
  + FALSE (Default)
* FORCE - Force them to reload the files even the files have been loaded previously.
  + TRUE
  + FALSE
* PURGE - Specifies whether to remove source files from Stages automatically after the data is loaded Successfully.
  + TRUE
  + FALSE (Default)

Snowpipe

* Snowpipe helps to load data into target table once file appears in a bucket.
* Use case
  + If data needs to be available immediately for analysis.
  + Snowpipe uses Serverless feature instead of virtual warehouses

How Snowpipe Works internally.?

* Once source files are uploaded to S3, S3 Notification (event hander) will trigger Snowpipe, Snowpipe will use COPY command to load data from S3 into Snowflake.
* It will Serverless load to load data.

A computer screen shot of a server

Description automatically generated

High Level Steps to create Snowpipe: -

* Create Stage
* Create COPY command and test it whether it works or not.
* Create Pipe (Create pipe as an object with COPY command)
* Create S3 Notification - to trigger Snowpipe

A screenshot of a computer

Description automatically generated

Types of Tables

Why do we have these different table types?

* To manage storage cost.

Permanent

* Most used table types
* default table type.
* It will be created by CREATE TABLE command
* features applicable for Permanent tables are,
  + Time travel retention period (0 to 90 days)
  + Fail safe
* But cost is little high when compared to other table types.
* Active until table dropped

when is it required.? (Use cases)

* General-permanent data

Transient

* Second commonly used table. it will be created by CREATE TRANSIENT TABLE command.
* Features applicable for Transient tables are,
  + Time travel retention period (0 to 1 day)
  + Fail safe (No)
  + Active untill table dropped

when is it required.? (Use cases)

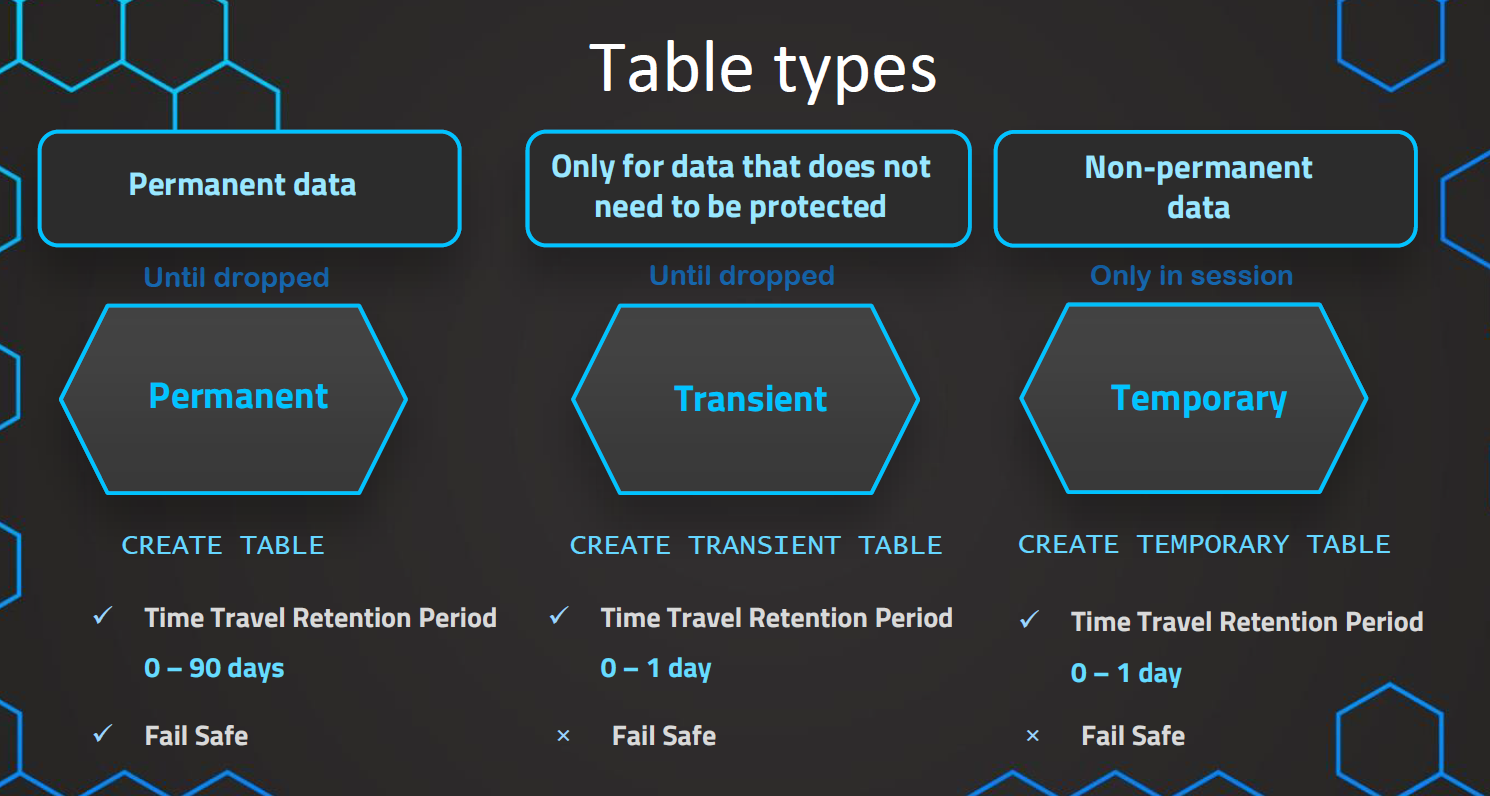
* Storage cost is low
* Only for data storage, no need for data to be protected.
* For large tables. will be used to save fail safe cost.

Temporary

* It will be created by CREATE TEMPORARY TABLE command.
* Features applicable for Temporary tables are,
  + Time travel retention period (0 to 1 day)
  + Fail safe (No)
* Only applicable for current session. When you are closing worksheet, these temporary tables are deleted completely. (No time travel retention period after deleted that table)
* Active until session closed.

when is it required.? (Use cases)

* Non-permanent data



Time Travel

* Time travel is a Snowflake feature That we can use to restore data that we have updated or deleted by accidently.

Time travel Period

* Standard Edition - Upto 1 day time travel
* Enterprise or Higher - Upto 90 days Time travel
* Business Critical - Upto 90 days Time travel
* Virtual private - Upto 90 days Time travel

Default retention period - 1 day.

Three methods to do Time travel

* at 🡪 at (OFFSET => -60\*1.5)
* Before 🡪 before (timestamp => '2021-04-15 17:47:50.581'::timestamp)
* using Query ID 🡪 before (statement => '019b9ee5-0500-8473-0043-4d8300073062')

A computer screen shot of a program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Fail Safe

* Part of Data protection Life Cycle.
* The purpose of Failsafe is to protect the historical data in the cluster when disaster happening.
* Disaster Recovery mechanism.
* Default fail safe period for permanent tables are 7 days. (which is not configurable)
  + Transient tables - 0 days.
* Period starts immediately after time travel period ends.
* No user interaction & need to call Snowflake support, they will recover the data.

Current data storage 🡪 Time travel 🡪 Fail safe Zone.

It will be automatically moved to fail safe zone.

A screen shot of a computer

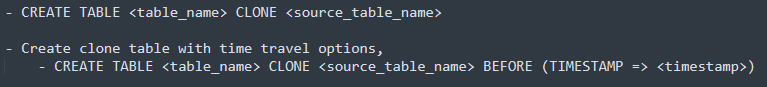
Description automatically generated

Zero Copy Cloning

* Create Copies of database, Schema or Table without moving data.
* Most advanced feature, it helps to create another database, schema, table with pointing same storage bucket to avoid the storage cost.
* It is Metadata operation.
* Quick and cost-efficient process for cloning our data.
* Cloned object is completely independent from original table.

Use cases

* Creating backups for development purposes.
* Works with Time travel also.



* Any structure of the object and metadata is inherited.
  + cluster key, comments etc.
* Data storage objects are eligible for cloning (Permanent and Transient only)
  + Databases
  + Schemas
  + Tables

Yes. We can clone temporary table to another temporary table.

* Also, we can clone Configuration objects as well,
  + Stages
  + File formats
  + Tasks

Swapping tables

A screen shot of a computer program

Description automatically generated

A black background with white text and blue squares

Description automatically generated

A diagram of a swapping tables

Description automatically generated

A screen shot of a computer

Description automatically generated

Snowflake's Column Level Security Features

Dynamic Data Masking

* Snowflake Dynamic Data Masking (DDM) is a column-level security feature that uses [masking policies](https://www.chaosgenius.io/blog/snowflake-data-governance/#what-is-masking-policy) to selectively mask sensitive data in table and view columns at query time.
* This means the underlying data is not altered in the database, but rather masked as it is retrieved.
* Dynamic Data Masking is more commonly used as it allows users to easily implement data masking without the need for external functions

A screenshot of a computer

Description automatically generated

External Tokenization

* External Tokenization enables the masking of sensitive data before it is loaded into Snowflake, which is achieved by utilizing an external function to tokenize the data and subsequently loading the tokenized data into Snowflake.
* External Tokenization, on the other hand, involves a more complex setup and is typically not widely implemented in organizations.

Streams

* When we build data warehouse, we need to take data from multiple data sources and perform ETL on top of it and write data into Data warehouse.
* Also, we need to take Delta as well. It may be new record, updated records, deleted records.
* To handle CDC in Snowflake, we are using Streams.
* Stream object is monitoring table about the DML changes (insert, update and delete) happening on the table.
* if three records are changed as part of CDC, those three records will be captured in Streams, along with that, three more columns would be added to Streams such as
  + METADATA$ACTION - whether data is updated, inserted, or deleted
  + METADATA$UPDATE
  + METADATA$ROW\_ID
* No additional cost been charged for those original columns which are captured from Source tables. charges will be applicable for those three extra columns.

How Stream works

A diagram of data processing

Description automatically generated

A diagram of a server

Description automatically generated

How to Create Stream

A screen shot of a computer code

Description automatically generated

Once, Stream records are updated to Target Table, then Stream records will be deleted.

Types of Streams

* Standard
* Append-Only

A screenshot of a black screen

Description automatically generated

Materialized Views

* We have a View that is queried very frequently and that will take long time to process the data. It will cause very bad experience to users. Also, more compute consumption, Cost will be expensive for every run.
* Solution is to create Materialized view. Which means create view, store the query results as Materialized view.
* Results will be stored in a Separate table, and this will be updated automatically (By automated service of Snowflake) based on the base table changes. So, it will bring updated data as output of Materialized view.

When to use MV?

* If the query is queried very frequently and that will take long time to process the data.
* The underlying data should not be changed very frequently. we can go for MV
* If the underlying table data changes frequently, then go for Task with Merge (Streams) logic.
* Maintenance Costs

Limitations

* Only available in Enterprise Edition.
* Joins will not be supported.
* Limited amount of Aggregated function available.
* No UDFs will be used inside.
* No having clause
* No order by clause
* No Limit clause.

How to Create

A screenshot of a computer program

Description automatically generated

How to refresh Materialized Views

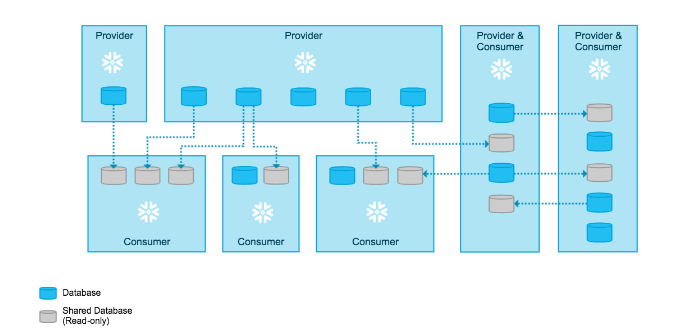
* Snowflake materialized views are different than materliazed views on other databases. Two important points:
  + Materialized views **provide always current data**. So, No need to refresh Separately.

Data Sharing

* Usually, Data sharing would be complicated process, we need to take backup and share it to the consumer, and If any changes on the source data, we need to take again and update to the consumer.
* But it is very simple concept to data sharing, Data sharing without actual copy of data and UpToDate. Compute resources and storage are tightly coupled together.
* So, Consumer can be able to get the data up to date. Shared data can be consumed by the own compute resources.
* Non-Snowflake users can also access through Reader account.

You can share the following Snowflake objects:

* Databases
* Tables
* Dynamic tables
* External tables
* Iceberg tables
* Secure views
* Secure materialized views
* Secure user-defined functions (UDFs)



Concepts

Providers

* Data provider who is going share data into Customers.

Consumers

* Data Consumers who is going to Consume data.

How to share To Snowflake Users

* Option 1: Grant privileges on objects to a share via a database role.
* Option 2: Grant privileges on objects directly to a share.

A diagram of a diagram

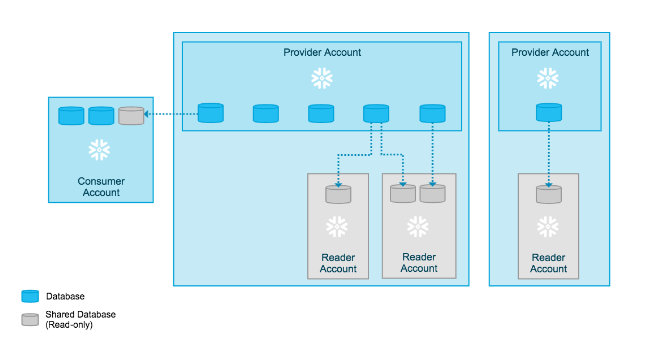
Description automatically generated

Benefits of Share

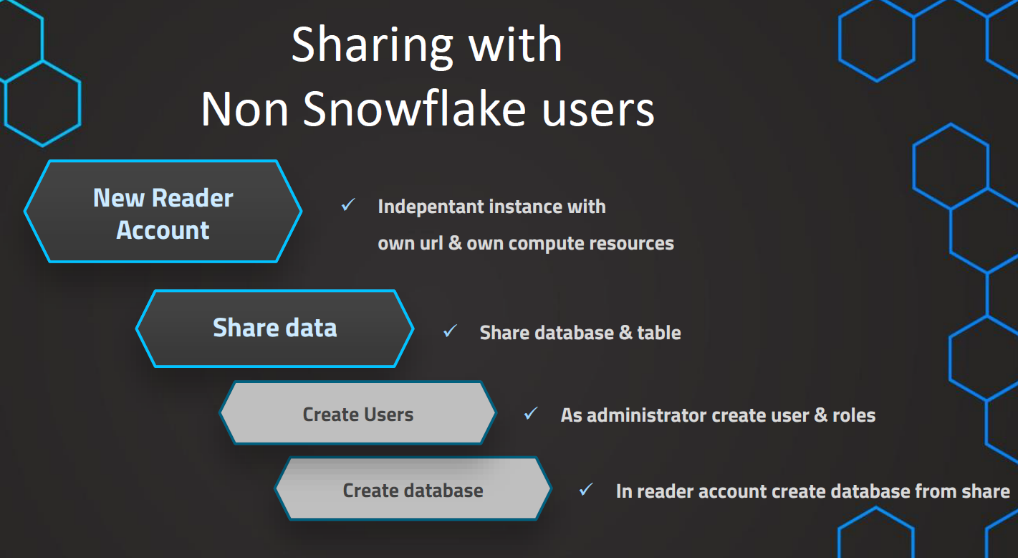
* New objects added to a share become immediately available to all consumers, providing real-time access to shared data.
* Updates to existing objects in a share become immediately available to all consumers.
* Access to a share (or any of the objects in a share) can be revoked at any time.

To Non-Snowflake Users

* To sharing data into non-snowflake users, you can create reader accounts. Reader accounts (formerly known as “read-only accounts”) provide a quick, easy way to share data with out customer to become Snowflake Users.
* Each reader account belongs to the provider account that created it. As a provider, you use **shares** to share databases with reader accounts.
* Users in a reader account can query data that has been shared with the reader account, but cannot perform any of the DML tasks.



How to share



Scheduled Tasks

* Tasks can be used to schedule SQL Statements.
* It may be Standalone tasks and tree of tasks.

Two main ways to Schedule

* Schedule at Interval. (Every 1 minute, 5 minute)
* Schedule at 8:30 PM time.

Task Status

* Suspended (default)
* Resumed/Started

Objectives

* Understand tasks
* Create tasks
* Schedule Tasks
* Tree of tasks
* Check task history

A screenshot of a computer program

Description automatically generated